Oracle® Solaris 관리: 보안 서비스

부품 번호: E25888
2012년 3월

ORACLE

120410@25097

목차

머리말

시스템 관리 설명서: 보안 서비스는 Oracle Solaris 운영 체제(Oracle Solaris OS) 관리 정보의 중요한 부분을 다루는 설명서 모음의 일부입니다. 본 설명서는 사용자가 이미 최신 릴리스를 설치했고 사용할 모든 네트워킹 소프트웨어를 설정했다고 가정합니다. Oracle Solaris OS는 Secure Shell과 같은 많은 기능을 포함하는 Oracle Solaris 제품군의 일부입니다.

주 – 본 Oracle Solaris 릴리스는 프로세서 아키텍처의 SPARC 및 x86 제품군을 사용하는 시스템을 지원합니다. 지원되는 시스템은 **Oracle Solaris OS: 하드웨어 호환성 목록**을 참조하십시오. 이 설명서에서는 플랫폼 유형에 따른 구현 차이가 있는 경우 이에 대하여 설명합니다.

이 책의 대상

본 설명서는 Oracle Solaris를 실행하는 한 대 이상의 시스템을 관리하는 사용자를 대상으로 작성되었습니다. 본 설명서를 사용하려면 2년 이상의 UNIX 시스템 관리 경력이 있어야 합니다. UNIX 시스템 관리 교육 과정에 참석하는 것도 도움이 될 수 있습니다.

시스템 관리 설명서의 구성

시스템 관리 설명서에서 설명하는 항목 목록은 다음과 같습니다.

책 제목	내용
SPARC 플랫폼에서 Oracle Solaris 부트 및 종료	SPARC 플랫폼에서 시스템 부트 및 종료, 부트 서비스 관리, 부트 동작 수정, ZFS에서 부트, 부트 아카이브 관리 및 부트 문제 해결
x86 플랫폼에서 Oracle Solaris 부트 및 종료	x86 플랫폼에서 시스템 부트 및 종료, 부트 서비스 관리, 부트 동작 수정, ZFS에서 부트, 부트 아카이브 관리 및 부트 문제 해결

책 제목	내용
Oracle Solaris 관리: 일반 작업	Oracle Solaris 명령 사용, 시스템 부트 및 종료, 사용자 계정 및 그룹 관리, 서비스, 하드웨어 오류, 시스템 정보, 시스템 리소스 및 시스템 성능 관리, 소프트웨어, 인쇄, 콘솔 및 터미널 관리, 시스템 및 소프트웨어 문제 해결
Oracle Solaris 관리: 장치 및 파일 시스템	이동식 매체, 디스크 및 장치, 파일 시스템, 데이터 백업 및 복원
Oracle Solaris 관리: IP 서비스	TCP/IP 네트워크 관리, IPv4 및 IPv6 주소 관리, DHCP, IPsec, IKE, IP 필터 및 IPQoS
Oracle Solaris Administration: Naming and Directory Services	NIS에서 LDAP으로 전환을 비롯한 DNS, NIS 및 LDAP 이름 지정 및 디렉토리 서비스
Oracle Solaris 관리: 네트워크 인터페이스 및 네트워크 가상화	WiFi 무선을 포함하는 자동 및 수동 IP 인터페이스 구성, 브릿지, VLAN, 통합, LLDP 및 IPMP 관리, 가상 NIC 및 리소스 관리
Oracle Solaris 관리: 네트워크 서비스	웹 캐시 서버, 시간 관련 서비스, 네트워크 파일 시스템(NFS 및 Autofs), 메일, SLP, PPP
Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리	리소스 관리 기능으로 응용 프로그램의 사용 가능한 시스템 리소스 이용 방법 제어, Oracle Solaris Zones 소프트웨어 분할 기술로 운영 체제 서비스를 가상화하여 실행 중인 응용 프로그램에 고립된 환경 조성, Oracle Solaris 10 Zones를 통해 Oracle Solaris 11 커널에서 실행 중인 Oracle Solaris 10 환경 호스트
Oracle Solaris 관리: 보안 서비스	감사, 장치 관리, 파일 보안, BART, Kerberos 서비스, PAM, 암호화 프레임워크, 키 관리, 권한, RBAC, SASL, 보안 셸 및 바이러스 검사
Oracle Solaris Administration: SMB and Windows Interoperability	SMB 서비스를 통해 Oracle Solaris 시스템에서 SMB 클라이언트가 SMB 공유를 사용할 수 있도록 구성, SMB 클라이언트로 SMB 공유 액세스, 고유의 ID 매핑 서비스를 통해 Oracle Solaris 시스템과 Windows 시스템 간에 사용자 및 그룹 ID 매핑
Oracle Solaris 관리: ZFS 파일 시스템	ZFS 저장소 풀 및 파일 시스템 생성 및 관리, 스냅샷, 복제, 백업, ACL(액세스 제어 목록)을 사용하여 ZFS 파일 보호, Oracle Solaris 시스템의 ZFS를 설치된 영역에 사용
Trusted Extensions 구성 및 관리	Trusted Extensions와 관련된 시스템 설치, 구성 및 관리
Oracle Solaris 11 보안 지침	Oracle Solaris 시스템 보안 유지와 영역, ZFS, Trusted Extensions와 같은 보안 기능에 대한 사용 시나리오

책 제목	내용
Oracle Solaris 10에서 Oracle Solaris 11로 전환	설치 분야에서 Oracle Solaris 10에서 Oracle Solaris 11로 전환하기 위한 시스템 관리 정보 및 예제 제공, 장치, 디스크 및 파일 시스템 관리, 소프트웨어 관리, 네트워킹, 시스템 관리, 보안, 가상화, 데스크탑 기능, 사용자 계정 관리, 사용자 환경에 에뮬레이트된 볼륨, 문제 해결 및 데이터 복구

Oracle Support에 액세스

Oracle 고객은 My Oracle Support를 통해 온라인 지원에 액세스할 수 있습니다. 자세한 내용은 `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info`를 참조하거나, 청각 장애가 있는 경우 `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs`를 방문하십시오.

활자체 규약

다음 표는 이 책에서 사용되는 활자체 규약에 대해 설명합니다.

표 P–1 활자체 규약

활자체	설명	예
`AaBbCc123`	명령 및 파일, 디렉토리 이름; 컴퓨터 화면에 출력되는 내용입니다.	`.login` 파일을 편집하십시오. 모든 파일 목록을 보려면 `ls -a` 명령을 사용하십시오. `machine_name% you have mail.`
`AaBbCc123`	사용자가 입력하는 내용으로 컴퓨터 화면의 출력 내용과 대조됩니다.	`machine_name%` **`su`** `Password:`
AaBbCc123	새로 나오는 용어, 강조 표시할 용어입니다. 명령줄 변수를 실제 이름이나 값으로 바꾸십시오.	`rm` *filename* 명령을 사용하여 파일을 제거합니다.
AaBbCc123	책 제목, 장, 절	**사용자 설명서**의 6장을 읽으십시오. **캐시**는 로컬로 저장된 복사본입니다. 파일을 저장하면 **안 됩니다**. **주:** 일부 강조된 항목은 온라인에서 굵은체로 나타납니다.

명령 예의 셸 프롬프트

다음 표에는 Oracle Solaris OS에 포함된 셸의 기본 UNIX 시스템 프롬프트 및 수퍼유저 프롬프트가 나와 있습니다. 명령 예제에 표시된 기본 시스템 프롬프트는 Oracle Solaris 릴리스에 따라 다릅니다.

표 P–2 쉘 프롬프트

셸	프롬프트
Bash 셸, Korn 셸 및 Bourne 셸	`$`
수퍼유저용 Bash 셸, Korn 셸 및 Bourne 셸	`#`
C 셸	`machine_name%`
수퍼유저용 C 셸	`machine_name#`

제 1 부

보안 개요

이 설명서는 Oracle Solaris OS의 보안 향상 기능을 중점적으로 설명합니다. 이 설명서는 이러한 보안 기능의 시스템 관리자 및 사용자를 대상으로 합니다. 1 장, “보안 서비스(개요)”는 설명서의 항목을 소개합니다.

◆ ◆ ◆ 1 장

1 보안 서비스(개요)

Oracle Solaris OS의 보안 유지를 위해 소프트웨어에서는 다음과 같은 기능을 제공합니다.

- 29 페이지 "시스템 보안" – 침입자를 방지하고, 시스템 리소스 및 장치의 잘못된 사용을 방지하며, 사용자 또는 침입자가 파일을 악의적으로 수정하거나 실수로 수정하지 못하도록 보호합니다
- 30 페이지 "암호화 서비스" – 송신자와 지정된 수신자만 컨텐츠를 읽을 수 있도록 데이터를 스크램블하고, 암호화 공급자와 공개 키 객체를 관리합니다.
- 31 페이지 "인증 서비스" – 사용자를 안전하게 식별합니다. 이때 사용자의 이름과 특정 형태의 증명(보통 암호)이 필요합니다.
- 32 페이지 "암호화를 통한 인증" – 인증된 당사자들이 가로채기, 수정 또는 스푸핑 없이 통신할 수 있습니다.
- 32 페이지 "감사" – 파일 액세스, 보안 관련 시스템 호출, 인증 오류 등을 비롯하여 시스템에 대한 보안 변경의 원인을 파악합니다.
- 32 페이지 "보안 정책" – 시스템 또는 시스템의 네트워크에 대한 보안 지침의 설계 및 구현입니다.

시스템 보안

시스템 보안은 시스템 리소스가 적절하게 사용되도록 해줍니다. 액세스 제어를 통해 시스템 리소스에 액세스할 수 있는 사용자를 제한할 수 있습니다. 시스템 보안 및 액세스 제어를 위한 Oracle Solaris 기능은 다음과 같습니다.

- **로그인 관리 도구** – 사용자의 로그인 권한을 모니터링하고 제어하는 명령입니다. 58 페이지 "로그인 및 암호 보안(작업 맵)"을 참조하십시오.
- **하드웨어 액세스** – PROM에 대한 액세스를 제한하고, 시스템을 부트할 수 있는 사용자를 제한하는 명령입니다. 68 페이지 "시스템 하드웨어에 대한 액세스 제어(작업)"를 참조하십시오.

- **리소스 액세스** – 시스템 리소스의 잘못된 사용은 최소화하면서 시스템 리소스의 적절한 사용은 최대화하기 위한 도구 및 전략입니다. 45 페이지 “시스템 리소스에 대한 액세스 제어”를 참조하십시오.

  Oracle Solaris 영역에서 리소스 관리에 대한 내용은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 제I부, “Oracle Solaris 리소스 관리”를 참조하십시오.
- **역할 기반 액세스 제어(RBAC)** – 특정 관리 작업을 수행하도록 허용된 제한된 특수 사용자 계정에 대한 아키텍처입니다. 135 페이지 “역할 기반 액세스 제어(개요)”를 참조하십시오.
- **권한** – 작업을 수행하기 위한 프로세스에 대한 개별 권한입니다. 이러한 프로세스 권한은 커널에 적용됩니다. 146 페이지 “권한(개요)”을 참조하십시오.
- **장치 관리** – 장치 **정책**은 UNIX 권한으로 이미 보호된 장치를 추가로 보호합니다. 장치 **할당**은 주변 장치(예: 마이크 또는 CD-ROM 드라이브)에 대한 액세스를 제어합니다. 할당 해제 시 device-clean 스크립트가 장치에서 모든 데이터를 지울 수 있습니다. 43 페이지 “장치에 대한 액세스 제어”를 참조하십시오.
- **기본 감사 보고 도구(BART)** – 시스템에 있는 파일의 파일 속성에 대한 스냅샷(**매니페스트**)입니다. 여러 시스템에서 또는 한 시스템에서 시간별로 매니페스트를 비교하는 방식으로 파일 변경 사항을 모니터링하면 보안 위험을 줄일 수 있습니다. 6 장, “기본 감사 보고 도구 사용(작업)”을 참조하십시오.
- **파일 권한** – 파일 또는 디렉토리의 속성입니다. 권한으로 파일 읽기, 쓰기, 실행 또는 디렉토리 검색이 허용되는 사용자 및 그룹을 제한할 수 있습니다. 7 장, “파일에 대한 액세스 제어(작업)”를 참조하십시오.
- **바이러스 검사 소프트웨어** – vscan 서비스는 응용 프로그램에서 파일을 사용하기 전에 파일에 바이러스가 있는지 검사합니다. 파일 시스템의 클라이언트가 파일에 액세스하기 전에 파일 시스템에서는 이 서비스를 호출하여 최근 바이러스 정의에 대해 파일을 실시간으로 검사할 수 있습니다.

  실시간 검사는 타사 응용 프로그램에서 수행되며, 파일을 열었을 때와 파일을 닫은 후에 검사할 수 있습니다. 4 장, “바이러스 검사 서비스(작업)”를 참조하십시오.

암호화 서비스

암호화는 데이터를 암호화하고 해독하는 데 사용되는 과학으로, 무결성, 프라이버시 및 신뢰성을 보장하는 데 사용됩니다. 무결성은 데이터가 변경되지 않았음을 의미합니다. 프라이버시는 다른 사용자가 데이터를 읽을 수 없음을 의미합니다. 데이터 신뢰성은 전달된 데이터가 전송된 데이터임을 의미합니다. 사용자 인증은 사용자가 신원 증명을 하나 이상 제공했음을 의미합니다. 인증 방식은 수학적으로 데이터의 소스 또는 신원 증명을 확인합니다. 암호화 방식은 일반 관찰자가 데이터를 읽을 수 없도록 데이터를 스크램블합니다. 암호화 서비스는 응용 프로그램과 사용자에게 인증 및 암호화 방식을 제공합니다.

- **암호화 프레임워크** – RSA Security Inc. PKCS #11 Cryptographic Token Interface(Cryptoki)라는 표준을 기반으로 하는 커널 레벨 및 사용자 레벨 소비자에 대한 암호화 서비스의 중앙 프레임워크입니다. 용도로는 암호, IPsec 및 타사 응용 프로그램이 있습니다. 이 프레임워크는 암호화를 위해 하드웨어 및 소프트웨어 소스를 중앙에서 관리합니다. PKCS #11 라이브러리는 타사 개발자에게 자신의 응용 프로그램에 대한 암호화 요구 사항을 접목시킬 수 있는 API를 제공합니다. 11 장, “암호화 프레임워크(개요)”를 참조하십시오.
- **응용 프로그램별 암호화 방식** –
  - 보안 RPC에서 DES를 사용하려면 267 페이지 “보안 RPC 개요”를 참조하십시오.
  - Kerberos 서비스에서 DES, 3DES, AES 및 ARCFOUR를 사용하려면 19 장, “Kerberos 서비스 소개”를 참조하십시오.
  - Secure Shell에서 RSA, DSA 및 암호화(예: Blowfish)를 사용하려면 17 장, “Secure Shell 사용(작업)”을 참조하십시오.
  - 암호의 암호화 알고리즘은 63 페이지 “기본 암호 보안 처리 알고리즘 변경(작업)”을 참조하십시오.
- KMF(키 관리 프레임워크)는 정책, 키 및 인증서를 비롯한 공개 키 객체를 관리하기 위한 중앙 유틸리티를 제공합니다. KMF는 OpenSSL, NSS 및 PKCS #11 공개 키 기술용으로 이러한 객체를 관리합니다. 13 장, “키 관리 프레임워크”를 참조하십시오.

인증 서비스

인증은 미리 정의된 기준을 토대로 사용자나 서비스를 식별하는 방식입니다. 인증 서비스는 단순 이름-암호 쌍에서 더 정교한 시도-응답 시스템(예: 토큰 카드 및 수명 측정)에 이르기까지 다양합니다. 강력한 인증 방식은 해당 사용자만 알고 있는 사용자 제공 정보와 확인 가능한 개인 항목에 의존합니다. 사용자 이름은 사용자가 알고 있는 정보의 한 예입니다. 스마트 카드 또는 지문 등은 검증 가능합니다. 인증을 위한 Oracle Solaris 기능은 다음과 같습니다.

- **보안 RPC** – Diffie-Hellman 프로토콜을 사용하여 NFS 마운트 및 이름 지정 서비스(예: NIS)를 보호하는 인증 방식입니다. 267 페이지 “보안 RPC 개요”를 참조하십시오.
- **플러그 가능한 인증 모듈(PAM)** – 서비스를 재컴파일하지 않고 다양한 인증 기술을 시스템 항목 서비스에 접목할 수 있도록 해주는 프레임워크입니다. 몇 가지 시스템 항목 서비스로는 `login` 및 `ftp`가 있습니다. 15 장, “PAM 사용”을 참조하십시오.
- **Simple Authentication and Security Layer(SASL)** – 네트워크 프로토콜에 인증 및 보안 서비스를 제공하는 프레임워크입니다. 16 장, “SASL 사용”을 참조하십시오.
- **Secure Shell** – 비보안 네트워크를 거치는 통신을 암호화하는 보안 원격 로그인 및 전송 프로토콜입니다. 17 장, “Secure Shell 사용(작업)”을 참조하십시오.
- **Kerberos 서비스** – 인증을 통한 암호화를 제공하는 클라이언트-서버 아키텍처입니다. 제6부를 참조하십시오.

암호화를 통한 인증

암호화를 통한 인증은 기본 보안 통신입니다. 인증은 소스와 대상이 의도한 당사자인지 확인하는 데 도움이 됩니다. 암호화는 소스에서 통신을 코드화하고 대상에서 통신을 해독합니다. 암호화를 사용할 경우 침입자가 가로채기를 위해 관리하는 전송을 읽지 못하도록 합니다. 보안 통신을 위한 Oracle Solaris 기능은 다음과 같습니다.

- **Secure Shell** – 데이터 전송 및 대화식 사용자 네트워크 세션을 도청, 세션 하이재킹 및 "가로채기(man-in-the-middle)" 공격으로부터 보호하는 프로토콜입니다. 강력한 인증은 공개 키 암호화를 통해 제공됩니다. X Windows 서비스 및 기타 네트워크 서비스는 추가 보호를 위해 Secure Shell 연결을 통해 안전하게 터널링될 수 있습니다. 17 장, "Secure Shell 사용(작업)"을 참조하십시오.
- **Kerberos 서비스** – 암호화를 통한 인증을 제공하는 클라이언트-서버 아키텍처입니다. 제6부를 참조하십시오.
- **Internet Protocol Security Architecture(IPsec)** – IP 데이터그램 보호를 제공하는 아키텍처입니다. 보호에는 기밀성, 강력한 데이터 무결성, 데이터 인증 및 부분 시퀀스 무결성이 포함됩니다. **Oracle Solaris 관리: IP 서비스**의 제III부, "IP 보안"을 참조하십시오.

감사

감사는 시스템 보안 및 유지 관리성의 기본적인 개념입니다. 감사는 어떤 상황이 발생했는지 확인하기 위해 시스템에서 작업 및 이벤트 내역을 검사하는 프로세스입니다. 내역은 수행된 작업, 수행 시기 및 수행자 및 영향을 받는 대상에 대한 로그에 보존됩니다. 제7부를 참조하십시오.

보안 정책

보안 정책 또는 정책이라는 문구는 조직의 보안 지침을 나타내는 것으로, 이 설명서 전반에서 사용됩니다. 사이트의 보안 정책은 처리 중인 정보의 민감도를 정의하는 규칙 세트이자, 허용되지 않은 액세스로부터 정보를 보호하는 데 사용되는 측정치입니다. Secure Shell, 인증, RBAC, 권한 부여, 권한 및 리소스 제어와 같은 보안 기술을 통해 정보를 보호할 수 있습니다.

일부 보안 기술은 구현의 특정 측면을 설명할 때 단어 정책을 사용하기도 합니다. 예를 들어 Oracle Solaris에서는 감사 정책 옵션을 사용하여 감사 정책의 몇 가지 측면을 구성합니다. 다음 표에서는 단어 정책을 사용하여 구현의 특정 측면을 설명하는 기능에 대한 용어집, 매뉴얼 페이지 및 정보로 이동할 수 있습니다.

표 1–1 Oracle Solaris에서 "정책"이라는 단어 사용

"정책" 용어	선택된 매뉴얼 페이지	기타 정보
감사 정책	auditconfig(1M)	26 장, "감사(개요)"
암호화 프레임워크의 정책	cryptoadm(1M)	11 장, "암호화 프레임워크(개요)"
장치 정책	getdevpolicy(1M)	43 페이지 "장치에 대한 액세스 제어"
Kerberos 정책	krb5.conf(4)	23 장, "Kerberos 주체 및 정책 관리(작업)"
네트워크 정책	ipfilter(5), ipadm(1M), ike.config(4), ipsecconf(1M), routeadm(1M)	**Oracle Solaris 관리: IP 서비스**의 제III부, "IP 보안"
암호 정책	passwd(1), crypt.conf(4), policy.conf(4)	38 페이지 "로그인 제어 유지 관리"
공개 키 기술에 대한 정책	kmfcfg(1)	13 장, "키 관리 프레임워크"
RBAC 정책	rbac(5), policy.conf(4)	203 페이지 "policy.conf 파일"

제 2 부

시스템, 파일 및 장치 보안

이 절에서는 네트워크에 연결되지 않은 시스템에서 구성할 수 있는 보안을 다룹니다. 이 장에서는 디스크와 파일, 주변 장치에 대한 액세스를 계획, 모니터 및 제어하는 방법에 대해 설명합니다.

◆ ◆ ◆ 2 장

2 시스템 보안 관리(개요)

시스템의 정보를 안전하게 유지하는 것은 중요한 시스템 관리 책임입니다. 이 장에서는 시스템 보안 관리에 대한 개요 정보를 제공합니다.

다음은 이 장에 포함된 개요 정보 목록입니다.

컴퓨터 시스템에 대한 액세스 제어

작업 공간에서 서버에 연결된 모든 컴퓨터는 하나의 큰 다중 시스템으로 생각할 수 있습니다. 관리자는 이 대형 시스템의 보안을 책임집니다. 액세스 권한을 얻으려는 외부인으로부터 네트워크를 보호해야 합니다. 또한 네트워크 내의 컴퓨터에 있는 데이터의 무결성을 유지해야 합니다.

파일 레벨에서 Oracle Solaris는 파일, 디렉토리 및 장치를 보호하는 데 사용할 수 있는 표준 보안 기능을 제공합니다. 시스템 및 네트워크 레벨에서 보안 문제는 거의 동일합니다. 보안 방어의 최우선 업무는 시스템에 대한 액세스를 제어하는 것입니다.

다음을 사용하여 시스템 액세스를 제어하고 모니터할 수 있습니다.

- 55 페이지 “보안 문제 보고”

물리적 보안 유지 관리

시스템에 대한 액세스를 제어하려면 컴퓨터 환경의 물리적 보안을 유지 관리해야 합니다. 예를 들어, 로그인되어 있는 상태에서 아무도 없는 시스템은 허용되지 않은 액세스 위험에 노출됩니다. 침입자는 운영 체제 및 네트워크에 대한 액세스 권한을 얻을 수 있습니다. 컴퓨터의 주변 및 컴퓨터 하드웨어를 허용되지 않은 액세스로부터 물리적으로 보호해야 합니다.

SPARC 시스템은 하드웨어 설정에 대한 허용되지 않은 액세스로부터 보호할 수 있습니다. `eeprom` 명령을 사용하여 PROM에 액세스하기 위한 암호를 요구합니다. 자세한 내용은 68 페이지 “하드웨어 액세스에 대한 암호 요구 방법”을 참조하십시오. x86 하드웨어를 보호하려면 해당 공급업체 설명서를 참조하십시오.

로그인 제어 유지 관리

또한 시스템이나 네트워크에 대한 허용되지 않은 액세스를 막아야 하는데 암호 지정 및 로그인 제어를 통해 가능합니다. 시스템의 모든 계정에는 암호가 있어야 합니다. 암호는 단순한 인증 방식입니다. 암호가 없는 계정이 있으면 사용자 이름을 추측할 수 있는 침입자가 전체 네트워크에 액세스할 수 있습니다. 강력한 암호 알고리즘은 무단 공격으로부터 보호합니다.

사용자가 시스템에 로그인하면 `login` 명령이 이름 스위치 서비스 `svc:/system/name-service/switch`의 정보에 따라 알맞은 이름 지정 서비스 또는 디렉토리 서비스 데이터베이스를 확인합니다. 다음 데이터베이스는 로그인에 영향을 줄 수 있습니다.

- `files` – 로컬 시스템의 `/etc` 파일을 지정합니다.
- `ldap` – LDAP 서버의 LDAP 디렉토리 서비스를 지정합니다.
- `nis` – NIS 마스터 서버의 NIS 데이터베이스를 지정합니다.
- `dns` – 네트워크의 도메인 이름 서비스를 지정합니다.

이름 지정 서비스에 대한 설명은 `nscd(1M)` 매뉴얼 페이지를 참조하십시오. 이름 지정 서비스 및 디렉토리 서버에 대한 자세한 내용은 **Oracle Solaris Administration: Naming and Directory Services** 를 참조하십시오.

`login` 명령은 사용자가 제공한 사용자 이름과 암호를 확인합니다. 사용자 이름이 암호 데이터베이스에 없을 경우 `login` 명령은 시스템에 대한 액세스를 거부합니다. 지정된 사용자 이름에 대한 암호가 올바르지 않을 경우 `login` 명령은 시스템에 대한 액세스를 거부합니다. 사용자가 유효한 사용자 이름과 해당하는 암호를 제공할 경우 시스템은 사용자에게 시스템에 대한 액세스 권한을 부여합니다.

시스템 로그인을 성공한 후 PAM 모듈은 로그인을 응용 프로그램에 전달할 수 있습니다. 자세한 내용은 15 장, “PAM 사용”을 참조하십시오.

Oracle Solaris 시스템에서는 정교한 인증 및 권한 부여 방식을 사용할 수 있습니다. 네트워크 레벨의 인증 및 권한 부여 방식에 대한 자세한 내용은 52 페이지 "원격 액세스에 대한 인증 및 권한 부여"를 참조하십시오.

암호 정보 관리

사용자가 시스템에 로그인할 때 사용자 이름과 암호를 모두 제공해야 합니다. 로그인은 공개적으로 알려져 있더라도 암호는 비밀로 유지해야 합니다. 암호는 각 사용자만 알고 있어야 합니다. 사용자는 자신의 암호를 신중하게 선택하고 자주 변경해야 합니다.

암호는 사용자 계정을 설정할 때 처음 만들어집니다. 사용자 계정에 대한 보안 유지를 위해 사용자가 자신의 암호를 정기적으로 변경하도록 암호 유효 기간을 설정할 수 있습니다. 또한 암호를 잠가 사용자 계정을 사용 안함으로 설정할 수도 있습니다. 암호 관리에 대한 자세한 내용은 **Oracle Solaris 관리: 일반 작업**의 2 장, "사용자 계정 및 그룹 관리(개요)" 및 `passwd(1)` 매뉴얼 페이지를 참조하십시오.

로컬 암호

네트워크에서 로컬 파일을 사용하여 사용자를 인증하는 경우 암호 정보는 시스템의 `/etc/passwd` 및 `/etc/shadow` 파일에 보관됩니다. 사용자 이름 및 기타 정보는 `/etc/passwd` 파일에 보관됩니다. 암호화된 암호 자체는 별도의 **섀도우** 파일인 `/etc/shadow`에 보관됩니다. 이 보안 방식은 사용자가 암호화된 암호에 액세스하지 못하도록 막습니다. `/etc/passwd` 파일은 시스템 로그인 권한이 있는 모든 사용자가 사용할 수 있지만 `/etc/shadow` 파일은 수퍼유저만 읽을 수 있습니다. `passwd` 명령을 사용하여 로컬 시스템에서 사용자의 암호를 변경할 수 있습니다.

NIS 암호

네트워크에서 NIS를 사용하여 사용자를 인증하는 경우 암호 정보는 NIS 암호 맵에 보관됩니다. NIS는 암호 유효 기간을 지원하지 않습니다. `passwd -r nis` 명령을 사용하여 NIS 암호 맵에 저장된 사용자의 암호를 변경할 수 있습니다.

LDAP 암호

Oracle Solaris LDAP 이름 지정 서비스는 암호 정보 및 섀도우 정보를 LDAP 디렉토리 트리의 `ou=people` 컨테이너에 저장합니다. Oracle Solaris LDAP 이름 지정 서비스 클라이언트에서 `passwd -r ldap` 명령을 사용하여 사용자의 암호를 변경할 수 있습니다. LDAP 이름 지정 서비스는 암호를 LDAP 저장소에 저장합니다.

암호 정책은 Oracle Directory Server Enterprise Edition에서 적용됩니다. 특히 클라이언트의 `pam_ldap` 모듈은 Oracle Directory Server Enterprise Edition에서 적용되는 암호 정책 제어를 따릅니다. 자세한 내용은 **Oracle Solaris Administration: Naming and Directory Services**의 "LDAP Naming Services Security Model"을 참조하십시오.

암호 암호화

강력한 암호 암호화는 공격에 대한 1차 방어선을 제공합니다. Oracle Solaris 소프트웨어는 6가지 암호 암호화 알고리즘을 제공합니다. Blowfish, MD5 및 SHA 알고리즘은 UNIX 알고리즘보다 강력한 암호 암호화를 제공합니다.

암호 알고리즘 식별자

사이트에 대한 알고리즘 구성은 `/etc/security/policy.conf` 파일에 지정합니다. `policy.conf` 파일에서 알고리즘은 다음 표에 나온 대로 식별자로 이름이 지정됩니다. 식별자-알고리즘 매핑은 `/etc/security/crypt.conf` 파일을 참조하십시오.

표 2–1 암호 암호화 알고리즘

식별자	설명	알고리즘 매뉴얼 페이지
1	BSD 및 Linux 시스템의 MD5 알고리즘과 호환되는 MD5 알고리즘입니다.	crypt_bsdmd5(5)
2a	BSD 시스템의 Blowfish 알고리즘과 호환되는 Blowfish 알고리즘입니다.	crypt_bsdbf(5)
md5	BSD 및 Linux 버전의 MD5보다 강력한 것으로 알려져 있는 Sun MD5 알고리즘입니다.	crypt_sunmd5(5)
5	SHA256 알고리즘입니다. SHA는 Secure Hash Algorithm(보안 해시 알고리즘)의 약어입니다. 이 알고리즘은 SHA-2 계열에 속합니다. SHA256은 255자 암호를 지원합니다.	crypt_sha256(5)
6	SHA512 알고리즘입니다.	crypt_sha512(5)
__unix__	전통적인 UNIX 암호화 알고리즘입니다.	crypt_unix(5)

policy.conf 파일의 알고리즘 구성

다음은 `policy.conf` 파일의 기본 알고리즘 구성을 보여줍니다.

```
#
...
# crypt(3c) Algorithms Configuration
#
# CRYPT_ALGORITHMS_ALLOW specifies the algorithms that are allowed
to
# be used for new passwords.  This is enforced only in crypt_gensalt(3c).
#
CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6

# To deprecate use of the traditional unix algorithm, uncomment below
# and change CRYPT_DEFAULT= to another algorithm.  For example,
# CRYPT_DEFAULT=1 for BSD/Linux MD5.
#
#CRYPT_ALGORITHMS_DEPRECATE=__unix__
```

```
# The Oracle Solaris default is a SHA256 based algorithm.  To revert to
# the policy present in Solaris releases set CRYPT_DEFAULT=__unix__,
# which is not listed in crypt.conf(4) since it is internal to libc.
#
CRYPT_DEFAULT=5
...
```

CRYPT_DEFAULT에 대한 값을 변경할 경우 새 사용자의 암호가 새 값과 연결된 알고리즘으로 암호화됩니다.

기존 사용자가 자신의 암호를 변경할 경우 이전 암호의 암호화 방식은 새 암호를 암호화하는 데 사용되는 알고리즘에 영향을 줍니다. 예를 들어, CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6 및 CRYPT_DEFAULT=1을 생각할 수 있습니다. 다음 표는 암호화된 암호를 생성하는 데 사용되는 알고리즘을 보여줍니다.

식별자 = 암호 알고리즘		
초기 암호	**변경된 암호**	**설명**
1 = crypt_bsdmd5	동일한 알고리즘을 사용합니다.	1 식별자는 또한 CRYPT_DEFAULT 의 값입니다. 사용자의 암호는 계속해서 crypt_bsdmd5 알고리즘으로 암호화됩니다.
2a = crypt_bsdbf	동일한 알고리즘을 사용합니다.	2a 식별자는 CRYPT_ALGORITHMS_ALLOW 목록에 있습니다. 따라서 새 암호는 crypt_bsdbf 알고리즘으로 암호화됩니다.
md5 = crypt_md5	동일한 알고리즘을 사용합니다.	md5 식별자는 CRYPT_ALGORITHMS_ALLOW 목록에 있습니다. 따라서 새 암호는 crypt_md5 알고리즘으로 암호화됩니다.
5 = crypt_sha256	동일한 알고리즘을 사용합니다.	5 식별자는 CRYPT_ALGORITHMS_ALLOW 목록에 있습니다. 따라서 새 암호는 crypt_sha256 알고리즘으로 암호화됩니다.
6 = crypt_sha512	동일한 알고리즘을 사용합니다.	6 식별자는 CRYPT_ALGORITHMS_ALLOW 목록에 있습니다. 따라서 새 암호는 crypt_sha512 알고리즘으로 암호화됩니다.
__unix__ = crypt_unix	crypt_bsdmd5 알고리즘을 사용합니다.	__unix__ 식별자는 CRYPT_ALGORITHMS_ALLOW 목록에 없습니다. 따라서 crypt_unix 알고리즘을 사용할 수 없습니다. 새 암호는 CRYPT_DEFAULT 알고리즘으로 암호화됩니다.

알고리즘 선택 구성에 대한 자세한 내용은 policy.conf(4) 매뉴얼 페이지를 참조하십시오. 암호 암호화 알고리즘을 지정하려면 63 페이지 “기본 암호 보안 처리 알고리즘 변경(작업)”을 참조하십시오.

특수 시스템 계정

root 계정은 여러 특수 **시스템** 계정 중 하나입니다. 이러한 계정 중에서 root 계정만 암호가 지정되고 로그인할 수 있습니다. nuucp 계정은 파일 전송을 위해 로그인할 수 있습니다. 기타 시스템 계정은 root의 전체 권한을 사용하지 않고 파일을 보호하거나 관리 프로세스를 실행합니다.

주의 – 시스템 계정의 암호 설정은 변경하지 마십시오. Oracle Solaris의 시스템 계정은 안전한 보안 상태로 전달됩니다.

다음 표는 몇 가지 시스템 계정 및 사용을 나열합니다. 시스템 계정은 특수한 기능을 수행합니다. 각 계정은 100보다 작은 UID를 가집니다.

표 2–2 시스템 계정 및 사용

시스템 계정	UID	사용
root	0	거의 제한 사항이 없습니다. 다른 보호 및 권한을 대체할 수 있습니다. root 계정은 전체 시스템에 대한 액세스 권한을 가집니다. root 로그인에 대한 암호는 매우 신중하게 보호되어야 합니다. root 계정은 대부분의 Oracle Solaris 명령을 소유합니다.
daemon	1	백그라운드 처리를 제어합니다.
bin	2	일부 Oracle Solaris 명령을 소유합니다.
sys	3	많은 시스템 파일을 소유합니다.
adm	4	일부 관리 파일을 소유합니다.
lp	71	프린터에 대한 객체 데이터 파일 및 스풀링된 데이터 파일을 소유합니다.
uucp	5	UUCP(UNIX-UNIX 복사 프로그램)에 대한 객체 데이터 파일 및 스풀링된 데이터 파일을 소유합니다.
nuucp	9	원격 시스템에서 시스템에 로그인하고 파일 전송을 시작하는 데 사용됩니다.

원격 로그인

원격 로그인은 침입자를 위한 공격 경로가 될 수 있습니다. Oracle Solaris는 원격 로그인을 모니터, 제한 및 사용 안함으로 설정할 수 있는 여러 가지 명령을 제공합니다. 절차는 58 페이지 “로그인 및 암호 보안(작업 맵)”을 참조하십시오.

기본적으로 원격 로그인은 시스템 마우스, 키보드, 프레임 버퍼, 오디오 장치 등과 같은 특정 시스템 장치에 대한 제어 권한을 얻거나 읽을 수 없습니다. 자세한 내용은 logindevperm(4) 매뉴얼 페이지를 참조하십시오.

장치에 대한 액세스 제어

컴퓨터 시스템에 연결된 주변 장치는 보안 위험을 노출시킬 수 있습니다. 마이크는 대화를 가로채고 원격 시스템에 전송할 수 있습니다. CD-ROM은 CD-ROM 장치의 다음 사용자가 읽을 수 있는 정보를 남겨 둘 수 있습니다. 프린터는 원격으로 액세스할 수 있습니다. 시스템에 필수적인 장치도 보안 문제를 유발할 수 있습니다. 예를 들어, `bge0`과 같은 네트워크 인터페이스는 필수 장치로 간주됩니다.

Oracle Solaris 소프트웨어는 장치에 대한 액세스 제어를 위한 두 가지 방법을 제공합니다. **장치 정책**은 시스템에 필수적인 장치에 대한 액세스를 제한하거나 막습니다. 장치 정책은 커널에서 적용됩니다. **장치 할당**은 주변 장치에 대한 액세스를 제한하거나 막습니다. 장치 할당은 사용자 할당 시 적용됩니다.

장치 정책은 권한을 사용하여 커널에서 선택된 장치를 보호합니다. 예를 들어, `bge`와 같은 네트워크 인터페이스에 대한 장치 정책은 읽기 또는 쓰기에 대해 모든 권한을 요구합니다.

장치 할당은 권한 부여를 사용하여 프린터나 마이크와 같은 주변 장치를 보호합니다. 기본적으로 장치 할당은 사용으로 설정되지 않습니다. 사용으로 설정되면 장치 할당을 구성하여 장치 사용을 막거나 장치에 액세스하기 위한 권한 부여를 요구할 수 있습니다. 장치 사용이 할당되면 현재 사용자가 할당을 해제할 때까지 다른 사용자는 장치에 액세스할 수 없습니다.

Oracle Solaris 시스템은 여러 영역에서 장치에 대한 액세스를 제어하도록 구성할 수 있습니다.

- **장치 정책 설정** – Oracle Solaris에서 특정 장치에 액세스하는 프로세스가 권한 집합을 사용하여 실행되도록 요구할 수 있습니다. 이러한 권한이 없는 프로세스는 장치를 사용할 수 없습니다. 부트 시 Oracle Solaris 소프트웨어는 장치 정책을 구성합니다. 설치 중 장치 정책에서 타사 드라이버를 구성할 수 있습니다. 설치 후 관리자는 장치 정책을 장치에 추가할 수 있습니다.
- **장치 할당 설정** – 장치 할당을 사용으로 설정하면 장치 사용을 한 번에 한 사용자로 제한할 수 있습니다. 사용자가 몇 가지 보안 요구 사항을 충족하도록 추가로 요구할 수 있습니다. 예를 들어, 장치를 사용하려면 사용자가 권한을 부여받도록 요구할 수 있습니다.
- **장치 사용 금지** – 컴퓨터 시스템에서 어떤 사용자도 마이크와 같은 장치를 사용할 수 없도록 막을 수 있습니다. 컴퓨터 키오스크는 특정 장치를 사용하지 못하게 할 수 있는 좋은 예입니다.
- **장치를 특정 영역으로 제한** – 장치 사용을 비전역 영역에 지정할 수 있습니다. 자세한 내용은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 "비전역 영역에서 장치 사용"을 참조하십시오. 장치 및 영역에 대한 일반적인 설명은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 "영역에 구성된 장치"를 참조하십시오.

장치 정책(개요)

장치 정책 방식을 사용하여 장치를 여는 프로세스에 특정 권한이 필요하도록 지정할 수 있습니다. 장치 정책으로 보호된 장치는 장치 정책에서 지정하는 권한을 사용하여 실행되는 프로세스만 액세스할 수 있습니다. Oracle Solaris는 기본 장치 정책을 제공합니다. 예를 들어, `bge0`과 같은 네트워크 인터페이스의 경우 인터페이스에 액세스하는 프로세스가 `net_rawaccess` 권한을 사용하여 실행되도록 요구할 수 있습니다. 요구 사항은 커널에서 적용됩니다. 권한에 대한 자세한 내용은 146 페이지 "권한(개요)"을 참조하십시오.

이전 릴리스에서 장치 노드는 파일 권한만으로 보호되었습니다. 예를 들어, `sys` 그룹이 소유한 장치는 `sys` 그룹의 구성원만 열 수 있었습니다. 이제 파일 권한으로는 누가 장치를 열 수 있는지 예측할 수 없습니다. 대신 장치는 파일 권한**과** 장치 정책으로 보호됩니다. 예를 들어, `/dev/ip` 파일은 `666` 권한을 가집니다. 하지만 장치는 해당 권한을 가진 프로세스만 열 수 있습니다.

장치 정책의 구성은 감사할 수 있습니다. `AUE_MODDEVPLCY` 감사 이벤트는 장치 정책의 변경 사항을 기록합니다.

장치 정책에 대한 자세한 내용은 다음을 참조하십시오.

- 78 페이지 "장치 정책 구성(작업 맵)"
- 89 페이지 "장치 정책 명령"
- 153 페이지 "권한 및 장치"

장치 할당(개요)

장치 할당 방식을 사용하여 CD-ROM과 같은 주변 장치에 대한 액세스를 제한할 수 있습니다. 방식은 수동으로 관리합니다. 장치 할당이 사용으로 설정되지 않은 경우 주변 장치는 파일 권한만으로 보호됩니다. 예를 들어, 기본적으로 주변 장치는 다음 용도로 사용할 수 있습니다.

- 모든 사용자가 디스켓이나 CD-ROM을 읽고 쓸 수 있습니다.
- 모든 사용자가 마이크를 연결할 수 있습니다.
- 모든 사용자가 연결된 프린터에 액세스할 수 있습니다.

장치 할당은 장치를 권한이 부여된 사용자로 제한합니다. 또한 장치 할당은 장치에 전혀 액세스하지 못하도록 막을 수 있습니다. 장치를 할당하는 사용자는 장치 할당을 해제할 때까지 해당 장치에 대한 배타적 사용 권한을 가집니다. 장치 할당이 해제되면 device-clean 스크립트가 남아 있는 데이터를 지웁니다. device-clean 스크립트를 작성하여 스크립트가 없는 장치에서 정보를 지울 수 있습니다. 예는 96 페이지 "새 Device-Clean 스크립트 작성"을 참조하십시오.

장치 할당, 장치 할당 해제 및 할당 가능 장치 나열 시도는 감사할 수 있습니다. 감사 이벤트는 `other` 감사 클래스의 일부입니다.

장치 할당에 대한 자세한 내용은 다음을 참조하십시오.

- 81 페이지 "장치 할당 관리(작업 맵)"
- 90 페이지 "장치 할당"
- 91 페이지 "장치 할당 명령"

시스템 리소스에 대한 액세스 제어

시스템 관리자는 시스템 작업을 제어하고 모니터할 수 있습니다. 누가 어떤 리소스를 사용할 수 있는지에 대한 제한을 설정할 수 있습니다. 리소스 사용을 기록하고, 누가 리소스를 사용하고 있는지 모니터할 수 있습니다. 또한 리소스의 부적절한 사용을 최소화하도록 시스템을 설정할 수 있습니다.

수퍼유저 제한 및 모니터링

시스템에서는 수퍼유저 액세스를 위한 root 암호를 요구합니다. 기본 구성에서 사용자는 root로 시스템에 원격으로 로그인할 수 없습니다. 원격으로 로그인할 때 사용자는 자신의 사용자 이름으로 로그인한 다음 su 명령을 사용하여 root가 되어야 합니다. su 명령을 사용하는 사용자(특히, 수퍼유저 액세스 권한을 얻으려고 시도하는 사용자)를 모니터할 수 있습니다. 수퍼유저를 모니터하고 수퍼유저에 대한 액세스를 제한하는 절차는 66 페이지 "수퍼유저 모니터 및 제한(작업)"을 참조하십시오.

수퍼유저를 대체하는 역할 기반 액세스 제어 구성

Oracle Solaris의 기능 중 하나인 역할 기반 액세스 제어(RBAC)는 수퍼유저의 기능을 관리 역할로 분산시키기 위해 설계되었습니다. 수퍼유저인 root 사용자는 시스템의 모든 리소스에 대한 액세스 권한을 가집니다. RBAC를 사용하여 root를 고유 권한의 역할 집합으로 대체할 수 있습니다. 예를 들어, 사용자 계정 만들기를 처리하는 하나의 역할을 설정하고 시스템 파일 수정을 처리하는 또 하나의 역할을 설정할 수 있습니다. 기능 또는 기능 집합을 처리하는 역할을 설정했으면 root의 기능에서 해당 기능을 제거할 수 있습니다.

각 역할에서는 알려진 사용자가 자신의 사용자 이름과 암호를 사용하여 로그인하도록 요구합니다. 로그인 후 사용자는 지정된 역할 암호를 사용하여 역할을 맡습니다. 결과적으로 root 암호를 알게 된 사람도 시스템을 제한적으로만 손상시킬 수 있게 됩니다. RBAC에 대한 자세한 내용은 135 페이지 "역할 기반 액세스 제어(개요)"를 참조하십시오.

시스템 리소스의 의도하지 않은 악용 방지

다음 방법으로 관리자 자신 및 사용자가 의도하지 않은 오류를 범하지 않도록 막을 수 있습니다.

- PATH 변수를 올바르게 설정하면 트로이 목마가 실행되지 않도록 할 수 있습니다.
- 제한된 셸을 사용자에게 지정할 수 있습니다. 제한된 셸은 사용자가 자신의 작업에 필요한 시스템 부분으로만 안내하여 사용자 오류를 막습니다. 실제로 신중하게 설정하면 사용자가 효율적으로 작업하도록 도움을 주는 시스템 부분만 액세스하도록 할 수 있습니다.
- 사용자가 액세스할 필요가 없는 파일에 대해서는 제한적인 권한을 설정할 수 있습니다.

PATH 변수 설정

신중을 기하여 PATH 변수를 올바르게 설정해야 합니다. 그렇지 않으면 다른 사람이 심어둔 프로그램을 의도하지 않게 실행할 수 있습니다. 침투 프로그램은 데이터나 시스템을 손상시킬 수 있습니다. 보안 위험을 유발하는 이러한 종류의 프로그램을 **트로이 목마**라고 합니다. 예를 들어, 대체 su 프로그램을 공용 디렉토리에 두면 시스템 관리자가 이 대체 프로그램을 실행할 수 있습니다. 이러한 스크립트는 정상적인 su 명령과 똑같이 보입니다. 스크립트는 실행 후 자신을 제거하기 때문에 실제로 트로이 목마를 실행했음을 보여주는 증거가 없을 수 있습니다.

PATH 변수는 로그인 시 자동으로 설정됩니다. 경로는 .bashrc 및 /etc/profile과 같은 초기화 파일을 통해 설정됩니다. 현재 디렉토리(.)가 마지막에 오도록 사용자 검색 경로를 설정하면 이러한 종류의 트로이 목마 실행으로부터 보호됩니다. root 계정에 대한 PATH 변수에는 현재 디렉토리가 전혀 포함되면 안됩니다.

사용자에게 제한된 셸 지정

표준 셸은 사용자의 파일 열기, 명령 실행 등을 허용합니다. 제한된 셸은 사용자의 권한을 디렉토리 변경 및 명령 실행으로 제한합니다. 제한될 셸은 /usr/lib/rsh 명령으로 호출됩니다. 제한된 셸은 원격 셸(/usr/sbin/rsh)이 아닙니다.

제한된 셸은 다음과 같이 표준 셸과 다릅니다.

- 사용자는 사용자의 홈 디렉토리로 제한되므로 cd 명령을 사용하여 디렉토리를 변경할 수 없습니다. 따라서 사용자는 시스템 파일을 찾아볼 수 없습니다.
- 사용자는 PATH 변경할 수 없으므로 시스템 관리자가 설정한 경로에 있는 명령만 사용할 수 있습니다. 또한 사용자는 전체 경로 이름을 사용하여 명령이나 스크립트를 실행할 수 없습니다.
- 사용자는 > 또는 >>를 사용하여 출력을 리디렉션할 수 없습니다.

제한된 셸을 사용하여 사용자가 시스템 파일에 접근하지 못하도록 제한할 수 있습니다. 셸은 특정 작업을 수행해야 하는 사용자에 대한 제한된 환경을 만듭니다. 하지만 제한된

셸은 완전히 안전하지는 않으며, 능숙하지 않은 사용자가 의도하지 않게 시스템을 손상시키지 못하도록 하기 위한 목적으로만 설계되었습니다.

제한된 셸에 대한 자세한 내용은 `man -s1m rsh` 명령을 사용하여 `rsh(1M)` 매뉴얼 페이지를 참조하십시오.

파일의 데이터에 대한 액세스 제한

Oracle Solaris는 다중 사용자 환경이므로 파일 시스템 보안은 시스템에서 가장 기본적인 보안 위험입니다. 전통적인 UNIX 파일 보호를 사용하여 파일을 보호할 수 있습니다. 또한 보안 안전한 액세스 제어 목록(ACL)을 사용할 수 있습니다.

일부 사용자에게는 특정 파일을 읽을 수 있도록 허용하고, 다른 사용자에게는 특정 파일을 변경하거나 삭제할 수 있는 권한을 부여하고자 할 수 있습니다. 또한 다른 사용자에게 공개하고 싶지 않은 데이터가 있을 수 있습니다. 7 장, “파일에 대한 액세스 제어(작업)”에서 파일 권한을 설정하는 방법에 대해 설명합니다.

setuid 실행 파일 제한

실행 파일은 보안 위험이 될 수 있습니다. 많은 실행 프로그램은 제대로 작동하려면 `root`로 실행되어야 합니다. 이러한 `setuid` 프로그램은 사용자 ID를 `0`으로 설정하여 실행됩니다. 이러한 프로그램을 실행하는 사용자는 프로그램을 `root` ID로 실행하게 됩니다. `root` ID로 실행되는 프로그램은 보안을 염두에 두고 작성되지 않은 경우 보안 문제를 야기할 수 있습니다.

`setuid` 비트가 `root`로 설정된 Oracle에서 제공하는 실행 파일을 제외하고 `setuid` 프로그램의 사용을 허용하지 않아야 합니다. `setuid` 프로그램의 사용을 허용할 수 밖에 없는 경우 해당 사용을 제한해야 합니다. 보안 관리에는 `setuid` 프로그램이 거의 필요하지 않습니다.

자세한 내용은 123 페이지 “보안 손상으로부터 실행 파일 보호”를 참조하십시오. 절차는 129 페이지 “보안 위험이 있는 프로그램 보호(작업 맵)”를 참조하십시오.

기본 보안 구성 사용

기본적으로 Oracle Solaris가 설치되면 많은 수의 네트워크 서비스가 사용 안함으로 설정됩니다. 이 구성을 “기본 보안”(SBD, Secure by Default)이라고 합니다. SBD에서 네트워크 요청을 허용하는 유일한 네트워크 서비스는 `sshd` 데몬입니다. 기타 모든 네트워크 서비스는 사용 안함으로 설정되거나 로컬 요청만 처리합니다. `ftp`와 같은 개별 네트워크 서비스를 사용으로 설정하려면 Oracle Solaris의 SMF(서비스 관리 기능)를 사용합니다. 자세한 내용은 `netservices(1M)` 및 `smf(5)` 매뉴얼 페이지를 참조하십시오.

리소스 관리 기능 사용

Oracle Solaris 소프트웨어는 정교한 리소스 관리 기능을 제공합니다. 이러한 기능을 사용하여 서버 통합 환경에서 응용 프로그램별로 리소스 사용 할당, 일정 설정, 모니터 및 할당량 설정이 가능합니다. 리소스 제어 프레임워크를 사용하여 프로세스에서 소비하는 시스템 리소스에 대한 제약 조건을 설정할 수 있습니다. 이러한 제약 조건은 시스템 리소스 과다 사용을 시도하는 스크립트를 사용한 서비스 거부 공격을 막는 데 도움이 됩니다.

Oracle Solaris 리소스 관리 기능을 사용하여 특정 프로젝트에 대한 리소스를 지정할 수 있습니다. 또한 사용 가능한 리소스를 동적으로 조정할 수 있습니다. 자세한 내용은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 제I부, "Oracle Solaris 리소스 관리"를 참조하십시오.

Oracle Solaris 영역 사용

Oracle Solaris 영역은 Oracle Solaris OS의 단일 인스턴스 내에서 프로세스가 시스템의 나머지 부분과 격리된 응용 프로그램 실행 환경을 제공합니다. 이러한 분리는 하나의 영역에서 실행되는 프로세스가 다른 영역에서 실행되는 프로세스를 모니터링하거나 영향을 미치는 것을 방지합니다. 수퍼유저 권한으로 실행되는 프로세스라도 다른 영역의 작업을 보거나 영향을 미칠 수 없습니다.

Oracle Solaris 영역은 여러 응용 프로그램을 단일 서버에 두는 환경에 이상적입니다. 자세한 내용은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 제II부, "Oracle Solaris Zones"을 참조하십시오.

시스템 리소스 사용 모니터링

시스템 관리자는 시스템 작업을 모니터해야 합니다. 관리자는 다음을 포함하여 시스템의 모든 측면을 파악하고 있습니다.

- 정상적인 로드는 어느 정도인가?
- 누가 시스템에 대한 액세스 권한을 가지고 있는가?
- 개인 사용자가 언제 시스템에 액세스하는가?
- 시스템에서 일반적으로 어떤 프로그램이 실행되는가?

이러한 종류의 정보를 알고 있으면 사용 가능한 도구를 사용하여 시스템 사용을 감사하고 개별 사용자의 작업을 모니터할 수 있습니다. 모니터링은 보안 침입이 의심될 때 매우 유용합니다. 감사 서비스에 대한 자세한 내용은 26 장, "감사(개요)"를 참조하십시오.

파일 무결성 모니터링

시스템 관리자는 관리하는 시스템에 설치된 파일이 예상치 않은 방법으로 변경되지 않았는지 확인해야 합니다. 대규모 설치에서는 각 시스템의 소프트웨어 스택에 대한 비교 및 보고 도구를 사용하여 시스템을 추적할 수 있습니다. 기본 감사 보고 도구(BART)를 사용하여 시간에 따른 여러 시스템의 파일 레벨 검사를 수행함으로써 시스템을 종합적으로 검증할 수 있습니다. 여러 시스템에 걸쳐 또는 시간에 따라 한 시스템에서 BART **매니페스트**에 대한 변경 사항으로 시스템의 무결성을 검증할 수 있습니다. BART는 매니페스트 만들기, 매니페스트 비교 및 보고서 작성을 위한 규칙을 제공합니다. 자세한 내용은 6 장, “기본 감사 보고 도구 사용(작업)”을 참조하십시오.

파일에 대한 액세스 제어

Oracle Solaris는 다중 사용자 환경입니다. 다중 사용자 환경에서는 시스템에 로그인한 모든 사용자가 다른 사용자에게 속한 파일을 읽을 수 있습니다. 적절한 파일 권한을 가진 사용자는 다른 사용자에게 속한 파일을 사용할 수도 있습니다. 자세한 내용은 7 장, “파일에 대한 액세스 제어(작업)”를 참조하십시오. 적절한 파일 권한 설정에 대한 단계별 지침은 123 페이지 “파일 보호(작업)”를 참조하십시오.

암호화를 사용하여 파일 보호

다른 사용자가 파일에 액세스하지 못하도록 하여 파일을 안전하게 유지할 수 있습니다. 예를 들어, `600` 권한의 파일은 소유자 및 수퍼유저를 제외하고 읽을 수 없습니다. `700` 권한의 디렉토리도 마찬가지로 액세스할 수 없습니다. 하지만 암호를 알아내거나 `root` 암호를 알게 된 사람은 해당 파일에 액세스할 수 있습니다. 또한 다르게 액세스할 수 없는 파일은 시스템 파일이 오프라인 매체로 백업될 때마다 백업 테이프에 보존됩니다.

암호화 프레임워크는 파일을 보호하기 위한 `digest`, `mac` 및 `encrypt` 명령을 제공합니다. 자세한 내용은 11 장, “암호화 프레임워크(개요)”를 참조하십시오.

액세스 제어 목록 사용

ACL는 파일 권한에 비해 뛰어난 제어 기능을 제공할 수 있습니다. 전통적인 UNIX 파일 보호로는 충분하지 않을 때 ACL을 추가합니다. 전통적인 UNIX 파일 보호는 소유자, 그룹 및 기타의 세 사용자 클래스에 대해 읽기, 쓰기 및 실행 권한을 제공합니다. ACL은 더욱 세밀한 파일 보안을 제공합니다.

ACL을 사용하면 다음을 포함하여 세밀한 파일 권한을 정의할 수 있습니다.

- 소유자 파일 권한
- 소유자 그룹에 대한 파일 권한
- 소유자 그룹 외부의 다른 사용자에 대한 파일 권한

- 특정 사용자에 대한 파일 권한
- 특정 그룹에 대한 파일 권한
- 이전의 각 범주에 대한 기본 권한

ACL 사용에 대한 자세한 내용은 122 페이지 “액세스 제어 목록을 사용하여 UFS 파일 보호”를 참조하십시오. 액세스 제어 목록(ACL)을 사용하여 ZFS 파일을 보호하려면 **Oracle Solaris 관리: ZFS 파일 시스템**의 8 장, “ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호”을 참조하십시오.

시스템 간 파일 공유

네트워크 파일 서버는 공유 가능한 파일을 제어할 수 있습니다. 또한 네트워크 파일 서버는 어떤 클라이언트가 파일에 대한 액세스 권한을 가지고 이러한 클라이언트에 대해 어떤 유형의 액세스가 허용되는지 제어할 수 있습니다. 일반적으로 파일 서버는 모든 클라이언트나 특정 클라이언트에게 읽기-쓰기 액세스 권한 또는 읽기 전용 액세스 권한을 부여할 수 있습니다. 액세스 제어는 `share` 명령을 사용하여 리소스를 사용할 수 있게 되었을 때 지정됩니다.

ZFS 파일 시스템의 NFS 공유를 만들 경우 파일 시스템은 공유를 제거할 때까지 영구적으로 공유됩니다. 시스템이 재부트되면 SMF에서 공유를 자동으로 관리합니다. 자세한 내용은 **Oracle Solaris 관리: ZFS 파일 시스템**의 3 장, “Oracle Solaris ZFS와 전통적인 파일 시스템의 차이”를 참조하십시오.

공유 파일에 대한 root 액세스 제한

일반적으로 수퍼유저는 네트워크에서 공유된 파일 시스템에 대한 `root` 액세스가 허용되지 않습니다. NFS 시스템은 수퍼유저 사용자를 사용자 ID `60001`의 사용자 `nobody`로 변경하여 마운트된 파일 시스템에 대한 `root` 액세스를 막습니다. 사용자 `nobody`의 액세스 권한은 공용에 부여된 액세스 권한과 동일합니다. 사용자 `nobody`는 자격 증명이 없는 사용자의 액세스 권한을 가집니다. 예를 들어, 공용이 파일에 대한 실행 권한만 가지는 경우 사용자 `nobody`는 해당 파일을 실행할 수만 있습니다.

NFS 서버는 호스트별로 공유 파일 시스템에 대한 `root` 액세스 권한을 부여할 수 있습니다. 이러한 권한을 부여하려면 `share` 명령에 `root=`*hostname* 옵션을 사용합니다. 이 옵션은 신중하게 사용해야 합니다. NFS에서 보안 옵션에 대한 자세한 내용은 **Oracle Solaris 관리: 네트워크 서비스**의 6 장, “네트워크 파일 시스템 액세스(참조)”를 참조하십시오.

네트워크 액세스 제어

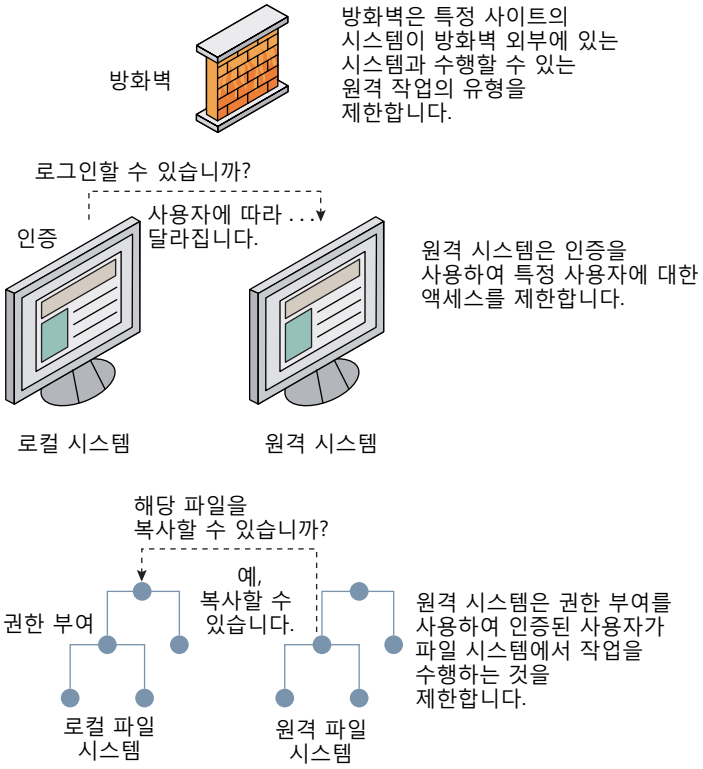
컴퓨터는 컴퓨터 **네트워크**의 일부인 경우가 많습니다. 네트워크를 통해 연결된 컴퓨터는 정보를 교환할 수 있습니다. 네트워크 컴퓨터는 네트워크에 있는 다른 컴퓨터의 데이터 및 기타 리소스에 액세스할 수 있습니다. 컴퓨터 네트워크는 강력하면서 정교한 컴퓨팅 환경을 만듭니다. 하지만 네트워크는 컴퓨터 보안을 복잡하게 만듭니다.

예를 들어, 컴퓨터 네트워크 내에서 개별 시스템은 정보의 공유를 허용합니다. 허용되지 않은 액세스는 보안 위험입니다. 많은 사람들이 네트워크에 액세스하기 때문에 (특히 사용자 오류를 통해) 허용되지 않은 액세스가 발생할 가능성이 높습니다. 암호의 부적절한 사용도 허용되지 않은 액세스를 허용할 수 있습니다.

네트워킹 보안 방식

네트워크 보안은 대개 원격 시스템의 작업 제한이나 차단을 기반으로 합니다. 다음 그림은 원격 작업에 지정할 수 있는 보안 제한 사항을 설명합니다.

그림 2–1 원격 작업에 대한 보안 제한 사항



원격 액세스에 대한 인증 및 권한 부여

인증은 특정 사용자가 원격 시스템에 액세스할 때 이러한 사용자에 대한 액세스를 제한할 수 있는 방법입니다. 인증은 시스템 레벨 및 네트워크 레벨 모두에서 설정할 수 있습니다. 사용자에게 원격 시스템에 대한 액세스 권한이 부여된 후 **권한 부여**를 통해 사용자가 수행할 수 있는 작업을 제한할 수 있습니다. 다음 표는 인증 및 권한 부여를 제공하는 서비스를 나열합니다.

표 2–3 원격 액세스에 대한 인증 서비스

서비스	설명	자세한 정보
IPsec	IPsec은 호스트 기반 및 인증서 기반 인증과 네트워크 트래픽 암호화를 제공합니다.	**Oracle Solaris 관리: IP 서비스**의 14 장, “IP 보안 아키텍처(개요)”
Kerberos	Kerberos는 암호화를 사용하여 시스템에 로그인하는 사용자를 인증하고 권한을 부여합니다.	예는 328 페이지 “Kerberos 서비스의 작동 방식”을 참조하십시오.

표 2–3 원격 액세스에 대한 인증 서비스 *(계속)*

서비스	설명	자세한 정보
LDAP	LDAP 디렉토리 서비스는 네트워크 레벨에서 인증과 권한 부여를 모두 제공할 수 있습니다.	**Oracle Solaris Administration: Naming and Directory Services**
원격 로그인 명령	원격 로그인 명령을 사용하여 사용자가 네트워크를 통해 원격 시스템에 로그인하고 해당 리소스를 사용할 수 있습니다. 원격 로그인 명령에는 `rlogin`, `rcp` 및 `ftp`가 포함됩니다. "신뢰할 수 있는 호스트"인 경우 인증이 자동으로 이루어집니다. 그렇지 않은 경우 자신에 대한 인증 요청을 받습니다.	**Oracle Solaris 관리: 네트워크 서비스**의 29 장, "원격 시스템 액세스(작업)"
SASL	SASL(Simple Authentication and Security Layer)은 네트워크 프로토콜에 인증 및 선택적 보안 서비스를 제공하는 프레임워크입니다. 플러그인을 사용하여 적절한 인증 프로토콜을 선택할 수 있습니다.	289 페이지 "SASL(개요)"
보안 RPC	보안 RPC는 원격 시스템에서 요청하는 사용자를 인증하여 네트워크 환경의 보안을 향상시킵니다. 보안 RPC에 대해 UNIX, DES 또는 Kerberos 인증 시스템을 사용할 수 있습니다.	267 페이지 "보안 RPC 개요"
	또한 보안 RPC는 NFS 환경에서 추가 보안을 제공하는 데 사용될 수도 있습니다. 보안 RPC를 사용하는 NFS 환경을 보안 NFS라고 합니다. 보안 NFS에서는 공개 키에 대해 Diffie-Hellman 인증을 사용합니다.	267 페이지 "NFS 서비스 및 보안 RPC"
Secure Shell	Secure Shell은 보안되지 않은 네트워크를 통해 네트워크 트래픽을 암호화합니다. Secure Shell은 암호, 공개 키 또는 둘 다 사용하여 인증을 제공합니다. Secure Shell은 공개 키에 대해 RSA 및 DSA 인증을 사용합니다.	293 페이지 "Secure Shell(개요)"

보안 RPC에 대한 가능한 대안은 Oracle Solaris **권한 있는 포트** 방식입니다. 권한 있는 포트에는 `1024`보다 작은 포트 번호가 지정됩니다. 클라이언트 시스템이 클라이언트의 자격 증명을 인증한 후 클라이언트는 권한 있는 포트를 사용하여 서버에 대한 연결을 설정합니다. 그러면 서버는 연결 포트 번호를 검사하여 클라이언트 자격 증명을 확인합니다.

Oracle Solaris 소프트웨어를 실행하지 않는 클라이언트는 권한 있는 포트를 사용하여 통신하지 못할 수 있습니다. 클라이언트가 포트를 통해 통신할 수 없을 경우 다음과 유사한 오류 메시지가 나타납니다.

```
"Weak Authentication
NFS request from unprivileged port"
```

방화벽 시스템

방화벽 시스템을 설정하여 외부 액세스로부터 네트워크의 리소스를 보호할 수 있습니다. **방화벽 시스템**은 내부 네트워크와 외부 네트워크 사이에서 장벽 기능을 수행하는 보안 호스트입니다. 내부 네트워크는 모든 다른 네트워크를 신뢰할 수 없는 네트워크로 취급합니다. 이 설정은 내부 네트워크와 통신하는 외부 네트워크(예: 인터넷) 사이에서 필수로 고려해야 합니다.

방화벽은 게이트웨이 및 장벽의 기능을 수행합니다. 방화벽은 네트워크 사이에서 데이터를 전달하는 게이트웨이의 기능을 수행합니다. 방화벽은 데이터가 네트워크를 자유롭게 통과하지 못하도록 차단하는 방벽의 기능을 수행합니다. 방화벽은 원격 네트워크의 호스트에 액세스하려면 내부 네트워크의 사용자에게 방화벽 시스템에 로그인하도록 요구합니다. 마찬가지로 외부 네트워크의 사용자는 내부 네트워크의 호스트에 대한 액세스 권한을 부여받기 전에 먼저 방화벽 시스템에 로그인해야 합니다.

또한 방화벽은 일부 내부 네트워크 사이에서도 유용할 수 있습니다. 예를 들어, 방화벽이나 보안 게이트웨이 컴퓨터를 설정하여 패킷 전송을 제한할 수 있습니다. 게이트웨이 컴퓨터가 패킷의 소스 주소 또는 대상 주소가 아닌 경우 게이트웨이는 두 네트워크 간의 패킷 교환을 금지할 수 있습니다. 또한 방화벽은 특정 프로토콜에 대해서만 패킷을 전달하도록 설정해야 합니다. 예를 들어, 메일 전송을 위한 패킷은 허용하지만 `telnet` 또는 `rlogin` 명령에 대한 패킷은 허용하지 않을 수 있습니다.

또한 내부 네트워크에서 보내는 모든 전자 메일은 먼저 방화벽 시스템으로 보내집니다. 그러면 방화벽에서 메일을 외부 네트워크의 호스트로 전송합니다. 또한 방화벽은 모든 수신 전자 메일을 받아서 메일을 내부 네트워크의 호스트로 배포합니다.

주의 – 방화벽은 권한이 부여되지 않은 사용자가 네트워크의 호스트에 액세스하지 못하도록 막습니다. 방화벽에서는 엄격하고 강력하게 적용되는 보안을 유지하는 반면, 네트워크의 다른 호스트에 대한 보안은 여유롭게 설정할 수 있습니다. 하지만 이 경우 방화벽 시스템에 침투할 수 있는 공격자는 내부 네트워크의 다른 모든 호스트에 대한 액세스 권한을 얻게 됩니다.

방화벽 시스템에는 신뢰할 수 있는 호스트가 없어야 합니다. **신뢰할 수 있는 호스트**는 사용자가 암호를 제공하지 않고도 로그인할 수 있는 호스트입니다. 방화벽 시스템은 해당 파일 시스템을 공유하거나 다른 서버의 파일 시스템을 마운트하면 안됩니다.

Oracle Solaris의 IPsec 및 IP 필터 기능이 방화벽 보호 기능을 제공할 수 있습니다. 네트워크 트래픽 보호에 대한 자세한 내용은 **Oracle Solaris 관리: IP 서비스**의 제III부, "IP 보안"을 참조하십시오.

암호화 및 방화벽 시스템

대부분의 로컬 영역 네트워크에서는 **패킷**이라고 하는 블록으로 컴퓨터 간에 데이터를 전송합니다. **패킷 스매싱**이라고 하는 절차를 통해 외부 네트워크에서 권한이 부여되지 않은 사용자가 데이터를 손상시킬 수 있습니다.

패킷 스매싱의 경우 패킷이 대상에 도달하기 전에 패킷을 가로챕니다. 그런 다음 침입자는 임의의 데이터를 컨텐츠에 주입하고 패킷을 다시 원래 경로에 둡니다. 로컬 영역 네트워크에서는 패킷이 서버를 포함한 모든 시스템에 동시에 도달하기 때문에 패킷 스매싱은 불가능합니다. 하지만 패킷 스매싱은 게이트웨이에서 가능하므로 네트워크의 모든 게이트웨이를 보호해야 합니다.

가장 위험한 공격은 데이터의 무결성에 영향을 미치는 것입니다. 이러한 공격에는 패킷 컨텐츠 변경 또는 사용자 가장이 포함됩니다. 도청이 포함된 공격은 데이터 무결성을 침해하지 못합니다. 도청은 나중에 재생을 위해 대화를 녹음합니다. 도청은 사용자를 가장하지 못합니다. 도청 공격은 데이터 무결성을 공격하지 못하지만 개인 정보에 영향을 줍니다. 네트워크를 이동하는 데이터를 암호화하여 개인 정보를 포함한 민감한 정보를 보호할 수 있습니다.

- 보안되지 않은 네트워크를 통한 원격 작업을 암호화하려면 17 장, “Secure Shell 사용(작업)”을 참조하십시오.
- 네트워크를 통과하는 데이터를 암호화하고 인증하려면 19 장, “Kerberos 서비스 소개”를 참조하십시오.
- IP 데이터그램을 암호화하려면 **Oracle Solaris 관리: IP 서비스**의 14 장, “IP 보안 아키텍처(개요)”를 참조하십시오.

보안 문제 보고

의심스런 보안 침입이 발생할 경우 CERT/CC(Computer Emergency Response Team/Coordination Center)에 문의할 수 있습니다. CERT/CC는 Carnegie Mellon University의 Software Engineering Institute에 위치한 DARPA(Defense Advanced Research Projects Agency) 펀드 프로젝트입니다. 이 기관에서 보안 문제를 해결하는 데 도움을 줄 수 있습니다. 또한 이 기관은 특정 요구에 더욱 알맞은 다른 Computer Emergency Response Team을 안내해 줄 수 있습니다. 최신 연락처 정보는 CERT/CC (`http://www.cert.org/contact_cert/`) 웹 사이트를 참조하십시오.

◆ ◆ ◆ 3 장

3 시스템에 대한 액세스 제어(작업)

이 장에서는 Oracle Solaris 시스템에 액세스할 수 있는 사용자를 제어하는 절차에 대해 설명합니다.

다음은 이 장에 포함된 정보 목록입니다.

시스템 보안에 대한 개략적인 내용은 2 장, "시스템 보안 관리(개요)"를 참조하십시오.

시스템 액세스 제어(작업 맵)

컴퓨터는 가장 약한 시작점 수준으로 보안됩니다. 다음 작업 맵에서는 모니터 및 보안해야 할 영역을 보여 줍니다.

작업	설명	수행 방법
사용자 로그인을 모니터, 허용 및 거부합니다.	비정상적인 로그인 작업을 모니터합니다. 일시적으로 로그인을 방지합니다.	58 페이지 "로그인 및 암호 보안(작업 맵)"
강력한 암호 보안 처리를 제공합니다.	사용자 암호 보안 처리 알고리즘을 지정합니다. 추가 알고리즘을 설치합니다.	63 페이지 "기본 암호 보안 처리 알고리즘 변경(작업)"
수퍼유저 작업을 모니터 및 제한합니다.	정기적으로 수퍼유저 작업을 모니터합니다. root 사용자의 원격 로그인을 방지합니다.	66 페이지 "수퍼유저 모니터 및 제한(작업)"
하드웨어 설정에 대한 액세스를 방지합니다.	일반 사용자가 PROM에 액세스하지 못하도록 합니다.	68 페이지 "시스템 하드웨어에 대한 액세스 제어(작업)"

로그인 및 암호 보안(작업)

원격 로그인을 제한하고, 사용자에게 암호를 가지도록 요구하고, root 계정이 복잡한 암호를 가지도록 할 수 있습니다. 실패한 액세스 시도를 모니터하고 일시적으로 로그인을 사용 안함으로 설정할 수도 있습니다.

로그인 및 암호 보안(작업 맵)

다음 작업 맵에서는 사용자 로그인을 모니터하고 사용자 로그인을 사용 안함으로 설정하는 절차에 대해 설명합니다.

작업	설명	수행 방법
root 암호를 변경합니다.	root 계정이 암호 요구 사항을 준수하는지 확인합니다.	58 페이지 “root 암호 변경 방법”
사용자의 로그인 상태를 표시합니다.	사용자의 로그인 계정에 대한 포괄적인 정보(예: 전체 이름 및 암호 에이징 정보)를 나열합니다.	59 페이지 “사용자의 로그인 상태 표시 방법”
암호가 없는 사용자를 찾습니다.	계정에 암호가 필요하지 않은 사용자만 찾습니다.	60 페이지 “암호가 없는 사용자 표시 방법”
일시적으로 로그인을 사용 안함으로 설정합니다.	시스템 종료 또는 일반적인 유지 관리의 일부로 시스템에 대한 사용자 로그인을 거부합니다.	60 페이지 “일시적으로 사용자 로그인을 사용 안함으로 설정하는 방법”
실패한 로그인 시도를 저장합니다.	다섯 번의 시도 후에도 올바른 암호를 제공하지 못한 사용자의 로그를 만듭니다.	61 페이지 “실패한 로그인 시도 모니터 방법”
실패한 모든 로그인 시도를 저장합니다.	실패한 로그인 시도의 로그를 만듭니다.	62 페이지 “실패한 모든 로그인 시도 모니터 방법”

▼ root 암호 변경 방법

root 암호를 변경할 때는 시스템의 모든 사용자에게 적용되는 암호 요구 사항을 준수해야 합니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

- **암호를 변경합니다.**

```
# passwd root
New Password:
Re-enter new Password:
passwd: password successfully changed for root
```

암호가 요구 사항을 준수하지 않을 경우 화면에 메시지가 인쇄됩니다. 메시지는 정보 제공용입니다. 세 번의 시도 후 명령을 다시 실행하여 암호를 변경해야 합니다.

```
passwd: Password too short - must be at least 6 characters.
passwd: The password must contain at least 2 alphabetic character(s).
passwd: The password must contain at least 1 numeric or special character(s).
```

▼ 사용자의 로그인 상태 표시 방법

시작하기 전에 root 역할을 가진 사용자여야 합니다.

- **logins 명령을 사용하여 사용자의 로그인 상태를 표시합니다.**

  ```
  # logins -x -l username
  ```

  `-x` 확장된 일련의 로그인 상태 정보를 표시합니다.

  `-l` *username* 지정된 사용자에 대한 로그인 상태를 표시합니다. *username* 변수는 사용자의 로그인 이름입니다. 여러 로그인 이름은 쉼표로 구분됩니다.

  logins 명령은 적절한 암호 데이터베이스를 사용하여 사용자의 로그인 상태를 가져옵니다. 데이터베이스는 로컬 /etc/passwd 파일 또는 이름 지정 서비스용 암호 데이터베이스일 수 있습니다. 자세한 내용은 logins(1M) 매뉴얼 페이지를 참조하십시오.

예 3–1 사용자의 로그인 상태 표시

다음 예에서는 사용자 jdoe에 대한 로그인 상태가 표시됩니다.

```
# logins -x -l jdoe
jdoe       500     staff           10   Jaylee Jaye Doe
                   /home/jdoe
                   /bin/bash
                   PS 010103 10 7 -1
```

`jdoe` 사용자의 로그인 이름을 식별합니다.

`500` 사용자 ID(UID)를 식별합니다.

`staff` 사용자의 기본 그룹을 식별합니다.

`10` 그룹 ID(GID)를 식별합니다.

Jaylee Jaye Doe 설명을 식별합니다.

`/home/jdoe` 사용자의 홈 디렉토리를 식별합니다.

`/bin/bash` 로그인 셸을 식별합니다.

```
PS 010170 10 7 -1
```

다음과 같은 암호 에이징 정보를 지정합니다.

- 마지막으로 암호가 변경된 날짜
- 다음 번 변경까지 필요한 기간(일)
- 변경이 필요하기까지 남은 기간(일)
- 경고 기간

▼ 암호가 없는 사용자 표시 방법

시작하기 전에 root 역할을 가진 사용자여야 합니다.

● **logins 명령을 사용하여 암호가 없는 사용자를 모두 표시합니다.**

```
# logins -p
```

-p 옵션은 암호가 없는 사용자 목록을 표시합니다. logins 명령은 분산된 이름 지정 서비스가 nsswitch.conf 파일에 지정되지 않은 경우 로컬 시스템의 passwd 데이터베이스를 사용합니다.

예 3–2 암호가 없는 사용자 표시

다음 예에서는 사용자 pmorph에게 암호가 없습니다.

```
# logins -p
pmorph          501     other           1       Polly Morph
#
```

▼ 일시적으로 사용자 로그인을 사용 안함으로 설정하는 방법

시스템 종료 또는 일반적인 유지 관리 중 일시적으로 사용자 로그인을 사용 안함으로 설정합니다. 수퍼유저 로그인은 영향을 받지 않습니다. 자세한 내용은 nologin(4) 매뉴얼 페이지를 참조하십시오.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 **텍스트 편집기에서 /etc/nologin 파일을 만듭니다.**

```
# vi /etc/nologin
```

2 **시스템 가용성에 대한 메시지를 포함합니다.**

3 **파일을 닫은 후 저장합니다.**

예 3–3 사용자 로그인을 사용 안함으로 설정

이 예에서는 사용자에게 시스템을 사용할 수 없는 것으로 통지됩니다.

```
# vi /etc/nologin
(Add system message here)

# cat /etc/nologin
***No logins permitted.***

***The system will be unavailable until 12 noon.***
```

시스템을 실행 레벨 0(단일 사용자 모드)으로 전환하여 로그인을 사용 안함으로 설정할 수도 있습니다. 시스템을 단일 사용자 모드로 전환하는 방법은 **x86 플랫폼에서 Oracle Solaris 부트 및 종료**의 3 장, "시스템 종료(작업)"를 참조하십시오.

▼ 실패한 로그인 시도 모니터 방법

이 절차에서는 터미널 창의 실패한 로그인 시도를 캡처합니다. 이 절차에서는 데스크탑 로그인 시도의 실패한 로그인을 캡처하지 않습니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 **loginlog 파일을 /var/adm 디렉토리에 만듭니다.**

```
# touch /var/adm/loginlog
```

2 **loginlog 파일에서 root 사용자에 대한 읽기/쓰기 권한을 설정합니다.**

```
# chmod 600 /var/adm/loginlog
```

3 **loginlog 파일에서 그룹 멤버쉽을 sys로 변경합니다.**

```
# chgrp sys /var/adm/loginlog
```

4 **로그가 작동하는지 확인합니다.**

예를 들어, 잘못된 암호로 시스템에 다섯 번 로그인합니다. 그런 다음 /var/adm/loginlog 파일을 표시합니다.

```
# more /var/adm/loginlog
jdoe:/dev/pts/2:Tue Nov  4 10:21:10 2010
jdoe:/dev/pts/2:Tue Nov  4 10:21:21 2010
jdoe:/dev/pts/2:Tue Nov  4 10:21:30 2010
jdoe:/dev/pts/2:Tue Nov  4 10:21:40 2010
jdoe:/dev/pts/2:Tue Nov  4 10:21:49 2010
#
```

loginlog 파일에 실패한 시도마다 하나의 항목이 포함되어 있습니다. 각 항목에는 사용자의 로그인 이름, tty 장치 및 시도 실패 횟수가 들어 있습니다. 사용자의 시도 실패 횟수가 다섯 번 미만인 경우 실패한 시도가 기록되지 않습니다.

loginlog 파일이 커지면 컴퓨터 시스템에 침입하려는 시도가 있는 것일 수 있습니다. 따라서 정기적으로 이 파일의 내용을 확인하고 지우십시오. 자세한 내용은 loginlog(4) 매뉴얼 페이지를 참조하십시오.

▼ 실패한 모든 로그인 시도 모니터 방법

이 절차에서는 실패한 모든 로그인 시도를 syslog 파일에 캡처합니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 SYSLOG 및 SYSLOG_FAILED_LOGINS 에 대한 적합한 값을 사용하여 /etc/default/login 파일을 설정합니다.

항목이 변경되도록 /etc/default/login 파일을 편집합니다. **SYSLOG=YES**의 주석 처리가 해제되어 있는지 확인합니다.

```
# grep SYSLOG /etc/default/login
# SYSLOG determines whether the syslog(3) LOG_AUTH facility should be used
SYSLOG=YES
# The SYSLOG_FAILED_LOGINS variable is used to determine how many failed
#SYSLOG_FAILED_LOGINS=5
SYSLOG_FAILED_LOGINS=0
#
```

2 로깅 정보를 보유할 수 있는 올바른 권한으로 파일을 만듭니다.

a. authlog 파일을 /var/adm 디렉토리에 만듭니다.

```
# touch /var/adm/authlog
```

b. authlog 파일에서 root 사용자에 대한 읽기/쓰기 권한을 설정합니다.

```
# chmod 600 /var/adm/authlog
```

c. authlog 파일에서 그룹 멤버쉽을 sys로 변경합니다.

```
# chgrp sys /var/adm/authlog
```

3 실패한 암호 시도가 기록되도록 syslog.conf 파일을 편집합니다.

authlog 파일로 실패를 보냅니다.

a. syslog.conf 파일에 다음 항목을 입력합니다.

syslog.conf에 있는 동일한 행의 필드는 탭으로 구분됩니다.

```
auth.notice      <Press Tab>  /var/adm/authlog
```

b. system-log 서비스를 새로 고칩니다.

```
# svcadm refresh system/system-log
```

4 로그가 작동하는지 확인합니다.

예를 들어, 일반 사용자로 잘못된 암호를 사용하여 시스템에 로그인합니다. 그런 다음 수퍼유저로 /var/adm/authlog 파일을 표시합니다.

```
# more /var/adm/authlog
Nov  4 14:46:11 example1 login: [ID 143248 auth.notice]
 Login failure on /dev/pts/8 from example2, stacey
#
```

5 정기적으로 /var/adm/authlog 파일을 모니터합니다.

예 3–4 세 번의 로그인 실패 후 액세스 시도 기록

앞서 설명된 절차를 따르되, /etc/default/login 파일에서 SYSLOG_FAILED_LOGINS의 값을 3으로 설정합니다.

예 3–5 세 번의 로그인 실패 후 연결 해제

/etc/default/login 파일에서 RETRIES 항목의 주석 처리를 해제한 다음 RETRIES의 값을 3으로 설정합니다. 편집 내용이 즉시 적용됩니다. 한 세션에서 로그인을 세 번 재시도하면 시스템에서 연결을 해제합니다.

기본 암호 보안 처리 알고리즘 변경(작업)

기본적으로 사용자 암호는 crypt_sha256 알고리즘으로 보안 처리됩니다. 기본 암호 보안 처리 알고리즘을 변경하여 다른 암호화 알고리즘을 사용할 수 있습니다.

▼ 암호 보안 처리 알고리즘 지정 방법

이 절차에서는 MD5 알고리즘의 BSD-Linux 버전이 기본 암호화 알고리즘으로, 사용자가 암호를 변경할 때 사용됩니다. 이 알고리즘은 Oracle Solaris, BSD 및 Linux 버전의 UNIX를 실행하는 혼합 시스템 네트워크에 적합합니다. 암호 보안 처리 알고리즘 및 알고리즘 식별자 목록은 표 2–1을 참조하십시오.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

- **선택한 암호화 알고리즘에 대한 식별자를 지정합니다.**

  /etc/security/policy.conf 파일에서 CRYPT_DEFAULT 변수에 대한 값으로 식별자를 입력합니다.

  파일에서 선택한 내용을 설명할 주석을 처리할 수도 있습니다.

```
# cat  /etc/security/policy.conf
...
CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6
#
# Use the version of MD5 (5) that works with Linux and BSD systems.
# Passwords previously encrypted with SHA256 (1) will be encrypted
# with MD5 when users change their passwords.
#
#
#CRYPT_DEFAULT=5
CRYPT_DEFAULT=1
```

  이 예에서는 알고리즘 구성이 암호 보안 처리에 sha256 알고리즘이 사용되지 않도록 합니다. 암호가 sha256 모듈로 보안 처리된 사용자는 암호를 변경할 때 암호가 crypt_bsdmd5로 보안 처리됩니다.

  알고리즘 선택 구성에 대한 자세한 내용은 policy.conf(4) 매뉴얼 페이지를 참조하십시오.

예 3–6 이기종 환경에서의 암호 보안 처리 알고리즘 제약

이 예에서는 BSD 및 Linux 시스템이 포함된 네트워크에서 관리자가 암호를 모든 시스템에서 사용 가능한 것으로 구성합니다. 일부 네트워크 응용 프로그램에서는 SHA512 암호화를 처리할 수 없으므로 관리자가 허용되는 알고리즘 목록에 해당 식별자를 포함시키지 않습니다. 관리자는 SHA256 알고리즘 5를 CRYPT_DEFAULT 변수에 대한 값으로 유지합니다. CRYPT_ALGORITHMS_ALLOW 변수에는 BSD 및 Linux 시스템과 호환되는 MD5 식별자와 BSD 시스템과 호환되는 Blowfish 식별자가 포함됩니다. 5는 CRYPT_DEFAULT 알고리즘이므로 CRYPT_ALGORITHMS_ALLOW 목록에 나열되지 않아도 됩니다. 하지만 유지 관리를 위해 관리자는 CRYPT_ALGORITHMS_ALLOW 목록에 5를 배치하고 CRYPT_ALGORITHMS_DEPRECATE 목록에 사용되지 않은 식별자를 배치합니다.

```
CRYPT_ALGORITHMS_ALLOW=1,2a,5
#CRYPT_ALGORITHMS_DEPRECATE=__unix__,md5,6
CRYPT_DEFAULT=5
```

▼ NIS 도메인에 대한 새 암호 알고리즘 지정 방법

NIS 도메인의 사용자가 암호를 변경하면 NIS 클라이언트는 /etc/security/policy.conf 파일에서 로컬 알고리즘 구성을 확인합니다. NIS 클라이언트 시스템이 암호를 보안 처리합니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1. **NIS 클라이언트의 /etc/security/policy.conf 파일에서 암호 보안 처리 알고리즘을 지정합니다.**
2. **수정된 /etc/security/policy.conf 파일을 NIS 도메인의 모든 클라이언트 시스템에 복사합니다.**
3. **혼동을 최소화하기 위해 수정된 /etc/security/policy.conf 파일을 NIS 루트 서버 및 슬레이브 서버에 복사합니다.**

▼ LDAP 도메인에 대한 새 암호 알고리즘 지정 방법

LDAP 클라이언트가 제대로 구성되면 LDAP 클라이언트는 새 암호 알고리즘을 사용할 수 있습니다. LDAP 클라이언트의 작동 방식은 NIS 클라이언트의 작동 방식과 동일합니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1. **LDAP 클라이언트의 /etc/security/policy.conf 파일에서 암호 보안 처리 알고리즘을 지정합니다.**
2. **수정된 policy.conf 파일을 LDAP 도메인의 모든 클라이언트 시스템에 복사합니다.**
3. **클라이언트의 /etc/pam.conf 파일에 pam_ldap 모듈이 사용되지 않는지 확인합니다.**

   pam_ldap.so.1이 포함된 항목 앞에 주석 기호(#)가 있는지 확인합니다. server_policy 옵션은 pam_authtok_store.so.1 모듈과 함께 사용하지 마십시오.

   클라이언트의 pam.conf 파일에 있는 PAM 항목은 로컬 알고리즘 구성에 따라 암호가 보안 처리될 수 있도록 합니다. 또한 PAM 항목은 암호가 인증될 수 있도록 합니다.

   LDAP 도메인의 사용자가 암호를 변경하면 LDAP 클라이언트는 /etc/security/policy.conf 파일에서 로컬 알고리즘 구성을 확인합니다. LDAP 클라이언트 시스템이 암호를 보안 처리합니다. 그러면 클라이언트가 {crypt} 태그를 가진 보안 처리된 암호를 서버로 보냅니다. 서버에서는 태그를 통해 암호가 이미 보안 처리되었음을 확인하고, 암호를 그대로 저장합니다. 인증을 위해 클라이언트는 저장된 암호를 서버에서 검색합니다. 그런 다음 사용자가 입력한 암호를 기반으로 클라이언트가 생성한 암호화된 버전과 저장된 암호를 비교합니다.

주 – LDAP 서버에서 암호 정책 제어를 사용하려면 `pam.conf` 파일의 `pam_authtok_store` 항목과 함께 `server_policy` 옵션을 사용하십시오. 그러면 Oracle Directory Server Enterprise Edition의 암호화 방식을 사용하여 서버에서 암호가 보안 처리됩니다. 절차는 **Oracle Solaris Administration: Naming and Directory Services**의 11 장, "Setting Up Oracle Directory Server Enterprise Edition With LDAP Clients (Tasks)"을 참조하십시오.

수퍼유저 모니터 및 제한(작업)

역할 기반 액세스 제어(RBAC) 설정을 통해서도 수퍼유저 계정을 사용할 수 있습니다. RBAC에 대한 개략적인 내용은 135 페이지 "역할 기반 액세스 제어(개요)"를 참조하십시오. RBAC를 설정하려면 9 장, "역할 기반 액세스 제어 사용(작업)"을 참조하십시오.

▼ su 명령을 사용 중인 사용자 모니터 방법

`sulog` 파일은 사용자에서 수퍼유저로의 전환에 사용되는 `su` 시도뿐 아니라 `su` 명령의 모든 사용을 나열합니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

- **정기적으로 `/var/adm/sulog` 파일의 내용을 모니터합니다.**

```
# more /var/adm/sulog
SU 12/20 16:26 + pts/0 stacey-root
SU 12/21 10:59 + pts/0 stacey-root
SU 01/12 11:11 + pts/0 root-rimmer
SU 01/12 14:56 + pts/0 jdoe-root
SU 01/12 14:57 + pts/0 jdoe-root
```

항목에는 다음 정보가 표시됩니다.

- 명령이 입력된 날짜 및 시간
- 시도 성공 여부. 플러스 기호(+)는 시도 성공을 나타냅니다. 마이너스 기호(-)는 시도 실패를 나타냅니다.
- 명령이 실행된 포트
- 사용자 이름 및 전환된 ID의 이름

이 파일의 `su` 로깅은 `/etc/default/su` 파일의 다음 항목을 통해 기본으로 사용으로 설정되어 있습니다.

```
SULOG=/var/adm/sulog
```

일반 오류 ???가 포함된 항목은 su 명령에 대한 제어 터미널을 식별할 수 없음을 나타냅니다. 일반적으로 데스크탑이 표시되기 전에 시스템에서 su 명령을 호출하면 ???가 SU 10/10 08:08 + ??? root-root에서처럼 포함됩니다. 사용자가 데스크탑 세션을 시작한 후에는 ttynam 명령이 제어 터미널의 값을 sulog: SU 10/10 10:10 + pts/3 jdoe-root에 반환합니다.

su 명령이 명령줄에서 호출되지 않았음을 나타내는 항목은 SU 10/10 10:20 + ??? root-oracle과 유사하게 표시될 수 있습니다. Trusted Extensions 사용자가 GUI를 사용하여 oracle 역할로 전환되었을 수도 있습니다.

▼ 수퍼유저 로그인 제한 및 모니터 방법

이 방법은 root의 로컬 시스템에 대한 액세스 시도를 즉시 감지합니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 /etc/default/login 파일에서 CONSOLE 항목을 확인합니다.

```
CONSOLE=/dev/console
```

기본적으로 콘솔 장치는 /dev/console로 설정되어 있습니다. 이 설정을 사용하여 root가 콘솔에 로그인할 수 있습니다. root는 원격으로 로그인할 수 없습니다.

2 root가 원격으로 로그인할 수 없는지 확인합니다.

원격 시스템에서 root로 로그인해 봅니다.

```
mach2 % ssh -l root mach1
Password:       <Type root password of mach1>
Password:
Password:
Permission denied (gssapi-keyex,gssapi-with-mic,publickey,keyboard-interactive).
```

기본 구성에서 root가 역할이고, 역할은 로그인할 수 없습니다. 또한 기본 구성에서 ssh 프로토콜이 root 사용자 로그인을 막습니다.

3 root로 로그인하려는 시도를 모니터합니다.

기본적으로 root로 로그인하려는 시도는 SYSLOG 유틸리티를 통해 콘솔에 인쇄됩니다.

a. 데스크탑에서 터미널 콘솔을 엽니다.

b. 다른 창에서는 su 명령을 사용하여 수퍼유저로 로그인합니다.

```
% su -
Password:       <Type root password>
#
```

메시지가 터미널 콘솔에 인쇄됩니다.

```
Sep 7 13:22:57 mach1 su: 'su root' succeeded for jdoe on /dev/pts/6
```

예 3–7 수퍼유저 액세스 시도 기록

이 예에서는 수퍼유저 시도가 SYSLOG를 통해 기록되지 않습니다. 따라서 관리자가 /etc/default/su 파일의 #CONSOLE=/dev/console 항목에서 주석을 제거하여 해당 시도를 기록하고 있습니다.

```
# CONSOLE determines whether attempts to su to root should be logged
# to the named device
#
CONSOLE=/dev/console
```

사용자가 수퍼유저로 로그인하려고 시도하면 터미널 콘솔에 해당 시도가 인쇄됩니다.

```
SU 09/07 16:38 + pts/8 jdoe-root
```

일반 오류 /etc/default/login 파일에 기본 CONSOLE 항목이 포함된 경우 원격 시스템에서 수퍼유저로 로그인하려는 사용자는 먼저 자신의 사용자 이름으로 로그인해야 합니다. 사용자 이름으로 로그인한 후 su 명령을 사용하여 수퍼유저로 로그인할 수 있습니다.

콘솔에 Mar 16 16:20:36 mach1 login: ROOT LOGIN /dev/pts/14 FROM mach2.Example.COM과 유사한 항목이 표시되면 시스템에서 원격 root 로그인을 허용하고 있는 것입니다. 원격 수퍼유저 액세스를 방지하려면 /etc/default/login 파일에서 #CONSOLE=/dev/console 항목을 CONSOLE=/dev/console로 변경하십시오.

시스템 하드웨어에 대한 액세스 제어(작업)

하드웨어 설정에 액세스하려는 경우 암호를 제공하도록 하여 물리적 시스템을 보호할 수 있습니다. 사용자가 중단 시퀀스를 통해 윈도우화 시스템을 종료하지 못하도록 하여 시스템을 보호할 수도 있습니다.

BIOS를 보호하려면 공급업체 설명서를 참조하십시오.

▼ 하드웨어 액세스에 대한 암호 요구 방법

시작하기 전에 Device Security, Maintenance and Repair 또는 System Administrator 권한 프로파일이 지정되어 있어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**

   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2. **터미널 창에서 PROM 보안 모드를 입력합니다.**

```
# eeprom security-mode=command

Changing PROM password:
New password:       <Type password>
Retype new password:       <Retype password>
```

   command 또는 full 값을 선택합니다. 자세한 내용은 eeprom(1M) 매뉴얼 페이지를 참조하십시오.

   위 명령을 입력할 때 PROM 암호를 제공하라는 메시지가 표시되지 않을 경우 시스템에 이미 PROM 암호가 있는 것입니다.

3. **(옵션) PROM 암호를 변경하려면 다음 명령을 입력합니다.**

```
# eeprom security-password=       Press Return
Changing PROM password:
New password:       <Type password>
Retype new password:       <Retype password>
```

   새 PROM 보안 모드 및 암호는 즉시 적용되지만, 다음 번 부트 시 통지될 수도 있습니다.

주의 – PROM 암호를 잊어버리지 마십시오. 이 암호 없이는 하드웨어를 사용할 수 없습니다.

▼ 시스템 중단 시퀀스를 사용 안함으로 설정하는 방법

주 – 일부 서버 시스템에는 키 스위치가 있습니다. 키 스위치가 안전한 위치에서 설정된 경우 스위치가 소프트웨어 키보드 중단 설정을 대체합니다. 따라서 다음 절차를 통해 변경한 내용이 구현되지 않을 수 있습니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1. **KEYBOARD_ABORT의 값을 disable로 변경합니다.**

   /etc/default/kbd 파일의 enable 행을 주석 처리합니다. 그런 다음 disable 행을 추가합니다.

```
# cat /etc/default/kbd
...
# KEYBOARD_ABORT affects the default behavior of the keyboard abort
# sequence, see kbd(1) for details.  The default value is "enable".
```

```
# The optional value is "disable".  Any other value is ignored.
...
#KEYBOARD_ABORT=enable
KEYBOARD_ABORT=disable
```

2 **키보드 기본값을 업데이트합니다.**

```
# kbd -i
```

◆ ◆ ◆ 4 장

4 바이러스 검사 서비스(작업)

이 장에서는 바이러스 방지 소프트웨어 사용에 대한 정보를 제공하고 다음 항목을 다룹니다.

바이러스 검사 정보

데이터는 다양한 **검사 엔진**을 사용하는 검사 서비스인 Vscan을 통해 바이러스로부터 보호됩니다. 검사 엔진은 타사 응용 프로그램으로, 외부 호스트에 상주하며 파일에서 알려진 바이러스가 있는지 검사합니다. 파일 시스템이 Vscan 서비스를 지원하며 서비스가 사용으로 설정되었고 파일 유형이 제외되지 않은 경우 파일이 바이러스 검사 대상이 됩니다. 그런 다음 파일이 이전에 최신 바이러스 정의로 검사되지 않았거나 파일이 마지막 검사 이후 수정된 경우 열기 및 닫기 작업을 수행할 때 파일에 대해 바이러스 검사가 수행됩니다.

여러 검사 엔진을 사용하도록 Vscan 서비스를 구성할 수 있습니다. Vscan 서비스는 최소한 두 개의 검사 엔진을 사용하는 것이 좋습니다. 바이러스 검사 요청은 사용 가능한 모든 검사 엔진에 분산됩니다. 표 4–1에서는 최신 패치로 구성된 경우 지원되는 검사 엔진을 보여 줍니다.

표 4–1 바이러스 방지 검사 엔진 소프트웨어

바이러스 방지 소프트웨어	ICAP 지원
Symantec Antivirus Scan Engine 4.3	지원됨
Symantec Antivirus Scan Engine 5.1	지원됨

표 4–1 바이러스 방지 검사 엔진 소프트웨어 (계속)

바이러스 방지 소프트웨어	ICAP 지원
Computer Associates eTrust AntiVirus 7.1 Computer Associates Integrated Threat Management 8.1	지원되지 않음[1]
Trend Micro Interscan Web Security Suite (IWSS) 2.5	지원됨
McAfee Secure Internet Gateway 4.5	지원됨

[1] Computer Associates Antivirus Scan Engine용 Sun StorageTek 5000 NAS ICAP Server를 설치해야 합니다. Sun Download Center (http://www.oracle.com/technetwork/indexes/downloads/index.html)에서 패키지를 다운로드하십시오.

Vscan 서비스 정보

실시간 검사 방법의 이점은 파일이 사용되기 **전**에 최신 바이러스 정의로 검사된다는 것입니다. 이 접근 방식을 사용하면 바이러스로 인해 데이터가 손상되기 전에 바이러스를 감지할 수 있습니다.

다음은 바이러스 검사 프로세스에 대한 설명입니다.

1. 사용자가 클라이언트에서 파일을 열면 Vscan 서비스는 파일이 이전에 최신 바이러스 정의로 검사되었는지 여부 및 파일이 마지막 검사 이후 수정되었는지 여부를 기반으로 파일을 검사해야 할지 여부를 결정합니다.
   - 파일을 검사해야 할 경우 파일이 검사 엔진으로 전송됩니다. 검사 엔진에 대한 연결을 실패할 경우 다른 검사 엔진으로 파일이 전송됩니다. 사용 가능한 검사 엔진이 없을 경우 바이러스 검사가 실패하고 파일에 대한 액세스가 거부될 수 있습니다.
   - 파일을 검사하지 않아도 될 경우 클라이언트가 파일에 액세스할 수 있습니다.
2. 검사 엔진은 최신 바이러스 정의를 사용하여 파일을 검사합니다.
   - 바이러스가 감지되면 파일이 격리된 것으로 표시됩니다. 격리된 파일은 읽거나 실행하거나 이름을 바꿀 수 없지만 삭제할 수 있습니다. 시스템 로그는 격리된 파일의 이름과 바이러스 이름을 기록하고 감사가 사용으로 설정된 경우 동일한 정보의 감사 레코드가 만들어집니다.
   - 파일이 감염되지 않은 경우 파일에 검사 기록 태그가 지정되고 클라이언트가 파일에 액세스할 수 있습니다.

Vscan 서비스 사용(작업)

바이러스 검사 파일은 다음 요구 사항이 충족되는 경우 사용할 수 있습니다.

- 하나 이상의 검사 엔진이 설치 및 구성되어 있습니다.
- 파일이 바이러스 검사를 지원하는 파일 시스템에 상주합니다.
- 바이러스 검사가 파일 시스템에서 사용으로 설정되어 있습니다.
- Vscan 서비스가 사용으로 설정되어 있습니다.
- Vscan 서비스가 지정된 파일 유형의 파일을 검사하도록 구성되어 있습니다.

다음 표에서는 Vscan 서비스를 설정하기 위해 수행할 작업에 대해 설명합니다.

작업	설명	수행 방법
검사 엔진을 설치합니다.	표 4–1에 나열된 지원되는 타사 제품 중 하나 이상을 설치 및 구성합니다.	제품 설명서를 참조하십시오.
파일 시스템이 바이러스 검사를 허용하도록 합니다.	ZFS 파일 시스템에서 바이러스 검사를 사용으로 설정합니다. 기본적으로 검사는 사용 안함으로 설정되어 있습니다.	73 페이지 "파일 시스템에서 바이러스 검사를 사용으로 설정하는 방법"
Vscan 서비스를 사용으로 설정합니다.	검사 서비스를 시작합니다.	74 페이지 "Vscan 서비스를 사용으로 설정하는 방법"
Vscan 서비스에 검사 엔진을 추가합니다.	Vscan 서비스에 특정 검사 엔진을 포함합니다.	74 페이지 "검사 엔진 추가 방법"
Vscan 서비스를 구성합니다.	Vscan 등록 정보를 확인 및 변경합니다.	74 페이지 "Vscan 등록 정보 확인 방법" 75 페이지 "Vscan 등록 정보 변경 방법"
특정 파일 유형에 대한 Vscan 서비스를 구성합니다.	검사에 포함 및 제외할 파일 유형을 지정합니다.	75 페이지 "바이러스 검사에서 파일을 제외하는 방법"

▼ 파일 시스템에서 바이러스 검사를 사용으로 설정하는 방법

파일 시스템 명령을 사용하여 파일의 바이러스 검사를 허용할 수 있습니다. 예를 들어, 바이러스 검사에 ZFS 파일 시스템을 포함하려면 `zfs`(1M) 명령을 사용합니다.

시작하기 전에 ZFS 파일 시스템 관리 또는 ZFS 저장소 관리 권한 프로파일에 지정되어 있어야 합니다. ZFS 파일 시스템에서는 특정 사용자에게 일부 관리 작업을 위임할 수 있습니다. 위임된 관리에 대한 자세한 내용은 **Oracle Solaris 관리: ZFS 파일 시스템**의 9 장, "Oracle Solaris ZFS 위임 관리"를 참조하십시오.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.
2. **ZFS 파일 시스템(예: `pool/volumes/vol1`)에서 바이러스 검사를 사용으로 설정합니다.**
   ```
   # zfs set vscan=on path/pool/volumes/vol1
   ```

▼ Vscan 서비스를 사용으로 설정하는 방법

시작하기 전에 VSCAN 관리 권한 프로파일에 지정되어 있어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.
2. **`svcadm(1M)` 명령을 사용하여 바이러스 검사를 사용으로 설정합니다.**
   ```
   # svcadm enable vscan
   ```

▼ 검사 엔진 추가 방법

시작하기 전에 VSCAN 관리 권한 프로파일에 지정되어 있어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.
2. **기본 등록 정보로 Vscan 서비스에 검사 엔진을 추가하려면 다음을 입력합니다.**
   ```
   #vscanadm add-engine engine_ID
   ```
   명령에 대한 설명은 `vscanadm(1M)` 명령 매뉴얼 페이지를 참조하십시오.

▼ Vscan 등록 정보 확인 방법

시작하기 전에 VSCAN 관리 권한 프로파일에 지정되어 있어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.
2. **모든 검사 엔진 또는 특정 검사 엔진의 Vscan 서비스에 대한 등록 정보를 확인합니다.**
   - **특정 검사 엔진의 등록 정보를 확인하려면 다음을 입력합니다.**
     ```
     # vscanadm get-engine engineID
     ```
   - **모든 검사 엔진의 등록 정보를 확인하려면 다음을 입력합니다.**
     ```
     # vscanadm get-engine
     ```

- **Vscan 서비스의 등록 정보 중 하나를 확인하려면 다음을 입력합니다.**

  ```
  # vscanadm get -p property
  ```

  여기서 *property*는 vscanadm(1M) 명령 매뉴얼 페이지에 설명된 매개변수 중 하나입니다.

  예를 들어, 검사할 수 있는 파일의 최대 크기를 확인하려면 다음을 입력합니다.

  ```
  # vscanadm get max-size
  ```

▼ Vscan 등록 정보 변경 방법

특정 검사 엔진의 등록 정보 및 Vscan 서비스의 일반 등록 정보를 변경할 수 있습니다. 여러 검사 엔진에서 검사할 파일의 크기를 제한하므로 Vscan 서비스의 *max-size* 등록 정보는 검사 엔진에서 허용하는 최대 크기보다 작거나 같은 값으로 설정해야 합니다. 그런 다음 최대 크기보다 커서 검사되지 않는 파일에 액세스할 수 있는지 여부를 정의합니다.

시작하기 전에 VSCAN 관리 권한 프로파일에 지정되어 있어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**

   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2. **vscanadm show 명령을 사용하여 현재 등록 정보를 확인합니다.**

3. **최대 바이러스 검사 크기를 128MB 등으로 설정합니다.**

   ```
   # vscanadm set -p max-size=128M
   ```

4. **크기로 인해 검사되지 않는 파일에 대한 액세스가 거부되도록 지정합니다.**

   ```
   # vscanadm set -p max-size-action=deny
   ```

   명령에 대한 설명은 vscanadm(1M) 명령 매뉴얼 페이지를 참조하십시오.

▼ 바이러스 검사에서 파일을 제외하는 방법

바이러스 방지 보호를 사용으로 설정하면 특정 유형의 모든 파일이 바이러스 검사에서 제외되도록 지정할 수 있습니다. Vscan 서비스는 시스템 성능에 영향을 끼치므로 특정 파일 유형이 바이러스 검사 대상으로 지정되도록 하여 시스템 리소스를 절약할 수 있습니다.

시작하기 전에 VSCAN 관리 권한 프로파일에 지정되어 있어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**

   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **바이러스 검사에 포함된 모든 파일 유형 목록을 확인합니다.**

```
# vscanadm get -p types
```

3 **다음과 같이 바이러스가 검사될 파일의 유형을 지정합니다.**

- **JPEG 유형 등의 특정 파일 유형을 바이러스 검사에서 제외합니다.**

  ```
  # vscanadm set -p types=-jpg,+*
  ```

- **실행 파일 등의 특정 파일 유형을 바이러스 검사에 포함합니다.**

  ```
  # vscanadm set -p types=+exe,-*
  ```

자세한 내용은 vscanadm(1M) 매뉴얼 페이지를 참조하십시오.

◆ ◆ ◆ 5 장

5 장치에 대한 액세스 제어(작업)

이 장에서는 장치 보호를 위한 단계별 지침을 참조 절과 함께 제공합니다.

다음은 이 장에 포함된 정보 목록입니다.

장치 보호에 대한 개요 정보는 43 페이지 "장치에 대한 액세스 제어"를 참조하십시오.

장치 구성(작업 맵)

다음 작업 맵은 장치에 대한 액세스 관리를 위해 수행할 작업을 가리킵니다.

작업	수행 방법
장치 정책을 관리합니다.	78 페이지 "장치 정책 구성(작업 맵)"
장치 할당을 관리합니다.	81 페이지 "장치 할당 관리(작업 맵)"
장치 할당을 사용합니다.	86 페이지 "장치 할당(작업)"

장치 정책 구성(작업)

장치 정책은 시스템에 필수적인 장치에 대한 액세스를 제한하거나 금지합니다. 정책은 커널에서 시행됩니다.

장치 정책 구성(작업 맵)

다음 작업 맵은 장치 정책에 관련된 장치 구성 절차를 가리킵니다.

작업	설명	수행 방법
시스템의 장치에 대한 장치 정책을 봅니다.	장치 및 해당 장치 정책을 나열합니다.	78 페이지 "장치 정책을 보는 방법"
장치 사용을 위한 권한을 요구합니다.	권한을 사용하여 장치를 보호합니다.	79 페이지 "기존 장치의 장치 정책을 변경하는 방법"
장치에서 권한 요구 사항을 제거합니다.	장치에 액세스하는 데 필요한 권한을 제거하거나 완화합니다.	예 5–3
장치 정책의 변경 사항을 감사합니다.	장치 정책의 변경 사항을 감사 증적에 기록합니다.	80 페이지 "장치 정책의 변경 사항을 감사하는 방법"
`/dev/arp`에 액세스합니다.	Oracle Solaris IP MIB-II 정보를 얻습니다.	80 페이지 "`/dev/*` 장치에서 IP MIB-II 정보를 검색하는 방법"

▼ 장치 정책을 보는 방법

- **시스템의 모든 장치에 대한 장치 정책을 표시합니다.**

```
% getdevpolicy | more
DEFAULT
read_priv_set=none
write_priv_set=none
ip:*
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
...
```

예 5–1 특정 장치에 대한 장치 정책 보기

이 예에서 세 장치에 대한 장치 정책이 표시됩니다.

```
% getdevpolicy /dev/allkmem /dev/ipsecesp /dev/bge
/dev/allkmem
read_priv_set=all
write_priv_set=all
```

```
/dev/ipsecesp
read_priv_set=sys_net_config
write_priv_set=sys_net_config
/dev/bge
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
```

▼ 기존 장치의 장치 정책을 변경하는 방법

시작하기 전에 Device Security 권한 프로파일이 지정되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 장치에 정책을 추가합니다.

```
# update_drv -a -p policy device-driver
```

`-a` *device-driver*에 대한 *policy*를 지정합니다.

`-p` *policy* *device-driver*에 대한 장치 정책입니다. 장치 정책은 두 개의 권한 세트를 지정합니다. 한 세트는 장치 읽기에 필요합니다. 다른 세트는 장치 쓰기에 필요합니다.

device-driver 장치 드라이버입니다.

자세한 내용은 `update_drv(1M)` 매뉴얼 페이지를 참조하십시오.

예 5–2 기존 장치에 정책 추가

다음 예에서 `ipnat` 장치에 장치 정책이 추가됩니다.

```
# getdevpolicy /dev/ipnat
/dev/ipnat
read_priv_set=none
write_priv_set=none
# update_drv -a \
-p 'read_priv_set=net_rawaccess write_priv_set=net_rawaccess' ipnat
# getdevpolicy /dev/ipnat
/dev/ipnat
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
```

예 5–3 장치에서 정책 제거

다음 예에서 `ipnat` 장치에 대한 장치 정책에서 읽기 권한 세트가 제거됩니다.

```
# getdevpolicy /dev/ipnat
/dev/ipnat
read_priv_set=net_rawaccess
```

```
write_priv_set=net_rawaccess
# update_drv -a -p write_priv_set=net_rawaccess ipnat
# getdevpolicy /dev/ipnat
/dev/ipnat
read_priv_set=none
write_priv_set=net_rawaccess
```

▼ 장치 정책의 변경 사항을 감사하는 방법

기본적으로 as 감사 클래스는 AUE_MODDEVPLCY 감사 이벤트를 포함합니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**

   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2. **AUE_MODDEVPLCY 감사 이벤트를 포함하는 감사 클래스를 미리 선택합니다.**

   ```
   # auditconfig -getflags
   current-flags
   # auditconfig -setflags current-flags,as
   ```

   자세한 지침은 548 페이지 "감사 클래스를 사전 선택하는 방법"을 참조하십시오.

▼ /dev/* 장치에서 IP MIB-II 정보를 검색하는 방법

Oracle Solaris IP MIB-II 정보를 검색하는 응용 프로그램이 /dev/arp(/dev/ip 아님)를 열어야 합니다.

1. **/dev/ip 및 /dev/arp의 장치 정책을 확인합니다.**

   ```
   % getdevpolicy /dev/ip /dev/arp
   /dev/ip
   read_priv_set=net_rawaccess
   write_priv_set=net_rawaccess
   /dev/arp
   read_priv_set=none
   write_priv_set=none
   ```

   /dev/ip를 읽고 쓰려면 net_rawaccess 권한이 필요합니다. /dev/arp에 필요한 권한은 없습니다.

2. **/dev/arp를 열고 tcp 및 udp 모듈을 푸시합니다.**

   필요한 권한은 없습니다. 이 방법은 /dev/ip를 열고 arp, tcp, udp 모듈을 푸시하는 것과 같습니다. 이제 /dev/ip을 열려면 권한이 필요하므로 /dev/arp 메소드를 사용하는 것이 좋습니다.

장치 할당 관리(작업)

장치 할당은 주변 장치에 대한 액세스를 제한하거나 금지합니다. 제한은 사용자 할당 타임에 시행됩니다. 기본적으로 사용자는 할당 가능한 장치에 액세스하기 위한 인증이 있어야 합니다.

장치 할당 관리(작업 맵)

다음 작업 맵은 장치 할당을 사용으로 설정하고 구성하는 절차를 가리킵니다. 장치 할당은 기본적으로 사용으로 설정되지 않습니다. 장치 할당을 사용으로 설정한 후 장치 할당 지침은 86 페이지 "장치 할당(작업)"을 참조하십시오.

작업	설명	수행 방법
장치를 할당 가능하도록 만듭니다. 장치 할당을 사용 안함으로 설정합니다.	한번에 하나의 사용자에 장치가 할당되도록 합니다. 모든 장치에서 할당 제한을 제거합니다.	81 페이지 "장치 할당을 사용으로 설정하는 방법"
장치를 할당할 수 있는 인증을 사용자에게 부여합니다.	사용자에게 장치 할당 인증을 지정합니다.	82 페이지 "장치를 할당할 수 있도록 사용자를 인증하는 방법"
시스템의 할당 가능한 장치를 봅니다.	할당 가능한 장치와 장치의 상태를 나열합니다.	83 페이지 "장치에 대한 할당 정보를 보는 방법"
강제로 장치를 할당합니다.	즉시 필요한 사용자에게 장치를 할당합니다.	83 페이지 "강제로 장치 할당"
강제로 장치를 할당 해제합니다.	현재 사용자에게 할당된 장치를 해제합니다.	84 페이지 "강제로 장치 할당 해제"
장치의 할당 등록 정보를 변경합니다.	장치 할당을 위한 요구 사항을 변경합니다.	84 페이지 "할당 가능한 장치를 변경하는 방법"
device-clean 스크립트를 만듭니다.	물리적 장치에서 데이터를 비웁니다.	96 페이지 "새 Device-Clean 스크립트 작성"
장치 할당을 감사합니다.	장치 할당을 감사 증적에 기록합니다.	85 페이지 "장치 할당을 감사하는 방법"

▼ 장치 할당을 사용으로 설정하는 방법

시작하기 전에 Device Security 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**
자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **장치 할당 서비스를 사용으로 설정하고 서비스가 사용으로 설정되었는지 확인합니다.**

```
# svcadm enable svc:/system/device/allocate
# svcs -x allocate
svc:/system/device/allocate:default (device allocation)
 State: online since September 10, 2011 01:10:11 PM PDT
   See: allocate(1)
   See: deallocate(1)
   See: list_devices(1)
   See: device_allocate(1M)
   See: mkdevalloc(1M)
   See: mkdevmaps(1M)
   See: dminfo(1M)
   See: device_maps(4)
   See: /var/svc/log/system-device-allocate:default.log
Impact: None.
```

장치 할당 서비스를 사용 안함으로 설정하려면 `disable` 하위 명령을 사용합니다.

```
# svcadm disable device/allocate
```

▼ 장치를 할당할 수 있도록 사용자를 인증하는 방법

시작하기 전에 User Security 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **적절한 인증 및 명령을 포함하는 권한 프로파일을 만듭니다.**

일반적으로, `solaris.device.allocate` 인증을 포함하는 권한 프로파일을 만듭니다. 170 페이지 "감사 프로파일을 만들거나 변경하는 방법"의 지침을 따릅니다. 다음과 같은 적절한 등록 정보를 권한 프로파일에 제공합니다.

- 권한 프로파일 이름: `Device Allocation`
- 부여된 인증: `solaris.device.allocate`
- 보안 속성 포함 명령: `exec_attr` 데이터베이스에서 `sys_mount` 권한으로 `mount`를 실행하고, `sys_mount` 권한으로 `umount`를 실행합니다.

3 **권한 프로파일에 대한 역할을 만듭니다.**

165 페이지 "역할을 만드는 방법"의 지침을 따릅니다. 다음 역할 등록 정보를 길잡이로 사용합니다.

- 역할 이름: `devicealloc`
- 역할 전체 이름: `Device Allocator`
- 역할 설명: `Allocates and mounts allocated devices`
- 권한 프로파일: `Device Allocation`

  이 권한 프로파일은 역할에 포함된 프로파일 목록의 첫번째여야 합니다.

4 **장치 할당이 허가된 모든 사용자에게 역할을 지정합니다.**

5 **장치 할당 사용 방법을 사용자에게 알려줍니다.**
이동식 매체 할당의 예는 86 페이지 "장치를 할당하는 방법"을 참조하십시오.

▼ 장치에 대한 할당 정보를 보는 방법

시작하기 전에 81 페이지 "장치 할당을 사용으로 설정하는 방법"을 완료했습니다.

Device Security 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**
자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **시스템에 할당 가능한 장치에 대한 정보를 표시합니다.**

```
# list_devices device-name
```

여기서 *device-name*은 다음 중 하나입니다.

- `audio`[*n*] – 마이크로폰 및 스피커입니다.
- `fd`[*n*] – 디스켓 드라이브입니다.
- `rmdisk`[*n*] – 이동식 매체 장치입니다.
- `sr`[*n*] – CD-ROM 드라이브입니다.
- `st`[*n*] – 테이프 드라이브입니다.

일반 오류 `list_devices` 명령이 다음과 비슷한 오류 메시지를 반환하는 경우 장치 할당이 사용으로 설정되지 않았거나 정보를 검색할 권한이 부족한 것입니다.

```
list_devices: No device maps file entry for specified device.
```

명령을 성공하려면 장치 할당을 사용으로 설정하고 `solaris.device.revoke` 인증을 가진 역할을 맡습니다.

▼ 강제로 장치 할당

누군가 장치 할당 해제를 잊어버렸을 때 강제 할당이 사용됩니다. 사용자가 장치를 즉시 필요로 할 때에도 강제 할당을 사용할 수 있습니다.

시작하기 전에 `solaris.device.revoke` 인증이 지정되어야 합니다.

1 **내 역할에 적절한 인증이 있는지 확인합니다.**

```
$ auths
solaris.device.allocate solaris.device.revoke
```

2 **장치가 필요한 사용자에게 강제로 장치를 할당합니다.**

이 예에서 테이프 드라이브가 사용자 jdoe에 강제로 할당됩니다.

```
$ allocate -U jdoe
```

▼ 강제로 장치 할당 해제

프로세스를 종료하거나 사용자가 로그아웃할 때 사용자에 할당된 장치가 자동으로 할당 해제되지 않습니다. 사용자가 장치 할당 해제를 잊어버렸을 때 강제 할당 해제가 사용됩니다.

시작하기 전에 solaris.device.revoke 인증이 지정되어야 합니다.

1 **내 역할에 적절한 인증이 있는지 확인합니다.**

```
$ auths
solaris.device.allocate solaris.device.revoke
```

2 **강제로 장치 할당을 해제합니다.**

이 예에서 프린터가 강제로 할당 해제됩니다. 이제 다른 사용자가 프린터를 할당할 수 있습니다.

```
$ deallocate -f /dev/lp/printer-1
```

▼ 할당 가능한 장치를 변경하는 방법

시작하기 전에 이 절차가 성공하려면 장치 할당을 사용으로 설정해야 합니다. 장치 할당을 사용으로 설정하려면 81 페이지 "장치 할당을 사용으로 설정하는 방법"을 참조하십시오. 사용자는 수퍼유저여야 합니다.

- **인증이 필요한지 지정하거나, solaris.device.allocate 인증을 지정합니다.**

device_allocate 파일에서 장치 항목의 다섯번째 필드를 변경합니다.

```
audio;audio;reserved;reserved;solaris.device.allocate;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;solaris.device.allocate;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;solaris.device.allocate;/etc/security/lib/sr_clean
```

여기서 solaris.device.allocate는 사용자가 장치를 사용하려면 solaris.device.allocate 인증이 필요함을 나타냅니다.

예 5–4 임의 사용자가 장치를 할당하도록 허가

다음 예에서 시스템의 임의 사용자가 임의 장치를 할당할 수 있습니다. device_allocate 파일에서 모든 장치 항목의 다섯번째 필드는 at 기호(@)로 변경되었습니다.

```
# vi /etc/security/device_allocate
audio;audio;reserved;reserved;@;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;@;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;@;/etc/security/lib/sr_clean
...
```

예 5–5 일부 주변 장치의 사용을 금지

다음 예에서 오디오 장치를 사용할 수 없습니다. `device_allocate` 파일에서 오디오 장치 항목의 다섯번째 필드는 별표(*)로 변경되었습니다.

```
# vi /etc/security/device_allocate
audio;audio;reserved;reserved;*;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;solaris device.allocate;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;solaris device.allocate;/etc/security/lib/sr_clean
...
```

예 5–6 모든 주변 장치의 사용을 금지

다음 예에서 어떤 주변 장치도 사용할 수 없습니다. `device_allocate` 파일에서 모든 장치 항목의 다섯번째 필드는 별표(*)로 변경되었습니다.

```
# vi /etc/security/device_allocate
audio;audio;reserved;reserved;*;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;*;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;*;/etc/security/lib/sr_clean
...
```

▼ 장치 할당을 감사하는 방법

기본적으로 장치 할당 명령은 `other` 감사 클래스에 속합니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2. **`ot` 감사 클래스를 미리 선택합니다.**

   ```
   # auditconfig -getflags
   current-flags
   # auditconfig -setflags current-flags,ot
   ```

   자세한 지침은 548 페이지 “감사 클래스를 사전 선택하는 방법”을 참조하십시오.

장치 할당(작업)

장치 할당은 한번에 하나의 사용자에 장치가 사용되도록 예약합니다. 마운트 지점이 필요한 장치를 마운트해야 합니다. 다음 절차는 장치를 할당하는 방법을 보여줍니다.

▼ 장치를 할당하는 방법

시작하기 전에 81 페이지 "장치 할당을 사용으로 설정하는 방법"에 설명된 대로 장치 할당을 사용으로 설정해야 합니다. 인증이 필요한 경우 사용자에 인증이 있어야 합니다.

1 장치를 할당합니다.

장치 이름으로 장치를 지정합니다.

```
% allocate device-name
```

2 장치가 할당되었는지 확인합니다.

동일한 명령을 실행합니다.

```
% allocate device-name
allocate. Device already allocated.
```

예 5–7 마이크로폰 할당

이 예에서 사용자 jdoe가 마이크로폰 audio를 할당합니다.

```
% whoami
jdoe
% allocate audio
```

예 5–8 프린터 할당

이 예에서 사용자가 프린터를 할당합니다. 사용자가 printer-1을 할당 해제하거나 프린터가 다른 사용자에게 강제로 할당되기 전까지는 아무도 프린터에 인쇄할 수 없습니다.

```
% allocate /dev/lp/printer-1
```

강제 할당 해제의 예는 84 페이지 "강제로 장치 할당 해제"를 참조하십시오.

예 5–9 테이프 드라이브 할당

이 예에서 사용자 jdoe가 테이프 드라이브 st0을 할당합니다.

```
% whoami
jdoe
% allocate st0
```

일반 오류 `allocate` 명령이 장치를 할당할 수 없는 경우 콘솔 창에 오류 메시지가 표시됩니다. 할당 오류 메시지 목록은 `allocate(1)` 매뉴얼 페이지를 참조하십시오.

▼ 할당된 장치를 마운트하는 방법

적절한 권한을 부여받은 경우 자동으로 장치가 마운트됩니다. 장치 마운트를 실패한 경우 이 절차를 따릅니다.

시작하기 전에 장치를 할당받았습니다. 장치 마운트에 필요한 권한이 지정되었습니다. 필요한 권한을 부여하려면 82 페이지 "장치를 할당할 수 있도록 사용자를 인증하는 방법"을 참조하십시오.

1 장치를 할당하고 마운트할 수 있는 역할을 맡습니다.

```
% su - role-name
Password:       <Type role-name password>
$
```

2 역할의 홈 디렉토리에 마운트 지점을 만들고 보호합니다.

처음으로 마운트 지점을 사용할 때만 이 단계를 수행해야 합니다.

```
$ mkdir mount-point ; chmod 700 mount-point
```

3 할당 가능한 장치를 나열합니다.

```
$ list_devices -l
List of allocatable devices
```

4 장치를 할당합니다.

장치 이름으로 장치를 지정합니다.

```
$ allocate device-name
```

5 장치를 마운트합니다.

```
$ mount -o ro -F filesystem-type device-path mount-point
```

구문 설명은 다음과 같습니다.

`-o ro`	장치가 읽기 전용으로 마운트됨을 나타냅니다. 장치에 쓰기 가능한 경우를 나타내려면 `-o rw`를 사용하십시오.
`-F` *filesystem-type*	장치의 파일 시스템 포맷을 나타냅니다. 일반적으로, CD-ROM은 HSFS 파일 시스템으로 포맷됩니다. 디스켓은 대개 PCFS 파일 시스템으로 포맷됩니다.
device-path	장치의 경로를 나타냅니다. `list_devices -l` 명령의 출력은 *device-path*를 포함합니다.
mount-point	단계 2에서 만든 마운트 지점을 나타냅니다.

예 5–10 CD-ROM 드라이브 할당

이 예에서 사용자가 CD-ROM 드라이브 `sr0`을 할당하고 마운트할 수 있는 역할을 맡습니다. 드라이브는 HSFS 파일 시스템으로 포맷됩니다.

```
% roles
devicealloc
% su - devicealloc
Password:       <Type devicealloc password>
$ mkdir /home/devicealloc/mymnt
$ chmod 700 /home/devicealloc/mymnt
$ list_devices -l
...
device: sr0 type: sr files: /dev/sr0 /dev/rsr0 /dev/dsk/c0t2d0s0 ...
...
$ allocate sr0
$ mount -o ro -F hsfs /dev/sr0 /home/devicealloc/mymnt
$ cd /home/devicealloc/mymnt ; ls
List of the contents of CD-ROM
```

일반 오류 `mount` 명령이 장치를 마운트할 수 없는 경우 `mount: insufficient privileges` 오류 메시지가 표시됩니다. 다음 사항을 확인합니다.

- 프로파일 셸에서 `mount` 명령을 실행 중인지 확인합니다. 역할을 맡은 경우 역할에 프로파일 셸이 있습니다. `mount` 명령으로 프로파일이 지정된 사용자인 경우 프로파일 셸을 만들어야 합니다. 사용 가능한 프로파일 셸 목록은 `pfexec(1)`을 참조하십시오.
- 지정된 마운트 지점을 소유하는지 확인합니다. 마운트 지점에 대한 읽기, 쓰기, 실행 액세스가 있어야 합니다.

여전히 할당된 장치를 마운트할 수 없는 경우 관리자에게 문의하십시오.

▼ 장치 할당을 해제하는 방법

할당 해제를 수행하면 사용자 작업 완료 시 다른 사용자가 장치를 할당하고 사용할 수 있습니다.

시작하기 전에 장치에 할당되어야 합니다.

1 장치가 마운트된 경우 장치 마운트를 해제합니다.

```
$ cd $HOME
$ umount mount-point
```

2 장치를 할당 해제합니다.

```
$ deallocate device-name
```

예 5–11 마이크로폰 할당 해제

이 예에서 사용자 jdoe가 마이크로폰 audio의 할당을 해제합니다.

```
% whoami
jdoe
% deallocate audio0
```

예 5–12 CD-ROM 드라이브 할당 해제

이 예에서 Device Allocator 역할이 CD-ROM 드라이브 할당을 해제합니다. 메시지를 인쇄한 후에 CD-ROM이 배출됩니다.

```
$ whoami
devicealloc
$ cd /home/devicealloc
$ umount /home/devicealloc/mymnt
$ ls /home/devicealloc/mymnt
$
$ deallocate sr0
/dev/sr0:      326o
/dev/rsr0:     326o
...
sr_clean: Media in sr0 is ready.  Please, label and store safely.
```

장치 보호(참조)

Oracle Solaris의 장치는 장치 정책으로 보호됩니다. 주변 장치를 장치 할당으로 보호할 수 있습니다. 장치 정책은 커널로 시행됩니다. 장치 할당은 선택적으로 사용으로 설정되며, 사용자 레벨에서 시행됩니다.

장치 정책 명령

장치 관리 명령은 로컬 파일의 장치 정책을 운영합니다. 장치 정책은 권한 요구 사항을 포함할 수 있습니다. Device Management 및 Device Security 권한 프로파일이 지정된 사용자는 장치를 관리할 수 있습니다.

다음 표는 장치 관리 명령을 나열합니다.

표 5–1 장치 관리 명령

명령 매뉴얼 페이지	목적
devfsadm(1M)	실행 중인 시스템에서 장치 및 장치 드라이버를 관리합니다. 또한 장치 정책을 로드합니다. devfsadm 명령은 디스크, 테이프, 포트, 오디오 및 의사 장치에 대한 /dev 링크를 정리할 수 있습니다. 이름이 지정된 드라이버의 장치를 재구성할 수도 있습니다.
getdevpolicy(1M)	하나 이상의 장치와 연관된 정책을 표시합니다. 이 명령은 임의 사용자가 실행할 수 있습니다.
add_drv(1M)	실행 중인 시스템에 새 장치 드라이버를 추가합니다. 새 장치에 장치 정책을 추가하는 옵션을 포함합니다. 일반적으로, 이 명령은 장치 드라이버를 설치 중일 때 스크립트로 호출됩니다.
update_drv(1M)	기존 장치 드라이버의 속성을 업데이트합니다. 장치에 대해 장치 정책을 업데이트하는 옵션을 포함합니다. 일반적으로, 이 명령은 장치 드라이버를 설치 중일 때 스크립트로 호출됩니다.
rem_drv(1M)	장치 또는 장치 드라이버를 제거합니다.

장치 할당

장치 할당을 사용하여 데이터 손실, 컴퓨터 바이러스 및 기타 보안 침해로부터 사이트를 보호할 수 있습니다. 장치 정책과 달리, 장치 할당은 선택 사항입니다. 장치 할당은 인증을 사용하여 할당 가능한 장치에 대한 액세스를 제한합니다.

장치 할당의 구성 요소

장치 할당 방식의 구성 요소는 다음과 같습니다.

- svc:/system/device/allocate 서비스. 자세한 내용은 smf(5) 매뉴얼 페이지 및 기타 장치 할당 명령에 대한 매뉴얼 페이지를 참조하십시오.
- allocate, deallocate, dminfo, list_devices 명령. 자세한 내용은 91 페이지 “장치 할당 명령”을 참조하십시오.
- Device Management 및 Device Security 권한 프로파일. 자세한 내용은 91 페이지 “장치 할당 권한 프로파일”을 참조하십시오.
- 각 할당 가능한 장치에 대한 device-clean 스크립트

이러한 명령과 스크립트는 다음 로컬 파일을 사용하여 장치 할당을 구현합니다.

- /etc/security/device_allocate 파일. 자세한 내용은 device_allocate(4) 매뉴얼 페이지를 참조하십시오.
- /etc/security/device_maps 파일. 자세한 내용은 device_maps(4) 매뉴얼 페이지를 참조하십시오.

- 각 할당 가능한 장치에 대한 /etc/security/dev 디렉토리의 잠금 파일
- 각 할당 가능한 장치와 연관된 잠금 파일의 변경된 속성

주 – /etc/security/dev 디렉토리는 차후 릴리스의 Oracle Solaris에서 지원되지 않을 수 있습니다.

장치 할당 서비스

svc:/system/device/allocate 서비스는 장치 할당을 제어합니다. 이 서비스는 기본적으로 꺼져 있습니다. 서비스를 사용으로 설정하려면 svcadm enable svc:/system/device/allocate 명령을 실행합니다.

장치 할당 권한 프로파일

장치 및 장치 할당을 관리하려면 Device Management 및 Device Security 권한 프로파일이 필요합니다.

이러한 권한 프로파일은 다음 인증을 포함합니다.

- solaris.device.allocate – 장치를 할당하는 데 필요합니다.
- solaris.device.cdrw – CD-ROM 읽기 및 쓰기에 필요합니다.
- solaris.device.config – 장치의 속성을 구성하는 데 필요합니다.
- solaris.device.grant – 내게 지정된 장치 인증을 다른 사용자에게 위임하는 데 필요합니다.
- solaris.device.mount.alloptions.fixed – 고정된 장치를 마운트할 때 마운트 옵션을 지정하는 데 필요합니다.
- solaris.device.mount.alloptions.removable – 이동식 장치를 마운트할 때 마운트 옵션을 지정하는 데 필요합니다.
- solaris.device.mount.fixed – 고정된 장치를 마운트하는 데 필요합니다.
- solaris.device.mount.removable – 이동식 장치를 마운트하는 데 필요합니다.
- solaris.device.revoke – 장치를 취소하거나 회수하는 데 필요합니다.

장치 할당 명령

대문자 옵션과 함께 사용되는 allocate, deallocate, list_devices 명령은 관리 명령입니다. 그렇지 않으면 이러한 명령은 사용자 명령입니다. 다음 표는 장치 할당 명령을 나열합니다.

표 5–2 장치 할당 명령

명령 매뉴얼 페이지	목적
dminfo(1M)	할당 가능한 장치를 장치 유형별, 장치 이름별, 전체 경로 이름별로 검색합니다.
list_devices(1)	할당 가능한 장치의 상태를 나열합니다. device_maps 파일에 나열된 장치와 연관된 장치 특수 파일을 모두 나열합니다. -U 옵션을 사용하여 지정된 사용자 ID에 할당 가능한 또는 할당된 장치를 나열합니다. 이 옵션으로 다른 사용자에게 할당 가능한 또는 할당된 장치를 확인할 수 있습니다. solaris.device.revoke 인증이 있어야 합니다.
allocate(1)	한 사용자가 사용하도록 할당 가능한 장치를 예약합니다. 기본적으로 사용자가 장치를 할당하려면 solaris.device.allocate 인증이 있어야 합니다. 사용자 인증이 필요하지 않도록 device_allocate 파일을 수정해야 합니다. 그러면 시스템의 사용자가 사용할 장치가 할당되도록 요청할 수 있습니다.
deallocate(1)	장치에서 할당 예약을 제거합니다.

할당 명령에 대한 인증

기본적으로 사용자가 할당 가능한 장치를 예약하려면 solaris.device.allocate 인증이 있어야 합니다. solaris.device.allocate 인증이 포함되도록 권한 프로파일을 만들려면 82 페이지 “장치를 할당할 수 있도록 사용자를 인증하는 방법”을 참조하십시오.

관리자가 장치의 할당 상태를 변경하려면 solaris.device.revoke 인증이 있어야 합니다. 예를 들어, allocate 및 list_devices 명령에 -U 옵션을 사용하거나 deallocate 명령에 -F 옵션을 사용하려면 solaris.device.revoke 인증이 필요합니다.

자세한 내용은 205 페이지 “인증이 필요한 선택된 명령”을 참조하십시오.

할당 오류 상태

deallocate 명령이 할당 해제를 실패하거나 allocate 명령이 할당을 실패할 때 장치가 **할당 오류 상태**에 놓입니다. 할당 가능한 장치가 할당 오류 상태에 있을 때 장치를 강제로 할당 해제해야 합니다. Device Management 권한 프로파일이나 Device Security 권한 프로파일을 가진 사용자/역할만 할당 오류 상태를 처리할 수 있습니다.

deallocate 명령을 -F 옵션과 함께 사용하면 강제로 할당 해제됩니다. 또는 allocate -U를 사용하여 장치를 사용자에 지정할 수 있습니다. 일단 장치를 할당하면 오류 메시지가 나타날 경우 조사할 수 있습니다. 장치 관련 문제를 수정한 후에 강제로 할당 해제할 수 있습니다.

device_maps 파일

장치 할당을 설정할 때 장치 맵이 만들어집니다. /etc/security/device_maps 파일은 각 할당 가능한 장치와 연관된 장치 이름, 장치 유형, 장치 특수 파일을 포함합니다.

device_maps 파일은 각 장치에 대한 장치 특수 파일 매핑을 정의하며, 이는 대부분의 경우 직관적이 아닙니다. 이 파일을 사용하여 프로그램은 어떤 장치 특수 파일이 어떤 장치에 매핑되는지 찾을 수 있습니다. 예를 들어, dminfo 명령을 사용하여 할당 가능한 장치를 설정할 때 지정된 장치 이름, 장치 유형, 장치 특수 파일을 검색할 수 있습니다. dminfo 명령은 device_maps 파일을 사용하여 이 정보를 보고합니다.

각 장치는 다음과 같은 형식의 한 라인으로 입력됩니다.

device-name:*device-type*:*device-list*

예 5–13 샘플 device_maps 입력

다음은 디스켓 드라이브 fd0에 대한 device_maps 파일의 입력 예입니다.

```
fd0:\
fd:\
/dev/diskette /dev/rdiskette /dev/fd0a /dev/rfd0a \
/dev/fd0b /dev/rfd0b /dev/fd0c /dev/fd0 /dev/rfd0c /dev/rfd0:\
```

device_maps 파일에서 다음 행에 입력을 계속하려면 백슬래시(\)로 끝날 수 있습니다. 주석을 포함할 수도 있습니다. 파운드 기호(#)는 백슬래시 바로 앞에 오지 않는 다음 개행 전까지 모든 후속 텍스트를 주석 처리합니다. 선행 및 후행 공백이 필드에 허용됩니다. 필드 정의는 다음과 같습니다.

device-name 장치의 이름을 지정합니다. 현재 장치 이름 목록은 83 페이지 "장치에 대한 할당 정보를 보는 방법"을 참조하십시오.

device-type 일반 장치 유형을 지정합니다. 일반 이름은 st, fd, rmdisk, audio와 같은 장치 클래스의 이름입니다. *device-type* 필드는 관련 장치를 논리적으로 그룹화합니다.

device-list 물리적 장치와 연관된 장치 특수 파일을 나열합니다. *device-list*는 특정 장치에 액세스가 허용된 특수 파일을 모두 포함해야 합니다. 목록이 불완전하면 악의적인 사용자가 개인 정보를 계속 얻거나 수정할 수 있습니다. *device-list* 필드의 유효한 항목은 /dev 디렉토리에 위치한 장치 파일을 반영합니다.

device_allocate 파일

/etc/security/device_allocate 파일을 수정하여 장치를 할당 가능에서 할당 불가능으로 변경하거나 새 장치를 추가할 수 있습니다. 샘플 device_allocate 파일은 다음과 같습니다.

```
st0;st;;;;/etc/security/lib/st_clean
fd0;fd;;;;/etc/security/lib/fd_clean
sr0;sr;;;;/etc/security/lib/sr_clean
audio;audio;;;*;/etc/security/lib/audio_clean
```

device_allocate 파일의 항목에 장치가 할당 가능함을 특별히 언급하지 않는 한, 장치가 할당 가능함을 의미하지는 않습니다. 샘플 device_allocate 파일에서 오디오 장치 항목의 다섯번째 필드에는 별표(*)가 있습니다. 다섯번째 필드의 별표는 장치가 할당 불가능함을 시스템에 알려줍니다. 따라서 장치를 사용할 수 없습니다. 이 필드에 다른 값을 입력하거나 값이 없으면 장치를 사용할 수 있음을 나타냅니다.

device_allocate 파일에서 각 장치는 다음과 같은 형식의 한 라인으로 입력됩니다.

device-name;*device-type*;reserved;reserved;*auths*;*device-exec*

device_allocate 파일에서 다음 행에 입력을 계속하려면 백슬래시(\)로 끝날 수 있습니다. 주석을 포함할 수도 있습니다. 파운드 기호(#)는 백슬래시 바로 앞에 오지 않는 다음 개행 전까지 모든 후속 텍스트를 주석 처리합니다. 선행 및 후행 공백이 필드에 허용됩니다. 필드 정의는 다음과 같습니다.

device-name 장치의 이름을 지정합니다. 현재 장치 이름 목록은 83 페이지 "장치에 대한 할당 정보를 보는 방법"을 참조하십시오.

device-type 일반 장치 유형을 지정합니다. 일반 이름은 st, fd, sr과 같은 장치 클래스의 이름입니다. *device-type* 필드는 관련 장치를 논리적으로 그룹화합니다. 장치를 할당 가능하도록 만들 때 device_maps 파일의 *device-type* 필드에서 장치 이름을 검색합니다.

reserved Sun은 나중에 사용하도록 reserved로 표시된 두 필드를 예약합니다.

auths 장치가 할당 가능한지 여부를 지정합니다. 이 필드의 별표(*)는 장치가 할당 불가능함을 나타냅니다. 인증 문자열 또는 빈 필드는 장치가 할당 가능함을 나타냅니다. 예를 들어, *auths* 필드의 문자열 solaris.device.allocate 는 장치를 할당하려면 solaris.device.allocate 인증이 필요함을 나타냅니다. 이 필드의 at 기호(@)는 임의 사용자에 의해 장치가 할당 가능함을 나타냅니다.

device-exec 할당 프로세스 동안 정리 및 객체 재사용 보호와 같은 특수 처리를 위해 호출할 스크립트의 경로 이름을 제공합니다. *device-exec* 스크립트는 장치가 deallocate 명령으로 작동할 때 언제든지 실행됩니다.

예를 들어, 다음과 같은 sr0 장치의 입력은 solaris.device.allocate 인증을 가진 사용자에 의해 CD-ROM 드라이브가 할당 가능함을 나타냅니다.

```
sr0;sr;reserved;reserved;solaris.device.allocate;/etc/security/lib/sr_clean
```

기본 장치 및 정의된 특성을 받아들이기로 결정할 수 있습니다. 새 장치를 설치한 후에 항목을 수정할 수 있습니다. 사용 전에 할당되어야 할 장치는 해당 장치 시스템의 device_allocate 및 device_maps 파일에 정의해야 합니다. 현재 카트리지 테이프

드라이브, 디스켓 드라이브, CD-ROM 드라이브, 이동식 매체 드라이브 및 오디오 칩이 할당 가능한 것으로 고려됩니다. 이러한 장치 유형에는 device-clean 스크립트가 있습니다.

주 – Xylogics 테이프 드라이브나 Archive 테이프 드라이브는 SCSI 장치에 제공된 `st_clean` 스크립트를 사용합니다. 터미널, 그래픽 타블렛 및 기타 할당 가능한 장치에 대한 고유의 device-clean 스크립트를 만들어야 합니다. 스크립트는 해당 유형의 장치에 대한 객체 재사용 요구 사항을 충족해야 합니다.

Device-Clean 스크립트

장치 할당은 이른바 객체 재사용 요구 사항의 일부를 충족합니다. *device-clean* 스크립트는 재사용 전에 물리적 장치에서 모든 사용 가능한 데이터를 비우도록 보안 요구 사항을 처리합니다. 다른 사용자에 의해 장치가 할당 가능하기 전에 데이터를 지웁니다. 기본적으로 카트리지 테이프 드라이브, 디스켓 드라이브, CD-ROM 드라이브 및 오디오 장치는 device-clean 스크립트가 필요합니다. Oracle Solaris가 스크립트를 제공합니다. 이 절은 device-clean 스크립트가 하는 일을 설명합니다.

테이프용 Device-Clean 스크립트

`st_clean` device-clean 스크립트는 세 가지 테이프 장치를 지원합니다.

- SCSI ¼인치 테이프
- Archive ¼인치 테이프
- Open-reel ½인치 테이프

`st_clean` 스크립트는 `mt` 명령에 `rewoffl` 옵션을 사용하여 장치를 정리합니다. 자세한 내용은 `mt(1)` 매뉴얼 페이지를 참조하십시오. 스크립트가 시스템 부트 중 실행되면 장치를 질의하여 장치가 온라인인지 확인합니다. 장치가 온라인이면 장치 안에 매체가 있는지 확인합니다. ¼인치 테이프 장치 안에 매체가 있으면 할당 오류 상태에 놓입니다. 할당 오류 상태에 놓이면 관리자가 수동으로 장치를 정리해야 합니다.

정상적인 시스템 운영 중에 `deallocate` 명령을 대화식 모드로 실행할 때 매체를 제거하라는 메시지가 나타납니다. 매체를 장치에서 제거할 때까지 할당 해제가 지연됩니다.

디스켓 및 CD-ROM 드라이브용 Device-Clean 스크립트

다음 device-clean 스크립트가 디스켓 및 CD-ROM 드라이브용으로 제공됩니다.

- **`fd_clean` 스크립트 –** 디스켓용 device-clean 스크립트입니다.
- **`sr_clean` 스크립트 –** CD-ROM 드라이브용 device-clean 스크립트입니다.

스크립트가 eject 명령을 사용하여 드라이브에서 매체를 제거합니다. eject 명령을 실패하면 장치가 할당 오류 상태에 놓입니다. 자세한 내용은 eject(1) 매뉴얼 페이지를 참조하십시오.

오디오용 Device-Clean 스크립트

오디오 장치는 audio_clean 스크립트로 정리합니다. 스크립트가 AUDIO_GETINFO ioctl 시스템 호출을 수행하여 장치를 읽습니다. 그런 다음 AUDIO_SETINFO ioctl 시스템 호출을 수행하여 장치 구성을 기본값으로 재설정합니다.

새 Device-Clean 스크립트 작성

시스템에 할당 가능한 장치를 더 추가하면 고유의 device-clean 스크립트를 만들어야 합니다. deallocate 명령은 device-clean 스크립트에 매개변수를 전달합니다. 여기에 표시된 매개변수는 장치 이름을 포함하는 문자열입니다. 자세한 내용은 device_allocate(4) 매뉴얼 페이지를 참조하십시오.

clean-script `-[I|i|f|S]` *device-name*

device-clean 스크립트는 성공은 "0", 실패는 "0"보다 큰 값을 반환해야 합니다. -I, -f, -S 옵션이 스크립트의 실행 모드를 결정합니다.

`-I` 시스템 부트 중에만 필요합니다. 모든 출력은 시스템 콘솔로 이동해야 합니다. 매체 강제 배출을 실패하거나 불가능한 경우 장치가 할당 오류 상태에 놓여야 합니다.

`-i` -I 옵션과 비슷하지만, 단 출력이 표시되지 않습니다.

`-f` 강제 정리용입니다. 옵션이 대화식이고 사용자가 프롬프트에 응답할 수 있다고 가정합니다. 이 옵션을 사용하는 스크립트는 정리의 한 부분을 실패할 경우 정리를 완료하려고 시도해야 합니다.

`-S` 표준 정리용입니다. 옵션이 대화식이고 사용자가 프롬프트에 응답할 수 있다고 가정합니다.

◆ ◆ ◆ 6 장

6 기본 감사 보고 도구 사용(작업)

이 장에서는 시스템에서 파일의 매니페스트를 만드는 방법과 매니페스트를 사용하여 시스템의 무결성을 검사하는 방법에 대해 설명합니다. 기본 감사 보고 도구(BART)를 사용하여 시간에 따른 시스템의 파일 레벨 검사를 수행함으로써 시스템을 종합적으로 검증할 수 있습니다.

다음은 이 장에 포함된 정보 목록입니다.

기본 감사 보고 도구(개요)

BART는 파일 시스템 레벨에서 전적으로 작동하는 파일 추적 도구입니다. BART를 사용하면 배치된 시스템에 설치된 소프트웨어 스택의 구성 요소에 대한 정보를 빠르고 쉽고 안정적으로 수집할 수 있습니다. BART를 사용하면 시간 소모가 많은 관리 작업을 단순화함으로써 시스템의 네트워크 관리 비용을 크게 줄일 수 있습니다.

BART를 사용하면 알려진 기준과 비교하여 시스템에서 발생한 파일 레벨 변경 사항을 확인할 수 있습니다. BART를 사용하여 완전하게 설치 및 구성된 시스템에서 기준 또는 **제어** 매니페스트를 만들 수 있습니다. 그러면 나중에 시스템의 스냅샷을 이 기준과 비교하여 설치된 이후 시스템에서 발생한 파일 레벨 변경 사항을 나열하는 보고서를 생성할 수 있습니다.

`bart` 명령은 표준 UNIX 명령입니다. 나중에 처리를 위해 `bart` 명령의 출력을 파일로 리디렉션할 수 있습니다.

BART 기능

BART는 강력하면서도 유연한 단순 구문에 초점을 맞추어 설계되었습니다. 이 도구를 사용하여 시간에 따른 시스템의 매니페스트를 생성할 수 있습니다. 그런 다음 시스템의 파일을 검증할 필요가 있을 때 이전 매니페스트와 새 매니페스트를 비교하여 보고서를 생성할 수 있습니다. BART를 사용하는 또 하나의 방법은 여러 유사한 시스템의 매니페스트를 생성하고 시스템 대 시스템 비교를 실행하는 것입니다. BART와 기존 감사 도구의 가장 큰 차이점은 BART가 추적되는 정보와 보고되는 정보 측면 모두에서 유연하다는 점입니다.

BART의 추가 이점 및 용도에는 다음이 포함됩니다.

- 파일 레벨에서 Oracle Solaris 소프트웨어를 실행하는 시스템의 카탈로그화를 위한 효율적이고 쉬운 방법을 제공합니다.
- 모니터할 파일을 정의할 수 있고 필요할 때 프로파일을 수정할 수 있습니다. 이 유연성을 통해 로컬 사용자 정의를 모니터하고 소프트웨어를 쉽고 효율적으로 재구성할 수 있습니다.
- 시스템에서 신뢰할 수 있는 소프트웨어를 실행하고 있는지 확인합니다.
- 시간에 따른 시스템의 파일 레벨 변경 사항을 모니터하여 손상되거나 비정상적인 파일을 찾아낼 수 있습니다.
- 시스템 성능 문제를 해결하는 데 도움을 줍니다.

BART 구성 요소

BART에는 두 가지 주요 구성 요소와 한 가지 선택적 구성 요소가 있습니다.

- BART 매니페스트
- BART 보고서
- BART 규칙 파일

BART 매니페스트

`bart create` 명령을 사용하여 특정 시간에 시스템의 파일 레벨 스냅샷을 만듭니다. 출력은 **매니페스트**라는 파일 및 파일 속성의 카탈로그입니다. 매니페스트에는 시스템의 모든 파일 또는 특정 파일에 대한 정보가 나열됩니다. 여기에는 MD5 체크섬과 같이 고유하게 식별하는 정보를 포함할 수 있는 파일의 속성에 대한 정보가 포함됩니다. MD5 체크섬에 대한 자세한 내용은 `md5(3EXT)` 매뉴얼 페이지를 참조하십시오. 매니페스트는 클라이언트와 서버 시스템 간에 저장 및 전송할 수 있습니다.

주 – BART는 동일 유형의 파일 시스템을 제외하고 파일 시스템 경계를 넘지 **않습니다**. 이 제약 조건은 `bart create` 명령의 출력을 더욱 가능하게 만듭니다. 예를 들어, 인수 없는 `bart create` 명령은 모든 ZFS 파일 시스템을 루트(/) 디렉토리에 카탈로그화합니다. 하지만 NFS 또는 TMPFS 파일 시스템이나 마운트된 CD-ROM은 카탈로그화되지 않습니다. 매니페스트를 만들 때 네트워크에 있는 파일 시스템에 대한 감사를 시도하지 마십시오. BART를 사용하여 네트워크 파일 시스템을 모니터하면 별로 가치가 없는 매니페스트를 생성하는 데 많은 리소스가 소모될 수 있습니다.

BART 매니페스트에 대한 자세한 내용은 110 페이지 “BART 매니페스트 파일 형식”을 참조하십시오.

BART 보고서

보고 도구는 비교할 두 개의 매니페스트와 플래그를 지정할 불일치를 나타내는 선택적 사용자 제공 규칙의 세 가지 입력을 사용합니다.

`bart compare` 명령을 사용하여 **제어 매니페스트** 및 **테스트 매니페스트**의 두 매니페스트를 비교합니다. 이러한 매니페스트는 `bart create` 명령과 함께 사용하는 동일한 파일 시스템, 옵션 및 규칙 파일로 준비해야 합니다.

`bart compare` 명령의 출력은 두 매니페스트 간의 파일별 불일치를 나열하는 보고서입니다. **불일치**는 두 매니페스트에 대해 카탈로그화된 해당 파일의 속성 변경 사항입니다. 두 매니페스트 간의 파일 항목 추가나 삭제도 불일치로 간주됩니다.

불일치를 보고할 때 두 가지 제어 레벨이 있습니다.

- 매니페스트를 생성할 때
- 보고서를 생성할 때

매니페스트를 생성하는 것이 두 매니페스트 간의 불일치를 보고하는 것보다 비용이 높으므로 이러한 제어 레벨은 의도적입니다. 매니페스트를 만들었으면 `bart compare` 명령을 서로 다른 규칙 파일과 함께 실행하여 여러 관점에서 매니페스트를 비교할 수 있습니다.

BART 보고서에 대한 자세한 내용은 113 페이지 “BART 보고”를 참조하십시오.

BART 규칙 파일

규칙 파일은 `bart` 명령에 대한 선택적 입력으로 사용할 수 있는 텍스트 파일입니다. 이 파일에서는 포함 및 제외 규칙을 사용합니다. 규칙 파일은 사용자 정의 매니페스트 및 보고서를 만드는 데 사용됩니다. 규칙 파일을 사용하여 카탈로그화할 파일 집합 및 해당 파일 집합에 대해 모니터할 속성을 간결한 구문으로 표현할 수 있습니다. 매니페스트를 비교할 때 규칙 파일은 매니페스트 간의 불일치에 대해 플래그를 지정하는 데 도움이 됩니다. 규칙 파일을 사용하면 시스템의 파일에 대한 특정 정보를 효율적으로 수집할 수 있습니다.

규칙 파일은 텍스트 편집기를 사용하여 만듭니다. 규칙 파일을 사용하여 다음 작업을 수행할 수 있습니다.

- `bart create` 명령을 사용하여 시스템의 모든 파일 또는 특정 파일에 대한 정보를 나열하는 매니페스트를 만듭니다.
- `bart compare` 명령을 사용하여 파일 시스템의 특정 속성을 모니터하는 보고서를 생성합니다.

주 – 서로 다른 목적으로 여러 규칙 파일을 만들 수 있습니다. 하지만 규칙 파일을 사용하여 매니페스트를 만드는 경우 매니페스트를 비교할 때 동일한 규칙 파일을 사용해야 합니다. 규칙 파일로 만들어진 매니페스트를 비교할 때 동일 규칙을 사용하지 않을 경우 `bart compare` 명령의 출력에 잘못된 불일치가 많이 나열됩니다.

또한 규칙 파일에는 사용자 오류로 인해 구문 오류 및 기타 모호한 정보가 포함될 수 있습니다. 규칙 파일에 잘못된 정보가 포함되어 있을 경우 이러한 사용자 오류도 보고됩니다.

규칙 파일을 사용하여 시스템의 특정 파일 및 파일 속성을 모니터하려면 계획이 필요합니다. 규칙 파일을 만들기 전에 모니터할 시스템의 파일 및 파일 속성을 결정합니다. 달성하려는 목적에 따라 매니페스트 만들기, 매니페스트 비교 또는 기타 목적으로 규칙 파일을 사용할 수 있습니다.

BART 규칙 파일에 대한 자세한 내용은 112 페이지 “BART 규칙 파일 형식” 및 `bart_rules(4)` 매뉴얼 페이지를 참조하십시오.

BART 사용(작업)

`bart` 명령을 일반 사용자, 수퍼유저 또는 역할을 맡은 사용자로 실행할 수 있습니다. `bart` 명령을 일반 사용자로 실행할 경우 자신의 홈 디렉토리에 있는 파일과 같이 액세스 권한을 가진 파일만 카탈로그화 및 모니터할 수 있습니다. `bart` 명령을 실행할 때 수퍼유저가 되면 만드는 매니페스트에 모니터하려고 하는 숨겨진 파일 및 개인 파일에 대한 정보가 포함된다는 장점이 있습니다. 권한이 제한된 파일(예: `/etc/passwd` 또는 `/etc/shadow` 파일)에 대한 정보를 카탈로그화 및 모니터해야 하는 경우 수퍼유저로 `bart` 명령을 실행합니다. 역할 기반 액세스 제어 사용에 대한 자세한 내용은 135 페이지 “역할 기반 액세스 제어(개요)”를 참조하십시오.

BART 보안 고려 사항

`bart` 명령을 수퍼유저로 실행하면 누구나 출력을 읽을 수 있게 됩니다. 이 출력에는 비밀로 유지해야 하는 파일 이름이 포함되어 있을 수 있습니다. `bart` 명령을 실행할 때 수퍼유저가 될 경우 출력 보호를 위한 적절한 조치를 취하십시오. 예를 들어, 제한된 권한의 출력 파일을 생성하는 옵션을 사용합니다.

주 – 이 장의 절차 및 예는 수퍼유저가 실행한 `bart` 명령을 보여줍니다. 다르게 지정되지 않은 경우 수퍼유저로 `bart` 명령 실행은 선택 사항입니다.

BART 사용(작업 맵)

작업	설명	수행 방법
BART 매니페스트를 만듭니다.	시스템에 설치된 모든 파일에 대한 정보 목록을 생성합니다.	101 페이지 "매니페스트를 만드는 방법"
사용자 정의 BART 매니페스트를 만듭니다.	시스템에 설치된 특정 파일에 대한 정보 목록을 생성합니다.	103 페이지 "매니페스트를 사용자 정의하는 방법"
BART 매니페스트를 비교합니다.	시간에 따른 시스템 변경 사항을 비교하는 보고서를 생성합니다. 또는 하나 이상의 시스템을 제어 시스템과 비교하는 보고서를 생성합니다.	104 페이지 "시간에 따라 동일 시스템에 대한 매니페스트를 비교하는 방법" 106 페이지 "여러 시스템의 매니페스트를 비교하는 방법"
(옵션) BART 보고서를 사용자 정의합니다.	다음 중 하나의 방법으로 사용자 정의 BART 보고서를 생성합니다. ■ 속성 지정 ■ 규칙 파일 사용	108 페이지 "파일 속성을 지정하여 BART 보고서를 사용자 정의하는 방법" 109 페이지 "규칙 파일을 사용하여 BART 보고서를 사용자 정의하는 방법"

▼ 매니페스트를 만드는 방법

초기 Oracle Solaris 소프트웨어 설치 직후 시스템의 매니페스트를 만들 수 있습니다. 이 유형의 매니페스트는 시간에 따라 동일 시스템에 대한 변경 사항 비교를 위한 기준을 제공합니다. 또는 이 매니페스트를 사용하여 다른 시스템에 대한 매니페스트와 비교할 수 있습니다. 예를 들어, 네트워크에 있는 각 시스템의 스냅샷을 만든 다음 각 테스트 매니페스트를 제어 매니페스트와 비교할 경우 테스트 시스템과 기준 구성을 동기화하기 위해 수행해야 하는 작업을 빠르게 결정할 수 있습니다.

시작하기 전에 시스템 매니페스트를 만들려면 `root` 역할을 가진 사용자여야 합니다.

1 Oracle Solaris 소프트웨어 설치 후 제어 매니페스트를 만들고 출력을 파일로 리디렉션합니다.

```
# bart create options > control-manifest
```

`-R` 매니페스트에 대한 루트 디렉토리를 지정합니다. 규칙에서 지정한 모든 경로는 이 디렉토리에 대한 상대 경로로 해석됩니다. 매니페스트에서 보고하는 모든 경로는 이 디렉토리에 대한 상대 경로입니다.

`-I` 명령줄 또는 표준 입력에서 읽은 카탈로그화할 개별 파일 목록을 사용합니다.

`-r` 이 매니페스트에 대한 규칙 파일 이름입니다. –가 `-r` 옵션과 함께 사용되면 표준 입력에서 규칙 파일을 읽습니다.

`-n` 파일 목록의 모든 일반 파일에 대한 내용 서명을 해제합니다. 이 옵션은 성능을 향상시키는 데 사용할 수 있습니다. 또는 시스템 로그 파일과 같이 파일 목록의 내용이 변경될 것으로 예상되는 경우 이 옵션을 사용할 수 있습니다.

2 매니페스트의 내용을 검토합니다.

3 나중에 사용하기 위해 매니페스트를 저장합니다.

매니페스트에 대한 의미 있는 이름을 선택합니다. 예를 들어, 매니페스트가 만들어진 시스템 이름과 날짜를 사용합니다.

예 6–1 시스템의 모든 파일에 대한 정보를 나열하는 매니페스트 만들기

`bart create` 명령을 옵션 없이 실행할 경우 시스템에 설치된 모든 파일에 대한 정보가 카탈로그화됩니다. 중앙 이미지에서 많은 시스템을 설치할 때 이 유형의 매니페스트를 기준으로 사용합니다. 또는 설치가 동일한지 확인하려는 경우 이 유형의 매니페스트를 사용하여 비교를 실행합니다.

예를 들면 다음과 같습니다.

```
# bart create
! Version 1.1
! HASH SHA256
! Wednesday, September 07, 2011 (22:22:27)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/ D 1024 40755 user::rwx,group::r-x,mask:r-x,other:r-x
3ebc418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090 0 0
.
.
.
/zone D 512 40755 user::rwx group::r-x,mask:r-x,other:r-x 3f81e892
154de3e7bdfd0d57a074c9fae0896a9e2e04bebfe5e872d273b063319e57f334 0 0
.
.
.
```

각 매니페스트는 헤더와 항목으로 구성됩니다. 각 매니페스트 파일 항목은 파일 유형에 따라 단일 행입니다. 예를 들어, 위의 출력에서 각 매니페스트 항목에 대해 F 유형은 파일을 지정하고 D 유형은 디렉토리를 지정합니다. 또한 크기, 내용, 사용자 ID, 그룹 ID 및 권한에 대한 정보가 나열됩니다. 출력의 파일 항목은 특수 문자를 올바르게 처리하기 위해 파일 이름의 인코딩된 버전별로 정렬됩니다. 모든 항목은 파일 이름을 기준으로 오름차순으로 정렬됩니다. 내장된 개행 또는 탭 문자를 포함하는 파일 이름과 같은 모든 비표준 파일 이름은 정렬되기 전에 비표준 문자가 따옴표로 묶입니다.

!로 시작하는 행은 매니페스트에 대한 메타 데이터를 제공합니다. 매니페스트 버전 행은 매니페스트 사양 버전을 나타냅니다. 해시 행은 사용된 해시 방식을 나타냅니다. 날짜 행은 매니페스트가 만들어진 날짜를 날짜 형식으로 표시합니다. date(1) 매뉴얼 페이지를 참조하십시오. 일부 행은 매니페스트 비교 도구에서 무시됩니다. 무시되는 행에는 빈 행, 공백만 구성하는 행 및 #으로 시작하는 주석이 포함됩니다.

▼ 매니페스트를 사용자 정의하는 방법

다음 중 하나의 방법으로 매니페스트를 사용자가 정의할 수 있습니다.

- 하위 트리 지정

  시스템에서 개별 하위 트리에 대한 매니페스트를 만들면 큰 디렉토리의 전체 내용이 아닌 특정 파일에 대한 변경 사항을 효율적으로 모니터할 수 있습니다. 시스템에서 특정 하위 트리의 기준 매니페스트를 만든 다음 정기적으로 동일 하위 트리의 테스트 매니페스트를 만들 수 있습니다. bart compare 명령을 사용하여 제어 매니페스트와 테스트 매니페스트를 비교합니다. 이 옵션을 사용하면 중요 파일 시스템을 효율적으로 모니터하여 파일이 침입자의 공격을 받았는지 여부를 확인할 수 있습니다.

- 파일 이름 지정

  전체 시스템을 카탈로그화하는 매니페스트를 만들 경우 시간이 더 오래 걸리고, 공간을 더 많이 차지하며, 더 많은 비용이 들기 때문에 시스템의 특정 파일에 대한 정보만 나열하고자 할 때 bart 명령의 이 옵션을 사용하도록 선택할 수 있습니다.

- 규칙 파일 사용

  규칙 파일을 사용하여 해당 시스템의 특정 파일 및 특정 하위 트리에 대한 정보를 나열하는 사용자 정의 매니페스트를 만듭니다. 또한 규칙 파일을 사용하여 특정 파일 속성을 모니터할 수 있습니다. 규칙 파일을 사용하여 매니페스트를 만들고 비교하면 하나 이상의 파일이나 하위 트리에 대해 여러 속성을 지정할 수 있는 유연성이 제공됩니다. 반면 명령줄에서는 만들거나 보고하는 각 매니페스트에 대한 모든 파일에 적용되는 전역 속성 정의만 지정할 수 있습니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 카탈로그화 및 모니터할 파일을 결정합니다.

2. **Oracle Solaris 소프트웨어 설치 후 다음 중 하나의 옵션을 사용하여 사용자 정의 매니페스트를 만듭니다.**

   - 하위 트리 지정:

     ```
     # bart create -R root-directory
     ```

   - 파일 이름 지정:

     ```
     # bart create -I filename...
     ```

     예를 들면 다음과 같습니다.

     ```
     # bart create -I /etc/system /etc/passwd /etc/shadow
     ```

   - 규칙 파일 사용:

     ```
     # bart create -r rules-file
     ```

3. **매니페스트의 내용을 검토합니다.**

4. **나중에 사용하기 위해 매니페스트를 저장합니다.**

▼ 시간에 따라 동일 시스템에 대한 매니페스트를 비교하는 방법

시간에 따라 동일 시스템에 대한 파일 레벨 변경 사항을 모니터하려고 할 때 이 절차를 사용합니다. 이 유형의 매니페스트는 손상되거나 비정상적인 파일 찾기, 보안 침입 감지 또는 시스템의 성능 문제 해결에 도움을 줄 수 있습니다.

시작하기 전에 공용 객체가 포함된 매니페스트를 만들고 비교하려면 `root` 역할을 가진 사용자여야 합니다.

1. **Oracle Solaris 소프트웨어 설치 후 시스템에서 모니터할 파일의 제어 매니페스트를 만듭니다.**

   ```
   # bart create -R /etc > control-manifest
   ```

2. **시스템에 대한 변경 사항을 모니터하려고 할 때마다 제어 매니페스트와 동일하게 준비된 테스트 매니페스트를 만듭니다.**

   ```
   # bart create -R /etc > test-manifest
   ```

3. **제어 매니페스트와 테스트 매니페스트를 비교합니다.**

   ```
   # bart compare options control-manifest test-manifest > bart-report
   ```

   `-r` 이 비교에 대한 규칙 파일 이름입니다. `-r` 옵션과 함께 –를 사용하면 지시어가 표준 입력에서 읽게 됩니다.

`-i`	사용자가 명령줄에서 전역 `IGNORE` 지시어를 설정할 수 있습니다.
`-p`	프로그래밍 구문 분석을 위해 표준 비현지화 출력을 생성하는 프로그래밍 모드입니다.
control-manifest	제어 시스템에 대한 `bart create` 명령의 출력입니다.
test-manifest	테스트 시스템에 대한 `bart create` 명령의 출력입니다.

4 BART 보고서에서 이상한 점을 검토합니다.

예 6–2 시간에 따라 동일 시스템에 대한 매니페스트 비교

이 예에서는 두 특정 시점 사이에 `/etc` 디렉토리에서 발생한 변경 사항을 모니터하는 방법을 보여줍니다. 이 유형의 비교를 통해 시스템의 중요 파일이 손상되었는지 여부를 빠르게 확인할 수 있습니다.

- 제어 매니페스트를 만듭니다.

```
# bart create -R /etc > system1.control.090711
! Version 1.1
! HASH SHA256
! Wednesday, September 07, 2011 (11:11:17)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/.cpr_config F 2236 100644 owner@:read_data/write_data/append_data/read_xattr/wr
ite_xattr/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchr
onize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:all
ow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
4e271c59 0 0 3ebc418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090
/.login F 1429 100644 owner@:read_data/write_data/append_data/read_xattr/write_x
attr/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchronize
:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,ev
eryone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
4bf9d6d7 0 3 ff6251a473a53de68ce8b4036d0f569838cff107caf1dd9fd04701c48f09242e
.
.
.
```

- `/etc` 디렉토리에 대한 변경 사항을 모니터하려고 할 때 테스트 매니페스트를 만듭니다.

```
# bart create -R /etc > system1.test.101011
Version 1.1
! HASH SHA256
! Monday, October 10, 2011 (10:10:17)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
```

```
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/.cpr_config F 2236 100644 owner@:read_data/write_data/append_data/read_xattr/wr
ite_xattr/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchr
onize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:all
ow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
4e271c59 0 0 3ebc418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090
.
.
.
```

- 제어 매니페스트와 테스트 매니페스트를 비교합니다.

```
# bart compare system1.control.090711 system1.test.101011
/security/audit_class
mtime  4f272f59
```

위의 출력은 `audit_class` 파일에 대한 수정 시간이 제어 매니페스트가 만들어진 이후 변경되었음을 나타냅니다. 이 보고서를 사용하여 소유권, 날짜, 내용 또는 기타 파일 속성이 변경되었는지 여부를 조사할 수 있습니다. 이 유형의 정보를 쉽게 사용할 수 있으면 누가 파일을 조작했는지 및 언제 변경 사항이 발생했는지 분석할 수 있습니다.

▼ 여러 시스템의 매니페스트를 비교하는 방법

시스템 대 시스템 비교를 실행하여 기준 시스템과 다른 시스템 간에 파일 레벨 차이점이 있는지 여부를 빠르게 확인할 수 있습니다. 예를 들어, 기준 시스템에 특정 버전의 Oracle Solaris 소프트웨어를 설치하고 다른 시스템에 동일한 패키지가 설치되어 있는지 여부를 알고자 하는 경우 이러한 시스템에 대한 매니페스트를 만든 다음 제어 매니페스트와 테스트 매니페스트를 비교할 수 있습니다. 이 유형의 비교는 제어 시스템과 비교하는 각 테스트 시스템에 대한 파일 내용의 불일치를 나열합니다.

시작하기 전에 시스템 매니페스트를 비교하려면 `root` 역할을 가진 사용자여야 합니다.

1 Oracle Solaris 소프트웨어 설치 후 제어 매니페스트를 만듭니다.

```
# bart create options > control-manifest
```

2 제어 매니페스트를 비교합니다.

3 테스트 시스템에서 동일한 `bart` 옵션을 사용하여 매니페스트를 만들고 출력을 파일로 리디렉션합니다.

```
# bart create options > test1-manifest
```

테스트 매니페스트에 대한 고유하고 의미 있는 이름을 선택합니다.

4. **매니페스트를 비교할 준비가 될 때까지 테스트 매니페스트를 시스템의 중앙 위치에 저장합니다.**

5. **매니페스트를 비교하고자 할 때 제어 매니페스트를 테스트 매니페스트의 위치에 복사합니다. 또는 테스트 매니페스트를 제어 시스템에 복사합니다.**

   예를 들면 다음과 같습니다.

   # **cp** *control-manifest /net/test-server/bart/manifests*

   테스트 시스템이 NFS 마운트 시스템이 아닌 경우 FTP 또는 기타 신뢰할 수 있는 방법을 사용하여 제어 매니페스트를 테스트 시스템에 복사합니다.

6. **제어 매니페스트와 테스트 매니페스트를 비교하고 출력을 파일로 리디렉션합니다.**

   # **bart compare** *control-manifest test1-manifest > test1.report*

7. **BART 보고서에서 이상한 점을 검토합니다.**

8. **제어 매니페스트와 비교하고자 하는 각 테스트 매니페스트에 대해 4단계부터 9단계까지 반복합니다.**

   각 테스트 시스템에 대해 동일한 bart 옵션을 사용합니다.

예 6–3 제어 시스템의 매니페스트와 다른 시스템의 매니페스트 비교

이 예에서는 제어 매니페스트와 다른 시스템의 테스트 매니페스트를 비교하여 /usr/bin 디렉토리의 내용에 대한 변경 사항을 모니터하는 방법을 설명합니다.

- 제어 매니페스트를 만듭니다.

```
# bart create -R /usr/bin > control-manifest.090711
! Version 1.1
! HASH SHA256
! Wednesday, September 07, 2011 (11:11:17)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/2to3 F 105 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attribut
es/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:read
_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:re
ad_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4bf9d261 0
 2 154de3e7bdfd0d57a074c9fae0896a9e2e04bebfe5e872d273b063319e57f334
/7z F 509220 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attribu
tes/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:rea
d_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:r
ead_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4dadc48a 0
```

```
 2 3ecd418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090
...
```

- 제어 시스템과 비교하고자 하는 각 시스템에 대한 테스트 매니페스트를 만듭니다.

```
# bart create -R /usr/bin > system2-manifest.101011
! Version 1.1
! HASH SHA256
! Monday, October 10, 2011 (10:10:22)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/2to3 F 105 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attribut
es/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:read
_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:re
ad_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4bf9d261 0
 2 154de3e7bdfd0d57a074c9fae0896a9e2e04bebfe5e872d273b063319e57f334
...
```

- 매니페스트를 비교하고자 할 때 매니페스트를 동일한 위치에 복사합니다.

```
# cp control-manifest /net/system2.central/bart/manifests
```

- 제어 매니페스트와 테스트 매니페스트를 비교합니다.

```
# bart compare control-manifest  system2.test > system2.report
/su:
  gid  control:3  test:1
/ypcat:
  mtime  control:3fd72511  test:3fd9eb23
```

위의 출력은 `/usr/bin` 디렉토리에 있는 `su` 파일의 그룹 ID가 제어 시스템의 그룹 ID와 동일하지 않음을 나타냅니다. 이 정보는 테스트 시스템에 다른 버전의 소프트웨어가 설치되었는지 여부 또는 누군가 파일을 조작했는지 여부를 판단하는 데 도움이 됩니다.

▼ 파일 속성을 지정하여 BART 보고서를 사용자 정의하는 방법

이 절차는 선택 사항이며 명령줄에서 파일 속성을 지정하여 BART 보고서를 사용자 정의하는 방법을 설명합니다. 시스템의 모든 파일 또는 특정 파일에 대한 정보를 나열하는 기준 매니페스트를 만드는 경우 특정 디렉토리, 하위 디렉토리 또는 파일에 대한 변경 사항을 모니터해야 할 때마다 `bart compare` 명령을 실행하여 서로 다른 속성을 지정할 수 있습니다. 명령줄에서 서로 다른 파일 속성을 지정하여 동일 매니페스트에 대해 서로 다른 유형의 비교를 실행할 수 있습니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

1. **모니터하려고 하는 파일 속성을 결정합니다.**
2. **Oracle Solaris 소프트웨어 설치 후 제어 매니페스트를 만듭니다.**
3. **변경 사항을 모니터하려고 할 때 테스트 매니페스트를 만듭니다.**
   제어 매니페스트와 동일하게 테스트 매니페스트를 준비합니다.
4. **매니페스트를 비교합니다.**
   예를 들면 다음과 같습니다.

   ```
   # bart compare -i dirmtime,lnmtime,mtime control-manifest.121503 \
   test-manifest.010504 > bart.report.010504
   ```

   콤마는 명령줄 구문에서 지정하는 각 속성을 구분합니다.
5. **BART 보고서에서 이상한 점을 검토합니다.**

▼ 규칙 파일을 사용하여 BART 보고서를 사용자 정의하는 방법

이 절차도 선택 사항이며 `bart compare` 명령에 규칙 파일을 입력으로 사용하여 BART 보고서를 사용자 정의하는 방법을 설명합니다. 규칙 파일을 사용하면 BART 보고서를 사용자 정의할 수 있으며, 하나 이상의 파일이나 하위 트리에 대해 여러 속성을 지정할 수 있는 유연성을 제공합니다. 서로 다른 규칙 파일을 사용하여 동일한 매니페스트에 대해 서로 다른 비교를 실행할 수 있습니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

1. **모니터하려고 하는 파일 및 파일 속성을 결정합니다.**
2. **텍스트 편집기를 사용하여 적당한 지시어가 있는 규칙 파일을 만듭니다.**
3. **Oracle Solaris 소프트웨어 설치 후 만든 규칙 파일을 사용하여 제어 매니페스트를 만듭니다.**

   ```
   # bart create -r rules-file > control-manifest
   ```
4. **제어 매니페스트와 동일하게 준비된 테스트 매니페스트를 만듭니다.**

   ```
   # bart create -r rules-file > test-manifest
   ```
5. **동일한 규칙 파일을 사용하여 제어 매니페스트와 테스트 매니페스트를 비교합니다.**

   ```
   # bart compare -r rules-file control-manifest test-manifest > bart.report
   ```
6. **BART 보고서에서 이상한 점을 검토합니다.**

예 6–4 규칙 파일을 사용하여 BART 보고서 사용자 정의

다음 규칙 파일에는 bart create 및 bart compare 명령 모두에 대한 지시어가 포함되어 있습니다. 규칙 파일은 bart create 명령이 /usr/bin 디렉토리의 내용에 대한 정보를 나열하도록 지시합니다. 또한 규칙 파일은 bart compare 명령이 동일 디렉토리에서 크기 및 내용 변경 사항만 추적하도록 지시합니다.

```
# Check size and content changes in the /usr/bin directory.
# This rules file only checks size and content changes.
# See rules file example.

IGNORE all
CHECK size contents
/usr/bin
```

- 만든 규칙 파일을 사용하여 제어 매니페스트를 만듭니다.

  ```
  # bart create -r bartrules.txt > usr_bin.control-manifest.121003
  ```

- /usr/bin 디렉토리에 대한 변경 사항을 모니터하려고 할 때마다 테스트 매니페스트를 만듭니다.

  ```
  # bart create -r bartrules.txt > usr_bin.test-manifest.121103
  ```

- 동일한 규칙 파일을 사용하여 매니페스트를 비교합니다.

  ```
  # bart compare -r bartrules.txt usr_bin.control-manifest \
  usr_bin.test-manifest
  ```

- bart compare 명령의 출력을 조사합니다.

  ```
   /usr/bin/gunzip:  add
  /usr/bin/ypcat:
    delete
  ```

위의 출력에서 bart compare 명령은 /usr/bin 디렉토리의 불일치를 보고했습니다. 출력은 /usr/bin/ypcat 파일이 삭제되고 /usr/bin/gunzip 파일이 추가되었음을 나타냅니다.

BART 매니페스트, 규칙 파일 및 보고서(참조)

이 절에서는 BART에서 사용하고 만드는 파일의 형식을 설명합니다.

BART 매니페스트 파일 형식

각 매니페스트 파일 항목은 파일 유형에 따라 단일 행입니다. 각 항목은 파일의 이름인 *fname*으로 시작합니다. 파일 이름에 내장된 특수 문자로 인한 구문 분석 문제를 막기 위해 파일 이름은 인코딩됩니다. 자세한 내용은 112 페이지 “BART 규칙 파일 형식”을 참조하십시오.

이후 필드는 다음 파일 속성을 나타냅니다.

type 다음 값을 가질 수 있는 파일 유형입니다.

- B - 블록 장치 노드의 경우
- C - 문자 장치 노드의 경우
- D - 디렉토리의 경우
- F - 파일의 경우
- L - 심볼릭 링크의 경우
- P - 파이프의 경우
- S - 소켓의 경우

size 바이트 단위의 파일 크기입니다.

mode 파일의 권한을 나타내는 8진수 숫자입니다.

acl 파일에 대한 ACL 속성입니다. ACL 속성이 있는 파일의 경우 여기에는 `acltotext()`의 출력이 포함됩니다.

uid 이 항목 소유자의 숫자 사용자 ID입니다.

gid 이 항목 소유자의 숫자 그룹 ID입니다.

dirmtime 1970년 1월 1일 00:00:00 UTC 이후 디렉토리에 대한 마지막 수정 시간(초 단위)입니다.

lnmtime 1970년 1월 1일 00:00:00 UTC 이후 링크에 대한 마지막 수정 시간(초 단위)입니다.

mtime 1970년 1월 1일 00:00:00 UTC 이후 파일에 대한 마지막 수정 시간(초 단위)입니다.

contents 파일의 체크섬 값입니다. 이 속성은 일반 파일에 대해서만 지정됩니다. 컨텍스트 검사를 해제하거나 체크섬을 계산할 수 없는 경우 이 필드의 값은 –입니다.

dest 심볼릭 링크의 대상입니다.

devnode 장치 노드의 값입니다. 이 속성은 문자 장치 파일 및 블록 장치 파일 전용입니다.

BART 매니페스트에 대한 자세한 내용은 `bart_manifest(4)` 매뉴얼 페이지를 참조하십시오.

BART 규칙 파일 형식

bart 명령에 대한 입력 파일은 텍스트 파일입니다. 이러한 파일은 매니페스트에 포함될 파일 및 보고서에 포함될 파일 속성을 지정하는 행으로 구성됩니다. 두 BART 기능 모두에서 동일한 입력 파일을 사용할 수 있습니다. #으로 시작하는 행, 빈 행 및 공백이 포함된 행은 도구에서 무시됩니다.

입력 파일에는 세 가지 유형의 지시어가 있습니다.

- 하위 트리 지시어(선택적 패턴 일치 수정자 포함)
- CHECK 지시어
- IGNORE 지시어

예 6–5 규칙 파일 형식

```
<Global CHECK/IGNORE Directives>
<subtree1> [pattern1..]
<IGNORE/CHECK Directives for subtree1>

<subtree2> [pattern2..]
<subtree3> [pattern3..]
<subtree4> [pattern4..]
<IGNORE/CHECK Directives for subtree2, subtree3, subtree4>
```

주 – 모든 지시어는 순서대로 읽혀지며, 나중에 나온 지시어가 먼저 나온 지시어를 대체할 수 있습니다.

행마다 하나의 하위 트리 지시어가 있습니다. 지시어는 절대 경로 이름 다음에 0개 이상의 패턴 일치 명령문으로 시작**되어야** 합니다.

규칙 파일 속성

bart 명령에서는 CHECK 및 IGNORE 명령문을 사용하여 추적하거나 무시할 속성을 정의합니다. 각 속성에는 연결된 키워드가 있습니다.

키워드 속성은 다음과 같습니다.

- acl
- all
- contents
- dest
- devnode
- dirmtime
- gid
- lnmtime

- mode
- mtime
- size
- type
- uid

all 키워드는 모든 파일 속성을 가리킵니다.

인용 구문

BART에서 사용하는 규칙 파일 사양 언어는 비표준 파일 이름을 나타내기 위한 표준 UNIX 인용 구문입니다. 내장된 탭, 공백, 개행 또는 특수 문자는 도구에서 파일 이름을 읽을 수 있도록 8진수 형식으로 인코딩됩니다. 이 비표준 인용 구문은 특정 파일 이름(예: 내장된 캐리지 리턴이 포함된 파일 이름)이 명령 파이프라인에서 올바르게 처리되지 않도록 합니다. 규칙 사양 언어에서는 셸 구문만을 사용하여 설명하기 어렵거나 충분하지 않은 복잡한 파일 이름 필터링 조건의 표현을 허용합니다.

BART 규칙 파일 또는 BART에서 사용하는 인용 구문에 대한 자세한 내용은 bart_rules(4) 매뉴얼 페이지를 참조하십시오.

BART 보고

기본 모드에서 bart compare 명령은 다음 예에 나온 대로 수정된 디렉토리 시간 기록(dirmtime)를 제외하고 시스템에 설치된 모든 파일을 검사합니다.

```
CHECK all
IGNORE    dirmtime
```

규칙 파일을 제공할 경우 CHECK all 및 IGNORE dirmtime의 전역 지시어가 이 순서대로 자동으로 규칙 파일 앞에 붙습니다.

BART 출력

다음 종료 값이 반환됩니다.

0 성공

1 파일을 처리할 때 치명적이지 않은 오류(예: 권한 문제)

>1 치명적인 오류(예: 잘못된 명령줄 옵션)

보고 방식은 상세 정보 출력과 프로그래밍 출력의 두 가지 출력 유형을 제공합니다.

- 상세 정보 출력은 기본 출력이며 현지화되고 여러 행에 표시됩니다. 상세 정보 출력은 지역화되며 사람이 읽을 수 있습니다. `bart compare` 명령이 두 시스템 매니페스트를 비교하는 경우 파일 차이 목록이 생성됩니다.

  예를 들면 다음과 같습니다.

  filename attribute control:xxxx test:yyyy

  filename 제어 매니페스트와 테스트 매니페스트 간에 서로 다른 파일 이름입니다.

  attribute 비교되는 매니페스트 간에 서로 다른 파일 속성 이름입니다. *xxxx*는 제어 매니페스트의 속성 값이고 *yyyy*는 테스트 매니페스트의 속성 값입니다. 여러 속성에 대한 불일치가 같은 파일에서 발생하는 경우 각 차이가 별도의 행에 나타납니다.

  다음은 `bart compare` 명령에 대한 기본 출력의 예입니다. 속성 차이는 `/etc/passwd` 파일에 대한 것입니다. 출력은 `size`, `mtime` 및 `contents` 속성이 변경되었음을 나타냅니다.

```
/etc/passwd:
size     control:74     test:81
mtime control:3c165879     test:3c165979
contents     control:daca28ae0de97afd7a6b91fde8d57afa
test:84b2b32c4165887355317207b48a6ec7
```

- `bart compare` 명령을 실행할 때 `-p` 옵션을 사용할 경우 프로그래밍 출력이 생성됩니다. 이 출력은 프로그래밍 조작에 알맞은 형식으로 생성됩니다. 프로그래밍 출력은 다른 프로그램에서 쉽게 구문 분석할 수 있으며 다른 도구에 대한 입력으로 사용되도록 설계되었습니다.

  예를 들면 다음과 같습니다.

  filename attribute control-val test-val [*attribute control-val test-val*]*

  filename 기본 형식의 *filename* 속성과 같습니다.

  attribute control-val test-val 각 파일에 대한 제어 매니페스트와 테스트 매니페스트 간에 서로 다른 파일 속성 설명입니다.

`bart` 명령에서 지원하는 속성 목록은 112 페이지 "규칙 파일 속성"을 참조하십시오.

BART에 대한 자세한 내용은 `bart(1M)` 매뉴얼 페이지를 참조하십시오.

◆ ◆ ◆ 7 장

7 파일에 대한 액세스 제어(작업)

이 장에서는 Oracle Solaris에서 파일을 보호하는 방법에 대해 설명합니다. 또한 시스템을 손상시킬 수 있는 사용 권한을 가진 파일로부터 시스템을 보호하는 방법에 대해 설명합니다.

주 – 액세스 제어 목록(ACL)을 사용하여 ZFS 파일을 보호하려면 **Oracle Solaris 관리: ZFS 파일 시스템**의 8 장, "ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호"을 참조하십시오.

다음은 이 장에 포함된 정보 목록입니다.

UNIX 사용 권한으로 파일 보호

UNIX 파일 사용 권한 및 ACL을 통해 파일을 보안할 수 있습니다. 고정된 비트가 설정된 파일 및 실행 파일에는 특수한 보안 조치가 필요합니다.

파일 확인 및 보안 명령

이 표에서는 파일과 디렉토리를 모니터 및 보안하는 명령에 대해 설명합니다.

표 7–1 파일 및 디렉토리 보안 명령

명령	설명	매뉴얼 페이지
`ls`	디렉토리의 파일 및 파일 정보를 나열합니다.	`ls(1)`

표 7–1 파일 및 디렉토리 보안 명령 (계속)

명령	설명	매뉴얼 페이지
chown	파일 소유권을 변경합니다.	chown(1)
chgrp	파일의 그룹 소유권을 변경합니다.	chgrp(1)
chmod	파일 사용 권한을 변경합니다. 문자 및 기호를 사용하는 심볼릭 모드 또는 8진수를 사용하는 절대 모드를 사용하여 파일 사용 권한을 변경할 수 있습니다.	chmod(1)

파일 및 디렉토리 소유권

기존 UNIX 파일 사용 권한은 다음과 같은 세 가지 사용자 클래스에 소유권을 지정할 수 있습니다.

- **사용자** – 파일 또는 디렉토리 소유자(일반적으로 파일을 만든 사용자)입니다. 파일 소유자는 파일 읽기 권한, 파일 쓰기(파일 변경) 권한 또는 파일 실행 권한(파일이 명령인 경우)을 가지는 사용자를 결정할 수 있습니다.
- **그룹** – 사용자 그룹의 구성원입니다.
- **기타** – 파일 소유자가 아니며 그룹 구성원이 아닌 기타 모든 사용자입니다.

일반적으로 파일 소유자는 파일 사용 권한을 지정하거나 수정할 수 있습니다. 또한 root 계정은 파일 소유권을 변경할 수 있습니다. 시스템 정책을 대체하려면 예 7–2를 참조하십시오.

파일 유형은 일곱 가지 중 하나일 수 있습니다. 각 유형은 다음과 같은 기호로 표시됩니다.

-(마이너스 기호) 텍스트 또는 프로그램

b 블록 특정 파일

c 문자 특정 파일

d 디렉토리

l 심볼릭 링크

s 소켓

D 도어

P 명명된 파이프(FIFO)

UNIX 파일 사용 권한

다음 표에서는 파일 또는 디렉토리 사용자의 각 클래스에 부여할 수 있는 사용 권한을 나열하고 설명합니다.

표 7–2 파일 및 디렉토리 사용 권한

기호	사용 권한	객체	설명
r	읽기	파일	지정된 사용자가 파일을 열고 파일 내용을 읽을 수 있습니다.
		디렉토리	지정된 사용자가 디렉토리의 파일을 나열할 수 있습니다.
w	쓰기	파일	지정된 사용자가 파일 내용을 수정하거나 파일을 삭제할 수 있습니다.
		디렉토리	지정된 사용자가 파일을 추가하거나 디렉토리에 링크를 추가할 수 있습니다. 파일을 제거하거나 디렉토리의 링크를 제거할 수도 있습니다.
x	실행	파일	지정된 사용자가 파일을 실행할 수 있습니다(파일이 프로그램 또는 셸 스크립트인 경우). exec(2) 시스템 호출 중 하나로 프로그램을 실행할 수도 있습니다.
		디렉토리	지정된 사용자가 파일을 열거나 디렉토리의 파일을 실행할 수 있습니다. 디렉토리 및 하위 디렉토리를 만들 수도 있습니다.
-	거부됨	파일 및 디렉토리	지정된 사용자가 파일을 읽거나 쓰거나 실행할 수 없습니다.

해당 파일 사용 권한은 일반 파일을 비롯하여 장치, 소켓, 명명된 파이프(FIFO) 등의 특수 파일에 적용됩니다.

심볼릭 링크의 경우 링크가 가리키는 파일의 사용 권한이 적용됩니다.

해당 디렉토리에 대해 제한적인 파일 사용 권한을 설정하여 디렉토리 및 하위 디렉토리의 파일을 보호할 수 있습니다. 단, 수퍼유저는 시스템의 모든 파일 및 디렉토리에 대한 액세스 권한을 가집니다.

특수 파일 사용 권한(setuid, setgid 및 고정된 비트)

실행 파일 및 공용 디렉토리에 대해 세 가지 특수 유형의 사용 권한(setuid, setgid 및 고정된 비트)을 사용할 수 있습니다. 해당 사용 권한이 설정되면 실행 파일을 실행하는 모든 사용자가 실행 파일 소유자(또는 그룹)의 ID를 사용합니다.

특수 사용 권한에는 보안 위험이 따르므로 특수 사용 권한을 설정할 때는 각별히 신중해야 합니다. 예를 들어, 특정 사용자가 사용자 ID(UID)를 0(root의 UID)으로 설정하는 프로그램을 실행하여 수퍼유저 권한을 얻을 수 있습니다. 또한 모든 사용자가 자신이 소유한 파일에 대해 특수 사용 권한을 설정할 수 있으므로 다른 보안 위험에 노출됩니다.

수퍼유저 권한을 얻기 위해 setuid 사용 권한 및 setgid 사용 권한이 무단으로 사용되고 있지 않은지 시스템을 모니터해야 합니다. root 또는 bin 이외의 다른 사용자에게 관리

프로그램의 소유권을 부여하는 사용 권한은 의심스러운 것입니다. 이 특수 사용 권한을 사용하는 모든 파일을 검색하여 나열하려면 130 페이지 “특수 파일 사용 권한이 있는 파일을 찾는 방법”을 참조하십시오.

setuid 사용 권한

실행 파일에 대해 setuid 사용 권한이 설정되면 파일 소유자를 기반으로 이 파일을 실행하는 프로세스에 액세스 권한이 부여됩니다. 액세스 권한은 실행 파일을 실행 중인 사용자를 기반으로 하는 것이 **아닙니다**. 이 특수 사용 권한에 따라 사용자는 일반적으로 소유자에게만 제공되는 파일 및 디렉토리 액세스 권한을 얻을 수 있습니다.

예를 들어, passwd 명령에 대한 setuid 사용 권한은 사용자가 암호를 변경할 수 있도록 합니다. setuid 사용 권한이 있는 passwd 명령은 다음과 유사합니다.

```
-r-sr-sr-x   3 root     sys       28144 Jun 17 12:02 /usr/bin/passwd
```

이 특수 사용 권한에는 보안 위험이 따릅니다. 정해진 일부 사용자는 프로세스 실행이 끝난 후에도 setuid 프로세스로 부여받은 사용 권한을 유지할 수 있습니다.

주 – 프로그램의 예약된 UID(0–100)로 setuid 사용 권한을 사용하면 유효 UID가 제대로 설정되지 않을 수 있습니다. 셸 스크립트를 사용하십시오. 또는 setuid 사용 권한에 예약된 UID를 사용하지 마십시오.

---

setgid 사용 권한

setgid 사용 권한은 setuid 사용 권한과 유사합니다. 프로세스의 유효 그룹 ID(GID)가 파일을 소유한 그룹으로 변경되며 해당 그룹에게 부여된 사용 권한을 기반으로 사용자에게 액세스 권한이 부여됩니다. /usr/bin/mail 명령의 setgid 사용 권한은 다음과 같습니다.

```
-r-x--s--x   1 root    mail      67504 Jun 17 12:01 /usr/bin/mail
```

setgid 사용 권한이 디렉토리에 적용되면 이 디렉토리에서 만들어진 파일은 디렉토리가 속한 그룹에 속합니다. 파일은 만들기 프로세스가 속한 그룹에 속하지 않습니다. 디렉토리에서 쓰기 및 실행 권한을 가지는 사용자는 해당 디렉토리에서 파일을 만들 수 있습니다. 단, 파일은 사용자가 속한 그룹이 아닌 디렉토리를 소유한 그룹에 속합니다.

수퍼유저 권한을 얻기 위해 setgid 사용 권한이 무단으로 사용되고 있지 않은지 시스템을 모니터해야 합니다. root 또는 bin 이외의 다른 비정상적인 그룹에 프로그램에 대한 그룹 액세스 권한을 부여하는 사용 권한은 의심스러운 것입니다. 이 사용 권한을 사용하는 모든 파일을 검색하여 나열하려면 130 페이지 “특수 파일 사용 권한이 있는 파일을 찾는 방법”을 참조하십시오.

고정된 비트

고정된 비트는 디렉토리 내의 파일을 보호하는 사용 권한 비트입니다. 디렉토리에 고정된 비트가 설정된 경우 파일 소유자, 디렉토리 소유자 또는 권한 있는 사용자만 파일을 삭제할 수 있습니다. 권한 있는 사용자의 예로 root 사용자를 들 수 있습니다. 고정된 비트는 사용자가 공용 디렉토리(예: /tmp)에서 다른 사용자의 파일을 삭제하지 못하도록 합니다.

```
drwxrwxrwt 7  root  sys   400 Sep  3 13:37 tmp
```

TMPFS 파일 시스템에서 공용 디렉토리를 설정할 때 수동으로 고정된 비트를 설정해야 합니다. 지침은 예 7–5를 참조하십시오.

기본 umask 값

파일 또는 디렉토리를 만들 때 일련의 기본 사용 권한이 사용됩니다. 시스템 기본값은 공개됩니다. 텍스트 파일의 666 사용 권한은 모든 사용자에게 읽기 및 쓰기 권한을 부여합니다. 디렉토리 및 실행 파일의 777 사용 권한은 모든 사용자에게 읽기, 쓰기 및 실행 권한을 부여합니다. 일반적으로 사용자는 셸 초기화 파일(예: .bashrc 및 .kshrc.user)의 시스템 기본값을 대체합니다. 관리자는 /etc/profile 파일에서 기본값을 설정할 수도 있습니다.

umask 명령으로 지정된 값은 기본값에서 제외됩니다. 이 프로세스는 chmod 명령이 권한을 부여하는 것과 동일한 방법으로 사용 권한을 거부합니다. 예를 들어, chmod 022 명령은 그룹 및 기타에 쓰기 권한을 부여합니다. umask 022 명령은 그룹 및 기타에 대한 쓰기 권한을 거부합니다.

다음 표에서는 일반적인 umask 값과 해당 값이 실행 파일에 끼치는 영향을 보여 줍니다.

표 7–3 다양한 보안 레벨에 대한 umask 설정

보안 레벨	umask 설정	허용되지 않는 사용 권한
허가(744)	022	그룹 및 기타에 대한 w
보통(740)	027	그룹에 대한 w, 기타에 대한 rwx
보통(741)	026	그룹에 대한 w, 기타에 대한 rw
심각(700)	077	그룹 및 기타에 대한 rwx

umask 값 설정에 대한 자세한 내용은 umask(1) 매뉴얼 페이지를 참조하십시오.

파일 사용 권한 모드

chmod 명령을 통해 파일 사용 권한을 변경할 수 있습니다. 사용 권한을 변경하려면 수퍼유저나 파일 또는 디렉토리의 소유자여야 합니다.

chmod 명령을 사용하여 다음 두 가지 모드 중 하나로 사용 권한을 설정할 수 있습니다.

- **절대 모드** – 숫자를 사용하여 파일 사용 권한을 나타냅니다. 절대 모드를 사용하여 사용 권한을 변경하는 경우 8진수 모드 숫자별로 각 세 문자의 사용 권한을 나타냅니다. 절대 모드는 사용 권한 설정에 사용되는 가장 일반적인 방법입니다.
- **심볼릭 모드** – 문자와 기호의 조합을 통해 사용 권한을 추가하거나 사용 권한을 제거합니다.

다음 표에서는 절대 모드로 파일 사용 권한을 설정하는 데 사용할 8진수 값을 나열합니다. 세 세트의 이러한 숫자를 사용하여 소유자, 그룹 및 기타에 대한 사용 권한을 이 순서대로 설정할 수 있습니다. 예를 들어, 값 644는 소유자에 대해 읽기 및 쓰기 권한을 설정하고 그룹 및 기타에 대해 읽기 전용 권한을 설정합니다.

표 7–4 절대 모드로 파일 사용 권한 설정

8진수 값	파일 사용 권한 세트	사용 권한 설명
0	---	권한 없음
1	--x	실행 권한만
2	-w-	쓰기 권한만
3	-wx	쓰기 및 실행 권한
4	r--	읽기 권한만
5	r-x	읽기 및 실행 권한
6	rw-	읽기 및 쓰기 권한
7	rwx	읽기, 쓰기 및 실행 권한

다음 표에서는 심볼릭 모드로 파일 사용 권한을 설정하는 데 사용할 기호를 나열합니다. 기호는 사용 권한을 설정 또는 변경할 사용자, 수행할 연산 및 지정 또는 변경하려는 사용 권한을 지정할 수 있습니다.

표 7–5 심볼릭 모드로 파일 사용 권한 설정

기호	기능	설명
u	*who*	사용자(소유자)
g	*who*	그룹

표 7–5 심볼릭 모드로 파일 사용 권한 설정 *(계속)*

기호	기능	설명
o	*who*	기타
a	*who*	모두
=	*operator*	지정
+	*operator*	추가
-	*operator*	제거
r	*permissions*	읽기
w	*permissions*	쓰기
x	*permissions*	실행
l	*permissions*	필수 잠금, `setgid` 비트 설정, 그룹 실행 비트 해제
s	*permissions*	`setuid` 또는 `setgid` 비트 설정
t	*permissions*	고정된 비트 설정, 기타에 대한 실행 비트 설정

기능 열에 지정된 *who operator permissions*는 파일 또는 디렉토리 사용 권한을 변경하는 기호를 지정합니다.

who 사용 권한을 변경할 사용자를 지정합니다.

operator 수행할 연산을 지정합니다.

permissions 변경할 사용 권한을 지정합니다.

절대 모드 또는 심볼릭 모드로 특수 파일 사용 권한을 설정할 수 있습니다. 하지만 디렉토리에 대한 `setuid` 사용 권한을 설정하거나 제거하려면 심볼릭 모드를 사용해야 합니다. 절대 모드에서는 사용 권한 세 문자의 왼쪽에 새 8진수 값을 추가하여 특수 사용 권한을 설정합니다. 다음 표에서는 특수 파일 사용 권한을 설정하는 데 사용할 8진수 값을 나열합니다.

표 7–6 절대 모드로 특수 파일 사용 권한 설정

8진수 값	특수 파일 사용 권한
1	고정된 비트
2	`setgid`
4	`setuid`

액세스 제어 목록을 사용하여 UFS 파일 보호

기존 UNIX 파일 보호에서는 세 가지 사용자 클래스(파일 소유자, 파일 그룹 및 기타)에 대해 읽기, 쓰기 및 실행 권한을 제공합니다. UFS 파일 시스템에서는 액세스 제어 목록(ACL)이 다음 작업을 가능하게 하는 향상된 파일 보안을 제공합니다.

- 파일 소유자, 그룹, 기타, 특정 사용자 및 그룹에 대한 파일 사용 권한 정의
- 위 범주 각각에 대한 기본 사용 권한 정의

주 – ZFS 파일 시스템의 ACL 및 NFSv4 파일의 ACL은 **Oracle Solaris 관리: ZFS 파일 시스템**의 8 장, "ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호"를 참조하십시오.

예를 들어, 그룹의 모든 사용자가 파일을 읽을 수 있도록 하려는 경우 해당 파일에 대해 그룹 읽기 권한을 부여하면 됩니다. 이제 그룹의 특정 사용자만 해당 파일에 쓸 수 있도록 하려고 한다고 가정합니다. 표준 UNIX는 해당 파일 보안 레벨을 제공하지 않지만 ACL은 이 파일 보안 레벨을 제공합니다.

UFS 파일 시스템에서는 `setfacl` 명령을 통해 파일에 대한 ACL 항목이 설정됩니다. UFS ACL 항목은 콜론으로 구분된 다음 필드로 구성됩니다.

entry-type:[*uid*|*gid*]:*perms*

entry-type 파일 사용 권한을 설정할 ACL 항목의 유형입니다. 예를 들어, *entry-type*은 `user`(파일 소유자) 또는 `mask`(ACL 마스크)일 수 있습니다.

uid 사용자 이름 또는 사용자 ID(UID)입니다.

gid 그룹 이름 또는 그룹 ID(GID)입니다.

perms *entry-type*에 대해 설정된 사용 권한을 나타냅니다. *perms*는 심볼릭 문자 `rwx` 또는 8진수로 표시될 수 있습니다. 이러한 숫자는 `chmod` 명령에 사용되는 숫자와 동일합니다.

다음 예에서는 ACL 항목이 사용자 `stacey`에 대해 읽기 및 쓰기 권한을 설정합니다.

```
user:stacey:rw-
```

주의 – UFS 파일 시스템 속성(예: ACL)은 UFS 파일 시스템에서만 지원됩니다. 따라서 ACL 항목이 있는 파일을 `/tmp` 디렉토리(일반적으로 TMPFS 파일 시스템으로 마운트됨)에 복원하거나 복사할 경우 ACL 항목이 손실됩니다. UFS 파일의 임시 저장소로 `/var/tmp` 디렉토리를 사용하십시오.

UFS 파일 시스템의 ACL에 대한 자세한 내용은 Oracle Solaris 10 릴리스용 **시스템 관리 설명서: 보안 서비스**를 참조하십시오.

보안 손상으로부터 실행 파일 보호

프로그램은 스택의 데이터를 읽고 씁니다. 일반적으로 프로그램은 코드용으로 특별히 지정된 메모리의 읽기 전용 부분에서 실행됩니다. 스택 버퍼 오버플로우를 야기하는 일부 공격은 스택에 새 코드를 삽입하여 프로그램이 해당 코드를 실행하도록 합니다. 스택 메모리에서 실행 권한을 제거하면 이러한 공격이 성공하지 못하도록 방지됩니다. 즉, 대부분의 프로그램은 실행 가능 스택을 사용하지 않고도 제대로 작동할 수 있습니다.

64비트 프로세스에는 항상 실행할 수 없는 스택이 사용됩니다. `noexec_user_stack` 변수를 통해 32비트 프로세스의 스택을 실행할 수 있는지 여부를 지정할 수 있습니다. 32비트 SPARC ABI를 준수하도록 기본값은 스택을 실행 가능한 것으로 지정하는 0입니다.

이 변수를 설정하면 스택에서 코드를 실행하려고 시도하는 프로그램에 `SIGSEGV` 신호가 전송됩니다. 일반적으로 이 신호가 전송되면 프로그램이 코어 덤프와 함께 종료됩니다. 또한 해당 프로그램은 잘못된 프로그램의 이름, 프로세스 ID 및 프로그램을 실행한 사용자의 실제 UID를 포함하는 경고 메시지를 생성합니다. 예를 들면 다음과 같습니다.

```
a.out[347] attempt to execute code on stack by uid 555
```

메시지는 `syslog kern` 기능이 `notice` 레벨로 설정된 경우 `syslog` 데몬을 통해 기록됩니다. 기본적으로 이 로깅은 `syslog.conf` 파일에서 설정되므로 콘솔과 `/var/adm/messages` 파일에 메시지가 전송됩니다. 자세한 내용은 `syslogd(1M)` 및 `syslog.conf(4)` 매뉴얼 페이지를 참조하십시오.

`syslog` 메시지는 잠재적인 보안 문제를 관찰하는 데 유용합니다. 또한 메시지는 `noexec_user_stack` 변수를 설정하여 올바른 작업으로부터 금지된 실행 가능 스택에 종속되어 있는 유효한 프로그램을 식별합니다. 메시지가 기록되지 않도록 하려면 `/etc/system` 파일에서 로그 변수 `noexec_user_stack_log`를 0으로 설정하십시오. 메시지가 기록되고 있지 않더라도 계속해서 `SIGSEGV` 신호는 실행 중인 프로그램이 코어 덤프와 함께 종료되도록 할 수 있습니다.

명시적으로 프로그램이 스택을 실행 가능 스택으로 표시하도록 하려는 경우 `mprotect()` 함수를 사용할 수 있습니다. 자세한 내용은 `mprotect(2)` 매뉴얼 페이지를 참조하십시오. 또한 `-M /usr/lib/ld/map.noexstk`로 프로그램을 컴파일하여 시스템 차원의 설정에 관계없이 스택을 실행할 수 없는 스택으로 설정할 수 있습니다.

파일 보호(작업)

다음 절차에서는 UNIX 사용 권한으로 파일을 보호하고, 보안 위험이 있는 파일을 찾고, 해당 파일로 인한 손상으로부터 시스템을 보호합니다.

UNIX 사용 권한으로 파일 보호(작업 맵)

다음 작업 맵에서는 파일 사용 권한을 나열하고, 파일 사용 권한을 변경하고, 특수 파일 사용 권한으로 파일을 보호하는 절차에 대해 설명합니다.

작업	수행 방법
파일 정보를 표시합니다.	124 페이지 "파일 정보 표시 방법"
로컬 파일 소유권을 변경합니다.	125 페이지 "파일 소유자 변경 방법" 126 페이지 "파일의 그룹 소유권 변경 방법"
로컬 파일 사용 권한을 변경합니다.	126 페이지 "심볼릭 모드로 파일 사용 권한 변경 방법" 127 페이지 "절대 모드로 파일 사용 권한 변경 방법" 128 페이지 "절대 모드로 특수 파일 사용 권한 변경 방법"

▼ 파일 정보 표시 방법

`ls` 명령을 사용하여 디렉토리의 모든 파일에 대한 정보를 표시합니다.

- **다음 명령을 입력하여 현재 디렉토리에 있는 모든 파일의 긴 목록을 표시합니다.**

```
% ls -la
```

`-l` 사용자 소유권, 그룹 소유권 및 파일 사용 권한이 포함된 긴 형식을 표시합니다.

`-a` 점(.)으로 시작하는 숨겨진 파일을 비롯하여 모든 파일을 표시합니다.

예 7–1 파일 정보 표시

다음 예에서는 `/sbin` 디렉토리에 있는 파일의 부분 목록이 표시됩니다.

```
% cd /sbin
% ls -la
total 4960
drwxr-xr-x   2 root     sys           64 Dec  8 11:57 ./
drwxr-xr-x  39 root     root          41 Dec  8 15:20 ../
-r-xr-xr-x   1 root     bin        21492 Dec  1 20:55 autopush*
-r-xr-xr-x   1 root     bin        33680 Oct  1 11:36 beadm*
-r-xr-xr-x   1 root     bin       184360 Dec  1 20:55 bootadm*
lrwxrwxrwx   1 root     root          21 Jun  7  2010 bpgetfile -> ...
-r-xr-xr-x   1 root     bin        86048 Dec  1 20:55 cryptoadm*
-r-xr-xr-x   1 root     bin        12828 Dec  1 20:55 devprop*
-r-xr-xr-x   1 root     bin       130132 Dec  1 20:55 dhcpagent*
-r-xr-xr-x   1 root     bin        13076 Dec  1 20:55 dhcpinfo*

   .
```

.
.

각 행에는 파일 정보가 다음 순서대로 표시됩니다.

- 파일 유형 – 예: d. 파일 유형 목록은 116 페이지 "파일 및 디렉토리 소유권"을 참조하십시오.
- 사용 권한 – 예: r-xr-xr-x. 설명은 116 페이지 "파일 및 디렉토리 소유권"을 참조하십시오.
- 하드 링크 수 – 예: 2
- 파일 소유자 – 예: root
- 파일 그룹 – 예: bin
- 파일 크기(바이트) – 예: 21308
- 파일이 만들어진 날짜 또는 파일이 마지막으로 변경된 날짜 – 예: Dec 9 15:55
- 파일 이름 – 예: dhcpinfo

▼ 파일 소유자 변경 방법

시작하기 전에 파일 또는 디렉토리 소유자가 아닌 경우 객체 액세스 관리 권한 프로파일에 지정되어 있어야 합니다. 공용 객체인 파일을 변경하려면 수퍼유저여야 합니다.

1 파일 사용 권한을 표시합니다.

```
% ls -l example-file
-rw-r--r--   1 janedoe   staff   112640 May 24 10:49 example-file
```

2 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

3 파일 소유자를 변경합니다.

```
# chown stacey example-file
```

4 파일 소유자가 변경되었는지 확인합니다.

```
# ls -l example-file
-rw-r--r--   1 stacey   staff   112640 May 26 08:50 example-file
```

NFS 마운트된 파일 시스템에서는 소유권 및 그룹 변경 제한 사항이 추가로 적용됩니다. 자세한 내용은 **Oracle Solaris 관리: 네트워크 서비스**의 6 장, "네트워크 파일 시스템 액세스(참조)"를 참조하십시오.

예 7–2 사용자가 고유 파일의 소유권을 변경할 수 있도록 설정

보안 고려 사항 – 특별한 이유가 있는 경우에만 rstchown 변수 설정을 0으로 변경해야 합니다. 기본 설정은 공간 커터를 무시하기 위해 기타에 속한 것처럼 하여 사용자가 파일을 나열하지 못하도록 합니다.

이 예에서는 /etc/system 파일에서 rstchown 변수의 값을 0으로 설정합니다. 이와 같이 설정하면 파일 소유자가 chown 명령을 사용하여 파일 소유권을 다른 사용자로 변경할 수 있습니다. 또한 소유자는 chgrp 명령을 사용하여 파일의 그룹 소유권을 소유자가 속하지 않은 그룹으로 설정할 수 있습니다. 시스템을 재부트하면 변경 사항이 적용됩니다.

```
set rstchown = 0
```

자세한 내용은 chown(1) 및 chgrp(1) 매뉴얼 페이지를 참조하십시오.

▼ 파일의 그룹 소유권 변경 방법

시작하기 전에 파일 또는 디렉토리 소유자가 아닌 경우 객체 액세스 관리 권한 프로파일에 지정되어 있어야 합니다. 공용 객체인 파일을 변경하려면 수퍼유저여야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**

   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2. **파일의 그룹 소유권을 변경합니다.**

   ```
   $ chgrp scifi example-file
   ```

   그룹 설정에 대한 자세한 내용은 **Oracle Solaris 관리: 일반 작업**의 2 장, “사용자 계정 및 그룹 관리(개요)”를 참조하십시오.

3. **파일의 그룹 소유권이 변경되었는지 확인합니다.**

   ```
   $ ls -l example-file
    -rw-r--r--   1 stacey   scifi   112640 June 20 08:55  example-file
   ```

   예 7–2를 참조하십시오.

▼ 심볼릭 모드로 파일 사용 권한 변경 방법

다음 절차에서는 사용자가 소유한 파일의 사용 권한을 변경합니다.

1. **심볼릭 모드로 사용 권한을 변경합니다.**

   ```
   % chmod who operator permissions filename
   ```

who	사용 권한을 변경할 사용자를 지정합니다.
operator	수행할 연산을 지정합니다.
permissions	변경할 사용 권한을 지정합니다. 유효한 기호 목록은 표 7–5를 참조하십시오.
filename	파일 또는 디렉토리를 지정합니다.

2 파일 사용 권한이 변경되었는지 확인합니다.

```
% ls -l filename
```

주 – 파일 또는 디렉토리 소유자가 아닌 경우 객체 액세스 관리 권한 프로파일에 지정되어 있어야 합니다. 공용 객체인 파일을 변경하려면 수퍼유저여야 합니다.

예 7–3 심볼릭 모드로 사용 권한 변경

다음 예에서는 기타에서 읽기 권한을 제거합니다.

```
% chmod o-r example-file1
```

다음 예에서는 로컬 파일에 사용자, 그룹 및 기타에 대한 읽기 및 실행 권한이 추가됩니다.

```
$ chmod a+rx example-file2
```

다음 예에서는 로컬 파일에 그룹에 대한 읽기, 쓰기 및 실행 권한이 지정됩니다.

```
$ chmod g=rwx example-file3
```

▼ 절대 모드로 파일 사용 권한 변경 방법

다음 절차에서는 사용자가 소유한 파일의 사용 권한을 변경합니다.

1 절대 모드로 사용 권한을 변경합니다.

```
% chmod nnn filename
```

nnn 파일 소유자, 파일 그룹 및 기타(해당 순서대로)에 대한 사용 권한을 나타내는 8진수 값을 지정합니다. 유효한 8진수 값 목록은 표 7–4를 참조하십시오.

filename 파일 또는 디렉토리를 지정합니다.

주 – `chmod` 명령을 사용하여 ACL 항목이 있는 파일의 파일 그룹 사용 권한을 변경하면 파일 그룹 사용 권한과 ACL 마스크가 모두 새 사용 권한으로 변경됩니다. 새 ACL 마스크 사용 권한은 파일에 ACL 항목이 있는 기타 사용자 및 그룹에 대한 사용 권한을 변경할 수 있습니다. 모든 ACL 항목에 대해 적절한 사용 권한이 설정되도록 하려면 `getfacl` 명령을 사용하십시오. 자세한 내용은 `getfacl(1)` 매뉴얼 페이지를 참조하십시오.

---

2 파일 사용 권한이 변경되었는지 확인합니다.

```
% ls -l filename
```

주 – 파일 또는 디렉토리 소유자가 아닌 경우 객체 액세스 관리 권한 프로파일에 지정되어 있어야 합니다. 공용 객체인 파일을 변경하려면 수퍼유저여야 합니다.

예 7–4 절대 모드로 사용 권한 변경

다음 예에서는 공용 디렉토리의 사용 권한이 744(읽기, 쓰기, 실행/읽기 전용/읽기 전용)에서 755(읽기, 쓰기, 실행/읽기 및 실행/읽기 및 실행)로 변경됩니다.

```
# ls -ld public_dir
drwxr--r--  1 jdoe   staff    6023 Aug  5 12:06 public_dir
# chmod 755 public_dir
# ls -ld public_dir
drwxr-xr-x  1 jdoe   staff    6023 Aug  5 12:06 public_dir
```

다음 예에서는 실행 가능 셸 스크립트의 사용 권한이 읽기 및 쓰기에서 읽기, 쓰기 및 실행으로 변경됩니다.

```
% ls -l my_script
-rw------- 1 jdoe   staff    6023 Aug  5 12:06 my_script
% chmod 700 my_script
% ls -l my_script
-rwx------ 1 jdoe   staff    6023 Aug  5 12:06 my_script
```

▼ 절대 모드로 특수 파일 사용 권한 변경 방법

시작하기 전에 파일 또는 디렉토리 소유자가 아닌 경우 객체 액세스 관리 권한 프로파일에 지정되어 있어야 합니다. 공용 객체인 파일을 변경하려면 수퍼유저여야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 절대 모드로 특수 파일 사용 권한을 변경합니다.

```
% chmod nnnn filename
```

nnnn 파일 또는 디렉토리 사용 권한을 변경하는 8진수 값을 지정합니다. 맨 왼쪽에 있는 8진수 값은 특수 파일 사용 권한을 설정합니다. 특수 사용 권한에 대한 유효한 8진수 값 목록은 표 7–6을 참조하십시오.

filename 파일 또는 디렉토리를 지정합니다.

주 – chmod 명령을 사용하여 ACL 항목이 있는 파일의 파일 그룹 사용 권한을 변경하면 파일 그룹 사용 권한과 ACL 마스크가 모두 새 사용 권한으로 변경됩니다. 새 ACL 마스크 사용 권한은 파일에 ACL 항목이 있는 추가 사용자 및 그룹에 대한 사용 권한을 변경할 수 있습니다. 모든 ACL 항목에 대해 적절한 사용 권한이 설정되도록 하려면 getfacl 명령을 사용하십시오. 자세한 내용은 getfacl(1) 매뉴얼 페이지를 참조하십시오.

3 파일 사용 권한이 변경되었는지 확인합니다.

```
% ls -l filename
```

예 7–5 절대 모드로 특수 파일 사용 권한 설정

다음 예에서는 dbprog 파일에 대해 setuid 사용 권한이 설정됩니다.

```
# chmod 4555 dbprog
# ls -l dbprog
-r-sr-xr-x   1 db     staff        12095 May  6 09:29 dbprog
```

다음 예에서는 dbprog2 파일에 대해 setgid 사용 권한이 설정됩니다.

```
# chmod 2551 dbprog2
# ls -l dbprog2
-r-xr-s--x   1 db     staff       24576 May  6 09:30 dbprog2
```

다음 예에서는 public_dir 디렉토리에 대해 고정된 비트 사용 권한이 설정됩니다.

```
# chmod 1777 public_dir
# ls -ld public_dir
drwxrwxrwt   2 jdoe   staff          512 May 15 15:27 public_dir
```

보안 위험이 있는 프로그램 보호(작업 맵)

다음 작업 맵에서는 시스템에서 위험성이 있는 실행 파일을 찾고 프로그램이 실행 가능 스택을 악용하지 못하도록 하는 절차에 대해 설명합니다.

작업	설명	수행 방법
특수 사용 권한을 가진 파일을 찾습니다.	setuid 비트가 설정되었지만 root 사용자가 소유하지 않은 파일을 찾습니다.	130 페이지 “특수 파일 사용 권한이 있는 파일을 찾는 방법”
실행 가능 스택이 오버플로우되지 않도록 합니다.	프로그램이 실행 가능 스택을 악용하지 않도록 합니다.	131 페이지 “프로그램이 실행 가능 스택을 사용하지 못하도록 하는 방법”

작업	설명	수행 방법
실행 가능 스택 메시지가 기록되지 않도록 합니다.	실행 가능 스택 메시지의 기록을 해제합니다.	예 7–7

▼ 특수 파일 사용 권한이 있는 파일을 찾는 방법

이 절차에서는 프로그램에서 setuid 및 setgid 사용 권한의 잠재적인 무단 사용을 찾습니다. root 또는 bin 이외의 다른 사용자에게 소유권을 부여하는 실행 파일은 의심스러운 것입니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 find 명령을 사용하여 setuid 사용 권한이 있는 파일을 찾습니다.

```
# find directory -user root -perm -4000 -exec ls -ldb {} \; >/tmp/filename
```

find *directory*	지정된 *directory*에서 시작하는 마운트된 경로를 모두 확인합니다. directory는 root(/), sys, bin 또는 mail일 수 있습니다.
-user root	root가 소유한 파일만 표시합니다.
-perm -4000	사용 권한이 4000으로 설정된 파일만 표시합니다.
-exec ls -ldb	find 명령 출력을 ls -ldb 형식으로 표시합니다.
/tmp/*filename*	find 명령 결과가 포함된 파일입니다.

2 /tmp/*filename*에 결과를 표시합니다.

```
# more /tmp/filename
```

setuid 사용 권한에 대한 배경 정보는 118 페이지 “setuid 사용 권한”을 참조하십시오.

예 7–6 setuid 사용 권한이 있는 파일 찾기

다음 예의 출력에서는 rar이라는 그룹의 사용자가 /usr/bin/sh의 개인용 복사본을 만들고 setuid 사용 권한을 root로 설정했음을 보여 줍니다. 따라서 /usr/rar/bin/sh 프로그램이 root 권한으로 실행됩니다.

이 출력은 /var/tmp/chkprm 디렉토리를 아카이브로 이동하여 나중에 참조할 수 있도록 저장되었습니다.

```
# find / -user root -perm -4000 -exec ls -ldb {} \; > /var/tmp/ckprm
# cat /var/tmp/ckprm
-r-sr-xr-x 1 root bin 38836 Aug 10 16:16 /usr/bin/at
-r-sr-xr-x 1 root bin 19812 Aug 10 16:16 /usr/bin/crontab
---s--x--x 1 root sys 46040 Aug 10 15:18 /usr/bin/ct
-r-sr-xr-x 1 root sys 12092 Aug 11 01:29 /usr/lib/mv_dir
-r-sr-sr-x 1 root bin 33208 Aug 10 15:55 /usr/lib/lpadmin
```

```
-r-sr-sr-x 1 root bin 38696 Aug 10 15:55 /usr/lib/lpsched
---s--x--- 1 root rar 45376 Aug 18 15:11 /usr/rar/bin/sh
-r-sr-xr-x 1 root bin 12524 Aug 11 01:27 /usr/bin/df
-rwsr-xr-x 1 root sys 21780 Aug 11 01:27 /usr/bin/newgrp
-r-sr-sr-x 1 root sys 23000 Aug 11 01:27 /usr/bin/passwd
-r-sr-xr-x 1 root sys 23824 Aug 11 01:27 /usr/bin/su
# mv /var/tmp/ckprm /export/sysreports/ckprm
```

▼ 프로그램이 실행 가능 스택을 사용하지 못하도록 하는 방법

32비트 실행 가능 스택의 보안 위험에 대한 설명은 123 페이지 "보안 손상으로부터 실행 파일 보호"를 참조하십시오.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 /etc/system 파일을 편집하고 다음 행을 추가합니다.

```
set noexec_user_stack=1
```

2 시스템을 다시 부트합니다.

```
# reboot
```

예 7–7 실행 가능 스택 메시지 로깅을 사용 안함으로 설정

이 예에서는 실행 가능 스택 메시지 로깅이 사용 안함으로 설정된 후 시스템이 재부트됩니다.

```
# cat /etc/system
set noexec_user_stack=1
set noexec_user_stack_log=0
# reboot
```

참조 자세한 내용은 다음을 참조하십시오.

- http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_overview
- http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_continued
- http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_concluded

제 3 부

역할, 권한 프로파일 및 권한

이 절은 RBAC(역할 기반 액세스 제어) 및 프로세스 권한 관리를 다룹니다. RBAC 구성 요소에는 역할, 권한 프로파일, 인증이 있습니다. 프로세스 권한 관리는 권한을 통해 구현됩니다. 권한과 RBAC를 함께 사용하면 수퍼유저의 시스템 관리보다 더 안전한 관리 대안을 제시합니다.

◆ ◆ ◆ 8 장

8 역할 및 권한 사용(개요)

Oracle Solaris의 RBAC(역할 기반 액세스 제어) 기능과 Oracle Solaris의 권한 기능은 수퍼유저보다 더 안전한 대안을 제시합니다. 이 장에서는 RBAC 및 권한에 대한 개요 정보를 제공합니다.

다음은 이 장에 포함된 개요 정보 목록입니다.

역할 기반 액세스 제어(개요)

RBAC(역할 기반 액세스 제어)는 보통 root 역할로 제한되는 작업에 대한 사용자 액세스를 제어하기 위한 보안 기능입니다. 프로세스 및 사용자에 보안 속성을 적용하여 여러 관리자 사이에 수퍼유저 능력을 분담할 수 있습니다. 프로세스 권한 관리는 **권한**을 통해 구현됩니다. 사용자 권한 관리는 RBAC를 통해 구현됩니다.

- 프로세스 권한 관리의 설명은 146 페이지 "권한(개요)"을 참조하십시오.
- RBAC 작업에 대한 내용은 9 장, "역할 기반 액세스 제어 사용(작업)"을 참조하십시오.
- 참고 사항은 10 장, "Oracle Solaris의 보안 속성(참조)"을 참조하십시오.

RBAC: 수퍼유저 모델의 대안

전통적인 UNIX 시스템에서 root 사용자는 수퍼유저라고도 하며 전권을 갖습니다. root로 실행되는 프로그램이나 setuid 프로그램은 전권을 갖습니다. root 사용자는 모든 파일을 읽거나 쓰고, 모든 프로그램을 실행하며, 모든 프로세스에 종료 신호를 보낼 수 있습니다. 실질적으로, 수퍼유저가 될 수 있는 사람이면 누구나 사이트의 방화벽을 수정하고, 감사 증적을 변경하고, 기밀 레코드를 읽고, 전체 네트워크를 종료할 수 있습니다. setuid 프로그램을 하이재킹할 경우 시스템에 무엇이든 할 수 있습니다.

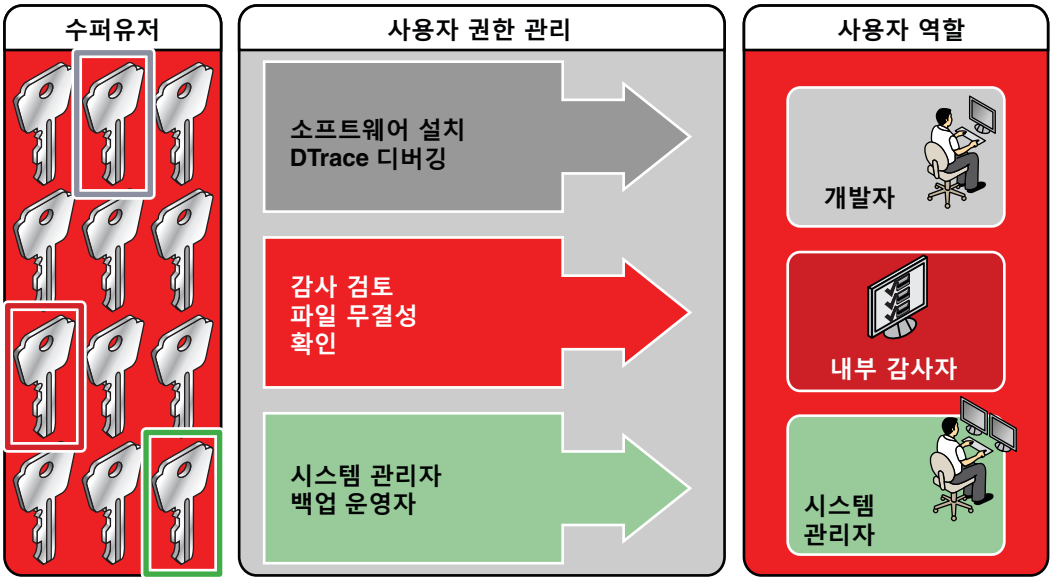
RBAC(역할 기반 액세스 제어)는 all-or-nothing 수퍼유저 모델보다 더 안전한 대안을 제시합니다. RBAC를 사용하면 더욱 세분화된 레벨에서 보안 정책을 시행할 수 있습니다. RBAC는 **최소 권한**의 보안 원칙을 사용합니다. 최소 권한이란, 사용자가 정확하게 작업 수행에 필요한 권한만 할당받아야 한다는 것입니다. 일반 사용자 권한으로 응용 프로그램 사용, 작업 상태 확인, 파일 인쇄, 새 파일 만들기 등을 충분히 수행할 수 있습니다. 일반 사용자 능력 밖의 기능은 권한 프로파일로 그룹화됩니다. 수퍼유저 능력이 필요한 작업을 수행하려는 사용자는 적절한 권한 프로파일이 포함된 역할을 맡습니다.

RBAC는 수퍼유저 능력을 **권한 프로파일**에 모읍니다. 이러한 권한 프로파일은 **역할**이라는 특수한 사용자 계정에 지정됩니다. 그러면 사용자가 수퍼유저 능력이 필요한 작업을 수행하는 역할을 맡을 수 있습니다. 미리 정의된 권한 프로파일이 Oracle Solaris 소프트웨어와 함께 제공됩니다. 역할을 만들고 프로파일을 지정합니다.

권한 프로파일은 광범위한 능력을 제공할 수 있습니다. 예를 들어, System Administrator 권한 프로파일을 통해 프린터 관리 및 cron 작업과 같은 보안에 관련되지 않은 작업을 수행할 수 있습니다. 권한 프로파일을 좁게 정의할 수도 있습니다. 예를 들어, Cron Management 권한 프로파일은 at 및 cron 작업을 관리합니다. 역할을 만들 때 광범위한 능력이나 제한된 능력(또는 둘 다)을 지정할 수 있습니다.

다음 그림은 RBAC가 신뢰된 사용자에 어떻게 권한을 분배하는지 보여줍니다.

그림 8–1 RBAC 권한 분배



RBAC 모델에서 수퍼유저가 하나 이상의 역할을 만듭니다. 역할은 권한 프로파일을 기반으로 합니다. 그런 다음 수퍼유저가 작업을 수행하도록 신뢰된 사용자에게 역할을 지정합니다. 사용자가 사용자 이름으로 로그인합니다. 로그인 후에 사용자는 제한된 관리 명령 및 그래픽 사용자 인터페이스(GUI) 도구를 실행할 수 있는 역할을 맡습니다.

역할 설정의 유연성 덕분에 다양한 보안 정책이 가능합니다. Oracle Solaris와 함께 제공되는 역할은 몇 개 없지만 다양한 역할을 쉽게 구성할 수 있습니다. 대부분의 역할은 동일한 이름의 권한 프로파일을 기반으로 합니다.

- **root** – `root` 사용자와 같은 강력한 역할입니다. 그러나 이 `root`는 로그인할 수 없습니다. 일반 사용자가 로그인한 후에 지정된 `root` 역할을 맡아야 합니다. 이 역할은 기본적으로 구성됩니다.
- **System Administrator** – 보안에 관련되지 않은 작업을 위한 비교적 덜 강력한 역할입니다. 이 역할은 파일 시스템, 메일 및 소프트웨어 설치를 관리할 수 있습니다. 그러나 이 역할은 암호를 설정할 수 없습니다.
- **Operator** – 백업 및 프린터 관리 등의 작업을 위한 하급 관리자 역할입니다.

주 – Media Backup 권한 프로파일은 전체 루트 파일 시스템에 액세스할 수 있습니다. 따라서 Media Backup 및 Operator 권한 프로파일은 하급 관리자용이긴 하지만 신뢰할 수 있는 사용자인지 확인해야 합니다.

하나 이상의 보안 역할을 구성하고 싶을 수 있습니다. Information Security, User Security, Zone Security라는 세 개의 권한 프로파일과 그 보충 프로파일이 보안을 처리합니다. Network Security는 Information Security 권한 프로파일의 보충 프로파일입니다.

이러한 역할은 구현할 필요가 없습니다. 역할은 조직의 보안 요구와 상관 관계가 있습니다. 하나의 전략은 보안, 네트워킹, 방화벽 관리와 같은 분야에 특수 목적의 관리자용 역할을 설정하는 것입니다. 또 다른 전략은 단일의 강력한 관리자 역할을 고급 사용자 역할과 함께 만드는 것입니다. 고급 사용자 역할은 고유 시스템의 일부를 수정하도록 허가된 사용자입니다.

수퍼유저 모델과 RBAC 모델이 공존할 수 있습니다. 다음 표는 RBAC 모델에서 사용 가능한, 수퍼유저부터 제한된 일반 사용자까지 단계적 등급을 요약한 것입니다. 양쪽 모델에서 추적할 수 있는 관리 작업이 포함됩니다. 권한 혼자 시스템에 미치는 효과 요약은 표 8–2를 참조하십시오.

표 8–1 수퍼유저 모델과 RBAC+권한 모델 비교

시스템에서 사용자 능력	수퍼유저 모델	RBAC 모델
전체 수퍼유저 능력을 가진 수퍼유저가 될 수 있음	실행 가능	실행 가능
전체 사용자 능력을 가진 사용자로 로그인할 수 있음	실행 가능	실행 가능

표 8–1 수퍼유저 모델과 RBAC+권한 모델 비교 (계속)

시스템에서 사용자 능력	수퍼유저 모델	RBAC 모델
제한된 능력을 가진 수퍼유저가 될 수 있음	실행 불가능	실행 가능
사용자로 로그인할 수 있고, 때때로 수퍼유저 능력을 가질 수 있음	실행 가능, `setuid` 프로그램만 사용	실행 가능, `setuid` 프로그램과 RBAC 사용
관리 능력을 가졌으나 전체 수퍼유저 능력은 없는 사용자로 로그인할 수 있음	실행 불가능	실행 가능, RBAC와 함께 직접 지정된 권한 및 인증 사용
일반 사용자보다 적은 능력을 가진 사용자로 로그인할 수 있음	실행 불가능	실행 가능, RBAC와 함께 제거된 권한 사용
수퍼유저 작업을 추적할 수 있음	실행 가능, `su` 명령 감사	실행 가능, `pfexec()` 호출 감사 또한 `root` 역할을 맡은 사용자의 이름이 감사 증적에 있음

RBAC 요소 및 기본 개념

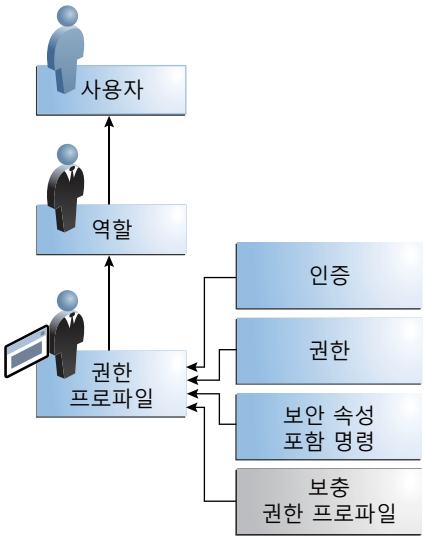
Oracle Solaris의 RBAC 모델은 다음 요소를 소개합니다.

- **인증** – 사용자나 역할이 추가 권한이 필요한 일련의 작업을 수행할 수 있는 사용 권한입니다. 예를 들어, 설치 시 보안 정책은 일반 사용자에게 `solaris.device.cdrw` 인증을 제공합니다. 이 인증을 통해 사용자는 CD-ROM 장치를 읽고 쓸 수 있습니다. 인증 목록은 `/etc/security/auth_attr` 파일을 참조하십시오.
- **권한** – 명령, 사용자, 역할, 시스템에 부여할 수 있는 별개의 권한입니다. 프로세스를 성공하려면 권한이 필요합니다. 예를 들어, `proc_exec` 권한을 통해 프로세스가 `execve()`를 호출할 수 있습니다. 일반 사용자는 기본 권한을 갖습니다. 기본 권한을 보려면 `ppriv -vl basic` 명령을 실행합니다.
- **보안 속성** – 프로세스가 작업을 수행할 수 있는 속성입니다. 전형적인 UNIX 환경에서 보안 속성을 통해 프로세스가 일반 사용자에게 금지된 작업을 수행할 수 있습니다. 예를 들어, `setuid` 및 `setgid` 프로그램에는 보안 속성이 있습니다. RBAC 모델에서 인증 및 권한은 `setuid` 및 `setgid` 프로그램과 더불어 보안 속성입니다. 이러한 속성은 사용자에 지정할 수 있습니다. 예를 들어, `solaris.device.allocate` 권한이 부여된 사용자는 장치에 배타적 사용을 할당할 수 있습니다. 권한을 프로세스에 둘 수 있습니다. 예를 들어, `file_flag_set` 권한을 가진 프로세스는 immutable, no-unlink, append-only 파일 속성을 설정할 수 있습니다.
- **권한 있는 응용 프로그램** – **보안 속성**을 검사하여 시스템 컨트롤을 대체할 수 있는 응용 프로그램 또는 명령입니다. 전형적인 UNIX 환경과 RBAC 모델에서 `setuid` 및 `setgid`를 사용하는 프로그램은 권한 있는 응용 프로그램입니다. RBAC 모델에서 성공을 위해 권한 또는 인증이 필요한 프로그램 역시 권한 있는 응용 프로그램입니다. 자세한 내용은 142 페이지 “권한 있는 응응 프로그램 및 RBAC”를 참조하십시오.

- **권한 프로파일** – 역할이나 사용자에 지정할 수 있는 보안 속성 모음입니다. 권한 프로파일에는 인증, 직접 지정된 권한, 보안 속성 포함 명령 및 기타 권한 프로파일이 포함될 수 있습니다. 다른 프로파일 내의 프로파일을 보충 권한 프로파일이라고 합니다. 권한 프로파일은 보안 속성을 그룹화하는 편리한 방법입니다.
- **역할** – 권한 있는 응용 프로그램을 실행하기 위한 특수한 신원입니다. 특수한 신원은 지정된 사용자만 맡을 수 있습니다. `root` 역할을 포함한 역할로 실행되는 시스템에는 수퍼유저가 불필요합니다. 수퍼유저 능력은 여러 역할로 분배됩니다. 예를 들어, 2-역할 시스템에서 보안 작업은 보안 역할에 의해 처리됩니다. 다른 한 역할은 보안에 관련되지 않은 시스템 관리 작업을 처리합니다. 역할을 더욱 세분화할 수 있습니다. 예를 들어, 시스템에 암호화 프레임워크, 프린터, 시스템 시간, 파일 시스템, 감사 등을 처리하기 위한 별도의 관리 역할을 포함할 수 있습니다.

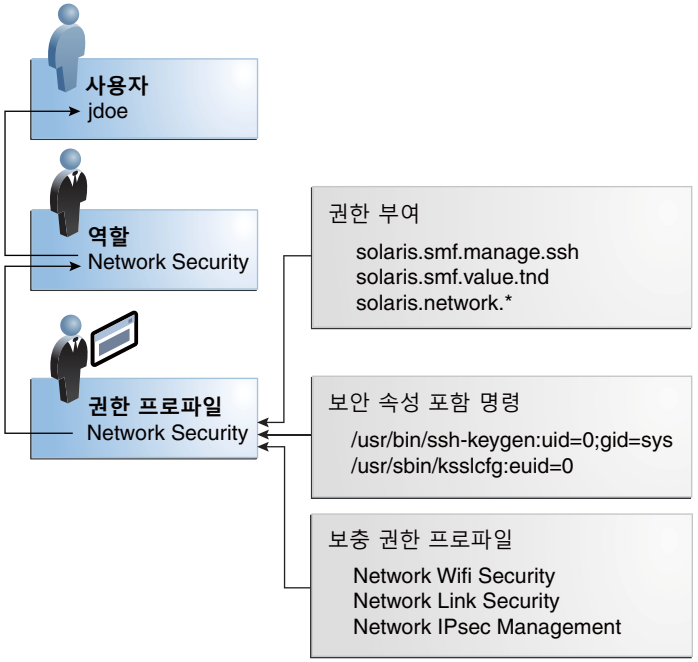
다음 그림은 RBAC 요소가 어떻게 상호 작용하는지 보여줍니다.

그림 8–2 RBAC 요소 관계



다음 그림은 Network Security 역할과 Network Security 권한 프로파일을 사용하여 RBAC 관계를 보여줍니다.

그림 8–3 RBAC 요소 관계의 예



Network Security 역할은 IPsec, Wifi 및 네트워크 링크를 관리하는 데 사용됩니다. 역할이 사용자 jdoe에 지정됩니다. jdoe가 해당 역할로 전환한 후 역할 암호를 제공하면 역할을 맡을 수 있습니다. 관리자는 역할 암호가 아닌 사용자 암호를 받아들이도록 역할을 사용자 정의할 수 있습니다.

그림 8–3에서 Network Security 권한 프로파일이 Network Security 역할에 지정됩니다. Network Security 권한 프로파일은 Network Wifi Security, Network Link Security, Network IPsec Management 순으로 평가되는 보충 프로파일을 포함합니다. 이러한 보충 프로파일이 역할의 주요 업무를 담당합니다.

Network Security 권한 프로파일에는 3개의 직접 지정된 인증, 0개의 직접 지정된 권한, 2개의 보안 속성 포함 명령이 있습니다. 보충 권한 프로파일에는 직접 지정된 인증이 있고, 이들 중 2개에는 보안 속성 포함 명령이 있습니다. Network Security 역할에서 jdoe는 이러한 프로파일에 지정된 인증을 모두 가지며, 이러한 프로파일의 보안 속성 포함 명령을 모두 실행할 수 있습니다. jdoe는 네트워크 보안을 관리할 수 있습니다.

권한 에스컬레이션

Oracle Solaris는 보안을 구성할 때 관리자에게 커다란 유연성을 부여합니다. 처음 설치 시 이 소프트웨어는 권한 에스컬레이션을 허용하지 않습니다. 사용자나 프로세스가

의도했던 것보다 더 많은 관리 권한을 얻을 때 권한 에스컬레이션이 발생합니다. 이 경우 권한은 단순한 권한이 아니라 보안 속성을 의미합니다.

Oracle Solaris 소프트웨어에는 root 역할에만 지정된 보안 속성이 있습니다. 다른 보안 보호를 그대로 둔 채, 관리자는 root 역할용으로 설계된 속성을 다른 계정에 지정할 수 있지만, 이러한 지정 작업은 몹시 주의를 기울여야 합니다.

다음 권한 프로파일과 인증은 비루트 계정의 권한을 에스컬레이트할 수 있습니다.

- **Media Restore 권한 프로파일** – 이 프로파일이 존재하지만, 어떤 다른 권한 프로파일에 속하지 않습니다. Media Restore는 전체 루트 파일 시스템에 액세스할 수 있으므로 이를 사용하여 권한 에스컬레이션이 기능합니다. 고의로 수정된 파일이나 대체 매체를 복원할 수 있습니다. 기본적으로 root 역할에 이 권한 프로파일이 포함됩니다.
- **solaris.*.assign 인증** – 이 인증이 존재하지만, 어떤 권한 프로파일이나 계정에 지정되지 않습니다. solaris.*.assign 인증을 가진 계정은 계정 자체에 지정되지 않은 보안 속성을 다른 사람에게 지정할 수 있습니다. 예를 들어, solaris.profile.assign 인증을 가진 역할은 역할 자체에 지정되지 않은 권한 프로파일을 다른 계정에 지정할 수 있습니다. 기본적으로 root 역할에만 solaris.*.assign 인증이 있습니다.

  최적의 사용법은, solaris.*.assign 인증이 아닌 solaris.*.delegate 인증을 지정하는 것입니다. solaris.*.delegate 인증은 위임자가 보유한 보안 속성만 다른 계정에 지정할 수 있습니다. 예를 들어, solaris.profile.delegate 인증이 지정된 역할은 역할 자체에 지정된 권한 프로파일을 다른 사용자와 역할에 지정할 수 있습니다.

권한 보안 속성에 영향을 주는 에스컬레이션은 208 페이지 “권한 에스컬레이션 금지”를 참조하십시오.

RBAC 인증

인증은 역할이나 사용자에 부여할 수 있는 별개의 권한입니다. 인증은 사용자 응용 프로그램 레벨에서 정책을 시행합니다.

인증을 역할이나 사용자에 직접 지정할 수 있지만, 최적의 사용법은 권한 프로파일에 인증을 포함하는 것입니다. 그런 다음 권한 프로파일이 역할에 추가되고, 역할이 사용자에 지정됩니다. 예제는 그림 8–3을 참조하십시오.

delegate 또는 assign 단어가 포함된 인증은 사용자나 역할이 보안 속성을 다른 사람에게 지정할 수 있습니다.

권한 에스컬레이션을 금지하려면 assign 인증을 계정에 지정하지 마십시오.

- delegate 인증은 위임자가 보유한 보안 속성만 다른 사람에게 지정할 수 있습니다. 예를 들어, solaris.profile.delegate 인증이 지정된 역할은 역할 자체에 지정된 권한 프로파일을 다른 사람에게 지정할 수 있습니다.
- assign 인증은 지정자가 보유하지 않은 보안 속성을 다른 사람에게 지정할 수 있습니다. 예를 들어, solaris.profile.assign 인증을 가진 역할은 권한 프로파일을 다른 사람에게 지정할 수 있습니다.

solaris.*.assign 인증은 전달되지만, 어떤 프로파일에도 포함되지 않습니다. 기본적으로 root 역할에만 solaris.*.assign 인증이 있습니다.

RBAC 호환 응용 프로그램은 응용 프로그램이나 그 안의 특정 작업에 액세스를 부여하기 전에 사용자의 인증을 검사할 수 있습니다. 이 검사는 UID=0에 대한 전통적인 UNIX 응용 프로그램의 검사를 대체합니다. 인증에 대한 자세한 내용은 다음 절을 참조하십시오.

- 200 페이지 “인증”
- 202 페이지 “auth_attr 데이터베이스”
- 205 페이지 “인증이 필요한 선택된 명령”

인증 및 권한

권한은 커널에서 보안 정책을 시행합니다. 인증과 권한의 차이점은 보안 정책을 시행하는 레벨에 있습니다. 적절한 권한 없이 프로세스는 커널을 통해 권한 있는 작업을 수행하는 것을 금지할 수 있습니다. 적절한 인증 없이 사용자는 권한 있는 응응 프로그램을 사용하거나 권한 있는 응응 프로그램 내에서 보안에 민감한 작업을 수행하는 것을 금지할 수 있습니다. 더 자세한 권한 설명은 146 페이지 “권한(개요)”을 참조하십시오.

권한 있는 응응 프로그램 및 RBAC

시스템 컨트롤을 대체할 수 있는 응용 프로그램 및 명령은 권한 있는 응응 프로그램으로 간주됩니다. UID=0과 같은 보안 속성, 권한 및 인증은 응용 프로그램에 권한을 부여합니다.

UID 및 GID를 검사하는 응용 프로그램

root(UID=0) 또는 다른 특수한 UID/GID를 검사하는 권한 있는 응용 프로그램이 UNIX 환경에 오랫동안 존재해 왔습니다. 권한 프로파일 방식을 통해 특수한 ID가 필요한 명령을 격리할 수 있습니다. 누구나 액세스할 수 있는 명령의 ID를 변경하는 대신, 권한 프로파일에 지정된 보안 속성 포함 명령을 배치할 수 있습니다. 그러면 해당 권한 프로파일을 가진 사용자/역할이 수퍼유저가 되지 않고도 프로그램을 실행할 수 있습니다.

실제 또는 유효로 ID를 지정할 수 있습니다. 유효 ID를 지정하는 것이 실제 ID를 지정하는 것보다 선호됩니다. 유효 ID는 파일 사용 권한 비트의 setuid 기능과 같습니다. 유효 ID는 감사용 UID를 식별하기도 합니다. 그러나 일부 셸 스크립트 및 프로그램은 root의 실제 UID가 필요하므로 실제 UID도 설정할 수 있습니다. 예를 들어, reboot 명령은 유효 UID보다 실제 UID가 필요합니다. 유효 ID가 명령을 실행하기에 부족한 경우 실제 ID를 명령에 지정해야 합니다.

권한을 검사하는 응용 프로그램

권한 있는 응용 프로그램은 권한 사용을 검사할 수 있습니다. RBAC 권한 프로파일 방식을 통해 보안 속성이 필요한 특수한 명령에 권한을 지정할 수 있습니다. 그런 다음, 권한 프로파일에서 지정된 보안 속성 포함 명령을 격리할 수 있습니다. 그러면 해당 권한 프로파일을 가진 사용자/역할이 명령 성공을 위해 필요한 권한만으로 명령을 실행할 수 있습니다.

권한을 검사하는 명령은 다음과 같습니다.

- Kerberos 명령 - kadmin, kprop, kdb5_util
- 네트워크 명령 - ipadm, routeadm, snoop
- 파일 및 파일 시스템 명령 - chmod, chgrp, mount
- 프로세스를 제어하는 명령 - kill, pcred, rcapadm

권한 포함 명령을 권한 프로파일에 추가하려면 170 페이지 “감사 프로파일을 만들거나 변경하는 방법” 및 profiles(1) 매뉴얼 페이지를 참조하십시오. 특정 프로파일에서 권한을 검사하는 명령을 확인하려면 156 페이지 “모든 정의된 보안 속성을 보는 방법”을 참조하십시오.

인증을 검사하는 응용 프로그램

Oracle Solaris는 추가적으로 인증을 검사하는 명령을 제공합니다. 정의상, root 사용자는 모든 인증을 가집니다. 따라서 root 사용자는 어떤 응용 프로그램도 실행할 수 있습니다. 인증을 검사하는 응용 프로그램은 다음과 같습니다.

- 감사 관리 명령 - auditconfig, auditreduce
- 프린터 관리 명령 - lpadmin, lpfilter
- 일괄 처리 작업 관련 명령 - at, atq, batch, crontab
- 장치 지향적 명령 - allocate, deallocate, list_devices, cdrw.

스크립트나 프로그램에서 인증을 테스트하려면 예 9–16을 참조하십시오. 인증이 필요한 프로그램을 작성하려면 **Developer’s Guide to Oracle Solaris 11 Security**의 “About Authorizations”를 참조하십시오.

RBAC 권한 프로파일

권한 프로파일은 관리 권한이 필요한 작업을 수행하기 위해 역할/사용자에 지정할 수 있는 보안 속성 모음입니다. 권한 프로파일에는 인증, 권한, 지정된 보안 속성 포함 명령 및 기타 권한 프로파일이 포함될 수 있습니다. 권한 프로파일에 지정된 권한은 모든 명령에 효력을 발휘합니다. 또한 권한 프로파일은 초기 상속 가능한 세트를 감소/확장하거나 권한의 제한 세트를 감소하는 항목을 포함합니다.

권한 프로파일에 대한 자세한 내용은 다음 절을 참조하십시오.

- 197 페이지 "권한 프로파일"
- 203 페이지 "prof_attr 데이터베이스"
- 203 페이지 "exec_attr 데이터베이스"

RBAC 역할

역할은 권한 있는 응용 프로그램을 실행할 수 있는 특수한 유형의 사용자 계정입니다. 역할은 사용자 계정과 동일한 방법으로 만듭니다. 역할에는 홈 디렉토리, 그룹 지정, 암호 등이 있습니다. 권한 프로파일 및 인증은 역할에 관리 능력을 제공합니다. 역할은 다른 역할이나 다른 사용자로부터 능력을 상속할 수 없습니다. 별개의 역할이 수퍼유저 능력을 분담하므로 더 안전한 관리 방법입니다.

사용자가 역할을 맡을 때 역할의 속성이 모든 사용자 속성을 대체합니다. 역할 정보는 passwd, shadow, user_attr 데이터베이스에 저장됩니다. 역할의 동작을 감사할 수 있습니다. 역할 설정에 대한 자세한 내용은 다음 절을 참조하십시오.

- 163 페이지 "RBAC 구현을 계획하는 방법"
- 165 페이지 "역할을 만드는 방법"
- 178 페이지 "역할의 보안 속성을 변경하는 방법"

역할을 여러 사용자에 지정할 수 있습니다. 동일한 역할을 맡은 모든 사용자는 동일한 역할 홈 디렉토리를 사용하고, 동일한 환경에서 작동하며, 동일한 파일에 액세스할 수 있습니다. 사용자가 명령줄에서 su 명령을 실행하고 역할 이름과 암호를 제공하면 역할을 맡을 수 있습니다. 기본적으로 사용자는 **역할의** 암호를 제공하여 인증됩니다. 관리자는 **사용자의** 암호를 제공하여 인증되도록 시스템을 구성할 수 있습니다. 절차는 183 페이지 "사용자가 고유의 암호를 사용하여 역할을 맡도록 설정하는 방법"을 참조하십시오.

역할을 직접 로그인할 수 없습니다. 사용자가 로그인 후에 역할을 맡습니다. 역할을 맡은 사용자는 현재 역할을 끝내기 전까지 다른 역할을 맡을 수 없습니다. 역할을 끝낸 사용자는 다른 역할을 맡을 수 있습니다.

root는 Oracle Solaris의 역할이므로 익명의 root 로그인을 금지합니다. 프로파일 셸 명령 pfexec가 감사 중인 경우 감사 증적에 사용자 로그인의 실제 UID, 사용자가 맡은 역할, 역할이 수행한 작업이 포함됩니다. 시스템이나 특정 사용자의 역할 운영을 감사하려면 169 페이지 "역할을 감사하는 방법"을 참조하십시오.

소프트웨어와 함께 제공된 권한 프로파일은 역할에 매핑되도록 설계되었습니다. 예를 들어, System Administrator 권한 프로파일을 사용하여 System Administrator 역할을 만들 수 있습니다. 역할을 구성하려면 165 페이지 “역할을 만드는 방법”을 참조하십시오.

프로파일 셸 및 RBAC

사용자 및 역할은 프로파일 셸에서 권한 있는 응용 프로그램을 실행할 수 있습니다. **프로파일 셸**은 권한 프로파일에 포함된 보안 속성을 인식하는 특수한 셸입니다. 관리자가 로그인 셸로 특정 사용자에게 프로파일 셸을 지정할 수 있습니다. 또는 사용자가 역할을 맡기 위해 `su` 명령을 실행할 때 프로파일 셸이 시작됩니다. Oracle Solaris에서 모든 셸에는 대응하는 프로파일 셸이 있습니다. 예를 들어, Bourne 셸(`sh`), Bash 셸(`csh`), Korn 셸(`ksh`)에 대응하는 프로파일 셸은 각각 `pfsh`, `pfbash`, `pfksh`입니다. 프로파일 셸 목록은 `pfexec(1)` 매뉴얼 페이지를 참조하십시오.

권한 프로파일에 직접 지정된 사용자의 로그인 셸이 프로파일 셸이 아닌 경우 보안 속성 포함 명령을 실행하려면 프로파일 셸을 호출해야 합니다. 유용성 및 보안 고려 사항은 145 페이지 “보안 속성을 직접 지정할 때 보안 고려 사항”을 참조하십시오.

프로파일 셸에서 실행된 모든 명령을 감사할 수 있습니다. 자세한 내용은 169 페이지 “역할을 감사하는 방법”을 참조하십시오.

이름 서비스 범위 및 RBAC

이름 서비스 범위는 RBAC를 이해하는 데 중요한 개념입니다. 역할의 범위를 개별 호스트로 제한할 수 있습니다. 다른 방법으로, LDAP과 같은 이름 지정 서비스로 제공된 모든 호스트를 범위에 포함할 수 있습니다. 시스템의 이름 서비스 범위는 이름 스위치 서비스 `svc:/system/name-service/switch`에 지정됩니다. 첫번째 일치 시 조회를 멈춥니다. 예를 들어, 권한 프로파일이 두 이름 서비스 범위에 존재하는 경우 첫번째 이름 서비스 범위의 항목만 사용됩니다. `files`가 첫번째 일치일 경우 역할의 범위가 로컬 호스트로 제한됩니다.

보안 속성을 직접 지정할 때 보안 고려 사항

일반적으로, 사용자는 역할을 통해 관리 능력을 얻습니다. 인증, 권한 및 권한 있는 명령은 권한 프로파일로 그룹화됩니다. 권한 프로파일은 역할에 포함되고, 역할은 사용자에 지정됩니다.

권한 프로파일과 보안 속성의 직접 지정도 가능합니다.

- 권한 프로파일, 권한 및 인증은 사용자에 직접 지정할 수 있습니다.
- 권한 및 인증은 사용자와 역할에 직접 지정할 수 있습니다.

그러나 권한의 직접 지정은 안전한 방법이 아닙니다. 직접 지정된 권한을 가진 사용자/역할은 커널을 통해 이 권한이 필요한 어디서든 보안 정책을 대체할 수 있습니다. 더 안전한 방법은 권한 프로파일에서 명령의 보안 속성으로 권한을 지정하는 것입니다. 그런 다음, 해당 권한 프로파일을 가진 누구나 해당 명령에만 권한을 사용할 수 있습니다.

인증은 사용자 레벨에서 작동하므로 인증의 직접 지정은 권한의 직접 지정보다 덜 위험할 수 있습니다. 그러나 인증을 통해 사용자가 감사 플래그 지정과 같은 고도의 보안 작업을 수행할 수 있습니다.

보안 속성을 직접 지정할 때 유용성 고려 사항

권한 프로파일과 보안 속성의 직접 지정은 유용성에 영향을 미칠 수 있습니다.

- 직접 지정된 권한 및 인증, 그리고 직접 지정된 권한 프로파일의 명령 및 인증이 효력을 발휘하려면 프로파일 셸에서 해석해야 합니다. 기본적으로 사용자에는 프로파일 셸이 지정되지 않습니다.

  프로파일 셸을 열고 해당 셸에서 명령을 실행해야 한다는 것을 잊지 마십시오.
- 인증의 개별적 지정은 확장 불가능합니다. 그리고 직접 지정된 인증이 작업을 수행하기에 부족할 수 있습니다. 작업에 권한 있는 명령이 필요할 수 있습니다.

  권한 프로파일은 인증과 권한 있는 명령을 함께 묶도록 설계되었습니다. 또한 확장 가능합니다.

권한(개요)

프로세스 권한 관리에서는 명령, 사용자, 역할, 시스템 레벨에서 프로세스가 제한됩니다. Oracle Solaris는 **권한**을 통해 프로세스 권한 관리를 구현합니다. 권한을 사용하면 시스템에서 전체 수퍼유저 능력을 보유한 하나의 사용자나 프로세스와 연관된 보안 위험을 줄일 수 있습니다. 권한 및 RBAC는 전통적인 수퍼유저 모델의 강력한 대체 모델입니다.

- RBAC에 대한 내용은 135 페이지 “역할 기반 액세스 제어(개요)”를 참조하십시오.
- 권한 관리 방법에 대한 내용은 186 페이지 “권한 사용(작업)”을 참조하십시오.
- 권한에 대한 참고 사항은 206 페이지 “권한”을 참조하십시오.

권한으로 커널 프로세스 보호

권한은 프로세스가 작업을 수행하는 데 필요한 별개의 권한입니다. 권한은 커널에서 시행됩니다. 권한의 **기본 세트**의 한도 내에서 운영되는 프로그램은 시스템 보안 정책의 한도 내에서 운영됩니다. 시스템 보안 정책의 한도 밖에서 운영되는 프로그램의 예로 `setuid` 프로그램이 있습니다. 권한을 사용할 경우 프로그램이 `setuid`를 호출할 필요가 없습니다.

권한은 시스템에서 가능한 작업 종류를 별개로 열거합니다. 정확히 프로그램 성공을 위해 필요한 권한만으로 프로그램을 실행할 수 있습니다. 예를 들어, 파일을 조작하는 프로그램에 file_dac_write 및 file_flag_set 권한이 필요할 수 있습니다. 이 기능을 이용하면 프로그램을 root로 실행할 필요가 없습니다.

전통적으로, 시스템은 권한 모델을 따르지 않았습니다. 오히려 수퍼유저 모델을 사용했습니다. 수퍼유저 모델에서는 프로세스가 root 또는 사용자로 실행됩니다. 사용자 프로세스는 사용자의 디렉토리와 파일에서 작동하도록 제한되었습니다. root 프로세스는 시스템의 어디든지 디렉토리 및 파일을 만들 수 있습니다. 사용자의 디렉토리 밖에 디렉토리를 만들어야 하는 프로세스는 UID=0, 즉 root로 실행됩니다. 시스템 파일을 보호하기 위해 보안 정책이 DAC(임의 액세스 제어)에 의존했습니다. 장치 노드가 DAC로 보호되었습니다. 예를 들어, sys 그룹이 소유한 장치는 sys 그룹의 구성원만 열 수 있었습니다.

그러나 setuid 프로그램, 파일 사용 권한 및 관리 계정은 오용되기 쉽습니다. setuid 프로세스가 허가한 동작이 작업 완료를 위해 필요한 것보다 훨씬 많습니다. 그러면 전권의 root 사용자로 실행된 침입자에 의해 setuid 프로그램이 손상될 수 있습니다. 마찬가지로, root 암호에 액세스할 수 있는 사용자가 전체 시스템을 손상시킬 수 있습니다.

이와 반대로, 권한으로 정책을 시행하는 시스템은 사용자 능력과 root 능력 사이에 단계적 등급을 허용합니다. 일반 사용자 능력을 벗어난 작업을 수행하려면 사용자에 권한을 부여할 수 있고, root는 현재 보유한 root보다 적은 권한으로 제한될 수 있습니다. RBAC를 사용하여 권한으로 실행되는 명령을 권한 프로파일에 격리하고 하나의 사용자나 역할에 지정할 수 있습니다. 표 8–1은 RBAC+권한 모델이 제공하는, 사용자 능력과 루트 능력 사이의 단계적 등급을 요약한 것입니다.

권한 모델이 수퍼유저 모델보다 훨씬 안전합니다. 프로세스에서 제거된 권한은 악용될 수 없습니다. 프로세스 권한은 프로그램이나 관리 계정이 모든 능력에 액세스하지 못하도록 합니다. 프로세스 권한은 민감한 파일에 추가 보호 조치를 제공할 수 있으며, DAC 보호만으로는 액세스에 악용될 수 있습니다.

그런 다음 프로그램과 프로세스에 필요한 능력만으로 권한을 제한할 수 있습니다. 이 기능을 **최소 권한의 원칙**이라고 합니다. 최소 권한을 구현하는 시스템에서는 프로세스를 탈취하는 침입자가 프로세스가 가진 권한에만 액세스할 수 있습니다. 시스템의 나머지는 손상될 수 없습니다.

권한 설명

권한은 분야에 따라 논리적으로 그룹화됩니다.

- FILE **권한** – file 문자열로 시작하는 권한은 파일 시스템 객체에서 작동합니다. 예를 들어, file_dac_write 권한은 파일에 쓰는 동안 임의 액세스 제어를 대체합니다.
- IPC **권한** – ipc 문자열로 시작하는 권한은 IPC 객체 액세스 제어를 대체합니다. 예를 들어, ipc_dac_read 권한을 통해 프로세스가 DAC로 보호된 원격 공유 메모리를 읽을 수 있습니다.

- NET 권한 – net 문자열로 시작하는 권한은 특정 네트워크 기능에 액세스를 제공합니다. 예를 들어, net_rawaccess 권한을 통해 장치가 네트워크에 연결할 수 있습니다.
- PROC 권한 – proc 문자열로 시작하는 권한을 통해 프로세스 자체의 제한된 등록 정보를 수정할 수 있습니다. PROC 권한은 매우 제한된 효과를 가진 권한입니다. 예를 들어, proc_clock_highres 권한을 통해 프로세스가 고해상도 타이머를 사용할 수 있습니다.
- SYS 권한 – sys 문자열로 시작하는 권한은 다양한 시스템 등록 정보에 무제한 액세스를 제공합니다. 예를 들어, sys_linkdir 권한을 통해 프로세스가 디렉토리의 하드 링크를 연결 및 분리할 수 있습니다.

기타 논리적 그룹에는 CONTRACT, CPC, DTRACE, GRAPHICS, VIRT, WIN, XVM 등이 있습니다.

일부 권한은 시스템에 제한된 효과를 미치고, 일부는 광범위한 효과를 미칩니다. proc_taskid 권한의 정의는 제한된 효과를 나타냅니다.

```
proc_taskid
        Allows a process to assign a new task ID to the calling process.
```

file_setid 권한의 정의는 광범위한 효과를 나타냅니다.

```
net_rawaccess
        Allow a process to have direct access to the network layer.
```

privileges(5) 매뉴얼 페이지는 모든 권한을 설명합니다. ppriv -lv 명령은 모든 권한의 설명을 표준 출력으로 인쇄합니다.

권한 있는 시스템의 관리상 차이점

권한이 있는 시스템과 권한이 없는 시스템 사이에는 몇 가지 눈에 띄는 차이점이 있습니다. 다음 표는 일부 차이점을 나열합니다.

표 8–2 권한 있는 시스템과 권한 없는 시스템 사이의 눈에 띄는 차이점

기능	권한 없음	권한
데몬	데몬이 root로 실행됩니다.	데몬이 사용자 daemon으로 실행됩니다. 예를 들어, 적절한 권한이 지정되었고 daemon으로 실행되는 데몬에는 lockd, nfsd, rpcbind가 있습니다.
로그 파일 소유권	root가 로그 파일을 소유합니다.	로그 파일을 만든 daemon이 로그 파일을 소유합니다. root 사용자는 파일을 소유하지 않습니다.
오류 메시지	오류 메시지가 수퍼유저를 참조합니다. 예를 들어, chroot: not superuser입니다.	오류 메시지가 권한 사용을 반영합니다. 예를 들어, chroot 실패에 해당하는 오류 메시지는 chroot: exec failed입니다.

표 8–2 권한 있는 시스템과 권한 없는 시스템 사이의 눈에 띄는 차이점 (계속)

기능	권한 없음	권한
setuid 프로그램	프로그램이 setuid를 사용하여 일반 사용자가 수행할 수 없는 작업을 완료합니다.	많은 setuid 프로그램이 권한으로 실행되도록 변경되었습니다. 예를 들어, 권한을 사용하는 명령에는 audit, ikeadm, ipadm, ipsecconf, ping, traceroute, newtask 등이 있습니다.
파일 사용 권한	장치 사용 권한을 DAC로 제어합니다. 예를 들어, sys 그룹의 구성원이 /dev/ip를 열 수 있습니다.	파일 사용 권한(DAC)이 장치를 열 수 있는 사람을 예측하지 않습니다. 장치는 DAC 및 장치 정책으로 보호됩니다. 예를 들어, /dev/ip 파일에 666 사용 권한이 있지만 적절한 권한을 가진 프로세스로만 장치를 열 수 있습니다. 원시 소켓은 여전히 DAC로 보호됩니다.
감사 이벤트	su 명령 사용의 감사에 많은 관리 기능이 관여합니다.	권한 사용의 감사에 대부분의 관리 기능이 관여합니다. pm, ps, ex, ua, as 감사 클래스는 장치 정책과 권한 사용을 모니터하는 감사 이벤트를 포함합니다.
프로세스	프로세스를 소유한 사람이 프로세스를 보호합니다.	권한으로 프로세스를 보호합니다. 프로세스 권한 및 프로세스 플래그는 /proc/<pid> 디렉토리에서 새 항목 priv로 표시됩니다.
디버깅	코어 덤프에 권한에 대한 참조가 없습니다.	코어 덤프의 ELF 노트 섹션은 NT_PRPRIV 및 NT_PRPRIVINFO 노트에 프로세스 권한 및 플래그에 대한 정보를 포함합니다. ppriv 명령 및 기타 명령은 적절히 크기 조정된 세트의 적절한 개수를 보여줍니다. 정확히 비트 세트의 비트를 권한 이름에 매핑합니다.

권한 및 시스템 리소스

Oracle Solaris 릴리스에서 project.max-locked-memory 및 zone.max-locked-memory 리소스 컨트롤을 사용하여 PRIV_PROC_LOCK_MEMORY 권한에 지정된 프로세스의 메모리 소비를 제한할 수 있습니다. 이 권한으로 프로세스가 물리적 메모리의 페이지를 잠글 수 있습니다.

PRIV_PROC_LOCK_MEMORY 권한을 권한 프로파일에 지정하면 이 권한을 가진 프로세스에 모든 메모리를 잠그는 능력을 제공할 수 있습니다. 보호 조치로, 권한 사용자가 모든 메모리를 잠그지 못하도록 리소스 컨트롤을 설정하십시오. 비전역 영역에서 실행되는 권한 있는 프로세스의 경우 zone.max-locked-memory 리소스 컨트롤을 설정합니다. 시스템에서 실행되는 권한 있는 프로세스의 경우 프로젝트를 만들고 project.max-locked-memory 리소스 컨트롤을 설정합니다. 이러한 리소스 컨트롤에 대한 내용은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 6 장, “리소스 제어(개요)” 및 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 16 장, “비전역 영역 구성(개요)”을 참조하십시오.

권한이 구현되는 방법

모든 프로세스에는 프로세스가 특정 권한을 사용할 수 있는지 여부를 결정하는 4개의 권한 세트가 있습니다. 커널이 자동으로 권한의 **유효 세트**를 계산합니다. 권한의 초기 **상속 가능한 세트**를 수정할 수 있습니다. 권한을 사용하도록 코딩된 프로그램은 권한의 **허가된 세트**를 줄일 수 있습니다. 권한의 **제한 세트**를 축소할 수 있습니다.

- **유효 권한 세트 또는 E** – 현재 발효 중인 권한 세트입니다. 프로세스는 허가된 세트에 속한 권한을 유효 세트에 추가할 수 있습니다. 또한 E에서 권한을 제거할 수도 있습니다.
- **허가된 권한 세트 또는 P** – 사용할 수 있는 권한 세트입니다. 상속 또는 지정을 통해 얻은 권한을 프로그램에 사용할 수 있습니다. 실행 프로파일은 프로그램에 권한을 지정하는 하나의 방법입니다. `setuid` 명령은 `root`가 가진 모든 권한을 프로그램에 지정합니다. 허가된 세트에서 권한을 제거할 수 있지만 세트에 권한을 추가할 수는 없습니다. P에서 제거된 권한은 자동으로 E에서 제거됩니다.

  권한 인식 프로그램은 사용하지 않는 권한을 허가된 세트에서 제거합니다. 이렇게 하면 프로그램이나 악의적 프로세스에서 불필요한 권한을 악용할 수 없습니다. 권한 인식 프로그램에 대한 자세한 내용은 **Developer's Guide to Oracle Solaris 11 Security**의 2 장, "Developing Privileged Applications"를 참조하십시오.
- **상속 가능한 권한 세트 또는 I** – `exec`의 호출에서 프로세스가 상속할 수 있는 권한 세트입니다. `setuid` 프로그램의 특수한 경우를 제외하면, `exec`의 호출 후에 허가된 세트와 유효 세트는 같습니다.

  `setuid` 프로그램의 경우 `exec`의 호출 후에 상속 가능한 세트가 먼저 제한 세트로 제약됩니다. 그런 다음, 상속된 권한 세트(I)에서 제한 세트에 속한 권한(L)을 뺀 값이 해당 프로세스의 P 및 E에 지정됩니다.
- **제한 권한 세트 또는 L** – 프로세스와 그 자식에 사용 가능한 권한에 대한 외부 제한입니다. 기본적으로 제한 세트는 모든 권한입니다. 프로세스가 제한 세트를 축소할 수 있지만 제한 세트를 확장할 수는 없습니다. L은 I를 제한하는 데 사용됩니다. 결과적으로, L은 `exec` 시간에 P 및 E를 제한합니다.

  사용자가 권한 지정 프로그램을 포함하는 프로파일에 지정된 경우 대개 해당 프로그램을 실행할 수 있습니다. 수정되지 않은 시스템에서 프로그램의 지정된 권한은 사용자의 제한 세트 내에 있습니다. 프로그램에 지정된 권한은 사용자의 허가된 세트의 일부가 됩니다. 권한이 지정된 프로그램을 실행하려면 사용자가 프로파일 셸에서 프로그램을 실행해야 합니다.

커널은 **기본 권한 세트**를 인식합니다. 수정되지 않은 시스템에서 각 사용자의 초기 상속 가능한 세트는 로그인 시 기본 세트와 같습니다. 기본 세트를 수정할 수 없는 반면, 사용자가 기본 세트에서 상속한 권한은 수정할 수 있습니다.

수정되지 않은 시스템에서 로그인 시 사용자의 권한 세트는 다음과 비슷합니다.

```
E (Effective): basic
I (Inheritable): basic
```

```
P (Permitted): basic
L (Limit): all
```

따라서 로그인 시 모든 사용자는 상속 가능한 세트, 허가된 세트, 유효 세트에 기본 세트를 포함합니다. 사용자의 제한 세트는 (전역 또는 비전역) 영역의 기본 제한 세트와 같습니다. 사용자의 유효 세트에 더 많은 권한을 넣으려면 권한 프로파일을 사용자에 지정해야 합니다. 권한 프로파일은 사용자가 권한을 추가한 명령을 포함합니다. 또한 위험을 동반하더라도 사용자나 역할에 직접 권한을 지정할 수도 있습니다. 위험 설명은 145 페이지 “보안 속성을 직접 지정할 때 보안 고려 사항”을 참조하십시오.

프로세스가 권한을 얻는 방법

프로세스는 권한을 상속할 수 있습니다. 또는 프로세스가 권한에 지정될 수 있습니다. 프로세스는 부모 프로세스에서 권한을 상속합니다. 로그인 시, 사용자의 초기 상속 가능한 권한 세트는 사용자의 프로세스에 사용 가능한 권한을 결정합니다. 사용자 초기 로그인의 모든 자식 프로세스는 해당 세트를 상속합니다.

프로그램, 사용자, 역할에 직접 권한을 지정할 수도 있습니다. 프로그램에 권한이 필요할 때 권한 프로파일에서 프로그램의 실행 파일에 권한을 지정합니다. 프로그램을 실행하도록 허가된 사용자나 역할이 프로그램을 포함하는 프로파일에 지정됩니다. 로그인 시 또는 프로파일 셸을 입력할 때, 프로그램의 실행 파일을 프로파일 셸에 입력하면 프로그램이 권한으로 실행됩니다. 예를 들어, Object Access Management 프로파일을 포함하는 역할은 `chmod` 명령을 `file_chown` 권한으로 실행할 수 있습니다.

역할/사용자가 추가 권한이 직접 지정된 프로그램을 실행할 때 지정된 권한이 역할/사용자의 상속 가능한 세트에 추가됩니다. 권한이 지정된 프로그램의 자식 프로세스는 부모의 권한을 상속합니다. 자식 프로세스에 부모 프로세스보다 더 많은 권한이 필요한 경우 자식 프로세스에 해당 권한을 직접 지정해야 합니다.

권한을 사용하도록 코딩된 프로그램을 권한 인식 프로그램이라고 합니다. **권한 인식** 프로그램은 프로그램 실행 중 권한 사용을 켜고 끕니다. 운용 환경에서 성공하려면 프로그램이 켜고 끄는 권한을 지정해야 합니다.

권한 인식 코드의 예는 **Developer’s Guide to Oracle Solaris 11 Security**의 2 장, “Developing Privileged Applications”를 참조하십시오. 권한이 필요한 프로그램에 권한을 지정하려면 예 9–14를 참조하십시오.

권한 지정

보안 관리자의 자격으로 권한을 지정할 책임이 있습니다. 최적의 사용법은 권한 프로파일에서 명령에 권한을 지정하는 것입니다. 그런 다음 권한 프로파일을 역할이나 사용자에 지정합니다.

사용자, 역할, 권한 프로파일에 직접 권한을 지정할 수도 있습니다. 세션 동안 책임감 있게 권한을 사용할 것으로 신뢰되는 일부 사용자에게는 권한을 직접 지정할 수 있습니다. 직접 지정의 좋은 후보는 proc_clock_highres와 같은 제한된 영향을 미치는 권한입니다. 직접 지정의 나쁜 후보는 file_dac_write와 같은 폭넓은 영향을 미치는 권한입니다.

사용자나 시스템에 권한이 거부될 수도 있습니다. 사용자나 시스템의 초기 상속 가능한 세트 또는 제한 세트에서 권한을 제거할 때는 주의를 기울여야 합니다.

사용자 또는 역할의 권한 확장

사용자 및 역할은 상속 가능한 권한 세트를 갖습니다. 제한 세트는 초기에 모든 권한이므로 확장할 수 없습니다. 사용자, 역할, 시스템에 대한 초기 상속 가능한 세트를 확장할 수 있습니다. 상속 가능한 세트에 속하지 않는 권한을 프로세스에 지정할 수도 있습니다.

두 가지 방법으로 사용 가능한 권한을 확장할 수 있습니다.

- 사용자, 역할, 시스템에 대한 초기 상속 가능한 세트를 확장할 수 있습니다.
- 상속 가능한 세트에 속하지 않는 권한을 프로세스에 명시적으로 지정할 수도 있습니다.

프로세스별 권한 지정이 가장 정확한 권한 추가 방법입니다. 사용자에 역할을 지정하여 사용자가 수행할 수 있는 권한 있는 작업의 수를 확장할 수 있습니다. 역할은 추가된 권한을 가진 명령을 포함하는 권한 프로파일에 지정됩니다. 사용자가 역할을 맡으면 역할의 프로파일 셸을 얻습니다. 권한 프로파일의 명령을 역할의 셸에 입력할 때 추가된 권한으로 명령이 실행됩니다.

사용자가 맡은 역할이 아닌, 사용자에게 권한 프로파일을 지정할 수도 있습니다. 사용자가 pfksh와 같은 프로파일 셸을 열면 사용자의 권한 프로파일의 명령을 권한으로 실행할 수 있습니다. 일반 셸에서 명령은 권한으로 실행되지 않습니다. 권한 있는 프로세스는 권한 있는 셸에서만 실행할 수 있습니다.

사용자, 역할, 시스템에 대한 초기 상속 가능한 권한 세트를 확장하는 것은 위험한 권한 지정 방법입니다. 상속 가능한 세트의 모든 권한은 허가된 세트와 유효 세트에 속합니다. 사용자나 역할이 셸에 입력하는 모든 명령은 직접 지정된 권한을 사용할 수 있습니다. 직접 지정된 권한을 통해 사용자/역할의 관리 책임의 한도를 벗어난 작업을 쉽게 수행할 수 있습니다.

시스템에서 초기 상속 가능한 권한 세트에 추가할 때 시스템에 로그온한 모든 사용자는 더 큰 기본 권한 세트를 받습니다. 이러한 직접 지정으로 시스템의 모든 사용자는 일반 사용자의 한도를 벗어난 작업을 쉽게 수행할 수 있습니다.

주 – 제한 세트는 초기에 모든 권한이므로 확장할 수 없습니다.

사용자 또는 역할의 권한 제한

권한을 제거하여 사용자나 역할이 특정 작업을 수행하는 것을 막을 수 있습니다. 초기 상속 가능한 세트 및 제한 세트에서 권한을 제거할 수 있습니다. 초기 상속 가능한 세트 또는 제한 세트가 기본 세트보다 작은 경우 세트를 분배하기 전에 권한 제거를 주의 깊게 테스트해야 합니다. 초기 상속 가능한 세트에서 권한을 제거하면 사용자의 로그인을 막을 수 있습니다. 제한 세트에서 권한을 제거할 때 기존의 `setuid` 프로그램에 제거된 권한이 필요한 경우 프로그램을 실패할 수 있습니다.

스크립트에 권한 지정

스크립트는 명령과 비슷한 실행 파일입니다. 따라서 권한 프로파일에서 명령에 권한을 추가하듯이 스크립트에 권한을 추가할 수 있습니다. 권한 프로파일에 지정된 사용자/역할이 프로파일 셸에서 스크립트를 실행할 때 추가된 명령으로 스크립트가 실행됩니다. 스크립트에 권한이 필요한 명령이 포함된 경우 추가된 권한을 가진 명령 역시 지정된 권한 프로파일에 있어야 합니다.

권한 인식 프로그램은 프로세스별 권한을 제한할 수 있습니다. 권한 인식 프로그램의 임무는 프로그램에 필요한 권한만 실행 파일에 지정하는 것입니다. 그런 다음 프로그램을 테스트하여 프로그램이 작업 수행을 성공하는지 확인합니다. 또한 프로그램이 권한 사용을 오용하지 않는지 검사합니다.

권한 및 장치

권한 모델은 권한을 사용하여 시스템 인터페이스를 보호하며, 수퍼유저 모델은 파일 사용 권한만으로 보호합니다. 권한 있는 시스템에서 파일 사용 권한은 인터페이스를 보호하기에 너무 약합니다. `proc_owner`와 같은 권한은 파일 사용 권한을 대체하고 시스템 전체에 대한 전체 액세스 권한을 부여할 수 있습니다.

따라서 Oracle Solaris에서 장치 디렉토리의 소유권은 장치를 열기에 충분하지 않습니다. 예를 들어 `sys` 그룹의 구성원은 더 이상 자동으로 `/dev/ip` 장치를 열도록 허용되지 않습니다. `/dev/ip`의 파일 사용 권한은 `0666`이지만, 장치를 열려면 `net_rawaccess` 권한이 필요합니다.

장치 정책은 권한으로 제어합니다. `getdevpolicy` 명령은 모든 장치에 대한 장치 정책을 표시합니다. 장치 구성 명령 `devfsadm`은 장치 정책을 설치합니다. `devfsadm` 명령은 장치 읽기/쓰기를 위해 권한 세트를 `open`과 바인드합니다. 자세한 내용은 `getdevpolicy(1M)` 및 `devfsadm(1M)` 매뉴얼 페이지를 참조하십시오.

장치 정책을 통해 장치 열기 권한 부여에 유연성이 확대됩니다. 기본 장치 정책과 비교해 다른 권한이나 많은 권한이 필요할 수 있습니다. 장치 정책 및 드라이버에 대한 권한 요구 사항을 적절히 수정할 수 있습니다. 장치 드라이버를 설치, 추가, 업데이트할 때 권한을 수정할 수 있습니다.

`add_drv` 및 `update_drv` 명령을 사용하여 장치 정책 항목 및 장치 특정 권한을 수정할 수 있습니다. 장치 정책을 변경하려면 전체 권한 세트를 보유한 프로세스를 실행 중이어야 합니다. 자세한 내용은 `add_drv(1M)` 및 `update_drv(1M)` 매뉴얼 페이지를 참조하십시오.

권한 및 디버깅

Oracle Solaris는 권한 실패를 디버깅하는 도구를 제공합니다. `ppriv` 명령과 `truss` 명령은 디버깅 출력을 제공합니다. 예제는 `ppriv(1)`매뉴얼 페이지를 참조하십시오. 절차는 192 페이지 “프로그램에 필요한 권한을 확인하는 방법”을 참조하십시오. 또한 `dtrace` 명령을 사용할 수 있습니다. 자세한 내용은 `dtrace(1M)` 매뉴얼 페이지를 참조하십시오.

◆ ◆ ◆ 9 장

9 역할 기반 액세스 제어 사용(작업)

이 장에서는 별개의 역할을 사용하여 수퍼유저 기능을 분배하기 위한 작업을 다룹니다. 역할이 사용할 수 있는 방식에는 권한 프로파일, 권한 부여 및 권한이 포함됩니다. 다음은 이 장에 포함된 작업 맵 목록입니다.

- 155 페이지 "RBAC 사용(작업)"
- 186 페이지 "권한 사용(작업)"

RBAC 개요는 135 페이지 "역할 기반 액세스 제어(개요)"를 참조하십시오. 참고 사항은 10 장, "Oracle Solaris의 보안 속성(참조)"을 참조하십시오. 권한을 사용하려면 186 페이지 "권한 사용(작업)"을 참조하십시오.

RBAC 사용(작업)

RBAC를 사용하려면 계획 및 RBAC 구성이 필요하고 역할을 맡는 방법을 알아야 합니다. 역할에 익숙해지면 RBAC를 사용자 정의하여 새로운 작업을 처리할 수 있습니다. 다음 작업 맵은 권한 사용을 포함한 주요 작업을 가리킵니다.

작업	설명	수행 방법
기본 RBAC 구성을 사용합니다.	초기 설치를 수정하지 않고 RBAC를 보고 사용합니다.	156 페이지 "RBAC 기본값 보기 및 사용(작업 맵)"
RBAC를 계획, 구성, 사용합니다.	사이트에 대해 RBAC를 구성합니다.	163 페이지 "RBAC 초기 구성(작업 맵)"
RBAC를 관리합니다.	사이트의 RBAC 구성을 업데이트합니다.	176 페이지 "RBAC 관리(작업 맵)"
권한을 관리하고 사용합니다.	사용자, 역할, 시스템, 프로세스에서 권한을 추가 및 제거합니다. 권한을 사용합니다. 권한 사용을 보고 디버그합니다.	186 페이지 "권한 사용(작업)"

RBAC 기본값 보기 및 사용(작업)

사용자는 기본적으로 권한에 할당됩니다. 시스템의 모든 사용자에 대한 권한은 /etc/security/policy.conf 파일에서 할당됩니다.

RBAC 기본값 보기 및 사용(작업 맵)

Oracle Solaris 설치에서 시스템은 사용자 권한과 프로세스 권한으로 구성됩니다. 더 이상의 구성 없이 RBAC를 보고 사용하려면 다음 작업 맵을 사용하십시오.

작업	설명	수행 방법
보안 속성 데이터베이스의 내용을 봅니다.	시스템의 모든 권한 부여, 권한 프로파일 및 보안 속성 포함 명령을 나열합니다.	156 페이지 "모든 정의된 보안 속성을 보는 방법"
권한을 봅니다.	권한 프로파일, 권한 부여, 권한 및 할당된 역할을 나열합니다.	157 페이지 "할당된 권한을 보는 방법"
root 역할을 맡습니다.	초기 사용자는 관리 권한을 얻습니다.	159 페이지 "역할을 맡는 방법"
관리자로 전환합니다.	관리 권한에 할당된 사용자에게 이러한 권한을 사용하기 위한 여러 방법을 제공할 수 있습니다.	160 페이지 "관리 권한을 얻는 방법"

▼ 모든 정의된 보안 속성을 보는 방법

다음 명령을 사용하여 시스템의 모든 권한 부여, 권한 프로파일 및 보안 속성 포함 명령을 나열합니다. 모든 정의된 권한을 나열하려면 187 페이지 "시스템의 권한을 나열하는 방법"을 참조하십시오.

1 모든 권한 부여를 나열합니다.

```
% getent auth_attr | more
solaris.:::All Solaris Authorizations::help=AllSolAuthsHeader.html
solaris.account.:::Account Management::help=AccountHeader.html
...
solaris.zone.login:::Zone Login::help=ZoneLogin.html
solaris.zone.manage:::Zone Deployment::help=ZoneManage.html
```

2 모든 권한 프로파일을 나열합니다.

```
% getent prof_attr | more
All:::Execute any command as the user or role:help=RtAll.html
Audit Configuration:::Configure Solaris Audit:auths=solaris.smf.value.audit;
help=RtAuditCfg.html
...
Zone Management:::Zones Virtual Application Environment Administration:
help=RtZoneMngmnt.html
Zone Security:::Zones Virtual Application Environment Security:auths=solaris.zone.*,
solaris.auth.delegate;help=RtZoneSecurity.html ...
```

3 **모든 보안 속성 포함 명령을 나열합니다.**

```
% getent exec_attr | more
All:solaris:cmd:::*:
Audit Configuration:solaris:cmd:::/usr/sbin/auditconfig:privs=sys_audit
...
Zone Security:solaris:cmd:::/usr/sbin/txzonemgr:uid=0
Zone Security:solaris:cmd:::/usr/sbin/zonecfg:uid=0 ...
```

▼ 할당된 권한을 보는 방법

다음 명령을 사용하여 RBAC 할당 사항을 봅니다. 모든 할당 가능한 권한을 보려면 156 페이지 "모든 정의된 보안 속성을 보는 방법"을 참조하십시오.

1 **내 권한 부여를 나열합니다.**

```
% auths
solaris.device.cdrw,solaris.device.mount.removable,solaris.mail.mailq
```

이러한 권한 부여는 기본적으로 모든 사용자에 할당됩니다.

2 **내 권한 프로파일을 나열합니다.**

```
% profiles
Basic Solaris User
All
```

이러한 권한 프로파일은 기본적으로 모든 사용자에 할당됩니다.

3 **내 할당된 역할을 나열합니다.**

```
% roles
root
```

이 역할은 기본적으로 초기 사용자에 할당됩니다. `No roles`는 역할이 할당되지 않음을 나타냅니다.

4 **기본 셸의 권한을 나열합니다.**

```
% ppriv $$
1234:    /bin/csh
flags = <none>
    E: basic
    I: basic
    P: basic
    L: all
```

모든 사용자는 기본적으로 기본 권한 세트에 할당됩니다. 제한 세트는 모든 권한입니다.

```
% ppriv -vl basic
file_link_any
        Allows a process to create hardlinks to files owned by a uid
        different from the process' effective uid.
file_read
        Allows a process to read objects in the filesystem.
```

```
file_write
        Allows a process to modify objects in the filesystem.
net_access
        Allows a process to open a TCP, UDP, SDP or SCTP network endpoint.
proc_exec
        Allows a process to call execve().
proc_fork
        Allows a process to call fork1()/forkall()/vfork()
proc_info
        Allows a process to examine the status of processes other
        than those it can send signals to.  Processes which cannot
        be examined cannot be seen in /proc and appear not to exist.
proc_session
        Allows a process to send signals or trace processes outside its session.
```

5 권한 프로파일의 명령에 대한 권한을 나열합니다.

```
% profiles -l
  Basic Solaris User
   /usr/bin/cdda2wav.bin   privs=file_dac_read,sys_devices,
     proc_priocntl,net_privaddr
   /usr/bin/cdrecord.bin   privs=file_dac_read,sys_devices,
     proc_lock_memory,proc_priocntl,net_privaddr
   /usr/bin/readcd.bin     privs=file_dac_read,sys_devices,net_privaddr
  All
   *
```

사용자의 권한 프로파일에는 특정 권한으로 실행되는 명령이 포함될 수 있습니다. Basic Solaris User 프로파일은 사용자가 CD-ROM을 읽고 쓸 수 있는 명령을 포함합니다.

예 9–1 사용자의 권한 부여 나열

```
% auths username
solaris.device.cdrw,solaris.device.mount.removable,solaris.mail.mailq
```

예 9–2 사용자 또는 역할의 권한 프로파일 나열

다음 명령은 특정 사용자의 권한 프로파일을 나열합니다.

```
% profiles jdoe
jdoe:
          Basic Solaris User
          All
```

다음 명령은 `cryptomgt` 역할의 권한 프로파일을 나열합니다.

```
% profiles cryptomgt
cryptomgt:
          Crypto Management
          Basic Solaris User
          All
```

다음 명령은 `root` 역할의 권한 프로파일을 나열합니다.

```
% profiles root
root:
          All
          Console User
          Network Wifi Info
          Desktop Removable Media User
          Suspend To RAM
          Suspend To Disk
          Brightness
          CPU Power Management
          Network Autoconf User
          Basic Solaris User
```

예 9–3 사용자의 할당된 역할 나열

다음 명령은 특정 사용자의 할당된 역할을 나열합니다.

```
% roles jdoe
root
```

예 9–4 특정 명령에 대한 사용자의 권한 나열

다음 명령은 일반 사용자의 권한 프로파일에서 권한 있는 명령을 나열합니다.

```
% profiles -l jdoe
jdoe:
  Basic Solaris User
   /usr/bin/cdda2wav.bin   privs=file_dac_read,sys_devices,
     proc_priocntl,net_privaddr
   /usr/bin/cdrecord.bin   privs=file_dac_read,sys_devices,
     proc_lock_memory,proc_priocntl,net_privaddr
   /usr/bin/readcd.bin     privs=file_dac_read,sys_devices,net_privaddr
  All
   *
```

▼ 역할을 맡는 방법

시작하기 전에 역할에 이미 할당되어야 합니다. 이름 지정 서비스를 해당 정보로 업데이트해야 합니다.

1 터미널 창에서 맡을 수 있는 역할을 확인합니다.

```
% roles
Comma-separated list of role names is displayed
```

2 su 명령을 사용하여 역할을 맡습니다.

```
% su - rolename
Password:       <Type rolename password>
$
```

`su - rolename` 명령은 해당 셸을 역할의 프로파일 셸로 변경합니다. 프로파일 셸은 권한 부여, 권한, 세트 ID 비트와 같은 보안 속성을 인식합니다.

3 **(옵션) 지금 역할에 속해 있는지 확인합니다.**

```
$ /usr/bin/whoami
rolename
```

이제 이 터미널 창에서 역할 작업을 수행할 수 있습니다.

4 **(옵션) 역할 기능을 봅니다.**

절차는 157 페이지 "할당된 권한을 보는 방법"을 참조하십시오.

예 9–5 root 역할 맡기

다음 예에서 초기 사용자가 root 역할을 맡고 역할의 셸에 권한을 나열합니다.

```
% roles
root
% su - root
Password:       <Type root password>
#       Prompt changes to root prompt
# ppriv $$
1200:   pfksh
flags = <none>
        E: all
        I: basic
        P: all
        L: all
```

권한에 대한 내용은 146 페이지 "권한(개요)"을 참조하십시오.

▼ 관리 권한을 얻는 방법

관리 권한은 프로파일 셸을 실행 중일 때 적용됩니다. 기본적으로 역할 계정은 프로파일 셸에 할당됩니다. 역할은 특정 관리 작업에 할당된 특수한 계정으로, 일반적으로 감사 파일 검토와 같은 관리 활동에 관련됩니다.

root 역할에서 초기 사용자는 모든 관리 권한을 가진 수퍼유저입니다. root 역할로 다른 역할을 만들 수 있습니다.

시작하기 전에 시스템을 관리하려면 일반 사용자에 할당되지 않은 권한이 있어야 합니다. 수퍼유저가 아닌 경우 역할, 관리 권한 프로파일 또는 특정 권한이나 권한 부여에 할당되어야 합니다.

- **다음 방법 중 하나를 선택하여 관리 명령을 실행합니다.**
  터미널 창을 엽니다.

  - **root가 됩니다.**

    ```
    % su -
    Password:       Type the root password
    #
    ```

    주 – 이 방법은 root가 사용자인지 아니면 역할인지 여부에 상관없이 작동합니다. 파운드 기호(#) 프롬프트는 지금 수퍼유저임을 나타냅니다.

  - **할당된 역할을 맡습니다.**
    다음 예에서 네트워크 관리 역할을 맡습니다. 이 역할에는 Network Management 권한 프로파일이 포함됩니다.

    ```
    % su - networkadmin
    Password:       Type the networkadmin password
    $
    ```

    지금 프로파일 셸에 있습니다. 이 셸에서 snoop, route, dladm 및 기타 명령을 실행할 수 있습니다. 프로파일 셸에 대한 자세한 내용은 145 페이지 “프로파일 셸 및 RBAC”를 참조하십시오.

    참고 – 역할 기능을 보려면 157 페이지 “할당된 권한을 보는 방법”의 단계를 사용하십시오.

  - **pfbash 명령을 사용하여 관리 권한으로 실행되는 셸을 만듭니다.**
    예를 들어, 다음 명령 세트는 pfbash 셸의 네트워크 패킷을 조사할 수 있습니다.

    ```
    % pfbash
    $ anoop
    ```

    net_observability 권한이 할당되지 않은 경우 snoop: cannot open "net0": Permission denied와 비슷한 오류 메시지와 함께 snoop 명령을 실패합니다. 권한이 직접 또는 권한 프로파일이나 역할을 통해 할당된 경우 이 명령을 성공합니다. 또한 이 셸에서 추가로 권한 있는 명령을 실행할 수 있습니다.

  - **pfexec 명령을 사용하여 관리 권한으로 실행되는 프로세스를 만듭니다.**
    권한 프로파일에서 권한 있는 명령 이름으로 pfexec 명령을 실행합니다. 예를 들어, 다음 명령은 네트워크 패킷을 조사할 수 있습니다.

    ```
    % pfexec snoop
    ```

    pfbash와 동일한 권한 제한이 pfexec에 적용됩니다. 그러나 다른 권한 있는 명령을 실행하려면 권한 있는 명령을 입력하기 전에 pfexec를 다시 입력해야 합니다.

예 9–6 역할 사용의 편의성을 위해 인증 캐싱

이 예에서 관리자가 네트워크를 관리하는 역할을 구성하되, 사용자의 인증을 캐싱하여 사용 편의성을 제공합니다. 먼저, 관리자가 역할을 만들고 할당합니다.

```
# roleadd -K roleauth=user -P "Network Management" netmgt
# usermod -R +netmgt jdoe
```

jdoe가 역할로 전환할 때 -c 옵션을 사용하는 경우 snoop 출력 표시에 앞서 암호를 요구합니다.

```
% su - netmgt -c snoop options
Password:

     snoop output
```

인증이 캐싱되지 않는 경우 jdoe가 즉시 명령을 다시 실행하면 암호 프롬프트가 나타납니다.

관리자가 인증을 캐싱하도록 pam.conf 파일을 구성합니다. 그러면 초기에 암호가 필요하지만, 그 후에 특정 시간이 지날 때까지 암호를 요구하지 않습니다. 관리자가 모든 사용자 정의된 pam.conf 스택을 파일 끝에 놓습니다.

```
# vi /etc/pam.conf
...
#
## Cache authentication for switched user
#
su      auth required           pam_unix_cred.so.1
su      auth sufficient         pam_tty_tickets.so.1
su      auth requisite          pam_authtok_get.so.1
su      auth required           pam_dhkeys.so.1
su      auth required           pam_unix_auth.so.1
```

입력한 후에 관리자가 입력에 오타, 누락, 반복이 있는지 검사합니다.

전체 su 스택이 필요합니다. pam_tty_tickets.so.1 모듈은 캐시를 제공합니다. PAM에 대한 자세한 내용은 pam.conf(4) 매뉴얼 페이지와 15 장, “PAM 사용”을 참조하십시오.

su PAM 스택을 pam.conf 파일에 추가한 후에는 netmgt 역할이 일련의 명령을 실행할 때 한번만 암호를 묻습니다.

```
% su - netmgt -c snoop options
Password:

     snoop output
% su - netmgt -c snoop options
     snoop output
...
```

사이트에 대해 RBAC 사용자 정의(작업)

RBAC의 초기 구성에는 특정 역할을 맡을 수 있는 사용자를 만들고, 역할을 만들고, 적절한 사용자에 할당하는 과정이 포함됩니다.

RBAC 초기 구성(작업 맵)

다음 작업 맵을 사용하여 사이트에서 RBAC를 계획하고 초기에 구현합니다. 일부 작업은 순서가 있습니다.

작업	설명	수행 방법
1. RBAC를 계획합니다.	사이트의 보안 요구 사항을 조사하고 사이트에서 RBAC 사용 방법을 결정합니다.	163 페이지 "RBAC 구현을 계획하는 방법"
2. 역할을 맡을 수 있는 사용자를 구성합니다.	관리 역할을 맡을 수 있는 사용자가 존재하는지 확인합니다.	**Oracle Solaris 관리: 일반 작업**의 "사용자 계정 설정 및 관리(작업 맵)"
3. 역할을 만듭니다.	역할을 만들어서 사용자에 할당합니다.	165 페이지 "역할을 만드는 방법" 167 페이지 "역할을 할당하는 방법"
(권장) 역할 동작을 감사합니다.	역할 동작을 기록하는 감사 이벤트가 포함된 감사 클래스를 미리 선택합니다.	169 페이지 "역할을 감사하는 방법"
권한 프로파일을 만들거나 변경합니다.	권한 프로파일을 만듭니다. 또는 권한 프로파일에서 보안 속성이나 보충 권한 프로파일을 수정합니다. 명령에 권한을 추가합니다.	170 페이지 "감사 프로파일을 만들거나 변경하는 방법" 예 9–14
레거시 응용 프로그램을 보안합니다.	레거시 응용 프로그램에 대해 세트 ID 사용 권한을 설정합니다. 스크립트가 세트 ID를 가진 명령을 포함할 수 있습니다. 레거시 응용 프로그램이 적절히 권한 부여를 검사할 수 있습니다.	171 페이지 "RBAC 등록 정보를 레거시 응용 프로그램에 추가하는 방법" 예 9–16
보안 속성 할당 문제를 해결합니다.	할당된 보안 속성을 사용자, 역할, 프로세스에 사용할 수 없는지 디버그합니다.	173 페이지 "RBAC 및 권한 할당 문제를 해결하는 방법"

▼ RBAC 구현을 계획하는 방법

RBAC는 조직이 정보 리소스를 관리하는 데 필수적인 부분입니다. RBAC 기능에 대한 지식과 조직의 보안 요구 사항을 바탕으로 계획을 수립해야 합니다.

주 – 기본 권한은 `/etc/security/policy.conf` 파일에서 할당됩니다.

1 **기본 RBAC 개념을 익힙니다.**

135 페이지 "역할 기반 액세스 제어(개요)"를 읽어 보십시오. RBAC를 사용한 시스템 관리는 전통적인 UNIX 관리 방법과 매우 다릅니다. 구현을 시작하기 전에 RBAC 개념을 익히려면 10 장, "Oracle Solaris의 보안 속성(참조)"을 참조하십시오.

2 **보안 정책을 조사합니다.**

조직의 보안 정책은 시스템의 잠재적 위협을 기술하고, 각 위협의 위험도를 측정하고, 이러한 위협에 맞서는 전략을 제시합니다. RBAC를 통해 보안 관련 작업을 고립시키는 것도 전략의 일부일 수 있습니다. 설치된 RBAC 구성을 있는 그대로 사용할 수도 있지만, 보안 정책에 따라 사용자 정의가 필요할 수 있습니다.

3 **조직이 요구하는 RBAC 수준을 결정합니다.**

보안 요구 사항에 따라, 다음과 같이 다양한 RBAC 수준을 사용할 수 있습니다.

- **루트 역할 –** 이 방법이 기본적으로 제공됩니다. 임의 사용자가 `root`로 로그인하지 못하게 합니다. 대신, 사용자는 `root` 역할을 맡기 전에 할당된 로그인을 사용하여 로그인해야 합니다.
- **별개의 역할 –** 이 방법은 제공된 권한 프로파일을 기반으로 역할을 만듭니다. 책임 수준, 작업 범위, 작업 유형에 따라 역할을 할당할 수 있습니다. 예를 들어, System Administrator 역할은 수퍼유저가 수행할 수 있는 많은 작업을 수행할 수 있고, Network IPsec Management 역할은 IPsec을 관리할 수 있습니다.

  또한 보안 책임을 다른 책임과 구분할 수 있습니다. User Management 역할은 사용자를 만들 수 있고, User Security 역할은 역할 및 권한 프로파일과 같은 보안 속성을 할당할 수 있습니다. 그러나 User Security 역할은 사용자를 만들 수 없고, User Management 역할은 권한 프로파일을 사용자에 할당할 수 없습니다.
- **루트 역할 없음 –** 이 방법은 시스템의 기본 구성을 변경해야 합니다. 이 구성에서는 `root`의 암호를 아는 모든 사용자가 시스템에 로그인하여 수정할 수 있습니다. 어떤 사용자가 수퍼유저로 활동하는지 알 수 없습니다.

4 **어떤 역할이 조직에 적절한지 결정합니다.**

권장된 역할 기능과 기본 권한 프로파일을 검토하십시오. 기본 권한 프로파일을 통해 관리자가 단일 프로파일을 사용하여 권장된 역할을 구성할 수 있습니다.

권한 프로파일을 더 조사하려면 다음 중 하나를 수행합니다.

- 시스템에서 사용 가능한 권한 프로파일은 `getent prof_attr` 명령을 사용하십시오.
- 이 설명서에서 일반적인 권한 프로파일의 요약은 197 페이지 "권한 프로파일"을 참조하십시오.

5 **추가적 역할이나 권한 프로파일이 조직에 적절한지 결정합니다.**

사이트에서 제한된 액세스를 이용할 수 있는 다른 응용 프로그램이나 응용 프로그램 제품군을 찾아보십시오. 서비스 거부 문제를 일으킬 수 있는 보안에 민감한 응용 프로그램이나 특수한 관리자 교육이 필요한 응용 프로그램이 RBAC의 좋은 후보입니다. 조직의 보안 요구 사항을 처리하기 위해 역할 및 권한 프로파일을 사용자 정의할 수 있습니다.

a. **어떤 명령이 새 작업에 필요한지 확인합니다.**

b. **이 작업에 어떤 권한 프로파일이 적절한지 결정합니다.**

기존 권한 프로파일이 이 작업을 처리할 수 있는지, 또는 별도의 권한 프로파일을 만들어야 하는지 확인합니다.

주 – Media Backup 및 Media Restore 권한 프로파일은 전체 루트 파일 시스템에 액세스할 수 있습니다. 따라서 이러한 권한 프로파일은 신뢰된 사용자에게만 적절히 할당됩니다. 다른 방법으로, 이러한 권한 프로파일을 할당하지 않도록 선택할 수 있습니다. 기본적으로 `root` 역할만 백업 및 보안을 위해 신뢰할 수 있습니다.

---

c. **이 권한 프로파일에 어떤 역할이 적절한지 확인합니다.**

이 작업에 대한 권한 프로파일을 기존 역할에 할당해야 하는지, 또는 새 역할을 만들어야 하는지 결정합니다. 기존 역할을 사용하는 경우 이 역할에 할당된 사용자에게 원래 권한 프로파일이 적절한지 확인합니다. 새 권한 프로파일의 순서를 지정하여 필요한 권한으로 명령이 실행되도록 합니다. 순서 지정에 대한 내용은 199 페이지 “지정된 보안 속성의 검색 순서”를 참조하십시오.

6 **어떤 사용자가 어떤 역할에 할당되어야 하는지 결정합니다.**

최소한의 특권의 원칙에 따라 사용자의 신뢰 수준에 적합한 역할을 할당합니다. 사용자가 수행할 필요가 없는 작업을 실행하지 못하게 금지하면 잠재적 문제를 줄일 수 있습니다.

▼ 역할을 만드는 방법

로컬에서 LDAP 저장소에 역할을 만들 수 있습니다.

시작하기 전에 역할을 만들고 초기 암호를 할당하려면 User Management 권한 프로파일에 할당되어야 합니다. 보안 속성을 역할에 할당하려면 User Security 권한 프로파일에 할당되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 역할을 만들려면 `roleadd` 명령을 사용합니다.

명령의 RBAC 인수는 다음과 같습니다.

```
# roleadd [-e expire] [-f inactive] [-s shell] [-m] [-S repository] \
[-A authorization-list] -K key=value] rolename
```

`-e` *expire* — 역할이 만료되는 날짜입니다. 임시 역할을 만들려면 이 옵션을 사용하십시오.

`-f` *inactive* — 역할 사용 사이에 허용된 최대 기간(일)입니다. *inactive* 값을 초과하면 역할을 사용할 수 없습니다. 기본값은 `0`이며, 만료 날짜가 없습니다.

`-m` — 기본 위치에 *rolename*의 홈 디렉토리를 만듭니다.

`-s` *shell* — *rolename*의 로그인 셸입니다. 이 셸은 프로파일 셸이어야 합니다. 프로파일 셸 목록은 `pfexec(1)` 매뉴얼 페이지를 참조하십시오.

참고 – 시스템의 `/usr/bin` 디렉토리에서 프로파일 셸을 나열할 수도 있습니다(예: `ls /usr/bin/pf*sh`).

`-S` *repository* — `files` 또는 `ldap` 중 하나입니다. 기본값은 로컬 파일입니다.

`-A` *authorization-list* — 콤마로 구분된 하나 이상의 권한 부여입니다. 권한 부여 목록은 `/etc/security/auth_attr` 파일을 참조하십시오.

`-K` *key=value* — **키=값** 쌍입니다. 이 옵션은 반복할 수 있습니다. 사용 가능한 키에는 `audit_flags`, `auths`, `profiles`, `project`, `defaultpriv`, `limitpriv`, `lock_after_retries`, `roleauth` 등이 있습니다. 값 설정에 필요한 키, 해당 값 및 권한 부여에 대한 내용은 `user_attr(4)` 매뉴얼 페이지를 참조하십시오.

rolename — 새 역할의 이름입니다. 허용 가능한 문자열의 제한 사항은 `roleadd(1M)` 매뉴얼 페이지를 참조하십시오.

참고 – 역할 이름이 권한 프로파일의 이름을 반영할 때 역할의 목적을 쉽게 이해할 수 있습니다. 예를 들어, Audit Review 권한 프로파일을 `auditreview` 역할에 할당하여 감사 레코드를 읽기, 필터링, 아카이브할 수 있도록 합니다.

예를 들어, 다음 명령은 로컬 User Administrator 역할과 홈 디렉토리를 만듭니다.

```
# roleadd -c "User Administrator role, local" -s /usr/bin/pfbash \
-m -K profiles="User Security,User Management"  useradm
80 blocks
# ls /export/home/useradm
local.cshrc     local.login     local.profile
```

3 **역할의 초기 암호를 만듭니다.**

```
# passwd -r files useradmPassword:        <Type useradm password>
Confirm Password:       <Retype useradm password>
#
```

주 – 일반적으로, 역할 계정은 여러 사용자에 할당됩니다. 따라서 관리자는 대개 역할 암호를 만들어서 대역 외에서 사용자에게 역할 암호를 알려줍니다.

---

4 **역할을 사용자에 할당하려면 usermod 명령을 실행합니다.**

절차는 167 페이지 "역할을 할당하는 방법" 및 예 9–10을 참조하십시오.

예 9–7 LDAP 저장소에 User Administrator 역할 만들기

이 예에서 관리자의 사이트가 LDAP 저장소를 사용합니다. 다음 명령을 실행하여 LDAP에 User Administrator 역할을 만듭니다.

```
# roleadd -c "User Administrator role, LDAP" -s /usr/bin/pfbash \
-m -S ldap -K profiles="User Security,User Management"  useradm
```

예 9–8 책임 구분용 역할 만들기

이 예에서 관리자의 사이트가 LDAP 저장소를 사용합니다. 다음 명령을 실행하여 두 개의 역할을 만듭니다. usermgt 역할은 사용자를 만들고, 홈 디렉토리를 제공하고, 초기 암호를 할당하고, 기타 비보안 작업을 수행할 수 있습니다. usersec 역할은 사용자를 만들 수 없지만, 사용자 암호를 변경하고 다른 RBAC 등록 정보를 변경할 수 있습니다.

```
# roleadd -c "User Management role, LDAP" -s /usr/bin/pfbash \
-m -S ldap -K profiles="User Management"  usermgt
# roleadd -c "User Security role, LDAP" -s /usr/bin/pfbash \
-m -S ldap -K profiles="User Security"  usersec
```

예 9–9 장치 및 파일 보안 역할 만들기

이 예에서 관리자가 이 시스템에 대한 장치 및 파일 보안 역할을 만듭니다.

```
# roleadd -c "Device and File System Security admin, local" -s /usr/bin/pfbash \
-m -K profiles="Device Security,File System Security"  devflsec
```

▼ 역할을 할당하는 방법

이 절차는 역할을 사용자에 할당하고, 이름 캐시 데몬을 다시 시작하고, 사용자가 역할을 맡는 방법을 보여줍니다.

시작하기 전에 165 페이지 "역할을 만드는 방법"에 설명된 대로 역할을 추가하고 암호를 할당했습니다.

대부분의 사용자 보안 속성을 수정하려면 User Security 권한 프로파일에 할당되어야 합니다. 사용자의 감사 플래그를 수정하려면 수퍼유저여야 합니다. 기타 속성을 수정하려면 User Management 권한 프로파일에 할당되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 역할을 사용자에 할당합니다.

usermod [-S *repository***] [***RBAC-arguments***]** *login*

예를 들어, 로컬 사용자에 역할을 할당합니다.

```
# usermod -R +useradm jdoe-local
```

usermod 명령의 옵션은 usermod(1M) 매뉴얼 페이지나 165 페이지 "역할을 만드는 방법"에서 단계 2의 설명을 참조하십시오.

3 변경 사항을 적용하려면 이름 서비스 캐시 데몬을 다시 시작합니다.

```
# svcadm restart system/name-service-cache
```

예 9–10 암호화를 관리하는 역할 만들기 및 할당

이 예에서 LDAP 네트워크의 관리자가 암호화 프레임워크를 관리하는 역할을 만들어서 UID 1111에 할당합니다. 할당 사항을 적용하려면 nscd 데몬을 다시 시작합니다.

```
# roleadd -c "Cryptographic Services manager" \
-g 14 -m -u 104 -s /usr/bin/pfksh \
-S ldap -K profiles="Crypto Management" cryptmgt
# passwd cryptmgt
New Password:       <Type cryptmgt password>
Confirm password:       <Retype cryptmgt password>
# usermod -u 1111 -R +cryptmgt
# svcadm restart system/name-service-cache
```

UID 1111을 가진 사용자가 로그인하여 역할을 맡고 할당된 보안 속성을 표시합니다.

```
% su - cryptmgt
Password:       <Type cryptmgt password>
Confirm Password:       <Retype cryptmgt password>
$ profiles -l
      Crypto Management
          /usr/bin/kmfcfg             euid=0
          /usr/sbin/cryptoadm         euid=0
          /usr/sfw/bin/CA.pl          euid=0
```

```
            /usr/sfw/bin/openssl        euid=0
$
```

암호화 프레임워크에 대한 내용은 11 장, “암호화 프레임워크(개요)”를 참조하십시오. 프레임워크를 관리하려면 235 페이지 “암호화 프레임워크 관리(작업 맵)”를 참조하십시오.

▼ 역할을 감사하는 방법

역할이 수행하는 동작을 감사할 수 있습니다. 감사 레코드에는 역할을 맡은 사용자의 로그인 이름인 rolename과 역할이 수행한 동작이 포함됩니다.

`116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as` 감사 이벤트는 역할 동작을 캡처합니다. as, ex, ps, ua 클래스 중 하나를 미리 선택하면 역할 동작이 감사됩니다.

시작하기 전에 감사를 구성하려면 Audit Configuration 권한 프로파일이 지정되어야 합니다. 감사 서비스를 사용으로 설정하거나 새로 고치려면 Audit Control 권한 프로파일에 할당되어야 합니다.

1. **역할의 감사를 감사 계획에 넣습니다.**

   계획 정보는 27 장, “감사 계획”을 참조하십시오.

2. **필요한 보안 속성을 가진 관리자가 됩니다.**

   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

3. **as, ex, ps, ua 클래스 중 하나를 미리 선택합니다.**

   - **감사 서비스가 사용으로 설정된 경우 미리 선택한 클래스를 검토합니다.**

     ```
     # auditconfig -getflags
     ```

     as, ex, ps, ua 클래스 중 하나를 미리 선택한 경우 역할 동작이 감사되고 있습니다. 그렇지 않으면, 이러한 클래스 중 하나를 기존 클래스에 추가합니다.

     ```
     # auditconfig -setflags existing preselections,as
     ```

   - **감사가 아직 사용으로 설정되지 않은 경우 역할 동작을 감사할 클래스를 미리 선택합니다.**

     ```
     # auditconfig -setflags as
     ```

     이 예에서 관리자가 as 클래스를 선택합니다. 이 클래스는 다른 감사 이벤트를 포함합니다. 클래스에 포함된 감사 이벤트를 보려면 예 28–25에 표시된 대로 `auditrecord` 명령을 사용합니다.

4 **감사 서비스를 사용으로 설정하거나 새로 고칩니다.**

```
# audit -s
```

▼ 감사 프로파일을 만들거나 변경하는 방법

제공된 권한 프로파일에 필요한 컬렉션 보안 속성이 없을 때 권한 프로파일을 만들거나 변경할 수 있습니다. 권한 프로파일에 대해 알아보려면 144 페이지 "RBAC 권한 프로파일"을 참조하십시오.

새 권한 프로파일을 만드는 가장 쉬운 방법은 기존 권한 프로파일을 복사하여 수정하는 것입니다.

시작하기 전에 권한 프로파일을 만들거나 변경하려면 File Security 권한 프로파일에 할당되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **기존 프로파일에서 새 권한 프로파일을 만듭니다.**

```
# profiles [-S repository] existing-profile-name
```

새 이름을 묻는 메시지가 표시됩니다. 기존 권한 프로파일의 내용이 새 프로파일에 중복됩니다.

3 **계속해서 새 권한 프로파일을 수정합니다.**

다음 예제에 표시된 대로 보충 권한 프로파일, 권한 부여 및 기타 보안 속성을 추가하거나 제거합니다.

예 9–11 기존 프로파일에서 새 권한 프로파일 만들기

이 예에서 관리자가 LDAP 저장소에 Console User 권한 프로파일을 사용자 정의합니다.

```
# profiles -S ldap Console User
New name: ExampleCo Console User
ExampleCo Console User >
Description > Manage MyCompany Systems as the Console User
Help > ExCoConsUser.html
```

관리자가 이 권한 프로파일에 대한 `roleauth` 속성을 설정합니다.

```
roleauth=yes
```

예 9–12 권한 프로파일에서 기본 권한 제거

다음 예에서 테스트를 거친 후 보안 관리자가 SunRayUser 권한 프로파일에 할당된 모든 사용자로부터 기본 권한을 제거합니다. 이들은 proc_session 권한 사용이 금지됩니다. 즉, 이러한 사용자는 현재 세션 밖에 있는 프로세스를 조사할 수 없습니다.

```
$ profiles -K defaultpriv=basic,!proc_session SunRayUser
```

예 9–13 권한 프로파일의 제한 세트에서 권한 제거

다음 예에서 테스트를 거친 후 보안 관리자가 SunRayUser 권한 프로파일에 할당된 모든 사용자로부터 제한 권한을 제거합니다. 이렇게 하면 해당 사용자가 다른 사용자의 프로세스를 볼 수 없습니다.

```
$ profiles -K limitpriv=all,!proc_session SunRayUser
```

예 9–14 명령에 권한 추가

이 예에서 보안 관리자가 권한 프로파일의 응용 프로그램에 권한을 추가합니다. 응용 프로그램은 권한 인식형입니다.

```
# profiles -p SiteApp
profiles:SiteApp> set desc="Site application"
profiles:SiteApp> add cmd=/opt/site-app/bin/site-cmd
profiles:SiteApp:site-cmd> add privs=proc_fork,proc_taskid
profiles:SiteApp:site-cmd> end
profiles:SiteApp> exit
```

확인을 위해 관리자가 site-cmd를 선택합니다.

```
# profiles -p SiteApp "select cmd=/opt/site-app/bin/site-cmd; info;end"
Found profile in files repository.
  id=/opt/site-app/bin/site-cmd
  privs=proc_fork,proc_taskid
```

참조 보안 속성 할당 문제를 해결하려면 173 페이지 “RBAC 및 권한 할당 문제를 해결하는 방법”을 참조하십시오. 배경 지식은 199 페이지 “지정된 보안 속성의 검색 순서”를 참조하십시오.

▼ RBAC 등록 정보를 레거시 응용 프로그램에 추가하는 방법

레거시 응용 프로그램은 명령 또는 명령 세트입니다. 권한 프로파일의 각 명령에 대해 보안 속성이 설정됩니다. 그러면 권한 프로파일이 역할에 포함됩니다. 역할을 맡은 사용자는 보안 속성으로 레거시 응용 프로그램을 실행할 수 있습니다.

시작하기 전에 권한 프로파일을 만들려면 Information Security 또는 Rights Management 권한 프로파일에 할당되어야 합니다. 권한 프로파일을 할당하려면 User Security 권한 프로파일에 할당되어야 합니다.

1. **레거시 응용 프로그램을 구현하는 명령에 보안 속성을 추가합니다.**

   다른 명령과 마찬가지 방법으로 레거시 응용 프로그램에 보안 속성을 추가합니다. 보안 속성 포함 명령을 권한 프로파일에 추가해야 합니다. 레거시 명령에 대해 `euid=0` 또는 `uid=0` 보안 속성을 명령에 제공합니다. 세부 절차는 170 페이지 "감사 프로파일을 만들거나 변경하는 방법"을 참조하십시오.

   a. **레거시 응용 프로그램에 대해 새 권한 프로파일을 만듭니다.**

      단계는 170 페이지 "감사 프로파일을 만들거나 변경하는 방법"을 참조하십시오.

   b. **필요한 보안 속성을 가진 명령을 추가합니다.**

      예제는 예 9–14를 참조하십시오.

2. **역할의 프로파일 목록에 권한 프로파일을 넣습니다.**

   권한 프로파일을 역할에 할당하려면 예 9–10을 참조하십시오.

예 9–15 스크립트의 명령에 보안 속성 추가

스크립트의 명령을 성공하려면 `setuid` 또는 `setgid` 비트 세트가 필요한 경우 스크립트 실행 파일 **및** 명령에서 권한 프로파일에 보안 속성을 추가해야 합니다. 그런 다음, 권한 프로파일을 역할에 포함하고 역할을 사용자에 할당합니다. 사용자가 역할을 맡고 스크립트를 실행할 때 보안 속성으로 명령이 실행됩니다.

예 9–16 스크립트 또는 프로그램에서 권한 부여 검사

권한 부여용 스크립트를 만들려면 `auths` 명령에 기반한 테스트를 추가해야 합니다. 이 명령에 대한 자세한 내용은 `auths(1)` 매뉴얼 페이지를 참조하십시오.

예를 들어, 다음 라인은 사용자에 `$1` 인수로 제공된 권한 부여가 있는지 테스트합니다.

```
if [ ‘/usr/bin/auths|/usr/xpg4/bin/grep $1‘ ]; then
        echo Auth granted
else
        echo Auth denied
fi
```

테스트가 더 완벽하려면 와일드카드를 사용한 권한 부여를 검사하는 논리를 테스트에 포함해야 합니다. 예를 들어, 사용자가 `solaris.system.date` 권한 부여를 가지고 있는지 테스트하려면 다음 문자열을 검사해야 합니다.

- `solaris.system.date`
- `solaris.system.*`
- `solaris.*`

프로그램을 작성하는 경우 `getauthattr()` 함수를 사용하여 권한 부여를 테스트합니다.

▼ RBAC 및 권한 할당 문제를 해결하는 방법

사용자나 역할의 프로세스가 할당된 보안 속성으로 실행되지 않는 이유에는 여러 인자가 관여합니다.

- 보안 속성의 철자가 틀립니다. 철자가 틀린 권한 부여는 자동으로 실패합니다.
- 사용자나 역할이 할당이 포함된 이름 지정 서비스를 사용하지 않습니다.
- 예상한 할당이 해당 속성의 첫번째 할당이 아닙니다.

  사용자나 역할의 보안 속성을 검색한 후 인증 시 할당되는 순서에 따라 성공 여부가 결정됩니다. 단, 권한 부여는 예외입니다. 검색 중 사용자나 역할에 할당된 권한 부여는 누적됩니다. 이와 반대로, 권한 할당 및 권한 프로파일의 보안 속성 할당은 검색에 종속됩니다. 첫번째 할당을 성공하고, 그 이후의 할당은 무시됩니다.
- 명령이 프로파일 셸에서 실행 중이 아닙니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

1 이름 지정 서비스를 확인하고 다시 시작합니다.

a. 사용자나 역할에 대한 보안 할당이 시스템에서 사용으로 설정된 이름 지정 서비스에 속하는지 확인합니다.

b. 이름 서비스 캐시 `svc:/system/name-service/cache`를 다시 시작합니다.

`nscd` 데몬의 TTL(time-to-live) 간격이 길어질 수 있습니다. 데몬을 다시 시작하여 이름 지정 서비스를 현재 데이터로 업데이트합니다.

2 보안 속성이 할당되는 위치를 확인합니다.

`userattr -v` 명령의 값으로 보안 속성을 사용합니다. 예를 들어, 다음 명령은 사용자 `jdoe`에 대해 어떤 보안 속성이 할당되고 어떤 곳에서 할당이 이루어졌는지 나타냅니다.

```
# userattr -v audit_flags jdoe      Modifications to the system defaults
user_attr: fw:no
# userattr -v auths jdoe      Assigned authorizations
```

```
solaris.admin.wusb.read,solaris.device.cdrw,solaris.device.mount.removable,
solaris.mail.mailq,solaris.profmgr.read,solaris.smf.manage.audit,
solaris.smf.value.audit
# userattr -v audit_flags jdoe       Modifications to audit preselection mask
# userattr -v auths jdoe       Assigned authorizations
# userattr -v defaultpriv jdoe       Modifications to basic user privileges
# userattr -v limitpriv jdoe       Modifications to limit privileges
# userattr -v lock_after_retries jdoe       Automatic lockout attribute
# userattr -v profiles jdoe       Assigned rights profiles
user_attr: Audit Review,Stop
# userattr roles jdoe       Assigned roles
user_attr : cryptomgt,infosec
```

3 직접 만든 권한 프로파일의 경우 적절한 보안 속성을 명령에 할당했는지 확인합니다.

예를 들어, 일부 명령을 성공하려면 `euid=0`이 아닌 `uid=0`이 필요합니다. 일부 명령의 측면에서 권한 부여가 필요할 수 있습니다.

4 보안 속성을 사용자에 제공할 수 없는 경우 다음을 확인합니다.

a. 보안 속성이 사용자에 직접 할당되었는지 확인합니다.

`userattr` 명령을 사용합니다.

b. 보안 속성이 직접 할당되지 않은 경우 사용자에 직접 할당된 권한 프로파일을 확인합니다.

i. 순서대로, 권한 프로파일 목록에서 보안 속성 할당을 검사합니다.

목록에서 가장 빠른 권한 프로파일의 속성 값이 사용자가 사용할 수 있는 값입니다. 이 값이 올바르지 않으면 해당 권한 프로파일의 값을 변경하거나 프로파일 목록의 순서를 바꿉니다.

권한 있는 명령의 경우 권한이 `defaultpriv` 키워드에 지정되었는지 확인합니다. 이 할당은 특정 명령에 대한 권한과 더불어 제공됩니다.

ii. 속성 할당이 목록에 없으면 사용자가 할당된 역할을 확인합니다.

속성이 역할에 할당된 경우 보안 속성을 얻으려면 사용자가 역할을 맡아야 합니다. 속성이 여러 개의 역할에 할당된 경우 목록에서 가장 빠른 역할의 할당이 적용됩니다. 이 값이 올바르지 않으면 목록의 첫번째 역할에 올바른 값을 할당하거나 역할 할당의 순서를 바꿉니다.

5 사용자나 역할에 직접 권한을 할당한 경우 실패한 명령을 성공하려면 권한 부여가 필요한지 확인합니다.

주 – 일부 명령의 측면에서 권한 부여가 필요할 수 있습니다. 최적의 사용법은, 권한을 직접 할당하기보다 관리 명령이 포함된 권한 프로파일을 할당하는 것입니다.

관리 명령이 포함된 권한 프로파일을 검토하십시오. 권한 부여가 포함된 권한 프로파일이 존재하는 경우 권한이 아닌, 권한 프로파일을 사용자에 할당합니다. 명령이 포함된 권한 프로파일은 다른 어떤 권한 프로파일보다 앞서 순서를 배치합니다.

6 사용자에 대한 명령을 계속 실패하는 경우 다음을 확인합니다.

a. 사용자가 프로파일 셸에서 명령을 실행 중인지 확인합니다.

관리 명령은 프로파일 셸에서 실행해야 합니다. 사용자 오류를 줄이려면 프로파일 셸을 사용자의 로그인 셸로 할당할 수 있습니다. 또는 사용자가 프로파일 셸에서 관리 명령을 실행하도록 미리 알려줄 수 있습니다.

b. 사용자에 직접 할당된 보안 속성이 있을 경우 명령 성공을 막는지 확인합니다.

특히, 사용자의 `defaultpriv` 및 `limitpriv` 속성 값을 확인합니다.

c. 어떤 권한 프로파일이나 역할이 명령을 포함하는지 확인합니다.

i. 순서대로, 권한 프로파일 목록에서 보안 속성 포함 명령을 검사합니다.

권한 프로파일 목록에서 가장 빠른 값이 사용자가 사용할 수 있는 값입니다. 이 값이 올바르지 않으면 해당 권한 프로파일의 값을 변경하거나 프로파일 목록의 순서를 바꿉니다.

특히, 프로파일의 `defaultpriv` 및 `limitpriv` 속성 값을 확인합니다.

ii. 속성 할당이 목록에 없으면 사용자가 할당된 역할을 확인합니다.

명령이 역할에 할당된 경우 보안 속성을 얻으려면 사용자가 역할을 맡아야 합니다. 속성이 여러 개의 역할에 할당된 경우 목록에서 가장 빠른 역할의 할당이 적용됩니다. 이 값이 올바르지 않으면 목록의 첫번째 역할에 올바른 값을 할당하거나 역할 할당의 순서를 바꿉니다.

7 역할에 대한 명령을 실패하는 경우 다음을 확인합니다.

관리 명령을 성공하려면 권한이 필요합니다. 일부 명령의 측면에서 권한 부여가 필요할 수 있습니다. 최적의 사용법은 관리 명령이 포함된 권한 프로파일을 할당하는 것입니다.

a. 역할에 직접 할당된 보안 속성이 있을 경우 명령 성공을 막는지 확인합니다.

특히, 역할의 `defaultpriv` 및 `limitpriv` 속성 값을 확인합니다.

b. 순서대로, 권한 프로파일 목록에서 보안 속성 포함 명령을 검사합니다.

권한 프로파일 목록에서 가장 빠른 값이 사용자가 사용할 수 있는 값입니다. 이 값이 올바르지 않으면 해당 권한 프로파일의 값을 변경하거나 프로파일 목록의 순서를 바꿉니다.

RBAC 관리(작업)

RBAC를 구성한 후에 사용 중인 경우 다음 절차에 따라 시스템에서 RBAC를 유지 관리하고 수정합니다.

RBAC 관리(작업 맵)

다음 작업 맵은 RBAC(역할 기반 액세스 제어)를 초기에 구현한 후에 RBAC를 유지 관리하기 위한 절차를 가리킵니다.

작업	설명	수행 방법
역할 암호를 변경합니다.	권한이 부여된 사용자나 역할이 다른 역할의 암호를 변경합니다.	177 페이지 "역할의 암호를 변경하는 방법"
역할의 할당된 권한을 수정합니다.	역할의 보안 속성을 수정합니다.	178 페이지 "역할의 보안 속성을 변경하는 방법" 예 9–19
사용자의 권한을 수정합니다.	일반 사용자에 보안 속성을 추가하거나 제거합니다.	179 페이지 "사용자의 RBAC 등록 정보를 변경하는 방법" 예 9–24 예 9–12
권한 프로파일에서 사용자의 권한을 수정합니다.	감사 플래그, 기본 권한과 같은 보안 속성 값을 권한 프로파일에서 할당합니다.	예 9–21 예 9–13
제한된 프로파일 셸을 만듭니다.	사용자나 역할이 소프트웨어의 모든 명령에 전체 액세스하지 못하도록 합니다.	182 페이지 "관리자를 명시적으로 할당된 권한으로 제한하는 방법"
시스템에서 기본 권한을 제거합니다.	특수한 용도의 시스템을 만듭니다.	예 9–25
사용자의 권한을 제한합니다.	사용자의 기본 또는 제한 권한 세트를 제한합니다.	예 9–21

작업	설명	수행 방법
사용자가 역할을 맡으려면 사용자 암호를 제공하도록 합니다.	사용자 암호로 역할에 인증되도록 사용자의 보안 속성을 수정합니다. 이 동작은 Linux 역할 동작과 비슷합니다.	183 페이지 "사용자가 고유의 암호를 사용하여 역할을 맡도록 설정하는 방법"
root를 사용자로 변경합니다.	시스템을 폐기하기 전에 root 역할을 사용자로 변경합니다.	184 페이지 "root 역할을 사용자로 변경하는 방법"

이러한 절차는 사용자, 역할, 권한 프로파일에 대한 보안 속성을 관리합니다. 기본 사용자 관리 절차는 **Oracle Solaris 관리: 일반 작업**의 2 장, "사용자 계정 및 그룹 관리(개요)"를 참조하십시오.

▼ 역할의 암호를 변경하는 방법

시작하기 전에 root 역할을 가진 사용자여야 합니다.

- **passwd 명령을 실행합니다.**

  ```
  # passwd [-r naming-service] target-rolename
  ```

  `-r` *naming-service* — files 또는 ldap 저장소에 암호 변경을 적용합니다. 기본 저장소는 files입니다. 저장소를 지정하지 않으면 모든 저장소에서 암호가 변경됩니다.

  target-rolename — 수정하려는 기존 역할의 이름입니다.

  더 많은 명령 옵션은 passwd(1) 매뉴얼 페이지를 참조하십시오.

예 9–17 역할의 암호 변경

이 예에서 root 역할이 로컬 devmgt 역할의 암호를 변경합니다.

```
# passwd -r files  devmgt
New password:       Type new password
Confirm password:       Retype new password
```

이 예에서 root 역할이 LDAP 디렉토리 서비스에서 devmgt 역할의 암호를 변경합니다.

```
# passwd -r ldap devmgt
New password:       Type new password
Confirm password:       Retype new password
```

이 예에서 root 역할이 파일 및 LDAP에서 devmgt 역할의 암호를 변경합니다.

```
# passwd devmgt
New password:       Type new password
Confirm password:       Retype new password
```

▼ 역할의 보안 속성을 변경하는 방법

시작하기 전에 역할의 암호 및 감사 플래그를 제외한, 역할의 보안 속성을 변경하려면 User Security 권한 프로파일에 할당되어야 합니다. 역할 등록 정보에는 권한 프로파일 및 권한 부여가 포함됩니다. 감사 플래그를 할당하거나 역할의 암호를 변경하려면 root 역할을 맡아야 합니다.

주 – 암호를 변경하려면 177 페이지 "역할의 암호를 변경하는 방법"을 참조하십시오.

---

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 rolemod 명령을 사용합니다.

이 명령은 로컬 이름 지정 서비스 또는 LDAP에 정의된 역할의 속성을 수정합니다. -A, -P, -R 옵션의 값을 - 또는 ++로 수정할 수 있습니다. -는 현재 지정된 값에서 해당 값을 빼는 것입니다. ++는 현재 지정된 값에 해당 값을 더하는 것입니다.

rolemod 명령에 대한 자세한 내용은 다음을 참조하십시오.

- 간단한 설명은 165 페이지 "역할을 만드는 방법"에서 roleadd 명령의 설명을 참조하십시오.
- 이 명령의 모든 인수는 rolemod(1M) 매뉴얼 페이지를 참조하십시오.
- -K 옵션에 대한 키 값 목록은 user_attr(4) 매뉴얼 페이지를 참조하십시오.

다음 명령은 LDAP 저장소에서 devmgt 역할의 할당된 권한 프로파일을 바꿉니다.

```
$ rolemod  -P "Device Management,File Management" -S ldap devadmin
```

예 9–18 로컬 역할의 보안 속성 변경

이 예에서 보안 관리자가 prtmgt 역할에 VSCAN Management 권한 프로파일이 포함되도록 수정합니다.

```
$ rolemod -c "Handles printers and virus scanning" \
-P "Printer Management,VSCAN Management,All" prtmgt
```

이러한 권한 프로파일은 policy.conf 파일을 통해 부여된 프로파일에 추가됩니다.

예 9–19 역할에 직접 권한 할당

이 예에서 보안 관리자가 시스템 시간에 영향을 미치는 매우 특정한 권한으로 systime 역할을 신뢰합니다.

```
$ rolemod -K priv=proc_clock_highres systime
```

priv 키워드의 값은 항상 역할의 프로세스에서 권한 목록에 속합니다.

▼ 사용자의 RBAC 등록 정보를 변경하는 방법

사용자 등록 정보에는 로그인 셸, 권한 프로파일, 역할이 포함됩니다. 사용자에게 관리 기능을 제공하는 가장 안전한 방법은 사용자에 역할을 할당하는 것입니다. 설명은 145 페이지 "보안 속성을 직접 지정할 때 보안 고려 사항"을 참조하십시오.

시작하기 전에 사용자의 암호 및 감사 플래그를 제외한, 사용자의 보안 속성을 변경하려면 User Security 권한 프로파일에 할당되어야 합니다. 감사 플래그를 지정하거나 역할의 암호를 변경하려면 root 역할을 맡아야 합니다. 기타 사용자 속성을 변경하려면 User Management 권한 프로파일에 할당되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 usermod 명령을 사용합니다.

이 명령은 로컬 이름 지정 서비스 또는 LDAP 이름 지정 서비스에 정의된 사용자의 속성을 수정합니다. 이 명령의 RBAC 인수는 useradd 명령의 인수와 비슷합니다. user_attr(4) 매뉴얼 페이지의 설명과 예 9–23에 표시된 내용을 참조하십시오.

다음 예에서 LDAP 사용자가 devmgt 역할에 할당됩니다. 이 역할은 이전의 역할 할당을 대체합니다. devmgt 역할은 LDAP 이름 지정 서비스에 존재해야 합니다.

```
$ usermod -R devmgt -S ldap jdoe-ldap
```

다음 예에서 이 역할이 이전의 역할 할당에 추가됩니다.

```
$ usermod -R +devmgt -S ldap jdoe-ldap
```

예 9–20 로컬 사용자에 역할 할당

이 예에서 사용자 jdoe가 이제 System Administrator인 sysadmin의 역할을 맡을 수 있습니다.

```
$ userattr roles jdoe
secdevice
$ usermod -R secdevice,sysadmin jdoe
$ userattr roles jdoe
secdevice,sysadmin
```

예 9–21 사용자의 제한 세트에서 권한 제거

다음 예에서 jdoe의 초기 로그인에서 시작된 모든 세션이 sys_linkdir 권한 사용을 금지합니다. 즉, 사용자가 su 명령을 실행한 후에도 디렉토리에 하드 링크를 만들거나 디렉토리 링크를 해제할 수 없습니다.

```
$ usermod -K limitpriv=all,!sys_linkdir jdoe
$ userattr limitpriv jdoe
all,!sys_linkdir
```

예 9–22 DHCP를 관리할 수 있는 사용자 만들기

이 예에서 보안 관리자가 LDAP에 사용자를 만듭니다. 로그인 시 jdoe-dhcp 사용자가 DHCP를 관리할 수 있습니다.

```
# useradd -P "DHCP Management" -s /usr/bin/pfbash -S ldap  jdoe-dhcp
```

사용자가 로그인 셸로 pfbash에 할당되었기 때문에 DHCP Management 권한 프로파일의 보안 속성을 사용자의 보안 셸에서 사용할 수 있습니다.

예 9–23 사용자에 직접 권한 부여 할당

이 예에서 보안 관리자가 화면 밝기를 조절할 수 있는 로컬 사용자를 만듭니다.

```
# useradd -c "Screened JDoe, local" -s /usr/bin/pfbash \
-A solaris.system.power.brightness  jdoe-scr
```

이 권한 부여는 사용자의 기존 권한 부여 할당에 추가됩니다.

예 9–24 사용자에 직접 권한 할당

이 예에서 보안 관리자가 시스템 시간에 영향을 미치는 매우 특정한 권한으로 사용자 jdoe를 신뢰합니다.

```
$ usermod -K defaultpriv=basic,proc_clock_highres jdoe
```

defaultpriv 키워드의 값이 기존 값을 대체합니다. 따라서 basic 권한을 보유할 사용자에 대해 basic 값이 지정됩니다. 기본 구성에서 모든 사용자는 기본 권한을 갖습니다.

▼ 사용자를 데스크탑 응용 프로그램으로 제한하는 방법

Oracle Solaris 사용자를 데스크탑 액세스로만 제한할 수 있습니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 프로파일 셸을 로그인 셸로 사용자에 할당합니다.

예를 들어, pfbash 셸을 사용자에 할당할 수 있습니다.

```
# usermod -s /usr/bin/pfbash username
```

이제 모든 사용자 프로세스가 RBAC 통제하에 있습니다.

2 Oracle 데스크탑에서 사용자가 기본 애플릿을 실행할 수 있는 권한 프로파일을 만듭니다.

다음 명령은 권한 프로파일을 만듭니다. end 명령은 추가된 명령에 보안 속성이 필요 없음을 나타냅니다. LDAP 저장소에 권한 프로파일을 만들려면 -S ldap 옵션을 사용합니다.

```
# profiles -p "Desktop Applets"
profiles:Desktop Applets> set desc="Can use basic desktop applications"
profiles:Desktop Applets> add cmd=/usr/bin/nautilus;end
profiles:Desktop Applets> add cmd=/usr/bin/dbus-launch;end
profiles:Desktop Applets> add cmd=/usr/lib/dbus-daemon;end
profiles:Desktop Applets> add cmd=/usr/lib/clock-applet;end
profiles:Desktop Applets> add cmd=/usr/lib/gconfd-2;end
profiles:Desktop Applets> add cmd=/usr/lib/gvfsd;end
profiles:Desktop Applets> add cmd=/usr/lib/gvfsd-metadata;end
profiles:Desktop Applets> add cmd=/usr/lib/gvfsd-trash;end
profiles:Desktop Applets> add cmd=/usr/lib/gvfs-hal-volume-monitor;end
profiles:Desktop Applets> add cmd=/usr/lib/gnome-pty-helper;end
profiles:Desktop Applets> add cmd=/usr/lib/utmp_update;end
profiles:Desktop Applets> add cmd=/usr/bin/sh;end
profiles:Desktop Applets> add cmd=/usr/bin/bash;end
profiles:Desktop Applets> add cmd=/usr/bin/csh;end
profiles:Desktop Applets> add cmd=/usr/bin/ksh;end
profiles:Desktop Applets> commit
profiles:Desktop Applets> exit
```

3 권한 프로파일의 입력이 올바른지 확인합니다.

입력에 오타, 누락, 반복과 같은 오류가 있는지 검토합니다.

```
# profiles -p "Desktop Applets" info
Found profile in files repository.
name=Desktop Applets
desc=Can use basic desktop applications
cmd=/usr/bin/nautilus
cmd=/usr/bin/dbus-launch
cmd=/usr/lib/dbus-daemon
cmd=/usr/lib/clock-applet
cmd=/usr/lib/gconfd-2
cmd=/usr/lib/gvfsd
```

```
cmd=/usr/lib/gvfsd-metadata
cmd=/usr/lib/gvfsd-trash
cmd=/usr/lib/gvfs-hal-volume-monitor
cmd=/usr/lib/gnome-pty-helper
cmd=/usr/lib/utmp_update
cmd=/usr/bin/sh
cmd=/usr/bin/bash
cmd=/usr/bin/csh
cmd=/usr/bin/ksh
```

참고 – 데스크탑 아이콘을 가진 응용 프로그램 또는 응용 프로그램 클래스에 대해 권한 프로파일을 만들 수 있습니다. 그런 다음, 이 새 권한 프로파일에 대한 보충 권한 프로파일로 Desktop Applets를 추가합니다. 더불어, 이러한 권한 프로파일을 통해 사용자가 적절한 데스크탑 응용 프로그램을 사용할 수 있습니다.

4 **Desktop Applets 권한 프로파일과 Stop 권한 프로파일을 사용자에 할당합니다.**

```
# usermod -P "Desktop Applets,Stop" username
```

이 사용자는 Basic Solaris User 권한 프로파일이나 Console User 권한 프로파일을 보유하지 않습니다. 따라서 이 사용자는 Desktop Applets 권한 프로파일의 명령 이외의 다른 명령을 실행할 수 없습니다. 예를 들어, 사용자는 터미널 창에 액세스할 수 없습니다.

자세한 내용은 197 페이지 “권한 프로파일”, 199 페이지 “지정된 보안 속성의 검색 순서” 및 **Trusted Extensions 구성 및 관리**의 “사용자를 데스크탑 응용 프로그램으로 제한하는 방법”을 참조하십시오.

`usermod` 명령은 로컬 이름 지정 서비스 또는 LDAP에 정의된 사용자 속성을 수정합니다. 이 명령의 인수는 `usermod(1M)` 매뉴얼 페이지를 참조하십시오.

▼ 관리자를 명시적으로 할당된 권한으로 제한하는 방법

다음 두 가지 방법으로 역할이나 사용자를 제한된 수의 관리 작업으로 제한할 수 있습니다.

- Stop 권한 프로파일을 사용할 수 있습니다.

  Stop 권한 프로파일은 제한된 셸을 만드는 가장 간단한 방법입니다. `policy.conf` 파일에 지정된 권한 부여 및 권한 프로파일은 고려되지 않습니다. 기본 구성에서는 역할이나 사용자가 Basic Solaris User 권한 프로파일, Console User 권한 프로파일 또는 `solaris.device.cdrw` 권한 부여에 할당되지 않습니다.
- 시스템에서 `policy.conf` 파일을 수정하고, 역할이나 사용자가 관리 작업에 해당 시스템을 사용할 수 있도록 합니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

- **Stop 권한 프로파일을 프로파일 목록의 마지막 프로파일로 추가합니다.**

  예를 들어, 감사 평가만 수행하도록 auditrev 역할을 제한할 수 있습니다.

  ```
  # rolemod -P "Audit Review,Stop" auditrev
  ```

  auditrev 역할에 Console User 권한 프로파일이 없기 때문에 감사자가 시스템을 종료할 수 없습니다. 이 역할에 solaris.device.cdrw 권한 부여가 없기 때문에 감사자가 CD-ROM 드라이브에서 읽기/쓰기를 수행할 수 없습니다. 이 역할에 Basic Solaris User 권한 프로파일이 없기 때문에 이 역할에서 Audit Review 권한 프로파일의 명령 이외의 다른 명령을 실행할 수 없습니다. 예를 들어, ls 명령이 실행되지 않습니다. 역할이 File Browser를 사용하여 감사 파일을 봅니다.

  자세한 내용은 197 페이지 “권한 프로파일” 및 199 페이지 “지정된 보안 속성의 검색 순서”를 참조하십시오.

  rolemod 명령은 로컬 이름 지정 서비스 또는 LDAP에 정의된 역할의 속성을 수정합니다. 이 명령의 인수는 rolemod(1M) 매뉴얼 페이지를 참조하십시오. 165 페이지 “역할을 만드는 방법”에 설명된 대로 RBAC 인수 목록은 roleadd 명령의 목록과 비슷합니다.

예 9–25 사용자에 제공되는 권한을 제한하도록 시스템 수정

이 예에서 관리자가 네트워크 관리에만 사용되는 시스템을 만듭니다. 관리자가 Basic Solaris User 권한 프로파일과 solaris.device.cdrw 권한 부여를 policy.conf 파일에서 제거합니다. Console User 권한 프로파일은 제거되지 않습니다. policy.conf 결과 파일에서 영향을 받는 라인은 다음과 같습니다.

```
...
#AUTHS_GRANTED=solaris.device.cdrw
#PROFS_GRANTED=Basic Solaris User
CONSOLE_USER=Console User
...
```

권한 부여, 명령, 권한 프로파일이 명시적으로 할당된 사용자만 이 시스템을 사용할 수 있습니다. 로그인 후에 권한이 부여된 사용자가 관리 업무를 수행할 수 있습니다. 권한이 부여된 사용자가 시스템 앞에 앉으면 Console User의 권한을 갖습니다.

▼ 사용자가 고유의 암호를 사용하여 역할을 맡도록 설정하는 방법

기본적으로 사용자가 역할을 맡으려면 역할의 암호를 입력해야 합니다. 이 절차에 따라 Linux 환경에서 역할을 맡듯이 Oracle Solaris에서 역할을 맡을 수 있습니다.

시작하기 전에 User Security 권한 프로파일이 포함된 역할을 맡고 있어야 합니다. 이 역할은 변경하려는 roleauth 값이 속한 역할일 수 없습니다.

- **사용자 암호로 역할을 인증하도록 합니다.**

  ```
  $ rolemod -K roleauth=user rolename
  ```

  이 역할을 맡으려면 할당된 사용자가 역할을 위해 특별히 만든 암호가 아닌, 고유의 암호를 사용할 수 있습니다.

예 9–26 권한 프로파일을 사용할 때 역할이 할당된 사용자의 암호를 사용하도록 설정

이 예에서 root 역할이 로컬 시스템에서 secadmin 역할에 대한 roleauth의 값을 변경합니다.

```
# profiles -K roleauth=user "System Administrator"
```

Security Administrator 권한 프로파일에 할당된 사용자가 역할을 맡을 때 암호를 묻는 메시지가 나타납니다. 다음 시퀀스에서 역할 이름은 secadmin입니다.

```
% su - secadmin
Password:       Type user password
$       /** You are now in a profile shell with administrative rights**/
```

사용자가 다른 역할에 할당된 경우도 해당 역할로 인증하기 위해 고유의 암호를 사용합니다.

예 9–27 LDAP 저장소에서 역할에 대한 roleauth의 값 변경

이 예에서 root 역할을 통해 secadmin 역할을 맡을 수 있는 모든 사용자가 역할을 맡을 때 고유의 암호를 사용하도록 합니다. 이 기능은 LDAP 서버에서 관리되는 모든 시스템에 대해 해당 사용자에게 부여됩니다.

```
# rolemod -S ldap -K roleauth=user secadmin
# profiles -S ldap -K roleauth=user "Security Administrator"
```

일반 오류 roleauth=user가 역할에 설정된 경우 사용자 암호를 통해 인증된 역할이 해당 역할에 할당된 모든 권한에 액세스할 수 있습니다. 이 키워드는 검색에 종속적입니다. 자세한 내용은 199 페이지 “지정된 보안 속성의 검색 순서”를 참조하십시오.

▼ root 역할을 사용자로 변경하는 방법

관리자가 네트워크에서 제거된 시스템을 폐기할 때 root를 사용자로 변경할 수 있습니다. 이 경우 root로 시스템에 로그인하면 간단히 정리됩니다.

시작하기 전에 User Management 및 User Security 권한 프로파일에 지정된 관리자여야 합니다.

1 root 역할 지정을 로컬 사용자로부터 제거합니다.

예를 들어, 두 사용자로부터 역할 할당을 제거합니다.

```
% su - root
Password: a!2@3#4$5%6^7
# roles jdoe
root
# roles kdoe
root
# roles ldoe
secadmin
# usermod -R "" jdoe
# usermod -R "" kdoe
#
```

2 root 역할을 사용자로 변경합니다.

```
# rolemod -K type=normal root
```

현재 root 역할에 속한 사용자는 그대로 남고, 루트 액세스를 가진 다른 사용자는 root에 su를 실행하거나 root 사용자로 시스템에 로그인할 수 있습니다.

3 변경 사항을 확인합니다.

다음 명령 중 하나를 사용할 수 있습니다.

```
# getent user_attr root
root::::auths=solaris.*;profiles=All;audit_flags=lo\:no;lock_after_retries=no;
min_label=admin_low;clearance=admin_high
```

type 키워드가 출력에서 누락되거나 normal과 같은 경우 계정은 역할이 아닙니다.

```
# userattr type root
```

출력이 비어 있거나 normal을 나열하는 경우 계정은 역할이 아닙니다.

예 9–28 root 역할이 시스템 구성에 사용되지 못하도록 금지

이 예에서 사이트 보안 정책에 따라 root 계정이 시스템을 유지 관리하지 못하도록 해야 합니다. 관리자가 시스템을 유지 관리하는 역할을 만들고 테스트했습니다. 이러한 역할에는 모든 보안 프로파일과 System Administrator 권한 프로파일이 포함됩니다. 신뢰된 사용자에게 백업을 복원할 수 있는 역할이 지정되었습니다. 시스템, 사용자, 권한 프로파일에 대한 감사 플래그를 변경할 수 있는 역할은 없습니다.

root 계정이 시스템 유지 관리에 사용되지 못하도록 하려면 보안 관리자가 루트 role 지정을 제거합니다. root 계정이 단일 사용자 모드로 시스템에 로그인할 수 있어야 하므로 계정이 암호를 유지합니다.

```
# rolemod -K roles= jdoe
# userattr roles jdoe
```

예 9–29 root 사용자를 root 역할로 변경

이 예에서 root 사용자가 root 사용자를 역할로 되돌립니다.

먼저, root가 root 계정을 역할로 변경하고 변경 사항을 확인합니다.

```
# rolemod -K type=role root
# getent user_attr root
root::::type=role;auths=solaris.*;profiles=All;audit_flags=lo\:no;
lock_after_retries=no;min_label=admin_low;clearance=admin_high
```

그런 다음, root가 root 역할을 로컬 사용자에 지정합니다.

```
# usermod -R root jdoe
```

일반 오류 데스크탑 환경에서 root가 역할일 때 root로 직접 로그인할 수 없습니다. 진단 메시지는 root가 시스템의 역할임을 나타냅니다.

root 역할을 맡을 수 있는 로컬 계정이 없는 경우 하나 만듭니다. root로서 단일 사용자 모드로 시스템에 로그인하여 로컬 사용자 계정 및 암호를 만들고 새 계정에 root 역할을 지정합니다. 그런 다음, 새 사용자로 로그인하여 root 역할을 맡습니다.

권한 사용(작업)

다음 작업 맵은 시스템에서 권한을 관리하고 사용하기 위한 단계별 지침을 가리킵니다.

작업	설명	수행 방법
명령을 실행할 때 권한을 사용합니다.	내게 할당된 권한과 시스템에서 사용 가능한 권한을 나열합니다.	186 페이지 “권한 확인(작업 맵)”
사이트에서 권한을 사용합니다.	권한을 할당, 제거, 추가하고 권한 사용을 디버그합니다.	190 페이지 “권한 관리(작업 맵)”

권한 확인(작업 맵)

사용자가 권한에 직접 할당된 경우 권한은 모든 셸에서 유효합니다. 사용자에게 직접 권한이 지정되지 않은 경우 사용자가 프로파일 셸을 열어야 합니다. 예를 들어 지정된 권한을 가진 명령이 사용자의 권한 프로파일 목록에 있는 권한 프로파일에 있는 경우 사용자가 프로파일 셸에서 명령을 실행해야 합니다.

다음 작업 맵은 내게 할당된 권한을 보기 위한 절차를 가리킵니다

작업	설명	수행 방법
정의된 권한을 봅니다.	Oracle Solaris 권한 및 해당 정의를 나열합니다.	187 페이지 “시스템의 권한을 나열하는 방법”
임의 셸에서 사용자로 권한을 봅니다.	직접 할당된 권한을 보여줍니다. 모든 프로세스가 이러한 권한으로 실행됩니다.	188 페이지 “직접 할당된 권한을 확인하는 방법”
프로파일 셸에서 권한 있는 명령을 봅니다.	사용자가 할당된 권한 프로파일을 통해 실행할 수 있는 권한 있는 명령을 보여줍니다.	189 페이지 “실행할 수 있는 권한 있는 명령을 확인하는 방법”
임의 셸에서 역할로 권한을 봅니다.	역할이 할당된 권한 프로파일을 통해 실행할 수 있는 권한 있는 명령을 보여줍니다.	189 페이지 “실행할 수 있는 권한 있는 명령을 확인하는 방법”

▼ 시스템의 권한을 나열하는 방법

다음 절차는 권한 이름 및 정의를 보는 방법을 보여줍니다.

- **터미널 창에서 온라인으로 권한을 볼 수 있습니다.**
  - **privileges(5) 매뉴얼 페이지를 참조하여 권한을 나열합니다.**

```
% man privileges
Standards, Environments, and Macros                 privileges(5)

NAME
     privileges - process privilege model
...
     The defined privileges are:

     PRIV_CONTRACT_EVENT

         Allow a process to request reliable delivery  of  events
         to an event endpoint.

         Allow a process to include events in the critical  event
         set  term  of  a  template  which  could be generated in
         volume by the user.
...
```

이 권한 형식은 개발자가 사용합니다.

  - **ppriv 명령을 사용하여 권한을 나열합니다.**

```
% ppriv -lv | more
contract_event
    Allows a process to request critical events without limitation.
    Allows a process to request reliable delivery of all events on
    any event queue.
...
win_upgrade_sl
        Allows a process to set the sensitivity label of a window
        resource to a sensitivity label that dominates the existing
        sensitivity label.
        This privilege is interpreted only if the system is configured
        with Trusted Extensions.
```

이 권한 형식은 useradd, roleadd, usermod , rolemod 명령을 사용하여 사용자 및 역할에 권한을 할당하고 profiles 명령을 사용하여 권한 프로파일에 권한을 할당할 수 있습니다.

▼ 직접 할당된 권한을 확인하는 방법

다음 절차는 권한이 직접 할당되었는지 확인하는 방법을 보여줍니다.

주의 – 직접 할당된 권한의 부적절한 사용은 의도하지 않은 보안 침해를 일으킬 수 있습니다. 설명은 145 페이지 “보안 속성을 직접 지정할 때 보안 고려 사항”을 참조하십시오.

1. **프로세스가 사용할 수 있는 권한을 나열합니다.**
   절차는 191 페이지 “프로세스의 권한을 확인하는 방법”을 참조하십시오.

2. **동작을 호출하고 임의 셸에서 명령을 실행합니다.**
   유효 세트에 나열된 권한은 세션 전체 동안 유효합니다. 기본 세트와 더불어 권한이 직접 할당된 경우 해당 권한이 유효 세트에 나열됩니다.

예 9–30 사용자 - 직접 지정된 권한 확인

사용자에게 직접 권한이 지정된 경우 해당 사용자의 기본 세트에는 원래 제공되는 기본 세트보다 더 많은 권한이 포함되어 있습니다. 이 예에서 사용자는 항상 proc_clock_highres 권한에 액세스할 수 있습니다.

```
% /usr/bin/whoami
jdoe
% ppriv -v $$
1800:   pfksh
flags = <none>
        E: file_link_any,...,proc_clock_highres,proc_session
        I: file_link_any,...,proc_clock_highres,proc_session
        P: file_link_any,...,proc_clock_highres,proc_session
        L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,sys_time
% ppriv -vl proc_clock_highres
        Allows a process to use high resolution timers.
```

예 9–31 역할 - 직접 지정된 권한 확인

역할은 관리 셸 또는 프로파일 셸을 사용합니다. 역할을 맡은 사용자는 역할의 셸을 사용하여 역할에 직접 할당된 권한을 나열할 수 있습니다. 다음 예에서 realtime 역할은 날짜 및 시간 프로그램을 처리하기 위해 권한이 직접 할당되었습니다.

```
% su - realtime
Password:       <Type realtime password>
$ /usr/bin/whoami
```

```
realtime
$ ppriv -v $$
1600:   pfksh
flags = <none>
        E: file_link_any,...,proc_clock_highres,proc_session,sys_time
        I: file_link_any,...,proc_clock_highres,proc_session,sys_time
        P: file_link_any,...,proc_clock_highres,proc_session,sys_time
        L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,sys_time
```

▼ 실행할 수 있는 권한 있는 명령을 확인하는 방법

사용자가 직접 할당된 권한이 아닌 경우 권한 프로파일을 통해 권한 있는 명령에 액세스를 얻습니다. 권한 프로파일의 명령은 프로파일 셸에서 실행해야 합니다.

1 할당된 권한 프로파일을 확인합니다.

```
% profiles
Audit Review
Console User
Suspend To RAM
Suspend To Disk
Brightness
CPU Power Management
Network Autoconf
Desktop Print Management
Network Wifi Info
Desktop Removable Media User
Basic Solaris User
All
```

2 Audit Review 프로파일에서 권한을 확인합니다.

```
profiles -l
Audit Review

  solaris.audit.read

  /usr/sbin/auditreduce  euid=0
  /usr/sbin/auditstat    euid=0
  /usr/sbin/praudit      euid=0
```

Audit Review 권한 프로파일을 통해 `auditreduce`, `auditstat`, `praudit` 명령을 유효 UID `0`으로 실행하고 `solaris.audit.read` 권한 부여를 할당할 수 있습니다.

예 9–32 역할의 권한 있는 명령 확인

이 예에서 사용자가 할당된 역할을 맡고 권한 프로파일 중 하나에 포함된 명령을 나열합니다.

```
% roles
devadmin
% su - devadmin
```

```
Password:       Type devadmin password
$ profiles -l
Device Security
          /usr/bin/kbd        uid=0;gid=sys
          /usr/sbin/add_allocatable    euid=0
          /usr/sbin/add_drv        uid=0
          /usr/sbin/devfsadm        uid=0
          /usr/sbin/eeprom        uid=0
          /usr/sbin/list_devices        euid=0
          /usr/sbin/rem_drv        uid=0
          /usr/sbin/remove_allocatable    euid=0
          /usr/sbin/strace        euid=0
          /usr/sbin/update_drv        uid=0
```

예 9–33 역할의 권한 있는 명령 실행

이 예에서 admin 역할이 useful.script 파일에 대한 사용 권한을 변경할 수 있습니다.

```
% whoami
jdoe
% ls -l useful.script
-rwxr-xr-- 1 elsee eng 262 Apr 2 10:52 useful.script
chgrp admin useful.script
chgrp: useful.script: Not owner
% su - admin
Password:       <Type admin password>
$ /usr/bin/whoami
admin
$ chgrp admin useful.script
$ chown admin useful.script
$ ls -l useful.script
-rwxr-xr-- 1 admin admin 262 Apr 2 10:53 useful.script
```

권한 관리(작업 맵)

사용자 및 역할에 대한 권한을 관리하는 가장 안전한 방법은 권한 프로파일의 명령에 권한을 사용하도록 구성하는 것입니다. 그러면 권한 프로파일이 역할에 포함됩니다. 역할이 사용자에 할당됩니다. 사용자가 할당된 역할을 맡을 때 프로파일 셸에서 실행되도록 권한 있는 명령을 사용할 수 있습니다. 다음 절차는 권한을 할당하고 권한을 제거하고 권한 사용을 디버그하는 방법을 보여줍니다.

다음 작업 맵은 권한을 할당, 제거, 디버그하고 권한 있는 명령이 포함된 스크립트를 실행하기 위한 절차를 가리킵니다.

작업	설명	수행 방법
어떤 권한이 프로세스에 있는지 확인합니다.	프로세스에 대한 유효, 상속 가능한, 허가된, 제한 권한 세트를 나열합니다.	191 페이지 "프로세스의 권한을 확인하는 방법"
어떤 권한이 프로세스에서 누락되었는지 확인합니다.	실패한 프로세스를 성공하는 데 필요한 권한을 나열합니다.	192 페이지 "프로그램에 필요한 권한을 확인하는 방법"
명령에 권한을 추가합니다.	권한 프로파일의 명령에 권한을 추가합니다. 사용자 또는 역할을 권한 프로파일에 할당할 수 있습니다. 그런 다음 사용자가 프로파일 셸에서 할당된 권한을 가진 명령을 실행할 수 있습니다.	예 9–14
사용자 또는 역할에 권한을 할당합니다.	사용자나 역할의 상속 가능한 권한 세트를 확장합니다. 이 절차는 주의해서 사용해야 합니다.	예 9–24
사용자의 권한을 제한합니다.	사용자의 기본 권한 세트를 제한합니다. 이 절차는 주의해서 사용해야 합니다.	예 9–12
권한 있는 셸 스크립트를 실행합니다.	셸 스크립트와 셸 스크립트의 명령에 권한을 추가합니다. 그런 다음, 프로파일 셸에서 스크립트를 실행합니다.	194 페이지 "권한 있는 명령으로 셸 스크립트를 실행하는 방법"

▼ 프로세스의 권한을 확인하는 방법

이 절차는 프로세스에 사용 가능한 권한을 확인하는 방법을 보여줍니다. 특정 명령에 할당된 권한은 목록에 포함되지 않습니다.

- **셸 프로세스에 사용 가능한 권한을 나열합니다.**

```
% ppriv pid
$ ppriv -v pid
```

pid 프로세스 번호입니다. 부모 셸의 프로세스 번호를 명령에 전달하려면 이중 달러 기호(`$$`)를 사용합니다.

`-v` 권한 이름의 상세 정보 목록을 제공합니다.

예 9–34 현재 셸의 권한 확인

다음 예에서 사용자의 셸 프로세스에서 부모 프로세스의 권한이 나열됩니다. 두번째 예에서 권한의 전체 이름이 나열됩니다. 출력의 단문자는 다음 권한 세트를 가리킵니다.

`E` 유효 권한 세트입니다.

`I` 상속 가능한 권한 세트입니다.

`P` 허가된 권한 세트입니다.

`L` 제한 권한 세트입니다.

```
% ppriv $$
1200:   -csh
flags = <none>
        E: basic
        I: basic
        P: basic
        L: all
% ppriv -v $$
1200:   -csh
flags = <none>
        E: file_link_any,net_access,proc_exec,proc_fork,proc_info,proc_session
        I: file_link_any,net_access,proc_exec,proc_fork,proc_info,proc_session
        P: file_link_any,net_access,proc_exec,proc_fork,proc_info,proc_session
        L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,sys_time
```

예 9–35 사용자가 맡을 수 있는 역할의 권한 확인

역할은 관리 셸 또는 프로파일 셸을 사용합니다. 역할을 맡고 역할의 셸을 사용하여 역할에 직접 할당된 권한을 나열해야 합니다. 다음 예에서 `sysadmin` 역할에 직접 할당된 권한이 없습니다.

```
% su - sysadmin
Password:      <Type sysadmin password>
$ /usr/bin/whoami
sysadmin
$ ppriv -v $$
1400:   pfksh
flags = <none>
        E: file_link_any,file_read,file_write,net_access,proc_exec,proc_fork,
           proc_info,proc_session
        I: file_link_any,file_read,file_write,net_access,proc_exec,proc_fork,
           proc_info,proc_session
        P: file_link_any,file_read,file_write,net_access,proc_exec,proc_fork,
           proc_info,proc_session
        L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,win_upgrade_sl
```

▼ 프로그램에 필요한 권한을 확인하는 방법

이 절차는 명령이나 프로세스를 성공하는 데 필요한 권한을 확인합니다.

시작하기 전에 이 디버깅 절차가 작동하려면 명령이나 프로세스를 실패해야 합니다.

1 ppriv 디버깅 명령의 인수로 실패하는 명령을 입력합니다.

```
% ppriv -eD touch /etc/acct/yearly
touch[5245]: missing privilege "file_dac_write"
     (euid = 130, syscall = 224) needed at zfs_zaccess+0x258
touch: cannot create /etc/acct/yearly: Permission denied
```

2 **/etc/name_to_sysnum 파일에서 syscall 번호를 찾아서 어떤 시스템 호출이 실패하는지 확인합니다.**

```
% grep 224 /etc/name_to_sysnum
creat64                 224
```

예 9–36 truss 명령을 사용하여 권한 사용 조사

truss 명령은 일반 셸에서 권한 사용을 디버그할 수 있습니다. 예를 들어, 다음 명령은 실패하는 touch 프로세스를 디버그합니다.

```
% truss -t creat touch /etc/acct/yearly
creat64("/etc/acct/yearly", 0666)
                        Err#13 EACCES [file_dac_write]
touch: /etc/acct/yearly cannot create
```

확장된 /proc 인터페이스가 truss 출력에 오류 코드 뒤에 누락된 권한을 보고합니다.

예 9–37 ppriv 명령을 사용하여 프로파일 셸의 권한 사용 조사

ppriv 명령은 프로파일 셸에서 권한 사용을 디버그할 수 있습니다. 권한 프로파일을 사용자에 할당하고 권한 프로파일에 권한 포함 명령이 포함된 경우 프로파일 셸에 명령을 입력해야 합니다. 권한 있는 명령을 일반 셸에 입력하면 명령이 권한으로 실행되지 않습니다.

이 예에서 jdoe 사용자는 objadmin 역할을 맡을 수 있습니다. objadmin 역할에는 Object Access Management 권한 프로파일이 포함됩니다. 이 권한 프로파일을 통해 objadmin 역할은 objadmin이 소유하지 않은 파일에 대한 사용 권한을 변경할 수 있습니다.

다음 발췌 부분에서 jdoe가 useful.script 파일에 대한 사용 권한 변경을 실패합니다.

```
jdoe% ls -l useful.script
-rw-r--r--  1 aloe  staff  2303 Apr 10 10:10 useful.script
jdoe% chown objadmin useful.script
chown: useful.script: Not owner
jdoe% ppriv -eD chown objadmin useful.script
chown[11444]: missing privilege "file_chown"
            (euid = 130, syscall = 16) needed at zfs_zaccess+0x258
chown: useful.script: Not owner
```

jdoe가 objadmin 역할을 맡을 때 파일에 대한 사용 권한이 변경됩니다.

```
jdoe% su - objadmin
Password:      <Type objadmin password>
$ ls -l useful.script
-rw-r--r--  1 aloe  staff  2303 Apr 10 10:10 useful.script
$ chown objadmin useful.script
$ ls -l useful.script
-rw-r--r--  1 objadmin  staff  2303 Apr 10 10:10 useful.script
```

```
$ chgrp admin useful.script
$ ls -l objadmin.script
-rw-r--r--  1 objadmin  admin  2303 Apr 10 10:11 useful.script
```

예 9–38 root 사용자가 소유한 파일 변경

이 예는 권한 에스컬레이션에 대한 보호 조치를 보여줍니다. 설명은 208 페이지 "권한 에스컬레이션 금지"를 참조하십시오. 파일은 root 사용자가 소유합니다. 비교적 덜 강력한 역할인 objadmin이 파일 소유권을 변경하려면 모든 권한이 필요하므로 작업을 실패합니다.

```
jdoe% su - objadmin
Password:      <Type objadmin password>
$ cd /etc; ls -l system
-rw-r--r--  1 root  sys   1883 Oct 10 10:20 system
$ chown objadmin system
chown: system: Not owner
$ ppriv -eD chown objadmin system
chown[11481]: missing privilege "ALL"
     (euid = 101, syscall = 16) needed at zfs_zaccess+0x258
chown: system: Not owner
```

▼ 권한 있는 명령으로 셸 스크립트를 실행하는 방법

주 – 권한이 필요한 명령을 실행하는 셸 스크립트를 만들 때 적절한 권한 프로파일에 할당된 권한을 가진 명령을 포함해야 합니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 첫번째 라인에서 /bin/pfsh 또는 다른 프로파일 셸로 스크립트를 시작합니다.

```
#!/bin/pfsh
# Copyright (c) 2011 by Oracle
```

2 스크립트의 명령에 필요한 권한을 확인합니다.

```
% ppriv -eD script-full-path
```

3 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

4 **스크립트에 대한 권한 프로파일을 만들거나 수정합니다.**

셸 스크립트와 셸 스크립트의 명령을 필요한 보안 속성으로 권한 프로파일에 추가해야 합니다. 단계는 170 페이지 “감사 프로파일을 만들거나 변경하는 방법”을 참조하십시오.

5 **권한 프로파일을 역할에 추가하고 역할을 사용자에 할당합니다.**

스크립트를 실행하려면 사용자가 역할을 맡고 역할의 프로파일 셸에서 스크립트를 실행합니다.

- 권한 프로파일을 역할에 추가하려면 178 페이지 “역할의 보안 속성을 변경하는 방법”을 참조하십시오.
- 사용자에게 역할을 지정하려면 예 9–20을 참조하십시오.

◆ ◆ ◆ 10장

10 Oracle Solaris의 보안 속성(참조)

이 장에서는 RBAC 및 권한에 대한 참조 자료를 제공합니다. 다음은 이 장에 포함된 참조 정보 목록입니다.

- 197 페이지 "권한 프로파일"
- 199 페이지 "지정된 보안 속성의 검색 순서"
- 200 페이지 "인증"
- 201 페이지 "RBAC 데이터베이스"
- 204 페이지 "RBAC 명령"
- 206 페이지 "권한 처리용 관리 명령"
- 207 페이지 "권한 정보 포함 파일"
- 207 페이지 "권한 및 감사"
- 208 페이지 "권한 에스컬레이션 금지"
- 209 페이지 "레거시 응용 프로그램 및 권한 모델"

RBAC 사용에 대한 내용은 9 장, "역할 기반 액세스 제어 사용(작업)"을 참조하십시오. 개요 정보는 135 페이지 "역할 기반 액세스 제어(개요)"를 참조하십시오.

권한을 사용하려면 186 페이지 "권한 사용(작업)"을 참조하십시오. 개요 정보는 146 페이지 "권한(개요)"을 참조하십시오.

권한 프로파일

이 절은 일반적인 권한 프로파일을 설명합니다. 권한 프로파일은 인증 및 기타 보안 속성, 보안 속성 포함 명령, 보충 권한 프로파일을 간편하게 모은 것입니다. Oracle Solaris는 많은 권한 프로파일을 제공합니다. 이들이 사용자 요구를 충족하지 않으면 기존 것을 수정하여 새로 만들 수 있습니다.

가장 강력한 권한 프로파일에서 가장 약한 순으로 지정되어야 합니다. 자세한 내용은 199 페이지 "지정된 보안 속성의 검색 순서"를 참조하십시오.

- **System Administrator 권한 프로파일** – 보안에 관련되지 않은 대부분의 작업을 수행할 수 있는 프로파일을 제공합니다. 이 프로파일에는 강력한 역할을 만들기 위한 여러 다른 프로파일이 포함됩니다. All 권한 프로파일은 보충 권한 프로파일 목록 끝에 지정됩니다. `profiles` 명령은 프로파일의 내용을 표시합니다.

  ```
  % profiles -p "System Administrator" info
  ```

- **Operator 권한 프로파일** – 파일 및 오프라인 매체를 관리하기 위한 제한된 능력을 제공합니다. 이 프로파일에는 단순한 역할을 만들기 위한 보충 권한 프로파일이 포함됩니다. `profiles` 명령은 프로파일의 내용을 표시합니다.

  ```
  % profiles -p Operator info
  ```

- **Printer Management 권한 프로파일** – 인쇄를 처리하기 위한 제한된 수의 명령 및 인증을 제공합니다. 이 프로파일은 관리 분야를 다루는 여러 프로파일 중 하나입니다. `profiles` 명령은 프로파일의 내용을 표시합니다.

  ```
  % profiles -p "Printer Management" info
  ```

- **Basic Solaris User 권한 프로파일** – 사용자가 보안 정책의 한도 내에서 시스템을 사용할 수 있습니다. 이 프로파일은 `policy.conf` 파일에 기본적으로 나열됩니다. Basic Solaris User 권한 프로파일에서 제공하는 편의성은 사이트 보안 요구 사항과 균형을 이루어야 합니다. 더 엄격한 보안이 필요한 사이트는 `policy.conf` 파일에서 이 프로파일을 제거하거나 Stop 권한 프로파일을 지정하는 것이 좋습니다. `profiles` 명령은 프로파일의 내용을 표시합니다.

  ```
  % profiles -p "Basic Solaris User" info
  ```

- **Console User 권한 프로파일** – 워크스테이션 소유자에 대해 컴퓨터에 앉은 사람에게 인증, 명령 및 작업 액세스를 제공합니다. `profiles` 명령은 프로파일의 내용을 표시합니다.

  ```
  % profiles -p "Console User" info
  ```

- **All 권한 프로파일** – 역할에 대해 보안 속성이 없는 명령에 액세스를 제공합니다. 이 프로파일은 제한된 권한을 가진 사용자에 적합할 수 있습니다. `profiles` 명령은 프로파일의 내용을 표시합니다.

  ```
  % profiles -p All info
  ```

- **Stop 권한 프로파일** – 더 이상의 프로파일 평가를 중지하는 특수한 권한 프로파일입니다. 이 프로파일은 `policy.conf` 파일에서 `AUTHS_GRANTED`, `PROFS_GRANTED`, `CONSOLE_USER` 변수의 평가를 금지합니다. 이 프로파일로 역할 및 사용자에게 제한된 프로파일 셸을 제공할 수 있습니다.

주 – Stop 프로파일은 권한 지정에 간접적인 영향을 미칩니다. Stop 프로파일 후에 나열된 권한 프로파일은 평가되지 않습니다. 따라서 이러한 프로파일의 권한을 가진 명령은 효력이 없습니다. 이 프로파일을 사용하려면 182 페이지 “관리자를 명시적으로 할당된 권한으로 제한하는 방법”을 참조하십시오.

`profiles` 명령은 프로파일의 내용을 표시합니다.

```
% profiles -p Stop info
```

각 권한 프로파일에는 연관된 도움말 파일이 있습니다. 도움말 파일은 HTML 형식이고 사용자 정의할 수 있습니다. 파일은 /usr/lib/help/profiles/locale/C 디렉토리에 상주합니다.

권한 프로파일의 내용 보기

권한 프로파일의 내용을 보는 세 가지 방법이 있습니다.

- getent 명령으로 시스템에 있는 모든 권한 프로파일의 내용을 볼 수 있습니다. 샘플 출력은 156 페이지 "모든 정의된 보안 속성을 보는 방법"을 참조하십시오.
- profiles -p "*Profile Name*" info 명령으로 특정 권한 프로파일의 내용을 볼 수 있습니다.
- profiles -l *account-name* 명령으로 특정 사용자나 역할에 지정된 권한 프로파일의 내용을 볼 수 있습니다.

자세한 내용은 getent(1M) 및 profiles(1) 매뉴얼 페이지를 참조하십시오.

지정된 보안 속성의 검색 순서

직접 또는 권한 프로파일을 통해 사용자나 역할에 보안 속성을 지정할 수 있습니다. 검색 순서는 사용되는 보안 속성 값에 영향을 미칩니다. 첫번째 발견된 속성 인스턴스의 값이 사용됩니다.

주 – 인증 순서는 중요하지 않습니다. 인증은 누적형입니다.

사용자가 로그인할 때 다음 검색 순서대로 보안 속성이 지정됩니다.

- useradd 및 usermod 명령으로 사용자에 지정된 **보안 속성**. 목록은 202 페이지 "user_attr 데이터베이스"를 참조하십시오.
- useradd 및 usermod 명령으로 사용자에 지정된 **권한 프로파일**. 이러한 지정 항목은 순차적으로 검색됩니다.

  목록의 첫번째 프로파일과 해당 권한 프로파일 목록, 목록의 두번째 프로파일과 해당 프로파일 목록 등의 순서입니다. auths 값(누적형)을 제외하면, 값의 첫번째 인스턴스는 시스템이 사용하는 것입니다. 권한 프로파일의 속성에는 사용자와 보충 프로파일의 모든 보안 속성이 포함됩니다. 목록은 202 페이지 "user_attr 데이터베이스"를 참조하십시오.
- **Console User 권한 프로파일** 값. 설명은 197 페이지 "권한 프로파일"을 참조하십시오.

- **Stop 권한 프로파일**이 지정된 경우 보안 속성의 평가가 중지됩니다. Stop 프로파일이 지정된 후에는 어떤 속성도 지정되지 않습니다. Stop 프로파일은 Console User 권한 프로파일을 평가한 후 `policy.conf` 파일의 기타 보안 속성(`AUTHS_GRANTED` 포함)을 평가하기 전에 평가됩니다. 설명은 197 페이지 "권한 프로파일"을 참조하십시오.
- `policy.conf` 파일의 **Basic Solaris User 권한 프로파일** 값.
- `policy.conf` 파일의 **AUTHS_GRANTED** 값.
- `policy.conf` 파일의 **PROFS_GRANTED** 값.
- `policy.conf` 파일의 **PRIV_DEFAULT** 값.
- `policy.conf` 파일의 **PRIV_LIMIT** 값.

인증

RBAC **인증**은 역할이나 사용자에 부여할 수 있는 별개의 권한입니다. 사용자가 응용 프로그램이나 그 안의 특정 작업에 액세스를 얻기 전에 RBAC 호환 응용 프로그램에서 인증을 검사합니다.

인증은 사용자 레벨이므로 확장 가능합니다. 인증이 필요한 프로그램을 작성하고, 인증을 시스템에 추가하고, 이러한 인증에 대한 권한 프로파일을 만들고, 프로그램 사용이 허용된 사용자나 역할에 권한 프로파일을 지정할 수 있습니다.

인증 이름 지정 규약

인증은 내부적으로 사용되는 이름이 있습니다. 예를 들어, `solaris.system.date`는 인증의 이름입니다. 인증에는 그래픽 사용자 인터페이스(GUI)에 나타나는 간단한 설명이 있습니다. 예를 들어, `Set Date & Time`은 `solaris.system.date` 인증의 설명입니다.

규약상 인증 이름은 인터넷 공급자 이름, 주제 영역, 하위 영역, 기능의 역순으로 구성됩니다. 인증 이름의 부분은 점으로 구분됩니다. 그 예로 `com.xyzcorp.device.access`가 있습니다. 이 규약의 예외는 Oracle Solaris의 인증으로, 인터넷 이름 대신 접두어 `solaris`를 사용합니다. 이름 지정 규약에 따라 관리자는 계층적 방식으로 인증을 적용할 수 있습니다. 와일드카드 문자(*)는 점 오른쪽의 문자열을 나타낼 수 있습니다.

인증 세분성의 예

인증 사용 방법의 예로 다음을 고려해 보십시오. Network Link Security 역할의 사용자가 `solaris.network.link.security` 인증으로 제한되고, Network Security 역할에 보충 프로파일로 Network Link Security 권한 프로파일이 있고 `solaris.network.*` 및 `solaris.smf.manage.ssh` 인증이 있습니다.

인증의 위임 기관

접미어 delegate로 끝나는 인증을 통해 사용자나 역할이 동일한 접두어로 시작하는 지정된 인증을 다른 사용자에게 위임할 수 있습니다.

solaris auth.delegate 인증은 이러한 사용자나 역할에 지정된 인증을 다른 사용자에게 위임할 수 있습니다.

예를 들어, solaris auth.delegate 및 solaris.network.wifi.wep 인증을 가진 역할은 solaris.network.wifi.wep 인증을 다른 사용자나 역할에 위임할 수 있습니다. 마찬가지로, solaris auth.delegate 및 solaris.network.wifi.wep 인증을 가진 역할은 solaris.network.wifi.wep 인증을 다른 사용자나 역할에 위임할 수 있습니다.

RBAC 데이터베이스

다음 데이터베이스는 RBAC 요소에 대한 데이터를 저장합니다.

- **확장된 사용자 속성 데이터베이스**(user_attr) – 사용자와 역할을 인증, 권한, 키워드 및 권한 프로파일과 연관시킵니다.
- **권한 프로파일 속성 데이터베이스**(prof_attr) – 권한 프로파일을 정의하고 프로파일의 지정된 인증, 권한, 키워드를 나열하고 연관된 도움말 파일을 식별합니다.
- **인증 속성 데이터베이스**(auth_attr) – 인증과 해당 속성을 정의하고 연관된 도움말 파일을 식별합니다.
- **실행 속성 데이터베이스**(exec_attr) – 특정 권한 프로파일에 지정된 보안 속성 포함 명령을 식별합니다.

policy.conf 데이터베이스는 모든 사용자에 적용된 인증, 권한 및 권한 프로파일을 포함합니다. 자세한 내용은 203 페이지 “policy.conf 파일”을 참조하십시오.

RBAC 데이터베이스 및 이름 지정 서비스

RBAC 데이터베이스의 이름 서비스 범위는 이름 지정 서비스 스위치 svc:/system/name-service/switch에 대한 SMF 서비스에 정의됩니다. RBAC 데이터베이스에 대한 이 서비스의 등록 정보는 auth_attr, password, prof_attr입니다. password 등록 정보는 passwd 및 user_attr 데이터베이스에 대한 이름 지정 서비스 우선 순위를 설정합니다. prof_attr 등록 정보는 prof_attr 및 exec_attr 데이터베이스에 대한 이름 지정 서비스 우선 순위를 설정합니다.

다음 출력에서 auth_attr, password, prof_attr 항목이 나열되지 않습니다. 따라서 RBAC 데이터베이스는 files 이름 지정 서비스를 사용하고 있습니다.

```
# svccfg -s name-service/switch listprop config
config                          application
config/value_authorization      astring       solaris.smf.value.name-service.switch
```

```
config/default                  astring       files
config/host                     astring       "files ldap dns"
config/printer                  astring       "user files ldap"
```

user_attr 데이터베이스

user_attr 데이터베이스는 passwd 및 shadow 데이터베이스를 보충하는 사용자 및 역할 정보를 포함합니다.

다음 보안 속성은 roleadd, rolemod, useradd, usermod, profiles 명령으로 설정할 수 있습니다.

- 사용자의 경우 roles 키워드가 하나 이상의 정의된 역할을 지정합니다.
- 역할의 경우 roleauth 키워드의 user 값을 사용하여 역할 암호가 아닌 사용자 암호로 인증할 수 있습니다. 기본적으로 값은 role입니다.
- 사용자 또는 역할의 경우 다음 속성을 설정할 수 있습니다.
  - audit_flags 키워드 - 감사 마스크를 수정합니다. audit_flags(5) 매뉴얼 페이지를 참조하십시오.
  - auths 키워드 - 인증을 지정합니다. auths(1) 매뉴얼 페이지를 참조하십시오.
  - defaultpriv 키워드 - 기본값의 기본 권한 세트에서 권한을 추가하거나 제거합니다. 150 페이지 "권한이 구현되는 방법"을 참조하십시오.
  - limitpriv 키워드 - 기본값의 제한 권한 세트에서 권한을 추가하거나 제거합니다. 150 페이지 "권한이 구현되는 방법"을 참조하십시오.

    이러한 권한은 항상 유효하며, 명령의 속성이 아닙니다. privileges(5) 매뉴얼 페이지와 150 페이지 "권한이 구현되는 방법"을 참조하십시오.
  - projects 키워드 - 기본 프로젝트를 추가합니다. project(4) 매뉴얼 페이지를 참조하십시오.
  - lock_after_retries 키워드 - 값이 yes인 경우 재시도 횟수가 /etc/default/login 파일에 허용된 수를 초과하면 시스템이 잠깁니다.
  - profiles 키워드 - 권한 프로파일을 지정합니다.

자세한 내용은 user_attr(4) 매뉴얼 페이지를 참조하십시오. 이 데이터베이스의 내용을 보려면 getent user_attr 명령을 사용하십시오. 자세한 내용은 getent(1M) 매뉴얼 페이지와 156 페이지 "모든 정의된 보안 속성을 보는 방법"을 참조하십시오.

auth_attr 데이터베이스

모든 인증은 auth_attr 데이터베이스에 저장됩니다. 인증은 사용자, 역할 또는 권한 프로파일에 지정할 수 있습니다. 선호 방법은, 권한 프로파일에 인증을 배치하고 역할의 프로파일 목록에 프로파일을 포함한 후 역할을 사용자에 지정하는 것입니다.

이 데이터베이스의 내용을 보려면 getent prof_attr 명령을 사용하십시오. 자세한 내용은 getent(1M) 매뉴얼 페이지와 156 페이지 "모든 정의된 보안 속성을 보는 방법"을 참조하십시오.

prof_attr 데이터베이스

prof_attr 데이터베이스는 권한 프로파일에 지정된 이름, 설명, 도움말 파일 위치, 권한 및 인증을 저장합니다. 권한 프로파일에 지정된 명령 및 보안 속성은 exec_attr 데이터베이스에 저장됩니다. 자세한 내용은 203 페이지 "exec_attr 데이터베이스"를 참조하십시오.

자세한 내용은 prof_attr(4) 매뉴얼 페이지를 참조하십시오. 이 데이터베이스의 내용을 보려면 getent exec_attr 명령을 사용하십시오. 자세한 내용은 getent(1M) 매뉴얼 페이지와 156 페이지 "모든 정의된 보안 속성을 보는 방법"을 참조하십시오.

exec_attr 데이터베이스

exec_attr 데이터베이스는 성공을 위해 보안 속성이 필요한 명령을 정의합니다. 명령은 권한 프로파일의 일부입니다. 보안 속성 포함 명령은 프로파일을 지정받은 역할이나 사용자가 실행할 수 있습니다.

자세한 내용은 exec_attr(4) 매뉴얼 페이지를 참조하십시오. 이 데이터베이스의 내용을 보려면 getent 명령을 사용하십시오. 자세한 내용은 getent(1M) 매뉴얼 페이지와 156 페이지 "모든 정의된 보안 속성을 보는 방법"을 참조하십시오.

policy.conf 파일

policy.conf 파일은 특정 권한 프로파일, 특정 인증, 특정 권한을 모든 사용자에 부여하는 방법을 제공합니다. 파일의 관련 항목은 **키=값** 쌍으로 구성됩니다.

- AUTHS_GRANTED=*authorizations* – 하나 이상의 인증을 가리킵니다.
- PROFS_GRANTED=*rights profiles* – 하나 이상의 권한 프로파일을 가리킵니다.
- CONSOLE_USER=Console User – Console User 권한 프로파일을 가리킵니다. 이 프로파일은 콘솔 사용자를 위한 편리한 인증 세트와 함께 제공됩니다. 이 프로파일을 사용자 정의할 수 있습니다. 프로파일 내용을 보려면 197 페이지 "권한 프로파일"을 참조하십시오.
- PRIV_DEFAULT=*privileges* – 하나 이상의 권한을 가리킵니다.
- PRIV_LIMIT=*privileges* – 모든 권한을 가리킵니다.

다음 예는 policy.conf 데이터베이스의 일반적인 값을 보여줍니다.

```
# grep AUTHS /etc/security/policy
AUTHS_GRANTED=solaris.device.cdrw

# grep PROFS /etc/security/policy
PROFS_GRANTED=Basic Solaris User

# grep PRIV /etc/security/policy

#PRIV_DEFAULT=basic
#PRIV_LIMIT=all
```

권한에 대한 자세한 내용은 146 페이지 “권한(개요)”을 참조하십시오.

RBAC 명령

이 절은 RBAC 관리에 사용되는 명령을 나열합니다. 또한 인증으로 액세스를 제어할 수 있는 명령이 나열된 표를 제공합니다.

RBAC를 관리하는 명령

다음 명령은 RBAC 정보를 검색 및 설정합니다.

표 10–1 RBAC 관리 명령

명령 매뉴얼 페이지	설명
auths(1)	사용자에 대한 인증을 표시합니다.
getent(1M)	user_attr, prof_attr, exec_attr 데이터베이스의 내용을 나열하는 인터페이스입니다.
nscd(1M)	이름 서비스 캐시 데몬으로 user_attr, prof_attr, exec_attr 데이터베이스를 캐싱하는 데 유용합니다. 데몬을 다시 시작하려면 svcadm 명령을 사용합니다.
pam_roles(5)	PAM용 역할 계정 관리 모듈입니다. 역할을 맡기 위해 인증을 검사합니다.
pfexec(1)	exec_attr 데이터베이스에 지정된 보안 속성 포함 명령을 실행하기 위해 프로파일 셸에서 사용됩니다.
policy.conf(4)	시스템 보안 정책에 대한 구성 파일입니다. 부여된 인증, 부여된 권한 및 기타 보안 정보를 나열합니다.
profiles(1)	지정된 사용자에 대한 권한 프로파일을 표시합니다. 로컬 시스템 또는 LDAP 네트워크에서 권한 프로파일을 만들거나 수정합니다.
roles(1)	지정된 사용자가 맡을 수 있는 역할을 표시합니다.
roleadd(1M)	로컬 시스템 또는 LDAP 네트워크에 역할을 추가합니다.
roleadd(1M)	로컬 시스템 또는 LDAP 네트워크에 역할을 추가합니다.

표 10–1 RBAC 관리 명령 *(계속)*

명령 매뉴얼 페이지	설명
rolemod(1M)	로컬 시스템 또는 LDAP 네트워크에서 역할의 등록 정보를 수정합니다.
userattr(1)	사용자나 역할 계정에 지정된 특정 권한의 값을 표시합니다.
useradd(1M)	시스템 또는 LDAP 네트워크에 사용자 계정을 추가합니다. -R 옵션은 사용자의 계정에 역할을 지정합니다.
userdel(1M)	시스템 또는 LDAP 네트워크에서 사용자 로그인을 삭제합니다.
usermod(1M)	시스템에서 사용자의 계정 등록 정보를 수정합니다.

인증이 필요한 선택된 명령

다음 표는 Oracle Solaris 시스템에서 명령 옵션을 제한하기 위해 인증이 사용되는 방법의 예를 제공합니다. 자세한 인증 설명은 200 페이지 “인증”을 참조하십시오.

표 10–2 명령 및 연관된 인증

명령 매뉴얼 페이지	인증 요구 사항
at(1)	solaris.jobs.user - 모든 옵션에 필요합니다(at.allow 파일도 at.deny 파일도 존재하지 않을 때).
atq(1)	solaris.jobs.admin - 모든 옵션에 필요합니다.
cdrw(1)	solaris.device.cdrw - 모든 옵션에 필요하고 policy.conf 파일에 기본적으로 부여됩니다.
crontab(1)	solaris.jobs.user - 작업을 제출하는 옵션에 필요합니다(crontab.allow 파일도 crontab.deny 파일도 존재하지 않을 때). solaris.jobs.admin - 다른 사용자의 crontab 파일을 나열/수정하는 옵션에 필요합니다.
allocate(1)	solaris.device.allocate(또는 기타 device_allocate 파일에 지정된 인증) - 장치를 할당하려면 필요합니다. solaris.device.revoke(또는 기타 device_allocate 파일에 지정된 인증) - 다른 사용자에 장치를 할당하려면 필요합니다(-F 옵션).
deallocate(1)	solaris.device.allocate(또는 기타 device_allocate 파일에 지정된 인증) - 다른 사용자의 장치 할당을 해제하려면 필요합니다. solaris.device.revoke(또는 기타 device_allocate 파일에 지정된 인증) - 지정된 장치(-F 옵션) 또는 모든 장치(-I 옵션)를 강제 할당 해제하려면 필요합니다.
list_devices(1)	solaris.device.revoke - 다른 사용자의 장치를 나열하려면 필요합니다(-U 옵션).

표 10–2 명령 및 연관된 인증 *(계속)*

명령 매뉴얼 페이지	인증 요구 사항
roleadd(1M)	solaris.user.manage - 역할을 만들려면 필요합니다. solaris.account.activate - 초기 암호를 설정하려면 필요합니다. solaris.account.setpolicy - 계정 잠금 및 암호 에이징과 같은 암호 정책을 설정하려면 필요합니다.
roledel(1M)	solaris.passwd.assign 인증 - 암호를 삭제하려면 필요합니다.
rolemod(1M)	solaris.passwd.assign 인증 - 암호를 변경하려면 필요합니다. solaris.account.setpolicy - 계정 잠금 및 암호 에이징과 같은 암호 정책을 변경하려면 필요합니다.
sendmail(1M)	solaris.mail - 메일 부속 시스템 기능에 액세스하려면 필요합니다. solaris.mail.mailq - 메일 대기열을 보려면 필요합니다.
useradd(1M)	solaris.user.manage - 사용자를 만들려면 필요합니다. solaris.account.activate - 초기 암호를 설정하려면 필요합니다. solaris.account.setpolicy - 계정 잠금 및 암호 에이징과 같은 암호 정책을 설정하려면 필요합니다.
userdel(1M)	solaris.passwd.assign 인증 - 암호를 삭제하려면 필요합니다.
usermod(1M)	solaris.passwd.assign 인증 - 암호를 변경하려면 필요합니다. solaris.account.setpolicy - 계정 잠금 및 암호 에이징과 같은 암호 정책을 변경하려면 필요합니다.

권한

권한 제약 프로세스는 커널에서 구현되며 명령, 사용자, 역할, 시스템 레벨에서 프로세스를 제약할 수 있습니다.

권한 처리용 관리 명령

다음 표는 권한 처리에 사용 가능한 명령을 나열합니다.

표 10–3 권한 처리용 명령

목적	명령	매뉴얼 페이지
프로세스 권한 조사	ppriv -v *pid*	ppriv(1)
프로세스 권한 설정	ppriv -s *spec*	
시스템에 권한 나열	ppriv -l	
권한 및 해당 설명 나열	ppriv -lv *priv*	
권한 실패 디버그	ppriv -eD *failed-operation*	

표 10–3 권한 처리용 명령 (계속)

목적	명령	매뉴얼 페이지
새 사용자에 권한 지정	useradd	useradd(1M)
기존 사용자에 권한 추가	usermod	usermod(1M)
권한 프로파일에 권한 지정	profiles	profiles(1)
새 역할에 권한 지정	roleadd	roleadd(1M)
기존 역할에 권한 추가	rolemod	rolemod(1M)
장치 정책 보기	getdevpolicy	getdevpolicy(1M)
장치 정책 설정	devfsadm	devfsadm(1M)
열린 장치에서 장치 정책 업데이트	update_drv -p *policy driver*	update_drv(1M)
장치에 장치 정책 추가	add_drv -p *policy driver*	add_drv(1M)

권한 정보 포함 파일

다음 파일은 권한에 대한 정보를 포함합니다.

표 10–4 권한 정보를 포함하는 파일

파일 및 매뉴얼 페이지	권한 정보	설명
/etc/security/policy.conf	PRIV_DEFAULT	시스템의 상속 가능한 권한 세트
policy.conf(4)	PRIV_LIMIT	시스템의 제한 권한 세트
syslog.conf	디버그 메시지용 시스템 로그 파일	권한 디버깅 로그
syslog.conf(4)	priv.debug 항목에 설정된 경로	

권한 및 감사

권한 사용을 감사할 수 있습니다. 프로세스가 권한을 사용할 때 언제든지 upriv 감사 토큰의 감사 증적에 권한 사용이 기록됩니다. 권한 이름이 레코드의 일부일 때 텍스트 표현이 사용됩니다. 다음 감사 이벤트는 권한 사용을 기록합니다.

- AUE_SETPPRIV **감사 이벤트** – 권한 세트를 변경할 때 감사 레코드를 생성합니다. AUE_SETPPRIV 감사 이벤트는 pm 클래스에 속합니다.
- AUE_MODALLOCPRIV **감사 이벤트** – 커널 밖에서 권한을 추가할 때 감사 레코드를 생성합니다. AUE_MODALLOCPRIV 감사 이벤트는 ad 클래스에 속합니다.
- AUE_MODDEVPLCY **감사 이벤트** – 장치 정책을 변경할 때 감사 레코드를 생성합니다. AUE_MODDEVPLCY 감사 이벤트는 ad 클래스에 속합니다.

- AUE_PFEXEC **감사 이벤트** – pfexec()가 사용으로 설정된 채 execve()를 호출할 때 감사 레코드를 생성합니다. AUE_PFEXEC 감사 이벤트는 as, ex, ps, ua 감사 클래스에 속합니다. 권한의 이름이 감사 레코드에 포함됩니다.

기본 세트에 속하는 권한의 성공적 사용은 감사되지 않습니다. 사용자의 기본 세트에서 제거된 기본 권한을 사용하려는 시도는 감사됩니다.

권한 에스컬레이션 금지

커널은 **권한 에스컬레이션**을 금지합니다. 프로세스가 의도했던 것보다 더 많은 권한을 얻을 때 권한 에스컬레이션이라고 합니다. 프로세스가 의도한 것보다 많은 권한을 얻지 못하도록 하려면 취약한 시스템 개조를 위해 전체 권한 세트가 필요합니다. 예를 들어, root(UID=0)가 소유한 파일/프로세스는 전체 권한 세트를 가진 프로세스만 변경할 수 있습니다. root 계정이 root 소유의 파일을 변경하는 데에는 권한이 필요하지 않습니다. 그러나 비루트 사용자가 root 소유의 파일을 변경하려면 모든 권한이 있어야 합니다.

마찬가지로, 장치 액세스를 제공하는 작업은 유효 세트의 모든 권한이 필요합니다.

file_chown_self 및 proc_owner 권한은 권한 에스컬레이션이 적용됩니다. file_chown_self 권한을 통해 프로세스가 해당 파일을 제공할 수 있습니다. proc_owner 권한을 통해 프로세스가 소유하지 않은 프로세스를 검사할 수 있습니다.

file_chown_self 권한은 rstchown 시스템 변수로 제한됩니다. rstchown 변수를 0으로 설정할 때 file_chown_self 권한이 시스템 및 모든 사용자의 초기 상속 가능한 세트에서 제거됩니다. rstchown 시스템 변수에 대한 자세한 내용은 chown(1) 매뉴얼 페이지를 참조하십시오.

file_chown_self 권한은 가장 안전하게 특정 명령에 지정되고, 프로파일에 배치되고, 프로파일 셸에 사용하기 위해 역할에 지정됩니다.

proc_owner 권한은 프로세스 UID를 0으로 전환하기에 충분하지 않습니다. 임의 UID에서 UID=0으로 프로세스를 전환하려면 모든 권한이 필요합니다. proc_owner 권한은 시스템의 모든 파일에 무제한 읽기 액세스를 제공하므로 가장 안전하게 특정 명령에 지정되고, 프로파일에 배치되고, 프로파일 셸에 사용하기 위해 역할에 지정됩니다.

주의 – 사용자의 초기 상속 가능한 세트에 file_chown_self 권한이나 proc_owner 권한을 포함하도록 사용자 계정을 수정할 수 있습니다. 이러한 강력한 권한을 사용자, 역할, 시스템의 상속 가능한 권한 세트에 배치하려면 우선적인 보안 이유가 있어야 합니다.

장치에 대한 권한 에스컬레이션을 금지하는 방법은 153 페이지 “권한 및 장치”를 참조하십시오.

레거시 응용 프로그램 및 권한 모델

레거시 응용 프로그램을 수용하기 위해 수퍼유저 및 권한 모델과 함께 권한 구현이 작동합니다. 커널은 PRIV_AWARE 플래그를 자동으로 추적하여 프로그램이 권한과 연동하도록 설계되었는지 나타냅니다. 권한을 인식하지 못하는 자식 프로세스를 고려하십시오. 부모 프로세스에서 상속한 권한은 자식의 허가된 세트와 유효 세트에서 사용할 수 있습니다. 자식 프로세스가 UID를 0으로 설정하면 전체 수퍼유저 능력을 얻지 못할 수 있습니다. 프로세스의 유효 세트와 허가된 세트는 자식의 제한 세트의 권한으로 제약됩니다. 따라서 권한 인식 프로세스의 제한 세트는 권한을 인식하지 못하는 자식 프로세스의 루트 권한을 제약합니다.

제 4 부

암호화 서비스

이 절에서는 Oracle Solaris에서 제공하는 중앙화된 암호화 및 공개 키 기술 기능에 대해 설명합니다.

◆ ◆ ◆ 11 장

11 암호화 프레임워크(개요)

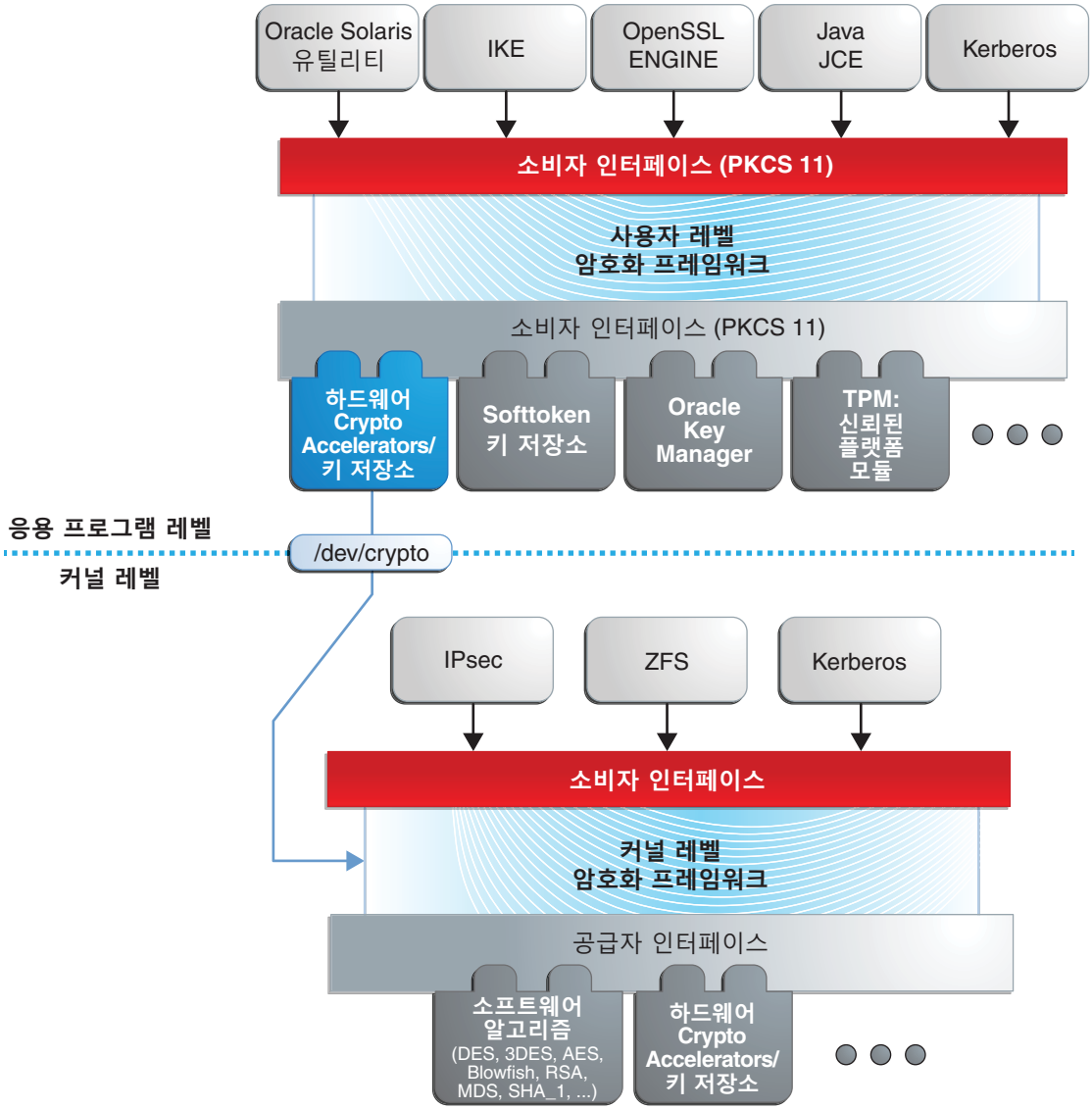
이 장에서는 Oracle Solaris의 암호화 프레임워크 기능을 설명합니다. 다음은 이 장에 포함된 정보 목록입니다.

암호화 프레임워크를 관리하고 사용하려면 12 장, “암호화 프레임워크(작업)”를 참조하십시오.

암호화 프레임워크 소개

암호화 프레임워크는 암호화 요구 사항을 처리하기 위한 알고리즘 및 PKCS #11 라이브러리의 공통 저장소를 제공합니다. PKCS #11 라이브러리는 RSA Security Inc.의 PKCS #11 암호화 토큰 인터페이스(Cryptoki) 표준에 따라 구현됩니다.

그림 11–1 암호화 프레임워크 레벨



커널 레벨에서 프레임워크는 현재 Kerberos 및 IPsec에 대한 암호화 요구 사항을 처리합니다. 사용자 레벨 소비자는 `libsasl` 및 IKE를 포함합니다. 커널 SSL(`kssl`) 프록시가 암호화 프레임워크를 사용합니다. 자세한 내용은 **Oracle Solaris 관리: 네트워크 서비스**의 "웹 서버에서 Secure Sockets Layer 프로토콜 사용" 및 `ksslcfg`(1M) 매뉴얼 페이지를 참조하십시오.

미국 수출법에 따라 개방형 암호화 인터페이스의 사용이 제한되어야 합니다. 암호화 프레임워크는 현행 법규에 맞게 커널 암호화 공급자와 PKCS #11 암호화 공급자의 서명을 요구합니다. 자세한 내용은 218 페이지 “타사 소프트웨어에 대한 이진 서명”을 참조하십시오.

프레임워크에서 암호화 서비스 **공급자**의 서비스를 Oracle Solaris의 많은 **소비자**가 사용할 수 있습니다. 공급자의 다른 이름은 **플러그인**입니다. 다음 세 가지 유형의 플러그인을 허용합니다.

- **사용자 레벨 플러그인** – PKCS #11 라이브러리(예: `pkcs11_softtoken.so.1`)를 사용하여 서비스를 제공하는 공유 객체입니다.
- **커널 레벨 플러그인** – 소프트웨어의 암호화 알고리즘(예: AES)을 구현하는 커널 모듈입니다.

  프레임워크에서 대부분의 알고리즘은 x86(SSE2 명령 세트 포함) 및 SPARC 하드웨어에 맞게 최적화되어 있습니다.
- **하드웨어 플러그인** – 장치 드라이버 및 연관된 하드웨어 가속기입니다. 그 예로 Niagara 칩, ncp 및 n2cp 장치 드라이버가 있습니다. 하드웨어 가속기는 운영 체제에서 값비싼 암호화 작업 부담을 덜어줍니다. 그 예로 Sun Crypto Accelerator 6000 보드가 있습니다.

프레임워크는 사용자 레벨 공급자에 대한 표준 인터페이스인 PKCS #11, v2.11 라이브러리를 구현합니다. 타사 응용 프로그램에서 라이브러리를 사용하여 공급자에 연결할 수 있습니다. 또한 서명된 라이브러리, 서명된 커널 알고리즘 모듈, 서명된 장치 드라이버를 프레임워크에 추가할 수 있습니다. 이러한 플러그인은 `pkgadd` 유틸리티가 타사 소프트웨어를 설치할 때 추가됩니다. 프레임워크의 주요 구성 요소에 대한 도표는 **Developer’s Guide to Oracle Solaris 11 Security**의 8 장, “Introduction to the Oracle Solaris Cryptographic Framework”를 참조하십시오.

암호화 프레임워크의 용어

다음 정의 및 예제 목록은 암호화 프레임워크를 작업할 때 유용합니다.

- **알고리즘** – 암호화 알고리즘입니다. 입력을 암호화하거나 해시하는 확립된 순환적 계산 프로시저입니다. 암호화 알고리즘은 대칭 또는 비대칭일 수 있습니다. 대칭 알고리즘은 암호화 및 해독에 동일한 키를 사용합니다. 비대칭 알고리즘은 공개 키 암호화에 사용되며 두 개의 키가 필요합니다. 해시 함수 역시 알고리즘입니다.

  알고리즘의 예는 다음과 같습니다.

  - 대칭 알고리즘 - AES, ARCFOUR
  - 비대칭 알고리즘 - Diffie-Hellman, RSA
  - 해시 함수 - MD5
- **소비자** – 공급자로부터 전달된 암호화 서비스의 사용자입니다. 소비자는 응용 프로그램, 최종 사용자 또는 커널 작업일 수 있습니다.

소비자의 예는 다음과 같습니다.

- 응용 프로그램 - IKE
- 최종 사용자 - `encrypt` 명령을 실행하는 일반 사용자
- 커널 작업 - IPsec

- **방식** – 특정 목적을 위한 알고리즘 모드의 적용입니다.

  예를 들어, 인증에 적용된 DES 방식(예: CKM_DES_MAC)은 암호화에 적용된 DES 방식(예: CKM_DES_CBC_PAD)과 별도의 방식입니다.

- **Metaslot** – 프레임워크에 로드된 다른 슬롯들의 기능을 결합한 단일 슬롯입니다. metaslot을 사용하면 프레임워크를 통해 사용 가능한 공급자의 기능을 쉽게 다룰 수 있습니다. metaslot을 사용하는 응용 프로그램이 작업을 요청하면 metaslot이 작업을 수행할 실제 슬롯을 알아냅니다. metaslot 기능은 구성 가능하지만, 구성이 필요하지 않습니다. metaslot은 기본적으로 켜져 있습니다. metaslot을 구성하려면 `cryptoadm(1M)` 매뉴얼 페이지를 참조하십시오.
- **모드** – 암호화 알고리즘의 버전입니다. 예를 들어, CBC(Cipher Block Chaining)는 ECB(Electronic Code Book)와 다른 모드입니다. AES 알고리즘에는 CKM_AES_ECB 및 CKM_AES_CBC의 두 가지 모드가 있습니다.
- **정책** – 어떤 방식을 사용할지 관리자가 선택합니다. 기본적으로 모든 공급자와 모든 방식을 사용할 수 있습니다. 어떤 방식을 사용 안함으로 설정하면 정책에 적용됩니다. 사용 안함으로 설정된 방식을 사용으로 설정하면 역시 정책에 적용됩니다.
- **공급자** – 소비자가 사용하는 암호화 서비스입니다. 공급자는 프레임워크에 플러그인되므로 **플러그인**이라고도 합니다.

  공급자의 예는 다음과 같습니다.

  - PKCS #11 라이브러리 - `pkcs11_softtoken.so`
  - 암호화 알고리즘의 모듈 - `aes`, `arcfour`
  - 장치 드라이버 및 연관된 하드웨어 가속기 - Sun Crypto Accelerator 6000용 `mca` 드라이버

- **슬롯** – 하나 이상의 암호화 장치에 대한 인터페이스입니다. 물리적 판독기나 기타 장치 인터페이스에 해당하는 각 슬롯은 토큰을 포함할 수 있습니다. 토큰은 프레임워크의 암호화 장치에 대한 논리적 뷰를 제공합니다.
- **토큰** – 슬롯에서 토큰은 프레임워크의 암호화 장치에 대한 논리적 뷰를 제공합니다.

암호화 프레임워크의 범위

프레임워크는 공급자를 제공하는 관리자용, 사용자용, 개발자용 명령을 제공합니다.

- **관리 명령** – `cryptoadm` 명령은 사용 가능한 공급자와 그 기능을 나열하는 `list` 하위 명령을 제공합니다. 일반 사용자가 `cryptoadm list` 및 `cryptoadm --help` 명령을 실행할 수 있습니다.

  기타 모든 `cryptoadm` 하위 명령을 실행하려면 Crypto Management 권한 프로파일이 포함된 역할을 맡거나 수퍼유저가 되어야 합니다. `disable`, `install`, `uninstall`과 같은 하위 명령을 프레임워크 관리에 사용할 수 있습니다. 자세한 내용은 `cryptoadm`(1M) 매뉴얼 페이지를 참조하십시오.

  `svcadm` 명령을 사용하여 `kcfd` 데몬을 관리하고 커널에서 암호화 정책을 새로 고칠 수 있습니다. 자세한 내용은 `svcadm`(1M) 매뉴얼 페이지를 참조하십시오.
- **사용자 레벨 명령** – `digest` 및 `mac` 명령은 파일 무결성 서비스를 제공합니다. `encrypt` 및 `decrypt` 명령은 도청으로부터 파일을 보호합니다. 이러한 명령을 사용하려면 222 페이지 “암호화 프레임워크로 파일 보호(작업 맵)”를 참조하십시오.

암호화 프레임워크의 관리 명령

`cryptoadm` 명령은 실행 중인 암호화 프레임워크를 관리합니다. 명령은 Crypto Management 권한 프로파일의 일부입니다. 이 프로파일은 암호화 프레임워크의 보안 관리를 위해 역할에 지정할 수 있습니다. `cryptoadm` 명령은 다음을 관리합니다.

- 암호화 공급자 정보 표시
- 공급자 방식 사용 또는 사용 안함
- metaslot 사용 또는 사용 안함

`svcadm` 명령을 사용하여 암호화 서비스 데몬 `kcfd`를 새로 고치고 사용 또는 사용 안함으로 설정할 수 있습니다. 이 명령은 Oracle Solaris의 SMF(서비스 관리 기능) 기능의 일부입니다. `svc:/system/cryptosvcs`는 암호화 프레임워크에 대한 서비스 인스턴스입니다. 자세한 내용은 `smf`(5) 및 `svcadm`(1M) 매뉴얼 페이지를 참조하십시오.

암호화 프레임워크의 사용자 레벨 명령

암호화 프레임워크는 파일 무결성을 검사하고 파일을 암호화하고 파일을 해독하기 위한 사용자 레벨 명령을 제공합니다. 별도의 명령 `elfsign`을 통해 공급자가 프레임워크와 함께 사용할 이진을 서명할 수 있습니다.

- `digest` **명령** – 하나 이상의 파일 또는 stdin에 대한 메시지 다이제스트를 계산합니다. 다이제스트는 파일 무결성 확인에 유용합니다. 다이제스트 함수의 예로 SHA1 및 MD5가 있습니다.

- `mac` **명령** – 하나 이상의 파일 또는 stdin에 대한 MAC(메시지 인증 코드)를 계산합니다. MAC은 데이터를 인증된 메시지와 연관시킵니다. MAC을 사용하여 수신자는 메시지가 발신자로부터 왔는지, 메시지가 변조되지 않았는지 확인할 수 있습니다. `sha1_mac` 및 `md5_hmac` 방식은 MAC을 계산할 수 있습니다.
- `encrypt` **명령** – 파일 또는 stdin을 대칭 암호화합니다. `encrypt -l` 명령은 사용 가능한 알고리즘을 나열합니다. 사용자 레벨 라이브러리 아래에 나열된 방식을 `encrypt` 명령에 사용할 수 있습니다. 프레임워크는 사용자 암호화를 위해 AES, DES, 3DES(3중 DES), ARCFOUR 방식을 제공합니다.
- `decrypt` **명령** – `encrypt` 명령으로 암호화된 파일 또는 stdin을 해독합니다. `decrypt` 명령은 원본 파일을 암호화하는 데 사용된 것과 동일한 키 및 방식을 사용합니다.

타사 소프트웨어에 대한 이진 서명

`elfsign` 명령은 암호화 프레임워크와 함께 사용할 공급자를 서명할 수 있습니다. 일반적으로, 이 명령은 공급자의 개발자가 실행합니다.

`elfsign` 명령에는 인증서를 요청하고 이진을 서명하고 이진에서 서명을 확인하는 하위 명령이 있습니다. 서명되지 않은 이진은 암호화 프레임워크에서 사용할 수 없습니다. 검증 가능한 서명된 이진이 있는 공급자는 프레임워크를 사용할 수 있습니다.

암호화 프레임워크의 플러그인

타사 공급자를 암호화 프레임워크에 플러그인할 수 있습니다. 타사 공급자는 다음 객체 중 하나일 수 있습니다.

- PKCS #11 공유 라이브러리
- 암호화 알고리즘, MAC 함수, 다이제스트 함수와 같은 로드 가능한 커널 소프트웨어 모듈
- 하드웨어 가속기용 커널 장치 드라이버

공급자의 객체를 Oracle의 인증서로 서명해야 합니다. 인증서 요청은 타사가 선택한 개인 키와 Oracle이 제공한 인증서를 기반으로 합니다. 인증서 요청을 Oracle로 보내면 타사를 등록한 후 인증서를 발행합니다. 그런 다음 타사 공급자 객체를 Oracle의 인증서로 서명합니다.

로드 가능한 커널 소프트웨어 모듈과 하드웨어 가속기용 커널 장치 드라이버도 커널에 등록해야 합니다. 등록은 암호화 프레임워크 SPI(서비스 공급자 인터페이스)를 통해 이루어집니다.

암호화 서비스 및 영역

전역 영역과 각 비전역 영역에는 고유의 `/system/cryptosvc` 서비스가 있습니다. 전역 영역에서 암호화 서비스를 사용으로 설정하거나 새로 고치면 `kcfd` 데몬이 전역 영역에서 시작되고, 전역 영역에 대한 사용자 레벨 정책이 설정되고, 시스템에 대한 커널 정책이 설정됩니다. 비전역 영역에서 서비스를 사용으로 설정하거나 새로 고치면 `kcfd` 데몬이 영역에서 시작되고, 영역에 대한 사용자 레벨 정책이 설정됩니다. 커널 정책은 전역 영역에서 설정되었습니다.

영역에 대한 자세한 내용은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 제II부, "Oracle Solaris Zones"를 참조하십시오. 지속성 응용 프로그램을 관리하는 SMF에 대한 자세한 내용은 **Oracle Solaris 관리: 일반 작업**의 6 장, "서비스 관리(개요)" 및 `smf(5)` 매뉴얼 페이지를 참조하십시오.

◆ ◆ ◆ 12 장

12 암호화 프레임워크(작업)

이 장에서는 암호화 프레임워크의 사용 방법을 설명합니다. 다음은 이 장에 포함된 정보 목록입니다.

암호화 프레임워크 사용(작업 맵)

다음 작업 맵은 암호화 프레임워크를 사용하기 위한 작업을 가리킵니다.

작업	설명	수행 방법
개별 파일 또는 파일 세트를 보호합니다.	파일 내용이 변조되지 않았는지 확인합니다. 침입자가 파일을 읽지 못하도록 합니다. 이러한 절차는 일반 사용자가 수행할 수 있습니다.	222 페이지 “암호화 프레임워크로 파일 보호(작업 맵)”
프레임워크를 관리합니다.	소프트웨어 공급자를 추가, 구성, 제거합니다. 하드웨어 공급자 방식을 사용 또는 사용 안함으로 설정합니다. 이러한 절차는 관리 절차입니다.	235 페이지 “암호화 프레임워크 관리(작업 맵)”

암호화 프레임워크로 파일 보호(작업)

이 절에서는 대칭 키를 생성하는 방법, 파일 무결성을 위한 체크섬을 만드는 방법, 도청으로부터 파일을 보호하는 방법을 설명합니다. 이 절의 명령은 일반 사용자가 실행할 수 있습니다. 개발자는 이러한 명령을 사용하는 스크립트를 작성할 수 있습니다.

암호화 프레임워크로 파일 보호(작업 맵)

암호화 프레임워크는 파일을 보호하도록 도울 수 있습니다. 다음 작업 맵은 사용 가능한 알고리즘을 나열하고 암호화 기법으로 파일을 보호하기 위한 절차를 가리킵니다.

작업	설명	수행 방법
대칭 키를 생성합니다.	사용자가 지정한 알고리즘에 사용할 무작위 키를 생성합니다.	222 페이지 “dd 명령을 사용하여 대칭 키를 생성하는 방법”
	사용자가 지정한 길이의 키를 생성합니다. 선택적으로 파일, PKCS #11 키 저장소 또는 NSS 키 저장소에 키를 저장합니다.	224 페이지 “pktool 명령을 사용하여 대칭 키를 생성하는 방법”
파일 무결성을 보장하는 체크섬을 제공합니다.	수신자의 파일 복사본이 보낸 파일과 같은지 확인합니다.	228 페이지 “파일의 다이제스트를 계산하는 방법”
MAC(메시지 인증 코드)으로 파일을 보호합니다.	본인이 발신자임을 메시지 수신자에게 확인합니다.	230 페이지 “파일의 MAC을 계산하는 방법”
파일을 암호화하고, 암호화된 파일을 해독합니다.	파일을 암호화하여 파일 내용을 보호합니다. 파일을 해독하려면 암호화 매개변수를 제공합니다.	232 페이지 “파일을 암호화 및 해독하는 방법”

▼ dd 명령을 사용하여 대칭 키를 생성하는 방법

파일을 암호화하고 파일의 MAC을 생성하려면 키가 필요합니다. 키는 난수 풀에서 파생되어야 합니다.

키를 만들려면 다음 세 가지 옵션이 있습니다.

- 사이트에 난수 생성기가 있는 경우 생성기를 사용합니다.
- 키를 생성하고 저장하려면 224 페이지 “pktool 명령을 사용하여 대칭 키를 생성하는 방법”을 참조하십시오.
- 그렇지 않으면 이 절차를 사용합니다. 이 절차에서는 비트 단위의 키 크기를 제공해야 합니다. 이와 반대로, pktool 명령은 지정한 알고리즘에 따라 올바른 키 크기를 결정합니다.

1 알고리즘에 필요한 키 길이를 결정합니다.

a. 사용 가능한 알고리즘을 나열합니다.

```
% encrypt -l
Algorithm       Keysize:  Min   Max (bits)
----------------------------------------
aes                       128   128
arcfour                     8   128
des                        64    64
```

```
3des                        192   192

% mac -l
Algorithm       Keysize:  Min   Max (bits)
------------------------------------------
des_mac                    64    64
sha1_hmac                   8   512
md5_hmac                    8   512
sha256_hmac                 8   512
sha384_hmac                 8  1024
sha512_hmac                 8  1024
```

b. dd 명령에 전달할 바이트 단위의 키 길이를 결정합니다.

최소 및 최대 키 크기를 8로 나눕니다. 최소 및 최대 키 크기가 서로 다른 경우 중간 키 크기가 가능합니다. 예를 들어, `sha1_hmac` 및 `md5_hmac` 함수에 대해 `dd` 명령에 8, 16, 64 값을 전달할 수 있습니다.

2 대칭 키를 생성합니다.

```
% dd if=/dev/urandom of=keyfile bs=n count=n
```

`if=`*file* 입력 파일입니다. 무작위 키의 경우 `/dev/urandom` 파일을 사용합니다.

`of=`*keyfile* 생성된 키를 보유하는 출력 파일입니다.

`bs=`*n* 바이트 단위의 키 크기입니다. 길이를 바이트로 표시하려면 비트 키 길이를 8로 나눕니다.

`count=`*n* 입력 블록의 수입니다. *n*의 숫자는 1이어야 합니다.

3 보호된 디렉토리에 키를 저장합니다.

사용자 이외의 다른 사람이 키 파일을 읽을 수 없어야 합니다.

```
% chmod 400 keyfile
```

예 12–1 AES 알고리즘의 키 만들기

다음 예에서 AES 알고리즘의 보안 키를 만듭니다. 또한 나중에 해독을 위해 키를 저장합니다. AES 방식은 128비트 키를 사용합니다. `dd` 명령에서 키가 16바이트로 표현됩니다.

```
% ls -al ~/keyf
drwx------   2 jdoe  staff        512 May 3 11:32 ./
% dd if=/dev/urandom of=$HOME/keyf/05.07.aes16 bs=16  count=1
% chmod 400 ~/keyf/05.07.aes16
```

예 12–2 DES 알고리즘의 키 만들기

다음 예에서 DES 알고리즘의 보안 키를 만듭니다. 또한 나중에 해독을 위해 키를 저장합니다. DES 방식은 64비트 키를 사용합니다. `dd` 명령에서 키가 8바이트로 표현됩니다.

```
% dd if=/dev/urandom of=$HOME/keyf/05.07.des8 bs=8  count=1
% chmod 400 ~/keyf/05.07.des8
```

예 12–3 3DES 알고리즘의 키 만들기

다음 예에서 3DES 알고리즘의 보안 키를 만듭니다. 또한 나중에 해독을 위해 키를 저장합니다. 3DES 방식은 192비트 키를 사용합니다. dd 명령에서 키가 24바이트로 표현됩니다.

```
% dd if=/dev/urandom of=$HOME/keyf/05.07.3des.24 bs=24 count=1
% chmod 400 ~/keyf/05.07.3des.24
```

예 12–4 MD5 알고리즘의 키 만들기

다음 예에서 MD5 알고리즘의 보안 키를 만듭니다. 또한 나중에 해독을 위해 키를 저장합니다. dd 명령에서 키가 64바이트로 표현됩니다.

```
% dd if=/dev/urandom of=$HOME/keyf/05.07.mack64 bs=64 count=1
% chmod 400 ~/keyf/05.07.mack64
```

▼ pktool 명령을 사용하여 대칭 키를 생성하는 방법

일부 응용 프로그램에서 통신을 암호화 및 해독하려면 대칭 키가 필요합니다. 이 절차에서 대칭 키를 만들고 저장합니다.

- 사이트에 난수 생성기가 있는 경우 생성기를 사용하여 키에 대한 난수를 생성할 수 있습니다. 이 절차에서는 사이트의 난수 생성기를 사용하지 않습니다.
- 대신, dd 명령을 /dev/urandom 장치와 함께 입력으로 사용할 수 있습니다. dd 명령은 키를 저장하지 않습니다. 절차는 222 페이지 “dd 명령을 사용하여 대칭 키를 생성하는 방법”을 참조하십시오.

1 (옵션) 키 저장소를 사용하려면 하나 만듭니다.

- **PKCS #11 키 저장소를 만들고 초기화하려면 256 페이지 “pktool setpin 명령을 사용하여 암호문을 생성하는 방법”을 참조하십시오.**
- **NSS 데이터베이스를 만들고 초기화하려면 예 13–5를 참조하십시오.**

2 대칭 키로 사용할 난수를 생성합니다.

다음 방법 중 하나를 사용합니다.

- **키를 생성하고 파일에 저장합니다.**

  파일에 저장된 키의 이점은, /etc/inet/secret/ipseckeys 파일 또는 IPsec과 같은 응용 프로그램의 키 파일에서 사용할 키를 이 파일로부터 추출할 수 있다는 것입니다.

  ```
  % pktool genkey keystore=file outkey=key-fn \
  [keytype=generic|specific-symmetric-algorithm] [keylen=size-in-bits] \
  [dir=directory] [print=n]
  ```

  keystore
  : file 값은 키에 대해 파일 유형의 저장소 위치를 지정합니다.

  outkey=*key-fn*
  : keystore=file일 때 파일 이름입니다.

  keytype=*specific-symmetric-algorithm*
  : 임의 길이의 대칭 키의 경우 값이 generic입니다. 특정 알고리즘에 대해 aes, arcfour, des, 3des를 지정합니다.

  keylen=*size-in-bits*
  : 비트 단위의 키 길이입니다. 숫자는 8로 나눌 수 있어야 합니다. des 또는 3des에 지정하지 **마십시오**.

  dir=*directory*
  : *key-fn*에 대한 디렉토리 경로입니다. 기본적으로 *directory*가 현재 디렉토리입니다.

  print=n
  : 터미널 창에 키를 인쇄합니다. 기본적으로 print의 값은 n입니다.

- **키를 생성하고 PKCS #11 키 저장소에 저장합니다.**

  PKCS #11 키 저장소의 이점은, 레이블로 키를 검색할 수 있다는 것입니다. 이 방법은 파일을 암호화하고 해독하는 키에 유용합니다. 이 방법을 사용하기 전에 단계 1을 완료해야 합니다.

  ```
  % pktool genkey label=key-label \
  [keytype=generic|specific-symmetric-algorithm] [keylen=size-in-bits] \
  [token=token] [sensitive=n] [extractable=y] [print=n]
  ```

  label=*key-label*
  : 사용자가 지정한 키의 레이블입니다. 레이블로 키 저장소에서 키를 검색할 수 있습니다.

  keytype=*specific-symmetric-algorithm*
  : 임의 길이의 대칭 키의 경우 값이 generic입니다. 특정 알고리즘에 대해 aes, arcfour, des, 3des를 지정합니다.

  keylen=*size-in-bits*
  : 비트 단위의 키 길이입니다. 숫자는 8로 나눌 수 있어야 합니다. des 또는 3des에 지정하지 **마십시오**.

token=*token*
토큰 이름입니다. 기본적으로 토큰은 Sun Software PKCS#11 softtoken입니다.

sensitive=n
키의 민감도를 지정합니다. 값이 y이면 print=y 인수를 사용하여 키를 인쇄할 수 없습니다. 기본적으로 sensitive의 값은 n입니다.

extractable=y
키 저장소에서 키를 추출할 수 있는지 지정합니다. 키가 추출되지 않도록 하려면 n을 지정합니다.

print=n
터미널 창에 키를 인쇄합니다. 기본적으로 print의 값은 n입니다.

- **키를 생성하고 NSS 키 저장소에 저장합니다.**
  이 방법을 사용하기 전에 단계 1을 완료해야 합니다.

```
% pktool keystore=nss genkey label=key-label \
[keytype=[keytype=generic|specific-symmetric-algorithm] [keylen=size-in-bits] [token=token] \
[dir=directory-path] [prefix=database-prefix]
```

keystore
nss 값은 키에 대해 NSS 유형의 저장소 위치를 사용합니다.

label=*key-label*
사용자가 지정한 키의 레이블입니다. 레이블로 키 저장소에서 키를 검색할 수 있습니다.

keytype=*specific-symmetric-algorithm*
임의 길이의 대칭 키의 경우 값이 generic입니다. 특정 알고리즘에 대해 aes, arcfour, des, 3des를 지정합니다.

keylen=*size-in-bits*
비트 단위의 키 길이입니다. 숫자는 8로 나눌 수 있어야 합니다. des 또는 3des에 지정하지 **마십시오**.

token=*token*
토큰 이름입니다. 기본적으로 토큰은 NSS 내부 토큰입니다.

dir=*directory*
NSS 데이터베이스에 대한 디렉토리 경로입니다. 기본적으로 *directory*는 현재 디렉토리입니다.

prefix=*directory*
NSS 데이터베이스의 접두어입니다. 기본값은 접두어 없음입니다.

print=n
터미널 창에 키를 인쇄합니다. 기본적으로 print의 값은 n입니다.

3 (옵션) 키가 존재하는지 확인합니다.

키를 저장한 위치에 따라 다음 명령 중 하나를 사용합니다.

- ***key-fn* 파일에서 키를 확인합니다.**

```
% pktool list keystore=file objtype=key infile=key-fn
Found n keys.
Key #1 - keytype:location (keylen)
```

- **PKCS #11 또는 NSS 키 저장소에서 키를 확인합니다.**

```
$ pktool list objtype=key
Enter PIN for keystore:
Found n keys.
Key #1 - keytype:location (keylen)
```

예 12–5 pktool 명령을 사용하여 대칭 키 만들기

다음 예에서 사용자가 처음으로 PKCS #11 키 저장소를 만들고, 응용 프로그램에 대한 대형 대칭 키를 생성합니다. 마지막으로, 키가 키 저장소에 있는지 확인합니다.

```
# pktool setpin
Create new passphrase:      easily-remembered-hard-to-detect-password
Re-enter new passphrase:     Retype password
Passphrase changed.
% pktool genkey label=specialappkey keytype=generic keylen=1024
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

% pktool list objtype=key
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

Found 1 keys.
Key #1 - symmetric:  specialappkey (1024 bits)
```

예 12–6 pktool 명령을 사용하여 DES 키 만들기

다음 예에서 DES 알고리즘의 보안 키를 만듭니다. 나중에 해독을 위해 로컬 파일에 키를 저장합니다. 명령이 `400` 사용 권한으로 파일을 보호합니다. 키를 만들 때 `print=y` 옵션이 터미널 창에 생성된 키를 표시합니다.

DES 방식은 64비트 키를 사용합니다. 키 파일을 소유한 사용자가 `od` 명령을 사용하여 키를 검색합니다.

```
% pktool genkey keystore=file outkey=64bit.file1 keytype=des print=y
        Key Value ="a3237b2c0a8ff9b3"
% od -x 64bit.file1
0000000 a323 7b2c 0a8f f9b3
```

예 12–7 IPsec 보안 연관에 대한 대칭 키 만들기

다음 예에서 관리자가 수동으로 IPsec SA에 대한 키 관련 자료를 만들고 파일에 저장합니다. 그런 다음, 관리자가 /etc/inet/secret/ipseckeys 파일로 키를 복사하고 원본 파일을 삭제합니다.

- 먼저, 관리자가 IPsec 정책에 필요한 키를 만들고 표시합니다.

```
# pktool genkey keystore=file outkey=ipencrin1 keytype=generic keylen=192 print=y
   Key Value ="294979e512cb8e79370dabecadc3fcbb849e78d2d6bd2049"
# pktool genkey keystore=file outkey=ipencrout1 keytype=generic keylen=192 print=y
   Key Value ="9678f80e33406c86e3d1686e50406bd0434819c20d09d204"
# pktool genkey keystore=file outkey=ipspi1 keytype=generic keylen=32 print=y
   Key Value ="acbeaa20"
# pktool genkey keystore=file outkey=ipspi2 keytype=generic keylen=32 print=y
   Key Value ="19174215"
# pktool genkey keystore=file outkey=ipsha21 keytype=generic keylen=256 print=y
   Key Value ="659c20f2d6c3f9570bcee93e96d95e2263aca4eeb3369f72c5c786af4177fe9e"
# pktool genkey keystore=file outkey=ipsha22 keytype=generic keylen=256 print=y
   Key Value ="b041975a0e1fce0503665c3966684d731fa3dbb12fcf87b0a837b2da5d82c810"
```

- 그런 다음, 관리자가 다음 /etc/inet/secret/ipseckeys 파일을 만듭니다.

```
##   SPI values require a leading 0x.
##   Backslashes indicate command continuation.
##
## for outbound packets on this system
add esp spi 0xacbeaa20 \
   src 192.168.1.1 dst 192.168.2.1 \
   encr_alg aes auth_alg sha256  \
   encrkey  294979e512cb8e79370dabecadc3fcbb849e78d2d6bd2049 \
   authkey  659c20f2d6c3f9570bcee93e96d95e2263aca4eeb3369f72c5c786af4177fe9e
##
## for inbound packets
add esp spi 0x19174215 \
   src 192.168.2.1 dst 192.168.1.1 \
   encr_alg aes auth_alg sha256  \
   encrkey 9678f80e33406c86e3d1686e50406bd0434819c20d09d204 \
   authkey b041975a0e1fce0503665c3966684d731fa3dbb12fcf87b0a837b2da5d82c810
```

- ipseckeys 파일의 구문이 유효한지 확인한 후에 관리자가 원본 키 파일을 삭제합니다.

```
# ipseckey -c /etc/inet/secret/ipseckeys
# rm ipencrin1 ipencrout1 ipspi1 ipspi2 ipsha21 ipsha22
```

- 관리자가 ssh 명령 또는 다른 보안 방식을 사용하여 ipseckeys 파일을 통신 시스템에 복사합니다. 통신 시스템에서 보호 사항이 반전됩니다. ipseckeys 파일의 첫번째 항목이 인바운드 패킷을 보호하고, 두번째 항목이 아웃바운드 패킷을 보호합니다. 통신 시스템에는 키가 생성되지 않습니다.

▼ 파일의 다이제스트를 계산하는 방법

파일의 다이제스트를 계산할 때 다이제스트 출력을 비교하여 파일이 변조되지 않았는지 확인할 수 있습니다. 다이제스트는 원본 파일을 고치지 않습니다.

1 사용 가능한 다이제스트 알고리즘을 나열합니다.

```
% digest -l
md5
sha1
sha256
sha384
sha512
```

2 파일의 다이제스트를 계산하고 다이제스트 목록을 저장합니다.

`digest` 명령으로 알고리즘을 제공합니다.

`% digest -v -a` *algorithm input-file* `>` *digest-listing*

`-v` 다음 형식으로 출력을 표시합니다.

algorithm (*input-file*) = *digest*

`-a` *algorithm* 파일의 다이제스트를 계산하는 데 사용할 알고리즘입니다. 단계 1의 출력에 나타난 대로 알고리즘을 입력합니다.

input-file `digest` 명령에 대한 입력 파일입니다.

digest-listing `digest` 명령에 대한 출력 파일입니다.

예 12–8 MD5 방식으로 다이제스트 계산

다음 예에서 `digest` 명령이 MD5 방식을 사용하여 전자 메일 첨부 파일에 대한 다이제스트를 계산합니다.

```
% digest -v -a md5 email.attach >>  $HOME/digest.emails.05.07
% cat  ~/digest.emails.05.07
md5 (email.attach) = 85c0a53d1a5cc71ea34d9ee7b1b28b01
```

`-v` 옵션을 사용하지 않으면 동봉한 정보 없이 다이제스트가 저장됩니다.

```
% digest -a md5 email.attach >>  $HOME/digest.emails.05.07
% cat  ~/digest.emails.05.07
85c0a53d1a5cc71ea34d9ee7b1b28b01
```

예 12–9 SHA1 방식으로 다이제스트 계산

다음 예에서 `digest` 명령이 SHA1 방식을 사용하여 디렉토리 목록을 제공합니다. 결과가 파일에 배치됩니다.

```
% digest -v -a sha1 docs/* > $HOME/digest.docs.legal.05.07
% more ~/digest.docs.legal.05.07
sha1 (docs/legal1) = 1df50e8ad219e34f0b911e097b7b588e31f9b435
sha1 (docs/legal2) = 68efa5a636291bde8f33e046eb33508c94842c38
sha1 (docs/legal3) = 085d991238d61bd0cfa2946c183be8e32cccf6c9
sha1 (docs/legal4) = f3085eae7e2c8d008816564fdf28027d10e1d983
```

▼ 파일의 MAC을 계산하는 방법

메시지 인증 코드(또는 MAC)는 파일의 다이제스트를 계산하고 보안 키를 사용하여 다이제스트를 한층 더 보호합니다. MAC은 원본 파일을 고치지 않습니다.

1 사용 가능한 방식을 나열합니다.

```
% mac -l
Algorithm       Keysize:  Min   Max
-----------------------------------
des_mac                   64    64
sha1_hmac                  8   512
md5_hmac                   8   512
sha256_hmac                8   512
sha384_hmac                8  1024
sha512_hmac                8  1024
```

2 적절한 길이의 대칭 키를 생성합니다.

두 가지 옵션이 있습니다. 키 생성에 사용할 암호문을 제공할 수 있습니다. 또는 키를 제공할 수 있습니다.

- 암호문을 제공하는 경우 암호문을 저장하거나 기억해야 합니다. 암호문을 온라인으로 저장할 경우 본인만 암호문 파일을 읽을 수 있어야 합니다.
- 키를 제공하는 경우 방식에 맞는 올바른 크기여야 합니다. 절차는 222 페이지 “dd 명령을 사용하여 대칭 키를 생성하는 방법”을 참조하십시오. 또한 `pktool` 명령을 사용할 수 있습니다. 절차 및 일부 예제는 224 페이지 “pktool 명령을 사용하여 대칭 키를 생성하는 방법”을 참조하십시오.

3 파일의 MAC을 만듭니다.

`mac` 명령으로 키를 제공하고 대칭 키 알고리즘을 사용합니다.

`% mac [-v] -a` *algorithm* `[-k` *keyfile* `| -K` *key-label* `[-T` *token*`]]` *input-file*

`-v` 다음 형식으로 출력을 표시합니다.

algorithm (*input-file*) = *mac*

`-a` *algorithm* MAC을 계산하는 데 사용할 알고리즘입니다. `mac -l` 명령의 출력에 나타난 대로 알고리즘을 입력합니다.

`-k` *keyfile* 알고리즘이 지정된 길이의 키를 포함하는 파일입니다.

`-K` *key-label* PKCS #11 키 저장소에서 키의 레이블입니다.

`-T` *token* 토큰 이름입니다. 기본적으로 토큰은 `Sun Software PKCS#11 softtoken`입니다. `-K` *key-label* 옵션을 사용할 때만 사용됩니다.

input-file MAC에 대한 입력 파일입니다.

예 12–10 DES_MAC 및 암호문으로 MAC 계산

다음 예에서 DES_MAC 방식과 암호문에서 파생된 키를 사용하여 전자 메일 첨부 파일이 인증됩니다. MAC 목록이 파일에 저장됩니다. 암호문이 파일에 저장된 경우 사용자 이외의 다른 사람이 파일을 읽을 수 없어야 합니다.

```
% mac -v -a des_mac email.attach
Enter passphrase:      <Type passphrase>
des_mac (email.attach) = dd27870a
% echo "des_mac (email.attach) = dd27870a" >> ~/desmac.daily.05.07
```

예 12–11 MD5_HMAC 및 키 파일로 MAC 계산

다음 예에서 MD5_HMAC 방식과 보안 키를 사용하여 전자 메일 첨부 파일이 인증됩니다. MAC 목록이 파일에 저장됩니다.

```
% mac -v -a md5_hmac -k $HOME/keyf/05.07.mack64 email.attach
md5_hmac (email.attach) = 02df6eb6c123ff25d78877eb1d55710c
% echo "md5_hmac (email.attach) = 02df6eb6c123ff25d78877eb1d55710c" \
>> ~/mac.daily.05.07
```

예 12–12 SHA1_HMAC 및 키 파일로 MAC 계산

다음 예에서 SHA1_HMAC 방식과 보안 키를 사용하여 디렉토리 매니페스트가 인증됩니다. 결과가 파일에 배치됩니다.

```
% mac -v -a sha1_hmac \
-k $HOME/keyf/05.07.mack64 docs/* > $HOME/mac.docs.legal.05.07
% more ~/mac.docs.legal.05.07
sha1_hmac (docs/legal1) = 9b31536d3b3c0c6b25d653418db8e765e17fe07a
sha1_hmac (docs/legal2) = 865af61a3002f8a457462a428cdb1a88c1b51ff5
sha1_hmac (docs/legal3) = 076c944cb2528536c9aebd3b9fbe367e07b61dc7
sha1_hmac (docs/legal4) = 7aede27602ef6e4454748cbd3821e0152e45beb4
```

예 12–13 SHA1_HMAC 및 키 레이블로 MAC 계산

다음 예에서 SHA1_HMAC 방식과 보안 키를 사용하여 디렉토리 매니페스트가 인증됩니다. 사용자의 PKCS #11 키 저장소에 결과가 놓입니다. 사용자가 초기에 `pktool setpin` 명령을 사용하여 키 저장소와 키 저장소의 암호를 만들었습니다.

```
% mac -a sha1_hmac -K legaldocs0507 docs/*
Enter pin for Sun Software PKCS#11 softtoken:      Type password
```

키 저장소에서 MAC을 검색하려면 사용자가 상세 정보 표시 옵션을 사용하고 키 레이블과 인증된 디렉토리의 이름을 제공합니다.

```
% mac -v -a sha1_hmac -K legaldocs0507  docs/*
Enter pin for Sun Software PKCS#11 softtoken:     Type password
sha1_hmac (docs/legal1) = 9b31536d3b3c0c6b25d653418db8e765e17fe07a
sha1_hmac (docs/legal2) = 865af61a3002f8a457462a428cdb1a88c1b51ff5
sha1_hmac (docs/legal3) = 076c944cb2528536c9aebd3b9fbe367e07b61dc7
sha1_hmac (docs/legal4) = 7aede27602ef6e4454748cbd3821e0152e45beb4
```

▼ 파일을 암호화 및 해독하는 방법

파일을 암호화할 때 원본 파일은 제거되거나 변경되지 않습니다. 출력 파일이 암호화됩니다.

encrypt 명령에 공통적인 오류 해결 방법은 예제 뒤의 절을 참조하십시오.

1 적절한 길이의 대칭 키를 만듭니다.

두 가지 옵션이 있습니다. 키 생성에 사용할 암호문을 제공할 수 있습니다. 또는 키를 제공할 수 있습니다.

- 암호문을 제공하는 경우 암호문을 저장하거나 기억해야 합니다. 암호문을 온라인으로 저장할 경우 본인만 암호문 파일을 읽을 수 있어야 합니다.
- 키를 제공하는 경우 방식에 맞는 올바른 크기여야 합니다. 절차는 222 페이지 "dd 명령을 사용하여 대칭 키를 생성하는 방법"을 참조하십시오. 또한 pktool 명령을 사용할 수 있습니다. 절차 및 일부 예제는 224 페이지 "pktool 명령을 사용하여 대칭 키를 생성하는 방법"을 참조하십시오.

2 파일을 암호화합니다.

encrypt 명령으로 키를 제공하고 대칭 키 알고리즘을 사용합니다.

```
% encrypt -a algorithm [-v] \
[-k keyfile | -K key-label [-T token]] [-i input-file] [-o output-file]
```

-a *algorithm*	파일을 암호화하는 데 사용할 알고리즘입니다. encrypt -l 명령의 출력에 나타난 대로 알고리즘을 입력합니다.
-k *keyfile*	알고리즘이 지정된 길이의 키를 포함하는 파일입니다. encrypt -l 명령의 출력에 각 알고리즘의 키 길이가 비트 단위로 나열됩니다.
-K *key-label*	PKCS #11 키 저장소에서 키의 레이블입니다.
-T *token*	토큰 이름입니다. 기본적으로 토큰은 Sun Software PKCS#11 softtoken입니다. -K *key-label* 옵션을 사용할 때만 사용됩니다.
-i *input-file*	암호화하려는 입력 파일입니다. 이 파일은 명령에 의해 바뀌지 않습니다.
-o *output-file*	입력 파일의 암호화된 형태인 출력 파일입니다.

예 12–14 파일을 암호화하기 위한 AES 키 만들기

다음 예에서 사용자가 암호화 및 해독에 사용할 AES 키를 만들고 기존 PKCS #11 키 저장소에 저장합니다. 키가 존재하는지 확인하고 키를 사용할 수 있지만, 키 자체를 볼 수는 없습니다.

```
% pktool genkey label=MyAESkeynumber1 keytype=aes keylen=256
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

% pktool list objtype=key
Enter PIN for Sun Software PKCS#11 softtoken  :<Type password>
Found 1 key
Key #1 - Sun Software PKCS#11 softtoken: MyAESkeynumber1 (256)
```

키를 사용하여 파일을 암호화하려면 레이블로 키를 검색합니다.

```
% encrypt -a aes -K MyAESkeynumber1 -i encryptthisfile -o encryptedthisfile
```

encryptedthisfile 파일을 해독하려면 레이블로 키를 검색합니다.

```
% decrypt -a aes -K MyAESkeynumber1 -i encryptedthisfile -o sameasencryptthisfile
```

예 12–15 AES 및 암호문으로 암호화 및 해독

다음 예에서 AES 알고리즘으로 파일이 암호화됩니다. 키가 암호문에서 생성됩니다. 암호문이 파일에 저장된 경우 사용자 이외의 다른 사람이 파일을 읽을 수 없어야 합니다.

```
% encrypt -a aes -i ticket.to.ride -o ~/enc/e.ticket.to.ride
Enter passphrase:      <Type passphrase>
Re-enter passphrase:       Type passphrase again
```

입력 파일 ticket.to.ride가 여전히 원본 형태로 존재합니다.

출력 파일을 해독하려면 파일을 암호화한 것과 동일한 암호문 및 암호화 방식을 사용합니다.

```
% decrypt -a aes -i ~/enc/e.ticket.to.ride -o ~/d.ticket.to.ride
Enter passphrase:      <Type passphrase>
```

예 12–16 AES 및 키 파일로 암호화 및 해독

다음 예에서 AES 알고리즘으로 파일이 암호화됩니다. AES 방식은 128비트 또는 16바이트 키를 사용합니다.

```
% encrypt -a aes -k ~/keyf/05.07.aes16 \
-i ticket.to.ride -o ~/enc/e.ticket.to.ride
```

입력 파일 ticket.to.ride가 여전히 원본 형태로 존재합니다.

출력 파일을 해독하려면 파일을 암호화한 것과 동일한 키 및 암호화 방식을 사용합니다.

```
% decrypt -a aes -k ~/keyf/05.07.aes16  \
-i ~/enc/e.ticket.to.ride -o ~/d.ticket.to.ride
```

예 12–17 ARCFOUR 및 키 파일로 암호화 및 해독

다음 예에서 ARCFOUR 알고리즘으로 파일이 암호화됩니다. ARCFOUR 알고리즘은 8비트(1바이트), 64비트(8바이트) 또는 128비트(16바이트) 키를 수용합니다.

```
% encrypt -a arcfour -i personal.txt \
-k ~/keyf/05.07.rc4.8 -o ~/enc/e.personal.txt
```

출력 파일을 해독하려면 파일을 암호화한 것과 동일한 키 및 암호화 방식을 사용합니다.

```
% decrypt -a arcfour -i ~/enc/e.personal.txt \
-k ~/keyf/05.07.rc4.8 -o ~/personal.txt
```

예 12–18 3DES 및 키 파일로 암호화 및 해독

다음 예에서 3DES 알고리즘으로 파일이 암호화됩니다. 3DES 알고리즘에 192비트 또는 24바이트 키가 필요합니다.

```
% encrypt -a 3des -k ~/keyf/05.07.des24 \
-i ~/personal2.txt -o ~/enc/e.personal2.txt
```

출력 파일을 해독하려면 파일을 암호화한 것과 동일한 키 및 암호화 방식을 사용합니다.

```
% decrypt -a 3des -k ~/keyf/05.07.des24 \
-i ~/enc/e.personal2.txt -o ~/personal2.txt
```

일반 오류 다음 메시지는 encrypt 명령에 제공한 키가 사용 중인 알고리즘에서 허가되지 않음을 나타냅니다.

- encrypt: unable to create key for crypto operation:(**암호화 작업에 대한 키를 작성할 수 없습니다.**) CKR_ATTRIBUTE_VALUE_INVALID
- encrypt: failed to initialize crypto operation:(**암호화 작업을 초기화하지 못했습니다.**) CKR_KEY_SIZE_RANGE

알고리즘의 요구 사항을 충족하지 않는 키를 전달하면 더 좋은 키를 제공해야 합니다.

- 한가지 옵션은 암호문을 사용하는 것입니다. 그러면 프레임워크가 요구 사항을 충족하는 키를 제공합니다.
- 두번째 옵션은 알고리즘이 수용하는 키 크기를 전달하는 것입니다. 예를 들어, DES 알고리즘에 64비트 키가 필요합니다. 3DES 알고리즘에 192비트 키가 필요합니다.

암호화 프레임워크 관리(작업)

이 절에서는 암호화 프레임워크에서 소프트웨어 공급자 및 하드웨어 공급자를 관리하는 방법을 설명합니다. 원하는 경우 소프트웨어 공급자 및 하드웨어 공급자를 사용에서 제할 수 있습니다. 예를 들어, 한 소프트웨어 공급자에서 알고리즘 구현을 사용 안함으로 설정할 수 있습니다. 그런 다음, 다른 소프트웨어 공급자에서 알고리즘을 사용하도록 시스템에 강제 적용할 수 있습니다.

암호화 프레임워크 관리(작업 맵)

다음 작업 맵은 암호화 프레임워크에서 소프트웨어 및 하드웨어 공급자를 관리하기 위한 절차를 가리킵니다.

작업	설명	수행 방법
암호화 프레임워크에서 공급자를 나열합니다.	암호화 프레임워크에서 사용할 수 있는 알고리즘, 라이브러리 및 하드웨어 장치를 나열합니다.	236 페이지 “사용 가능한 공급자를 나열하는 방법”
소프트웨어 공급자를 추가합니다.	PKCS #11 라이브러리 또는 커널 모듈을 암호화 프레임워크에 추가합니다. 공급자에 서명해야 합니다.	239 페이지 “소프트웨어 공급자를 추가하는 방법”
사용자 레벨 방식의 사용을 금지합니다.	소프트웨어 방식을 사용에서 제합니다. 방식을 다시 사용으로 설정할 수 있습니다.	241 페이지 “사용자 레벨 방식의 사용을 금지하는 방법”
커널 모듈에서 방식을 일시적으로 사용 안함으로 설정합니다.	방식을 일시적으로 사용에서 제합니다. 대개 테스트에 사용됩니다.	242 페이지 “커널 소프트웨어 공급자의 사용을 금지하는 방법”
공급자 설치를 제거합니다.	커널 소프트웨어 공급자를 사용에서 제합니다.	예 12–27
사용 가능한 하드웨어 공급자를 나열합니다.	연결된 하드웨어를 표시하고, 하드웨어가 제공하는 방식을 표시하고, 사용으로 설정된 방식을 표시합니다.	245 페이지 “하드웨어 공급자를 나열하는 방법”
하드웨어 공급자에서 방식을 사용 안함으로 설정합니다.	하드웨어 가속기에서 선택한 방식이 사용되지 않는지 확인합니다.	246 페이지 “하드웨어 공급자 방식 및 기능을 사용 안하는 방법”
암호화 서비스를 다시 시작하거나 새로 고칩니다.	암호화 서비스가 사용 가능한지 확인합니다.	247 페이지 “모든 암호화 서비스를 새로 고치거나 다시 시작하는 방법”

▼ 사용 가능한 공급자를 나열하는 방법

암호화 프레임워크는 여러 유형의 소비자에 대한 알고리즘을 제공합니다.

- 사용자 레벨 공급자는 `libpkcs11` 라이브러리로 링크된 응용 프로그램에 PKCS #11 암호화 인터페이스를 제공합니다.
- 커널 소프트웨어 공급자는 IPsec, Kerberos 및 기타 Oracle Solaris 커널 구성 요소에 대한 알고리즘을 제공합니다.
- 커널 하드웨어 공급자는 커널 소비자에 사용 가능한 알고리즘을 `pkcs11_kernel` 라이브러리를 통해 응용 프로그램에 제공합니다.

1 간단한 형식으로 공급자를 나열합니다.

주 – 공급자 목록의 내용 및 형식은 여러 Oracle Solaris 릴리스마다 다릅니다. 시스템에서 `cryptoadm list` 명령을 실행하여 시스템이 지원하는 공급자를 확인하십시오.

일반 사용자는 사용자 레벨의 방식만 사용할 수 있습니다.

```
% cryptoadm list
User-level providers:
Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
Provider: /usr/lib/security/$ISA/pkcs11_tpm.so

Kernel software providers:
    des
    aes
    arcfour
    blowfish
    ecc
    sha1
    sha2
    md4
    md5
    rsa
    swrand

Kernel hardware providers:
    ncp/0
```

2 암호화 프레임워크에서 공급자 및 해당 방식을 나열합니다.

다음 출력에 모든 방식이 나열됩니다. 그러나 나열된 방식 중 일부는 사용하지 못할 수 있습니다. 관리자가 사용 승인한 방식만 나열하려면 예 12–20을 참조하십시오.

표시 목적상 출력이 잘립니다.

```
% cryptoadm list -m
User-level providers:
=====================

Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
```

```
/usr/lib/security/$ISA/pkcs11_kernel.so: no slots presented.

Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
Mechanisms:
CKM_DES_CBC
CKM_DES_CBC_PAD
CKM_DES_ECB
CKM_DES_KEY_GEN
CKM_DES_MAC_GENERAL
...
CKM_ECDSA_SHA1
CKM_ECDH1_DERIVE

Provider: /usr/lib/security/$ISA/pkcs11_tpm.so
/usr/lib/security/$ISA/pkcs11_tpm.so: no slots presented.

Kernel software providers:
==========================
des: CKM_DES_ECB,CKM_DES_CBC,CKM_DES3_ECB,CKM_DES3_CBC
aes: CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM,CKM_AES_GCM,CKM_AES_GMAC
arcfour: CKM_RC4
blowfish: CKM_BLOWFISH_ECB,CKM_BLOWFISH_CBC
ecc: CKM_EC_KEY_PAIR_GEN,CKM_ECDH1_DERIVE,CKM_ECDSA,CKM_ECDSA_SHA1
sha1: CKM_SHA_1,CKM_SHA_1_HMAC,CKM_SHA_1_HMAC_GENERAL
sha2: CKM_SHA256,CKM_SHA256_HMAC,CKM_SHA256_HMAC_GENERAL,CKM_SHA384,CKM_SHA384_HMAC,
CKM_SHA384_HMAC_GENERAL,CKM_SHA512,CKM_SHA512_HMAC,CKM_SHA512_HMAC_GENERAL
md4: CKM_MD4
md5: CKM_MD5,CKM_MD5_HMAC,CKM_MD5_HMAC_GENERAL
rsa: CKM_RSA_PKCS,CKM_RSA_X_509,CKM_MD5_RSA_PKCS,CKM_SHA1_RSA_PKCS,
CKM_SHA256_RSA_PKCS,CKM_SHA384_RSA_PKCS,CKM_SHA512_RSA_PKCS
swrand: No mechanisms presented.

Kernel hardware providers:
==========================
ncp/0: CKM_DSA,CKM_RSA_X_509,CKM_RSA_PKCS,CKM_RSA_PKCS_KEY_PAIR_GEN,
CKM_DH_PKCS_KEY_PAIR_GEN,CKM_DH_PKCS_DERIVE,CKM_EC_KEY_PAIR_GEN,
CKM_ECDH1_DERIVE,CKM_ECDSA
```

예 12–19 기존 암호화 방식 찾기

다음 예에서 사용자 레벨 라이브러리 pkcs11_softtoken이 제공하는 모든 방식이 나열됩니다.

```
% cryptoadm list -m provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
Mechanisms:
CKM_DES_CBC
CKM_DES_CBC_PAD
CKM_DES_ECB
CKM_DES_KEY_GEN
CKM_DES_MAC_GENERAL
CKM_DES_MAC
...
CKM_ECDSA
CKM_ECDSA_SHA1
CKM_ECDH1_DERIVE
```

예 12–20 사용 가능한 암호화 방식 찾기

정책에 따라 사용 가능한 방식이 결정됩니다. 관리자가 정책을 설정합니다. 관리자는 특정 공급자에서 방식이 사용되지 않도록 선택할 수 있습니다. -p 옵션은 관리자가 설정한 정책에 의해 허가된 방식 목록을 표시합니다.

```
% cryptoadm list -p
User-level providers:
=====================
/usr/lib/security/$ISA/pkcs11_kernel.so: all mechanisms are enabled.
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
except CKM_MD5. random is enabled.
/usr/lib/security/$ISA/pkcs11_tpm.so: all mechanisms are enabled.

Kernel software providers:
==========================
des: all mechanisms are enabled.
aes: all mechanisms are enabled.
arcfour: all mechanisms are enabled.
blowfish: all mechanisms are enabled.
ecc: all mechanisms are enabled.
sha1: all mechanisms are enabled.
sha2: all mechanisms are enabled.
md4: all mechanisms are enabled.
md5: all mechanisms are enabled.
rsa: all mechanisms are enabled.
swrand: random is enabled.

Kernel hardware providers:
==========================
ncp/0: all mechanisms are enabled. random is enabled.
```

예 12–21 어떤 암호화 방식이 어떤 기능을 수행하는지 확인

방식은 서명 또는 키 생성과 같은 특정 암호화 기능을 수행합니다. -v -m 옵션은 모든 방식과 해당 기능을 표시합니다.

이 경우 관리자가 CKM_ECDSA* 방식이 사용될 수 있는 기능이 무엇인지 확인할 수 있습니다.

```
% cryptoadm list -vm
User-level providers:
=====================

Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
/usr/lib/security/$ISA/pkcs11_kernel.so: no slots presented.

Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
...
CKM_ECDSA       112 571  .  .  .  .  X  .  X  .  .  .  .  .  .
CKM_ECDSA_SHA1  112 571  .  .  .  .  X  .  X  .  .  .  .  .  .
...
```

목록에 의하면, 이러한 사용자 레벨 방식을 /usr/lib/security/$ISA/pkcs11_softtoken.so 라이브러리에서 사용할 수 있습니다.

각 항목은 방식에 대한 정보 조각을 나타냅니다. 이러한 ECC 방식에 나타나는 목록은 다음과 같습니다.

- 최소 길이 – 112바이트
- 최대 길이 – 571바이트
- 하드웨어 – 하드웨어에 사용할 수 없습니다.
- 암호화 – 데이터를 암호화하는 데 사용되지 않습니다.
- 해독 – 데이터를 해독하는 데 사용되지 않습니다.
- 다이제스트 – 메시지 다이제스트를 만드는 데 사용되지 않습니다.
- 서명 – 데이터를 서명하는 데 사용됩니다.
- 서명 + 복구 – 데이터를 서명으로부터 복구할 수 있는 경우 데이터를 서명하는 데 사용되지 않습니다.
- 확인 – 서명된 데이터를 확인하는 데 사용됩니다.
- 확인 + 복구 – 서명으로부터 복구할 수 있는 데이터를 확인하는 데 사용되지 않습니다.
- 키 생성 – 개인 키를 생성하는 데 사용되지 않습니다.
- 쌍 생성 – 키 쌍을 생성하는 데 사용되지 않습니다.
- 래핑 – 래핑하는 데 사용되지 않습니다. 즉, 기존 키를 암호화합니다.
- 언래핑 – 래핑된 키를 언래핑하는 데 사용되지 않습니다.
- 파생 – 기본 키로부터 새 키를 파생하는 데 사용되지 않습니다.

▼ 소프트웨어 공급자를 추가하는 방법

시작하기 전에 Crypto Management 권한 프로파일에 할당되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 시스템에 사용 가능한 소프트웨어 공급자를 나열합니다.

```
% cryptoadm list
User-level providers:
Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_tpm.so: all mechanisms are enabled.

Kernel software providers:
    des
    aes
    arcfour
    blowfish
    sha1
```

```
    sha2
    md4
    md5
    rsa
    swrand

Kernel hardware providers:
    ncp/0
```

3 저장소에서 공급자를 추가합니다.

기존 공급자 소프트웨어는 Oracle에서 인증서를 발행했습니다.

4 공급자를 새로 고칩니다.

소프트웨어 공급자를 추가한 경우 또는 하드웨어와 지정된 정책을 추가한 경우 공급자를 새로 고쳐야 합니다.

```
# svcadm refresh svc:/system/cryptosvc
```

5 목록에서 새 공급자를 찾습니다.

이 경우 새 커널 소프트웨어 공급자가 설치되었습니다.

```
# cryptoadm list
...
Kernel software providers:
    des
    aes
    arcfour
    blowfish
    ecc
    sha1
    sha2
    md4
    md5
    rsa
    swrand
  sha3      <-- added provider
...
```

예 12–22 사용자 레벨 소프트웨어 공급자 추가

다음 예에서 서명된 PKCS #11 라이브러리가 설치됩니다.

```
# pkgadd -d /cdrom/cdrom0/SolarisNew
    Answer the prompts
# svcadm refresh system/cryptosvc
# cryptoadm list
user-level providers:
=========================
    /usr/lib/security/$ISA/pkcs11_kernel.so
    /usr/lib/security/$ISA/pkcs11_softtoken.so
    /usr/lib/security/$ISA/pkcs11_tpm.so
    /opt/lib/$ISA/libpkcs11.so.1      <-- added provider
```

암호화 프레임워크로 라이브러리를 테스트 중인 개발자가 수동으로 라이브러리를 설치할 수 있습니다.

```
# cryptoadm install provider=/opt/lib/\$ISA/libpkcs11.so.1
```

▼ 사용자 레벨 방식의 사용을 금지하는 방법

라이브러리 공급자의 암호화 방식 중 일부를 사용하면 안되는 경우 선택한 방식을 제거할 수 있습니다. 이 절차는 예제로 `pkcs11_softtoken` 라이브러리에서 DES 방식을 사용합니다.

시작하기 전에 Crypto Management 권한 프로파일에 할당되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 특정 사용자 레벨 소프트웨어 공급자가 제공한 방식을 나열합니다.

```
% cryptoadm list -m provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so:
CKM_DES_CBC,CKM_DES_CBC_PAD,CKM_DES_ECB,CKM_DES_KEY_GEN,
CKM_DES3_CBC,CKM_DES3_CBC_PAD,CKM_DES3_ECB,CKM_DES3_KEY_GEN,
CKM_AES_CBC,CKM_AES_CBC_PAD,CKM_AES_ECB,CKM_AES_KEY_GEN,
...
```

3 사용할 수 있는 방식을 나열합니다.

```
$ cryptoadm list -p
user-level providers:
=====================
...
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled.
random is enabled.
...
```

4 사용하면 안되는 방식을 비활성화합니다.

```
$ cryptoadm disable provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so \
> mechanism=CKM_DES_CBC,CKM_DES_CBC_PAD,CKM_DES_ECB
```

5 사용할 수 있는 방식을 나열합니다.

```
$ cryptoadm list -p provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
except CKM_DES_ECB,CKM_DES_CBC_PAD,CKM_DES_CBC. random is enabled.
```

예 12–23 사용자 레벨 소프트웨어 공급자 방식을 사용으로 설정

다음 예에서 사용 안함으로 설정된 DES 방식을 다시 사용할 수 있도록 만듭니다.

```
$ cryptoadm list -m provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so:
CKM_DES_CBC,CKM_DES_CBC_PAD,CKM_DES_ECB,CKM_DES_KEY_GEN,
CKM_DES3_CBC,CKM_DES3_CBC_PAD,CKM_DES3_ECB,CKM_DES3_KEY_GEN,
...
$ cryptoadm list -p provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
except CKM_DES_ECB,CKM_DES_CBC_PAD,CKM_DES_CBC. random is enabled.
$ cryptoadm enable provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so \
> mechanism=CKM_DES_ECB
$ cryptoadm list -p provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
except CKM_DES_CBC_PAD,CKM_DES_CBC. random is enabled.
```

예 12–24 모든 사용자 레벨 소프트웨어 공급자 방식을 사용으로 설정

다음 예에서 사용자 레벨 라이브러리의 모든 방식이 사용으로 설정됩니다.

```
$ cryptoadm enable provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so all
$ cryptoadm list -p provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled.
random is enabled.
```

예 12–25 사용자 레벨 소프트웨어 공급자 가용성을 영구적으로 제거

다음 예에서 `libpkcs11.so.1` 라이브러리가 제거됩니다.

```
$ cryptoadm uninstall provider=/opt/lib/\$ISA/libpkcs11.so.1
$ cryptoadm list
user-level providers:
    /usr/lib/security/$ISA/pkcs11_kernel.so
    /usr/lib/security/$ISA/pkcs11_softtoken.so
    /usr/lib/security/$ISA/pkcs11_tpm.so

kernel software providers:
...
```

▼ 커널 소프트웨어 공급자의 사용을 금지하는 방법

암호화 프레임워크가 AES와 같은 여러 모드의 공급자를 제공하는 경우 느리거나 손상된 방식을 사용에서 제할 수 있습니다. 이 절차는 예제로 AES 알고리즘을 사용합니다.

시작하기 전에 Crypto Management 권한 프로파일에 할당되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 특정 커널 소프트웨어 공급자가 제공한 방식을 나열합니다.

```
$ cryptoadm list -m provider=aes
aes: CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM,CKM_AES_GCM,CKM_AES_GMAC
```

3 사용할 수 있는 방식을 나열합니다.

```
$ cryptoadm list -p provider=aes
aes: all mechanisms are enabled.
```

4 사용하면 안되는 방식을 비활성화합니다.

```
$ cryptoadm disable provider=aes mechanism=CKM_AES_ECB
```

5 사용할 수 있는 방식을 나열합니다.

```
$ cryptoadm list -p provider=aes
aes: all mechanisms are enabled, except CKM_AES_ECB.
```

예 12–26 커널 소프트웨어 공급자 방식을 사용으로 설정

다음 예에서 사용 안함으로 설정된 AES 방식을 다시 사용할 수 있도록 만듭니다.

```
cryptoadm list -m provider=aes
aes: CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM,CKM_AES_GCM,CKM_AES_GMAC
$ cryptoadm list -p provider=aes
aes: all mechanisms are enabled, except CKM_AES_ECB.
$ cryptoadm enable provider=aes mechanism=CKM_AES_ECB
$ cryptoadm list -p provider=aes
aes: all mechanisms are enabled.
```

예 12–27 커널 소프트웨어 공급자 가용성을 일시적으로 제거

다음 예에서 AES 공급자를 일시적으로 사용에서 제합니다. 설치를 제거하는 동안 공급자가 자동으로 로드되지 못하게 하려면 `unload` 하위 명령이 유용합니다. 예를 들어, `unload` 하위 명령은 공급자에 영향을 미치는 패치를 설치할 때 사용됩니다.

```
$ cryptoadm unload provider=aes
```

```
$ cryptoadm list
...
Kernel software providers:
    des
    aes (inactive)
    arcfour
    blowfish
    ecc
    sha1
    sha2
    md4
    md5
    rsa
    swrand
```

암호화 프레임워크를 새로 고칠 때까지 AES 공급자를 사용할 수 없습니다.

```
$ svcadm refresh system/cryptosvc
```

```
$ cryptoadm list
...
Kernel software providers:
    des
    aes
    arcfour
    blowfish
    ecc
    sha1
    sha2
    md4
    md5
    rsa
    swrand
```

커널 소비자가 커널 소프트웨어 공급자를 사용 중인 경우 소프트웨어가 언로드되지 않습니다. 오류 메시지가 표시되고 공급자를 계속 사용할 수 있습니다.

예 12–28 소프트웨어 공급자 가용성을 영구적으로 제거

다음 예에서 AES 공급자를 사용에서 제합니다. 일단 제거된 AES 공급자는 커널 소프트웨어 공급자의 정책 목록에 나타나지 않습니다.

```
$ cryptoadm uninstall provider=aes

$ cryptoadm list
...
Kernel software providers:
    des
    arcfour
    blowfish
    ecc
    sha1
    sha2
    md4
    md5
    rsa
    swrand
```

커널 소비자가 커널 소프트웨어 공급자를 사용 중인 경우 오류 메시지가 표시되고 공급자를 계속 사용할 수 있습니다.

예 12–29 제거된 커널 소프트웨어 공급자 재설치

다음 예에서 AES 커널 소프트웨어 공급자가 재설치됩니다.

```
$ cryptoadm install provider=aes \
mechanism=CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM,CKM_AES_GCM,CKM_AES_GMAC

$ cryptoadm list
...
Kernel software providers:
    des
```

```
aes
arcfour
blowfish
ecc
sha1
sha2
md4
md5
rsa
swrand
```

▼ 하드웨어 공급자를 나열하는 방법

하드웨어 공급자는 자동으로 찾아서 로드됩니다. 자세한 내용은 `driver.conf(4)` 매뉴얼 페이지를 참조하십시오.

시작하기 전에 암호화 프레임워크 내에서 사용하려는 하드웨어가 있을 때 하드웨어가 커널에서 SPI로 등록됩니다. 프레임워크가 하드웨어 드라이버를 서명했는지 확인합니다. 특히, Sun이 발행한 인증서로 드라이버의 객체 파일을 서명했는지 확인합니다.

예를 들어, Sun Crypto Accelerator 6000 보드(`mca`), UltraSPARC T1 및 T2 프로세서의 암호화 가속기용 ncp 드라이버(`ncp`), UltraSPARC T2 프로세서의 n2cp 드라이버(`n2cp`)가 하드웨어 방식을 프레임워크로 플러그인합니다.

공급자 서명 얻기에 대한 내용은 218 페이지 “타사 소프트웨어에 대한 이진 서명”을 참조하십시오.

1 시스템에서 사용 가능한 하드웨어 공급자를 나열합니다.

```
% cryptoadm list
...
kernel hardware providers:
   ncp/0
```

2 칩 또는 보드가 제공하는 방식을 나열합니다.

```
% cryptoadm list -m provider=ncp/0
ncp/0:
CKM_DSA
CKM_RSA_X_509
...
CKM_ECDH1_DERIVE
CKM_ECDSA
```

3 칩 또는 보드에 사용할 수 있는 방식을 나열합니다.

```
% cryptoadm list -p provider=ncp/0
ncp/0: all mechanisms are enabled.
```

▼ 하드웨어 공급자 방식 및 기능을 사용 안하는 방법

하드웨어 공급자에서 방식과 난수 기능을 선택적으로 사용 안할 수 있습니다. 다시 사용으로 설정하려면 예 12–30을 참조하십시오. 이 예제의 하드웨어인 Sun Crypto Accelerator 1000 보드는 난수 생성기를 제공합니다.

시작하기 전에 Crypto Management 권한 프로파일에 할당되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2. **사용하지 않을 방식 또는 기능을 선택합니다.**
   하드웨어 공급자를 나열합니다.

   ```
   # cryptoadm list
   ...
   Kernel hardware providers:
       dca/0
   ```

   - **선택한 방식을 사용 안함으로 설정합니다.**

     ```
     # cryptoadm list -m provider=dca/0
     dca/0: CKM_RSA_PKCS, CKM_RSA_X_509, CKM_DSA, CKM_DES_CBC, CKM_DES3_CBC
     random is enabled.
     # cryptoadm disable provider=dca/0 mechanism=CKM_DES_CBC,CKM_DES3_CBC
     # cryptoadm list -p provider=dca/0
     dca/0: all mechanisms are enabled except CKM_DES_CBC,CKM_DES3_CBC.
     random is enabled.
     ```

   - **난수 생성기를 사용 안함으로 설정합니다.**

     ```
     # cryptoadm list -p provider=dca/0
     dca/0: all mechanisms are enabled. random is enabled.
     # cryptoadm disable provider=dca/0 random
     # cryptoadm list -p provider=dca/0
     dca/0: all mechanisms are enabled. random is disabled.
     ```

   - **모든 방식을 사용 안함으로 설정합니다. 난수 생성기는 사용 안함으로 설정하지 마십시오.**

     ```
     # cryptoadm list -p provider=dca/0
     dca/0: all mechanisms are enabled. random is enabled.
     # cryptoadm disable provider=dca/0 mechanism=all
     # cryptoadm list -p provider=dca/0
     dca/0: all mechanisms are disabled. random is enabled.
     ```

   - **하드웨어에서 모든 기능 및 방식을 사용 안함으로 설정합니다.**

     ```
     # cryptoadm list -p provider=dca/0
     dca/0: all mechanisms are enabled. random is enabled.
     # cryptoadm disable provider=dca/0 all
     # cryptoadm list -p provider=dca/0
     dca/0: all mechanisms are disabled. random is disabled.
     ```

예 12–30 하드웨어 공급자에서 방식 및 기능을 사용으로 설정

다음 예에서 하드웨어 조각에 사용 안함으로 설정된 방식이 선택적으로 사용으로 설정됩니다.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled except CKM_DES_ECB,CKM_DES3_ECB

.
random is enabled.
# cryptoadm enable provider=dca/0 mechanism=CKM_DES3_ECB
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled except CKM_DES_ECB.
random is enabled.
```

다음 예에서 난수 생성기만 사용으로 설정됩니다.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_MD5,CKM_MD5_HMAC,....
random is disabled.
# cryptoadm enable provider=dca/0 random
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_MD5,CKM_MD5_HMAC,....
random is enabled.
```

다음 예에서 방식만 사용으로 설정됩니다. 난수 생성기는 계속 사용할 수 없습니다.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_MD5,CKM_MD5_HMAC,....
random is disabled.
# cryptoadm enable provider=dca/0 mechanism=all
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled. random is disabled.
```

다음 예에서 보드의 모든 기능 및 방식이 사용으로 설정됩니다.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_DES_ECB,CKM_DES3_ECB.
random is disabled.
# cryptoadm enable provider=dca/0 all
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled. random is enabled.
```

▼ 모든 암호화 서비스를 새로 고치거나 다시 시작하는 방법

기본적으로 암호화 프레임워크는 사용으로 설정됩니다. 어떤 이유로 `kcfd` 데몬을 실패할 때 SMF(서비스 관리 기능)를 사용하여 암호화 서비스를 다시 시작할 수 있습니다. 자세한 내용은 `smf(5)` 및 `svcadm(1M)` 매뉴얼 페이지를 참조하십시오. 암호화 서비스 다시 시작이 영역에 미치는 영향은 219 페이지 “암호화 서비스 및 영역”을 참조하십시오.

시작하기 전에 Crypto Management 권한 프로파일에 할당되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 암호화 서비스의 상태를 확인합니다.

```
% svcs cryptosvc
 STATE          STIME    FMRI
offline         Dec_09   svc:/system/cryptosvc:default
```

3 암호화 서비스를 사용으로 설정합니다.

```
# svcadm enable svc:/system/cryptosvc
```

예 12–31 암호화 서비스 새로 고침

다음 예에서 암호화 서비스를 전역 영역에서 새로 고칩니다. 따라서 모든 비전역 영역의 커널 레벨 암호화 정책도 새로 고쳐집니다.

```
# svcadm refresh system/cryptosvc
```

◆ ◆ ◆ 13장

13 키 관리 프레임워크

Oracle Solaris의 키 관리 프레임워크(Key Management Framework, KMF) 기능은 공개 키 객체 관리용 도구 및 프로그래밍 인터페이스를 제공합니다. 공개 키 객체는 X.509 인증서 및 공개/개인 키 쌍을 포함합니다. 이러한 객체의 저장 형식은 다양할 수 있습니다. 또한 KMF는 응용 프로그램의 X.509 인증서 사용을 정의하는 정책 관리용 도구를 제공합니다. KMF는 타사 플러그인을 지원합니다.

공개 키 기술 관리

키 관리 프레임워크(KMF)는 공개 키 기술(PKI) 관리에 대한 통일된 접근법을 제공합니다. Oracle Solaris에는 PKI 기술을 이용하는 여러 다른 응용 프로그램이 있습니다. 각 응용 프로그램은 고유의 프로그래밍 인터페이스, 키 저장소 방식, 관리 유틸리티를 제공합니다. 응용 프로그램이 정책 시행 방식을 제공하는 경우 방식이 해당 응용 프로그램에만 적용됩니다. KMF와 함께 응용 프로그램은 통일된 관리 도구 세트, 단일 프로그래밍 인터페이스 세트, 단일 정책 시행 방식을 사용합니다. 이러한 기능은 해당 인터페이스를 채택한 모든 응용 프로그램의 PKI 요구를 관리합니다.

KMF는 다음 인터페이스를 사용하여 공개 키 기술 관리를 통일합니다.

- **pktool 명령** – 이 명령은 인증서와 같은 PKI 객체를 다양한 키 저장소에서 관리합니다.
- **kmfcfg 명령** – 이 명령은 PKI 정책 데이터베이스 및 타사 플러그인을 관리합니다.

  PKI 정책 결정은 검증 방법과 같은 작업을 포함합니다. 또한 PKI 정책은 인증서의 범위를 제한할 수 있습니다. 예를 들어, PKI 정책이 인증서를 특정 목적으로만 사용할 수 있는지 검증할 수 있습니다. 이러한 정책은 해당 인증서가 다른 요청에 사용되는 것을 금지합니다.
- **KMF 라이브러리** – 이 라이브러리는 기본 키 저장소 방식을 끌어내는 프로그래밍 인터페이스를 포함합니다.

응용 프로그램이 하나의 특정 키 저장소 방식을 선택할 필요는 없지만, 한 방식에서 다른 방식으로 마이그레이션할 수 있습니다. 지원되는 키 저장소는 PKCS #11, NSS, OpenSSL입니다. 라이브러리가 플러그인 가능한 프레임워크를 포함하므로 새 키 저장소 방식을 추가할 수 있습니다. 따라서 새 방식을 사용하는 응용 프로그램을 조금만 바꾸면 새 키 저장소를 사용할 수 있습니다.

주 – 실행 중인 OpenSSL의 버전을 확인하려면 `openssl version`을 입력합니다. 결과는 다음과 유사합니다.

```
OpenSSL 1.0.0d 8 Feb 2011
```

키 관리 프레임워크 유틸리티

KMF는 키 저장소 관리 방법을 제공하고 이러한 키 사용에 대한 전체 정책을 제공합니다. KMF는 세 가지 공개 키 기술에 대한 정책, 키, 인증서를 관리합니다.

- PKCS #11 공급자(암호화 프레임워크)의 토큰
- NSS(Network Security Services)
- OpenSSL, 파일 기반 키 저장소

`kmfcfg` 도구는 KMF 정책 항목을 만들고 수정, 삭제할 수 있습니다. 또한 프레임워크에 대한 플러그인을 관리합니다. KMF는 `pktool` 명령을 통해 키 저장소를 관리합니다. 자세한 내용은 `kmfcfg(1)` 및 `pktool(1)` 매뉴얼 페이지와 다음 절을 참조하십시오.

KMF 정책 관리

KMF 정책은 데이터베이스에 저장됩니다. 이 정책 데이터베이스는 KMF 프로그래밍 인터페이스를 사용하는 모든 응용 프로그램에서 내부적으로 액세스할 수 있습니다. 데이터베이스는 KMF 라이브러리에서 관리되는 키 및 인증서의 사용을 제약할 수 있습니다. 응용 프로그램이 인증서를 확인하려고 시도할 때 정책 데이터베이스를 검사합니다. `kmfcfg` 명령은 정책 데이터베이스를 수정합니다.

KMF 플러그인 관리

`kmfcfg` 명령은 다음과 같은 플러그인의 하위 명령을 제공합니다.

- `list plugin` – KMF에서 관리되는 플러그인을 나열합니다.
- `install` *plugin* – 모듈의 경로 이름으로 플러그인을 설치하고 플러그인에 대한 키 저장소를 만듭니다. KMF에서 플러그인을 제거하려면 키 저장소를 제거합니다.
- `uninstall` *plugin* – 키 저장소를 제거하여 KMF에서 플러그인을 제거합니다.

- `modify` *plugin* – `debug`와 같은 플러그인 코드에 저장된 옵션과 함께 플러그인이 실행되도록 합니다.

자세한 내용은 `kmfcfg(1)` 매뉴얼 페이지를 참조하십시오. 절차는 262 페이지 "KMF에서 타사 플러그인을 관리하는 방법"을 참조하십시오.

KMF 키 저장소 관리

KMF는 세 가지 공개 키 기술인 PKCS #11 토큰, NSS, OpenSSL에 대한 키 저장소를 관리합니다. 이러한 기술은 모두 `pktool` 명령으로 다음 작업을 수행할 수 있습니다.

- 자체 서명된 인증서를 생성합니다.
- 인증서 요청을 생성합니다.
- 대칭 키를 생성합니다.
- 공개/개인 키 쌍을 생성합니다.
- 서명할 외부 인증 기관(CA)에 보낼 PKCS #10 인증서 서명 요청(CSR)을 생성합니다.
- PKCS #10 CSR을 서명합니다.
- 객체를 키 저장소로 가져옵니다.
- 키 저장소의 객체를 나열합니다.
- 키 저장소에서 객체를 삭제합니다.
- CRL을 다운로드합니다.

PKCS #11 및 NSS 기술의 경우 `pktool` 명령으로 암호문을 생성하여 PIN을 설정할 수도 있습니다.

- 키 저장소에 대한 암호문을 생성합니다.
- 키 저장소의 객체에 대한 암호문을 생성합니다.

`pktool` 유틸리티 사용의 예는 `pktool(1)` 매뉴얼 페이지와 251 페이지 "키 관리 프레임워크 사용(작업 맵)"을 참조하십시오.

키 관리 프레임워크 사용(작업)

이 절은 `pktool` 명령을 사용하여 암호, 암호문, 파일, 키 저장소, 인증서, CRL과 같은 공개 키 객체를 관리하는 방법을 설명합니다.

키 관리 프레임워크 사용(작업 맵)

키 관리 프레임워크(KMF)를 통해 중앙에서 공개 키 기술을 관리할 수 있습니다.

작업	설명	수행 방법
인증서를 만듭니다.	PKCS #11, NSS, SSL에서 사용할 인증서를 만듭니다.	252 페이지 “pktool gencert 명령을 사용하여 인증서를 만드는 방법”
인증서를 내보냅니다.	인증서와 해당 지원 키가 포함된 파일을 만듭니다. 파일은 암호로 보호할 수 있습니다.	255 페이지 “PKCS #12 형식의 인증서 및 개인 키를 내보내는 방법”
인증서를 가져옵니다.	다른 시스템에서 인증서를 가져옵니다.	253 페이지 “인증서를 키 저장소로 가져오는 방법”
	다른 시스템에서 PKCS #12 형식의 인증서를 가져옵니다.	예 13–2
암호문을 생성합니다.	PKCS #11 키 저장소나 NSS 키 저장소에 액세스하기 위한 암호문을 생성합니다.	256 페이지 “pktool setpin 명령을 사용하여 암호문을 생성하는 방법”
대칭 키를 생성합니다.	파일 암호화, 파일의 MAC 만들기 및 응용 프로그램에 사용할 대칭 키를 생성합니다.	224 페이지 “pktool 명령을 사용하여 대칭 키를 생성하는 방법”
키 쌍을 생성합니다.	응용 프로그램에 사용할 공개/개인 키 쌍을 생성합니다.	257 페이지 “pktool genkeypair 명령을 사용하여 키 쌍을 생성하는 방법”
PKCS #10 CSR을 생성합니다.	서명할 외부 인증 기관(CA)을 위한 PKCS #10 인증서 서명 요청(CSR)을 생성합니다.	pktool(1) 매뉴얼 페이지
PKCS #10 CSR을 서명합니다.	PKCS #10 CSR을 서명합니다.	261 페이지 “pktool signcsr 명령을 사용하여 인증서 요청을 서명하는 방법”
KMF에 플러그인을 추가합니다.	플러그인을 설치, 수정, 나열합니다. 또한 KMF에서 플러그인을 제거합니다.	262 페이지 “KMF에서 타사 플러그인을 관리하는 방법”

▼ pktool gencert 명령을 사용하여 인증서를 만드는 방법

이 절차는 자체 서명된 인증서를 만들고 PKCS #11 키 저장소에 인증서를 저장합니다. 이 작업의 일부로 RSA 공개/개인 키 쌍도 생성됩니다. 개인 키는 인증서와 함께 키 저장소에 저장됩니다.

1 자체 서명된 인증서를 생성합니다.

```
% pktool gencert [keystore=keystore] label=label-name \
subject=subject-DN serial=hex-serial-number
```

`keystore=`*keystore* 공개 키 객체의 유형별로 키 저장소를 지정합니다. 값은 `nss`, `pkcs11`, `ssl`일 수 있습니다. 이 키워드는 선택 사항입니다.

`label=`*label-name* 발행자가 인증서에 제공하는 고유한 이름을 지정합니다.

`subject=`*subject-DN* 인증서에 대한 식별 이름을 지정합니다.

`serial=`*hex-serial-number* 16진수 형식의 일련 번호를 지정합니다. 인증서의 발행자가 `0x0102030405`와 같은 숫자를 선택합니다.

2 키 저장소의 내용을 확인합니다.

```
% pktool list
Found number certificates.
1. (X.509 certificate)
      Label:  label-name
      ID: Fingerprint that binds certificate to private key
      Subject: subject-DN
      Issuer:  distinguished-name
      Serial:  hex-serial-number
n. ...
```

이 명령은 키 저장소의 모든 인증서를 나열합니다. 다음 예에서 키 저장소는 하나의 인증서만 포함합니다.

예 13–1 pktool을 사용하여 자체 서명된 인증서 만들기

다음 예에서 `My Company`의 사용자가 자체 서명된 인증서를 만들고 PKCS #11 객체용 키 저장소에 인증서를 저장합니다. 키 저장소는 처음에 비어 있습니다. 키 저장소가 초기화되지 않은 경우 softtoken의 PIN이 `changeme`입니다.

```
% pktool gencert keystore=pkcs11 label="My Cert" \
subject="C=US, O=My Company, OU=Security Engineering Group, CN=MyCA" \
serial=0x000000001
Enter pin for Sun Software PKCS#11 softtoken:     Type PIN for token
```

```
% pktool list
Found 1 certificates.
1. (X.509 certificate)
      Label: My Cert
      ID: 12:82:17:5f:80:78:eb:44:8b:98:e3:3c:11:c0:32:5e:b6:4c:ea:eb
      Subject: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
      Issuer: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
      Serial: 0x01
```

▼ 인증서를 키 저장소로 가져오는 방법

이 절차는 PEM 또는 원시 DER로 인코딩된 PKI 정보가 담긴 파일을 키 저장소로 가져오는 방법을 설명합니다. 내보내기 절차는 예 13–4를 참조하십시오.

1 인증서를 가져옵니다.

```
% pktool import keystore=keystore infile=infile-name label=label-name
```

2 개인 PKI 객체를 가져오는 경우 프롬프트가 표시되면 암호를 제공합니다.

a. 프롬프트에 파일의 암호를 제공합니다.

PKCS #12 형식의 내보내기 파일과 같은 개인 PKI 정보를 가져오는 경우 파일에 암호가 필요합니다. 가져오기 중인 파일의 작성자가 사용자에게 PKCS #12 암호를 알려줍니다.

```
Enter password to use for accessing the PKCS12 file:      Type PKCS #12 password
```

b. 프롬프트에 키 저장소의 암호를 입력합니다.

```
Enter pin for Sun Software PKCS#11 softtoken:      Type PIN for token
```

3 키 저장소의 내용을 확인합니다.

```
% pktool list
Found number certificates.
1. (X.509 certificate)
      Label:  label-name
      ID: Fingerprint that binds certificate to private key
      Subject: subject-DN
      Issuer:  distinguished-name
      Serial:  hex-serial-number
2. ...
```

예 13–2 PKCS #12 파일을 키 저장소로 가져오기

다음 예에서 사용자가 PKCS #12 파일을 타사로부터 가져옵니다. `pktool import` 명령은 `gracedata.p12` 파일에서 개인 키 및 인증서를 추출하여 사용자의 선호 키 저장소에 저장합니다.

```
% pktool import keystore=pkcs11 infile=gracedata.p12 label=GraceCert
Enter password to use for accessing the PKCS12 file:      Type PKCS #12 password
Enter pin for Sun Software PKCS#11 softtoken:      Type PIN for token
Found 1 certificate(s) and 1 key(s) in gracedata.p12
% pktool list
Found 1 certificates.
1. (X.509 certificate)
        Label: GraceCert
        ID: 12:82:17:5f:80:78:eb:44:8b:98:e3:3c:11:c0:32:5e:b6:4c:ea:eb
        Subject: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
        Issuer: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
        Serial: 0x01
```

예 13–3 X.509 인증서를 키 저장소로 가져오기

다음 예에서 사용자가 PEM 형식의 X.509 인증서를 사용자의 선호 키 저장소로 가져옵니다. 이 공개 인증서는 암호로 보호되지 않습니다. 사용자의 공개 키 저장소도 암호로 보호되지 않습니다.

```
% pktool import keystore=pkcs11 infile=somecert.pem label="TheirCompany Root Cert"
% pktool list
Found 1 certificates.
1. (X.509 certificate)
        Label: TheirCompany Root Cert
        ID: 21:ae:83:98:24:d1:1f:cb:65:5b:48:75:7d:02:47:cf:98:1f:ec:a0
        Subject: C=US, O=TheirCompany, OU=Security, CN=TheirCompany Root CA
        Issuer: C=US, O=TheirCompany, OU=Security, CN=TheirCompany Root CA
        Serial: 0x01
```

▼ PKCS #12 형식의 인증서 및 개인 키를 내보내는 방법

PKCS #12 형식의 파일을 만들어서 개인 키 및 연관된 X.509 인증서를 다른 시스템으로 내보낼 수 있습니다. 파일 액세스는 암호로 보호됩니다.

1 내보낼 인증서를 찾습니다.

```
% pktool list
Found number certificates.
1. (X.509 certificate)
      Label:  label-name
      ID: Fingerprint that binds certificate to private key
      Subject: subject-DN
      Issuer:  distinguished-name
      Serial:  hex-serial-number
2. ...
```

2 키 및 인증서를 내보냅니다.

pktool list 명령에서 키 저장소 및 레이블을 사용합니다. 내보내기 파일에 대한 파일 이름을 제공합니다. 이름에 공백이 포함된 경우 이름 주위를 큰 따옴표로 둘러쌉니다.

```
% pktool export keystore=keystore outfile=outfile-name label=label-name
```

3 내보내기 파일을 암호로 보호합니다.

프롬프트에 키 저장소의 현재 암호를 입력합니다. 이 시점에 내보내기 파일의 암호를 만듭니다. 파일을 가져올 때 수신자가 이 암호를 제공해야 합니다.

```
Enter pin for Sun Software PKCS#11 softtoken:      Type PIN for token
Enter password to use for accessing the PKCS12 file:      Create PKCS #12 password
```

참고 – 내보내기 파일에서 별도로 암호를 보냅니다. 최적의 사용법으로 대역 외에서(예: 전화 통화 중) 암호를 알려줄 것을 제안합니다.

예 13–4 PKCS #12 형식의 인증서 및 개인 키 내보내기

다음 예에서 사용자가 개인 키 및 연관된 X.509 인증서를 표준 PKCS #12 파일로 내보냅니다. 이 파일을 다른 키 저장소로 가져올 수 있습니다. PKCS #11 암호는 소스 키

저장소를 보호합니다. PKCS #12 암호는 PKCS #12 파일의 개인 데이터를 보호하는 데 사용됩니다. 파일을 가져오려면 이 암호가 필요합니다.

```
% pktool list
Found 1 certificates.
1. (X.509 certificate)
      Label: My Cert
      ID: 12:82:17:5f:80:78:eb:44:8b:98:e3:3c:11:c0:32:5e:b6:4c:ea:eb
      Subject: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
      Issuer: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
      Serial: 0x01

% pktool export keystore=pkcs11 outfile=mydata.p12 label="My Cert"
Enter pin for Sun Software PKCS#11 softtoken:      Type PIN for token
Enter password to use for accessing the PKCS12 file:      Create PKCS #12 password
```

그런 다음 사용자가 수신자에게 전화를 걸어서 PKCS #12 암호를 알려줍니다.

▼ pktool setpin 명령을 사용하여 암호문을 생성하는 방법

키 저장소의 객체에 대한, 그리고 키 저장소 자체에 대한 암호문을 생성할 수 있습니다. 객체나 키 저장소에 액세스하려면 암호문이 필요합니다. 키 저장소의 객체에 대한 암호문 생성의 예는 예 13–4를 참조하십시오.

1 키 저장소 액세스를 위한 암호문을 생성합니다.

```
% pktool setpin keystore=nss|pkcs11 dir=directory
```

2 프롬프트에 대답합니다.

키 저장소에 아직 암호가 설정되지 않은 경우 Return 키를 눌러 암호를 만듭니다.

```
Enter current token passphrase:      Press the Return key
Create new passphrase:      Type the passphrase that you want to use
Re-enter new passphrase:      Retype the passphrase
Passphrase changed.
```

이제 키 저장소가 *passphrase*(으)로 보호됩니다. 암호문을 잊어버린 경우 키 저장소의 객체에 액세스할 수 없습니다.

예 13–5 키 저장소를 암호문으로 보호

다음 예는 NSS 데이터베이스에 대한 암호문을 설정하는 방법을 보여줍니다. 암호문을 만든 적이 없으므로 사용자가 첫번째 프롬프트에 Return 키를 누릅니다.

```
% pktool setpin keystore=nss dir=/var/nss
Enter current token passphrase:      Press the Return key
```

```
Create new passphrase:     has8n0NdaH
Re-enter new passphrase:   has8n0NdaH
Passphrase changed.
```

▼ pktool genkeypair 명령을 사용하여 키 쌍을 생성하는 방법

일부 응용 프로그램은 공개/개인 키 쌍이 필요합니다. 이 절차에서 이러한 키 쌍을 만들어서 저장합니다.

1 (옵션) 키 저장소를 사용하려면 키 저장소를 만듭니다.

- **PKCS #11 키 저장소를 만들고 초기화하려면** **256 페이지 "pktool setpin 명령을 사용하여 암호문을 생성하는 방법"을 참조하십시오.**
- **NSS 키 저장소를 만들고 초기화하려면** **예 13–5를 참조하십시오.**

2 키 쌍을 만듭니다.

다음 방법 중 하나를 사용합니다.

- **키 쌍을 만들어서 파일에 저장합니다.**

  디스크의 파일에서 직접 키를 읽는 응용 프로그램의 경우 파일 기반 키가 생성됩니다. 일반적으로, OpenSSL 암호화 라이브러리를 직접 사용하는 응용 프로그램은 응용 프로그램의 키 및 인증서를 파일에 저장해야 합니다.

  주 – `file` 키 저장소는 타원 곡선(ec) 키 및 인증서를 지원하지 않습니다.

```
% pktool genkeypair keystore=file outkey=key-filename \
[format=der|pem] [keytype=rsa|dsa] [keylen=key-size]
```

`keystore=file`
  `file` 값은 키에 대해 파일 유형의 저장소 위치를 지정합니다.

`outkey=`*key-filename*
  키 쌍이 저장된 파일의 이름을 지정합니다.

`format=der|pem`
  키 쌍의 인코딩 형식을 지정합니다. `der` 출력은 이진이고 `pem` 출력은 ASCII입니다.

`keytype=rsa|dsa`
  `file` 키 저장소에 저장할 수 있는 키 쌍의 유형을 지정합니다. 정의는 DSA 및 RSA를 참조하십시오.

keylen=*key-size*
키의 길이를 비트 단위로 지정합니다. 숫자는 8로 나눌 수 있어야 합니다. 가능한 키 크기를 결정하려면 `cryptoadm list -vm` 명령을 사용합니다.

- **키 쌍을 만들어서 PKCS #11 키 저장소에 저장합니다.**

  이 방법을 사용하기 전에 단계 1을 완료해야 합니다.

  PKCS #11 키 저장소는 하드웨어 장치에 객체를 저장하는 데 사용됩니다. 이 장치에는 암호화 프레임워크로 플러그인할 수 있는 Sun Crypto Accelerator 6000 카드, 신뢰된 플랫폼 모듈(TPM) 장치, 스마트 카드 등이 있습니다. 또한 PKCS #11을 사용하여 `softtoken`(디스크의 개인 하위 디렉토리에 객체를 저장하는 소프트웨어 기반 토큰)에 객체를 저장할 수 있습니다. 자세한 내용은 `pkcs11_softtoken(5)` 매뉴얼 페이지를 참조하십시오.

  지정한 레이블로 키 저장소에서 키 쌍을 검색할 수 있습니다.

  ```
  % pktool genkeypair label=key-label \
  [token=token[:manuf[:serial]]] \
  [keytype=rsa|dsa|ec]  [curve=ECC-Curve-Name]]\
  [keylen=key-size] [listcurves]
  ```

  label=*key-label*
  키 쌍의 레이블을 지정합니다. 키 쌍은 자체 레이블로 키 저장소에서 검색할 수 있습니다.

  token=*token*[:*manuf*[:*serial*]]
  토큰 이름을 지정합니다. 기본적으로 토큰 이름은 `Sun Software PKCS#11 softtoken`입니다.

  keytype=rsa|dsa|ec [curve=*ECC-Curve-Name*]
  키 쌍 유형을 지정합니다. 타원 곡선(ec) 유형의 경우 선택적으로 곡선 이름을 지정합니다. 곡선 이름은 `listcurves` 옵션에 출력으로 나열됩니다.

  keylen=*key-size*
  키의 길이를 비트 단위로 지정합니다. 숫자는 8로 나눌 수 있어야 합니다.

  listcurves
  ec 키 유형에 대한 `curve=` 옵션에 값으로 사용할 수 있는 타원 곡선 이름을 나열합니다.

- **키 쌍을 생성하여 NSS 키 저장소에 저장합니다.**

  NSS 키 저장소는 주 암호화 인터페이스로 NSS를 이용하는 서버에서 사용됩니다. 예를 들어, Oracle iPlanet 웹 서버는 객체 저장소에 NSS 데이터베이스를 사용합니다.

  이 방법을 사용하기 전에 단계 1을 완료해야 합니다.

  ```
  % pktool keystore=nss genkeypair label=key-nickname \
  [token=token[:manuf[:serial]]] \
  [dir=directory-path] [prefix=database-prefix] \
  [keytype=rsa|dsa|ec] [curve=ECC-Curve-Name]] \
  [keylen=key-size] [listcurves]
  ```

`keystore=nss`
`nss` 값은 키에 대해 NSS 유형의 저장소 위치를 사용합니다.

`label=`*nickname*
키 쌍의 레이블을 지정합니다. 키 쌍은 자체 레이블로 키 저장소에서 검색할 수 있습니다.

`token=`*token*[`:`*manuf*[`:`*serial*]]
토큰 이름을 지정합니다. 기본적으로 토큰은 `Sun Software PKCS#11 softtoken`입니다.

`dir=`*directory*
NSS 데이터베이스에 대한 디렉토리 경로를 지정합니다. 기본적으로 *directory*가 현재 디렉토리입니다.

`prefix=`*database-prefix*
NSS 데이터베이스에 대한 접두어를 지정합니다. 기본값은 접두어 없음입니다.

`keytype=rsa|dsa|ec [curve=`*ECC-Curve-Name*`]`
키 쌍 유형을 지정합니다. 타원 곡선 유형의 경우 선택적으로 곡선 이름을 지정합니다. 곡선 이름은 `listcurves` 옵션에 출력으로 나열됩니다.

`keylen=`*key-size*
키의 길이를 비트 단위로 지정합니다. 숫자는 8로 나눌 수 있어야 합니다.

`listcurves`
`ec` 키 유형에 대한 `curve=` 옵션에 값으로 사용할 수 있는 타원 곡선 이름을 나열합니다.

3 (옵션) 키가 존재하는지 확인합니다.

키를 저장한 위치에 따라 다음 명령 중 하나를 사용합니다.

- ***key-filename* 파일에서 키를 확인합니다.**

```
% pktool list keystore=file objtype=key infile=key-filename
Found n keys.
Key #1 - keytype:location (keylen)
```

- **PKCS #11 키 저장소에서 키를 확인합니다.**

```
$ pktool list objtype=key
Enter PIN for keystore:
Found n keys.
Key #1 - keytype:location (keylen)
```

- **NSS 키 저장소에서 키를 확인합니다.**

```
% pktool list keystore=nss dir=directory objtype=key
```

예 13–6 pktool 명령을 사용하여 키 쌍 만들기

다음 예에서 사용자가 처음으로 PKCS #11 키 저장소를 만듭니다. RSA 키 쌍에 대한 키 크기를 결정한 후에 응용 프로그램에 대한 키 쌍을 생성합니다. 마지막으로, 키 쌍이 키 저장소에 있는지 확인합니다. RSA 키 쌍의 두번째 인스턴스를 하드웨어에 저장할 수 있습니다. 사용자가 token 인수를 지정하지 않았으므로 키 쌍이 Sun Software PKCS#11 softtoken으로 저장됩니다.

```
# pktool setpin
Create new passphrase:      Easily remembered, hard-to-detect password
Re-enter new passphrase:    Retype password
Passphrase changed.
% cryptoadm list -vm | grep PAIR
...
CKM_DSA_KEY_PAIR_GEN         512  1024 .  .  .
CKM_RSA_PKCS_KEY_PAIR_GEN    256  4096 .  .  .
...
CKM_RSA_PKCS_KEY_PAIR_GEN    512  2048 X  .  .
ecc: CKM_EC_KEY_PAIR_GEN,CKM_ECDH1_DERIVE,CKM_ECDSA,CKM_ECDSA_SHA1
% pktool genkeypair label=specialappkeypair keytype=rsa keylen=2048
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

% pktool list
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

Found 1 keys.
Key #1 - keypair:  specialappkeypair (2048 bits)
```

예 13–7 타원 곡선 알고리즘을 사용하는 키 쌍 만들기

다음 예에서 사용자가 타원 곡선(ec) 키 쌍을 키 저장소에 추가하고, 곡선 이름을 지정하고, 키 쌍이 키 저장소에 존재하는지 확인합니다.

```
% pktool genkeypair listcurves
secp112r1, secp112r2, secp128r1, secp128r2, secp160k1
.
.
.
c2pnb304w1, c2tnb359v1, c2pnb368w1, c2tnb431r1, prime192v2
prime192v3
% pktool genkeypair label=eckeypair keytype=ec curves=c2tnb431r1
% pktool list
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

Found 2 keys.
Key #1 - keypair:  specialappkeypair (2048 bits)
Key #2 - keypair:  eckeypair (c2tnb431r1)
```

▼ pktool signcsr 명령을 사용하여 인증서 요청을 서명하는 방법

이 절차는 PKCS #10 인증서 서명 요청(CSR)을 서명하는 데 사용됩니다. CSR은 PEM 또는 DER 형식일 수 있습니다. 서명 프로세스가 X.509 v3 인증서를 발행합니다. PKCS #10 CSR을 생성하려면 pktool(1) 매뉴얼 페이지를 참조하십시오.

시작하기 전에 인증 기관(CA)으로서 CSR을 받아서 파일에 저장했습니다.

1 pktool signcsr 명령의 필수 인수를 위해 다음 정보를 수집합니다.

`signkey` 서명자의 키를 PKCS #11 키 저장소에 저장한 경우 `signkey`는 이 개인 키를 검색하는 *label*입니다.

서명자의 키를 NSS 키 저장소나 file 키 저장소에 저장한 경우 `signkey`는 이 개인 키를 보유하는 파일 이름입니다.

`csr` CSR의 파일 이름을 지정합니다.

`serial` 서명된 인증서의 일련 번호를 지정합니다.

`outcer` 서명된 인증서에 대한 파일 이름을 지정합니다.

`issuer` CA 발행자 이름을 식별 이름(DN) 형식으로 지정합니다.

`signcsr` 하위 명령의 선택적 인수에 대한 내용은 pktool(1) 매뉴얼 페이지를 참조하십시오.

2 요청을 서명하고 인증서를 발행합니다.

예를 들어, 다음 명령은 PKCS #11 저장소에서 서명자의 키로 인증서를 서명합니다.

```
# pktool signcsr signkey=CASigningKey \
csr=fromExampleCoCSR \
serial=0x12345678 \
outcert=ExampleCoCert2010 \
issuer="O=Oracle Corporation, \
     OU=Oracle Solaris Security Technology, L=Redwood City, ST=CA, C=US, \
     CN=rootsign Oracle"
```

다음 명령은 파일에서 서명자의 키로 인증서를 서명합니다.

```
# pktool signcsr signkey=CASigningKey \
csr=fromExampleCoCSR \
serial=0x12345678 \
outcert=ExampleCoCert2010 \
issuer="O=Oracle Corporation, \
     OU=Oracle Solaris Security Technology, L=Redwood City, ST=CA, C=US, \
     CN=rootsign Oracle"
```

3 **인증서를 요청자에게 보냅니다.**

전자 메일, 웹 사이트 또는 기타 방식을 사용하여 인증서를 요청자에게 전달할 수 있습니다.

예를 들어, 전자 메일을 사용하여 `ExampleCoCert2010` 파일을 요청자에게 보낼 수 있습니다.

▼ KMF에서 타사 플러그인을 관리하는 방법

키 저장소 이름을 제공하여 플러그인을 식별합니다. KMF에 플러그인을 추가할 때 소프트웨어가 키 저장소 이름으로 플러그인을 식별합니다. 플러그인이 옵션을 받아들이도록 정의할 수 있습니다. 이 절차는 KMF에서 플러그인을 제거하는 방법을 포함합니다.

1 **플러그인을 설치합니다.**

```
% /usr/bin/kmfcfg install keystore=keystore-name \
modulepath=path-to-plugin [option="option-string"]
```

구문 설명은 다음과 같습니다.

keystore-name – 제공한 키 저장소에 대한 고유한 이름입니다.

path-to-plugin – KMF 플러그인에 대한 공유 라이브러리 객체의 전체 경로를 지정합니다.

option-string – 공유 라이브러리 객체의 선택적 인수를 지정합니다.

2 **플러그인을 나열합니다.**

```
% kmfcfg list plugin
keystore-name:path-to-plugin [(built-in)] | [;option=option-string]
```

3 **플러그인을 제거하려면 설치를 해제하고 제거되었는지 확인합니다.**

```
% kmfcfg uninstall keystore=keystore-name
% kmfcfg plugin list
```

예 13–8 KMF 플러그인을 옵션과 함께 호출

다음 예에서 관리자가 KMF 플러그인을 사이트 특정 디렉토리에 저장합니다. 플러그인이 debug 옵션을 받아들이도록 정의됩니다. 관리자가 플러그인을 추가하고 플러그인이 설치되었는지 확인합니다.

```
# /usr/bin/kmfcfg install keystore=mykmfplug \
modulepath=/lib/security/site-modules/mykmfplug.so
# kmfcfg list plugin
KMF plugin information:
-----------------------
pkcs11:kmf_pkcs11.so.1 (built-in)
file:kmf_openssl.so.1 (built-in)
```

```
nss:kmf_nss.so.1 (built-in)
mykmfplug:/lib/security/site-modules/mykmfplug.so
# kmfcfg modify plugin keystore=mykmfplug option="debug"
# kmfcfg list plugin
KMF plugin information:
----------------------
...
mykmfplug:/lib/security/site-modules/mykmfplug.so;option=debug
```

이제 플러그인이 디버깅 모드로 실행됩니다.

제 5 부

인증 서비스 및 보안 통신

이 절에서는 네트워크로 연결되지 않은 시스템 또는 두 시스템 간에 구성할 수 있는 인증 서비스를 설명합니다.

- 14 장, “네트워크 서비스 인증(작업)”
- 15 장, “PAM 사용”
- 16 장, “SASL 사용”
- 17 장, “Secure Shell 사용(작업)”
- 18 장, “Secure Shell(참조)”

인증된 사용자 및 시스템의 네트워크를 구성하려면 제6부를 참조하십시오.

◆ ◆ ◆ 14장

14 네트워크 서비스 인증(작업)

이 장에서는 보안 RPC를 사용하여 NFS 마운트에서 호스트 및 사용자를 인증하는 방법에 대한 정보를 제공합니다. 다음은 이 장에 포함된 항목 목록입니다.

보안 RPC 개요

보안 RPC(원격 프로시저 호출)는 인증 방식을 사용하여 원격 프로시저를 보호합니다. Diffie-Hellman 인증 방식은 서비스에 대해 요청하는 호스트 및 사용자를 모두 인증합니다. 인증 방식에서는 데이터 암호화 표준(DES) 암호화를 사용합니다. 보안 RPC를 사용하는 응용 프로그램에는 NFS 및 NIS 이름 지정 서비스가 포함됩니다.

NFS 서비스 및 보안 RPC

NFS를 사용하여 여러 호스트에서 네트워크를 통해 파일을 공유할 수 있습니다. NFS 서비스에서는 서버에 여러 클라이언트에 대한 데이터 및 리소스가 포함됩니다. 클라이언트는 서버가 클라이언트와 공유하는 파일 시스템에 대한 액세스 권한을 가집니다. 클라이언트 시스템에 로그인한 사용자는 서버에서 파일 시스템을 마운트하여 파일 시스템에 액세스할 수 있습니다. 클라이언트 시스템의 사용자에게는 파일이 로컬에 있는 것처럼 보입니다. NFS의 가장 일반적인 용도 중 하나는 시스템을 사무실에 설치하고, 모든 사용자 파일을 중앙 위치에 저장하는 것입니다. `mount` 명령에 대한 `-nosuid` 옵션과 같은 NFS 서비스의 기능을 사용하면 무단 사용자가 장치 및 파일 시스템을 열지 못하도록 할 수 있습니다.

NFS 서비스는 보안 RPC를 사용하여 네트워크를 통해 요청하는 사용자를 인증합니다. 이 프로세스를 **보안** *NFS*라고 합니다. Diffie-Hellman 인증 방식 AUTH_DH에서는 DES 암호화를 사용하여 인증된 액세스를 확인합니다. AUTH_DH 방식은 AUTH_DES라고 부르기도 합니다. 자세한 내용은 다음을 참조하십시오.

- 보안 NFS를 설정하고 관리하려면 **Oracle Solaris 관리: 네트워크 서비스**의 "보안 NFS 시스템 관리"를 참조하십시오.
- RPC 인증에 포함된 트랜잭션에 대한 개요는 269 페이지 "Diffie-Hellman 인증 구현"을 참조하십시오.

보안 NFS에서 DES 암호화

데이터 암호화 표준(DES) 암호화 함수는 56비트 키를 사용하여 데이터를 암호화합니다. 두 자격 증명 사용자 또는 주체가 동일한 DES 키를 알고 있는 경우 키를 사용하여 텍스트를 암호화하고 해독함으로써 비밀 통신이 가능합니다. DES는 비교적 빠른 암호화 방식입니다.

DES 키만 사용할 경우 침입자가 동일한 키로 암호화된 암호화 텍스트 메시지를 충분히 수집하여 키를 알아내고 메시지를 해독할 수 있다는 위험이 있습니다. 이러한 이유로 보안 NFS와 같은 보안 시스템에서는 키를 자주 변경해야 합니다.

Kerberos 인증

Kerberos는 MIT에서 개발한 인증 시스템입니다. Kerberos의 일부 암호화는 DES를 기준으로 합니다. Kerberos V4 지원은 더 이상 보안 RPC의 일부로 제공되지 않습니다. 하지만 RPCSEC_GSS를 사용하는 Kerberos V5의 클라이언트측 및 서버측 구현은 이 릴리스에 포함되어 있습니다. 자세한 내용은 19 장, "Kerberos 서비스 소개"를 참조하십시오.

Diffie-Hellman 인증 및 보안 RPC

Diffie-Hellman(DH) 사용자 인증 방식은 침입자가 알아내기가 쉽지 않습니다. 클라이언트와 서버는 각자 개인 키를 가지며, 공개 키와 함께 사용하여 공통 키를 만들어 냅니다. 개인 키는 **비밀 키**라고도 합니다. 클라이언트와 서버는 공통 키를 사용하여 서로 통신합니다. 공통 키는 DES와 같이 합의한 암호화 함수를 사용하여 암호화됩니다.

인증은 보내는 시스템에서 공통 키를 사용하여 현재 시간을 암호화할 수 있는 기능을 기준으로 합니다. 그런 다음 받는 시스템에서 해독하고 현재 시간과 비교하여 확인합니다. 클라이언트와 서버의 시간은 동기화되어야 합니다. 자세한 내용은 **Oracle Solaris 관리: 네트워크 서비스**의 "NTP(Network Time Protocol) 관리(작업)"를 참조하십시오.

공개 키와 개인 키는 NIS 데이터베이스에 저장됩니다. NIS는 키를 publickey 맵에 저장합니다. 이 파일에는 모든 잠재 사용자에 대한 공개 키 및 개인 키가 포함되어 있습니다.

시스템 관리자는 NIS 맵을 설정하고 각 사용자에 대한 공개 키 및 개인 키를 생성할 책임이 있습니다. 개인 키는 사용자의 암호를 사용하여 암호화된 형태로 저장됩니다. 이 프로세스는 개인 키를 사용자만 알도록 합니다.

Diffie-Hellman 인증 구현

이 절에서는 Diffie-Hellman 인증(AUTH_DH)을 사용하는 클라이언트-서버 세션에서 일련의 트랜잭션을 설명합니다.

보안 RPC에 대한 공개 키 및 개인 키 생성

트랜잭션 이전에 관리자는 newkey 또는 nisaddcred 명령을 실행하여 공개 키 및 비밀 키를 생성합니다. 각 사용자는 고유한 공개 키와 비밀 키를 가집니다. 공개 키는 공용 데이터베이스에 저장됩니다. 비밀 키는 동일한 데이터베이스에 암호화된 형태로 저장됩니다. chkey 명령은 키 쌍을 변경합니다.

보안 RPC에 대해 keylogin 명령 실행

대개 로그인 암호는 보안 RPC 암호와 동일합니다. 이 경우 keylogin 명령이 필요하지 않습니다. 하지만 암호가 다른 경우 사용자가 로그인한 다음 keylogin 명령을 실행해야 합니다.

keylogin 명령은 사용자에게 보안 RPC 암호를 물어봅니다. 그런 다음 명령은 암호를 사용하여 비밀 키를 해독합니다. 그런 다음 keylogin 명령은 해독된 비밀 키를 **키 서버** 프로그램에 전달합니다. 키 서버는 모든 컴퓨터의 로컬 인스턴스에 있는 RPC 서비스입니다. 키 서버는 해독된 비밀 키를 저장하고 사용자가 서버와 보안 RPC 트랜잭션을 시작할 때까지 기다립니다.

로그인 암호와 RPC 암호가 같을 경우 로그인 프로세스는 비밀 키를 키 서버에 전달합니다. 암호가 서로 달라야 하는 경우 사용자는 항상 keylogin 명령을 실행해야 합니다. keylogin 명령이 ~/.login, ~/.cshrc 또는 ~/.profile 파일과 같은 사용자의 환경 구성 파일에 포함된 경우 사용자가 로그인할 때마다 keylogin 명령이 자동으로 실행됩니다.

보안 RPC에 대한 컨버세이션 키 생성

사용자가 서버와 트랜잭션을 시작하면 다음이 발생합니다.

1. 키 서버가 임의로 컨버세이션 키를 생성합니다.
2. 커널은 컨버세이션 키 및 기타 정보를 사용하여 클라이언트의 시간 기록을 암호화합니다.
3. 키 서버는 공개 키 데이터베이스에서 서버의 공개 키를 조회합니다. 자세한 내용은 publickey(4) 매뉴얼 페이지를 참조하십시오.

4. 키 서버는 클라이언트의 비밀 키와 서버의 공개 키를 사용하여 공통 키를 만듭니다.
5. 키 서버는 공통 키를 사용하여 컨버세이션 키를 암호화합니다.

보안 RPC에서 서버와 처음 연결

그러면 암호화된 시간 기록 및 암호화된 컨버세이션 키를 포함하는 전송이 서버로 보내집니다. 전송에는 자격 증명 및 검증기가 포함됩니다. 자격 증명에는 세 가지 구성 요소가 포함됩니다.

- 클라이언트의 네트워크 이름
- 공통 키를 사용하여 암호화된 컨버세이션 키
- 컨버세이션 키를 사용하여 암호화된 "창"

창은 서버의 시계와 클라이언트의 시간 기록 사이에 허용되어야 한다고 클라이언트가 요구하는 시간 차이입니다. 서버의 시계와 시간 기록 사이의 차이가 창보다 클 경우 서버는 클라이언트의 요청을 거부합니다. 클라이언트가 RPC 세션을 시작하기 전에 먼저 서버와 동기화하므로 일반적인 상황에서 이러한 거부는 발생하지 않습니다.

클라이언트의 검증기에는 다음이 포함됩니다.

- 암호화된 시간 기록
- 1이 작은 지정된 창의 암호화된 검증기

창 검증기는 누군가 사용자를 가장하고자 하는 경우 필요합니다. 가장하는 사람은 자격 증명 및 검증기의 암호화된 필드를 채우는 대신 임의의 비트를 삽입하는 프로그램을 작성할 수 있습니다. 서버는 컨버세이션 키를 임의의 키로 해독합니다. 그런 다음 서버는 키를 사용하여 창 및 시간 기록 해독을 시도합니다. 결과는 임의의 숫자입니다. 하지만 수천 번의 시도 후 임의의 창/시간 기록 쌍은 인증 시스템을 통과할 수 있습니다. 창 검증기는 가짜 자격 증명을 인증할 수 있는 가능성을 줄입니다.

보안 RPC에서 컨버세이션 키 해독

서버가 클라이언트로부터 전송을 수신하면 다음이 발생합니다.

1. 서버에 로컬인 키 서버가 공개 키 데이터베이스에서 클라이언트의 공개 키를 조회합니다.
2. 키 서버는 클라이언트의 공개 키와 서버의 비밀 키를 사용하여 공통 키를 추론합니다. 공통 키는 클라이언트에서 계산된 동일한 공통 키입니다. 계산을 위해서는 비밀 키 중 하나를 알아야 하므로 서버와 클라이언트만 공통 키를 계산할 수 있습니다.
3. 커널은 공통 키를 사용하여 컨버세이션 키를 해독합니다.
4. 커널은 키 서버를 호출하고 해독된 컨버세이션 키를 사용하여 클라이언트의 시간 기록을 해독합니다.

보안 RPC에서 서버에 정보 저장

서버에서 클라이언트의 시간 기록을 해독한 후 서버는 4가지 정보 항목을 자격 증명 테이블에 저장합니다.

- 클라이언트의 컴퓨터 이름
- 컨버세이션 키
- 창
- 클라이언트의 시간 기록

서버는 처음 3가지 항목을 나중에 사용을 위해 저장합니다. 서버는 재생을 막기 위해 클라이언트의 시간 기록을 저장합니다. 서버는 마지막 시간 기록보다 시간상으로 이후의 시간 기록만 수락합니다. 결과적으로 재생된 트랜잭션은 반드시 거부됩니다.

주 – 이러한 인증에서는 호출자의 이름이 어떠한 방식으로든 암시적으로 인증되어야 합니다. 키 서버에서 DES를 사용하면 교착 상태가 발생하므로 키 서버는 DES 인증을 사용하여 호출자를 인증할 수 없습니다. 교착 상태를 피하기 위해 키 서버는 사용자 ID(UID)별로 비밀 키를 저장하고 로컬 `root` 프로세스에만 요청을 허용합니다.

보안 RPC에서 클라이언트에 검증기 반환

서버는 클라이언트에 검증기를 반환하며, 여기에는 다음이 포함됩니다.

- 서버가 자격 증명 캐시에 기록하는 인덱스 ID
- 컨버세이션 키로 암호화된 클라이언트의 시간 기록 빼기 1

클라이언트의 시간 기록에서 1을 빼는 이유는 시간 기록이 오래되었음을 확인하기 위함입니다. 오래된 시간 기록은 클라이언트 검증기로 재사용할 수 없습니다.

보안 RPC에서 서버 인증

클라이언트는 검증기를 받고 서버를 인증합니다. 클라이언트는 서버만 클라이언트가 보낸 시간 기록을 알고 있으므로 서버만 검증기를 보낼 수 있다는 사실을 알 수 있습니다.

보안 RPC에서 트랜잭션 처리

첫번째 트랜잭션 이후 모든 트랜잭션에서 클라이언트는 인덱스 ID를 서버에 반환합니다. 또한 클라이언트는 다른 암호화된 시간 기록을 보냅니다. 서버는 컨버세이션 키로 암호화된 클라이언트의 시간 기록 빼기 1을 돌려 보냅니다.

보안 RPC에서 인증 관리(작업)

마운트된 NFS 파일 시스템에서 사용을 위해 인증을 요구하여 네트워크의 보안을 높입니다.

보안 RPC 관리(작업 맵)

다음 작업 맵에서는 NIS 및 NFS에 대한 보안 RPC를 구성하는 절차를 안내합니다.

작업	설명	수행 방법
1. 키 서버를 시작합니다.	키를 생성하여 사용자를 인증할 수 있도록 합니다.	272 페이지 "보안 RPC 키 서버를 다시 시작하는 방법"
2. NIS 호스트에서 자격 증명을 설정합니다.	호스트의 root 사용자를 NIS 환경에서 인증할 수 있도록 합니다.	272 페이지 "NIS 호스트에 대한 Diffie-Hellman 키를 설정하는 방법"
3. NIS 사용자에게 키를 제공합니다.	사용자를 NIS 환경에서 인증할 수 있도록 합니다.	274 페이지 "NIS 사용자에 대한 Diffie-Hellman 키를 설정하는 방법"
4. 인증을 사용하여 NFS 파일을 공유합니다.	NFS 서버가 인증을 사용하여 공유 파일 시스템을 안전하게 보호할 수 있도록 합니다.	275 페이지 "Diffie-Hellman 인증을 사용하여 NFS 파일을 공유하는 방법"

▼ 보안 RPC 키 서버를 다시 시작하는 방법

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 keyserv 데몬이 실행 중인지 확인합니다.

```
# svcs \*keyserv\*
STATE    STIME    FMRI
disabled Dec_14   svc:/network/rpc/keyserv
```

2 키 서버 서비스가 온라인이 아닌 경우 서비스를 사용으로 설정합니다.

```
# svcadm enable network/rpc/keyserv
```

▼ NIS 호스트에 대한 Diffie-Hellman 키를 설정하는 방법

이 절차는 NIS 도메인의 모든 호스트에서 수행해야 합니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 기본 이름 지정 서비스가 NIS가 아닌 경우 publickey 맵을 이름 지정 서비스에 추가합니다.

a. 이름 지정 서비스에 대한 config/default의 값이 nis가 아닌지 확인합니다.

```
# svccfg -s name-service/switch listprop config
config                      application
config/value_authorization  astring       solaris.smf.value.name-service.switch
config/default              astring       files
config/host                 astring       "files nis dns"
config/printer              astring       "user files nis"
```

config/default의 값이 nis인 경우 여기에서 중지할 수 있습니다.

b. publickey에 대한 이름 지정 서비스를 nis로 설정합니다.

```
# svccfg
# svccfg -s name-service/switch setprop config/publickey = astring: "nis"
# svccfg -s name-service/switch:default refresh
```

c. publickey 값을 확인합니다.

```
# svccfg
# svccfg -s name-service/switch listprop
config                      application
config/value_authorization  astring       solaris.smf.value.name-service.switch
config/default              astring       files
config/host                 astring       "files nis dns"
config/printer              astring       "user files nis"
config/publickey            astring       nis
```

이 시스템에서는 기본값인 files와 다르므로 publickey의 값이 나열됩니다.

2 newkey 명령을 사용하여 새 키 쌍을 만듭니다.

```
# newkey -h hostname
```

여기서 *hostname*은 클라이언트의 이름입니다.

예 14–1 NIS 클라이언트에서 root에 대한 새 키 설정

다음 예에서는 earth가 보안 NIS 클라이언트로 설정됩니다. 관리자에게는 Name Service Security 권한 프로파일이 지정되었습니다.

```
# newkey -h earth
Adding new key for unix.earth@example.com
New Password:       <Type password>
Retype password:    <Retype password>
Please wait for the database to get updated...
Your new key has been successfully stored away.
#
```

▼ NIS 사용자에 대한 Diffie-Hellman 키를 설정하는 방법

이 절차는 NIS 도메인의 모든 사용자에 대해 수행해야 합니다.

시작하기 전에 시스템 관리자만 NIS 마스터 서버에 로그인되었을 때 사용자에 대한 새 키를 생성할 수 있습니다. 관리자에게는 Name Service Security 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2. **사용자에 대한 새 키를 만듭니다.**

   ```
   # newkey -u username
   ```

   여기서 *username*은 사용자의 이름입니다. 시스템에서 암호를 물어봅니다. 일반 암호를 입력할 수 있습니다. 개인 키는 일반 암호를 사용하여 암호화된 형태로 저장됩니다.

3. **사용자에게 로그인하고 chkey -p 명령을 입력하도록 합니다.**
   이 명령을 통해 사용자는 사용자만 알 수 있는 암호로 개인 키를 다시 암호화할 수 있습니다.

주 – chkey 명령은 사용자에 대한 새 키 쌍을 만드는 데 사용할 수 있습니다.

예 14–2 NIS에서 새 사용자 키 설정 및 암호화

이 예에서는 수퍼유저가 키를 설정합니다.

```
# newkey -u jdoe
Adding new key for unix.12345@example.com
New Password:       <Type password>
Retype password:    <Retype password>
Please wait for the database to get updated...
Your new key has been successfully stored away.
#
```

그런 다음 사용자 jdoe는 개인 암호를 사용하여 키를 다시 암호화합니다.

```
% chkey -p
Updating nis publickey database.
Reencrypting key for unix.12345@example.com
Please enter the Secure-RPC password for jdoe:    <Type password>
Please enter the login password for jdoe:         <Type password>
Sending key change request to centralexample...
```

▼ Diffie-Hellman 인증을 사용하여 NFS 파일을 공유하는 방법

이 절차는 액세스에 대해 인증을 요구하여 NFS 서버에서 공유 파일 시스템을 보호합니다.

시작하기 전에 Diffie-Hellman 공개 키 인증이 네트워크에서 사용으로 설정되어야 합니다. 네트워크에서 인증을 사용으로 설정하려면 272 페이지 "NIS 호스트에 대한 Diffie-Hellman 키를 설정하는 방법"을 완료하십시오.

이 작업을 수행하려면 System Management 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2. **NFS 서버에서 Diffie-Hellman 인증을 사용하여 파일 시스템을 공유합니다.**

   ```
   # share -F nfs -o sec=dh /filesystem
   ```

   여기서 *filesystem*은 공유할 파일 시스템입니다.

   `-o sec=dh` 옵션은 파일 시스템에 액세스하려면 이제 `AUTH_DH` 인증이 필요하다는 것을 의미합니다.

3. **NFS 클라이언트에서 Diffie-Hellman 인증을 사용하여 파일 시스템을 마운트합니다.**

   ```
   # mount -F nfs -o sec=dh server:filesystem mount-point
   ```

   server — *filesystem*을 공유하는 시스템의 이름입니다.

   filesystem — 공유할 파일 시스템의 이름입니다(예: `opt`).

   mount-point — 마운트 지점의 이름입니다(예: `/opt`).

   `-o sec=dh` 옵션은 `AUTH_DH` 인증을 사용하여 파일 시스템을 마운트합니다.

◆ ◆ ◆ 15 장

15 PAM 사용

이 장에서는 PAM(플러그 가능한 인증 모듈) 프레임워크를 다룹니다. PAM은 인증 서비스를 Oracle Solaris OS에 "플러그인"할 수 있는 방법을 제공합니다. PAM은 시스템에 액세스할 때 여러 인증 서비스에 대한 지원을 제공합니다.

PAM(개요)

PAM(플러그 가능한 인증 모듈) 프레임워크를 통해 시스템 항목 서비스(예: `login`, `ftp`, `telnet`)를 변경하지 않고도 새로운 인증 서비스를 "플러그인"할 수 있습니다. 또한 PAM을 사용하여 Kerberos 등의 다른 보안 방식과 UNIX 로그인을 통합할 수 있습니다. 계정, 자격 증명, 세션 및 암호 관리를 위한 방식도 이 프레임워크를 사용하여 "플러그인"할 수 있습니다.

PAM 사용 이점

PAM 프레임워크를 사용하면 사용자 인증을 위한 시스템 항목 서비스(예: `ftp`, `login`, `telnet`, `rsh`) 사용을 구성할 수 있습니다. PAM이 제공하는 몇 가지 이점은 다음과 같습니다.

- 유연한 구성 정책
  - 응용 프로그램별 인증 정책
  - 기본 인증 방식을 선택할 수 있는 기능
  - 높은 보안 시스템에서 여러 권한 부여를 요구할 수 있는 기능
- 최종 사용자의 사용 편의성
  - 암호가 여러 인증 서비스에 대해 동일한 경우 암호 재입력 없음

  - 사용자가 여러 명령을 입력할 필요 없이 여러 인증 서비스에 대해 사용자에게 암호를 요구할 수 있는 기능
- 사용자 인증 서비스에 선택적 옵션을 전달할 수 있는 기능
- 시스템 항목 서비스를 변경할 필요 없이 사이트별 보안 정책을 구현할 수 있는 기능

PAM 프레임워크 소개

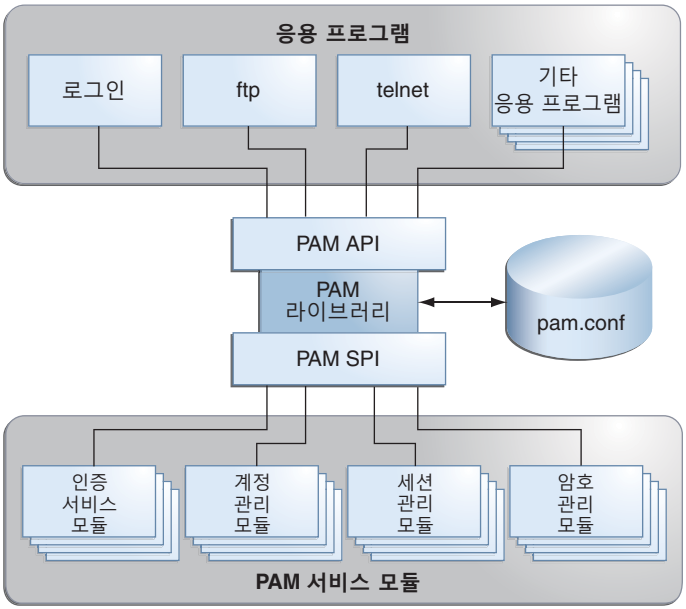
PAM 프레임워크는 네 부분으로 구성됩니다.

- PAM 소비자
- PAM 라이브러리
- pam.conf(4) 구성 파일
- PAM 서비스 모듈(공급자라도도 함)

프레임워크는 인증 관련 작업이 발생하도록 일관성 있는 방식을 제공합니다. 이 방식을 통해 응용 프로그램 개발자는 정책의 의미를 알 필요 없이 PAM 서비스를 사용할 수 있습니다. 알고리즘은 중앙에서 제공됩니다. 알고리즘은 개별 응용 프로그램과 독립적으로 수정할 수 있습니다. PAM을 사용하면 관리자는 응용 프로그램을 변경하지 않고도 특정 시스템의 요구에 인증 프로세스를 맞출 수 있습니다. 조정은 PAM 구성 파일인 pam.conf를 통해 수행됩니다.

다음 그림은 PAM 아키텍처를 나타냅니다. 응용 프로그램은 PAM API(application programming interface)를 통해 PAM 라이브러리와 통신합니다. PAM 모듈은 PAM SPI(서비스 공급자 인터페이스)를 통해 PAM 라이브러리와 통신합니다. 따라서 PAM 라이브러리를 통해 응용 프로그램과 모듈이 서로 통신할 수 있습니다.

그림 15–1 PAM 아키텍처



이 릴리스에 대한 PAM 변경 사항

Oracle Solaris 11 Express 릴리스용 PAM 프레임워크에는 새 `pam_allow` 모듈이 포함됩니다. 모듈을 사용하여 보안을 적용하지 않고 모든 사용자에게 액세스 권한을 부여할 수 있습니다. 모듈은 주의해서 사용해야 합니다. 자세한 내용은 `pam_allow(5)` 매뉴얼 페이지를 참조하십시오.

PAM(작업)

이 절에서는 PAM 프레임워크에서 특정 보안 정책을 사용하도록 하기 위해 필요할 수 있는 몇 가지 작업을 설명합니다. PAM 구성 파일과 관련된 몇 가지 보안 문제에 대해 알고 있어야 합니다. 보안 문제에 대한 자세한 내용은 280 페이지 “PAM 구현 계획”을 참조하십시오.

PAM(작업 맵)

작업	설명	수행 방법
PAM 설치에 대해 계획합니다.	구성 문제를 고려하고 소프트웨어 구성 프로세스를 시작하기 전에 구성에 대한 결정을 내립니다.	280 페이지 “PAM 구현 계획”
새 PAM 모듈을 추가합니다.	일반 소프트에 속하지 않은 요구 사항을 충족하려면 사이트별 모듈을 작성하고 설치해야 할 수 있습니다. 이 절차에서는 이러한 새 PAM 모듈을 설치하는 방법을 설명합니다.	281 페이지 “PAM 모듈을 추가하는 방법”
`~/.rhosts`를 통해 액세스를 차단합니다.	`~/.rhosts`를 통해 액세스를 막음으로써 보안을 높입니다.	281 페이지 “PAM을 사용하여 원격 시스템에서 Rhost 스타일 액세스를 막는 방법”
오류 로깅을 시작합니다.	`syslog`를 통해 PAM 오류 메시지 로깅을 시작합니다.	282 페이지 “PAM 오류 보고서를 로깅하는 방법”

PAM 구현 계획

기본적으로 제공되는 `pam.conf` 구성 파일은 표준 보안 정책을 구성합니다. 이 정책은 많은 상황에서 작동해야 합니다. 서로 다른 보안 정책을 구현해야 하는 경우 초점을 맞추어야 하는 문제는 다음과 같습니다.

- 무엇이 필요한지, 특히 어떤 PAM 서비스 모듈을 선택해야 하는지 결정합니다.
- 특수한 구성 옵션이 필요한 서비스를 파악합니다. 해당하는 경우 `other`를 사용합니다.
- 모듈이 실행되어야 하는 순서를 결정합니다.
- 각 모듈에 대한 제어 플래그를 선택합니다. 모든 제어 플래그에 대한 자세한 내용은 283 페이지 “PAM 스택이 작동하는 방식”을 참조하십시오.
- 각 모듈에 필요한 옵션을 선택합니다. 각 모듈에 대한 매뉴얼 페이지에는 특수 옵션이 나열되어 있어야 합니다.

다음은 PAM 구성 파일을 변경하기 전에 고려해야 할 몇 가지 제안 사항입니다.

- 각 모듈 유형에 대해 `other` 항목을 사용하여 모든 응용 프로그램이 `/etc/pam.conf`에 포함될 필요가 없도록 합니다.
- `binding`, `sufficient` 및 `optional` 제어 플래그의 보안 구현을 고려합니다.
- 모듈과 연결된 매뉴얼 페이지를 검토합니다. 이러한 매뉴얼 페이지는 각 모듈의 작동 방식, 사용 가능한 옵션 및 스택 모듈 사이의 상호 작용을 이해하는 데 도움을 줄 수 있습니다.

주의 – PAM 구성 파일이 잘못 구성되거나 파일이 손상될 경우 사용자가 로그인하지 못하게 될 수 있습니다. `sulogin` 명령은 PAM을 사용하지 않으므로 시스템을 단일 사용자 모드로 부트하고 문제를 수정하려면 root 암호가 필요할 수 있습니다.

`/etc/pam.conf` 파일을 변경한 후 시스템 액세스 권한이 있을 때 가능한 많이 파일을 검토하여 문제를 해결하십시오. 변경 사항으로 영향을 받을 수 있는 모든 명령을 테스트합니다. 예는 새 모듈을 `telnet` 서비스에 추가하는 것입니다. 이 예에서는 `telnet` 명령을 사용하고 변경 사항으로 서비스가 예상대로 작동하는지 확인합니다.

▼ PAM 모듈을 추가하는 방법

이 절차에서는 새 PAM 모듈을 추가하는 방법을 보여줍니다. 사이트별 보안 정책을 포함하거나 타사 응용 프로그램을 지원하기 위해 새 모듈을 만들 수 있습니다.

1. **관리자로 전환합니다.**
   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2. **사용해야 하는 제어 플래그 및 기타 옵션을 결정합니다.**
   제어 플래그에 대한 자세한 내용은 283 페이지 “PAM 스택이 작동하는 방식”을 참조하십시오.

3. **모듈 파일이 root의 소유이고 권한이 555가 되도록 소유권 및 권한이 설정되었는지 확인합니다.**

4. **PAM 구성 파일 `/etc/pam.conf`를 편집하고 이 모듈을 해당 서비스에 추가합니다.**

5. **모듈이 제대로 추가되었는지 확인합니다.**
   구성 파일이 잘못 구성된 경우를 대비하여 시스템이 재부트되기 **전에** 테스트해야 합니다. 시스템을 재부트하기 전에 `ssh`와 같은 직접 서비스를 사용하여 로그인하고 `su` 명령을 실행합니다. 서비스는 시스템이 부트될 때 한 번만 생성되는 데몬일 수 있습니다. 그런 경우 모듈이 추가되었는지 확인하려면 시스템을 재부트해야 합니다.

▼ PAM을 사용하여 원격 시스템에서 Rhost 스타일 액세스를 막는 방법

1. **관리자로 전환합니다.**
   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **PAM 구성 파일에서 rhosts_auth.so.1이 포함된 모든 행을 제거합니다.**

이 단계는 rlogin 세션 중 ~/.rhosts 파일이 읽혀지지 않도록 합니다. 따라서 이 단계는 원격 시스템에서 로컬 시스템에 대한 인증되지 않은 액세스를 막습니다. ~/.rhosts 또는 /etc/hosts.equiv 파일의 존재 여부나 내용에 상관없이 모든 rlogin 액세스에는 암호가 필요합니다.

3 **rsh 서비스를 사용 안함으로 설정합니다.**

~/.rhosts 파일에 대한 다른 인증되지 않은 액세스를 막으려면 rsh 서비스를 사용 안함으로 설정합니다.

```
# svcadm disable network/shell
```

▼ PAM 오류 보고서를 로깅하는 방법

1 **관리자로 전환합니다.**

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **필요한 로깅 레벨에 대해 /etc/syslog.conf 파일을 구성합니다.**

로깅 레벨에 대한 자세한 내용은 syslog.conf(4)를 참조하십시오.

3 **syslog 데몬에 대한 구성 정보를 새로 고칩니다.**

```
# svcadm refresh system/system-log
```

PAM 구성(참조)

PAM 구성 파일 pam.conf(4)는 login, rlogin, su 및 cron과 같은 시스템 서비스에 대해 PAM 서비스 모듈을 구성하는 데 사용됩니다. 시스템 관리자가 이 파일을 관리합니다. pam.conf의 항목 순서가 잘못되면 예측하지 못한 역효과가 발생할 수 있습니다. 예를 들어, 잘못 구성된 pam.conf는 사용자를 잠그게 되므로 복구하려면 단일 사용자 모드가 필요할 수 있습니다. 순서 설정에 대한 설명은 283 페이지 "PAM 스택이 작동하는 방식"을 참조하십시오.

PAM 구성 파일 구문

구성 파일 항목의 형식은 다음과 같습니다.

service-name module-type control-flag module-path module-options

service-name 서비스의 이름(예: ftp, login, passwd)입니다. 응용 프로그램에서는 응용 프로그램이 제공하는 서비스에 대해 서로 다른 서비스 이름을 사용할 수 있습니다. 예를 들어, Oracle Solaris 보안 셸 데몬에서는

sshd-none, sshd-password, sshd-kbdint, sshd-pubkey 및 sshd-hostbased의 서비스 이름을 사용합니다. service-name *other*는 와일드카드 service-name으로 사용되는 사전 정의된 이름입니다. 특정 service-name을 구성 파일에서 찾을 수 없는 경우 *other*에 대한 구성 파일이 사용됩니다.

module-type 서비스의 유형(즉, auth, account, session 또는 password)입니다.

control-flag 서비스에 대한 통합된 성공 또는 실패 값을 결정하는 모듈의 역할을 나타냅니다. 유효한 제어 플래그는 binding, include, optional, required, requisite 및 sufficient입니다. 이러한 플래그 사용에 대한 자세한 내용은 283 페이지 "PAM 스택이 작동하는 방식"을 참조하십시오.

module-path 서비스를 구현하는 라이브러리 객체의 경로입니다. 경로 이름이 절대 경로가 아닌 경우 경로 이름은 /usr/lib/security/$ISA/에 대한 상대 경로로 간주됩니다. libpam이 디렉토리에서 응용 프로그램의 특정 아키텍처를 찾도록 하려면 아키텍처 종속 매크로 $ISA를 사용합니다.

module-options 서비스 모듈에 전달되는 옵션입니다. 모듈의 매뉴얼 페이지에서는 해당 모듈에서 허용되는 옵션을 설명합니다. 일반적인 모듈 옵션에는 nowarn 및 debug가 포함됩니다.

PAM 스택이 작동하는 방식

응용 프로그램이 다음 함수를 호출할 경우 libpam은 구성 파일 /etc/pam.conf를 읽어 어떤 모듈이 이 서비스에 대한 작업에 참여하는지 결정합니다.

- pam_authenticate(3PAM)
- pam_acct_mgmt(3PAM)
- pam_setcred(3PAM)
- pam_open_session(3PAM)
- pam_close_session(3PAM)
- pam_chauthtok(3PAM)

/etc/pam.conf에 이 서비스의 작업에 대한 모듈이 하나(예: 인증 또는 계정 관리)만 포함된 경우 해당 모듈의 결과가 작업의 결과를 결정합니다. 예를 들어, passwd 응용 프로그램에 대한 기본 인증 작업에는 하나의 모듈 pam_passwd_auth.so.1이 포함됩니다.

```
passwd  auth required           pam_passwd_auth.so.1
```

반면 서비스의 작업에 대해 정의된 모듈이 여러 개인 경우 해당 모듈을 **스택**이라 하고 *PAM* **스택**이 해당 서비스에 대해 존재합니다. 예를 들어, pam.conf에 다음 항목이 포함된 경우를 생각해 보십시오.

```
login   auth requisite          pam_authtok_get.so.1
login   auth required           pam_dhkeys.so.1
login   auth required           pam_unix_cred.so.1
login   auth required           pam_unix_auth.so.1
login   auth required           pam_dial_auth.so.1
```

이러한 항목은 `login` 서비스에 대한 샘플 `auth` 스택을 나타냅니다. 이 스택의 결과를 결정하려면 개별 모듈의 결과 코드에 **통합 프로세스**가 필요합니다. 통합 프로세스에서 모듈은 `/etc/pam.conf` 에 지정된 대로 순서대로 실행됩니다. 각 성공 또는 실패 코드는 모듈의 제어 플래그에 따라 전체 결과에 통합됩니다. 제어 플래그는 스택의 조기 종료를 유발할 수 있습니다. 예를 들어, `requisite` 모듈이 실패하거나 `sufficient` 또는 `binding` 모듈이 성공할 수 있습니다. 스택이 처리된 후 개별 결과는 하나의 전체 결과로 합쳐서 응용 프로그램에 전달됩니다.

제어 플래그는 PAM 모듈이 서비스에 대한 액세스를 결정하는 데 수행하는 역할을 나타냅니다. 제어 플래그 및 효과는 다음과 같습니다.

- **Binding** – 실패한 이전 required 모듈이 없는 경우 binding 모듈의 요구 사항을 충족하는 성공은 이 응용 프로그램에 즉시 성공을 반환합니다. 이러한 조건이 충족되지 않는다면 모듈은 더 이상 실행되지 않습니다. 실패의 경우 required 실패가 기록되고 모듈 처리가 계속됩니다.
- **Include** – PAM 스택의 이 시점에서 사용될 별도의 PAM 구성 파일에서 행을 추가합니다. 이 플래그는 성공이나 실패 동작을 제어하지 않습니다. 새 파일이 읽혀지면 PAM 포함 스택이 증가됩니다. 새 파일에서 스택 확인이 완료되면 포함 스택 값이 감소합니다. 파일의 끝에 도달하고 PAM 포함 스택이 0이면 스택 처리가 종료됩니다. PAM 포함 스택에 대한 최대 수는 32입니다.
- **Optional** – optional 모듈의 요구 사항을 충족하는 성공은 서비스를 사용하는 데 필요하지 않습니다. 실패의 경우 optional 실패가 기록됩니다.
- **Required** – reauired 모듈의 요구 사항을 충족하는 성공은 서비스를 사용하는 데 필요합니다. 실패의 경우 이 서비스에 대한 나머지 모듈이 실행된 후 오류가 반환됩니다. 서비스에 대한 최종 성공은 실패를 보고한 binding 또는 required 모듈이 없을 경우에만 반환됩니다.
- **Requisite** – requisite 모듈의 요구 사항을 충족하는 성공은 서비스를 사용하는 데 필요합니다. 실패의 경우 모듈의 추가 실행 없이 즉시 오류가 반환됩니다. 함수가 응용 프로그램에 성공을 반환할 수 있으려면 서비스에 대한 모든 requisite 모듈이 성공을 반환해야 합니다.
- **Sufficient** – 발생한 이전 required 실패가 없는 경우 sufficient 모듈의 성공은 모듈의 추가 실행 없이 응용 프로그램에 즉시 성공을 반환합니다. 실패의 경우 optional 실패가 기록됩니다.

다음 두 다이어그램은 통합 프로세스에서 액세스가 어떻게 결정되는지 보여줍니다. 첫번째 다이어그램은 제어 플래그의 각 유형에 대해 성공 또는 실패가 어떻게 기록되는지 나타냅니다. 두번째 다이어그램은 통합된 값이 어떻게 결정되는지 보여줍니다.

그림 15–2 PAM 스택: 제어 플래그의 효과

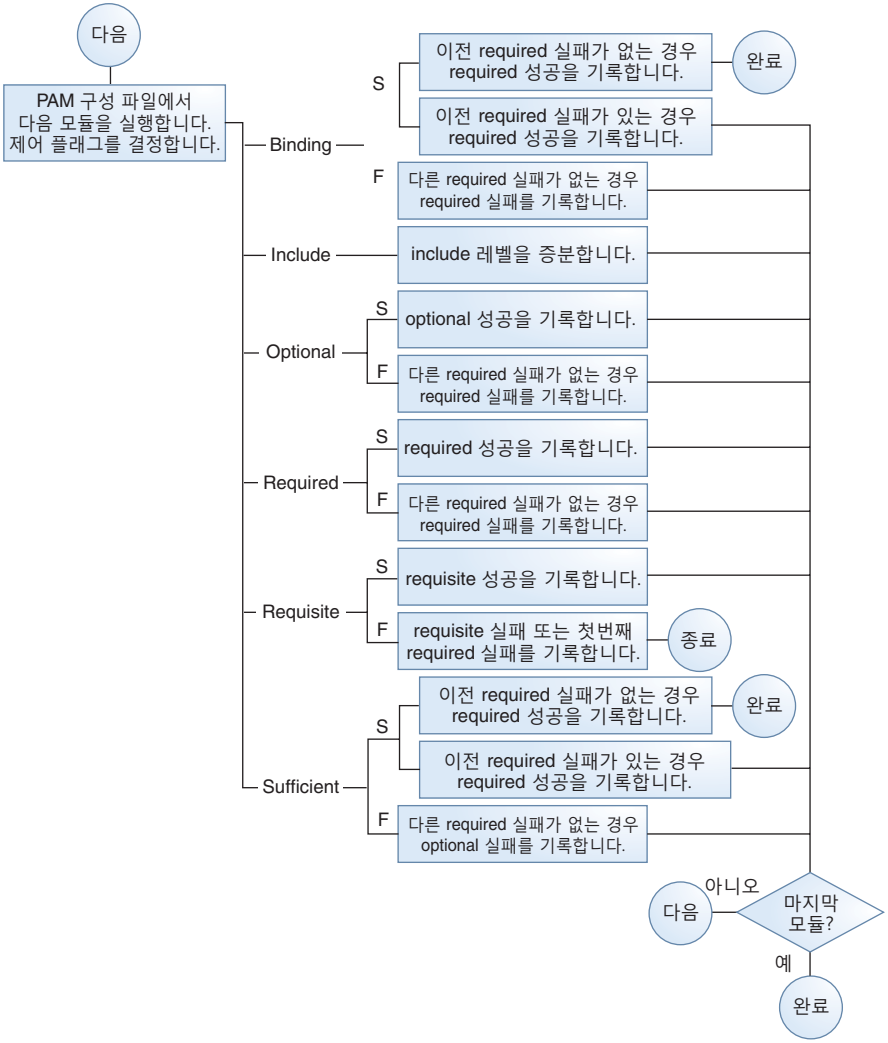
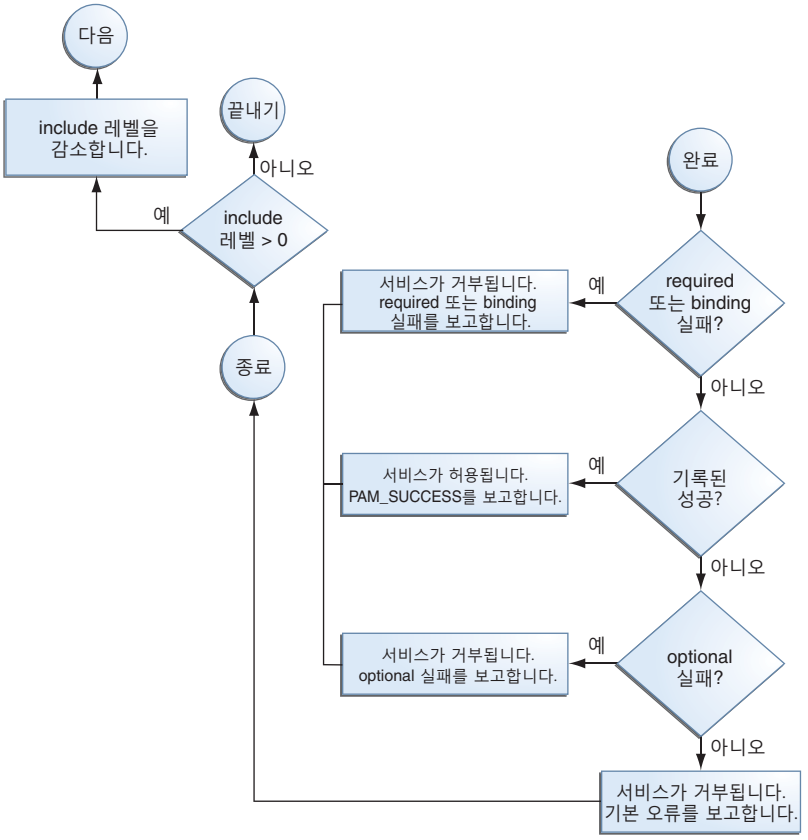


그림 15–3 PAM 스택: 통합된 값이 결정되는 방식



PAM 스택 예

인증을 요청하는 rlogin 서비스의 다음 예를 살펴보십시오.

예 15–1 일반적인 PAM 구성 파일의 일부

이 예의 pam.conf 파일에는 rlogin 서비스에 대한 다음 내용이 있습니다.

```
# Authentication management
...
# rlogin service
rlogin  auth sufficient         pam_rhosts_auth.so.1
rlogin  auth requisite          pam_authtok_get.so.1
rlogin  auth required           pam_dhkeys.so.1
rlogin  auth required           pam_unix_auth.so.1
...
```

예 15–1 일반적인 PAM 구성 파일의 일부 *(계속)*

rlogin 서비스에서 인증을 요청하면 libpam이 먼저 pam_rhosts_auth(5) 모듈을 실행합니다. 제어 플래그는 pam_rhosts_auth 모듈에 대해 sufficient로 설정됩니다. pam_rhosts_auth 모듈에서 사용자를 인증할 수 있는 경우 처리가 중지되고 응용 프로그램에 성공이 반환됩니다.

pam_rhosts_auth 모듈에서 사용자 인증을 실패할 경우 다음 PAM 모듈인 pam_authtok_get(5)이 실행됩니다. 이 모듈에 대한 제어 플래그는 requisite로 설정됩니다. pam_authtok_get를 실패할 경우 인증 프로세스가 종료되고 rlogin에 실패가 반환됩니다.

pam_authtok_get을 성공할 경우 다음 두 모듈인 pam_dhkeys(5) 및 pam_unix_auth(5)가 실행됩니다. 두 모듈은 required로 설정된 제어 플래그를 가지므로 개별적으로 실패 반환 여부에 상관없이 프로세스가 계속됩니다. pam_unix_auth가 실행된 후 남아 있는 rlogin 인증에 대한 모듈은 없습니다. 이 시점에서 pam_dhkeys 또는 pam_unix_auth가 실패를 반환할 경우 사용자는 rlogin을 통해 액세스가 거부됩니다.

◆ ◆ ◆ 16 장

16 SASL 사용

이 장에서는 SASL(Simple Authentication and Security Layer)에 대한 정보를 제공합니다.

SASL(개요)

SASL(Simple Authentication and Security Layer)은 네트워크 프로토콜에 인증 및 선택적 보안 서비스를 제공하는 프레임워크입니다. 응용 프로그램이 SASL 라이브러리 `/usr/lib/libsasl.so`를 호출하여 응용 프로그램과 다양한 SASL 방식 사이에 접착층을 제공합니다. 방식은 인증 프로세스에 사용되며 선택적 보안 서비스를 제공합니다. SASL의 버전은 몇몇 변경 사항과 함께 Cyrus SASL에서 파생됩니다.

SASL은 다음 서비스를 제공합니다.

- 플러그인의 로드
- 보안 방식의 선택을 돕기 위해 응용 프로그램에서 필요한 보안 옵션 확인
- 응용 프로그램에 사용 가능한 플러그인 나열
- 특정 인증 시도에 대해 사용 가능한 방식 목록에서 최상의 방식 선택
- 응용 프로그램과 선택한 방식 사이에 인증 데이터 경로 지정
- SASL 협상 정보를 응용 프로그램에 다시 제공

SASL(참조)

다음 절은 SASL 구현에 대한 정보를 제공합니다.

SASL 플러그인

SASL 플러그인은 보안 방식, 사용자 정규화, 보조 등록 정보 검색을 지원합니다. 기본적으로, 동적으로 로드된 32비트 플러그인이 /usr/lib/sasl에 설치되고, 64비트 플러그인이 /usr/lib/sasl/ *$ISA*에 설치됩니다. 다음 보안 방식 플러그인이 제공됩니다.

crammd5.so.1	CRAM-MD5로, 권한 부여 없이 인증만 지원합니다.
digestmd5.so.1	DIGEST-MD5로, 권한 부여는 물론 인증, 무결성, 프라이버시를 지원합니다.
gssapi.so.1	GSSAPI로, 권한 부여는 물론 인증, 무결성, 프라이버시를 지원합니다. GSSAPI 보안 방식에는 작동 중인 Kerberos 기반구조가 필요합니다.
plain.so.1	PLAIN으로, 인증 및 권한 부여를 지원합니다.

더불어, EXTERNAL 보안 방식 플러그인과 INTERNAL 사용자 정규화 플러그인이 libsasl.so.1로 내장됩니다. EXTERNAL 방식은 인증 및 권한 부여를 지원합니다. 외부 보안 소스가 제공하는 경우 무결성 및 프라이버시를 지원합니다. INTERNAL 플러그인은 필요한 경우 영역 이름을 사용자 이름에 추가합니다.

Oracle Solaris 릴리스는 이 시점에 auxprop 플러그인을 제공하지 않습니다. CRAM-MD5 및 DIGEST-MD5 방식 플러그인이 서버측에서 완전히 작동하려면 사용자가 일반 텍스트 암호를 검색하는 auxprop 플러그인을 제공해야 합니다. PLAIN 플러그인은 추가로 암호 확인 지원이 필요합니다. 암호 확인 지원은 서버 응용 프로그램에 콜백, auxprop 플러그인, saslauthd, pwcheck 중 하나일 수 있습니다. salauthd 및 pwcheck 데몬은 Oracle Solaris 릴리스에 제공되지 않습니다. 향상된 상호 운용성을 위해 mech_list SASL 옵션을 사용하여 완전히 작동하는 방식으로 서버 응용 프로그램을 제한할 수 있습니다.

SASL 환경 변수

기본적으로 클라이언트 인증 이름은 getenv("LOGNAME")으로 설정됩니다. 이 변수는 클라이언트나 플러그인에서 재설정할 수 있습니다.

SASL 옵션

/etc/sasl/*app*.conf 파일에 설정할 수 있는 옵션을 사용하여 libsasl 및 플러그인의 동작을 서버측에서 수정할 수 있습니다. *app* 변수는 서버에서 정의된 응용 프로그램의 이름입니다. 서버 *app*에 대한 문서는 응용 프로그램 이름을 지정해야 합니다.

다음 옵션이 지원됩니다.

auto_transition	일반 텍스트 인증을 성공할 때 사용자를 다른 방식으로 자동으로 이행합니다.

`auxprop_login` 사용할 보조 등록 정보 플러그인의 이름을 나열합니다.

`canon_user_plugin` 사용할 canon_user 플러그인을 선택합니다.

`mech_list` 서버 응용 프로그램에서 사용하도록 허용된 방식을 나열합니다.

`pwcheck_method` 암호 확인에 사용되는 방식을 나열합니다. 현재 `auxprop`가 유일하게 허용된 값입니다.

`reauth_timeout` 빠른 재인증을 위해 인증 정보가 캐싱되는 시간 길이(분)를 설정합니다. 이 옵션은 DIGEST-MD5 플러그인에서 사용됩니다. 이 옵션을 0으로 설정하면 재인증이 사용되지 않습니다.

다음 옵션은 지원되지 않습니다.

`plugin_list` 사용 가능한 방식을 나열합니다. 옵션은 플러그인의 동적 로드 동작을 바꾸기 때문에 사용되지 않습니다.

`saslauthd_path` `saslauthd` 데몬과 통신에 사용되는 saslauthd 도어의 위치를 정의합니다. `saslauthd` 데몬은 Oracle Solaris 릴리스에 포함되지 않습니다. 따라서 이 옵션도 포함되지 않습니다.

`keytab` GSSAPI 플러그인에서 사용되는 `keytab` 파일의 위치를 정의합니다. 기본 `keytab` 위치를 설정하는 대신 `KRB5_KTNAME` 환경 변수를 사용하십시오.

다음 옵션은 Cyrus SASL에 없는 옵션입니다. 그러나 Oracle Solaris 릴리스에 추가되었습니다.

`use_authid` GSS 클라이언트 보안 컨텍스트를 만들 때 기본 자격 증명을 사용하기보다 클라이언트 자격 증명을 획득합니다. 기본적으로 기본 클라이언트 Kerberos 식별이 사용됩니다.

`log_level` 서버에 대한 원하는 로깅 레벨을 설정합니다.

◆ ◆ ◆ 17 장

17 Secure Shell 사용(작업)

Oracle Solaris의 Secure Shell 기능을 사용하면 비보안 네트워크를 통해 원격 호스트에 안전하게 액세스할 수 있습니다. 셸은 원격 로그인 및 원격 파일 전송 명령을 제공합니다. 다음은 이 장에서 다루는 항목을 나열한 것입니다.

참조 정보는 18 장, “Secure Shell(참조)”을 참조하십시오.

Secure Shell(개요)

Secure Shell에서 인증은 암호 또는 공개 키를 사용하거나 두 가지를 모두 사용하여 제공됩니다. 모든 네트워크 트래픽은 암호화됩니다. 따라서 Secure Shell은 침입자가 가로챈 통신을 읽지 못하도록 합니다. 또한 Secure Shell은 침입자의 시스템 속임수를 방지합니다.

Secure Shell을 필요 시 VPN(가상 사설망)으로 사용할 수도 있습니다. VPN은 X 윈도우 시스템 트래픽을 전달할 수도 있고, 암호화된 네트워크 링크를 통해 로컬 시스템과 원격 시스템 간에 개별 포트 번호를 연결할 수도 있습니다.

Secure Shell을 통해 다음 작업을 수행할 수 있습니다.

- 비보안 네트워크를 통해 안전하게 다른 호스트에 로그인합니다.
- 두 호스트 간에 안전하게 파일을 복사합니다.
- 원격 호스트에서 안전하게 명령을 실행합니다.

서버측에서 Secure Shell은 Secure Shell 프로토콜의 두 가지 버전, 버전 1(v1) 및 버전 2를 제공합니다. 버전 2(v2)가 더 안전합니다. Secure Shell은 v2로 마이그레이션하고 있는 사용자를 지원하는 용도로만 v1을 제공합니다. v1에 대한 자세한 내용은 **System Administration Guide: Security Services**를 참조하십시오.

Secure Shell 인증

Secure Shell은 원격 호스트에 대한 연결에 사용할 공개 키 및 암호 인증 방법을 제공합니다. 개인 키는 네트워크를 통과되지 않으므로 공개 키 인증이 암호 인증보다 강력한 인증 방식입니다.

인증 방법은 다음 순서로 시도됩니다. 구성이 인증 방법을 충족하지 않을 경우 다음 방법이 시도됩니다.

- **GSS-API** – `mech_krb5`(Kerberos V), `mech_dh`(AUTH_DH) 등의 GSS-API 방식에 자격 증명을 사용하여 클라이언트 및 서버를 인증합니다. GSS-API에 대한 자세한 내용은 **Developer's Guide to Oracle Solaris 11 Security**의 "Introduction to GSS-API"를 참조하십시오.
- **호스트 기반 인증** – 호스트 키 및 rhosts 파일을 사용합니다. 클라이언트의 RSA 및 DSA 공개/개인 호스트 키를 사용하여 클라이언트를 인증합니다. rhosts 파일을 사용하여 사용자에 대해 클라이언트에게 권한을 부여합니다.
- **공개 키 인증** – RSA 및 DSA 공개/개인 키로 사용자를 인증합니다.
- **암호 인증** – PAM을 사용하여 사용자를 인증합니다. v2의 키보드 인증 방법은 PAM의 임의적인 프롬프트를 허용합니다. 자세한 내용은 `sshd`(1M) 매뉴얼 페이지의 `SECURITY` 절을 참조하십시오.

다음 표에서는 원격 호스트에 로그인하려고 시도 중인 사용자를 인증하기 위한 요구 사항을 보여 줍니다. 사용자는 로컬 호스트인 클라이언트에 있습니다. 원격 호스트인 서버는 `sshd` 데몬을 실행 중입니다. 표에서는 Secure Shell 인증 방법, 호환되는 프로토콜 버전 및 호스트 요구 사항을 보여 줍니다.

표 17–1 Secure Shell 인증 방법

인증 방법	로컬 호스트(클라이언트) 요구 사항	원격 호스트(서버) 요구 사항
GSS-API	GSS 방식에 대한 개시자 자격 증명입니다.	GSS 방식에 대한 승인자 자격 증명입니다. 자세한 내용은 312 페이지 "Secure Shell에서 GSS 자격 증명 취득"을 참조하십시오.

호스트 기반	사용자 계정
	`/etc/ssh/ssh_host_rsa_key` 또는 `/etc/ssh/ssh_host_dsa_key`의 로컬 호스트 개인 키
	`/etc/ssh/ssh_config`의 `HostbasedAuthentication yes`
		사용자 계정
		`/etc/ssh/known_hosts` 또는 `~/.ssh/known_hosts`의 로컬 호스트 공개 키
		`/etc/ssh/sshd_config`의 `HostbasedAuthentication yes`
		`/etc/ssh/sshd_config`의 `IgnoreRhosts no`
		`/etc/ssh/shosts.equiv`, `/etc/hosts.equiv`, `~/.rhosts` 또는 `~/.shosts`의 로컬 호스트 항목

표 17–1 Secure Shell 인증 방법 (계속)

인증 방법	로컬 호스트(클라이언트) 요구 사항	원격 호스트(서버) 요구 사항
RSA 또는 DSA 공개 키	사용자 계정 ~/.ssh/id_rsa 또는 ~/.ssh/id_dsa의 개인 키 ~/.ssh/id_rsa.pub 또는 ~/.ssh/id_dsa.pub의 사용자 공개 키	사용자 계정 ~/.ssh/authorized_keys의 사용자 공개 키
암호 기반	사용자 계정	사용자 계정 PAM을 지원합니다.
서버에서만 RSA(v1)를 사용한 .rhosts	사용자 계정 /etc/ssh/ssh_host_rsa1_key의 로컬 호스트 공개 키	사용자 계정 /etc/ssh/ssh_known_hosts 또는 ~/.ssh/known_hosts의 로컬 호스트 공개 키 /etc/ssh/sshd_config의 IgnoreRhosts no /etc/ssh/shosts.equiv, /etc/hosts.equiv, ~/.shosts 또는 ~/.rhosts의 로컬 호스트 항목

기업의 Secure Shell

Oracle Solaris 시스템의 Secure Shell에 대한 포괄적인 설명은 2003년 6월 발행된 Jason Reid의 **Secure Shell in the Enterprise**(ISBN 0-13-142900-0)를 참조하십시오. 이 책은 Sun Microsystems Press에서 발행한 Sun BluePrints Series의 일부입니다.

Secure Shell 및 OpenSSH 프로젝트

Secure Shell은 OpenSSH (http://www.openssh.com) 프로젝트의 포크입니다. OpenSSH의 후속 버전에서 발견된 위험성에 대한 보안 수정 사항은 개별 버그 수정 및 기능으로 Secure Shell에 통합되었습니다. Secure Shell 포크에 대한 내부 개발은 계속 진행 중입니다.

다음은 이 릴리스의 Secure Shell에서 v2 프로토콜에 대해 구현된 기능입니다.

- ForceCommand 키워드 – 사용자가 명령줄에서 입력하는 내용에 관계없이 지정된 명령을 강제로 실행합니다. 이 키워드는 Match 블록에서 가장 유용합니다. sshd_config 구성 옵션은 $HOME/.ssh/authorized_keys의 command="..." 옵션과 유사합니다.
- AES-128 암호문 보호 – 이 릴리스에서 ssh-keygen 명령으로 생성된 개인 키는 AES-128 알고리즘으로 보호됩니다. 이 알고리즘은 암호문 변경 등으로 새로 생성된 키 및 다시 암호화된 키를 보호합니다.
- sftp-server 명령에 대한 -u 옵션 – 사용자가 파일 및 디렉토리에 대해 명시적 umask를 설정할 수 있도록 합니다. 이 옵션은 사용자의 기본 umask를 대체합니다. 예는 sshd_config(4) 매뉴얼 페이지의 Subsystem에 대한 설명을 참조하십시오.

- Match 블록에 대한 추가 키워드 – AuthorizedKeysFile, ForceCommand 및 HostbasedUsesNameFromPacketOnly가 Match 블록에서 지원됩니다. 기본적으로 AuthorizedKeysFile의 값은 $HOME/.ssh/authorized_keys이며 HostbasedUsesNameFromPacketOnly는 no입니다. Match 블록을 사용하려면 300 페이지 "SSH 시스템 기본값에 대한 사용자 및 호스트 예외를 만드는 방법"을 참조하십시오.

프로젝트에 대한 버그 수정을 제공하면서 Oracle Solaris 엔지니어는 Secure Shell의 포크에 다음 Oracle Solaris 기능도 통합했습니다.

- PAM - Secure Shell에 PAM이 사용됩니다. OpenSSH UsePAM 구성 옵션은 지원되지 않습니다.
- 권한 구분 - Secure Shell에 OpenSSH 프로젝트의 권한 구분 코드가 사용되지 않습니다. Secure Shell은 감사, 레코드 보관 및 키 갱신에 대한 처리와 세션 프로토콜에 대한 처리를 구분합니다.

  Secure Shell 권한 구분 코드는 항상 설정되어 있으며 해제할 수 없습니다. OpenSSH UsePrivilegeSeparation 옵션은 지원되지 않습니다.
- 로케일 - Secure Shell은 RFC 4253 **Secure Shell Transfer Protocol**에 정의된 언어 협상을 전체적으로 지원합니다. 사용자가 로그인한 후 사용자의 로그인 셸 프로파일이 Secure Shell의 협상된 로케일 설정을 대체할 수 있습니다.
- 감사 - Secure Shell은 Solaris 감사 서비스에 완전히 통합되었습니다. 감사 서비스에 대한 자세한 내용은 제7부를 참조하십시오.
- GSS-API 지원 - GSS-API는 사용자 인증 **및** 초기 키 교환에 사용할 수 있습니다. GSS-API는 RFC4462 **Generic Security Service Application Program Interface**에 정의되어 있습니다.
- 프록시 명령 - Secure Shell은 SOCKS5 및 HTTP 프로토콜에 대한 프록시 명령을 제공합니다. 예는 309 페이지 "방화벽 외부의 호스트에 대한 기본 연결 설정 방법"을 참조하십시오.

Oracle Solaris 릴리스에서 Secure Shell은 OpenSSH 프로젝트의 SSH_OLD_FORWARD_ADDR 호환성 플래그를 다시 동기화합니다. 2011년 3월 당시 Secure Shell 버전은 1.5였습니다.

Secure Shell 및 FIPS-140 지원

Secure Shell 작업에 Sun Crypto Accelerator 6000 카드를 사용하면 Secure Shell은 레벨 3의 FIPS-140 지원으로 실행됩니다. 레벨 3 하드웨어는 물리적 변조를 방지하고, ID 기반 인증을 사용하고, 하드웨어의 기타 인터페이스에서 중요한 보안 매개변수를 처리하는 인터페이스를 격리할 수 있는 것으로 인증되었습니다.

Secure Shell(작업 맵)

다음 작업 맵에서는 Secure Shell 구성 및 Oracle Solaris의 Secure Shell 기능 사용과 관련된 작업 맵에 대한 링크를 제공합니다.

작업	설명	수행 방법
Secure Shell을 구성합니다.	관리자에게 사용자에 대한 Secure Shell 구성을 안내합니다.	297 페이지 “Secure Shell 구성(작업 맵)”
Secure Shell을 사용합니다.	사용자에게 Secure Shell 사용을 안내합니다.	301 페이지 “Secure Shell 사용(작업 맵)”

Secure Shell 구성(작업)

기본적으로 Secure Shell에서는 호스트 기반 인증 및 두 프로토콜 사용이 사용으로 설정되어 있지 않습니다. 이러한 기본값을 변경하려면 관리 개입이 필요합니다. 포트 전달이 작동되도록 하려는 경우에도 관리 개입이 필요합니다.

Secure Shell 구성(작업 맵)

다음 작업 맵에서는 Secure Shell 구성 절차에 대해 설명합니다.

작업	설명	수행 방법
호스트 기반 인증을 구성합니다.	클라이언트와 서버에서 호스트 기반 인증을 구성합니다.	297 페이지 “Secure Shell에 대한 호스트 기반 인증 설정 방법”
포트 전달을 구성합니다.	사용자가 포트 전달을 사용할 수 있도록 합니다.	300 페이지 “Secure Shell에서 포트 전달을 구성하는 방법”
SSH 시스템 기본값에 대한 예외를 구성합니다.	사용자, 호스트, 그룹 및 주소에 대해 시스템 기본값과 다른 SSH 설정을 지정합니다.	300 페이지 “SSH 시스템 기본값에 대한 사용자 및 호스트 예외를 만드는 방법”

▼ Secure Shell에 대한 호스트 기반 인증 설정 방법

다음 절차에서는 서버에서 클라이언트의 개인 키가 인증에 사용되는 공개 키 시스템을 설정합니다. 사용자는 공개/개인 키 쌍을 만들어야 합니다.

절차에서 언급되는 **클라이언트**와 **로컬 호스트**라는 용어는 사용자가 ssh 명령을 입력한 시스템을 나타냅니다. **서버**와 **원격 호스트**라는 용어는 클라이언트가 연결하려고 시도 중인 시스템을 나타냅니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1. **클라이언트에서 호스트 기반 인증을 사용으로 설정합니다.**

   클라이언트 구성 파일 /etc/ssh/ssh_config에서 다음 항목을 입력합니다.

   ```
   HostbasedAuthentication yes
   ```

   파일 구문은 ssh_config(4) 매뉴얼 페이지를 참조하십시오.

2. **서버에서 호스트 기반 인증을 사용으로 설정합니다.**

   서버 구성 파일 /etc/ssh/sshd_config에서 동일한 항목을 입력합니다.

   ```
   HostbasedAuthentication yes
   ```

   파일 구문은 sshd_config(4) 매뉴얼 페이지를 참조하십시오.

3. **서버에서 클라이언트가 신뢰할 수 있는 호스트로 인식될 수 있도록 하는 파일을 구성합니다.**

   자세한 내용은 sshd(1M) 매뉴얼 페이지의 FILES 절을 참조하십시오.

   - **서버의 /etc/ssh/shosts.equiv 파일에 클라이언트를 항목으로 추가합니다.**

     client-host

   - **또는 사용자에게 ~/.shosts 파일에 클라이언트에 대한 항목을 추가하도록 할 수도 있습니다.**

     client-host

4. **서버에서 sshd 데몬이 신뢰할 수 있는 호스트 목록에 액세스할 수 있는지 확인합니다.**

   /etc/ssh/sshd_config 파일에서 IgnoreRhosts를 no로 설정합니다.

   ```
   ## sshd_config
   IgnoreRhosts no
   ```

5. **사이트의 Secure Shell 사용자에게 두 호스트에 대한 계정이 있는지 확인합니다.**

6. **다음 중 하나를 수행하여 서버에서 클라이언트의 공개 키를 삽입합니다.**

   - **서버에서 sshd_config 파일을 수정한 다음 사용자에게 ~/.ssh/known_hosts 파일에 클라이언트의 공개 호스트 키를 추가하도록 합니다.**

     ```
     ## sshd_config
     IgnoreUserKnownHosts no
     ```

     사용자 지침은 302 페이지 “Secure Shell에서 사용할 공개/개인 키 쌍 생성 방법”을 참조하십시오.

- **서버에 클라이언트의 공개 키를 복사합니다.**

  호스트 키는 /etc/ssh 디렉토리에 저장되어 있습니다. 일반적으로 첫번째 부트 시 sshd 데몬이 키를 생성합니다.

  a. **서버에서 /etc/ssh/ssh_known_hosts 파일에 키를 추가합니다.**

     클라이언트에서 한 행에 백슬래시 없이 명령을 입력합니다.

     ```
     # cat /etc/ssh/ssh_host_dsa_key.pub | ssh RemoteHost \
     'cat >> /etc/ssh/ssh_known_hosts && echo "Host key copied"'
     ```

  b. **프롬프트가 표시되면 로그인 암호를 제공합니다.**

     파일이 복사되면 "Host key copied" 메시지가 표시됩니다.

     /etc/ssh/ssh_known_hosts 파일의 각 행은 공백으로 구분된 필드로 구성됩니다.

     hostnames algorithm-name publickey comment

  c. **/etc/ssh/ssh_known_hosts 파일을 편집하고 복사된 항목에 *RemoteHost*를 첫번째 필드로 추가합니다.**

     ```
     ## /etc/ssh/ssh_known_hosts File
     RemoteHost  <copied entry>
     ```

예 17–1 호스트 기반 인증 설정

다음 예에서는 각 호스트가 서버와 클라이언트로 구성됩니다. 각 호스트의 사용자는 다른 호스트에 대한 ssh 연결을 시작할 수 있습니다. 다음은 각 호스트를 서버와 클라이언트로 만드는 구성입니다.

- 각 호스트에서 Secure Shell 구성 파일에는 다음 항목이 포함되어 있습니다.

  ```
  ## /etc/ssh/ssh_config
  HostBasedAuthentication yes
  #
  ## /etc/ssh/sshd_config
  HostBasedAuthentication yes
  IgnoreRhosts no
  ```

- 각 호스트에서 shosts.equiv 파일에는 다른 호스트에 대한 항목이 포함되어 있습니다.

  ```
  ## /etc/ssh/shosts.equiv on machine2
  machine1

  ## /etc/ssh/shosts.equiv on machine1
  machine2
  ```

- 각 호스트의 공개 키는 다른 호스트의 /etc/ssh/ssh_known_hosts 파일에 있습니다.

  ```
  ## /etc/ssh/ssh_known_hosts on machine2
  ... machine1

  ## /etc/ssh/ssh_known_hosts on machine1
  ... machine2
  ```

- 사용자는 다음과 같이 두 호스트에 대한 계정을 가집니다.

```
## /etc/passwd on machine1
jdoe:x:3111:10:J Doe:/home/jdoe:/bin/sh

## /etc/passwd on machine2
jdoe:x:3111:10:J Doe:/home/jdoe:/bin/sh
```

▼ Secure Shell에서 포트 전달을 구성하는 방법

포트 전달은 로컬 포트가 원격 호스트로 전달될 수 있도록 합니다. 소켓이 로컬측의 포트를 수신 대기하도록 효과적으로 할당됩니다. 마찬가지로 원격측에서도 포트를 지정할 수 있습니다.

주 – Secure Shell 포트 전달에는 TCP 연결이 사용되어야 합니다. Secure Shell은 포트 전달을 위해 UDP 연결을 지원하지 않습니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

1 포트 전달을 허용하도록 원격 서버의 Secure Shell 설정을 구성합니다.

`/etc/ssh/sshd_config` 파일에서 `AllowTcpForwarding`의 값을 `yes`로 변경합니다.

```
# Port forwarding
AllowTcpForwarding yes
```

2 Secure Shell 서비스를 다시 시작합니다.

```
remoteHost# svcadm restart network/ssh:default
```

지속 서비스 관리에 대한 자세한 내용은 **Oracle Solaris 관리: 일반 작업**의 6 장, “서비스 관리(개요)” 및 `svcadm`(1M) 매뉴얼 페이지를 참조하십시오.

3 포트 전달을 사용할 수 있는지 확인합니다.

```
remoteHost# /usr/bin/pgrep -lf sshd
 1296 ssh -L 2001:remoteHost:23 remoteHost
```

▼ SSH 시스템 기본값에 대한 사용자 및 호스트 예외를 만드는 방법

이 절차에서는 `/etc/ssh/sshd_config` 파일의 전역 섹션 뒤에 조건부 `Match` 블록을 추가합니다. `Match` 블록 뒤의 키워드/값 쌍은 일치 항목으로 지정된 사용자, 그룹, 호스트 또는 주소에 대한 예외를 지정합니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

1 `sshd_config` 파일을 편집합니다.

2 **기본 설정 이외의 다른 SSH 키워드 설정을 사용하도록 사용자, 그룹, 호스트 또는 주소를 구성합니다.**

Match 블록을 전역 설정 뒤에 배치합니다.

주 – 파일의 전역 섹션에는 기본 설정이 나열될 수도 있고 나열되지 않을 수도 있습니다. 기본값은 sshd_config(4) 매뉴얼 페이지를 참조하십시오.

TCP 전달을 사용할 수 있도록 허용되지 않아야 할 사용자가 있을 수도 있습니다. 다음 예에서는 public 그룹의 사용자 및 이름이 test로 시작하는 사용자가 TCP 전달을 사용할 수 없습니다.

```
## sshd_config file
## Global settings

# Example (reflects default settings):
#
# Host *
#   ForwardAgent no
#   ForwardX11 no
#   PubkeyAuthentication yes
#   PasswordAuthentication yes
#   FallBackToRsh no
#   UseRsh no
#   BatchMode no
#   CheckHostIP yes
#   StrictHostKeyChecking ask
#   EscapeChar ~
Match Group public
 AllowTcpForwarding no
Match User test*
 AllowTcpForwarding no
```

Match 블록 구문에 대한 자세한 내용은 sshd_config(4) 매뉴얼 페이지를 참조하십시오.

Secure Shell 사용(작업)

Secure Shell은 로컬 셸과 원격 셸 간의 보안 액세스를 제공합니다. 자세한 내용은 ssh_config(4) 및 ssh(1) 매뉴얼 페이지를 참조하십시오.

Secure Shell 사용(작업 맵)

다음 작업 맵에서는 사용자의 Secure Shell 사용 절차에 대해 설명합니다.

작업	설명	수행 방법
공개/개인 키 쌍을 만듭니다.	공개 키 인증이 필요한 사이트의 Secure Shell에 대한 액세스를 사용으로 설정합니다.	302 페이지 “Secure Shell에서 사용할 공개/개인 키 쌍 생성 방법”

작업	설명	수행 방법
암호문을 변경합니다.	개인 키를 인증하는 구문을 변경합니다.	304 페이지 “Secure Shell 개인 키에 대한 암호문 변경 방법”
Secure Shell을 사용하여 로그인합니다.	원격으로 로그인할 때는 암호화된 Secure Shell 통신을 제공합니다. 이 프로세스는 rsh 명령을 사용하는 것과 유사합니다.	304 페이지 “Secure Shell을 사용하여 원격 호스트에 로그인하는 방법”
암호를 입력하지 않고 Secure Shell에 로그인합니다.	Secure Shell에 사용자 암호를 제공하는 에이전트를 사용하여 로그인할 수 있도록 합니다.	305 페이지 “Secure Shell에서 암호 프롬프트를 줄이는 방법”
Secure Shell에서 포트 전달을 사용합니다.	TCP를 통한 Secure Shell 연결에서 사용할 로컬 포트 또는 원격 포트를 지정합니다.	307 페이지 “Secure Shell에서 포트 전달을 사용하는 방법”
Secure Shell을 사용하여 파일을 복사합니다.	호스트 간에 안전하게 파일을 복사합니다.	308 페이지 “Secure Shell을 사용하여 파일을 복사하는 방법”
방화벽 내부의 호스트에서 방화벽 외부의 호스트에 안전하게 연결합니다.	HTTP 또는 SOCKS5와 호환되는 Secure Shell 명령을 사용하여 방화벽으로 분리된 호스트를 연결합니다.	309 페이지 “방화벽 외부의 호스트에 대한 기본 연결 설정 방법”

▼ Secure Shell에서 사용할 공개/개인 키 쌍 생성 방법

사용자 사이트에서 호스트 기반 인증 또는 사용자 공개 키 인증을 구현한 경우 사용자는 공개/개인 키 쌍을 생성해야 합니다. 추가 옵션은 ssh-keygen(1) 매뉴얼 페이지를 참조하십시오.

시작하기 전에 시스템 관리자에게 호스트 기반 인증이 구성되어 있는지 여부를 확인합니다.

1 키 생성 프로그램을 시작합니다.

```
myLocalHost% ssh-keygen -t rsa
Generating public/private rsa key pair.
...
```

여기서 -t는 알고리즘 유형으로, rsa, dsa 또는 rsa1 중 하나입니다.

2 키를 보관할 파일의 경로를 지정합니다.

기본적으로 RSA v2 키를 나타내는 파일 이름 id_rsa가 괄호 안에 표시됩니다. Return 키를 눌러 이 파일을 선택할 수 있습니다. 또는 대체 파일 이름을 입력할 수도 있습니다.

```
Enter file in which to save the key (/home/jdoe/.ssh/id_rsa):     <Press Return>
```

.pub 문자열을 개인 키 파일 이름에 추가하면 공개 키의 파일 이름이 자동으로 만들어집니다.

3 **키를 사용하는 데 필요한 암호문을 입력합니다.**

이 암호문은 개인 키를 암호화하는 데 사용됩니다. 널 항목은 **사용하지 않는 것이 좋습니다**. 암호문은 입력할 때 표시되지 않습니다.

```
Enter passphrase (empty for no passphrase):      <Type passphrase>
```

4 **확인용으로 암호문을 다시 입력합니다.**

```
Enter same passphrase again:      <Type passphrase>
Your identification has been saved in /home/jdoe/.ssh/id_rsa.
Your public key has been saved in /home/jdoe/.ssh/id_rsa.pub.
The key fingerprint is:
0e:fb:3d:57:71:73:bf:58:b8:eb:f3:a3:aa:df:e0:d1 jdoe@myLocalHost
```

5 **결과를 확인합니다.**

키 파일의 경로가 올바른지 확인합니다.

```
% ls ~/.ssh
id_rsa
id_rsa.pub
```

이 단계에서는 공개/개인 키 쌍이 만들어져 있습니다.

6 **적절한 옵션을 선택합니다.**

- **관리자가 호스트 기반 인증을 구성한 경우 로컬 호스트의 공개 키를 원격 호스트에 복사해야 할 수도 있습니다.**

  이제 원격 호스트에 로그인할 수 있습니다. 자세한 내용은 304 페이지 “Secure Shell을 사용하여 원격 호스트에 로그인하는 방법”을 참조하십시오.

  a. **한 행에 백슬래시 없이 명령을 입력합니다.**

  ```
  % cat /etc/ssh/ssh_host_dsa_key.pub | ssh RemoteHost \
  'cat >> ~./ssh/known_hosts && echo "Host key copied"'
  ```

  b. **프롬프트가 표시되면 로그인 암호를 제공합니다.**

  ```
  Enter password:      <Type password>
  Host key copied
  %
  ```

- **사이트에서 공개 키를 통한 사용자 인증을 사용하는 경우 원격 호스트에서 `authorized_keys` 파일을 채웁니다.**

  a. **공개 키를 원격 호스트에 복사합니다.**

  한 행에 백슬래시 없이 명령을 입력합니다.

  ```
  myLocalHost% cat $HOME/.ssh/id_rsa.pub | ssh myRemoteHost \
  'cat >> .ssh/authorized_keys && echo "Key copied"'
  ```

b. 프롬프트가 표시되면 로그인 암호를 제공합니다.

파일이 복사되면 "Key copied" 메시지가 표시됩니다.

```
Enter password:       Type login password
Key copied
myLocalHost%
```

7 (옵션) 암호문에 대한 프롬프트를 줄입니다.

절차는 305 페이지 “Secure Shell에서 암호 프롬프트를 줄이는 방법”을 참조하십시오. 자세한 내용은 `ssh-agent(1)` 및 `ssh-add(1)` 매뉴얼 페이지를 참조하십시오.

▼ Secure Shell 개인 키에 대한 암호문 변경 방법

다음 절차에서는 개인 키를 변경하지 않습니다. 이 절차에서는 개인 키에 대한 인증 방식인 암호문을 변경합니다. 자세한 내용은 `ssh-keygen(1)` 매뉴얼 페이지를 참조하십시오.

- **암호문을 변경합니다.**

  `-p` 옵션을 사용하여 `ssh-keygen` 명령을 입력하고 프롬프트에 응답합니다.

```
myLocalHost% ssh-keygen -p
Enter file which contains the private key (/home/jdoe/.ssh/id_rsa):     <Press Return>
Enter passphrase (empty for no passphrase):      <Type passphrase>
Enter same passphrase again:       <Type passphrase>
```

  여기서 `-p`는 개인 키 파일의 암호문 변경을 요청합니다.

▼ Secure Shell을 사용하여 원격 호스트에 로그인하는 방법

1 Secure Shell 세션을 시작합니다.

`ssh` 명령을 입력하고 원격 호스트의 이름 및 로그인을 지정합니다.

```
myLocalHost% ssh myRemoteHost -l username
```

프롬프트가 원격 호스트의 신뢰성을 묻습니다.

```
The authenticity of host 'myRemoteHost' can't be established.
RSA key fingerprint in md5 is: 04:9f:bd:fc:3d:3e:d2:e7:49:fd:6e:18:4f:9c:26
Are you sure you want to continue connecting(yes/no)?
```

일반적으로 이 프롬프트는 원격 호스트에 대한 초기 연결에 표시됩니다.

2 **프롬프트가 표시되면 원격 호스트 키의 신뢰성을 확인합니다.**

- **원격 호스트의 신뢰성을 확인할 수 없을 경우 no를 입력하고 시스템 관리자에게 문의하십시오.**

  ```
  Are you sure you want to continue connecting(yes/no)? no
  ```

  관리자가 전역 /etc/ssh/ssh_known_hosts 파일을 업데이트합니다. 업데이트된 ssh_known_hosts 파일은 이 프롬프트가 표시되지 않도록 합니다.

- **원격 호스트의 신뢰성을 확인한 경우 프롬프트에 응답하고 다음 단계를 계속합니다.**

  ```
  Are you sure you want to continue connecting(yes/no)? yes
  ```

3 **Secure Shell에 대해 자신을 인증합니다.**

a. **프롬프트가 표시되면 암호문을 입력합니다.**

```
Enter passphrase for key '/home/jdoe/.ssh/id_rsa':      <Type passphrase>
```

b. **프롬프트가 표시되면 계정 암호를 입력합니다.**

```
jdoe@myRemoteHost's password:      <Type password>
Last login: Wed Sep  7 09:07:49 2011 from myLocalHost
Oracle Corporation      SunOS 5.11       September 2011
myRemoteHost%
```

4 **원격 호스트에서 트랜잭션을 수행합니다.**

보낸 명령이 암호화되고, 수신한 응답이 암호화됩니다.

5 **Secure Shell 연결을 해제합니다.**

완료되면 **exit**를 입력하거나 일반적인 셸 종료 방법을 사용합니다.

```
myRemoteHost% exit
myRemoteHost% logout
Connection to myRemoteHost closed
myLocalHost%
```

▼ Secure Shell에서 암호 프롬프트를 줄이는 방법

암호문 및 암호를 입력하지 않고 Secure Shell을 사용하려는 경우 에이전트 데몬을 사용할 수 있습니다. 세션 시작 시 데몬을 시작합니다. 그런 다음 ssh-add 명령을 사용하여 에이전트 데몬으로 개인 키를 저장합니다. 호스트마다 계정이 다른 경우 세션에 필요한 키를 추가합니다.

필요한 경우 다음 절차의 설명에 따라 수동으로 에이전트 데몬을 시작할 수 있습니다.

1 **에이전트 데몬을 시작합니다.**

```
myLocalHost% eval 'ssh-agent'
Agent pid 9892
```

2 에이전트 데몬이 시작되었는지 확인합니다.

```
myLocalHost% pgrep ssh-agent
9892
```

3 에이전트 데몬에 개인 키를 추가합니다.

ssh-add 명령을 입력합니다.

```
myLocalHost% ssh-add
Enter passphrase for /home/jdoe/.ssh/id_rsa:      <Type passphrase>
Identity added: /home/jdoe/.ssh/id_rsa(/home/jdoe/.ssh/id_rsa)
myLocalHost%
```

4 Secure Shell 세션을 시작합니다.

```
myLocalHost% ssh myRemoteHost -l jdoe
```

암호문 프롬프트가 표시되지 않습니다.

예 17–2 ssh-add 옵션 사용

이 예에서는 jdoe가 에이전트 데몬에 두 개의 키를 추가합니다. -l 옵션이 데몬에 저장된 모든 키를 나열하는 데 사용됩니다. 세션 종료 시 -D 옵션이 에이전트 데몬에서 모든 키를 제거하는 데 사용됩니다.

```
myLocalHost% ssh-agent
myLocalHost% ssh-add
Enter passphrase for /home/jdoe/.ssh/id_rsa:      <Type passphrase>
Identity added: /home/jdoe/.ssh/id_rsa(/home/jdoe/.ssh/id_rsa)
myLocalHost% ssh-add /home/jdoe/.ssh/id_dsa
Enter passphrase for /home/jdoe/.ssh/id_dsa:      <Type passphrase>
Identity added:
/home/jdoe/.ssh/id_dsa(/home/jdoe/.ssh/id_dsa)

myLocalHost% ssh-add -l
md5 1024 0e:fb:3d:53:71:77:bf:57:b8:eb:f7:a7:aa:df:e0:d1
/home/jdoe/.ssh/id_rsa(RSA)
md5 1024 c1:d3:21:5e:40:60:c5:73:d8:87:09:3a:fa:5f:32:53
/home/jdoe/.ssh/id_dsa(DSA)
```

User conducts Oracle Solaris Secure Shell transactions

```
myLocalHost% ssh-add -D
Identity removed:
/home/jdoe/.ssh/id_rsa(/home/jdoe/.ssh/id_rsa.pub)
/home/jdoe/.ssh/id_dsa(DSA)
```

▼ Secure Shell에서 포트 전달을 사용하는 방법

로컬 포트가 원격 호스트에 전달되도록 지정할 수 있습니다. 소켓이 로컬측의 포트를 수신 대기하도록 효과적으로 할당됩니다. 보안 채널을 통해 이 포트에서 원격 호스트로의 연결이 설정됩니다. 예를 들어, IMAP4를 사용하여 원격으로 전자 메일을 얻기 위해 포트 143을 지정할 수 있습니다. 마찬가지로 원격측에서도 포트를 지정할 수 있습니다.

시작하기 전에 포트 전달을 사용하려면 관리자가 원격 Secure Shell 서버에서 포트 전달을 사용으로 설정했어야 합니다. 자세한 내용은 300 페이지 “Secure Shell에서 포트 전달을 구성하는 방법”을 참조하십시오.

- **보안 포트 전달을 사용하려면 다음 옵션 중 하나를 선택합니다.**

  - **원격 포트에서 보안 통신을 수신하도록 로컬 포트를 설정하려면 두 포트를 지정합니다.**

    원격 통신을 수신 대기하는 로컬 포트를 지정합니다. 또한 통신을 전달하는 원격 호스트 및 원격 포트를 지정합니다.

    ```
    myLocalHost% ssh -L localPort:remoteHost:remotePort
    ```

  - **로컬 포트에서 보안 연결을 수신하도록 원격 포트를 설정하려면 두 포트를 지정합니다.**

    원격 통신을 수신 대기하는 원격 포트를 지정합니다. 또한 통신을 전달하는 로컬 호스트 및 로컬 포트를 지정합니다.

    ```
    myLocalHost% ssh -R remotePort:localhost:localPort
    ```

예 17–3 로컬 포트 전달을 사용하여 메일 수신

다음 예에서는 로컬 포트 전달을 사용하여 원격 서버에서 안전하게 메일을 수신할 수 있는 방법을 보여 줍니다.

```
myLocalHost% ssh -L 9143:myRemoteHost:143 myRemoteHost
```

이 명령은 myLocalHost의 포트 9143에서 포트 143으로 연결을 전달합니다. 포트 143은 myRemoteHost의 IMAP v2 서버 포트입니다. 메일 응용 프로그램을 실행한 사용자는 localhost:9143에서처럼 IMAP 서버에 대한 로컬 포트 번호를 지정합니다.

localhost와 myLocalHost를 혼동하지 마십시오. myLocalHost는 가상 호스트 이름이며, localhost는 로컬 시스템을 식별하는 키워드입니다.

예 17–4 원격 포트 전달을 사용하여 방화벽 외부에서 통신

이 예에서는 기업 환경의 사용자가 외부 네트워크의 호스트에서 회사 방화벽 내부의 호스트로 연결을 전달할 수 있는 방법을 보여 줍니다.

```
myLocalHost% ssh -R 9022:myLocalHost:22 myOutsideHost
```

이 명령은 myOutsideHost의 포트 9022에서 로컬 호스트의 포트 22 sshd 서버로 연결을 전달합니다.

```
myOutsideHost% ssh -p 9022 localhost
myLocalHost%
```

▼ Secure Shell을 사용하여 파일을 복사하는 방법

다음 절차에서는 scp 명령을 사용하여 호스트 간에 암호화된 파일을 복사하는 방법을 보여 줍니다. 로컬 호스트와 원격 호스트 간 또는 두 원격 호스트 간에 암호화된 파일을 복사할 수 있습니다. scp 명령은 인증 프롬프트를 표시합니다. 자세한 내용은 scp(1) 매뉴얼 페이지를 참조하십시오.

또한 sftp 보안 파일 전송 프로그램을 사용할 수 있습니다. 자세한 내용은 sftp(1) 매뉴얼 페이지를 참조하십시오. 예는 예 17–5를 참조하십시오.

주 – 감사 서비스는 ft 감사 클래스를 통해 sftp 트랜잭션을 감사할 수 있습니다. scp의 경우 감사 서비스는 ssh 세션에 대한 액세스 및 종료를 감사할 수 있습니다.

---

1 보안 복사 프로그램을 시작합니다.

소스 파일, 원격 대상의 사용자 이름 및 대상 디렉토리를 지정합니다.

```
myLocalHost% scp myfile.1 jdoe@myRemoteHost:~
```

2 프롬프트가 표시되면 암호문을 제공합니다.

```
Enter passphrase for key '/home/jdoe/.ssh/id_rsa':      <Type passphrase>
myfile.1       25% |*******                     |    640 KB  0:20 ETA
myfile.1
```

암호문을 입력하면 진행 상황이 표시됩니다. 위 출력의 두번째 행을 참조하십시오. 다음과 같이 진행 상황이 표시됩니다.

- 파일 이름
- 파일 전송 백분율
- 파일 전송 백분율을 나타내는 일련의 별표
- 데이터 전송량
- 전체 파일의 예상 도착 시간(ETA), 즉 남은 시간

예 17–5 sftp 명령 사용 시 포트 지정

이 예에서는 사용자가 sftp 명령으로 특정 포트를 사용하려고 합니다. 사용자는 -o 옵션을 사용하여 포트를 지정합니다.

```
% sftp -o port=2222 guest@RemoteFileServer
```

▼ 방화벽 외부의 호스트에 대한 기본 연결 설정 방법

Secure Shell을 사용하여 방화벽 내부의 호스트에서 방화벽 외부의 호스트로의 연결을 설정할 수 있습니다. 이 작업을 수행하려면 구성 파일에서 또는 명령줄 옵션으로 ssh에 대한 프록시 명령을 지정합니다. 명령줄 옵션은 예 17–6을 참조하십시오.

일반적으로 구성 파일을 통해 ssh 상호 작용을 사용자 정의할 수 있습니다.

- ~/.ssh/config에서 고유의 개인 파일을 사용자 정의할 수 있습니다.
- 또한 관리 구성 파일 /etc/ssh/ssh_config의 설정을 사용할 수도 있습니다.

두 가지 유형의 프록시 명령으로 파일을 사용자 정의할 수 있습니다. 프록시 명령 중 하나는 HTTP 연결에 사용되며, 나머지 하나는 SOCKS5 연결에 사용됩니다. 자세한 내용은 ssh_config(4) 매뉴얼 페이지를 참조하십시오.

1 구성 파일에서 프록시 명령 및 호스트를 지정합니다.

다음 구문을 사용하여 필요에 따라 행을 여러 개 추가합니다.

```
[Host outside-host]
ProxyCommand proxy-command [-h proxy-server] \
[-p proxy-port] outside-host|%h outside-port|%p
```

Host *outside-host*
: 명령줄에서 원격 호스트 이름이 지정된 경우 프록시 명령 지정을 인스턴스로 제한합니다. *outside-host*에 와일드카드를 사용하면 일련의 호스트에 프록시 명령 지정이 적용됩니다.

proxy-command
: 프록시 명령을 지정합니다.

  명령은 다음 중 하나일 수 있습니다.

  - HTTP 연결의 경우 /usr/lib/ssh/ssh-http-proxy-connect
  - SOCKS5 연결의 경우 /usr/lib/ssh/ssh-socks5-proxy-connect

-h *proxy-server* 및 -p *proxy-port*
: 해당 옵션은 각각 프록시 서버와 프록시 포트를 지정합니다. 있을 경우 프록시는 프록시 서버 및 프록시 포트를 지정하는 환경 변수(예: HTTPPROXY, HTTPPROXYPORT, SOCKS5_PORT, SOCKS5_SERVER 및 http_proxy)를 대체합니다. http_proxy 변수는 URL을 지정합니다. 옵션이 사용되지 않을 경우 관련 환경 변수를 설정해야 합니다. 자세한 내용은 ssh-socks5-proxy-connect(1) 및 ssh-http-proxy-connect(1) 매뉴얼 페이지를 참조하십시오.

outside-host
: 연결할 특정 호스트를 지정합니다. 명령줄에서 호스트를 지정하려면 %h 대체 인수를 사용합니다.

outside-port
: 연결할 특정 포트를 지정합니다. 명령줄에서 포트를 지정하려면 %p 대체 인수를 사용합니다. Host *outside-host* 옵션을 사용하지 않고 %h 및 %p를 지정하면 ssh 명령이

호출될 때마다 호스트 인수에 프록시 명령이 적용됩니다.

2 **외부 호스트를 지정하여 Secure Shell을 실행합니다.**

예를 들어, 다음 명령어를 입력합니다.

```
myLocalHost% ssh myOutsideHost
```

이 명령은 개인 구성 파일에서 myOutsideHost에 대한 프록시 명령 지정을 검색합니다. 지정을 찾을 수 없을 경우 명령은 시스템 차원의 구성 파일 /etc/ssh/ssh_config 에서 찾습니다. ssh 명령이 프록시 명령으로 대체됩니다.

예 17–6 명령줄에서 방화벽 외부의 호스트에 연결

구성 파일에서 프록시 명령을 지정하는 방법은 309 페이지 "방화벽 외부의 호스트에 대한 기본 연결 설정 방법"에서 설명됩니다. 이 예에서는 ssh 명령줄에서 프록시 명령이 지정됩니다.

```
% ssh -o'Proxycommand=/usr/lib/ssh/ssh-http-proxy-connect \
-h myProxyServer -p 8080 myOutsideHost 22' myOutsideHost
```

ssh 명령에 대한 -o 옵션은 명령줄에서 프록시 명령을 지정하는 방법을 제공합니다. 이 예의 명령은 다음을 수행합니다.

- ssh를 HTTP 프록시 명령으로 대체합니다.
- 포트 8080 및 myProxyServer를 프록시 서버로 사용합니다.
- myOutsideHost의 포트 22에 연결합니다.

◆ ◆ ◆ 18 장

18 Secure Shell(참조)

이 장에서는 Oracle Solaris의 Secure Shell 기능에 포함된 구성 옵션에 대해 설명합니다. 다음은 이 장에 포함된 참조 정보 목록입니다.

Secure Shell 구성 절차는 17 장, “Secure Shell 사용(작업)”을 참조하십시오.

일반적인 Secure Shell 세션

일반적으로 Secure Shell 데몬(`sshd`)은 부트 시 네트워크 서비스가 시작될 때 시작됩니다. 데몬은 클라이언트로부터의 연결을 수신 대기합니다. Secure Shell 세션은 사용자가 `ssh`, `scp` 또는 `sftp` 명령을 실행할 때 시작됩니다. 새 `sshd` 데몬은 각 수신 연결에 대해 포크됩니다. 포크된 데몬은 키 교환, 암호화, 인증, 명령 실행 및 클라이언트와의 데이터 교환을 처리합니다. 이러한 세션 특성은 클라이언트측 구성 파일 및 서버측 구성 파일로 결정됩니다. 명령줄 인수는 구성 파일의 설정을 대체할 수 있습니다.

클라이언트와 서버는 상호 간에 자체적으로 인증되어야 합니다. 인증에 성공하면 사용자가 원격으로 명령을 실행하고 호스트 간에 데이터를 복사할 수 있습니다.

Secure Shell의 세션 특성

`sshd` 데몬의 서버측 동작은 `/etc/ssh/sshd_config` 파일의 키워드 설정으로 제어됩니다. 예를 들어, `sshd_config` 파일은 서버에 대한 액세스가 허용되는 인증 유형을 제어합니다. 서버측 동작은 `sshd` 데몬이 시작된 경우 명령줄 옵션으로도 제어될 수 있습니다.

클라이언트측 동작은 다음 우선 순위에 따라 Secure Shell 키워드로 제어됩니다.

- 명령줄 옵션
- 사용자의 구성 파일 `~/.ssh/config`
- 시스템 차원의 구성 파일 `/etc/ssh/ssh_config`

예를 들어, 사용자는 명령줄에서 `-c aes256–ctr,aes128-ctr,arcfour`를 지정하여 `aes128–ctr`보다 우선하는 시스템 차원의 구성 `Ciphers` 설정을 대체할 수 있습니다. 그러면 첫번째 암호 `aes256–ctr`이 우선합니다.

Secure Shell의 인증 및 키 교환

Secure Shell 프로토콜은 클라이언트 사용자/호스트 인증 및 서버 호스트 인증을 지원합니다. Secure Shell 세션을 보호하기 위해 암호화 키가 교환됩니다. Secure Shell은 다양한 인증 및 키 교환 방법을 제공합니다. 일부 방법은 선택 사항입니다. 클라이언트 인증 방식은 표 17–1에서 나열됩니다. 서버는 알려진 호스트 공개 키를 사용하여 인증됩니다.

인증의 경우 Secure Shell은 사용자 인증 및 주로 암호가 사용되는 일반적인 대화식 인증을 지원합니다. 또한 Secure Shell은 사용자 공개 키 및 신뢰할 수 있는 호스트 공개 키를 통한 인증을 지원합니다. 키는 RSA 또는 DSA일 수 있습니다. 세션 키는 서버 인증 단계에서 서명된 일시적인 Diffie-Hellman 키 교환으로 구성됩니다. 또한 Secure Shell은 인증에 GSS 자격 증명을 사용할 수 있습니다.

Secure Shell에서 GSS 자격 증명 취득

Secure Shell에서 인증에 GSS-API를 사용하려면 서버에 GSS-API 승인자 자격 증명이 있고 클라이언트에 GSS-API 개시자 자격 증명이 있어야 합니다. `mech_dh` 및 `mech_krb5`에 대한 지원이 제공됩니다.

`mech_dh`의 경우 `root`가 `keylogin` 명령을 실행했으면 서버에 GSS-API 승인자 자격 증명이 있는 것입니다.

`mech_krb5`의 경우 서버에 해당하는 호스트 주체의 `/etc/krb5/krb5.keytab`에 유효한 항목이 있으면 서버에 GSS-API 승인자 자격 증명이 있는 것입니다.

다음 중 하나가 완료된 경우 클라이언트에 `mech_dh`에 대한 개시자 자격 증명이 있는 것입니다.

- `keylogin` 명령이 실행된 경우
- `pam_dhkeys` 모듈이 `pam.conf` 파일에서 사용된 경우

다음 중 하나가 완료된 경우 클라이언트에 `mech_krb5`에 대한 개시자 자격 증명이 있는 것입니다.

- `kinit` 명령이 실행된 경우
- `pam_krb5` 모듈이 `pam.conf` 파일에서 사용된 경우

보안 RPC에서 mech_dh를 사용하는 방법은 14 장, “네트워크 서비스 인증(작업)”을 참조하십시오. mech_krb5를 사용하는 방법은 19 장, “Kerberos 서비스 소개”를 참조하십시오. 방식에 대한 자세한 내용은 mech(4) 및 mech_spnego(5) 매뉴얼 페이지를 참조하십시오.

Secure Shell의 명령 실행 및 데이터 전달

인증이 완료되면 사용자가 일반적으로 셸을 요청하거나 명령을 실행하여 Secure Shell을 사용할 수 있습니다. 사용자는 ssh 명령 옵션을 통해 요청을 생성할 수 있습니다. 예를 들어, 의사 tty를 할당하거나 X11 연결 또는 TCP/IP 연결을 전달하거나 보안 연결을 통해 ssh-agent 인증 프로그램을 사용으로 설정하는 요청을 생성할 수 있습니다.

기본적인 사용자 세션 구성 요소는 다음과 같습니다.

1. 사용자가 세션 모드를 시작하는 셸 또는 명령 실행을 요청합니다.

   이 모드에서는 데이터가 클라이언트측 터미널을 통해 전송 또는 수신됩니다. 서버측에서는 데이터가 셸 또는 명령을 통해 전송됩니다.
2. 데이터 전송이 완료되면 사용자 프로그램이 종료됩니다.
3. 기존 연결을 제외하고 모든 X11 전달 및 TCP/IP 전달이 중지됩니다. 기존 X11 연결 및 TCP/IP 연결은 열린 상태로 유지됩니다.
4. 서버가 클라이언트로 종료 상태 메시지를 보냅니다. 열린 상태로 유지되었던 전달된 포트 등 모든 연결이 해제되면 클라이언트가 서버에 대한 연결을 해제합니다. 그런 다음 클라이언트가 종료됩니다.

Secure Shell의 클라이언트 및 서버 구성

Secure Shell 세션의 특성은 구성 파일로 제어됩니다. 구성 파일은 명령줄의 옵션에 의해 특정 수준으로 대체될 수 있습니다.

Secure Shell의 클라이언트 구성

대부분의 경우 Secure Shell 세션의 클라이언트측 특성은 시스템 차원의 구성 파일 /etc/ssh/ssh_config로 제어됩니다. ssh_config 파일의 설정은 사용자의 구성 파일 ~/.ssh/config로 대체할 수 있습니다. 사용자는 명령줄에서 두 구성 파일을 대체할 수도 있습니다.

서버의 /etc/ssh/sshd_config 파일 설정에 따라 서버가 허용하는 클라이언트 요청이 결정됩니다. 서버 구성 설정 목록은 314 페이지 “Secure Shell의 키워드”를 참조하십시오. 자세한 내용은 sshd_config(4) 매뉴얼 페이지를 참조하십시오.

클라이언트 구성 파일의 키워드는 314 페이지 “Secure Shell의 키워드”에서 나열됩니다. 키워드에 기본값이 있을 경우 값이 제공됩니다. 해당 키워드는 ssh(1), scp(1), sftp(1) 및 ssh_config(4) 매뉴얼 페이지에서 자세히 설명됩니다. 영문자순 키워드 목록 및 동등한 명령줄 대체는 표 18–8을 참조하십시오.

Secure Shell의 서버 구성

Secure Shell 세션의 서버측 특성은 /etc/ssh/sshd_config 파일로 제어됩니다. 서버 구성 파일의 키워드는 314 페이지 “Secure Shell의 키워드”에서 나열됩니다. 키워드에 기본값이 있을 경우 값이 제공됩니다. 키워드에 대한 자세한 설명은 sshd_config(4) 매뉴얼 페이지를 참조하십시오.

Secure Shell의 키워드

다음 표에서는 키워드 및 해당 기본값(있을 경우)을 나열합니다. 키워드는 영문자순으로 표시됩니다. 클라이언트에 적용되는 키워드는 ssh_config 파일에 있으며, 서버에 적용되는 키워드는 sshd_config 파일에 있습니다. 두 파일에서 설정되는 키워드도 있습니다. v1 프로토콜을 실행 중인 Secure Shell 서버의 키워드는 표시됩니다.

표 18–1 Secure Shell 구성 파일의 키워드(A~Escape)

키워드	기본값	위치
AllowGroups	기본값 없음	서버
AllowTcpForwarding	yes	서버
AllowUsers	기본값 없음	서버
AuthorizedKeysFile	~/.ssh/authorized_keys	서버
Banner	/etc/issue	서버
Batchmode	no	클라이언트
BindAddress	기본값 없음	클라이언트
CheckHostIP	yes	클라이언트
ChrootDirectory	no	서버
Cipher	blowfish, 3des	클라이언트
Ciphers	aes128-ctr, aes128-cbc, 3des-cbc, blowfish-cbc, arcfour	모두
ClearAllForwardings	no	클라이언트
ClientAliveCountMax	3	서버

표 18–1 Secure Shell 구성 파일의 키워드(A~Escape) (계속)

키워드	기본값	위치
ClientAliveInterval	0	서버
Compression	no	모두
CompressionLevel	기본값 없음	클라이언트
ConnectionAttempts	1	클라이언트
ConnectTimeout	시스템 TCP 시간 초과	클라이언트
DenyGroups	기본값 없음	서버
DenyUsers	기본값 없음	서버
DisableBanner	no	클라이언트
DynamicForward	기본값 없음	클라이언트
EscapeChar	~	클라이언트

표 18–2 Secure Shell 구성 파일의 키워드(Fall~Local)

키워드	기본값	위치
FallBackToRsh	no	클라이언트
ForwardAgent	no	클라이언트
ForwardX11	no	클라이언트
ForwardX11Trusted	yes	클라이언트
GatewayPorts	no	모두
GlobalKnownHostsFile	/etc/ssh/ssh_known_hosts	클라이언트
GSSAPIAuthentication	yes	모두
GSSAPIDelegateCredentials	no	클라이언트
GSSAPIKeyExchange	yes	모두
GSSAPIStoreDelegateCredentials	yes	서버
HashKnownHosts	no	클라이언트
Host	* 자세한 내용은 318 페이지 “Secure Shell의 호스트 특정 매개변수”를 참조하십시오.	클라이언트
HostbasedAuthentication	no	모두
HostbasedUsesNameFromPacketOnly	no	서버

표 18–2 Secure Shell 구성 파일의 키워드(Fall~Local) (계속)

키워드	기본값	위치
HostKey	/etc/ssh/ssh_host_key	서버, v1
HostKey	/etc/ssh/host_rsa_key, /etc/ssh/host_dsa_key	서버
HostKeyAlgorithms	ssh-rsa, ssh-dss	클라이언트
HostKeyAlias	기본값 없음	클라이언트
HostName	기본값 없음	클라이언트
IdentityFile	~/.ssh/id_dsa, ~/.ssh/id_rsa	클라이언트
IgnoreIfUnknown	기본값 없음	클라이언트
IgnoreRhosts	yes	서버
IgnoreUserKnownHosts	yes	서버
KbdInteractiveAuthentication	yes	모두
KeepAlive	yes	모두
KeyRegenerationInterval	3600(초)	서버
ListenAddress	기본값 없음	서버
LocalForward	기본값 없음	클라이언트

표 18–3 Secure Shell 구성 파일의 키워드(Login~R)

키워드	기본값	위치
LoginGraceTime	120(초)	서버
LogLevel	info	모두
LookupClientHostnames	yes	서버
MACs	hmac-sha1,hmac-md5	모두
Match	기본값 없음	서버
MaxStartups	10:30:60	서버
NoHostAuthenticationForLocalHost	no	클라이언트
NumberOfPasswordPrompts	3	클라이언트
PAMServiceName	기본값 없음	서버
PAMServicePrefix	기본값 없음	서버

표 18–3 Secure Shell 구성 파일의 키워드(Login~R) *(계속)*

키워드	기본값	위치
PasswordAuthentication	yes	모두
PermitEmptyPasswords	no	서버
PermitRootLogin	no	서버
PermitUserEnvironment	no	서버
PidFile	/system/volatile/sshd.pid	서버
Port	22	모두
PreferredAuthentications	hostbased,publickey,keyboard-interactive,passwor	클라이언트
PreUserauthHook	기본값 없음	서버
PrintLastLog	yes	서버
PrintMotd	no	서버
Protocol	2,1	모두
ProxyCommand	기본값 없음	클라이언트
PubkeyAuthentication	yes	모두
RekeyLimit	1G~4G	클라이언트
RemoteForward	기본값 없음	클라이언트
RhostsAuthentication	no	서버, v1
RhostsRSAAuthentication	no	서버, v1
RSAAuthentication	no	서버, v1

표 18–4 Secure Shell 구성 파일의 키워드(S~X)

키워드	기본값	위치
ServerAliveCountMax	3	클라이언트
ServerAliveInterval	0	클라이언트
ServerKeyBits	512~768	서버, v1
StrictHostKeyChecking	ask	클라이언트
StrictModes	yes	서버
Subsystem	sftp /usr/lib/ssh/sftp-server	서버

표 18–4 Secure Shell 구성 파일의 키워드(S~X) (계속)

키워드	기본값	위치
SyslogFacility	auth	서버
UseOpenSSLEngine	yes	모두
UsePrivilegedPort	no	모두
User	기본값 없음	클라이언트
UserKnownHostsFile	~/.ssh/known_hosts	클라이언트
UseRsh	no	클라이언트
VerifyReverseMapping	no	서버
X11DisplayOffset	10	서버
X11Forwarding	yes	서버
X11UseLocalHost	yes	서버
XAuthLocation	/usr/openwin/bin/xauth	모두

Secure Shell의 호스트 특정 매개변수

로컬 호스트마다 Secure Shell 특성이 다른 것이 유용한 경우 관리자는 /etc/ssh/ssh_config 파일에서 호스트 또는 정규 표현식에 따라 적용할 별도의 매개변수 세트를 정의할 수 있습니다. 이 작업을 완료하려면 파일에서 Host 키워드로 항목을 그룹화하면 됩니다. Host 키워드를 사용하지 않을 경우 사용자가 작업 중인 로컬 호스트에 클라이언트 구성 파일의 항목이 적용됩니다.

Secure Shell 및 로그인 환경 변수

sshd_config 파일에 다음 Secure Shell 키워드가 설정되지 않은 경우 /etc/default/login 파일에서 동등한 항목의 값을 가져옵니다.

/etc/default/login의 항목	sshd_config의 키워드 및 값
CONSOLE=*	PermitRootLogin=without-password
#CONSOLE=*	PermitRootLogin=yes
PASSREQ=YES	PermitEmptyPasswords=no
PASSREQ=NO	PermitEmptyPasswords=yes
#PASSREQ	PermitEmptyPasswords=no

/etc/default/login의 항목	sshd_config의 키워드 및 값
TIMEOUT=*secs*	LoginGraceTime=*secs*
#TIMEOUT	LoginGraceTime=120
RETRIES 및 SYSLOG_FAILED_LOGINS	password 및 keyboard-interactive 인증 방법에만 적용됩니다.

사용자 로그인 셸의 초기화 스크립트에서 다음 변수가 설정되면 sshd 데몬에 해당 값이 사용됩니다. 변수가 설정되지 않으면 데몬에 기본값이 사용됩니다.

TIMEZONE TZ 환경 변수의 설정을 제어합니다. 설정되지 않은 경우 데몬이 시작되었으면 sshd 데몬에 TZ 값이 사용됩니다.

ALTSHELL SHELL 환경 변수의 설정을 제어합니다. 기본값은 ALTSHELL=YES이며, 이 경우 sshd 데몬에 사용자 셸의 값이 사용됩니다. ALTSHELL=NO인 경우 SHELL 값이 설정되지 않습니다.

PATH PATH 환경 변수의 설정을 제어합니다. 값이 설정되지 않은 경우 기본 경로는 /usr/bin입니다.

SUPATH root에 대한 PATH 환경 변수의 설정을 제어합니다. 값이 설정되지 않은 경우 기본 경로는 /usr/sbin:/usr/bin입니다.

자세한 내용은 login(1) 및 sshd(1M) 매뉴얼 페이지를 참조하십시오.

Secure Shell의 알려진 호스트 유지 관리

다른 호스트와 안전하게 통신해야 할 각 호스트의 로컬 호스트 /etc/ssh/ssh_known_hosts 파일에는 서버의 공개 키가 저장되어 있어야 합니다. 스크립트를 사용하여 /etc/ssh/ssh_known_hosts 파일을 업데이트할 수 있기는 하지만 스크립트는 중요한 보안 위험성에 노출되므로 이 방법은 사용하지 않는 것이 좋습니다.

/etc/ssh/ssh_known_hosts 파일은 다음과 같이 보안 방식을 통해서만 배포해야 합니다.

- 보안 연결(예: Secure Shell, IPsec 또는 알려진 신뢰할 수 있는 시스템의 Kerberos화된 ftp)을 통해 배포
- 시스템 설치 시 배포

침입자가 보거스 공개 키를 known_hosts 파일에 삽입하여 액세스 권한을 얻을 수 있는 가능성을 없애려면 ssh_known_hosts 파일의 알려진 신뢰할 수 있는 소스를 사용해야 합니다. ssh_known_hosts 파일은 설치 중 배포할 수 있습니다. 나중에 scp 명령을 사용하는 스크립트를 통해 최신 버전을 얻을 수 있습니다.

Secure Shell 파일

다음 표에서는 중요한 Secure Shell 파일 및 제안되는 파일 사용 권한을 보여 줍니다.

표 18–5 Secure Shell 파일

파일 이름	설명	제안되는 사용 권한 및 소유자
`/etc/ssh/sshd_config`	Secure Shell 데몬 `sshd`에 대한 구성 데이터를 포함합니다.	`-rw-r--r-- root`
`/etc/ssh/ssh_host_dsa_key` 또는 `/etc/ssh/ssh_host_rsa_key`	호스트 개인 키를 포함합니다.	`-rw------- root`
host-private-key`.pub`	호스트 공개 키(예: `/etc/ssh/ssh_host_rsa_key.pub`)를 포함합니다. 로컬 `known_hosts` 파일에 호스트 키를 복사하는 데 사용됩니다.	`-rw-r--r-- root`
`/system/volatile/sshd.pid`	Secure Shell 데몬 `sshd`의 프로세스 ID를 포함합니다. 여러 데몬이 실행 중인 경우 파일에 마지막으로 시작된 데몬이 포함됩니다.	`-rw-r--r-- root`
`~/.ssh/authorized_keys`	사용자 계정에 로그인할 수 있는 사용자의 공개 키를 보관합니다.	`-rw-r--r--` *username*
`/etc/ssh/ssh_known_hosts`	클라이언트가 안전하게 통신할 수 있는 모든 호스트의 호스트 공개 키를 포함합니다. 파일은 관리자가 채웁니다.	`-rw-r--r-- root`
`~/.ssh/known_hosts`	클라이언트가 안전하게 통신할 수 있는 모든 호스트의 호스트 공개 키를 포함합니다. 파일은 자동으로 유지 관리됩니다. 사용자가 알 수 없는 호스트를 사용하여 연결할 때마다 원격 호스트 키가 파일에 추가됩니다.	`-rw-r--r--` *username*
`/etc/default/login`	해당하는 `sshd_config` 매개변수가 설정되지 않은 경우 `sshd` 데몬에 대한 기본값을 제공합니다.	`-r--r--r-- root`
`/etc/nologin`	이 파일이 존재할 경우 `sshd` 데몬은 `root`만 로그인할 수 있도록 합니다. 로그인하려고 시도 중인 사용자에게 이 파일의 내용이 표시됩니다.	`-rw-r--r-- root`
`~/.rhosts`	사용자가 암호 없이 로그인할 수 있는 호스트를 지정하는 호스트/사용자 이름 쌍을 포함합니다. `rlogind` 및 `rshd` 데몬에도 이 파일이 사용됩니다.	`-rw-r--r--` *username*
`~/.shosts`	사용자가 암호 없이 로그인할 수 있는 호스트를 지정하는 호스트/사용자 이름 쌍을 포함합니다. 다른 유틸리티는 이 파일을 사용하지 않습니다. 자세한 내용은 `sshd(1M)` 매뉴얼 페이지의 `FILES` 절을 참조하십시오.	`-rw-r--r--` *username*
`/etc/hosts.equiv`	`.rhosts` 인증에서 사용되는 호스트를 포함합니다. `rlogind` 및 `rshd` 데몬에도 이 파일이 사용됩니다.	`-rw-r--r-- root`

표 18–5 Secure Shell 파일 *(계속)*

파일 이름	설명	제안되는 사용 권한 및 소유자
`/etc/ssh/shosts.equiv`	호스트 기반 인증에서 사용되는 호스트를 포함합니다. 다른 유틸리티는 이 파일을 사용하지 않습니다.	`-rw-r--r-- root`
`~/.ssh/environment`	로그인 시 초기 지정을 포함합니다. 기본적으로 이 파일은 읽히지 않습니다. 이 파일이 읽히도록 하려면 `sshd_config` 파일에서 `PermitUserEnvironment` 키워드를 `yes`로 설정해야 합니다.	`-rw-r--r--` *username*
`~/.ssh/rc`	사용자 셸이 시작되기 전에 실행되는 초기화 루틴을 포함합니다. 샘플 초기화 루틴은 `sshd`(1M) 매뉴얼 페이지를 참조하십시오.	`-rw-r--r--` *username*
`/etc/ssh/sshrc`	관리자가 지정한 호스트 특정 초기화 루틴을 포함합니다.	`-rw-r--r-- root`
`/etc/ssh/ssh_config`	클라이언트 시스템에서 시스템 설정을 구성합니다.	`-rw-r--r-- root`
`~/.ssh/config`	시스템 설정을 대체하는 사용자 설정을 구성합니다.	`-rw-r--r--` *username*

다음 표에서는 키워드 또는 명령 옵션이 대체할 수 있는 Secure Shell 파일을 나열합니다.

표 18–6 Secure Shell 파일 위치 대체

파일 이름	키워드 대체	명령줄 대체
`/etc/ssh/ssh_config`		`ssh -F` *config-file* `scp -F` *config-file*
`~/.ssh/config`		`ssh -F` *config-file*
`/etc/ssh/host_rsa_key` `/etc/ssh/host_dsa_key`	`HostKey`	
`~/.ssh/identity` `~/.ssh/id_dsa`, `~/.ssh/id_rsa`	`IdentityFile`	`ssh -i` *id-file* `scp -i` *id-file*
`~/.ssh/authorized_keys`	`AuthorizedKeysFile`	
`/etc/ssh/ssh_known_hosts`	`GlobalKnownHostsFile`	
`~/.ssh/known_hosts`	`UserKnownHostsFile` `IgnoreUserKnownHosts`	

Secure Shell 명령

다음 표에서는 주요 Secure Shell 명령을 요약합니다.

표 18–7 Secure Shell의 명령

명령 매뉴얼 페이지	설명
ssh(1)	사용자를 원격 시스템에 로그인하고 원격 시스템에서 안전하게 명령을 실행합니다. 이 명령은 `rlogin` 및 `rsh` 명령에 대한 Secure Shell 대체입니다. `ssh` 명령은 비보안 네트워크를 통해 신뢰할 수 없는 두 호스트 간에 암호화된 보안 통신을 가능하게 합니다. 또한 X11 연결 및 임의적 TCP/IP 포트는 보안 채널을 통해 전달될 수 있습니다.
sshd(1M)	Secure Shell에 대한 데몬입니다. 데몬은 클라이언트로부터의 연결을 수신 대기하며 비보안 네트워크를 통해 신뢰할 수 없는 두 호스트 간에 암호화된 보안 통신을 가능하게 합니다.
ssh-add(1)	인증 에이전트 `ssh-agent`에 RSA 또는 DSA ID를 추가합니다. ID를 **키**라고도 합니다.
ssh-agent(1)	공개 키 인증에 사용되는 개인 키를 보관합니다. `ssh-agent` 프로그램은 X-세션 또는 로그인 세션 시작 시 시작됩니다. 기타 모든 창 및 다른 프로그램은 `ssh-agent` 프로그램의 클라이언트로 시작됩니다. 환경 변수 사용을 통해 에이전트는 사용자가 `ssh` 명령을 사용하여 다른 시스템에 로그인할 때 배치되고 인증에 사용될 수 있습니다.
ssh-keygen(1)	Secure Shell에 대한 인증 키를 생성 및 관리합니다.
ssh-keyscan(1)	다양한 Secure Shell 호스트의 공개 키를 수집합니다. `ssh_known_hosts` 파일 작성 및 확인을 지원합니다.
ssh-keysign(1M)	`ssh` 명령이 로컬 호스트의 호스트 키에 액세스하는 데 사용합니다. Secure Shell v2를 통한 호스트 기반 인증 중 필요한 디지털 서명을 생성합니다. 사용자가 아닌 `ssh` 명령이 이 명령을 호출합니다.
scp(1)	암호화된 `ssh` 전송을 통해 네트워크의 호스트 간에 안전하게 파일을 복사합니다. `rcp` 명령과 달리 `scp` 명령은 인증에 암호 정보가 필요한 경우 암호 또는 암호문을 묻습니다.
sftp(1)	`ftp` 명령과 유사한 대화식 파일 전송 프로그램입니다. `ftp` 명령과 달리 `sftp` 명령은 암호화된 `ssh` 전송을 통해 모든 작업을 수행합니다. 명령은 연결 후 지정된 호스트 이름에 로그인하고 대화식 명령 모드를 시작합니다.

다음 표에서는 Secure Shell 키워드를 대체하는 명령 옵션을 나열합니다. 키워드는 `ssh_config` 및 `sshd_config` 파일에서 지정됩니다.

표 18–8 Secure Shell 키워드에 대한 명령줄 대체

키워드	ssh 명령줄 대체	scp 명령줄 대체
`BatchMode`		`scp -B`
`BindAddress`	`ssh -b` *bind-addr*	`scp -a` *bind-addr*
`Cipher`	`ssh -c` *cipher*	`scp -c` *cipher*
`Ciphers`	`ssh -c` *cipher-spec*	`scp -c` *cipher-spec*

표 18–8 Secure Shell 키워드에 대한 명령줄 대체 *(계속)*

키워드	ssh 명령줄 대체	scp 명령줄 대체
Compression	ssh -C	scp -C
DynamicForward	ssh -D *SOCKS4-port*	
EscapeChar	ssh -e *escape-char*	
ForwardAgent	ssh -A(사용) ssh -a(사용 안함)	
ForwardX11	ssh -X(사용) ssh -x(사용 안함)	
GatewayPorts	ssh -g	
IPv4	ssh -4	scp -4
IPv6	ssh -6	scp -6
LocalForward	ssh -L *localport:remotehost:remoteport*	
MACS	ssh -m *mac-spec*	
Port	ssh -p *port*	scp -P *port*
Protocol	ssh -2(v2 전용)	
RemoteForward	ssh -R *remoteport:localhost:localport*	

제 6 부

Kerberos 서비스

이 절에서는 다음 장에서 다루는 Kerberos 서비스의 구성, 관리 및 사용에 대한 정보를 제공합니다.

- 19 장, “Kerberos 서비스 소개”
- 20 장, “Kerberos 서비스 계획”
- 21 장, “Kerberos 서비스 구성(작업)”
- 22 장, “Kerberos 오류 메시지 및 문제 해결”
- 23 장, “Kerberos 주체 및 정책 관리(작업)”
- 24 장, “Kerberos 응용 프로그램 사용(작업)”
- 25 장, “Kerberos 서비스(참조)”

◆ ◆ ◆ 19장

19 Kerberos 서비스 소개

이 장에서는 Kerberos 서비스에 대해 소개합니다. 다음은 이 장에 포함된 개요 정보 목록입니다.

Kerberos 서비스란?

Kerberos **서비스**는 네트워크를 통해 보안 트랜잭션을 제공하는 클라이언트-서버 구조입니다. 이 서비스는 무결성 및 프라이버시를 비롯하여 강력한 사용자 인증을 제공합니다. **인증**은 네트워크 트랜잭션의 송신인과 수신자가 맞는지 보증합니다. 이 서비스는 또한 앞뒤로 전달되는 데이터의 유효성을 확인(**무결성**)하고 전송 중 데이터를 암호화합니다(**프라이버시**). Kerberos 서비스를 사용할 경우 다른 시스템에 로그인, 명령 실행, 데이터 교환 및 안전한 파일 전송 등이 가능합니다. 또한 이 서비스는 관리자가 서비스 및 시스템에 대한 액세스를 제한할 수 있도록 해주는 **인증** 서비스를 제공합니다. 또한 Kerberos 사용자는 다른 사용자가 자신의 계정에 액세스하는 것을 규제할 수 있습니다.

Kerberos 서비스는 **단일 사인 온***(SSO)* 시스템이므로, 세션당 한 번만 서비스에 대해 자신을 인증하기만 하면 됩니다. 그러면 세션 동안의 이후 모든 트랜잭션이 자동으로 보안 처리됩니다. 서비스에 대해 인증을 받은 후에는 Kerberos 기반 명령(예: `ftp` 또는 `rsh`)을 사용할 때마다 인증하거나, NFS 파일 시스템의 데이터에 액세스할 필요가 없습니다. 따라서 네트워크를 통해 암호를 전송할 경우 암호가 인터셉트될 가능성이 있는데, 이 서비스를 사용할 때마다 이러한 방식으로 암호를 전송할 필요가 없습니다.

Oracle Solaris 릴리스의 Kerberos 서비스는 MIT(Massachusetts Institute of Technology)에서 개발한 Kerberos V5 네트워크 인증 프로토콜을 기반으로 합니다. Kerberos V5 제품을 사용해 본 적이 있는 사용자라면 Oracle Solaris 버전을 매우 친숙하게 찾을 수 있을 것입니다. Kerberos V5 프로토콜은 **사실상** 네트워크 보안을 위한 산업 표준이기 때문에

Oracle Solaris 버전에서는 다른 시스템과의 상호 운용성이 향상됩니다. 즉, Oracle Solaris 릴리스의 Kerberos 서비스는 Kerberos V5 프로토콜을 사용하는 시스템에서 작동하기 때문에 이기종 네트워크를 통해서도 보안 트랜잭션이 가능합니다. 또한 이 서비스는 도메인 간에 그리고 단일 서비스 내에서 인증과 보안을 제공합니다.

Kerberos 서비스는 Oracle Solaris 응용 프로그램을 실행하는 데 있어 유연성을 제공합니다. 네트워크 서비스(예: NFS 서비스, `telnet`, `ftp`)에 대해 Kerberos 기반 요청 및 Kerberos 기반이 아닌 요청을 모두 허용하도록 서비스를 구성할 수 있습니다. 그러면 Kerberos 서비스가 사용으로 설정되지 않은 시스템에서 현재 응용 프로그램이 실행 중이더라도 계속 작동합니다. 물론 Kerberos 기반 네트워크만 허용하도록 Kerberos 서비스를 구성할 수도 있습니다.

Kerberos 서비스가 제공하는 보안 방식은 GSS-API(Generic Security Service Application Programming Interface)를 사용하는 응용 프로그램을 사용할 경우 인증, 무결성 및 프라이버시를 위해 Kerberos 사용을 허용합니다. 그러나 다른 보안 방식이 개발된 경우 응용 프로그램이 Kerberos 서비스를 계속 사용할 필요는 없습니다. 이 서비스는 모듈 방식으로 GSS-API에 통합되었으므로 GSS-API를 사용하는 응용 프로그램이 필요에 가장 적합한 보안 방식을 사용할 수 있습니다.

Kerberos 서비스의 작동 방식

다음은 Kerberos 인증 시스템에 대한 개요입니다. 자세한 설명은 510 페이지 “Kerberos 인증 시스템의 작동 방식”을 참조하십시오.

사용자의 관점에서 Kerberos 서비스는 Kerberos 세션이 시작되면 대개 눈에 띄지 않습니다. `rsh` 또는 `ftp` 등의 명령도 동일하게 작동합니다. Kerberos 세션을 초기화하면 더 이상 로그인하거나 Kerberos 암호를 제공할 필요가 없습니다.

Kerberos 시스템은 **티켓**이라는 개념에 중점을 둡니다. 티켓은 사용자나 서비스(예: NFS 서비스)를 식별하는 전자 정보 집합입니다. 운전 면허증이 사용자의 신원을 확인해 주고 소지하고 있는 운전 면허를 나타내듯이, 티켓은 사용자와 사용자의 네트워크 액세스 권한을 식별합니다. Kerberos 기반 트랜잭션을 수행하는 경우(예: 다른 시스템에 원격 로그인하는 경우) 티켓에 대한 요청이 투명하게 KDC(**키 배포 센터**)로 전송됩니다. KDC는 데이터베이스에 액세스하여 사용자의 신원을 인증하고 다른 시스템에 액세스할 수 있는 권한을 부여하는 티켓을 반환합니다. “투명하게”란 명시적으로 티켓을 요청할 필요가 없음을 의미합니다. 요청은 `rlogin` 명령의 일부로 수행됩니다. 인증된 클라이언트만 특정 서비스에 대한 티켓을 가져올 수 있으므로 사용 중인 ID로는 다른 클라이언트가 `rlogin`을 사용할 수 없습니다.

티켓은 특정 속성과 연관됩니다. 예를 들어 티켓이 **전달 가능** 티켓일 수 있습니다. 이는 새 인증 프로세스 없이도 다른 시스템에서 티켓을 사용할 수 있음을 의미합니다. 또한 **후일자** 티켓일 수도 있습니다. 이는 지정된 시간까지 티켓이 유효하지 않음을 의미합니다. 예를 들어 티켓을 어떻게 사용하여 어떤 사용자가 어떤 유형의 티켓을 얻을 수 있는지는 **정책**에 의해 설정됩니다. 정책은 Kerberos 서비스가 설치되거나 관리될 때 결정됩니다.

주 – 자격 증명과 **티켓**이라는 용어를 자주 접하게 될 것입니다. 보다 넓은 Kerberos 관점에서 이 둘은 서로 바꿔 사용할 수 있습니다. 그러나 기술적인 측면에서 자격 증명은 티켓과 해당 세션에 대한 **세션 키**가 추가된 것입니다. 이 차이는 510 페이지 “Kerberos를 사용하여 서비스에 대한 액세스 권한 얻기”에 더 자세히 설명되어 있습니다.

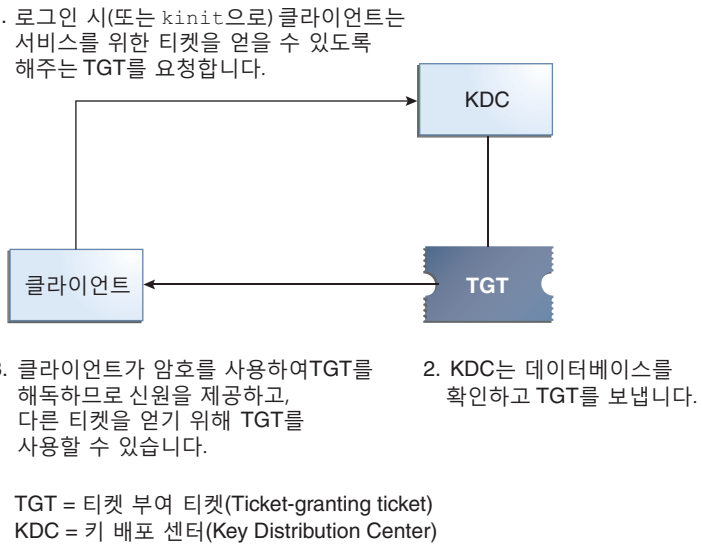
다음 절에서는 Kerberos 인증에 대해 추가로 설명합니다.

초기 인증: TGT(티켓 부여 티켓)

Kerberos 인증은 모든 후속 인증에 대해 허용되는 초기 인증과 후속 인증 자체의 두 단계로 구성됩니다.

다음 그림은 초기 인증이 이루어지는 방식을 보여줍니다.

그림 19–1 Kerberos 세션에 대한 초기 인증



1. 클라이언트(사용자 또는 NFS 등의 서비스)는 KDC(키 배포 센터)에서 TGT(**티켓 부여 티켓**)를 요청하여 Kerberos 세션을 시작합니다. 이 요청은 대개 로그인 시 자동으로 수행됩니다.

TGT(티켓 부여 티켓)는 특정 서비스의 다른 티켓을 얻는 데 필요합니다. TGT(티켓 부여 티켓)는 여권과 비슷하다고 생각하십시오. TGT(티켓 부여 티켓)는 여권처럼 사용자의 신원을 확인하며 여러 개의 "비자"를 얻을 수 있도록 해줍니다. 여기서 "비자"(티켓)는 외국에 사용하기 위한 것이 아니라 원격 시스템 또는 네트워크 서비스에 사용하기 위한 것입니다. 여권과 비자처럼 TGT(티켓 부여 티켓) 및 기타 여러 티켓의 수명은 제한되어 있습니다. 차이점이라면 "Kerberos화된" 명령은 사용자에게 여권이 있음을 알고 있어 자동으로 비자를 얻는다는 것입니다. 즉, 사용자가 트랜잭션을 직접 수행할 필요가 없습니다.

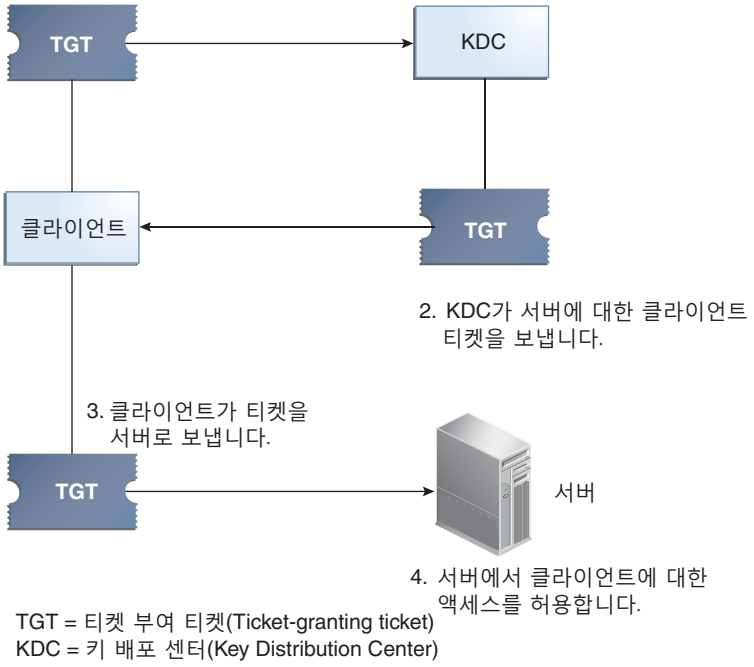
TGT(티켓 부여 티켓)가 네 곳의 스키 리조트에서 3일 동안 사용할 수 있는 스키 패스라고 생각해 볼 수도 있습니다. 방문하려는 리조트의 패스를 보여 주면 패스 유효 기간 동안에는 해당 리조트의 리프트 티켓을 받을 수 있습니다. 리프트 티켓이 있으면 해당 리조트에서 원하는 만큼 스키를 탈 수 있습니다. 다음 날 다른 리조트를 방문한 경우 다시 패스를 보여 주면 새 리조트의 리프트 티켓을 추가로 얻을 수 있습니다. 차이점이라면 Kerberos 기반 명령은 사용자에게 주말용 스키 패스가 있음을 알고 있어 자동으로 리프트 티켓을 얻는다는 점입니다. 따라서 사용자가 트랜잭션을 직접 수행할 필요가 없습니다.

2. KDC는 TGT(티켓 부여 티켓)를 만들어 암호화된 형태로 다시 클라이언트로 보냅니다. 클라이언트는 클라이언트의 암호를 사용하여 TGT(티켓 부여 티켓)를 해독합니다.
3. 이제 유효한 TGT(티켓 부여 티켓)가 생겼으므로, 클라이언트에서는 TGT(티켓 부여 티켓)가 유효한 동안 모든 종류의 네트워크 작업(예: `rlogin` 또는 `telnet`)에 대한 티켓을 요청할 수 있습니다. 이 티켓은 보통 몇 시간 동안만 지속됩니다. 클라이언트에서는 고유 네트워크 작업을 수행할 때마다 KDC로부터 해당 작업에 대한 티켓을 요청합니다.

후속 Kerberos 인증

클라이언트가 초기 인증을 받은 후 후속 인증이 수행될 때마다 다음 그림에 표시된 패턴을 따릅니다.

그림 19–2 Kerberos 인증을 사용하여 서비스에 대한 액세스 권한 얻기



1. 클라이언트는 KDC에 신원 증명서로 TGT(티켓 부여 티켓)를 전송하여 KDC로부터 다른 시스템에 대한 원격 로그인용 특정 서비스에 대한 티켓을 요청합니다.
2. KDC는 특정 서비스에 대한 티켓을 만들어 클라이언트로 보냅니다.

   예를 들어 `joe`라는 사용자가 필요한 `krb5` 인증과 공유된 NFS 파일 시스템에 액세스하려 한다고 가정합시다. 이 사용자는 이미 인증을 받았기 때문에 즉, 이미 TGT(티켓 부여 티켓)가 있으므로 파일에 액세스하려고 하면 NFS 클라이언트 시스템이 자동으로 그리고 투명하게 KDC로부터 NFS 서비스에 대한 티켓을 얻습니다.

   예를 들어 `joe`라는 사용자가 `boston` 서버에서 `rlogin`을 사용한다고 가정합시다. 이 사용자는 이미 인증을 받았기 때문에 즉, 이미 TGT(티켓 부여 티켓)가 있으므로 자동으로 그리고 투명하게 `rlogin` 명령의 일부로 티켓을 얻습니다. 이 티켓이 있으면 만료될 때까지 원하는 만큼 `boston`에 원격으로 로그인할 수 있습니다. `joe`가 `denver` 시스템에 원격 로그인하고 싶은 경우 1단계에서처럼 또 다른 티켓을 얻습니다.
3. 클라이언트가 티켓을 서버로 보냅니다.

NFS 서비스를 사용할 경우 NFS 클라이언트가 자동으로 그리고 투명하게 NFS 서비스에 대한 티켓을 NFS 서버로 보냅니다.

4. 서버에서 클라이언트 액세스를 허용합니다.

이러한 단계에서 서버는 KDC와 통신하지 않습니다. 그렇지만 첫번째 클라이언트와 마찬가지로 서버 자체는 KDC에 등록되어 있습니다. 간단히 나타내기 위해 이 부분은 생략했습니다.

Kerberos 원격 응용 프로그램

joe와 같은 사용자가 사용할 수 있는 Kerberos 기반(또는 "Kerberos화된") 명령은 다음과 같습니다.

- `ftp`
- `rcp`
- `rlogin`
- `rsh`
- `ssh`
- `telnet`

이러한 응용 프로그램은 같은 이름의 Solaris 응용 프로그램과 같습니다. 그러나 Kerberos 주체를 사용하여 트랜잭션을 인증하도록 확장되었으므로 Kerberos 기반 보안을 제공합니다. 주체에 대한 자세한 내용은 332 페이지 "Kerberos 주체"를 참조하십시오.

이러한 명령은 494 페이지 "Kerberos 사용자 명령"에 자세히 설명되어 있습니다.

Kerberos 주체

Kerberos 서비스의 클라이언트는 **주체**로 식별됩니다. 주체는 KDC가 티켓을 지정할 수 있는 고유 ID입니다. 주체는 사용자(예: `joe`) 또는 서비스(예: `nfs` 또는 `telnet`)일 수 있습니다.

관례상 주체 이름은 **기본 요소**, **인스턴스**, **영역**이라는 세 개의 구성 요소로 구분됩니다. 예를 들면 일반적인 Kerberos 주체는 `joe/admin@ENG.EXAMPLE.COM`입니다. 위 예에서 각 요소의 역할은 다음과 같습니다.

- `joe`는 기본 요소입니다. 기본 요소는 여기에 표시된 사용자 이름 또는 서비스(예: `nfs`)일 수 있습니다. 기본 요소는 `host`라는 단어일 수도 있습니다. 이 단어는 해당 주체가 다양한 네트워크 서비스 `ftp`, `rcp`, `rlogin` 등을 제공하도록 설정된 서비스 주체임을 의미합니다.
- `admin`은 인스턴스입니다. 인스턴스는 사용자 주체의 경우 선택적이지만, 서비스 주체의 경우에는 필수입니다. 예를 들어 사용자 `joe`가 가끔 시스템 관리자 역할을 수행하는 경우 `joe/admin`을 사용하여 자신과 자신의 일반적인 사용자 ID를 구별할 수 있습니다. 마찬가지로, `joe`에게 서로 다른 두 호스트에 대한 계정이 있을 경우 `joe/denver.example.com` 및 `joe/boston.example.com`과 같이 서로 다른 인스턴스를 사용하는 두 개의 주체 이름을 사용할 수 있습니다. Kerberos 서비스는 `joe`와 `joe/admin`을 완전히 다른 두 주체로 취급합니다.

  서비스 주체의 경우 인스턴스는 정규화된 호스트 이름입니다. `bigmachine.eng.example.com`은 이러한 인스턴스의 예입니다. 이 예에서 기본 요소/인스턴스는 `ftp/bigmachine.eng.example.com` 또는 `host/bigmachine.eng.example.com`일 수 있습니다.
- `ENG.EXAMPLE.COM`은 Kerberos 영역입니다. 영역은 333 페이지 “Kerberos 영역”에 설명되어 있습니다.

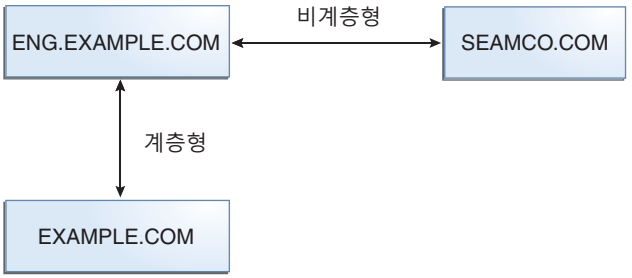
다음은 모두 유효한 주체 이름입니다.

- `joe`
- `joe/admin`
- `joe/admin@ENG.EXAMPLE.COM`
- `nfs/host.eng.example.com@ENG.EXAMPLE.COM`
- `host/eng.example.com@ENG.EXAMPLE.COM`

Kerberos 영역

영역은 도메인과 유사한 논리적 그룹으로, 동일한 **마스터** *KDC* 아래에 있는 시스템 그룹을 정의합니다. 그림 19–3은 영역 간의 관계를 보여줍니다. 일부 영역은 계층형으로, 한 영역이 다른 영역의 수퍼 세트입니다. 또 다른 영역은 비계층형(또는 “직접”)으로, 두 영역 간의 매핑을 정의해야 합니다. Kerberos 서비스의 특징은 영역 간 인증을 허용한다는 점입니다. 각 영역에는 KDC에 있는 다른 영역에 대한 주체 항목만 있으면 됩니다. 이 Kerberos 기능을 **영역 간 인증**이라고 합니다.

그림 19–3 Kerberos 영역



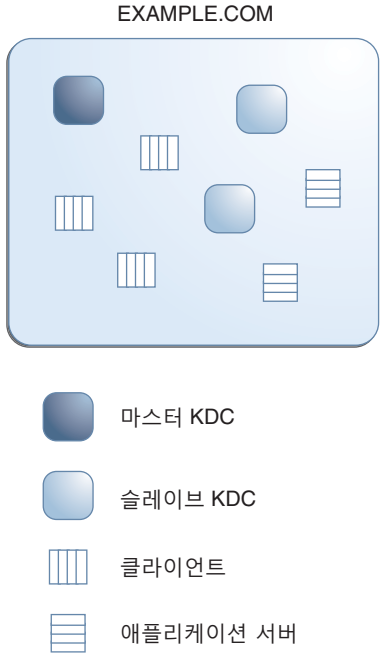
Kerberos 서버

각 영역에는 주체 데이터베이스의 마스터 복사본을 유지 관리하는 서버가 있어야 합니다. 이 서버를 **마스터** *KDC* **서버**라고 합니다. 또한 각 영역에는 주체 데이터베이스의 복제 복사본을 포함하는 **슬레이브** *KDC* **서버**도 한 개 이상 있어야 합니다. 마스터 KDC 서버와 슬레이브 KDC 서버 모두 인증을 설정하는 데 사용되는 티켓을 만듭니다.

영역에는 Kerberos **애플리케이션 서버**가 포함될 수도 있습니다. 이 서버는 Kerberos화된 서비스(예: `ftp`, `telnet`, `rsh` 및 NFS)에 대한 액세스를 제공합니다. SEAM 1.0 또는 1.0.1을 설치한 경우 영역에 Kerberos 네트워크 애플리케이션 서버가 포함되어 있을 수 있지만, 이 소프트웨어는 이 릴리스와 함께 제공되지 않습니다.

다음 그림은 가상 영역에 포함될 수 있는 요소를 보여줍니다.

그림 19–4 일반 Kerberos 영역



Kerberos 보안 서비스

Kerberos 서비스는 사용자 보안 인증 이외에도 두 가지 보안 서비스를 제공합니다.

- **무결성** – 인증이 네트워크에 있는 클라이언트가 해당 권한을 가진 대상인지 확인하는 것과 마찬가지로, 무결성은 클라이언트가 보내는 데이터가 유효하며 전송 중 변경되지 않았는지 확인합니다. 무결성은 데이터의 암호화된 체크섬을 통해 수행됩니다. 무결성에도 사용자 인증이 포함됩니다.
- **프라이버시** – 프라이버시는 보안을 한층 더 강화합니다. 프라이버시는 전송된 데이터의 무결성을 확인할 뿐 아니라 전송하기 전에 데이터를 암호화하여 도청자로부터 데이터를 보호합니다. 프라이버시도 사용자를 인증합니다.

개발자의 경우 RPCSEC_GSS 프로그래밍 인터페이스를 사용하여 보안 서비스를 선택하도록 RPC 기반 응용 프로그램을 설계할 수 있습니다.

여러 Kerberos 릴리스의 구성 요소

Kerberos 서비스의 구성 요소는 여러 릴리스에 포함되어 왔습니다. 처음에는 Kerberos 서비스 및 Kerberos 서비스를 지원하기 위한 기본 운영 체제 변경 사항이 "Sun Enterprise Authentication Mechanism"(줄여서 SEAM)이라는 제품 이름으로 릴리스되었습니다. Oracle Solaris 소프트웨어에 포함되는 SEAM 제품의 구성 요소가 많아질수록 SEAM 릴리스의 컨텐츠는 줄었습니다. Oracle Solaris 10 릴리스부터는 SEAM 제품의 모든 구성 요소가 포함되어 더 이상 SEAM 제품이 필요하지 않게 되었습니다. SEAM 제품 이름은 기록상의 이유로 설명서에 포함되어 있습니다.

다음 표는 각 릴리스에 포함된 구성 요소에 대해 설명합니다. 각 제품 릴리스는 연대순으로 나열됩니다. 모든 구성 요소는 다음 절에 설명되어 있습니다.

표 19–1 Kerberos 릴리스 컨텐츠

릴리스 이름	내용
SEAM 1.0(Solaris Easy Access Server 3.0)	Solaris 2.6 및 7 릴리스용 Kerberos 서비스의 전체 릴리스
Kerberos 서비스(Solaris 8 릴리스)	Kerberos 클라이언트 소프트웨어만
SEAM 1.0.1(Solaris 8 Admin Pack)	Solaris 8 릴리스용 Kerberos KDC 및 원격 응용 프로그램
Kerberos 서비스(Solaris 9 릴리스)	Kerberos KDC 및 클라이언트 소프트웨어만
SEAM 1.0.2	Solaris 9 릴리스용 Kerberos 원격 응용 프로그램
Kerberos 서비스(Oracle Solaris 10 릴리스 이상)	향상된 기능이 포함된 Kerberos 서비스의 전체 릴리스

Oracle Solaris 10 릴리스에 포함된 향상된 기능에 대한 자세한 내용은 Kerberos 구성 요소를 참조하십시오.

Kerberos 구성 요소

Kerberos V5 제품의 MIT 배포판과 마찬가지로, Oracle Solaris 릴리스의 Kerberos 서비스는 다음으로 구성됩니다.

- KDC(키 배포 센터)
  - Kerberos 데이터베이스 관리 데몬 – `kadmind`.
  - Kerberos 티켓 처리 데몬 – `krb5kdc`.
  - 데이터베이스 관리 프로그램 – `kadmin`(마스터 전용), `kadmin.local` 및 `kdb5_util`
  - 데이터베이스 전파 소프트웨어 – `kprop`(슬레이브 전용) 및 `kpropd`.
- 자격 증명 관리를 위한 사용자 프로그램 – `kinit`, `klist` 및 `kdestroy`
- Kerberos 암호를 변경하기 위한 사용자 프로그램 – `kpasswd`.

- 원격 응용 프로그램 – `ftp`, `rcp`, `rlogin`, `rsh`, `ssh` 및 `telnet`
- 원격 응용 프로그램 데몬 – `ftpd`, `rlogind`, `rshd`, `sshd` 및 `telnetd`
- Keytab 관리 유틸리티 – `ktutil`
- GSS-API(Generic Security Service Application Programming Interface) – 새 방식이 추가될 때마다 응용 프로그램을 재컴파일할 필요 없이 응용 프로그램에서 여러 개의 보안 방식을 사용할 수 있도록 해줍니다. GSS-API는 응용 프로그램을 여러 운영 체제에 이식 가능하게 해주는 표준 인터페이스를 사용합니다. 또한 응용 프로그램에 인증을 비롯한 무결성 및 프라이버시 보안 서비스가 포함되도록 해줍니다. `ftp` 및 `ssh`는 GSS-API를 사용합니다.
- RPCSEC_GSS API(Application Programming Interface) – NFS 서비스가 Kerberos 인증을 사용할 수 있도록 해줍니다. RPCSEC_GSS는 사용 중인 방식과 관계없이 보안 서비스를 제공하는 보안 종류입니다. RPCSEC_GSS는 GSS-API 계층을 기반으로 합니다. 플러그 가능 GSS_API 기반 보안 방식은 RPCSEC_GSS를 사용하는 응용 프로그램에서 사용할 수 있습니다.

또한 Oracle Solaris 릴리스의 Kerberos 서비스에는 다음 항목도 포함됩니다.

- Kerberos 관리 GUI 기반 도구(`gkadmin`) – 주체 및 주체 정책을 관리할 수 있도록 해줍니다. 이 Java 기술 기반의 GUI는 `kadmin` 명령 대신 사용됩니다.
- PAM용 Kerberos V5 서비스 모듈 – Kerberos 서비스에 대해 인증, 계정 관리, 세션 관리 및 암호 관리를 제공합니다. 이 모듈을 사용하면 Kerberos 인증이 사용자에게 투명하게 수행될 수 있습니다.
- 커널 모듈 – NFS 서비스에서 사용하도록 Kerberos 서비스의 커널 기반 구현을 제공하므로 성능이 크게 향상됩니다.

Oracle Solaris 11 릴리스의 Kerberos 정보

이 절에서는 Oracle Solaris 11 릴리스에서 제공하는 변경 사항에 대해 설명합니다.

- Kerberos 서비스는 MIT 1.8 릴리스와 동기화되었습니다. 포함된 기능은 다음과 같습니다.
  - 취약한 암호화 유형인 `arcfour-hmac-md5-exp`, `des-cbc-md5` 및 `des-cbc-crc`는 기본적으로 허용되지 않습니다. 그러나 `/etc/krb5/krb5.conf` 파일의 `allow_weak_crypto = true` 선언을 추가하면 취약한 암호화 알고리즘을 사용할 수 있습니다.
  - `/etc/krb5/krb5.conf` 파일에서 `permitted_enctypes` 관계는 선택적 `DEFAULT` 키워드를 + 또는 – `enctyp_family`와 함께 사용하여 기본 집합에서 지정된 암호화 유형 제거를 추가할 수 있습니다.
  - 대부분의 경우 클라이언트 측에서 `domain_realm` 매핑 테이블에 대한 필요를 제거할 수 있는데, 이를 위해서는 KDC에서 최소 참조 지원을 구현한 다음 해당 프로토콜을 통해 매핑 정보를 클라이언트에 제공하십시오. 서비스 주체 `name service/canonical-fqdn@LOCAL.REALM`에 대한 요청을 로컬 KDC로 보낸 다음

참조를 요청하면 클라이언트가 `domain_realm` 매핑 테이블 없이도 작동할 수 있습니다. 이 기능은 특정 이름 유형 또는 특정 형식을 사용하는 서비스 주체 이름으로 제한됩니다. KDC에서는 `domain_realm` 매핑 테이블만 사용할 수 있습니다. DNS에 대한 블록화 쿼리를 사용할 수 없습니다.

- Kerberos 데이터베이스에 LDAP 백엔드를 사용 중인 경우 주체 항목에 대한 별칭을 만들 수 있습니다. 서비스에 다른 호스트 이름으로 액세스할 수 있는 경우 또는 DNS를 사용하여 트 이름을 정규화할 수 없는 경우 즉, 간략한 형식을 사용하는 경우에 주체 별칭 지원이 유용합니다. 서비스가 알려진 여러 주체 이름에 대한 별칭을 사용할 수 있으므로, keytab 파일에는 실제 서비스 주체에 대한 하나의 키 집합만 필요합니다.
- `kvno` 유틸리티를 사용하여 `/etc/krb5/krb5.keytab`에 지정된 서비스 주체 키와 관련된 문제를 진단할 수 있습니다.
- `kadmin ktadd` 명령은 `-norandkey` 옵션을 지원하여 `kadmind` 명령이 난수화된 새 키를 만들지 못하도록 합니다. `-norandkey` 옵션은 암호 파생 키가 있는 주체에 대한 keytab을 만들려는 경우에 유용합니다. 따라서 암호를 지정할 필요 없이 `kinit` 명령을 실행하는 데 사용할 수 있는 keytab을 만들 수 있습니다.
- 지정된 시간 제한 내에 사전 인증이 특정 횟수만큼 실패할 경우 주체가 잠길 수 있습니다. 자세한 내용은 400 페이지 “계정 잠금 구성 방법”을 참조하십시오.
- `OK_AS_DELEGATE` 플래그는 위임된 자격 증명을 수락할 정도로 중간 서버를 신뢰하는지 여부와 관련하여 KDC가 로컬 영역 정책을 클라이언트로 전달할 수 있도록 해줍니다. 자세한 내용은 348 페이지 “위임을 위해 서비스 신뢰”를 참조하십시오.

- Kerberos에 대한 통계 지점을 통계적으로 정의한 사용자 레벨 집합이 추가되었습니다. 이러한 검사는 Kerberos 프로토콜 메시지에 대한 논리적 뷰를 제공합니다. 예제를 보려면 444 페이지 “Kerberos 서비스에서 DTrace 사용”을 참조하십시오.
- `kclient` 스크립트가 향상되었습니다. 이 스크립트에는 Microsoft Active Directory 서버 조인 기능이 포함됩니다. 지침은 386 페이지 “대화식으로 Kerberos 클라이언트를 구성하는 방법” 및 389 페이지 “Active Directory 서버에 대한 Kerberos 클라이언트 구성 방법”을 참조하십시오. 또한 이 스크립트에는 클라이언트에 대한 KDC 서버를 식별하는 데 사용할 수 있는 `-T` 옵션이 포함됩니다. 이 스크립트에 대한 모든 옵션은 `kclient(1M)` 매뉴얼 페이지에 설명되어 있습니다.
- `/etc/krb5/kadm5.keytab` 파일은 더 이상 필요하지 않습니다. 따라서 이제 이 파일에 저장된 키는 Kerberos 데이터베이스에서 직접 읽을 수 없습니다.
- 디렉토리 서버에서 LDAP을 사용하여 Kerberos 주체 및 정책 레코드에 액세스하기 위한 지원이 추가되었습니다. 이러한 변경은 관리를 간소화하고 KDC 및 디렉토리 서버의 배치에 따라 더 향상된 가용성을 제공합니다. LDAP 관련 절차 목록은 422 페이지 “LDAP 디렉토리 서버에서 KDC 관리”를 참조하십시오.

- 새 `kdcmgr` 명령은 KDC를 자동으로 또는 대화식으로 설정하는 데 사용됩니다. 이 명령으로 마스터 및 슬레이브 KDC 서버가 생성됩니다. 또한 `kdcmgr` 명령을 `status` 옵션과 함께 사용할 경우 로컬 호스트에 설치된 KDC에 대한 정보가 표시됩니다. 표 21–1에서 자동 및 대화식 절차에 대한 포인터를 참조하십시오.
- 추가 설정이 필요하지 않은 Oracle Solaris 클라이언트에 대한 지원이 이 릴리스에 추가되었습니다. Kerberos 서비스 및 일부 기본값이 변경되었습니다. Kerberos 클라이언트는 적절하게 구성된 환경에서 클라이언트 측 구성 없이 작동합니다. 자세한 내용은 346 페이지 “클라이언트 구성 옵션”을 참조하십시오.

◆ ◆ ◆ 20장

20 Kerberos 서비스 계획

이 장은 Kerberos 서비스의 설치와 유지 관리를 담당하는 관리자가 검토해야 합니다. 이 장에서는 관리자가 서비스를 설치하거나 구성하기 전에 해결해야 하는 여러 설치 및 구성 옵션에 대해 설명합니다.

다음은 시스템 관리자 또는 기타 지식을 갖춘 지원 인력이 학습해야 하는 항목입니다.

Kerberos 배치를 계획하는 이유

Kerberos 서비스를 설치하기 전에 몇 가지 구성 문제를 해결해야 합니다. 처음 설치한 후 구성을 변경할 수는 있지만 변경할 경우 구현하기 힘들 수 있습니다. 또한 약간이라도 변경할 경우 KDC를 재구성해야 하므로, Kerberos 구성을 계획할 경우 장기 목표를 고려하는 것이 더 좋습니다.

Kerberos 기반구조를 배치하는 데는 KDC 설치, 호스트 키 만들기, 사용자 마이그레이션과 같은 작업이 필요합니다. Kerberos 배치를 재구성하는 것은 초기 배치를 수행하는 것만큼 어려우므로, 재구성해야 하는 경우가 발생하지 않도록 신중하게 배치를 계획하십시오.

Kerberos 영역 계획

영역은 도메인과 유사한 논리적 그룹으로, 동일한 마스터 KDC 아래에 있는 시스템 그룹을 정의합니다. DNS 도메인 이름을 설정하는 경우와 마찬가지로, Kerberos 서비스를 구성하기 전에 영역 이름, 각 영역의 개수 및 크기, 영역 간 인증을 위한 다른 영역과의 관계와 같은 문제를 해결해야 합니다.

영역 이름

영역 이름은 ASCII 문자열로 구성될 수 있습니다. 보통 영역 이름이 대문자인 경우를 제외하고는 DNS 도메인 이름과 같습니다. 이 규칙은 친숙한 이름을 사용하면서 Kerberos 서비스 관련 문제와 DNS 이름 공간 관련 문제를 구별하는 데 도움이 됩니다. DNS를 사용하지 않거나 다른 문자를 사용하도록 선택한 경우 원하는 문자열을 사용할 수 있습니다. 그러나 구성 프로세스에 더 많은 작업이 필요합니다. 표준 인터넷 명명 구조를 준수하는 영역 이름을 사용하는 것이 좋습니다.

영역 수

설치에 필요한 영역 수는 다음과 같은 몇 가지 요인에 따라 달라집니다.

- 지원될 클라이언트 수. 한 영역에 클라이언트가 너무 많이 있을 경우 관리하기 어려워지므로 결과적으로 영역을 분할해야 합니다. 지원 가능한 클라이언트 수를 결정하는 주요 요인은 다음과 같습니다.
  - 각 클라이언트가 생성하는 Kerberos 양
  - 물리적 네트워크의 대역폭
  - 호스트 속도

  설치 시마다 제한 사항이 달라지므로 최대 클라이언트 수를 결정하는 규칙은 없습니다.
- 클라이언트가 떨어져 있는 거리. 클라이언트가 서로 다른 지역에 있는 경우 여러 개의 작은 영역을 설정할 수 있습니다.
- KDC로 설치할 수 있는 호스트 수. 각 영역에는 마스터 서버와 슬레이브 서버 각각 하나씩, 최소 두 개의 KDC 서버가 있어야 합니다.

관리 도메인과 Kerberos 영역을 조정하는 것이 좋습니다. Kerberos V 영역은 영역이 대응되는 DNS 도메인의 여러 하위 도메인으로 확장될 수 있습니다.

영역 계층 구조

영역 간 인증을 위해 여러 개의 영역을 구성하는 경우 영역을 서로 연결하는 방법을 결정해야 합니다. 영역 간에 계층 관계를 설정하여 관련 도메인에 대한 자동 경로를 제공할 수 있습니다. 물론 계층 체인의 모든 영역이 제대로 구성되어 있어야 합니다. 자동 경로를 사용하면 관리 부담이 줄어듭니다. 그러나 여러 레벨의 도메인이 있을 경우 기본 경로에는 너무 많은 트랜잭션이 필요하기 때문에 기본 경로를 사용하지 않을 수 있습니다.

신뢰 관계를 직접 설정하도록 선택할 수도 있습니다. 직접 신뢰 관계는 두 계층 영역 간에 너무 많은 레벨이 있거나 계층 관계가 존재하지 않을 경우에 가장 유용합니다. 연결을 사용하는 모든 호스트의 `/etc/krb5/krb5.conf` 파일에 연결을 정의해야 합니다. 따라서 약간의 추가 작업이 필요합니다. 직접 신뢰 관계는 추이적 관계라고도 합니다. 지침은 333 페이지 “Kerberos 영역”을 참조하십시오. 여러 영역에 대한 구성 절차는 372 페이지 “영역 간 인증 구성”을 참조하십시오.

호스트 이름과 영역 간 매핑

호스트 이름과 영역 이름 간 매핑은 `krb5.conf` 파일의 `domain_realm` 섹션에 정의되어 있습니다. 요구 사항에 따라 이러한 매핑은 전체 도메인 및 개별 호스트에 대해 정의될 수 있습니다.

DNS를 사용하여 KDC에 대한 정보를 조회할 수도 있습니다. DNS를 사용하면 변경할 때마다 모든 클라이언트에서 `krb5.conf` 파일을 편집할 필요가 없기 때문에 정보를 변경하는 것이 더 쉽습니다. 자세한 내용은 `krb5.conf(4)` 매뉴얼 페이지를 참조하십시오.

Solaris Kerberos 클라이언트는 Active Directory 서버와 더 잘 상호 운영됩니다. 영역과 호스트 간 매핑을 제공하도록 Active Directory 서버를 구성할 수 있습니다.

클라이언트 및 서비스 주체 이름

Kerberos 서비스를 사용 중인 경우 모든 호스트에서 DNS를 사용으로 설정해야 합니다. DNS를 사용할 경우 주체는 각 호스트의 FQDN(정규화된 도메인 이름)을 포함해야 합니다. 예를 들어 호스트 이름은 `boston`이고 DNS 도메인 이름은 `example.com`이며 영역 이름은 `EXAMPLE.COM`인 경우, 호스트에 대한 주체 이름은 `host/boston.example.com@EXAMPLE.COM`이어야 합니다. 이 설명서의 예제에서는 DNS가 구성되어 있으며 각 호스트에 대해 FQDN을 사용합니다.

Kerberos 서비스는 DNS를 통해 호스트 별칭을 정규화하며, 연관된 서비스의 주체를 구성할 때 정규화된 형식(cname)을 사용합니다. 따라서 서비스 주체를 만드는 경우 서비스 주체 이름의 호스트 이름 구성 요소는 서비스를 호스팅하는 시스템 호스트 이름의 표준화된 형식이어야 합니다.

다음은 Kerberos 서비스가 호스트 이름을 정규화하는 방식에 대한 예제입니다. 사용자가 "ssh alpha.example.com" 명령을 실행한다고 가정합니다, 여기서 `alpha.example.com`은 cname `beta.example.com`에 대한 DNS 호스트 별칭입니다. ssh가 Kerberos를 호출하고 `alpha.example.com`에 대한 호스트 서비스 티켓을 요청할 경우, Kerberos 서비스는 `alpha.example.com`을 `beta.example.com`으로 정규화하고 KDC로부터 서비스 주체 "host/beta.example.com"에 대한 티켓을 요청합니다.

주체 이름에 호스트의 FQDN이 포함된 경우, `/etc/resolv.conf` 파일에서 DNS 도메인 이름을 설명하는 문자열과 일치시키는 것이 중요합니다. 주체에 대해 FQDN을 지정한 경우 Kerberos 서비스에서 DNS 도메인 이름이 소문자여야 합니다. DNS 도메인 이름에는 대문자와 소문자가 포함될 수 있지만 호스트 주체를 만드는 경우에는 소문자만 사용해야 합니다. 예를 들어 DNS 도메인 이름이 `example.com`, `Example.COM` 또는 다른 변형이든지 중요하지 않습니다. 호스트에 대한 주체 이름은 `host/boston.example.com@EXAMPLE.COM`입니다.

또한 DNS 클라이언트 서비스가 실행 중이지 않은 경우 대부분의 데몬이나 명령이 시작되지 않도록 서비스 관리 기능이 구성되었습니다. `kdb5_util`, `kadmind` 및 `kpropd` 데몬과 `kprop` 명령은 모두 DNS 서비스에 의존하도록 구성되었습니다. Kerberos 서비스 및 SMF를 사용하여 제공되는 기능을 완전히 활용하려면 모든 호스트에서 DNS 클라이언트 서비스를 사용으로 설정해야 합니다.

KDC 및 관리 서비스용 포트

기본적으로 KDC에는 포트 `88` 및 포트 `750`이 사용되며, KDC 관리 데몬에는 포트 `749`가 사용됩니다. 다른 포트 번호를 사용할 수 있습니다. 그러나 포트 번호를 변경한 경우 모든 클라이언트에서 `/etc/services` 및 `/etc/krb5/krb5.conf` 파일을 변경해야 합니다. 이 파일 이외에도 각 KDC의 `/etc/krb5/kdc.conf` 파일도 업데이트해야 합니다.

슬레이브 KDC 수

슬레이브 KDC는 마스터 KDC와 마찬가지로 클라이언트에 대한 자격 증명을 생성합니다. 또한 마스터를 사용할 수 없는 경우 백업을 제공합니다. 각 영역에는 최소 한 개의 슬레이브 KDC가 있어야 합니다. 다음 요인에 따라 슬레이브 KDC가 추가로 필요할 수 있습니다.

- 영역에 있는 물리적 세그먼트 수. 일반적으로 나머지 영역 없이도 적어도 각 세그먼트가 작동할 수 있도록 네트워크를 설정해야 합니다. 이를 위해서는 각 세그먼트에서 KDC에 액세스할 수 있어야 합니다. 이 경우 KDC는 마스터 또는 슬레이브일 수 있습니다.
- 영역에 있는 클라이언트 수. 슬레이브 KDC 서버를 더 추가하면 현재 서버에 대한 로드를 줄일 수 있습니다.

슬레이브 KDC는 아주 많이 추가할 수 있습니다. 설치된 KDC 서버가 많을수록 영역 전체에서 업데이트된 데이터를 가져오는 시간이 오래 걸릴 수 있으므로, KDC

데이터베이스를 각 서버로 전파해야 합니다. 또한 각 슬레이브에는 KDC 데이터베이스의 복사본이 보존되어 있으므로 슬레이브가 많을수록 보안 유출의 위험이 높아집니다.

이외에도 하나 이상의 슬레이브 KDC가 마스터 KDC로 교체되도록 손쉽게 구성할 수 있습니다. 이와 같은 방식으로 최소 하나의 슬레이브 KDC를 구성할 경우 얻게 되는 이점은 어떤 이유로 마스터 KDC에서 오류가 발생할 경우 마스터 KDC로 손쉽게 교체할 시스템이 미리 구성되어 있다는 점입니다. 교체 가능한 슬레이브 KDC를 구성하는 방법은 402 페이지 “마스터 KDC와 슬레이브 KDC 교체”를 참조하십시오.

UNIX 자격 증명과 GSS 자격 증명 간 매핑

Kerberos 서비스는 매핑을 필요로 하는 GSS 응용 프로그램(예: NFS)에 GSS 자격 증명 이름과 UNIX 사용자 ID(UID) 간 기본 매핑을 제공합니다. Kerberos 서비스를 사용할 경우 GSS 자격 증명 이름은 Kerberos 주체 이름과 같습니다. 기본 매핑 알고리즘은 한 구성 요소의 Kerberos 주체 이름을 가져와서 주체의 기본 이름인 해당 구성 요소를 사용하여 UID를 조회하는 것입니다. 조회는 기본 영역 또는 `/etc/krb5/krb5.conf`의 `auth_to_local_realm` 매개변수를 사용하여 허용되는 영역에서 수행됩니다. 예를 들어 사용자 주체 이름 `bob@EXAMPLE.COM`은 암호 테이블을 사용하여 UNIX 사용자 `bob`의 UID에 매핑됩니다. 사용자 주체 이름 `bob/admin@EXAMPLE.COM`은 `admin`이라는 인스턴스 구성 요소를 포함하고 있으므로 매핑되지 않습니다. 사용자 자격 증명에 대한 기본 매핑으로 충분할 경우, GSS 자격 증명 테이블을 채울 필요가 없습니다. 이전 릴리스에서는 NFS 서비스가 작동하려면 GSS 자격 증명 테이블을 채워야 했습니다. 인스턴스 구성 요소를 포함하는 주체 이름을 매핑하려는 경우와 같이 기본 매핑으로 충분하지 않을 경우, 다른 방법을 사용해야 합니다. 자세한 내용은 다음을 참조하십시오.

- 380 페이지 “자격 증명 테이블을 만드는 방법”
- 380 페이지 “자격 증명 테이블에 단일 항목 추가 방법”
- 381 페이지 “영역 간 자격 증명 매핑 제공 방법”
- 444 페이지 “GSS 자격 증명에서 UNIX 자격 증명으로 매핑”

Kerberos 영역으로 자동 사용자 마이그레이션

기본 Kerberos 영역에 유효한 사용자 계정이 없는 UNIX 사용자는 PAM 프레임워크를 사용하여 자동으로 마이그레이션할 수 있습니다. 특히 `pam_krb5_migrate` 모듈이 PAM 서비스의 인증 스택에 사용됩니다. Kerberos 주체가 없는 사용자가 자신의 암호를 사용하여 시스템에 성공적으로 로그인하면 해당 사용자에 대해 Kerberos 주체가 자동으로 생성되도록 서비스가 설정됩니다. 새 주체 암호는 UNIX 암호와 같습니다. `pam_krb5_migrate` 모듈의 사용법은 397 페이지 “Kerberos 영역에서 사용자의 자동 마이그레이션을 구성하는 방법”을 참조하십시오.

사용할 데이터베이스 전파 시스템

마스터 KDC에 저장된 데이터베이스는 정기적으로 슬레이브 KDC로 전파해야 합니다. 데이터베이스 전파는 증분 방식으로 구성할 수 있습니다. 증분 프로세스 전파는 전체 데이터베이스가 아닌 업데이트된 정보만 슬레이브 KDC로 전파합니다. 데이터베이스 전파에 대한 자세한 내용은 406 페이지 “Kerberos 데이터베이스 관리”를 참조하십시오.

증분 전파를 사용하지 않을 경우 가장 먼저 해결해야 할 문제 중 하나는 슬레이브 KDC의 업데이트 간격입니다. 업데이트를 완료하는 데 걸리는 시간과 최신 정보를 모든 클라이언트에 제공해야 하는 필요성을 비교 검토해야 합니다.

한 영역에 여러 개의 KDC가 있는 대형 설치의 경우, 프로세스가 병렬로 수행되도록 하나 이상의 슬레이브가 데이터를 전파할 수 있습니다. 이 전략을 사용하면 업데이트에 걸리는 시간은 줄어들지만 영역 관리의 복잡도는 더 커집니다. 이 전략에 대한 자세한 설명은 418 페이지 “병렬 전파 설정”을 참조하십시오.

영역 내에서 클럭 동기화

Kerberos 인증 시스템에 참여하는 모든 호스트의 내부 클럭은 지정된 최대 시간 내에서 동기화되어야 합니다. **클럭 불균형**이라고 하는 이 기능은 또 다른 Kerberos 보안 검사를 제공합니다. 참여하는 호스트 사이에 클럭 불균형을 초과할 경우 요청이 거부됩니다.

모든 클럭을 동기화하는 한 가지 방법은 NTP(Network Time Protocol) 소프트웨어를 사용하는 것입니다. 자세한 내용은 400 페이지 “KDC와 Kerberos 클라이언트 간의 클럭 동기화”를 참조하십시오. 클럭을 동기화하는 다른 방법도 있으므로 반드시 NTP를 사용할 필요는 없습니다. 그러나 클럭 불균형으로 인한 액세스 실패를 방지하려면 특정 형태의 동기화를 사용해야 합니다.

클라이언트 구성 옵션

Solaris 10 릴리스의 새로운 기능은 `kclient` 구성 유틸리티입니다. 이 유틸리티는 대화식 모드 또는 비대화식 모드로 실행할 수 있습니다. 대화식 모드에서는 Kerberos별 매개변수 값을 입력하는 프롬프트가 사용자에게 표시되므로, 사용자가 클라이언트를 구성할 때 기존 설치를 변경할 수 있습니다. 비대화식 모드에서는 이전에 설정한 매개변수 값을 사용하는 파일이 사용됩니다. 또한 비대화식 모드에서는 명령줄 옵션도 사용할 수 있습니다. 대화식 모드와 비대화식 모드 모두 수동 프로세스에 비해 필요한 단계가 적으므로 프로세스를 더 신속하게 수행할 수 있으며 오류가 발생할 가능성이 낮습니다.

Solaris Express Developer Edition 1/08 릴리스에서는 제로 구성 Kerberos 클라이언트를 허용하도록 변경되었습니다. 사용 환경에서 다음 규칙을 준수한다면 Solaris Kerberos 클라이언트에 대해 명시적인 구성 절차를 수행할 필요가 없습니다.

- KDC에 대해 SRV 레코드를 반환하도록 DNS를 구성합니다.
- 영역 이름이 DNS 도메인 이름과 일치하거나 KDC에서 지원을 참조합니다.
- Kerberos 클라이언트에는 `keytab` 파일이 필요하지 않습니다.

Kerberos 클라이언트를 명시적으로 구성하는 것이 더 나은 경우도 있습니다.

- 참조가 사용되지 않은 경우 제로 구성 논리는 호스트의 DNS 도메인 이름에 따라 영역을 결정합니다. 이 경우 약간의 보안 위험이 발생하지만, `dns_lookup_realm`을 사용으로 설정하는 경우보다는 덜 위험합니다.
- `pam_krb5` 모듈은 `keytab`의 호스트 키 항목에 의존합니다. 이 요구 사항은 `krb5.conf` 파일에서 사용 안함으로 설정할 수 있지만, 이는 보안상의 이유로 권장되지 않습니다. `krb5.conf(4)` 매뉴얼 페이지를 참조하십시오.
- 제로 구성 프로세스는 직접 구성보다 덜 효율적이며 DNS에 대한 의존도가 더 높습니다. 이 프로세스는 직접적으로 구성된 클라이언트보다 더 많이 DNS 조회를 수행합니다.

모든 클라이언트 구성 프로세스에 대한 설명은 384 페이지 “Kerberos 클라이언트 구성”을 참조하십시오.

클라이언트 로그인 보안 향상

로그인 시 클라이언트는 `pam_krb5` 모듈을 사용하여 최신 TGT를 발행한 KDC와 `/etc/krb5/krb5.keytab`에 저장된 클라이언트 호스트 주체를 발행한 KDC가 동일한지 확인합니다. `pam_krb5` 모듈은 인증 스택에 구성된 경우에 KDC를 확인합니다. 클라이언트 호스트 주체를 저장하지 않는 DHCP 클라이언트와 같은 일부 구성의 경우 이 검사를 사용 안함으로 설정해야 합니다. 이 검사를 해제하려면 `krb5.conf` 파일의 `verify_ap_req_nofail` 옵션을 false로 설정해야 합니다. 자세한 내용은 395 페이지 “TGT(티켓 부여 티켓) 확인을 사용 안함으로 설정하는 방법”을 참조하십시오.

KDC 구성 옵션

KDC를 구성하는 방법은 여러 가지입니다. 가장 간단한 방법은 `kdcmgr` 유틸리티를 사용하여 KDC를 자동으로 또는 대화식으로 구성하는 것입니다. 자동 버전의 경우 명령줄 옵션을 사용하여 구성 매개변수를 정의해야 합니다. 이 방법은 특히 스크립트의 경우에 유용합니다. 대화식 버전의 경우 필요한 모든 정보를 입력하는 프롬프트를 표시합니다. 이 명령 사용에 대한 지침은 표 21–1을 참조하십시오.

LDAP을 사용하여 Kerberos용 데이터베이스 파일을 관리하기 위한 지원도 제공됩니다. 지침은 360 페이지 “LDAP 데이터 서버를 사용하도록 KDC를 구성하는 방법”을

참조하십시오. LDAP을 사용하면 사이트에 대한 관리가 간소화되므로 Kerberos 데이터베이스와 설정된 기존 디렉토리 서버 간에 더 나은 조정이 필요합니다.

위임을 위해 서비스 신뢰

일부 응용 프로그램의 경우 클라이언트는 다른 서비스에 연결할 때 대신 작동할 서버로 권한을 위임할 수 있습니다. 이 경우 클라이언트가 자격 증명을 중간 서버로 전송해야 합니다. 서버에 대한 서비스 티켓을 얻기 위한 클라이언트 기능은 위임된 자격 증명을 수락할 정도로 서버를 신뢰하는지 여부에 대한 정보를 클라이언트에 전달하지 않습니다. `kadmin` 명령에 대한 `ok_to_auth_as_delegate` 옵션은 이러한 자격 증명을 수락할 정도로 중간 서버를 신뢰하는지 여부와 관련하여 KDC가 로컬 영역 정책을 클라이언트로 전달할 수 있도록 해줍니다.

KDC 응답의 암호화된 부분에 있는 자격 증명 티켓 플래그 복사본에는 티켓에 지정된 서버가 영역의 정책에 따라 위임을 수신할 적합한 대상으로 결정되었음을 클라이언트에 표시하도록 `ok_to_auth_as_delegate` 옵션이 설정되어 있습니다. 클라이언트는 이 정보를 사용하여 (프록시 또는 전송된 TGT를 부여하는 방식으로) 자격 증명을 이 서버로 위임할지 여부를 결정합니다. 이 옵션을 설정할 때 관리자는 서비스가 위임된 자격 증명을 사용할지 여부뿐만 아니라 보안 및 서비스가 실행되는 서버의 배치를 고려해야 합니다.

Kerberos 암호화 유형

암호화 유형은 Kerberos 서비스에 사용된 암호화 알고리즘, 암호화 모드 및 해시 알고리즘을 지정하는 식별자입니다. Kerberos 서비스의 키에는 서비스가 키로 암호화 작업을 수행할 때 사용할 암호화 알고리즘 및 모드를 식별할 수 있는 연관된 암호화 유형이 지정되어 있습니다. 지원되는 암호화 유형은 다음과 같습니다.

- `des-cbc-md5`
- `des-cbc-crc`
- `des3-cbc-sha1-kd`
- `arcfour-hmac-md5`
- `arcfour-hmac-md5-exp`
- `aes128-cts-hmac-sha1-96`
- `aes256-cts-hmac-sha1-96`

주 – Solaris 10 8/07 이전 릴리스에서는 번들화되지 않은 강력한 암호화 패키지가 설치된 경우에 `aes256-cts-hmac-sha1-96` 암호화 유형을 Kerberos 서비스에 사용할 수 있습니다.

암호화 유형을 변경하려면 주체 데이터베이스를 새로 만들 때 변경해야 합니다. KDC, 서버 및 클라이언트 간의 상호 작용 때문에 기존 데이터베이스에서 암호화 유형을

변경하는 것은 어려운 일입니다. 데이터베이스를 다시 만드는 경우가 아니면 이러한 매개변수를 설정되지 않은 상태로 두십시오. 자세한 내용은 513 페이지 “Kerberos 암호화 유형 사용”을 참조하십시오.

주 – 설치된 마스터 KDC에서 Solaris 10 릴리스가 실행 중이지 않은 경우 마스터 KDC를 업그레이드하기 전에 슬레이브 KDC를 Solaris 10 릴리스로 업그레이드해야 합니다. Solaris 10 마스터 KDC는 이전 슬레이브에서 처리할 수 없는 새 암호화 유형을 사용하게 됩니다.

취약한 암호화 유형인 `arcfour-hmac-md5-exp`, `des-cbc-md5` 및 `des-cbc-crc`는 기본적으로 Oracle Solaris 11 릴리스에서 사용할 수 없습니다. 이 암호화 유형을 계속 사용해야 하는 경우 `/etc/krb5/krb5.conf` 파일의 `libdefaults` 섹션에서 `allow_weak_crypto = true`를 설정하십시오.

그래픽 Kerberos 관리 도구의 온라인 도움말 URL

온라인 도움말 URL은 그래픽 Kerberos 관리 도구 `gkadmin`에 사용되므로, URL을 올바르게 정의해야 “Help Contents(도움말 목차)“ 메뉴가 작동합니다. 이 매뉴얼의 HTML 버전은 해당 서버에 설치할 수 있습니다. 또는 `http://www.oracle.com/technetwork/indexes/documentation/index.html`에 있는 컬렉션을 사용할 수 있습니다.

URL은 Kerberos 서비스를 사용할 호스트를 구성할 때 `krb5.conf` 파일에 지정됩니다. URL은 이 설명서의 **Kerberos 주체 및 정책 관리(작업)** 장의 448 페이지 “SEAM 도구” 절을 가리켜야 합니다. 다른 위치가 더 적합한 경우 다른 HTML 페이지를 선택할 수 있습니다.

◆ ◆ ◆ 21 장

21 Kerberos 서비스 구성(작업)

이 장에서는 KDC 서버, 네트워크 애플리케이션 서버, NFS 서버 및 Kerberos 클라이언트에 대한 구성 절차를 제공합니다. 이러한 절차 중 대부분에는 수퍼유저 액세스 권한이 필요하므로 해당 절차는 시스템 관리자 또는 고급 사용자가 수행해야 합니다. 영역 간 구성 절차 및 기타 KDC 서버 관련 항목도 다룹니다.

다음 항목을 다룹니다.

Kerberos 서비스 구성(작업 맵)

구성 프로세스의 단계는 다른 단계에 종속되므로 특정 순서로 완료되어야 합니다. 해당 절차에서는 Kerberos 서비스 사용에 필요한 서비스를 설정하는 경우가 많습니다. 순서가 중요하지 않은 다른 절차는 적절할 때 완료할 수 있습니다. 다음 작업 맵에서는 Kerberos 설치에 대해 제안되는 순서를 보여 줍니다.

작업	설명	수행 방법
1. Kerberos 설치를 계획합니다.	소프트웨어 구성 프로세스를 시작하기 전에 구성 문제를 해결할 수 있도록 합니다. 사전 계획을 통해 장기 실행에 드는 시간과 기타 리소스를 절약할 수 있습니다.	20 장, “Kerberos 서비스 계획”
2. (옵션) NTP를 설치합니다.	NTP(Network Time Protocol) 소프트웨어 또는 다른 클럭 동기화 프로토콜을 구성합니다. Kerberos 서비스가 제대로 작동하려면 영역 내 모든 시스템의 클럭이 동기화되어야 합니다.	400 페이지 “KDC와 Kerberos 클라이언트 간의 클럭 동기화”
3. KDC 서버를 구성합니다.	영역에 대한 마스터 KDC 및 슬레이브 KDC 서버와 KDC 데이터베이스를 구성하고 구축합니다.	353 페이지 “KDC 서버 구성”
4. (옵션) KDC 서버에서 보안 수준을 향상시킵니다.	KDC 서버에서 보안 위반이 발생하지 않도록 합니다.	425 페이지 “KDC 서버에 대한 액세스 제한 방법”
5. (옵션) 교체 가능한 KDC 서버를 구성합니다.	마스터 KDC와 슬레이브 KDC 교체 작업을 간편하게 수행할 수 있도록 합니다.	402 페이지 “교체 가능한 슬레이브 KDC 구성 방법”

추가 Kerberos 서비스 구성(작업 맵)

필요한 단계가 완료되면 적절할 때 다음 절차를 사용할 수 있습니다.

작업	설명	수행 방법
영역 간 인증을 구성합니다.	영역 간 통신을 사용으로 설정합니다.	372 페이지 “영역 간 인증 구성”
Kerberos 애플리케이션 서버를 구성합니다.	서버가 Kerberos 인증을 사용하는 `ftp`, `telnet`, `rsh` 등의 서비스를 지원할 수 있도록 합니다.	375 페이지 “Kerberos 네트워크 애플리케이션 서버 구성”
Kerberos 클라이언트를 구성합니다.	클라이언트가 Kerberos 서비스를 사용할 수 있도록 합니다.	384 페이지 “Kerberos 클라이언트 구성”
Kerberos NFS 서버를 구성합니다.	서버가 Kerberos 인증을 필요로 하는 파일 시스템을 공유할 수 있도록 합니다.	378 페이지 “Kerberos NFS 서버 구성”
애플리케이션 서버에서 보안 수준을 향상시킵니다.	인증된 트랜잭션으로만 액세스를 제한하여 애플리케이션 서버에서 보안 수준을 향상시킵니다.	424 페이지 “Kerberos화된 응용 프로그램만 사용으로 설정하는 방법”

KDC 서버 구성

Kerberos 소프트웨어를 설치한 후에는 KDC 서버를 구성해야 합니다. 마스터 KDC와 하나 이상의 슬레이브 KDC를 구성하면 자격 증명을 발행하는 서비스가 제공됩니다. 이러한 자격 증명은 Kerberos 서비스의 기본 사항이므로 다른 작업을 시도하기 전에 KDC를 설치해야 합니다.

마스터 KDC와 슬레이브 KDC의 가장 큰 차이점은 마스터 KDC만 데이터베이스 관리 요청을 처리할 수 있다는 것입니다. 예를 들어, 암호 변경 또는 새 주체 추가 작업은 마스터 KDC에서 완료해야 합니다. 그런 다음 해당 변경 사항을 KDC로 전파할 수 있습니다. 슬레이브 KDC와 마스터 KDC는 모두 자격 증명을 생성합니다. 이 기능은 마스터 KDC가 응답할 수 없는 경우 중복을 제공합니다.

표 21–1 KDC 서버 구성(작업 맵)

작업	설명	수행 방법
마스터 KDC를 구성합니다.	스크립트에 적합한 자동 프로세스를 사용하여 영역에 대한 마스터 KDC 서버와 데이터베이스를 구성하고 구축합니다.	354 페이지 “자동으로 마스터 KDC를 구성하는 방법”
	대부분의 설치에 적합한 대화식 프로세스를 사용하여 영역에 대한 마스터 KDC 서버와 데이터베이스를 구성하고 구축합니다.	354 페이지 “대화식으로 마스터 KDC를 구성하는 방법”
	보다 복잡한 설치에 필요한 수동 프로세스를 사용하여 영역에 대한 마스터 KDC 서버와 데이터베이스를 구성하고 구축합니다.	356 페이지 “수동으로 마스터 KDC를 구성하는 방법”
	수동 프로세스와 KDC용 LDAP을 사용하여 영역에 대한 마스터 KDC 서버와 데이터베이스를 구성하고 구축합니다.	360 페이지 “LDAP 데이터 서버를 사용하도록 KDC를 구성하는 방법”
슬레이브 KDC 서버를 구성합니다.	스크립트에 적합한 자동 프로세스를 사용하여 영역에 대한 슬레이브 KDC 서버를 구성하고 구축합니다.	366 페이지 “자동으로 슬레이브 KDC를 구성하는 방법”
	대부분의 설치에 적합한 대화식 프로세스를 사용하여 영역에 대한 슬레이브 KDC 서버를 구성하고 구축합니다.	367 페이지 “대화식으로 슬레이브 KDC를 구성하는 방법”
	보다 복잡한 설치에 필요한 수동 프로세스를 사용하여 영역에 대한 슬레이브 KDC 서버를 구성하고 구축합니다.	368 페이지 “수동으로 슬레이브 KDC를 구성하는 방법”
KDC 서버에서 주체 키를 새로 고칩니다.	새 암호화 유형이 사용되도록 KDC 서버에서 세션 키를 업데이트합니다.	372 페이지 “마스터 서버에서 TGS(티켓 부여 서비스) 키를 새로 고치는 방법”

▼ 자동으로 마스터 KDC를 구성하는 방법

Oracle Solaris 11 릴리스에서는 다음 절차를 사용하여 자동으로 마스터 KDC를 구성할 수 있습니다.

1 관리자가 되거나 Kerberos Server Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 KDC를 만듭니다.

`kdcmgr` 유틸리티를 실행하여 KDC를 만듭니다. 마스터 키 암호와 관리 주체에 대한 암호를 모두 제공해야 합니다.

```
kdc1# kdcmgr -a kws/admin -r EXAMPLE.COM create master

Starting server setup
---------------------------------------

Setting up /etc/krb5/kdc.conf

Setting up /etc/krb5/krb5.conf

Initializing database '/var/krb5/principal' for realm 'EXAMPLE.COM',
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:      <Type the password>
Re-enter KDC database master key to verify:      <Type it again>

Authenticating as principal root/admin@EXAMPLE.COM with password.
WARNING: no policy specified for kws/admin@EXAMPLE.COM; defaulting to no policy
Enter password for principal "kws/admin@EXAMPLE.COM":      <Type the password>
Re-enter password for principal "kws/admin@EXAMPLE.COM":      <Type it again>
Principal "kws/admin@EXAMPLE.COM" created.

Setting up /etc/krb5/kadm5.acl.

-------------------------------------------------
Setup COMPLETE.

kdc1#
```

▼ 대화식으로 마스터 KDC를 구성하는 방법

Oracle Solaris 릴리스에서는 다음 절차를 사용하여 대화식으로 마스터 KDC를 구성할 수 있습니다.

1 **관리자가 되거나 Kerberos Server Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.**

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **KDC를 만듭니다.**

kdcmgr 유틸리티를 실행하여 KDC를 만듭니다. 마스터 키 암호와 관리 주체에 대한 암호를 모두 제공해야 합니다.

```
kdc1# kdcmgr create master

Starting server setup
---------------------------------------

Enter the Kerberos realm: EXAMPLE.COM

Setting up /etc/krb5/kdc.conf

Setting up /etc/krb5/krb5.conf

Initializing database '/var/krb5/principal' for realm 'EXAMPLE.COM',
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:      <Type the password>
Re-enter KDC database master key to verify:      <Type it again>

Enter the krb5 administrative principal to be created: kws/admin

Authenticating as principal root/admin@EXAMPLE.COM with password.
WARNING: no policy specified for kws/admin@EXAMPLE.COM; defaulting to no policy
Enter password for principal "kws/admin@EXAMPLE.COM":      <Type the password>
Re-enter password for principal "kws/admin@EXAMPLE.COM":      <Type it again>
Principal "kws/admin@EXAMPLE.COM" created.

Setting up /etc/krb5/kadm5.acl.

--------------------------------------------------
Setup COMPLETE.

kdc1#
```

예 21–1 KDC 서버 상태 표시

kdcmgr status 명령을 사용하여 마스터 또는 슬레이브 KDC 서버에 대한 정보를 표시할 수 있습니다.

▼ 수동으로 마스터 KDC를 구성하는 방법

이 절차에서는 증분 전파가 구성됩니다. 또한 다음 구성 매개변수가 사용됩니다.

- 영역 이름 = `EXAMPLE.COM`
- DNS 도메인 이름 = `example.com`
- 마스터 KDC = `kdc1.example.com`
- `admin` 주체 = `kws/admin`
- 온라인 도움말 URL = `http://download.oracle.com/docs/cd/E23824_01/html/821-1456/aadmin-23.html`

주 – 349 페이지 "그래픽 Kerberos 관리 도구의 온라인 도움말 URL"에 설명된 대로 절을 가리키도록 URL을 조정하십시오.

시작하기 전에 이 절차를 수행하려면 호스트가 DNS를 사용하도록 구성되어 있어야 합니다. 이 마스터를 교체 가능한 것으로 설정하려는 경우 구체적인 이름 지정 지침은 402 페이지 "마스터 KDC와 슬레이브 KDC 교체"를 참조하십시오.

1 마스터 KDC에서 수퍼유저로 로그인합니다.

2 Kerberos 구성 파일(`krb5.conf`)을 편집합니다.

영역 이름 및 서버 이름을 변경해야 합니다. 이 파일에 대한 자세한 설명은 `krb5.conf(4)` 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/krb5.conf
[libdefaults]
        default_realm = EXAMPLE.COM

[realms]
        EXAMPLE.COM = {
        kdc = kdc1.example.com
        admin_server = kdc1.example.com
        }

[domain_realm]
        .example.com = EXAMPLE.COM
#
# if the domain name and realm name are equivalent,
# this entry is not needed
#
[logging]
        default = FILE:/var/krb5/kdc.log
        kdc = FILE:/var/krb5/kdc.log

[appdefaults]
    gkadmin = {
        help_url = http://download.oracle.com/docs/cd/E23824_01/html/821-1456/aadmin-23.html
        }
```

이 예에서는 default_realm, kdc, admin_server 및 모든 domain_realm 항목에 대한 행이 변경되었습니다. 또한 help_url을 정의하는 행이 편집되었습니다.

주 – 암호화 유형을 제한하려는 경우 default_tkt_enctypes 또는 default_tgs_enctypes 행을 설정할 수 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 513 페이지 "Kerberos 암호화 유형 사용"을 참조하십시오.

---

3 KDC 구성 파일(kdc.conf)을 편집합니다.

영역 이름을 변경해야 합니다. 이 파일에 대한 자세한 설명은 kdc.conf(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM = {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                sunw_dbprop_enable = true
                sunw_dbprop_master_ulogsize = 1000
                }
```

이 예에서는 realms 섹션의 영역 이름 정의가 변경되었습니다. 또한 realms 섹션에서 증분 전파를 사용으로 설정하고 KDC 마스터가 로그에 유지할 업데이트 수를 선택하는 행이 추가되었습니다.

주 – 암호화 유형을 제한하려는 경우 permitted_enctypes, supported_enctypes 또는 master_key_type 행을 설정할 수 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 513 페이지 "Kerberos 암호화 유형 사용"을 참조하십시오.

---

4 kdb5_util 명령을 사용하여 KDC 데이터베이스를 만듭니다.

kdb5_util 명령은 KDC 데이터베이스를 만듭니다. 또한 -s 옵션과 함께 사용할 경우 이 명령은 kadmind 및 krb5kdc 데몬이 시작되기 전에 KDC를 자체적으로 인증하는 데 사용되는 stash 파일을 만듭니다.

```
kdc1 # /usr/sbin/kdb5_util create -s
Initializing database '/var/krb5/principal' for realm 'EXAMPLE.COM'
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:      <Type the key>
Re-enter KDC database master key to verify:      <Type it again>
```

5 액세스 제어 목록 파일(kadm5.acl)을 편집합니다.

채워진 /etc/krb5/kadm5.acl 파일에는 KDC를 관리할 수 있도록 허용된 모든 주체 이름이 포함되어야 합니다.

```
kws/admin@EXAMPLE.COM    *
```

이 항목은 EXAMPLE.COM 영역의 kws/admin 주체가 KDC의 주체 또는 정책을 수정할 수 있도록 합니다. 기본 설치에는 모든 admin 주체와 일치하는 별표(*)가 포함됩니다. 이 기본 설치의 경우 보안 위험이 있을 수 있으므로 모든 admin 주체의 목록을 포함하는 것이 더 안전합니다. 자세한 내용은 kadm5.acl(4) 매뉴얼 페이지를 참조하십시오.

6 kadmin.local 명령을 시작하고 주체를 추가합니다.

다음 하위 단계에서 Kerberos 서비스에 사용되는 주체를 만듭니다.

```
kdc1 # /usr/sbin/kadmin.local
kadmin.local:
```

a. 데이터베이스에 관리 주체를 추가합니다.

admin 주체는 필요에 따라 여러 개 추가할 수 있습니다. KDC 구성 프로세스를 완료하려면 admin 주체를 하나 이상 추가해야 합니다. 이 예에서는 kws/admin 주체가 추가됩니다. "kws" 대신 적절한 주체 이름으로 대체할 수 있습니다.

```
kadmin.local: addprinc kws/admin
Enter password for principal kws/admin@EXAMPLE.COM:     <Type the password>
Re-enter password for principal kws/admin@EXAMPLE.COM:     <Type it again>
Principal "kws/admin@EXAMPLE.COM" created.
kadmin.local:
```

b. kiprop 주체를 만듭니다.

kiprop 주체는 마스터 KDC로부터의 업데이트를 허가하는 데 사용됩니다.

```
kadmin.local: addprinc -randkey kiprop/kdc1.example.com
Principal "kiprop/kdc1.example.com@EXAMPLE.COM" created.
kadmin.local:
```

c. kadmin.local을 종료합니다.

다음 단계에 필요한 모든 주체를 추가한 것입니다.

```
kadmin.local: quit
```

7 Kerberos 데몬을 시작합니다.

```
kdc1 # svcadm enable -r network/security/krb5kdc
kdc1 # svcadm enable -r network/security/kadmin
```

8 kadmin을 시작하고 주체를 더 추가합니다.

이 단계에서 그래픽 Kerberos 관리 도구를 사용하여 주체를 추가할 수 있습니다. 이 작업을 수행하려면 이 절차의 앞부분에서 만든 admin 주체 이름 중 하나로 로그인해야 합니다. 하지만 간소화를 위해 다음 명령줄 예가 제공됩니다.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. 마스터 KDC host 주체를 만듭니다.

host 주체는 Kerberos화된 응용 프로그램(예: kprop)이 변경 사항을 슬레이브 KDC에 전파하는 데 사용합니다. 또한 이 주체는 ssh 등의 응용 프로그램을 사용하여 KDC 서버에 대한 보안 원격 액세스를 제공하는 데 사용됩니다. 주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

```
kadmin: addprinc -randkey host/kdc1.example.com
Principal "host/kdc1.example.com@EXAMPLE.COM" created.
kadmin:
```

b. (옵션) kclient 주체를 만듭니다.

이 주체는 kclient 유틸리티가 Kerberos 클라이언트를 설치하는 동안 사용합니다. 이 유틸리티를 사용하지 않으려는 경우 주체를 추가할 필요가 없습니다. kclient 유틸리티 사용자가 이 암호를 사용해야 합니다.

```
kadmin: addprinc clntconfig/admin
Enter password for principal clntconfig/admin@EXAMPLE.COM:     <Type the password>
Re-enter password for principal clntconfig/admin@EXAMPLE.COM:      <Type it again>
Principal "clntconfig/admin@EXAMPLE.COM" created.
kadmin:
```

c. 마스터 KDC의 keytab 파일에 마스터 KDC의 host 주체를 추가합니다.

keytab 파일에 host 주체를 추가하면 sshd 등의 애플리케이션 서버가 자동으로 이 주체를 사용할 수 있습니다.

```
kadmin: ktadd host/kdc1.example.com
Entry for principal host/kdc1.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

d. kadmin을 종료합니다.

```
kadmin: quit
```

9 **(옵션) NTP 또는 다른 클럭 동기화 방식을 사용하여 마스터 KDC의 클럭을 동기화합니다.**

NTP(Network Time Protocol)를 설치하여 사용할 필요가 없습니다. 하지만 인증이 성공하려면 모든 클럭이 `krb5.conf` 파일의 `libdefaults` 섹션에 정의된 기본 시간에 속해야 합니다. NTP에 대한 자세한 내용은 400 페이지 "KDC와 Kerberos 클라이언트 간의 클럭 동기화"를 참조하십시오.

10 **슬레이브 KDC를 구성합니다.**

중복을 제공하려면 슬레이브 KDC를 하나 이상 설치해야 합니다. 구체적인 지침은 368 페이지 "수동으로 슬레이브 KDC를 구성하는 방법"을 참조하십시오.

▼ LDAP 데이터 서버를 사용하도록 KDC를 구성하는 방법

다음 절차에 따라 LDAP 데이터 서버를 사용하도록 KDC를 구성할 수 있습니다.

이 절차에서는 다음 구성 매개변수가 사용됩니다.

- 영역 이름 = `EXAMPLE.COM`
- DNS 도메인 이름 = `example.com`
- 마스터 KDC = `kdc1.example.com`
- 디렉토리 서버 = `dsserver.example.com`
- `admin` 주체 = `kws/admin`
- LDAP 서비스의 FMRI = `svc:/application/sun/ds:ds--var-opt-SUNWdsee-dsins1`
- 온라인 도움말 URL = `http://download.oracle.com/docs/cd/E23824_01/html/821-1456/aadmin-23.html`

주 – 349 페이지 "그래픽 Kerberos 관리 도구의 온라인 도움말 URL"에 설명된 대로 절을 가리키도록 URL을 조정하십시오.

시작하기 전에 또한 이 절차를 수행하려면 호스트가 DNS를 사용하도록 구성되어 있어야 합니다. 성능을 향상시키려면 KDC 및 LDAP 디렉토리 서비스를 동일한 서버에 설치하십시오. 또한 디렉토리 서버가 실행 중이어야 합니다. 다음 절차는 Sun Directory Server Enterprise Edition 7.0 릴리스를 사용하는 서버에서 작동합니다.

1 **KDC에서 수퍼유저로 로그인합니다.**

2 **SSL을 사용하여 디렉토리 서버에 연결하도록 마스터 KDC를 구성합니다.**

다음 단계에서는 디렉토리 서버의 자체 서명된 인증서를 사용하도록 Oracle Solaris 릴리스 KDC를 구성합니다. 인증서가 만료된 경우 "To Manage Self-Signed Certificates"의 인증서 갱신 절차를 따르십시오.

a. **디렉토리 서버에서 자체 서명된 디렉토리 서버 인증서를 내보냅니다.**

```
# /export/sun-ds6.1/ds6/bin/dsadm show-cert -F der /export/sun-ds6.1/directory2 \
        defaultCert > /tmp/defaultCert.cert.der
```

b. **마스터 KDC에서 디렉토리 서버 인증서를 가져옵니다.**

```
# pktool setpin keystore=nss dir=/var/ldap
# chmod a+r /var/ldap/*.db
# pktool import keystore=nss objtype=cert trust="CT" infile=/tmp/defaultCert.certutil.der \
        label=defaultCert dir=/var/ldap
```

c. **마스터 KDC에서 SSL이 작동 중인지 테스트합니다.**

이 예에서는 cn=directory manager 항목에 관리 권한이 있는 것으로 간주합니다.

```
/usr/bin/ldapsearch -Z -P /var/ldap -D "cn=directory manager" \
        -h dsserver.example.com -b "" -s base objectclass='*'
Subject:
    "CN=dsserver.example.com,CN=636,CN=Directory Server,O=Example Corporation
```

CN=dsserver.example.com 항목에는 간단한 버전이 아닌 정규화된 호스트 이름이 포함되어야 합니다.

3 **필요한 경우 LDAP 디렉토리를 채웁니다.**

4 **기존 스키마에 Kerberos 스키마를 추가합니다.**

```
# ldapmodify -h dsserver.example.com -D "cn=directory manager" -f /usr/share/lib/ldif/kerberos.ldif
```

5 **LDAP 디렉토리에 Kerberos 컨테이너를 만듭니다.**

krb5.conf 파일에 다음 항목을 추가합니다.

a. **데이터베이스 유형을 정의합니다.**

realms 섹션에 database_module을 정의하는 항목을 추가합니다.

```
database_module = LDAP
```

b. **데이터베이스 모듈을 정의합니다.**

```
[dbmodules]
    LDAP = {
        ldap_kerberos_container_dn = "cn=krbcontainer,dc=example,dc=com"
        db_library = kldap
        ldap_kdc_dn = "cn=kdc service,ou=profile,dc=example,dc=com"
        ldap_kadmind_dn = "cn=kadmin service,ou=profile,dc=example,dc=com"
        ldap_cert_path = /var/ldap
        ldap_servers = ldaps://dsserver.example.com
    }
```

c. LDAP 디렉토리에 KDC를 만듭니다.

이 명령은 krbcontainer 및 여러 개의 다른 객체를 만듭니다. 또한 /var/krb5/.k5.EXAMPLE.COM 마스터 키 stash 파일을 만듭니다.

```
# kdb5_ldap_util -D "cn=directory manager" create -P abcd1234 -r EXAMPLE.COM -s
```

6 KDC 바인드 고유 이름(DN) 암호를 숨깁니다.

해당 암호는 KDC가 DS에 바인드할 때 사용합니다. KDC는 KDC가 사용 중인 액세스의 유형에 따라 다른 역할을 사용합니다.

```
# kdb5_ldap_util stashsrvpw "cn=kdc service,ou=profile,dc=example,dc=com"
# kdb5_ldap_util stashsrvpw "cn=kadmin service,ou=profile,dc=example,dc=com"
```

7 KDC 서비스 역할을 추가합니다.

a. 다음과 같은 내용으로 kdc_roles.ldif 파일을 만듭니다.

```
dn: cn=kdc service,ou=profile,dc=example,dc=com
cn: kdc service
sn: kdc service
objectclass: top
objectclass: person
userpassword: test123

dn: cn=kadmin service,ou=profile,dc=example,dc=com
cn: kadmin service
sn: kadmin service
objectclass: top
objectclass: person
userpassword: test123
```

b. LDAP 디렉토리에 역할 항목을 만듭니다.

```
# ldapmodify -a -h dsserver.example.com -D "cn=directory manager" -f kdc_roles.ldif
```

8 KDC 관련 역할에 대한 ACL을 설정합니다.

```
# cat << EOF | ldapmodify -h dsserver.example.com -D "cn=directory manager"
# Set kadmin ACL for everything under krbcontainer.
dn: cn=krbcontainer,dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///cn=krbcontainer,dc=example,dc=com")(targetattr="krb*")(version 3.0;\
      acl kadmin_ACL; allow (all)\
      userdn = "ldap:///cn=kadmin service,ou=profile,dc=example,dc=com";)

# Set kadmin ACL for everything under the people subtree if there are
# mix-in entries for krb princs:
dn: ou=people,dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///ou=people,dc=example,dc=com")(targetattr="krb*")(version 3.0;\
      acl kadmin_ACL; allow (all)\
      userdn = "ldap:///cn=kadmin service,ou=profile,dc=example,dc=com";)
EOF
```

9 Kerberos 구성 파일(krb5.conf)을 편집합니다.

영역 이름 및 서버 이름을 변경해야 합니다. 이 파일에 대한 자세한 설명은 krb5.conf(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/krb5.conf
[libdefaults]
        default_realm = EXAMPLE.COM

[realms]
        EXAMPLE.COM = {
        kdc = kdc1.example.com
        admin_server = kdc1.example.com
        }

[domain_realm]
        .example.com = EXAMPLE.COM
#
# if the domain name and realm name are equivalent,
# this entry is not needed
#
[logging]
        default = FILE:/var/krb5/kdc.log
        kdc = FILE:/var/krb5/kdc.log

[appdefaults]
    gkadmin = {
        help_url = http://download.oracle.com/docs/cd/E23824_01/html/821-1456/aadmin-23.html
        }
```

이 예에서는 default_realm, kdc, admin_server 및 모든 domain_realm 항목에 대한 행이 변경되었습니다. 또한 help_url을 정의하는 행이 편집되었습니다.

주 – 암호화 유형을 제한하려는 경우 default_tkt_enctypes 또는 default_tgs_enctypes 행을 설정할 수 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 513 페이지 “Kerberos 암호화 유형 사용”을 참조하십시오.

---

10 KDC 구성 파일(kdc.conf)을 편집합니다.

영역 이름을 변경해야 합니다. 이 파일에 대한 자세한 설명은 kdc.conf(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM = {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                sunw_dbprop_enable = true
                sunw_dbprop_master_ulogsize = 1000
                }
```

이 예에서는 realms 섹션의 영역 이름 정의가 변경되었습니다. 또한 realms 섹션에서 증분 전파를 사용으로 설정하고 KDC 마스터가 로그에 유지할 업데이트 수를 선택하는 행이 추가되었습니다.

주 – 암호화 유형을 제한하려는 경우 permitted_enctypes, supported_enctypes 또는 master_key_type 행을 설정할 수 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 513 페이지 “Kerberos 암호화 유형 사용”을 참조하십시오.

---

11 액세스 제어 목록 파일(kadm5.acl)을 편집합니다.

채워진 /etc/krb5/kadm5.acl 파일에는 KDC를 관리할 수 있도록 허용된 모든 주체 이름이 포함되어야 합니다.

```
kws/admin@EXAMPLE.COM    *
```

이 항목은 EXAMPLE.COM 영역의 kws/admin 주체가 KDC의 주체 또는 정책을 수정할 수 있도록 합니다. 기본 설치에는 모든 admin 주체와 일치하는 별표(*)가 포함됩니다. 이 기본 설치의 경우 보안 위험이 있을 수 있으므로 모든 admin 주체의 목록을 포함하는 것이 더 안전합니다. 자세한 내용은 kadm5.acl(4) 매뉴얼 페이지를 참조하십시오.

12 kadmin.local 명령을 시작하고 주체를 추가합니다.

다음 하위 단계에서 Kerberos 서비스에 사용되는 주체를 만듭니다.

```
kdc1 # /usr/sbin/kadmin.local
kadmin.local:
```

a. 데이터베이스에 관리 주체를 추가합니다.

admin 주체는 필요에 따라 여러 개 추가할 수 있습니다. KDC 구성 프로세스를 완료하려면 admin 주체를 하나 이상 추가해야 합니다. 이 예에서는 kws/admin 주체가 추가됩니다. "kws" 대신 적절한 주체 이름으로 대체할 수 있습니다.

```
kadmin.local: addprinc kws/admin
Enter password for principal kws/admin@EXAMPLE.COM:     <Type the password>
Re-enter password for principal kws/admin@EXAMPLE.COM:     <Type it again>
Principal "kws/admin@EXAMPLE.COM" created.
kadmin.local:
```

b. kadmin.local을 종료합니다.

다음 단계에 필요한 모든 주체를 추가한 것입니다.

```
kadmin.local: quit
```

13 (옵션) Kerberos 서비스에 대한 LDAP 종속성을 구성합니다.

LDAP 및 KDC 서버가 동일한 호스트에서 실행되고 있으며 LDAP 서비스가 SMF FMRI로 구성된 경우 Kerberos 데몬에 대한 LDAP 서비스에 종속성을 추가합니다. LDAP 서비스가 다시 시작되는 경우 이 종속성이 KDC 서비스를 다시 시작합니다.

a. krb5kdc 서비스에 종속성을 추가합니다.

```
# svccfg -s security/krb5kdc
svc:/network/security/krb5kdc> addpg dsins1 dependency
svc:/network/security/krb5kdc> setprop dsins1/entities = \
    fmri: "svc:/application/sun/ds:ds--var-opt-SUNWdsee-dsins1"
svc:/network/security/krb5kdc> setprop dsins1/grouping = astring: "require_all"
svc:/network/security/krb5kdc> setprop dsins1/restart_on = astring: "restart"
svc:/network/security/krb5kdc> setprop dsins1/type = astring: "service"
svc:/network/security/krb5kdc> exit
```

b. kadmin 서비스에 종속성을 추가합니다.

```
# svccfg -s security/kadmin
svc:/network/security/kadmin> addpg dsins1 dependency
svc:/network/security/kadmin> setprop dsins1/entities =\
    fmri: "svc:/application/sun/ds:ds--var-opt-SUNWdsee-dsins1"
svc:/network/security/kadmin> setprop dsins1/grouping = astring: "require_all"
svc:/network/security/kadmin> setprop dsins1/restart_on = astring: "restart"
svc:/network/security/kadmin> setprop dsins1/type = astring: "service"
svc:/network/security/kadmin> exit
```

14 Kerberos 데몬을 시작합니다.

```
kdc1 # svcadm enable -r network/security/krb5kdc
kdc1 # svcadm enable -r network/security/kadmin
```

15 kadmin을 시작하고 주체를 더 추가합니다.

이 단계에서 GUI Kerberos 관리 도구를 사용하여 주체를 추가할 수 있습니다. 이 작업을 수행하려면 이 절차의 앞부분에서 만든 admin 주체 이름 중 하나로 로그인해야 합니다. 하지만 간소화를 위해 다음 명령줄 예가 제공됩니다.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. 마스터 KDC host 주체를 만듭니다.

host 주체는 Kerberos화된 응용 프로그램(예: klist 및 kprop)에 사용됩니다. 클라이언트는 인증된 NFS 파일 시스템을 마운트할 때 이 주체를 사용합니다. 주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

```
kadmin: addprinc -randkey host/kdc1.example.com
Principal "host/kdc1.example.com@EXAMPLE.COM" created.
kadmin:
```

b. (옵션) kclient 주체를 만듭니다.

이 주체는 kclient 유틸리티가 Kerberos 클라이언트를 설치하는 동안 사용합니다. 이 유틸리티를 사용하지 않으려는 경우 주체를 추가할 필요가 없습니다. kclient 유틸리티 사용자가 이 암호를 사용해야 합니다.

```
kadmin: addprinc clntconfig/admin
Enter password for principal clntconfig/admin@EXAMPLE.COM:      <Type the password>
Re-enter password for principal clntconfig/admin@EXAMPLE.COM:      <Type it again>
Principal "clntconfig/admin@EXAMPLE.COM" created.
kadmin:
```

c. 마스터 KDC의 keytab 파일에 마스터 KDC의 host 주체를 추가합니다.

keytab 파일에 host 주체를 추가하면 이 주체가 자동으로 사용될 수 있습니다.

```
kadmin: ktadd host/kdc1.example.com
Entry for principal host/kdc1.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

d. kadmin을 종료합니다.

```
kadmin: quit
```

16 (옵션) NTP 또는 다른 클럭 동기화 방식을 사용하여 마스터 KDC의 클럭을 동기화합니다.

NTP(Network Time Protocol)를 설치하여 사용할 필요가 없습니다. 하지만 인증이 성공하려면 모든 클럭이 krb5.conf 파일의 libdefaults 섹션에 정의된 기본 시간에 속해야 합니다. NTP에 대한 자세한 내용은 400 페이지 “KDC와 Kerberos 클라이언트 간의 클럭 동기화”를 참조하십시오.

17 슬레이브 KDC를 구성합니다.

중복을 제공하려면 슬레이브 KDC를 하나 이상 설치해야 합니다. 구체적인 지침은 368 페이지 “수동으로 슬레이브 KDC를 구성하는 방법”을 참조하십시오.

▼ 자동으로 슬레이브 KDC를 구성하는 방법

Oracle Solaris 릴리스에서는 다음 절차를 사용하여 자동으로 슬레이브 KDC를 구성할 수 있습니다.

1 **관리자가 되거나 Kerberos Server Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.**

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **KDC를 만듭니다.**

kdcmgr 유틸리티를 실행하여 KDC를 만듭니다. 마스터 키 암호와 관리 주체에 대한 암호를 모두 제공해야 합니다.

```
kdc2# kdcmgr -a kws/admin -r EXAMPLE.COM create -m kdc1 slave

Starting server setup
---------------------------------------

Setting up /etc/krb5/kdc.conf

Setting up /etc/krb5/krb5.conf
Obtaining TGT for kws/admin ...
Password for kws/admin@EXAMPLE.COM:      <Type the password>

Setting up /etc/krb5/kadm5.acl.

Setting up /etc/krb5/kpropd.acl.

Waiting for database from master...
Waiting for database from master...
Waiting for database from master...
kdb5_util: Cannot find/read stored master key while reading master key
kdb5_util: Warning: proceeding without master key
Enter KDC database master key:      <Type the password>

-------------------------------------------------
Setup COMPLETE.

kdc2#
```

▼ 대화식으로 슬레이브 KDC를 구성하는 방법

다음 절차에 따라 대화식으로 슬레이브 KDC를 구성할 수 있습니다.

1 **관리자가 되거나 Kerberos Server Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.**

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 KDC를 만듭니다.

kdcmgr 유틸리티를 실행하여 KDC를 만듭니다. 마스터 키 암호와 관리 주체에 대한 암호를 모두 제공해야 합니다.

```
kdc1# kdcmgr create slave

Starting server setup
-------------------------------------

Enter the Kerberos realm: EXAMPLE.COM
What is the master KDC's host name?: kdc1

Setting up /etc/krb5/kdc.conf

Setting up /etc/krb5/krb5.conf
Obtaining TGT for kws/admin ...
Password for kws/admin@EXAMPLE.COM:      <Type the password>

Setting up /etc/krb5/kadm5.acl.

Setting up /etc/krb5/kpropd.acl.

Waiting for database from master...
Waiting for database from master...
Waiting for database from master...
kdb5_util: Cannot find/read stored master key while reading master key
kdb5_util: Warning: proceeding without master key
Enter KDC database master key:      <Type the password>

-------------------------------------------------
Setup COMPLETE.

kdc2#
```

▼ 수동으로 슬레이브 KDC를 구성하는 방법

이 절차에서는 이름이 kdc2인 새 슬레이브 KDC가 구성됩니다. 또한 증분 전파가 구성됩니다. 이 절차에서는 다음 구성 매개변수를 사용합니다.

- 영역 이름 = EXAMPLE.COM
- DNS 도메인 이름 = example.com
- 마스터 KDC = kdc1.example.com
- 슬레이브 KDC = kdc2.example.com
- admin 주체 = kws/admin

시작하기 전에 마스터 KDC를 구성해야 합니다. 이 슬레이브를 교체 가능한 것으로 설정하려는 경우 구체적인 지침은 402 페이지 "마스터 KDC와 슬레이브 KDC 교체"를 참조하십시오.

1 마스터 KDC에서 수퍼유저로 로그인합니다.

2 마스터 KDC에서 kadmin을 시작합니다.

마스터 KDC를 구성할 때 만든 admin 주체 이름 중 하나로 로그인해야 합니다.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. 마스터 KDC에서 데이터베이스에 슬레이브 host 주체를 추가합니다(아직 추가하지 않은 경우).

슬레이브가 작동하려면 host 주체가 있어야 합니다. 주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

```
kadmin: addprinc -randkey host/kdc2.example.com
Principal "host/kdc2.example.com@EXAMPLE.COM" created.
kadmin:
```

b. 마스터 KDC에서 kiprop 주체를 만듭니다.

kiprop 주체는 마스터 KDC로부터의 증분 전파를 허가하는 데 사용됩니다.

```
kadmin: addprinc -randkey kiprop/kdc2.example.com
Principal "kiprop/kdc2.example.com@EXAMPLE.COM" created.
kadmin:
```

c. kadmin을 종료합니다.

```
kadmin: quit
```

3 마스터 KDC에서 Kerberos 구성 파일(krb5.conf)을 편집합니다.

슬레이브마다 항목을 하나씩 추가해야 합니다. 이 파일에 대한 자세한 설명은 krb5.conf(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/krb5.conf
 .
 .
[realms]
                EXAMPLE.COM = {
                kdc = kdc1.example.com
                kdc = kdc2.example.com
                admin_server = kdc1.example.com
        }
```

4 마스터 KDC에서 kadm5.acl에 kiprop 항목을 추가합니다.

이 항목은 마스터 KDC가 kdc2 서버에 대한 증분 전파 요청을 수신할 수 있도록 합니다.

```
kdc1 # cat /etc/krb5/kadm5.acl
*/admin@EXAMPLE.COM *
kiprop/kdc2.example.com@EXAMPLE.COM p
```

5 kadm5.acl 파일의 새 항목이 사용되도록 마스터 KDC에서 kadmind를 다시 시작합니다.

```
kdc1 # svcadm restart network/security/kadmin
```

6 모든 슬레이브 KDC에서 마스터 KDC 서버의 KDC 관리 파일을 복사합니다.

마스터 KDC 서버가 각 KDC 서버에 필요한 정보를 업데이트했으므로 모든 슬레이브 KDC에서 이 단계를 수행해야 합니다. ftp 또는 유사한 전송 방식을 사용하여 마스터 KDC의 다음 파일을 복사할 수 있습니다.

- /etc/krb5/krb5.conf
- /etc/krb5/kdc.conf

7 모든 슬레이브 KDC에서 마스터 KDC에 대한 항목과 각 슬레이브 KDC를 데이터베이스 전파 구성 파일 kpropd.acl에 추가합니다.

모든 슬레이브 KDC 서버에서 이 정보를 업데이트해야 합니다.

```
kdc2 # cat /etc/krb5/kpropd.acl
host/kdc1.example.com@EXAMPLE.COM
host/kdc2.example.com@EXAMPLE.COM
```

8 모든 슬레이브 KDC에서 Kerberos 액세스 제어 목록 파일 kadm5.acl이 채워져 있지 않은지 확인합니다.

수정되지 않은 kadm5.acl 파일은 다음과 같이 표시됩니다.

```
kdc2 # cat /etc/krb5/kadm5.acl
*/admin@___default_realm___ *
```

파일에 kiprop 항목이 있을 경우 제거합니다.

9 새 슬레이브에서 kdc.conf의 항목을 변경합니다.

sunw_dbprop_master_ulogsize 항목을 sunw_dbprop_slave_poll을 정의하는 항목으로 바꿉니다. 이 항목은 폴링 시간을 2분으로 설정합니다.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM= {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                sunw_dbprop_enable = true
                sunw_dbprop_slave_poll = 2m
        }
```

10 새 슬레이브에서 kadmin 명령을 시작합니다.

마스터 KDC를 구성할 때 만든 admin 주체 이름 중 하나로 로그인해야 합니다.

```
kdc2 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. kadmin을 사용하여 슬레이브의 keytab 파일에 슬레이브의 host 주체를 추가합니다.

이 항목은 kprop 및 기타 Kerberos화된 응용 프로그램이 작동할 수 있도록 합니다. 주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

```
kadmin: ktadd host/kdc2.example.com
Entry for principal host/kdc2.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc2.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc2.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc2.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc2.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

b. 슬레이브 KDC의 keytab 파일에 kiprop 주체를 추가합니다.

krb5.keytab 파일에 kiprop 주체를 추가하면 증분 전파가 시작될 때 kpropd 명령이 자체적으로 인증할 수 있습니다.

```
kadmin: ktadd kiprop/kdc2.example.com
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

c. kadmin을 종료합니다.

```
kadmin: quit
```

11 새 슬레이브에서 Kerberos 전파 데몬을 시작합니다.

```
kdc2 # svcadm enable network/security/krb5_prop
```

12 새 슬레이브에서 kdb5_util을 사용하여 stash 파일을 만듭니다.

```
kdc2 # /usr/sbin/kdb5_util stash
kdb5_util: Cannot find/read stored master key while reading master key
kdb5_util: Warning: proceeding without master key

Enter KDC database master key:      <Type the key>
```

13 (옵션) 새 슬레이브 KDC에서 NTP 또는 다른 클럭 동기화 방식을 사용하여 마스터 KDC의 클럭을 동기화합니다.

NTP(Network Time Protocol)를 설치하여 사용할 필요가 없습니다. 하지만 인증이 성공하려면 모든 클럭이 `krb5.conf` 파일의 `libdefaults` 섹션에 정의된 기본 시간에 속해야 합니다. NTP에 대한 자세한 내용은 400 페이지 "KDC와 Kerberos 클라이언트 간의 클럭 동기화"를 참조하십시오.

14 새 슬레이브에서 KDC 데몬(krb5kdc)을 시작합니다.

```
kdc2 # svcadm enable network/security/krb5kdc
```

▼ 마스터 서버에서 TGS(티켓 부여 서비스) 키를 새로 고치는 방법

KDC 서버가 Solaris 10 릴리스 이전에 만들어져서 TGS(티켓 부여 서비스) 주체에 DES 키만 있을 경우 이 키는 TGT(티켓 부여 티켓) 세션 키의 암호화 유형을 DES로 제한합니다. KDC가 더 강력한 암호화 유형을 추가로 지원하는 릴리스로 업데이트되면 관리자는 KDC가 생성한 모든 세션 키에 더 강력한 암호화가 사용되도록 할 수 있습니다. 하지만 기존 TGS 주체의 키가 새 암호화 유형을 포함하도록 새로 고쳐지지 않을 경우 TGT 세션 키가 계속 DES로 제한됩니다. 다음 절차에서는 추가 암호화 유형을 사용할 수 있도록 키를 새로 고칩니다.

- **TGS 서비스 주체 키를 새로 고칩니다.**

```
kdc1 % /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin: cpw -randkey krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

예 21–2 마스터 서버에서 주체 키 새로 고침

`root`로 KDC 마스터에 로그온하면 다음 명령을 사용하여 TGS 서비스 주체를 새로 고칠 수 있습니다.

```
kdc1 # kadmin.local -q 'cpw -randkey krbtgt/EXAMPLE.COM@EXAMPLE.COM'
```

영역 간 인증 구성

한 영역의 사용자가 다른 영역에서 인증될 수 있도록 여러 가지 방법으로 영역을 연결할 수 있습니다. 영역 간 인증을 수행하려면 두 영역 간에 공유되는 보안 키를 설정합니다. 영역의 관계는 계층 관계 또는 방향 관계일 수 있습니다(343 페이지 "영역 계층 구조" 참조).

▼ 계층 영역 간 인증 설정 방법

이 절차의 예에서는 두 개의 영역(ENG.EAST.EXAMPLE.COM 및 EAST.EXAMPLE.COM)을 사용합니다. 영역 간 인증이 양방향에서 설정됩니다. 이 절차는 두 영역의 마스터 KDC에서 완료해야 합니다.

시작하기 전에 각 영역에 대한 마스터 KDC를 구성해야 합니다. 인증 프로세스를 완전히 테스트하려면 여러 Kerberos 클라이언트를 구성해야 합니다.

1 첫번째 마스터 KDC에서 수퍼유저로 로그인합니다.

2 두 영역에 대한 TGT(티켓 부여 티켓) 서비스 주체를 만듭니다.

마스터 KDC를 구성할 때 만든 admin 주체 이름 중 하나로 로그인해야 합니다.

```
# /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin: addprinc krbtgt/ENG.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM
Enter password for principal krgtgt/ENG.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM:      <Type password>
kadmin: addprinc krbtgt/EAST.EXAMPLE.COM@ENG.EAST.EXAMPLE.COM
Enter password for principal krgtgt/EAST.EXAMPLE.COM@ENG.EAST.EXAMPLE.COM:      <Type password>
kadmin: quit
```

주 – 각 서비스 주체에 대해 지정된 암호는 두 KDC에서 같아야 합니다. 따라서 서비스 주체 krbtgt/ENG.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM에 대한 암호는 두 영역에서 동일해야 합니다.

3 Kerberos 구성 파일(krb5.conf)에 모든 영역에 대한 도메인 이름을 정의할 항목을 추가합니다.

```
# cat /etc/krb5/krb5.conf
[libdefaults]
 .
 .
[domain_realm]
        .eng.east.example.com = ENG.EAST.EXAMPLE.COM
        .east.example.com = EAST.EXAMPLE.COM
```

이 예에서는 ENG.EAST.EXAMPLE.COM 및 EAST.EXAMPLE.COM 영역에 대한 도메인 이름이 정의됩니다. 파일은 하향식으로 검색되므로 하위 도메인을 먼저 포함시켜야 합니다.

4 Kerberos 구성 파일을 이 영역의 모든 클라이언트에 복사합니다.

영역 간 인증이 작동하려면 모든 시스템(슬레이브 KDC 및 기타 서버 포함)에 Kerberos 구성 파일(/etc/krb5/krb5.conf)의 새 버전이 설치되어야 합니다.

5 두번째 영역에서 위 단계를 모두 반복합니다.

▼ 직접 영역 간 인증 설정 방법

이 절차의 예에서는 두 개의 영역(ENG.EAST.EXAMPLE.COM 및 SALES.WEST.EXAMPLE.COM)을 사용합니다. 영역 간 인증이 양방향에서 설정됩니다. 이 절차는 두 영역의 마스터 KDC에서 완료해야 합니다.

시작하기 전에 각 영역에 대한 마스터 KDC를 구성해야 합니다. 인증 프로세스를 완전히 테스트하려면 여러 Kerberos 클라이언트를 구성해야 합니다.

1 마스터 KDC 서버 중 하나에서 수퍼유저로 로그인합니다.

2 두 영역에 대한 TGT(티켓 부여 티켓) 서비스 주체를 만듭니다.

마스터 KDC를 구성할 때 만든 admin 주체 이름 중 하나로 로그인해야 합니다.

```
# /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin: addprinc krbtgt/ENG.EAST.EXAMPLE.COM@SALES.WEST.EXAMPLE.COM
Enter password for principal
  krgtgt/ENG.EAST.EXAMPLE.COM@SALES.WEST.EXAMPLE.COM:      <Type the password>
kadmin: addprinc krbtgt/SALES.WEST.EXAMPLE.COM@ENG.EAST.EXAMPLE.COM
Enter password for principal
  krgtgt/SALES.WEST.EXAMPLE.COM@ENG.EAST.EXAMPLE.COM:      <Type the password>
kadmin: quit
```

주 – 각 서비스 주체에 대해 지정된 암호는 두 KDC에서 같아야 합니다. 따라서 서비스 주체 krbtgt/ENG.EAST.EXAMPLE.COM@SALES.WEST.EXAMPLE.COM에 대한 암호는 두 영역에서 동일해야 합니다.

3 Kerberos 구성 파일에서 원격 영역에 대한 직접 경로를 정의할 항목을 추가합니다.

이 예에서는 ENG.EAST.EXAMPLE.COM 영역의 클라이언트를 보여 줍니다. SALES.WEST.EXAMPLE.COM 영역의 적절한 정의를 가져오려면 영역 이름을 교체해야 합니다.

```
# cat /etc/krb5/krb5.conf
[libdefaults]
 .
 .
[capaths]
    ENG.EAST.EXAMPLE.COM = {
        SALES.WEST.EXAMPLE.COM = .
    }

    SALES.WEST.EXAMPLE.COM = {
         ENG.EAST.EXAMPLE.COM = .
    }
```

4 **Kerberos 구성 파일을 현재 영역의 모든 클라이언트에 복사합니다.**
영역 간 인증이 작동하려면 모든 시스템(슬레이브 KDC 및 기타 서버 포함)에 Kerberos 구성 파일(`/etc/krb5/krb5.conf`)의 새 버전이 설치되어야 합니다.

5 **두번째 영역에 대해 위 단계를 모두 반복합니다.**

Kerberos 네트워크 애플리케이션 서버 구성

네트워크 애플리케이션 서버는 `ftp`, `rcp`, `rlogin`, `rsh`, `ssh` 및 `telnet` 네트워크 응용 프로그램 중 하나 이상을 사용하여 액세스를 제공하는 호스트입니다. 몇 단계만으로 서버에서 이러한 명령의 Kerberos 버전을 사용으로 설정할 수 있습니다.

▼ Kerberos 네트워크 애플리케이션 서버 구성 방법

이 절차에서는 다음 구성 매개변수를 사용합니다.

- 애플리케이션 서버 = `boston`
- `admin` 주체 = `kws/admin`
- DNS 도메인 이름 = `example.com`
- 영역 이름 = `EXAMPLE.COM`

시작하기 전에 이 절차를 수행하려면 마스터 KDC가 구성되어 있어야 합니다. 프로세스를 완전히 테스트하려면 여러 Kerberos 클라이언트를 구성해야 합니다.

1 **서버에서 수퍼유저로 로그인합니다.**

2 **(옵션) NTP 클라이언트 또는 다른 클럭 동기화 방식을 설치합니다.**
NTP에 대한 자세한 내용은 400 페이지 “KDC와 Kerberos 클라이언트 간의 클럭 동기화”를 참조하십시오.

3 **새 서버에 대한 주체를 추가하고 서버의 keytab 파일을 업데이트합니다.**
다음 명령을 실행하면 host 주체 유무가 보고됩니다.

```
boston # klist -k |grep host
4 host/boston.example.com@EXAMPLE.COM
4 host/boston.example.com@EXAMPLE.COM
4 host/boston.example.com@EXAMPLE.COM
4 host/boston.example.com@EXAMPLE.COM
```

명령이 주체를 반환하지 않을 경우 다음 단계에 따라 새 주체를 만듭니다.

GUI Kerberos 관리 도구를 사용하여 주체를 추가하는 방법은 457 페이지 “새 Kerberos 주체를 만드는 방법”에서 설명됩니다. 다음 단계의 예에서는 명령줄을 사용하여 필요한 주체를 추가하는 방법을 보여 줍니다. 마스터 KDC를 구성할 때 만든 admin 주체 이름 중 하나로 로그인해야 합니다.

```
boston # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. 서버의 host 주체를 만듭니다.

host 주체의 용도는 다음과 같습니다.

- 원격 명령(예: rsh 및 ssh) 사용 시 트래픽 인증
- pam_krb5가 host 주체를 통해 사용자의 Kerberos 자격 증명이 신뢰할 수 있는 KDC에서 온 것인지 확인하여 KDC 스푸핑 공격을 방지하는 데 사용
- root 사용자가 root 주체 없이도 Kerberos 자격 증명을 자동으로 확보할 수 있도록 허용. 이는 공유에 Kerberos 자격 증명이 필요한 수동 NFS 마운트를 수행할 때 유용할 수 있습니다.

원격 응용 프로그램을 사용하는 트래픽을 Kerberos 서비스를 통해 인증하려는 경우 이 주체가 필요합니다. 서버에 연결된 호스트 이름이 여러 개인 경우 호스트 이름의 FQDN 형식을 사용하여 각 호스트 이름에 대한 주체를 만듭니다.

```
kadmin: addprinc -randkey host/boston.example.com
Principal "host/boston.example.com" created.
kadmin:
```

b. 서버의 keytab 파일에 서버의 host 주체를 추가합니다.

kadmin 명령이 실행되고 있지 않을 경우 /usr/sbin/kadmin -p *kws*/admin과 유사한 명령을 사용하여 다시 시작합니다.

서버에 연결된 호스트 이름이 여러 개인 경우 각 호스트 이름에 대한 keytab에 주체를 추가합니다.

```
kadmin: ktadd host/boston.example.com
Entry for principal host/boston.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/boston.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/boston.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/boston.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/boston.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

c. kadmin을 종료합니다.

```
kadmin: quit
```

▼ FTP 실행 시 Kerberos를 통한 일반 보안 서비스 사용 방법

일반 보안 서비스(GSS)를 사용하면 응용 프로그램이 인증, 무결성 및 프라이버시를 위해 간편하게 Kerberos를 사용할 수 있습니다. 다음 단계에서는 ProFTPD에 대해 GSS 서비스를 사용으로 설정하는 방법을 보여 줍니다.

1 FTP 서버에서 수퍼유저로 로그인합니다.

2 FTP 서버에 대한 주체를 추가하고 서버의 keytab 파일을 업데이트합니다.

변경 사항이 이전에 적용된 경우 이러한 단계가 필요하지 않을 수도 있습니다.

a. kadmin 명령을 시작합니다.

```
ftpserver1 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

b. FTP 서버에 대한 host 서비스 주체를 추가합니다.

```
kadmin: addprinc -randkey host/ftpserver1.example.com
```

c. 서버의 keytab 파일에 host 서비스 주체를 추가합니다.

```
kadmin: ktadd host/ftpserver1.example.com
```

3 FTP 서버에 대해 GSS를 사용으로 설정합니다.

/etc/proftpd.conf 파일을 다음과 같이 변경합니다.

```
# cat /etc/proftpd.conf
#User          ftp
#Group         ftp

User           root
Group          root

UseIPv6 off

LoadModule     mod_gss.c

GSSEngine      on
GSSKeytab      /etc/krb5/krb5.keytab
```

4 FTP 서버를 다시 시작합니다.

```
# svcadm restart network/ftp
```

Kerberos NFS 서버 구성

NFS 서비스는 UNIX 사용자 ID(UID)를 사용하여 사용자를 식별하며 GSS 자격 증명을 직접 사용할 수 없습니다. 자격 증명을 UID로 변환하려면 사용자 자격 증명을 UNIX UID에 매핑하는 자격 증명 테이블을 만들어야 할 수 있습니다. 기본 자격 증명 매핑에 대한 자세한 내용은 345 페이지 "UNIX 자격 증명과 GSS 자격 증명 간 매핑"을 참조하십시오. 이 절의 절차에서는 Kerberos NFS 서버 구성, 자격 증명 테이블 관리 및 NFS 마운트된 파일 시스템에 대한 Kerberos 보안 모드 시작에 필요한 작업을 중점적으로 다룹니다. 다음 작업 맵에서는 이 절에서 다루는 작업에 대해 설명합니다.

표 21–2 Kerberos NFS 서버 구성(작업 맵)

작업	설명	수행 방법
Kerberos NFS 서버를 구성합니다.	서버가 Kerberos 인증을 필요로 하는 파일 시스템을 공유할 수 있도록 합니다.	378 페이지 "Kerberos NFS 서버 구성 방법"
자격 증명 테이블을 만듭니다.	기본 매핑으로 충분하지 않을 경우 GSS 자격 증명과 UNIX 사용자 ID 간의 매핑을 제공하는 데 사용할 수 있는 자격 증명 테이블을 생성합니다.	380 페이지 "자격 증명 테이블을 만드는 방법"
사용자 자격 증명을 UNIX UID에 매핑하는 자격 증명 테이블을 변경합니다.	자격 증명 테이블에서 정보를 업데이트합니다.	380 페이지 "자격 증명 테이블에 단일 항목 추가 방법"
두 개의 유사 영역 간에 자격 증명 매핑을 만듭니다.	영역이 암호 파일을 공유하는 경우 영역 간에 UID를 매핑하는 방법에 대한 지침을 제공합니다.	381 페이지 "영역 간 자격 증명 매핑 제공 방법"
Kerberos 인증과 파일 시스템을 공유합니다.	Kerberos 인증이 필요하도록 보안 모드와 파일 시스템을 공유합니다.	382 페이지 "Kerberos 보안 모드가 여러 개인 보안 NFS 환경 설정 방법"

▼ Kerberos NFS 서버 구성 방법

이 절차에서는 다음 구성 매개변수가 사용됩니다.

- 영역 이름 = `EXAMPLE.COM`
- DNS 도메인 이름 = `example.com`
- NFS 서버 = `denver.example.com`
- `admin` 주체 = `kws/admin`

1 NFS 서버에서 수퍼유저로 로그인합니다.

2 Kerberos NFS 서버 구성 필수 조건을 완료합니다.

마스터 KDC를 구성해야 합니다. 프로세스를 완전히 테스트하려면 여러 클라이언트가 필요합니다.

3 **(옵션) NTP 클라이언트 또는 다른 클럭 동기화 방식을 설치합니다.**

NTP(Network Time Protocol)를 설치하여 사용할 필요가 없습니다. 하지만 인증이 성공하려면 모든 클럭이 `krb5.conf` 파일에서 `clockskew` 관계에 정의된 최대 차이 범위 내에서 KDC 서버의 시간과 동기화되어야 합니다. NTP에 대한 자세한 내용은 400 페이지 "KDC와 Kerberos 클라이언트 간의 클럭 동기화"를 참조하십시오.

4 **NFS 서버를 Kerberos 클라이언트로 구성합니다.**

384 페이지 "Kerberos 클라이언트 구성"의 지침을 따릅니다.

5 **kadmin을 시작합니다.**

457 페이지 "새 Kerberos 주체를 만드는 방법"에 설명된 대로 그래픽 Kerberos 관리 도구를 사용하여 주체를 추가할 수 있습니다. 이 작업을 수행하려면 마스터 KDC를 구성할 때 만든 `admin` 주체 이름 중 하나로 로그인해야 합니다. 하지만 다음 예에서는 명령줄을 사용하여 필요한 주체를 추가하는 방법을 보여 줍니다.

```
denver # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. **서버의 NFS 서비스 주체를 만듭니다.**

주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

시스템에서 NFS 데이터에 액세스하는 데 사용할 수 있는 각 고유 인터페이스에 대해 이 단계를 반복합니다. 호스트에 고유 이름이 있는 인터페이스가 여러 개인 경우 각 고유 이름에는 고유한 NFS 서비스 주체가 있어야 합니다.

```
kadmin: addprinc -randkey nfs/denver.example.com
Principal "nfs/denver.example.com" created.
kadmin:
```

b. **서버의 keytab 파일에 서버의 NFS 서비스 주체를 추가합니다.**

단계 a에서 만든 각 고유 서비스 주체에 대해 이 단계를 반복합니다.

```
kadmin: ktadd nfs/denver.example.com
Entry for principal nfs/denver.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs denver.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

c. **kadmin을 종료합니다.**

```
kadmin: quit
```

6 **(옵션) 필요한 경우 특수한 GSS 자격 증명 맵을 만듭니다.**

일반적으로 Kerberos 서비스는 GSS 자격 증명과 UNIX UID 간에 적절한 맵을 생성합니다. 기본 매핑은 345 페이지 "UNIX 자격 증명과 GSS 자격 증명 간 매핑"에서 설명됩니다. 기본 매핑으로 충분하지 않을 경우 자세한 내용은 380 페이지 "자격 증명 테이블을 만드는 방법"을 참조하십시오.

7 **Kerberos 보안 모드와 NFS 파일 시스템을 공유합니다.**

자세한 내용은 382 페이지 "Kerberos 보안 모드가 여러 개인 보안 NFS 환경 설정 방법"을 참조하십시오.

▼ 자격 증명 테이블을 만드는 방법

gsscred 자격 증명 테이블은 NFS 서버가 Kerberos 자격 증명을 UID에 매핑하는 데 사용합니다. 기본적으로 주체 이름의 주요 부분이 UNIX 로그인 이름과 일치됩니다. Kerberos 인증을 사용하는 NFS 서버에서 파일 시스템을 마운트할 NFS 클라이언트의 경우 기본 매핑으로 충분하지 않으면 이 테이블을 만들어야 합니다.

1 **NFS 서버에서 수퍼유저로 로그인합니다.**

2 **/etc/gss/gsscred.conf를 편집하고 보안 방식을 변경합니다.**

방식을 files로 변경합니다.

3 **gsscred 명령을 사용하여 자격 증명 테이블을 만듭니다.**

```
# gsscred -m kerberos_v5 -a
```

gsscred 명령은 svc:/system/name-service/switch:default 서비스의 passwd 항목으로 나열되는 모든 소스에서 정보를 수집합니다. 로컬 암호 항목이 자격 증명 테이블에 포함되지 않도록 하려는 경우 일시적으로 files 항목을 제거해야 할 수도 있습니다. 자세한 내용은 gsscred(1M) 매뉴얼 페이지를 참조하십시오.

▼ 자격 증명 테이블에 단일 항목 추가 방법

시작하기 전에 이 절차를 수행하려면 gsscred 테이블이 NFS 서버에서 이미 만들어져 있어야 합니다. 지침은 380 페이지 "자격 증명 테이블을 만드는 방법"을 참조하십시오.

1 **NFS 서버에서 수퍼유저로 로그인합니다.**

2 **gsscred 명령을 사용하여 자격 증명 테이블에 항목을 추가합니다.**

```
# gsscred -m mech [ -n name [ -u uid ]] -a
```

mech 사용할 보안 방식을 정의합니다.

name KDC에 정의된 대로 사용자에 대한 주체 이름을 정의합니다.

uid 암호 데이터베이스에 정의된 대로 사용자에 대한 UID를 정의합니다.

-a 주체 이름 매핑에 UID를 추가합니다.

예 21–3 자격 증명 테이블에 여러 구성 요소 주체 추가

다음 예에서는 UID 3736에 매핑된 이름이 sandy/admin인 주체에 대해 항목이 추가됩니다.

```
# gsscred -m kerberos_v5 -n sandy/admin -u 3736 -a
```

예 21–4 자격 증명 테이블에 다른 도메인의 주체 추가

다음 예에서는 UID 3736에 매핑된 이름이 sandy/admin@EXAMPLE.COM인 주체에 대해 항목이 추가됩니다.

```
# gsscred -m kerberos_v5 -n sandy/admin@EXAMPLE.COM -u 3736 -a
```

▼ 영역 간 자격 증명 매핑 제공 방법

이 절차에서는 동일한 암호 파일을 사용하는 영역 간에 적절한 자격 증명 매핑을 제공합니다. 이 예에서는 CORP.EXAMPLE.COM 및 SALES.EXAMPLE.COM 영역에서 동일한 암호 파일을 사용합니다. bob@CORP.EXAMPLE.COM 및 bob@SALES.EXAMPLE.COM에 대한 자격 증명이 동일한 UID에 매핑됩니다.

1. **클라이언트 시스템에 수퍼유저로 로그인합니다.**

2. **클라이언트 시스템에서 krb5.conf 파일에 항목을 추가합니다.**

```
# cat /etc/krb5/krb5.conf
[libdefaults]
        default_realm = CORP.EXAMPLE.COM
 .
[realms]
    CORP.EXAMPLE.COM = {
        .
        auth_to_local_realm = SALES.EXAMPLE.COM
        .
    }
```

예 21–5 동일한 암호 파일을 사용하여 영역 간에 자격 증명 매핑

이 예에서는 동일한 암호 파일을 사용하는 영역 간에 적절한 자격 증명 매핑을 제공합니다. 이 예에서는 CORP.EXAMPLE.COM 및 SALES.EXAMPLE.COM 영역에서 동일한 암호

파일을 사용합니다. bob@CORP.EXAMPLE.COM 및 bob@SALES.EXAMPLE.COM에 대한 자격 증명이 동일한 UID에 매핑됩니다. 클라이언트 시스템에서 krb5.conf 파일에 항목을 추가합니다.

```
# cat /etc/krb5/krb5.conf
[libdefaults]
        default_realm = CORP.EXAMPLE.COM
 .
[realms]
    CORP.EXAMPLE.COM = {
        .
        auth_to_local_realm = SALES.EXAMPLE.COM
        .
    }
```

일반 오류 자격 증명 매핑 문제 해결 프로세스에 대한 지원은 444 페이지 “GSS 자격 증명에서 UNIX 자격 증명으로 매핑”을 참조하십시오.

▼ Kerberos 보안 모드가 여러 개인 보안 NFS 환경 설정 방법

이 절차에서는 NFS 서버가 다른 보안 모드 또는 종류를 사용하여 보안 NFS 액세스를 제공할 수 있도록 합니다. 클라이언트가 NFS 서버와 보안 종류를 협상하는 경우 클라이언트가 액세스할 수 있는 서버에서 제공하는 첫번째 종류가 사용됩니다. 이 종류는 NFS 서버가 공유하는 파일 시스템의 모든 후속 클라이언트 요청에 사용됩니다.

1 NFS 서버에서 수퍼유저로 로그인합니다.

2 keytab 파일에 NFS 서비스 주체가 있는지 확인합니다.

klist 명령은 keytab 파일이 있는지 여부를 보고하고 주체를 표시합니다. keytab 파일이 존재하지 않거나 NFS 서비스 주체가 존재하지 않는 것으로 결과가 표시되면 378 페이지 “Kerberos NFS 서버 구성 방법”에 나오는 모든 단계의 완료를 확인해야 합니다.

```
# klist -k
Keytab name: FILE:/etc/krb5/krb5.keytab
KVNO Principal
---- --------------------------------------------------------
   3 nfs/denver.example.com@EXAMPLE.COM
   3 nfs/denver.example.com@EXAMPLE.COM
   3 nfs/denver.example.com@EXAMPLE.COM
   3 nfs/denver.example.com@EXAMPLE.COM
```

3 /etc/nfssec.conf 파일에서 Kerberos 보안 모드를 사용으로 설정합니다.

/etc/nfssec.conf 파일을 편집하고 Kerberos 보안 모드 앞에 배치된 "#"을 제거합니다.

```
# cat /etc/nfssec.conf
 .
```

```
.
#
# Uncomment the following lines to use Kerberos V5 with NFS
#
krb5            390003  kerberos_v5     default -               # RPCSEC_GSS
krb5i           390004  kerberos_v5     default integrity       # RPCSEC_GSS
krb5p           390005  kerberos_v5     default privacy         # RPCSEC_GSS
```

4 적절한 보안 모드와 파일 시스템을 공유합니다.

share -F nfs -o sec=*mode file-system*

mode 파일 시스템 공유 시 사용할 보안 모드를 지정합니다. 여러 보안 모드를 사용 중인 경우 목록의 첫번째 모드가 기본 모드로 사용됩니다.

file-system 공유할 파일 시스템에 대한 경로를 정의합니다.

명명된 파일 시스템의 파일에 액세스하려고 시도하는 모든 클라이언트에는 Kerberos 인증이 필요합니다. 파일에 액세스하려면 NFS 클라이언트의 사용자 주체가 인증되어야 합니다.

5 (옵션) 자동 마운트가 사용 중인 경우 기본 모드가 아닌 다른 보안 모드가 선택되도록 auto_master 데이터베이스를 편집합니다.

파일 시스템에 액세스하는 데 자동 마운트를 사용하고 있지 않거나 기본 보안 모드 선택을 그대로 적용할 수 있을 경우 이 절차를 따르지 않아도 됩니다.

file-system auto_home -nosuid,sec=*mode*

6 (옵션) 수동으로 mount 명령을 실행하여 기본 모드가 아닌 다른 모드를 통해 파일 시스템에 액세스합니다.

또는 mount 명령을 사용하여 보안 모드를 지정할 수도 있지만 이 대체 방법은 자동 마운트를 사용하지 않습니다.

 # mount -F nfs -o sec=*mode file-system*

예 21–6 하나의 Kerberos 보안 모드와 파일 시스템 공유

이 예에서는 Kerberos 인증이 성공해야만 NFS 서비스를 통해 파일에 액세스할 수 있습니다.

```
# share -F nfs -o sec=krb5 /export/home
```

예 21–7 여러 Kerberos 보안 모드와 파일 시스템 공유

이 예에서는 세 개의 모든 Kerberos 보안 모드가 선택되었습니다. 사용되는 모드는 클라이언트와 NFS 서버 간에 협상됩니다. 명령의 첫번째 모드가 실패하면 다음 모드가 시도됩니다. 자세한 내용은 nfssec(5) 매뉴얼 페이지를 참조하십시오.

```
# share -F nfs -o sec=krb5:krb5i:krb5p /export/home
```

Kerberos 클라이언트 구성

Kerberos 클라이언트에는 Kerberos 서비스를 사용해야 하는 네트워크의 KDC 서버가 아닌 호스트가 포함됩니다. 이 절에서는 root 인증을 사용하여 NFS 파일 시스템을 마운트하는 자세한 내용과 함께 Kerberos 클라이언트 설치 절차를 제공합니다.

Kerberos 클라이언트 구성(작업 맵)

다음 작업 맵에는 Kerberos 클라이언트 설정과 관련된 모든 절차가 포함되어 있습니다. 각 행에서는 작업 식별자, 해당 작업을 수행하고자 하는 이유 설명, 작업에 대한 링크를 차례로 제공합니다.

작업	설명	수행 방법
Kerberos 클라이언트 설치 프로파일을 설정합니다.	Kerberos 클라이언트를 자동으로 설치하는 데 사용할 수 있는 클라이언트 설치 프로파일을 생성합니다.	385 페이지 “Kerberos 클라이언트 설치 프로파일을 만드는 방법”
Kerberos 클라이언트를 구성합니다.	수동으로 Kerberos 클라이언트를 설치합니다. 각 클라이언트 설치에 고유 설치 매개변수가 필요한 경우 이 절차를 사용하십시오.	390 페이지 “수동으로 Kerberos 클라이언트를 구성하는 방법”
	자동으로 Kerberos 클라이언트를 설치합니다. 각 클라이언트에 대한 설치 매개변수가 동일한 경우 이 절차를 사용하십시오.	385 페이지 “자동으로 Kerberos 클라이언트를 구성하는 방법”
	대화식으로 Kerberos 클라이언트를 설치합니다. 설치 매개변수 중 일부만 변경해야 할 경우 이 절차를 사용하십시오.	386 페이지 “대화식으로 Kerberos 클라이언트를 구성하는 방법”
	자동으로 Active Directory 서버의 Kerberos 클라이언트를 설치합니다.	389 페이지 “Active Directory 서버에 대한 Kerberos 클라이언트 구성 방법”
클라이언트가 root 사용자로 NFS 파일 시스템에 액세스할 수 있도록 합니다.	클라이언트가 root 액세스를 사용하여 공유되는 NFS 파일 시스템을 마운트할 수 있도록 클라이언트에서 root 주체를 만듭니다. 또한 cron 작업을 실행할 수 있도록 클라이언트가 NFS 파일 시스템에 대한 비대화식 root 액세스를 설정할 수 있도록 합니다.	396 페이지 “Kerberos로 보호된 NFS 파일 시스템에 root 사용자로 액세스하는 방법”
클라이언트 TGT(티켓 부여 티켓)를 발행한 KDC의 확인을 사용 안함으로 설정합니다.	로컬 keytab 파일에 host 주체가 저장되지 않은 클라이언트가 TGT를 발행한 KDC와 host 주체를 발행한 서버가 동일한지 확인하는 보안 점검을 건너 뛸 수 있도록 합니다.	395 페이지 “TGT(티켓 부여 티켓) 확인을 사용 안함으로 설정하는 방법”

▼ Kerberos 클라이언트 설치 프로파일을 만드는 방법

이 절차에서는 Kerberos 클라이언트를 설치할 때 사용할 수 있는 kclient 프로파일을 만듭니다. kclient 프로파일을 사용하면 입력 오류 발생 가능성이 줄어듭니다. 또한 이 프로파일을 사용하면 대화식 프로세스에 비해 사용자 개입이 줄어듭니다.

1 **수퍼유저로 로그인합니다.**

2 **kclient 설치 프로파일을 만듭니다.**

샘플 kclient 프로파일은 다음과 같이 표시될 수 있습니다.

```
client# cat /net/denver.example.com/export/install/profile
REALM EXAMPLE.COM
KDC kdc1.example.com
ADMIN clntconfig
FILEPATH /net/denver.example.com/export/install/krb5.conf
NFS 1
DNSLOOKUP none
```

▼ 자동으로 Kerberos 클라이언트를 구성하는 방법

시작하기 전에 이 절차에서는 설치 프로파일을 사용합니다. 385 페이지 "Kerberos 클라이언트 설치 프로파일을 만드는 방법"을 참조하십시오.

1 **관리자가 되거나 Kerberos Client Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.**

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **kclient 설치 스크립트를 실행합니다.**

프로세스를 완료하려면 `clntconfig` 주체에 대한 암호를 제공해야 합니다.

```
client# /usr/sbin/kclient -p /net/denver.example.com/export/install/profile

Starting client setup
-------------------------------------------------

kdc1.example.com

Setting up /etc/krb5/krb5.conf.

Obtaining TGT for clntconfig/admin ...
Password for clntconfig/admin@EXAMPLE.COM:      <Type the password>

nfs/client.example.com entry ADDED to KDC database.
nfs/client.example.com entry ADDED to keytab.

host/client.example.com entry ADDED to KDC database.
host/client.example.com entry ADDED to keytab.
```

```
Copied /net/denver.example.com/export/install/krb5.conf.

-------------------------------------------------
Setup COMPLETE.

client#
```

예 21–8 명령줄 대체를 사용하여 자동으로 Kerberos 클라이언트 구성

다음 예에서는 설치 프로파일에 설정된 DNSARG 및 KDC 매개변수를 대체합니다.

```
# /usr/sbin/kclient -p /net/denver.example.com/export/install/profile\
-d dns_fallback -k kdc2.example.com

Starting client setup
-------------------------------------------------

kdc1.example.com

Setting up /etc/krb5/krb5.conf.

Obtaining TGT for clntconfig/admin ...
Password for clntconfig/admin@EXAMPLE.COM:      <Type the password>

nfs/client.example.com entry ADDED to KDC database.
nfs/client.example.com entry ADDED to keytab.

host/client.example.com entry ADDED to KDC database.
host/client.example.com entry ADDED to keytab.

Copied /net/denver.example.com/export/install/krb5.conf.

-------------------------------------------------
Setup COMPLETE.

client#
```

▼ 대화식으로 Kerberos 클라이언트를 구성하는 방법

이 절차에서는 설치 프로파일 없이 kclient 설치 유틸리티를 사용합니다. Oracle Solaris 11 릴리스에서는 kclient 유틸리티가 향상되어 사용이 간편해지고 Active Directory 서버에서 작동할 수 있게 되었습니다. 자세한 내용은 389 페이지 “Active Directory 서버에 대한 Kerberos 클라이언트 구성 방법”을 참조하십시오. 이전 릴리스에서 kclient를 실행하는 예는 예 21–10을 참조하십시오.

1 관리자가 되거나 Kerberos Client Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 `kclient` 설치 스크립트를 실행합니다.

다음 정보를 제공해야 합니다.

- Kerberos 영역 이름
- KDC 마스터 호스트 이름
- KDC 슬레이브 호스트 이름
- 로컬 영역에 매핑할 도메인
- Kerberos 인증에 사용할 PAM 서비스 이름 및 옵션

a. KDC 서버가 Oracle Solaris 릴리스를 실행하고 있지 않은지 여부를 지정합니다.

이 시스템이 Oracle Solaris 릴리스를 실행하고 있지 않은 KDC 서버의 클라이언트인 경우 KDC를 실행하고 있는 서버의 유형을 정의해야 합니다. 사용 가능한 서버는 Microsoft Active Directory, MIT KDC 서버, Heimdal KDC 서버 및 Shishi KDC 서버입니다.

b. Kerberos 조회에 DNS를 사용해야 하는지 여부를 선택합니다.

Kerberos 조회에 DNS를 사용할 경우 사용하려는 DNS 조회 옵션을 입력해야 합니다. 유효한 옵션은 `dns_lookup_kdc`, `dns_lookup_realm` 및 `dns_fallback`입니다. 이러한 값에 대한 자세한 내용은 `krb5.conf(4)` 매뉴얼 페이지를 참조하십시오.

c. Kerberos 영역의 이름 및 마스터 KDC 호스트 이름을 정의합니다.

이 정보는 `/etc/krb5/krb5.conf` 구성 파일에 추가됩니다.

d. 슬레이브 KDC가 존재하는지 여부를 선택합니다.

영역에 슬레이브 KDC가 있을 경우 슬레이브 KDC 호스트 이름을 입력해야 합니다. 이 정보는 클라이언트의 구성 파일에 추가 KDC 항목을 만드는 데 사용됩니다.

e. 서비스 또는 호스트 키가 필요한지 여부를 지정합니다.

일반적으로 클라이언트 시스템이 Kerberos화된 서비스를 호스트하는 경우 서비스 또는 호스트 키가 필요하지 않습니다.

f. 클라이언트가 클러스터의 구성원인지 여부를 지정합니다.

클라이언트가 클러스터의 구성원인 경우 클러스터의 논리적 이름을 제공해야 합니다. 논리적 호스트 이름은 서비스 키를 만들 때 사용되며, 클러스터에서 Kerberos 서비스를 호스트하는 경우 필요합니다.

g. 현재 영역에 매핑할 도메인 또는 호스트를 식별합니다.

이 매핑은 다른 도메인이 클라이언트의 기본 영역에 속할 수 있도록 합니다.

h. 클라이언트가 Kerberos화된 NFS를 사용할지 여부를 지정합니다.

클라이언트가 Kerberos를 사용하는 NFS 서비스를 호스트할 경우 NFS 서비스 키를 만들어야 합니다.

i. /etc/pam.conf 파일을 업데이트해야 할지 여부를 지정합니다.

이를 통해 인증에 Kerberos를 사용할 PAM 서비스를 설정할 수 있습니다. 서비스 이름 및 Kerberos 인증을 사용할 방식을 나타내는 플래그를 입력해야 합니다. 유효한 플래그 옵션은 다음과 같습니다.

- `first` – Kerberos 인증을 먼저 사용하고 Kerberos 인증을 실패한 경우에만 UNIX를 사용합니다.
- `only` – Kerberos 인증만 사용합니다.
- `optional` – 선택적으로 Kerberos 인증을 사용합니다.

j. 마스터 /etc/krb5/krb5.conf 파일을 복사해야 할지 여부를 선택합니다.

이 옵션을 설정하면 `kclient`에 대한 인수가 충분하지 않을 경우 특정 구성 정보를 사용할 수 있습니다.

예 21–9 kclient 설치 유틸리티 실행

```
client# /usr/sbin/kclient

Starting client setup
-----------------------------------------------------

Is this a client of a non-Solaris KDC ? [y/n]: n
        No action performed.
Do you want to use DNS for kerveros lookups ? [y/n]: n
        No action performed.
Enter the Kerberos realm: EXAMPLE.COM
Specify the KDC hostname for the above realm: kdc1.example.com

Note, this system and the KDC's time must be within 5 minutes of each other for
Kerberos to function. Both systems should run some form of time synchronization
system like Network Time Protocol (NTP).
Do you have any slave KDC(s) ? [y/n]: y
Enter a comma-separated list of slave KDC host names: kdc2.example.com

Will this client need service keys ? [y/n]: n
        No action performed.
Is this client a member of a cluster that uses a logical host name ? [y/n]: n
        No action performed.
Do you have multiple domains/hosts to map to realm ? [y/n]: y
Enter a comma-separated list of domain/hosts to map to the default realm: engineering.example.com, \
        example.com

Setting up /etc/krb5/krb5.conf.

Do you plan on doing Kerberized nfs ? [y/n]: y
Do you want to update /etc/pam.conf ? [y/n]: y
Enter a comma-separated list of PAM service names in the following format:
service:{first|only|optional}: xscreensaver:first
Configuring /etc/pam.conf.

Do you want to copy over the master krb5.conf file ? [y/n]: n
        No action performed.
```

```
-------------------------------------------------
Setup COMPLETE.
```

예 21–10 Oracle Solaris 10 릴리스에서 kclient 설치 유틸리티 실행

다음 출력에서는 kclient 명령 실행 결과를 보여 줍니다.

```
client# /usr/sbin/kclient

Starting client setup
-------------------------------------------------

Do you want to use DNS for kerberos lookups ? [y/n]: n
        No action performed.
Enter the Kerberos realm: EXAMPLE.COM
Specify the KDC hostname for the above realm: kdc1.example.com

Setting up /etc/krb5/krb5.conf.

Enter the krb5 administrative principal to be used: clntconfig/admin
Obtaining TGT for clntconfig/admin ...
Password for clntconfig/admin@EXAMPLE.COM:      <Type the password>
Do you plan on doing Kerberized nfs ? [y/n]: n

host/client.example.com entry ADDED to KDC database.
host/client.example.com entry ADDED to keytab.

Do you want to copy over the master krb5.conf file ? [y/n]: y
Enter the pathname of the file to be copied: \
/net/denver.example.com/export/install/krb5.conf

Copied /net/denver.example.com/export/install/krb5.conf.

-------------------------------------------------
Setup COMPLETE !
#
```

▼ Active Directory 서버에 대한 Kerberos 클라이언트 구성 방법

이 절차에서는 설치 프로파일 없이 kclient 설치 유틸리티를 사용합니다.

1. **수퍼유저로 로그인합니다.**
2. **(옵션) 클라이언트에 대한 DNS 리소스 레코드 생성을 사용으로 설정합니다.**

```
client# sharectl set -p ddns_enable=true smb
```

3 **kclient 유틸리티를 실행합니다.**

-T 옵션을 사용하면 KDC 서버 유형이 선택됩니다. 이 경우 Active Directory 서버가 선택됩니다.

```
client# kclient -T ms_ad
```

기본적으로 관리자 주체에 대한 암호를 제공해야 합니다.

예 21–11 kclient를 사용하여 Active Directory 서버에 대한 Kerberos 클라이언트 구성

다음 출력에서는 ms_ad(Microsoft Active Directory) 서버 유형 인수를 사용하여 kclient 명령을 실행한 결과를 보여 줍니다. 이름이 EXAMPLE.COM인 Active Directory 도메인에 클라이언트가 결합됩니다.

```
client# /usr/sbin/kclient -T ms_ad

Starting client setup
-----------------------------------------------------

Attempting to join 'CLIENT' to the 'EXAMPLE.COM' domain.
Password for Administrator@EXAMPLE.COM:      <Type the password>
Forest name found: example.com
Looking for local KDCs, DCs and global catalog servers (SVR RRs).

Setting up /etc/krb5/krb5.conf

Creating the machine account in AD via LDAP.
-----------------------------------------------------
Setup COMPLETE.
#
```

▼ 수동으로 Kerberos 클라이언트를 구성하는 방법

이 절차에서는 다음 구성 매개변수가 사용됩니다.

- 영역 이름 = EXAMPLE.COM
- DNS 도메인 이름 = example.com
- 마스터 KDC = kdc1.example.com
- 슬레이브 KDC = kdc2.example.com
- NFS 서버 = denver.example.com
- 클라이언트 = client.example.com
- admin 주체 = kws/admin
- 사용자 주체 = mre
- 온라인 도움말 URL = http://download.oracle.com/docs/cd/E23824_01/html/821-1456/aadmin-23.html

주 – 349 페이지 “그래픽 Kerberos 관리 도구의 온라인 도움말 URL”에 설명된 대로 절을 가리키도록 URL을 조정하십시오.

1 **수퍼유저로 로그인합니다.**

2 **Kerberos 구성 파일(`krb5.conf`)을 편집합니다.**

Kerberos 기본 버전에서 파일을 변경하려면 영역 이름 및 서버 이름을 변경해야 합니다. `gkadmin`에 대한 도움말 파일의 경로도 식별해야 합니다.

```
kdc1 # cat /etc/krb5/krb5.conf
[libdefaults]
        default_realm = EXAMPLE.COM

[realms]
        EXAMPLE.COM = {
        kdc = kdc1.example.com
        kdc = kdc2.example.com
        admin_server = kdc1.example.com
        }

[domain_realm]
        .example.com = EXAMPLE.COM
#
# if the domain name and realm name are equivalent,
# this entry is not needed
#
[logging]
        default = FILE:/var/krb5/kdc.log
        kdc = FILE:/var/krb5/kdc.log

[appdefaults]
    gkadmin = {
        help_url = http://download.oracle.com/docs/cd/E23824_01/html/821-1456/aadmin-23.html
```

주 – 암호화 유형을 제한하려는 경우 `default_tkt_enctypes` 또는 `default_tgs_enctypes` 행을 설정할 수 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 513 페이지 “Kerberos 암호화 유형 사용”을 참조하십시오.

3 **(옵션) KDC를 찾는 데 사용되는 프로세스를 변경합니다.**

기본적으로 Kerberos 영역과 KDC 간의 매핑은 다음 순서로 확인됩니다.

- `krb5.conf`의 `realms` 섹션에 있는 정의
- DNS에서 SRV 레코드 조회

`dns_lookup_kdc` 또는 `dns_fallback`을 `krb5.conf` 파일의 `libdefaults` 섹션에 추가하여 이 동작을 변경할 수 있습니다. 자세한 내용은 `krb5.conf(4)` 매뉴얼 페이지를 참조하십시오. 항상 참조가 먼저 시도됩니다.

4 **(옵션) 호스트에 대한 영역을 확인하는 데 사용되는 프로세스를 변경합니다.**

기본적으로 호스트와 영역 간의 매핑은 다음 순서로 확인됩니다.

- KDC가 참조를 지원하는 경우 KDC가 클라이언트에 호스트가 속한 영역을 알릴 수 있음
- `krb5.conf` 파일의 `domain_realm` 정의
- 호스트의 DNS 도메인 이름
- 기본 영역

`dns_lookup_kdc` 또는 `dns_fallback`을 `krb5.conf` 파일의 `libdefaults` 섹션에 추가하여 이 동작을 변경할 수 있습니다. 자세한 내용은 `krb5.conf(4)` 매뉴얼 페이지를 참조하십시오. 항상 참조가 먼저 시도됩니다.

5 (옵션) NTP 또는 다른 클럭 동기화 방식을 사용하여 클라이언트의 클럭을 마스터 KDC의 클럭과 동기화합니다.

NTP(Network Time Protocol)를 설치하여 사용할 필요가 없습니다. 하지만 인증이 성공하려면 모든 클럭이 `krb5.conf` 파일의 `clockskew` 관계에 정의된 최대 차이 범위 내에서 KDC 서버의 시간과 동기화되어야 합니다. NTP에 대한 자세한 내용은 400 페이지 “KDC와 Kerberos 클라이언트 간의 클럭 동기화”를 참조하십시오.

6 `kadmin`을 시작합니다.

457 페이지 “새 Kerberos 주체를 만드는 방법”에 설명된 대로 그래픽 Kerberos 관리 도구를 사용하여 주체를 추가할 수 있습니다. 이 작업을 수행하려면 마스터 KDC를 구성할 때 만든 `admin` 주체 이름 중 하나로 로그인해야 합니다. 하지만 다음 예에서는 명령줄을 사용하여 필요한 주체를 추가하는 방법을 보여 줍니다.

```
denver # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. (옵션) 사용자 주체가 존재하지 않을 경우 사용자 주체를 만듭니다.

이 호스트에 연결된 사용자에게 주체가 지정되지 않은 경우에만 사용자 주체를 만들어야 합니다.

```
kadmin: addprinc mre
Enter password for principal mre@EXAMPLE.COM:      <Type the password>
Re-enter password for principal mre@EXAMPLE.COM:      <Type it again>
kadmin:
```

b. (옵션) `root` 주체를 만들고 서버의 keytab 파일에 주체를 추가합니다.

클라이언트가 NFS 서비스를 사용하여 마운트된 파일 시스템에 대해 `root` 액세스를 실행할 수 있도록 이 단계를 수행해야 합니다. cron 작업을 `root`로 실행하는 등 비대화식 `root` 액세스가 필요한 경우에도 이 단계를 수행해야 합니다.

클라이언트가 NFS 서비스를 사용하여 마운트된 원격 파일 시스템에 대한 `root` 액세스를 필요로 하지 않을 경우 이 단계를 건너 뛸 수 있습니다. 영역 차원의 `root` 주체가 만들어지지 않도록 하려면 `root` 주체가 두번째 구성 요소(Kerberos

클라이언트 시스템의 호스트 이름)와 쌍으로 구성된 구성 요소 주체여야 합니다. 주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

```
kadmin: addprinc -randkey root/client.example.com
Principal "root/client.example.com" created.
kadmin: ktadd root/client.example.com
Entry for principal root/client.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

c. host 주체를 만들고 서버의 keytab 파일에 주체를 추가합니다.

host 주체는 원격 액세스 서비스가 인증을 제공하는 데 사용합니다. 이 주체는 keytab 파일에 자격 증명이 없을 경우 root가 자격 증명을 확보할 수 있도록 합니다.

```
kadmin: addprinc -randkey host/denver.example.com
Principal "host/denver.example.com@EXAMPLE.COM" created.
kadmin: ktadd host/denver.example.com
Entry for principal host/denver.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

d. (옵션) 서버의 keytab 파일에 서버의 NFS 서비스 주체를 추가합니다.

클라이언트가 Kerberos 인증을 사용하여 NFS 파일 시스템에 액세스해야 하는 경우에만 이 단계를 수행해야 합니다.

```
kadmin: ktadd nfs/denver.example.com
Entry for principal nfs/denver.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

e. kadmin을 종료합니다.

```
kadmin: quit
```

7 (옵션) NFS에서 Kerberos를 사용으로 설정합니다.

a. /etc/nfssec.conf 파일에서 Kerberos 보안 모드를 사용으로 설정합니다.

/etc/nfssec.conf 파일을 편집하고 Kerberos 보안 모드 앞에 배치된 "#"을 제거합니다.

```
# cat /etc/nfssec.conf
 .
 .
#
# Uncomment the following lines to use Kerberos V5 with NFS
#
krb5            390003  kerberos_v5     default -               # RPCSEC_GSS
krb5i           390004  kerberos_v5     default integrity       # RPCSEC_GSS
krb5p           390005  kerberos_v5     default privacy         # RPCSEC_GSS
```

b. DNS를 사용으로 설정합니다.

svc:/network/dns/client:default 서비스가 사용으로 설정되지 않은 경우 사용으로 설정합니다. 자세한 내용은 resolv.conf(4) 매뉴얼 페이지를 참조하십시오.

c. gssd 서비스를 다시 시작합니다.

```
# svcadm restart network/rpc/gss
```

8 클라이언트가 TGT를 자동으로 갱신하도록 하거나 사용자에게 Kerberos 티켓 만료를 경고하려면 /etc/krb5/warn.conf 파일에서 항목을 만듭니다.

자세한 내용은 warn.conf(4) 매뉴얼 페이지를 참조하십시오.

예 21–12 비Solaris KDC를 사용하는 Kerberos 클라이언트 설정

비Solaris KDC에서 작동하도록 Kerberos 클라이언트를 설정할 수 있습니다. 이 경우 /etc/krb5/krb5.conf 파일의 realms 섹션에 행이 포함되어야 합니다. 이 행은 클라이언트가 Kerberos 암호 변경 서버와 통신하는 동안 사용되는 프로토콜을 변경합니다. 이 행의 형식은 다음과 같습니다.

```
[realms]
                EXAMPLE.COM = {
                kdc = kdc1.example.com
                kdc = kdc2.example.com
                admin_server = kdc1.example.com
                kpasswd_protocol = SET_CHANGE
        }
```

예 21–13 호스트 및 도메인 이름과 Kerberos 영역 간의 매핑에 대한 DNS TXT 레코드

```
@ IN SOA kdc1.example.com root.kdc1.example.com (
                                1989020501   ;serial
                                10800        ;refresh
```

```
                                3600         ;retry
                                3600000      ;expire
                                86400 )      ;minimum

                          IN    NS      kdc1.example.com.
kdc1                      IN    A       192.146.86.20
kdc2                      IN    A       192.146.86.21

_kerberos.example.com.              IN      TXT     "EXAMPLE.COM"
_kerberos.kdc1.example.com.         IN      TXT     "EXAMPLE.COM"
_kerberos.kdc2.example.com.         IN      TXT     "EXAMPLE.COM"
```

예 21–14 Kerberos 서버 위치에 대한 DNS SRV 레코드

이 예에서는 KDC, `admin` 서버 및 `kpasswd` 서버 각각에 대한 레코드를 정의합니다.

```
@ IN SOA kdc1.example.com root.kdc1.example.com (
                                1989020501   ;serial
                                10800        ;refresh
                                3600         ;retry
                                3600000      ;expire
                                86400 )      ;minimum

                                   IN      NS      kdc1.example.com.
kdc1                               IN      A       192.146.86.20
kdc2                               IN      A       192.146.86.21

_kerberos._udp.EXAMPLE.COM         IN      SRV 0 0 88  kdc2.example.com
_kerberos._tcp.EXAMPLE.COM         IN      SRV 0 0 88  kdc2.example.com
_kerberos._udp.EXAMPLE.COM         IN      SRV 1 0 88  kdc1.example.com
_kerberos._tcp.EXAMPLE.COM         IN      SRV 1 0 88  kdc1.example.com
_kerberos-adm._tcp.EXAMPLE.COM     IN      SRV 0 0 749 kdc1.example.com
_kpasswd._udp.EXAMPLE.COM          IN      SRV 0 0 749 kdc1.example.com
```

▼ TGT(티켓 부여 티켓) 확인을 사용 안함으로 설정하는 방법

이 절차에서는 로컬 `/etc/krb5/krb5.keytab` 파일에 저장된 host 주체의 KDC와 TGT(티켓 부여 티켓)를 발행한 KDC가 동일한지 확인하는 보안 점검을 사용 안함으로 설정합니다. 이 점검은 DNS 스푸핑 공격을 방지합니다. 하지만 일부 클라이언트 구성의 경우 host 주체를 사용하지 못할 수 있으므로 클라이언트가 작동하려면 이 점검을 사용 안함으로 설정해야 합니다. 다음은 이 점검을 사용 안함으로 설정해야 할 구성입니다.

- 클라이언트 IP 주소는 동적으로 지정됩니다. DHCP 클라이언트를 예로 들 수 있습니다.
- 클라이언트는 서비스를 호스트하도록 구성되지 않으므로 만들어진 host 주체가 없습니다.
- 호스트 키는 클라이언트에 저장되지 않습니다.

1 수퍼유저로 로그인합니다.

2 **krb5.conf 파일을 변경합니다.**

verify_ap_req_nofail 옵션이 false로 설정된 경우 TGT 확인 프로세스가 사용으로 설정되지 않습니다. 이 옵션에 대한 자세한 내용은 krb5.conf(4) 매뉴얼 페이지를 참조하십시오.

```
client # cat /etc/krb5/krb5.conf
[libdefaults]
        default_realm = EXAMPLE.COM
        verify_ap_req_nofail = false
  ...
```

주 – verify_ap_req_nofail 옵션은 krb5.conf 파일의 [libdefaults] 또는 [realms] 섹션에 입력할 수 있습니다. [libdefaults] 섹션에 옵션을 입력할 경우 모든 영역에 설정이 사용됩니다. [realms] 섹션에 옵션을 입력할 경우 정의된 영역에만 설정이 적용됩니다.

▼ Kerberos로 보호된 NFS 파일 시스템에 root 사용자로 액세스하는 방법

이 절차를 수행하면 클라이언트가 Kerberos 인증을 필요로 하는 NFS 파일 시스템에 root ID 권한으로 액세스할 수 있습니다. 특히 NFS 파일 시스템이 -o sec=krb5,root=client1.sun.com 등의 옵션과 공유되는 경우 이 절차를 수행하십시오.

- **kadmin을 시작합니다.**

  457 페이지 "새 Kerberos 주체를 만드는 방법"에 설명된 대로 GUI Kerberos 관리 도구를 사용하여 주체를 추가할 수 있습니다. 이 작업을 수행하려면 마스터 KDC를 구성할 때 만든 admin 주체 이름 중 하나로 로그인해야 합니다. 하지만 다음 예에서는 명령줄을 사용하여 필요한 주체를 추가하는 방법을 보여 줍니다.

  ```
  denver # /usr/sbin/kadmin -p kws/admin
  Enter password:      <Type kws/admin password>
  kadmin:
  ```

  a. **NFS 클라이언트에 대한 root 주체를 만듭니다.**

  이 주체는 Kerberos 인증이 필요한 NFS 마운트된 파일 시스템에 대해 root와 동등한 액세스를 제공하는 데 사용됩니다. 영역 차원의 root 주체가 만들어지지 않도록 하려면 root 주체가 두번째 구성 요소(Kerberos 클라이언트 시스템의 호스트 이름)와 쌍으로 구성된 구성 요소 주체여야 합니다. 주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

  ```
  kadmin: addprinc -randkey root/client.example.com
  Principal "root/client.example.com" created.
  kadmin:
  ```

b. 서버의 keytab 파일에 root 주체를 추가합니다.

클라이언트가 NFS 서비스를 사용하여 마운트된 파일 시스템에 대해 root 액세스를 실행할 수 있도록 root 주체를 추가한 경우 이 단계를 수행해야 합니다. cron 작업을 root로 실행하는 등 비대화식 root 액세스가 필요한 경우에도 이 단계를 수행해야 합니다.

```
kadmin: ktadd root/client.example.com
Entry for principal root/client.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

c. kadmin을 종료합니다.

```
kadmin: quit
```

▼ Kerberos 영역에서 사용자의 자동 마이그레이션을 구성하는 방법

Kerberos 주체가 없는 사용자는 기존 Kerberos 영역으로 자동 마이그레이션될 수 있습니다. 마이그레이션을 수행하려면 /etc/pam.conf의 서비스 인증 스택에서 pam_krb5_migrate 모듈을 스택으로 구성하여 사용 중인 서비스에 PAM 프레임워크를 사용합니다.

이 예에서는 gdm 및 other PAM 서비스 이름이 자동 마이그레이션을 사용하도록 구성됩니다. 사용되는 구성 매개변수는 다음과 같습니다.

- 영역 이름 = EXAMPLE.COM
- 마스터 KDC = kdc1.example.com
- 마이그레이션 서비스를 호스트하는 시스템 = server1.example.com
- 마이그레이션 서비스 주체 = host/server1.example.com

시작하기 전에 server1을 EXAMPLE.COM 영역의 Kerberos 클라이언트로 설정합니다. 자세한 내용은 384 페이지 “Kerberos 클라이언트 구성”을 참조하십시오.

1 수퍼유저로 로그인합니다.

2 server1에 대한 호스트 서비스 주체가 존재하는지 여부를 확인합니다.

server1의 keytab 파일에 있는 호스트 서비스 주체는 마스터 KDC에 대해 서버를 인증하는 데 사용됩니다.

```
server1 # klist -k
Keytab name: FILE:/etc/krb5/krb5.keytab
    KVNO Principal
    ---- -------------------------------------------------
       3 host/server1.example.com@EXAMPLE.COM
       3 host/server1.example.com@EXAMPLE.COM
       3 host/server1.example.com@EXAMPLE.COM
       3 host/server1.example.com@EXAMPLE.COM
```

3 PAM 구성 파일을 변경합니다.

a. gdm 서비스에 대한 항목을 추가합니다.

```
# cat /etc/pam.conf
 .
 .
#
# gdm service
#
gdm       auth requisite          pam_authtok_get.so.1
gdm       auth required           pam_dhkeys.so.1
gdm       auth required           pam_unix_cred.so.1
gdm       auth sufficient         pam_krb5.so.1
gdm       auth requisite          pam_unix_auth.so.1
gdm       auth optional           pam_krb5_migrate.so.1
```

b. (옵션) 필요한 경우 강제로 암호를 즉시 변경합니다.

강제로 Kerberos 암호를 즉시 변경하려는 경우 새로 만들어진 Kerberos 계정의 암호 만료 시간을 현재 시간으로 설정할 수 있습니다. 만료 시간을 현재 시간으로 설정하려면 pam_krb5_migrate 모듈을 사용하는 행에 expire_pw 옵션을 추가합니다. 자세한 내용은 pam_krb5_migrate(5) 매뉴얼 페이지를 참조하십시오.

```
# cat /etc/pam.conf
 .
 .
gdm      auth optional           pam_krb5_migrate.so.1 expire_pw
```

c. 계정 스택에 pam_krb5 모듈을 추가합니다.

그러면 Kerberos의 암호가 만료될 때 액세스가 차단됩니다.

```
# cat /etc/pam.conf
 .
 .
#
# Default definition for Account management
# Used when service name is not explicitly mentioned for account management
#
other   account requisite       pam_roles.so.1
other   account required        pam_krb5.so.1
other   account required        pam_unix_account.so.1
```

d. 암호 스택에 pam_krb5 모듈을 추가합니다.

그러면 암호가 만료될 때 암호가 업데이트됩니다.

```
# cat /etc/pam.conf
 .
 .
#
# Default definition for Password management
# Used when service name is not explicitly mentioned for password management
#
other   password required       pam_dhkeys.so.1
other   password requisite      pam_authtok_get.so.1
other   password requisite      pam_authtok_check.so.1
other   password sufficient     pam_krb5.so.1
other   password required       pam_authtok_store.so.1
```

4 마스터 KDC에서 액세스 제어 파일을 업데이트합니다.

다음 항목은 root 사용자를 제외한 모든 사용자에 대한 host/server1.example.com 서비스 주체에 마이그레이션 및 조회 권한을 허가합니다. 마이그레이션되지 않아야 하는 사용자는 U 권한을 사용하여 kadm5.acl 파일에 나열되어야 합니다. 이러한 항목은 전체 또는 ui 허가 항목 앞에 와야 합니다. 자세한 내용은 kadm5.acl(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/kadm5.acl
host/server1.example.com@EXAMPLE.COM U root
host/server1.example.com@EXAMPLE.COM ui *
*/admin@EXAMPLE.COM *
```

5 마스터 KDC에서 Kerberos 관리 데몬을 다시 시작합니다.

이 단계를 수행하면 kadmind 데몬이 새 kadm5.acl 항목을 사용할 수 있습니다.

```
kdc1 # svcadm restart network/security/kadmin
```

6 마스터 KDC에서 pam.conf 파일에 항목을 추가합니다.

다음 항목은 kadmind 데몬이 k5migrate PAM 서비스를 사용하고 마이그레이션해야 할 계정에 대해 UNIX 사용자 암호를 검증할 수 있도록 합니다.

```
# grep k5migrate /etc/pam.conf
k5migrate       auth    required        pam_unix_auth.so.1
k5migrate       account required        pam_unix_account.so.1
```

▼ 계정 잠금 구성 방법

- **kadmin을 시작합니다.**

```
denver # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. 계정 잠금 매개변수를 사용하여 정책을 만듭니다.

다음 예에서는 이름이 `default`인 정책을 만드는 데 `add_policy` 하위 명령이 사용됩니다. `300`초의 범위 동안 인증이 세 번 실패하면 `900`초 동안의 계정 잠금이 트리거됩니다.

```
kadmin: add_policy -maxfailure 3 -failurecountinterval "300 seconds"\
-lockoutduration "900 seconds" default
```

b. kadmin을 종료합니다.

```
kadmin: quit
```

예 21–15 잠긴 주체의 잠금 해제

다음 예에서는 사용자 주체의 잠금이 해제됩니다.

```
# kadmin
kadmin: add_policy -unlock principal
```

KDC와 Kerberos 클라이언트 간의 클럭 동기화

Kerberos 인증 시스템에 참여하는 모든 호스트의 내부 클럭은 지정된 최대 시간 범위(**클럭 불균형**이라고 함) 내에서 동기화되어야 합니다. 이 요구 사항을 충족하면 다른 Kerberos 보안 점검이 제공됩니다. 참여 호스트 간에 클럭 불균형이 초과되면 클라이언트 요청이 거부됩니다.

또한 클럭 불균형은 재생된 요청을 인식 및 거부하기 위해 애플리케이션 서버가 모든 Kerberos 프로토콜 메시지를 추적해야 할 기간을 결정합니다. 따라서 클럭 불균형 값이 클수록 애플리케이션 서버가 수집해야 할 정보가 많아집니다.

최대 클럭 불균형의 기본값은 300초(5분)입니다. `krb5.conf` 파일의 `libdefaults` 섹션에서 이 기본값을 변경할 수 있습니다.

주 – 보안상 클럭 불균형을 300초 이상으로 늘리지 마십시오.

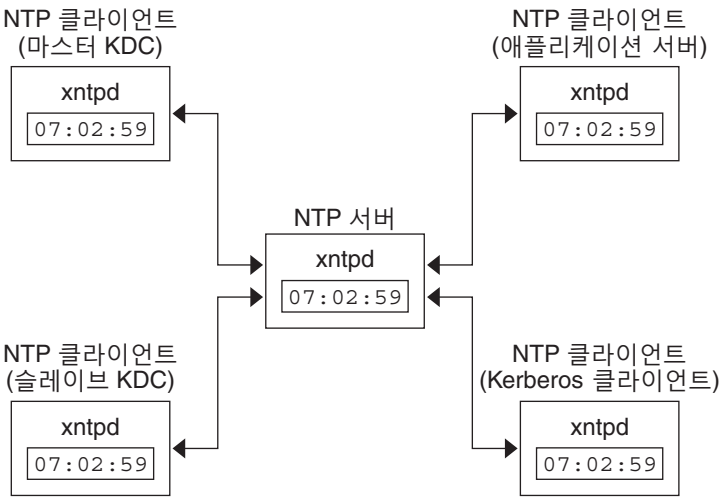
KDC와 Kerberos 클라이언트 간의 클럭은 동기화 상태로 유지되어야 하므로 동기화를 위해서는 NTP(Network Time Protocol) 소프트웨어를 사용해야 합니다. Oracle Solaris 소프트웨어에는 University of Delaware의 NTP 공용 도메인 소프트웨어가 포함되어 있습니다.

주 – 클럭 동기화의 다른 방법은 `rdate` 명령 및 `cron` 작업을 사용하는 것으로, 이 프로세스는 NTP를 사용하는 것보다 간단할 수 있습니다. 하지만 이 절에서는 NTP 사용을 집중적으로 다룹니다. 네트워크를 사용하여 클럭을 동기화할 경우 클럭 동기화 프로토콜이 자체적으로 보안되어야 합니다.

NTP를 통해 네트워크 환경에서 정확한 시간 또는 네트워크 클럭 동기화를 관리하거나 모두 관리할 수 있습니다. 기본적으로 NTP는 서버-클라이언트 구현입니다. 마스터 클럭으로 사용할 하나의 시스템(NTP 서버)을 선택합니다. 그런 다음 해당 클럭이 마스터 클럭과 동기화되도록 기타 모든 시스템(NTP 클라이언트)을 설정합니다.

클럭을 동기화하기 위해 NTP는 인터넷 표준 시간 서버와의 계약에 따라 UNIX 시스템 시간을 설정 및 유지 관리하는 `xntpd` 데몬을 사용합니다. 다음은 이 서버-클라이언트 NTP 구현의 예를 보여 줍니다.

그림 21–1 NTP를 사용하여 클럭 동기화



KDC 클라이언트와 Kerberos 클라이언트의 클럭이 동기화 상태로 유지되도록 하는 과정에서는 다음 단계가 구현됩니다.

1. 네트워크에서 NTP 서버를 설정합니다. 이 서버는 마스터 KDC를 제외한 모든 시스템일 수 있습니다. NTP 서버 작업을 찾으려면 **Oracle Solaris 관리: 네트워크 서비스**의 "NTP(Network Time Protocol) 관리(작업)"를 참조하십시오.
2. 네트워크에서 KDC 및 Kerberos 클라이언트를 구성하면 NTP 서버의 NTP 클라이언트가 되도록 설정됩니다. NTP 클라이언트 작업을 찾으려면 **Oracle Solaris 관리: 네트워크 서비스**의 "NTP(Network Time Protocol) 관리(작업)"를 참조하십시오.

마스터 KDC와 슬레이브 KDC 교체

마스터 KDC와 슬레이브 KDC를 간편하게 교체하려면 이 절의 절차를 사용해야 합니다. 마스터 KDC 서버가 특정 이유로 실패한 경우 또는 새 하드웨어 설치 등으로 인해 마스터 KDC를 다시 설치해야 하는 경우에만 마스터 KDC와 슬레이브 KDC를 교체해야 합니다.

▼ 교체 가능한 슬레이브 KDC 구성 방법

마스터 KDC로 사용할 수 있도록 설정하려는 슬레이브 KDC에서 이 절차를 수행하십시오. 이 절차에서는 증분 전파를 사용 중인 것으로 간주합니다.

1 KDC 설치 중 마스터 KDC 및 교체 가능한 슬레이브 KDC에 대한 별칭 이름을 사용합니다.

KDC에 대한 호스트 이름을 정의하는 경우 각 시스템의 DNS에 별칭이 포함되어 있는지 확인합니다. `/etc/krb5/krb5.conf` 파일에서 호스트를 정의하는 경우에도 별칭 이름을 사용합니다.

2 슬레이브 KDC 설치 단계를 따릅니다.

교체에 앞서 이 서버가 영역의 기타 모든 슬레이브 KDC로 작동해야 합니다. 지침은 368 페이지 "수동으로 슬레이브 KDC를 구성하는 방법"을 참조하십시오.

3 마스터 KDC 명령을 이동합니다.

마스터 KDC 명령이 이 슬레이브 KDC에서 실행되지 않도록 하려면 `kprop`, `kadmind` 및 `kadmin.local` 명령을 예약된 위치로 이동합니다.

```
kdc4 # mv /usr/lib/krb5/kprop /usr/lib/krb5/kprop.save
kdc4 # mv /usr/lib/krb5/kadmind /usr/lib/krb5/kadmind.save
kdc4 # mv /usr/sbin/kadmin.local /usr/sbin/kadmin.local.save
```

▼ 마스터 KDC와 슬레이브 KDC 교체 방법

이 절차에서 교체하려는 마스터 KDC 서버의 이름은 `kdc1`이며, 새 마스터 KDC가 될 슬레이브 KDC의 이름은 `kdc4`입니다. 이 절차에서는 증분 전파를 사용 중인 것으로 간주합니다.

시작하기 전에 이 절차를 수행하려면 슬레이브 KDC 서버가 교체 가능한 슬레이브로 설정되어 있어야 합니다. 자세한 내용은 402 페이지 "교체 가능한 슬레이브 KDC 구성 방법"을 참조하십시오.

1 수퍼유저로 로그인합니다.

2 새 마스터 KDC에서 kadmin을 시작합니다.

```
kdc4 # /usr/sbin/kadmin -p kws/admin
Enter password:       <Type kws/admin password>
kadmin:
```

a. kadmind 서비스에 대한 새 주체를 만듭니다.

다음 예에서는 두 행의 첫번째 addprinc 명령을 보여 주지만 명령은 한 행에만 입력되어야 합니다.

```
kadmin: addprinc -randkey -allow_tgs_req +password_changing_service -clearpolicy \
       changepw/kdc4.example.com
Principal "changepw/kdc4.example.com@ENG.SUN.COM" created.
kadmin: addprinc -randkey -allow_tgs_req -clearpolicy kadmin/kdc4.example.com
Principal "kadmin/kdc4.example.com@EXAMPLE.COM" created.
kadmin:
```

b. kadmin을 종료합니다.

```
kadmin: quit
```

3 새 마스터 KDC에서 강제로 동기화를 수행합니다.

다음 단계에서는 슬레이브 서버에서 강제로 전체 KDC 업데이트를 수행합니다.

```
kdc4 # svcadm disable network/security/krb5kdc
kdc4 # rm /var/krb5/principal.ulog
```

4 새 마스터 KDC에서 업데이트가 완료되었는지 확인합니다.

```
kdc4 # /usr/sbin/kproplog -h
```

5 새 마스터 KDC에서 KDC 서비스를 다시 시작합니다.

```
kdc4 # svcadm enable -r network/security/krb5kdc
```

6 새 마스터 KDC에서 업데이트 로그를 지웁니다.

이러한 단계는 새 마스터 KDC 서버에 대한 업데이트 로그를 다시 초기화합니다.

```
kdc4 # svcadm disable network/security/krb5kdc
kdc4 # rm /var/krb5/principal.ulog
```

7 이전 마스터 KDC에서 kadmind 및 krb5kdc 프로세스를 강제 종료합니다.

kadmind 프로세스를 강제 종료할 경우 변경 사항이 KDC 데이터베이스에 적용되지 않습니다.

```
kdc1 # svcadm disable network/security/kadmin
kdc1 # svcadm disable network/security/krb5kdc
```

8 이전 마스터 KDC에서 전파 요청 폴링 시간을 지정합니다.

/etc/krb5/kdc.conf에서 sunw_dbprop_master_ulogsize 항목을 주석 처리하고 sunw_dbprop_slave_poll을 정의하는 항목을 추가합니다. 이 항목은 폴링 시간을 2분으로 설정합니다.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM= {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                sunw_dbprop_enable = true
#               sunw_dbprop_master_ulogsize = 1000
                sunw_dbprop_slave_poll = 2m
        }
```

9 이전 마스터 KDC에서 마스터 KDC 명령 및 kadm5.acl 파일을 이동합니다.

마스터 KDC 명령이 실행되지 않도록 하려면 kprop, kadmind 및 kadmin.local 명령을 예약된 위치로 이동합니다.

```
kdc1 # mv /usr/lib/krb5/kprop /usr/lib/krb5/kprop.save
kdc1 # mv /usr/lib/krb5/kadmind /usr/lib/krb5/kadmind.save
kdc1 # mv /usr/sbin/kadmin.local /usr/sbin/kadmin.local.save
kdc1 # mv /etc/krb5/kadm5.acl /etc/krb5/kadm5.acl.save
```

10 DNS 서버에서 마스터 KDC에 대한 별칭 이름을 변경합니다.

서버를 변경하려면 example.com 영역 파일을 편집하고 masterkdc에 대한 항목을 변경합니다.

```
masterkdc IN CNAME kdc4
```

11 DNS 서버에서 인터넷 도메인 이름 서버를 다시 시작합니다.

다음 명령을 실행하여 새 별칭 정보를 재로드합니다.

```
# svcadm refresh network/dns/server
```

12 새 마스터 KDC에서 마스터 KDC 명령 및 슬레이브 kpropd.acl 파일을 이동합니다.

```
kdc4 # mv /usr/lib/krb5/kprop.save /usr/lib/krb5/kprop
kdc4 # mv /usr/lib/krb5/kadmind.save /usr/lib/krb5/kadmind
kdc4 # mv /usr/sbin/kadmin.local.save /usr/sbin/kadmin.local
kdc4 # mv /etc/krb5/kpropd.acl /etc/krb5/kpropd.acl.save
```

13 새 마스터 KDC에서 Kerberos 액세스 제어 목록 파일(kadm5.acl)을 만듭니다.

채워진 /etc/krb5/kadm5.acl 파일에는 KDC를 관리할 수 있도록 허용된 모든 주체 이름이 포함되어야 합니다. 또한 파일은 증분 전파에 대한 요청을 생성하는 슬레이브를 모두 나열해야 합니다. 자세한 내용은 kadm5.acl(4) 매뉴얼 페이지를 참조하십시오.

```
kdc4 # cat /etc/krb5/kadm5.acl
kws/admin@EXAMPLE.COM   *
kiprop/kdc1.example.com@EXAMPLE.COM p
```

14 새 마스터 KDC에서 kdc.conf 파일에 업데이트 로그 크기를 지정합니다.

sunw_dbprop_slave_poll 항목을 주석 처리하고 sunw_dbprop_master_ulogsize를 정의하는 항목을 추가합니다. 항목은 로그 크기를 1000개 항목으로 설정합니다.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM= {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                sunw_dbprop_enable = true
#               sunw_dbprop_slave_poll = 2m
                sunw_dbprop_master_ulogsize = 1000
        }
```

15 새 마스터 KDC에서 kadmind 및 krb5kdc를 시작합니다.

```
kdc4 # svcadm enable -r network/security/krb5kdc
kdc4 # svcadm enable -r network/security/kadmin
```

16 이전 마스터 KDC에서 kiprop 서비스 주체를 추가합니다.

krb5.keytab 파일에 kiprop 주체를 추가하면 증분 전파 서비스에 대해 kpropd 데몬이 자체적으로 인증할 수 있습니다.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Authenticating as pricipal kws/admin@EXAMPLE.COM with password.
Enter password:     <Type kws/admin password>
kadmin: ktadd kiprop/kdc1.example.com
Entry for principal kiprop/kdc1.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc1.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc1.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc1.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc1.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin: quit
```

17 **이전 마스터 KDC에서 `krb5.conf`에 나열된 각 KDC의 항목을 전파 구성 파일 `kpropd.acl`에 추가합니다.**

```
kdc1 # cat /etc/krb5/kpropd.acl
host/kdc1.example.com@EXAMPLE.COM
host/kdc2.example.com@EXAMPLE.COM
host/kdc3.example.com@EXAMPLE.COM
host/kdc4.example.com@EXAMPLE.COM
```

18 **이전 마스터 KDC에서 `kpropd` 및 `krb5kdc`를 시작합니다.**

```
kdc1 # svcadm enable -r network/security/krb5_prop
kdc1 # svcadm enable -r network/security/krb5kdc
```

Kerberos 데이터베이스 관리

Kerberos 데이터베이스는 Kerberos의 중심이므로 올바르게 유지 관리되어야 합니다. 이 절에서는 Kerberos 데이터베이스 관리(예: 데이터베이스 백업 및 복원, 증분 또는 병렬 전파 설정, stash 파일 관리) 방법에 대한 몇 가지 절차를 제공합니다. 데이터베이스 초기 설정 단계는 356 페이지 “수동으로 마스터 KDC를 구성하는 방법”에서 설명됩니다.

Kerberos 데이터베이스 백업 및 전파

마스터 KDC에서 슬레이브 KDC로 Kerberos 데이터베이스를 전파하는 작업은 가장 중요한 구성 작업 중 하나입니다. 전파가 충분히 자주 실행되지 않으면 마스터 KDC와 슬레이브 KDC의 동기화가 끊깁니다. 따라서 마스터 KDC의 작동이 중지될 경우 슬레이브 KDC가 최신 데이터베이스 정보를 가지지 못합니다. 또한 로드 균형 조정 용도로 슬레이브 KDC가 마스터 KDC로 구성된 경우 슬레이브 KDC를 마스터 KDC로 사용하는 클라이언트가 최신 정보를 가지지 못합니다. 따라서 전파가 충분히 자주 실행되도록 하거나 Kerberos 데이터베이스 변경 빈도에 따라 서버에서 증분 전파를 구성해야 합니다. 수동으로 데이터베이스를 전파하면 관리 오버헤드가 늘어나므로 수동 전파보다 증분 전파가 선호됩니다. 또한 데이터베이스 전체 전파를 실행하는 경우 비효율성이 발생할 수 있습니다.

마스터 KDC를 구성할 때 자동으로 Kerberos 데이터베이스를 `/var/krb5/slave_datatrans` 덤프 파일에 백업하여 슬레이브 KDC에 전파하도록 `cron` 작업의 `kprop_script` 명령을 설정합니다. 하지만 다른 파일과 마찬가지로 Kerberos 데이터베이스도 손상될 수 있습니다. 슬레이브 KDC에서 데이터가 손상된 경우 다음 번 데이터베이스 자동 전파 시 복사본이 다시 설치되므로 특별한 통지가 없을 수 있습니다. 하지만 마스터 KDC에서 데이터가 손상된 경우 다음 번 전파 중 손상된 데이터베이스가 모든 슬레이브 KDC에 전파됩니다. 또한 손상된 백업이 손상되지 않은 마스터 KDC의 이전 백업 파일을 겹쳐 씁니다.

이 시나리오에는 "안전한" 백업 복사본이 없으므로 주기적으로 `slave_datatrans` 덤프 파일이 다른 위치에 복사되거나 `kdb5_util`의 `dump` 명령을 사용하여 별도의 다른 백업

복사본이 만들어지도록 cron 작업도 설정해야 합니다. 그런 다음 데이터베이스가 손상되면 kdb5_util의 load 명령을 사용하여 마스터 KDC에서 최신 백업을 복원할 수 있습니다.

데이터베이스 덤프 파일에는 주체 키가 포함되어 있으므로 권한이 없는 사용자가 액세스하지 못하도록 파일을 보호해야 합니다. 기본적으로 데이터베이스 덤프 파일은 root로만 읽기 및 쓰기 권한을 가집니다. 허용되지 않은 액세스를 방지하려면 kprop 명령을 통해서만 데이터베이스 덤프 파일을 전파하십시오. 그러면 전달하려는 데이터가 암호화됩니다. 또한 kprop는 슬레이브 KDC로만 데이터를 전파하므로 허용되지 않은 호스트로 데이터베이스 덤프 파일이 잘못 전송될 가능성이 최소화됩니다.

주의 – Kerberos 데이터베이스가 전파된 후 업데이트되는 경우 및 나중에 데이터베이스가 다음 번 전파 전에 손상되는 경우 KDC 슬레이브에 업데이트가 포함되지 않습니다. 업데이트는 손실됩니다. 따라서 정기적으로 예약된 전파 전에 Kerberos 데이터베이스에 중요한 업데이트를 추가할 경우 데이터가 손실되지 않도록 수동으로 데이터베이스를 전파해야 합니다.

kpropd.acl 파일

슬레이브 KDC의 kpropd.acl 파일은 host 주체 이름 목록(한 행당 하나의 이름)을 제공합니다. 이 목록에는 KDC가 전파를 통해 업데이트된 데이터베이스를 수신할 수 있는 시스템이 지정됩니다. 마스터 KDC를 사용하여 모든 슬레이브 KDC를 전파할 경우 각 슬레이브의 kpropd.acl 파일에는 마스터 KDC의 host 주체 이름만 포함되어야 합니다.

단, 본 설명서의 Kerberos 설치 및 후속 구성 단계에서는 마스터 KDC와 슬레이브 KDC에 동일한 kpropd.acl 파일을 추가하도록 안내합니다. 이 파일에는 모든 KDC host 주체 이름이 포함되어 있습니다. 전파하는 KDC를 일시적으로 사용할 수 없을 경우 이 구성을 통해 KDC에서의 전파를 실행할 수 있습니다. 또한 모든 KDC의 복사본이 동일하게 유지되도록 유지 관리가 간편한 구성을 설정합니다.

kprop_script 명령

kprop_script 명령은 kprop 명령을 사용하여 Kerberos 데이터베이스를 다른 KDC에 전파합니다. kprop_script 명령이 슬레이브 KDC에서 실행되면 Kerberos 데이터베이스의 슬레이브 KDC 복사본을 다른 KDC에 전파합니다. kprop_script는 전파할 KDC를 표시하는 인수에 대해 공백으로 구분된 호스트 이름 목록을 승인합니다.

kprop_script가 실행되면 /var/krb5/slave_datatrans 파일에 Kerberos 데이터베이스 백업을 만들고 지정된 KDC에 해당 파일을 복사합니다. Kerberos 데이터베이스는 전파가 완료될 때까지 잠깁니다.

▼ Kerberos 데이터베이스 백업 방법

1 관리자가 되거나 Kerberos Server Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 `kdb5_util` 명령의 `dump` 명령을 사용하여 Kerberos 데이터베이스를 백업합니다.

```
# /usr/sbin/kdb5_util dump [-verbose] [-d dbname] [filename [principals...]]
```

`-verbose` 백업하려는 각 주체 및 정책의 이름을 인쇄합니다.

dbname 백업할 데이터베이스의 이름을 정의합니다. 파일에 대한 절대 경로를 지정할 수 있습니다. `-d` 옵션을 지정하지 않을 경우 기본 데이터베이스 이름은 `/var/krb5/principal`입니다.

filename 데이터베이스 백업에 사용되는 파일을 정의합니다. 파일에 대한 절대 경로를 지정할 수 있습니다. 파일을 지정하지 않을 경우 표준 출력에 데이터베이스가 덤프됩니다.

principals 백업할 하나 이상의 주체 목록(공백으로 구분됨)을 정의합니다. 정규화된 주체 이름을 사용해야 합니다. 주체를 지정하지 않을 경우 전체 데이터베이스가 백업됩니다.

예 21–16 Kerberos 데이터베이스 백업

다음 예에서는 이름이 `dumpfile`인 파일에 Kerberos 데이터베이스가 백업됩니다. **-상세 정보 표시** 옵션이 지정되었으므로 각 주체가 백업된 대로 인쇄됩니다.

```
# kdb5_util dump -verbose dumpfile
kadmin/kdc1.eng.example.com@ENG.EXAMPLE.COM
krbtgt/ENG.EXAMPLE.COM@ENG.EXAMPLE.COM
kadmin/history@ENG.EXAMPLE.COM
pak/admin@ENG.EXAMPLE.COM
pak@ENG.EXAMPLE.COM
changepw/kdc1.eng.example.com@ENG.EXAMPLE.COM
```

다음 예에서는 Kerberos 데이터베이스의 `pak` 및 `pak/admin` 주체가 백업됩니다.

```
# kdb5_util dump -verbose dumpfile pak/admin@ENG.EXAMPLE.COM pak@ENG.EXAMPLE.COM
pak/admin@ENG.EXAMPLE.COM
pak@ENG.EXAMPLE.COM
```

▼ Kerberos 데이터베이스 복원 방법

1 마스터 KDC에서 수퍼유저로 로그인합니다.

2 마스터에서 KDC 데몬을 중지합니다.

```
kdc1 # svcadm disable network/security/krb5kdc
kdc1 # svcadm disable network/security/kadmin
```

3 kdb_util 명령의 load 명령을 사용하여 Kerberos 데이터베이스를 복원합니다.

```
# /usr/sbin/kdb5_util load [-verbose] [-d dbname] [-update] [filename]
```

`-verbose` 복원하려는 각 주체 및 정책의 이름을 인쇄합니다.

dbname 복원할 데이터베이스의 이름을 정의합니다. 파일에 대한 절대 경로를 지정할 수 있습니다. `-d` 옵션을 지정하지 않을 경우 기본 데이터베이스 이름은 `/var/krb5/principal`입니다.

`-update` 기존 데이터베이스를 업데이트합니다. 그렇지 않으면 새 데이터베이스가 만들어지거나 기존 데이터베이스를 겹쳐 씁니다.

filename 데이터베이스를 복원할 파일을 정의합니다. 파일에 대한 절대 경로를 지정할 수 있습니다.

4 KDC 데몬을 시작합니다.

```
kdc1 # svcadm enable -r network/security/krb5kdc
kdc1 # svcadm enable -r network/security/kadmin
```

예 21–17 Kerberos 데이터베이스 복원

다음 예에서는 이름이 `database1`인 데이터베이스가 `dumpfile` 파일의 현재 디렉토리에 복원됩니다. `-update` 옵션이 지정되지 않았으므로 복원을 통해 새 데이터베이스가 만들어집니다.

```
# kdb5_util load -d database1 dumpfile
```

▼ 서버 업그레이드 후 Kerberos 데이터베이스 변환 방법

KDC 데이터베이스가 Solaris 8 또는 Solaris 9 릴리스를 실행하는 서버에 만들어진 경우 데이터베이스를 변환하면 향상된 데이터베이스 형식을 사용할 수 있습니다.

시작하기 전에 데이터베이스가 오래된 형식을 사용하고 있는지 확인합니다.

1 **마스터에서 KDC 데몬을 중지합니다.**

```
kdc1 # svcadm disable network/security/krb5kdc
kdc1 # svcadm disable network/security/kadmin
```

2 **데이터베이스의 임시 복사본을 저장할 디렉토리를 만듭니다.**

```
kdc1 # mkdir /var/krb5/tmp
kdc1 # chmod 700 /var/krb5/tmp
```

3 **KDC 데이터베이스를 덤프합니다.**

```
kdc1 # kdb5_util dump /var/krb5/tmp/prdb.txt
```

4 **현재 데이터베이스 파일의 복사본을 저장합니다.**

```
kdc1 # cd /var/krb5
kdc1 # mv princ* tmp/
```

5 **데이터베이스를 로드합니다.**

```
kdc1 # kdb5_util load /var/krb5/tmp/prdb.txt
```

6 **KDC 데몬을 시작합니다.**

```
kdc1 # svcadm enable -r network/security/krb5kdc
kdc1 # svcadm enable -r network/security/kadmin
```

▼ 증분 전파를 사용하도록 마스터 KDC를 재구성하는 방법

이 절차의 단계를 통해 증분 전파를 사용하도록 기존 마스터 KDC를 재구성할 수 있습니다. 이 절차에서는 다음 구성 매개변수가 사용됩니다.

- 영역 이름 = `EXAMPLE.COM`
- DNS 도메인 이름 = `example.com`
- 마스터 KDC = `kdc1.example.com`
- 슬레이브 KDC = `kdc2.example.com`
- `admin` 주체 = `kws/admin`

1 **수퍼유저로 로그인합니다.**

2 **`kdc.conf`에 항목을 추가합니다.**

증분 전파를 사용으로 설정하고 KDC 마스터가 로그에 유지할 수 있는 업데이트 수를 선택해야 합니다. 자세한 내용은 `kdc.conf(4)` 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
        kdc_ports = 88,750

[realms]
```

```
EXAMPLE.COM= {
        profile = /etc/krb5/krb5.conf
        database_name = /var/krb5/principal
        acl_file = /etc/krb5/kadm5.acl
        kadmind_port = 749
        max_life = 8h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        sunw_dbprop_enable = true
        sunw_dbprop_master_ulogsize = 1000
}
```

3 kiprop 주체를 만듭니다.

kiprop 주체는 마스터 KDC 서버를 인증하고 마스터 KDC로부터의 업데이트를 허가하는 데 사용됩니다.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin: addprinc -randkey kiprop/kdc1.example.com
Principal "kiprop/kdc1.example.com@EXAMPLE.COM" created.
kadmin: addprinc -randkey kiprop/kdc2.example.com
Principal "kiprop/kdc2.example.com@EXAMPLE.COM" created.
kadmin:
```

4 마스터 KDC에서 kadm5.acl에 kiprop 항목을 추가합니다.

이 항목은 마스터 KDC가 kdc2 서버의 증분 전파 요청을 수신할 수 있도록 합니다.

```
kdc1 # cat /etc/krb5/kadm5.acl
*/admin@EXAMPLE.COM *
kiprop/kdc2.example.com@EXAMPLE.COM p
```

5 root crontab 파일의 kprop 행을 주석 처리합니다.

이 단계는 마스터 KDC가 KDC 데이터베이스 복사본을 전파하지 않도록 합니다.

```
kdc1 # crontab -e
#ident  "@(#)root       1.20    01/11/06 SMI"
#
# The root crontab should be used to perform accounting data collection.
#
# The rtc command is run to adjust the real time clock if and when
# daylight savings time changes.
#
10 3 * * * /usr/sbin/logadm
15 3 * * 0 /usr/lib/fs/nfs/nfsfind
1 2 * * * [ -x /usr/sbin/rtc ] && /usr/sbin/rtc -c > /dev/null 2>&1
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
#10 3 * * * /usr/lib/krb5kprop_script kdc2.example.sun.com #SUNWkr5ma
```

6 kadmind를 다시 시작합니다.

```
kdc1 # svcadm restart network/security/kadmin
```

7 증분 전파를 사용하는 모든 슬레이브 KDC 서버를 재구성합니다.

전체 지침은 412 페이지 "증분 전파를 사용하도록 슬레이브 KDC를 재구성하는 방법"을 참조하십시오.

▼ 증분 전파를 사용하도록 슬레이브 KDC를 재구성하는 방법

1 수퍼유저로 로그인합니다.

2 kdc.conf에 항목을 추가합니다.

첫번째 새 항목은 증분 전파를 사용으로 설정합니다. 두번째 새 항목은 폴링 시간을 2분으로 설정합니다.

```
kdc2 # cat /etc/krb5/kdc.conf
[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM= {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                sunw_dbprop_enable = true
                sunw_dbprop_slave_poll = 2m
        }
```

3 krb5.keytab 파일에 kiprop 주체를 추가합니다.

```
kdc2 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin: ktadd kiprop/kdc2.example.com
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin: quit
```

4 **kpropd를 다시 시작합니다.**

```
kdc2 # svcadm restart network/security/krb5_prop
```

▼ 전체 전파를 사용하도록 슬레이브 KDC를 구성하는 방법

이 절차에서는 Solaris 10 릴리스를 실행 중인 슬레이브 KDC 서버가 전체 전파를 사용하도록 재구성하는 방법을 보여 줍니다. 일반적으로 마스터 KDC 서버가 Solaris 9 릴리스 또는 이전 릴리스를 실행 중인 경우에만 이 절차를 사용합니다. 이 경우 마스터 KDC 서버는 증분 전파를 지원할 수 없으므로 전파가 작동하려면 슬레이브를 구성해야 합니다.

이 절차에서는 이름이 `kdc3`인 슬레이브 KDC가 구성됩니다. 이 절차에서는 다음 구성 매개변수를 사용합니다.

- 영역 이름 = `EXAMPLE.COM`
- DNS 도메인 이름 = `example.com`
- 마스터 KDC = `kdc1.example.com`
- 슬레이브 KDC = `kdc2.example.com` 및 `kdc3.example.com`
- `admin` 주체 = `kws/admin`
- 온라인 도움말 URL = `http://download.oracle.com/docs/cd/E23824_01/html/821-1456/aadmin-23.html`

주 – 349 페이지 “그래픽 Kerberos 관리 도구의 온라인 도움말 URL”에 설명된 대로 절을 가리키도록 URL을 조정하십시오.

시작하기 전에 마스터 KDC를 구성해야 합니다. 이 슬레이브를 교체 가능한 것으로 설정하려는 경우 구체적인 지침은 402 페이지 “마스터 KDC와 슬레이브 KDC 교체”를 참조하십시오.

1 **마스터 KDC에서 수퍼유저로 로그인합니다.**

2 마스터 KDC에서 kadmin을 시작합니다.

마스터 KDC를 구성할 때 만든 admin 주체 이름 중 하나로 로그인해야 합니다.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. 마스터 KDC에서 데이터베이스에 슬레이브 host 주체를 추가합니다(아직 추가하지 않은 경우).

슬레이브가 작동하려면 host 주체가 있어야 합니다. 주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

```
kadmin: addprinc -randkey host/kdc3.example.com
Principal "host/kdc3@EXAMPLE.COM" created.
kadmin:
```

b. kadmin을 종료합니다.

```
kadmin: quit
```

3 마스터 KDC에서 Kerberos 구성 파일(krb5.conf)을 편집합니다.

슬레이브마다 항목을 하나씩 추가해야 합니다. 이 파일에 대한 자세한 설명은 krb5.conf(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/krb5.conf
 .
 .
[realms]
                EXAMPLE.COM = {
                kdc = kdc1.example.com
                kdc = kdc2.example.com
                kdc = kdc3.example.com
                admin_server = kdc1.example.com
        }
```

4 마스터 KDC에서 마스터 KDC에 대한 항목과 각 슬레이브 KDC를 kpropd.acl 파일에 추가합니다.

이 파일에 대한 자세한 설명은 kprop(1M) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/kpropd.acl
host/kdc1.example.com@EXAMPLE.COM
host/kdc2.example.com@EXAMPLE.COM
host/kdc3.example.com@EXAMPLE.COM
```

5 모든 슬레이브 KDC에서 마스터 KDC 서버의 KDC 관리 파일을 복사합니다.

마스터 KDC 서버가 각 KDC 서버에 필요한 정보를 업데이트했으므로 모든 슬레이브 KDC에서 이 단계를 수행해야 합니다. `ftp` 또는 유사한 전송 방식을 사용하여 마스터 KDC의 다음 파일을 복사할 수 있습니다.

- `/etc/krb5/krb5.conf`
- `/etc/krb5/kdc.conf`
- `/etc/krb5/kpropd.acl`

6 모든 슬레이브 KDC에서 Kerberos 액세스 제어 목록 파일 `kadm5.acl`이 채워져 있지 않은지 확인합니다.

수정되지 않은 `kadm5.acl` 파일은 다음과 같이 표시됩니다.

```
kdc2 # cat /etc/krb5/kadm5.acl
*/admin@___default_realm___ *
```

파일에 `kiprop` 항목이 있을 경우 제거합니다.

7 새 슬레이브에서 `kadmin` 명령을 시작합니다.

마스터 KDC를 구성할 때 만든 `admin` 주체 이름 중 하나로 로그인해야 합니다.

```
kdc2 # /usr/sbin/kadmin -p kws/admin
Enter password:      <Type kws/admin password>
kadmin:
```

a. `kadmin`을 사용하여 슬레이브의 keytab 파일에 슬레이브의 `host` 주체를 추가합니다.

이 항목은 `kprop` 및 기타 Kerberos화된 응용 프로그램이 작동할 수 있도록 합니다. 주체 인스턴스가 호스트 이름인 경우 FQDN은 이름 서비스의 도메인 이름 대소문자에 관계없이 소문자로 지정되어야 합니다.

```
kadmin: ktadd host/kdc3.example.com
Entry for principal host/kdc3.example.com with kvno 3, encryption type AES-256 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc3.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc3.example.com with kvno 3, encryption type Triple DES cbc
          mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc3.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc3.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

b. `kadmin`을 종료합니다.

```
kadmin: quit
```

8 마스터 KDC에서 `crontab -e`를 실행하여 자동으로 백업을 실행하는 `cron` 작업에 슬레이브 KDC 이름을 추가합니다.

`kprop_script` 행의 끝에 각 슬레이브 KDC 서버의 이름을 추가합니다.

```
10 3 * * * /usr/lib/krb5/kprop_script kdc2.example.com kdc3.example.com
```

백업 시간을 변경할 수도 있습니다. 이 항목은 매일 오전 3시 10분에 백업 프로세스를 시작합니다.

9 새 슬레이브에서 Kerberos 전파 데몬을 시작합니다.

```
kdc3 # svcadm enable network/security/krb5_prop
```

10 마스터 KDC에서 kprop_script를 사용하여 데이터베이스를 백업 및 전파합니다.

데이터베이스 백업 복사본을 이미 사용할 수 있을 경우 다른 백업을 완료할 필요가 없습니다. 추가 지침은 417 페이지 "수동으로 슬레이브 KDC에 Kerberos 데이터베이스를 전파하는 방법"을 참조하십시오.

```
kdc1 # /usr/lib/krb5/kprop_script kdc3.example.com
Database propagation to kdc3.example.com: SUCCEEDED
```

11 새 슬레이브에서 kdb5_util을 사용하여 stash 파일을 만듭니다.

```
kdc3 # /usr/sbin/kdb5_util stash
kdb5_util: Cannot find/read stored master key while reading master key
kdb5_util: Warning: proceeding without master key

Enter KDC database master key:      <Type the key>
```

12 (옵션) 새 슬레이브 KDC에서 NTP 또는 다른 클럭 동기화 방식을 사용하여 마스터 KDC 클럭을 동기화합니다.

NTP(Network Time Protocol)를 설치하여 사용할 필요가 없습니다. 하지만 인증이 성공하려면 모든 클럭이 krb5.conf 파일의 libdefaults 섹션에 정의된 기본 시간에 속해야 합니다. NTP에 대한 자세한 내용은 400 페이지 "KDC와 Kerberos 클라이언트 간의 클럭 동기화"를 참조하십시오.

13 새 슬레이브에서 KDC 데몬(krb5kdc)을 시작합니다.

```
kdc3 # svcadm enable network/security/krb5kdc
```

▼ KDC 서버 동기화 여부 확인 방법

증분 전파가 구성된 경우 이 절차는 슬레이브 KDC의 정보가 업데이트되었는지 확인합니다.

1 수퍼유저로 로그인합니다.

2 KDC 마스터 서버에서 kproplog 명령을 실행합니다.

```
kdc1 # /usr/sbin/kproplog -h
```

3 KDC 슬레이브 서버에서 kproplog 명령을 실행합니다.

```
kdc2 # /usr/sbin/kproplog -h
```

4 최종 일련 번호 값과 최종 시간 기록 값이 일치하는지 확인합니다.

예 21–18 KDC 서버 동기화 여부 확인

다음은 마스터 KDC 서버에서 `kproplog` 명령을 실행한 결과의 샘플입니다.

```
kdc1 # /usr/sbin/kproplog -h

Kerberos update log (/var/krb5/principal.ulog)
Update log dump:
    Log version #: 1
    Log state: Stable
    Entry block size: 2048
    Number of entries: 2500
    First serial #: 137966
    Last serial #: 140465
    First time stamp: Fri Nov 28 00:59:27 2004
    Last time stamp: Fri Nov 28 01:06:13 2004
```

다음은 슬레이브 KDC 서버에서 `kproplog` 명령을 실행한 결과의 샘플입니다.

```
kdc2 # /usr/sbin/kproplog -h

Kerberos update log (/var/krb5/principal.ulog)
Update log dump:
    Log version #: 1
    Log state: Stable
    Entry block size: 2048
    Number of entries: 0
    First serial #: None
    Last serial #: 140465
    First time stamp: None
    Last time stamp: Fri Nov 28 01:06:13 2004
```

최종 일련 번호와 최종 시간 기록의 값이 같은지 확인합니다. 값이 같을 경우 슬레이브가 마스터 KDC 서버와 동기화된 것입니다.

슬레이브 KDC 서버 출력에서 슬레이브 KDC 서버의 업데이트 로그에 업데이트 항목이 존재하지 않는지 확인합니다. 슬레이브 KDC 서버는 마스터 KDC 서버와 달리 일련의 업데이트를 유지하지 않으므로 항목이 존재하지 않습니다. 또한 KDC 슬레이브 서버는 관련 정보가 아닌 첫번째 일련 번호 또는 첫번째 시간 기록에 대한 정보를 포함하지 않습니다.

▼ 수동으로 슬레이브 KDC에 Kerberos 데이터베이스를 전파하는 방법

이 절차에서는 `kprop` 명령을 사용하여 Kerberos 데이터베이스를 전파하는 방법을 보여 줍니다. 주기적 `cron` 작업 이외에 슬레이브 KDC를 마스터 KDC와 동기화해야 할 경우 이

절차를 사용하십시오. kprop_script와 달리 kprop를 사용하면 Kerberos 데이터베이스의 새 백업을 만들지 않고도 현재 데이터베이스 백업을 전파할 수 있습니다.

주 – 증분 전파를 사용 중인 경우 이 절차를 사용하지 마십시오.

1. **관리자가 되거나 Kerberos Server Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.**
   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.
2. **마스터 KDC에서 수퍼유저로 로그인합니다.**
3. **(옵션) kdb5_util 명령을 사용하여 데이터베이스를 백업합니다.**

   ```
   # /usr/sbin/kdb5_util dump /var/krb5/slave_datatrans
   ```
4. **kprop 명령을 사용하여 슬레이브 KDC에 데이터베이스를 전파합니다.**

   ```
   # /usr/lib/krb5/kprop -f /var/krb5/slave_datatrans slave-KDC
   ```

예 21–19 kprop_script를 사용하여 수동으로 슬레이브 KDC에 Kerberos 데이터베이스 전파

주기적 cron 작업 이외에 데이터베이스를 백업하여 슬레이브 KDC에 전파하려는 경우에도 다음과 같이 kprop_script 명령을 사용할 수 있습니다.

```
# /usr/lib/krb5/kprop_script slave-KDC
```

병렬 전파 설정

대부분의 경우 마스터 KDC는 슬레이브 KDC에 Kerberos 데이터베이스를 전파하는 데만 사용됩니다. 하지만 사이트에 KDC가 여러 개 있을 경우 전파 프로세스를 공유하는 로드(**병렬 전파**라고 함)를 사용할 수 있습니다.

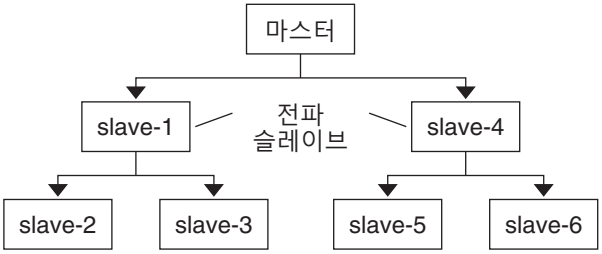
주 – 증분 전파를 사용 중인 경우 이 절차를 사용하지 마십시오.

병렬 전파를 사용하면 특정 슬레이브 KDC가 마스터 KDC와 전파 역할을 공유할 수 있습니다. 이 역할 공유를 통해 전파를 더 빠르게 완료하고 마스터 KDC에 대한 작업을 줄일 수 있습니다.

예를 들어, 사이트에 마스터 KDC가 한 개 있고 슬레이브 KDC가 여섯 개 있으며(그림 21–2 참조) 여기서 slave-1에서 slave-3까지는 하나의 논리적 그룹화로 구성되고 slave-4에서 slave-6까지는 다른 논리적 그룹화로 구성된다고 가정합니다. 이

경우 병렬 전파를 설정하려면 마스터 KDC가 데이터베이스를 slave-1 및 slave-4에 전파하도록 할 수 있습니다. 이런 방법으로 해당 KDC 슬레이브는 데이터베이스를 그룹 내 KDC 슬레이브에 전파할 수 있습니다.

그림 21–2 병렬 전파 구성 예



병렬 전파 설정을 위한 구성 단계

다음은 자세한 단계별 절차가 아니지만 병렬 전파를 사용으로 설정할 수 있는 높은 레벨의 구성 단계 목록입니다. 이러한 단계에는 다음이 포함됩니다.

1. 마스터 KDC에서 후속 전파(**전파 슬레이브**)를 수행할 KDC 슬레이브에 대해서만 인수가 포함되도록 cron 작업의 kprop_script 항목을 변경합니다.
2. 각 전파 슬레이브에서 전파할 슬레이브에 대한 인수가 포함되도록 cron 작업에 kprop_script 항목을 추가합니다. 병렬 전파를 성공적으로 수행하려면 새 Kerberos 데이터베이스와 함께 전파 슬레이브가 자체적으로 전파된 후 cron 작업이 실행되도록 설정해야 합니다.

주 – 전파 슬레이브를 전파하는 데 걸리는 시간은 네트워크 대역폭, Kerberos 데이터베이스 크기 등의 인자에 따라 다릅니다.

---

3. 각 슬레이브 KDC에서 적절한 전파 권한을 설정합니다. 이 단계를 완료하려면 kpropd.acl 파일에 전파하는 KDC의 host 주체 이름을 추가합니다.

예 21–20 병렬 전파 설정

그림 21–2의 예를 사용하면 마스터 KDC의 kprop_script 항목이 다음과 같이 표시됩니다.

```
0 3 * * * /usr/lib/krb5/kprop_script slave-1.example.com slave-4.example.com
```

slave-1의 kprop_script 항목은 다음과 같이 표시됩니다.

예 21–20 병렬 전파 설정 *(계속)*

```
0 4 * * * /usr/lib/krb5/kprop_script slave-2.example.com slave-3.example.com
```

마스터가 전파를 수행하고 1시간 후에 슬레이브에서 전파가 시작됩니다.

전파 슬레이브의 `kpropd.acl` 파일에는 다음 항목이 포함됩니다.

```
host/master.example.com@EXAMPLE.COM
```

`slave-1`이 전파하려는 KDC 슬레이브의 `kpropd.acl` 파일에는 다음 항목이 포함됩니다.

```
host/slave-1.example.com@EXAMPLE.COM
```

stash 파일 관리

stash **파일**에는 Kerberos 데이터베이스를 만들 때 자동으로 만들어지는 Kerberos 데이터베이스용 마스터 키가 포함되어 있습니다. stash 파일이 손상된 경우 `kdb5_util` 유틸리티의 `stash` 명령을 사용하여 손상된 파일을 바꿀 수 있습니다. `kdb5_util`의 `destroy` 명령을 사용하여 Kerberos 데이터베이스를 제거한 후에만 stash 파일을 제거해야 합니다. stash 파일은 데이터베이스와 함께 자동으로 제거되지 않으므로 정리를 완료하려면 stash 파일을 제거해야 합니다.

▼ stash 파일 제거 방법

1 stash 파일이 포함된 KDC에 수퍼유저로 로그인합니다.

2 stash 파일을 제거합니다.

```
# rm stash-file
```

여기서 *stash-file*은 stash 파일의 경로입니다. 기본적으로 stash 파일은 `/var/krb5/.k5.realm`에 있습니다.

주 – stash 파일을 다시 만들어야 할 경우 `kdb5_util` 명령의 `-f` 옵션을 사용할 수 있습니다.

▼ 새 마스터 키 사용 방법

1 관리자가 되거나 Kerberos Server Management 프로파일에 지정된 역할이나 사용자 이름을 맡습니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 새 마스터 키를 만듭니다.

이 명령은 임의로 생성된 새 마스터 키를 추가합니다. -s 옵션은 새 마스터 키가 기본 keytab에 저장되도록 요청합니다.

```
# kdb5_util add_mkey -s

Creating new master key for master key principal 'K/M@EXAMPLE.COM'
You will be prompted for a new database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:      <Type the password>
Re-enter KDC database master key to verify:      <Type it again>
```

3 새 마스터 키가 존재하는지 확인합니다.

```
# kdb5_util list_mkeys
Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-128 CTS mode with 96-bit SHA-1 HMAC, No activate time set
KNVO: 1, Enctype: DES cbc mode with RSA-MD5, Active on: Wed Dec 31 18:00:00 CST 2001 *
```

이 출력의 별표는 현재 활성 상태의 마스터 키를 식별합니다.

4 새로 만들어진 마스터 키가 활성화될 시간을 설정합니다.

```
# date
Fri Jul 1 17:57:00 CDT 2011
# kdb5_util use_mkey 2 'now+2days'
# kdb5_util list_mkeys
Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-128 CTS mode with 96-bit SHA-1 HMAC, Active on: Sun Jul 03 17:57:15 CDT 2011
KNVO: 1, Enctype: DES cbc mode with RSA-MD5, Active on: Wed Dec 31 18:00:00 CST 2001 *
```

이 예에서는 새 마스터 키가 모든 KDC에 전파되도록 충분한 시간을 확보하기 위해 날짜가 2일로 설정됩니다. 환경에 맞게 날짜를 조정합니다.

5 (옵션) 새 주체를 만든 후 새 마스터 키가 사용되고 있는지 확인합니다.

```
# kadmin.local -q 'getprinc jimf' |egrep 'Principal|MKey'
Authenticating as principal root/admin@EXAMPLE.COM with password.
Principal: jimf@EXAMPLE.COM
MKey: vno 2
```

이 예에서는 MKey: vno 2를 통해 주체의 보안 키가 새로 만들어진 마스터 키 2에 의해 보호되고 있음을 알 수 있습니다.

6 새 마스터 키로 사용자 주체 보안 키를 다시 암호화합니다.

명령 끝에 패턴 인수를 추가할 경우 패턴과 일치하는 주체가 업데이트됩니다. 이 명령 구문에 -n 옵션을 추가하여 업데이트할 주체를 식별합니다.

```
# kdb5_util update_princ_encryption -f -v
Principals whose keys WOULD BE re-encrypted to master key vno 2:
updating: host/kdc1.example.com@EXAMPLE.COM
skipping:   jimf@EXAMPLE.COM
updating: kadmin/changepw@EXAMPLE.COM
updating: kadmin/history@EXAMPLE.COM
updating: kdc/admin@EXAMPLE.COM
```

```
updating: host/kdc2.example.com@EXAMPLE.COM
6 principals processed: 5 updated, 1 already current
```

7 이전 마스터 키를 제거합니다.

주체 보안 키를 보호하는 데 더 이상 마스터 키가 사용되지 않을 경우 마스터 키 주체에서 제거할 수 있습니다. 이 명령은 주체가 키를 사용하고 있을 경우 키를 제거하지 않습니다. 이 명령에 -n 옵션을 추가하여 올바른 마스터 키가 제거될지 확인합니다.

```
# kdb5_util purge_mkeys -f -v
Purging the follwing master key(s) from K/M@EXAMPLE.COM:
KNVO: 1
1 key(s) purged.
```

8 이전 마스터 키가 제거되었는지 확인합니다.

```
# kdb5_util list_mkeys
Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-128 CTS mode with 96-bit SHA-1 HMAC, Active on: Sun Jul 03 17:57:15 CDT 2011 *
```

9 stash 파일을 업데이트합니다.

```
# kdb5_util stash
Using existing stashed keys to update stash file.
```

10 stash 파일이 업데이트되었는지 확인합니다.

```
# klist -kt /var/krb5/.k5.EXAMPLE.COM
Keytab name: FILE:.k5.EXAMPLE.COM
KVNO Timestamp             Principal
---- --------------- ------------------------------------------------------
   2 05/07/2011 15:08 K/M@EXAMPLE.COM
```

LDAP 디렉토리 서버에서 KDC 관리

LDAP 디렉토리 서버를 사용하는 대부분의 KDC 관리 작업은 DB2 서버에 대한 관리 작업과 동일합니다. LDAP에서만 작동하는 몇 가지 새 작업이 있습니다.

표 21–3 LDAP을 사용하도록 KDC 서버 구성(작업 맵)

작업	설명	수행 방법
마스터 KDC를 구성합니다.	수동 프로세스와 KDC용 LDAP을 사용하여 영역에 대한 마스터 KDC 서버와 데이터베이스를 구성하고 구축합니다.	360 페이지 “LDAP 데이터 서버를 사용하도록 KDC를 구성하는 방법”
Kerberos 주체 속성과 비Kerberos 객체 클래스 유형을 함께 사용합니다.	Kerberos 레코드와 함께 저장된 정보가 다른 LDAP 데이터베이스와 공유될 수 있도록 합니다.	423 페이지 “비Kerberos 객체 클래스 유형에서 Kerberos 주체 속성을 함께 사용하는 방법”
영역을 삭제합니다.	영역에 연결된 데이터를 모두 제거합니다.	423 페이지 “LDAP 디렉토리 서버에서 영역 삭제 방법”

▼ 비Kerberos 객체 클래스 유형에서 Kerberos 주체 속성을 함께 사용하는 방법

이 절차에서는 비Kerberos 객체 클래스 유형에 Kerberos 주체 속성을 연결할 수 있도록 합니다. 이 절차에서는 krbprincipalaux, krbTicketPolicyAux 및 krbPrincipalName 속성이 people 객체 클래스에 연결됩니다.

이 절차에서는 다음 구성 매개변수가 사용됩니다.

- 디렉토리 서버 = dsserver.example.com
- 사용자 주체 = willf@EXAMPLE.COM

1 수퍼유저로 로그인합니다.

2 people 객체 클래스에서 각 항목을 준비합니다.

각 항목에 대해 이 단계를 반복합니다.

```
cat << EOF | ldapmodify -h dsserver.example.com -D "cn=directory manager"
dn: uid=willf,ou=people,dc=example,dc=com
changetype: modify
objectClass: krbprincipalaux
objectClass: krbTicketPolicyAux
krbPrincipalName: willf@EXAMPLE.COM
EOF
```

3 영역 컨테이너에 하위 트리 속성을 추가합니다.

이 단계는 ou=people,dc=example,dc=com 컨테이너와 기본 EXAMPLE.COM 컨테이너에서 주체 항목을 검색할 수 있도록 합니다.

```
# kdb5_ldap_util -D "cn=directory manager" modify \
            -subtrees 'ou=people,dc=example,dc=com' -r EXAMPLE.COM
```

4 (옵션) KDC 레코드가 DB2에 저장된 경우 DB2 항목을 마이그레이션합니다.

a. DB2 항목을 덤프합니다.

```
# kdb5_util dump > dumpfile
```

b. 데이터베이스를 LDAP 서버로 로드합니다.

```
# kdb5_util load -update dumpfile
```

5 (옵션) KDC에 주체 속성을 추가합니다.

```
# kadmin.local -q 'addprinc willf'
```

▼ LDAP 디렉토리 서버에서 영역 삭제 방법

이 절차는 다른 LDAP 디렉토리 서버가 영역을 처리하도록 구성된 경우 사용할 수 있습니다.

1 **수퍼유저로 로그인합니다.**

2 **영역을 삭제합니다.**

```
# kdb5_ldap_util -D "cn=directory manager" destroy
```

Kerberos 서버에서 보안 수준 향상

Kerberos 애플리케이션 서버 및 KDC 서버에서 보안 수준을 향상시키려면 다음 단계를 수행하십시오.

표 21–4 Kerberos 서버에서 보안 수준 향상(작업 맵)

작업	설명	수행 방법
Kerberos 인증을 사용한 액세스를 사용으로 설정합니다.	Kerberos 인증만 허용하도록 서버에 대한 네트워크 액세스를 제한합니다.	424 페이지 “Kerberos화된 응용 프로그램만 사용으로 설정하는 방법”
KDC 서버에 대한 액세스를 제한합니다.	KDC 서버 및 해당 데이터의 보안 수준을 향상시킵니다.	425 페이지 “KDC 서버에 대한 액세스 제한 방법”
사전 파일을 사용하여 암호 보안 수준을 향상시킵니다.	사전에 대해 새 암호를 확인하여 새 암호의 보안 수준을 향상시킵니다.	426 페이지 “사전 파일을 사용하여 암호 보안 수준을 향상시키는 방법”

▼ Kerberos화된 응용 프로그램만 사용으로 설정하는 방법

이 절차에서는 Kerberos 인증 트랜잭션만 사용하도록 `telnet`, `ftp`, `rcp`, `rsh` 및 `rlogin`을 실행 중인 서버에 대한 네트워크 액세스를 제한합니다.

1 **수퍼유저로 로그인합니다.**

2 **`telnet` 서비스에 대한 `exec` 등록 정보를 변경합니다.**

유효한 인증 정보를 제공할 수 있는 사용자로 액세스가 제한되도록 `telnet`에 대한 `exec` 등록 정보에 `-a user` 옵션을 추가합니다.

```
# inetadm -m svc:/network/telnet:default exec="/usr/sbin/in.telnetd -a user"
```

3 **(옵션) 아직 구성되지 않은 경우 `telnet` 서비스에 대한 `exec` 등록 정보를 변경합니다.**

Kerberos 인증 연결만 허용되도록 `ftp`에 대한 `exec` 등록 정보에 `-a` 옵션을 추가합니다.

```
# inetadm -m svc:/network/ftp:default exec="/usr/sbin/in.ftpd -a"
```

4 **다른 서비스를 사용 안함으로 설정합니다.**

`in.rshd` 및 `in.rlogind` 데몬은 사용 안함으로 설정해야 합니다.

```
# svcadm disable network/shell
# svcadm disable network/login:rlogin
```

▼ KDC 서버에 대한 액세스 제한 방법

마스터 KDC 서버와 슬레이브 KDC 서버에는 로컬에 저장된 KDC 데이터베이스의 복사본이 있습니다. 이러한 서버에 대한 액세스를 제한하여 데이터베이스 보안을 유지하는 것은 Kerberos 설치의 전반적인 보안을 위해 중요합니다.

1 **수퍼유저로 로그인합니다.**

2 **필요에 따라 원격 서비스를 사용 안함으로 설정합니다.**

보안 KDC 서버를 제공하려면 불필요한 모든 네트워크 서비스를 사용 안함으로 설정해야 합니다. 구성에 따라 해당 서비스 중 일부는 이미 사용 안함으로 설정되었을 수 있습니다. `svcs` 명령으로 서비스 상태를 확인합니다. 대부분의 경우 실행해야 할 유일한 서비스는 `krb5kdc`와 `krdb5_kprop`(KDC가 슬레이브인 경우)이거나 `kadmin`(KDC가 마스터인 경우)입니다. 또한 루프백 tli(`ticlts`, `ticotsord` 및 `ticots`)를 사용하는 서비스는 사용 상태로 유지할 수 있습니다.

```
# svcadm disable network/comsat
# svcadm disable network/dtspc/tcp
# svcadm disable network/finger
# svcadm disable network/login:rlogin
# svcadm disable network/rexec
# svcadm disable network/shell
# svcadm disable network/talk
# svcadm disable network/tname
# svcadm disable network/uucp
# svcadm disable network/rpc_100068_2-5/rpc_udp
```

3 **KDC를 지원하는 하드웨어에 대한 액세스를 제한합니다.**

물리적 액세스를 제한하려면 KDC 서버 및 해당 모니터가 보안 설비에 있어야 합니다. 사용자는 어떤 방식으로도 이 서버에 액세스할 수 없어야 합니다.

4 **로컬 디스크 또는 KDC 슬레이브에서 KDC 데이터베이스 백업을 저장합니다.**

테이프가 안전하게 저장된 경우에만 KDC의 테이프 백업을 수행합니다. keytab 파일의 복사본에 대해 동일한 단계를 수행합니다. 다른 시스템과 공유되지 않는 로컬 파일 시스템에 해당 파일을 저장하는 것이 좋습니다. 저장소 파일 시스템은 마스터 KDC 서버 또는 슬레이브 KDC에 있을 수 있습니다.

▼ 사전 파일을 사용하여 암호 보안 수준을 향상시키는 방법

Kerberos 서비스는 새 자격 증명을 만들 때 사전 파일을 사용하여 사전에 있는 단어가 암호로 사용되지 못하도록 할 수 있습니다. 사전에 있는 용어를 암호로 사용하지 못하도록 하면 다른 사람이 암호를 쉽게 추측할 수 없습니다. 기본적으로 `/var/krb5/kadm5.dict` 파일이 사용되지만 비어 있습니다.

1 마스터 KDC에서 수퍼유저로 로그인합니다.

2 KDC 구성 파일(`kdc.conf`)을 편집합니다.

서비스에 사전 파일을 사용하도록 알릴 행을 추가해야 합니다. 이 예에서는 `spell` 유틸리티와 함께 포함된 사전이 사용됩니다. 구성 파일에 대한 자세한 설명은 `kdc.conf(4)` 매뉴얼 페이지를 참조하십시오.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM = {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                sunw_dbprop_enable = true
                sunw_dbprop_master_ulogsize = 1000
                dict_file = /usr/share/lib/dict/words
                }
```

3 Kerberos 데몬을 다시 시작합니다.

```
kdc1 # svcadm restart -r network/security/krb5kdc
kdc1 # svcadm restart -r network/security/kadmin
```

◆ ◆ ◆ 22장

22 Kerberos 오류 메시지 및 문제 해결

이 장에서는 Kerberos 서비스를 사용할 때 표시될 수 있는 오류 메시지에 대한 해결 방법을 제공합니다. 또한 다양한 문제에 대한 몇 가지 문제 해결 팁도 제공합니다. 이 장에서 다루는 오류 메시지 및 문제 해결 정보는 다음과 같습니다.

Kerberos 오류 메시지

이 절에서는 각 오류가 발생하는 이유와 해결 방법을 비롯하여 Kerberos 오류 메시지에 대한 정보를 제공합니다.

SEAM 도구 오류 메시지

`Unable to view the list of principals or policies; use the Name field.`

원인: 로그인한 `admin` 주체가 Kerberos ACL 파일(`kadm5.acl`)에서 나열 권한(`l`)을 보유하고 있지 않습니다. 따라서 주체 목록 또는 정책 목록을 볼 수 없습니다.

해결책: 이를 위해 Name(이름) 필드에 주체 및 정책 이름을 사용하거나, 적절한 권한이 있는 주체로 로그인해야 합니다.

`JNI: Java array creation failed`

`JNI: Java class lookup failed`

`JNI: Java field lookup failed`

`JNI: Java method lookup failed`

`JNI: Java object lookup failed`

`JNI: Java object field lookup failed`

`JNI: Java string access failed`

`JNI: Java string creation failed`

원인: SEAM 도구에서 사용하는 Java Native Interface(`gkadmin`)에 심각한 문제가 있습니다.

해결책: `gkadmin`을 종료한 후 다시 시작하십시오. 문제가 계속되면 버그를 보고하십시오.

일반 Kerberos 오류 메시지(A-M)

이 절에서는 Kerberos 명령, Kerberos 데몬, PAM 프레임워크, GSS 인터페이스, NFS 서비스 및 Kerberos 라이브러리에 대한 일반 오류 메시지 목록(A-M)을 제공합니다.

`All authentication systems disabled; connection refused`

원인: 이 버전의 `rlogind`는 인증 방식을 지원하지 않습니다.

해결책: `rlogind`가 `-k` 옵션과 함께 호출되었는지 확인하십시오.

`Another authentication mechanism must be used to access this host`

원인: 인증을 수행할 수 없습니다.

해결책: 클라이언트가 Kerberos V5 인증 방식을 사용하고 있는지 확인하십시오.

`Authentication negotiation has failed, which is required for encryption. Good bye.`

원인: 서버와 인증을 협상할 수 없습니다.

해결책: `telnet` 명령을 `toggle authdebug` 명령과 함께 호출하여 인증 디버깅을 시작하고, 추가 내용은 디버그 메시지를 확인하십시오. 또한 유효한 자격 증명이 있는지도 확인하십시오.

`Bad krb5 admin server hostname while initializing kadmin interface`

원인: `krb5.conf` 파일에서 `admin_server`에 대해 잘못된 호스트 이름이 구성되었습니다.

해결책: 마스터 KDC에 대해 올바른 호스트 이름이 `krb5.conf` 파일의 `admin_server` 행에 지정되었는지 확인하십시오.

`Bad lifetime value`
원인: 제공된 수명 값이 유효하지 않거나 올바르지 않은 형식입니다.

해결책: 제공된 값이 `kinit(1)` 매뉴얼 페이지의 Time Formats 절과 일치하는지 확인하십시오.

`Bad start time value`
원인: 제공된 시작 시간 값이 유효하지 않거나 올바르지 않은 형식입니다.

해결책: 제공된 값이 `kinit(1)` 매뉴얼 페이지의 Time Formats 절과 일치하는지 확인하십시오.

`Cannot contact any KDC for requested realm`
원인: 요청한 영역에 응답하는 KDC가 없습니다.

해결책: 적어도 하나의 KDC(마스터 또는 슬레이브)에 연결 가능한지 또는 `krb5kdc` 데몬이 KDC에서 실행 중인지 확인하십시오. `/etc/krb5/krb5.conf` 파일에서 구성된 KDC(`kdc` = *kdc-name*)의 목록을 확인하십시오.

`Cannot determine realm for host: host is '`*hostname*`'`
원인: Kerberos가 호스트에 대한 영역 이름을 확인할 수 없습니다.

해결책: 기본 영역 이름이 있는지 또는 도메인 이름 매핑이 Kerberos 구성 파일(`krb5.conf`)에 설정되었는지 확인하십시오.

`Cannot find a kadmin KDC entry in krb5.conf(4) or DNS Service Location records for realm '` *realmname*`'`

`Cannot find a kpassword KDC entry in krb5.conf(4) or DNS Service Location records for realm '`*realmname*`'`

`Cannot find a master KDC entry in krb5.conf(4) or DNS Service Location records for realm '`*realmname*`'`

`Cannot find any KDC entries in krb5.conf(4) or DNS Service Location records for realm '`*realmname*`'`
원인: `krb5.conf` 파일 또는 DNS 서버 레코드가 올바르지 않게 구성되었습니다.

해결책: Kerberos 구성 파일(`/etc/krb5/krb5.conf`) 또는 KDC에 대한 DNS 서버 레코드가 제대로 구성되었는지 확인하십시오.

`Cannot find address for '`*hostname*`': '`*error-string*`'`
원인: 지정된 호스트 이름에 대한 DNS 레코드에서 주소를 찾을 수 없습니다.

해결책: DNS에서 호스트 레코드를 수정하거나, DNS 조회 프로세스에서 오류를 수정하십시오.

`Cannot find KDC for requested realm`
원인: 요청한 영역에서 KDC를 찾을 수 없습니다.

해결책: Kerberos 구성 파일(krb5.conf)의 realm 섹션에 KDC가 지정되었는지 확인하십시오.

`cannot initialize realm` *realm-name*

원인: KDC에 stash 파일이 없는 것일 수 있습니다.

해결책: KDC에 stash 파일이 있는지 확인하십시오. 없을 경우 kdb5_util 명령을 사용하여 stash 파일을 만든 다음 krb5kdc 명령을 다시 시작하십시오.

`Cannot resolve KDC for requested realm`

원인: Kerberos가 영역에 대한 KDC를 확인할 수 없습니다.

해결책: Kerberos 구성 파일(krb5.conf)의 realm 섹션에 KDC가 지정되었는지 확인하십시오.

`Cannot resolve network address for KDCs '`*hostname*`' discovered via DNS Service Location records for realm '`*realm-name*`'`

`Cannot resolve network address for KDCs '`*hostname*`' specified in krb5.conf(4) for realm '`*realm-name*`'`

원인: krb5.conf 파일 또는 DNS 서버 레코드가 올바르지 않게 구성되었습니다.

해결책: Kerberos 구성 파일(/etc/krb5/krb5.conf) 및 KDC에 대한 DNS 서버 레코드가 제대로 구성되었는지 확인하십시오.

`Cannot reuse password`

원인: 지정한 암호가 이전에 이 주체에 의해 사용되었습니다.

해결책: 이전에 선택한 적이 없는 암호를 선택하되, 암호가 적어도 KDC 데이터베이스에 주체별로 보존된 암호 수 내에 있지 않아야 합니다. 이 정책은 주체 정책에 따라 적용됩니다.

`Can't get forwarded credentials`

원인: 자격 증명 전달을 설정할 수 없습니다.

해결책: 주체가 전달 가능 자격 증명을 보유하고 있는지 확인하십시오.

`Can't open/find Kerberos configuration file`

원인: Kerberos 구성 파일(krb5.conf)을 사용할 수 없습니다.

해결책: krb5.conf 파일이 올바른 위치에서 사용 가능하며 올바른 권한을 보유하고 있는지 확인하십시오. root는 이 파일을 쓸 수 있어야 하며 그 외의 다른 사용자는 읽을 수 있어야 합니다.

`Client 'principal ' not found in Kerberos database`
원인: Kerberos 데이터베이스에서 주체가 누락되었습니다.

해결책: Kerberos 데이터베이스에 클라이언트 주체를 추가하십시오.

`Client 'principal' pre-authentication failed`
원인: 주체에 대한 사전 인증이 실패했습니다.

해결책: 사용자가 올바른 암호를 사용하고 있는지 확인하십시오.

`Client did not supply required checksum--connection rejected`
원인: 체크섬을 사용하는 인증이 클라이언트와 협상되지 않았습니다. 클라이언트가 초기 연결을 지원하지 않는 이전 Kerberos V5 프로토콜을 사용하고 있을 수 있습니다.

해결책: 클라이언트가 초기 연결을 지원하는 Kerberos V5 프로토콜을 사용하고 있는지 확인하십시오.

`Client/server realm mismatch in initial ticket request: 'client-principal' requesting ticket 'service-principal'`
원인: 초기 티켓 영역에서 클라이언트와 서버 간 영역 불일치가 발생했습니다.

해결책: 통신 중인 서버가 클라이언트와 동일한 영역에 있는지 또는 영역 구성이 올바른지 확인하십시오.

`Client or server has a null key`
원인: 주체에 널 키가 있습니다.

해결책: `kadmin`의 `cpw` 명령을 사용하여 주체가 널이 아닌 키를 갖도록 수정하십시오.

`Clock skew too great: 'client' requesting ticket 'service-principal ' from KDC 'KDC-hostname' ( KDC-time). Skew is value`

`Clock skew too great: 'client' AP request with ticket for 'service-principal '. Skew is value (allowable value)`
원인: 클라이언트와 KDC 서버 또는 애플리케이션 서버에 보고된 시간 차가 너무 큽니다.

해결책: 시계가 동기화 상태를 유지하도록 NTP(네트워크 시간 프로토콜)를 구성하십시오. 자세한 내용은 400 페이지 “KDC와 Kerberos 클라이언트 간의 클럭 동기화”를 참조하십시오.

`Communication failure with server while initializing kadmin interface`
원인: 관리 서버(마스터 KDC라고도 함)에 대해 지정된 호스트에서 `kadmind` 데몬이 실행 중이지 않습니다.

해결책: 마스터 KDC에 대해 올바른 호스트 이름을 지정했는지 확인하십시오. 올바른 호스트 이름을 지정한 경우, 지정한 마스터 KDC에서 `kadmind`이 실행 중인지 확인하십시오.

`Credentials cache file permissions incorrect`
원인: 자격 증명 캐시(`/tmp/krb5cc_`*uid*)에 대해 적합한 읽기 또는 쓰기 권한을 가지고 있습니다.

해결책: 자격 증명 캐시에 대한 읽기 또는 쓰기 권한이 있는지 확인하십시오.

`Credentials cache I/O operation failed` *XXX*
원인: 시스템의 자격 증명 캐시(`/tmp/krb5cc_`*uid*)에 쓰는 중 Kerberos에서 문제가 발생했습니다.

해결책: 자격 증명 캐시가 제거되었는지, `df` 명령을 사용하여 장치에 남은 공간이 있는지 확인하십시오.

`Decrypt integrity check failed`
원인: 잘못된 티켓이 있을 수 있습니다.

해결책: 다음 두 조건을 확인하십시오.

- 자격 증명이 유효한지 확인하십시오. `kdestroy`를 사용하여 티켓을 삭제한 다음 `kinit`를 사용하여 티켓을 새로 만드십시오.
- 대상 호스트에 서비스 키의 버전이 올바른 keytab 파일이 있는지 확인하십시오. Kerberos 데이터베이스에서 서비스 주체(예: `host/`*FQDN-hostname*)의 키 버전 번호를 확인하려면 `kadmin`을 사용하십시오. 또한 대상 호스트에서 `klist -k`를 사용하여 동일한 키 버전 번호를 갖는지도 확인하십시오.

`Decrypt integrity check failed for client 'principal' and server 'hostname'`
원인: 잘못된 티켓이 있을 수 있습니다.

해결책: 자격 증명이 유효한지 확인하십시오. `kdestroy` 명령을 사용하여 티켓을 삭제한 다음 `kinit` 명령을 사용하여 티켓을 새로 만드십시오.

`Encryption could not be enabled. Goodbye.`
원인: 서버와 암호화를 협상할 수 없습니다.

해결책: `telnet` 명령을 `toggle encdebug` 명령과 함께 호출하여 인증 디버깅을 시작하고, 추가 내용은 디버그 메시지를 확인하십시오.

`Failed to find realm for` *principal* `in keytab`
원인: *principal*에 포함된 영역 이름이 keytab 파일에 저장된 주체의 영역 이름과 일치하지 않습니다.

해결책: 주체가 올바른 영역을 사용하고 있는지 확인하십시오.

`failed to obtain credentials cache`
원인: `kadmin` 초기화 중 `kadmin`이 `admin` 주체에 대한 자격 증명을 얻으려고 시도했는데 오류가 발생했습니다.

해결책: `kadmin`을 실행할 때 올바른 주체와 암호를 사용했는지 확인하십시오.

`Field is too long for this implementation`
원인: Kerberos화된 응용 프로그램에서 전송 중인 메시지 크기가 너무 큽니다. 전송 프로토콜이 UDP인 경우 이 오류가 발생할 수 있습니다. 이 경우 기본 최대 메시지 크기는 65535바이트입니다. 또한 Kerberos 서비스에서 전송한 프로토콜 메시지 내에 개별 필드에 대한 제한이 있습니다.

해결책: KDC 서버의 `/etc/krb5/kdc.conf` 파일에서 전송을 UDP로 제한하지 않았는지 확인하십시오.

`GSS-API (or Kerberos) error`
원인: 이 메시지는 일반 GSS-API 또는 Kerberos 오류 메시지로, 서로 다른 여러 문제로 인해 발생할 수 있습니다.

해결책: `/var/krb5/kdc.log` 파일을 확인하여 이 오류가 발생했을 때 기록된 더 구체적인 오류 메시지를 찾으십시오.

`Hostname cannot be canonicalized for '`*`hostname`*`': '`*`error-string`*`'`
원인: Kerberos 클라이언트가 서버에 대한 완전 수식 호스트 이름을 찾을 수 없습니다.

해결책: 서버 호스트 이름이 DNS에 정의되어 있는지, 호스트 이름-주소 및 주소-호스트 이름 매핑이 일치하는지 확인하십시오.

`Illegal cross-realm ticket`
원인: 전송한 티켓에 올바른 상호 영역이 없습니다. 영역에 올바른 신뢰 관계가 설정되지 않았을 수 있습니다.

해결책: 사용 중인 영역에 올바른 신뢰 관계가 설정되었는지 확인하십시오.

`Improper format of Kerberos configuration file`
원인: Kerberos 구성 파일에 잘못된 항목이 있습니다.

해결책: `krb5.conf` 파일의 모든 관계 뒤에 "=" 기호와 값이 있는지 확인하십시오. 또한 각 하위 섹션에 대한 쌍에 대괄호가 있는지도 확인하십시오.

`Inappropriate type of checksum in message`
원인: 메시지에 잘못된 체크섬 유형이 포함되었습니다.

해결책: `krb5.conf` 및 `kdc.conf` 파일에 유효한 체크섬 유형이 지정되었는지 확인하십시오.

`Incorrect net address`
원인: 네트워크 주소 불일치가 있습니다. 전달 중이었던 티켓의 네트워크 주소가 티켓이 처리된 네트워크 주소와 다릅니다. 이 메시지는 티켓 전송 중에 표시될 수 있습니다.

해결책: 네트워크 주소가 올바른지 확인하십시오. `kdestroy`를 사용하여 티켓을 삭제한 다음 `kinit`를 사용하여 티켓을 새로 만드십시오.

`Invalid credential was supplied`

`Service key not available`
원인: 자격 증명 캐시에 있는 서비스 티켓이 올바르지 않을 수 있습니다.

해결책: 이 서비스를 사용하기 전에 현재 자격 증명 캐시를 삭제한 다음 `kinit`를 다시 실행하십시오.

`Invalid flag for file lock mode`
원인: 내부 Kerberos 오류가 발생했습니다.

해결책: 버그를 보고하십시오.

`Invalid message type specified for encoding`
원인: Kerberos가 Kerberos화된 응용 프로그램에서 전송한 메시지 유형을 인식할 수 없습니다.

해결책: 사이트 또는 공급업체에서 개발한 Kerberos화된 응용 프로그램을 사용 중인 경우, Kerberos를 올바르게 사용하고 있는지 확인하십시오.

`Invalid number of character classes`
원인: 주체에 대해 지정한 암호가 주체 정책에서 요구하는 충분한 암호 클래스를 포함하고 있지 않습니다.

해결책: 정책이 요구하는 최소 암호 클래스 수를 갖는 암호를 지정했는지 확인하십시오.

`KADM err: Memory allocation failure`
원인: `kadmin`을 실행하기에 메모리가 부족합니다.

해결책: 메모리를 해제한 후 다시 `kadmin`을 실행해 보십시오.

`kadmin: Bad encryption type while changing host/`*FQDN*`'s key`
원인: Solaris 10 8/07 릴리스 이후부터는 기본 릴리스에 더 많은 기본 암호화 유형이 포함됩니다. 이전 버전의 소프트웨어에서 실행 중인 KDC에서는 지원되지 않는 암호화 유형을 클라이언트가 요청할 수 있습니다.

해결책: 이 문제를 해결하기 위한 몇 가지 해결 방법이 있습니다. 구현하기 가장 쉬운 방법부터 나열됩니다.

1. SUNWcry 및 SUNWcryr 패키지를 KDC 서버에 추가합니다. 그러면 KDC에서 지원하는 암호화 유형 수가 늘어납니다.
2. 클라이언트의 `krb5.conf`에서 `permitted_enctypes`를 설정하여 `aes256` 암호화 유형이 포함되지 않도록 합니다. 이 단계는 새 클라이언트마다 수행해야 합니다.

`KDC can't fulfill requested option`

원인: KDC에서 요청한 옵션을 허용하지 않습니다. 후일자 또는 전달 가능 옵션을 요청했는데 KDC에서 이를 허용하지 않는 것이 문제일 수 있습니다. 또한 TGT 갱신을 요청했지만 갱신 가능 TGT가 없는 것도 문제일 수 있습니다.

해결책: KDC에서 허용하지 않는 옵션 또는 사용할 수 없는 티켓의 유형을 요청하고 있는지 확인하십시오.

`KDC policy rejects request`

원인: KDC 정책이 요청을 허용하지 않습니다. 예를 들어 KDC에 대한 요청에 IP 주소가 없습니다. 전달을 요청했는데 KDC에서 이를 허용하지 않습니다.

해결책: 올바른 옵션과 함께 `kinit`를 사용하고 있는지 확인하십시오. 필요한 경우 주체와 연관된 정책을 수정하거나, 요청을 허용하도록 주체의 속성을 변경하십시오. `kadmin`을 사용하여 정책이나 주체를 수정할 수 있습니다.

`KDC reply did not match expectation: KDC not found. Probably got an unexpected realm referral`

원인: KDC 응답에 예상 주체 이름이 포함되어 있지 않거나, 응답의 다른 값이 올바르지 않습니다.

해결책: 통신 중인 KDC가 RFC4120을 준수하는지, 전송하는 요청이 Kerberos V5 요청인지, KDC가 사용 가능한지 확인하십시오.

`kdestroy: Could not obtain principal name from cache`

원인: 자격 증명 캐시가 누락되었거나 손상되었습니다.

해결책: 제공된 캐시 위치가 올바른지 확인하십시오. TGT를 제거하고 필요한 경우 `kinit`를 사용하여 새 TGT를 얻으십시오.

`kdestroy: No credentials cache file found while destroying cache`

원인: 자격 증명 캐시(`/tmp/krb5c_` *uid*)가 누락되었거나 손상되었습니다.

해결책: 제공된 캐시 위치가 올바른지 확인하십시오. TGT를 제거하고 필요한 경우 `kinit`를 사용하여 새 TGT를 얻으십시오.

`kdestroy: TGT expire warning NOT deleted`

원인: 자격 증명 캐시가 누락되었거나 손상되었습니다.

해결책: 제공된 캐시 위치가 올바른지 확인하십시오. TGT를 제거하고 필요한 경우 `kinit`를 사용하여 새 TGT를 얻으십시오.

`Kerberos authentication failed`
원인: Kerberos 암호가 올바르지 않거나, 암호가 UNIX 암호와 동기화되지 않았을 수 있습니다.

해결책: 암호가 동기화되지 않은 경우 다른 암호를 지정하여 Kerberos 인증을 완료해야 합니다. 사용자가 자신의 원래 암호를 잊어버렸을 수 있습니다.

`Kerberos V5 refuses authentication`
원인: 서버와 인증을 협상할 수 없습니다.

해결책: `telnet` 명령을 `toggle authdebug` 명령과 함께 호출하여 인증 디버깅을 시작하고, 추가 내용은 디버그 메시지를 확인하십시오. 또한 유효한 자격 증명이 있는지도 확인하십시오.

`Key table entry not found`
원인: 네트워크 애플리케이션 서버의 keytab 파일에 서비스 주체에 대한 항목이 없습니다.

해결책: Kerberos화된 서비스를 제공할 수 있도록 적합한 서비스 주체를 서버의 keytab 파일에 추가하십시오.

`Key table file '`*filename*`' not found`
원인: 지정된 키 테이블 파일이 없습니다.

해결책: 키 테이블 파일을 만드십시오.

`Key version` *number* `is not available for principal` *principal*
원인: 키 버전이 애플리케이션 서버의 키 버전과 일치하지 않습니다.

해결책: `klist -k` 옵션을 사용하여 애플리케이션 서버의 키 버전을 확인하십시오.

`Key version number for principal in key table is incorrect`
원인: keytab 파일의 주체 키 버전이 Kerberos 데이터베이스의 버전과 다릅니다. 서비스 키가 변경되었거나 이전 서비스 티켓을 사용하고 있을 수 있습니다.

해결책: 서비스 키가 변경된 경우(예: `kadmin`을 사용하여), 새 키를 추출한 다음 서비스가 실행 중인 호스트 keytab 파일에 저장해야 합니다.

또는 이전 키를 포함하는 이전 서비스 티켓을 사용하고 있을 수 있습니다. `kdestroy` 명령을 실행한 다음 `kinit` 명령을 다시 실행할 수 있습니다.

`kinit: gethostname failed`
원인: 로컬 네트워크 구성의 오류로 인해 `kinit`가 실패했습니다.

해결책: 호스트가 올바르게 구성되었는지 확인하십시오.

`login: load_modules: can not open module /usr/lib/security/pam_krb5.so.1`
원인: Kerberos PAM 모듈이 누락되었거나 유효한 실행 파일 이진이 아닙니다.

해결책: Kerberos PAM 모듈이 `/usr/lib/security` 디렉토리에 있으며 유효한 실행 파일 이진인지 확인하십시오. 또한 `/etc/pam.conf` 파일이 `pam_krb5.so.1`에 대한 올바른 경로를 포함하고 있는지도 확인하십시오.

`Looping detected getting initial creds: 'client-principal' requesting ticket 'service-principal'. Max loops is value. Make sure a KDC is available.`
원인: Kerberos가 초기 티켓을 얻으려고 여러 번 시도했지만 실패했습니다.

해결책: 적어도 하나의 KDC가 인증 요청에 응답하는지 확인하십시오.

`Master key does not match database`
원인: 로드된 데이터베이스 덤프가 마스터 키를 포함하는 데이터베이스에서 생성되지 않았습니다. 마스터 키는 `/var/krb5/.k5.`*REALM*에 있습니다.

해결책: 로드된 데이터베이스 덤프의 마스터 키가 `/var/krb5/.k5.`*REALM*에 있는 마스터 키와 일치하는지 확인하십시오.

`Matching credential not found`
원인: 요청의 일치하는 자격 증명을 찾을 수 없습니다. 자격 증명 캐시에 사용할 수 없는 자격 증명이 요청에 필요합니다.

해결책: `kdestroy`를 사용하여 티켓을 삭제한 다음 `kinit`를 사용하여 티켓을 새로 만드십시오.

`Message out of order`
원인: 순차적 프라이버시를 사용하여 전송된 메시지가 잘못된 순서로 도착했습니다. 전송 중 일부 메시지가 손실되었을 수 있습니다.

해결책: Kerberos 세션을 다시 초기화해야 합니다.

`Message stream modified`
원인: 계산된 체크섬과 메시지 체크섬 간에 불일치가 있습니다. 전송 중 메시지가 수정되었을 수 있는데, 이는 보안 누출을 나타내는 것일 수 있습니다.

해결책: 메시지가 네트워크를 통해 올바르게 전송되고 있는지 확인하십시오. 이 메시지는 또한 전송 중에 메시지가 변조되었을 가능성을 나타내는 것일 수도 있으므로, `kdestroy`를 사용하여 티켓을 삭제한 다음 사용 중인 Kerberos 서비스를 다시 초기화하십시오.

일반 Kerberos 오류 메시지(N-Z)

이 절에서는 Kerberos 명령, Kerberos 데몬, PAM 프레임워크, GSS 인터페이스, NFS 서비스 및 Kerberos 라이브러리에 대한 일반 오류 메시지 목록(N-Z)을 제공합니다.

`No credentials cache file found`
원인: Kerberos가 자격 증명 캐시(`/tmp/krb5cc_`*uid*)를 찾을 수 없습니다.

해결책: 자격 증명 파일이 있으며 읽기 가능한지 확인하십시오. 그렇지 않을 경우 `kinit`를 다시 수행해 보십시오.

`No credentials were supplied, or the credentials were unavailable or inaccessible`

`No credential cache found`
원인: 사용자의 자격 증명 캐시가 올바르지 않거나 없습니다.

해결책: 서비스를 시작하기 전에 사용자가 `kinit`를 실행해야 합니다.

`No credentials were supplied, or the credentials were unavailable or inaccessible`

`No principal in keytab (' `*filename*`') matches desired name` *principal*
원인: 서버 인증 중 오류가 발생했습니다.

해결책: 호스트 또는 서비스 주체가 서버의 Keytab 파일에 있는지 확인하십시오.

`Operation requires "`*privilege*`" privilege`
원인: 사용 중인 `admin` 주체가 `kadm5.acl` 파일에 구성된 적합한 권한을 보유하고 있지 않습니다.

해결책: 적합한 권한이 있는 주체를 사용하십시오. 또는 `kadm5.acl` 파일을 수정하여 사용 중인 주체가 적합한 권한을 보유하도록 구성하십시오. 보통 `/admin`을 이름의 일부로 사용하는 주체는 적합한 권한을 보유하고 있습니다.

`PAM-KRB5 (auth): krb5_verify_init_creds failed: Key table entry not found`
원인: 원격 응용 프로그램이 로컬 `/etc/krb5/krb5.keytab` 파일에 있는 호스트의 서비스 주체를 읽으려고 했는데, 해당 주체가 존재하지 않습니다.

해결책: 호스트의 Keytab 파일에 호스트의 서비스 주체를 추가하십시오.

`Password is in the password dictionary`
원인: 지정한 암호가 사용 중인 암호 사전에 있습니다. 이 암호는 적합한 암호가 아닙니다.

해결책: 암호 클래스가 혼합된 암호를 선택하십시오.

`Permission denied in replay cache code`
원인: 시스템의 재생 캐시를 열 수 없습니다. 서버가 현재 사용자 ID 대신 다른 사용자 ID로 처음 실행되었을 수 있습니다.

해결책: 재생 캐시에 적합한 권한이 있는지 확인하십시오. 재생 캐시는 Kerberos화된 응용 프로그램이 실행 중인 호스트에 저장됩니다. 비`root` 사용자의 재생 캐시 파일은 `/var/krb5/rcache/rc_`*service_name*`_`*uid*입니다. 루트 사용자의 재생 캐시 파일은 `/var/krb5/rcache/root/rc_` *service_name*입니다.

`Protocol version mismatch`
원인: Kerberos V4 요청이 KDC로 전송되었을 가능성이 가장 높습니다. Kerberos 서비스는 Kerberos V5 프로토콜만 지원합니다.

해결책: 응용 프로그램이 Kerberos V5 프로토콜을 사용하고 있는지 확인하십시오.

`Request is a replay`
원인: 요청이 이미 이 서버로 전송되어 처리되었습니다. 티켓을 도난 당했을 수 있으며 다른 사람이 티켓을 재사용하려고 합니다.

해결책: 잠시 기다렸다가 요청을 다시 발행하십시오.

`Requested principal and ticket don't match: Requested principal is '`*service-principal*`' and TGT principal is '`*TGT-principal*`'`
원인: 연결 중인 서비스 주체와 보유하고 있는 서비스 티켓이 일치하지 않습니다.

해결책: DNS가 올바르게 작동하는지 확인하십시오. 다른 공급업체의 소프트웨어를 사용 중인 경우 해당 소프트웨어가 주체 이름을 올바르게 사용하고 있는지 확인하십시오.

`Requested protocol version not supported`
원인: Kerberos V4 요청이 KDC로 전송되었을 가능성이 가장 높습니다. Kerberos 서비스는 Kerberos V5 프로토콜만 지원합니다.

해결책: 응용 프로그램이 Kerberos V5 프로토콜을 사용하고 있는지 확인하십시오.

`Service key `*service-principal*` not available`
원인: 이름이 지정된 서비스 주체가 애플리케이션 서버의 Keytab 파일에 없습니다.

해결책: 서비스 주체가 일치하는지 또는 애플리케이션 서버의 Keytab 파일에 포함되었는지 확인하십시오.

`Server refused to negotiate authentication, which is required for encryption. Good bye.`
원인: 원격 응용 프로그램이 클라이언트의 Kerberos 인증을 수락할 수 없거나, 수락하지 않도록 구성되었습니다.

해결책: 인증 협상이 가능한 원격 응용프로그램을 제공하거나, 인증을 설정하는 적합한 플래그를 사용하도록 응용 프로그램을 구성하십시오.

`Server refused to negotiate encryption. Good bye.`
원인: 서버와 암호화를 협상할 수 없습니다.

해결책: `telnet` 명령을 `toggle encdebug` 명령과 함께 호출하여 인증 디버깅을 시작하고, 추가 내용은 디버그 메시지를 확인하십시오.

`Server rejected authentication (during sendauth exchange)`
원인: 통신하려는 서버가 인증을 거부했습니다. 이 오류는 대개 Kerberos 데이터베이스 전파 중에 발생합니다. 몇 가지 공통된 원인으로 인해 `kpropd.acl` 파일, DNS 또는 keytab 파일 관련 문제가 발생할 수 있습니다.

해결책: `kprop`가 아닌 다른 응용 프로그램을 실행할 때 이 오류가 발생하는 경우 서버의 Keytab 파일이 올바른지 확인하십시오.

`Server` *service-principal* `not found in Kerberos database`
원인: 서비스 주체가 올바르지 않거나 주체 데이터베이스에서 누락되었습니다.

해결책: 서비스 주체가 올바르며 데이터베이스에 있는지 확인하십시오.

`Target name principal '` *principal*`' does not match` *service-principal*
원인: 사용 중인 서비스 주체가 애플리케이션 서버가 사용 중인 서비스 주체와 일치하지 않습니다.

해결책: 애플리케이션 서버에서 서비스 주체가 Keytab 파일에 포함되었는지 확인하십시오. 클라이언트의 경우 올바른 서비스 주체를 사용하고 있는지 확인하십시오.

`The ticket isn't for us`

`Ticket/authenticator don't match`
원인: 티켓과 인증자 간에 불일치가 있습니다. 요청의 주체 이름이 서비스 주체의 이름과 일치하지 않을 수 있습니다. 서비스에는 비FQDN 이름이 필요한데 주체의 FQDN 이름으로 티켓을 전송했거나, 서버에 FQDN 이름이 필요한데 비FQDN 이름을 전송했기 때문입니다.

해결책: `kprop`가 아닌 다른 응용 프로그램을 실행할 때 이 오류가 발생하는 경우 서버의 Keytab 파일이 올바른지 확인하십시오.

`Ticket expired`
원인: 티켓 시간이 만료되었습니다.

해결책: `kdestroy`를 사용하여 티켓을 삭제한 다음 `kinit`를 사용하여 티켓을 새로 만드십시오.

`Ticket is ineligible for postdating`
원인: 주체가 후일자 티켓을 허용하지 않습니다.

해결책: `kadmin`을 사용하여 후일자를 허용하도록 주체를 수정하십시오.

`Ticket not yet valid: '`*client-principal* `' requesting ticket '`*service-principal*`' from '`*kdc-hostname*`' (`*time*`). TGT start time is `*time*`.`
원인: 후일자 티켓이 아직 유효하지 않습니다.

해결책: 올바른 날짜를 사용하여 새 티켓을 만들거나, 현재 티켓이 유효해질 때까지 기다리십시오.

`Truncated input file detected`
원인: 작업에 사용된 데이터베이스 덤프 파일이 완전한 덤프 파일이 아닙니다.

해결책: 덤프 파일을 다시 만들거나, 다른 데이터베이스 덤프 파일을 사용하십시오.

`Unable to securely authenticate user ... exit`
원인: 서버와 인증을 협상할 수 없습니다.

해결책: `telnet` 명령을 `toggle authdebug` 명령과 함께 호출하여 인증 디버깅을 시작하고, 추가 내용은 디버그 메시지를 확인하십시오. 또한 유효한 자격 증명이 있는지도 확인하십시오.

`Unknown encryption type: `*name*
원인: 자격 증명에 포함된 암호화 유형을 사용할 수 없습니다.

해결책: `klist -e` 명령을 사용하여 클라이언트에서 사용할 암호화 유형을 결정하십시오. 애플리케이션 서버가 적어도 하나의 암호화 유형을 지원하는지 확인하십시오.

`Wrong principal in request`
원인: 티켓에 잘못된 주체 이름이 있습니다. 이 오류는 DNS 또는 FQDN 문제를 나타내는 것일 수 있습니다.

해결책: 서비스 주체가 티켓의 주체와 일치하는지 확인하십시오.

Kerberos 문제 해결

이 절에서는 Kerberos 소프트웨어에 대한 문제 해결 정보를 제공합니다.

▼ 키 버전 번호로 문제를 식별하는 방법

KDC에서 사용하는 키 버전 번호(KVNO)와 시스템에 호스트된 서비스의 `/etc/krb5/krb5.keytab`에 저장된 서비스 주체 키가 일치하지 않는 경우가 있습니다. 키 테이블 파일을 새 키로 업데이트하지 않은 상태에서 KDC에 새 키 집합이 생성된 경우 KVNO가 동기화되지 않을 수 있습니다. 이 문제는 다음 절차를 사용하여 진단할 수 있습니다.

1 `keytab` 항목을 나열합니다.

각 주체의 KVNO가 목록에 포함됩니다.

```
# klist -k
Keytab name: FILE:/etc/krb5/krb5.keytab
KVNO Principal
---- --------------------------------------------------------------------------
   2 host/denver.example.com@EXAMPLE.COM
   2 host/denver.example.com@EXAMPLE.COM
   2 host/denver.example.com@EXAMPLE.COM
   2 nfs/denver.example.com@EXAMPLE.COM
   2 nfs/denver.example.com@EXAMPLE.COM
   2 nfs/denver.example.com@EXAMPLE.COM
   2 nfs/denver.example.com@EXAMPLE.COM
```

2 `host` 키를 사용하여 초기 자격 증명을 확보합니다.

```
# kinit -k
```

3 KDC에서 사용하는 KVNO를 확인합니다.

```
# kvno nfs/denver.example.com
nfs/denver.example.com@EXAMPLE.COM: kvno = 3
```

여기에 나열된 KVNO는 2 대신 3입니다.

krb5.conf 파일의 형식 관련 문제

`krb5.conf` 파일의 형식이 올바르지 않은 경우, 다음과 같은 오류 메시지가 터미널 창에 표시되거나 로그 파일에 기록됩니다.

```
Improper format of Kerberos configuration file while initializing krb5 library
```

`krb5.conf` 파일의 형식과 관련된 문제가 있을 경우 연관된 서비스가 공격에 취약할 수 있습니다. Kerberos 기능을 사용하도록 허용하기 전에 문제를 해결해야 합니다.

Kerberos 데이터베이스 전파 관련 문제

Kerberos 데이터베이스 전파에 실패한 경우 슬레이브 KDC와 마스터 KDC 간, 그리고 마스터 KDC에서 슬레이브 KDC 서버까지 `/usr/bin/rlogin -x`를 실행하십시오.

액세스를 제한하도록 KDC가 설정된 경우 rlogin은 사용 안함으로 설정되었으므로 이 문제를 해결하는 데 사용할 수 없습니다. KDC에서 rlogin을 사용으로 설정하려면 eklogin 서비스를 사용으로 설정해야 합니다.

```
# svcadm enable svc:/network/login:eklogin
```

문제 해결을 완료한 후에는 eklogin 서비스를 사용 안함으로 설정해야 합니다.

rlogin이 작동하지 않는 경우, KDC의 Keytab 파일 때문에 문제가 발생할 수 있습니다. rlogin 작동할 경우, rlogin 및 전파 소프트웨어에서는 동일한 host/ *host-name* 주체를 사용하기 때문에 Keytab 파일이나 이름 서비스에 문제가 있는 것이 아닙니다. 이 경우 kpropd.acl 파일이 올바른지 확인하십시오.

Kerberos화된 NFS 파일 시스템 마운트 관련 문제

- Kerberos화된 NFS 파일 시스템 마운트에 실패한 경우 /var/rcache/root 파일이 NFS 서버에 있는지 확인하십시오. root가 파일 시스템을 소유하고 있지 않은 경우 이를 제거하고 다시 마운트해 보십시오.
- Kerberos화된 NFS 파일 시스템에 액세스하는 중 문제가 있는 경우 사용자의 시스템과 NFS 서버에서 gssd 서비스가 사용으로 설정되었는지 확인하십시오.
- Kerberos화된 NFS 파일 시스템에 액세스하려고 할 때 **잘못된 인수** 또는 **잘못된 디렉토리** 오류 메시지가 표시된다면 NFS 파일 시스템을 마운트하려고 할 때 완전 수식 DNS 이름을 사용하는 것이 문제일 수 있습니다. 마운트되는 호스트가 서버 Keytab 파일에 있는 서비스 주체의 호스트 이름 부분과 같지 않습니다.

  서버에 이더넷 인터페이스가 여러 개 있는데 "호스트당 복수 주소 레코드" 체계 대신 "인터페이스당 이름" 체계를 사용하도록 DNS를 설정한 경우에도 이 문제가 발생할 수 있습니다. Kerberos 서비스의 경우 다음과 같이 호스트당 복수 주소 레코드를 설정해야 합니다.[1] :

```
my.host.name.   A       1.2.3.4
                A       1.2.4.4
                A       1.2.5.4

my-en0.host.name.       A       1.2.3.4
my-en1.host.name.       A       1.2.4.4
my-en2.host.name.       A       1.2.5.4

4.3.2.1         PTR     my.host.name.
4.4.2.1         PTR     my.host.name.
4.5.2.1         PTR     my.host.name.
```

이 예의 설정은 서로 다른 인터페이스에 대한 하나의 참조 및 서버 Keytab 파일의 세 서비스 주체 대신 단일 서비스 주체를 허용합니다.

[1] Ken Hornstein, “Kerberos FAQ,” [http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html#kerbdns], accessed 10 March 2010.

root 사용자로 인증 관련 문제

시스템에서 수퍼유저가 되려고 시도하면서 호스트의 Keytab 파일에 root 주체를 이미 추가했는데 인증에 실패할 경우, 확인해야 할 두 가지 잠재적인 문제가 있습니다. 먼저 Keytab 파일의 root 주체가 완전 수식 호스트 이름을 인스턴스로 사용하는지 확인하십시오. 그럴 경우 시스템이 DNS 클라이언트로 올바르게 설정되었는지 /etc/resolv.conf 파일을 확인하십시오.

GSS 자격 증명에서 UNIX 자격 증명으로 매핑

자격 증명 매핑을 모니터링할 수 있으려면 먼저 /etc/gss/gsscred.conf 파일에서 다음 행의 주석 처리를 해제하십시오.

```
SYSLOG_UID_MAPPING=yes
```

다음으로, gssd 서비스가 /etc/gss/gsscred.conf 파일에서 정보를 가져오도록 지정하십시오.

```
# pkill -HUP gssd
```

이제 gssd가 자격 증명 매핑을 요청하므로 자격 증명 매핑을 모니터링할 수 있어야 합니다. syslog.conf 파일이 debug 보안 레벨을 사용하여 auth 시스템 설비에 대해 구성된 경우, 매핑이 syslogd에 의해 기록됩니다.

Kerberos 서비스에서 DTrace 사용

이 예에서는 KDC에서 사전 인증이 필요한지, 필요할 경우 어떤 사전 인증 유형이 지원되는지 알고자 합니다. 먼저 권한이 있는 사용자로 로그인하여 다음과 같이 D 프로그램 소스 파일을 만드십시오.

```
# cat kerberos_preauth.d
kerberos$target:::krb_error-read
{
    self->preauth = args[1]->kerror_error_code ==
        "KDC_ERR_PREAUTH_REQUIRED(25)" ?  "required" : "not required";

    printf(" - Preauthentication is %s for this KDC.\n", self->preauth);
}

kerberos$target:::krb_error-read
/ self->preauth == "required" /
{
    printf(" - This KDC supports the following preauth types: %s.",
        args[1]->kerror_e_data);
}
```

그런 다음 preauth.d 소스 파일을 컴파일하여 답을 얻으십시오.

```
# dtrace -qs kerberos_preauth.d -c "kinit -k"
 - Preauthentication is required for this KDC.
 - This KDC supports the following preauth types: ENC_TIMESTAMP(2)
FX_FAST(136) PK_ETYPE_INFO2(19) SAM_RESPONSE(13) FX_COOKIE(133).
```

다양한 사전 인증 유형에 대한 자세한 내용은 RFC 4120을 참조하십시오.

◆ ◆ ◆ 23 장

23 Kerberos 주체 및 정책 관리(작업)

이 장은 주체 및 주체와 관련된 정책을 관리하는 절차를 제공합니다. 또한 호스트의 keytab 파일을 관리하는 방법에 대해서도 설명합니다.

이 장은 주체와 정책을 관리해야 하는 사용자가 이용해야 합니다. 이 장을 이용하기 전에 계획 고려 사항을 비롯하여 주체와 정책에 대해 잘 알고 있어야 합니다. 19 장, "Kerberos 서비스 소개" 및 20 장, "Kerberos 서비스 계획"을 각각 참조하십시오.

다음은 이 장에서 다루는 정보를 나열한 것입니다.

Kerberos 주체 및 정책을 관리하는 방법

마스터 KDC의 Kerberos 데이터베이스에는 영역의 모든 Kerberos 주체, 해당 암호, 정책 및 기타 관리 정보가 포함되어 있습니다. 주체를 생성 및 삭제하고 해당 속성을 수정하려면 `kadmin` 또는 `gkadmin` 명령을 사용할 수 있습니다.

`kadmin` 명령은 Kerberos 주체, 정책 및 keytab 파일을 유지 관리할 수 있는 명령줄 인터페이스를 제공합니다. 다음 두 가지 버전의 `kadmin` 명령이 있습니다.

- `kadmin` – 네트워크의 어떤 위치에서도 안전하게 작동하도록 Kerberos 인증을 사용합니다.
- `kadmin.local` – 마스터 KDC에서 직접 실행해야 합니다.

Kerberos를 사용하여 사용자를 인증하는 `kadmin` 이외에는 두 버전의 기능이 동일합니다. 원격 버전을 사용할 수 있도록 충분한 데이터베이스를 설정할 수 있으려면 로컬 버전이 필요합니다.

또한 Oracle Solaris 릴리스는 대화식 GUI(사용자 인터페이스)인 SEAM 도구 gkadmin을 제공하는데, 이 도구도 kadmin 명령과 동일한 기능을 제공합니다. 자세한 내용은 448 페이지 “SEAM 도구”를 참조하십시오.

SEAM 도구

SEAM 도구(gkadmin)는 Kerberos 주체와 정책을 유지 관리할 수 있는 대화식 GUI(그래픽 사용자 인터페이스)입니다. 이 도구는 kadmin 명령과 동일한 기능을 제공합니다. 그러나 keytab 파일의 관리는 지원하지 않습니다. keytab 파일은 kadmin 명령을 사용하여 관리해야 합니다. 이에 대해서는 477 페이지 “Keytab 파일 관리”에 설명되어 있습니다.

kadmin 명령과 마찬가지로, SEAM 도구는 네트워크의 어떤 위치에서도 안전하게 작동하도록 Kerberos 인증 및 암호화된 RPC를 사용합니다. SEAM 도구를 사용하여 수행할 수 있는 작업은 다음과 같습니다.

- 기본값 또는 기존 주체를 기반으로 하는 주체 새로 만들기
- 기존 정책을 기반으로 하는 정책 새로 만들기
- 주체에 대한 설명 추가
- 새 주체를 만들기 위한 기본값 설정
- 도구를 종료하지 않고 다른 주체로 로그인
- 주체 목록 및 정책 목록 인쇄 또는 저장
- 주체 목록 및 정책 목록 보기 및 검색

SEAM 도구는 또한 상황에 맞는 도움말과 일반 온라인 도움말도 제공합니다.

다음 작업 맵은 SEAM 도구로 수행할 수 있는 작업을 나타냅니다.

- 452 페이지 “Kerberos 주체 관리(작업 맵)”
- 465 페이지 “Kerberos 정책 관리(작업 맵)”

또한 SEAM 도구에서 지정하거나 볼 수 있는 모든 주체 속성과 정책 속성에 대한 설명을 보려면 473 페이지 “SEAM 도구 패널 설명”으로 이동하십시오.

SEAM 도구에 해당하는 명령줄 명령

이 절에서는 SEAM 도구와 동일한 기능을 제공하는 kadmin 명령을 보여줍니다. 이러한 명령은 X Window 시스템을 실행하지 않아도 사용할 수 있습니다. 이 장에 설명된 대부분의 절차는 SEAM 도구를 사용하지만, 해당하는 명령줄 명령을 사용하는 예제를 제공하는 절차도 많이 있습니다.

표 23–1 SEAM 도구에 해당하는 명령줄 명령

SEAM 도구 절차	해당하는 kadmin 명령
주체 목록 보기	list_principals 또는 get_principals

표 23–1 SEAM 도구에 해당하는 명령줄 명령 (계속)

SEAM 도구 절차	해당하는 kadmin 명령
주체 속성 보기	get_principal
새 주체 만들기	add_principal
주체 복제	해당하는 명령줄 명령 없음
주체 수정	modify_principal 또는 change_password
주체 삭제	delete_principal
새 주체를 만들기 위한 기본값 설정	해당하는 명령줄 명령 없음
정책 목록 보기	list_policies 또는 get_policies
정책 속성 보기	get_policy
새 정책 만들기	add_policy
정책 복제	해당하는 명령줄 명령 없음
정책 수정	modify_policy
정책 삭제	delete_policy

SEAM 도구로만 수정되는 파일

SEAM 도구로만 수정되는 파일은 $HOME/.gkadmin 파일입니다. 이 파일에는 새 주체를 만들기 위한 기본값이 포함되어 있습니다. 이 파일은 Edit(편집) 메뉴에서 Properties(등록 정보)를 선택하여 업데이트할 수 있습니다.

SEAM 도구의 인쇄 및 온라인 도움말 기능

SEAM 도구는 인쇄 기능과 온라인 도움말 기능을 제공합니다. Print(인쇄) 메뉴를 통해 프린터나 파일로 전송할 수 있는 항목은 다음과 같습니다.

- 지정된 KDC에 있는 사용 가능한 주체 목록
- 지정된 KDC에 있는 사용 가능한 정책 목록
- 현재 선택된 주체 또는 로드된 주체
- 현재 선택된 정책 또는 로드된 정책

상황에 맞는 도움말과 일반 도움말은 Help(도움말) 메뉴에서 액세스할 수 있습니다. Help(도움말) 메뉴에서 상황에 맞는 도움말을 선택하면 상황에 맞는 도움말 창이 표시되고 도구가 도움말 모드로 전환됩니다. 도움말 모드에서 창에 있는 필드, 레이블 또는 버튼을 누르면 해당 항목에 대한 도움말이 도움말 창에 표시됩니다. 도구의 일반 모드로 다시 전환하려면 Help(도움말) 창에서 Dismiss(없애기)를 누르십시오.

Help Contents(도움말 목차)를 선택할 수도 있습니다. 그러면 이 장에 제공되는 일반 개요 및 작업 정보에 대한 포인터를 제공하는 HTML 브라우저가 열립니다.

SEAM 도구에서 대형 목록 처리

사이트에 많은 수의 주체와 정책이 누적되기 시작하면 SEAM 도구가 주체와 정책 목록을 로드하고 표시하는 데 걸리는 시간이 점점 길어집니다. 따라서 이 도구를 사용 시 생산성이 저하됩니다. 이 문제를 해결하는 방법에는 여러 가지가 있습니다.

먼저 SEAM 도구가 목록을 로드하지 않도록 하여 목록을 로드하는 데 걸리는 시간을 완전히 제거할 수 있습니다. 이 옵션은 Edit(편집) 메뉴에서 Properties(등록 정보)를 선택한 다음 Show Lists(목록 표시)의 선택을 취소하여 설정할 수 있습니다. 도구가 목록을 로드하지 않을 경우 목록이 표시되지 않으므로 더 이상 목록 패널을 사용하여 주체나 정책을 선택할 수 없습니다. 대신, 제공된 새 Name(이름) 필드에 주체 또는 정책 이름을 입력한 다음 수행할 작업을 선택해야 합니다. 실제로 이름을 입력하는 것은 목록에서 항목을 선택하는 것과 같습니다.

대형 목록을 처리하는 다른 방법은 이 목록을 캐싱하는 것입니다. 실제로 제한된 시간 동안 목록을 캐싱하는 것은 SEAM 도구의 기본 동작입니다. 이 경우에도 처음에는 SEAM 도구가 목록을 캐시로 로드해야 합니다. 그러나 그 이후부터는 목록을 다시 검색하는 대신 캐시를 사용할 수 있습니다. 이 옵션을 사용하면 계속해서 서버에서 목록을 로드해야 할 필요가 없으므로 오랜 시간이 걸리지 않습니다.

목록 캐싱은 Edit(편집) 메뉴에서 Properties(등록 정보)를 선택하여 설정할 수 있습니다. 캐시 설정에는 두 가지가 있습니다. 목록을 영구적으로 캐싱하거나, 도구가 목록을 서버에서 캐시로 재로드해야 하는 기간을 지정할 수 있습니다.

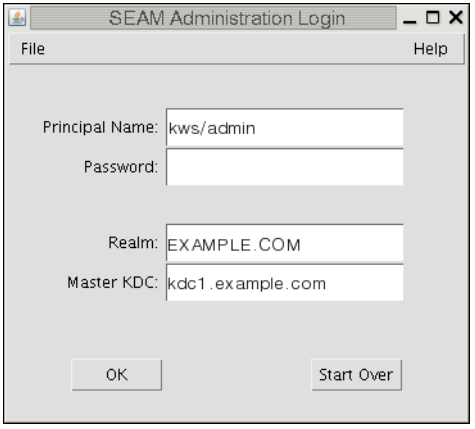
목록을 캐싱할 경우에도 목록 패널에서 주체와 정책을 선택할 수 있으므로, 첫번째 옵션과 마찬가지로 SEAM 도구를 사용하는 방식에 영향을 주지 않습니다. 또한 캐싱할 경우 다른 사용자의 변경 사항을 볼 수는 없지만, 본인이 변경한 내용이 있을 경우 서버와 캐시 모두에서 목록이 업데이트되기 때문에 본인의 변경 사항을 기준으로 최신 목록 정보를 볼 수 있습니다. 다른 변경 사항을 표시하고 목록의 최신 복사본을 가져오도록 캐시를 업데이트하려는 경우, 서버에서 캐시를 새로 고치고 싶을 때마다 Refresh(새로 고침) 메뉴를 사용할 수 있습니다.

▼ SEAM 도구를 시작하는 방법

1 **gkadmin 명령을 사용하여 SEAM 도구를 시작합니다.**

```
$ /usr/sbin/gkadmin
```

SEAM Administration Login(SEAM 관리 로그인) 창이 표시됩니다.



2 **기본값을 사용하지 않으려면 새 기본값을 지정합니다.**

창에 자동으로 기본값이 입력됩니다. 기본 주체 이름은 USER 환경 변수의 현재 ID에 `/admin`을 추가(*username*`/admin`)하여 결정됩니다. 기본 Realm(영역) 및 Master KDC(마스터 KDC) 필드는 `/etc/krb5/krb5.conf` 파일에서 선택됩니다. 기본값을 검색하려면 Start Over(시작)를 누릅니다.

주 – 각 주체 이름이 수행할 수 있는 관리 작업은 Kerberos ACL 파일 `/etc/krb5/kadm5.acl`에 지정되어 있습니다. 제한된 권한에 대한 자세한 내용은 476 페이지 “제한된 Kerberos 관리 권한으로 SEAM 도구 사용”을 참조하십시오.

---

3 **지정된 주체 이름에 대한 암호를 입력합니다.**

4 **확인을 누릅니다.**

모든 주체를 표시하는 창이 표시됩니다.

Kerberos 주체 관리

이 절에서는 SEAM 도구로 주체를 관리하는 데 사용되는 단계별 지침을 제공합니다. 사용 가능한 해당 명령줄 명령의 예제도 제공합니다.

Kerberos 주체 관리(작업 맵)

작업	설명	수행 방법
주체 목록 보기	Principals(주체) 탭을 누르면 주체 목록이 표시됩니다.	453 페이지 “Kerberos 주체 목록을 보는 방법”
주체 속성 보기	Principal List(주체 목록)에서 Principal(주체)을 선택한 다음 Modify(수정) 버튼을 누르면 주체의 속성이 표시됩니다.	455 페이지 “Kerberos 주체의 속성을 보는 방법”
새 주체 만들기	Principal List(주체 목록) 패널에서 Create New(새로 만들기) 버튼을 누르면 새 주체를 만들 수 있습니다.	457 페이지 “새 Kerberos 주체를 만드는 방법”
주체 복제	Principal List(주체 목록)에서 복제할 주체를 선택한 다음 Duplicate(복제) 버튼을 누르면 주체를 복제할 수 있습니다.	460 페이지 “Kerberos 주체를 복제하는 방법”
주체 수정	Principal List(주체 목록)에서 수정할 주체를 선택한 다음 Modify(수정) 버튼을 누르면 주체를 수정할 수 있습니다. 주체 이름은 수정할 수 없습니다. 주체 이름을 바꾸려면 주체를 복제하고 새 이름을 지정한 다음 이를 저장하고 이전 주체를 삭제해야 합니다.	460 페이지 “Kerberos 주체를 수정하는 방법”
주체 삭제	Principal List(주체 목록)에서 삭제할 주체를 선택한 다음 Delete(삭제) 버튼을 누르면 주체가 삭제됩니다.	461 페이지 “Kerberos 주체를 삭제하는 방법”
새 주체를 만들기 위한 기본값 설정	Edit(편집) 메뉴에서 Properties(등록 정보)를 선택하면 새 주체를 만들기 위한 기본값을 설정할 수 있습니다.	462 페이지 “새 Kerberos 주체를 만들기 위한 기본값을 설정하는 방법”
Kerberos 관리 권한 수정(`kadm5.acl` 파일)	**명령줄에서만 수행됩니다.** Kerberos 관리 권한에 따라 주체가 Kerberos 데이터베이스에 대해 수행할 수 있는 작업(예: 추가 및 수정)이 결정됩니다. 주체별로 Kerberos 관리 권한을 수정하려면 `/etc/krb5/kadm5.acl` 파일을 편집해야 합니다.	463 페이지 “Kerberos 관리 권한을 수정하는 방법”

자동으로 새 Kerberos 주체 만들기

SEAM 도구는 사용하기 간편하지만, 이 도구로는 새 주체를 자동으로 만들 수 없습니다. 10개 또는 100개의 새 주체를 단시간에 추가해야 할 경우에 특히 자동화가 유용합니다. 이 경우 Bourne 셸 스크립트에서 `kadmin.local` 명령을 사용하면 됩니다.

다음 셸 스크립트 행은 새 주체를 자동으로 만드는 방법에 대한 예제입니다.

```
awk '{ print "ank +needchange -pw", $2, $1 }' < /tmp/princnames |
        time /usr/sbin/kadmin.local> /dev/null
```

이 예제는 쉽게 읽을 수 있도록 두 개의 행으로 분할되어 있습니다. 이 스크립트는 주체 이름과 해당 암호를 포함하는 `princnames` 파일을 읽은 다음 Kerberos 데이터베이스에 이 파일을 추가합니다. 하나 이상의 공백으로 구분된 각 행에 주체 이름과 해당 암호를 포함하는 `princnames` 파일을 만들어야 합니다. `+needchange` 옵션은 주체를 처음으로 사용하는 경우 로그인 중 새 암호를 입력하라는 프롬프트를 사용자에게 표시하도록 주체를 구성합니다. 이렇게 하면 `princnames` 파일의 암호에 대해 보안 위험이 발생하지 않습니다.

보다 정교한 스크립트를 작성할 수 있습니다. 예를 들어 스크립트가 이름 서비스의 정보를 사용하여 주체 이름에 대한 사용자 이름 목록을 얻을 수 있습니다. 사이트 요구 사항 및 스크립트 환경에 따라 수행할 수 있는 작업과 수행 방식이 결정됩니다.

▼ Kerberos 주체 목록을 보는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

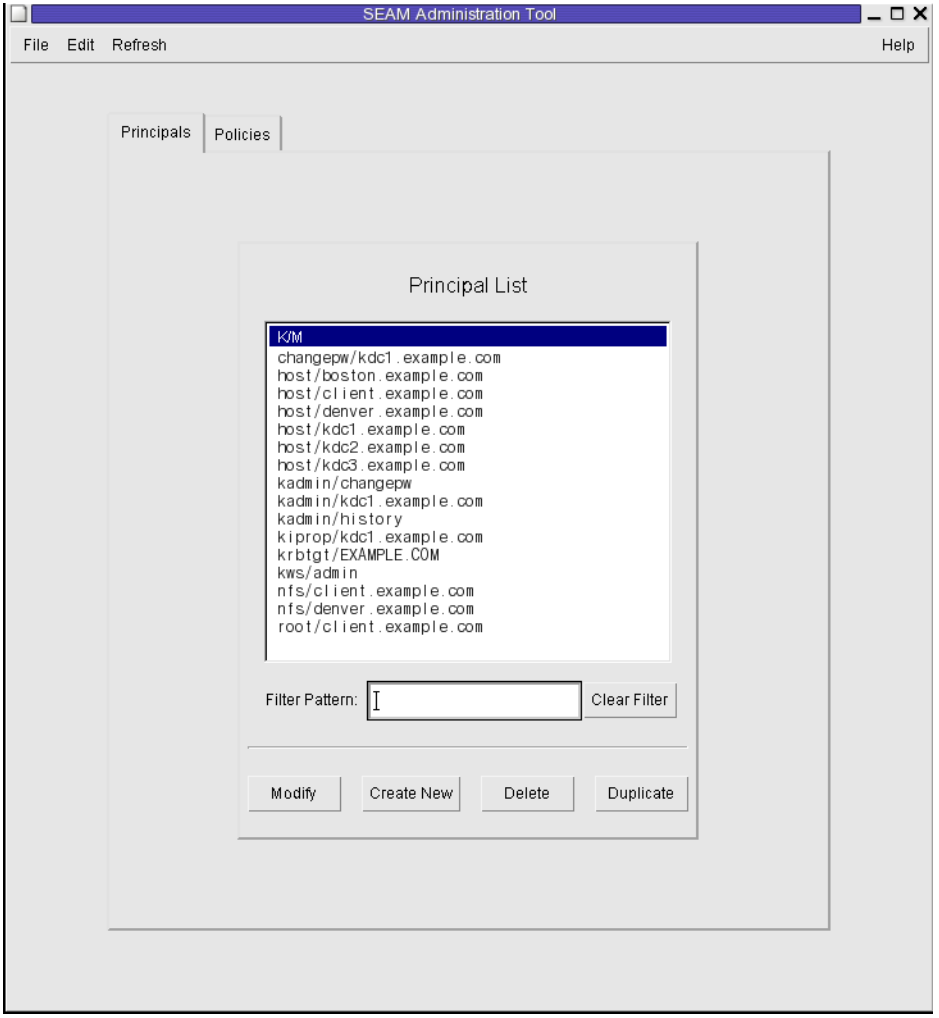
1 필요한 경우 SEAM 도구를 시작합니다.

자세한 내용은 451 페이지 “SEAM 도구를 시작하는 방법”을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Principals(주체) 탭을 누릅니다.**

주체 목록이 표시됩니다.



3 **특정 주체 또는 주체 하위 목록을 표시합니다.**

Filter(필터) 필드에 필터 문자열을 입력한 다음 Enter 키를 누릅니다. 필터가 성공하면 필터와 일치하는 주체 목록이 표시됩니다.

필터 문자열은 하나 이상의 문자로 구성되어야 합니다. 필터 방식은 대소문자를 구분하므로 필터에 대해 적합한 대소문자를 사용해야 합니다. 예를 들어 필터 문자열 `ge`를 입력할 경우, 필터 방식에 따라 `ge` 문자열을 포함하는 주체(예: `george` 또는 `edge`)만 표시됩니다.

전체 주체 목록을 표시하려면 Clear Filter(필터 지우기)를 누릅니다.

예 23–1 Kerberos 주체 목록 보기(명령줄)

다음 예에서 kadmin의 list_principals 명령은 kadmin*와 일치하는 모든 주체를 나열합니다. list_principals 명령과 함께 와일드카드를 사용할 수 있습니다.

```
kadmin: list_principals kadmin*
kadmin/changepw@EXAMPLE.COM
kadmin/kdc1.example.con@EXAMPLE.COM
kadmin/history@EXAMPLE.COM
kadmin: quit
```

▼ Kerberos 주체의 속성을 보는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

1. **필요한 경우 SEAM 도구를 시작합니다.**
   자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

   ```
   $ /usr/sbin/gkadmin
   ```

2. **Principals(주체) 탭을 누릅니다.**

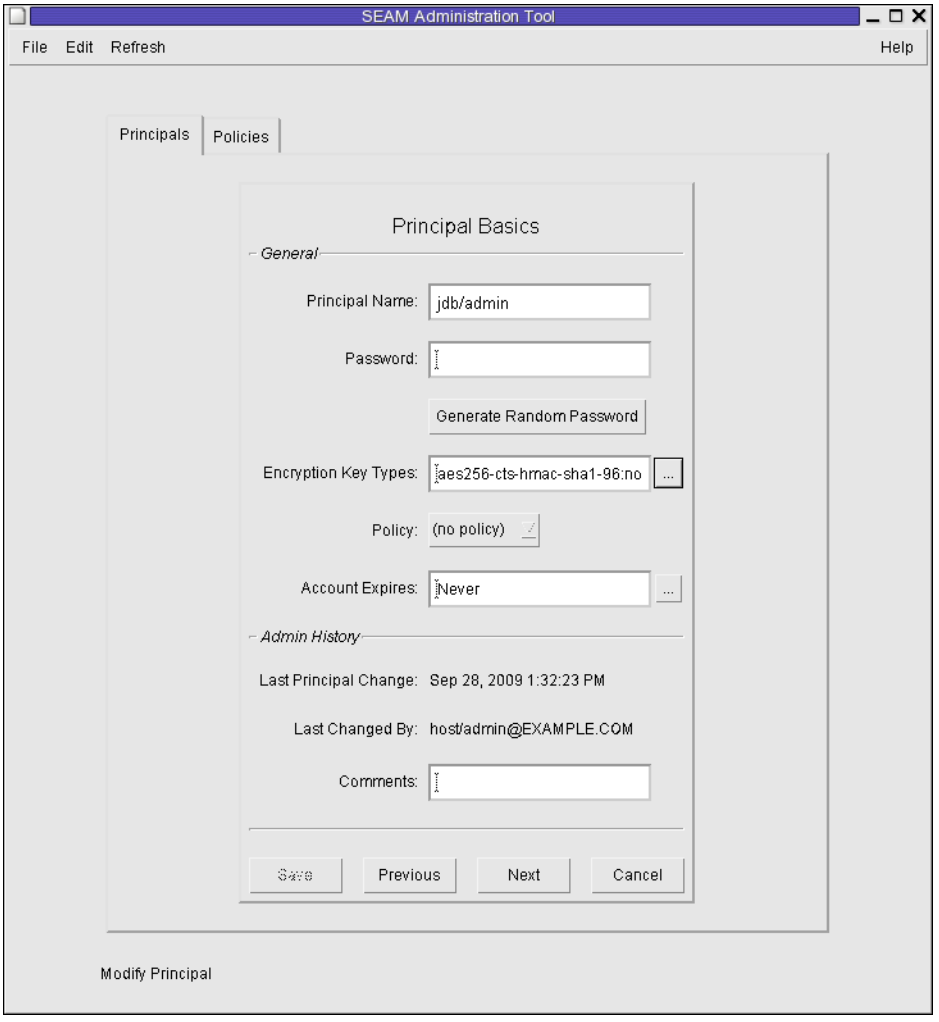
3. **목록에서 보려는 주체를 선택한 다음 Modify(수정)를 누릅니다.**
   몇 가지 주체 속성을 포함하는 Principal Basics(주체 기본 사항) 패널이 표시됩니다.

4. **주체 속성을 모두 보려면 계속 Next(다음)를 누릅니다.**
   세 개의 창에 속성 정보가 포함되어 있습니다. 각 창의 다양한 속성에 대한 정보를 보려면 Help(도움말) 메뉴에서 Context-Sensitive Help(상황에 맞는 도움말)를 선택합니다. 또는 모든 주체 속성 설명을 보려면 473 페이지 "SEAM 도구 패널 설명"으로 이동합니다.

5. **보기를 완료했으면 Cancel(취소)을 누릅니다.**

예 23–2 Kerberos 주체의 속성 보기

다음 예는 jdb/admin 주체를 보고 있는 경우의 첫번째 창을 보여줍니다.



예 23–3 Kerberos 주체의 속성 보기(명령줄)

다음 예에서 `kadmin`의 `get_principal` 명령은 `jdb/admin` 주체의 속성을 표시합니다.

```
kadmin: getprinc jdb/admin
Principal: jdb/admin@EXAMPLE.COM

Expiration date: [never]
Last password change: [never]

Password expiration date: Wed Apr 14 11:53:10 PDT 2011
Maximum ticket life: 1 day 16:00:00
Maximum renewable life: 1 day 16:00:00
Last modified: Mon Sep 28 13:32:23 PST 2009 (host/admin@EXAMPLE.COM)
```

```
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 1
Key: vno 1, AES-256 CTS mode with 96-bit SHA-1 HMAC, no salt
Key: vno 1, AES-128 CTS mode with 96-bit SHA-1 HMAC, no salt
Key: vno 1, Triple DES with HMAC/sha1, no salt
Key: vno 1, ArcFour with HMAC/md5, no salt
Key: vno 1, DES cbc mode with RSA-MD5, no salt
Attributes: REQUIRES_HW_AUTH
Policy: [none]
kadmin: quit
```

▼ 새 Kerberos 주체를 만드는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

1 **필요한 경우 SEAM 도구를 시작합니다.**

자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

주 – 새 정책이 필요한 주체를 새로 만드는 경우 먼저 새 정책을 만든 다음 새 주체를 만들어야 합니다. 469 페이지 "새 Kerberos 정책을 만드는 방법"으로 이동하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Principals(주체) 탭을 누릅니다.**

3 **New(새로 만들기)를 누릅니다.**

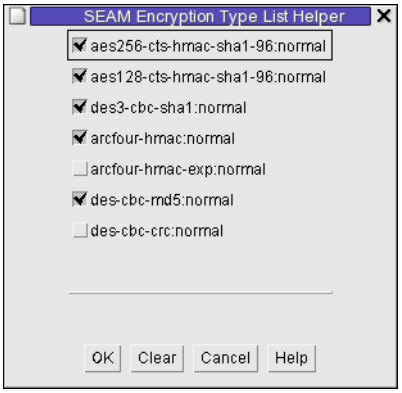
몇 가지 주체 속성을 포함하는 Principal Basics(주체 기본 사항) 패널이 표시됩니다.

4 **주체 이름과 암호를 지정합니다.**

주체 이름과 암호는 필수 항목입니다.

5 **주체의 암호화 유형을 지정합니다.**

암호화 키 유형 필드의 오른쪽에 있는 상자를 누르면 사용 가능한 모든 암호화 키 유형을 표시하는 새 창이 열립니다. 필요한 암호화 유형을 선택한 후 OK(확인)를 누릅니다.



6 **주체에 대한 정책을 지정합니다.**

7 **주체의 속성 값을 지정하고 Next(다음)를 눌러 추가 속성을 지정합니다.**

세 개의 창에 속성 정보가 포함되어 있습니다. 각 창의 다양한 속성에 대한 정보를 보려면 Help(도움말) 메뉴에서 Context-Sensitive Help(상황에 맞는 도움말)를 선택합니다. 또는 모든 주체 속성 설명을 보려면 473 페이지 "SEAM 도구 패널 설명"으로 이동합니다.

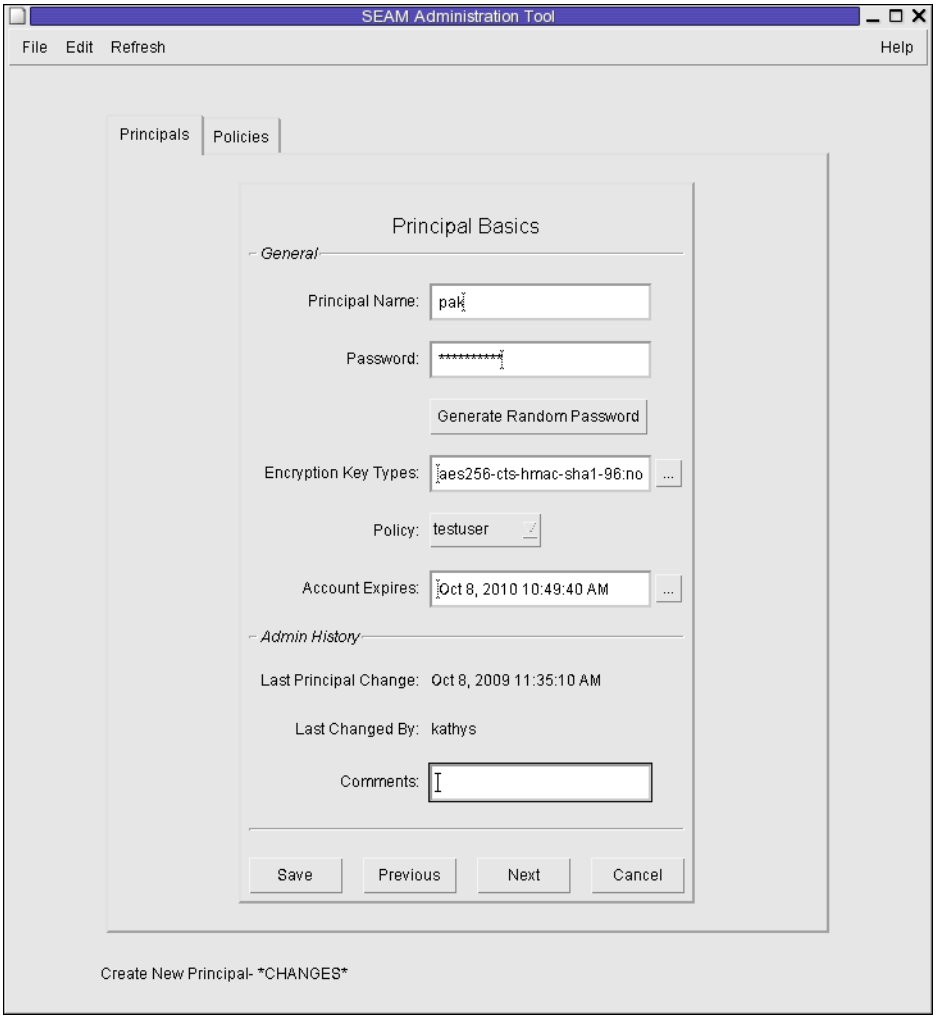
8 **Save(저장)를 눌러 주체를 저장하거나, 마지막 패널에서 Done(완료)을 누릅니다.**

9 **필요한 경우 `/etc/krb5/kadm5.acl` 파일에서 새 주체에 대한 Kerberos 관리 권한을 설정합니다.**

자세한 내용은 463 페이지 "Kerberos 관리 권한을 수정하는 방법"을 참조하십시오.

예 23–4 새 Kerberos 주체 만들기

다음 예는 `pak`이라는 새 주체를 만들 때 표시되는 Principal Basics(주체 기본 사항) 패널을 보여줍니다. 정책은 `testuser`로 설정되어 있습니다.



예 23–5 새 Kerberos 주체 만들기(명령줄)

다음 예에서 `kadmin`의 `add_principal` 명령은 `pak`이라는 새 주체를 만듭니다. 주체의 정책은 `testuser`로 설정됩니다.

```
kadmin: add_principal -policy testuser pak
Enter password for principal "pak@EXAMPLE.COM": <Type the password>
Re-enter password for principal "pak@EXAMPLE.COM": <Type the password again>
Principal "pak@EXAMPLE.COM" created.
kadmin: quit
```

▼ Kerberos 주체를 복제하는 방법

이 절차는 기존 주체 속성의 일부 또는 전체를 사용하여 새 주체를 만드는 방법에 대해 설명합니다. 이 절차의 경우 해당하는 명령줄 명령은 없습니다.

1 **필요한 경우 SEAM 도구를 시작합니다.**

자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Principals(주체) 탭을 누릅니다.**

3 **목록에서 복제하려는 주체를 선택한 다음 Duplicate(복제)를 누릅니다.**

Principal Basics(주체 기본 사항) 패널이 표시됩니다. 선택한 주체의 모든 속성이 복제됩니다. 단, Principal Name(주체 이름) 및 Password(암호) 필드는 복제되지 않고 비어 있습니다.

4 **주체 이름과 암호를 지정합니다.**

주체 이름과 암호는 필수 항목입니다. 선택한 주체의 정확한 복제본을 만들려면 Save(저장)를 누르고 단계 7로 이동합니다.

5 **주체의 속성에 대해 다른 값을 지정하고 Next(다음)를 눌러 추가 속성을 지정합니다.**

세 개의 창에 속성 정보가 포함되어 있습니다. 각 창의 다양한 속성에 대한 정보를 보려면 Help(도움말) 메뉴에서 Context-Sensitive Help(상황에 맞는 도움말)를 선택합니다. 또는 모든 주체 속성 설명을 보려면 473 페이지 "SEAM 도구 패널 설명"으로 이동합니다.

6 **Save(저장)를 눌러 주체를 저장하거나, 마지막 패널에서 Done(완료)을 누릅니다.**

7 **필요한 경우 `/etc/krb5/kadm5.acl` 파일에서 주체에 대한 Kerberos 관리 권한을 설정합니다.**

자세한 내용은 463 페이지 "Kerberos 관리 권한을 수정하는 방법"을 참조하십시오.

▼ Kerberos 주체를 수정하는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

1 **필요한 경우 SEAM 도구를 시작합니다.**

자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Principals(주체) 탭을 누릅니다.**

3 **목록에서 수정하려는 주체를 선택한 다음 Modify(수정)를 누릅니다.**
떫 가지 주체 속성을 포함하는 Principal Basics(주체 기본 사항) 패널이 표시됩니다.

4 **주체의 속성을 수정하고 Next(다음)를 눌러 추가 속성을 지정합니다.**
세 개의 창에 속성 정보가 포함되어 있습니다. 각 창의 다양한 속성에 대한 정보를 보려면 Help(도움말) 메뉴에서 Context-Sensitive Help(상황에 맞는 도움말)를 선택합니다. 또는 모든 주체 속성 설명을 보려면 473 페이지 "SEAM 도구 패널 설명"으로 이동합니다.

주 – 주체 이름은 수정할 수 없습니다. 주체 이름을 바꾸려면 주체를 복제하고 새 이름을 지정한 다음 이를 저장하고 이전 주체를 삭제해야 합니다.

---

5 **Save(저장)를 눌러 주체를 저장하거나, 마지막 패널에서 Done(완료)을 누릅니다.**

6 **`/etc/krb5/kadm5.acl` 파일에서 주체에 대한 Kerberos 관리 권한을 수정합니다.**
자세한 내용은 463 페이지 "Kerberos 관리 권한을 수정하는 방법"을 참조하십시오.

예 23–6 Kerberos 주체의 암호 수정(명령줄)

다음 예에서 `kadmin`의 `change_password` 명령은 `jdb` 주체의 암호를 수정합니다. `change_password` 명령을 사용하면 암호를 주체의 암호 내역에 있는 암호로 변경할 수 없습니다.

```
kadmin: change_password jdb
Enter password for principal "jdb": <Type the new password>
Re-enter password for principal "jdb": <Type the password again>
Password for "jdb@EXAMPLE.COM" changed.
kadmin: quit
```

주체의 다른 속성을 수정하려면 `kadmin`의 `modify_principal` 명령을 사용해야 합니다.

▼ Kerberos 주체를 삭제하는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

1 **필요한 경우 SEAM 도구를 시작합니다.**
자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Principals(주체) 탭을 누릅니다.**

3 **목록에서 삭제하려는 주체를 선택한 다음 Delete(삭제)를 누릅니다.**
삭제를 확인하면 주체가 삭제됩니다.

4 **Kerberos ACL(액세스 제어 목록) 파일 /etc/krb5/kadm5.acl에서 주체를 제거합니다.**
자세한 내용은 463 페이지 "Kerberos 관리 권한을 수정하는 방법"을 참조하십시오.

예 23–7 Kerberos 주체 삭제(명령줄)

다음 예에서 kadmin의 delete_principal 명령은 jdb 주체를 삭제합니다.

```
kadmin: delete_principal pak
Are you sure you want to delete the principal "pak@EXAMPLE.COM"? (yes/no): yes
Principal "pak@EXAMPLE.COM" deleted.
Make sure that you have removed this principal from all ACLs before reusing.
kadmin: quit
```

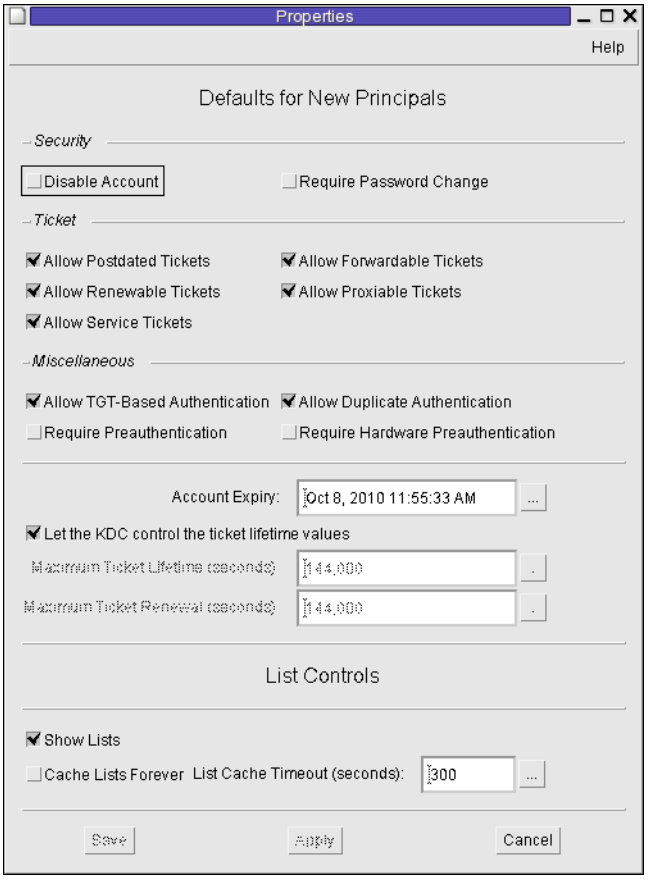
▼ 새 Kerberos 주체를 만들기 위한 기본값을 설정하는 방법

이 절차의 경우 해당하는 명령줄 명령은 없습니다.

1 **필요한 경우 SEAM 도구를 시작합니다.**
자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Edit(편집) 메뉴에서 Properties(등록 정보)를 선택합니다.**
Properties(등록 정보) 창이 표시됩니다.



3 **새 주체를 만들 때 사용할 기본값을 선택합니다.**
각 창의 다양한 속성에 대한 정보를 보려면 Help(도움말) 메뉴에서 Context-Sensitive Help(상황에 맞는 도움말)를 선택합니다.

4 **저장을 누릅니다.**

▼ Kerberos 관리 권한을 수정하는 방법

사이트에 사용자 주체는 많이 있지만, 소수의 사용자만 Kerberos 데이터베이스를 관리할 수 있습니다. Kerberos 데이터베이스 관리 권한은 Kerberos ACL(액세스 제어 목록) 파일 `kadm5.acl`에 따라 결정됩니다. `kadm5.acl` 파일을 사용하여 개인 주체에 대한 권한을

허용하거나 허용하지 않을 수 있습니다. 또는 주체 이름에 '*' 와일드카드를 사용하여 주체 그룹에 대한 권한을 지정할 수 있습니다.

1 마스터 KDC에서 수퍼유저로 로그인합니다.

2 `/etc/krb5/kadm5.acl` 파일을 편집합니다.

`kadm5.acl` 파일의 항목 형식은 다음과 같습니다.

principal privileges [*principal-target*]

principal 권한이 부여될 주체를 지정합니다. 주체 이름의 일부에 '*' 와일드카드가 포함될 수 있는데, 이는 주체 그룹에 대해 동일한 권한을 제공할 경우에 유용합니다. 예를 들어 `admin` 인스턴스를 포함하는 모든 주체를 지정하려는 경우, `*/admin@`*realm*을 사용하십시오.

`admin` 인스턴스는 일반적으로 Kerberos 주체별로 별도의 권한(예: Kerberos 데이터베이스에 대한 관리 액세스 권한)을 부여하는 데 사용됩니다. 예를 들어 사용자 `jdb`에게는 관리용 주체인 `jdb/admin`이 있을 수 있습니다. 즉, 사용자 `jdb`는 실제로 해당 권한을 사용해야 하는 경우에만 `jdb/admin` 티켓을 얻게 됩니다.

privileges 주체가 수행하거나 수행할 수 없는 작업을 지정합니다. 이 필드는 다음 문자 또는 이러한 문자의 대문자 목록으로 된 문자열로 구성됩니다. 문자가 대문자이거나 지정되지 않은 경우 작업이 허용되지 않습니다. 소문자일 경우 작업이 허용됩니다.

`a`	주체 또는 정책의 추가를 허용하거나 허용하지 않습니다.
`d`	주체 또는 정책의 삭제를 허용하거나 허용하지 않습니다.
`m`	주체 또는 정책의 수정을 허용하거나 허용하지 않습니다.
`c`	주체 암호 변경을 허용하거나 허용하지 않습니다.
`i`	Kerberos 데이터베이스에 대한 조회를 허용하거나 허용하지 않습니다.
`l`	Kerberos 데이터베이스에서 주체 또는 정책의 나열을 허용하거나 허용하지 않습니다.
`x` 또는 `*`	모든 권한을 허용합니다(`admcil`).

principal-target 이 필드에 주체가 지정된 경우 *principal*이 *principal-target*에서 작동하는 경우에만 *privileges*가 *principal*에 적용됩니다. 주체 이름의 일부에 '*' 와일드카드가 포함될 수 있는데, 이는 주체를 그룹화할 경우에 유용합니다.

예 23–8 Kerberos 관리 권한 수정

`kadm5.acl` 파일의 다음 항목은 `admin` 인스턴스를 포함하는 `EXAMPLE.COM` 영역의 주체에 Kerberos 데이터베이스에 대한 모든 권한을 부여합니다.

```
*/admin@EXAMPLE.COM *
```

kadm5.acl 파일의 다음 항목은 root 인스턴스를 포함하는 주체를 추가 및 나열하고 이 주체에 대해 조회할 수 있는 권한을 jdb@EXAMPLE.COM 주체에 부여합니다.

```
jdb@EXAMPLE.COM ali */root@EXAMPLE.COM
```

Kerberos 정책 관리

이 절에서는 SEAM 도구로 정책을 관리하는 데 사용되는 단계별 지침을 제공합니다. 사용 가능한 해당 명령줄 명령의 예제도 제공합니다.

Kerberos 정책 관리(작업 맵)

작업	설명	수행 방법
정책 목록 보기	Policies(정책) 탭을 누르면 정책 목록이 표시됩니다.	466 페이지 “Kerberos 정책 목록을 보는 방법”
정책 속성 보기	Policy List(정책 목록)에서 정책을 선택한 다음 Modify(수정) 버튼을 누르면 정책의 속성이 표시됩니다.	467 페이지 “Kerberos 정책의 속성을 보는 방법”
새 정책 만들기	Policy List(정책 목록) 패널에서 Create New(새로 만들기) 버튼을 누르면 새 정책을 만들 수 있습니다.	469 페이지 “새 Kerberos 정책을 만드는 방법”
정책 복제	Policy List(정책 목록)에서 복제할 정책을 선택한 다음 Duplicate(복제) 버튼을 누르면 정책을 복제할 수 있습니다.	471 페이지 “Kerberos 정책을 복제하는 방법”
정책 수정	Policy List(정책 목록)에서 수정할 정책을 선택한 다음 Modify(수정) 버튼을 누르면 정책을 수정할 수 있습니다. 정책 이름은 수정할 수 없습니다. 정책 이름을 바꾸려면 정책을 복제하고 새 이름을 지정한 다음 이를 저장하고 이전 정책을 삭제해야 합니다.	471 페이지 “Kerberos 정책을 수정하는 방법”
정책 삭제	Policy List(정책 목록)에서 삭제할 정책을 선택한 다음 Delete(삭제) 버튼을 누르면 정책이 삭제됩니다.	472 페이지 “Kerberos 정책을 삭제하는 방법”

▼ Kerberos 정책 목록을 보는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

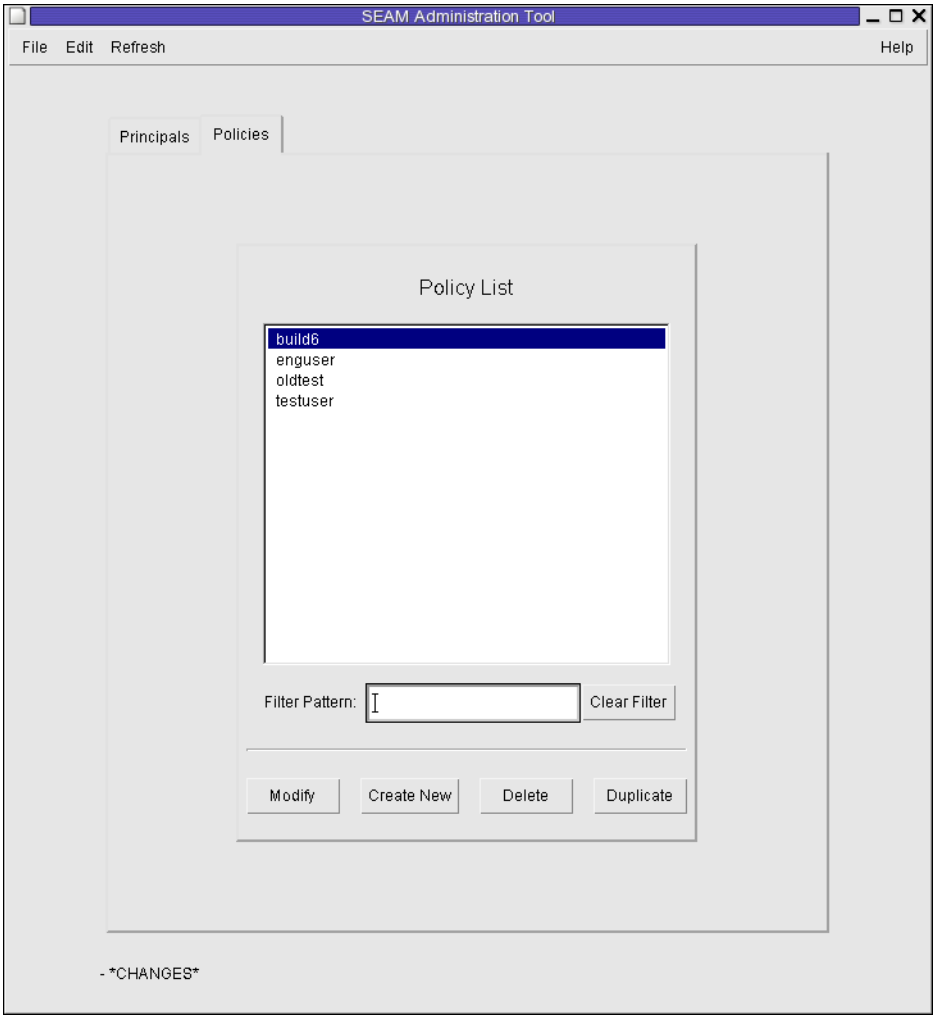
1 **필요한 경우 SEAM 도구를 시작합니다.**

자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Policies(정책) 탭을 누릅니다.**

정책 목록이 표시됩니다.



3 **특정 정책 또는 정책 하위 목록을 표시합니다.**

Filter(필터) 필드에 필터 문자열을 입력한 다음 Enter 키를 누릅니다. 필터가 성공하면 필터와 일치하는 정책 목록이 표시됩니다.

필터 문자열은 하나 이상의 문자로 구성되어야 합니다. 필터 방식은 대소문자를 구분하므로 필터에 대해 적합한 대소문자를 사용해야 합니다. 예를 들어 필터 문자열 ge를 입력할 경우, 필터 방식에 따라 ge 문자열을 포함하는 정책(예: george 또는 edge)만 표시됩니다.

전체 정책 목록을 표시하려면 Clear Filter(필터 지우기)를 누릅니다.

예 23–9 Kerberos 정책 목록 보기(명령줄)

다음 예에서 kadmin의 list_policies 명령은 *user*와 일치하는 모든 정책을 나열합니다. list_policies 명령과 함께 와일드카드를 사용할 수 있습니다.

```
kadmin: list_policies *user*
testuser
enguser
kadmin: quit
```

▼ Kerberos 정책의 속성을 보는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

1 **필요한 경우 SEAM 도구를 시작합니다.**

자세한 내용은 451 페이지 “SEAM 도구를 시작하는 방법”을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Policies(정책) 탭을 누릅니다.**

3 **목록에서 보려는 정책을 선택한 다음 Modify(수정)를 누릅니다.**

Policy Details(정책 세부 정보) 패널이 표시됩니다.

4 **보기를 완료했으면 Cancel(취소)을 누릅니다.**

예 23–10 Kerberos 정책의 속성 보기

다음 예는 test 정책을 보고 있는 경우의 Policy Details(정책 세부 정보) 패널을 보여줍니다.

예 23–11 Kerberos 정책의 속성 보기(명령줄)

다음 예에서 kadmin의 get_policy 명령은 enguser 정책의 속성을 표시합니다.

```
kadmin: get_policy enguser
Policy: enguser
Maximum password life: 2592000
Minimum password life: 0
Minimum password length: 8
Minimum number of password character classes: 2
Number of old keys kept: 3
Reference count: 0
kadmin: quit
```

참조 수는 이 정책을 사용하는 주체 수를 나타냅니다.

▼ 새 Kerberos 정책을 만드는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

1. **필요한 경우 SEAM 도구를 시작합니다.**
   자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

   ```
   $ /usr/sbin/gkadmin
   ```

2. **Policies(정책) 탭을 누릅니다.**

3. **New(새로 만들기)를 누릅니다.**
   Policy Details(정책 세부 정보) 패널이 표시됩니다.

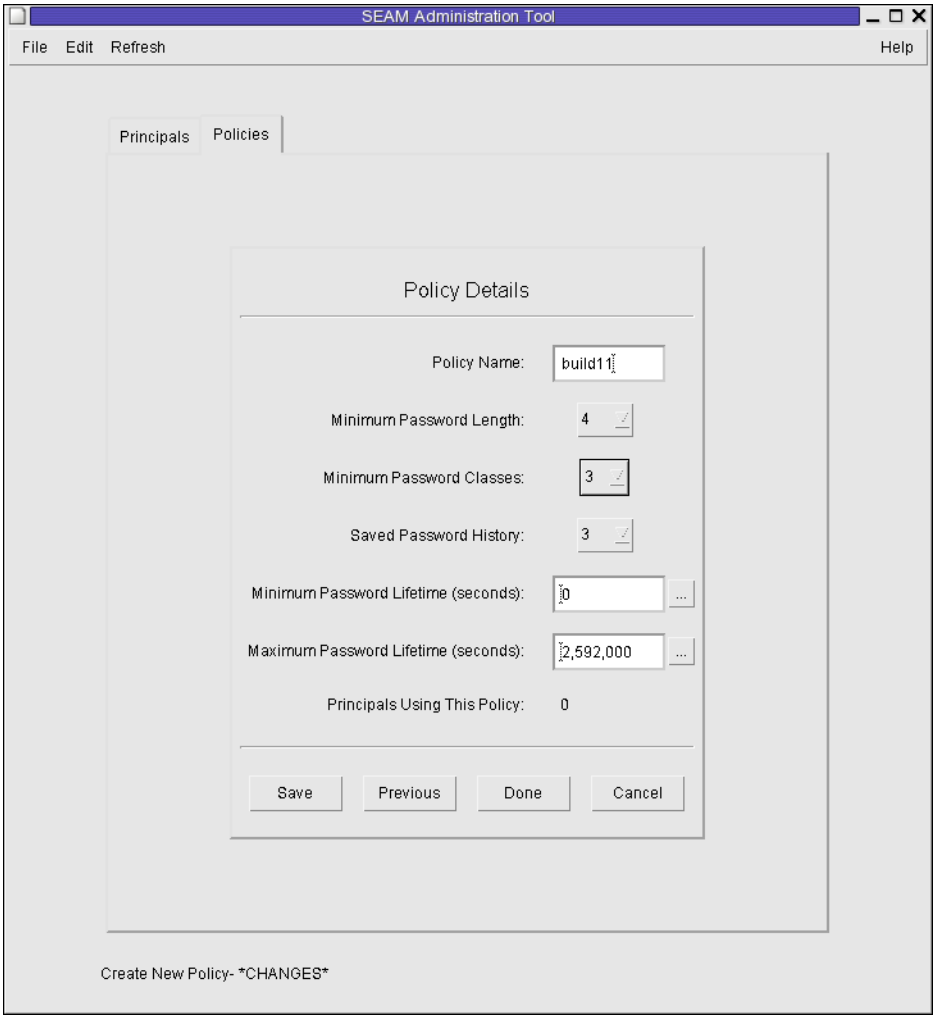
4. **Policy Name(정책 이름) 필드에 정책 이름을 지정합니다.**
   정책 이름은 필수 항목입니다.

5. **정책 속성의 값을 지정합니다.**
   이 창의 다양한 속성에 대한 정보를 보려면 Help(도움말) 메뉴에서 Context-Sensitive Help(상황에 맞는 도움말)를 선택합니다. 또는 모든 정책 속성 설명을 보려면 표 23–5로 이동합니다.

6. **Save(저장)를 눌러 정책을 저장하거나 Done(완료)을 누릅니다.**

예 23–12 새 Kerberos 정책 만들기

다음 예에서는 `build11`이라는 새 정책이 생성되었습니다. Minimum Password Classes(최소 암호 클래스)는 3으로 설정되어 있습니다.



예 23–13 새 Kerberos 정책 만들기(명령줄)

다음 예에서 `kadmin`의 `add_policy` 명령은 `build11` 정책을 만듭니다. 이 정책을 사용하려면 암호에 적어도 세 개의 문자 클래스가 필요합니다.

```
$ kadmin
kadmin: add_policy -minclasses 3 build11
kadmin: quit
```

▼ Kerberos 정책을 복제하는 방법

이 절차는 기존 정책 속성의 일부 또는 전체를 사용하여 새 정책을 만드는 방법에 대해 설명합니다. 이 절차의 경우 해당하는 명령줄 명령은 없습니다.

1 **필요한 경우 SEAM 도구를 시작합니다.**

자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Policies(정책) 탭을 누릅니다.**

3 **목록에서 복제하려는 정책을 선택한 다음 Duplicate(복제)를 누릅니다.**

Policy Details(정책 세부 정보) 패널이 표시됩니다. 선택한 정책의 모든 속성이 복제됩니다. 단, Policy Name(정책 이름) 필드는 복제되지 않고 비어 있습니다.

4 **Policy Name(정책 이름) 필드에 복제된 정책의 이름을 지정합니다.**

정책 이름은 필수 항목입니다. 선택한 정책의 정확한 복제본을 만들려면 단계 6으로 이동합니다.

5 **정책 속성에 대해 다른 값을 지정합니다.**

이 창의 다양한 속성에 대한 정보를 보려면 Help(도움말) 메뉴에서 Context-Sensitive Help(상황에 맞는 도움말)를 선택합니다. 또는 모든 정책 속성 설명을 보려면 표 23–5로 이동합니다.

6 **Save(저장)를 눌러 정책을 저장하거나 Done(완료)을 누릅니다.**

▼ Kerberos 정책을 수정하는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

1 **필요한 경우 SEAM 도구를 시작합니다.**

자세한 내용은 451 페이지 "SEAM 도구를 시작하는 방법"을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Policies(정책) 탭을 누릅니다.**

3 **목록에서 수정하려는 정책을 선택한 다음 Modify(수정)를 누릅니다.**

Policy Details(정책 세부 정보) 패널이 표시됩니다.

4 **정책 속성을 수정합니다.**

이 창의 다양한 속성에 대한 정보를 보려면 Help(도움말) 메뉴에서 Context-Sensitive Help(상황에 맞는 도움말)를 선택합니다. 또는 모든 정책 속성 설명을 보려면 표 23–5로 이동합니다.

주 – 정책 이름은 수정할 수 없습니다. 정책 이름을 바꾸려면 정책을 복제하고 새 이름을 지정한 다음 이를 저장하고 이전 정책을 삭제해야 합니다.

---

5 **Save(저장)를 눌러 정책을 저장하거나 Done(완료)을 누릅니다.**

예 23–14 Kerberos 정책 수정(명령줄)

다음 예에서 `kadmin`의 `modify_policy` 명령은 `build11` 정책에 대해 최소 암호 길이를 5자로 수정합니다.

```
$ kadmin
kadmin: modify_policy -minlength 5 build11
kadmin: quit
```

▼ Kerberos 정책을 삭제하는 방법

해당하는 명령줄 명령 예는 이 절차 뒤에 나옵니다.

주 – 정책을 삭제하기 전에 해당 정책을 현재 사용하고 있는 모든 주체에서 정책을 취소해야 합니다. 이를 위해서는 주체의 Policy(정책) 속성을 수정해야 합니다. 주체가 정책을 사용하고 있으면 삭제할 수 없습니다.

---

1 **필요한 경우 SEAM 도구를 시작합니다.**

자세한 내용은 451 페이지 “SEAM 도구를 시작하는 방법”을 참조하십시오.

```
$ /usr/sbin/gkadmin
```

2 **Policies(정책) 탭을 누릅니다.**

3 **목록에서 삭제하려는 정책을 선택한 다음 Delete(삭제)를 누릅니다.**

삭제를 확인하면 정책이 삭제됩니다.

예 23–15 Kerberos 정책 삭제(명령줄)

다음 예에서 `kadmin` 명령의 `delete_policy` 명령은 `build11` 정책을 삭제합니다.

```
kadmin: delete_policy build11
Are you sure you want to delete the policy "build11"? (yes/no): yes
kadmin: quit
```

정책을 삭제하기 전에 해당 정책을 현재 사용하고 있는 모든 주체에서 정책을 취소해야 합니다. 이를 위해서는 영향을 받는 주체에서 `kadmin`의 `modify_principal-policy` 명령을 사용해야 합니다. 주체가 정책을 사용하고 있으면 `delete_policy` 명령이 실패합니다.

SEAM 도구 참조

이 절에서는 SEAM 도구의 각 패널에 대해 설명합니다. 또한 제한된 권한으로 SEAM 도구를 사용하는 방법도 설명합니다.

SEAM 도구 패널 설명

이 절에서는 SEAM 도구에서 지정하거나 볼 수 있는 각 주체와 정책 속성에 대해 설명합니다. 속성은 해당 속성이 표시된 패널로 구성됩니다.

표 23–2 SEAM 도구의 Principal Basics(주체 기본 사항) 패널 속성

속성	설명
주체 이름	주체 이름입니다. 즉, 정규화된 주체 이름의 *primary*/ *instance* 부분입니다. 주체는 KDC가 티켓을 지정할 수 있는 고유 ID입니다. 주체를 수정하는 경우 주체 이름은 편집할 수 없습니다.
Password	주체에 대한 암호입니다. Generate Random Password(임의 암호 생성) 버튼을 사용하여 주체에 대한 임의 암호를 만들 수 있습니다.
Policy	주체에 사용 가능한 정책에 대한 메뉴입니다.
Account Expires	주체의 계정이 만료되는 날짜 및 시간입니다. 계정이 만료되면 주체는 더 이상 TGT(티켓 부여 티켓)를 가져오고 로그인할 수 없습니다.
Last Principal Change	주체에 대한 정보를 마지막으로 수정한 날짜입니다(읽기 전용).
Last Changed By	이 주체의 계정을 마지막으로 수정한 주체의 이름입니다(읽기 전용).
설명	주체와 관련된 설명입니다(예: "임시 계정").

표 23–3 SEAM 도구의 Principal Details(주체 세부 정보) 패널 속성

속성	설명
Last Success	주체가 마지막으로 로그인에 성공한 날짜 및 시간입니다(읽기 전용).
Last Failure	주체가 마지막으로 로그인에 실패한 날짜 및 시간입니다(읽기 전용).
Failure Count	주체에 대한 로그인 실패가 발생한 횟수입니다(읽기 전용).
Last Password Change	주체의 암호가 마지막으로 변경된 날짜 및 시간입니다(읽기 전용).
Password Expires	주체의 현재 암호가 만료되는 날짜 및 시간입니다.

표 23–3 SEAM 도구의 Principal Details(주체 세부 정보) 패널 속성 *(계속)*

속성	설명
Key Version	주체에 대한 키 버전 번호입니다. 이 속성은 일반적으로 암호가 손상된 경우에만 변경됩니다.
Maximum Lifetime (seconds)	주체에 대해 티켓을 부여할 수 있는 최대 기간입니다(갱신 없음).
Maximum Renewal (seconds)	주체에 대해 기존 티켓을 갱신할 수 있는 최대 기간입니다.

표 23–4 SEAM 도구의 Principal Flags(주체 플래그) 패널 속성

속성(라디오 버튼)	설명
Disable Account	선택할 경우 주체가 로그인할 수 없습니다. 이 속성을 사용하여 주체 계정을 일시적으로 고정할 수 있습니다.
Require Password Change	선택할 경우 주체의 현재 암호가 만료되므로, 사용자가 `kpasswd` 명령을 사용하여 새 암호를 만들어야 합니다. 이 속성은 보안 유출이 발생하여 이전 암호를 바꿔야 하는 경우에 유용합니다.
Allow Postdated Tickets	선택할 경우 주체가 후일자 티켓을 얻을 수 있습니다. 예를 들면 몇 시간 후에 실행해야 하는 `cron` 작업에 대해 후일자 티켓을 사용해야 하지만, 티켓 수명이 짧은 관계로 미리 티켓을 얻을 수 없는 경우입니다.
Allow Forwardable Tickets	선택할 경우 주체가 전달 가능 티켓을 얻을 수 있습니다. 전달 가능 티켓은 단일 사인 온(SSO) 세션을 제공하기 위해 원격 호스트로 전송되는 티켓입니다. 예를 들어 전달 가능 티켓을 사용하면서 `ftp` 또는 `rsh`를 통해 자신을 인증할 경우, 암호를 다시 입력하지 않고 NFS 서비스와 같은 다른 서비스를 사용할 수 있습니다.
Allow Renewable Tickets	선택할 경우 주체가 갱신 가능 티켓을 얻을 수 있습니다. 주체는 (첫번째 티켓이 만료된 후 새 티켓을 가져올 필요 없이) 갱신 가능한 티켓의 만료 날짜나 시간을 자동으로 확장할 수 있습니다. 현재는 NFS 서비스가 티켓을 갱신할 수 있는 티켓 서비스입니다.
Allow Renewable Tickets	선택할 경우 주체가 프록시 가능 티켓을 얻을 수 있습니다. 프록시 가능 티켓은 서비스가 클라이언트를 대신하여 클라이언트 작업을 수행하는 데 사용할 수 있는 티켓입니다. 프록시 가능 티켓을 사용하면 서비스가 클라이언트 ID를 사용하여 다른 서비스의 티켓을 얻을 수 있습니다. 그러나 TGT(티켓 부여 티켓)를 얻을 수는 없습니다.
Allow Service Tickets	선택할 경우 주체에 대한 서비스 티켓을 발행할 수 있습니다. `kadmin/`*hostname* 및 `changepw/` *hostname* 주체에 대한 서비스 티켓이 발행되도록 해서는 안됩니다. 이렇게 하면 해당 주체만 KDC 데이터베이스를 업데이트할 수 있습니다.
Allow TGT-Based Authentication	선택할 경우 서비스 주체가 다른 주체에 서비스를 제공할 수 있습니다. 더 구체적으로, 이 속성을 사용하면 KDC가 서비스 주체에 대해 서비스 티켓을 발행할 수 있습니다. 이 속성은 서비스 주체의 경우에만 유효합니다. 선택을 취소할 경우 서비스 주체에 대해 서비스 티켓을 발행할 수 없습니다.

표 23–4 SEAM 도구의 Principal Flags(주체 플래그) 패널 속성 *(계속)*

속성(라디오 버튼)	설명
Allow Duplicate Authentication	선택할 경우 사용자 주체가 다른 사용자 주체에 대한 서비스 티켓을 얻을 수 있습니다. 이 속성은 사용자 주체의 경우에만 유효합니다. 선택을 취소할 경우 사용자 주체가 서비스 주체에 대한 서비스 티켓을 얻을 수는 있지만, 다른 사용자 주체에 대한 서비스 티켓은 얻을 수 없습니다.
Required Preauthentication	선택할 경우 해당 주체가 실제로 TGT(티켓 부여 티켓)를 요청한 주체임이 (소프트웨어를 통해) 인증될 때까지 KDC가 요청된 TGT를 주체에게 전송하지 않습니다. 이 사전 인증은 보통 DES 카드와 같은 추가 암호를 통해 수행됩니다. 선택을 취소할 경우 KDC가 요청된 TGT를 주체에게 전송하기 전에 주체를 사전 인증할 필요가 없습니다.
Required Hardware Authentication	선택할 경우 해당 주체가 실제로 TGT(티켓 부여 티켓)를 요청한 주체임이 (하드웨어를 통해) 인증될 때까지 KDC가 요청된 TGT를 주체에게 전송하지 않습니다. 예를 들어 하드웨어 사전 인증은 Java Ring Reader에서 발생할 수 있습니다. 선택을 취소할 경우 KDC가 요청된 TGT를 주체에게 전송하기 전에 주체를 사전 인증할 필요가 없습니다.

표 23–5 SEAM 도구의 Policy Basics(정책 기본 사항) 창 속성

속성	설명
Policy Name	정책의 이름입니다. 정책은 주체의 암호와 티켓을 제어하는 규칙 세트입니다. 정책을 수정하는 경우 정책 이름은 편집할 수 없습니다.
Minimum Password Length	주체 암호의 최소 길이입니다.
Minimum Password Classes	주체의 암호에 필요한 최소한의 서로 다른 문자 유형 수입니다. 예를 들어 최소 클래스 값이 2인 경우, 암호의 서로 다른 문자 유형이 문자 및 숫자(hi2mom)와 같이 최소 두 개여야 합니다. 값이 3인 경우, 암호의 서로 다른 문자 유형이 문자, 숫자, 구두점(hi2mom!)과 같이 최소 세 개여야 합니다. 값이 그 이상인 경우에도 마찬가지입니다. 값이 1인 경우는 암호 문자 유형 수에 제한이 없습니다.
Saved Password History	주체가 이전에 사용한 암호 수 및 재사용할 수 없는 이전 암호 목록입니다.
Minimum Password Lifetime (seconds)	암호를 변경하기 전까지 사용해야 하는 최소 기간입니다.
Maximum Password Lifetime (seconds)	암호를 변경하기 전까지 사용할 수 있는 최대 기간입니다.
Principals Using This Policy	이 정책이 현재 적용된 주체 수입니다 (읽기 전용).

제한된 Kerberos 관리 권한으로 SEAM 도구 사용

admin 주체가 Kerberos 데이터베이스를 관리할 수 있는 모든 권한을 가지고 있는 경우 SEAM 도구의 모든 기능을 사용할 수 있습니다. 그러나 주체 목록을 보거나 주체의 암호를 변경할 수만 있는 등 제한된 권한을 가지고 있을 수도 있습니다. 제한된 Kerberos 관리 권한으로도 SEAM 도구를 사용할 수 있습니다. 그러나 SEAM 도구의 여러 부분이 사용자가 보유하고 있지 않은 Kerberos 관리 권한을 기준으로 변경됩니다. 표 23–6은 사용자의 Kerberos 관리 권한을 기준으로 SEAM 도구가 변경되는 방식을 보여줍니다.

나열 권한이 없을 경우 SEAM 도구가 시각적으로 가장 많이 변경됩니다. 나열 권한이 없을 경우 List(목록) 패널에 조작할 주체 및 정책 목록이 표시되지 않습니다. 대신, List(목록) 패널의 Name(이름) 필드를 사용하여 조작할 주체나 정책을 지정해야 합니다.

SEAM 도구에 로그인했는데 작업을 수행할 수 있는 충분한 권한이 없을 경우, 다음과 같은 메시지가 표시되고 다시 SEAM Administration Login(SEAM 관리 로그인) 창으로 돌아갑니다.

```
Insufficient privileges to use gkadmin: ADMCIL. Please try using another principal.
```

Kerberos 데이터베이스를 관리할 수 있도록 주체에 대한 권한을 변경하려면 463 페이지 “Kerberos 관리 권한을 수정하는 방법”으로 이동하십시오.

표 23–6 제한된 Kerberos 관리 권한으로 SEAM 도구 사용

허용되지 않는 권한	SEAM 도구에 미치는 영향
a(추가)	Principal List(주체 목록) 및 Policy List(정책 목록) 패널에서 Create New(새로 만들기) 및 Duplicate(복제) 버튼을 사용할 수 없습니다. 추가 권한이 없을 경우 주체나 정책을 새로 만들거나 이들을 복제할 수 없습니다.
d(삭제)	Principal List(주체 목록) 및 Policy List(정책 목록) 패널에서 Delete(삭제) 버튼을 사용할 수 없습니다. 삭제 권한이 없을 경우 주체나 정책을 삭제할 수 없습니다.
m(수정)	Principal List(주체 목록) 및 Policy List(정책 목록) 패널에서 Modify(수정) 버튼을 사용할 수 없습니다. 수정 권한이 없을 경우 주체나 정책을 수정할 수 없습니다. 또한 Modify(수정) 버튼을 사용할 수 없는 경우 암호 변경 권한이 있더라도 주체의 암호를 수정할 수 없습니다.
c(암호 변경)	Principal Basics(주체 기본 사항) 패널의 Password(암호) 필드가 읽기 전용이며 변경할 수 없습니다. 암호 변경 권한이 없을 경우 주체의 암호를 수정할 수 없습니다. 암호 변경 권한이 있더라도 수정 권한이 있어야 주체의 암호를 변경할 수 있습니다.

표 23–6 제한된 Kerberos 관리 권한으로 SEAM 도구 사용 *(계속)*

허용되지 않는 권한	SEAM 도구에 미치는 영향
i(데이터베이스 조회)	Principal List(주체 목록) 및 Policy List(정책 목록) 패널에서 Modify(수정) 및 Duplicate(복제) 버튼을 사용할 수 없습니다. 조회 권한이 없을 경우 주체나 정책을 수정하거나 복제할 수 없습니다. 또한 Modify(수정) 버튼을 사용할 수 없는 경우 암호 변경 권한이 있더라도 주체의 암호를 수정할 수 없습니다.
l(나열)	List(목록) 패널의 주체 및 정책 목록을 사용할 수 없습니다. 나열 권한이 없을 경우 List(목록) 패널의 Name(이름) 필드를 사용하여 조작할 주체나 정책을 지정해야 합니다.

Keytab 파일 관리

서비스를 제공하는 모든 호스트에는 *keytab*("키 테이블"의 줄임말)이라는 로컬 파일이 있습니다. keytab에는 **서비스 키**라고 하는 해당 서비스에 대한 주체가 포함되어 있습니다. 서비스 키는 서비스가 KDC에 대해 자신을 인증하는 데 사용되며, Kerberos 및 서비스 자체를 통해서만 알려집니다. 예를 들어 Kerberos화된 NFS 서버가 있는 경우, 이 서버에는 `nfs` 서비스 주체를 포함하는 keytab 파일이 있습니다.

keytab 파일에 서비스 키를 추가하려면 `kadmin`의 `ktadd` 명령을 사용하여 호스트의 keytab 파일에 해당 서비스 주체를 추가하십시오. 서비스 주체를 keytab 파일에 추가하는 것이기 때문에 주체가 이미 Kerberos 데이터베이스에 있어야 `kadmin`이 주체가 있는지 확인할 수 있습니다. Kerberos화된 서비스를 제공하는 애플리케이션 서버의 경우 keytab 파일은 기본적으로 `/etc/krb5/krb5.keytab`에 있습니다.

keytab은 사용자 암호와 비슷합니다. 사용자가 자신의 암호를 보호하는 것이 중요하듯이, 애플리케이션 서버가 해당 keytab 파일을 보호하는 것도 똑같이 중요합니다. keytab 파일은 항상 로컬 디스크에 저장하고 `root` 사용자만 읽을 수 있도록 설정해야 합니다. 또한 비보안 상태의 네트워크를 통해 keytab 파일을 전송해서도 안됩니다.

`root` 주체를 호스트의 keytab 파일에 추가해야 하는 특별한 경우가 있습니다. Kerberos 클라이언트의 사용자가 Kerberos화된 NFS 파일 시스템을 마운트하려는데 이때 루트와 동등한 액세스 권한이 필요한 경우, 클라이언트의 `root` 주체를 클라이언트의 keytab 파일에 추가해야 합니다. 또는 사용자가 자동 마운트를 사용하고 있지만 `root` 액세스 권한으로 Kerberos화된 NFS 파일 시스템을 마운트하려는 경우 항상 `kinit` 명령을 `root`로 사용하여 클라이언트의 `root` 주체에 대한 자격 증명을 얻어야 합니다.

keytab 파일을 관리할 때 사용할 수 있는 또 다른 명령은 `ktutil` 명령입니다. `ktutil`은 `kadmin`과 마찬가지로 Kerberos 데이터베이스와 상호 작용하지 않기 때문에 이 대화식 명령을 사용하면 Kerberos 관리 권한 없이도 로컬 호스트의 keytab 파일을 관리할 수 있습니다. 따라서 주체가 keytab 파일에 추가되면 `ktutil`을 사용하여 keytab 파일의 키 목록을 확인하거나 일시적으로 서비스에 대한 인증을 사용 안함으로 설정할 수 있습니다.

주 – kadmin의 ktadd 명령을 사용하여 keytab 파일에서 주체를 변경하면 새 키가 생성되어 keytab 파일에 추가됩니다.

Keytab 파일(작업 맵)

작업	설명	수행 방법
keytab 파일에 서비스 주체 추가	kadmin의 ktadd 명령을 사용하여 keytab 파일에 서비스 주체를 추가할 수 있습니다.	478 페이지 "Keytab 파일에 Kerberos 서비스 주체를 추가하는 방법"
keytab 파일에서 서비스 주체 제거	kadmin의 ktremove 명령을 사용하여 keytab 파일에서 서비스를 제거할 수 있습니다.	479 페이지 "Keytab 파일에서 서비스 주체를 제거하는 방법"
keytab 파일에 키 목록(주체 목록) 표시	ktutil 명령을 사용하여 keytab 파일에 키 목록을 표시할 수 있습니다.	480 페이지 "Keytab 파일에 키 목록(주체)을 표시하는 방법"
호스트에서 일시적으로 서비스에 대한 인증을 사용 안함으로 설정	이 절차를 통해 kadmin 권한 없이 호스트에서 일시적으로 서비스에 대한 인증을 사용 안함으로 신속하게 설정할 수 있습니다. ktutil을 사용하여 서버의 keytab 파일에서 서비스 주체를 삭제하기 전에 원본 keytab 파일을 임시 위치로 복사합니다. 서비스를 다시 사용으로 설정하려는 경우 원본 keytab 파일을 다시 적절한 위치로 복사합니다.	481 페이지 "호스트에서 일시적으로 서비스에 대한 인증을 사용 안함으로 설정하는 방법"

▼ Keytab 파일에 Kerberos 서비스 주체를 추가하는 방법

1. **주체가 이미 Kerberos 데이터베이스에 있는지 확인합니다.**
   자세한 내용은 453 페이지 "Kerberos 주체 목록을 보는 방법"을 참조하십시오.

2. **keytab 파일에 주체를 추가해야 하는 호스트에 수퍼유저로 로그인합니다.**

3. **kadmin 명령을 시작합니다.**

   ```
   # /usr/sbin/kadmin
   ```

4. **ktadd 명령을 사용하여 keytab 파일에 주체를 추가합니다.**

   kadmin: **ktadd** [**-e** *enctype*] [**-k** *keytab*] [**-q**] [*principal* | **-glob** *principal-exp*]

   -e *enctype* krb5.conf 파일에 정의된 암호화 유형 목록을 대체합니다.

`-k` *keytab*	keytab 파일을 지정합니다. 기본적으로 `/etc/krb5/krb5.keytab`이 사용됩니다.
`-q`	간단한 정보를 표시합니다.
principal	keytab 파일에 추가될 주체를 지정합니다. `host`, `root`, `nfs` 및 `ftp`를 서비스 주체로 추가할 수 있습니다.
`-glob` *principal-exp*	주체 표현식을 지정합니다. *principal-exp*와 일치하는 모든 주체가 keytab 파일에 추가됩니다. 주체 표현식 규칙은 `kadmin`의 `list_principals` 명령에 대한 규칙과 같습니다.

5 kadmin 명령을 종료합니다.

```
kadmin: quit
```

예 23–16 Keytab 파일에 서비스 주체 추가

다음 예에서는 KDC가 `denver`의 네트워크 서비스를 인증할 수 있도록 `denver`의 `host` 주체가 `denver`의 keytab 파일에 추가됩니다.

```
denver # /usr/sbin/kadmin
kadmin: ktadd host/denver.example.com
Entry for principal host/denver.example.com with kvno 3, encryption type AES-256 CTS
          mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type AES-128 CTS mode
          with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type Triple DES cbc mode
          with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type ArcFour
          with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type DES cbc mode
          with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin: quit
```

▼ Keytab 파일에서 서비스 주체를 제거하는 방법

1 keytab 파일에서 제거해야 하는 서비스 주체가 있는 호스트에 수퍼유저로 로그인합니다.

2 kadmin 명령을 시작합니다.

```
# /usr/sbin/kadmin
```

3 (옵션) keytab 파일에 주체(키)의 현재 목록을 표시하려면 ktutil 명령을 사용합니다.

자세한 지침은 480 페이지 “Keytab 파일에 키 목록(주체)을 표시하는 방법”을 참조하십시오.

4 ktremove 명령을 사용하여 keytab 파일에서 주체를 제거합니다.

```
kadmin: ktremove [-k keytab] [-q] principal [kvno | all | old ]
```

`-k` *keytab*	keytab 파일을 지정합니다. 기본적으로 `/etc/krb5/krb5.keytab`이 사용됩니다.
`-q`	간단한 정보를 표시합니다.
principal	keytab 파일에서 제거할 주체를 지정합니다.
kvno	키 버전 번호가 *kvno*와 일치하는 지정된 주체에 대한 모든 항목을 제거합니다.
`all`	지정된 주체에 대한 모든 항목을 제거합니다.
`old`	키 버전 번호가 가장 높은 주체를 제외하고, 지정된 주체에 대한 모든 항목을 제거합니다.

5 kadmin 명령을 종료합니다.

```
kadmin: quit
```

예 23–17 Keytab 파일에서 서비스 주체 제거

다음 예에서는 `denver`의 `host` 주체가 `denver`의 keytab 파일에서 제거됩니다.

```
denver # /usr/sbin/kadmin
kadmin: ktremove host/denver.example.com@EXAMPLE.COM
kadmin: Entry for principal host/denver.example.com@EXAMPLE.COM with kvno 3
  removed from keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin: quit
```

▼ Keytab 파일에 키 목록(주체)을 표시하는 방법

1 keytab 파일이 있는 호스트에 수퍼유저로 로그인합니다.

주 – 다른 사용자 소유의 keytab 파일을 만들 수는 있지만, keytab 파일의 기본 위치를 사용하려면 `root` 소유권이 필요합니다.

---

2 ktutil 명령을 시작합니다.

```
# /usr/bin/ktutil
```

3 read_kt 명령을 사용하여 keytab 파일을 키 목록 버퍼로 읽어 들입니다.

```
ktutil: read_kt keytab
```

4 list 명령을 사용하여 키 목록 버퍼를 표시합니다.

```
ktutil: list
```

현재 키 목록 버퍼가 표시됩니다.

5 ktutil 명령을 종료합니다.

```
ktutil: quit
```

예 23–18 Keytab 파일에 키 목록(주체) 표시

다음 예에서는 denver 호스트의 /etc/krb5/krb5.keytab 파일에 키 목록을 표시합니다.

```
denver # /usr/bin/ktutil
    ktutil: read_kt /etc/krb5/krb5.keytab
    ktutil: list
slot KVNO Principal
---- ---- ---------------------------------------
   1    5 host/denver@EXAMPLE.COM
    ktutil: quit
```

▼ 호스트에서 일시적으로 서비스에 대한 인증을 사용 안함으로 설정하는 방법

때때로 네트워크 애플리케이션 서버에서 일시적으로 rlogin 또는 ftp 등의 서비스에 대한 인증 방식을 사용 안함으로 설정해야 할 수 있습니다. 예를 들어 유지 보수 절차를 수행 중인 동안에는 사용자가 시스템에 로그인하지 못하도록 하고자 할 수 있습니다. ktutil 명령을 사용하면 kadmin 권한 없이도 서버의 keytab 파일에서 서비스 주체를 제거할 수 있으므로 이 작업이 가능합니다. 인증을 다시 사용으로 설정하려면 저장한 원본 keytab 파일을 다시 원본 위치로 복사하기만 하면 됩니다.

주 – 기본적으로 대부분의 서비스는 인증이 필요하도록 설정되어 있습니다. 인증이 필요하도록 서비스가 설정되지 않은 경우 서비스에 대한 인증을 사용 안함으로 설정하더라도 서비스가 계속 작동합니다.

---

1 keytab 파일이 있는 호스트에 수퍼유저로 로그인합니다.

주 – 다른 사용자 소유의 keytab 파일을 만들 수는 있지만, keytab 파일의 기본 위치를 사용하려면 root 소유권이 필요합니다.

---

2 현재 keytab 파일을 임시 파일에 저장합니다.

3 **ktutil 명령을 시작합니다.**

```
# /usr/bin/ktutil
```

4 **read_kt 명령을 사용하여 keytab 파일을 키 목록 버퍼로 읽어 들입니다.**

```
ktutil: read_kt keytab
```

5 **list 명령을 사용하여 키 목록 버퍼를 표시합니다.**

```
ktutil: list
```

현재 키 목록 버퍼가 표시됩니다. 사용 안함으로 설정하려는 서비스에 대한 슬롯 번호를 메모해 둡니다.

6 **호스트 서비스를 일시적으로 사용 안함으로 설정하려면 delete_entry 명령을 사용하여 키 목록 버퍼에서 특정 서비스 주체를 제거합니다.**

```
ktutil: delete_entry slot-number
```

여기서 *slot-number*는 list 명령으로 표시되는 삭제할 서비스 주체의 슬롯 번호를 지정합니다.

7 **write_kt 명령을 사용하여 새 keytab 파일에 키 목록 버퍼를 씁니다.**

```
ktutil: write_kt new-keytab
```

8 **ktutil 명령을 종료합니다.**

```
ktutil: quit
```

9 **새 keytab 파일을 이동합니다.**

```
# mv new-keytab keytab
```

10 **서비스를 다시 사용으로 설정하려는 경우 임시(원본) keytab 파일을 다시 원본 위치로 복사합니다.**

예 23–19 호스트에서 일시적으로 서비스를 사용 안함으로 설정

다음 예에서는 denver 호스트의 host 서비스가 일시적으로 사용 안함으로 설정되었습니다. denver에서 호스트 서비스를 다시 사용으로 설정하려면 krb5.keytab.temp 파일을 /etc/krb5/krb5.keytab 파일로 복사하십시오.

```
denver # cp /etc/krb5/krb5.keytab /etc/krb5/krb5.keytab.temp
denver # /usr/bin/ktutil
    ktutil:read_kt /etc/krb5/krb5.keytab
    ktutil:list
slot KVNO Principal
---- ---- ---------------------------------------
   1    8 root/denver@EXAMPLE.COM
   2    5 host/denver@EXAMPLE.COM
    ktutil:delete_entry 2
    ktutil:list
```

```
slot KVNO Principal
---- ---- -------------------------------------
   1    8 root/denver@EXAMPLE.COM
    ktutil:write_kt /etc/krb5/new.krb5.keytab
    ktutil: quit
denver # cp /etc/krb5/new.krb5.keytab /etc/krb5/krb5.keytab
```

◆ ◆ ◆ 24장

24 Kerberos 응용 프로그램 사용(작업)

이 장은 Kerberos 서비스가 구성되어 있는 시스템의 사용자를 대상으로 하며, 제공되는 "Kerberos화된" 명령과 서비스를 사용하는 방법에 대해 설명합니다. 이 장의 내용을 읽기 전에 이러한 명령(Kerberos화되지 않은 버전에 있음)에 대해 이미 잘 알고 있어야 합니다.

이 장은 일반 독자를 대상으로 하기 때문에 티켓 획득, 확인 및 삭제 등 티켓에 대한 내용을 다룹니다. 또한 Kerberos 암호 선택 또는 변경에 대한 내용도 다룹니다.

다음은 이 장에서 다루는 정보를 나열한 것입니다.

Oracle Solaris Kerberos 제품에 대한 개요는 19 장, "Kerberos 서비스 소개"를 참조하십시오.

Kerberos 티켓 관리

이 절에서는 티켓 획득, 확인 및 삭제 방법에 대해 설명합니다. 티켓에 대한 개요는 328 페이지 "Kerberos 서비스의 작동 방식"을 참조하십시오.

티켓의 이점

SEAM 릴리스 또는 Oracle Solaris 릴리스가 설치된 경우 Kerberos는 `login` 명령에 포함되어 있어 있으므로 로그인하면 자동으로 티켓이 획득됩니다. Kerberos화된 명령 `rsh`, `rcp`, `telnet` 및 `rlogin`은 보통 티켓의 복사본을 다른 시스템에 전달하도록 설정되어 있으므로, 티켓이 이러한 시스템에 액세스하도록 요청할 필요가 없습니다. 구성에 이 자동 전달이 포함되어 있지 않을 수 있지만, 이는 기본 동작입니다. 티켓 전달에 대한 자세한 내용은 494 페이지 "Kerberos화된 명령 개요" 및 497 페이지 "Kerberos 티켓 전달"을 참조하십시오.

티켓 수명에 대한 자세한 내용은 507 페이지 “티켓 수명”을 참조하십시오.

Kerberos 티켓 만들기

일반적으로 PAM이 제대로 구성된 경우, 로그인하면 티켓이 자동으로 생성되므로 티켓을 획득하기 위해 특별한 작업을 수행하지 않아도 됩니다. 그러나 티켓이 만료된 경우에는 티켓을 만들어야 합니다. 또한 예를 들어 `rlogin -l`을 사용하여 시스템에 다른 사용자로 로그인하는 경우 기본 주체가 아닌 다른 주체를 사용해야 합니다.

티켓을 만들려면 `kinit` 명령을 사용하십시오.

```
% /usr/bin/kinit
```

`kinit` 명령은 암호를 입력하라는 메시지를 표시합니다. `kinit` 명령의 전체 구문은 `kinit(1)` 매뉴얼 페이지를 참조하십시오.

예 24–1 Kerberos 티켓 만들기

이 예는 사용자 `jennifer`가 자신의 시스템에 티켓을 만드는 명령을 보여줍니다.

```
% kinit
Password for jennifer@ENG.EXAMPLE.COM:       <Type password>
```

여기서 사용자 `david`는 `-l` 옵션을 사용하여 세 시간 동안 유효한 티켓을 만듭니다.

```
% kinit -l 3h david@EXAMPLE.ORG
Password for david@EXAMPLE.ORG:       <Type password>
```

이 예는 사용자 `david`가 자신이 사용할 전달 가능 티켓(`-f` 옵션 사용)을 만드는 명령을 보여줍니다. 이 전달 가능 티켓을 사용하면 사용자가 두번째 시스템에 로그인한 다음 `telnet`을 통해 세번째 시스템에 로그인할 수 있습니다.

```
% kinit -f david@EXAMPLE.ORG
Password for david@EXAMPLE.ORG:          <Type password>
```

전달 가능 티켓의 작동 방식에 대한 자세한 내용은 497 페이지 “Kerberos 티켓 전달” 및 506 페이지 “티켓의 유형”을 참조하십시오.

Kerberos 티켓 확인

모든 티켓이 비슷한 것은 아닙니다. 예를 들면 어떤 티켓은 **전달 가능** 티켓이고, 다른 티켓은 **후일자** 티켓일 수 있습니다. 또 다른 티켓은 전송 가능 티켓인 동시에 후일자 티켓일 수 있습니다. `klist` 명령을 `-f` 옵션과 함께 사용하면 사용자가 보유한 티켓과 이러한 티켓의 속성을 확인할 수 있습니다.

```
% /usr/bin/klist -f
```

다음 기호는 `klist`에 의해 표시되는 각 티켓과 연관된 속성입니다.

A 사전 인증됨

D 후일자 가능

d 후일자

F 전달 가능

f 전달됨

I 초기

i 잘못됨

P 프록시 가능

p 프록시

R 갱신 가능

506 페이지 “티켓의 유형”에서는 티켓의 여러 가지 속성에 대해 설명합니다.

예 24–2 Kerberos 티켓 확인

이 예는 사용자 `jennifer`가 보유한 **초기** 티켓이 **전송 가능**(F)한 **후일자**(d) 티켓이지만, 아직 검증되지 않았음(i)을 보여줍니다.

```
% /usr/bin/klist -f
Ticket cache: /tmp/krb5cc_74287
Default principal: jennifer@EXAMPLE.COM

Valid starting                Expires                Service principal
09 Mar 04 15:09:51  09 Mar 04 21:09:51  nfs/EXAMPLE.COM@EXAMPLE.COM
        renew until 10 Mar 04 15:12:51, Flags: Fdi
```

다음 예는 사용자 `david`가 보유한 두 티켓이 다른 호스트에서 자신의 호스트로 **전달되었음**(f)을 보여줍니다. 이 티켓은 **전달 가능**(F) 티켓이기도 합니다.

```
% klist -f
Ticket cache: /tmp/krb5cc_74287
Default principal: david@EXAMPLE.COM
```

예 24–2 Kerberos 티켓 확인 *(계속)*

```
Valid starting                Expires                Service principal
07 Mar 04 06:09:51  09 Mar 04 23:33:51  host/EXAMPLE.COM@EXAMPLE.COM
        renew until 10 Mar 04 17:09:51, Flags: fF

Valid starting                Expires                Service principal
08 Mar 04 08:09:51  09 Mar 04 12:54:51  nfs/EXAMPLE.COM@EXAMPLE.COM
        renew until 10 Mar 04 15:22:51, Flags: fF
```

다음 예는 -e 옵션을 사용하여 세션 키 및 티켓의 암호화 유형을 표시하는 방법을 보여줍니다. -a 옵션은 이름 서비스가 변환을 수행할 수 있는 경우 호스트 주소를 호스트 이름에 매핑하는 데 사용됩니다.

```
% klist -fea
Ticket cache: /tmp/krb5cc_74287
Default principal: david@EXAMPLE.COM

Valid starting                Expires                Service principal
07 Mar 04 06:09:51  09 Mar 04 23:33:51  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 10 Mar 04 17:09:51, Flags: FRIA
        Etype(skey, tkt): DES cbc mode with RSA-MD5, DES cbc mode with CRC-32
        Addresses: client.example.com
```

Kerberos 티켓 삭제

현재 세션 중에 획득한 모든 Kerberos 티켓을 삭제하려면 kdestroy 명령을 사용하십시오. 이 명령은 자격 증명 캐시를 삭제하여 자격 증명과 티켓을 삭제합니다. 이 명령은 보통 필요하지 않지만 kdestroy를 실행하면 로그인하지 않은 동안 자격 증명 캐시가 손상될 가능성이 줄어듭니다.

티켓을 삭제하려면 kdestroy 명령을 사용하십시오.

```
% /usr/bin/kdestroy
```

kdestroy 명령은 티켓을 **모두** 삭제합니다. 티켓을 선택적으로 삭제할 수는 없습니다.

사용자가 자리를 비운 상태여서 침입자가 자신의 권한을 사용하는 것에 대해 우려되는 경우 kdestroy 또는 화면을 잠그는 화면 보호기를 사용해야 합니다.

Kerberos 암호 관리

Kerberos 서비스가 구성된 경우 일반 Solaris 암호와 Kerberos 암호, 두 개의 암호가 생깁니다. 두 암호를 동일하게 설정하거나 다르게 설정할 수 있습니다.

암호 선택에 대한 권장 사항

암호에는 입력 가능한 거의 모든 문자가 포함될 수 있습니다. 그러나 기본적으로 Control 키와 Return 키는 사용할 수 없습니다. 좋은 암호는 본인은 쉽게 기억할 수 있지만 다른 사람은 쉽게 추측할 수 없는 암호입니다. 잘못된 암호의 예는 다음과 같습니다.

- 사전에 있는 단어
- 일반 이름 또는 잘 알려진 이름
- 유명인 또는 유명 캐릭터의 이름
- 임의 형태로 된 본인의 이름 또는 사용자 이름(예: 이름 거꾸로 쓰기, 두 번 반복해서 쓰기 등)
- 배우자의 이름, 자녀 이름 또는 애완동물의 이름
- 본인의 생일 또는 친척의 생일
- 주민등록번호, 운전 면허증 번호, 여권 번호 또는 기타 유사 신분증 번호
- 이 설명서 또는 다른 설명서에 표시된 샘플 암호

좋은 암호는 8자 이상입니다. 또한 암호에는 대소문자, 숫자, 문장 부호 등의 문자를 혼합해서 사용해야 합니다. 다음은 이 설명서에 표시되지 않았다면 좋았을 암호 예제입니다.

- 머리글자어(예: “I2LMHinSF” - “I too left my heart in San Francisco”로 유추할 수 있음)
- 발음하기 좋은 무의미한 단어(예: “WumpaBun” 또는 “WangDangdoodle!”)
- 정교하게 철자를 잘못 쓴 문구(예: “6o'cluck” 또는 “RrriotGrrrlsRrrule!”)

주의 – 이 예제는 사용하지 마십시오. 이 설명서에 표시된 암호는 침입자가 가장 먼저 시도하는 암호입니다.

암호 변경

PAM이 제대로 구성된 경우 Kerberos 암호를 두 가지 방법으로 변경할 수 있습니다.

- 일반 UNIX passwd 명령 사용. Kerberos 서비스가 구성된 경우 passwd 명령이 자동으로 새 Kerberos 암호를 묻는 메시지를 표시합니다.

  kpasswd 대신 passwd를 사용할 경우의 이점은 UNIX 암호와 Kerberos 암호를 동시에 설정할 수 있다는 점입니다. 그러나 일반적으로 passwd를 사용하여 두 암호를 변경할 **필요**가 없습니다. 대개 UNIX 암호만 변경하고 Kerberos 암호를 그대로 두든지 아니면 그 반대일 수 있습니다.

  주 – passwd의 동작은 PAM 모듈이 구성된 방식에 따라 다릅니다. 일부 구성에서 두 암호를 모두 변경해야 할 수 있습니다. 일부 사이트에서는 UNIX 암호를 변경해야 하고, 다른 사이트에서는 Kerberos 암호를 변경해야 합니다.

- kpasswd 명령 사용. kpasswd는 passwd와 매우 비슷합니다. kpasswd는 Kerberos 암호만 변경한다는 점이 다릅니다. UNIX 암호를 변경하려는 경우에는 passwd를 사용해야 합니다.

  또 다른 차이점이라면 kpasswd는 유효한 UNIX 사용자가 아닌 Kerberos 주체에 대한 암호를 변경할 수 있다는 점입니다. 예를 들어 david/admin은 Kerberos 주체이기는 하지만 실제 UNIX 사용자는 아니므로, passwd 대신 kpasswd를 사용해야 합니다.

암호를 변경하면 변경 사항이 시스템을 통해(특히 광역 네트워크를 통해) 전파되는 데 약간의 시간이 걸립니다. 시스템이 설정된 방식에 따라 몇 분에서 몇 시간까지 지연될 수 있습니다. 암호를 변경한 즉시 새 Kerberos 티켓을 획득해야 하는 경우 새 암호를 먼저 사용해 보십시오. 새 암호가 작동하지 않을 경우 이전 암호를 사용하여 다시 시도하십시오.

시스템 관리자는 Kerberos V5 프로토콜을 통해 허용 가능한 암호에 대한 기준을 사용자별로 설정할 수 있습니다. 이러한 기준은 사용자별로 설정된 **정책**(또는 기본 정책)에 정의되어 있습니다. 정책에 대한 자세한 내용은 465 페이지 “Kerberos 정책 관리”를 참조하십시오.

예를 들어 사용자 jennifer의 정책(jenpol이라고 함)이 암호가 8자 이상이고 최소 두 가지 유형의 문자를 혼합해서 사용하도록 요구한다고 가정합시다. 따라서 kpasswd는 “sloth”를 암호로 사용하려는 시도를 거부합니다.

```
% kpasswd
kpasswd: Changing password for jennifer@ENG.EXAMPLE.COM.
Old password:        <Jennifer types her existing password>
kpasswd: jennifer@ENG.EXAMPLE.COM's password is controlled by
the policy jenpol
which requires a minimum of 8 characters from at least 2 classes
(the five classes are lowercase, uppercase, numbers, punctuation,
```

```
and all other characters).
New password:      <Jennifer types 'sloth'>
New password (again):       <Jennifer re-types 'sloth'>
kpasswd: New password is too short.
Please choose a password which is at least 4 characters long.
```

여기서 jennifer는 "slothrop49"를 암호로 사용합니다. "slothrop49"는 8자 이상이고 서로 다른 두 유형의 문자(숫자 및 소문자)를 포함하므로 기준을 충족합니다.

```
% kpasswd
kpasswd: Changing password for jennifer@ENG.EXAMPLE.COM.
Old password:       <Jennifer types her existing password>
kpasswd: jennifer@ENG.EXAMPLE.COM's password is controlled by
the policy jenpol
which requires a minimum of 8 characters from at least 2 classes
(the five classes are lowercase, uppercase, numbers, punctuation,
and all other characters).
New password:       <Jennifer types 'slothrop49'>
New password (again):       <Jennifer re-types 'slothrop49'>
Kerberos password changed.
```

예 24–3 암호 변경

다음 예에서 사용자 david는 passwd로 UNIX 암호와 Kerberos 암호를 모두 변경합니다.

```
% passwd
    passwd:  Changing password for david
    Enter login password:                   <Type the current UNIX password>
    New password:                           <Type the new UNIX password>
    Re-enter password:                      <Confirm the new UNIX password>
    Old KRB5 password:                      <Type the current Kerberos password>
    New KRB5 password:                      <Type the new Kerberos password>
    Re-enter new KRB5 password:             <Confirm the new Kerberos password>
```

passwd는 UNIX 암호와 Kerberos 암호를 모두 묻습니다. 이 동작은 기본 구성으로 설정됩니다. 이 경우 사용자 david는 다음에 표시된 것과 같이, kpasswd를 사용하여 자신의 Kerberos 암호를 다른 값으로 변경해야 합니다.

이 예는 사용자 david가 kpasswd로 자신의 Kerberos 암호만 변경함을 보여줍니다.

```
% kpasswd
kpasswd: Changing password for david@ENG.EXAMPLE.COM.
Old password:               <Type the current Kerberos password>
New password:               <Type the new Kerberos password>
New password (again):       <Confirm the new Kerberos password>
Kerberos password changed.
```

예 24–3 암호 변경 *(계속)*

이 예에서 사용자 david는 Kerberos 주체 david/admin(유효한 UNIX 사용자가 아님)에 대한 암호를 변경합니다. 이때 kpasswd를 사용해야 합니다.

```
% kpasswd david/admin
kpasswd:  Changing password for david/admin.
Old password:                <Type the current Kerberos password>
New password:                <Type the new Kerberos password>
New password (again):        <Type the new Kerberos password>
Kerberos password changed.
```

계정에 대한 액세스 권한 부여

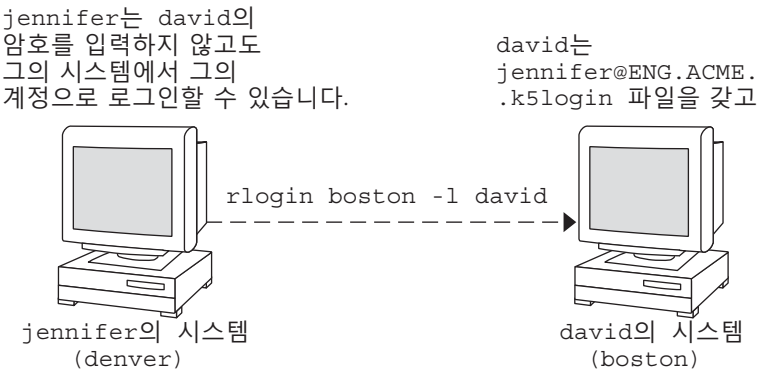
자신의 계정에 로그인할 수 있는 액세스 권한을 다른 사용자에게 부여해야 하는 경우, 홈 디렉토리에 .k5login 파일을 배치하면 암호 노출 없이 Kerberos를 통해 이러한 권한을 부여할 수 있습니다. .k5login 파일은 액세스 권한을 부여하려는 각 사용자에게 해당하는 하나 이상의 Kerberos 주체 목록입니다. 각 주체는 별도의 행에 있어야 합니다.

사용자 david가 다음과 비슷한 .k5login 파일을 자신의 홈 디렉토리에 보존한다고 가정합시다.

```
jennifer@ENG.EXAMPLE.COM
joe@EXAMPLE.ORG
```

이 파일은 사용자 jennifer와 joe가 이미 각 영역에 Kerberos 티켓을 보유하고 있다면 david의 ID를 사용하도록 허용합니다. 예를 들어 jennifer는 david의 암호를 제공하지 않고도 그의 시스템(boston)에 원격으로 로그인할 수 있습니다.

그림 24–1 .k5login 파일을 사용하여 계정에 대한 액세스 권한 부여



david의 홈 디렉토리가 NFS 마운트 디렉토리인 경우, Kerberos V5 프로토콜을 사용하면 다른(세번째) 시스템에서 jennifer가 전달 가능 티켓을 보유하고 있어야 자신의 홈 디렉토리에 액세스할 수 있습니다. 전달 가능 티켓 사용 예제는 486 페이지 “Kerberos 티켓 만들기”를 참조하십시오.

네트워크를 통해 다른 시스템에 로그인하는 경우 자신의 Kerberos 주체를 해당 시스템의 .k5login 파일에 포함시킬 수 있습니다.

.k5login 파일을 사용하는 것이 암호를 알려 주는 것보다 훨씬 안전한 이유는 다음과 같습니다.

- .k5login 파일에서 주체를 제거하여 언제든지 액세스 권한을 제거할 수 있습니다.
- 자신의 홈 디렉토리에 있는 .k5login 파일에 이름이 지정된 사용자 주체가 해당 시스템(.k5login 파일이 NFS를 통해 공유되는 경우, 시스템 집합)에서 자신의 계정에 대한 전체 액세스 권한을 갖습니다. 그러나 Kerberos화된 서비스는 자신이 아닌 해당 사용자의 ID를 기준으로 액세스를 인증합니다. 따라서 jennifer는 joe의 시스템에 로그인하여 작업을 수행할 수 있습니다. 그러나 jennifer가 ftp 또는 rlogin과 같은 Kerberos화된 프로그램을 사용하는 경우, 자신의 이름으로 작업을 수행합니다.
- Kerberos는 티켓을 획득한 사용자에 대한 로그를 보존하므로, 시스템 관리자는 필요한 경우 특정 시간에 다른 사용자의 사용자 ID를 사용할 수 있는 사용자를 확인할 수 있습니다.

.k5login 파일을 사용하는 한 가지 일반적인 방법은 이 파일을 root의 홈 디렉토리에 삽입하는 것입니다. 그러면 해당 시스템에 대한 root 액세스 권한이 나열된 Kerberos 주체에 부여됩니다. 이 구성을 사용하면 시스템 관리자가 로컬에서 root가 되거나, root 암호를 알려 줄 필요도 없고 다른 사용자가 네트워크를 통해 root 암호를 입력할 필요도 없이 원격에서 root로 로그인할 수 있습니다.

예 24–4 .k5login 파일을 사용하여 계정에 대한 액세스 권한 부여

jennifer가 boston.example.com 시스템에 root로 로그인한다고 가정합시다. 그녀의 주체 이름은 boston.example.com에 있는 root 홈 디렉토리의 .k5login 파일에 있으므로, 그녀가 다시 자신의 암호를 입력할 필요가 없습니다.

```
% rlogin boston.example.com -l root -x
This rlogin session is using DES encryption for all data transmissions.
Last login: Thu Jun 20 16:20:50 from daffodil
SunOS Release 5.7 (GENERIC) #2: Tue Nov 14 18:09:31 EST 1998
boston[root]%
```

Kerberos 사용자 명령

Kerberos V5 제품은 **단일 사인 온***(SSO)* 시스템이므로, 자신의 암호를 한 번만 입력하면 됩니다. Kerberos는 잘 알려진 기존 네트워크 프로그램의 각 제품군에 내장되어 있기 때문에 Kerberos V5 프로그램이 인증(및 선택적 암호화)을 수행합니다. Kerberos V5 응용 프로그램은 Kerberos 기능이 추가된 기존 UNIX 네트워크 프로그램의 버전입니다.

예를 들어 Kerberos화된 프로그램을 사용하여 원격 호스트에 연결할 경우, 프로그램, KDC 및 원격 호스트는 일련의 신속한 협상을 수행합니다. 이러한 협상이 완료되면 프로그램은 사용자를 대신해 원격 호스트에 대해 사용자의 ID를 검증하고, 원격 호스트는 사용자에게 액세스 권한을 부여합니다.

Kerberos화된 명령은 가장 먼저 Kerberos를 사용하여 인증을 시도합니다. Kerberos 인증이 실패하면 해당 명령과 함께 사용된 옵션에 따라 오류가 발생하거나 UNIX 인증이 시도됩니다. 자세한 내용은 각 Kerberos 명령 매뉴얼 페이지의 Kerberos Security 절을 참조하십시오.

Kerberos화된 명령 개요

Kerberos화된 네트워크 서비스는 인터넷을 통해 다른 시스템에 연결하는 프로그램으로, 다음과 같은 프로그램이 있습니다.

- ftp
- rcp
- rlogin
- rsh
- ssh
- telnet

이러한 프로그램에는 Kerberos 티켓을 투명하게 사용하여 인증 및 선택적 암호화를 원격 호스트와 협상하는 기능이 있습니다. 대부분의 경우 Kerberos는 사용자의 신원을 입증할 수 있는 증거를 제공하기 때문에 티켓 사용을 위해 암호를 입력할 필요가 없다는 점만 알고 있을 것입니다.

Kerberos V5 네트워크 프로그램에는 다음과 같은 기능의 옵션이 포함되어 있습니다.

- 사용자의 티켓을 다른 호스트로 전달(처음에 전달 가능 티켓을 획득한 경우)
- 사용자와 원격 호스트 간에 전송된 데이터 암호화

주 – 이 절에서는 사용자가 이미 이러한 프로그램의 Kerberos화되지 않은 버전을 잘 알고 있다고 가정하고 Kerberos V5 패키지에 추가된 Kerberos 기능을 중점적으로 설명합니다. 여기에 설명된 명령에 대한 자세한 설명은 해당 매뉴얼 페이지를 참조하십시오.

다음 Kerberos 옵션이 `ftp`, `rcp`, `rlogin`, `rsh` 및 `telnet`에 추가되었습니다.

`-a` 기존 티켓을 사용하여 자동 로그인을 시도합니다. 이름이 현재 사용자 ID와 같은 경우 `getlogin()`이 반환하는 사용자 이름을 사용합니다. 자세한 내용은 `telnet`(1) 매뉴얼 페이지를 참조하십시오.

`-f` **재전달할 수 없는** 티켓을 원격 호스트로 전달합니다. 이 옵션은 `-F` 옵션과 함께 사용할 수 없습니다. 이 두 옵션은 동일한 명령에 함께 사용할 수 없습니다.

세번째 호스트에서 다른 Kerberos 기반 서비스에 대해 자신을 인증해야 할 경우 티켓을 전달할 수 있습니다. 예를 들어 다른 시스템에 원격으로 로그인한 다음 거기에서 세번째 시스템에 원격으로 로그인할 수 있습니다.

원격 호스트의 홈 디렉토리가 Kerberos V5 방식을 사용하는 NFS 마운트 디렉토리인 경우 전달 가능 티켓을 사용해야 합니다. 그렇지 않으면 홈 디렉토리에 액세스할 수 없습니다. 즉, 처음에 시스템 1에 로그인한다고 가정합시다. 시스템 1에서 원격으로 자산의 홈 시스템인 시스템 2에 로그인한 다음 시스템 3의 홈 디렉토리를 마운트합니다. `-f` 또는 `-F` 옵션을 `rlogin`과 함께 사용한 경우가 아니라면 티켓을 시스템 3에 전달할 수 없으므로 자신의 홈 디렉토리에 연결할 수 없습니다.

기본적으로 `kinit`는 전달 가능한 TGT(티켓 부여 티켓)를 획득합니다. 그러나 이 경우 구성이 다를 수 있습니다.

티켓 전달에 대한 자세한 내용은 497 페이지 “Kerberos 티켓 전달”을 참조하십시오.

`-F` TGT의 **재전달 가능** 복사본을 원격 시스템에 전달합니다. `-f`와 비슷하지만, 추가(네번째 또는 다섯번째) 시스템에 액세스할 수 있도록

해줍니다. 따라서 -F 옵션은 -f 옵션의 수퍼 세트로 간주할 수 있습니다. -F 옵션은 -f 옵션과 함께 사용할 수 없습니다. 이 두 옵션은 동일한 명령에 함께 사용할 수 없습니다.

티켓 전달에 대한 자세한 내용은 497 페이지 “Kerberos 티켓 전달”을 참조하십시오.

-k *realm* krb5.conf 파일을 사용하여 영역 자체를 결정하는 대신 지정된 *realm*에서 원격 호스트에 대한 티켓을 요청합니다.

-K 티켓을 사용하여 원격 호스트에 대해 인증하지만, 자동으로 로그인되지는 않습니다.

-m *mechanism* /etc/gss/mech 파일에 나열된 것과 같이, 사용할 GSS-API 보안 방식을 지정합니다. 기본적으로 kerberos_v5로 설정됩니다.

-x 이 세션을 암호화합니다.

-X *auth-type* 인증의 *auth-type* 유형을 사용 안함으로 설정합니다.

다음 표는 명령의 특정 옵션을 보여줍니다. “X”는 명령에 해당 옵션이 있음을 나타냅니다.

표 24–1 네트워크 명령의 Kerberos 옵션

	ftp	rcp	rlogin	rsh	telnet
-a					X
-f	X		X	X	X
-F			X	X	X
-k		X	X	X	X
-K					X
-m	X				
-x	X	X	X	X	X
-X					X

또한 ftp를 사용할 경우 프롬프트에서 세션의 보호 레벨을 설정할 수 있습니다.

clear 보호 레벨을 “clear”(보호 없음)로 설정합니다. 이 보호 레벨은 기본값입니다.

private 보호 레벨을 “private”로 설정합니다. 데이터 전송 시 암호화를 통해 기밀성 및 무결성 보호됩니다. 그러나 일부 Kerberos 사용자만 프라이버시 서비스를 사용할 수 있습니다.

safe 보호 레벨을 “safe”로 설정합니다. 데이터 전송 시 암호화 체크섬을 통해 무결성 보호됩니다.

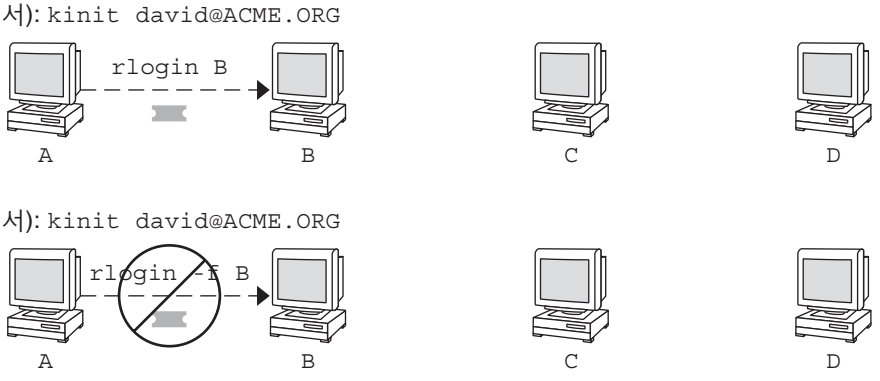
`protect`와 위에 표시된 보호 레벨(`clear`, `private` 또는 `safe`) 중 하나를 입력하여 `ftp` 프롬프트에서 보호 레벨을 설정할 수도 있습니다.

Kerberos 티켓 전달

494 페이지 “Kerberos화된 명령 개요”에 설명된 것과 같이, 일부 명령의 경우 `-f` 또는 `-F` 옵션을 사용하여 티켓을 전달할 수 있습니다. 티켓을 전달하면 네트워크 트랜잭션을 "연쇄적으로" 수행할 수 있습니다. 예를 들어 한 시스템에 원격으로 로그인한 다음 거기에서 다른 시스템에 원격으로 로그인할 수 있습니다. `-f` 옵션을 사용하면 티켓을 전달할 수 있고, `-F` 옵션을 사용하면 전달된 티켓을 재전달할 수 있습니다.

다음 그림에서 사용자 `david`는 `kinit`를 사용하여 전달할 수 없는 TGT(티켓 부여 티켓)를 획득합니다. `-f` 옵션을 지정하지 않았기 때문에 이 티켓은 전달할 수 없는 티켓입니다. 시나리오 1에서는 시스템 B에 원격으로 로그인할 수는 있지만, 더 이상은 로그인할 수 없습니다. 시나리오 2에서는 david가 전달할 수 없는 티켓을 전달하려고 시도했기 때문에 `rlogin -f` 명령이 실패합니다.

그림 24–2 전달할 수 없는 티켓 사용



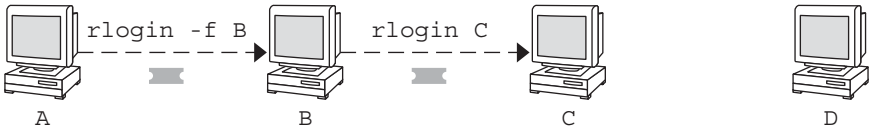
실제로 Kerberos 구성 파일은 `kinit`가 기본적으로 전달 가능 티켓을 획득하도록 설정됩니다. 그러나 사용자의 구성은 다를 수 있습니다. 설명을 위해, `kinit -f`를 사용하여 호출되지 않는 한 `kinit`가 전달 가능 TGT를 획득하지 **않는다고** 가정합니다. 그러나 `kinit`에는 `-F` 옵션이 없습니다. TGT는 전송 가능하거나 전송 불가능합니다.

다음 그림에서 사용자 `david`는 `kinit -f`를 사용하여 전달 가능 TGT를 획득합니다. 시나리오 3에서는 david가 `rlogin`과 함께 전달 가능 티켓을 사용하므로 시스템 C에

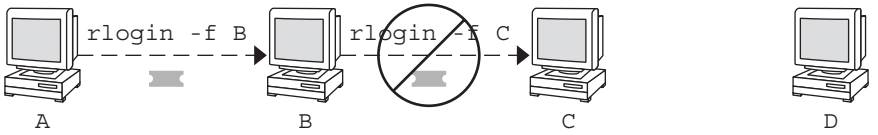
연결할 수 있습니다. 시나리오 4에서는 티켓을 재전달할 수 없으므로 rlogin 명령이 또 다시 실패합니다. 시나리오 5에서와 같이 -F 옵션을 대신 사용하면 두번째 rlogin이 성공하고 티켓을 시스템 D로 재전달할 수 있습니다.

그림 24–3 전달 가능 없는 티켓 사용

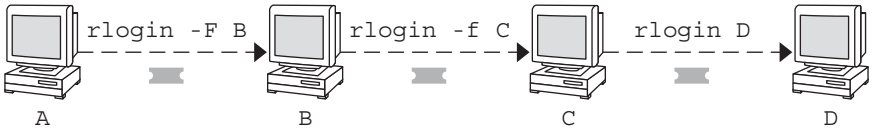
3. (A에서): kinit -f david@ACME.ORG



4. (A에서): kinit -f david@ACME.ORG



5. (A에서): kinit -f david@ACME.ORG



Kerberos화된 명령 사용(예제)

다음 예제는 Kerberos화된 명령에 대한 옵션이 어떻게 작동하는지 보여줍니다.

예 24–5 telnet과 함께 -a, -f 및 -x 옵션 사용

이 예에서 사용자 david는 이미 로그인했으며 telnet을 통해 denver.example.com 시스템에 연결하려고 합니다. 그는 -f 옵션을 사용하여 기존 티켓을 전달하고, -x 옵션을 사용하여 세션을 암호화하며, -a 옵션을 사용하여 자동으로 로그인을 수행합니다. 세번째 호스트의 서비스를 사용하지 않을 것이기 때문에 그는 -F 대신 -f를 사용할 수 있습니다.

```
% telnet -a -f -x denver.example.com
Trying 128.0.0.5...
Connected to denver.example.com. Escape character is '^]'.
[ Kerberos V5 accepts you as "david@eng.example.com" ]
[ Kerberos V5 accepted forwarded credentials ]
SunOS 5.9: Tue May 21 00:31:42 EDT 2004  Welcome to SunOS
%
```

예 24–5 telnet과 함께 -a, -f 및 -x 옵션 사용 *(계속)*

david의 시스템은 Kerberos를 사용하여 denver.example.com에 대해 그를 인증한 다음 자동으로 david로 로그인되었습니다. 따라서 암호화된 세션, 그를 기다리고 있는 티켓 복사본이 있으므로 자신의 암호를 입력할 필요가 없습니다. Kerberos 버전이 아닌 telnet을 사용한 경우, 암호를 입력하라는 메시지가 표시되므로 네트워크를 통해 암호를 암호화되지 않은 상태로 전송했을 수 있습니다. 이때 침입자가 네트워크 트래픽을 관찰하고 있었다면 david의 암호를 알게 되었을 것입니다.

Kerberos 티켓을 전달할 경우 telnet(및 여기에 설명된 다른 명령)은 종료될 때 티켓을 삭제합니다.

예 24–6 -F 옵션과 함께 rlogin 사용

여기서는 사용자 jennifer가 자신의 시스템인 boston.example.com에 로그인하려고 합니다. 그녀는 -F 옵션을 사용하여 기존 티켓을 전달한 다음 -x 옵션을 사용하여 세션을 암호화합니다. 그녀는 boston에 로그인한 후 티켓 재전달이 필요한 다른 네트워크 트랜잭션을 수행할 수 있으므로 -f 대신 -F를 선택합니다. 물론 기존 티켓을 전달하기 때문에 암호를 입력할 필요가 없습니다.

```
% rlogin boston.example.com -F -x
This rlogin session is using encryption for all transmissions.
Last login Mon May 19 15:19:49 from daffodil
SunOS Release 5.9 (GENERIC) #2 Tue Nov 14 18:09:3 EST 2003
%
```

예 24–7 ftp에서 보호 레벨 설정

joe는 ftp를 통해 denver.example.com 시스템의 ~joe/MAIL 디렉토리에서 자신의 메일에 연결하여 세션을 암호화한다고 가정합시다. 다음과 같이 교환이 이루어집니다.

```
% ftp -f denver.example.com
Connected to denver.example.com
220 denver.example.org FTP server (Version 6.0) ready.
334 Using authentication type GSSAPI; ADAT must follow
GSSAPI accepted as authentication type
GSSAPI authentication succeeded Name (daffodil.example.org:joe)
232 GSSAPI user joe@MELPOMENE.EXAMPLE.COM is authorized as joe
230 User joe logged in.
Remote system type is UNIX.
Using BINARY mode to transfer files.
ftp> protect private
200 Protection level set to Private
ftp> cd ~joe/MAIL
250 CWD command successful.
ftp> get RMAIL
227 Entering Passive Mode (128,0,0,5,16,49)
150 Opening BINARY mode data connection for RMAIL (158336 bytes).
226 Transfer complete. 158336 bytes received in 1.9 seconds (1.4e+02 Kbytes/s)
ftp> quit
%
```

예 24–7 ftp에서 보호 레벨 설정 *(계속)*

세션을 암호화하기 위해 `joe`는 보호 레벨을 `private`로 설정합니다.

◆ ◆ ◆ 25장

25 Kerberos 서비스(참조)

이 장에서는 Kerberos 제품에 포함된 여러 가지 파일, 명령 및 데몬에 대해 설명합니다. 또한 Kerberos 인증의 작동 방식에 대한 자세한 정보를 제공합니다.

이 장에서 다루는 참조 정보는 다음과 같습니다.

Kerberos 파일

이 절에서는 Kerberos 서비스에서 사용하는 몇 가지 파일에 대해 설명합니다.

표 25–1 Kerberos 파일

파일 이름	설명
`~/.gkadmin`	SEAM 도구에서 새 주체를 만드는 데 사용되는 기본값입니다.
`~/.k5login`	Kerberos 계정에 대한 액세스 권한을 부여하는 주체 목록입니다.
`/etc/krb5/kadm5.acl`	KDC 관리자의 주체 이름과 해당 Kerberos 관리 권한을 포함하고 있는 Kerberos 액세스 제어 목록 파일입니다.

표 25–1 Kerberos 파일 *(계속)*

파일 이름	설명
`/etc/krb5/kadm5.keytab`	더 이상 사용되지 않음: 이 파일은 Oracle Solaris 11 릴리스에서 제거되었습니다.
`/etc/krb5/kdc.conf`	KDC 구성 파일입니다.
`/etc/krb5/kpropd.acl`	Kerberos 데이터베이스 전파 구성 파일입니다.
`/etc/krb5/krb5.conf`	Kerberos 영역 구성 파일입니다.
`/etc/krb5/krb5.keytab`	네트워크 응용 프로그램 서버에 대한 Keytab 파일입니다.
`/etc/krb5/warn.conf`	Kerberos 티켓 만료 경고 및 자동 갱신 구성 파일입니다.
`/etc/pam.conf`	PAM 구성 파일입니다.
`/tmp/krb5cc_`*uid*	기본 자격 증명 캐시입니다. 여기서 *uid*는 사용자의 십진수 UID입니다.
`/tmp/ovsec_adm.`*xxxxxx*	암호 변경 작업의 수명에 대한 임시 자격 증명 캐시입니다. 여기서 *xxxxxx*는 임의 문자열입니다.
`/var/krb5/.k5.`*REALM*	KDC 마스터 키의 복사본을 포함하는 KDC stash 파일입니다.
`/var/krb5/kadmin.log`	`kadmind`에 대한 로그 파일입니다.
`/var/krb5/kdc.log`	KDC에 대한 로그 파일입니다.
`/var/krb5/principal`	Kerberos 주체 데이터베이스입니다.
`/var/krb5/principal.kadm5`	정책 정보를 포함하는 Kerberos 관리 데이터베이스입니다.
`/var/krb5/principal.kadm5.lock`	Kerberos 관리 데이터베이스 잠금 파일입니다.
`/var/krb5/principal.ok`	Kerberos 데이터베이스가 성공적으로 초기화될 때 생성되는 Kerberos 주체 데이터베이스 초기화 파일입니다.
`/var/krb5/principal.ulog`	증분 전파에 대한 업데이트를 포함하는 Kerberos 업데이트 로그입니다.
`/var/krb5/slave_datatrans`	전파를 위해 `kprop_script` 스크립트가 사용하는 KDC의 백업 파일입니다.
`/var/krb5/slave_datatrans_`*slave*	지정된 *slave*에 대한 전체 업데이트를 수행할 때 생성되는 임시 덤프 파일입니다.

Kerberos 명령

이 절에서는 Kerberos 제품에 포함된 몇 가지 명령에 대해 설명합니다.

표 25–2 Kerberos 명령

명령	설명
/usr/bin/ftp	FTP(File Transfer Protocol) 프로그램입니다.
/usr/bin/kdestroy	Kerberos 티켓을 삭제합니다.
/usr/bin/kinit	Kerberos TGT(티켓 부여 티켓)를 얻어 캐시에 저장합니다.
/usr/bin/klist	현재 Kerberos 티켓을 표시합니다.
/usr/bin/kpasswd	Kerberos 암호를 변경합니다.
/usr/bin/ktutil	Kerberos Keytab 파일을 관리합니다.
/usr/bin/kvno	Kerberos 주체의 키 버전 번호를 나열합니다.
/usr/bin/rcp	원격 파일 복사 프로그램입니다.
/usr/bin/rlogin	원격 로그인 프로그램입니다.
/usr/bin/rsh	원격 셸 프로그램입니다.
/usr/bin/telnet	Kerberos화된 telnet 명령입니다.
/usr/lib/krb5/kprop	Kerberos 데이터베이스 전파 프로그램입니다.
/usr/sbin/gkadmin	주체 및 정책을 관리하는 데 사용되는 Kerberos 데이터베이스 관리 GUI 프로그램입니다.
/usr/sbin/gsscred	gsscred 테이블 항목을 관리합니다.
/usr/sbin/kadmin	주체, 정책 및 Keytab 파일을 관리하는 데 사용되는 원격 Kerberos 데이터베이스 관리 프로그램입니다(Kerberos 인증과 함께 실행됨).
/usr/sbin/kadmin.local	주체, 정책 및 Keytab 파일을 관리하는 데 사용되는 로컬 Kerberos 데이터베이스 관리 프로그램으로, Kerberos 인증 없이 실행되며 마스터 KDC에서 실행해야 합니다.
/usr/sbin/kclient	설치 프로파일과 함께 사용되거나 설치 프로파일 없이 사용되는 Kerberos 클라이언트 설치 스크립트입니다.
/usr/sbin/kdb5_ldap_util	Kerberos 데이터베이스용 LDAP 컨테이너를 만듭니다.
/usr/sbin/kdb5_util	Kerberos 데이터베이스 및 stash 파일을 만듭니다.
/usr/sbin/kgcmgr	Kerberos 마스터 및 슬레이브 KDC를 구성합니다.

표 25–2 Kerberos 명령 *(계속)*

명령	설명
/usr/sbin/kproplog	업데이트 로그에 업데이트 항목에 대한 요약 정보를 나열합니다.

Kerberos 데몬

다음 표는 Kerberos 제품이 사용하는 데몬에 대해 설명합니다.

표 25–3 Kerberos 데몬

데몬	설명
/usr/sbin/in.ftpd	FTP(File Transfer Protocol) 데몬입니다.
/usr/lib/krb5/kadmind	Kerberos 데이터베이스 관리 데몬입니다.
/usr/lib/krb5/kpropd	Kerberos 데이터베이스 전파 데몬입니다.
/usr/lib/krb5/krb5kdc	Kerberos 티켓 처리 데몬입니다.
/usr/lib/krb5/ktkt_warnd	Kerberos 티켓 만료 경고 및 자동 갱신 데몬입니다.
/usr/sbin/in.rlogind	원격 로그인 데몬입니다.
/usr/sbin/in.rshd	원격 셸 데몬입니다.
/usr/sbin/in.telnetd	telnet 데몬입니다.

Kerberos 용어

다음 절에서는 Kerberos 용어와 해당 용어에 대한 설명을 제공합니다. 이러한 용어는 Kerberos 설명서 전체에서 사용됩니다. Kerberos 개념을 파악하려면 이러한 용어를 숙지해야 합니다.

Kerberos 관련 용어

KDC를 관리하기 위해서는 이 절의 용어를 잘 알고 있어야 합니다.

키 배포 센터 또는 *KDC*는 자격 증명 발행을 담당하는 Kerberos 구성 요소입니다. 이러한 자격 증명은 KDC 데이터베이스에 저장된 정보를 사용하여 생성됩니다. 각 영역에는 최소 두 개의 KDC 즉, 한 개의 마스터와 최소 한 개의 슬레이브가 필요합니다. 모든 KDC에서 자격 증명을 생성하지만, 마스터 KDC만 KDC 데이터베이스에 대한 변경 사항을 처리합니다.

stash **파일**에는 KDC에 대한 마스터 키가 포함되어 있습니다. 이 키는 `kadmind` 및 `krb5kdc` 명령을 시작하기 전에 자동으로 KDC를 인증하기 위해 서버가 재부트될 때 사용됩니다. 이 파일에는 마스터 키가 포함되어 있기 때문에 파일 및 파일 백업이 보안 상태로 보존되어야 합니다. 이 파일은 `root`에 대한 읽기 전용 권한으로 생성됩니다. 파일을 보안 상태로 보존하려면 권한을 변경하지 마십시오. 파일이 손상되면 이 키를 사용하여 KDC 데이터베이스를 액세스하거나 수정할 수 있습니다.

인증 관련 용어

인증 프로세스를 이해하기 위해서는 이 절의 용어를 알고 있어야 합니다. 프로그래머와 시스템 관리자는 이러한 용어에 익숙해야 합니다.

클라이언트는 사용자의 워크스테이션에서 실행되는 소프트웨어입니다. 클라이언트에서 실행되는 Kerberos 소프트웨어는 이 프로세스 중 많은 요청을 생성합니다. 따라서 이 소프트웨어의 작업과 사용자의 작업을 구분하는 것이 중요합니다.

서버 및 **서비스**라는 용어는 대개 서로 바꿔서 사용됩니다. 명확히 하자면, **서버**라는 용어는 Kerberos 소프트웨어가 실행 중인 물리적 시스템을 정의하는 데 사용됩니다. **서비스**는 서버에서 지원되는 특정 기능(예: `ftp` 또는 `nfs`)에 해당합니다. 설명서에서는 서버를 서비스의 일부로 설명하는 경우가 많은데, 이는 이러한 용어의 의미를 흐리게 합니다. 따라서 **서버**는 물리적 시스템을 나타내고, **서비스**라는 용어는 소프트웨어를 나타냅니다.

Kerberos 제품은 두 가지 유형의 키를 사용합니다. 한 유형의 키는 암호에서 파생된 키입니다. 암호에서 파생된 키는 각 사용자 주체에게 제공되며 해당 사용자와 KDC만 알 수 있습니다. Kerberos 제품에서 사용하는 다른 유형의 키는 암호와 관련이 없는 임의 키입니다. 따라서 사용자 주체가 사용하기에 적합하지 않습니다. 임의 키는 보통 KDC에서 생성하는 Keytab 및 세션 키에 항목이 있는 서비스 주체에 사용됩니다. 서비스는 개별적으로 실행될 수 있도록 해주는 Keytab의 키에 액세스할 수 있으므로 서비스 주체는 임의 키를 사용할 수 있습니다. 세션 키는 KDC에서 생성되고 클라이언트와 서버 간에 공유되어 클라이언트와 서비스 간에 보안 트랜잭션을 제공합니다.

티켓은 사용자 ID를 서버나 서비스로 안전하게 전달하는 데 사용되는 정보 패킷으로, 티켓은 단일 클라이언트에만, 그리고 특정 서버의 특정 서비스에만 유효합니다. 티켓은 다음으로 구성됩니다.

- 서비스의 주체 이름
- 사용자의 주체 이름
- 사용자 호스트의 IP 주소
- 시간 기록
- 티켓 수명을 정의하는 값
- 세션 키 복사본

이러한 데이터는 모두 서버의 서비스 키로 암호화됩니다. KDC에서는 아래에 설명된 자격 증명에 포함되어 있는 티켓을 발행합니다. 티켓은 발행된 후 만료될 때까지 재사용할 수 있습니다.

자격 증명은 티켓 및 일치하는 세션 키를 포함하는 정보 패킷으로, 요청 주체의 키로 암호화됩니다. 일반적으로 KDC에서는 클라이언트에서 보내는 티켓 요청에 대한 응답으로 자격 증명을 생성합니다.

인증자는 서버에서 클라이언트 사용자 주체를 인증하기 위해 사용하는 정보로, 사용자의 주체 이름, 시간 기록 및 기타 데이터를 포함합니다. 티켓과 달리, 인증자는 보통 서비스에 대한 액세스가 요청될 때 한 번만 사용할 수 있습니다. 인증자는 클라이언트와 서버가 공유하는 세션 키를 사용하여 암호화됩니다. 일반적으로 클라이언트는 인증자를 만들어 서버 또는 서비스의 티켓과 함께 전송하여 서버 또는 서비스에서 인증됩니다.

티켓의 유형

티켓에는 티켓의 사용 방식을 제어하는 등록 정보가 있습니다. 이러한 등록 정보는 티켓을 만들 때 지정되지만, 나중에 사용자가 티켓의 등록 정보를 수정할 수도 있습니다. 예를 들어 티켓은 `forwardable`에서 `forwarded`로 변경될 수 있습니다. 티켓 등록 정보는 `klist` 명령으로 확인할 수 있습니다. 487 페이지 “Kerberos 티켓 확인”을 참조하십시오.

티켓은 다음 용어 중 하나 이상으로 설명할 수 있습니다.

전달 가능/전달됨
: 전달 가능 티켓은 한 호스트에서 다른 호스트로 전송될 수 있으므로, 클라이언트 재인증이 필요하지 않습니다. 예를 들어 사용자 `david`가 사용자 `jennifer`의 시스템에서 전달 가능 티켓을 얻은 경우, 새 티켓 없이도 자신의 시스템에 로그인할 수 있으므로 다시 인증할 필요가 없습니다. 전달 가능 티켓에 대한 예는 예 24–1을 참조하십시오.

초기
: 초기 티켓이란 TGT(티켓 부여 티켓)를 기반으로 하지 않고 직접 발행되는 티켓입니다. 암호가 달라지는 응용 프로그램과 같은 일부 서비스의 경우 티켓을 초기로 표시해야 클라이언트에 보안 키가 있음을 입증할 수 있습니다. 초기 티켓은 클라이언트가 TGT(티켓 부여 티켓)에 의존하지 않고 최근에 인증되었음을 나타내며, 오랫동안 지속되었을 수 있습니다.

잘못됨
: 잘못된 티켓은 아직 사용 가능하지 않은 후일자 티켓으로, 검증될 때까지 애플리케이션 서버에서 거부됩니다. 티켓을 검증하려면 티켓 시작 시간이 경과한 후 클라이언트가 TGS(티켓 부여 서비스) 요청에서 `VALIDATE` 플래그를 설정하여 티켓을 KDC에 제공해야 합니다.

후일자 가능/후일자
: 후일자 티켓이란 생성 후 지정된 시간이 경과해야 유효해지는 티켓입니다. 예를 들어 티켓이 도난 당하더라도 일괄 처리

	작업이 실행될 때까지는 사용할 수 없으므로, 이러한 티켓은 나중에 야간에 실행하려는 일괄 처리 작업에 유용합니다. 후일자 티켓은 발행된 후 시작 시간이 경과할 때까지 유효하지 않은 상태로 유지되며, 클라이언트에서는 KDC의 검증을 요청합니다. 후일자 티켓은 보통 TGT(티켓 부여 티켓)의 만료 시간까지 유효합니다. 그러나 티켓이 갱신 가능한 것으로 표시된 경우 티켓의 수명은 TGT(티켓 부여 티켓)의 전체 수명 지속 기간과 동일하게 설정됩니다.
프록시 가능/프록시	때때로 서비스가 주체를 대신해 작업을 수행해야 하는 경우가 있습니다. 티켓을 만들 때 프록시의 주체 이름을 지정해야 합니다. Oracle Solaris 릴리스에서는 프록시 가능 또는 프록시 티켓을 지원하지 않습니다. 프록시 가능 티켓은 전달 가능 티켓과 비슷하지만, 단일 서비스에 대해서만 유효한 반면 전달 가능 티켓은 서비스에 클라이언트 ID의 완전한 사용 권한을 부여합니다. 따라서 전달 가능 티켓은 일종의 수퍼 프록시로 간주될 수 있습니다.
갱신 가능	티켓을 매우 오랫동안 사용하는 것은 보안상 위험하므로 티켓을 갱신 가능하도록 지정할 수 있습니다. 갱신 가능 티켓에는 두 개의 만료 시기가 있습니다. 즉, 티켓의 현재 인스턴스가 만료되는 시기와 티켓의 최대 수명(1주)입니다. 클라이언트에서 계속 티켓을 사용하려는 경우 첫번째 만료가 발생하기 전에 티켓을 갱신합니다. 예를 들어 모든 티켓의 최대 수명이 10시간일 경우 한 티켓은 한 시간 동안 유효할 수 있습니다. 티켓을 보유하고 있는 클라이언트가 티켓을 한 시간 이상 보존하려는 경우에는 해당 시간 내에 티켓을 갱신해야 합니다. 티켓이 최대 티켓 수명(10시간)에 도달하면 자동으로 만료되므로 갱신할 수 없습니다.

티켓 속성을 확인하는 방법은 487 페이지 “Kerberos 티켓 확인”을 참조하십시오.

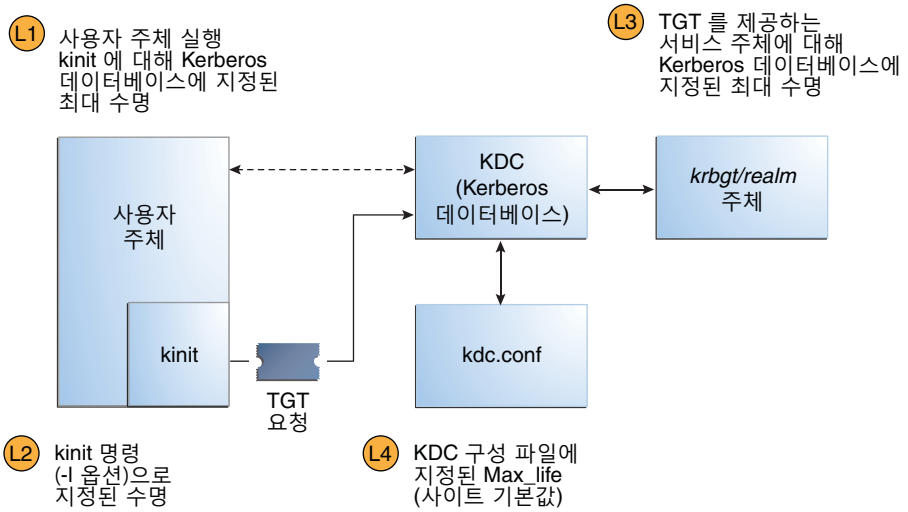
티켓 수명

주체가 TGT(티켓 부여 티켓)를 포함한 티켓을 얻으면 티켓의 수명은 다음 수명 값 중 가장 작은 값으로 설정됩니다.

- `kinit`의 `-l` 옵션에 지정된 수명 값(`kinit`를 사용하여 티켓을 얻은 경우). 기본적으로 `kinit`가 최대 수명 값에 사용되었습니다.
- `kdc.conf` 파일에 지정된 최대 수명 값(`max_life`)
- 티켓을 제공하는 서비스 주체의 Kerberos 데이터베이스에 지정된 최대 수명 값. `kinit`의 경우 서비스 주체는 `krbtgt/`*realm*입니다.
- 티켓을 요청하는 사용자 주체의 Kerberos 데이터베이스에 지정된 최대 수명 값

그림 25–1은 TGT의 수명이 결정되는 방식과 네 가지 수명 값이 비롯되는 위치를 보여줍니다. 이 그림은 TGT의 수명이 결정되는 방식을 보여주지만, 기본적으로 주체가 티켓을 얻을 때 발생하는 방식과 동일합니다. 단, `kinit`가 수명 값을 제공하지 않고, 티켓을 제공하는 서비스 주체가 (`krbtgt/`*realm* 주체 대신) 최대 수명 값을 제공한다는 점만 다릅니다.

그림 25–1 TGT의 수명이 결정되는 방식



티켓 수명 = L1, L2, L3, 및 L4의 최소 값

갱신 가능 티켓의 수명도 네 값 중 가장 작은 값에서 결정되지만, 다음과 같이 갱신 가능한 수명 값이 대신 사용됩니다.

- `kinit`의 `-r` 옵션으로 지정된 갱신 가능한 수명 값(`kinit`를 사용하여 티켓을 얻었거나 갱신한 경우)
- `kdc.conf` 파일에 지정된 최대 갱신 가능한 수명 값(`max_renewable_life`)
- 티켓을 제공하는 서비스 주체의 Kerberos 데이터베이스에 지정된 최대 갱신 가능한 수명 값. `kinit`의 경우 서비스 주체는 `krbtgt/`*realm*입니다.
- 티켓을 요청하는 사용자 주체의 Kerberos 데이터베이스에 지정된 갱신 가능한 최대 수명 값

Kerberos 주체 이름

각 티켓은 주체 이름으로 식별됩니다. 주체 이름으로 사용자나 서비스를 식별할 수 있습니다. 다음은 몇 가지 주체 이름에 대한 예제입니다.

표 25–4 Kerberos 주체 이름 예제

주체 이름	설명
changepw/kdc1.example.com@EXAMPLE.COM	암호를 변경할 때 KDC에 액세스할 수 있도록 해주는 마스터 KDC 서버에 대한 주체입니다.
clntconfig/admin@EXAMPLE.COM	kclient 설치 유틸리티에서 사용하는 주체입니다.
ftp/boston.example.com@EXAMPLE.COM	ftp 서비스에서 사용하는 주체입니다. 이 주체는 host 주체 대신 사용할 수 있습니다.
host/boston.example.com@EXAMPLE.COM	Kerberos화된 응용 프로그램(예: klist 및 kprop) 및 서비스(예: ftp 및 telnet)에서 사용하는 주체입니다. 이 주체를 host 또는 서비스 주체라고 합니다. 이 주체는 NFS 마운트를 인증하는 데 사용되며, 클라이언트에 발행된 TGT가 올바른 KDC에서 제공되는지 클라이언트에서 확인하는 데도 사용됩니다.
K/M@EXAMPLE.COM	마스터 키 이름 주체입니다. 마스터 KDC별로 하나의 마스터 키 이름이 연관됩니다.
kadmin/history@EXAMPLE.COM	다른 주체에 대한 암호 사용 기록을 보존하는 데 사용되는 키를 포함하는 주체입니다. 하나의 마스터 KDC에는 이러한 주체 중 하나가 포함되어 있습니다.
kadmin/kdc1.example.com@EXAMPLE.COM	kadmind 명령을 사용하여 KDC에 액세스할 수 있도록 해주는 마스터 KDC 서버에 대한 주체입니다.
kadmin/changepw.example.com@EXAMPLE.COM	Oracle Solaris 릴리스에서 실행 중이지 않은 클라이언트의 암호 변경 요청을 수락하는 데 사용되는 주체입니다.
krbtgt/EXAMPLE.COM@EXAMPLE.COM	이 주체는 TGT(티켓 부여 티켓)를 생성할 때 사용됩니다.
krbtgt/EAST.EXAMPLE.COM@WEST.EXAMPLE.COM	이 주체는 영역 간 TGT(티켓 부여 티켓)의 한 예입니다.
nfs/boston.example.com@EXAMPLE.COM	NFS 서비스에서 사용하는 주체입니다. 이 주체는 host 주체 대신 사용할 수 있습니다.
root/boston.example.com@EXAMPLE.COM	클라이언트의 root 계정과 연관된 주체입니다. 이 주체는 root 주체라고 하며 NFS 마운트 파일 시스템에 대한 root 액세스 권한을 제공합니다.
username@EXAMPLE.COM	사용자 주체입니다.
username/admin@EXAMPLE.COM	KDC 데이터베이스를 관리하는 데 사용할 수 있는 admin 주체입니다.

Kerberos 인증 시스템의 작동 방식

ID를 제공하는 티켓 및 일치하는 세션 키를 제공할 경우 응용 프로그램을 통해 원격 시스템에 로그인할 수 있습니다. 세션 키는 해당 사용자에 대한 정보와 액세스하려는 서비스를 포함하고 있습니다. 티켓과 세션 키는 사용자가 처음으로 로그인할 때 KDC에서 모든 사용자에 대해 생성됩니다. 티켓 및 일치하는 세션 키가 자격 증명을 구성합니다. 여러 네트워크 서비스를 사용 중인 경우 사용자가 여러 자격 증명을 수집할 수 있습니다. 그러나 특정 서버에서 실행되는 자격 증명은 서버당 하나만 있어야 합니다. 예를 들어 `boston`이라는 서버에서 `ftp` 서비스에 액세스하는 데는 하나의 자격 증명이 필요합니다. 다른 서버에서 `ftp` 서비스에 액세스하려면 해당 서버의 자격 증명이 필요합니다.

자격 증명을 만들고 저장하는 프로세스는 투명합니다. 자격 증명은 KDC에서 만들어 요청자에게 전송합니다. 수신된 자격 증명은 자격 증명 캐시에 저장됩니다.

Kerberos 서비스가 DNS 및 nsswitch 서비스와 상호 작용하는 방식

Kerberos 서비스는 DNS를 사용하여 호스트 이름을 분석하도록 컴파일됩니다. 호스트 이름 분석이 수행될 때 `nsswitch` 서비스는 검사하지 않습니다.

Kerberos를 사용하여 서비스에 대한 액세스 권한 얻기

특정 서버에서 특정 서비스에 액세스하려면 사용자가 두 개의 자격 증명을 얻어야 합니다. 첫번째 자격 증명은 TGT(티켓 부여 티켓)에 대한 자격 증명입니다. TGS(티켓 부여 서비스)가 이 자격 증명을 해독하면 사용자가 액세스를 요청하는 서버에 대해 또 다른 자격 증명이 생성됩니다. 그러면 이 두번째 자격 증명을 사용하여 서버의 서비스에 대한 액세스를 요청할 수 있습니다. 서버가 두번째 자격 증명을 성공적으로 해독하면 사용자에게 액세스 권한이 부여됩니다. 다음 절에서 이 프로세스에 대해 더 자세히 설명합니다.

TGS(티켓 부여 서비스)에 대한 자격 증명 얻기

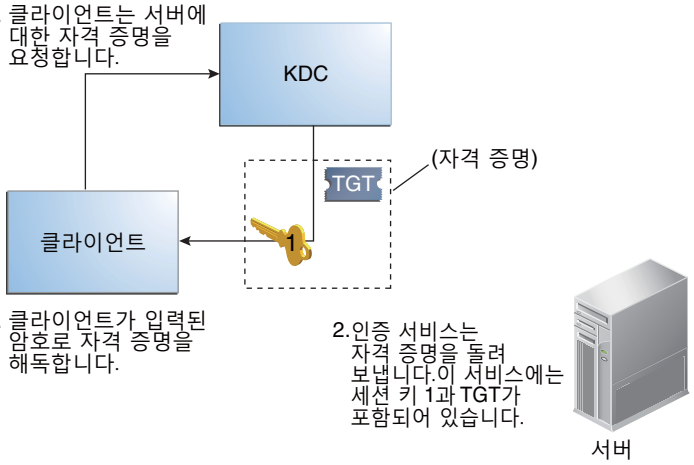
1. 클라이언트가 특정 사용자 주체에 대한 인증 서버로 요청을 보내 인증 프로세스를 시작합니다. 이 요청은 암호화 없이 전송됩니다. 요청에는 보안 정보가 포함되어 있지 않으므로 암호화를 사용할 필요가 없습니다.
2. 인증 서비스에서 요청을 받으면 사용자의 주체 이름을 KDC 데이터베이스에서 조회합니다. 주체가 데이터베이스의 항목과 일치할 경우 인증 서비스가 해당 주체에 대한 개인 키를 얻습니다. 그런 다음 클라이언트 및 TGS(티켓 부여 서비스)에서 사용할 세션 키(세션 키 1) 및 TGS(티켓 부여 서비스)에 대한 티켓(티켓 1)을

생성합니다. 이 티켓을 TGT(**티켓 부여 티켓**)라고도 합니다. 세션 키와 티켓은 사용자의 개인 키를 사용하여 암호화되며, 정보가 다시 클라이언트로 전송됩니다.

3. 클라이언트가 이 정보를 사용하여 세션 키 1 및 티켓 1을 해독합니다. 사용자 주체의 경우 개인 키를 사용합니다. 개인 키는 해당 사용자 및 KDC 데이터베이스만 알고 있어야 하므로 패킷 내의 정보가 안전해야 합니다. 클라이언트에서는 자격 증명 캐시에 정보를 저장합니다.

이 프로세스 중 일반적으로 사용자에게 암호를 입력하라는 메시지가 표시됩니다. 사용자가 지정한 암호와 KDC 데이터베이스에 저장된 개인 키를 작성하는 데 사용된 암호가 같을 경우, 클라이언트에서는 인증 서비스가 보낸 정보를 성공적으로 해독할 수 있습니다. 이제 TGS(티켓 부여 서비스)에 사용할 자격 증명이 클라이언트에 생겨 클라이언트가 서버에 대한 자격 증명을 요청할 수 있습니다.

그림 25–2 TGS(티켓 부여 서비스)에 대한 자격 증명 얻기



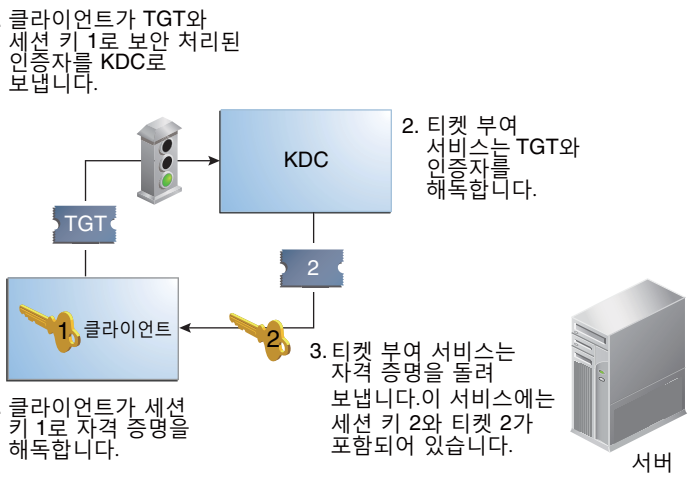
TGT = 티켓 부여 티켓(Ticket-granting ticket)
KDC = 키 배포 센터(Key Distribution Center)

서버에 대한 자격 증명 얻기

1. 특정 서버에 대한 액세스를 요청하려면 먼저 클라이언트가 해당 서버에 대한 자격 증명을 인증 서비스로부터 얻어야 합니다. 510 페이지 “TGS(티켓 부여 서비스)에 대한 자격 증명 얻기”를 참조하십시오. 그런 다음 요청을 TGS(티켓 부여 서비스)로 보냅니다. 이 서비스에는 서비스 주체 이름(티켓 1)과 세션 키 1로 암호화된 인증자가 포함되어 있습니다. 티켓 1은 원래 TGS(티켓 부여 서비스)의 서비스 키를 사용하여 인증 서비스에 의해 암호화되었습니다.

2. TGS(티켓 부여 서비스)의 서비스 키는 TGS(티켓 부여 서비스)에서 알고 있기 때문에 티켓 1을 해독할 수 있습니다. 티켓 1의 정보에는 세션 키 1이 포함되어 있으므로, TGS(티켓 부여 서비스)는 인증자를 해독할 수 있습니다. 이때 사용자 주체는 TGS(티켓 부여 서비스)를 사용하여 인증됩니다.
3. 성공적으로 인증되면 TGS(티켓 부여 서비스)가 사용자 주체 및 서버에 대한 세션 키(세션 키 2) 및 서버에 대한 티켓(티켓 2)을 생성합니다. 그런 다음 세션 키 2와 티켓 2가 세션 키 1을 사용하여 암호화됩니다. 세션 키 1은 클라이언트 및 TGS(티켓 부여 서비스)만 알고 있기 때문에 이 정보는 안전하며 네트워크를 통해 안전하게 전송할 수 있습니다.
4. 클라이언트가 이 정보 패킷을 받으면 세션 키 1을 사용하여 정보를 해독합니다. 세션 키 1은 자격 증명 캐시에 저장되어 있습니다. 클라이언트가 서버에서 사용할 자격 증명을 얻었습니다. 이제 클라이언트가 해당 서버의 특정 서비스에 대한 액세스를 요청할 수 있습니다.

그림 25–3 서버에 대한 자격 증명 얻기



TGT = 티켓 부여 티켓(Ticket-granting ticket)
KDC = 키 배포 센터(Key Distribution Center)

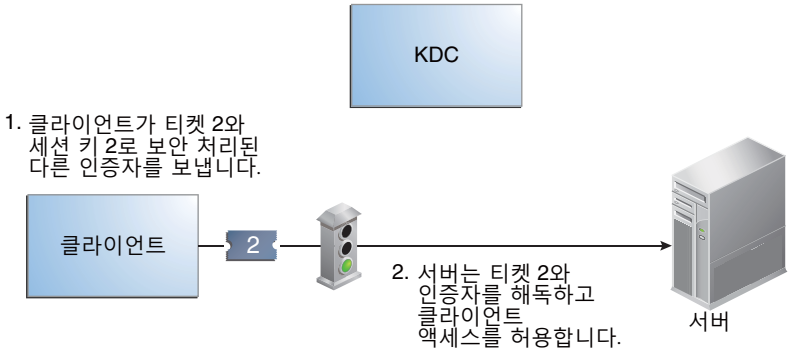
특정 서비스에 대한 액세스 권한 얻기

1. 특정 서비스에 대한 액세스를 요청하려면 먼저 클라이언트가 TGS(티켓 부여 서비스)에 대한 자격 증명을 인증 서버로부터 얻고 서버 자격 증명을 TGS(티켓 부여 서비스)로부터 얻어야 합니다. 510 페이지 “TGS(티켓 부여 서비스)에 대한 자격 증명

얻기" 및 511 페이지 "서버에 대한 자격 증명 얻기"를 참조하십시오. 그러면 티켓 2 및 다른 인증자를 포함하는 서버로 요청을 보낼 수 있습니다. 인증자는 세션 키 2를 사용하여 암호화됩니다.

2. 티켓 2는 TGS(티켓 부여 서비스)의 서비스 키를 사용하여 TGS(티켓 부여 서비스)에 의해 암호화되었습니다. 서비스 키는 서비스 주체만 알고 있으므로 서비스가 티켓 2를 해독하고 세션 키 2를 가져올 수 있습니다. 그런 다음 세션 키 2를 사용하여 인증자를 해독할 수 있습니다. 인증자를 성공적으로 해독하면 서비스에 대한 액세스 권한이 클라이언트에 부여됩니다.

그림 25–4 특정 서비스에 대한 액세스 권한 얻기



Kerberos 암호화 유형 사용

암호화 유형으로 암호화 작업을 수행할 때 사용할 암호화 알고리즘과 모드를 식별할 수 있습니다. `aes`, `des3-cbc-sha1` 및 `rc4–hmac` 암호화 유형으로는 더 높은 강도의 암호화 작업에 사용할 수 있는 키를 만들 수 있습니다. 이처럼 높은 강도의 작업을 수행하면 Kerberos 서비스의 전반적인 보안이 향상됩니다.

주 – Solaris 10 8/07 이전 릴리스에서는 번들화되지 않은 강력한 암호화 패키지가 설치된 경우에 `aes256-cts-hmac-sha1-96` 암호화 유형을 Kerberos 서비스에 사용할 수 있습니다.

클라이언트가 KDC로부터 티켓을 요청할 경우 KDC는 클라이언트 및 서버 모두와 호환되는 암호화 유형을 사용하는 키를 사용해야 합니다. Kerberos 프로토콜은 KDC가 티켓 응답의 클라이언트 부분에 특정 암호화 유형을 사용하도록 클라이언트에 요청할 수는 있지만, 서버에서 암호화 유형을 KDC로 지정하도록 허용하지 않습니다.

주 – 설치된 마스터 KDC에서 Solaris 10 릴리스가 실행 중이지 않은 경우 마스터 KDC를 업그레이드하기 전에 슬레이브 KDC를 Solaris 10 릴리스로 업그레이드해야 합니다. Solaris 10 마스터 KDC는 이전 슬레이브에서 처리할 수 없는 새 암호화 유형을 사용하게 됩니다.

다음 목록은 암호화 유형을 변경하기 전에 고려해야 하는 몇 가지 문제에 대해 설명합니다.

- KDC는 주체 데이터베이스에서 서버 주체 항목과 연관된 첫번째 키/암호화 유형을 서버에서 지원한다고 가정합니다.
- KDC에서는 주체에 대해 생성된 키가 주체가 인증될 시스템과 호환되는지 확인해야 합니다. 기본적으로 `kadmin` 명령은 지원되는 모든 암호화 유형에 대한 키를 만듭니다. 주체가 사용되는 시스템에서 암호화 유형의 이 기본 집합을 지원하지 않을 경우 주체를 생성할 때 암호화 유형을 제한해야 합니다. 암호화 유형을 제한하려면 `kadmin addprinc`에서 `-e` 플래그를 사용하거나, `kdc.conf` 파일에서 `supported_enctypes` 매개변수를 이 일부로 설정하십시오. `supported_enctypes` 매개변수는 Kerberos 영역에 있는 대부분의 시스템이 기본 암호화 유형 집합의 일부를 지원하는 경우에 사용해야 합니다. `supported_enctypes`를 설정하면 특정 영역에 대한 주체를 만들 때 암호화 유형의 기본 집합인 `kadmin addprinc`이 사용되도록 지정됩니다. 일반적으로 다음 두 방법 중 하나를 사용하여 Kerberos에서 사용하는 암호화 유형을 제어하는 것이 가장 좋습니다.
- 시스템에서 지원하는 암호화 유형을 결정할 때 시스템에서 실행 중인 Kerberos의 버전과 서버 주체를 작성할 서버 응용 프로그램에서 지원하는 암호화 알고리즘을 고려해야 합니다. 예를 들어 `nfs/hostname` 서비스 주체를 만들 때 암호화 유형을 해당 호스트의 NFS 서버에서 지원하는 유형으로 제한해야 합니다. Solaris 10 릴리스에서 지원되는 모든 암호화 유형은 NFS 서버에서도 지원됩니다.
- `kdc.conf` 파일의 `master_key_enctype` 매개변수를 사용하면 주체 데이터베이스의 항목을 암호화하는 마스터 키의 암호화 유형을 제어할 수 있습니다. KDC 주체 데이터베이스가 이미 생성된 경우에는 이 매개변수를 사용하지 마십시오. `master_key_enctype` 매개변수를 데이터베이스 생성 시에 사용하여 기본 마스터 키의 암호화 유형을 `des-cbc-crc`에서 더 강력한 암호화 유형으로 변경할 수 있습니다. 모든 슬레이브 KDC가 선택한 암호화 유형을 지원하며, 슬레이브 KDC를 구성할 때 `kdc.conf`에 동일한 `master_key_enctype` 항목이 있는지 확인하십시오. `kdc.conf`에서 `supported_enctypes`가 설정된 경우, `master_key_enctype`이 `supported_enctypes`의 암호화 유형 중 하나로 설정되었는지도 확인하십시오. 이러한 문제를 제대로 처리하지 않으면 마스터 KDC가 슬레이브 KDC와 함께 작동하지 않을 수 있습니다.
- 클라이언트의 경우, `krb5.conf`의 두 매개변수를 통해 KDC에서 티켓을 가져올 때 클라이언트가 요청하는 암호화 유형을 제어할 수 있습니다. `default_tkt_enctypes` 매개변수는 클라이언트가 KDC로부터 TGT(티켓 부여 티켓)를 요청할 때 사용할 암호화 유형을 지정합니다. TGT는 클라이언트가 더 효율적인 방식으로 다른 서버 티켓을 획득하는 데 사용됩니다. `default_tkt_enctypes`를 설정하면 클라이언트가 TGT를 사용하여 서버 티켓을 요청할 때(이를 TGS 요청이라고 함) 클라이언트와 KDC

간의 통신을 보호하는 데 사용되는 암호화 유형을 클라이언트에서 어느 정도 제어할 수 있습니다. `default_tkt_enctypes`에 지정된 암호화 유형은 KDC에 저장된 주체 데이터베이스의 주체 키 암호화 유형 중에서 하나라도 일치해야 합니다. 그렇지 않을 경우 TGT 요청이 실패합니다. `default_tkt_enctypes` 매개변수는 상호 운용성 문제의 원인이 될 수 있으므로 대부분의 경우 이 매개변수를 설정하지 않는 것이 좋습니다. 기본적으로 클라이언트 코드는 지원되는 모든 암호화 유형을 요청하며, KDC는 주체 데이터베이스에서 찾은 키를 기준으로 암호화 유형을 선택합니다.

- `default_tgs_enctypes` 매개변수는 서버 티켓을 얻는 데 사용되는 TGS 요청에서 클라이언트가 요청하는 암호화 유형을 제한합니다. 이 매개변수는 또한 클라이언트와 서버가 공유하는 세션 키를 만들 때 KDC에서 사용하는 암호화 유형도 제한합니다. 예를 들어 보안 NFS를 수행할 때 클라이언트가 3DES 암호화만 사용하려고 하는 경우 `default_tgs_enctypes`를 `des3-cbc-sha1`으로 설정해야 합니다. 클라이언트 및 서버 주체의 주체 데이터베이스에 `des-3-cbc-sha1` 키가 있는지 확인하십시오. `default_tkt_enctype`과 마찬가지로, KDC와 서버에서 자격 증명이 제대로 설정되지 않은 경우 상호 운용성 문제가 발생할 수 있으므로 대부분의 경우 이 매개변수를 설정하지 않는 것이 좋습니다.
- 서버의 경우, `kdc.conf`의 `permitted_enctypes`를 사용하여 서버에서 허용하는 암호화 유형을 제어할 수 있습니다. 또한 `keytab` 항목을 만들 때 사용되는 암호화 유형을 지정할 수 있습니다. 마찬가지로, 일반적으로 암호화 유형을 제어하는 데 이 방법을 사용하지 않는 것이 좋습니다. 대신 KDC는 사용할 키 또는 암호화 유형을 결정할 때 서버 응용 프로그램과 통신하지 않으므로 KDC가 사용할 암호화 유형을 결정하도록 하십시오.

gsscred 테이블 사용

`gsscred` 테이블은 기본 매핑만으로 충분하지 않은 경우 NFS 서버가 Kerberos 사용자를 식별하려고 할 때 사용됩니다. NFS 서비스는 UNIX ID를 사용하여 사용자를 식별합니다. 이러한 ID는 사용자 주체 또는 자격 증명의 일부가 아닙니다. `gsscred` 테이블은 GSS 자격 증명에서 UNIX UID(암호 파일에 있음)로의 추가 매핑을 제공합니다. 이 테이블은 KDC 데이터베이스가 채워진 후에 생성되고 관리되어야 합니다. 자세한 내용은 345 페이지 “UNIX 자격 증명과 GSS 자격 증명 간 매핑”을 참조하십시오.

클라이언트 요청이 수신되면 NFS 서비스는 자격 증명 이름을 UNIX ID에 매핑하려고 합니다. 매핑에 실패하면 `gsscred` 테이블이 검사됩니다.

Oracle Solaris Kerberos 및 MIT Kerberos 간의 주요 차이점

Kerberos 서비스의 Solaris 10 버전은 MIT Kerberos 버전 1.2.1을 기반으로 합니다. 다음 목록은 MIT 1.2.1 버전에는 포함되지 않지만 Solaris 10 릴리스에는 포함된 향상된 기능입니다.

- Oracle Solaris 원격 응용 프로그램의 Kerberos 지원
- KDC 데이터베이스에 대한 증분 전파
- 클라이언트 구성 스크립트
- 지역화된 오류 메시지
- BSM 감사 레코드 지원
- GSS-API를 통한 Kerberos의 스레드 안전 사용
- 암호화에 대해 암호화 프레임워크 사용

이 버전에는 MIT 1.2.1 이후 버그 수정 사항도 포함되어 있습니다. 특히 1.2.5 btree 버그 수정 사항 및 1.3 TCP 지원이 추가되었습니다.

제 7 부

Oracle Solaris에서 감사

이 절에서는 감사 부속 시스템의 구성, 관리 및 사용에 대한 정보를 제공합니다.

◆ ◆ ◆ 26 장

26 감사(개요)

Oracle Solaris의 감사 부속 시스템은 시스템이 어떻게 사용되고 있는지에 대한 기록을 유지합니다. 감사 서비스에는 감사 데이터 분석을 도와주는 도구가 포함됩니다.

이 장에서는 Oracle Solaris에서 감사가 어떻게 작동하는지 소개합니다. 다음은 이 장에 포함된 정보 목록입니다.

계획 제안은 27 장, "감사 계획"을 참조하십시오. 사이트에서 감사 구성 절차는 28 장, "감사 관리(작업)"를 참조하십시오. 참조 정보는 29 장, "감사(참조)"를 참조하십시오.

감사란?

감사는 시스템 리소스 사용에 대한 데이터의 모음입니다. 감사 데이터는 보안 관련 시스템 이벤트의 레코드를 제공합니다. 그러면 이 데이터를 사용하여 호스트에서 발생하는 작업에 대한 책임을 지정할 수 있습니다. 성공적인 감사는 식별 및 인증의 두 가지 보안 기능으로 시작됩니다. 각 로그인 시 사용자가 사용자 이름을 제공하고 PAM 인증을 성공하면 고유한 **감사 사용자** *ID*가 즉각 생성되고 해당 사용자와 연결되며, 고유한 감사 세션 ID가 생성되고 해당 사용자의 프로세스와 연결됩니다. 감사 세션 ID는 해당 로그인 세션 중 시작된 모든 프로세스에서 상속합니다. 사용자가 다른 사용자로 전환할 경우 모든 사용자 작업은 동일한 감사 사용자 ID로 추적됩니다. ID 전환에 대한 자세한 내용은 su(1M) 매뉴얼 페이지를 참조하십시오. 기본적으로 시스템 부트 및 종료와 같은 특정 작업은 항상 감사됩니다.

감사 서비스를 통해 다음 작업이 가능합니다.

- 호스트에서 발생하는 보안 관련 이벤트 모니터링
- 네트워크 전역 감사 추적으로 이벤트 기록
- 잘못된 사용 또는 무단 작업 감지
- 개인 및 객체의 액세스 패턴 및 액세스 내역 검토
- 보호 방식을 우회하려는 시도 감지
- 사용자가 ID를 변경할 때 발생하는 확장된 권한 사용 감지

감사 용어 및 개념

다음 용어는 감사 서비스를 설명하는 데 사용됩니다. 일부 정의에는 좀더 자세한 설명에 대한 포인터가 포함되어 있습니다.

감사 클래스
: 감사 이벤트의 그룹화입니다. 감사 클래스는 감사할 이벤트 그룹을 선택할 수 있는 방법을 제공합니다.

  자세한 내용은 523 페이지 “감사 클래스 및 사전 선택”과 audit_flags(5), audit_class(4) 및 audit_event(4) 매뉴얼 페이지를 참조하십시오.

감사 파일 시스템
: 이진 형식의 감사 파일 저장소입니다.

  자세한 내용은 525 페이지 “감사 로그” 및 audit.log(4) 매뉴얼 페이지를 참조하십시오.

감사 이벤트
: 감사 가능한 보안 관련 시스템 작업입니다. 선택이 용이하도록 이벤트는 감사 클래스로 그룹화됩니다.

  자세한 내용은 522 페이지 “감사 이벤트” 및 audit_event(4) 매뉴얼 페이지를 참조하십시오.

감사 플래그
: 명령 또는 키워드에 대한 인수로 제공되는 감사 클래스입니다. 플래그는 더하기 기호나 빼기 기호를 앞에 붙여 클래스가 성공(+) 또는 실패(-)에 대해 감사되는지 나타낼 수 있습니다. 앞에 캐럿(^)이 있으면 성공이 감사되지 않음(^+) 또는 실패가 감사되지 않음(^-)을 나타냅니다.

  자세한 내용은 audit_flags(5) 매뉴얼 페이지 및 605 페이지 “감사 클래스 구문”을 참조하십시오.

감사 플러그인
: 대기열의 감사 레코드를 지정된 위치로 전송하는 모듈입니다. audit_binfile 플러그인은 이진 감사 파일을 만듭니다. 이진 파일은 감사 파일 시스템에 저장되는 감사 추적을 구성합니다. audit_remote 플러그인은 이진 감사 레코드를 원격 저장소로 보냅니다. audit_syslog 플러그인은 syslog 로그에서 선택한 감사 레코드를 요약합니다.

자세한 내용은 524 페이지 “감사 플러그인 모듈”과 모듈 매뉴얼 페이지 audit_binfile(5), audit_remote(5) 및 audit_syslog(5)를 참조하십시오.

감사 정책 — 사이트에서 사용 또는 사용 안함으로 설정할 수 있는 감사 옵션 집합입니다. 이러한 옵션에는 특정 종류의 감사 데이터를 기록할지 여부가 포함됩니다. 또한 옵션에는 감사 대기열이 가득 찼을 때 감사 가능한 작업을 일시 중지할지 여부도 포함됩니다.

자세한 내용은 538 페이지 “감사 정책 이해” 및 auditconfig(1M) 매뉴얼 페이지를 참조하십시오.

감사 레코드 — 감사 대기열에 수집되는 감사 데이터입니다. 하나의 감사 레코드는 단일 감사 이벤트를 설명합니다. 각 감사 레코드는 감사 토큰으로 구성됩니다.

자세한 내용은 524 페이지 “감사 레코드 및 감사 토큰” 및 audit.log(4) 매뉴얼 페이지를 참조하십시오.

감사 토큰 — 감사 레코드나 이벤트의 필드입니다. 각 감사 토큰은 사용자, 프로그램 또는 기타 객체와 같은 감사 이벤트의 속성의 설명합니다.

자세한 내용은 611 페이지 “감사 토큰 형식” 및 audit.log(4) 매뉴얼 페이지를 참조하십시오.

감사 추적 — 기본 플러그인 audit_binfile을 사용하는 모든 감사된 시스템의 감사 데이터를 저장하는 하나 이상의 감사 파일 모음입니다.

자세한 내용은 609 페이지 “감사 추적”을 참조하십시오.

사후 선택 — 감사 추적에서 검사할 감사 이벤트의 선택입니다. 기본 활성 플러그인 audit_binfile이 감사 추적을 만듭니다. 사후 선택 도구 auditreduce 명령이 감사 추적에서 레코드를 선택합니다.

자세한 내용은 auditreduce(1M) 및 praudit(1M) 매뉴얼 페이지를 참조하십시오.

사전 선택 — 모니터할 감사 클래스의 선택합니다. 사전 선택된 감사 클래스의 감사 이벤트는 감사 대기열에 수집됩니다. 사전 선택되지 않은 감사 클래스는 감사되지 않으므로 해당 이벤트가 대기열에 나타나지 않습니다.

자세한 내용은 523 페이지 “감사 클래스 및 사전 선택”과 audit_flags(5) 및 auditconfig(1M) 매뉴얼 페이지를 참조하십시오.

공용 객체 root 사용자가 소유하고 누구나 읽을 수 있는 파일입니다. 예를 들어, /etc 디렉토리 및 /usr/bin 디렉토리에 있는 파일은 공용 객체입니다. 공용 객체는 읽기 전용 이벤트에 대해 감사되지 않습니다. 예를 들어, file_read(fr) 감사 클래스가 사전 선택되더라도 공용 객체 읽기는 감사되지 않습니다. public 감사 정책 옵션을 변경하여 기본값을 대체할 수 있습니다.

감사 이벤트

감사 이벤트는 시스템에서 감사 가능한 작업을 나타냅니다. 감사 이벤트는 /etc/security/audit_event 파일에 나열됩니다. 각 감사 이벤트는 시스템 호출 또는 사용자 명령에 연결되고, 하나 이상의 감사 클래스에 지정됩니다. audit_event 파일의 형식에 대한 설명은 audit_event(4) 매뉴얼 페이지를 참조하십시오.

예를 들어, AUE_EXECVE 감사 이벤트는 execve() 시스템 호출을 감사합니다. auditrecord -e execve 명령은 이 항목을 표시합니다.

```
execve
  system call execve              See execve(2)
  event ID    23                  AUE_EXECVE
  class       ps,ex               (0x0000000040100000)
      header
      path
      [attribute]                 omitted on error
      [exec_arguments]            output if argv policy is set
      [exec_environment]          output if arge policy is set
      subject
      [use_of_privilege]
      return
```

감사 클래스 ps 또는 감사 클래스 ex를 사전 선택할 경우 모든 execve() 시스템 호출이 감사 대기열에 기록됩니다.

감사는 *attributable* 및 *non-attributable* 이벤트를 처리합니다. 감사 정책은 다음과 같이 이벤트를 *synchronous* 및 *asynchronous* 이벤트로 나눕니다.

- **지정 가능한 이벤트** – 사용자에게 지정할 수 있는 이벤트입니다. execve() 시스템 호출은 사용자에게 지정할 수 있으므로 호출이 지정 가능한 이벤트로 간주됩니다. 모든 지정 가능한 이벤트는 동기 이벤트입니다.
- **지정 불가능한 이벤트** – 커널 인터럽트 레벨에서 발생하거나 사용자가 인증되기 전에 발생하는 이벤트입니다. na 감사 클래스는 지정 불가능한 감사 이벤트를 처리합니다. 예를 들어, 시스템 부트는 지정 불가능한 이벤트입니다. 대부분의 지정 불가능한 이벤트는 비동기 이벤트입니다. 하지만 실패한 로그인과 같이 연결된 프로세스가 있는 지정 불가능한 이벤트는 동기 이벤트입니다.
- **동기 이벤트** – 시스템의 프로세스와 연결된 이벤트입니다. 시스템 이벤트의 대부분은 동기 이벤트입니다.

- **비동기 이벤트** – 프로세스와 연결되지 않아 차단하고 나중에 해제할 수 있는 프로세스가 없는 이벤트입니다. 초기 시스템 부트 및 PROM 진입/종료 이벤트가 비동기 이벤트의 예입니다.

감사 서비스에서 정의한 감사 이벤트 이외에 타사 응용 프로그램에서 감사 이벤트를 생성할 수 있습니다. 감사 이벤트 번호 32768부터 65535까지 타사 응용 프로그램에 대해 사용할 수 있습니다. 공급업체는 Oracle Solaris 담당자에게 연락하여 이벤트 번호를 예약하고 감사 인터페이스에 대한 액세스 권한을 얻어야 합니다.

감사 클래스 및 사전 선택

각 감사 이벤트는 **감사 클래스**에 속합니다. 감사 클래스는 많은 수의 감사 이벤트에 대한 편리한 컨테이너입니다. 감사할 클래스를 **사전 선택**할 경우 해당 클래스의 모든 이벤트가 감사 대기열에 기록됩니다. 예를 들어, `ps` 감사 클래스를 사전 선택하면 `execve()`, `fork ()` 및 기타 시스템 호출이 기록됩니다.

시스템의 이벤트 또는 특정 사용자가 시작한 이벤트에 대해 사전 선택할 수 있습니다.

- **시스템 전역 사전 선택** – `auditconfig` 명령에 `-setflags` 및 `-setnaflags` 옵션을 사용하여 감사에 대한 시스템 전역 기본값을 지정합니다.

  주 – `perzone` 정책이 설정된 경우 모든 영역에서 기본 감사 클래스를 지정할 수 있습니다. `perzone` 감사의 경우 기본값은 시스템 전역이 아닌 영역 전역입니다.

- **사용자 특정 사전 선택** – 사용자에 대한 감사 플래그를 구성하여 개별 사용자에 대해 시스템 전역 감사 기본값과 다르게 지정합니다. `useradd`, `roleadd`, `usermod` 및 `rolemod` 명령은 `user_attr` 데이터베이스에 `audit_flags` 보안 속성을 추가합니다. `profiles` 명령은 `prof_attr` 데이터베이스에 권한 프로파일에 대한 감사 플래그를 추가합니다.

  감사 사전 선택 마스크는 사용자에 대해 감사되는 이벤트의 클래스를 결정합니다. 사용자 사전 선택 마스크에 대한 설명은 608 페이지 “프로세스 감사 특성”을 참조하십시오. 어떻게 구성된 감사 플래그가 사용되는지에 대한 자세한 내용은 199 페이지 “지정된 보안 속성의 검색 순서”를 참조하십시오.

감사 클래스는 `/etc/security/audit_class` 파일에 정의되어 있습니다. 각 항목은 클래스에 대한 감사 마스크, 클래스에 대한 이름 및 클래스에 대한 설명을 포함합니다. 예를 들어, `lo` 및 `ps` 클래스 정의는 `audit_class` 파일에 다음과 같이 나타납니다.

```
0x0000000000001000:lo:login or logout
0x0000000000100000:ps:process start/stop
```

감사 클래스에는 `all` 및 `no`의 두 전역 클래스가 포함됩니다. 감사 클래스에 대한 설명은 `audit_class(4)` 매뉴얼 페이지를 참조하십시오. 클래스 목록은 `/etc/security/audit_class` 파일을 검토하십시오.

클래스에 감사 이벤트 매핑은 구성이 가능합니다. 클래스에서 이벤트를 제거하거나 클래스에 이벤트를 추가하고, 선택한 특정 이벤트에 대해 새 클래스를 만들 수 있습니다. 절차는 558 페이지 "감사 이벤트의 클래스 멤버쉽을 변경하는 방법"을 참조하십시오. 클래스에 매핑된 이벤트를 보려면 `auditrecord -c` *class* 명령을 사용합니다.

감사 레코드 및 감사 토큰

각 **감사 레코드**는 감사된 단일 이벤트의 발생을 기록합니다. 레코드에는 작업을 수행한 사람, 영향을 받는 파일, 시도된 작업, 작업이 발생한 위치 및 시기 등과 같은 정보가 포함됩니다. 다음 예는 `login` 감사 레코드를 보여줍니다.

```
header,69,2,login - local,,example_system,2010-10-10 10:10:10.020 -07:00
subject,jdoe,jdoe,staff,jdoe,staff,1210,4076076536,69 2 example_system
return,success,0
```

각 감사 이벤트에 대해 저장되는 정보의 유형은 **감사 토큰** 집합으로 정의됩니다. 이벤트에 대해 감사 레코드가 만들어질 때마다 레코드에는 해당 이벤트에 대해 정의된 토큰의 일부 또는 모두가 포함됩니다. 이벤트의 특성에 따라 기록되는 토큰이 결정됩니다. 위의 예에서 각 행은 감사 토큰의 이름으로 시작합니다. 감사 토큰의 내용은 토큰 이름 다음에 나옵니다. `header`, `subject` 및 `return` 감사 토큰은 함께 `login - local` 감사 레코드를 구성합니다. 감사 레코드를 구성하는 토큰을 표시하려면 `auditrecord -e` *event* 명령을 사용합니다.

`praudit` 출력의 예와 함께 각 감사 토큰의 구조에 대한 자세한 설명은 611 페이지 "감사 토큰 형식"을 참조하십시오. 감사 토큰의 이진 스트림에 대한 설명은 `audit.log(4)` 매뉴얼 페이지를 참조하십시오.

감사 플러그인 모듈

감사 대기열에서 사전 선택으로 지정한 레코드를 처리할 감사 플러그인 모듈을 지정할 수 있습니다. 적어도 하나의 플러그인은 활성화되어야 합니다. 기본적으로 `audit_binfile` 플러그인이 활성화됩니다. `auditconfig -setplugin` *plugin-name* 명령을 사용하여 플러그인을 구성합니다.

감사 서비스는 다음 플러그인을 제공합니다.

- `audit_binfile` 플러그인 – 이진 감사 파일로 감사 대기열의 전달을 처리합니다. 자세한 내용은 `audit.log(4)` 매뉴얼 페이지를 참조하십시오.
- `audit_remote` 플러그인 – 감사 대기열에서 구성된 원격 서버로 이진 감사 레코드의 보안 전달을 처리합니다. `audit_remote` 플러그인은 `libgss()` 라이브러리를 사용하여 서버를 인증합니다. 전송은 개인 정보 및 무결성을 위해 보호됩니다.
- `audit_syslog` 플러그인 – 감사 대기열에서 `syslog` 로그로 선택된 레코드의 전달을 처리합니다.

플러그인을 구성하려면 auditconfig(1M) 매뉴얼 페이지를 참조하십시오. 플러그인 구성의 예는 559 페이지 “감사 로그 구성(작업)”의 작업을 참조하십시오.

플러그인에 대한 자세한 내용은 audit_binfile(5), audit_remote(5) 및 audit_syslog(5) 매뉴얼 페이지를 참조하십시오.

감사 로그

감사 레코드는 감사 로그에 수집됩니다. 감사 서비스는 감사 레코드에 대한 세 가지 출력 모드를 제공합니다.

- **감사 파일**이라는 로그는 감사 레코드를 이진 형식으로 저장합니다. 시스템 또는 사이트의 감사 파일 집합은 완전한 감사 레코드를 제공합니다. 완전한 감사 레코드를 **감사 추적**이라고 합니다. 이러한 로그는 audit_binfile 플러그인으로 만들어지고, praudit 및 auditreduce 사후 선택 명령으로 검토할 수 있습니다.
- audit_remote 플러그인은 감사 레코드를 원격 저장소로 스트리밍합니다. 저장소에서는 감사 추적을 유지 관리하고 사후 선택 도구를 제공합니다.
- syslog 유틸리티는 감사 레코드의 텍스트 요약을 수집하고 저장합니다. syslog 레코드는 안전하지 않습니다. 다음 예는 login 감사 레코드에 대한 syslog 항목을 보여줍니다.

```
Oct 10  10:10:20 example_system auditd: [ID 6472 audit.notice] \
        login - login ok session 4076172534 by root as root:other
```

사이트에서는 모든 형식으로 감사 레코드를 수집하도록 감사를 구성할 수 있습니다. 이진 모드를 로컬에서 사용하거나 이진 파일을 원격 저장소로 보내거나 syslog 모드를 사용하거나 이러한 모드의 조합을 사용하도록 사이트의 시스템을 구성할 수 있습니다. 다음 표는 이진 감사 레코드를 syslog 감사 레코드와 비교한 것입니다.

표 26–1 이진, 원격 및 syslog 감사 레코드의 비교

기능	이진 및 원격 레코드	syslog 레코드
프로토콜	이진 – 파일 시스템에 기록합니다. 원격 – 원격 저장소로 스트리밍합니다.	원격 로깅을 위해 UDP를 사용합니다.
데이터 유형	이진	텍스트
레코드 길이	제한 없음	감사 레코드당 최대 1024자
위치	이진 – 시스템의 zpool에 저장 원격 – 원격 저장소	syslog.conf 파일에 지정된 위치에 저장

표 26–1 이진, 원격 및 syslog 감사 레코드의 비교 (계속)

기능	이진 및 원격 레코드	syslog 레코드
구성하는 방법	이진 – audit_binfile 플러그인에서 p_dir 속성을 설정합니다.	audit_syslog 플러그인을 활성화하고 syslog.conf 파일을 구성합니다.
	원격 – audit_remote 플러그인에서 p_hosts 속성을 설정하고 플러그인을 활성화합니다.	
읽는 방법	이진 – 일반적으로 배치 모드에서 XML로 브라우저 출력	실시간으로 또는 syslog에 대해 만든 스크립트로 검색
	원격 – 저장소에서 절차 결정	일반 텍스트 출력
완전성	완전성이 보장되며 올바른 순서로 나타남	완전성이 보장되지 않음
시간 기록	협정 세계시(UTC)	감사되는 시스템의 시간

이진 레코드가 가장 뛰어난 보안과 완전성을 제공합니다. 이진 출력은 Common Criteria (http://www.commoncriteriaportal.org/) 감사 요구 사항과 같은 보안 자격 증명 요구 사항을 충족합니다.

audit_binfile 플러그인은 스누핑으로부터 보호되는 파일 시스템에 레코드를 기록합니다. 단일 시스템에서 모든 이진 레코드는 순서대로 수집되고 표시됩니다. 한 감사 추적의 시스템이 여러 시간대에 분포되어 있는 경우 이진 로그의 UTC 시간 기록을 사용하여 정확한 비교가 가능합니다. praudit -x 명령을 사용하여 브라우저에서 XML로 레코드를 볼 수 있습니다. 또한 스크립트를 사용하여 XML 출력을 구문 분석할 수 있습니다.

audit_remote 플러그인은 원격 저장소에 레코드를 기록합니다. 저장소는 저장 및 사후 선택을 처리합니다.

반면, syslog 레코드는 높은 편의성과 유연성을 제공할 수 있습니다. 예를 들어, 다양한 소스에서 syslog 데이터를 수집할 수 있습니다. 또한 syslog.conf 파일에서 audit.notice 이벤트를 모니터할 때 syslog 유틸리티는 현재 시간 기록과 함께 감사 레코드 요약을 기록합니다. 워크스테이션, 서버, 방화벽 및 라우터를 포함한 다양한 소스에서 syslog 메시지에 대해 개발한 동일한 관리 및 분석 도구를 사용할 수 있습니다. 레코드는 실시간으로 보거나 원격 시스템에 저장할 수 있습니다.

syslog.conf를 사용하여 감사 레코드를 원격으로 저장하면 공격자가 로그 데이터를 변경하거나 삭제하지 못하도록 보호할 수 있습니다. 한편, 감사 레코드를 원격으로 저장할 경우 레코드가 서비스 거부 및 소스 주소 스푸핑과 같은 네트워크 공격을 받을 수 있습니다. 또한 UDP는 패킷을 삭제하거나 패킷을 순서 없이 전달할 수 있습니다. syslog 항목은 1024자로 제한되므로 일부 감사 레코드가 로그에서 잘릴 수 있습니다. 단일 시스템에서 일부 감사 레코드가 수집되지 않습니다. 레코드가 순서대로 표시되지 않을 수 있습니다. 각 감사 레코드에는 로컬 시스템의 날짜와 시간이 기록되므로 시간 기록에 의존하여 여러 시스템에 대한 감사 추적을 생성할 수 없습니다.

플러그인 및 감사 로그에 대한 자세한 내용은 다음을 참조하십시오.

- audit_binfile(5) 매뉴얼 페이지
- audit_syslog(5) 매뉴얼 페이지
- audit.log(4) 매뉴얼 페이지
- 563 페이지 "감사 추적에 대한 감사 공간을 지정하는 방법"
- 567 페이지 "syslog 감사 로그를 구성하는 방법"

감사 추적 저장 및 관리

audit_binfile 플러그인이 활성화되면 **감사 파일 시스템**이 감사 파일을 이진 형식으로 보관합니다. 일반적인 설치에서는 /var/audit 파일 시스템을 사용하며 추가 파일 시스템을 사용할 수 있습니다. 모든 감사 파일 시스템의 컨텐츠는 **감사 추적**을 구성합니다. 감사 레코드는 이러한 파일 시스템에 다음 순서대로 저장됩니다.

- **기본 감사 파일 시스템** – /var/audit 파일 시스템이며, 시스템에 대한 감사 파일의 기본 파일 시스템입니다.
- **보조 감사 파일 시스템** – 관리자의 지시에 따라 시스템에 대한 감사 파일이 보관되는 파일 시스템입니다.

파일 시스템은 audit_binfile 플러그인의 p_dir 속성에 인수로 지정됩니다. 목록의 앞에 있는 파일 시스템이 가득 찰 때까지 파일 시스템은 사용되지 않습니다. 파일 시스템 항목 목록의 예는 560 페이지 "감사 파일에 대한 ZFS 파일 시스템을 만드는 방법"을 참조하십시오.

기본 감사 루트 디렉토리에 감사 파일을 두면 감사 추적을 검토할 때 감사 검토자에게 도움이 됩니다. auditreduce 명령은 감사 루트 디렉토리를 사용하여 감사 추적의 모든 파일을 찾습니다. 기본 감사 루트 디렉토리는 /var/audit입니다. auditreduce 명령에 -M 옵션을 사용하여 특정 시스템의 감사 파일을 지정하고, -S 옵션을 사용하여 다른 감사 파일 시스템을 지정할 수 있습니다. 자세한 내용은 auditreduce(1M) 매뉴얼 페이지를 참조하십시오.

감사 서비스는 감사 추적의 파일을 결합하고 필터링하기 위한 명령을 제공합니다. auditreduce 명령은 감사 추적의 감사 파일을 병합할 수 있습니다. 또한 이 명령은 파일을 필터링하여 특정 이벤트를 찾을 수 있습니다. praudit 명령은 이진 파일을 읽습니다. praudit 명령에 대한 옵션은 스크립팅 및 브라우저 표시에 적당한 출력을 제공합니다.

신뢰할 수 있는 시간 기록 유지

여러 시스템의 감사 로그를 병합할 때 이러한 시스템의 날짜와 시간은 정확해야 합니다. 마찬가지로, 감사 로그를 원격 시스템에 보낼 때 기록 시스템과 저장소 시스템의 시계가 정확해야 합니다. NTP(Network Time Protocol)는 시스템 시계를 정확하고 알맞게 유지합니다. 자세한 내용은 **Oracle Solaris 관리: 네트워크 서비스**의 3 장, "시간 관련 서비스" 및 xntpd(1M) 매뉴얼 페이지를 참조하십시오.

원격 저장소 관리

audit_remote 플러그인이 활성화되면 원격 저장소가 감사 레코드를 관리합니다.

감사와 보안의 관련성

감사는 시스템 사용에 대한 의심스럽거나 비정상적인 패턴을 밝혀내어 잠재적인 보안 침입을 감지하는 데 도움을 줍니다. 또한 감사는 의심스런 작업을 특정 사용자로 역추적할 수 있는 방법을 제공하므로 침입을 지연시키는 역할도 수행합니다. 자신의 작업이 감사되고 있다는 사실을 알고 있는 사용자는 악의적인 작업을 덜 시도하게 됩니다.

컴퓨터 시스템, 특히 네트워크에 있는 시스템을 보호하기 위해서는 시스템 프로세스나 사용자 프로세스가 시작되기 전에 작업을 제어하는 방식이 필요합니다. 보안을 위해서는 작업이 발생할 때 작업을 모니터하는 도구가 필요합니다. 또한 보안을 위해서는 작업이 발생한 후 작업 보고서가 필요합니다.

대부분의 감사 작업에는 지정된 매개변수를 충족하는 현재 이벤트 모니터링 및 이벤트 보고가 포함되므로 모범 사례에서는 사용자가 로그인하거나 시스템 프로세스가 시작되기 전에 감사 매개변수를 설정하도록 합니다. 감사 서비스에서 이러한 이벤트를 모니터하고 보고하는 방법에 대한 자세한 내용은 27 장, “감사 계획” 및 28 장, “감사 관리(작업)”를 참조하십시오.

감사는 해커의 무단 침입을 막을 수는 없습니다. 하지만 감사 서비스는 특정 사용자가 특정 작업을 특정 시간과 날짜에 수행했다고 보고할 수 있습니다. 감사 보고서에서는 침입 경로와 사용자 이름으로 사용자를 식별할 수 있습니다. 이러한 정보는 터미널에 즉시 보고되고 나중에 분석을 위해 파일에 저장할 수 있습니다. 따라서 감사 서비스는 다음을 확인하는 데 도움을 주는 데이터를 제공합니다.

- 시스템 보안이 침해된 방법
- 원하는 레벨의 보안 유지를 위해 막아야 하는 보안 허점

감사가 작동하는 방식

감사는 지정된 이벤트가 발생할 때 감사 레코드를 생성합니다. 가장 일반적으로 감사 레코드를 생성하는 이벤트에는 다음이 포함됩니다.

- 시스템 시작 및 시스템 종료
- 로그인 및 로그아웃
- 프로세스 만들기/삭제 또는 스레드 만들기/삭제
- 객체 열기, 닫기, 만들기, 삭제 또는 이름 바꾸기
- 권한 기능 또는 역할 기반 액세스 제어(RBAC) 사용
- 식별 작업 및 인증 작업

- 프로세스 또는 사용자에 의한 권한 변경
- 관리 작업(예: 패키지 설치)
- 사이트 특정 응용 프로그램

감사 레코드는 세 가지 소스에서 생성됩니다.

- 응용 프로그램
- 비동기 감사 이벤트의 결과
- 프로세스 시스템 호출의 결과

관련 이벤트 정보가 캡처된 후 정보는 감사 레코드로 만들어집니다. 각 감사 레코드에는 이벤트를 식별하는 정보, 이벤트의 발생 원인, 이벤트의 시간 및 기타 관련 정보가 포함됩니다. 그러면 이 레코드는 활성 **플러그인**에 대한 감사 대기열에 들어갑니다. 모든 플러그인을 활성화할 수 있지만 적어도 하나의 플러그인은 활성화되어야 합니다.

기본적으로 하나의 플러그인은 활성화됩니다. 이 플러그인은 감사 레코드를 감사 파일에 기록하는 `audit_binfile` 플러그인입니다. 이러한 파일은 이진 형식으로 로컬에 저장됩니다. 활성 `audit_remote` 플러그인은 이러한 레코드를 원격 저장소로 보냅니다. 활성 `audit_syslog` 플러그인은 텍스트 요약을 `syslog` 유틸리티로 보냅니다. 그림은 그림 26–1을 참조하십시오.

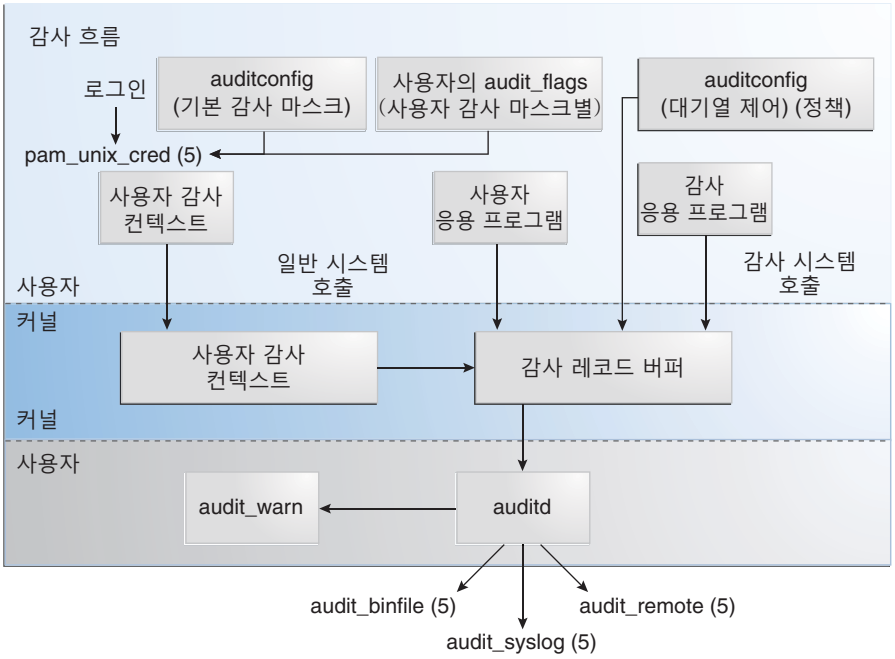
로컬에 저장된 감사 파일은 하나 이상의 ZFS 풀에 있을 수 있습니다. ZFS 풀은 로컬 저장소 관리를 용이하게 할 수 있습니다. 이러한 풀은 서로 다른 시스템 및 서로 다르지만 서로 연결된 네트워크에 있을 수 있습니다. 서로 연결된 감사 파일 모음은 **감사 추적**으로 간주됩니다.

자세한 내용은 529 페이지 “감사를 구성하는 방법”, 525 페이지 “감사 로그” 및 524 페이지 “감사 플러그인 모듈”을 참조하십시오.

감사를 구성하는 방법

시스템 구성 중 모니터할 감사 레코드의 클래스를 **사전 선택**하게 됩니다. 또한 개별 사용자에 대해 수행되는 감사의 정도를 세밀하게 조정할 수 있습니다. 다음 그림은 Oracle Solaris에서 감사의 자세한 흐름을 보여줍니다.

그림 26–1 감사의 흐름



데이터가 커널에서 수집된 후 플러그인은 데이터를 알맞은 위치로 배포합니다.

- `audit_binfile` 플러그인은 이진 감사 레코드를 `/var/audit` 파일 시스템에 둡니다. 사후 선택 도구를 사용하여 감사 추적의 관심 있는 부분을 검사할 수 있습니다.
- `audit_remote` 플러그인은 이진 감사 레코드를 보호된 링크를 통해 원격 저장소로 보냅니다.
- `audit_syslog` 플러그인은 감사 레코드의 텍스트 요약을 `syslog` 유틸리티로 보냅니다.

비전역 영역을 설치하는 시스템은 전역 영역의 모든 영역을 동일하게 감사할 수 있습니다. 또한 비전역 영역에서 서로 다른 레코드를 수집하도록 이러한 시스템을 구성할 수 있습니다. 자세한 내용은 604 페이지 “감사 및 Oracle Solaris 영역”을 참조하십시오.

Oracle Solaris 영역이 있는 시스템에 대한 감사

영역은 단일 인스턴스의 Oracle Solaris OS 내에 만들어지는 가상화 운영 체제 환경입니다. 감사 서비스는 영역에서의 작업을 포함한 전체 시스템을 감사합니다. 비전역 영역을 설치한 시스템은 단일 감사 서비스를 실행하여 모든 영역을 동일하게 감사할 수 있습니다. 또는 전역 영역을 포함하여 영역당 하나의 감사 서비스를 실행할 수 있습니다.

다음 조건을 충족하는 사이트는 단일 감사 서비스를 실행할 수 있습니다.

- 사이트에 단일 이미지 감사 추적이 필요합니다.
- 비전역 영역이 응용 프로그램 컨테이너로 사용됩니다. 영역이 관리 도메인의 일부입니다. 즉, 사용자 정의된 이름 지정 서비스 파일이 있는 비전역 영역이 없습니다.

  시스템의 모든 영역이 하나의 관리 도메인 내에 있는 경우 `zonename` 감사 정책을 사용하여 서로 다른 영역에서 구성된 감사 이벤트를 구별할 수 있습니다.
- 관리자가 낮은 감사 오버헤드를 원합니다. 전역 영역 관리자가 모든 영역을 동일하게 감사합니다. 또한 전역 영역의 감사 데몬이 시스템의 모든 영역을 서비스합니다.

다음 조건을 충족하는 사이트는 영역당 하나의 감사 서비스를 실행할 수 있습니다.

- 사이트에 단일 이미지 감사 추적이 필요하지 않습니다.
- 비전역 영역에 사용자 정의된 이름 지정 서비스 파일이 있습니다. 이러한 별도의 관리 도메인은 대개 서버로 작동합니다.
- 개별 영역 관리자가 자신이 관리하는 영역의 감사를 제어하고자 합니다. 영역별 감사에서 영역 관리자는 자신이 관리하는 영역에 대한 감사를 사용 또는 사용 안함으로 설정할지 결정할 수 있습니다.

영역별 감사의 장점은 각 영역에 대한 사용자 정의된 감사 추적과 영역을 기준으로 영역에 대한 감사를 사용 안함으로 설정할 수 있는 기능입니다. 이러한 장점은 관리 오버헤드로 약화될 수 있습니다. 각 영역 관리자가 감사를 관리해야 합니다. 각 영역은 고유의 감사 데몬에서 실행되고 고유의 감사 대기열과 감사 로그를 가집니다. 이러한 감사 로그는 관리해야 합니다.

이 릴리스의 감사 서비스 정보

감사에 다음 기능이 도입되었습니다.

- 감사는 서비스입니다. 601 페이지 “감사 서비스”를 참조하십시오.
- 감사 기능은 기본적으로 활성화됩니다.
- 감사 서비스를 사용 안함 또는 사용으로 설정할 때 재부트가 필요하지 않습니다.
- `auditconfig` 명령을 사용하여 감사 정책, 지정 불가능한 플래그, 지정 가능한 플래그, 플러그인 및 대기열 제어를 표시하고 변경합니다. `auditconfig(1M)` 매뉴얼 페이지를 참조하십시오.
- 공용 객체의 감사로 감사 추적에서 잡음이 덜 생성됩니다.
- 비커널 이벤트의 감사는 성능에 영향을 주지 않습니다.
- 기본적으로 `login/logout` 클래스의 이벤트는 시스템 및 `root` 계정에 대해 감사됩니다.

- Oracle Solaris는 audit_binfile, audit_remote 및 audit_syslog의 세 가지 플러그인을 제공합니다. audit_binfile(5), audit_remote(5) 및 audit_syslog(5) 매뉴얼 페이지를 참조하십시오.
- 전역 영역을 감사하지 않고도 비전역 영역을 감사할 수 있습니다. 비전역 영역의 감사에 대한 유일한 요구 사항은 perzone 감사 정책이 전역 영역에서 설정되어야 한다는 것입니다.
- 가능한 감사 클래스의 수가 32개에서 64개로 늘어났습니다. 처음 8개 상위 레벨 비트는 고객에 대해 예약되었습니다.
- 감사에 대한 권한 프로파일이 재구성되었습니다. 604 페이지 "감사 관리를 위한 권한 프로파일"을 참조하십시오.
- audit_flags 보안 속성을 사용하여 시스템 전역 감사와 다른 사용자 감사를 구성합니다. 이 키워드는 useradd, usermod, roleadd 및 rolemod 명령에 대한 인수입니다. audit_flags 값은 user_attr 데이터베이스에 저장됩니다. useradd(1M), usermod(1M), roleadd(1M), rolemod(1M) 및 user_attr(4) 매뉴얼 페이지를 참조하십시오.

  profiles 명령에 대한 always_audit 및 never_audit 키워드는 prof_attr 데이터베이스에서 audit_flags 보안 속성을 업데이트합니다. 자세한 내용은 profiles(1) 매뉴얼 페이지 및 199 페이지 "지정된 보안 속성의 검색 순서"를 참조하십시오.
- 새로운 감사 클래스가 정의되었습니다. ft 감사 클래스에는 파일 전송 감사 이벤트가 포함됩니다. ftp 및 sftp 명령은 이 클래스로 감사되는 이벤트 중에 있습니다. frcp 감사 클래스에는 관리자의 사전 선택 여부에 상관없이 기록되는 감사 이벤트가 포함됩니다. auditrecord -c *classname* 명령은 이러한 새 클래스의 감사 이벤트를 설명합니다.

◆ ◆ ◆ 27 장

27 감사 계획

이 장에서는 Oracle Solaris 설치에 대해 감사 서비스를 사용자 정의하는 방법을 설명합니다. 다음은 이 장에 포함된 계획 정보 목록입니다.

감사에 대한 개요는 26 장, "감사(개요)"를 참조하십시오. 사이트에서 감사 구성 절차는 28 장, "감사 관리(작업)"를 참조하십시오. 참조 정보는 29 장, "감사(참조)"를 참조하십시오.

감사 계획(작업)

감사할 작업의 종류에 대해 선별하고자 합니다. 동시에 유용한 감사 정보를 수집하고자 합니다. 그리고 누구를 감사하고 무엇을 감사할지 신중하게 계획해야 합니다. 기본 `audit_binfile` 플러그인을 사용하는 경우 감사 파일이 빠르게 커지면서 사용 가능한 공간을 채우므로 충분한 디스크 공간을 할당해야 합니다.

다음 작업 맵에서는 디스크 공간 및 기록할 이벤트를 계획하는 데 필요한 주요 작업을 안내합니다.

작업	수행 방법
비전역 영역에 대한 감사 전략을 결정합니다.	534 페이지 "영역에서 감사를 계획하는 방법"
감사 추적에 대한 저장소 공간을 계획합니다.	535 페이지 "감사 레코드의 저장소를 계획하는 방법"
감사할 대상(사용자 및 객체)을 결정합니다.	536 페이지 "감사할 대상(사용자 및 객체)을 계획하는 방법"

▼ 영역에서 감사를 계획하는 방법

시스템에 비전역 영역이 포함되어 있는 경우 전역 영역과 동일하게 영역을 감사하거나 각 비전역 영역에 대한 감사 서비스를 별도로 구성, 사용 및 사용 안함으로 설정할 수 있습니다. 예를 들어, 비전역 영역만 감사하고 전역 영역은 감사하지 않을 수 있습니다.

장단점에 대한 자세한 내용은 530 페이지 “Oracle Solaris 영역이 있는 시스템에 대한 감사”를 참조하십시오.

- **다음 옵션 중 하나를 선택합니다.**

  - **옵션 1 - 모든 영역에 대해 단일 감사 서비스를 구성합니다.**

    모든 영역을 동일하게 감사하면 단일 이미지 감사 추적이 만들어집니다. 단일 이미지 감사 추적은 `audit_binfile` 또는 `audit_remote` 플러그인을 사용할 때 발생하며, 시스템의 모든 영역이 한 관리 도메인의 일부입니다. 그러면 모든 영역의 레코드가 동일한 설정으로 사전 선택되므로 감사 레코드를 쉽게 비교할 수 있습니다.

    이 구성은 모든 영역을 한 시스템의 일부로 취급합니다. 전역 영역은 시스템에서 하나의 감사 서비스만 실행하고 모든 영역에 대한 감사 레코드를 수집합니다. 전역 영역에서만 `audit_class` 및 `audit_event` 파일을 사용자 정의한 다음 이러한 파일을 모든 비전역 영역에 복사합니다.

    a. **모든 영역에 대해 동일한 이름 지정 서비스를 사용합니다.**

    주 – 이름 지정 서비스 파일이 비전역 영역에서 사용자 정의되고 `perzone` 정책이 설정되지 않으면 감사 도구를 신중하게 사용하여 유용한 레코드를 선택해야 합니다. 한 영역의 사용자 ID는 다른 영역에서 동일한 ID를 가진 다른 사용자를 가리킬 수 있습니다.

    ---

    b. **감사 레코드에 영역의 이름이 포함되도록 합니다.**

    영역 이름을 감사 레코드의 일부로 두려면 전역 영역에서 `zonename` 정책을 설정합니다. 그러면 `auditreduce` 명령이 감사 추적에서 영역별로 감사 이벤트를 선택할 수 있습니다. 예는 `auditreduce(1M)` 매뉴얼 페이지를 참조하십시오.

    단일 이미지 감사 추적을 계획하려면 536 페이지 “감사할 대상(사용자 및 객체)을 계획하는 방법”을 참조하십시오. 첫번째 단계부터 시작합니다. 또한 전역 영역 관리자는 535 페이지 “감사 레코드의 저장소를 계획하는 방법”에 설명된 대로 저장소를 마련해 두어야 합니다.

  - **옵션 2 - 영역당 하나의 감사 서비스를 구성합니다.**

    서로 다른 영역에서 서로 다른 이름 지정 서비스 데이터베이스를 사용하거나 영역 관리자가 해당 영역의 감사를 제어하고자 하는 경우 영역별 감사를 구성하려면 선택합니다.

주 – 비전역 영역을 감사하려면 perzone 정책을 설정해야 하지만, 전역 영역에서는 감사 서비스를 사용으로 설정하지 않아도 됩니다. 비전역 영역 감사가 구성되고 해당 감사 서비스가 전역 영역과 별도로 사용 및 사용 안함으로 설정됩니다.

- 영역별 감사를 구성할 경우 전역 영역에서 perzone 감사 정책을 설정합니다. 비전역 영역이 처음으로 부트되기 전에 영역별 감사가 설정된 경우 감사는 영역의 최초 부트부터 시작됩니다. 감사 정책을 설정하려면 571 페이지 "영역별 감사를 구성하는 방법"을 참조하십시오.
- 각 영역 관리자가 영역에 대한 감사를 구성합니다.

  비전역 영역 관리자는 perzone 및 ahlt를 제외한 모든 정책 옵션을 설정할 수 있습니다.
- 각 영역 관리자는 영역의 감사를 사용 또는 사용 안함으로 설정할 수 있습니다.
- 검토 중 발생 영역으로 추적할 수 있는 레코드를 생성하려면 zonename 감사 정책을 설정합니다.

영역별 감사를 계획하려면 536 페이지 "감사할 대상(사용자 및 객체)을 계획하는 방법"을 참조하십시오. 첫번째 단계를 건너뛸 수 있습니다. 또한 audit_binfile 플러그인이 활성화된 경우 각 영역 관리자는 535 페이지 "감사 레코드의 저장소를 계획하는 방법"에 설명된 대로 모든 영역에 대한 저장소를 마련해 두어야 합니다.

▼ 감사 레코드의 저장소를 계획하는 방법

audit_binfile 플러그인은 감사 추적을 만듭니다. 감사 추적에는 전용 파일 공간이 필요합니다. 이 공간은 사용 가능하고 안전해야 합니다. 시스템에서는 초기 저장소에 대해 /var/audit 파일 시스템을 사용합니다. 감사 파일에 대해 추가 감사 파일 시스템을 구성할 수 있습니다. 다음 절차에서는 감사 추적 저장소를 계획할 때 해결해야 하는 문제를 다룹니다.

시작하기 전에 비전역 영역을 구현하는 경우 이 절차를 사용하기 전에 534 페이지 "영역에서 감사를 계획하는 방법"을 완료하십시오.

audit_binfile 플러그인을 사용하는 중입니다.

1 사이트에서 필요한 감사의 양을 결정합니다.

사이트의 보안 요구 사항과 감사 추적용 디스크 공간 가용성의 균형을 맞춥니다.

사이트 보안을 유지하면서 공간 요구 사항을 줄이는 방법과 감사 저장소를 설계하는 방법은 541 페이지 "감사 비용 제어" 및 542 페이지 "효율적으로 감사"를 참조하십시오.

실제 단계는 589 페이지 "생성되는 감사 레코드의 양을 줄이는 방법", 597 페이지 "전용 파일 시스템에서 감사 파일을 압축하는 방법" 및 예 28–28을 참조하십시오.

2 **감사할 시스템을 결정하고 감사 파일 시스템을 구성합니다.**

사용할 모든 파일 시스템 목록을 만듭니다. 구성에 대한 자세한 내용은 527 페이지 "감사 추적 저장 및 관리" 및 auditreduce(1M) 매뉴얼 페이지를 참조하십시오. 감사 파일 시스템을 지정하려면 563 페이지 "감사 추적에 대한 감사 공간을 지정하는 방법"을 참조하십시오.

3 **모든 시스템의 시계를 동기화합니다.**

자세한 내용은 527 페이지 "신뢰할 수 있는 시간 기록 유지"를 참조하십시오.

▼ 감사할 대상(사용자 및 객체)을 계획하는 방법

시작하기 전에 비전역 영역을 구현하는 경우 이 절차를 사용하기 전에 534 페이지 "영역에서 감사를 계획하는 방법"을 검토하십시오.

1 **단일 시스템 이미지 감사 추적을 원하는지 여부를 결정합니다.**

주 – 이 단계는 audit_binfile 플러그인에만 적용됩니다.

단일 관리 도메인 내의 시스템은 단일 시스템 이미지 감사 추적을 만들 수 있습니다. 시스템에서 서로 다른 이름 지정 서비스를 사용하는 경우 단계 2부터 시작합니다. 그런 다음 모든 시스템에 대해 나머지 계획 단계를 완료합니다.

사이트에 대해 단일 시스템 이미지 감사 추적을 만들려면 설치 환경의 모든 시스템이 다음과 같이 구성되어야 합니다.

- 모든 시스템에 대해 동일한 이름 지정 서비스를 사용합니다.

  감사 레코드의 올바른 구현을 위해서는 passwd, group 및 hosts 파일이 일관적이어야 합니다.
- 모든 시스템에서 동일하게 감사 서비스를 구성합니다. 서비스 설정 표시 및 수정에 대한 자세한 내용은 auditconfig(1M) 매뉴얼 페이지를 참조하십시오.
- 모든 시스템에 대해 동일한 audit_warn, audit_event 및 audit_class 파일을 사용합니다.

2 **감사 정책을 결정합니다.**

기본적으로 cnt 정책만 사용으로 설정됩니다.

auditconfig -lspolicy 명령을 사용하여 사용 가능한 정책 옵션에 대한 설명을 봅니다.

- 정책 옵션의 효과는 538 페이지 "감사 정책 이해"를 참조하십시오.
- cnt 정책의 효과는 607 페이지 "비동기 및 동기 이벤트에 대한 감사 정책"을 참조하십시오.
- 감사 정책을 설정하려면 553 페이지 "감사 정책을 변경하는 방법"을 참조하십시오.

3 **이벤트-클래스 매핑의 수정을 원하는지 여부를 결정합니다.**

거의 모든 상황에서 기본 매핑이면 충분합니다. 하지만 새 클래스를 추가하거나 클래스 정의를 변경하거나 특정 시스템 호출의 레코드가 유용하지 않다고 판단되는 경우 이벤트-클래스 매핑을 수정할 수도 있습니다.

예는 558 페이지 “감사 이벤트의 클래스 멤버쉽을 변경하는 방법”을 참조하십시오.

4 **사전 선택할 감사 클래스를 결정합니다.**

감사 클래스를 추가하거나 기본 클래스를 변경하는 가장 좋은 시기는 사용자가 시스템에 로그인하기 전입니다.

`auditconfig` 명령에 `-setflags` 및 `-setnaflags` 옵션을 사용하여 사전 선택하는 감사 클래스는 모든 사용자와 프로세스에 적용됩니다. 성공, 실패 또는 둘 다에 대해 클래스를 사전 선택할 수 있습니다.

감사 클래스 목록은 `/etc/security/audit_class` 파일을 검토하십시오.

5 **시스템 전역 사전 선택에 대한 사용자 수정을 결정합니다.**

일부 사용자가 시스템과 다르게 감사되어야 한다고 판단할 경우 `useradd`, `usermod`, `roleadd` 또는 `rolemod` 명령에 `audit_flags` 보안 속성을 사용합니다. 또한 `profiles` 명령을 사용하여 이 속성을 `prof_attr` 데이터베이스의 권한 프로파일에 추가할 수 있습니다. 사용자 사전 선택 마스크는 명시적인 감사 플래그로 권한 프로파일을 사용하는 사용자에 대해 수정됩니다.

절차는 549 페이지 “사용자의 감사 특성을 구성하는 방법”을 참조하십시오. 적용되는 감사 플래그 값에 대한 자세한 내용은 199 페이지 “지정된 보안 속성의 검색 순서”를 참조하십시오.

6 **`audit_warn` 전자 메일 별칭을 어떻게 관리할지 결정합니다.**

`audit_warn` 스크립트는 감사 시스템에서 관리 주의가 요구되는 상황을 감지할 때마다 실행됩니다. 기본적으로 `audit_warn` 스크립트는 전자 메일을 `audit_warn` 별칭에 보내고 메시지를 콘솔로 보냅니다.

별칭을 설정하려면 556 페이지 “`audit_warn` 전자 메일 별칭을 구성하는 방법”을 참조하십시오.

7 **감사 레코드를 어떤 형식으로 어디에 수집할지 결정합니다.**

세 가지 옵션이 있습니다.

- 기본적으로 이진 감사 레코드를 로컬에 저장합니다. 기본 저장소 디렉토리는 `/var/audit`.입니다 `audit_binfile` 플러그인을 추가로 구성하려면 560 페이지 “감사 파일에 대한 ZFS 파일 시스템을 만드는 방법”을 참조하십시오.
- `audit_remote` 플러그인을 사용하여 이진 감사 레코드를 보호된 원격 저장소로 스트리밍합니다. 파일에 대한 수신자가 있어야 합니다. 절차는 566 페이지 “원격 저장소에 감사 파일을 보내는 방법”을 참조하십시오.

- audit_syslog 플러그인을 사용하여 감사 레코드 요약을 syslog에 보냅니다. 절차는 567 페이지 “syslog 감사 로그를 구성하는 방법”을 참조하십시오.

  이진과 syslog 형식에 대한 비교는 525 페이지 “감사 로그”를 참조하십시오.

8 관리자에게 디스크 공간 축소에 대해 언제 경고할지 결정합니다.

주 – 이 단계는 audit_binfile 플러그인에만 적용됩니다.

감사 파일 시스템의 디스크 공간이 최소 여유 공간 비율 또는 소프트 한계 아래로 떨어지면 감사 서비스는 다음 사용 가능한 감사 디렉토리로 전환합니다. 그런 다음 서비스에서는 소프트 한계를 초과했다는 경고를 보냅니다.

최소 여유 공간 비율을 설정하려면 예 28–17을 참조하십시오.

9 모든 감사 디렉토리가 가득 찰 경우 어떤 작업을 수행할지 결정합니다.

주 – 이 단계는 audit_binfile 플러그인에만 적용됩니다.

기본 구성에서는 audit_binfile 플러그인이 활성화되고 cnt 정책이 설정됩니다. 이 구성에서는 커널 감사 대기열이 가득 차면 시스템이 계속 작동합니다. 시스템에서는 삭제되는 감사 레코드 수를 계산하지만 이벤트를 기록하지 않습니다. 더욱 높은 보안을 위해 cnt 정책을 사용 안함으로 설정하고 ahlt 정책을 사용으로 설정할 수 있습니다. 비동기 이벤트를 감사 대기열에 둘 수 없으면 ahlt 정책은 시스템을 중지시킵니다.

이러한 정책 옵션에 대한 자세한 내용은 607 페이지 “비동기 및 동기 이벤트에 대한 감사 정책”을 참조하십시오. 이러한 정책 옵션을 구성하려면 예 28–6을 참조하십시오.

하지만 audit_binfile 대기열이 가득 차고 다른 활성 플러그인에 대한 대기열이 가득 차지 않으면 커널 대기열이 가득 차지 않은 플러그인에 계속해서 레코드를 보냅니다. audit_binfile 대기열에서 다시 레코드를 수신할 수 있게 되면 감사 서비스가 레코드 보내기를 재개합니다.

주 – 적어도 하나의 플러그인에 대한 대기열이 감사 레코드를 수신하지 않으면 cnt 또는 ahlt 정책이 트리거되지 않습니다.

감사 정책 이해

감사 정책은 로컬 시스템에 대한 감사 레코드의 특성을 결정합니다. auditconfig 명령을 사용하여 이러한 정책을 설정합니다. 자세한 내용은 auditconfig(1M) 매뉴얼 페이지를 참조하십시오.

저장소 요구 사항 및 시스템 처리 수요를 최소화하기 위해 대부분의 감사 정책 옵션은 사용 안함으로 설정됩니다. 이러한 옵션은 감사 서비스의 등록 정보이며 시스템 부트 시 적용되는 정책을 결정합니다. 자세한 내용은 auditconfig(1M) 매뉴얼 페이지를 참조하십시오.

다음 표를 참조하여 사이트의 요구 사항이 하나 이상의 감사 정책 옵션을 사용으로 설정하여 발생하는 추가 오버헤드보다 우선하는지 여부를 결정합니다.

표 27–1 감사 정책 옵션의 효과

Policy Name	설명	정책 옵션을 변경하는 이유
ahlt	이 정책은 비동기 이벤트에만 적용됩니다. 사용 안함으로 설정할 경우, 이 정책은 감사 레코드를 생성하지 않고 이벤트가 완료되도록 허용합니다. 사용으로 설정할 경우, 이 정책은 감사 대기열이 가득 찰 경우 시스템을 중지시킵니다. 감사 대기열을 정리하고 감사 레코드에 사용 가능한 공간을 만들며 재부트하려면 관리자의 개입이 필요합니다. 이 정책은 전역 영역에서만 사용으로 설정할 수 있습니다. 정책은 모든 영역에 영향을 줍니다.	시스템 가용성이 보안보다 중요할 경우에는 사용 안함 옵션을 선택하는 것이 좋습니다. 보안이 가장 중요한 환경에서는 사용 옵션을 선택하는 것이 좋습니다. 자세한 내용은 607 페이지 “비동기 및 동기 이벤트에 대한 감사 정책”을 참조하십시오.
arge	사용 안함으로 설정하면 이 정책은 실행된 프로그램의 환경 변수를 execve 감사 레코드에서 뺍니다. 사용으로 설정하면 이 정책은 실행된 프로그램의 환경 변수를 execve 감사 레코드에 추가합니다. 결과 감사 레코드에는 이 정책이 사용 안함으로 설정될 때 더 자세한 정보가 포함됩니다.	사용 안함 옵션으로 설정하면 사용 옵션보다 적은 정보를 수집합니다. 비교는 591 페이지 “사용자의 모든 명령을 감사하는 방법”을 참조하십시오. 적은 수의 사용자를 감사할 때는 사용 옵션을 선택하는 것이 좋습니다. 또한 ex 감사 클래스의 프로그램에서 사용되고 있는 환경 변수가 의심스러운 경우 이 옵션이 유용합니다.
argv	사용 안함으로 설정하면 이 정책은 실행된 프로그램의 인수를 execve 감사 레코드에서 뺍니다. 사용으로 설정하면 이 정책은 실행된 프로그램의 인수를 execve 감사 레코드에 추가합니다. 결과 감사 레코드에는 이 정책이 사용 안함으로 설정될 때 더 자세한 정보가 포함됩니다.	사용 안함 옵션으로 설정하면 사용 옵션보다 적은 정보를 수집합니다. 비교는 591 페이지 “사용자의 모든 명령을 감사하는 방법”을 참조하십시오. 적은 수의 사용자를 감사할 때는 사용 옵션을 선택하는 것이 좋습니다. 또한 ex 감사 클래스에서 비정상적인 프로그램이 실행되고 있다고 판단되는 경우 이 옵션이 유용합니다.
cnt	사용 안함으로 설정하면 이 정책은 사용자 또는 응용 프로그램 실행을 차단합니다. 감사 대기열이 가득 차서 감사 레코드를 감사 추적에 추가할 수 없으면 차단이 발생합니다. 사용으로 설정하면 이 정책은 감사 레코드를 생성하지 않고 이벤트가 완료되도록 허용합니다. 정책에서는 삭제되는 감사 레코드의 수를 계산합니다.	보안이 가장 중요한 환경에서 사용 안함 옵션을 선택하는 것이 좋습니다. 시스템 가용성이 보안보다 중요할 때 사용 옵션을 선택하는 것이 좋습니다. 자세한 내용은 607 페이지 “비동기 및 동기 이벤트에 대한 감사 정책”을 참조하십시오.

표 27–1 감사 정책 옵션의 효과 *(계속)*

Policy Name	설명	정책 옵션을 변경하는 이유
group	사용 안함으로 설정하면 이 정책은 그룹 목록을 감사 레코드에 추가하지 않습니다. 사용으로 설정하면 이 정책은 그룹 목록을 모든 감사 레코드에 특별한 토큰으로 추가합니다.	사용 안함 옵션은 일반적으로 사이트 보안 요구 사항을 충족합니다. 주체가 어떤 보조 그룹에 속하는지 감사해야 하는 경우 사용 옵션을 선택하는 것이 좋습니다.
path	사용 안함으로 설정하면 이 정책은 시스템 호출 중 사용되는 하나의 경로만 감사 레코드에 기록합니다. 사용으로 설정하면 이 정책은 감사 이벤트와 함께 사용되는 모든 경로를 모든 감사 레코드에 기록합니다.	사용 안함 옵션은 하나의 경로만 감사 레코드에 추가합니다. 사용 옵션은 시스템 호출 중 사용되는 각 파일 이름이나 경로를 감사 레코드에 path 토큰으로 입력합니다.
perzone	사용 안함으로 설정하면 이 정책은 시스템에 대해 단일 감사 구성을 유지합니다. 하나의 감사 서비스가 전역 영역에서 실행됩니다. zonename 감사 토큰이 사전 선택된 경우 특정 영역의 감사 이벤트를 감사 레코드에서 찾을 수 있습니다. 사용으로 설정하면 이 정책은 각 영역에 대해 별도의 감사 구성, 감사 대기열 및 감사 로그를 유지합니다. 감사 서비스가 각 영역에서 실행됩니다. 이 정책은 전역 영역에서만 사용으로 설정할 수 있습니다.	각 영역에 대해 별도의 감사 로그, 대기열 및 데몬을 유지해야 하는 특별한 이유가 있을 경우 사용 안함 옵션이 유용합니다. zonename 감사 토큰으로 감사 레코드를 간단히 검사하여 효과적으로 시스템을 모니터할 수 없는 경우 사용 옵션이 유용합니다.
public	사용 안함으로 설정하면 이 정책은 파일 읽기가 사전 선택되었을 때 공용 객체의 읽기 전용 이벤트를 감사 추적에 추가하지 않습니다. 읽기 전용 이벤트를 포함하는 감사 클래스에는 fr, fa 및 cl이 있습니다. 사용으로 설정하면 이 정책은 해당하는 감사 클래스가 사전 선택된 경우 공용 객체의 모든 읽기 전용 감사 이벤트를 기록합니다.	사용 안함 옵션은 일반적으로 사이트 보안 요구 사항을 충족합니다. 사용 옵션은 거의 유용하지 않습니다.
seq	사용 안함으로 설정하면 이 정책은 시퀀스 번호를 모든 감사 레코드에 추가하지 않습니다. 사용으로 설정하면 이 정책은 시퀀스 번호를 모든 감사 레코드에 추가합니다. sequence 토큰에 시퀀스 번호가 포함됩니다.	감사가 부드럽게 실행되는 경우 사용 안함 옵션이면 충분합니다. cnt 정책이 사용으로 설정된 경우 사용 옵션을 선택하는 것이 좋습니다. seq 정책을 사용하여 데이터가 언제 폐기되었는지 확인할 수 있습니다. 또는 auditstat 명령을 사용하여 삭제된 레코드를 볼 수 있습니다.
trail	사용 안함으로 설정하면 이 정책은 trailer 토큰을 감사 레코드에 추가하지 않습니다. 사용으로 설정하면 이 정책은 trailer 토큰을 모든 감사 레코드에 추가합니다.	사용 안함 옵션은 더 작은 감사 레코드를 만듭니다. 사용 옵션은 trailer 토큰을 사용하여 각 감사 레코드의 끝을 분명하게 표시합니다. trailer 토큰은 종종 sequence 토큰과 함께 사용됩니다. trailer 토큰은 손상된 감사 추적 복구에 도움이 됩니다.

표 27–1 감사 정책 옵션의 효과 (계속)

Policy Name	설명	정책 옵션을 변경하는 이유
zonename	사용 안함으로 설정하면 이 정책은 zonename 토큰을 감사 레코드에 포함시키지 않습니다.	영역별로 감사 동작을 추적할 필요가 없는 경우 사용 안함 옵션이 유용합니다.
	사용으로 설정하면 이 정책은 zonename 토큰을 모든 감사 레코드에 포함시킵니다.	영역에 따라 레코드를 사후 선택하여 영역별로 감사 동작을 격리하고 비교하고자 하는 경우 사용 옵션이 유용합니다.

감사 비용 제어

감사는 시스템 리소스를 소모하므로 기록되는 세부 정보의 정도를 제어해야 합니다. 감사할 대상을 결정할 때 다음 감사 비용을 고려하십시오.

- 처리 시간 증가 비용
- 감사 데이터의 분석 비용

기본 플러그인 audit_binfile을 사용하는 경우 감사 데이터의 저장소 비용도 고려해야 합니다.

감사 데이터의 처리 시간 증가 비용

처리 시간 증가 비용은 감사 비용 중 가장 적은 부분을 차지합니다. 첫번째 이유는 감사는 일반적으로 이미지 처리, 복잡한 계산 등과 같이 프로세서를 많이 사용하는 작업 중에 발생하지 않는다는 것입니다. audit_binfile 플러그인을 사용하는 경우 또 하나의 이유는 감사 관리자가 사후 선택 작업을 감사되는 시스템에서 감사 데이터 분석 전용 시스템으로 이동할 수 있다는 것입니다. 마지막으로 커널 이벤트가 사전 선택되지 않으면 감사 서비스 영향 이외에 커널은 시스템 성능에 대한 별다른 영향을 주지 않습니다.

감사 데이터의 분석 비용

분석 비용은 대개 수집되는 감사 데이터의 양에 비례합니다. 분석 비용에는 감사 레코드를 병합하고 검토하는 데 필요한 시간이 포함됩니다.

audit_binfile 플러그인으로 수집된 레코드의 경우, 비용에는 레코드 및 해당 지원 이름 서비스 데이터베이스를 아카이브하고 레코드를 안전한 장소에 보관하는 데 필요한 시간도 포함됩니다. 지원 데이터베이스에는 groups, hosts 및 passwd가 포함됩니다.

생성하는 레코드가 적을수록 감사 추적을 분석하는 데 필요한 시간이 줄어듭니다. 542 페이지 “감사 데이터의 저장소 비용” 및 542 페이지 “효율적으로 감사” 절에서는 효율적으로 감사하는 방법에 대해 설명합니다. 효율적인 감사는 감사 데이터의 양을 줄이면서 사이트의 보안 목표를 달성할 수 있는 충분한 정보를 제공합니다.

감사 데이터의 저장소 비용

audit_binfile 플러그인을 사용하는 경우 저장소 비용은 감사 비용 중 가장 많은 부분을 차지합니다. 감사 데이터의 양은 다음에 따라 달라집니다.

- 사용자 수
- 시스템 수
- 사용량
- 필요한 추적 가능 및 책임 가능 정도

이러한 요소는 사이트마다 다르므로 공식으로 디스크 공간의 양을 사전에 결정하여 감사 데이터 저장소를 마련해 둘 수 없습니다. 다음 정보에 따라 작업을 수행합니다.

- 감사 클래스 이해

  감사를 구성하기 전에 클래스에 포함된 이벤트의 유형을 이해해야 합니다. 감사 이벤트-클래스 매핑을 변경하여 감사 레코드 수집을 최적화할 수 있습니다.

- 감사 클래스를 현명하게 사전 선택하여 생성되는 레코드의 양을 줄입니다.

  전체 감사(즉, all 클래스 사용)는 디스크 공간을 빠르게 채웁니다. 프로그램 컴파일과 같은 단순한 작업도 큰 감사 파일을 생성할 수 있습니다. 보통 크기의 프로그램이 1분 내에 수천 개의 감사 레코드를 생성할 수 있습니다.

  예를 들어, file_read 감사 클래스 fr을 빼면 감사 양을 크게 줄일 수 있습니다. 실패한 작업에 대해서만 감사하도록 선택하면 종종 감사 양을 줄일 수 있습니다. 예를 들어, 실패한 file_read 작업 -fr에 대해 감사하면 모든 file_read 이벤트에 대해 감사할 때보다 훨씬 적은 수의 레코드를 생성할 수 있습니다.

- audit_binfile 플러그인을 사용하는 경우 효율적인 감사 파일 관리도 중요합니다. 예를 들어, 감사 파일 전용인 ZFS 파일 시스템을 압축할 수 있습니다.
- 사이트에 대한 감사 철학을 개발합니다.

  철학은 합리적인 측정 단위를 기준으로 합니다. 이러한 측정 단위에는 사이트에서 필요한 추적 가능 양과 관리하는 사용자의 유형이 포함됩니다.

효율적으로 감사

다음 기술은 조직의 보안 목표를 달성하면서 더욱 효율적으로 감사하는 데 도움될 수 있습니다.

- 가능한 많은 감사 클래스에 대해 시스템 전역이 아닌 사용자 및 역할에 대한 감사 클래스만 사전 선택합니다.
- 특정 시점에 사용자의 일부만 무작위로 감사합니다.
- audit_binfile 플러그인이 활성화된 경우 파일을 필터링, 병합 및 압축하여 감사에 대한 디스크 저장소 요구 사항을 줄입니다. 파일 아카이브, 이동식 매체로 파일 전송 및 원격으로 파일 저장을 위한 절차를 개발합니다.
- 비정상적인 동작에 대해 감사 데이터를 실시간으로 모니터합니다.

- audit_syslog 플러그인 – syslog 파일에서 감사 레코드를 처리하기 위해 이미 개발한 관리 및 분석 도구를 확장할 수 있습니다.
- audit_binfile 플러그인 – 특정 작업에 대한 감사 추적을 모니터하기 위한 절차를 설정할 수 있습니다. 비정상적인 이벤트 감지 시 특정 사용자나 특정 시스템에 대한 감사 자동 증가를 트리거하는 스크립트를 작성할 수 있습니다.

  예를 들어, 다음을 수행하는 스크립트를 작성할 수 있습니다.

  1. 감사되는 시스템에서 감사 파일 만들기를 모니터합니다.
  2. tail 명령을 사용하여 감사 파일을 처리합니다.

     tail -0f 명령에서 praudit 명령을 통한 출력 파이프는 레코드가 생성될 때 감사 레코드 스트림을 반환할 수 있습니다. 자세한 내용은 tail(1) 매뉴얼 페이지를 참조하십시오.
  3. 비정상적인 메시지 유형 또는 기타 지표에 대해 이 스트림을 분석하고, 감사자에게 분석을 전달합니다.

     또는 스크립트를 사용하여 자동 응답을 트리거할 수 있습니다.
  4. 감사 파일 시스템에서 새로운 not_terminated 감사 파일 출현을 지속적으로 모니터합니다.
  5. 해당 파일이 더 이상 쓰여지지 않는 경우 남아 있는 tail 프로세스를 종료합니다.

◆ ◆ ◆ 28 장

28 감사 관리(작업)

이 장에서는 Oracle Solaris 시스템에서 감사를 구성하고 관리하는 데 유용한 절차를 제공합니다. 그리고 감사 추적을 관리하고 감사 서비스 문제를 해결하기 위한 지침도 포함되어 있습니다. 다음은 이 장에 포함된 정보 목록입니다.

감사 서비스에 대한 개요는 26 장, "감사(개요)"를 참조하십시오. 계획 제안은 27 장, "감사 계획"을 참조하십시오. 참조 정보는 29 장, "감사(참조)"를 참조하십시오.

감사 관리(작업 맵)

다음 작업 맵에서는 감사를 관리하는 데 필요한 주요 작업을 안내합니다. 문제 해결 절만 제외하고 작업은 순서대로 나열됩니다.

작업	설명	수행 방법
1. 감사를 계획합니다.	감사 서비스를 구성하기 전에 결정할 구성 문제가 포함되어 있습니다.	533 페이지 "감사 계획(작업)"
2. 감사를 구성합니다.	사용자 및 시스템에 대해 기록될 감사 이벤트를 설정합니다. 선택적으로 감사 정책, 감사 클래스-이벤트 매핑 및 대기열 제어를 수정합니다.	546 페이지 "감사 서비스 구성(작업 맵)"
	감사 레코드가 저장되는 위치 및 해당 형식을 결정하는 플러그인을 구성합니다.	559 페이지 "감사 로그 구성(작업)"

작업	설명	수행 방법
3. 감사를 사용으로 설정합니다.	감사 서비스를 시작합니다. 감사 서비스를 중지합니다.	572 페이지 “감사 서비스를 사용/사용 안함으로 설정(작업)”
	비전역 영역을 설치한 호스트에서 영역당 하나의 감사 서비스가 실행됩니다. 또는 영역에서 전역 영역 감사 서비스를 실행합니다.	568 페이지 “영역에서 감사 서비스 구성(작업)”
4. 감사 레코드를 관리합니다.	감사 추적에서 감사 데이터를 수집하고 분석합니다.	576 페이지 “로컬 시스템에서 감사 레코드 관리(작업 맵)”
감사 문제를 해결합니다.	감사 서비스 문제를 디버깅하고 해결합니다.	586 페이지 “감사 서비스 문제 해결(작업)”

감사 서비스 구성(작업)

네트워크에서 감사를 사용으로 설정하기 전에 해당 사이트 감사 요구 사항을 충족하도록 기본값을 수정할 수 있습니다. 가장 좋은 방법은 처음 사용자가 로그인하기 전에 가능한 많이 감사 구성을 사용자 정의하는 것입니다.

영역을 구현한 경우 전역 영역에서 모든 영역을 감사하거나 비전역 영역을 개별적으로 감사하도록 선택할 수 있습니다. 개요는 604 페이지 “감사 및 Oracle Solaris 영역”을 참조하십시오. 계획은 534 페이지 “영역에서 감사를 계획하는 방법”.을 참조하십시오 절차는 568 페이지 “영역에서 감사 서비스 구성(작업)”을 참조하십시오.

감사 서비스 구성(작업 맵)

다음 작업 맵에서는 감사 구성을 위한 절차를 안내합니다. 모든 작업은 선택 사항입니다.

작업	설명	수행 방법
감사 기본값을 표시합니다.	감사를 구성하기 전에 기본 정책, 대기열 제어, 플래그 및 플러그인 사용을 표시합니다.	547 페이지 “감사 서비스 기본값을 표시하는 방법”
감사되는 이벤트를 선택합니다.	시스템 전역 감사 클래스를 사전 선택합니다. 이벤트가 지정 가능한 경우 모든 사용자가 이 이벤트에 대해 감사됩니다.	548 페이지 “감사 클래스를 사전 선택하는 방법”
특정 사용자에 대해 감사되는 이벤트를 선택합니다.	시스템 전역 감사 클래스에 대한 사용자별 예외 사항을 설정합니다.	549 페이지 “사용자의 감사 특성을 구성하는 방법”
감사 정책을 지정합니다.	사이트에서 요구하는 추가 감사 데이터를 정의합니다.	553 페이지 “감사 정책을 변경하는 방법”
대기열 제어를 지정합니다.	기본 버퍼 크기, 대기열의 감사 레코드 및 버퍼에 감사 레코드 쓰기 간격을 수정합니다.	555 페이지 “감사 대기열 제어를 변경하는 방법”

작업	설명	수행 방법
audit_warn 전자 메일 별칭을 만듭니다.	감사 서비스에 주의가 필요할 때 전자 메일 경고를 받는 사람을 정의합니다.	556 페이지 "audit_warn 전자 메일 별칭을 구성하는 방법"
감사 로그를 구성합니다.	각 플러그인에 대한 감사 레코드의 위치를 구성합니다.	559 페이지 "감사 로그 구성(작업)"
감사 클래스를 추가합니다.	중요 이벤트를 포함할 새 감사 클래스를 만들어 감사 레코드의 수를 줄입니다.	557 페이지 "감사 클래스를 추가하는 방법"
이벤트-클래스 매핑을 변경합니다.	이벤트-클래스 매핑을 변경하여 감사 레코드의 수를 줄입니다.	558 페이지 "감사 이벤트의 클래스 멤버쉽을 변경하는 방법"

▼ 감사 서비스 기본값을 표시하는 방법

이 절차의 명령은 현재 감사 구성을 표시합니다. 이 절차의 출력은 구성되지 않은 시스템에서 가져온 것입니다.

시작하기 전에 Audit Configuration 또는 Audit Control 권한 프로파일이 지정되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 지정 가능한 이벤트에 대한 사전 선택된 클래스를 표시합니다.

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
```

lo는 login/logout 감사 클래스에 대한 플래그입니다. 마스크 출력의 형식은 (*success*, *failure*)입니다.

3 지정 불가능한 이벤트에 대한 사전 선택된 클래스를 표시합니다.

```
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

주 – 클래스에 지정된 이벤트 및 이에 따라 기록되는 이벤트를 보려면 auditrecord -c *class* 명령을 실행합니다.

---

4 감사 정책을 표시합니다.

```
$ auditconfig -getpolicy
configured audit policies = cnt
active audit policies = cnt
```

active 정책은 현재 정책이지만, 정책 값은 감사 서비스에서 저장하지 않습니다. *configured* 정책은 감사 서비스에서 저장하므로 감사 서비스를 다시 시작하면 정책이 복원됩니다.

5 감사 플러그인에 대한 정보를 표시합니다.

```
$ auditconfig -getplugin
Plugin: audit_binfile (active)
    Attributes: p_dir=/var/audit;p_fsize=0;p_minfree=1;

Plugin: audit_syslog (inactive)
    Attributes: p_flags=;

Plugin: audit_remote (inactive)
    Attributes: p_hosts=;p_retries=3;p_timeout=5;
```

audit_binfile 플러그인은 기본적으로 활성화됩니다.

6 감사 대기열 제어를 표시합니다.

```
$ auditconfig -getqctrl
  no configured audit queue hiwater mark
  no configured audit queue lowater mark
  no configured audit queue buffer size
  no configured audit queue delay
  active audit queue hiwater mark (records) = 100
  active audit queue lowater mark (records) = 10
  active audit queue buffer size (bytes) = 8192
  active audit queue delay (ticks) = 20
```

active 대기열 제어는 커널에서 현재 사용하고 있는 대기열 제어입니다. no configured 문자열은 시스템에서 기본값을 사용 중임을 나타냅니다.

7 기존 사용자에 대해 사전 선택된 감사 클래스를 표시합니다.

사용자를 찾은 다음 각 사용자의 audit_flags 속성 값을 표시합니다.

```
# who
adoe    pts/1        Oct 10 10:20    (:0.0)
adoe    pts/2        Oct 10 10:20    (:0.0)
jdoe    pts/5        Oct 12 12:20    (:0.0)
jdoe    pts/6        Oct 12 12:20    (:0.0)
...
# userattr audit_flags adoe
# userattr audit_flags jdoe
```

기본적으로 사용자는 시스템 전역 설정에 대해서만 감사됩니다.

userattr 명령에 대한 설명은 userattr(1) 매뉴얼 페이지를 참조하십시오. audit_flags 키워드에 대한 설명은 user_attr(4) 매뉴얼 페이지를 참조하십시오.

▼ 감사 클래스를 사전 선택하는 방법

모니터할 이벤트를 포함하는 감사 클래스를 사전 선택합니다. 사전 선택된 클래스에 없는 이벤트는 기록되지 않습니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **현재 사전 선택된 클래스를 결정합니다.**

```
# auditconfig -getflags
...

# auditconfig -getnaflags
,,,
```

출력에 대한 설명은 547 페이지 "감사 서비스 기본값을 표시하는 방법"을 참조하십시오.

3 **지정 가능한 클래스를 사전 선택합니다.**

```
# auditconfig -setflags lo,ps,fw
user default audit flags = ps,lo,fw(0x101002,0x101002)
```

이 명령은 로그인/로그아웃, 프로세스 시작/중지 및 파일 쓰기 클래스에서 이벤트의 성공 및 실패에 대해 감사합니다.

주 – `auditconfig -setflags` 명령은 현재 시스템 기본값에 클래스를 **추가**하지 않습니다. 이 명령은 시스템 기본값을 **바꾸므로** 사전 선택할 모든 클래스를 지정해야 합니다.

---

4 **지정 불가능한 클래스를 사전 선택합니다.**

`na` 클래스에는 대표적으로 PROM, 부트 및 지정 불가능한 마운트 이벤트가 포함됩니다.

```
# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```

`lo` 및 `na`는 `-setnaflags` 옵션에 유일하게 유용한 인수입니다.

주 – `auditconfig -setnaflags` 명령은 시스템 기본값을 **바꾸므로** 사전 선택할 모든 클래스를 지정해야 합니다.

▼ 사용자의 감사 특성을 구성하는 방법

시스템별 기준이 아닌 사용자별 기준으로 클래스를 사전 선택하면 시스템 성능에 대한 감사의 영향을 줄일 수 있는 경우가 있습니다. 또한 시스템과 약간 다르게 특정 사용자를 감사할 수도 있습니다.

각 사용자에 대한 감사 클래스 사전 선택은 `audit_flags` 보안 속성으로 지정됩니다. 608 페이지 "프로세스 감사 특성"에 설명된 대로 이러한 사용자 특정 값은 시스템에 대해 사전 선택된 클래스와 함께 사용자의 감사 마스크를 결정합니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

- **user_attr 또는 prof_attr 데이터베이스에서 감사 플래그를 설정합니다.**

  - **사용자에 대한 감사 플래그를 설정하려면 usermod 명령을 사용합니다.**

    ```
    # usermod -K audit_flags=fw:no jdoe
    ```

    audit_flags 키워드는 *always-audit*:*never-audit*입니다.

    always-audit 이 사용자에 대해 감사되는 감사 클래스를 나열합니다. 시스템 전역 클래스에 대한 수정 앞에는 캐럿(^)이 붙습니다. 시스템 전역 클래스에 추가된 클래스 앞에는 캐럿이 붙지 않습니다.

    never-audit 이러한 감사 이벤트가 시스템 전역으로 감사되더라도 사용자에 대해 감사되지 않는 감사 클래스를 나열합니다. 시스템 전역 클래스에 대한 수정 앞에는 캐럿(^)이 붙습니다.

    여러 감사 클래스를 지정하려면 클래스를 콤마로 구분합니다. 자세한 내용은 audit_flags(5) 매뉴얼 페이지를 참조하십시오.

  - **권한 프로파일에 대한 감사 플래그를 설정하려면 profiles 명령을 사용합니다.**

    ```
    # profiles -p "System Administrator"
    profiles:System Administrator> set name="Audited System Administrator"
    profiles:Audited System Administrator> set always_audit=fw,as
    profiles:Audited System Administrator> end
    profiles:Audited System Administrator> exit
    ```

    Audited System Administrator 권한 프로파일을 사용자나 역할에 지정할 경우 해당 사용자나 역할은 199 페이지 “지정된 보안 속성의 검색 순서”에 설명된 검색 순서에 따라 이러한 플래그에 대해 감사됩니다.

예 28–1 한 사용자에 대해 감사되는 이벤트 변경

이 예에서는 모든 사용자에 대한 감사 사전 선택 마스크가 다음과 같습니다.

```
# auditconfig -getflags
active user default audit flags = ss,lo(0x11000,0x11000)
configured user default audit flags = ss,lo(0x11000,0x11000)
```

관리자를 제외하고 로그인된 사용자가 없습니다.

시스템 리소스에 대한 AUE_PFEXEC 감사 이벤트의 영향을 줄이기 위해 관리자는 이 이벤트를 시스템 레벨에서 감사하지 않습니다. 대신 관리자는 사용자 jdoe에 대해 pf 클래스를 사전 선택합니다. pf 클래스는 예 28–10에서 만들어졌습니다.

```
# usermod -K audit_flags=pf:no jdoe
```

userattr 명령은 추가를 표시합니다.

```
# userattr audit_flags jdoe
pf:no
```

사용자 jdoe가 로그인할 때 jdoe의 감사 사전 선택 마스크는 시스템 기본값과 audit_flags 값의 조합입니다. 289는 jdoe 로그인 셸의 PID입니다.

```
# auditconfig -getpinfo 289
audit id = jdoe(1234)
process preselection mask = ss,pf,lo(0x0100000000000000,0x0100000008011000)
terminal id (maj,min,host) = 242,511,example1(192.168.160.171)
audit session id = 103203403
```

예 28–2 한 사용자에 대한 감사 사전 선택 예외 사항 수정

이 예에서는 모든 사용자에 대한 감사 사전 선택 마스크가 다음과 같습니다.

```
# auditconfig -getflags
active user default audit flags = ss,lo(0x11000,0x11000)
configured user default audit flags = ss,lo(0x11000,0x11000)
```

관리자를 제외하고 로그인된 사용자가 없습니다.

관리자는 jdoe 사용자에 대해 실패한 ss 이벤트를 수집하지 않도록 결정합니다.

```
# usermod -K audit_flags=^-ss:no jdoe
```

userattr 명령은 예외 사항을 표시합니다.

```
# userattr audit_flags jdoe
^-ss:no
```

사용자 jdoe가 로그인할 때 jdoe의 감사 사전 선택 마스크는 시스템 기본값과 audit_flags 값의 조합입니다. 289는 jdoe 로그인 셸의 PID입니다.

```
# auditconfig -getpinfo 289
audit id = jdoe(1234)
process preselection mask = +ss,lo(0x11000,0x1000)
terminal id (maj,min,host) = 242,511,example1(192.168.160.171)
audit session id = 103203403
```

예 28–3 선택한 사용자 감사, 시스템 전역 감사 없음

이 예에서는 4명의 선택된 사용자에 대한 로그인 및 역할 작업이 시스템에서 감사됩니다. 시스템에 대해 사전 선택된 감사 클래스는 없습니다.

먼저 관리자는 모든 시스템 전역 플래그를 제거합니다.

```
# auditconfig -setflags no
user default audit flags = no(0x0,0x0)
```

그런 다음 관리자는 4명의 사용자에 대해 2개의 감사 클래스를 사전 선택합니다. pf 클래스는 예 28–10에서 만들어졌습니다.

```
# usermod -K audit_flags=lo,pf:no jdoe
# usermod -K audit_flags=lo,pf:no kdoe
# usermod -K audit_flags=lo,pf:no pdoe
# usermod -K audit_flags=lo,pf:no zdoe
```

그런 다음 관리자는 root 역할에 대해 pf 클래스를 사전 선택합니다.

```
# userattr audit_flags root
# rolemod -K audit_flags=lo,pf:no root
# userattr audit_flags root
lo,pf:no
```

무단 침입 기록을 계속하기 위해 관리자는 지정 불가능한 로그인의 감사를 변경하지 않습니다.

```
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

예 28–4 사용자의 감사 플래그 제거

다음 예에서는 관리자가 모든 사용자별 감사 플래그를 제거합니다. 현재 로그인된 사용자의 기존 프로세스는 계속 감사됩니다.

관리자는 audit_flags 키워드를 no 값으로 설정하여 usermod 명령을 실행합니다.

```
# usermod -K audit_flags= jdoe
# usermod -K audit_flags= kdoe
# usermod -K audit_flags= ldoe
```

그런 다음 관리자는 제거를 확인합니다.

```
# userattr audit_flags jdoe
# userattr audit_flags kdoe
# userattr audit_flags ldoe
```

예 28–5 사용자 그룹에 대한 권한 프로파일 만들기

관리자는 사이트의 모든 관리 권한 프로파일이 pf 클래스를 명시적으로 감사하도록 하고자 합니다. 지정할 모든 권한 프로파일에 대해 관리자는 감사 플래그가 포함된 LDAP의 사이트별 버전을 만듭니다.

먼저, 관리자는 기존 권한 프로파일을 복제한 다음 이름을 변경하고 감사 플래그를 추가합니다.

```
# profiles -p "Network Wifi Management" -S ldap
profiles: Network Wifi Management> set name="Wifi Management"
profiles: Wifi Management> set desc="Audited wifi management"
profiles: Wifi Management> set audit_always=pf
profiles: Wifi Management> exit
```

사용할 모든 권한 프로파일에 대해 이 절차를 반복한 후 관리자는 Wifi Management 프로파일의 정보를 나열합니다.

```
# profiles -p "Wifi Management" -S ldap info
name=Wifi Management
desc=Audited wifi management
auths=solaris.network.wifi.config
help=RtNetWifiMngmnt.html
always_audit=pf
```

▼ 감사 정책을 변경하는 방법

감사 정책은 로컬 시스템에 대한 감사 레코드의 특성을 결정합니다. 감사 정책을 변경하여 감사된 명령에 대한 자세한 정보를 기록하거나 영역 이름을 모든 레코드에 추가하거나 기타 사이트 보안 요구 사항을 충족할 수 있습니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 현재 감사 정책을 봅니다.

```
$ auditconfig -getpolicy
...
```

출력에 대한 설명은 547 페이지 “감사 서비스 기본값을 표시하는 방법”을 참조하십시오.

3 사용 가능한 감사 옵션을 봅니다.

```
$ auditconfig -lspolicy
policy string    description:
ahlt             halt machine if it can not record an async event
all              all policies for the zone
arge             include exec environment args in audit recs
argv             include exec command line args in audit recs
cnt              when no more space, drop recs and keep a cnt
group            include supplementary groups in audit recs
none             no policies
path             allow multiple paths per event
perzone          use a separate queue and auditd per zone
public           audit public files
seq              include a sequence number in audit recs
trail            include trailer token in audit recs
windata_down     include downgraded window information in audit recs
windata_up       include upgraded window information in audit recs
zonename         include zonename token in audit recs
```

주 – perzone 및 ahlt 정책 옵션은 전역 영역에서만 설정할 수 있습니다. 특정 정책 옵션을 사용하는 장단점은 538 페이지 "감사 정책 이해"를 참조하십시오.

4 선택한 감사 정책 옵션을 사용 또는 사용 안함으로 설정합니다.

```
# auditconfig [ -t ] -setpolicy [prefix]policy[,policy...]
```

`-t` 선택 사항. 임시(또는 **활성**) 정책을 만듭니다. 디버깅 또는 테스트 목적으로 임시 정책을 설정할 수 있습니다.

임시 정책은 감사 서비스가 새로 고쳐질 때까지 또는 정책이 `auditconfig -setpolicy` 명령으로 수정될 때까지 유효합니다.

prefix +의 *prefix* 값은 정책 목록을 현재 정책에 추가합니다. -의 *prefix* 값은 정책 목록을 현재 정책에서 제거합니다. 접두어가 없으면 감사 정책이 재설정됩니다. 이 옵션을 사용하여 현재 감사 정책을 유지할 수 있습니다.

policy 사용으로 설정하거나 사용 안함으로 설정할 정책을 선택합니다.

예 28–6 ahlt 감사 정책 옵션

이 예에서는 cnt 정책이 사용 안함으로 설정되고 ahlt 정책이 사용으로 설정됩니다. 이 구성에서는 감사 대기열이 가득 찰 경우 시스템 사용이 정지되고 비동기 이벤트가 발생합니다. 동기 이벤트가 발생하면 스레드를 만든 프로세스가 멈춥니다. 이 구성은 보안이 가용성보다 중요한 경우 적합합니다. 자세한 내용은 607 페이지 "비동기 및 동기 이벤트에 대한 감사 정책"을 참조하십시오.

```
# auditconfig -setpolicy -cnt
# auditconfig -setpolicy +ahlt
```

ahlt 정책 앞의 더하기 기호(+)는 정책을 현재 정책 설정에 추가합니다. 더하기 기호가 없으면 ahlt 정책이 현재 정책 설정을 바꿉니다.

예 28–7 임시 감사 정책 설정

이 예에서는 감사 서비스가 사용으로 설정되었고 ahlt 감사 정책이 구성되었습니다. 관리자는 trail 감사 정책을 활성 정책(+trail)에 추가하지만, 감사 서비스에서 trail 감사 정책을 영구적으로(-t) 사용하도록 구성하지 않습니다. trail 정책은 손상된 감사 추적을 복구하는 데 유용합니다.

```
$ auditconfig -setpolicy ahlt
$ auditconfig -getpolicy
  configured audit policies = ahlt
  active audit policies = ahlt
$ auditconfig -t -setpolicy +trail
  configured audit policies = ahlt
  active audit policies = ahlt,trail
```

디버깅이 완료되면 관리자는 trail 정책을 사용 안함으로 설정합니다.

```
$ auditconfig -setpolicy -trail
$ auditconfig -getpolicy
  configured audit policies = ahlt
  active audit policies = ahlt
```

audit -s 명령을 실행하여 감사 서비스를 새로 고쳐도 감사 서비스의 다른 임시 값과 함께 이 임시 정책이 제거됩니다. 다른 임시 값의 예는 555 페이지 “감사 대기열 제어를 변경하는 방법”을 참조하십시오.

예 28–8 perzone 감사 정책 설정

이 예에서는 perzone 감사 정책이 전역 영역의 기존 정책에 추가됩니다. perzone 정책 설정은 영구 등록 정보로 저장되므로 perzone 정책은 세션 동안 및 감사 서비스가 다시 시작되어도 유효합니다.

```
$ auditconfig -getpolicy
  configured audit policies = cnt
  active audit policies = cnt
$ auditconfig -setpolicy +perzone
$ auditconfig -getpolicy
  configured audit policies = perzone,cnt
  active audit policies = perzone,cnt
```

▼ 감사 대기열 제어를 변경하는 방법

감사 서비스는 감사 대기열 매개변수에 대한 기본값을 제공합니다. auditconfig 명령을 사용하여 이러한 값을 검사, 변경 및 임시적으로 변경할 수 있습니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2. **감사 대기열 제어의 현재 값을 봅니다.**

   ```
   $ auditconfig -getqctrl
   ...
   ```

   출력에 대한 설명은 547 페이지 “감사 서비스 기본값을 표시하는 방법”을 참조하십시오.

3. **선택한 감사 대기열 제어를 수정합니다.**
   감사 대기열 제어의 예 및 설명은 auditconfig(1M) 매뉴얼 페이지를 참조하십시오.

   - 일부 또는 모든 감사 대기열 제어를 수정하려면 -setqctrl 옵션을 사용합니다.

```
# auditconfig [ -t ] -setqctrl hiwater lowater bufsz interval
```

예를 들어, 다른 제어를 설정하지 않고 *interval* 값을 `10`으로 설정합니다.

```
# auditconfig -setqctrl 0 0 0 10
```

- 특정 감사 대기열 제어를 수정하려면 해당 옵션을 지정합니다. **# auditconfig -setqdelay 10**에서와 같이 `-setqdelay` 옵션은 `-setqctrl 0 0 0 interval`과 동일합니다.

```
# auditconfig [ -t ] -setqhiwater value
# auditconfig [ -t ] -setqlowater value
# auditconfig [ -t ] -setqbufsz value
# auditconfig [ -t ] -setqdelay value
```

예 28–9 감사 대기열 제어를 기본값으로 재설정

관리자는 모든 감사 대기열 제어를 설정한 다음 저장소의 *lowater* 값을 기본값으로 다시 변경합니다.

```
# auditconfig -setqctrl 200 5 10216 10
# auditconfig -setqctrl 200 0 10216 10
configured audit queue hiwater mark (records) = 200
no configured audit queue lowater mark
configured audit queue buffer size (bytes) = 10216
configured audit queue delay (ticks) = 10
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 5
active audit queue buffer size (bytes) = 10216
active audit queue delay (ticks) = 10
```

나중에 관리자는 *lowater* 값을 현재 세션에 대한 기본값으로 설정합니다.

```
# auditconfig -setqlowater 10
# auditconfig -getqlowater
configured audit queue lowater mark (records) = 10
active audit queue lowater mark (records) = 10
```

▼ audit_warn 전자 메일 별칭을 구성하는 방법

`/etc/security/audit_warn` 스크립트는 관리자에게 주의가 필요할 수 있는 감사 이벤트를 알려주는 메일을 생성합니다. 이 스크립트를 사용자 정의하고 `root` 이외의 계정에 메일을 보낼 수 있습니다.

`perzone` 정책이 설정된 경우 비전역 관리자는 비전역 영역에서 `audit_warn` 전자 메일 별칭을 구성해야 합니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

- **audit_warn 전자 메일 별칭을 구성합니다.**

다음 옵션 중 하나를 선택합니다.

- **옵션 1** – audit_warn 스크립트에서 audit_warn 전자 메일 별칭을 다른 전자 메일 계정으로 바꿉니다.

  스크립트의 ADDRESS 행에서 audit_warn 전자 메일 별칭을 다른 주소로 변경합니다.

```
#ADDRESS=audit_warn             # standard alias for audit alerts
ADDRESS=audadmin              # role alias for audit alerts
```

주의 – 새 릴리스의 Oracle Solaris OS로 업그레이드하는 경우 사용자 정의된 audit_warn 파일과 audit_warn.new 파일을 수동으로 병합해야 합니다. 이 새 파일에는 중요 변경 사항이 포함되어 있을 수 있습니다. 업그레이드 시 preserve=renamenew 파일 작업에 대한 설명은 pkg(5) 매뉴얼 페이지를 참조하십시오.

- **옵션 2** – audit_warn 전자 메일을 다른 메일 계정으로 리디렉션합니다.

  이 경우 audit_warn 전자 메일 별칭을 적당한 메일 별칭 파일에 추가합니다. 로컬 /etc/mail/aliases 파일이나 이름 공간의 mail_aliases 데이터베이스에 별칭을 추가할 수 있습니다. root 및 audadmin 전자 메일 계정이 audit_warn 전자 메일 별칭의 구성원으로 추가된 경우 /etc/mail/aliases 항목은 다음과 유사합니다.

```
audit_warn: root,audadmin
```

  그런 다음 newaliases 명령을 실행하여 aliases 파일에 대한 임의 액세스 데이터베이스를 재구축합니다.

```
# newaliases
/etc/mail/aliases: 14 aliases, longest 10 bytes, 156 bytes total
```

▼ 감사 클래스를 추가하는 방법

고유의 감사 클래스를 만들 때 해당 사이트에 대해 감사하고자 하는 감사 이벤트를 추가하면 됩니다.

한 시스템에서 클래스를 추가하는 경우 감사하는 모든 시스템에 변경 사항을 복사합니다. 가장 좋은 방법은 감사 서비스를 사용으로 설정하기 전에 감사 클래스를 만드는 것입니다.

주의 – 새 릴리스의 Oracle Solaris OS로 업그레이드하는 경우 사용자 정의된 audit_class 파일과 audit_class.new 파일을 수동으로 병합해야 합니다. 이 새 파일에는 중요 변경 사항이 포함되어 있을 수 있습니다. 업그레이드 시 preserve=renamenew 파일 작업에 대한 설명은 pkg(5) 매뉴얼 페이지를 참조하십시오.

시작하기 전에 항목은 고유해야 합니다. 사용 가능한 비트를 선택해야 합니다. 고객이 사용 가능한 비트는 /etc/security/audit_class 파일에 설명되어 있습니다.

root 역할을 가진 사용자여야 합니다.

1 (옵션) audit_class 파일의 백업 복사본을 저장합니다.

```
# cp /etc/security/audit_class /etc/security/audit_class.orig
```

2 새 항목을 audit_class 파일에 추가합니다.

각 항목의 형식은 다음과 같습니다.

0x*64bitnumber*:*flag*:*description*

필드에 대한 설명은 audit_class(4) 매뉴얼 페이지를 참조하십시오. 기존 클래스 목록은 /etc/security/audit_class 파일을 참조하십시오.

예 28–10 새 감사 클래스 만들기

이 예에서는 역할에서 실행되는 관리 명령을 포함하는 클래스를 만듭니다. audit_class 파일에 추가된 항목은 다음과 같습니다.

```
0x0100000000000000:pf:profile command
```

항목은 새 pf 감사 클래스를 만듭니다. 예 28–11은 새 감사 클래스를 채웁니다.

일반 오류 audit_class 파일을 사용자 정의한 경우 시스템의 감사 사전 선택 마스크에 대한 사용자 예외 사항이 새 감사 클래스와 일관성이 있는지 확인합니다. audit_flags 값이 audit_class 파일의 일부가 아닌 경우 오류가 발생합니다.

▼ 감사 이벤트의 클래스 멤버쉽을 변경하는 방법

감사 이벤트의 클래스 멤버쉽을 변경하여 기존 감사 클래스의 크기를 줄이거나 이벤트를 고유의 클래스에 추가할 수 있습니다.

주의 – audit_event 파일에서 이벤트를 주석 처리하지 마십시오. 이 파일은 praudit 명령에서 이진 감사 파일을 읽는 데 사용됩니다. 아카이브된 감사 파일은 파일에 나열된 이벤트를 포함할 수 있습니다.

한 시스템에서 감사 이벤트-클래스 매핑을 재구성하는 경우 감사하는 모든 시스템에 변경 사항을 복사합니다. 가장 좋은 방법은 사용자가 로그인하기 전에 이벤트-클래스 매핑을 변경하는 것입니다.

주의 – 새 릴리스의 Oracle Solaris OS로 업그레이드하는 경우 사용자 정의된 `audit_event` 파일과 `audit_event.new` 파일을 수동으로 병합해야 합니다. 이 새 파일에는 중요 변경 사항이 포함되어 있을 수 있습니다. 업그레이드 시 `preserve=renamenew` 파일 작업에 대한 설명은 `pkg(5)` 매뉴얼 페이지를 참조하십시오.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

1 (옵션) `audit_event` 파일의 백업 복사본을 저장합니다.

```
# cp /etc/security/audit_event /etc/security/audit_event.orig
```

2 이벤트의 *class-list*를 변경하여 특정 이벤트가 속한 클래스를 변경합니다.

각 항목의 형식은 다음과 같습니다.

number:*name*:*description*:*class-list*

number 감사 이벤트 ID입니다.

name 감사 이벤트의 이름입니다.

description 일반적으로 감사 레코드 만들기를 트리거하는 시스템 호출 또는 실행 파일입니다.

class-list 콤마로 구분된 감사 클래스 목록입니다.

예 28–11 기존 감사 이벤트를 새 클래스에 매핑

이 예에서는 기존 감사 이벤트를 예 28–10에서 만들어진 새 클래스에 매핑합니다. 기본적으로 `AUE_PFEXEC` 감사 이벤트는 `ps`, `ex`, `ua` 및 `as`의 4개 클래스에 매핑됩니다. 새 클래스를 만들면 관리자는 다른 4개 클래스의 이벤트를 감사하지 않고 `AUE_PFEXEC` 이벤트를 감사할 수 있습니다.

```
# grep pf /etc/security/audit_class
0x0100000000000000:pf:profile command
# vi /etc/security/audit_event
116:AUE_PFEXEC:execve(2) with pfexec enabled:pf
# auditconfig -setflags lo,pf
user default audit flags = pf,lo(0x0100000000001000,0x0100000000001000)
```

감사 로그 구성(작업)

`audit_binfile` 및 `audit_syslog` 의 두 감사 플러그인은 구성 가능한 위치로 감사 로그를 보냅니다. 다음 작업은 이러한 로그를 구성하는 데 유용합니다.

감사 로그 구성(작업 맵)

다음 작업 맵에서는 다양한 플러그인에 대한 감사 로그를 구성하기 위한 절차를 안내합니다. 모든 작업은 선택 사항입니다.

작업	설명	수행 방법
audit_binfile 플러그인에 대한 로컬 저장소를 추가합니다.	감사 파일에 대한 로컬 디스크 공간을 만들고 파일 권한으로 보호합니다.	560 페이지 "감사 파일에 대한 ZFS 파일 시스템을 만드는 방법"
audit_binfile 플러그인에 대한 저장소를 지정합니다.	이진 감사 레코드에 대한 디렉토리를 식별합니다.	563 페이지 "감사 추적에 대한 감사 공간을 지정하는 방법"
audit_remote 플러그인에 대한 저장소를 구성합니다.	보호 방식을 통해 원격 저장소로 감사 레코드를 보낼 수 있습니다.	566 페이지 "원격 저장소에 감사 파일을 보내는 방법"
audit_syslog 플러그인에 대한 저장소를 구성합니다.	감사 이벤트를 텍스트 형식으로 syslog에 스트리밍할 수 있습니다.	567 페이지 "syslog 감사 로그를 구성하는 방법"

▼ 감사 파일에 대한 ZFS 파일 시스템을 만드는 방법

다음 절차에서는 감사 파일에 대한 ZFS 풀과 해당하는 파일 시스템 및 마운트 지점을 만드는 방법을 설명합니다. 기본적으로 /var/audit 파일 시스템에는 audit_binfile 플러그인에 대한 감사 파일이 포함됩니다.

시작하기 전에 ZFS File System Management 및 ZFS Storage Management 권한 프로파일이 지정되어야 합니다. ZFS Storage Management 권한 프로파일을 사용하여 저장소 풀을 만들 수 있습니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 필요한 디스크 공간 크기를 결정합니다.

호스트당 200MB 이상의 디스크 공간을 지정합니다. 하지만 필요한 감사의 양에 따라 디스크 공간 요구 사항이 결정됩니다. 따라서 디스크 공간 요구 사항은 이 그림보다 훨씬 높을 수 있습니다.

주 – 기본 클래스 사전 선택은 lo 클래스의 모든 기록되는 이벤트 인스턴스(로그인, 로그아웃, 역할 지정 등)에 대해 약 80바이트씩 증가하는 파일을 /var/audit에 만듭니다.

---

3 **미러링되는 ZFS 저장소 풀을 만듭니다.**

`zpool create` 명령은 ZFS 파일 시스템에 대한 컨테이너인 저장소 풀을 만듭니다. 자세한 내용은 **Oracle Solaris 관리: ZFS 파일 시스템**의 1 장, "Oracle Solaris ZFS 파일 시스템(소개)"을 참조하십시오.

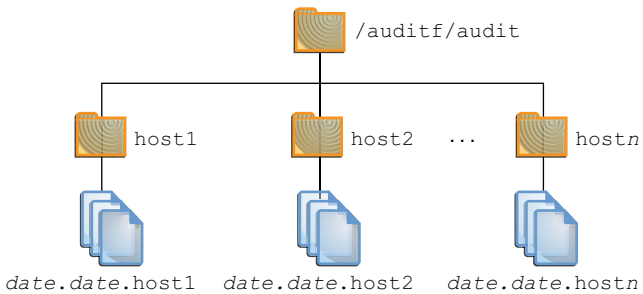
```
# zpool create audit-pool mirror disk1 disk2
```

예를 들어, `c3t1d0` 및 `c3t2d0`의 두 디스크에서 `auditp` 풀을 만들고 미러링합니다.

```
# zpool create auditp mirror c3t1d0 c3t2d0
```

4 **감사 파일에 대한 ZFS 파일 시스템 및 마운트 지점을 만듭니다.**

하나의 명령으로 파일 시스템 및 마운트 지점을 만듭니다. 생성 시 파일 시스템이 마운트됩니다. 예를 들어, 다음 그림은 호스트 이름으로 저장되는 감사 추적 저장소를 보여줍니다.



주 – 파일 시스템을 암호화하려는 경우 생성 시 파일 시스템을 암호화해야 합니다. 예는 예 28–12를 참조하십시오.

암호화에는 관리가 필요합니다. 예를 들어, 마운트 시 암호문이 필요합니다. 자세한 내용은 **Oracle Solaris 관리: ZFS 파일 시스템**의 "ZFS 파일 시스템 암호화"를 참조하십시오.

```
# zfs create -o mountpoint=/mountpoint audit-pool/mountpoint
```

예를 들어, `auditf` 파일 시스템에 대한 `/audit` 마운트 지점을 만듭니다.

```
# zfs create -o mountpoint=/audit auditp/auditf
```

5 **감사 파일에 대한 ZFS 파일 시스템을 만듭니다.**

```
# zfs create -p auditp/auditf/system
```

예를 들어, `sys1` 시스템에 대한 암호화되지 않은 ZFS 파일 시스템을 만듭니다.

```
# zfs create -p auditp/auditf/sys1
```

6 **(옵션) 감사 파일에 대한 추가 파일 시스템을 만듭니다.**

추가 파일 시스템을 만드는 한 가지 이유는 감사 오버플로우를 막기 위함입니다. 단계 9에 나온 대로 파일 시스템당 ZFS 할당량을 설정할 수 있습니다. audit_warn 전자 메일 별칭은 각 할당량에 도달하면 알려줍니다. 공간을 확보하기 위해 닫힌 감사 파일을 원격 서버로 이동할 수 있습니다.

```
# zfs create -p auditp/auditf/sys1.1
# zfs create -p auditp/auditf/sys1.2
```

7 **상위 감사 파일 시스템을 보호합니다.**

다음 ZFS 등록 정보는 풀의 모든 파일 시스템에 대해 off로 설정됩니다.

```
# zfs set devices=off auditp/auditf
# zfs set exec=off auditp/auditf
# zfs set setuid=off auditp/auditf
```

8 **풀의 감사 파일을 압축합니다.**

일반적으로 압축은 ZFS의 파일 시스템 레벨에서 설정됩니다. 하지만 이 풀의 모든 파일 시스템에는 감사 파일이 포함되므로 압축은 풀에 대한 최상위 레벨 데이터 집합에서 설정됩니다.

```
# zfs set compression=on auditp
```

또한 **Oracle Solaris 관리: ZFS 파일 시스템**의 "ZFS 압축, 중복 제거 및 암호화 등록 정보 간의 상호 작용"을 참조하십시오.

9 **할당량을 설정합니다.**

상위 파일 시스템, 종속 파일 시스템 또는 둘 다에서 할당량을 설정할 수 있습니다. 상위 감사 파일 시스템에서 할당량을 설정할 경우 종속 파일 시스템에 대한 할당량을 설정하면 제한이 추가됩니다.

a. **상위 감사 파일 시스템에서 할당량을 설정합니다.**

다음 예에서는 auditp 풀의 두 디스크가 모두 할당량에 도달하면 audit_warn 스크립트가 감사 관리자에게 알려줍니다.

```
# zfs set quota=510G auditp/auditf
```

b. **종속 감사 파일 시스템에서 할당량을 설정합니다.**

다음 예에서는 auditp/auditf/*system* 파일 시스템에 대한 할당량에 도달하면 audit_warn 스크립트가 감사 관리자에게 알려줍니다.

```
# zfs set quota=170G auditp/auditf/sys1
# zfs set quota=170G auditp/auditf/sys1.1
# zfs set quota=165G auditp/auditf/sys1.2
```

10 **대량 풀의 경우 감사 파일의 크기를 제한합니다.**

기본적으로 감사 파일은 풀의 크기까지 커질 수 있습니다. 관리 용이성을 위해 감사 파일의 크기를 제한합니다. 예 28–14를 참조하십시오.

예 28–12 감사 파일에 대한 암호화된 파일 시스템 만들기

사이트 보안 요구 사항을 준수하기 위해 관리자는 암호화를 사용하여 감사 파일 시스템을 만듭니다. 그런 다음 관리자는 마운트 지점을 설정합니다.

```
# zfs create -o encryption=on auditp/auditf
Enter passphrase for auditp/auditf': /** Type 8-character minimum passphrase**/
Enter again: /** Confirm passphrase **/
# zfs set -o mountpoint=/audit auditp/auditf
```

관리자가 auditf 파일 시스템에 추가 파일 시스템을 만들면 이러한 종속 파일 시스템도 암호화됩니다.

예 28–13 /var/audit 디렉토리에서 할당량 설정

이 예에서는 관리자가 기본 감사 파일 시스템에서 할당량을 설정합니다. 이 할당량에 도달하면 audit_warn 스크립트가 감사 관리자에게 경고합니다.

```
# zfs set quota=252G rpool/var/audit
```

▼ 감사 추적에 대한 감사 공간을 지정하는 방법

이 절차에서는 audit_binfile 플러그인에 대한 속성을 사용하여 감사 추적에 추가 디스크 공간을 지정합니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2. **audit_binfile 플러그인에 대한 속성을 결정합니다.**
   audit_binfile(5) 매뉴얼 페이지의 OBJECT ATTRIBUTES 절을 참조하십시오.

```
# man audit_binfile
...
OBJECT ATTRIBUTES
     The p_dir attribute specifies where the audit files will be
     created. The directories are listed in the order in which
     they are to be used.

     The p_minfree attribute defines the percentage of free space
     that the audit system requires before the audit daemon invokes
     the audit_warn script.

     The p_fsize attribute defines the maximum size in bytes that
     an  audit  file can become before it is automatically closed
     and a new audit file  opened.  ...
```

3 **감사 추적에 디렉토리를 추가하려면 p_dir 속성을 지정합니다.**

기본 파일 시스템은 /var/audit입니다.

```
# auditconfig -setplugin audit_binfile active p_dir=/audit/sys1.1,/var/audit
```

위의 명령은 /audit/sys1.1 파일 시스템을 감사 파일에 대한 기본 디렉토리로 설정하고 기본 /var/audit 파일 시스템을 보조 디렉토리로 설정합니다. 이 시나리오에서는 /var/audit가 마지막 의존 디렉토리입니다. 이 구성이 성공하려면 /audit/sys1.1 파일 시스템이 존재해야 합니다.

560 페이지 "감사 파일에 대한 ZFS 파일 시스템을 만드는 방법"에서 유사한 파일 시스템을 만들었습니다.

4 **감사 서비스를 새로 고칩니다.**

auditconfig -setplugin 명령은 **구성된** 값을 설정합니다. 이 값은 감사 서비스의 등록 정보이므로 서비스를 새로 고치거나 다시 시작해도 복원됩니다. 감사 서비스가 새로 고쳐지거나 다시 시작되면 구성된 값이 **활성**이 됩니다. 구성된 값 및 활성 값에 대한 자세한 내용은 auditconfig(1M) 매뉴얼 페이지를 참조하십시오.

```
# audit -s
```

예 28–14 audit_binfile 플러그인에 대한 파일 크기 제한

다음 예에서는 이진 감사 파일의 크기가 특정 크기로 설정됩니다. 크기는 메가바이트로 지정됩니다.

```
# auditconfig -setplugin audit_binfile active p_fsize=4M
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile (active)
    Attributes: p_dir=/var/audit;p_fsize=4M;p_minfree=1;
```

기본적으로 감사 파일은 무제한으로 커질 수 있습니다. 더 작은 감사 파일을 만들기 위해 관리자는 4MB의 파일 크기 제한을 지정합니다. 제한 크기에 도달하면 감사 서비스는 새 파일을 만듭니다. 파일 크기 제한은 관리자가 감사 서비스를 새로 고친 후 적용됩니다.

```
# audit -s
```

예 28–15 감사 플러그인에 여러 변경 사항 지정

다음 예에서는 처리량이 많고 ZFS 풀이 큰 시스템의 관리자가 audit_binfile 플러그인에 대한 대기열 크기, 이진 파일 크기 및 소프트 제한 경고를 변경합니다. 관리자는 감사 파일이 4GB까지 커질 수 있도록 허용하고, ZFS 풀의 2%가 남으면 경고를 받으며, 허용된 할당량 크기를 두 배로 늘립니다. 기본 대기열 크기는 active audit queue hiwater mark (records) = 100과 같이 커널 감사 대기열에 대한 고수위 마크인 100입니다.

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile (active)
    Attributes: p_dir=/var/audit;p_fsize=2G;p_minfree=1;
# auditconfig -setplugin audit_binfile active "p_minfree=2;p_fsize=4G" 200
```

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile (active)
    Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
    Queue size: 200
```

변경된 지정 사항은 관리자가 감사 서비스를 새로 고친 후 적용됩니다.

```
# audit -s
```

예 28–16 감사 플러그인에 대한 대기열 크기 제거

다음 예에서는 audit_binfile 플러그인에 대한 대기열 크기가 제거됩니다.

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile (active)
    Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
    Queue size: 200
# auditconfig -setplugin audit_binfile active "" ""
# auditconfig -getplugin audit_binfile
 Plugin: audit_binfile (active)
    Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
```

마지막 비어 있는 큰 따옴표("")는 플러그인에 대한 대기열 크기를 기본값으로 설정합니다.

플러그인에 대한 qsize 지정 변경 사항은 관리자가 감사 서비스를 새로 고친 후 적용됩니다.

```
# audit -s
```

예 28–17 경고에 대한 소프트 제한 설정

이 예에서는 모든 감사 파일 시스템에 대한 최소 사용 가능 공간 레벨을 설정하여 파일 시스템의 2%를 아직 사용할 수 있을 때 경고를 보냅니다.

```
# auditconfig -setplugin audit_binfile active p_minfree=2
```

기본값은 1%입니다. 대형 ZFS 풀의 경우 적당히 낮은 백분율을 선택합니다. 예를 들어, 16TB 풀의 10%는 16GB이므로 충분한 디스크 공간이 남아 있을 때 감사 관리자에게 경고를 보내게 됩니다. 값이 2이면 약 2GB의 디스크 공간이 남아 있을 때 audit_warn 메시지를 보냅니다.

audit_warn 전자 메일 별칭이 경고를 수신합니다. 별칭을 설정하려면 556 페이지 “audit_warn 전자 메일 별칭을 구성하는 방법”을 참조하십시오.

또한 대형 풀의 경우 관리자는 파일 크기를 3GB로 제한할 수 있습니다.

```
# auditconfig -setplugin audit_binfile active p_fsize=3G
```

플러그인에 대한 p_minfree 및 p_fsize 지정 사항은 관리자가 감사 서비스를 새로 고친 후 적용됩니다.

```
# audit -s
```

▼ 원격 저장소에 감사 파일을 보내는 방법

이 절차에서는 audit_remote 플러그인에 대한 속성을 사용하여 원격 감사 저장소에 감사 추적을 보냅니다.

시작하기 전에 원격 저장소에서 감사 파일의 수신자여야 합니다. Audit Configuration 권한 프로파일이 지정되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 audit_remote 플러그인에 대한 속성을 결정합니다.

audit_remote(5) 매뉴얼 페이지의 OBJECT ATTRIBUTES 절을 참조하십시오.

```
# man audit_remote
...
OBJECT ATTRIBUTES
     The p_hosts attribute specifies the remote servers.
     You can also specify the port number and the GSS-API
     mechanism.

     The p_retries attribute specifies the number of retries for
     connecting and sending data. The default is 3.

     The p_timeout attribute specifies the number of seconds
     in which a connection times out.
```

기본 포트는 solaris_audit IANA 지정 포트인 16162/tcp입니다. 기본 방식은 kerberos_v5입니다. 시간 초과 기본값은 5초입니다. 플러그인에 대한 대기열 크기도 지정할 수 있습니다.

3 원격 호스트를 지정하려면 p_hosts 속성을 사용합니다.

```
# auditconfig -setplugin audit_remote active p_hosts=rhost1:16088:kerberos_v5
```

4 재시도 횟수를 지정하려면 p_retries 속성을 사용합니다.

```
# auditconfig -setplugin audit_remote active p_retries=5
```

5 연결 시간 초과 길이를 지정하려면 p_timeout 속성을 사용합니다.

```
# auditconfig -setplugin audit_remote active p_timeout=3
```

6 감사 서비스를 새로 고칩니다.

감사 서비스는 새로 고쳐질 때 감사 플러그인 변경 사항을 읽습니다.

```
# audit -s
```

▼ syslog 감사 로그를 구성하는 방법

감사 서비스에서 감사 대기열의 감사 레코드 중 일부나 모두를 syslog 유틸리티에 복사하도록 지시할 수 있습니다. 이진 감사 데이터와 텍스트 요약을 모두 기록할 경우 이진 데이터는 완전한 감사 레코드를 제공하고, 요약은 실시간 검토를 위해 데이터를 필터링합니다.

시작하기 전에 audit_syslog 플러그인을 구성하려면 Audit Configuration 권한 프로파일이 지정되어야 합니다. syslog 유틸리티를 구성하려면 root 역할을 가진 사용자여야 합니다.

1 audit_syslog 플러그인에 보낼 감사 클래스를 선택하고 플러그인을 활성으로 만듭니다.

주 – p_flags 감사 클래스는 시스템 기본값으로 또는 사용자나 권한 프로파일의 감사 플래그로 사전 선택되어야 합니다. 사전 선택되지 않은 클래스에 대한 레코드는 수집되지 않습니다.

```
# auditconfig -setplugin audit_syslog active p_flags=lo,+as,-ss
```

2 syslog 유틸리티를 구성합니다.

a. audit.notice 항목을 syslog.conf 파일에 추가합니다.

항목에는 로그 파일의 위치가 포함됩니다.

```
# cat /etc/syslog.conf
...
audit.notice       /var/adm/auditlog
```

b. 로그 파일을 만듭니다.

```
# touch /var/adm/auditlog
```

c. syslog 서비스에 대한 구성 정보를 새로 고칩니다.

```
# svcadm refresh system/system-log
```

3 감사 서비스를 새로 고칩니다.

감사 서비스는 새로 고쳐질 때 감사 플러그인 변경 사항을 읽습니다.

```
# audit -s
```

4 정기적으로 syslog 로그 파일을 아카이브합니다.

감사 서비스는 확장 출력을 생성할 수 있습니다. 로그를 관리하려면 logadm(1M) 매뉴얼 페이지를 참조하십시오.

예 28–18 syslog 출력에 대한 감사 클래스 지정

다음 예에서는 syslog 유틸리티가 사전 선택된 감사 클래스를 일부를 수집합니다. pf 클래스는 예 28–10에서 만들어졌습니다.

```
# auditconfig -setnaflags lo,na
# auditconfig -setflags lo,ss
# usermod -K audit_flags=pf:no jdoe
# auditconfig -setplugin audit_syslog active p_flags=lo,+na,-ss,+pf
```

auditconfig 명령에 대한 인수는 시스템에서 모든 로그인/로그아웃, 지정 불가능 및 시스템 상태 감사 레코드의 변경 사항을 수집하도록 지시합니다. audit_syslog 플러그인 항목은 syslog 유틸리티에서 모든 로그인, 성공한 지정 불가능 이벤트 및 시스템의 상태의 실패한 변경 사항을 수집하도록 지시합니다.

jdoe 사용자의 경우 이진 감사 레코드에는 pfexec 명령 호출의 모든 사용이 포함됩니다. 이러한 이벤트를 사후 선택에 사용할 수 있으려면 audit_binfile 또는 audit_remote 플러그인이 활성화되어야 합니다. syslog 유틸리티는 pfexec 명령에 대해 성공한 호출을 수집합니다.

예 28–19 원격 시스템에 syslog 감사 레코드 두기

syslog.conf 파일의 audit.notice 항목이 원격 시스템을 가리키도록 변경할 수 있습니다. 예를 들어, 로컬 시스템의 이름은 sys1.1입니다. 원격 시스템은 remote1입니다.

```
sys1.1 # cat /etc/syslog.conf
...
audit.notice       @remote1
```

remote1 시스템에 있는 syslog.conf 파일의 audit.notice 항목은 로그 파일을 가리킵니다.

```
remote1 # cat /etc/syslog.conf
...
audit.notice       /var/adm/auditlog
```

영역에서 감사 서비스 구성(작업)

감사 서비스는 영역의 감사 이벤트를 포함한 전체 시스템을 감사합니다. 비전역 영역을 설치한 시스템은 모든 영역을 동일하게 감사하거나 영역별로 감사를 구성할 수 있습니다. 배경 지식은 530 페이지 “Oracle Solaris 영역이 있는 시스템에 대한 감사”를 참조하십시오. 계획하려면 534 페이지 “영역에서 감사를 계획하는 방법”을 참조하십시오.

비전역 영역을 전역 영역과 동일하게 감사할 경우 감사 서비스가 전역 영역에서 실행됩니다. 서비스는 전역 영역 및 모든 비전역 영역에서 감사 레코드를 수집합니다. 비전역 영역 관리자에게는 감사 레코드에 대한 액세스 권한이 없을 수 있습니다.

주 – 전역 영역 관리자는 비전역 영역에 있는 사용자의 감사 마스크를 수정하도록 선택할 수 있습니다.

비전역 영역을 개별적으로 감사할 경우 별도의 감사 서비스가 감사되는 각 영역에서 실행됩니다. 각 영역에서는 고유의 감사 레코드를 수집합니다. 레코드는 비전역 영역 및 비전역 영역 루트의 전역 영역에 표시됩니다.

▼ 감사를 위해 동일하게 모든 영역을 구성하는 방법

이 절차에서는 모든 영역을 동일하게 감사할 수 있습니다. 이 방법에는 가장 작은 컴퓨터 오버헤드와 관리 리소스가 요구됩니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

1 감사를 위해 전역 영역을 구성합니다.

다음 예외 사항을 적용하여 546 페이지 “감사 서비스 구성(작업 맵)”에서 작업을 완료합니다.

- `perzone` 감사 정책을 사용으로 설정하지 않습니다.
- 감사 서비스를 사용으로 설정하지 않습니다. 감사를 위해 비전역 영역을 구성한 후 감사 서비스를 사용으로 설정합니다.
- `zonename` 정책을 설정합니다. 이 정책은 영역의 이름을 모든 감사 레코드에 추가합니다.

```
# auditconfig -setpolicy +zonename
```

2 감사 구성 파일을 수정한 경우 전역 영역에서 모든 비전역 영역으로 복사합니다.

`audit_class` 또는 `audit_event` 파일을 수정한 경우 두 가지 방법 중 하나로 복사합니다.

- 파일을 루프백 마운트할 수 있습니다.
- 파일을 복사할 수 있습니다.

비전역 영역이 실행되고 있어야 합니다.

- **변경된 `audit_class` 및 `audit_event` 파일을 루프백 파일 시스템(`lofs`)으로 마운트합니다.**

  a. 전역 영역에서 비전역 영역을 정지합니다.

  ```
  # zoneadm -z non-global-zone halt
  ```

b. 전역 영역에서 수정한 모든 감사 구성 파일에 대해 읽기 전용 루프백 마운트를 만듭니다.

```
# zonecfg -z non-global-zone
 add fs
    set special=/etc/security/audit-file
    set dir=/etc/security/audit-file
    set type=lofs
    add options [ro,nodevices,nosetuid]
    commit
    end
 exit
```

c. 변경 사항을 적용하려면 비전역 영역을 부트합니다.

```
# zoneadm -z non-global-zone boot
```

나중에 전역 영역에서 감사 구성 파일을 수정할 경우 영역을 재부트하여 비전역 영역에서 루프백 마운트된 파일을 새로 고칩니다.

- **파일을 복사합니다.**

  a. 전역 영역에서 비전역 영역의 `/etc/security` 디렉토리를 나열합니다.

  ```
  # ls /zone/zonename/root/etc/security/
  ```

  b. 변경된 `audit_class` 및 `audit_event` 파일을 영역의 `/etc/security` 디렉토리에 복사합니다.

  ```
  # cp /etc/security/audit-file /zone/zonename/root/etc/security/audit-file
  ```

  나중에 전역 영역에서 이러한 파일 중 하나를 변경할 경우 비전역 영역에 파일을 다시 복사해야 합니다.

감사 서비스가 전역 영역에서 사용으로 설정되면 비전역 영역이 감사됩니다.

예 28–20 감사 구성 파일을 영역의 루프백 마운트로 마운트

이 예에서는 시스템 관리자가 `audit_class` , `audit_event` 및 `audit_warn` 파일을 수정했습니다.

`audit_warn` 파일은 전역 영역에서만 읽히므로 비전역 영역에 마운트할 필요가 없습니다.

이 시스템 `machine1`에서 관리자는 `machine1–webserver` 및 `machine1–appserver`의 두 비전역 영역을 만들었습니다. 관리자는 감사 구성 파일 수정을 마쳤습니다. 관리자가 나중에 파일을 수정할 경우 영역을 재부트하여 루프백 마운트를 다시 읽어야 합니다.

```
# zoneadm -z machine1-webserver halt
# zoneadm -z machine1-appserver halt
# zonecfg -z machine1-webserver
 add fs
    set special=/etc/security/audit_class
    set dir=/etc/security/audit_class
    set type=lofs
```

```
    add options [ro,nodevices,nosetuid]
    commit
    end
 add fs
    set special=/etc/security/audit_event
    set dir=/etc/security/audit_event
    set type=lofs
    add options [ro,nodevices,nosetuid]
    commit
    end
  exit
# zonecfg -z machine1-appserver
 add fs
    set special=/etc/security/audit_class
    set dir=/etc/security/audit_class
    set type=lofs
    add options [ro,nodevices,nosetuid]
    commit
    end
...
 exit
```

비전역 영역이 재부트되면 `audit_class` 및 `audit_event` 파일은 영역에서 읽기 전용입니다.

▼ 영역별 감사를 구성하는 방법

이 절차에서는 별도의 영역 관리자가 해당 영역에서 감사 서비스를 제어할 수 있습니다. 전체 정책 옵션 목록은 `auditconfig(1M)` 매뉴얼 페이지를 참조하십시오.

시작하기 전에 감사를 구성하려면 Audit Configuration 권한 프로파일이 지정되어야 합니다. 감사 서비스를 사용으로 설정하려면 Audit Control 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2. **전역 영역에서 감사를 구성합니다.**

   a. **546 페이지 "감사 서비스 구성(작업 맵)"의 작업을 완료합니다.**

   b. **perzone 감사 정책을 추가합니다. 명령은 예 28–8을 참조하십시오.**

   주 – 전역 영역에서 감사 서비스를 사용으로 설정할 필요는 없습니다.

3. **감사하고자 하는 각 비전역 영역에서 감사 파일을 구성합니다.**

   a. **546 페이지 "감사 서비스 구성(작업 맵)"의 작업을 완료합니다.**

b. 시스템 전역 감사 설정은 구성하지 않습니다.

특히 perzone 또는 ahlt 정책을 비전역 영역에 추가하지 않습니다.

4 영역에서 감사를 사용으로 설정합니다.

```
myzone# audit -s
```

예 28–21 비전역 영역에서 감사를 사용 안함으로 설정

이 예는 전역 영역에서 perzone 감사 정책을 설정한 경우 유효합니다. noaudit 영역의 영역 관리자는 해당 영역에 대한 감사를 사용 안함으로 설정합니다.

```
noauditzone # auditconfig -getcond
audit condition = auditing
noauditzone # audit -t
noauditzone # auditconfig -getcond
audit condition = noaudit
```

감사 서비스를 사용/사용 안함으로 설정(작업)

감사 서비스는 기본적으로 사용으로 설정되며 auditconfig 명령으로 구성됩니다. perzone 감사 정책이 전역 영역에서 설정된 경우 영역 관리자가 해당 비전역 영역에 대한 서비스를 사용/사용 안함으로 설정하고 새로 고칠 수 있습니다.

▼ 감사 서비스를 새로 고치는 방법

이 절차에서는 감사 서비스가 사용으로 설정된 후 감사 플러그인의 구성을 변경한 경우 감사 서비스를 업데이트합니다.

시작하기 전에 Audit Control 권한 프로파일이 지정되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 감사 서비스를 새로 고칩니다.

```
# audit -s
```

주 – 감사 서비스를 새로 고치면 모든 임시 구성 설정이 손실됩니다. 감사 정책 및 대기열 제어에서는 임시 설정을 허용합니다. 자세한 내용은 auditconfig(1M) 매뉴얼 페이지를 참조하십시오.

---

3 **현재 감사되고 있는 사용자의 사전 선택 마스크를 새로 고칩니다.**

감사 레코드는 각 프로세스와 연결된 감사 사전 선택 마스크를 기준으로 생성됩니다. 감사 서비스를 새로 고쳐도 기존 프로세스의 마스크는 변경되지 **않습니다**. 기존 프로세스에 대한 사전 선택 마스크를 명시적으로 재설정하려면 595 페이지 "로그인한 사용자의 사전 선택 마스크를 업데이트하는 방법"을 참조하십시오.

예 28–22 사용으로 설정된 감사 서비스 새로 고침

이 예에서는 관리자가 감사를 재구성하고 변경 사항을 확인한 다음 감사 서비스를 새로 고칩니다.

- 먼저, 관리자는 임시 정책을 추가합니다.

```
# auditconfig -t -setpolicy +zonename
# auditconfig -getpolicy
configured audit policies = ahlt,arge,argv,perzone
active audit policies = ahlt,arge,argv,perzone,zonename
```

- 그런 다음 관리자는 대기열 제어를 지정합니다.

```
# auditconfig -setqctrl 200 20 0 0
# auditconfig -getqctrl
configured audit queue hiwater mark (records) = 200
configured audit queue lowater mark (records) = 20
configured audit queue buffer size (bytes) = 8192
configured audit queue delay (ticks) = 20
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 20
active audit queue buffer size (bytes) = 8192
active audit queue delay (ticks) = 20
```

- 그런 다음 관리자는 플러그인 속성을 지정합니다.
  - `audit_binfile` 플러그인에 대해 관리자는 `qsize` 값을 제거합니다.

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile (active)
    Attributes: p_dir=/audit/sys1.1,/var/audit;
    p_minfree=2;p_fsize=4G;
Queue size: 200
# auditconfig -setplugin audit_binfile active  "" ""
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile (active)
    Attributes: p_dir=/audit/sys1.1,/var/audit
 p_minfree=2;p_fsize=4G;
```

    마지막 비어 있는 큰 따옴표("")는 플러그인에 대한 대기열 크기를 기본값으로 설정합니다.

  - `audit_syslog` 플러그인에 대해 관리자는 성공한 로그인/로그아웃 이벤트 및 실패한 실행 파일이 `syslog`에 보내지도록 구성합니다. 이 플러그인에 대한 `qsize`는 50으로 설정되었습니다.

```
# auditconfig -setplugin audit_syslog active p_flags=+lo,-ex 50
# auditconfig -getplugin audit_syslog
auditconfig -getplugin audit_syslog
```

```
Plugin: audit_syslog (active)
    Attributes: p_flags=+lo,-ex;
    Queue size: 50
```

- 관리자는 audit_remote 플러그인을 구성하거나 사용하지 않습니다.

- 그런 다음 관리자는 감사 서비스를 새로 고치고 구성을 확인합니다.
  - 임시 zonename 정책은 더 이상 설정되지 않습니다.

    ```
    # audit -s
    # auditconfig -getpolicy
    configured audit policies = ahlt,arge,argv,perzone
    active audit policies = ahlt,arge,argv,perzone
    ```

  - 대기열 제어는 동일하게 유지됩니다.

    ```
    # auditconfig -getqctrl
    configured audit queue hiwater mark (records) = 200
    configured audit queue lowater mark (records) = 20
    configured audit queue buffer size (bytes) = 8192
    configured audit queue delay (ticks) = 20
    active audit queue hiwater mark (records) = 200
    active audit queue lowater mark (records) = 20
    active audit queue buffer size (bytes) = 8192
    active audit queue delay (ticks) = 20
    ```

  - audit_binfile 플러그인에는 지정된 대기열 크기가 없습니다. audit_syslog 플러그인에는 지정된 대기열 크기가 있습니다.

    ```
    # auditconfig -getplugin
    Plugin: audit_binfile (active)
        Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;

    Plugin: audit_syslog (active)
        Attributes: p_flags=+lo,-ex;
        Queue size: 50
    ...
    ```

▼ 감사 서비스를 사용 안함으로 설정하는 방법

이 절차에서는 perzone 감사 정책이 설정되었을 때 전역 영역 및 비전역 영역에서 감사를 사용 안함으로 설정하는 방법을 설명합니다.

- perzone 감사 정책이 설정되지 않은 경우 감사는 모든 영역에 대해 사용 안함으로 설정됩니다.
- perzone 감사 정책이 전역 영역에서 설정된 경우 정책은 감사를 사용으로 설정한 비전역 영역에서 유효합니다.

  perzone 정책이 전역 영역에서 설정되었으므로 전역 영역 재부트 및 비전역 영역 재부트 시에도 비전역 영역에서 계속해서 감사 레코드를 수집합니다.

시작하기 전에 Audit Control 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**
자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **audit -t 명령을 실행하여 감사 서비스를 사용 안함으로 설정합니다.**
자세한 내용은 audit(1M) 및 auditd(1M) 매뉴얼 페이지를 참조하십시오.

- **전역 영역에서 감사 서비스를 사용 안함으로 설정합니다.**

  ```
  # audit -t
  ```

  perzone 감사 정책이 설정되지 않은 경우 이 명령은 모든 영역에서 감사를 사용 안함으로 설정합니다.

- **비전역 영역에서 감사 서비스를 사용 안함으로 설정합니다.**
  perzone 감사 정책이 설정된 경우 비전역 영역 관리자는 비전역 영역에서 서비스를 사용 안함으로 설정해야 합니다.

  ```
  zone1 # audit -t
  ```

▼ 감사 서비스를 사용으로 설정하는 방법

이 절차에서는 관리자가 서비스를 사용 안함으로 설정한 후 모든 영역에 대해 감사 서비스를 사용으로 설정할 수 있습니다. 비전역 영역에서 감사 서비스를 시작하려면 예 28–23을 참조하십시오.

시작하기 전에 감사 서비스를 사용 또는 사용 안함으로 설정하려면 Audit Control 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**
자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **audit -s 명령을 사용하여 감사 서비스를 사용으로 설정합니다.**

```
# audit -s
```

자세한 내용은 audit(1M) 매뉴얼 페이지를 참조하십시오.

3 **감사가 사용으로 설정되었는지 확인합니다.**

```
# auditconfig -getcond
audit condition = auditing
```

예 28–23 비전역 영역에서 감사를 사용으로 설정

이 예에서는 영역 관리자가 다음 작업을 수행한 후 zone1에 대해 감사 서비스를 사용으로 설정합니다.

- 전역 영역 관리자는 전역 영역에서 perzone 정책을 설정합니다.
- 비전역 영역 관리자는 감사 서비스 및 사용자별 사용자 정의를 구성합니다.

그런 다음 영역 관리자는 영역에 대해 감사 서비스를 사용으로 설정합니다.

```
zone1# audit -s
```

로컬 시스템에서 감사 레코드 관리(작업)

기본 플러그인 audit_binfile은 감사 추적을 만듭니다. 감사 추적을 관리하여 네트워크에서 사용자의 작업을 모니터할 수 있습니다. 감사는 많은 양의 데이터를 생성할 수 있습니다. 다음 작업에서는 이러한 모든 데이터로 작업하는 방법을 보여줍니다.

로컬 시스템에서 감사 레코드 관리(작업 맵)

다음 작업 맵에서는 감사 레코드를 선택, 분석 및 관리하기 위한 절차를 안내합니다.

작업	설명	수행 방법
감사 레코드의 형식을 표시합니다.	감사 이벤트에 대해 수집되는 정보의 종류 및 정보가 표시되는 순서를 보여줍니다.	577 페이지 “감사 레코드 정의를 표시하는 방법”
감사 레코드를 병합합니다.	여러 시스템의 감사 파일을 하나의 감사 추적으로 합칩니다.	578 페이지 “감사 추적에서 감사 파일을 병합하는 방법”
검사할 레코드를 선택합니다.	조사할 특정 이벤트를 선택합니다.	580 페이지 “감사 추적에서 감사 이벤트를 선택하는 방법”
감사 레코드를 표시합니다.	이진 감사 레코드를 볼 수 있습니다.	582 페이지 “이진 감사 파일의 내용을 보는 방법”
잘못 이름 지정된 감사 파일을 정리합니다.	감사 서비스에서 실수로 열어 둔 감사 파일에 종료 시간 기록을 제공합니다.	584 페이지 “not_terminated 감사 파일을 정리하는 방법”
감사 추적 오버플로우를 막습니다.	감사 파일 시스템이 가득 차지 않도록 막습니다.	585 페이지 “감사 추적 오버플로우를 막는 방법”

▼ 감사 레코드 정의를 표시하는 방법

auditrecord 명령은 감사 레코드 정의를 표시합니다. 정의는 감사 이벤트 번호, 감사 클래스, 선택 마스크 및 감사 이벤트의 레코드 형식을 제공합니다.

- **모든 감사 이벤트 레코드의 정의를 HTML 파일로 만듭니다.**

  -a 옵션은 모든 감사 이벤트 정의를 나열합니다. -h 옵션은 목록을 HTML 형식으로 만듭니다.

  ```
  % auditrecord -ah > audit.events.html
  ```

참고 – 브라우저에서 HTML 파일을 표시하면 브라우저의 Find(찾기) 도구를 사용하여 특정 감사 레코드 정의를 찾습니다.

자세한 내용은 auditrecord(1M) 매뉴얼 페이지를 참조하십시오.

예 28–24 프로그램의 감사 레코드 형식 표시

이 예에서는 login 프로그램으로 생성되는 모든 감사 레코드의 형식이 표시됩니다. 로그인 프로그램에는 rlogin, telnet, newgrp 및 Oracle Solaris의 보안 셸 기능이 포함됩니다.

```
% auditrecord -p login
...
login: logout
  program     various              See login(1)
  event ID    6153                 AUE_logout
  class       lo                  (0x0000000000001000)
...
newgrp
  program     newgrp               See newgrp login
  event ID    6212                 AUE_newgrp_login
  class       lo                  (0x0000000000001000)
...
rlogin
  program     /usr/sbin/login      See login(1) - rlogin
  event ID    6155                 AUE_rlogin
  class       lo                   (0x0000000000001000)
...
/usr/lib/ssh/sshd
  program     /usr/lib/ssh/sshd    See login - ssh
  event ID    6172                 AUE_ssh
  class       lo                   (0x0000000000001000)
...
telnet login
  program     /usr/sbin/login      See login(1) - telnet
  event ID    6154                 AUE_telnet
  class       lo                   (0x0000000000001000)
  ...
```

예 28–25 감사 클래스의 감사 레코드 형식 표시

이 예에서는 예 28–10에서 만들어진 pf 클래스의 모든 감사 레코드 형식이 표시됩니다.

```
% auditrecord -c pf

pfexec
  system call pfexec               See execve(2) with pfexec enabled
  event ID    116                  AUE_PFEXEC
  class       pf                   (0x0100000000000000)
      header
      path                    pathname of the executable
      path                    pathname of working directory
      [privileges]            privileges if the limit or inheritable set are changed
      [privileges]            privileges if the limit or inheritable set are changed
      [process]               process if ruid, euid, rgid or egid is changed
      exec_arguments
      [exec_environment]      output if arge policy is set
      subject
      [use_of_privilege]
      return
```

use_of_privilege 토큰은 권한이 사용될 때마다 기록됩니다. privileges 토큰은 제한 또는 상속 가능한 설정이 변경될 경우 기록됩니다. process 토큰은 ID가 변경될 경우 기록됩니다. 이러한 토큰이 레코드에 포함되기 위해 필요한 정책 옵션은 없습니다.

▼ 감사 추적에서 감사 파일을 병합하는 방법

모든 감사 디렉토리의 모든 감사 파일을 결합하면 전체 감사 추적의 내용을 분석할 수 있습니다. auditreduce 명령은 입력 파일의 모든 레코드를 단일 출력 파일로 병합합니다. 그런 다음 입력 파일은 삭제할 수 있습니다. 지정된 경로가 없는 경우 auditreduce 명령은 /var/audit 파일 시스템을 사용합니다.

주 – 감사 추적의 시간 기록은 협정 세계시(UTC)로 되어 있으므로 의미를 가지려면 날짜와 시간을 현재 시간대로 변환해야 합니다. auditreduce 명령이 아닌 표준 파일 명령으로 이러한 파일을 조작할 때는 항상 이 사항을 염두에 두십시오.

시작하기 전에 Audit Review 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2. **병합된 감사 파일을 저장할 파일 시스템을 만듭니다.**
   이 파일 시스템은 원래 파일을 저장하기 위해 560 페이지 “감사 파일에 대한 ZFS 파일 시스템을 만드는 방법”에서 만든 파일 시스템과 **다른** *zpool*에 있어야 합니다.

3 감사 추적의 감사 레코드를 병합합니다.

병합된 감사 파일을 저장할 디렉토리로 디렉토리를 변경합니다. 이 디렉토리에서 감사 레코드를 이름이 지정된 접미어가 있는 파일로 병합합니다. 로컬 시스템에 있는 감사 추적의 모든 디렉토리가 병합됩니다.

```
# cd audit-storage-directory
# auditreduce -Uppercase-option -O suffix
```

`auditreduce` 명령에 대한 대문자 옵션은 감사 추적의 파일을 조작합니다. 대문자 옵션에는 다음이 포함됩니다.

`-A`	감사 추적의 모든 파일을 선택합니다.
`-C`	완전한 파일만 선택합니다.
`-M`	특정 접미어가 있는 파일을 선택합니다. 접미어는 시스템 이름이거나 요약 파일에 대해 지정한 접미어일 수 있습니다.
`-O`	현재 디렉토리에서 *suffix* 접미어를 사용하여 시작 시간과 종료 시간 모두에 대해 14자의 시간 기록을 가진 감사 파일을 만듭니다.
`-R` *pathname*	대체 감사 루트 디렉토리인 *pathname*에서 감사 파일을 읽도록 지정합니다.
`-S` *server*	지정된 서버에서 감사 파일을 읽도록 지정합니다.

전체 옵션 목록은 `auditreduce(1M)` 매뉴얼 페이지를 참조하십시오.

4 병합된 파일을 다른 zpool에 있는 파일 시스템으로 이동합니다.

파일을 다른 시스템으로 이동하려면 `sftp` 명령을 사용합니다. 지침은 `sftp(1)` 매뉴얼 페이지를 참조하십시오.

예 28–26 감사 파일을 요약 파일로 복사

다음 예에서는 System Administrator 권한 프로파일이 지정된 관리자가 감사 추적의 모든 파일을 다른 파일 시스템의 병합된 파일로 복사합니다. `/var/audit/storage` 파일 시스템은 감사 루트 파일 시스템인 `/var/audit` 파일 시스템과 다른 별도의 디스크에 있습니다.

```
$ cd /var/audit/storage
$ auditreduce -A -O All
$ ls /var/audit/storage/*All
20100827183214.20100827215318.All
```

다음 예에서는 완전한 파일만 감사 추적에서 병합된 파일로 복사됩니다. 전체 경로는 `-0` 옵션의 값으로 지정됩니다. 경로의 마지막 구성 요소인 `Complete`는 접미어로 사용됩니다.

```
$ auditreduce -C -O /var/audit/storage/Complete
$ ls /var/audit/storage/*Complete
20100827183214.20100827214217.Complete
```

다음 예에서는 완전한 파일만 sys1.1 시스템에서 병합된 파일로 복사됩니다.

```
$ cd /var/audit/storage
$ auditreduce -M sys1.1 -O example1summ
$ ls /var/audit/storage/*summ
20100827183214.20100827214217.example1summ
```

예 28–27 감사 파일을 요약 파일로 이동

auditreduce 명령에 대한 -D 옵션은 감사 파일을 다른 위치로 복사할 때 해당 감사 파일을 삭제합니다. 다음 예에서는 sys1.1 시스템에 대한 완전한 감사 파일이 나중에 검사를 위해 audit_summary 파일 시스템으로 복사됩니다.

```
$ cd /var/audit/audit_summary
$ auditreduce -C -O daily_sys1.1 -D sys1.1
$ ls *sys1.1
20100827183214.20100827214217.daily_sys1.1
```

*daily_sys1.1 파일에 대한 입력인 sys1.1 시스템의 감사 파일은 명령이 성공적으로 완료되면 제거됩니다.

▼ 감사 추적에서 감사 이벤트를 선택하는 방법

검사를 위해 감사 레코드를 필터링할 수 있습니다. 전체 필터링 옵션 목록은 auditreduce(1M) 매뉴얼 페이지를 참조하십시오.

시작하기 전에 Audit Review 권한 프로파일이 지정되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 감사 추적 또는 지정된 감사 파일에서 원하는 레코드의 종류를 선택합니다.

auditreduce *-lowercase-option argument* [*optional-file*]

argument 소문자 옵션에 필요한 특정 인수입니다. 예를 들어, -c 옵션에는 ua와 같은 감사 클래스의 *argument*가 필요합니다.

-d 특정 날짜의 모든 이벤트를 선택합니다. *argument*에 대한 날짜 형식은 *yyymmdd*입니다. 다른 날짜 옵션인 -b 및 -a는 특정 날짜 전후의 이벤트를 선택합니다.

`-u` 특정 사용자에게 지정 가능한 모든 이벤트를 선택합니다. *argument*는 사용자 이름입니다. 다른 사용자 옵션인 `-e`는 유효 사용자 ID에 지정 가능한 모든 이벤트를 선택합니다.

`-c` 사전 선택된 감사 클래스의 모든 이벤트를 선택합니다. *argument*는 감사 클래스 이름입니다.

`-m` 특정 감사 이벤트의 모든 인스턴스를 선택합니다. *argument*는 감사 이벤트입니다.

optional-file 감사 파일의 이름입니다.

전체 옵션 목록은 `auditreduce(1M)` 매뉴얼 페이지를 참조하십시오.

예 28–28 감사 파일 결합 및 줄이기

`auditreduce` 명령은 입력 파일을 결합할 때 관심이 적은 레코드를 없앨 수 있습니다. 예를 들어, `auditreduce` 명령을 사용하여 1개월이 지난 감사 파일에서 로그인 및 로그아웃 레코드만 유지할 수 있습니다. 전체 감사 추적을 검색해야 하는 경우 백업 매체에서 추적을 복구할 수 있습니다.

```
# cd /var/audit/audit_summary
# auditreduce -O lo.summary -b 20100827 -c lo; compress *lo.summary
```

예 28–29 한 사용자의 감사 레코드를 요약 파일로 복사

이 예에서는 특정 사용자의 이름을 포함하는 감사 추적의 레코드가 병합됩니다. `-e` 옵션은 유효 사용자를 찾습니다. `-u` 옵션은 로그인 사용자를 찾습니다.

```
$ cd /var/audit/audit_summary
$ auditreduce -e tamiko -O tamiko
```

이 파일에서 특정 이벤트를 찾을 수 있습니다. 다음 예에서는 사용자가 2010년 9월 7일에 로그인 및 로그아웃한 시간이 확인됩니다. 사용자의 이름이 파일 접미어로 있는 파일만 확인됩니다. 날짜의 짧은 형식은 *yyyymmdd*입니다.

```
# auditreduce -M tamiko -O tamikolo -d 20100907 -u tamiko -c lo
```

예 28–30 선택한 레코드를 단일 파일로 복사

이 예에서는 특정 일에 대한 로그인 및 로그아웃 레코드가 감사 추적에서 선택됩니다. 레코드는 대상 파일로 병합됩니다. 대상 파일은 감사 루트 디렉토리를 포함하는 파일 시스템 이외의 파일 시스템에 쓰여집니다.

```
# auditreduce -c lo -d 20100827 -O /var/audit/audit_summary/logins
# ls /var/audit/audit_summary/*logins
/var/audit/audit_summary/20100827183936.20100827232326.logins
```

▼ 이진 감사 파일의 내용을 보는 방법

praudit 명령을 사용하여 이진 감사 파일의 내용을 볼 수 있습니다. auditreduce 명령에서 출력을 파이프하거나 특정 감사 파일을 읽을 수 있습니다. -x 옵션은 추가 처리에 유용합니다.

시작하기 전에 Audit Review 권한 프로파일이 지정되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 다음 praudit 명령 중 하나를 사용하여 용도에 가장 적합한 출력을 생성합니다.

다음 예에서는 동일 감사 이벤트에서 praudit 출력을 보여줍니다. sequence 및 trailer 토큰을 포함하도록 감사 정책이 설정되었습니다.

- praudit -s 명령은 짧은 형식(한 행당 하나의 토큰)으로 감사 레코드를 표시합니다. -l 옵션을 사용하여 각 레코드를 한 행에 표시합니다.

```
$ auditreduce -c lo | praudit -s
header,69,2,AUE_screenlock,,mach1,2010-10-14 08:02:56.348 -07:00
subject,jdoe,root,staff,jdoe,staff,856,50036632,82 0 mach1
return,success,0
sequence,1298
```

- praudit -r 명령은 원시 형식(한 행당 하나의 토큰)으로 감사 레코드를 표시합니다. -l 옵션을 사용하여 각 레코드를 한 행에 표시합니다.

```
$ auditreduce -c lo | praudit -r
21,69,2,6222,0x0000,10.132.136.45,1287070091,698391050
36,26700,0,10,26700,10,856,50036632,82 0 10.132.136.45
39,0,0
47,1298
```

- praudit -x 명령은 XML 형식(한 행당 하나의 토큰)으로 감사 레코드를 표시합니다. -l 옵션을 사용하여 한 행에 하나의 레코드에 대한 XML 출력을 표시합니다. 다음 목록은 이 인쇄 페이지에 맞추어 두 행의 출력으로 나누어집니다.

```
$ auditreduce -c lo | praudit -x
<record version="2" event="screenlock - unlock" host="mach1"
     iso8601="2010-10-14 08:28:11.698 -07:00">
<subject audit-uid="jdoe" uid="root" gid="staff" ruid="jdoe
    rgid="staff" pid="856" sid="50036632" tid="82 0 mach1"/>
<return errval="success" retval="0"/>
<sequence seq-num="1298"/>
</record>
```

예 28–31 전체 감사 추적 인쇄

인쇄 명령에 파이프를 사용하면 전체 감사 추적이 프린터로 출력됩니다. 보안상 이유로 프린터는 제한적인 액세스 권한을 가집니다.

```
# auditreduce | praudit | lp -d example.protected.printer
```

예 28–32 특정 감사 파일 보기

이 예에서는 요약 로그인 파일이 터미널 창에서 검사됩니다.

```
# cd /var/audit/audit_summary/logins
# praudit 20100827183936.20100827232326.logins | more
```

예 28–33 감사 레코드를 XML 형식으로 표시

이 예에서는 감사 레코드가 XML 형식으로 변환됩니다.

```
# praudit -x 20100827183214.20100827215318.logins > 20100827.logins.xml
```

XML 파일은 브라우저에서 표시할 수 있습니다. 파일의 내용은 스크립트로 관련 정보를 추출하여 작업할 수 있습니다.

예 28–34 스크립트를 사용하여 praudit 출력 처리

praudit 명령의 출력을 텍스트 행으로 처리하고자 할 수 있습니다. 예를 들어, auditreduce 명령에서 선택할 수 없는 레코드를 선택하고자 할 수 있습니다. 간단한 셸 스크립트를 사용하여 praudit 명령의 출력을 처리할 수 있습니다. 다음 간단한 예제 스크립트는 하나의 감사 레코드를 한 행에 표시하고 사용자 지정 문자열을 검색한 다음 감사 파일을 원래 형식으로 반환합니다.

```
#!/bin/sh
#
## This script takes an argument of a user-specified string.
#  The sed command prefixes the header tokens with Control-A
#  The first tr command puts the audit tokens for one record
#  onto one line while preserving the line breaks as Control-A
#
praudit | sed -e '1,2d' -e '$s/^file.*$//' -e 's/^header/^aheader/' \\
| tr '\\012\\001' '\\002\\012' \\
| grep "$1" \\       Finds the user-specified string
| tr '\\002' '\\012'       Restores the original newline breaks
```

스크립트의 ^a는 ^과 a의 두 문자가 아닌 Ctrl-A입니다. 접두어는 텍스트로 나타날 수 있는 header 문자열에서 header 토큰을 구분합니다.

일반 오류 다음과 유사한 메시지는 praudit 명령을 사용할 수 있는 충분한 권한이 없음을 나타냅니다.

```
praudit: Can't assign 20090408164827.20090408171614.sys1.1 to stdin.
```

프로파일 셸에서 praudit 명령을 실행합니다. Audit Review 권한 프로파일이 지정되어야 합니다.

▼ not_terminated 감사 파일을 정리하는 방법

비정상적인 시스템 중단이 발생할 경우 감사 파일이 열린 상태에서 감사 서비스가 종료됩니다. 또는 파일 시스템에 액세스할 수 없게 되고 시스템이 강제로 새로운 파일 시스템으로 전환됩니다. 이러한 경우 감사 파일이 더 이상 감사 레코드에 사용되지 않더라도 감사 파일은 종료 시간 기록으로 not_terminated 문자열을 가집니다. auditreduce -O 명령을 사용하여 파일에 올바른 시간 기록을 지정합니다.

시작하기 전에 Audit Review 권한 프로파일이 지정되어야 합니다.

1. **필요한 보안 속성을 가진 관리자가 됩니다.**
   자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2. **감사 파일 시스템에 not_terminated 문자열이 있는 파일을 만들어진 순서대로 나열합니다.**

   ```
   # ls -R1t audit-directory*/* | grep not_terminated
   ```

   -R 하위 디렉토리의 파일을 나열합니다.

   -t 최근에서 가장 오래된 순서로 파일을 나열합니다.

   -1 파일을 한 열로 나열합니다.

3. **오래된 not_terminated 파일을 정리합니다.**
   auditreduce -O 명령에 오래된 파일의 이름을 지정합니다.

   ```
   # auditreduce -O system-name old-not-terminated-file
   ```

4. **오래된 not_terminated 파일을 제거합니다.**

   ```
   # rm system-name old-not-terminated-file
   ```

예 28–35 닫힌 not_terminated 감사 파일 정리

다음 예에서는 not_terminated 파일을 찾고 이름을 바꾼 다음 원본을 제거합니다.

```
ls -R1t */* | grep not_terminated
.../egret.1/20100908162220.not_terminated.egret
.../egret.1/20100827215359.not_terminated.egret
# cd */egret.1
# auditreduce -O egret 20100908162220.not_terminated.egret
# ls -1t
20100908162220.not_terminated.egret       Current audit file
```

```
20100827230920.20100830000909.egret      Cleaned up audit file
20100827215359.not_terminated.egret      Input (old) audit file
# rm 20100827215359.not_terminated.egret
# ls -1t
20100908162220.not_terminated.egret      Current audit file
20100827230920.20100830000909.egret      Cleaned up audit file
```

새 파일의 시작 시간 기록에는 `not_terminated` 파일에서 첫 감사 이벤트의 시간이 반영됩니다. 종료 시간 기록에는 파일에서 마지막 감사 이벤트의 시간이 반영됩니다.

▼ 감사 추적 오버플로우를 막는 방법

보안 정책에서 모든 감사 데이터가 저장되도록 요구하는 경우 감사 레코드 손실을 막습니다.

시작하기 전에 `root` 역할을 가진 사용자여야 합니다.

1. **`audit_binfile` 플러그인에서 최소 사용 가능 크기를 설정합니다.**

   `p_minfree` 속성을 사용합니다.

   `audit_warn` 전자 메일 별칭은 디스크 공간이 최소 사용 가능 크기까지 채워지면 경고를 보냅니다. 예 28–17을 참조하십시오.

2. **감사 파일을 정기적으로 아카이브하도록 일정을 설정합니다.**

   파일을 오프라인 매체에 백업하여 감사 파일을 아카이브합니다. 또한 아카이브 파일 시스템으로 파일을 이동할 수도 있습니다.

   `syslog` 유틸리티를 사용하여 텍스트 감사 로그를 수집하는 경우 텍스트 로그를 아카이브합니다. 자세한 내용은 `logadm`(1M) 매뉴얼 페이지를 참조하십시오.

3. **아카이브된 감사 파일을 감사 파일 시스템에서 삭제하도록 일정을 설정합니다.**

4. **보조 정보를 저장하고 보관합니다.**

   감사 추적과 함께 감사 레코드를 해석하는 데 필요한 정보를 아카이브합니다. 최소한 `passwd`, `group` 및 `hosts` 파일을 저장합니다. 또한 `audit_event` 및 `audit_class` 파일을 아카이브할 수 있습니다.

5. **어떤 감사 파일이 아카이브되었는지 기록합니다.**

6. **아카이브된 매체를 적절히 보관합니다.**

7 **ZFS 압축을 사용으로 설정하여 필요한 파일 시스템 용량을 줄입니다.**

감사 파일 전용 ZFS 파일 시스템에서 압축을 사용하면 파일이 크게 줄어듭니다. 예는 597 페이지 "전용 파일 시스템에서 감사 파일을 압축하는 방법"을 참조하십시오.

또한 **Oracle Solaris 관리: ZFS 파일 시스템**의 "ZFS 압축, 중복 제거 및 암호화 등록 정보 간의 상호 작용"을 참조하십시오.

8 **요약 파일을 만들어 저장하는 감사 데이터의 양을 줄입니다.**

`auditreduce` 명령에 대한 옵션을 사용하여 감사 추적에서 요약 파일을 추출할 수 있습니다. 요약 파일에는 지정된 유형의 감사 이벤트에 대한 레코드만 포함됩니다. 요약 파일을 추출하려면 예 28–28 및 예 28–30을 참조하십시오.

감사 서비스 문제 해결(작업)

이 절에서는 다양한 감사 오류 메시지, 기본 설정 및 다른 도구에서 제공하는 감사를 다룹니다. 이러한 절차는 필요한 감사 이벤트를 기록하고 감사 문제를 디버깅하는 데 유용합니다.

감사 서비스 문제 해결(작업 맵)

다음 작업 맵에서는 감사 문제 해결을 위한 절차를 안내합니다.

문제점	해결 방법	수행 방법
감사를 구성했는데 감사 레코드가 기록되지 않는 이유는 무엇입니까?	감사 서비스 문제를 해결합니다.	587 페이지 "감사가 실행 중인지 확인하는 방법"
수집되는 감사 정보의 양을 줄이려면 어떻게 합니까?	감사하고자 하는 이벤트만 감사합니다.	589 페이지 "생성되는 감사 레코드의 양을 줄이는 방법"
사용자가 시스템에서 수행하는 모든 작업을 감사하려면 어떻게 합니까?	모든 명령에 대해 하나 이상의 사용자를 감사합니다.	591 페이지 "사용자의 모든 명령을 감사하는 방법"
기록되는 감사 이벤트를 변경하고 변경 사항을 기존 세션에 적용하려면 어떻게 합니까?	사용자의 사전 선택 마스크를 업데이트합니다.	595 페이지 "로그인한 사용자의 사전 선택 마스크를 업데이트하는 방법"
특정 파일에 대한 수정 사항을 찾으려면 어떻게 합니까?	파일 수정을 감사한 다음 `auditreduce` 명령을 사용하여 특정 파일을 찾습니다.	593 페이지 "특정 파일에 대한 변경 사항 감사 레코드를 찾는 방법"
감사 파일의 크기를 줄이려면 어떻게 합니까?	이진 감사 파일의 크기를 제한합니다.	597 페이지 "이진 감사 파일의 크기를 제한하는 방법"
감사 파일에 대한 파일 시스템 공간을 덜 사용하려면 어떻게 합니까?	ZFS 할당량 및 압축을 사용합니다.	597 페이지 "전용 파일 시스템에서 감사 파일을 압축하는 방법"

문제점	해결 방법	수행 방법
audit_event 파일에서 감사 이벤트를 제거하려면 어떻게 합니까?	audit_event 파일을 올바르게 업데이트합니다.	596 페이지 "특정 이벤트의 감사를 막는 방법"
Oracle Solaris 시스템에 대한 모든 로그인을 감사하려면 어떻게 합니까?	모든 시스템에서 로그인을 감사합니다.	598 페이지 "다른 운영 체제에서 로그인을 감사하는 방법"
FTP 전송에 대한 감사 레코드가 기록되지 않는 이유는 무엇입니까?	고유의 로그를 생성하는 유틸리티에 대해 알맞은 감사 도구를 사용합니다.	599 페이지 "FTP 및 SFTP 파일 전송을 감사하는 방법"

▼ 감사가 실행 중인지 확인하는 방법

감사 기능은 기본적으로 활성화됩니다. 감사가 사용 안함으로 설정되지 않았지만 감사 레코드가 활성 플러그인으로 보내지지 않는다고 판단되는 경우 다음 절차를 사용하여 문제를 식별합니다.

시작하기 전에 시스템 파일을 수정하려면 root 역할을 가진 사용자여야 합니다. 감사를 구성하려면 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1 감사가 실행 중인지 확인합니다.

다음 방법 중 하나를 사용합니다.

- **현재 감사 조건을 확인합니다.**

  다음 목록은 감사가 실행 중이 아님을 나타냅니다.

  ```
  # auditconfig -getcond
  audit condition = noaudit
  ```

  다음 목록은 감사가 실행 중임을 나타냅니다.

  ```
  # auditconfig -getcond
  audit condition = auditing
  ```

- **감사 서비스가 실행 중인지 확인합니다.**

  다음 목록은 감사가 실행 중이 아님을 나타냅니다.

  ```
  # svcs -x auditd
  svc:/system/auditd:default (Solaris audit daemon)
   State: disabled since Sun Oct 10 10:10:10 2010
  Reason: Disabled by an administrator.
     See: http://sun.com/msg/SMF-8000-05
     See: auditd(1M)
     See: audit(1M)
     See: auditconfig(1M)
     See: audit_flags(5)
     See: audit_binfile(5)
     See: audit_syslog(5)
     See: audit_remote(5)
     See: /var/svc/log/system-auditd:default.log
  Impact: This service is not running.
  ```

다음 목록은 감사 서비스가 실행 중임을 나타냅니다.

```
# svcs auditd
STATE          STIME    FMRI
online         10:10:10 svc:/system/auditd:default
```

감사 서비스가 실행 중이 아닌 경우 사용으로 설정합니다. 절차는 575 페이지 "감사 서비스를 사용으로 설정하는 방법"을 참조하십시오.

2 적어도 하나의 플러그인이 활성화되었는지 확인합니다.

```
# audit -v
```

활성화된 플러그인이 없는 경우 활성화합니다.

```
# auditconfig -setplugin audit_binfile active
```

3 사용자 정의된 감사 클래스를 만든 경우 클래스에 이벤트를 지정했는지 확인합니다.

예를 들어, 다음 플래그 목록은 Oracle Solaris 소프트웨어에서 제공하지 않은 pf 클래스를 포함합니다.

```
# auditconfig -getflags
active user default audit flags = pf,lo(0x0100000000000000,0x0100000000001000)
configured user default audit flags = pf,lo(0x0100000000000000,0x0100000000001000)
```

pf 클래스를 만드는 자세한 내용은 557 페이지 "감사 클래스를 추가하는 방법"을 참조하십시오.

a. 클래스가 audit_class 파일에서 정의되었는지 확인합니다.

감사 클래스가 정의되고 해당 마스크는 고유해야 합니다.

```
# grep pf /etc/security/audit_class      Verify class exists
0x0100000000000000:pf:profile
# grep 0x08000000 /etc/security/audit_class      Ensure mask is unique
0x0100000000000000:pf:profile
```

고유하지 않은 마스크를 바꿉니다. 클래스가 정의되지 않은 경우 정의합니다. 그렇지 않으면 auditconfig -setflags 명령을 유효한 값과 함께 실행하여 현재 플래그를 재설정합니다.

b. 이벤트가 클래스에 지정되었는지 확인합니다.

다음 방법 중 하나를 사용합니다.

```
# auditconfig -lsevent | egrep " pf|,pf|pf,"
AUE_PFEXEC      116 pf execve(2) with pfexec enabled

# auditrecord -c pf
     List of audit events assigned to pf class
```

이벤트가 클래스에 지정되지 않은 경우 적당한 이벤트를 이 클래스에 지정합니다.

4 **이전 단계에서 문제가 나타나지 않은 경우 전자 메일 및 로그 파일을 검토합니다.**

a. **audit_warn 별칭으로 전송된 전자 메일을 읽습니다.**

audit_warn 스크립트는 경고 메시지를 audit_warn 전자 메일 별칭으로 보냅니다. 올바르게 구성된 별칭이 없을 경우 메시지가 root 별칭으로 보내집니다.

b. **감사 서비스에 대한 로그 파일을 검토합니다.**

svcs -s auditd 명령의 출력은 감사 서비스에서 생성하는 감사 로그에 대한 전체 경로를 나열합니다. 예는 단계 1의 목록을 참조하십시오.

c. **시스템 로그 파일을 검토합니다.**

audit_warn 스크립트는 daemon.alert 메시지를 /var/log/syslog 파일에 씁니다.

/var/adm/messages 파일에는 정보가 포함되어 있을 수 있습니다.

5 **문제를 찾아 수정한 다음 감사 서비스를 사용으로 설정하거나 다시 시작합니다.**

```
# audit -s
```

▼ 생성되는 감사 레코드의 양을 줄이는 방법

사이트에서 감사되어야 하는 이벤트를 결정한 후 다음 제안에 따라 관리 가능한 감사 파일을 만듭니다.

시작하기 전에 감사 클래스를 사전 선택하고 감사 정책을 설정하려면 Audit Configuration 권한 프로파일이 지정되어야 합니다. 시스템 파일을 수정하고 감사 플래그를 사용자, 역할 및 권한 프로파일에 지정하려면 root 역할을 가진 사용자여야 합니다.

1 **기본 감사 정책을 사용합니다.**

특히, 이벤트 및 감사 토큰을 감사 추적에 추가하지 마십시오. 다음 정책은 감사 추적의 크기를 늘립니다.

- arge 정책 – execv 감사 이벤트에 환경 변수를 추가합니다.
- argv 정책 – execv 감사 이벤트에 명령 매개변수를 추가합니다.
- public 정책 – 파일 이벤트가 감사되는 경우 감사 가능한 이벤트가 공용 객체에 발생할 때마다 감사 추적에 이벤트를 추가합니다. 파일 클래스에는 fa, fc, fd, fm, fr, fw 및 cl이 포함됩니다. 공용 파일에 대한 정의는 520 페이지 “감사 용어 및 개념”을 참조하십시오.
- path 정책 – 선택적 path 토큰이 포함된 감사 이벤트에 path 토큰을 추가합니다.
- group 정책 – 선택적 newgroups 토큰이 포함된 감사 이벤트에 그룹 토큰을 추가합니다.
- seq 정책 – 모든 감사 이벤트에 시퀀스 토큰을 추가합니다.

- `trail` 정책 – 모든 감사 이벤트에 트레일러 토큰을 추가합니다.
- `windata_down` 정책 – Trusted Extensions로 구성된 시스템에서 레이블이 있는 창의 정보가 다운그레이드될 때 이벤트를 추가합니다.
- `windata_up` 정책 – Trusted Extensions로 구성된 시스템에서 레이블이 있는 창의 정보가 업그레이드될 때 이벤트를 추가합니다.
- `zonename` 정책 – 모든 감사 이벤트에 영역 이름을 추가합니다. 전역 영역이 유일하게 구성된 영역인 경우 모든 감사 이벤트에 `zone, global` 문자열을 추가합니다.

다음 감사 레코드는 `ls` 명령의 사용을 보여줍니다. `ex` 클래스가 감사되고 기본 정책을 사용 중입니다.

```
header,129,2,AUE_EXECVE,,mach1,2010-10-14 11:39:22.480 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
subject,jdoe,root,root,root,root,2404,50036632,82 0 mach1
return,success,0
```

다음은 모든 정책이 설정되었을 때 나타나는 동일한 레코드입니다.

```
header,1578,2,AUE_EXECVE,,mach1,2010-10-14 11:45:46.658 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
exec_args,2,ls,/etc/security
exec_env,49,MANPATH=/usr/share/man,USER=jdoe,GDM_KEYBOARD_LAYOUT=us,EDITOR=gedit,
  LANG=en_US.UTF-8,GDM_LANG=en_US.UTF-8,PS1=#,GDMSESSION=gnome,SESSIONTYPE=1,SHLVL=2,
  HOME=/home/jdoe,LOGNAME=jdoe,G_FILENAME_ENCODING=@locale,UTF-8, PRINTER=example-dbl,
...
path,/lib/ld.so.1
attribute,100755,root,bin,21,393073,18446744073709551615
subject,jdoe,root,root,root,root,2424,50036632,82 0 mach1
group,root,other,bin,sys,adm,uucp,mail,tty,lp,nuucp,daemon
return,success,0
zone,global
sequence,197
trailer,1578
```

2 `audit_syslog` 플러그인을 사용하여 일부 감사 이벤트를 `syslog`로 보냅니다.

그리고 이러한 감사 이벤트를 `audit_binfile` 또는 `audit_remote` 플러그인으로 보내지 않습니다. 이 전략은 `syslog` 로그로 보내는 감사 이벤트의 이진 레코드를 보관할 필요가 없을 경우에만 유효합니다.

3 더 적은 시스템 전역 감사 플래그를 설정하고 개별 사용자를 감사합니다.

시스템 전역으로 감사되는 감사 클래스의 수를 줄여 모든 사용자에 대한 감사의 양을 줄입니다.

`roleadd`, `rolemod`, `useradd` 및 `usermod` 명령에 `audit_flags` 키워드를 사용하여 특정 사용자 및 역할에 대한 이벤트를 감사합니다. 예는 예 28–18 및 `usermod(1M)` 매뉴얼 페이지를 참조하십시오.

profiles 명령의 always_audit 및 never_audit 등록 정보를 사용하여 특정 권한 프로파일에 대한 이벤트를 감사합니다. 자세한 내용은 profiles(1) 매뉴얼 페이지를 참조하십시오.

주 – 다른 보안 속성과 마찬가지로 감사 플래그는 검색 순서의 영향을 받습니다. 자세한 내용은 199 페이지 "지정된 보안 속성의 검색 순서"를 참조하십시오.

---

4 **고유의 사용자 정의된 감사 클래스를 만듭니다.**

해당 사이트에서 감사 클래스를 만들 수 있습니다. 모니터해야 하는 감사 이벤트만 이러한 감사 클래스에 추가합니다. 절차는 557 페이지 "감사 클래스를 추가하는 방법"을 참조하십시오.

주의 – 기존 감사 클래스 지정을 수정할 경우 최신 버전의 Oracle Solaris OS로 업그레이드할 때 이러한 수정 사항이 유지될 수 있습니다. 하지만 Oracle Solaris 파일의 최신 버전에는 수동으로 설치에 통합해야 하는 변경 사항이 포함될 수 있습니다. 설치 로그를 신중하게 검토하십시오. 자세한 내용은 pkg(5) 매뉴얼 페이지의 preserve=renamenew에 대한 설명을 참조하십시오.

▼ 사용자의 모든 명령을 감사하는 방법

보안 정책의 일부로 일부 사이트에서는 root 계정 및 관리 역할에서 실행하는 모든 명령의 감사 레코드를 요구합니다. 일부 사이트에서는 모든 사용자가 실행하는 모든 명령에 대한 감사 레코드를 요구할 수 있습니다. 또한 사이트에서는 명령 인수 및 환경이 기록되도록 요구할 수 있습니다.

시작하기 전에 감사 클래스를 사전 선택하고 감사 정책을 설정하려면 Audit Configuration 권한 프로파일이 지정되어야 합니다. 감사 플래그를 사용자, 역할 권한 프로파일에 지정하려면 root 역할을 가진 사용자여야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**

자세한 내용은 160 페이지 "관리 권한을 얻는 방법"을 참조하십시오.

2 **lo 및 ex 클래스를 감사합니다.**

ex 클래스는 exec() 및 execve() 함수에 대한 모든 호출을 감사합니다.

lo 클래스는 로그인, 로그아웃 및 화면 잠금을 감사합니다. 다음 출력은 ex 및 lo 클래스의 모든 이벤트를 나열합니다.

```
% auditconfig -lsevent | grep " lo "
AUE_login                       6152 lo login - local
AUE_logout                      6153 lo logout
AUE_telnet                      6154 lo login - telnet
AUE_rlogin                      6155 lo login - rlogin
```

```
AUE_rshd                        6158 lo rsh access
AUE_su                          6159 lo su
AUE_rexecd                      6162 lo rexecd
AUE_passwd                      6163 lo passwd
AUE_rexd                        6164 lo rexd
AUE_ftpd                        6165 lo ftp access
AUE_ftpd_logout                 6171 lo ftp logout
AUE_ssh                         6172 lo login - ssh
AUE_role_login                  6173 lo role login
AUE_newgrp_login                6212 lo newgrp login
AUE_admin_authenticate          6213 lo admin login
AUE_screenlock                  6221 lo screenlock - lock
AUE_screenunlock                6222 lo screenlock - unlock
AUE_zlogin                      6227 lo login - zlogin
AUE_su_logout                   6228 lo su logout
AUE_role_logout                 6229 lo role logout
AUE_smbd_session                6244 lo smbd(1m) session setup
AUE_smbd_logoff                 6245 lo smbd(1m) session logoff
AUE_ClientConnect               9101 lo client connection to x server
AUE_ClientDisconnect            9102 lo client disconn. from x server
% auditconfig -lsevent | egrep " ex |,ex |ex,"
AUE_EXECVE                        23 ex,ps execve(2)
```

- **관리 역할에 대해 이러한 클래스를 감사하려면 역할의 보안 속성을 수정합니다.**

  다음 예에서 root는 역할입니다. 사이트에서는 sysadm, auditadm 및 netadm의 세 역할을 만들었습니다. 모든 역할은 ex 및 lo 클래스에 있는 이벤트의 성공 및 실패에 대해 감사됩니다.

  ```
  # rolemod -K audit_flags=lo,ex:no root
  # rolemod -K audit_flags=lo,ex:no sysadm
  # rolemod -K audit_flags=lo,ex:no auditadm
  # rolemod -K audit_flags=lo,ex:no netadm
  ```

- **모든 사용자에 대해 이러한 클래스를 감사하려면 시스템 전역 플래그를 설정합니다.**

  ```
  # auditconfig -setflags lo,ex
  ```

  출력은 다음과 유사하게 나타납니다.

  ```
  header,129,2,AUE_EXECVE,,mach1,2010-10-14 12:17:12.616 -07:00
  path,/usr/bin/ls
  attribute,100555,root,bin,21,320271,18446744073709551615
  subject,jdoe,root,root,root,root,2486,50036632,82 0 mach1
  return,success,0
  ```

3. **명령에 대한 인수를 기록하려면 argv 정책을 추가합니다.**

   ```
   # auditconfig -setpolicy +argv
   ```

   exec_args 토큰은 명령 인수를 기록합니다.

   ```
   header,151,2,AUE_EXECVE,,mach1,2010-10-14 12:26:17.373 -07:00
   path,/usr/bin/ls
   attribute,100555,root,bin,21,320271,18446744073709551615
   exec_args,2,ls,/etc/security
   subject,jdoe,root,root,root,root,2494,50036632,82 0 mach1
   return,success,0
   ```

4 **명령이 실행되는 환경을 기록하려면 arge 정책을 추가합니다.**

```
# auditconfig -setpolicy +arge
```

exec_env 토큰은 명령 환경을 기록합니다.

```
header,1460,2,AUE_EXECVE,,mach1,2010-10-14 12:29:39.679 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
exec_args,2,ls,/etc/security
exec_env,49,MANPATH=/usr/share/man,USER=jdoe,GDM_KEYBOARD_LAYOUT=us,EDITOR=gedit,
LANG=en_US.UTF-8,GDM_LANG=en_US.UTF-8,PS1=#,GDMSESSION=gnome,SESSIONTYPE=1,SHLVL=2,
HOME=/home/jdoe,LOGNAME=jdoe,G_FILENAME_ENCODING=@locale,UTF-8,
PRINTER=example-dbl,...,_=/usr/bin/ls
subject,jdoe,root,root,root,root,2502,50036632,82 0 mach1
return,success,0
```

▼ 특정 파일에 대한 변경 사항 감사 레코드를 찾는 방법

목적이 /etc/passwd 및 /etc/default 디렉토리의 파일과 같이 제한된 수의 파일에 대한 파일 쓰기를 기록하는 것이라면 auditreduce 명령을 사용하여 파일을 찾습니다.

시작하기 전에 auditconfig 명령을 사용하려면 Audit Configuration 권한 프로파일이 지정되어야 합니다. auditreduce 명령을 사용하려면 Audit Review 권한 프로파일이 지정되어야 합니다. 감사 플래그를 사용자 및 역할에 지정하려면 root 역할을 가진 사용자여야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **fw 클래스를 감사합니다.**

사용자나 역할의 감사 플래그에 클래스를 추가하면 시스템 전역 감사 사전 선택 마스크에 클래스를 추가할 때보다 적은 레코드가 생성됩니다. 다음 단계 중 하나를 수행합니다.

- **fw 클래스를 특정 역할에 추가합니다.**

```
# rolemod -K audit_flags=fw:no root
# rolemod -K audit_flags=fw:no sysadm
# rolemod -K audit_flags=fw:no auditadm
# rolemod -K audit_flags=fw:no netadm
```

- **fw 클래스를 시스템 전역 플래그에 추가합니다.**

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
# auditconfig -setflags lo,fw
user default audit flags = lo,fw(0x1002,0x1002)
```

3 **또한 성공한 파일 쓰기를 감사합니다.**

성공을 감사하면 실패 및 성공을 감사할 때보다 적은 레코드가 생성됩니다. 다음 단계 중 하나를 수행합니다.

- **+fw 플래그를 특정 역할에 추가합니다.**

```
# rolemod -K audit_flags=+fw:no root
# rolemod -K audit_flags=+fw:no sysadm
# rolemod -K audit_flags=+fw:no auditadm
# rolemod -K audit_flags=+fw:no netadm
```

- **+fw 플래그를 시스템 전역 플래그에 추가합니다.**

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
# auditconfig -setflags lo,+fw
user default audit flags = lo,+fw(0x1002,0x1000)
```

- **시스템 전역 플래그가 성공 및 실패에 대해 감사하는 경우 특정 사용자 및 역할에 대한 예외 사항을 설정합니다.**

```
# auditconfig -getflags
active user default audit flags = lo,fw(0x1002,0x1002)
configured user default audit flags = lo,fw(0x1002,0x1002)
# rolemod -K audit_flags=^-fw:no root
# rolemod -K audit_flags=^-fw:no sysadm
# rolemod -K audit_flags=^-fw:no auditadm
# rolemod -K audit_flags=^-fw:no netadm
```

시스템 전역 플래그는 변경되지 않았지만 이러한 네 역할에 대한 사전 선택 마스크가 변경되었습니다.

```
# auditconfig -getflags
active user default audit flags = lo,fw(0x1002,0x1000)
configured user default audit flags = lo,fw(0x1002,0x1000)
```

4 **특정 파일에 대한 감사 레코드를 찾으려면 auditreduce 명령을 사용합니다.**

```
# auditreduce -o file=/etc/passwd,/etc/default -O filechg
```

auditreduce 명령은 file 인수의 모든 인스턴스에 대한 감사 추적을 검색합니다. 명령은 관심 파일의 경로 이름이 포함된 모든 레코드를 포함하는 filechg 접미어의 이진 파일을 만듭니다. -o file=*pathname* 옵션의 구문은 auditreduce(1M) 매뉴얼 페이지를 참조하십시오.

5 **filechg 파일을 찾으려면 praudit 명령을 사용합니다.**

```
# praudit *filechg
```

▼ 로그인한 사용자의 사전 선택 마스크를 업데이트하는 방법

이미 로그인한 사용자가 시스템 전역 감사 사전 선택 마스크의 변경 사항에 대해 감사되도록 하고자 합니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다. 사용자 세션을 종료하려면 Process Management 권한 프로파일이 지정되어야 합니다.

1 필요한 보안 속성을 가진 관리자가 됩니다.

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 이미 로그인한 사용자의 사전 선택 마스크를 업데이트합니다.

두 가지 옵션이 있습니다. 기존 세션을 종료하거나 `auditconfig` 명령을 사용하여 사전 선택 마스크를 업데이트할 수 있습니다.

- **사용자의 기존 세션을 종료합니다.**

  사용자는 로그아웃하고 다시 로그인할 수 있습니다. 또는 Process Management 권한 프로파일이 지정된 역할을 가진 사용자가 수동으로 활성 세션을 종료할 수 있습니다. 새 세션은 새로운 사전 선택 마스크를 상속합니다. 하지만 사용자 세션을 종료하는 것은 실용적이지 않을 수 있습니다.

- **각 로그인한 사용자의 사전 선택 마스크를 동적으로 변경합니다.**

  Audit Configuration 권한 프로파일이 지정된 역할을 가진 사용자가 시스템 전역 감사 사전 선택 마스크를 `lo`에서 `lo,ex`로 변경했다고 가정합니다.

  ```
  # auditconfig -setflags lo,ex
  ```

  a. 로그인한 일반 사용자 및 프로세스 ID를 나열합니다.

  ```
  # who -a
  jdoe  - vt/2         Jan 25 07:56  4:10   1597   (:0)
  jdoe  + pts/1        Jan 25 10:10   .     1706   (:0.0)
  ...
  jdoe  + pts/2        Jan 25 11:36  3:41   1706   (:0.0)
  ```

  b. 나중에 비교를 위해 각 사용자의 사전 선택 마스크를 표시합니다.

  ```
  # auditconfig -getpinfo 1706
  audit id = jdoe(1234)
  process preselection mask = lo(0x1000,0x1000)
  terminal id (maj,min,host) = 9426,65559,mach1(192.168.123.234)
  audit session id = 103203403
  ```

c. 사용자의 사전 선택 마스크를 수정합니다.

```
# auditconfig -setumask jdoe lo,ex       /* for this user */

# auditconfig -setsmask 103203403 lo,ex       /* for this session */

# auditconfig -setpmask 1706 lo,ex       /* for this process */
```

d. 사용자에 대한 사전 선택 마스크가 변경되었는지 확인합니다.

예를 들어, 마스크를 변경하기 전에 있었던 프로세스를 확인합니다.

```
# auditconfig -getpinfo 1706
audit id = jdoe(1234)
process preselection mask = ex,lo(0x40001000,0x40001000)
terminal id (maj,min,host) = 9426,65559,mach1(192.168.123.234)
audit session id = 103203403
```

▼ 특정 이벤트의 감사를 막는 방법

유지 관리를 목적으로 때때로 사이트에서 이벤트가 감사되지 않도록 막을 수 있습니다.

시작하기 전에 root 역할을 가진 사용자여야 합니다.

1 이벤트의 클래스를 no 클래스로 변경합니다.

예를 들어, 이벤트 26 및 27은 pm 클래스에 속해 있습니다.

```
## audit_event file
...
25:AUE_VFORK:vfork(2):ps
26:AUE_SETGROUPS:setgroups(2):pm
27:AUE_SETPGRP:setpgrp(2):pm
28:AUE_SWAPON:swapon(2):no
...
```

이러한 이벤트를 no 클래스로 변경합니다.

```
## audit_event file
...
25:AUE_VFORK:vfork(2):ps
26:AUE_SETGROUPS:setgroups(2):no
27:AUE_SETPGRP:setpgrp(2):no
28:AUE_SWAPON:swapon(2):no
...
```

pm 클래스가 현재 감사되는 경우 기존 세션에서는 이벤트 26 및 27을 여전히 감사합니다. 이러한 이벤트의 감사를 중지하려면 595 페이지 "로그인한 사용자의 사전 선택 마스크를 업데이트하는 방법"의 지침을 따라 사용자의 사전 선택 마스크를 업데이트해야 합니다.

주의 – audit_event 파일에서 이벤트를 주석 처리하지 마십시오. 이 파일은 praudit 명령에서 이진 감사 파일을 읽는 데 사용됩니다. 아카이브된 감사 파일은 파일에 나열된 이벤트를 포함할 수 있습니다.

2 **커널 이벤트를 새로 고칩니다.**

```
# auditconfig -conf
Configured 283 kernel events.
```

▼ 이진 감사 파일의 크기를 제한하는 방법

이진 감사 파일은 무제한으로 커집니다. 아카이브 및 검색을 용이하게 하기 위해 크기를 제한할 수 있습니다. 또한 원본 파일에서 더 작은 이진 파일을 만들 수도 있습니다.

시작하기 전에 p_fsize 속성을 설정하려면 Audit Configuration 권한 프로파일이 지정되어야 합니다. auditreduce 명령을 사용하려면 Audit Review 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **p_fsize 속성을 사용하여 개별 이진 감사 파일의 크기를 제한합니다.**

p_fsize 속성에 대한 설명은 audit_binfile(5) 매뉴얼 페이지의 OBJECT ATTRIBUTES 섹션을 참조하십시오.

예는 예 28–14를 참조하십시오.

3 **추가 분석을 위해 auditreduce 명령을 사용하여 레코드를 선택하고 이러한 레코드를 더 작은 파일에 씁니다.**

auditreduce -*lowercase* 옵션은 특정 레코드를 찾습니다.

auditreduce -*Uppercase* 옵션은 선택 항목을 파일에 씁니다. 자세한 내용은 auditreduce(1M) 매뉴얼 페이지를 참조하십시오.

▼ 전용 파일 시스템에서 감사 파일을 압축하는 방법

감사 파일은 커질 수 있습니다. 예 28–14에 나온 대로 파일 크기에 대한 상한을 설정할 수 있습니다. 이 절차에서는 압축을 사용하여 크기를 줄입니다.

시작하기 전에 ZFS File System Management 및 ZFS Storage Management 권한 프로파일이 지정되어야 합니다. ZFS Storage Management 권한 프로파일을 사용하여 저장소 풀을 만들 수 있습니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**
자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **감사 파일에 대한 전용 ZFS 파일 시스템을 만듭니다.**
절차는 560 페이지 “감사 파일에 대한 ZFS 파일 시스템을 만드는 방법”을 참조하십시오.

3 **다음 옵션 중 하나를 사용하여 ZFS 저장소 풀을 압축합니다.**
두 옵션 모두 감사 파일 시스템을 압축합니다. 감사 서비스를 새로 고치면 압축률이 표시됩니다.

압축을 설정하려면 `zfs set compression=on` *dataset* 명령을 사용합니다. 다음 예에서 ZFS 풀 `auditp/auditf`는 데이터 집합입니다.

- **기본 압축 알고리즘을 사용합니다.**

```
# zfs set compression=on auditp/auditf
# audit -s
# zfs get compressratio auditp/auditf
NAME           PROPERTY       VALUE  SOURCE
auditp/auditf  compressratio  4.54x  -
```

- **더 높은 압축 알고리즘을 사용합니다.**

```
# zfs set compression=gzip-9 auditp/auditf
# zfs get compression auditp/auditf
NAME           PROPERTY     VALUE     SOURCE
auditp/auditf  compression  gzip-9    local
# audit -s
# zfs get compressratio auditp/auditf
NAME           PROPERTY       VALUE  SOURCE
auditp/auditf  compressratio  16.89x  -
```

`gzip-9` 압축 알고리즘을 사용하면 기본 압축 알고리즘인 `lzjb`보다 1/3 적은 공간을 차지하는 파일이 생성됩니다. 자세한 내용은 **Oracle Solaris 관리: ZFS 파일 시스템**의 6장, “Oracle Solaris ZFS 파일 시스템 관리”를 참조하십시오.

▼ 다른 운영 체제에서 로그인을 감사하는 방법

Oracle Solaris OS는 소스와 상관없이 모든 로그인을 감사할 수 있습니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**
자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **지정 가능한 이벤트 및 지정 불가능한 이벤트에 대해 lo 클래스를 감사합니다.**

이 클래스는 로그인, 로그아웃 및 화면 잠금을 감사합니다. 이러한 클래스는 기본적으로 감사됩니다.

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

3 **값이 변경된 경우 lo 플래그를 추가합니다.**

```
# auditconfig -getflags
active user default audit flags = as,st(0x20800,0x20800)
configured user default audit flags = as,st(0x20800,0x20800)
# auditconfig -setflags lo,as,st
user default audit flags = as,lo,st(0x21800,0x21800)
# auditconfig -getnaflags
active non-attributable audit flags = na(0x400,0x400)
configured non-attributable audit flags = na(0x400,0x400)
# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```

주 – ssh 로그인을 감사하려면 시스템에서 Oracle Solaris의 ssh 데몬을 실행하고 있어야 합니다. 이 데몬은 Oracle Solaris 시스템에서 감사 서비스에 대해 수정됩니다. 자세한 내용은 295 페이지 “Secure Shell 및 OpenSSH 프로젝트”를 참조하십시오.

▼ FTP 및 SFTP 파일 전송을 감사하는 방법

FTP 서비스는 파일 전송 로그를 만듭니다. ssh 프로토콜로 실행되는 SFTP 서비스는 ft 감사 클래스를 사전 선택하여 감사할 수 있습니다. 두 서비스에 대한 로그인을 감사할 수 있습니다.

시작하기 전에 Audit Configuration 권한 프로파일이 지정되어야 합니다.

1 **필요한 보안 속성을 가진 관리자가 됩니다.**

자세한 내용은 160 페이지 “관리 권한을 얻는 방법”을 참조하십시오.

2 **FTP 서비스의 명령 및 파일 전송을 기록하려면 proftpd(8) 매뉴얼 페이지를 참조하십시오.**

사용 가능한 로깅 옵션은 “Logging Capabilities” 절을 참조하십시오. 특히, log commands 및 log transfers 옵션은 유용한 로그를 제공할 수 있습니다.

3 **sftp 액세스 및 파일 전송을 기록하려면 ft 클래스를 감사합니다.**

ft 클래스에는 다음 SFTP 트랜잭션이 포함됩니다.

```
% auditrecord -c ft
file transfer: chmod ...
```

```
file transfer: chown ...
file transfer: download ...
file transfer: mkdir ...
file transfer: upload ...
file transfer: remove ...
file transfer: rename ...
file transfer: rmdir ...
file transfer: session start ...
file transfer: session end ...
file transfer: symlink ...
file transfer: utimes
```

4 **FTP 서버에 대한 액세스를 기록하려면 lo 클래스를 감사합니다.**

다음 출력에 나온 대로 ftpd 데몬의 로그인 및 로그아웃으로 감사 레코드가 생성됩니다.

```
% auditrecord -c lo | more
...
in.ftpd
  program     /usr/sbin/in.ftpd    See ftp access
  event ID    6165                 AUE_ftpd
  class       lo                   (0x0000000000001000)
      subject
      [text]                       error message
      return

in.ftpd
  program     /usr/sbin/in.ftpd    See ftp logout
  event ID    6171                 AUE_ftpd_logout
  class       lo                   (0x0000000000001000)
      subject
      return
...
```

◆ ◆ ◆ 29 장

29 감사(참조)

이 장에서는 감사의 중요한 구성 요소에 대해 설명합니다. 다음은 이 장에 포함된 참조 정보 목록입니다.

감사에 대한 개요는 26 장, “감사(개요)”를 참조하십시오. 계획 제안은 27 장, “감사 계획”을 참조하십시오. 사이트에서 감사 구성 절차는 28 장, “감사 관리(작업)”를 참조하십시오.

감사 서비스

감사 서비스 `auditd`는 기본적으로 사용으로 설정됩니다. 서비스를 사용으로 설정하거나 새로 고치거나 사용 안함으로 설정하려면 572 페이지 “감사 서비스를 사용/사용 안함으로 설정(작업)”을 참조하십시오.

사용자 구성이 없으면 다음 기본값이 설정됩니다.

- 모든 로그인 이벤트가 감사됩니다.

  성공 및 실패 로그인 시도가 모두 감사됩니다.

- 모든 사용자가 로그인 및 로그아웃 이벤트(역할 맡기 및 화면 잠금 포함)에 대해 감사됩니다.
- audit_binfile 플러그인이 활성화됩니다. /var/audit 디렉토리가 감사 레코드를 저장하고, 감사 파일의 크기는 무제한이며, 대기열 크기는 레코드 100개입니다.
- cnt 정책이 설정됩니다.

  감사 레코드가 사용 가능한 디스크 공간을 채우면 시스템에서 삭제된 감사 레코드의 수를 추적합니다. 사용 가능한 디스크 공간의 1%가 남으면 경고가 발생합니다.
- 다음 감사 대기열 제어가 설정됩니다.
  - 레코드 잠금을 생성하기 전 감사 대기열의 최대 레코드 수 - 100
  - 차단된 감사 프로세스가 차단 해제되기 전 감사 대기열의 최대 레코드 수 - 10
  - 감사 대기열에 대한 버퍼 크기 - 8192바이트
  - 감사 추적에 감사 레코드 쓰기 간격 - 20초

기본값을 표시하려면 547 페이지 “감사 서비스 기본값을 표시하는 방법”을 참조하십시오.

감사 서비스를 사용하여 임시(또는 활성) 값을 설정할 수 있습니다. 이러한 값은 구성된(또는 등록 정보) 값과 다를 수 있습니다.

- 임시 값은 감사 서비스를 새로 고치거나 다시 시작할 때 복원되지 않습니다.

  감사 정책 및 감사 대기열 제어는 임시 값을 사용할 수 있습니다. 감사 플래그에는 임시 값이 없습니다.
- 구성된 값은 서비스의 등록 정보 값으로 저장되므로 감사 서비스를 새로 고치거나 다시 시작할 때 복원됩니다.

권한 프로파일은 감사 서비스를 관리할 수 있는 사용자를 제어합니다. 자세한 내용은 604 페이지 “감사 관리를 위한 권한 프로파일”을 참조하십시오.

기본적으로 모든 영역은 동일하게 감사됩니다. 604 페이지 “감사 및 Oracle Solaris 영역”을 참조하십시오.

감사 서비스 매뉴얼 페이지

다음 표에서는 감사 서비스에 대한 주요 관리 매뉴얼 페이지를 요약합니다.

매뉴얼 페이지	요약
audit(1M)	감사 서비스의 작업을 제어하는 명령입니다. audit -n은 audit_binfile 플러그인에 대한 새로운 감사 파일을 시작합니다. audit -s는 감사를 사용으로 설정하고 새로 고칩니다. audit -t는 감사를 사용 안함으로 설정합니다. audit -v는 적어도 하나의 플러그인이 활성화되었는지 확인합니다.
audit_binfile(5)	감사 레코드를 이진 파일로 보내는 기본 감사 플러그인입니다. 606 페이지 "감사 플러그인"도 참조하십시오.
audit_remote(5)	감사 레코드를 원격 수신자에게 보내는 감사 플러그인입니다.
audit_syslog(5)	감사 레코드의 텍스트 요약을 syslog 유틸리티로 보내는 감사 플러그인입니다.
audit_class(4)	감사 클래스의 정의를 포함하는 파일입니다. 8개 상위 순서 비트는 고객이 새 감사 클래스를 만드는 데 사용할 수 있습니다. 시스템 업그레이드 시 이 파일 수정의 효과는 557 페이지 "감사 클래스를 추가하는 방법"을 참조하십시오.
audit_event(4)	감사 이벤트의 정의를 포함하고 이벤트를 감사 클래스에 매핑하는 파일입니다. 매핑은 수정할 수 있습니다. 시스템 업그레이드 시 이 파일 수정의 효과는 558 페이지 "감사 이벤트의 클래스 멤버쉽을 변경하는 방법"을 참조하십시오.
audit_flags(5)	감사 클래스 사전 선택의 구문, 실패한 이벤트만 또는 성공한 이벤트만 선택하기 위한 접두어 및 기존 사전 선택을 수정하는 접두어를 설명합니다.
audit.log(4)	이진 감사 파일의 이름 지정, 파일의 내부 구조 및 모든 감사 토큰의 구조를 설명합니다.
audit_warn(1M)	감사 서비스에서 감사를 작성할 때 비정상적인 조건을 발견할 경우 전자 메일 별칭을 알려주는 스크립트입니다. 사이트에 대해 이 스크립트를 사용자 정의하여 수동 개입이 필요할 수 있는 조건을 경고할 수 있습니다. 또는 이러한 조건을 자동으로 처리할 방법을 지정할 수 있습니다.
auditconfig(1M)	감사 구성 매개변수를 검색하고 설정하는 명령입니다. 검색하고 설정할 수 있는 매개변수 목록을 표시하려면 auditconfig를 옵션 없이 입력합니다.
auditrecord(1M)	/etc/security/audit_event 파일의 감사 이벤트 정의를 표시하는 명령입니다. 샘플은 577 페이지 "감사 레코드 정의를 표시하는 방법"을 참조하십시오.
auditreduce(1M)	이진 형식으로 저장된 감사 레코드를 사후 선택하고 병합하는 명령입니다. 명령은 하나 이상의 입력 감사 파일에서 감사 레코드를 병합할 수 있습니다. 레코드는 이진 형식으로 유지됩니다. 대문자 옵션은 파일 선택에 영향을 미칩니다. 소문자 옵션은 레코드 선택에 영향을 미칩니다.

매뉴얼 페이지	요약
auditstat(1M)	커널 감사 통계를 표시하는 명령입니다. 예를 들어, 명령은 커널 감사 대기열의 레코드 수, 삭제된 레코드 수 및 사용자 프로세스가 시스템 호출 결과로 커널에서 생성한 감사 레코드 수를 표시할 수 있습니다.
praudit(1M)	표준 입력에서 이진 형식의 감사 레코드를 읽고 사전 선택 가능한 형식으로 레코드를 표시하는 명령입니다. 입력은 auditreduce 명령 또는 단일 감사 파일이나 감사 파일 목록에서 파이프할 수 있습니다. 또한 입력은 현재 감사 파일에 대해 tail -0f 명령으로 생성할 수 있습니다. 샘플 출력은 582 페이지 “이진 감사 파일의 내용을 보는 방법”을 참조하십시오.
syslog.conf(4)	audit_syslog 플러그인에 대해 감사 레코드의 텍스트 요약을 syslog 유틸리티로 보내도록 구성된 파일입니다.

감사 관리를 위한 권한 프로파일

Oracle Solaris는 감사 서비스 구성, 서비스 사용/사용 안함으로 설정 및 감사 추적 분석을 위한 권한 프로파일을 제공합니다. 감사 구성 파일을 편집하려면 root의 권한이 필요합니다.

- **Audit Configuration** – 관리자가 감사 서비스의 매개변수를 구성하고 auditconfig 명령을 실행할 수 있도록 합니다.
- **Audit Control** – 관리자가 감사 서비스를 시작, 새로 고침 및 사용 안함으로 설정하고 audit 명령을 실행하여 서비스를 시작, 새로 고침 또는 중지할 수 있도록 합니다.
- **Audit Review** – 관리자가 감사 레코드를 분석할 수 있도록 합니다. 이 권한 프로파일은 praudit 및 auditreduce 명령으로 감사 레코드를 읽을 수 있는 권한을 부여합니다. 또한 이 관리자는 auditstat 명령을 실행할 수 있습니다.
- **System Administrator** – Audit Review 권한 프로파일을 포함합니다. System Administrator 권한 프로파일을 가진 관리자는 감사 레코드를 분석할 수 있습니다.

감사 서비스를 처리할 역할을 구성하려면 163 페이지 “RBAC 초기 구성(작업 맵)”을 참조하십시오.

감사 및 Oracle Solaris 영역

비전역 영역은 전역 영역과 동일하게 감사하거나 고유의 플래그, 저장소 및 감사 정책을 설정할 수 있습니다.

모든 영역이 동일하게 감사되는 경우 전역 영역의 audit_class 및 audit_event 파일이 모든 영역에서 감사를 위한 클래스-이벤트 매핑을 제공합니다. +zonename 정책 옵션은 영역 이름으로 레코드를 사후 선택하는 데 유용합니다.

영역은 개별적으로 감사할 수도 있습니다. 정책 옵션 perzone이 전역 영역에서 설정된 경우 각 비전역 영역은 고유의 감사 서비스를 실행하고, 고유의 감사 대기열을 처리하며,

해당 감사 레코드의 내용과 위치를 지정합니다. 또한 비전역 영역은 대부분의 감사 정책 옵션을 설정할 수 있습니다. 전체 시스템에 영향을 미치는 정책은 설정할 수 없으므로 비전역 영역은 `ahlt` 또는 `perzone` 정책을 설정할 수 없습니다. 자세한 내용은 530 페이지 “Oracle Solaris 영역이 있는 시스템에 대한 감사” 및 534 페이지 “영역에서 감사를 계획하는 방법”을 참조하십시오.

영역에 대한 자세한 내용은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 제II부, “Oracle Solaris Zones”을 참조하십시오.

감사 클래스

Oracle Solaris는 많은 수의 감사 이벤트에 대한 편리한 컨테이너로 감사 클래스를 정의합니다.

감사 클래스를 재구성하고 새 감사 클래스를 만들 수 있습니다. 감사 클래스 이름은 최대 8자까지 가능합니다. 클래스 설명은 72자로 제한됩니다. 숫자 및 영숫자 이외의 문자가 허용됩니다. 자세한 내용은 `audit_class(4)` 매뉴얼 페이지 및 557 페이지 “감사 클래스를 추가하는 방법”을 참조하십시오.

주의 – `all` 클래스는 많은 양의 데이터를 생성하고 디스크를 빠르게 채울 수 있습니다. 모든 작업을 감사해야 하는 특별한 이유가 있을 경우에만 `all` 클래스를 사용하십시오.

감사 클래스 구문

감사 클래스의 이벤트는 성공, 실패 또는 둘 다에 대해 감사할 수 있습니다.

- 접두어가 없으면 이벤트 클래스가 성공 및 실패에 대해 감사됩니다.
- 더하기(+) 접두어가 있으면 이벤트 클래스가 성공에 대해서만 감사됩니다.
- 빼기(-) 접두어가 있으면 이벤트 클래스가 실패에 대해서만 감사됩니다.
- 접두어나 감사 플래그 앞에 캐럿(^)이 있으면 현재 사전 선택이 수정된 것입니다. 예를 들면 다음과 같습니다.
  - `ot`가 시스템에 대해 사전 선택되고 사용자의 사전 선택이 `^ot`인 경우 해당 사용자는 `other` 클래스의 이벤트에 대해 감사되지 않습니다.
  - `+ot`가 시스템에 대해 사전 선택되고 사용자의 사전 선택이 `^+ot`인 경우 해당 사용자는 `other` 클래스의 성공 이벤트에 대해 감사되지 않습니다.
  - `-ot`가 시스템에 대해 사전 선택되고 사용자의 사전 선택이 `^-ot`인 경우 해당 사용자는 `other` 클래스의 실패 이벤트에 대해 감사되지 않습니다.

감사 클래스 사전 선택 구문을 검토하려면 `audit_flags(5)` 매뉴얼 페이지를 참조하십시오.

감사 클래스 및 해당 접두어는 다음 명령에서 지정할 수 있습니다.

- `auditconfig` 명령 옵션 `-setflags` 및 `-setnaflags`에 대한 인수로 지정합니다.
- `audit_syslog` 플러그인의 `p_flags` 속성에 대한 값으로 지정합니다. `auditconfig -setplugin audit_syslog active` 명령에 대한 옵션으로 속성을 지정합니다.
- `useradd`, `usermod`, `roleadd` 및 `rolemod` 명령의 `-K audit_flags=`*always-audit-flags*`:`*never-audit-flags* 옵션에 대한 값으로 지정합니다.
- `profiles` 명령의 `-always_audit` 및 `-never_audit` 등록 정보에 대한 값으로 지정합니다.

감사 플러그인

감사 플러그인은 감사 대기열의 감사 레코드를 어떻게 처리할지 지정합니다. 감사 플러그인은 `audit_binfile`, `audit_remote` 및 `audit_syslog` 이름을 사용하여 `auditconfig -setplugin` 명령에 대한 인수로 지정됩니다. 플러그인은 다음 속성으로 추가 지정할 수 있습니다.

- `audit_binfile` 플러그인

  이진 데이터를 보낼 위치 - `p_dir` 속성

  관리자가 경고를 받기 전 디스크에 남은 최소 공간 - `p_minfree` 속성

  감사 파일의 최대 크기 - `p_fsize` 속성
- `audit_remote` 플러그인

  이진 감사 데이터를 보낼 원격 인증된 감사 서버 - `p_hosts` 속성

  원격 인증된 감사 서버에 접근하기 위한 시도 횟수 - `p_retries` 속성

  원격 인증된 감사 서버에 접근하기 위한 시도 간격(초) - `p_timeout` 속성
- `audit_syslog` 플러그인

  `syslog`로 보낼 감사 레코드의 텍스트 요약 선택 - `p_flags` 속성
- (모든 플러그인에 대상) 플러그인에 대해 대기되는 최대 감사 레코드 수 - `qsize` 속성

`audit_binfile(5)`, `audit_remote(5)`, `audit_syslog(5)` 및 `auditconfig(1M)` 매뉴얼 페이지를 참조하십시오.

감사 정책

감사 정책은 추가 정보가 감사 추적에 추가되는지 여부를 결정합니다.

`arge`, `argv`, `group`, `path`, `seq`, `trail`, `windata_down`, `windata_up` 및 `zonename` 정책은 감사 레코드에 토큰을 추가합니다. `windata_down` 및 `windata_up` 정책은 Oracle Solaris의 Trusted Extensions 기능에서 사용됩니다. 자세한 내용은 **Trusted Extensions 구성 및 관리**의 22 장, “Trusted Extensions 감사(개요)”를 참조하십시오.

나머지 정책은 토큰을 추가하지 않습니다. public 정책은 공용 파일의 감사를 제한합니다. perzone 정책은 비전역 영역에 대해 별도의 감사 대기열을 설정합니다. ahlt 및 cnt 정책은 감사 레코드를 전달할 수 없을 경우 어떻게 되는지 결정합니다. 자세한 내용은 607 페이지 "비동기 및 동기 이벤트에 대한 감사 정책"을 참조하십시오.

서로 다른 감사 정책 옵션의 효과는 538 페이지 "감사 정책 이해"를 참조하십시오. 감사 정책 옵션에 대한 설명은 auditconfig(1M) 매뉴얼 페이지의 -setpolicy 옵션을 참조하십시오. 사용 가능한 정책 옵션 목록을 표시하려면 auditconfig -lspolicy 명령을 실행합니다. 현재 정책을 표시하려면 auditconfig -getpolicy 명령을 실행합니다.

비동기 및 동기 이벤트에 대한 감사 정책

ahlt 정책과 cnt 정책은 함께 감사 대기열이 가득 차서 더 이상 이벤트를 수신할 수 없을 때 어떻게 되는지 제어합니다.

주 – 적어도 하나의 플러그인에 대한 대기열이 감사 레코드를 수신할 수 없으면 cnt 또는 ahlt 정책이 트리거되지 않습니다.

---

cnt 및 ahlt 정책은 서로 독립적이면서 관련되어 있습니다. 정책의 조합은 다음 효과를 가집니다.

- -ahlt +cnt는 제공되는 기본 정책입니다. 이 기본값은 이벤트를 기록할 수 없더라도 감사된 이벤트가 처리되도록 합니다.

  -ahlt 정책은 비동기 이벤트의 감사 레코드를 커널 감사 대기열에 둘 수 없는 경우 시스템에서 이벤트 수를 계산하고 처리를 계속하도록 지시합니다. 전역 영역에서 as_dropped 카운터가 수를 기록합니다.

  +cnt 정책은 동기 이벤트가 도착하고 이벤트를 커널 감사 대기열에 둘 수 없는 경우 시스템에서 이벤트 수를 계산하고 처리를 계속하도록 지시합니다. 영역의 as_dropped 카운터가 수를 기록합니다.

  -ahlt +cnt 구성은 처리를 계속하면 감사 레코드가 손실되더라도 처리를 계속해야 하는 사이트에서 일반적으로 사용됩니다. auditstat drop 필드는 영역에서 삭제된 감사 레코드의 수를 표시합니다.

- +ahlt -cnt 정책은 비동기 이벤트를 커널 감사 대기열에 추가할 수 없는 경우 처리를 정지하도록 지시합니다.

  +ahlt 정책은 비동기 이벤트의 감사 레코드를 커널 감사 대기열에 둘 수 없는 경우 모든 처리가 중지되도록 지시합니다. 시스템 패닉이 발생합니다. 비동기 이벤트가 감사 대기열에 들어가지 않으며 호출 스택의 포인터에서 복구해야 합니다.

  -cnt 정책은 동기 이벤트를 커널 감사 대기열에 둘 수 없는 경우 이벤트 전달을 시도하는 스레드가 차단되도록 지시합니다. 감사 공간을 사용할 수 있을 때까지 스레드는 일시 정지 대기열에 있습니다. 수가 계산되지 않습니다. 감사 공간을 사용할 수 있을 때까지 프로그램이 멈춘 것처럼 보일 수 있습니다.

+ahlt -cnt 구성은 모든 감사 이벤트의 레코드가 시스템 가용성보다 우선하는 사이트에서 일반적으로 사용됩니다. 감사 공간을 사용할 수 있을 때까지 프로그램이 멈춘 것처럼 보일 수 있습니다. auditstat wblk 필드는 스레드가 차단된 횟수를 표시합니다.

하지만 비동기 이벤트가 발생할 경우 시스템 패닉이 발생하고 더 이상 작동하지 않습니다. 감사 이벤트의 커널 대기열은 저장된 충돌 덤프에서 수동으로 복구할 수 있습니다. 비동기 이벤트가 감사 대기열에 들어가지 않으며 호출 스택의 포인터에서 복구해야 합니다.

- -ahlt -cnt 정책은 비동기 이벤트를 커널 감사 대기열에 둘 수 없는 경우 이벤트 수를 계산하고 처리를 계속하도록 지시합니다. 동기 이벤트를 커널 감사 대기열에 둘 수 없는 경우 이벤트 전달을 시도하는 스레드가 차단됩니다. 감사 공간을 사용할 수 있을 때까지 스레드는 일시 정지 대기열에 있습니다. 수가 계산되지 않습니다. 감사 공간을 사용할 수 있을 때까지 프로그램이 멈춘 것처럼 보일 수 있습니다.

  -ahlt -cnt 구성은 모든 동기 감사 이벤트의 기록이 비동기 감사 레코드의 일부 손실보다 우선하는 사이트에서 일반적으로 사용됩니다. auditstat wblk 필드는 스레드가 차단된 횟수를 표시합니다.

- +ahlt +cnt 정책은 비동기 이벤트를 커널 감사 대기열에 둘 수 없는 경우 시스템 패닉이 발생하도록 지시합니다. 동기 이벤트를 커널 감사 대기열에 둘 수 없는 경우 시스템에서 이벤트 수를 계산하고 처리를 계속합니다.

프로세스 감사 특성

다음 감사 특성은 최초 로그인 시 설정됩니다.

- **프로세스 사전 선택 마스크** – 사용자 감사 마스크가 지정된 경우 시스템 전역 감사 마스크와 사용자 특정 감사 마스크의 조합입니다. 사용자가 로그인하면 로그인 프로세스가 사전 선택된 클래스를 결합하여 사용자의 프로세스에 대한 **프로세스 사전 선택 마스크**를 설정합니다. 프로세스 사전 선택 마스크는 각 감사 클래스의 이벤트가 감사 레코드를 생성해야 하는지 여부를 지정합니다.

  다음 알고리즘은 시스템에서 사용자의 프로세스 사전 선택 마스크를 어떻게 얻는지 설명합니다.

  (system-wide default flags + *always-audit-classes*) - *never-audit-classes*

  auditconfig -getflags 명령 결과의 시스템 전역 감사 클래스를 사용자의 always_audit 키워드에 대한 *always-audit-classes* 값의 클래스에 추가합니다. 그런 다음 전체에서 사용자의 *never-audit-classes*의 클래스를 뺍니다. 또한 audit_flags(5) 매뉴얼 페이지를 참조하십시오.

- **감사 사용자 ID** – 사용자가 로그인할 때 프로세스는 변경 불가능한 감사 사용자 ID를 얻습니다. 이 ID는 사용자의 초기 프로세스로 시작된 모든 하위 프로세스에서 상속됩니다. 감사 사용자 ID는 책임을 적용하는 데 도움이 됩니다. 사용자가 역할을 맡은 후에도 감사 사용자 ID는 동일하게 유지됩니다. 각 감사 레코드에 저장된 감사 사용자 ID를 사용하여 특정 작업을 로그인 사용자로 항상 역추적할 수 있습니다.

- **감사 세션 ID** – 감사 세션 ID는 로그인 시 지정됩니다. 이 ID는 모든 하위 프로세스에서 상속됩니다.
- **터미널 ID** – 로컬 로그인의 경우, 터미널 ID는 로컬 시스템의 IP 주소와 사용자가 로그인한 물리적 장치를 식별하는 고유한 번호로 구성됩니다. 대부분의 경우 로그인은 콘솔을 통해 이루어집니다. 콘솔 장치에 해당하는 번호는 `0,0`입니다. 원격 로그인의 경우, 터미널 ID는 원격 호스트의 IP 주소와 원격 포트 번호 및 로컬 포트 번호로 구성됩니다.

감사 추적

감사 추적에는 이진 감사 파일이 포함됩니다. 추적은 `audit_binfile` 플러그인으로 만들어집니다. 감사 서비스는 감사 추적 레코드를 수집하고 플러그인에 보내면 플러그인이 디스크에 기록합니다.

이진 감사 파일 이름 지정 규칙

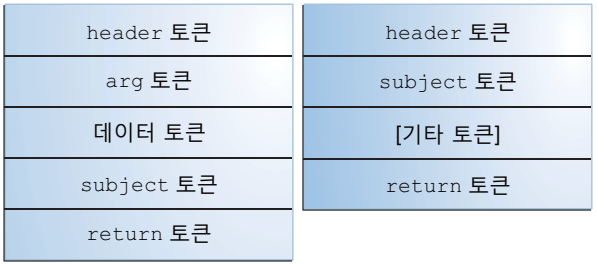
`audit_binfile` 플러그인은 이진 감사 파일을 만듭니다. 각 이진 감사 파일은 자체적으로 레코드의 모음입니다. 파일의 이름은 레코드가 생성된 기간과 레코드를 생성한 시스템을 식별합니다. 기간을 나타내는 시간 기록은 협정 세계시(UTC)로 지정되어 서로 다른 시간대에서도 올바른 순서로 정렬되도록 합니다.

자세한 내용은 `audit.log(4)` 매뉴얼 페이지를 참조하십시오. 열리고 닫힌 감사 파일 이름의 예는 584 페이지 “`not_terminated` 감사 파일을 정리하는 방법”을 참조하십시오.

감사 레코드 구조

감사 레코드는 감사 토큰의 시퀀스입니다. 각 감사 토큰에는 사용자 ID, 시간 및 날짜와 같은 이벤트 정보가 포함됩니다. `header` 토큰은 감사 레코드를 시작하고, 선택적 `trailer` 토큰은 레코드를 종료합니다. 기타 감사 토큰에는 감사 이벤트와 관련된 정보가 포함됩니다. 다음 그림은 일반적인 커널 감사 레코드 및 일반적인 사용자 레벨 감사 레코드를 보여줍니다.

그림 29–1 일반적인 감사 레코드 구조



감사 레코드 분석

감사 레코드 분석에는 감사 추적에서 레코드 사후 선택이 포함됩니다. 두 가지 방법 중 하나를 사용하여 수집된 이진 데이터를 구문 분석할 수 있습니다.

- `praudit` 명령을 사용할 수 있습니다. 명령에 대한 옵션은 서로 다른 텍스트 출력을 제공합니다. 예를 들어, `praudit -x` 명령은 스크립트 및 브라우저에 입력을 위한 XML을 제공합니다. `praudit` 출력에는 필드가 포함되지 않으며, 유일한 목적은 이진 데이터의 구문 분석을 돕는 것입니다. `praudit` 출력의 순서 및 형식은 Oracle Solaris 릴리스 사이에 보증되지 않습니다.

  `praudit` 출력의 예는 582 페이지 “이진 감사 파일의 내용을 보는 방법”을 참조하십시오.

  각 감사 토큰에 대한 `praudit` 출력의 예는 611 페이지 “감사 토큰 형식”의 개별 토큰을 참조하십시오.

- 이진 데이터 스트림 구문 분석을 위한 프로그램을 작성할 수 있습니다. 프로그램에서는 감사 레코드의 변형을 고려해야 합니다. 예를 들어, `ioctl()` 시스템 호출은 “잘못된 파일 이름”에 대한 감사 레코드를 만듭니다. 이 레코드에는 “잘못된 파일 설명자”에 대한 `ioctl()` 감사 레코드와 다른 토큰이 포함됩니다.
  - 각 감사 토큰에서 이진 데이터 순서에 대한 설명은 `audit.log(4)` 매뉴얼 페이지를 참조하십시오.
  - 매니페스트 값은 `/usr/include/bsm/audit.h` 파일을 참조하십시오.
  - 감사 레코드에서 토큰의 순서를 보려면 `auditrecord` 명령을 사용합니다. `auditrecord` 명령의 출력에는 서로 다른 매니페스트 값에 대해 서로 다른 토큰이 포함됩니다. 대괄호(`[]`)는 감사 토큰이 선택 사항임을 나타냅니다. 자세한 내용은 `auditrecord(1M)` 매뉴얼 페이지를 참조하십시오.

감사 토큰 형식

각 감사 토큰은 토큰 유형 식별자와 토큰에 대한 데이터로 구성됩니다. 다음 표는 각 토큰의 간략한 설명과 함께 토큰 이름을 나타냅니다. 더 이상 사용되지 않는 토큰은 이전 Solaris 릴리스와 호환성을 위해 유지됩니다.

표 29–1 감사에 대한 감사 토큰

토큰 이름	설명	자세한 정보
acl	액세스 제어 항목(ACE) 및 액세스 제어 목록(ACL) 정보	612 페이지 “acl 토큰”
arbitrary	형식 및 유형 정보가 있는 데이터	audit.log(4) 매뉴얼 페이지를 참조하십시오.
argument	시스템 호출 인수 값	613 페이지 “argument 토큰”
속성	파일 vnode 정보	613 페이지 “attribute 토큰”
cmd	명령 인수 및 환경 변수	613 페이지 “cmd 토큰”
exec_args	Exec 시스템 호출 인수	614 페이지 “exec_args 토큰”
exec_env	Exec 시스템 호출 환경 변수	614 페이지 “exec_env 토큰”
exit	프로그램 종료 정보	audit.log(4) 매뉴얼 페이지를 참조하십시오.
file	감사 파일 정보	614 페이지 “file 토큰”
fmri	프레임워크 관리 리소스 표시기	614 페이지 “fmri 토큰”
group	프로세스 그룹 정보	615 페이지 “group 토큰”
헤더	감사 레코드의 시작을 나타냄	615 페이지 “header 토큰”
ip	IP 헤더 정보	audit.log(4) 매뉴얼 페이지를 참조하십시오.
ip address	인터넷 주소	615 페이지 “ip address 토큰”
ip port	인터넷 포트 주소	616 페이지 “ip port 토큰”
ipc	시스템 V IPC 정보	616 페이지 “ipc 토큰”
IPC_perm	시스템 V IPC 객체 액세스 정보	617 페이지 “IPC_perm 토큰”
opaque	구조화되지 않은 데이터(지정되지 않은 형식)	audit.log(4) 매뉴얼 페이지를 참조하십시오.
path	경로 정보	617 페이지 “path 토큰”
path_attr	액세스 경로 정보	617 페이지 “path_attr 토큰”
권한	권한 집합 정보	617 페이지 “privilege 토큰”
프로세스	프로세스 정보	618 페이지 “process 토큰”

표 29–1 감사에 대한 감사 토큰 *(계속)*

토큰 이름	설명	자세한 정보
return	시스템 호출의 상태	618 페이지 “return 토큰”
sequence	시퀀스 번호	618 페이지 “sequence 토큰”
socket	소켓 유형 및 주소	618 페이지 “socket 토큰”
주체	주체 정보(process와 동일한 형식)	619 페이지 “subject 토큰”
text	ASCII 문자열	619 페이지 “text 토큰”
trailer	감사 레코드의 끝을 나타냄	619 페이지 “trailer 토큰”
use of authorization	권한 부여 사용	620 페이지 “use of authorization 토큰”
use of privilege	권한 사용	620 페이지 “use of privilege 토큰”
user	사용자 ID 및 사용자 이름	620 페이지 “user 토큰”
xclient	X 클라이언트 식별	620 페이지 “xclient 토큰”
zonename	영역의 이름	621 페이지 “zonename 토큰”
Trusted Extensions 토큰	label 및 X 창 시스템 정보	**Trusted Extensions 구성 및 관리**의 “Trusted Extensions 감사 참조”를 참조하십시오.

다음 토큰은 더 이상 사용되지 않습니다.

- liaison
- **호스트**
- tid

더 이상 사용되지 않는 토큰에 대한 정보는 해당 토큰이 포함된 릴리스의 참조 자료를 참조하십시오.

감사 레코드는 항상 header 토큰으로 시작됩니다. header 토큰은 감사 레코드가 감사 추적에서 시작되는 위치를 나타냅니다. 지정 가능한 이벤트의 경우 subject 및 process 토큰이 이벤트를 일으킨 프로세스의 값을 가리킵니다. 지정 불가능한 이벤트의 경우 process 토큰이 시스템을 가리킵니다.

acl 토큰

acl 토큰에는 ZFS 파일 시스템의 경우 액세스 제어 항목(ACE) 및 UFS 파일 시스템의 경우 액세스 제어 목록(ACL)에 대한 정보를 기록하기 위한 두 가지 형식이 있습니다.

acl 토큰이 UFS 파일 시스템에 대해 기록되는 경우 praudit -x 명령은 다음과 같이 필드를 표시합니다.

```
<acl type="1" value="root" mode="6"/>
```

acl 토큰이 ZFS 데이터 집합에 대해 기록되는 경우 praudit -x 명령은 다음과 같이 필드를 표시합니다.

```
<acl who="root" access_mask="default" flags="-i,-R" type="2"/>
```

argument 토큰

argument 토큰에는 시스템 호출의 인수에 대한 정보(시스템 호출의 인수 수, 인수 값 및 선택적 설명)가 포함됩니다. 이 토큰은 감사 레코드에서 32비트 정수 시스템 호출 인수를 허용합니다.

praudit -x 명령은 argument 토큰의 필드를 다음과 같이 표시합니다.

```
<argument arg-num="2" value="0x5401" desc="cmd"/>
```

attribute 토큰

attribute 토큰에는 파일 vnode의 정보가 포함됩니다.

attribute 토큰은 대개 path 토큰과 함께 사용됩니다. attribute 토큰은 경로 검색 중 생성됩니다. 경로 검색 오류가 발생할 경우 필요한 파일 정보를 얻을 수 있는 vnode가 없습니다. 따라서 attribute 토큰이 감사 레코드의 일부로 포함되지 않습니다. praudit -x 명령은 attribute 토큰의 필드를 다음과 같이 표시합니다.

```
<attribute mode="20620" uid="root" gid="tty" fsid="0" nodeid="9267" device="108233"/>
```

cmd 토큰

cmd 토큰은 명령과 연결된 인수 목록 및 환경 변수 목록을 기록합니다.

praudit -x 명령은 cmd 토큰의 필드를 표시합니다. 다음은 잘린 cmd 토큰입니다. 행은 표시 목적으로 줄바꿈되었습니다.

```
<cmd><arge>WINDOWID=6823679</arge>
<arge>COLORTERM=gnome-terminal</arge>
<arge>...LANG=C</arge>...<arge>HOST=machine1</arge>
<arge>LPDEST=printer1</arge>...</cmd>
```

exec_args 토큰

exec_args 토큰은 exec() 시스템 호출의 인수를 기록합니다.

praudit -x 명령은 exec_args 토큰의 필드를 다음과 같이 표시합니다.

```
<exec_args><arg>/usr/bin/sh</arg><arg>/usr/bin/hostname</arg></exec_args>
```

주 – exec_args 토큰은 argv 감사 정책 옵션이 활성화된 경우에만 출력됩니다.

exec_env 토큰

exec_env 토큰은 exec() 시스템 호출의 현재 환경 변수를 기록합니다.

praudit -x 명령은 exec_env 토큰의 필드를 표시합니다. 행은 표시 목적으로 줄바꿈되었습니다.

```
<exec_env><env>_=/usr/bin/hostname</env>
<env>LANG=C</env><env>PATH=/usr/bin:/usr/ucb</env>
<env><env>LOGNAME=jdoe</env><env>USER=jdoe</env>
<env>DISPLAY=:0</env><env>SHELL=/bin/csh</env>
<env>HOME=/home/jdoe</env><env>PWD=/home/jdoe</env><env>TZ=US/Pacific</env>
</exec_env>
```

주 – exec_env 토큰은 arge 감사 정책 옵션이 활성화된 경우에만 출력됩니다.

file 토큰

file 토큰은 새 감사 파일의 시작과 이전 파일이 비활성화된 경우 이전 감사 파일의 끝을 표시하는 특수 토큰입니다. 처음 file 토큰은 감사 추적에서 이전 파일을 식별합니다. 최종 file 토큰은 감사 추적에서 다음 파일을 식별합니다. 이러한 토큰은 연속된 감사 파일을 하나의 감사 추적으로 "연결"합니다.

praudit -x 명령은 file 토큰의 필드를 표시합니다. 행은 표시 목적으로 줄바꿈되었습니다.

```
<file iso8601="2009-04-08 14:18:26.200 -07:00">
/var/audit/machine1/files/20090408211826.not_terminated.machine1</file>
```

fmri 토큰

fmri 토큰은 결함 관리 리소스 표시기(FMRI)의 사용을 기록합니다. 자세한 내용은 smf(5) 매뉴얼 페이지를 참조하십시오.

praudit -x 명령은 fmri 토큰의 내용을 표시합니다.

```
<fmri service_instance="svc:/system/cryptosvc"</fmri>
```

group 토큰

group 토큰은 프로세스 자격 증명의 그룹 항목을 기록합니다.

praudit -x 명령은 groups 토큰의 필드를 다음과 같이 표시합니다.

```
<group><gid>staff</gid><gid>other</gid></group>
```

주 – group 토큰은 group 감사 정책 옵션이 활성화된 경우에만 출력됩니다.

header 토큰

header 토큰은 감사 레코드의 시작을 표시하는 특수 토큰입니다. header 토큰은 trailer 토큰과 결합하여 레코드의 다른 모든 토큰을 괄호로 묶습니다.

드물게, header 토큰이 하나 이상의 이벤트 수정자를 포함할 수 있습니다.

- fe는 실패한 감사 이벤트를 나타냅니다.
- fp는 실패한 권한 사용을 나타냅니다.
- na는 지정 불가능한 이벤트를 나타냅니다.

  ```
  header,52,2,system booted,na,mach1,2011-10-10 10:10:20.564 -07:00
  ```
- rd는 데이터를 객체에서 읽기를 나타냅니다.
- sp는 성공한 권한 사용을 나타냅니다.

  ```
  header,120,2,exit(2),sp,mach1,2011-10-10 10:10:10.853 -07:00
  ```
- wr은 데이터를 객체에 쓰기를 나타냅니다.

praudit 명령은 header 토큰을 다음과 같이 표시합니다.

```
header,756,2,execve(2),,machine1,2010-10-10 12:11:10.209 -07:00
```

praudit -x 명령은 header 토큰의 필드를 감사 레코드의 시작에 표시합니다. 행은 표시 목적으로 줄바꿈되었습니다.

```
<record version="2" event="execve(2)" host="machine1"
iso8601="2010-10-10 12:11:10.209 -07:00">
```

ip address 토큰

ip address 토큰에는 인터넷 프로토콜 주소(IP 주소)가 포함됩니다. IP 주소는 IPv4 또는 IPv6 형식으로 표시될 수 있습니다. IPv4 주소는 4바이트를 사용합니다. IPv6 주소는 1바이트를 사용하여 주소 유형을 설명하고, 16바이트를 사용하여 주소를 설명합니다.

praudit -x 명령은 ip address 토큰의 내용을 다음과 같이 표시합니다.

```
<ip_address>machine1</ip_address>
```

ip port 토큰

ip port 토큰에는 TCP 또는 UDP 포트 주소가 포함됩니다.

praudit 명령은 ip port 토큰을 다음과 같이 표시합니다.

```
ip port,0xf6d6
```

ipc 토큰

ipc 토큰에는 호출자가 특정 IPC 객체를 식별하는 데 사용되는 시스템 V IPC 메시지 핸들, 세마포어 핸들 또는 공유 메모리 핸들이 포함됩니다.

주 – IPC 객체 식별자는 감사 토큰의 컨텍스트 없는 성질에 위배됩니다. 전역 "이름"은 IPC 객체를 고유하게 식별하지 않습니다. 대신 IPC 객체가 핸들로 식별됩니다. 핸들은 IPC 객체가 활성화된 시간 동안에만 유효합니다. 하지만 IPC 객체의 식별이 문제가 되면 안됩니다. 시스템 V IPC 방식은 거의 사용되지 않으며, 방식은 모두 동일한 감사 클래스를 공유합니다.

다음 표는 IPC 객체 유형 필드에 가능한 값을 나타냅니다. 값은 /usr/include/bsm/audit.h 파일에 정의되어 있습니다.

표 29–2 IPC 객체 유형 필드에 대한 값

이름	값	설명
AU_IPC_MSG	1	IPC 메시지 객체
AU_IPC_SEM	2	IPC 세마포어 객체
AU_IPC_SHM	3	IPC 공유 메모리 객체

praudit -x 명령은 ipc 토큰의 필드를 다음과 같이 표시합니다.

```
<IPC ipc-type="shm" ipc-id="15"/>
```

IPC_perm 토큰

IPC_perm 토큰에는 시스템 V IPC 액세스 권한의 복사본이 포함됩니다. 이 토큰은 IPC 공유 메모리 이벤트, IPC 세마포어 이벤트 및 IPC 메시지 이벤트로 생성되는 감사 레코드에 추가됩니다.

praudit -x 명령은 IPC_perm 토큰의 필드를 표시합니다. 행은 표시 목적으로 줄바꿈되었습니다.

```
<IPC_perm uid="jdoe" gid="staff" creator-uid="jdoe"
creator-gid="staff" mode="100600" seq="0" key="0x0"/>
```

값은 IPC 객체와 연결된 IPC_perm 구조에서 가져옵니다.

path 토큰

path 토큰에는 객체에 대한 액세스 경로 정보가 포함됩니다.

praudit -x 명령은 path 토큰의 내용을 표시합니다.

```
<path>/export/home/srv/.xsession-errors</path>
```

path_attr 토큰

path_attr 토큰에는 객체에 대한 액세스 경로 정보가 포함됩니다. 액세스 경로는 path 토큰 객체 아래에 있는 속성 파일 객체의 시퀀스를 지정합니다. openat() 와 같은 시스템 호출이 속성 파일에 액세스합니다. 속성 파일 객체에 대한 자세한 내용은 fsattr(5) 매뉴얼 페이지를 참조하십시오.

praudit 명령은 다음과 같이 path_attr 토큰을 표시합니다.

```
path_attr,1,attr_file_name
```

privilege 토큰

privilege 토큰은 프로세스에 대한 권한 사용을 기록합니다. privilege 토큰은 기본 집합의 권한에 대해서는 기록되지 않습니다. 권한이 관리 작업으로 기본 집합에서 제거된 경우에는 해당 권한의 사용이 기록됩니다. 권한에 대한 자세한 내용은 146 페이지 “권한(개요)”을 참조하십시오.

praudit -x 명령은 privilege 토큰의 필드를 표시합니다.

```
<privilege set-type="Inheritable">ALL</privilege>
```

process 토큰

process 토큰에는 신호의 수신자와 같이 프로세스와 연결된 사용자에 대한 정보가 포함됩니다.

praudit -x 명령은 process 토큰의 필드를 표시합니다. 행은 표시 목적으로 줄바꿈되었습니다.

```
<process audit-uid="-2" uid="root" gid="root" ruid="root"
rgid="root" pid="567" sid="0" tid="0 0 0.0.0.0"/>
```

return 토큰

return 토큰에는 시스템 호출의 반환 상태(u_error) 및 프로세스 반환 값(u_rval1)이 포함됩니다.

return 토큰은 항상 시스템 호출에 대해 커널에서 생성한 감사 레코드의 일부로 반환됩니다. 응용 프로그램 감사에서 이 토큰은 종료 상태 및 기타 반환 값을 나타냅니다.

praudit 명령은 시스템 호출에 대한 return 토큰을 다음과 같이 표시합니다.

```
return,failure: Operation now in progress,-1
```

praudit -x 명령은 return 토큰의 필드를 다음과 같이 표시합니다.

```
<return errval="failure: Operation now in progress" retval="-1/">
```

sequence 토큰

sequence 토큰에는 시퀀스 번호가 포함됩니다. 시퀀스 번호는 감사 레코드가 감사 추적에 추가될 때마다 증분됩니다. 이 토큰은 디버깅에 유용합니다.

praudit -x 명령은 sequence 토큰의 내용을 표시합니다.

```
<sequence seq-num="1292"/>
```

주 – sequence 토큰은 seq 감사 정책 옵션이 활성화된 경우에만 출력됩니다.

socket 토큰

socket 토큰에는 인터넷 소켓을 설명하는 정보가 포함됩니다. 일부 인스턴스에서 토큰에는 원격 포트 및 원격 IP 주소만 포함됩니다.

praudit 명령은 socket 토큰의 인스턴스를 다음과 같이 표시합니다.

```
socket,0x0002,0x83b1,localhost
```

확장된 토큰은 소켓 유형 및 로컬 포트 정보를 포함한 정보를 추가합니다.

praudit -x 명령은 socket 토큰의 인스턴스를 다음과 같이 표시합니다. 행은 표시 목적으로 줄바꿈되었습니다.

```
<socket sock_domain="0x0002" sock_type="0x0002" lport="0x83cf"
laddr="example1" fport="0x2383" faddr="server1.Subdomain.Domain.COM"/>
```

subject 토큰

subject 토큰은 작업을 수행하거나 시도하는 사용자를 설명합니다. 형식은 process 토큰과 동일합니다.

subject 토큰은 항상 시스템 호출에 대해 커널에서 생성한 감사 레코드의 일부로 반환됩니다. praudit 명령은 subject 토큰을 다음과 같이 표시합니다.

```
subject,jdoe,root,root,root,root,1631,1421584480,8243 65558 machine1
```

praudit -x 명령은 subject 토큰의 필드를 표시합니다. 행은 표시 목적으로 줄바꿈되었습니다.

```
<subject audit-uid="jdoe" uid="root" gid="root" ruid="root"
rgid="root" pid="1631" sid="1421584480" tid="8243 65558 machine1"/>
```

text 토큰

text 토큰에는 텍스트 문자열이 포함됩니다.

praudit -x 명령은 text 토큰의 내용을 표시합니다.

```
<text>booting kernel</text>
```

trailer 토큰

header 및 trailer의 두 토큰은 감사 레코드의 끝점을 구별하고 다른 모든 토큰을 괄호로 묶는 특수 토큰입니다. header 토큰은 감사 레코드를 시작합니다. trailer 토큰은 감사 레코드를 끝냅니다. trailer 토큰은 선택적 토큰입니다. trailer 토큰은 trail 감사 정책 옵션이 설정된 경우에만 각 레코드의 마지막 토큰으로 추가됩니다.

트레일러가 설정된 상태에서 감사 레코드가 생성된 경우 auditreduce 명령은 트레일러가 레코드 헤더를 올바르게 가리키는지 확인할 수 있습니다. trailer 토큰은 감사 추적의 역추적을 지원합니다.

praudit 명령은 trailer 토큰을 다음과 같이 표시합니다.

```
trailer,136
```

use of authorization 토큰

use of authorization 토큰은 권한 부여 사용을 기록합니다.

praudit 명령은 use of authorization 토큰을 다음과 같이 표시합니다.

```
use of authorization,solaris.role.delegate

XXXX<use_of_authorization result="successful use of auth">solaris.role.delegate</use_of_auth>
```

use of privilege 토큰

use of privilege 토큰은 권한 사용을 기록합니다.

praudit -x 명령은 use of privilege 토큰의 필드를 다음과 같이 표시합니다.

```
<use_of_privilege result="successful use of priv">proc_setid</use_of_privilege>
```

user 토큰

user 토큰은 사용자 이름 및 사용자 ID를 기록합니다. 이 토큰은 사용자 이름이 호출자와 다른 경우에만 존재합니다.

praudit -x 명령은 user 토큰의 필드를 다음과 같이 표시합니다.

```
<user uid="123456" username="tester1"/>
```

xclient 토큰

xclient 토큰에는 X 서버에 대한 클라이언트 연결 수가 포함됩니다.

praudit -x 명령은 xlient 토큰의 내용을 다음과 같이 표시합니다.

```
<X_client>15</X_client>
```

zonename 토큰

zonename 토큰은 감사 이벤트가 발생한 영역을 기록합니다. 문자열 “global”은 전역 영역에서 발생한 감사 이벤트를 나타냅니다.

praudit -x 명령은 zonename 토큰의 내용을 표시합니다.

```
<zone name="graphzone"/>
```

용어집

AES Advanced Encryption Standard입니다. 대칭 128비트 블록 데이터 암호화 기술입니다. 미국 정부는 2000년 10월 알고리즘의 Rijndael 변형을 암호화 표준으로 채택했습니다. AES가 정부 표준으로 사용자 주체 암호화를 대체합니다.

Blowfish 32비트에서 448비트까지 가변 길이 키를 사용하는 대칭 블록 암호 알고리즘입니다. 작성자인 Bruce Schneier에 따르면 Blowfish는 키가 자주 변경되지 않는 응용 프로그램에 최적화되었다고 합니다.

DES Data Encryption Standard입니다. 1975년에 개발되고 1981년에 ANSI에 의해 ANSI X.3.92로 표준화된 대칭 키 암호화 방법입니다. DES에서는 56비트 키를 사용합니다.

Diffie-Hellman 프로토콜 공개 키 암호화라고도 합니다. 1976년 Diffie와 Hellman이 개발한 비대칭 암호화 키 계약 프로토콜입니다. 이 프로토콜을 사용하면 두 사용자가 사전 보안 없이 비보안 매체를 통해 보안 키를 교환할 수 있습니다. Diffie-Hellman은 Kerberos에서 사용됩니다.

DSA Digital Signature Algorithm입니다. 512비트에서 4096비트 사이의 가변 키 크기를 사용하는 공개 키 알고리즘입니다. 미국 정부 표준인 DSS는 최대 1024비트까지 지정합니다. DSA는 입력에 SHA1을 사용합니다.

FQDN 정규화된 도메인 이름입니다. 예를 들어, `central.example.com`이 있습니다(대조적으로 간단히 `denver`).

GSS-API Generic Security Service Application Programming Interface의 약자. Kerberos 서비스를 포함하여 다양한 모듈형 보안 서비스를 지원하는 네트워크 계층입니다. GSS-API는 보안 인증, 무결성, 프라이버시 서비스를 제공합니다. 인증, 무결성, 프라이버시도 참조하십시오.

KDC Key Distribution Center(키 배포 센터)의 약자. 세 가지 Kerberos V5 구성 요소가 있는 시스템입니다.

- 주체 및 키 데이터베이스
- 인증 서비스
- TGS(티켓 부여 서비스)

각 영역에는 마스터 KDC가 있고 하나 이상의 슬레이브 KDC가 있어야 합니다.

Kerberos 인증 서비스, 서비스에서 사용되는 프로토콜 또는 서비스 구현에 사용되는 코드입니다.

Oracle Solaris Kerberos 구현은 Kerberos V5 구현에 가장 근접합니다.

"Kerberos"와 "Kerberos V5"는 기술적으로 서로 다르지만 Kerberos 문서에서 종종 바꿔서 사용하기도 합니다.

Kerberos(Cerberus라고도 씀)는 그리스 신화에서 지옥의 문을 지키는 머리가 셋 달린 사나운 개입니다.

Kerberos 정책 Kerberos 서비스에서 암호 사용을 통제하는 규칙 세트입니다. 정책을 통해 주체의 액세스나 티켓 수명 매개변수를 규제할 수 있습니다.

key 1. 일반적으로, 두 가지의 주요 키 유형 중 하나입니다.

- **대칭 키** – 암호 해독 키와 똑같은 암호화 키입니다. 대칭 키는 파일을 암호화하는 데 사용됩니다.
- **비대칭 키** 또는 **공개 키** – Diffie-Hellman 또는 RSA와 같은 공개 키 알고리즘에 사용되는 키입니다. 공개 키에는 한 사용자에만 알려진 개인 키, 서버나 일반 리소스에서 사용되는 공개 키, 그리고 둘을 조합한 개인-공개 키 쌍이 포함됩니다. 개인 키는 **보안** 키라고도 합니다. 공개 키는 **공유** 키 또는 **공통** 키라고도 합니다.
- 2. keytab 파일의 항목(주체 이름)입니다. keytab 파일도 참조하십시오.

3. Kerberos에서 암호화 키로 사용되며 다음 세 가지 유형이 있습니다.

- **개인 키** – 주체 및 KDC에서 공유되고 시스템 한도 밖에서 배포되는 암호화 키입니다. 개인 키도 참조하십시오.
- **서비스 키** – 이 키는 개인 키와 동일한 목적을 제공하지만, 서버 및 서비스에서 사용됩니다. 서비스 키도 참조하십시오.
- **세션 키** – 단일 로그인 세션 기간으로 제한된 수명 동안 두 주체 간에 사용되는 임시 암호화 키입니다. 세션 키도 참조하십시오.

keytab 파일 하나 이상의 키(주체)를 포함하는 키 테이블 파일입니다. 사용자가 암호를 사용하는 것처럼 호스트나 서비스는 keytab 파일을 사용합니다.

kvno 키 버전 번호. 생성 순서대로 특정 키를 추적하는 시퀀스 번호입니다. 가장 높은 kvno가 최신의 가장 현재 키입니다.

MAC 1. MAC(메시지 인증 코드)를 참조하십시오.

2. 레이블 지정이라고도 합니다. 정부 보안 기술에서 MAC은 필수 액세스 제어입니다. MAC의 예로 Top Secret 및 Confidential과 같은 레이블이 있습니다. MAC은 DAC(임의 액세스 제어)과 대조를 이룹니다. DAC의 예로 UNIX 사용 권한이 있습니다.

3. 하드웨어에서 LAN의 고유한 시스템 주소입니다. 시스템이 이더넷에 있는 경우 MAC은 이더넷 주소입니다.

MAC (메시지 인증 코드) MAC는 데이터 무결성을 보장하고 데이터 출처를 인증합니다. MAC는 도청에 대한 보호 기능을 제공하지 않습니다.

MD5 디지털 서명을 포함하여 메시지 인증에 사용되는 반복적인 암호화 해시 기능입니다. 이 기능은 1991년 Rivest가 개발했습니다.

NTP Network Time Protocol의 약자. 네트워크 환경에서 정밀한 시간이나 네트워크 클럭 동기화(또는 둘 다)를 관리할 수 있는 델라웨어 대학교에서 설계한 소프트웨어입니다. NTP를 사용하여 Kerberos 환경에서 클럭 불균형을 유지 관리할 수 있습니다. 클럭 불균형도 참조하십시오.

PAM Pluggable Authentication Module의 약자. 여러 인증 방식에서 서비스를 재컴파일할 필요 없이 사용할 수 있는 프레임워크입니다. PAM에서는 로그인 시 Kerberos 세션 초기화가 가능합니다.

primary 주체 이름의 첫번째 부분입니다. 인스턴스, 주체 이름, 영역도 참조하십시오.

principal 1. 네트워크 통신에 참여하는 고유한 이름이 지정된 클라이언트/사용자 또는 서버/서비스입니다. Kerberos 트랜잭션에는 주체들(서비스 주체 및 사용자 주체) 간의 상호 작용 또는 주체와 KDC 간의 상호 작용이 관여합니다. 다시 말해서, 주체는 Kerberos가 티켓을 지정할 수 있는 고유한 개체입니다. 주체 이름, 서비스 주체, 사용자 주체도 참조하십시오.

2. (RPCSEC_GSS API) 클라이언트 주체, 서버 주체를 참조하십시오.

QOP Quality of Protection의 약자. 무결성 서비스나 프라이버시 서비스와 함께 사용되는 암호화 알고리즘을 선택할 수 있는 매개변수입니다.

RBAC Oracle Solaris 기능의 일종인 역할 기반 액세스 제어입니다. all-or-nothing 수퍼유저 모델의 대안입니다. RBAC를 사용하여 조직은 수퍼유저의 능력을 분리하여 역할이라는 특수한 사용자 계정에 지정할 수 있습니다. 특정 개인에게 그 책임에 따라 역할을 지정할 수 있습니다.

RBAC 정책 명령과 연관된 보안 정책입니다. 현재 유효한 정책은 `solaris`입니다. `solaris` 정책은 권한, 인증 및 `setuid` 보안 속성을 인식합니다.

RSA 디지털 서명 및 공개 키 암호 방식을 가져오는 방법입니다. 이 방법은 1978년 개발자 Rivest, Shamir 및 Adleman에 의해 처음 설명되었습니다.

SEAM Sun Enterprise Authentication Mechanism의 약자. 매사추세츠 공과대학에서 개발한 Kerberos V5 기술을 기반으로, 네트워크를 통해 사용자를 인증하기 위한 시스템의 초기 버전에 대한 제품이름입니다. 이제 제품을 Kerberos 서비스라고 부릅니다. SEAM은 다양한 Solaris 릴리스에 포함되지 않은 Kerberos 서비스의 일부를 참조합니다.

SHA1 Secure Hashing Algorithm입니다. 이 알고리즘은 2^{64} 미만의 입력 길이에서 작동하여 메시지 다이제스트를 생성합니다. SHA1 알고리즘은 DSA에 입력됩니다.

stash 파일 stash 파일은 KDC용 마스터 키의 암호화된 복사본을 포함합니다. `kadmind` 및 `krb5kdc` 프로세스를 시작하기 전에 KDC를 자동으로 인증하도록 서버를 재부트할 때 이 마스터 키가 사용됩니다. stash 파일에 마스터 키가 포함되므로 stash 파일과 그 백업을 안전하게 보관해야 합니다. 암호화가 훼손되면 키를 사용하여 KDC 데이터베이스를 액세스하거나 수정해야 합니다.

TGS Ticket-Granting Service(티켓 부여 서비스)의 약자. 티켓 발행을 담당하는 KDC의 부분입니다.

TGT Ticket-Granting Ticket(티켓 부여 티켓)의 약자. 클라이언트가 다른 서비스에 대한 티켓을 요청할 수 있는 KDC에서 발행한 티켓입니다.

VPN (가상 사설망) 암호화를 사용하고 공용 네트워크를 통한 사용자 연결을 터널링하여 보안 통신을 제공하는 네트워크입니다.

감사 정책 어떤 감사 이벤트를 기록할지 결정하는 전역 사용자별 설정입니다. 감사 서비스에 적용되는 전역 설정은 일반적으로 감사 증적에 포함할 선택적 정보 조각에 영향을 미칩니다. 두 설정 `cnt` 및 `ahlt`는 감사 대기열을 채울 때 시스템의 작업에 영향을 미칩니다. 예를 들어, 감사 정책에서 시퀀스 번호가 모든 감사 레코드에 속하도록 요구할 수 있습니다.

감사 추적 모든 호스트의 모든 감사 파일 모음입니다.

감사 파일 이진 감사 로그. 감사 파일은 감사 파일 시스템에 별도로 저장됩니다.

강화 호스트에 내재된 보안 취약성을 제거하도록 운영 체제의 기본 구성을 수정한 것입니다.

개인 키 각 사용자 주체에 제공되며 주체의 사용자와 KDC에만 알려진 키입니다. 사용자 주체의 경우 키는 사용자 암호를 기반으로 합니다. key도 참조하십시오.

개인 키 암호화 개인 키 암호화에서 발신자와 수신자는 암호화에 동일한 키를 사용합니다. 공개 키 암호화도 참조하십시오.

갱신 가능 티켓 장시간 존재하는 티켓은 보안 위험이 있으므로 티켓을 *renewable*로 지정할 수 있습니다. 갱신 가능 티켓에는 두 개의 만료 시간 a) 티켓의 현재 인스턴스가 만료되는 시간 b) 티켓의 최대 수명이 있습니다. 클라이언트가 티켓을 계속 사용하려면 첫번째 만료가 발생하기 전에 티켓을 갱신합니다. 예를 들어, 1시간 동안 유효한 티켓이 있고 모든 티켓은 최대 10시간의 수명을 가질 수 있습니다. 티켓을 보유하는 클라이언트가 1시간보다 더 오래 티켓을 보관하려면 티켓을 갱신해야 합니다. 티켓이 최대 티켓 수명에 도달하면 자동으로 만료되어 갱신할 수 없습니다.

검사 엔진 파일에 알려진 바이러스가 있는지 조사하는, 외부 호스트에 상주하는 타사 응용 프로그램입니다.

공개 키 기술에 대한 정책 키 관리 프레임워크(KMF)에서 정책은 인증서 사용을 관리합니다. KMF 정책 데이터베이스는 KMF 라이브러리에서 관리되는 키 및 인증서 사용을 제약할 수 있습니다.

공개 키 암호화 각 사용자가 두 개의 키, 즉 하나의 공개 키와 하나의 개인 키를 사용하는 암호화 체계입니다. 공개 키 암호화에서 발신자는 수신자의 공개 키를 사용하여 메시지를 암호화하고, 수신자는 개인 키를 사용하여 암호를 해독합니다. Kerberos 서비스는 개인 키 시스템입니다. 개인 키 암호화도 참조하십시오.

공급자 Oracle Solaris의 암호화 프레임워크 기능에서 소비자에게 제공된 암호화 서비스입니다. 공급자의 예로 PKCS #11 라이브러리, 커널 암호화 모듈, 하드웨어 가속기가 있습니다. 공급자는 암호화 프레임워크에 플러그인되므로 **플러그인**이라고도 합니다. 소비자의 예는 소비자를 참조하십시오.

공용 객체 `root` 사용자가 소유하고 어디서든 읽을 수 있는 파일입니다(예: `/etc` 디렉토리의 파일).

관계 `kdc.conf` 또는 `krb5.conf` 파일에 정의된 구성 변수 또는 관계입니다.

관리자 주체 *username* `/admin`(예: `jdoe/admin`) 형식의 이름을 가진 사용자 주체입니다. 관리자 주체는 일반 사용자 주체보다 더 많은 권한(예: 정책 변경)을 가질 수 있습니다. 주체 이름, 사용자 주체도 참조하십시오.

권한 Oracle Solaris 시스템에서 프로세스에 대한 별개의 권한입니다. 권한은 `root`인 프로세스를 좀 더 세부적으로 제어합니다. 권한은 커널에서 정의되고 시행됩니다. 권한에 대한 자세한 설명은 `privileges(5)` 매뉴얼 페이지를 참조하십시오.

권한 모델 수퍼유저 모델보다 더 엄격한 컴퓨터 시스템의 보안 모델입니다. 권한 모델에서 프로세스를 실행하려면 권한이 필요합니다. 시스템 운영은 관리자가 해당 프로세스에 보유한 권한으로 기반으로 별개의 부분으로 나눌 수 있습니다. 관리자의 로그인 프로세스에 권한을 지정할 수 있습니다. 또는 특정 명령에만 효력을 발휘하도록 권한을 지정할 수 있습니다.

권한 부여 1. Kerberos에서는 주체가 서비스를 사용할 수 있는지, 어떤 객체에 주체가 액세스할 수 있는지, 각 객체에 대해 허용된 액세스 유형 등을 결정하는 프로세스입니다.

2. RBAC(역할 기반 액세스 제어)에서는 보안 정책에 의해 금지된 일련의 작업을 수행하기 위해 역할/사용자에 지정할 수 있는(또는 권한 프로파일에 포함할 수 있는) 사용 권한입니다.

권한 세트 권한 모음입니다. 각 프로세스에는 프로세스가 특정 권한을 사용할 수 있는지 여부를 결정하는 4개의 권한 세트가 있습니다. 제한 세트, 유효 세트, 허가된 세트, 상속 가능한 세트를 참조하십시오.

또한 권한의 기본 세트는 로그인 시 사용자의 프로세스에 지정된 권한 모음입니다.

권한 에스컬레이션 지정된 보안 속성(대체 포함)이 허용하는 리소스 범위 밖에 있는 리소스에 액세스를 얻는 것입니다. 그 결과, 프로세스가 권한이 없는 작업을 수행할 수 있습니다.

권한 인식 코드를 통해 권한 사용을 켜고 끄는 프로그램, 스크립트, 명령입니다. 운용 환경에서 켜져 있는 권한을 프로세스에 제공해야 합니다. 프로그램의 사용자가 권한을 프로그램에 추가한 권한 프로파일을 사용하도록 하면 됩니다. 권한에 대한 자세한 설명은 `privileges(5)` 매뉴얼 페이지를 참조하십시오.

권한 있는 응용 프로그램 시스템 컨트롤을 대체할 수 있는 응용 프로그램입니다. 응용 프로그램이 특정 UID, GID, 인증 또는 권한과 같은 보안 속성을 검사합니다.

권한 프로파일 권한 또는 프로파일이라고도 합니다. 역할 또는 사용자에 지정할 수 있는 RBAC에 사용된 대체 모음입니다. 권한 프로파일은 인증, 권한, 보안 속성 포함 명령 및 기타 권한 프로파일로 구성됩니다.

기밀성 프라이버시를 참조하십시오.

기본 세트 로그인 시 사용자 프로세스에 지정되는 권한 세트입니다. 수정되지 않은 시스템에서 각 사용자의 초기 상속 가능한 세트는 로그인 시 기본 세트와 같습니다.

네트워크 애플리케이션 서버 `ftp`와 같은 네트워크 응용 프로그램을 제공하는 서버입니다. 영역에 여러 네트워크 애플리케이션 서버를 포함할 수 있습니다.

다이제스트 메시지 다이제스트를 참조하십시오.

단일 시스템 이미지 단일 시스템 이미지는 Oracle Solaris 감사에 사용되어 동일한 이름 지정 서비스를 사용하는 감사 시스템 그룹을 설명합니다. 이러한 시스템은 해당 감사 레코드를 중앙 감사 서버로 보내고, 여기서 레코드가 한 시스템에서 나온 것처럼 레코드를 비교할 수 있습니다.

동기 감사 이벤트 감사 이벤트의 다수를 차지합니다. 이러한 이벤트는 시스템의 프로세스와 연관됩니다. 프로세스와 연관된 출처를 알 수 없는 이벤트는 실패한 로그인과 같은 동기 이벤트입니다.

마스터 KDC 각 영역의 주 KDC로, Kerberos 관리 서버인 `kadmind`와 인증 및 티켓 부여 데몬인 `krb5kdc`를 포함합니다. 각 영역에는 적어도 하나의 마스터 KDC가 있어야 하고, 클라이언트에 인증 서비스를 제공하는 많은 중복된 슬레이브 KDC를 포함할 수 있습니다.

메시지 다이제스트 메시지 다이제스트는 메시지에서 계산된 해시 값입니다. 해시 값은 메시지를 거의 고유하게 식별합니다. 다이제스트는 파일 무결성 확인에 유용합니다.

무결성 사용자 인증과 더불어, 암호화 체크섬을 통해 전송된 데이터의 유효성을 제공하는 보안 서비스입니다. 인증, 프라이버시도 참조하십시오.

방식 1. 데이터 인증 또는 기밀성을 이루기 위한 암호화 기법을 지정하는 소프트웨어 패키지입니다. 예: Kerberos V5, Diffie-Hellman 공개 키.

2. Oracle Solaris의 암호화 프레임워크 기능에서 특정 목적을 위한 알고리즘의 구현입니다. 예를 들어, 인증에 적용된 DES 방식(예: CKM_DES_MAC)은 암호화에 적용된 DES 방식(예: CKM_DES_CBC_PAD)과 별도의 방식입니다.

보안 방식 방식을 참조하십시오.

보안 서비스 서비스를 참조하십시오.

보안 셸 비보안 네트워크를 통해 보안 원격 로그인 및 기타 보안 네트워크 서비스를 제공하기 위한 특수한 프로토콜입니다.

보안 속성 RBAC에서 수퍼유저가 아닌 일반 사용자가 관리 명령을 실행할 때 명령을 성공하게 해주는 보안 정책의 대체입니다. 수퍼유저 모델에서 `setuid` 및 `setgid` 프로그램이 보안 속성입니다. 이러한 속성을 명령에 적용할 때 누가 명령을 실행하든지 관계없이 명령을 성공합니다. 권한 모델에서 보안 속성은 권한입니다. 명령에 권한을 부여할 때 명령을 성공합니다. 권한 모델에서도 `setuid` 및 `setgid` 프로그램을 보안 속성으로 인식하므로 수퍼유저 모델과 호환됩니다.

보안 정책 정책을 참조하십시오.

보안 종류 종류를 참조하십시오.

보안 키 개인 키를 참조하십시오.

비동기 감사 이벤트 비동기 이벤트는 시스템 이벤트의 소수를 차지합니다. 이러한 이벤트는 어떤 프로세스와 연관되지 않으므로 프로세스를 차단했다가 나중에 깨울 수 없습니다. 비동기 이벤트의 예로, 초기 시스템 부트 및 PROM 진입/종료 이벤트가 있습니다.

사용자 주체 특정 사용자에 기인한 주체입니다. 사용자 주체의 기본 이름은 사용자 이름이고, 선택적 인스턴스는 의도한 해당 자격 증명 용도를 설명하는 데 사용되는 이름입니다(예: `jdoe` 또는 `jdoe/admin`). 사용자 인스턴스라고도 합니다. 서비스 주체도 참조하십시오.

상속 가능한 세트 `exec`의 호출에서 프로세스가 상속할 수 있는 권한 세트입니다.

서버 네트워크 클라이언트에 리소스를 제공하는 주체입니다. 예를 들어, `central.example.com` 시스템에 `ssh`를 제공하면 해당 시스템은 `ssh` 서비스를 제공하는 서버입니다. 서비스 주체도 참조하십시오.

서버 주체 (RPCSEC_GSS API) 서비스를 제공하는 주체입니다. 서버 주체는 *service*`@`*host* 형식으로 ASCII 문자열로 저장됩니다. 클라이언트 주체도 참조하십시오.

서비스 1. 종종 여러 대의 서버에 의해 네트워크 클라이언트에 제공된 리소스입니다. 예를 들어, `central.example.com` 시스템에 `rlogin`을 제공하는 경우 해당 시스템은 `rlogin` 서비스를 제공하는 서버입니다.

2. 인증 외의 보호 레벨을 제공하는 보안 서비스(무결성 또는 프라이버시)입니다. 무결성 및 프라이버시도 참조하십시오.

서비스 주체 서비스에 Kerberos 인증을 제공하는 주체입니다. 서비스 주체의 경우 기본 이름은 `ftp`와 같은 서비스 이름이고, 해당 인스턴스는 서비스를 제공하는 시스템의 정규화된 호스트 이름입니다. 호스트 주체, 사용자 주체도 참조하십시오.

서비스 키 서비스 주체 및 KDC에서 공유되고 시스템 한도 밖에서 배포되는 암호화 키입니다. key도 참조하십시오.

세션 키 인증 서비스 또는 TGS(티켓 부여 서비스)에서 생성된 키입니다. 세션 키는 클라이언트와 서비스 간에 보안 트랜잭션을 제공하기 위해 생성됩니다. 세션 키의 수명은 단일 로그인 세션으로 제한됩니다. key도 참조하십시오.

소비자 Oracle Solaris의 암호화 프레임워크 기능에서 소비자는 공급자로부터 전달된 암호화 서비스의 사용자입니다. 소비자는 응용 프로그램, 최종 사용자 또는 커널 작업일 수 있습니다. 소비자의 예로 Kerberos, IKE, IPsec 등이 있습니다. 공급자의 예는 공급자를 참조하십시오.

소프트웨어 공급자 Oracle Solaris의 암호화 프레임워크 기능에서 암호화 서비스를 제공하는 커널 소프트웨어 모듈 또는 PKCS #11 라이브러리입니다. 공급자도 참조하십시오.

수퍼유저 모델 컴퓨터 시스템의 전형적인 UNIX 보안 모델입니다. 수퍼유저 모델에서 관리자는 all-or-nothing 방식으로 시스템을 제어합니다. 일반적으로, 시스템을 관리하려면 사용자가 수퍼유저(`root`)를 맡고 모든 관리 작업을 수행할 수 있습니다.

슬레이브 KDC 마스터 KDC의 복사본으로, 대부분의 마스터 기능을 수행할 수 있습니다. 각 영역에는 대개 여러 개의 슬레이브 KDC(마스터 KDC는 하나만)가 있습니다. KDC, 마스터 KDC도 참조하십시오.

씨드 무작위 수를 생성하기 위한 숫자 스타터입니다. 스타터가 무작위 소스에서 기원할 때 씨드를 **무작위 씨드**라고 합니다.

알고리즘 암호화 알고리즘. 입력을 암호화하거나 해시하는 확립된 순환적 계산 프로시저입니다.

암호 정책 암호를 생성하는 데 사용할 수 있는 암호화 알고리즘입니다. 암호 변경 주기, 잘못된 입력 허용 개수, 기타 보안 고려 사항 등 암호와 관련한 일반적인 사안이라고 할 수 있습니다. 보안 정책에 암호가 필요합니다. 암호 정책에서는 암호를 MD5 알고리즘으로 암호화해야 하고, 추가로 암호 강도와 관련된 요구 사항이 필요할 수 있습니다.

암호문 개인 키를 암호문 사용자가 만들었는지 확인하는 데 사용되는 문구입니다. 좋은 암호문은 10-30자 길이로, 영문자와 숫자를 섞어서 만들고 단순한 문구와 단순한 이름을 피합니다. 통신을 암호화 및 해독하기 위해 개인 키 사용을 인증하려면 암호문을 묻는 메시지가 나타납니다.

암호화 알고리즘 알고리즘을 참조하십시오.

암호화 프레임워크의 정책 Oracle Solaris의 암호화 프레임워크 기능에서 정책은 기존 암호화 방식을 사용 안함으로 설정합니다. 그러면 방식을 사용할 수 없습니다. 암호화 프레임워크의 정책은 DES와 같은 공급자가 `CKM_DES_CBC`와 같은 특정 방식을 사용하는 것을 금지할 수 있습니다.

애플리케이션 서버 네트워크 애플리케이션 서버를 참조하십시오.

액세스 제어 목록(ACL) 액세스 제어 목록(ACL)은 전통적인 UNIX 파일 보호보다 좀 더 세부적인 파일 보안을 제공합니다. 예를 들어, ACL을 사용하여 파일에 그룹 읽기 액세스를 허용하면서 해당 그룹의 한 멤버만 파일 쓰기를 허용할 수 있습니다.

역할 지정된 사용자만 맡을 수 있는 권한 있는 응용 프로그램을 실행하기 위한 특수한 신원입니다.

영역 1. 단일 Kerberos 데이터베이스와 일련의 KDC(키 배포 센터)에 의해 제공된 논리적 네트워크입니다.

2. 주체 이름의 세번째 부분입니다. 주체 이름 `jdoe/admin@ENG.EXAMPLE.COM`의 경우 영역은 `ENG.EXAMPLE.COM`입니다. 주체 이름도 참조하십시오.

유효 세트 현재 프로세스에 발효 중인 권한 세트입니다.

이름 서비스 범위 역할이 작동하도록 허가된 범위입니다. 즉, NIS 또는 LDAP과 같은 지정된 이름 지정 서비스에서 제공하는 개별 호스트 또는 모든 호스트를 말합니다.

인스턴스 주체 이름의 두번째 부분으로, 인스턴스는 주체의 기본 부분을 한정합니다. 서비스 주체의 경우 인스턴스는 필수 사항입니다. 인스턴스는 `host/central.example.com`과 같이 호스트의 정규화된 도메인 이름입니다. 사용자 주체의 경우 인스턴스는 선택 사항입니다. 그러나 `jdoe` 및 `jdoe/admin`은 고유한 주체입니다. primary, 주체 이름, 서비스 주체, 사용자 주체도 참조하십시오.

인증 주체의 제시된 신원을 확인하는 프로세스입니다.

인증자 인증자는 티켓(KDC에서) 및 서비스(서버에서)를 요청할 때 클라이언트에 의해 전달됩니다. 최근 시점에서 확인할 수 있는 클라이언트 및 서버에만 알려진 세션 키를 사용하여 생성된 정보를 포함하므로 트랜잭션이 안전한 것으로 나타납니다. 티켓과 함께 사용할 경우 인증자는 사용자 주체를 인증할 수 있습니다. 인증자에는 사용자의 주체 이름, 사용자 호스트의 IP 주소, 시간 기록이 포함됩니다. 티켓과 달리, 인증자는 대개 서비스 액세스를 요청할 때 한번만 사용할 수 있습니다. 인증자는 클라이언트 및 서버에 대한 세션 키를 사용하여 암호화됩니다.

자격 증명 티켓 및 일치하는 세션 키를 포함하는 정보 패키지입니다. 주체의 신원을 인증하는 데 사용됩니다. 티켓, 세션 키도 참조하십시오.

자격 증명 캐시 KDC로부터 받은 자격 증명을 포함하는 저장 공간(대개 파일)입니다.

잘못된 티켓 아직 사용할 수 없는 후일자 티켓입니다. 잘못된 티켓은 유효해질 때까지 애플리케이션 서버에서 거부합니다. 유효화하려면 시작 시간이 지난 후에 `VALIDATE` 플래그 세트를 사용하여 TGS 요청의 클라이언트가 KDC에 잘못된 티켓을 제시해야 합니다. 후일자 티켓도 참조하십시오.

장치 정책 커널 레벨의 장치 보호입니다. 장치 정책은 장치에 두 개의 권한 세트로 구현됩니다. 첫번째 권한 세트는 장치의 읽기 액세스를 제어합니다. 두번째 권한 세트는 장치의 쓰기 액세스를 제어합니다. 정책도 참조하십시오.

장치 할당 사용자 레벨의 장치 보호입니다. 장치 할당은 하나의 장치를 한번에 한 사용자만 배타적으로 사용하도록 합니다. 장치 재사용 전에 장치 데이터를 비웁니다. 인증을 사용하여 장치 할당이 허가된 사용자를 제한할 수 있습니다.

전달 가능 티켓 클라이언트가 원격 호스트에서 티켓을 요청하기 위해 전체 인증 프로세스를 거치지 않고도 사용할 수 있는 티켓입니다. 예를 들어, 사용자 `jennifer`의 시스템에 있는 동안 사용자 `david`가 전달 가능 티켓을 얻은 경우 david는 새 티켓을 얻지 않고도(다시 인증받을 필요 없이) 자신의 시스템에 로그인할 수 있습니다. 프록시 가능 티켓도 참조하십시오.

정책 일반적으로, 의사결정 및 조치를 반영하거나 결정하는 계획이나 행동 방침입니다. 컴퓨터 시스템의 경우 정책은 대개 보안 정책을 의미합니다. 사이트의 보안 정책은 처리 중인 정보의 민감도를 정의하는 규칙 세트이자, 허용되지 않은 액세스로부터 정보를 보호하는 데 사용되는 측정치입니다. 예를 들어, 시스템을 감사하고 장치를 권한으로 보호하고 암호를 6주마다 변경하도록 보안 정책을 수립할 수 있습니다.

Oracle Solaris OS의 특정 영역에서 정책을 구현하는 방법은 감사 정책, 암호화 프레임워크의 정책, 장치 정책, Kerberos 정책, 암호 정책, RBAC 정책을 참조하십시오.

제한 세트 프로세스와 그 자식에 사용 가능한 권한에 대한 외부 제한입니다.

종류 전통적으로, **보안 종류** 및 **인증 종류**는 인증 유형(AUTH_UNIX, AUTH_DES, AUTH_KERB)을 지칭한 종류로서, 동일한 의미입니다. RPCSEC_GSS는 인증과 더불어 무결성과 프라이버시 서비스를 제공하지만 역시 보안 종류입니다.

주체 이름 1. *primary/instance@REALM* 형식의 주체 이름입니다. 인스턴스, primary, 영역도 참조하십시오.

2. (RPCSEC_GSS API) 클라이언트 주체, 서버 주체를 참조하십시오.

책임 구분 최소한의 특권 개념의 일부입니다. 책임 구분 하에서는, 한 사용자가 트랜잭션을 완성하는 모든 작업을 수행하거나 승인할 수 없습니다. 예를 들어, RBAC에서 보안 대체 지정으로부터 로그인 사용자 생성을 분리할 수 있습니다. 한 역할이 사용자를 만듭니다. 별도의 역할이 권한 프로파일, 역할, 기존 사용자의 권한과 같은 보안 속성을 지정할 수 있습니다.

초기 티켓 (기존 TGT(티켓 부여 티켓)에 기반하지 않고) 직접 발행된 티켓입니다. 암호를 변경하는 응용 프로그램과 같은 일부 서비스는 *initial*로 표시된 티켓이 필요할 수 있으므로 클라이언트가 보안 키를 알고 있다는 것을 스스로 보증해야 합니다. 초기 티켓은 클라이언트가 (오랫동안 존재해 왔던 TGT(티켓 부여 티켓)에 의존하는 대신) 최근에 자체 인증되었음을 나타내므로 이 보증은 매우 중요합니다.

최소한의 특권 지정된 프로세스를 수퍼유저 권력의 일부에만 제공하는 보안 모델입니다. 최소 권한 모델은 일반 사용자가 파일 시스템 마운트 및 파일 소유권 변경과 같은 개인적인 관리 작업을 수행할 수 있도록 충분한 권한을 지정합니다. 반면에 프로세스는 완전한 수퍼유저 권력(즉 모든 권한)이 아닌, 작업 완료에 필요한 권한으로만 실행됩니다. 버퍼 오버플로우 같은 프로그래밍 오류로 인한 손해는, 보호된 시스템 파일 읽기/쓰기 또는 시스템 정지 같은 중요한 능력에 액세스할 수 없는 비루트 사용자에 국한될 수 있습니다.

최소화 서버 실행에 필요한 최소 운영 체제를 설치합니다. 서버 작동에 직접적인 관련이 없는 소프트웨어는 설치되지 않거나 설치 후 삭제됩니다.

출처를 알 수 없는 감사 이벤트 AUE_BOOT 이벤트와 같이 개시자가 결정할 수 없는 감사 이벤트입니다.

클라이언트 좁은 의미로, 사용자 대신 네트워크 서비스를 이용하는 프로세스입니다. 예를 들어, `rlogin`을 사용하는 응용 프로그램이 있습니다. 어떤 경우 서버 자체가 다른 서버나 서비스의 클라이언트가 될 수 있습니다.

더 넓은 의미로, a) Kerberos 자격 증명을 수신하고 b) 서버에서 제공한 서비스를 이용하는 호스트입니다.

간단히 말하면, 서비스를 이용하는 주체입니다.

클라이언트 주체 (RPCSEC_GSS API) RPCSEC_GSS로 보안된 네트워크 서비스를 사용하는 클라이언트(사용자 또는 응용 프로그램)입니다. 클라이언트 주체 이름은 `rpc_gss_principal_t` 구조 형태로 저장됩니다.

클럭 불균형 Kerberos 인증 시스템에 참여하는 모든 호스트의 내부 시스템 클럭에 차이가 날 수 있는 최대 시간입니다. 참여하는 호스트 사이에 클럭 불균형을 초과할 경우 요청이 거부됩니다. 클럭 불균형은 `krb5.conf` 파일에 지정할 수 있습니다.

티켓 사용자의 신원을 서버나 서비스로 안전하게 전달하는 데 사용되는 정보 패킷입니다. 티켓은 단일 클라이언트에만, 그리고 특정 서버의 특정 서비스에만 유효합니다. 티켓에는 서비스의 주체 이름, 사용자의 주체 이름, 사용자 호스트의 IP 주소, 시간 기록, 티켓의 수명을 정의하는 값이 포함됩니다. 클라이언트 및 서비스에서 사용할 무작위 세션 키로 티켓이 생성됩니다. 일단 티켓이 만들어지면 만료될 때까지 재사용할 수 있습니다. 티켓은 새로운 인증자와 함께 제시될 때 클라이언트 인증에만 사용됩니다. 인증자, 자격 증명, 서비스, 세션 키도 참조하십시오.

티켓 파일 자격 증명 캐시를 참조하십시오.

프라이버시 전송된 데이터를 보내기 전에 암호화하는 보안 서비스입니다. 프라이버시에는 데이터 무결성과 사용자 인증도 포함됩니다. 인증, 무결성, 서비스도 참조하십시오.

프로파일 셸 RBAC에서 역할(또는 사용자)이 권한 프로파일에 지정된 권한 있는 응용 프로그램을 명령줄에서 실행할 수 있는 셸입니다. 프로파일 셸은 `pfsh`, `pfcsh`, `pfksh`입니다. 각각 Bourne 셸(`sh`), C 셸(`csh`), Korn 셸(`ksh`)에 해당합니다.

프록시 가능 티켓 클라이언트에 작업을 수행하는 대신, 서비스에서 사용할 수 있는 티켓입니다. 따라서 서비스가 클라이언트의 프록시 역할을 한다고 말할 수 있습니다. 티켓을 사용하여 서비스는 클라이언트의 신원을 차용할 수 있습니다. 프록시 가능 티켓을 사용하여 다른 서비스에 대한 서비스 티켓을 얻을 수 있지만, TGT(티켓 부여 티켓)는 얻을 수 없습니다. 프록시 가능 티켓과 전달 가능 티켓의 차이점은, 프록시 가능 티켓은 단일 작업에만 유효하다는 것입니다. 전달 가능 티켓도 참조하십시오.

하드웨어 공급자 Oracle Solaris의 암호화 프레임워크 기능에서 장치 드라이버 및 해당 하드웨어 가속기입니다. 하드웨어 공급자는 컴퓨터 시스템에서 값비싼 암호화 작업 부담을 덜어주므로 CPU 리소스를 확보하여 다른 용도로 사용할 수 있습니다. 공급자도 참조하십시오.

허가된 세트 프로세스에서 사용할 수 있는 권한 세트입니다.

호스트 네트워크를 통해 액세스 가능한 시스템입니다.

호스트 주체 `ftp`, `rcp`, `rlogin`과 같은 네트워크 서비스 범위를 제공하기 위해 (기본 이름 `host`로 서명된) 주체가 설정되는 특정 인스턴스의 서비스 주체입니다. 호스트 주체의 예는 `host/central.example.com@EXAMPLE.COM`입니다. 서버 주체도 참조하십시오.

후일자 티켓 후일자 티켓은 생성 후 지정된 시간까지 유효해지지 않습니다. 예를 들어, 이러한 티켓은 밤늦게 실행하려는 일괄 처리 작업에 유용합니다. 티켓을 훔친 경우 일괄 처리 작업이 실행될 때까지 티켓을 사용할 수 없기 때문입니다. 후일자 티켓을 발행할 때 *invalid*로 발행되고 a) 시작 시간이 지날 때까지 b) 클라이언트가 KDC에서 검증으로 요청할 때까지 해당 방법을 유지합니다. 후일자 티켓은 보통 TGT(티켓 부여 티켓)의 만료 시간까지 유효합니다. 그러나 후일자 티켓이 *renewable*로 표시된 경우 티켓의 수명이 보통 TGT(티켓 부여 티켓)의 전체 수명 기간과 똑같이 설정됩니다. 잘못된 티켓, 갱신 가능 티켓도 참조하십시오.

색인

B

C

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

감

관

구

권

규

그

금

기

나

만

매

메

명

모

무

문

물

바

방

백

복

삭

읽

임

자

작

재

저

전

절

패

효

후