Oracle® Solaris 11.2의 Kerberos 및 기타 인증 서비스 관리

부품 번호: E53970-02
2014년 8월

ORACLE

목차

표

코드 예

이 설명서 사용

- **개요** - 하나 이상의 Oracle Solaris 시스템에서 보안 인증을 관리하는 방법에 대해 설명합니다. 이 설명서에서는 PAM(플러그 가능한 인증 모듈), Kerberos, SASL(단순 인증 및 보안 계층) 및 NFS와 NIC용 보안 RPC에 대해 설명합니다. 부록에서는 Kerberos용 DTrace 프로브에 대해 설명하고 해당 사용 예를 보여 줍니다.
- **대상** - 시스템, 보안 및 네트워크 보안 관리자
- **필요한 지식** - 액세스 요구 사항 및 네트워크 보안 요구 사항

제품 설명서 라이브러리

이 제품에 대한 최신 정보 및 알려진 문제는 설명서 라이브러리(http://www.oracle.com/pls/topic/lookup?ctx=E56343)에서 확인할 수 있습니다.

Oracle 지원 액세스

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

피드백

http://www.oracle.com/goto/docfeedback에서 이 설명서에 대한 피드백을 보낼 수 있습니다.

♦ ♦ ♦ 1 장

1 플러그 가능한 인증 모듈 사용

이 장에서는 PAM(플러그 가능한 인증 모듈)에 대해 설명합니다. PAM은 Oracle Solaris OS에서 응용 프로그램 사용자에 대한 검사를 플러그인하기 위한 프레임워크를 제공합니다. PAM은 사용자 인증, 암호 변경 관리, 사용자 세션 닫기 및 열기, 계정 제한(예: 시간) 추적을 비롯하여 응용 프로그램 사용을 관리하기 위한 핵심 프레임워크입니다. PAM은 타사 응용 프로그램으로 확장이 가능하므로 시스템 서비스에 대한 액세스를 완벽하게 관리할 수 있습니다.

이 장에서는 다음 내용을 다룹니다.

Oracle Solaris 11.2 인증의 새로운 기능

이 절에서는 기존 고객을 위해 이번 릴리스의 PAM 및 Kerberos 기술에 새로 도입된 중요한 기능에 대해 설명합니다.

PAM의 새로운 기능

Oracle Solaris 11.2 릴리스에서 PAM 프레임워크의 새로운 기능은 다음과 같습니다.

- `pam_unix_account` 모듈은 모든 또는 지정된 PAM 서비스에 대한 시간 기반 액세스 제어를 지원합니다. 따라서 PAM을 호출하는 보안 관련 작업을 특정 날짜 및 시간으로 제한할 수 있습니다. 선택적으로 시간대를 제공할 수 있습니다. `pam_unix_account`(5) 매뉴얼 페이지를 참조하십시오.
- 사용자 및 역할에 `access_times` 및 `access_tz` 키워드를 지정할 수 있습니다. 모든 사용자 보안 속성과 마찬가지로 이러한 키워드도 모든 이름 서비스에서 지원됩니다. 자세한 내용은 `user_attr`(4) 매뉴얼 페이지를 참조하십시오.

Kerberos의 새로운 기능

Oracle Solaris 11.2 릴리스에서 Kerberos의 새로운 기능은 다음과 같습니다.

- Kerberos가 임의 시간에 실행되는 인증된 배치 작업을 지원합니다. `at`, `batch` 및 `cron` 등 지연된 실행을 지원하는 명령은 수동 개입이 없어도 Kerberos 서비스를 사용하여 인증을 제공할 수 있습니다. 자세한 내용은 "Kerberos 서비스에 대한 액세스를 위해 지연된 실행 구성" [117]을 참조하십시오.
- AI(자동 설치 프로그램)를 사용하여 Kerberos 클라이언트를 자동으로 구성할 수 있습니다. 이러한 클라이언트는 Kerberos화된 서비스를 즉시 호스트할 수 있습니다. 설치하는 동안 Kerberos 클라이언트 시스템이 자동으로 프로비전되므로 관리자가 보안 인프라를 빠르고 쉽게 배치하여 수고를 덜 수 있습니다.
  - 설치된 클라이언트 시스템에 추가 단계, 즉 Kerberos 구성을 지정하고 시스템에 대한 Kerberos 서비스 키를 만들기 위한 별도의 작업이 필요 없습니다.
  - 클라이언트 시스템에 대한 Kerberos 서비스 키를 제공하기 위한 옵션이 다양합니다.

  자동 설치 프로그램을 통한 Kerberos 구성에 대한 자세한 내용은 "Oracle Solaris 11.2 시스템 설치 "의 "AI를 사용한 Kerberos 클라이언트 구성 방법"을 참조하십시오.
- `ktkt_warn` 서비스가 기본적으로 사용 안함으로 설정되어 있습니다. 초기 TGT 자격 증명 만료에 대해 사용자에게 경고하거나 사용자의 초기 자격 증명을 자동으로 갱신하려면 이 서비스를 명시적으로 사용으로 설정해야 합니다. 자세한 내용은 "모든 TGT(티켓 부여 티켓) 자동 갱신" [108]을 참조하십시오.

Oracle Solaris에 포함된 다양한 버전의 Kerberos에 대한 자세한 내용은 "Components of Various Kerberos Releases" in "Oracle Solaris 11.1 Administration: Security Services"를 참조하십시오.

PAM 정보

PAM은 다양한 인증 기능을 수행하기 위한 응용 프로그램용 프레임워크를 제공합니다. PAM은 시스템 프로그램을 비롯한 응용 프로그램의 사용자에 대한 중앙 인증, 세션 관리, 암호 관리 및 계정 제한을 제공합니다. PAM은 사용자 관리 세부 정보가 변경되더라도 `login`, `su`, `ssh` 등의 프로그램이 변경되지 않게 해 줍니다. 사이트 응용 프로그램은 PAM을 사용하여 고유한 계정, 자격 증명, 세션 및 암호 요구 사항을 관리할 수 있습니다. PAM은 이러한 응용 프로그램에 "플러그인"됩니다.

PAM 프레임워크 소개

PAM 프레임워크는 네 부분으로 구성됩니다.

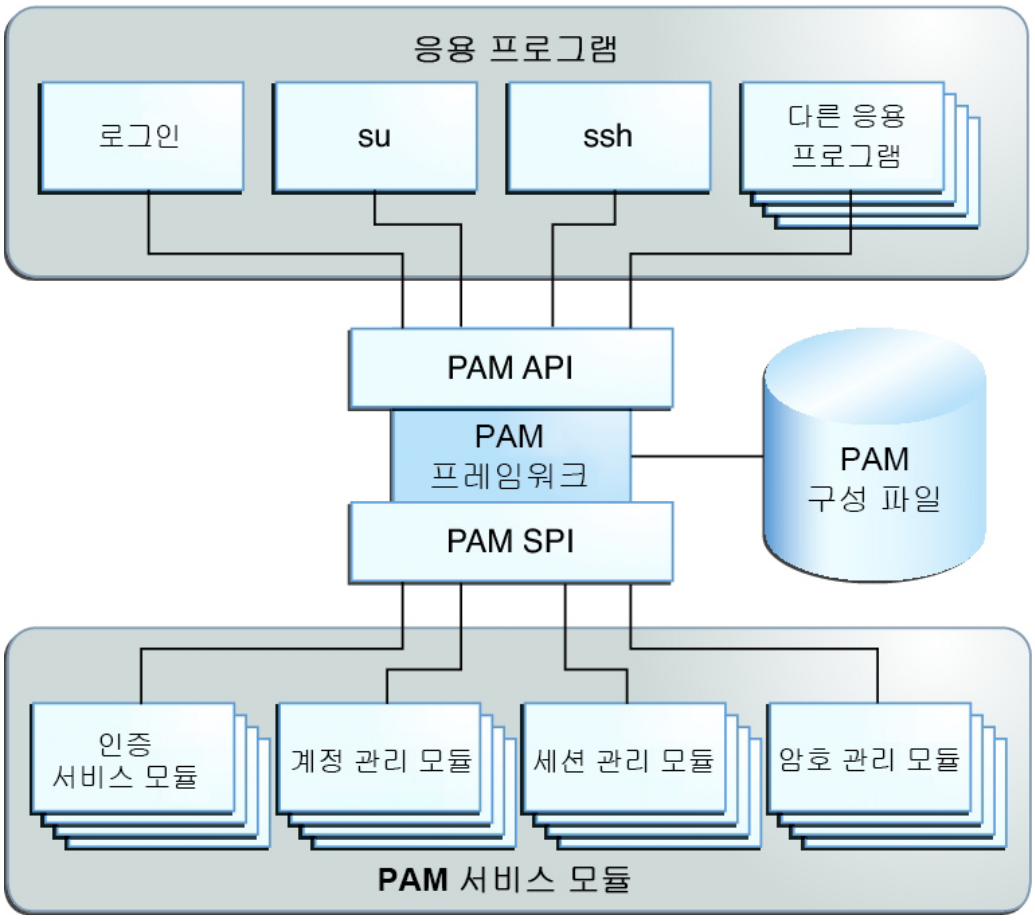
- PAM을 사용하는 응용 프로그램

- PAM 프레임워크
- PAM 서비스 모듈
- PAM 구성(선택한 모듈 및 사용자 지정 포함)

PAM 프레임워크는 인증 관련 작업이 수행되는 일관된 방식을 제공합니다. 이 방식을 통해 응용 프로그램 개발자는 인증 정책의 의미를 알 필요 없이 PAM 서비스를 사용할 수 있습니다. PAM을 사용하면 관리자는 응용 프로그램을 변경하지 않고도 특정 시스템의 요구에 인증 프로세스를 맞출 수 있습니다. 더 나아가 관리자는 PAM 구성을 조정할 수 있습니다.

다음 그림은 PAM 아키텍처를 나타냅니다.

그림 1-1 PAM 아키텍처



PAM 아키텍처는 다음과 같이 작동합니다.

- 응용 프로그램은 PAM API(Application Programming Interface)를 통해 PAM 프레임워크와 통신합니다.

  API 사용에 대한 자세한 내용은 pam(3PAM) 매뉴얼 페이지 및 "Developer's Guide to Oracle Solaris 11 Security "의 3 장, "Writing PAM Applications and Services"을 참조하십시오.
- PAM 서비스 모듈은 PAM SPI(서비스 공급자 인터페이스)를 통해 PAM 프레임워크와 통신합니다. 자세한 내용은 pam_sm(3PAM) 매뉴얼 페이지를 참조하십시오.

  선택한 서비스 모듈에 대한 간략한 설명은 "PAM 서비스 모듈" [35]과 pam.conf(4) 및 pam_user_policy(5) 매뉴얼 페이지를 참조하십시오.

관리자는 하나 이상의 모듈 시리즈를 구성하여 사이트 요구 사항을 관리할 수 있습니다. 이러한 모듈 시리즈를 PAM 스택이라고 합니다. 스택은 순서대로 평가됩니다. 응용 프로그램에 둘 이상의 PAM 스택이 필요한 경우 응용 프로그램 개발자가 서비스 이름을 둘 이상 만들어야 합니다. 예를 들어, sshd 데몬은 PAM 서비스 이름이 여러 개 필요하며 제공합니다. sshd 데몬에 대한 PAM 서비스 이름 목록을 보려면 sshd(1M) 매뉴얼 페이지에서 PAM을 검색하십시오. PAM 스택에 대한 자세한 내용은 "PAM 스택" [31]을 참조하십시오. "PAM 스택 예" [34]는 PAM 인증 스택을 단계별로 보여 줍니다.

PAM 사용 이점

PAM 프레임워크를 사용하면 사용자가 응용 프로그램을 사용하기 위해 충족해야 할 요구 사항을 구성할 수 있습니다. 다음은 PAM이 제공하는 몇 가지 이점입니다.

- 유연한 PAM 구성 정책
  - 서비스 이름별 인증 정책
  - 사이트 차원 PAM 정책 및 사용자별 PAM 정책
  - 관리자가 기본 인증 정책 선택
  - 높은 보안 시스템에서 여러 사용자 요구 사항 적용
- 최종 사용자의 사용 편의성
  - 여러 인증 서비스에 대해 암호가 동일한 경우 암호 재입력 없음
  - 사용자가 여러 명령을 입력할 필요 없이 여러 인증 서비스에 대해 사용자에게 암호를 요구할 수 있는 기능
- 관리자의 구성 편의성
  - PAM 서비스 모듈에 옵션을 전달할 수 있는 기능
  - 응용 프로그램을 변경할 필요 없이 사이트별 보안 정책을 구현할 수 있는 기능

사이트별 PAM 구성 계획

PAM 구성은 그 자체로 표준 보안 정책을 구현하며, 이러한 표준 보안 정책은 인증이 필요한 시스템 서비스(예: login 및 ssh)를 처리합니다. 일부 시스템 서비스에 대해 다른 보안 정책

을 구현하거나 타사 응용 프로그램용 정책을 만들어야 하는 경우 다음 문제를 고려해야 합니다.

- 제공된 구성 파일이 요구 사항을 충족하지 않는지 확인합니다.

  기본 구성을 테스트합니다. `/etc/security/pam_policy` 디렉토리에 있는 사용자별 파일을 테스트합니다. 기본 서비스 이름인 `other`가 요구 사항을 처리하는지 테스트합니다. "PAM 스택 예" [34]는 `other` 스택을 단계별로 보여 줍니다.
- 스택을 수정해야 할 서비스 이름을 모두 식별합니다. 서비스 이름의 PAM 스택을 수정하는 방법에 대한 예는 사이트별 PAM 구성 파일을 만드는 방법 [20]을 참조하십시오.
- PAM 프레임워크를 사용하도록 코딩된 타사 응용 프로그램이 있는 경우 해당 응용 프로그램이 사용하는 PAM 서비스 이름을 확인합니다.
- 각 서비스 이름에 대해 사용할 PAM 모듈을 확인합니다.

  5절의 매뉴얼 페이지에서 PAM 모듈을 검토합니다. 이러한 매뉴얼 페이지는 각 모듈의 작동 방식, 사용 가능한 옵션 및 스택 모듈 사이의 상호 작용에 대해 설명합니다. 선택한 모듈에 대한 간략한 요약은 "PAM 서비스 모듈" [35]을 참조하십시오. 외부 출처에서도 PAM 모듈을 제공합니다.
- 서비스 이름별로 모듈이 실행되는 순서를 확인합니다.
- 각 모듈에 대한 제어 플래그를 선택합니다. 제어 플래그에 대한 자세한 내용은 "PAM 스택" [31]을 참조하십시오. 제어 플래그는 보안에 영향을 줄 수 있습니다.

  시각적으로 보려면 그림 1-2. "PAM 스택: 제어 플래그의 효과" 및 그림 1-3. "PAM 스택: 통합된 값이 결정되는 방식"을 참조하십시오.
- 각 모듈에 필요한 옵션을 선택합니다. 각 모듈의 매뉴얼 페이지에는 해당 모듈에 사용할 수 있는 옵션이 나열되어 있습니다.
- PAM 구성을 사용하여 응용 프로그램 사용을 테스트합니다. `root` 역할, 기타 역할, 권한 있는 사용자 및 일반 사용자로 테스트합니다. 응용 프로그램을 사용하도록 허용되지 않은 사용자가 있는 경우에는 해당 사용자를 테스트합니다.

사용자별 PAM 정책 지정

`pam_user_policy` PAM 모듈을 사용하면 시스템 관리자가 사용자별로 특정 PAM 구성을 지정할 수 있습니다. 해당 모듈이 PAM 스택의 첫번째 모듈이고 사용자의 `pam_policy` 보안 속성에 PAM 구성 파일이 지정된 경우에는 이 파일에 사용자의 PAM 정책이 지정됩니다.

자세한 내용은 다음을 참조하십시오.

- `pam_user_policy`(5) 매뉴얼 페이지
- "PAM 스택" [31]
- 사이트별 PAM 구성 파일을 만드는 방법 [20]
- 예 1-3. "권한 프로파일을 사용하여 사용자별 PAM 정책 설정"

PAM 구성

기본 제공되는 PAM을 있는 그대로 사용할 수 있습니다. 이 절에서는 기본적으로 적용되지 않는 PAM 구성의 예를 제공합니다.

표 1-1 PAM 작업 맵

작업	설명	지침
PAM 설치에 대해 계획합니다.	사이트에 대한 PAM 사용자 정의를 계획하는 방법에 대해 설명합니다.	"사이트별 PAM 구성 계획" [18]
새 PAM 정책을 사용자에게 지정합니다.	여러 서비스에 대한 사용자별 인증 요구 사항을 사용자 정의합니다.	사이트별 PAM 구성 파일을 만드는 방법 [20]
암호화된 홈 디렉토리를 사용하여 사용자를 만듭니다.	암호화된 홈 디렉토리를 만들 수 있도록 PAM 스택을 수정합니다.	예 1-1. "수정된 PAM 스택을 사용하여 암호화된 홈 디렉토리 만들기"
새 PAM 모듈을 추가합니다.	사용자 정의 PAM 모듈을 설치 및 사용자 정의하는 방법을 설명합니다.	PAM 모듈을 추가하는 방법 [22]
기본 PAM 정책이 아닌 PAM 정책을 사용자에게 지정합니다.	Kerberos, LDAP 또는 로그인 조합을 사용하는 사이트의 광범위한 사용자에게 지정할 PAM 정책을 권한 프로파일에 추가하는 방법을 보여 줍니다.	수정된 PAM 정책을 지정하는 방법 [24]
기본 PAM 정책이 아닌 PAM 정책을 사용자에게 지정합니다.	사용자 정의 PAM 스택을 모든 시스템 이미지에 배포합니다.	수정된 PAM 정책을 지정하는 방법 [24]
오류 로깅을 시작합니다.	`syslog`를 통해 PAM 오류 메시지를 기록합니다.	PAM 오류 보고서를 로깅하는 방법 [27]
PAM 오류 문제를 해결합니다.	PAM 구성 오류를 찾고 해결하고 테스트하는 단계를 제공합니다.	PAM 구성 오류 문제를 해결하는 방법 [28]

▼ 사이트별 PAM 구성 파일을 만드는 방법

기본 구성에서는 `ssh` 및 `telnet` 항목 서비스를 `other` 서비스 이름에서 처리합니다. 이 절차에 나오는 PAM 구성 파일은 `ssh` 및 `telnet`에 대한 요구 사항을 변경합니다.

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **새 PAM 정책 구성 파일을 만듭니다.**
   `pfedit` 명령을 사용하여 파일을 만든 다음 `/opt`와 같은 사이트 구성 디렉토리에 가져다 놓습니다. `/etc/security/pam_policy` 디렉토리에 저장해도 됩니다.

   참고 - `/etc/security/pam_policy` 디렉토리에 있는 기존 파일은 수정하지 마십시오.

   파일에 설명 주석을 포함합니다.

   ```
   # pfedit /opt/local_pam/ssh-telnet-conf
   ```

```
#
# PAM configuration which uses UNIX authentication for console logins,
# (see pam.d/login), and LDAP for SSH keyboard-interactive logins
# This stack explicitly denies telnet logins.
#
sshd-kbdint  auth requisite          pam_authtok_get.so.1
sshd-kbdint  auth binding            pam_unix_auth.so.1 server_policy
sshd-kbdint  auth required           pam_unix_cred.so.1
sshd-kbdint  auth required           pam_ldap.so.1
#
telnet auth     requisite    pam_deny.so.1
telnet account  requisite    pam_deny.so.1
telnet session  requisite    pam_deny.so.1
telnet password requisite    pam_deny.so.1
```

2. **파일을 보호합니다.**

   root 소유권과 444 사용 권한으로 파일을 보호합니다.

```
# ls -l /opt/local_pam

total 5
-r--r--r--   1 root         4570 Jun 21 12:08 ssh-telnet-conf
```

3. **정책을 지정합니다.**

   수정된 PAM 정책을 지정하는 방법 [24]을 참조하십시오.

예 1-1 수정된 PAM 스택을 사용하여 암호화된 홈 디렉토리 만들기

기본적으로 zfs_pam_key 모듈은 /etc/security/pam_policy/unix 파일에 없습니다. 이 예에서는 관리자가 unix 버전 PAM 사용자별 정책을 만든 다음 새 버전을 사용하여 해당 홈 디렉토리가 암호화되는 사용자를 만듭니다.

```
# cp /etc/security/pam_policy/unix /opt/local_pam/unix-encrypt
# pfedit /opt/local_pam/unix-encrypt.conf
...
other   auth required           pam_unix_auth.so.1
other   auth required           pam_unix_cred.so.1
## pam_zfs_key auto-creates an encrypted home directory
##
other auth required           pam_zfs_key.so.1 create
```

관리자가 사용자를 추가할 때 이 정책 파일을 사용합니다. 암호화는 파일 시스템에 추가할 수 없습니다. 따라서 파일 시스템을 만들 때 암호화를 설정한 상태에서 만들어야 합니다. 자세한 내용은 zfs_encrypt(1M)을 참조하십시오.

관리자가 사용자를 만들고 암호를 지정합니다.

```
# useradd -K pam_policy=/opt/local_pam/unix-encrypt.conf jill
# passwd jill
New Password: xxxxxxxx
```

```
Re-enter new Password: xxxxxxxx
passwd: password successfully changed for jill
```

그런 다음 해당 사용자로 로그인하여 암호화된 홈 디렉토리를 만듭니다.

```
# su - jill
Password: xxxxxxxx
Creating home directory with encryption=on.
Your login password will be used as the wrapping key.
Oracle Corporation      SunOS 5.11      11.2    July 2014

# logout
```

ZFS 서비스 모듈에 대한 옵션은 pam_zfs_key(5) 매뉴얼 페이지를 참조하십시오.

마지막으로, 새 홈 디렉토리가 암호화된 파일 시스템인지 확인합니다.

```
# mount -p | grep ~jill
rpool/export/home/jill - /export/home/jill zfs - no
rw,devices,setuid,nonbmand,exec,rstchown,xattr,atime
# zfs get encryption,keysource rpool/export/home/jill
NAME                   PROPERTY    VALUE              SOURCE
rpool/export/home/jill  encryption  on                 local
rpool/export/home/jill  keysource   passphrase,prompt  local
```

▼ PAM 모듈을 추가하는 방법

이 절차에서는 새 PAM 모듈을 추가하고 보호하고 테스트하는 방법을 보여 줍니다. 사이트별 보안 정책을 사용하려는 경우나 타사 응용 프로그램을 지원해야 할 경우에 새 모듈이 필요할 수 있습니다. PAM 모듈을 만들려면 "Developer's Guide to Oracle Solaris 11 Security"의 3 장, "Writing PAM Applications and Services"을 참조하십시오.

참고 - PAM 서비스 모듈의 32비트 버전 및 64비트 버전을 설치해야 합니다.

시작하기 전에 "사이트별 PAM 구성 계획" [18]을 완료합니다.

root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **디스크에 두 버전의 PAM 서비스 모듈을 모두 설치한 후 보호합니다.**

   root 소유권 및 444 사용 권한으로 모듈 파일을 보호합니다.

```
# cd /opt/pam_modules
# ls -lR
.:
total 4
-r--r--r--   1 root     root        4570 Nov 27 12:34 pam_app1.so.1
```

```
drwxrwxrwx   2 root     root           3 Nov 27 12:38 sparcv9

./64:
total 1
-r--r--r--   1 root     root        4862 Nov 27 12:38 pam_app1.so.1
```

32비트 모듈은 /opt/pam_modules 디렉토리에 있고 64비트 모듈은 64 하위 디렉토리에 있습니다.

2. **모듈을 적절한 PAM 구성 파일에 추가합니다.**

   다음 예에는 app1이라는 새 응용 프로그램에 대한 모듈이 나옵니다. 모듈의 서비스 이름은 응용 프로그램 이름과 같습니다. app1 service-name 파일을 /etc/pam.d 디렉토리에 만듭니다. 이 파일의 첫번째 항목은 app1 서비스를 개별 사용자에게 지정할 수 있도록 해줍니다.

```
# cd /etc/pam.d
# pfedit app1
...
# PAM configuration
#
# app1 service
#
auth definitive         pam_user_policy.so.1
auth required           /opt/pam_modules/$ISA/pam.app1.so.1  debug
```

   모듈 경로의 $ISA 토큰은 PAM 프레임워크가 호출 응용 프로그램에 대해 서비스 모듈의 적절한 32비트 또는 64비트 아키텍처 버전을 가리키도록 지시합니다. 32비트 응용 프로그램의 경우 */a/b/*$ISA/*module*.so는 */a/b/module*.so가 되고, 64비트 응용 프로그램의 경우에는 */a/b/*64/*module*.so가 됩니다. 이 예제에서는 /opt/pam_modules 디렉토리에 32비트 pam.app1.so.1 서비스 모듈을 설치했고 /opt/pam_modules/64 디렉토리에 64비트 모듈을 설치했습니다.

   자세한 내용은 pfedit(1M) 및 pam.conf(4) 매뉴얼 페이지를 참조하십시오.

   선택한 사용자로만 app1 PAM 정책을 제한하려는 경우 예 1-2 "사용자별 PAM 정책 파일에 새 모듈 추가"를 참조하십시오.

3. **새 서비스를 테스트합니다.**

   login 또는 ssh를 사용하여 직접 로그인합니다. 그런 다음 새 모듈로 인해 영향을 받는 명령을 실행합니다. 해당 명령을 사용할 수 있도록 허용되는 사용자와 거부되는 사용자를 테스트합니다. 문제 해결에 대한 자세한 내용은 PAM 구성 오류 문제를 해결하는 방법 [28]을 참조하십시오.

4. **정책을 지정합니다.**

   수정된 PAM 정책을 지정하는 방법 [24]을 참조하십시오.

예 1-2 사용자별 PAM 정책 파일에 새 모듈 추가

이 예에서 app1 서비스는 일부 사용자만 사용하므로 관리자가 해당 서비스를 사용자별 정책으로 추가합니다.

```
# cd /etc/pam.d
# cp app1 /opt/local_pam/app1-conf
# pfedit /opt/local_pam/app1-conf

## app1 service
##
app1 auth definitive         pam_user_policy.so.1
app1 auth required           /opt/pam_modules/$ISA/pam_app1.so.1  debug
```

먼저 app1 파일을 pam.d 디렉토리에서 삭제합니다.

```
# rm /etc/pam.d/app1
```

그런 다음 app1-conf 정책을 시스템 관리자의 PAM 정책에 추가합니다.

```
# rolemod -K pam_policy=/opt/local_pam/app1-conf sysadmin
```

예 1-3 권한 프로파일을 사용하여 사용자별 PAM 정책 설정

이 예에서는 pam_policy 보안 속성을 사용하여 서로 다른 서비스의 사용자가 인증되도록 합니다. any PAM 정책 파일은 /etc/security/pam_policy 디렉토리에 제공됩니다. 파일의 주석은 이 정책에 대해 설명합니다.

이 디렉토리의 파일은 수정하지 마십시오.

```
# profiles -p "PAM Per-User Policy of Any" \
'set desc="Profile which sets pam_policy=any";
set pam_policy=any; exit;'
```

권한 프로파일을 지정하려면 수정된 PAM 정책을 지정하는 방법 [24]을 참조하십시오.

▼ 수정된 PAM 정책을 지정하는 방법

이 절차에서는 모든 시스템 이미지에 기본 PAM 정책이 아닌 정책을 구성합니다. 모든 파일을 복사한 후 새 PAM 정책 또는 수정된 PAM 정책을 개별 사용자나 모든 사용자에게 지정할 수 있습니다.

시작하기 전에 새 정책을 구현하는 PAM 구성 파일을 수정 및 테스트합니다.

root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **기본 PAM 파일이 아닌 PAM 파일을 모든 시스템 이미지에 추가합니다.**
   모든 새 PAM 모듈과 새 PAM 구성 파일 및 수정된 PAM 구성 파일을 모든 시스템 이미지에 추가해야 합니다.

   a. **먼저 모든 새 PAM 모듈을 모든 시스템 이미지에 추가합니다.**

i. **32비트 PAM 모듈을 해당 아키텍처 디렉토리에 추가합니다.**

ii. **64비트 PAM 모듈을 해당 아키텍처 디렉토리에 추가합니다.**

디렉토리 설정에 대한 예는 1단계 in PAM 모듈을 추가하는 방법 [22]를 참조하십시오.

b. **그 다음에는 모든 새 PAM 구성 파일을 모든 시스템 이미지에 추가합니다.**

예를 들어, `/opt/local_pam/ssh-telnet-conf` 파일을 모든 시스템 이미지에 추가합니다.

c. **그런 다음 모든 수정된 PAM 구성 파일을 모든 시스템 이미지에 추가합니다.**

예를 들어, 수정된 `/etc/pam.conf` 파일 및 모든 수정된 `/etc/pam.d/`*service-name-files*를 모든 시스템 이미지에 복사합니다.

2. **기본 PAM 정책이 아닌 PM 정책을 모든 사용자에게 지정합니다.**

a. **다음 방법 중 하나로 `policy.conf` 파일을 수정합니다.**

- **PAM 구성 파일을 `policy.conf` 파일의 `PAM_POLICY` 키워드에 추가합니다.**

```
# pfedit /etc/security/policy.conf
...
# PAM_POLICY=
PAM_POLICY=/opt/local_pam/ssh-telnet-conf
...
```

- **권한 프로파일을 `policy.conf` 파일의 `PROFS_GRANTED` 키워드에 추가합니다.**

예를 들어, 예 1-3. “권한 프로파일을 사용하여 사용자별 PAM 정책 설정”에 나온 PAM 사용자별 정책 Any 권한 프로파일을 지정합니다.

```
# pfedit /etc/security/policy.conf
...
AUTHS_GRANTED=
# PROFS_GRANTED=Basic Solaris User
PROFS_GRANTED=PAM Per-User Policy of Any,Basic Solaris User
...
```

b. **수정된 `policy.conf` 파일을 모든 시스템 이미지에 복사합니다.**

3. **기본 PAM 정책이 아닌 PAM 정책을 개별 사용자에게 지정하려는 경우에는 사용자에게 직접 정책을 지정하거나 사용자에게 지정된 권한 프로파일에 정책을 추가합니다.**

- **개별 사용자에게 직접 PAM 정책을 지정합니다.**

```
# usermod -K pam_policy="/opt/local_pam/ssh-telnet-conf" jill
```

- **PAM 정책을 권한 프로파일에 포함하고 프로파일을 개별 사용자에게 지정합니다.**
  이 예에서는 `ldap` PAM 정책을 사용합니다.

  ```
  # profiles -p "PAM Per-User Policy of LDAP" \
  'set desc="Profile which sets pam_policy=ldap";
  set pam_policy=ldap; exit;'
  ```

  그런 다음 권한 프로파일을 사용자에게 지정합니다.

  ```
  # usermod -P +"PAM Per-User Policy of LDAP" jill
  ```

예 1-4 선택한 사용자로 `ktelnet` PAM 스택 제한

관리자는 Kerberos 영역에서 `telnet`을 사용할 수 있는 사용자 수를 제한할 수 있습니다. 이를 위해 관리자는 `telnet` 서비스를 사용으로 설정하기 전에 기본 `ktelnet` 구성 파일을 변경한 다음 기본 `ktelnet` 파일을 `pam_policy` 디렉토리에 가져다 놓습니다.

먼저 사용자별 `ktelnet` 파일을 구성합니다.

```
# cp /etc/pam.d/ktelnet /etc/security/pam_policy/ktelnet-conf
# pfedit /etc/security/pam_policy/ktelnet-conf
...
# Kerberized telnet service
#
ktelnet  auth required          pam_unix_cred.so.1
ktelnet  auth required          pam_krb5.so.1
```

`4440` 사용 권한으로 파일을 보호합니다.

```
# chmod 444 /etc/security/pam_policy/ktelnet-conf
# ls -l /etc/security/pam_policy/ktelnet-conf
-r--r--r--   1 root     root         228 Nov 27 15:04 ktelnet-conf
```

그런 다음 `pam.d` 디렉토리의 `ktelnet` 파일을 수정합니다.

- 첫번째 항목은 사용자별 지정을 가능하게 합니다.
- 두번째 항목은 관리자가 `pam_policy=ktelnet`을 지정한 경우가 아니면 `ktelnet` 사용을 거부합니다.

```
# cp /etc/pam.d/ktelnet /etc/pam.d/ktelnet.orig
              # pfedit /etc/pam.d/ktelnet
...
# Denied Kerberized telnet service
#
auth definitive         pam_user_policy.so.1
auth required          pam_deny.so.1
```

관리자가 권한 있는 사용자, 일반 사용자 및 루트 역할로 구성을 테스트합니다. 구성이 전달되면 관리자가 `telnet` 서비스를 사용으로 설정하고 사용자별 정책을 Kerberos 관리자에게 지정합니다.

```
# svcadm enable telnet
# rolemod -S ldap -K pam_policy=ktelnet-conf kerbadmin
```

관리자가 수정된 파일을 모든 Kerberos 서버에 복사하고 해당 서버에서 `telnet`을 사용으로 설정합니다.

▼ PAM 오류 보고서를 로깅하는 방법

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **온라인 상태의 `system-log` 서비스 인스턴스를 확인합니다.**

```
# svcs system-log
STATE          STIME    FMRI
disabled       13:11:55 svc:/system/system-log:rsyslog
online         13:13:27 svc:/system/system-log:default
```

2. **필요한 로깅 레벨에 맞게 `syslog.conf` 파일을 구성합니다.**

로깅 레벨에 대한 자세한 내용은 `syslog.conf`(4) 매뉴얼 페이지의 `DESCRIPTION` 절을 참조하십시오. 대부분의 PAM 오류 보고는 `LOG_AUTH` 기능을 통해 수행됩니다.

예를 들어, 디버그 출력용 파일을 만듭니다.

```
# touch /var/adm/pam_debuglog
```

그런 다음 `syslog.conf` 항목을 추가하여 디버그 출력을 해당 파일로 보냅니다.

참고 - `rsyslog` 서비스 인스턴스가 온라인이면 `rsyslog.conf` 파일을 수정합니다.

```
# pfedit /etc/syslog.conf
...
*.debug         /var/adm/pam_debuglog
...
```

3. **`system-log` 서비스에 대한 구성 정보를 새로 고칩니다.**

```
# svcadm refresh system-log:default
```

참고 - `rsyslog` 서비스가 온라인이면 `system-log:rsyslog` 서비스 인스턴스를 새로 고칩니다.

▼ PAM 구성 오류 문제를 해결하는 방법

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **문제를 해결하려는 각 PAM 항목에 대해 `debug` 옵션을 추가합니다.**

   예를 들어, `/etc/pam.d/cron` 파일의 다음 항목은 서비스에 대한 디버그 출력을 만듭니다.

   ```
   account definitive      pam_user_policy.so.1     debug
   account required        pam_unix_account.so.1    debug
   ```

2. **적절한 레벨에 PAM 오류를 로깅하고 `syslog` 데몬을 새로 고칩니다.**

   자세한 내용은 PAM 오류 보고서를 로깅하는 방법 [27]을 참조하십시오.

3. **손상된 PAM 구성으로 인한 문제가 발생한 경우에는 다음을 수행합니다.**

   a. **한 터미널 창에서 응용 프로그램을 실행하고 다른 창에서 PAM 구성 파일을 수정합니다.**

   b. **응용 프로그램 창에서 변경 사항을 테스트하여 오류가 수정되는지 확인합니다.**

4. **손상된 PAM 구성으로 인해 로그인할 수 없는 문제가 발생한 경우에는 단일 사용자 모드로 부트한 다음 파일을 수정하고 재부트한 후 테스트합니다.**

   - **SPARC 시스템을 부트하려면 PROM 프롬프트에 다음 명령을 입력합니다.**

     ```
     ok > boot -s
     ```

   - **x86 시스템을 부트하려면 `-s` 옵션을 GRUB 메뉴의 커널 옵션에 추가합니다.**

   자세한 내용은 `boot`(1M) 및 `grub`(5) 매뉴얼 페이지를 참조하십시오.

5. **오류가 수정되었는지 확인합니다.**

   `login` 또는 `ssh`를 사용하여 직접 로그인합니다. 일반 사용자, 권한 있는 사용자 및 역할이 영향을 받는 명령을 사용할 수 있는지 테스트합니다.

PAM 구성 참조

이 절에서는 PAM 스택을 비롯한 PAM에 대해 자세히 설명합니다.

PAM 구성 파일

`login` 및 `ssh`와 같이 PAM 프레임워크를 사용하는 시스템 응용 프로그램은 `/etc/pam.d` 디렉토리에 있는 PAM 구성 파일에 구성됩니다. `/etc/pam.conf` 파일도 사용될 수 있습니다. 이러한 파일을 변경하면 시스템의 모든 사용자에게 영향을 줍니다.

`/etc/security/pam_policy` 디렉토리에도 PAM 구성 파일이 보관됩니다. 이러한 파일은 여러 서비스를 취급하며 사용자별로 지정할 수 있도록 설계되었습니다. 이 디렉토리의 파일은 수정하지 않아야 합니다.

- `/etc/pam.d` **디렉토리** - 와일드카드 파일 `other`를 비롯한 서비스별 PAM 구성 파일이 있습니다. 응용 프로그램에 대한 서비스를 추가하려면 *service-name* 파일을 추가합니다. 이 파일은 응용 프로그램에 사용되는 서비스 이름입니다. 적합한 경우 응용 프로그램에서 `other` 파일의 PAM 스택을 사용할 수 있습니다.

  `/etc/pam.d` 디렉토리의 서비스 파일은 대부분의 PAM 구현에 있는 기본 구성을 제공합니다. 이러한 파일은 `pkg`(5) 매뉴얼 페이지에 설명된 IPS 방식을 사용하여 자체 어셈블됩니다. 이 기본 동작은 다른 플랫폼간 PAM 응용 프로그램과의 상호 운용성을 간소화합니다. 자세한 내용은 `pam.conf`(4) 매뉴얼 페이지를 참조하십시오.
- `/etc/pam.conf` **파일** - 레거시 PAM 구성 및 정책 파일입니다. 이 파일은 비어 있는 상태로 제공됩니다. 선호되는 PAM 구성 방식에서는 `/etc/pam.d` 디렉토리의 파일을 사용합니다. 자세한 내용은 `pam.conf`(4) 매뉴얼 페이지를 참조하십시오.
- `/etc/security/pam_policy` **디렉토리** - 여러 서비스에 대한 정책이 포함된 PAM 정책 파일이 있습니다. 필요에 따라 이러한 파일을 개별 사용자, 개별 사용자 그룹 또는 모든 사용자에게 지정할 수 있습니다. 이렇게 지정할 경우 `pam.conf` 또는 `/etc/pam.d` 디렉토리에 있는 PAM 구성 파일을 겹쳐쓰게 됩니다. 이러한 파일은 수정하지 마십시오. 사용자별 파일을 추가하려면 사이트별 PAM 구성 파일을 만드는 방법 [20]을 참조하십시오.
  사용자별 파일에 대한 자세한 내용은 `pam_user_policy`(5) 매뉴얼 페이지를 참조하십시오.

보안 관리자는 모든 PAM 구성 파일을 관리합니다. 항목 순서가 잘못되면, 즉 PAM 스택이 올바르지 않으면 예측하지 못한 역효과가 발생할 수 있습니다. 예를 들어, 잘못 구성된 파일은 사용자를 잠그게 되므로 복구하려면 단일 사용자 모드가 필요할 수 있습니다. 자세한 내용은 "PAM 스택" [31] 및 PAM 구성 오류 문제를 해결하는 방법 [28]을 참조하십시오.

PAM 구성 검색 순서

PAM 프레임워크에 대한 응용 프로그램 호출은 다음과 같은 순서로 구성된 PAM 서비스를 검색합니다.

1. `/etc/pam.conf`에서 서비스 이름을 조회합니다.
2. `/etc/pam.d/`*service-name* 파일에 있는 특정 서비스가 사용됩니다.

3. /etc/pam.conf 파일에서 서비스 이름 other를 확인합니다.
4. /etc/pam.d/other 파일이 사용됩니다.

이 순서에 따라 기존 /etc/pam.conf 파일이 /etc/pam.d에 있는 서비스별 PAM 구성 파일과 함께 작동할 수 있습니다.

PAM 구성 파일 구문

pam.conf 파일 및 PAM 사용자별 파일은 pam.d 디렉토리에 있는 서비스별 파일과는 다른 구문을 사용합니다.

- /etc/pam.conf 파일 및 /etc/security/pam_policy 파일의 항목은 다음 두 형식 중 하나입니다.

  service-name module-type control-flag module-path module-options

  service-name module-type include *path-to-included-PAM-configuration*

- /etc/pam.d 디렉토리의 *service-name* 파일에 있는 항목에는 서비스 이름이 생략됩니다. 파일 이름이 서비스 이름을 제공합니다.

  module-type control-flag module-path module-options

  module-type include *path-to-included-PAM-configuration*

PAM 구성 파일 구문의 항목은 다음과 같습니다.

service-name

서비스의 이름(대소문자 구분 안함)입니다(예:login 또는 ssh). 응용 프로그램에서는 응용 프로그램이 제공하는 서비스에 대해 서로 다른 서비스 이름을 사용할 수 있습니다. 예를 들어 sshd 데몬이 제공하는 다양한 서비스의 서비스 이름을 확인하려면 sshd(1M) 매뉴얼 페이지에서 PAM을 검색하십시오.

미리 정의된 서비스 이름 "other"는 특정 서비스 구성이 제공되지 않은 경우 사용되는 기본 서비스 이름입니다.

module-type

서비스의 유형(즉, auth, account, session 또는 password)을 나타냅니다.

control-flag

서비스에 대한 성공 또는 실패 값을 결정하는 모듈의 역할을 나타냅니다. 유효한 제어 플래그는 “PAM 스택” [31]을 참조하십시오.

module-path

모듈 유형을 구현하는 모듈의 경로입니다. 경로 이름이 절대 경로가 아닌 경우 경로 이름은 /usr/lib/security/$ISA/ 경로에 대한 상대 경로로 간주됩니다. $ISA 매크로 또는

토큰은 PAM 프레임워크가 모듈 경로의 아키텍처 특정 디렉토리를 검색하도록 지시합니다.

module-options

서비스 모듈로 전달될 수 있는 `nowarn` 및 `debug`와 같은 옵션입니다. 모듈의 매뉴얼 페이지에서는 해당 모듈에 대한 옵션을 설명합니다.

path-to-included-PAM-configuration

PAM 구성 파일 또는 `/usr/lib/security` 디렉토리를 기준으로 한 파일 이름을 지정합니다.

PAM 스택

응용 프로그램이 다음 함수 중 하나를 호출하면 PAM 프레임워크는 PAM 구성 파일을 읽고 해당 응용 프로그램의 PAM 서비스 이름을 구현하는 모듈을 확인합니다.

- `pam_authenticate`(3PAM)
- `pam_acct_mgmt`(3PAM)
- `pam_setcred`(3PAM)
- `pam_open_session`(3PAM)
- `pam_close_session`(3PAM)
- `pam_chauthtok`(3PAM)

구성 파일에 모듈이 하나만 포함된 경우 해당 모듈의 결과가 작업의 결과를 결정합니다. 예를 들어, `passwd` 응용 프로그램에 대한 기본 인증 작업에는 `/etc/pam.d/passwd` 파일에 하나의 모듈(`pam_passwd_auth.so.1`)이 포함됩니다.

```
auth required           pam_passwd_auth.so.1
```

반면 서비스를 구현하는 모듈이 여러 개인 경우 해당 모듈을 스택이라 합니다. 즉, PAM 스택이 해당 서비스 이름에 대해 존재합니다. 예를 들어, `/etc/pam.d/login` 샘플 서비스에 있는 항목을 생각해 보십시오.

```
auth definitive         pam_user_policy.so.1
auth requisite          pam_authtok_get.so.1
auth required           pam_unix_auth.so.1
auth required           pam_dhkeys.so.1
auth required           pam_unix_cred.so.1
auth required           pam_dial_auth.so.1
```

이러한 항목은 `login` 서비스 이름에 대한 `auth` 스택을 만듭니다. 이 스택의 결과를 결정하려면 개별 모듈의 결과 코드에 통합 프로세스가 필요합니다.

통합 프로세스에서 모듈은 파일에 나오는 순서대로 실행됩니다. 각 성공 또는 실패 코드는 모듈의 제어 플래그에 따라 전체 결과에 통합됩니다. 제어 플래그는 스택의 조기 종료를 유발

할 수 있습니다. 예를 들어, `requisite` 또는 `definitive` 모듈이 실패하면 스택이 종료됩니다. 이전 실패한 없는 경우 `sufficient`, `definitive` 또는 `binding` 모듈이 성공하면 스택이 종료됩니다. 스택이 처리된 후 개별 결과는 하나의 전체 결과로 합쳐서 응용 프로그램에 전달됩니다. 플로우를 그림으로 보려면 그림 1-2. “PAM 스택: 제어 플래그의 효과” 및 그림 1-3. “PAM 스택: 통합된 값이 결정되는 방식”을 참조하십시오.

제어 플래그는 PAM 모듈이 성공 또는 실패를 결정하는 데 수행하는 역할을 나타냅니다. 제어 플래그 및 효과는 다음과 같습니다.

- **Binding** - 기록된 이전 실패가 없는 경우 binding 모듈의 요구 사항 충족에 성공하면 응용 프로그램에 즉시 성공을 반환합니다. 이러한 조건이 충족되지 않는다면 모듈은 더 이상 실행되지 않습니다.

  실패의 경우 required 실패가 기록되고 모듈 처리가 계속됩니다.
- **Definitive** - 기록된 이전 실패가 없는 경우 definitive 모듈의 요구 사항 충족에 성공하면 응용 프로그램에 즉시 성공을 반환합니다.

  이전 실패가 기록된 경우 모듈이 추가로 실행되지 않고 해당 실패가 응용 프로그램에 즉시 반환됩니다. 실패의 경우 모듈의 추가 실행 없이 즉시 오류가 반환됩니다.
- **Include** - PAM 스택의 이 시점에서 사용될 별도의 PAM 구성 파일에서 행을 추가합니다. 이 플래그는 성공이나 실패 동작을 제어하지 않습니다. 새 파일이 읽혀지면 PAM `include` 스택이 증가됩니다. 새 파일에서 스택 확인이 완료되면 `include` 스택 값이 감소합니다. 파일의 끝에 도달하고 PAM `include` 스택이 `0`이면 스택 처리가 종료됩니다. PAM `include` 스택에 대한 최대 수는 `32`입니다.
- **Optional** - optional 모듈의 요구 사항을 충족하는 성공은 서비스를 사용하는 데 필요하지 않습니다.

  실패의 경우 optional 실패가 기록됩니다.
- **Required** - reauired 모듈의 요구 사항을 충족하는 성공은 스택이 성공하는 데 필요합니다. 스택에 대한 최종 성공은 실패를 보고한 binding 또는 required 모듈이 없을 경우에만 반환됩니다.

  실패의 경우 이 서비스에 대한 나머지 모듈이 실행된 후 오류가 반환됩니다.
- **Requisite** - requisite 모듈의 요구 사항을 충족하는 성공은 스택이 성공하는 데 필요합니다. 스택이 응용 프로그램에 성공을 반환할 수 있으려면 스택의 모든 requisite 모듈이 성공을 반환해야 합니다.

  실패의 경우 모듈의 추가 실행 없이 즉시 오류가 반환됩니다.
- **Sufficient** - 기록된 이전 required 실패가 없는 경우 sufficient 모듈의 성공은 모듈의 추가 실행 없이 응용 프로그램에 즉시 성공을 반환합니다.

  실패의 경우 optional 실패가 기록됩니다.

다음 두 연결된 다이어그램은 통합 프로세스에서 결과가 어떻게 결정되는지 보여 줍니다.

- 첫번째 다이어그램은 제어 플래그의 각 유형에 대해 성공 또는 실패가 어떻게 기록되는지 보여 줍니다. 두번째 다이어그램은 결과를 보여 줍니다.
- 두번째 다이어그램은 통합된 값이 어떻게 결정되는지 보여줍니다. Optional 실패 및 required 실패는 실패를 반환하고 성공은 성공을 반환합니다. 이러한 반환 코드를 처리하는 방법은 응용 프로그램이 결정합니다.

그림 1-2 PAM 스택: 제어 플래그의 효과

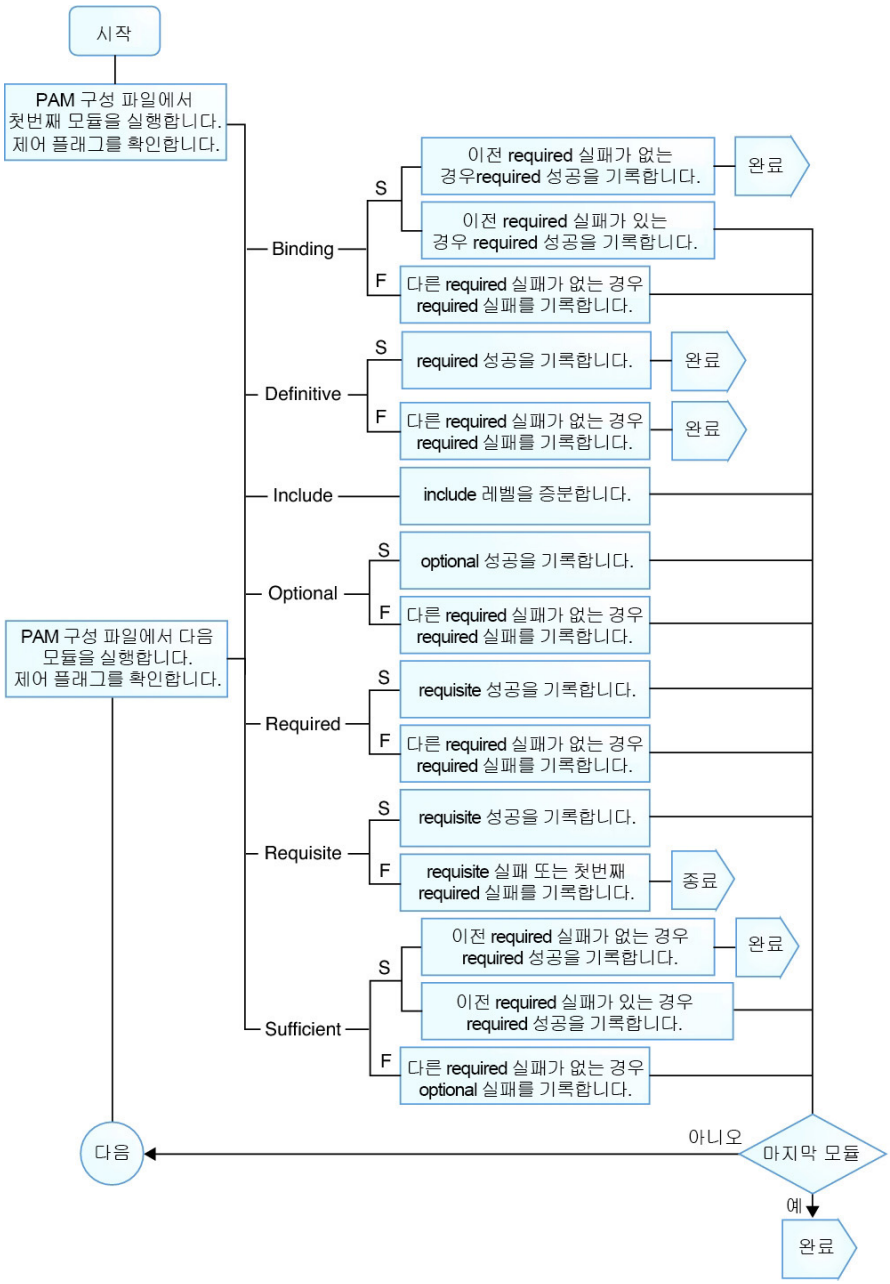
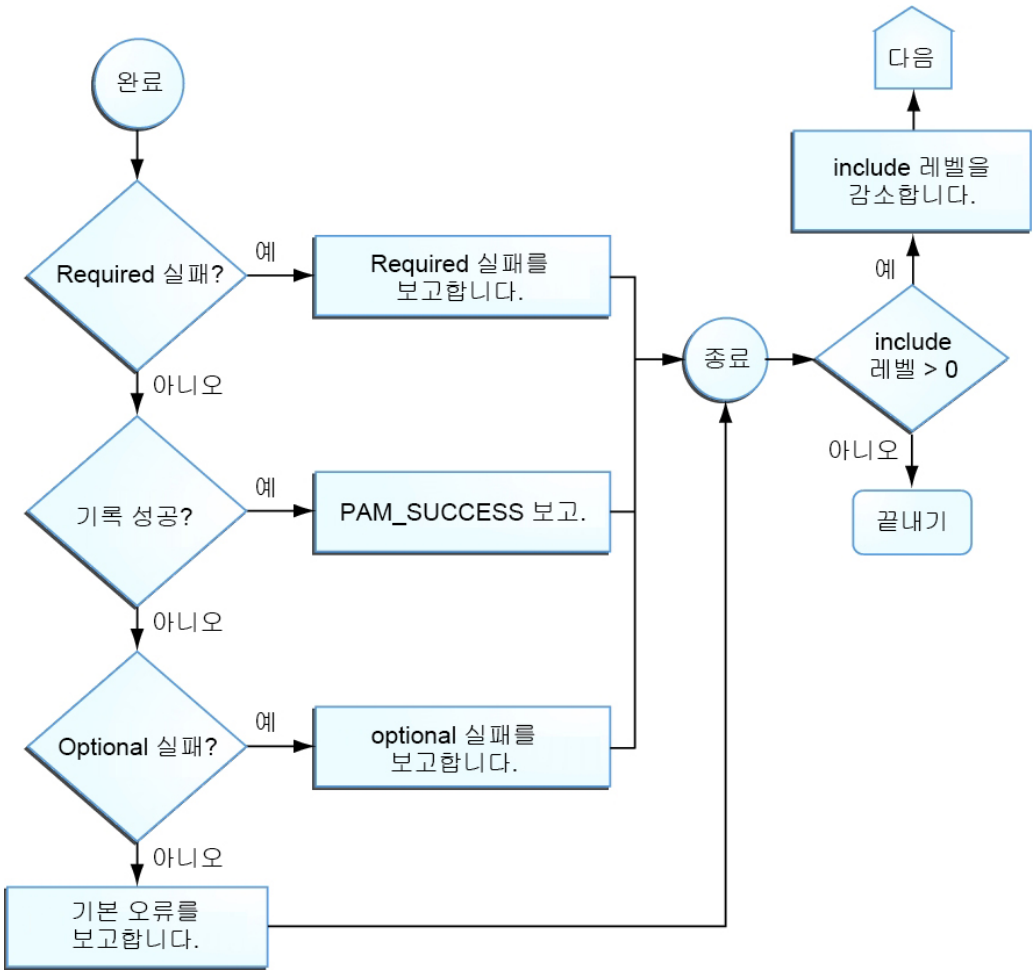


그림 1-3 PAM 스택: 통합된 값이 결정되는 방식



PAM 스택 예

다음 예에서는 샘플 `/etc/pam.d/other` 파일의 인증 관리를 위한 기본 정의를 보여 줍니다. 이러한 정의는 서비스별 인증 정의가 구성되지 않은 경우에 인증에 사용됩니다.

```
##
# Default definitions for Authentication management
# Used when service name is not explicitly mentioned for authentication
```

```
#
auth definitive         pam_user_policy.so.1
auth requisite          pam_authtok_get.so.1
auth required           pam_dhkeys.so.1
auth required           pam_unix_auth.so.1
auth required           pam_unix_cred.so.1
```

먼저 pam_user_policy.so 모듈을 사용하여 사용자의 PAM 정책을 확인합니다. definitive 제어 플래그는 구성된 PAM 스택에 대한 평가가 성공할 경우 이 시점에서 검사된 다른 모듈이 없기 때문에 응용 프로그램에 성공을 반환하도록 결정합니다. 구성된 PAM 스택 평가가 실패할 경우에는 응용 프로그램에 실패 코드가 반환되고 추가 검사가 수행되지 않습니다. 사용자별 PAM 정책이 이 사용자에 대해 지정되지 않았으면 다음 모듈이 실행됩니다.

사용자별 PAM 정책이 이 사용자에 대해 지정되지 않았으면 pam_authtok_get 모듈이 실행됩니다. 이 모듈에 대한 제어 플래그는 requisite로 설정됩니다. pam_authtok_get을 실패할 경우 인증 프로세스가 종료되고 응용 프로그램에 실패가 반환됩니다.

pam_authtok_get이 실패하지 않으면 다음 3개의 모듈이 실행됩니다. 이러한 모듈은 required 제어 플래그로 구성되므로 개별 실패가 반환되는지 여부에 관계없이 통합 프로세스가 계속됩니다. pam_unix_cred가 실행된 후에는 남은 모듈이 없습니다. 이때 모든 모듈이 성공하면 응용 프로그램에 성공이 반환됩니다. pam_dhkeys, pam_unix_auth 또는 pam_unix_cred가 실패를 반환한 경우 응용 프로그램에 실패가 반환됩니다.

PAM 서비스 모듈

이 절에서는 선택한 PAM 서비스 모듈을 나열합니다. 모듈은 매뉴얼 페이지별로 나열되고 그 뒤에 모듈의 용도와 모듈에 대한 간략한 설명이 나옵니다. 자세한 내용은 매뉴얼 페이지를 읽어보십시오.

Oracle Solaris에서 제공하는 모든 PAM 서비스 모듈 목록은 매뉴얼 페이지의 5절을 참조하십시오. 새로운 모듈이 정기적으로 추가됩니다. 예를 들어, 이 릴리스에서는 Windows 시스템 인증에 사용할 수 있는 수많은 모듈이 추가되었습니다. 사이트에서 타사 PAM 모듈을 추가할 수도 있습니다.

pam_allow(5) — 모든 호출에 대해 PAM_SUCCESS를 반환합니다. pam_deny(5) 매뉴얼 페이지도 참조하십시오.

pam_authtok_check(5) — 암호 변경에 대해 암호 토큰의 유효성을 검사합니다.

pam_authtok_get(5) — PAM 스택에 암호 프롬프트 기능을 제공합니다.

pam_authtok_store(5) — PAM_USER에 대한 암호 토큰을 업데이트합니다.

pam_deny(5) — 모든 호출에 대해 모듈 유형 기본 실패 반환 코드를 반환합니다. pam_allow(5) 매뉴얼 페이지도 참조하십시오.

pam_dhkeys(5)	두 PAM 서비스에 보안 RPC 인증 및 보안 RPC 인증 토큰 관리 기능을 제공합니다.
pam_krb5(5)	Kerberos 사용자의 ID를 확인하고 Kerberos 자격 증명 캐시를 관리하는 기능을 제공합니다.
pam_krb5_migrate(5)	PAM_USER를 클라이언트의 로컬 Kerberos 영역으로 마이그레이션하는 데 도움을 줍니다.
pam_ldap(5)	구성된 LDAP 디렉토리 서버를 통한 PAM 인증 및 계정 관리 스택에 필요한 기능을 제공합니다.
pam_list(5)	이 호스트에서 사용자 계정의 유효성을 검사하는 기능을 제공합니다. 이 유효성 검사는 호스트에 있는 사용자 및 넷 그룹 목록을 기반으로 합니다.
pam_passwd_auth(5)	암호 스택에 인증 기능을 제공합니다.
pam_pkcs11(5)	사용자가 X.509 인증서 및 PKCS#11 토큰에 저장된 해당 전용 개인 키를 사용하여 시스템에 로그인할 수 있도록 해줍니다.
pam_roles(5)	사용자가 역할을 맡을 권한이 부여되었는지 확인하고 역할에 의한 직접 로그인을 금지합니다.
pam_smb_passwd(5)	로컬 Oracle Solaris 사용자에 대한 SMB 암호 변경 또는 추가를 지원합니다. smb(4) 매뉴얼 페이지도 참조하십시오.
pam_smbfs_login(5)	Oracle Solaris 클라이언트와 해당 CIFS/SMB 서버 간에 암호를 동기화합니다.
pam_tsol_account(5)	레이블과 관련된 Trusted Extensions 계정 제한을 확인합니다.
pam_tty_tickets(5)	이전 인증 성공으로 생성된 티켓을 검사하기 위한 방식을 제공합니다.
pam_unix_account(5)	사용자 계정이 잠기거나 만료되지 않았는지 확인하고 사용자 암호를 변경할 필요가 없음을 확인하는 기능을 제공합니다. access_times 및 access_tz에 대한 검사를 포함합니다.
pam_unix_auth(5)	암호가 PAM_USER에 대해 올바른 암호인지 확인하는 기능을 제공합니다.
pam_unix_cred(5)	사용자 자격 증명 정보를 설정하는 기능을 제공합니다. 자격 증명 기능과는 독립적으로 인증 기능을 대체할 수 있도록 해줍니다.
pam_unix_session(5)	세션을 열거나 닫고 /var/adm/lastlog 파일을 업데이트합니다.

`pam_user_policy(5)` 사용자별 PAM 구성을 호출합니다.

`pam_zfs_key(5)` 사용자의 암호화된 홈 디렉토리에 대한 ZFS 암호화 문장암호를 로드 및 변경하는 기능을 제공합니다.

♦ ♦ ♦ 2 장

2 Kerberos 서비스 정보

이 장에서는 Kerberos 서비스에 대해 소개합니다. 이 장은 다음 정보를 포함합니다.

Kerberos 서비스란?

Kerberos 서비스는 네트워크를 통해 보안 트랜잭션을 제공하는 클라이언트-서버 구조입니다. 이 서비스는 무결성 및 프라이버시를 비롯하여 강력한 사용자 인증을 제공합니다. 인증은 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 권한 부여 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`, `ssh` 또는 NFS 파일 시스템의 데이터에 액세스하기 위한 명령)을 사용할 때마다 인증할 필요가 없습니다. 따라서 네트워크를 통해 암호를 전송할 경우 암호가 인터셉트될 가능성이 있는데, 이 서비스를 사용할 때마다 이러한 방식으로 암호를 전송할 필요가 없습니다.

Oracle Solaris의 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 버전이 매우 친숙할 것입니다. Kerberos V5 프로토콜은 사실상 네트워크 보안을 위한 업계 표준이기 때문에 Oracle Solaris 버전에서는

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 및 `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 네트워크 요청만 가능하도록 Kerberos 서비스를 구성할 수도 있습니다.

Kerberos 서비스의 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 선택할 수 있습니다.

Kerberos 서비스의 작동 방식

이 절에서는 Kerberos 인증 시스템에 대한 개요를 제공합니다. 자세한 설명은 “Kerberos 자격 증명이 서비스에 대한 액세스를 제공하는 방법” [52]을 참조하십시오.

사용자의 관점에서 Kerberos 서비스는 Kerberos 세션이 시작되면 대개 눈에 띄지 않습니다. `ssh` 또는 `ftp` 등의 명령도 동일하게 작동합니다. Kerberos 세션을 초기화하면 더 이상 로그인하거나 Kerberos 암호를 제공할 필요가 없습니다.

Kerberos 시스템은 티켓이라는 개념에 중점을 둡니다. 티켓은 사용자나 서비스(예: NFS 서비스)를 식별하는 전자 정보 세트입니다. 운전 면허증이 사용자의 신원을 확인해 주고 소지하고 있는 운전 면허를 나타내듯이, 티켓은 사용자와 사용자의 네트워크 액세스 권한을 식별합니다. Kerberos 기반 트랜잭션을 수행하는 경우(예: NFS 마운트된 파일을 요청하는 경우) 티켓에 대한 요청이 투명하게 KDC(키 배포 센터)로 전송됩니다. KDC는 데이터베이스에 액세스하여 사용자의 신원을 인증하고 NFS 서버에 액세스할 수 있는 권한을 부여하는 티켓을 반환합니다. “투명하게”란 명시적으로 티켓을 요청할 필요가 없음을 의미합니다. 요청은 서버에 대한 액세스를 시도할 때 발생합니다. 인증된 클라이언트만 특정 서비스에 대한 티켓을 가져올 수 있으므로 사용 중인 ID로는 다른 클라이언트가 NFS 서버에 액세스할 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 정책에 의해 설정됩니다. 정책은 Kerberos 서비스가 설치되거나 관리될 때 결정됩니다.

참고 - 자격 증명과 티켓이라는 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 키가 추가된 것입니다. 차이점에 대한 자세한 설명은 “Kerberos 자격 증명이 서비스에 대한 액세스를 제공하는 방법” [52]을 참조하십시오.

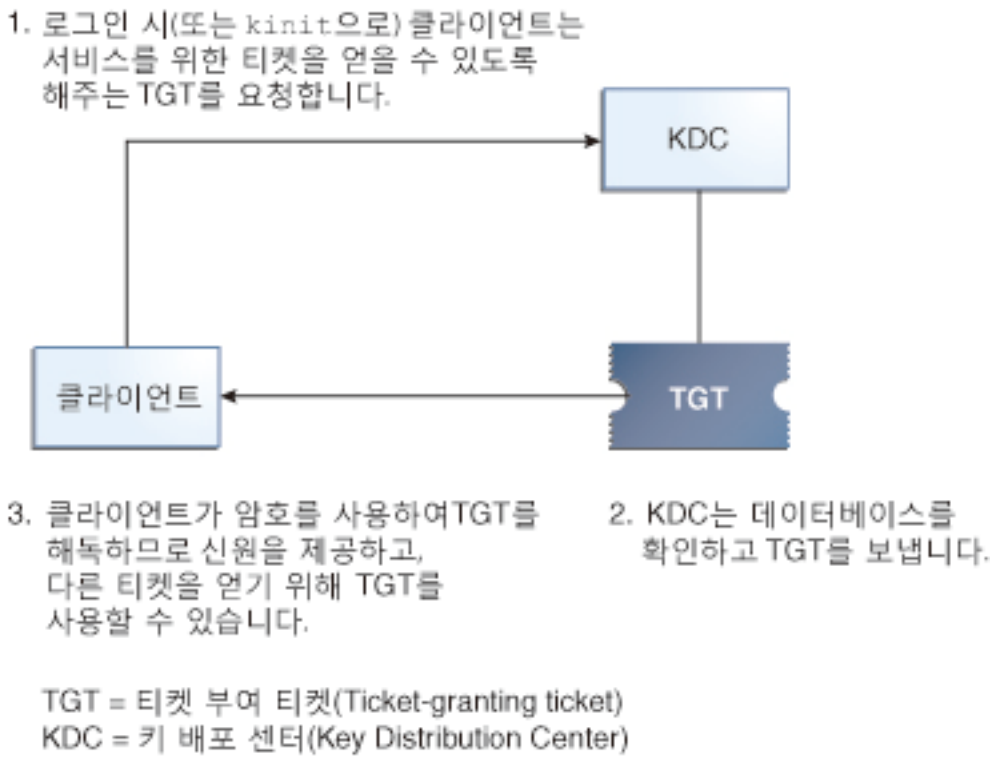
다음 절에서는 Kerberos 인증에 대해 추가로 설명합니다.

초기 인증: TGT(티켓 부여 티켓)

Kerberos 인증은 모든 후속 인증을 가능하게 하는 초기 인증과 후속 인증 자체의 두 단계로 구성됩니다.

다음 그림은 초기 인증이 이루어지는 방식을 보여줍니다.

그림 2-1 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"(티켓)를 얻을 수 있도록 해줍니다. 티켓은 외국이 아니라 원격 시스템 또는 네트워크 서비스에 액세스할 수 있도록 해줍니다. 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.

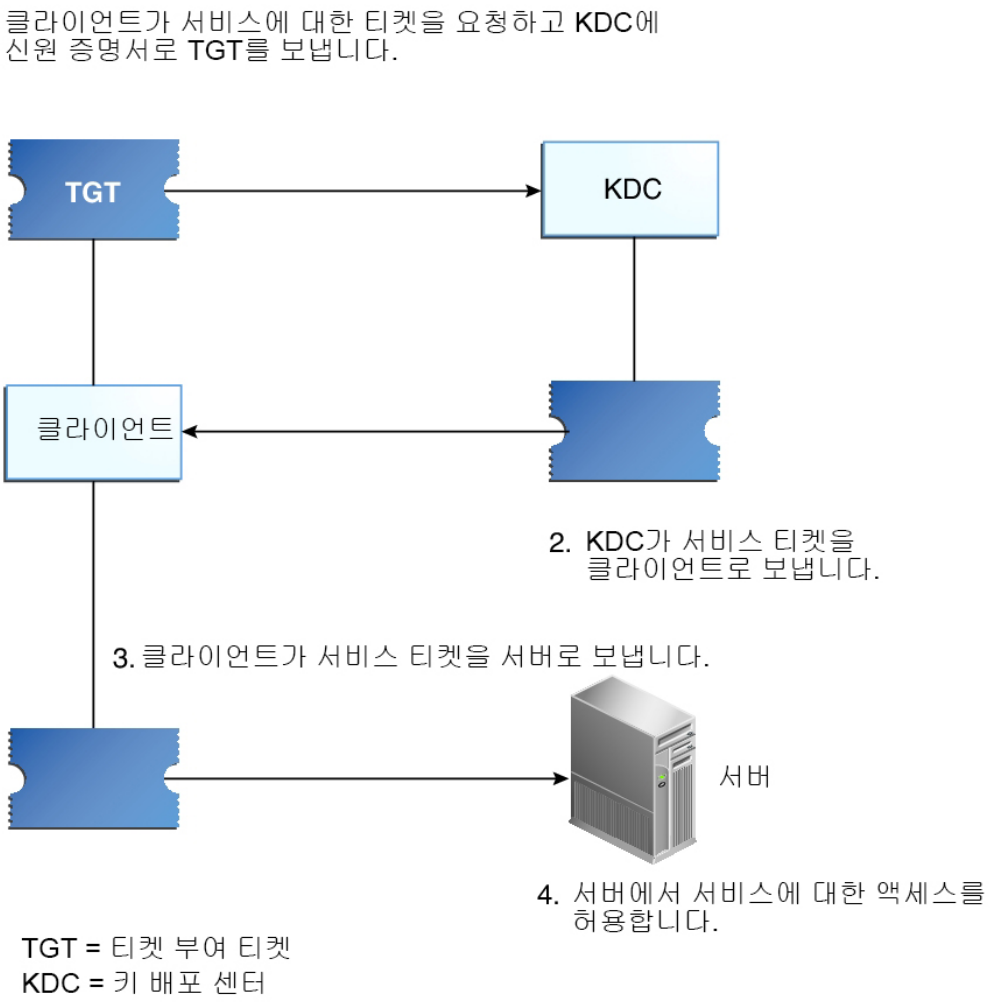
TGT(티켓 부여 티켓)가 네 곳의 스키 리조트에서 3일 동안 사용할 수 있는 스키 패스라고 생각해 볼 수도 있습니다. 방문하려는 리조트의 패스를 보여 주면 패스 유효 기간 동안에는 해당 리조트의 리프트 티켓을 받을 수 있습니다. 리프트 티켓이 있으면 해당 리조트에서 원하는 만큼 스키를 탈 수 있습니다. 다음 날 다른 리조트를 방문한 경우 다시 패스를 보여 주면 새 리조트의 리프트 티켓을 추가로 얻을 수 있습니다. 차이점이라면 Kerberos 기반 명령은 사용자에게 주말용 스키 패스가 있음을 알고 있어 자동으로 리프트 티켓을 얻는다는 점입니다.

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: `nfs` 또는 `ssh`)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 2-2 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.

   `jdoe`라는 사용자가 필요한 `krb5` 인증과 공유된 NFS 파일 시스템에 액세스하려 한다고 가정합시다. `jdoe`는 이미 인증을 받았기 때문에 즉, `jdoe`는 이미 TGT(티켓 부여 티켓)가 있으므로 `jdoe`가 파일에 액세스하려고 하면 NFS 클라이언트 시스템이 자동으로 그리고 투명하게 KDC로부터 NFS 서비스에 대한 티켓을 얻습니다. Kerberos화된 서비스를 사용하기 위해 `jdoe`는 1단계와 같이 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.

여기에 나오는 단계에서는 서버가 KDC와 통신하지 않는다고 가정하지만, 서버는 첫번째 클라이언트와 마찬가지로 자신을 KDC에 등록합니다. 간단히 나타내기 위해 이 부분은 생략되었습니다.

배치 작업에 대한 Kerberos 인증

cron, at, batch 등의 배치 작업은 지연된 실행 프로세스입니다. Kerberos 환경에서는 지연된 실행 프로세스를 포함한 모든 프로세스에 자격 증명이 필요합니다. 하지만 사용자의 자격 증명은 수명이 비교적 짧습니다. 기본적으로 사용자 자격 증명은 8시간 동안 유효하며 1주일 내에서 갱신 가능합니다. 이러한 시간 제한은 중요한 키가 악의적인 사용자에게 노출되지 않도록 하기 위한 것이지만 작업이 실행되는 것을 아무 때나 막을 수 있습니다.

Oracle Solaris에서는 Kerberos 서비스에 액세스하는 배치 작업을 사용자의 장기 키를 노출하지 않고 실행할 수 있습니다. 이러한 문제를 방지하기 위해 Kerberos 서비스, 사용자 이름 및 클라이언트 호스트 이름이 포함된 자격 증명을 세션별 사용자 자격 증명 캐시에 저장하는 방법이 있습니다. 배치 작업을 인증하는 데 PAM 모듈을 사용하는 방법도 있습니다. 호스트가 티켓을 얻을 수 있는 대상 서비스를 LDAP 디렉토리 서버에 중앙 집중식으로 저장할 수도 있습니다.

자세한 내용은 `pam_krb5_keytab`(5) 및 `pam_gss_s4u`(5) 매뉴얼 페이지와 "Kerberos 서비스에 대한 액세스를 위해 지연된 실행 구성" [117]을 참조하십시오.

Kerberos, DNS 및 이름 지정 서비스

Kerberos 서비스는 DNS를 사용하여 호스트 이름을 분석하도록 컴파일됩니다. 호스트 이름을 분석하기 위해 `nsswitch` 서비스는 검사하지 않습니다.

Kerberos 구성 요소

Kerberos는 영역, 네트워크 프로그램, 주체, 서버 및 기타 구성 요소를 포함합니다. 명령 및 모듈 목록은 "Kerberos 유틸리티" [48]를 참조하십시오.

Kerberos 네트워크 프로그램

참고 - 이 Oracle Solaris 릴리스에서는 ssh를 제외한 모든 원격 로그인 명령이 제거되었습니다. 제거된 다음 명령 중 하나를 사용하여 이전 시스템에 연결하려는 경우 그렇게 할 수 있습니다. 제거된 명령이 레거시 스크립트에서 사용되었기 때문에 최신 시스템에서 해당 명령을 사용하려는 경우에는 `svcadm enable login:rlogin`과 같이 해당 명령에 대해 SMF 서비스를 사용으로 설정해야 합니다. `ssh` 명령을 사용하도록 스크립트를 변경하는 방법도 있습니다. 마찬가지로, 이전 시스템에서 제거된 명령을 사용하여 최신 시스템에 연결하려는 경우에는 최신 시스템에서 해당 명령에 대해 이 서비스를 사용으로 설정해야 합니다.

`telnet`과 같은 제거된 명령은 취약한 암호화 키를 요구할 수 있습니다. 이러한 키는 `krb5.conf` 파일에 기본적으로 허용되지 않습니다. 자세한 내용은 "Kerberos에 지원되는 암호화 유형" [60] 및 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오.

자세한 내용은 "Oracle Solaris 11.2에서 시스템 및 연결된 장치의 보안 "의 "시스템 리소스에 대한 액세스 제어"를 참조하십시오. "Kerberos 사용자 명령" [159]에 나오는 매뉴얼 페이지도 참조하십시오.

사용자가 사용할 수 있는 Kerberos 기반("Kerberos화된") 명령은 다음과 같습니다.

- `ftp`
- `rcp`, `rlogin`, `rsh`
- `ssh`, `scp`, `sftp`
- `telnet`

이러한 응용 프로그램은 같은 이름의 Oracle Solaris 응용 프로그램과 같습니다. 그러나 Kerberos 주체를 사용하여 트랜잭션을 인증하도록 확장되었으므로 Kerberos 기반 보안을 제공합니다. 주체에 대한 자세한 내용은 "Kerberos 주체" [45]를 참조하십시오.

이러한 명령은 "Kerberos 사용자 명령" [159]에 자세히 설명되어 있습니다.

Kerberos 주체

Kerberos 서비스의 클라이언트는 주체로 식별됩니다. 주체는 KDC가 티켓을 지정할 수 있는 고유 ID입니다. 주체는 사용자(예: `jdoe`) 또는 서비스(예: `nfs`)일 수 있습니다.

관례상 주체 이름은 기본 요소, 인스턴스, 영역이라는 세 개의 구성 요소로 구분됩니다. 예를 들면 일반적인 Kerberos 주체는 `jdoe/admin@CORP.EXAMPLE.COM`입니다. 위 예에서 각 요소의 역할은 다음과 같습니다.

- `jdoe`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`, `scp`, `ssh` 등을 제공하도록 설정된 서비스 주체임을 의미합니다.

- admin은 인스턴스입니다. 인스턴스는 사용자 주체의 경우 선택적이지만, 서비스 주체의 경우에는 필수입니다. 예를 들어 사용자 jdoe가 가끔 시스템 관리자 역할을 수행하는 경우 jdoe/admin 주체를 사용하여 관리자와 사용자 ID를 구별할 수 있습니다. 마찬가지로, jdoe에게 서로 다른 두 호스트에 대한 계정이 있을 경우 계정에 jdoe/denver.example.com 및 jdoe/boston.example.com과 같이 서로 다른 인스턴스를 사용하는 두 개의 주체 이름을 사용할 수 있습니다. Kerberos 서비스는 jdoe와 jdoe/admin을 완전히 다른 두 주체로 취급합니다.

  서비스 주체의 경우 인스턴스는 정규화된 호스트 이름입니다. bigmachine.corp.example.com은 이러한 인스턴스의 예입니다. 이 예에서 기본 요소/인스턴스는 ftp/bigmachine.corp.example.com 또는 host/bigmachine.corp.example.com일 수 있습니다.
- CORP.EXAMPLE.COM은 Kerberos 영역입니다. 영역은 "Kerberos 영역" [46]에 설명되어 있습니다.

다음은 모두 유효한 주체 이름입니다.

- jdoe
- jdoe/admin
- jdoe/admin@CORP.EXAMPLE.COM
- nfs/host.corp.example.com@CORP.EXAMPLE.COM
- host/corp.example.com@CORP.EXAMPLE.COM

Kerberos 영역

영역은 도메인과 유사한 논리적 그룹으로, 동일한 마스터 *KDC* 아래에 있는 시스템 그룹을 정의합니다. 그림 2-3. "Kerberos 영역"은 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 영역 간 인증은 여러 영역 간의 인증을 지원합니다. 각 영역에는 KDC에 있는 다른 영역에 대한 주체 항목만 있으면 됩니다.

그림 2-3 Kerberos 영역

Kerberos 서버

각 영역에는 주체 데이터베이스의 마스터 복사본을 유지 관리하는 서버가 있어야 합니다. 이 서버를 마스터 *KDC* 서버라고 합니다. 또한 각 영역에는 주체 데이터베이스의 복제 복사본을 포함하는 슬레이브 *KDC* 서버도 한 개 이상 있어야 합니다. 마스터 KDC 서버와 슬레이브 KDC 서버 모두 인증을 설정하는 데 사용되는 티켓을 만듭니다.

영역에는 Kerberos 애플리케이션 서버가 포함될 수도 있습니다. 이 서버는 Kerberos화된 서비스(예: `ftp`, `ssh` 및 NFS)에 대한 액세스를 제공합니다.

다음 그림은 가상 영역에 포함될 수 있는 요소를 보여줍니다.

그림 2-4 일반 Kerberos 영역

Kerberos 유틸리티

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`
- 네트워크 응용 프로그램 - `ftp`, `rcp`, `rlogin`, `rsh`, `scp`, `sftp`, `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_GSS API는 사용 중인 방식과 관계없이 보안 서비스를 제공합니다. RPCSEC_GSS는 GSS-API 계층을 기반으로 합니다. 플러그 가능 GSS_API 기반 보안 방식은 RPCSEC_GSS를 사용하는 응용 프로그램에서 사용할 수 있습니다.

또한 Oracle Solaris의 Kerberos 서비스에는 다음 항목도 포함됩니다.

- PAM용 Kerberos V5 서비스 모듈 - 서비스에 대해 인증, 계정 관리, 세션 관리 및 암호 관리를 제공합니다. 이 모듈은 Kerberos 인증이 사용자에게 투명하게 수행되도록 합니다.
- Kerberos V5 사용자별 PAM 스택 - `/etc/security/pam_policy` 디렉토리에 있는 다양한 시나리오에 대한 PAM 구성 파일을 제공합니다.
- 커널 모듈 - NFS 서비스에서 사용하도록 Kerberos 서비스의 커널 기반 구현을 제공하므로 성능이 크게 향상됩니다.
- Kerberos 관리 GUI(`gkadmin`) - `kadmin` 명령의 대안인 Java™ 기술 기반 GUI에서 주체 및 주체 정책을 관리할 수 있도록 해줍니다.

자세한 내용은 7장. Kerberos 서비스 참조를 참조하십시오.

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.

Kerberos 암호화 유형

암호화 유형으로 암호화 작업을 수행할 때 사용할 암호화 알고리즘과 모드를 식별할 수 있습니다. 지원되는 암호화 유형 목록은 krb5.conf(4) 및 kdb5_util(1M) 매뉴얼 페이지를 참조하십시오.

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 클라이언트가 요청하는 것은 허용하지만 서버에서 암호화 유형을 KDC로 지정하는 것은 허용하지 않습니다.

암호화 유형을 변경하기 전에 다음 문제를 고려해야 합니다.

- KDC는 주체 데이터베이스에서 서버 주체 항목과 연관된 첫번째 키/암호화 유형을 서버에서 지원한다고 가정합니다.
- KDC에서, 주체에 대해 생성된 키가 주체를 인증할 시스템과 호환되는지 확인해야 합니다. 기본적으로 kadmin 명령은 지원되는 모든 암호화 유형에 대한 키를 만듭니다. 주체가 사용되는 시스템에서 암호화 유형의 이 기본 세트를 지원하지 않을 경우 주체를 생성할 때 암호화 유형을 제한해야 합니다. 암호화 유형을 제한하는 두 가지 권장 방법은 kadmin addprinc에 -e 플래그를 사용하는 것과 kdc.conf 파일의 supported_enctypes 매개변수를 해당 암호화 유형 세트로 설정하는 것입니다. supported_enctypes 매개변수는 Kerberos 영역에 있는 대부분의 시스템이 기본 암호화 유형 세트의 일부를 지원하는 경우에 사용합니다. supported_enctypes를 설정하면 특정 영역에 대한 주체를 만들 때 암호화 유형의 기본 세트인 kadmin addprinc이 사용되도록 지정됩니다.
- 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.
- 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 aes256-cts-hmac-sha1-96에서 다른 암호화 유형으로 변경할 수 있습니다. 모든 슬레이브 KDC가 선택한 암호화 유형을 지원하며, 슬레이브 KDC를 구성할 때 kdc.conf 파일에 동일한 master_key_enctype 항목이 있는지 확인하십시오. kdc.conf에서 supported_enctypes가 설정된 경우, master_key_enctype이 supported_enctypes의 암호화 유형 중 하나로 설정되었는지도 확인하십시오. 이러한 문제를 제대로 처리하지 않으면 마스터 KDC가 슬레이브 KDC와 함께 작동하지 않을 수 있습니다.
- 클라이언트의 경우, krb5.conf 파일의 매개변수를 통해 클라이언트가 KDC에서 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_enctypes를 설정하면 클라이언트가 TGT를 사용하여 서버 티켓을 요청할 때(이를 TGS 요청이라고 함) 클라이언트와 KDC 간의 통신을 보

호하는 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 않아야 합니다. 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`로 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 않아야 합니다.
- 서버의 경우, `kdc.conf` 파일의 `permitted_enctypes` 매개변수를 사용하여 서버에서 허용하는 암호화 유형을 제어할 수 있습니다. 또한 keytab 항목을 만들 때 사용되는 암호화 유형을 지정할 수 있습니다. 암호화 유형을 제어하는 데 이 방법을 사용하지 않도록 하십시오. 대신 KDC는 사용할 키 또는 암호화 유형을 결정할 때 서버 응용 프로그램과 통신하지 않으므로 KDC가 사용할 암호화 유형을 결정하도록 하십시오.

FIPS 140 알고리즘 및 Kerberos 암호화 유형

Kerberos가 Oracle Solaris의 FIPS 140 모드에서 실행되도록 구성할 수 있습니다. 영역에 레거시 응용 프로그램 또는 FIPS 140과 호환되지 않는 시스템이 포함된 경우 영역을 FIPS 140 모드로 실행할 수 없습니다.

FIPS 140 모드로 실행할 경우 Kerberos는 FIPS 140 공급자의 소비자가 됩니다. Oracle Solaris의 공급자는 암호화 프레임워크입니다. 암호화 프레임워크에 대해 유일한 FIPS 140 검증 Kerberos 암호화 유형은 `des3-cbc-sha1`입니다. 이 값은 기본값이 아닙니다. 지침은 FIPS 140 모드에서 실행되도록 Kerberos를 구성하는 방법 [70]을 참조하십시오.

참고 - FIPS 140-2 검증 암호화만 사용하도록 요구 사항이 엄격한 경우, Oracle Solaris 11.1 SRU 5.5 릴리스 또는 Oracle Solaris 11.1 SRU 3 릴리스를 실행해야 합니다. Oracle은 이 두 가지 릴리스에서 암호화 프레임워크에 대한 FIPS 140-2 검증을 마쳤습니다. Oracle Solaris 11.2는 이러한 검증을 기초로 제작되었으며 성능, 기능 및 안정성 문제를 해결하는 소프트웨어 향상 기능을 포함합니다. 이러한 향상 기능을 활용하기 위해서는 가능한 모든 경우에 Oracle Solaris 11.2를 FIPS 140-2 모드로 구성해야 합니다.

Kerberos 자격 증명이 서비스에 대한 액세스를 제공하는 방법

ID를 제공하는 티켓 및 일치하는 세션 키를 제공할 경우 원격 서비스를 통해 액세스할 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 `nfs` 서비스에 액세스하는 데는 하나의 자격 증명이 필요합니다. 다른 서버에서 `nfs` 서비스에 액세스하려면 별도의 자격 증명이 필요합니다.

특정 서버에서 특정 서비스에 액세스하려면 사용자가 두 개의 자격 증명을 얻어야 합니다. 첫 번째 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.

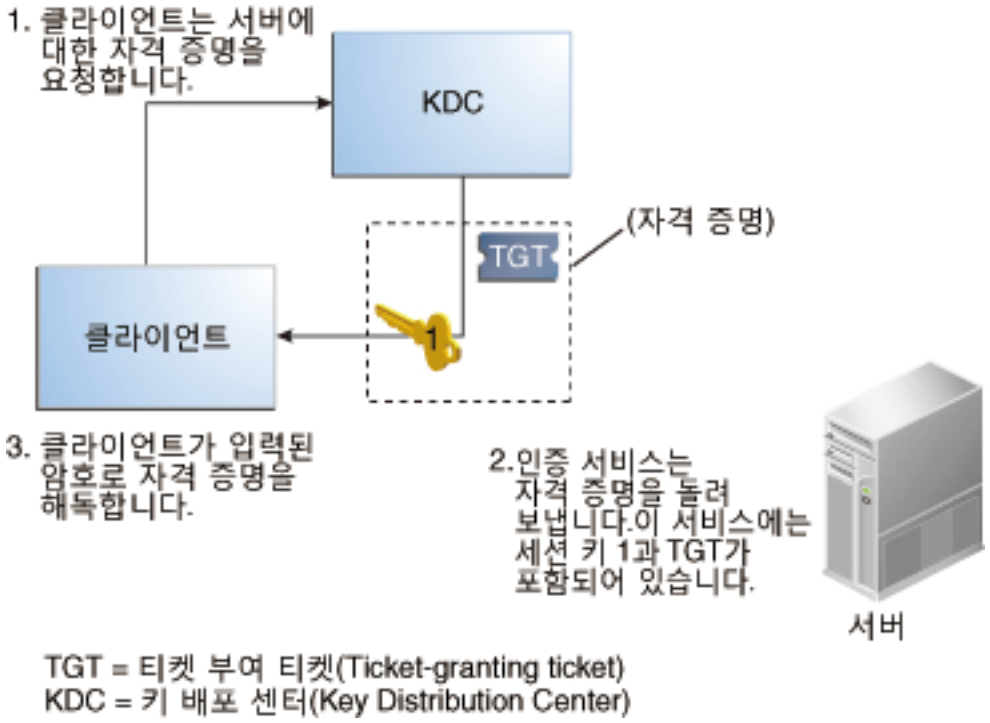
자격 증명을 만들고 저장하는 프로세스는 투명합니다. 자격 증명은 KDC에서 만들어 요청자에게 전송합니다. 수신된 자격 증명은 자격 증명 캐시에 저장됩니다.

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 2-5 TGS(티켓 부여 서비스)에 대한 자격 증명 얻기



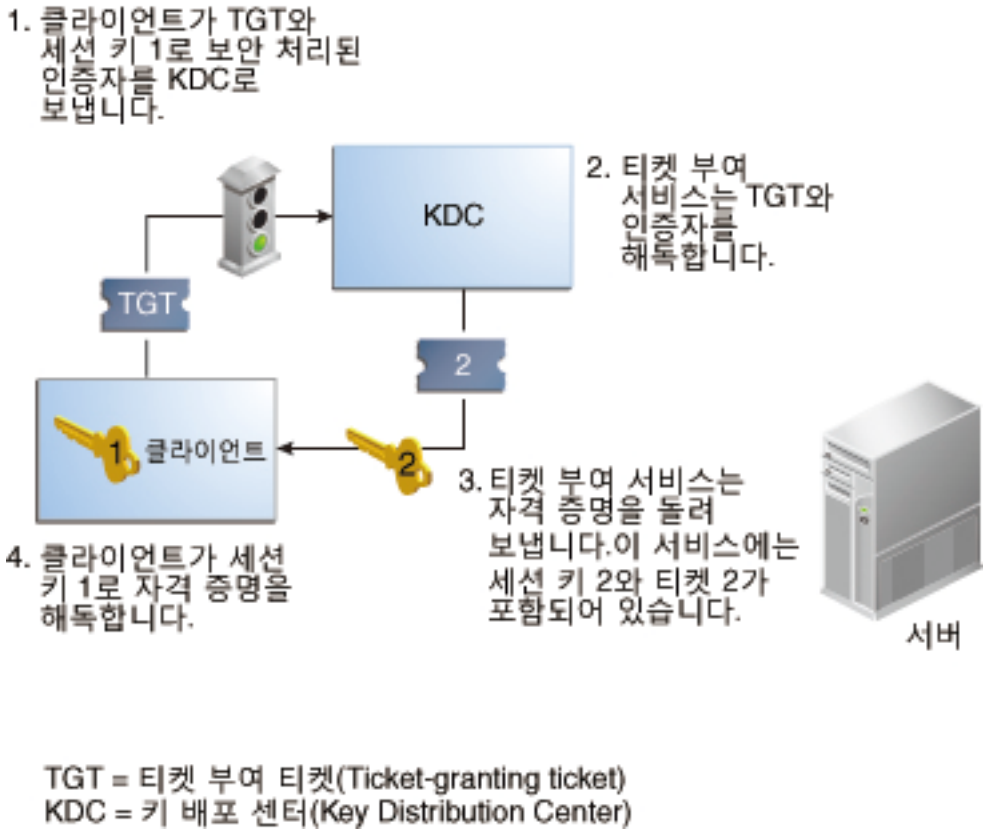
Kerberos화된 서버에 대한 자격 증명 얻기

1. 특정 서버에 대한 액세스를 요청하려면 먼저 클라이언트가 해당 서버에 대한 자격 증명을 인증 서비스로부터 얻어야 합니다. “TGS(티켓 부여 서비스)에 대한 자격 증명 얻기” [52]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 세

션 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 2-6 서버에 대한 자격 증명 얻기



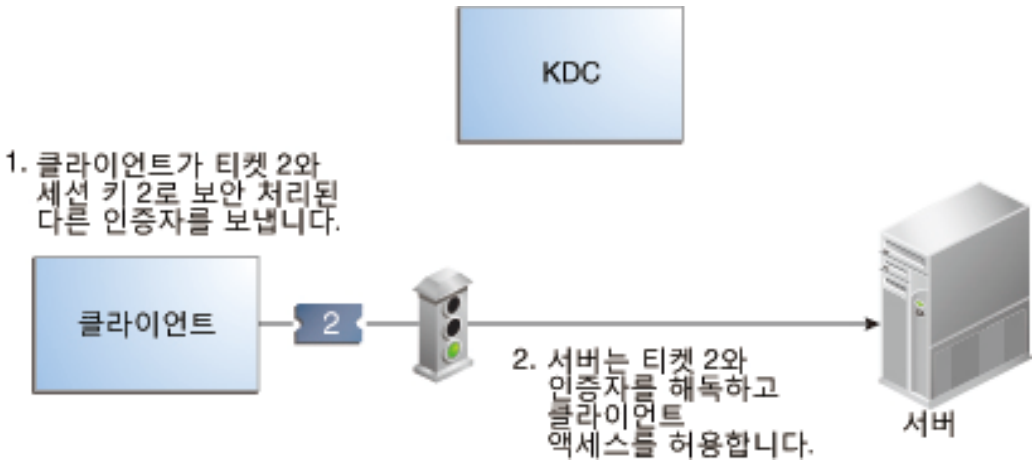
특정 Kerberos 서비스에 대한 액세스 권한 얻기

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. “TGS(티켓 부여 서비스)에 대한 자격 증명 얻기” [52] 및 “Kerberos

화된 서버에 대한 자격 증명 얻기" [53]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 2-7 특정 서비스에 대한 액세스 권한 얻기



Oracle Solaris Kerberos 및 MIT Kerberos 간의 주요 차이점

다음은 Oracle Solaris에는 포함되었지만 MIT Kerberos에는 포함되지 않은 향상된 기능입니다.

- Oracle Solaris 원격 응용 프로그램의 Kerberos 지원
- KDC 데이터베이스의 증분 전파
- 클라이언트 구성 스크립트
- 지역화된 오류 메시지
- Oracle Solaris 감사 레코드 지원
- GSS-API를 통한 Kerberos의 스레드 안전 사용
- 암호화에 대해 암호화 프레임워크 사용

♦ ♦ ♦ 3 장

3 Kerberos 서비스 계획

이 장에서는 관리자가 Kerberos 서비스를 구성하거나 배포하기 전에 해결해야 하는 여러 설치 및 구성 옵션에 대해 설명합니다.

Kerberos 배치 계획

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, Kerberos를 배치하기 전에 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.

Kerberos 영역 이름

영역 이름은 ASCII 문자열로 구성될 수 있습니다. 보통 영역 이름이 대문자인 경우를 제외하고는 DNS 도메인 이름과 같습니다. 이 규칙은 친숙한 이름을 그대로 사용하면서 Kerberos 서비스 관련 문제와 DNS 이름 공간 관련 문제를 구별하는 데 도움이 됩니다. 모든 문자열을 사용할 수 있지만 구성 및 유지 관리에 더 많은 작업이 필요하게 될 수도 있습니다. 표준 인터넷 명명 구조를 준수하는 영역 이름을 사용하십시오.

Kerberos 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 영역에 마스터 서버 하나와 슬레이브 서버 하나 이상, 즉 최소 두 개의 KDC 서버를 만들도록 계획합니다.

관리 도메인과 Kerberos 영역을 조정하는 것이 좋습니다. Kerberos V 영역은 영역이 대응되는 DNS 도메인의 여러 하위 도메인으로 확장될 수 있습니다.

Kerberos 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. 계층 체인의 모든 영역이 제대로 구성된 경우에는 이러한 자동 경로를 통해 관리 부담이 줄어들 수 있습니다. 그러나 여러 레벨의 도메인이 있을 경우 자동 경로에는 너무 많은 트랜잭션이 필요하기 때문에 자동 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 유용합니다. 연결을 사용하는 모든 호스트의 `/etc/krb5/krb5.conf` 파일에 연결을 정의해야 하므로 약간의 추가 작업이 필요합니다. 직접 신뢰 관계는 추이적 관계라고도 합니다. 이에 대한 그림은 “Kerberos 영역” [46]을 참조하십시오. 구성 절차는 “영역 간 인증 구성” [118]을 참조하십시오.

호스트 이름과 Kerberos 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 사용할 경우 정보를 변경할 때마다 모든 Kerberos 클라이언트에서 `krb5.conf` 파일을 편집할 필요가 없으므로 변경 작업이 더 용이합니다.

참고 - 이 설명서의 절차에서는 DNS를 사용하는 것으로 가정합니다.

자세한 내용은 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오.

Oracle Solaris의 Kerberos 클라이언트는 Active Directory 서버와 잘 상호 운영됩니다. 영역과 호스트 간 매핑을 제공하도록 Active Directory 서버를 구성할 수 있습니다.

Kerberos 클라이언트 및 서비스 주체 이름

Oracle Solaris의 Kerberos는 `name-service/switch` 서비스를 사용하지 않습니다. 대신 Kerberos 서비스는 DNS를 사용하여 호스트 이름을 분석합니다. 따라서 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`입니다. 이 설명서의 예에서는 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 제공하는 시스템 호스트 이름의 표준화된 형식이어야 합니다.

다음 예에서는 Kerberos 서비스가 호스트 이름을 정규화하는 방법을 보여 줍니다. 사용자가 `ssh alpha.example.com` 명령(여기서 `alpha.example.com`은 `beta.example.com`에 대한 DNS 호스트 별칭임)을 실행하면 Kerberos 서비스는 `alpha.example.com`을 `beta.example.com`으로 정규화합니다. KDC는 서비스 주체 *host*/`beta.example.com`에 대한 요청으로 티켓을 처리합니다.

주체 이름에 호스트의 FQDN이 포함된 경우, `/etc/resolv.conf` 파일에서 DNS 도메인 이름을 설명하는 문자열과 일치시켜야 합니다. 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 변형일 수 있습니다. 호스트에 대한 주체 이름은 host/boston.example.com@EXAMPLE.COM입니다.

또한 DNS 클라이언트 서비스가 실행 중이지 않은 경우 대부분의 데몬이나 명령이 시작되지 않도록 SMF(서비스 관리 기능)가 구성되었습니다. kdb5_util, kadmind 및 kpropd 데몬과 kprop 명령은 DNS 서비스에 의존하도록 구성되었습니다. Kerberos 서비스 및 SMF에서 제공되는 기능을 완전히 사용하려면 모든 호스트에서 DNS 클라이언트 서비스를 사용으로 설정해야 합니다.

Kerberos 영역 내에서 클럭 동기화

Kerberos 인증 시스템에 참여하는 모든 호스트의 내부 클럭은 지정된 최대 시간 내에서 동기화되어야 합니다. 클럭 불균형이라고 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 "KDC와 Kerberos 클라이언트 간의 클럭 동기화" [121]를 참조하십시오. 클럭을 동기화하는 다른 방법도 있지만 특정 형태의 동기화를 사용해야 합니다.

Kerberos에 지원되는 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 지정하는 연관된 암호화 유형이 지정되어 있습니다. 지원되는 암호화 유형 목록은 krb5.conf(4) 및 kdb5_util(1M) 매뉴얼 페이지를 참조하십시오.

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. 자세한 내용은 "Kerberos 암호화 유형" [50]을 참조하십시오.

des와 같은 취약한 암호화 유형은 기본적으로 허용되지 않습니다. 역방향 호환성이나 상호 운용성을 위해 취약한 암호화 유형을 사용해야 할 경우에는 /etc/krb5/krb5.conf 파일의 libdefaults 섹션에 있는 allow_weak_crypto를 true로 설정하십시오.

KDC 계획

KDC는 특정 포트를 사용합니다. KDC를 사용하려면 더 많은 티켓 부하를 처리할 추가 서버와 서버 간 동기화를 수행할 전파 기술이 필요합니다. 또한 암호화 유형이 중앙에서 관리됩니다. 처음 KDC를 구성할 때 선택할 수 있는 몇 가지 옵션이 있습니다.

KDC 및 관리 서비스용 포트

기본적으로 KDC에는 포트 88 및 포트 750이 사용되며, KDC 관리 데몬에는 포트 749가 사용됩니다. 다른 포트 번호도 사용할 수 있습니다. 그러나 포트 번호를 변경한 경우 모든 클라이언트에서 `/etc/services` 및 `/etc/krb5/krb5.conf` 파일을 변경해야 합니다. 또한 각 KDC에서 `/etc/krb5/kdc.conf` 파일을 업데이트해야 합니다.

슬레이브 KDC 수

슬레이브 KDC는 마스터 KDC와 마찬가지로 클라이언트에 대한 자격 증명을 생성합니다. 또한 마스터를 사용할 수 없는 경우 백업을 제공합니다. 영역당 슬레이브 KDC를 하나 이상은 만들도록 계획합니다.

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 KDC를 너무 많이 추가하지는 마십시오. KDC 데이터베이스를 각 서버로 전파해야 하므로, 설치된 KDC 서버가 많을수록 영역 전체에서 업데이트된 데이터를 가져오는 시간이 오래 걸릴 수 있습니다. 또한 각 슬레이브에는 KDC 데이터베이스의 복사본이 보존되어 있으므로 슬레이브가 많을수록 보안 유출의 위험이 높아집니다.

하나 이상의 슬레이브 KDC가 마스터 KDC로 교체되도록 구성합니다. 이와 같은 방식으로 하나 이상의 슬레이브 KDC를 구성할 경우 얻게 되는 이점은 어떤 이유로 마스터 KDC에서 오류가 발생할 경우 마스터 KDC로 교체할 시스템이 미리 구성되어 있다는 점입니다. 자세한 내용은 “마스터 KDC와 슬레이브 KDC 교체” [122]를 참조하십시오.

Kerberos 데이터베이스 전파

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 “Kerberos 데이터베이스 관리” [127]를 참조하십시오.

증분 전파를 사용하지 않을 경우 가장 먼저 해결해야 할 문제 중 하나는 슬레이브 KDC의 업데이트 간격입니다. 업데이트를 완료하는 데 필요한 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 관리의 복잡도는 더 커집니다. 자세한 내용은 “Kerberos에 대해 병렬 전파 설정” [135]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 “KDC 서버 구성” [69]을 참조하십시오.

LDAP을 사용하여 Kerberos용 데이터베이스 파일을 관리할 수도 있습니다. 자세한 내용은 LDAP 디렉토리 서버를 사용하도록 마스터 KDC를 구성하는 방법 [83]을 참조하십시오. LDAP을 사용하면 Kerberos 데이터베이스와 설정된 기존 디렉토리 서버 간에 조정이 필요한 사이트에서 관리가 간소화됩니다.

Kerberos 클라이언트 계획

Kerberos 클라이언트는 자동 설치하거나, 스크립트에서 설치하거나, 구성 파일을 편집하여 수동으로 구성할 수 있습니다. 보호된 네트워크에 로그인할 수 있도록 PAM 프레임워크가 `pam_ldap` 모듈을 제공합니다. `pam_ldap`(5) 매뉴얼 페이지를 참조하십시오. 또한 클라이언트가 서비스를 요청할 경우 마스터 KDC가 아닌 다른 서버에서 서비스에 대한 권한을 부여할 수 있습니다.

Kerberos 클라이언트 자동 설치 계획

Oracle Solaris AI(자동 설치 프로그램) 기능을 사용하여 Kerberos 클라이언트를 쉽고 빠르게 구성할 수 있습니다. AI 서버 관리자가 Kerberos 구성 프로파일을 만들어 AI 클라이언트에 할당합니다. 또한 AI 서버는 클라이언트 키를 제공합니다. 따라서 Kerberos 클라이언트는 설치하는 즉시 보안 서비스를 호스트할 수 있는, 완전히 프로비전된 Kerberos 시스템입니다. 자동 설치 프로그램을 사용하면 시스템 관리 및 유지 관리 비용을 절감할 수 있습니다.

`kclient` 명령을 사용하여 모든 KDC 유형의 클라이언트에 대한 자동 설치를 만들 수 있습니다.

- Oracle Solaris KDC의 클라이언트가 아닌 클라이언트에 AI를 사용할 수 있습니다. KDC 공급업체 목록은 `kclient`(1M) 매뉴얼 페이지를 참조하십시오.

모든 KDC 유형에 대해 미리 생성된 keytab 전송이 지원됩니다. Oracle Solaris KDC 및 MS AD도 자동 등록을 지원합니다.

- `kclient` 명령을 사용하여 AI에 사용할 Kerberos 구성 프로파일을 만들 수 있습니다. 자세한 내용은 `kclient`(1M) 매뉴얼 페이지 및 "Kerberos 클라이언트 구성 옵션" [63]을 참조하십시오. AI를 통한 Kerberos 클라이언트 구성에 대한 지침은 "Oracle Solaris 11.2 시스템 설치 "의 "AI를 사용한 Kerberos 클라이언트 구성 방법"을 참조하십시오.

Kerberos 클라이언트 구성 옵션

`kclient` 구성 유틸리티를 사용하거나 파일을 수동으로 편집하여 Kerberos 클라이언트를 구성할 수 있습니다. 이 유틸리티는 대화식 모드와 비대화식 모드로 실행할 수 있습니다. 대화식 모드에서는 Kerberos별 매개변수 값을 입력하는 프롬프트가 사용자에게 표시되므로, 사용자가 클라이언트를 구성할 때 변경할 수 있습니다. 비대화식 모드에서는 파일에 매개변수 값을 제공합니다. 또한 비대화식 모드에서는 명령줄 옵션을 추가할 수 있습니다. 대화식 모드와 비대화식 모드 모두 수동 구성에 비해 필요한 단계가 적으므로 더 신속하게 수행할 수 있으며 오류가 발생할 가능성이 낮습니다.

다음 설정이 이미 적용되고 있는 경우에는 Kerberos 클라이언트에 대한 명시적 구성이 필요하지 않습니다.

- KDC에 대해 SRV 레코드를 반환하도록 DNS를 구성합니다.
- 영역 이름이 DNS 도메인 이름과 일치하거나 KDC에서 참조를 지원합니다.
- Kerberos 클라이언트에 KDC 서버의 키와는 다른 키가 필요하지 않습니다.

하지만 다음과 같은 이유로 Kerberos 클라이언트를 명시적으로 구성할 수도 있습니다.

- 제로 구성 프로세스가 직접 구성된 클라이언트보다 더 많은 DNS 조회를 수행하므로 직접 구성보다 덜 효율적입니다.
- 참조가 사용되지 않은 경우 제로 구성 논리는 호스트의 DNS 도메인 이름에 따라 영역을 결정합니다. 이 구성의 경우 약간의 보안 위험이 발생하지만, `dns_lookup_realm`을 사용으로 설정하는 경우보다는 덜 위험합니다.
- `pam_krb5` 모듈이 keytab file(keytab 파일)의 호스트 키 항목에 의존합니다. 이 요구사항은 `krb5.conf` 파일에서 사용 안함으로 설정할 수 있지만, 이는 보안상의 이유로 권장되지 않습니다. 자세한 내용은 "Kerberos 클라이언트 로그인 보안" [63] 및 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오.

클라이언트 구성에 대한 자세한 설명은 "Kerberos 클라이언트 구성" [91]을 참조하십시오.

Kerberos 클라이언트 로그인 보안

로그인 시 클라이언트는 `pam_krb5` 모듈을 사용하여 최신 TGT를 발행한 KDC와 `/etc/krb5/krb5.keytab` 파일에 저장된 클라이언트 호스트 주체를 발행한 KDC가 동일한지 확인합니

다. 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 "TGT(티켓 부여 티켓) 확인을 사용 안함으로 설정" [102]을 참조하십시오.

Kerberos의 신뢰할 수 있는 위임 서비스

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 신뢰할 수 있는지 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.

클라이언트에 대한 KDC 응답의 암호화된 부분에 ok_to_auth_as_delegate 옵션이 설정된 자격 증명 티켓 플래그의 복사본이 포함될 수 있습니다. 클라이언트는 이 설정을 사용하여 (프록시 또는 전송된 TGT를 부여하는 방식으로) 자격 증명을 이 서버로 위임할지 여부를 결정할 수 있습니다. 이 옵션을 설정할 때는 서비스가 위임된 자격 증명을 사용할지 여부뿐만 아니라 보안 및 서비스가 실행되는 서버의 배치를 고려해야 합니다.

Kerberos에 UNIX 이름 및 자격 증명 사용 계획

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. 또한 기본 Kerberos 영역에 유효한 사용자 계정이 없는 UNIX 사용자는 PAM 프레임워크를 사용하여 자동으로 마이그레이션할 수 있습니다.

UNIX 자격 증명과 GSS 자격 증명 간 매핑

기본 매핑 알고리즘은 Kerberos 주체의 기본 이름을 사용하여 UID를 조회합니다. 조회는 기본 영역 또는 /etc/krb5/krb5.conf의 auth_to_local_realm 매개변수에 지정된 허용되는 영역에서 수행됩니다. 예를 들어 사용자 주체 이름 jdoe@EXAMPLE.COM은 암호 테이블을 사용하여 UNIX 사용자 jdoe의 UID에 매핑됩니다. 사용자 주체 이름 jdoe/admin@EXAMPLE.COM은 admin이라는 인스턴스 구성 요소를 포함하고 있으므로 매핑되지 않습니다.

사용자 자격 증명에 대한 기본 매핑으로 충분할 경우, GSS 자격 증명 테이블을 채울 필요가 없습니다. 인스턴스 구성 요소를 포함하는 주체 이름을 매핑하려는 경우와 같이 기본 매핑으로 충분하지 않을 경우, 다른 방법을 사용해야 합니다. 자세한 내용은 다음을 참조하십시오.

- 자격 증명 테이블을 만들고 수정하는 방법 [113]
- 영역 간 자격 증명 매핑 제공 방법 [114]
- “GSS 자격 증명에서 UNIX 자격 증명으로 매핑” [183]

gsscred 테이블

gsscred 테이블은 기본 매핑만으로 충분하지 않은 경우 NFS 서버가 Kerberos 사용자를 식별하려고 할 때 사용됩니다. NFS 서비스는 UNIX UID를 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.

클라이언트 요청이 수신되면 NFS 서비스는 자격 증명 이름을 UNIX UID에 매핑하려고 합니다. 매핑에 실패하면 gsscred 테이블이 검사됩니다.

Kerberos 영역으로 자동 사용자 마이그레이션

기본 Kerberos 영역에 유효한 사용자 계정이 없는 UNIX 사용자는 PAM 프레임워크를 사용하여 자동으로 마이그레이션할 수 있습니다. 즉, pam_krb5_migrate 모듈을 PAM 서비스의 인증 스택에 추가합니다. 이 경우 Kerberos 주체가 없는 사용자가 암호를 사용하여 시스템에 성공적으로 로그인하면 해당 사용자에 대해 Kerberos 주체가 자동으로 생성되도록 서비스가 구성됩니다. 새 주체 암호는 UNIX 암호와 같습니다. pam_krb5_migrate.so 모듈 사용에 대한 지침은 Kerberos 영역에서 사용자의 자동 마이그레이션을 구성하는 방법 [104]을 참조하십시오.

♦ ♦ ♦ 4 장

4 Kerberos 서비스 구성

이 장에서는 KDC 서버, 네트워크 애플리케이션 서버, NFS 서버 및 Kerberos 클라이언트에 대한 구성 절차를 제공합니다. 이러한 절차 중 대부분에는 `root` 액세스 권한이 필요하므로 해당 절차는 시스템 관리자 또는 고급 사용자가 수행해야 합니다. 영역 간 구성 절차 및 기타 KDC 서버 관련 항목도 다룹니다.

이 장에서는 다음 내용을 다룹니다.

Kerberos 서비스 구성

구성 프로세스의 일부 절차는 다른 절차에 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.

참고 - 이 절의 예제에는 Oracle Solaris에 대해 FIPS 140에서 검증되지 않은 기본 암호화 유형이 사용됩니다. FIPS 140 모드로 실행하려면 데이터베이스, 서버 및 클라이언트 커뮤니케이션에 대해 암호화 유형을 `des3-cbc-sha1` 암호화 유형으로 제한해야 합니다. KDC를 만들기 전에 FIPS 140 모드에서 실행되도록 Kerberos를 구성하는 방법 [70]에 따라 파일을 편집합니다.

표 4-1 Kerberos 서비스 구성 작업 맵

작업	설명	지침
1. Kerberos 설치를 계획합니다.	소프트웨어 구성 프로세스를 시작하기 전에 구성 문제를 해결합니다. 사전 계획을 통해 향후 시간과 기타 리소스를 절약할 수 있습니다.	3장. Kerberos 서비스 계획
2. KDC 서버를 구성합니다.	영역에 대한 마스터 KDC 및 슬레이브 KDC 서버와 KDC 데이터베이스를 구성하고 구축합니다.	"KDC 서버 구성" [69]
2a. (선택 사항) FIPS 140 모드에서 실행되도록 Kerberos를 구성합니다.	FIPS 140 검증 알고리즘만 사용으로 설정합니다.	FIPS 140 모드에서 실행되도록 Kerberos를 구성하는 방법 [70]
2b. (옵션) LDAP에서 실행되도록 Kerberos를 구성합니다.	LDAP 디렉토리 서버를 사용하도록 KDC를 구성합니다.	"LDAP 디렉토리 서버에서 KDC 관리" [89]
3. NTP(Network Time Protocol) 소프트웨어를 설치합니다.	네트워크의 모든 시스템에 시간을 제공하는 중앙 클럭을 만듭니다.	"KDC와 Kerberos 클라이언트 간의 클럭 동기화" [121]
4. (옵션) 교체 가능한 KDC 서버를 구성합니다.	마스터 KDC와 슬레이브 KDC 교체 작업을 간편하게 수행할 수 있도록 합니다.	교체 가능한 슬레이브 KDC 구성 방법 [123]
4. (옵션) KDC 서버에서 보안 수준을 향상시킵니다.	KDC 서버에서 보안 위반이 발생하지 않도록 합니다.	"KDC 서버에 대한 액세스 제한" [139]

추가 Kerberos 서비스 구성

필요한 단계가 완료되면 적절할 때 다음 절차를 수행합니다.

표 4-2 추가 Kerberos 서비스 구성 작업 맵

작업	설명	지침
영역 간 인증을 구성합니다.	영역 간 통신을 사용으로 설정합니다.	"영역 간 인증 구성" [118]
Kerberos 애플리케이션 서버를 구성합니다.	서버가 Kerberos 인증을 사용하여 `ftp`와 같은 서비스를 지원할 수 있도록 합니다.	"Kerberos 네트워크 애플리케이션 서버 구성" [109]
지연된 실행 서비스를 구성합니다.	`cron` 호스트가 임의 시간에 작업을 실행할 수 있도록 합니다.	"Kerberos 서비스에 대한 액세스를 위해 지연된 실행 구성" [117]
Kerberos NFS 서버를 구성합니다.	서버가 Kerberos 인증을 필요로 하는 파일 시스템을 공유할 수 있도록 합니다.	"Kerberos NFS 서버 구성" [111]
Kerberos 클라이언트를 구성합니다.	클라이언트가 Kerberos 서비스를 사용할 수 있도록 합니다.	"Kerberos 클라이언트 구성" [91]
인증을 지원할 수 있도록 클럭을 동기화합니다.	클럭 동기화 프로토콜을 구성합니다.	"KDC와 Kerberos 클라이언트 간의 클럭 동기화" [121]

작업	설명	지침
Kerberos 데이터베이스를 관리합니다.	Kerberos 데이터베이스를 유지 관리합니다.	“Kerberos 데이터베이스 관리” [127]
Kerberos 데이터베이스의 stash 파일을 관리합니다.	Kerberos 데이터베이스의 키를 처리합니다.	“Kerberos 데이터베이스의 stash 파일 관리” [136]

KDC 서버 구성

Kerberos 소프트웨어를 설치한 후에는 KDC(키 배포 센터) 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 구성해야 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. 슬레이브 KDC는 마스터 KDC가 응답할 수 없는 경우 중복을 제공합니다.

마스터 KDC 서버, 데이터베이스 및 추가 서버를 다양한 방식으로 구성하고 구축할 수 있습니다.

- 자동 - 스크립트를 사용하려는 경우에 권장
- 대화식 - 대부분의 설치에 적합함
- 수동 - 복잡한 설치에 필요
- 수동(LDAP 사용) - KDC와 함께 LDAP을 사용하는 경우 필요

표 4-3 KDC 서버 구성 작업 맵

작업	설명	지침
KDC 패키지를 설치합니다.	KDC를 만드는 데 필요한 패키지입니다.	KDC 패키지를 설치하는 방법 [70]
(선택 사항) FIPS 140 모드에서 실행되도록 Kerberos를 구성합니다.	FIPS 140 검증 알고리즘만 사용으로 설정합니다.	FIPS 140 모드에서 실행되도록 Kerberos를 구성하는 방법 [70]
스크립트를 사용하여 마스터 KDC를 구성합니다.	초기 구성을 간소화합니다.	`kdcmgr`을 사용하여 마스터 KDC를 구성하는 방법 [71] 예 4-1. “인수 없이 `kdcmgr` 명령 실행”
스크립트를 사용하여 슬레이브 KDC 서버를 구성합니다.	초기 구성을 간소화합니다.	`kdcmgr`을 사용하여 슬레이브 KDC를 구성하는 방법 [73]
마스터 KDC 서버를 수동으로 구성합니다.	초기 설치 중 KDC 구성 파일에 있는 모든 항목을 제어할 수 있도록 합니다.	수동으로 마스터 KDC를 구성하는 방법 [74]
슬레이브 KDC 서버를 수동으로 구성합니다.	초기 설치 중 KDC 구성 파일에 있는 모든 항목을 제어할 수 있도록 합니다.	수동으로 슬레이브 KDC를 구성하는 방법 [79]

작업	설명	지침
LDAP을 사용하도록 마스터 KDC를 수동으로 구성합니다.	LDAP을 사용하도록 마스터 KDC 서버를 구성합니다.	LDAP 디렉토리 서버를 사용하도록 마스터 KDC를 구성하는 방법 [83]
KDC 서버에서 주체 키를 새로 교체합니다.	강력한 암호화 유형을 사용하도록 레거시 KDC 서버의 세션 키를 업데이트합니다.	"마스터 서버에서 TGS(티켓 부여 서비스) 키 바꾸기" [88]

▼ KDC 패키지를 설치하는 방법

Kerberos 클라이언트 소프트웨어는 시스템에 기본적으로 설치됩니다. KDC(키 배포 센터)를 설치하려면 KDC 패키지를 추가해야 합니다.

시작하기 전에 패키지를 시스템에 추가할 수 있으려면 소프트웨어 설치 권한 프로파일이 지정되어 있어야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **KDC 패키지를 설치합니다.**

```
$ pkg install system/security/kerberos-5
```

자세한 내용은 pkg(1) 매뉴얼 페이지를 참조하십시오.

2. **(옵션) Kerberos 서비스를 나열합니다.**

서버 패키지를 추가하면 시스템에 두 가지 Kerberos 서비스가 추가로 제공됩니다. Kerberos 클라이언트 소프트웨어와 마찬가지로 이 두 서비스는 기본적으로 사용 안함으로 설정됩니다. 이러한 서비스를 사용으로 설정하기 전에 Kerberos를 구성합니다.

```
$ svcs krb5
STATE          STIME    FMRI
disabled       Sep_10   svc:/network/security/krb5kdc:default
disabled       Sep_10   svc:/network/security/krb5_prop:default
$ svcs | grep kerb
STATE          STIME    FMRI
disabled       Sep_07   svc:/system/kerberos/install:default
```

▼ FIPS 140 모드에서 실행되도록 Kerberos를 구성하는 방법

시작하기 전에 Kerberos를 FIPS 140 모드로 실행하려면 시스템에서 FIPS 140 모드를 사용으로 설정해야 합니다. "Oracle Solaris 11.2의 암호화 및 인증서 관리 "의 "FIPS 140이 사용으로 설정된 부트 환경 만들기 "를 참조하십시오.

1. **KDC의 암호화 유형을 편집합니다.**

kdc.conf 파일의 [realms] 섹션에서 KDC 데이터베이스에 대한 마스터 키 유형을 설정합니다.

```
# pfedit  /etc/krb5/kdc.conf
...
master_key_type = des3-cbc-sha1-kd
```

2. **같은 파일에서 다른 암호화 유형은 명시적으로 금지합니다.**

   명령을 실행하여 암호화를 설정할 수도 있기 때문에 구성 파일에서 명령에 대한 비FIPS 140 알고리즘 인수 사용을 금지해야 합니다.

   ```
   supported_enctypes = des3-cbc-sha1-kd:normal
   ```

3. **krb5.conf 파일의 [libdefaults] 섹션에서 트랜잭션에 대한 암호화 유형을 편집합니다.**

   이러한 매개변수는 Kerberos 서버, 서비스 및 클라이언트의 암호화 유형을 제한합니다.

   ```
   # pfedit /etc/krb5/krb5.conf
          default_tgs_enctypes = des3-cbc-sha1-kd
          default_tkt_enctypes = des3-cbc-sha1-kd
          permitted_enctypes = des3-cbc-sha1-kd
   ```

4. **같은 파일에서 약한 암호화 유형은 명시적으로 금지합니다.**

   ```
          allow_weak_enctypes = false
   ```

일반 오류 "Kerberos 암호화 유형" [50]을 참조하십시오.

▼ kdcmgr을 사용하여 마스터 KDC를 구성하는 방법

kdcmgr 스크립트는 마스터 및 슬레이브 KDC를 설치하는 데 사용할 수 있는 명령줄 인터페이스를 제공합니다. 마스터 KDC를 설치하는 경우 Kerberos 데이터베이스 암호와 관리자 암호를 각각 만들어야 합니다. 설치를 완료하려면 슬레이브 KDC에서 이 두 암호를 제공해야 합니다. 이러한 암호에 대한 자세한 내용은 kdcmgr(1M) 매뉴얼 페이지를 참조하십시오.

시작하기 전에 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **마스터 KDC를 만듭니다.**

   명령줄에서 kdcmgr 명령을 실행하고 관리자 및 영역을 지정합니다.

   Kerberos 데이터베이스 암호(마스터 키)와 관리 주체의 암호를 묻는 메시지가 표시됩니다. 암호를 묻는 프롬프트가 표시됩니다.

   ```
   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 strong password **/
Re-enter KDC database master key to verify: xxxxxxxx

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 strong password **/
Re-enter password for principal "kws/admin@EXAMPLE.COM": xxxxxxxx
Principal "kws/admin@EXAMPLE.COM" created.

Setting up /etc/krb5/kadm5.acl.

--------------------------------------------------
Setup COMPLETE.

kdc1#
```

참고 - 두 암호를 안전한 위치에 저장하고 보관하십시오.

2. **(옵션) 마스터 KDC의 상태를 표시합니다.**

```
# kdcmgr status
```

3. **NTP 또는 다른 방식을 사용하여 이 시스템의 클럭을 영역의 다른 클럭과 동기화합니다.**
   인증이 성공하려면 모든 클럭이 `krb5.conf` 파일의 `libdefaults` 섹션에 정의된 기본 시간에 속해야 합니다. 자세한 내용은 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오. NTP(Network Time Protocol)에 대한 자세한 내용은 "KDC와 Kerberos 클라이언트 간의 클럭 동기화" [121]를 참조하십시오.

참고 - 마스터 KDC는 NTP 서버로 사용할 수 없습니다. NTP 서버가 없는 경우 NTP 서버를 설치한 후 마스터 KDC로 돌아가 마스터 KDC를 NTP 서버의 클라이언트로 설정하십시오.

예 4-1 인수 없이 `kdcmgr` 명령 실행

이 예에서는 영역 이름 및 관리자 주체를 묻는 프롬프트가 표시되면 관리자가 해당 정보를 제공합니다.

```
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 strong password **/
Re-enter KDC database master key to verify: xxxxxxxx

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 strong password **/
Re-enter password for principal "kws/admin@EXAMPLE.COM": xxxxxxxx
Principal "kws/admin@EXAMPLE.COM" created.

Setting up /etc/krb5/kadm5.acl.

--------------------------------------------------
Setup COMPLETE.

kdc1#
```

▼ kdcmgr을 사용하여 슬레이브 KDC를 구성하는 방법

시작하기 전에 마스터 KDC 서버를 구성합니다.

root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **슬레이브 KDC를 만듭니다.**

   명령줄에서 kdcmgr 명령을 실행하고 관리자, 영역 및 마스터 KDC를 지정합니다.

   kdcmgr을 사용하여 마스터 KDC를 구성하는 방법 [71]에서 만든 두 암호(관리자 주체용 하나와 KDC 데이터베이스용 하나)를 묻는 프롬프트가 표시됩니다.

```
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: xxxxxxxx

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: xxxxxxxx

--------------------------------------------------
Setup COMPLETE.

kdc2#
```

2. **(옵션) KDC의 상태를 표시합니다.**

```
# kdcmgr status
```

3. **NTP 또는 다른 방식을 사용하여 이 시스템의 클럭을 영역의 다른 클럭과 동기화합니다.**

   NTP 서버가 없는 경우 이 시스템을 NTP 서버로 사용할 수 있습니다.

   인증이 성공하려면 모든 클럭이 `krb5.conf` 파일의 `libdefaults` 섹션에 정의된 기본 시간에 속해야 합니다. 자세한 내용은 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오. NTP(Network Time Protocol)에 대한 자세한 내용은 "KDC와 Kerberos 클라이언트 간의 클럭 동기화" [121]를 참조하십시오.

다음 순서 NTP 서버를 설치한 후 마스터 KDC로 돌아가 마스터 KDC를 NTP 서버의 클라이언트로 설정합니다.

▼ 수동으로 마스터 KDC를 구성하는 방법

이 절차에서는 증분 전파가 구성됩니다. 이 절차에서는 다음 구성 매개변수를 사용합니다.

- 영역 이름 = `EXAMPLE.COM`
- DNS 도메인 이름 = `example.com`
- 마스터 KDC = `kdc1.example.com`
- `admin` 주체 = `kws/admin`
- 온라인 도움말 URL = `http://docs.oracle.com/cd/E23824_01/html/821-1456/aadmin-23.html`

참고 - "gkadmin GUI" [142]에 설명된 대로 온라인 도움말 위치를 가리키도록 URL을 조정하십시오.

시작하기 전에 DNS를 사용하도록 호스트를 구성합니다. 이 마스터를 교체 가능한 것으로 설정하려는 경우 구체적인 이름 지정 지침은 "마스터 KDC와 슬레이브 KDC 교체" [122]를 참조하십시오.

`root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **KDC 패키지를 설치합니다.**
   KDC 패키지를 설치하는 방법 [70]의 지침을 따릅니다.

2. **Kerberos 구성 파일(`krb5.conf`)을 편집합니다.**
   이 파일에 대한 설명은 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오.
   이 예에서는 관리자가 `default_realm`, `kdc`, `admin_ server` 및 모든 `domain_realm` 영역에 대한 라인을 변경하고 `help_url` 항목을 편집합니다.

```
kdc1 # pfedit /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://docs.oracle.com/cd/E23824_01/html/821-1456/aadmin-23.html
}
```

참고 - 이전 Kerberos 시스템과 통신해야 하는 경우에는 암호화 유형을 제한해야 할 수도 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 "Kerberos 암호화 유형" [50]을 참조하십시오.

3. **KDC 구성 파일 `kdc.conf`에서 영역을 지정합니다.**

   이 파일에 대한 설명은 `kdc.conf`(4) 매뉴얼 페이지를 참조하십시오.

   이 예에서는 관리자가 영역 이름 정의뿐 아니라 증분 전파 및 로깅 기본값도 변경합니다.

```
kdc1 # pfedit /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
}
```

   참고 - 이전 Kerberos 시스템과 통신해야 하는 경우에는 암호화 유형을 제한해야 할 수도 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 "Kerberos 암호화 유형" [50]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. 자세한 내용은 `kdb5_util`(1M), `kadmind`(1M) 및 `krb5kdc`(1M) 매뉴얼 페이지를 참조하십시오.

```
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 strong password **/
Re-enter KDC database master key to verify: xxxxxxxx
```

   작은 정보 - 이 단계가 실패할 경우 KDC 주체가 해당 FQDN으로 식별되는지 확인하십시오.

```
# getent hosts IP-address-of-KDC
IP-address-of-KDC  kdc       /** This entry does not include FQDN **/
```

   그런 다음 FQDN을 `/etc/hosts` 파일의 첫번째 항목으로 추가합니다. 예를 들면 다음과 같습니다.

```
IP-address-of-KDC  kdc.kdc-principal.example.com kdc
```

5. **Kerberos 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    *
   ```

   위 항목은 EXAMPLE.COM 영역의 kws/admin 주체가 KDC의 주체 및 정책을 수정할 수 있도록 합니다. 기본 주체 항목은 모든 admin 주체와 일치하는 별표(*)입니다. 이 항목은 보안 위험을 야기할 수 있습니다. 모든 admin 주체 및 해당 권한을 명시적으로 나열하도록 파일을 수정합니다. 자세한 내용은 kadm5.acl(4) 매뉴얼 페이지를 참조하십시오.

6. **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 strong password **/
   Re-enter password
   for principal kws/admin@EXAMPLE.COM: xxxxxxxx
   Principal "kws/admin@EXAMPLE.COM" created.
   kadmin.local:
   ```

   자세한 내용은 kadmin(1M) 매뉴얼 페이지를 참조하십시오.

7. **Kerberos 데몬을 시작합니다.**

   ```
   kdc1 # svcadm enable -r network/security/krb5kdc
   kdc1 # svcadm enable -r network/security/kadmin
   ```

8. **kadmin을 시작하고 주체를 더 추가합니다.**

   ```
   kdc1 # /usr/sbin/kadmin -p kws/admin
   Enter password: xxxxxxxx
   kadmin:
   ```

   a. **마스터 KDC host 주체를 만듭니다.**

      호스트 주체는 Kerberos화된 응용 프로그램(예: kprop)이 변경 사항을 슬레이브 KDC에 전파하는 데 사용합니다. 또한 이 주체는 ssh 등의 네트워크 애플리케이션을 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. 자세한 내용은 kclient(1M) 매뉴얼 페이지를 참조하십시오.

```
kadmin: addprinc clntconfig/admin
Enter password for principal clntconfig/admin@EXAMPLE.COM:      /** Type strong  password **/
Re-enter password for principal clntconfig/admin@EXAMPLE.COM: xxxxxxxx
Principal "clntconfig/admin@EXAMPLE.COM" created.
kadmin:
```

참고 - 이 암호를 안전한 위치에 저장하고 보관하십시오.

c. **clntconfig/admin 주체에 권한을 지정합니다.**

kclient 설치 작업을 수행할 수 있는 충분한 권한을 clntconfig 주체에 지정하도록 kadm5.acl 파일을 편집합니다.

```
# pfedit /etc/krb5/kadm5.acl
...
clntconfig/admin@EXAMPLE.COM  acdilm
```

d. **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.
kadmin:
```

e. **kadmin을 종료합니다.**

```
kadmin: quit
```

9. **NTP 또는 다른 동기화 방식을 사용하여 이 시스템의 클럭을 영역에 있는 다른 클럭과 동기화합니다.**

인증이 성공하려면 모든 클럭이 krb5.conf 파일의 libdefaults 섹션에 정의된 기본 시간에 속해야 합니다. 자세한 내용은 krb5.conf(4) 매뉴얼 페이지를 참조하십시오. NTP(Network Time Protocol)에 대한 자세한 내용은 “KDC와 Kerberos 클라이언트 간의 클럭 동기화” [121]를 참조하십시오.

참고 - 마스터 KDC는 NTP 서버로 사용할 수 없습니다. NTP 서버가 없는 경우 NTP 서버를 설치한 후 마스터 KDC로 돌아가 마스터 KDC를 NTP 서버의 클라이언트로 설정하십시오.

10. **슬레이브 KDC를 구성합니다.**

    중복을 제공하려면 슬레이브 KDC를 하나 이상 설치해야 합니다. kdcmgr을 사용하여 슬레이브 KDC를 구성하는 방법 [73] 또는 수동으로 슬레이브 KDC를 구성하는 방법 [79]의 지침을 따릅니다.

▼ 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 구성합니다. 이 슬레이브를 교체 가능한 것으로 설정하려는 경우 마스터 KDC와 슬레이브 KDC 교체 방법 [123]의 지침을 따릅니다.

KDC 서버에서 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **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: xxxxxxxx
kadmin:
```

   자세한 내용은 kadmin(1M) 매뉴얼 페이지를 참조하십시오.

   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 증분 전파에 사용할 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
```

2. **마스터 KDC에서 Kerberos 구성 파일(krb5.conf)을 편집합니다.**

슬레이브마다 항목을 하나씩 추가해야 합니다. 이 파일에 대한 설명은 krb5.conf(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # pfedit /etc/krb5/krb5.conf
.
.
[realms]
EXAMPLE.COM = {
kdc = kdc1.example.com
kdc = kdc2.example.com
admin_server = kdc1.example.com
}
```

3. **마스터 KDC에서 kadm5.acl에 kiprop 항목을 추가합니다.**

이 항목은 마스터 KDC가 kdc2 서버에 대한 증분 전파 요청을 수신할 수 있도록 합니다.

```
kdc1 # pfedit /etc/krb5/kadm5.acl
*/admin@EXAMPLE.COM *
kiprop/kdc2.example.com@EXAMPLE.COM p
```

4. **kadm5.acl 파일의 새 항목이 사용되도록 마스터 KDC에서 kdmin 서비스를 다시 시작합니다.**

```
kdc1 # svcadm restart network/security/kadmin
```

5. **모든 슬레이브 KDC에서 마스터 KDC 서버의 KDC 관리 파일을 복사합니다.**

각 슬레이브 KDC에 마스터 KDC에 대한 최신 정보가 있어야 합니다. sftp 또는 유사한 전송 방식을 사용하여 마스터 KDC의 다음 파일을 복사할 수 있습니다.

- /etc/krb5/krb5.conf
- /etc/krb5/kdc.conf

6. **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 # pfedit /etc/krb5/kpropd.acl
host/kdc1.example.com@EXAMPLE.COM
host/kdc2.example.com@EXAMPLE.COM
```

7. **모든 슬레이브 KDC에서 Kerberos 액세스 제어 목록 파일 `kadm5.acl`이 채워져 있지 않은지 확인합니다.**

   수정되지 않은 `kadm5.acl` 파일은 다음 예와 같이 표시될 수 있습니다.

   ```
   kdc2 # pfedit /etc/krb5/kadm5.acl
   */admin@___default_realm___ *
   ```

   파일에 `kiprop` 항목이 있을 경우 제거합니다.

8. **새 슬레이브에서 `kdc.conf` 파일에 슬레이브의 폴링 간격을 정의합니다.**

   `sunw_dbprop_master_ulogsize` 항목을 슬레이브의 폴링 간격을 정의하는 항목으로 바꿉니다. 다음 항목은 폴링 시간을 2분으로 설정합니다.

   ```
   kdc1 # pfedit /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
   }
   ```

9. **새 슬레이브에서 `kadmin` 명령을 시작합니다.**

   마스터 KDC를 구성할 때 만든 `admin` 주체 이름 중 하나로 로그인합니다.

   ```
   kdc2 # /usr/sbin/kadmin -p kws/admin
   Enter password: xxxxxxxx
   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. 자세한 내용은 kprop(1M) 매뉴얼 페이지를 참조하십시오.

      ```
      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.
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.
kadmin:
```

c. **`kadmin`을 종료합니다.**

```
kadmin: quit
```

10. **새 슬레이브에서 Kerberos 전파 데몬을 시작합니다.**

```
kdc2 # svcadm enable network/security/krb5_prop
```

11. **새 슬레이브에서 `kdb5_util` 명령을 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: xxxxxxxx
```

자세한 내용은 kdb5_util(1M) 매뉴얼 페이지를 참조하십시오.

12. **NTP 또는 다른 방식을 사용하여 이 시스템의 클럭을 영역의 다른 클럭과 동기화합니다.**

인증이 성공하려면 모든 클럭이 `krb5.conf` 파일의 `libdefaults` 섹션에 정의된 기본 시간에 속해야 합니다. 자세한 내용은 krb5.conf(4) 매뉴얼 페이지를 참조하십시오. NTP(Network Time Protocol)에 대한 자세한 내용은 "KDC와 Kerberos 클라이언트 간의 클럭 동기화" [121]를 참조하십시오.

13. **새 슬레이브에서 KDC 데몬을 시작합니다.**

```
kdc2 # svcadm enable network/security/krb5kdc
```

다음 순서 NTP 서버를 설치한 후 마스터 KDC로 돌아가 마스터 KDC를 NTP 서버의 클라이언트로 설정합니다.

▼ LDAP 디렉토리 서버를 사용하도록 마스터 KDC를 구성하는 방법

이 절차에서는 다음 구성 매개변수를 사용합니다.

- 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://docs.oracle.com/cd/E23824_01/html/821-1456/aadmin-23.html`

참고 - “`gkadmin` GUI” [142]에 설명된 대로 온라인 도움말 위치를 가리키도록 URL을 조정하십시오.

시작하기 전에 DNS를 사용하도록 호스트를 구성합니다. 성능을 향상시키려면 KDC 및 LDAP 디렉토리 서비스를 동일한 서버에 설치하십시오. 또한 디렉토리 서버가 실행 중이어야 합니다. 다음 절차는 Oracle Directory Server Enterprise Edition을 사용하는 서버에서 작동합니다. 자세한 내용은 Oracle Identity Management - Documentation을 참조하십시오.

KDC 서버에서 `root` 역할을 맡아야 합니다. 자세한 내용은 “Oracle Solaris 11.2의 사용자 및 프로세스 보안 ”의 “지정된 관리 권한 사용”을 참조하십시오.

1. **SSL을 사용하여 디렉토리 서버에 연결하도록 마스터 KDC를 구성합니다.**
   다음 단계에서는 디렉토리 서버의 자체 서명된 인증서를 사용하도록 KDC를 구성합니다.

   a. **디렉토리 서버에서 자체 서명된 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.cert.der \
   label=defaultCert dir=/var/ldap
   ```

   자세한 내용은 `pktool(1)` 매뉴얼 페이지를 참조하십시오.

c. **마스터 KDC에서 SSL이 작동 중인지 테스트합니다.**

이 예에서는 `cn=directory manager` 항목에 관리 권한이 있는 것으로 간주합니다.

```
master# /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.

2. **필요한 경우 LDAP 디렉토리를 채웁니다.**

3. **LDAP의 기존 스키마에 Kerberos 스키마를 추가합니다.**

```
# ldapmodify -h dsserver.example.com -D "cn=directory manager" \
-f /usr/share/lib/ldif/kerberos.ldif
```

4. **LDAP 디렉토리에 Kerberos 컨테이너를 만듭니다.**

`krb5.conf` 파일에 다음 항목을 추가합니다.

a. **데이터베이스 유형을 정의합니다.**

`realms` 절에 `database_module`을 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 및 키에 대한 stash 파일을 만듭니다. 명령의 옵션에 대한 자세한 내용은 kdb5_ldap_util(1M) 매뉴얼 페이지를 참조하십시오.

```
# kdb5_ldap_util -D "cn=directory manager" create
   -P master-key -r EXAMPLE.COM -s
```

5. **KDC 바인드 고유 이름(DN) 암호를 숨깁니다.**

이러한 암호는 KDC가 디렉토리 서버에 바인딩될 때 사용됩니다. 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"
```

6. 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: xxxxxxxx

dn: cn=kadmin service,ou=profile,dc=example,dc=com
cn: kadmin service
sn: kadmin service
objectclass: top
objectclass: person
userpassword: xxxxxxxx
```

   b. LDAP 디렉토리에 역할 항목을 만듭니다.

```
# ldapmodify -a -h dsserver.example.com -D "cn=directory manager" -f kdc_roles.ldif
```

7. **kadmin** 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
```

8. Kerberos 구성 파일(**krb5.conf**)을 편집합니다.

영역 및 서버를 지정해야 합니다. 이 파일에 대한 설명은 krb5.conf(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # pfedit /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://docs.oracle.com/cd/E23824_01/html/821-1456/aadmin-23.html
}
```

참고 - "gkadmin GUI" [142]에 설명된 대로 온라인 도움말 위치를 가리키도록 URL을 조정하십시오.

이 예에서는 default_realm, kdc, admin_server 및 모든 domain_realm 항목에 대한 라인이 변경되었습니다. 온라인 도움말 URL도 변경되었습니다.

참고 - 이전 Kerberos 시스템과 통신해야 하는 경우에는 암호화 유형을 제한해야 할 수도 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 "Kerberos 암호화 유형" [50]을 참조하십시오.

9. **KDC 구성 파일(kdc.conf)을 편집합니다.**

   영역을 지정해야 합니다. 이 파일에 대한 설명은 kdc.conf(4) 매뉴얼 페이지를 참조하십시오.

   이 예에서는 관리자가 영역 이름 정의뿐 아니라 증분 전파 및 로깅 기본값도 변경합니다.

```
kdc1 # pfedit /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
}
```

참고 - 이전 Kerberos 시스템과 통신해야 하는 경우에는 암호화 유형을 제한해야 할 수도 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 "Kerberos 암호화 유형" [50]을 참조하십시오.

10. **Kerberos 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    *
    ```

    위 항목은 `EXAMPLE.COM` 영역의 `kws/admin` 주체가 KDC의 주체 및 정책을 수정할 수 있도록 합니다. 기본 주체 항목은 모든 `admin` 주체와 일치하는 별표(`*`)입니다. 이 항목은 보안 위험을 야기할 수 있습니다. 모든 `admin` 주체 및 해당 권한을 명시적으로 나열하도록 파일을 수정합니다. 자세한 내용은 `kadm5.acl`(4) 매뉴얼 페이지를 참조하십시오.

11. **`kadmin.local` 명령을 시작하고 `admin` 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 만들어야 합니다. 이 예에서는 `kws/admin` 주체를 만듭니다. "`kws`" 대신 적절한 주체 이름으로 대체할 수 있습니다.

       ```
       kadmin.local: addprinc kws/admin
       Enter password for principal kws/admin@EXAMPLE.COM:     /** Type strong password **/
       Re-enter password for principal kws/admin@EXAMPLE.COM: xxxxxxxx
       Principal "kws/admin@EXAMPLE.COM" created.
       kadmin.local:
       ```

    b. **`kadmin.local`을 종료합니다.**

       ```
       kadmin.local: quit
       ```

12. **(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
```

13. **7단계 through 9단계 in 수동으로 마스터 KDC를 구성하는 방법 [74]를 수행하여 LDAP에서 Kerberos 구성을 완료합니다.**

14. **슬레이브 KDC를 구성합니다.**

중복을 제공하려면 슬레이브 KDC를 하나 이상 설치해야 합니다. 수행 방법은 수동으로 슬레이브 KDC를 구성하는 방법 [79]을 참조하십시오.

마스터 서버에서 TGS(티켓 부여 서비스) 키 바꾸기

참고 - 모든 세션 키에 더 강력한 새 암호화 유형을 사용하려는 경우 키를 바꾸십시오.

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 TGS 주체가 모든 세션 키에 대해 더 강력한 암호화를 생성할 수 있도록 주체의 DES 키를 바꿔야 합니다.

원격에서 또는 마스터 서버에서 직접 키를 바꿀 수 있습니다. `changepw` 권한이 지정된 `admin` 주체여야 합니다.

- 모든 Kerberos 시스템에서 TGS 서비스 주체 키를 바꾸려면 `kadmin` 명령을 사용합니다.

```
kdc1 % /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
kadmin: cpw -randkey krbtgt/EXAMPLE.COM@EXAMPLE.COM
Enter TGS key: xxxxxxxx
Enter new TGS key:     /** Type strong password **/
Re-enter TGS key to verify: xxxxxxxx
```

cpw는 change_password 명령의 별칭입니다. -randkey 옵션은 새 암호를 묻는 프롬프트를 표시합니다.

- root로 KDC 마스터에 로그온하면 kadmin.local 명령을 사용할 수 있습니다. 새 데이터베이스 암호를 묻는 프롬프트가 표시됩니다.

```
kdc1 # kadmin.local -q 'cpw -randkey krbtgt/EXAMPLE.COM@EXAMPLE.COM'
```

참고 - 이 암호를 안전한 위치에 저장하고 보관하십시오.

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 4-4 LDAP을 사용하도록 KDC 서버 구성 작업 맵

작업	설명	지침
마스터 KDC를 구성합니다.	수동 프로세스와 KDC용 LDAP을 사용하여 영역에 대한 마스터 KDC 서버와 데이터베이스를 구성하고 구축합니다.	LDAP 디렉토리 서버를 사용하도록 마스터 KDC를 구성하는 방법 [83]
Kerberos 주체 속성과 비Kerberos 객체 클래스 유형을 함께 사용합니다.	Kerberos 레코드와 함께 저장된 정보가 다른 LDAP 데이터베이스와 공유될 수 있도록 합니다.	비Kerberos 객체 클래스 유형에서 Kerberos 주체 속성을 함께 사용하는 방법 [89]
영역을 삭제합니다.	영역에 연결된 데이터를 모두 제거합니다.	LDAP 디렉토리 서버에서 영역 삭제 방법 [90]

▼ 비Kerberos 객체 클래스 유형에서 Kerberos 주체 속성을 함께 사용하는 방법

이 절차에서는 krbprincipalaux, krbTicketPolicyAux 및 krbPrincipalName 속성이 people 객체 클래스에 연결됩니다.

이 절차에서는 다음 구성 매개변수를 사용합니다.

- 디렉토리 서버 = dsserver.example.com

- 사용자 주체 = `mre@EXAMPLE.COM`

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **people 객체 클래스에서 각 항목을 준비합니다.**
   디렉토리 서버에서 각 항목에 대해 이 단계를 반복합니다.

```
cat << EOF | ldapmodify -h dsserver.example.com -D "cn=directory manager"
dn: uid=mre,ou=people,dc=example,dc=com
changetype: modify
objectClass: krbprincipalaux
objectClass: krbTicketPolicyAux
krbPrincipalName: mre@EXAMPLE.COM
EOF
```

2. **영역 컨테이너에 하위 트리 속성을 추가합니다.**
   이 예에서는 `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
```

3. **(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
```

4. **(옵션) KDC에 주체 속성을 추가합니다.**

```
# kadmin.local -q 'addprinc mre'
```

▼ LDAP 디렉토리 서버에서 영역 삭제 방법

이 절차는 다른 LDAP 디렉토리 서버가 영역을 처리하도록 구성된 경우 사용할 수 있습니다.

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

- **영역을 삭제합니다.**

```
# kdb5_ldap_util -D "cn=directory manager" destroy
```

Kerberos 클라이언트 구성

Kerberos 클라이언트에는 네트워크의 모든 호스트(KDC 서버가 아니면서 동시에 Kerberos 서비스를 사용해야 하는 호스트)가 포함됩니다. 이 절에서는 root 인증을 사용하여 NFS 파일 시스템을 마운트하는 자세한 내용과 함께 Kerberos 클라이언트 설치 절차를 제공합니다.

클라이언트 구성 옵션은 서버 옵션과 비슷하며 AI(자동 설치 프로그램)가 추가되었습니다.

- AI - 여러 Kerberos 클라이언트를 쉽고 빠르게 설치하려는 경우 권장
- 자동 - 스크립트를 사용하려는 경우에 권장
- 대화식 - 대부분의 설치에 적합함
- 수동 - 복잡한 설치에 필요

다음 작업 맵에서는 이 절에서 다루는 작업에 대해 설명합니다.

표 4-5 Kerberos 클라이언트 구성 작업 맵

작업	설명	지침
AI(자동 설치 프로그램)를 사용하여 클라이언트를 설치합니다.	시스템 설치 중 Kerberos 클라이언트를 구성하려는 경우 적합합니다.	"Oracle Solaris 11.2 시스템 설치 "의 "AI를 사용한 Kerberos 클라이언트 구성 방법"
유사한 Kerberos 클라이언트에 사용할 수 있는 설치 프로파일을 만듭니다.	재사용 가능한 클라이언트 설치 프로파일을 만듭니다.	Kerberos 클라이언트 설치 프로파일을 만드는 방법 [91]
스크립트를 사용하여 클라이언트를 설치합니다.	각 클라이언트에 대한 설치 매개변수가 동일한 경우 적합합니다.	자동으로 Kerberos 클라이언트를 구성하는 방법 [92]
프롬프트에 응답하여 클라이언트를 설치합니다.	설치 매개변수 중 일부만 변경해야 할 경우 적합합니다.	대화식으로 Kerberos 클라이언트를 구성하는 방법 [94]
수동으로 클라이언트를 설치합니다.	각 클라이언트 설치에 고유 설치 매개변수가 필요한 경우 적합합니다.	수동으로 Kerberos 클라이언트를 구성하는 방법 [97]
Kerberos 클라이언트가 Active Directory 서버에 참여하도록 설정합니다.	자동으로 Active Directory 서버의 Kerberos 클라이언트를 설치합니다.	Kerberos 클라이언트가 Active Directory 서버에 참여하도록 설정하는 방법 [96]
클라이언트 TGT(티켓 부여 티켓)를 발행한 KDC의 확인을 사용 안함으로 설정합니다.	Kerberos 클라이언트의 로컬 keytab 파일에 저장된 호스트 주체가 없는 경우 KDC 확인을 간소화합니다.	"TGT(티켓 부여 티켓) 확인을 사용 안함으로 설정" [102]
클라이언트가 root NFS 파일 시스템에 액세스할 수 있도록 합니다.	클라이언트가 root 액세스 권한으로 NFS 파일 시스템을 마운트할 수 있도록 합니다. 또한 cron 작업이 실행될 수 있도록 클라이언트가 NFS 파일 시스템에 액세스할 수 있도록 합니다.	Kerberos로 보호된 NFS 파일 시스템에 root 사용자로 액세스하는 방법 [103]

▼ Kerberos 클라이언트 설치 프로파일을 만드는 방법

이 절차에서는 Kerberos 클라이언트를 설치할 때 사용할 수 있는 kclient 프로파일을 만듭니다. 프로파일을 사용하면 입력 오류 발생 가능성이 줄어듭니다. 또한 이 프로파일을 사용하면 대화식 프로세스에 비해 사용자 개입이 줄어듭니다.

참고 - 처음부터 완전히 구성된 Kerberos 클라이언트로 부트되는 시스템을 만들려면 “Configuring Security” in “Installing Oracle Solaris 11.2 Systems ”을 참조하십시오.

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 “Oracle Solaris 11.2의 사용자 및 프로세스 보안 ”의 “지정된 관리 권한 사용”을 참조하십시오.

1. **`kclient` 설치 프로파일을 만듭니다.**

   다음은 샘플 `kclient` 프로파일입니다.

```
client# pfedit kcprofile
REALM EXAMPLE.COM
KDC kdc1.example.com
ADMIN clntconfig
FILEPATH /net/denver.example.com/export/install/krb5.conf
NFS 1
DNSLOOKUP none
```

2. **파일을 보호하고 다른 클라이언트가 사용할 수 있도록 저장합니다.**

```
client# cp kcprofile /net/denver.example.com/export/install
denver# chown root kcprofile; chmod 644 kcprofile
```

▼ 자동으로 Kerberos 클라이언트를 구성하는 방법

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 “Oracle Solaris 11.2의 사용자 및 프로세스 보안 ”의 “지정된 관리 권한 사용”을 참조하십시오.

1. **kclient 프로파일을 만듭니다.**

   Kerberos 클라이언트 설치 프로파일을 만드는 방법 [91]의 설치 프로파일을 사용합니다.

2. **프로파일 인수 없이 `kclient` 명령을 실행합니다.**

   프로세스를 완료하려면 `clntconfig` 주체에 대한 암호를 제공해야 합니다. 이 암호는 “KDC 서버 구성” [69]에서 마스터 KDC를 구성할 때 이미 만들었습니다. 자세한 내용은 `kclient`(1M) 매뉴얼 페이지를 참조하십시오.

```
client# /usr/sbin/kclient -p /net/denver.example.com/export/install/kcprofile

Starting client setup
-------------------------------------------------

kdc1.example.com

Setting up /etc/krb5/krb5.conf.
```

```
Obtaining TGT for clntconfig/admin ...
Password for clntconfig/admin@EXAMPLE.COM: xxxxxxxx

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 4-2 설치 프로파일을 사용하여 Kerberos 클라이언트 구성

다음 예에서는 kcprofile 클라이언트 프로파일과 두 개의 명령줄 대체를 사용하여 클라이언트를 구성합니다.

```
# /usr/sbin/kclient -p /net/denver.example.com/export/install/kcprofile \
-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:  xxxxxxxx

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. 클라이언트가 Active Directory 서버에 참여하도록 설정하려는 경우 Kerberos 클라이언트가 Active Directory 서버에 참여하도록 설정하는 방법 [96]으로 이동합니다.

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **인수 없이 `kclient` 명령을 실행합니다.**

   ```
   client# /usr/sbin/kclient
   ```

   다음 정보를 묻는 프롬프트가 표시됩니다.

   - Kerberos 영역 이름
   - KDC 마스터 호스트 이름
   - KDC 슬레이브 호스트 이름
   - 로컬 영역에 매핑할 도메인
   - Kerberos 인증에 사용할 PAM 서비스 이름 및 옵션

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

2. **KDC 서버가 Oracle Solaris 릴리스를 실행하고 있지 않은 경우 `y`로 응답하고 KDC를 실행하는 서버의 유형을 정의합니다.**

   사용할 수 있는 서버 목록은 `kclient`(1M) 매뉴얼 페이지의 `-T` 옵션을 참조하십시오.

3. **Kerberos 조회에 DNS가 사용되어야 하는 경우에는 `y`로 응답하고 사용할 DNS 조회 옵션을 지정합니다.**

   유효한 옵션은 `dns_lookup_kdc`, `dns_lookup_realm` 및 `dns_fallback`입니다. 이러한 값에 대한 자세한 내용은 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오.

4. **Kerberos 영역의 이름 및 마스터 KDC 호스트 이름을 정의합니다.**

   이 정보는 `/etc/krb5/krb5.conf` 구성 파일에 추가됩니다.

5. **영역에 슬레이브 KDC가 있는 경우 `y`로 응답하고 슬레이브 KDC 호스트 이름을 제공합니다.**

   이 정보는 클라이언트의 구성 파일에 추가 KDC 항목을 만드는 데 사용됩니다.

6. **서비스 또는 호스트 키가 필요한 경우 `y`로 응답합니다.**

   일반적으로 클라이언트 시스템이 Kerberos화된 서비스를 호스트하는 경우 서비스 또는 호스트 키가 필요하지 않습니다.

7. **클라이언트가 클러스터의 구성원인 경우 `y`로 응답하고 클러스터의 논리적 이름을 제공합니다.**

논리적 호스트 이름은 서비스 키를 만들 때 사용되며, 클러스터에서 Kerberos 서비스를 호스트하는 경우 필요합니다.

8. **현재 영역에 매핑할 도메인 또는 호스트를 식별합니다.**
   이 매핑은 다른 도메인이 클라이언트의 기본 영역에 속할 수 있도록 합니다.

9. **클라이언트가 Kerberos화된 NFS를 사용할지 여부를 지정합니다.**
   클라이언트가 Kerberos를 사용하는 NFS 서비스를 호스트할 경우 NFS 서비스 키를 만들어야 합니다.

10. **새 PAM 정책을 만들어야 하는지 여부를 나타냅니다.**
    인증에 Kerberos를 사용할 PAM 서비스를 설정하려면 서비스 이름과 Kerberos 인증의 사용 방식을 나타내는 플래그를 제공합니다. 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.

    Kerberos용으로 제공된 PAM 서비스에 대한 자세한 내용은 `/etc/security/pam_policy`를 참조하십시오.

11. **마스터 `/etc/krb5/krb5.conf` 파일을 복사해야 하는지 여부를 지정합니다.**
    이 옵션을 설정하면 `kclient`에 대한 인수가 충분하지 않을 경우 특정 구성 정보를 사용할 수 있습니다.

예 4-3 `kclient` 스크립트 샘플 실행

```
...
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 kerberos lookups ? [y/n]: n
No action performed.
Enter the Kerberos realm: EXAMPLE.COM
Specify the KDC host name 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: corphdqtrs.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.
```

▼ Kerberos 클라이언트가 Active Directory 서버에 참여하도록 설정하는 방법

이 절차에서는 설치 프로파일 없이 `kclient` 명령을 사용합니다.

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **(옵션) 클라이언트에 대한 DNS 리소스 레코드 생성을 사용으로 설정합니다.**

   ```
   client# sharectl set -p ddns_enable=true smb
   ```

2. **`kclient` 명령을 실행합니다.**

   다음 출력에서는 클라이언트가 `EXAMPLE.COM`이라는 AD 도메인에 참여하도록 설정하기 위해 `kclient` 명령을 실행했을 때의 샘플 출력을 보여 줍니다.

   `-T` 옵션은 KDC 서버 유형(이 경우 Microsoft Active Directory(AD))을 선택합니다. 기본적으로 AD 서버의 관리자 주체에 대한 암호를 제공해야 합니다.

   ```
   client# /usr/sbin/kclient -T ms_ad
   Starting client setup
   -------------------------------------------------

   Attempting to join 'CLIENT' to the 'EXAMPLE.COM' domain.
   Password for Administrator@EXAMPLE.COM: xxxxxxxx
   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.
#
```

자세한 내용은 kclient(1M) 매뉴얼 페이지를 참조하십시오.

▼ 수동으로 Kerberos 클라이언트를 구성하는 방법

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
- NFS 서버 = denver.example.com
- 클라이언트 = client.example.com
- admin 주체 = kws/admin
- 사용자 주체 = mre
- 온라인 도움말 URL = http://docs.oracle.com/cd/E23824_01/html/821-1456/aadmin-23.html

시작하기 전에 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **Kerberos 구성 파일(krb5.conf)을 편집합니다.**

   Kerberos 구성 파일에서 영역 이름 및 서버 이름을 변경합니다. gkadmin에 대한 도움말 파일의 경로도 지정할 수 있습니다.

```
kdc1 # pfedit /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://www.example.com/doclib/OSMKA/aadmin-23.html
```

참고 - 이전 Kerberos 시스템과 통신해야 하는 경우에는 암호화 유형을 제한해야 할 수도 있습니다. 암호화 유형 제한과 관련된 문제에 대한 설명은 "Kerberos 암호화 유형" [50]을 참조하십시오.

2. **(옵션) KDC를 찾는 데 사용되는 프로세스를 변경합니다.**

   기본적으로 Kerberos 영역과 KDC 간의 매핑은 다음 순서로 확인됩니다.

   - `krb5.conf`의 `realms` 절에 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)를 참조하십시오. 항상 참조가 먼저 시도됩니다.

3. **(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.

4. **NTP 또는 다른 클럭 동기화 방식을 사용하여 클라이언트의 클럭을 마스터 KDC의 클럭과 동기화합니다.**

   인증이 성공하려면 모든 클럭이 `krb5.conf` 파일의 `clockskew` 관계에 정의된 최대 차이 범위 내에서 KDC 서버의 시간과 동기화되어야 합니다. 자세한 내용은 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오. NTP(Network Time Protocol)에 대한 자세한 내용은 "KDC와 Kerberos 클라이언트 간의 클럭 동기화" [121]를 참조하십시오.

5. **Kerberos 주체를 만듭니다.**

```
denver # /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
```

```
kadmin:
```

자세한 내용은 kadmin(1M) 매뉴얼 페이지를 참조하십시오.

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 strong password **/
Re-enter password for principal mre@EXAMPLE.COM: xxxxxxxx
kadmin:
```

b. **(옵션) root 주체를 만들고 서버의 keytab 파일에 주체를 추가합니다.**

참고 - 클라이언트가 NFS 마운트된 원격 파일 시스템에 대한 root 액세스를 필요로 하지 않을 경우 이 단계를 건너 뛸 수 있습니다.

---

cron 작업을 root로 실행하는 등 비대화식 root 액세스가 필요한 경우 이 단계를 수행합니다.

root 주체는 두 개의 구성 요소로 이루어진 주체여야 합니다. 영역 차원의 root 주체가 생성되지 않도록 두번째 구성 요소는 Kerberos 클라이언트 시스템의 호스트 이름이어야 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.
kadmin:
```

c. **host 주체를 만들고 서버의 keytab 파일에 주체를 추가합니다.**

host 주체는 원격 액세스 서비스가 인증을 제공하는 데 사용합니다. 이 주체는 keytab 파일에 아직 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.
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.
kadmin:
```

e. **kadmin을 종료합니다.**

```
kadmin: quit
```

6. **(옵션) NFS에서 Kerberos를 사용으로 설정합니다.**

a. **/etc/nfssec.conf 파일에서 Kerberos 보안 모드를 사용으로 설정합니다.**

/etc/nfssec.conf 파일에서 Kerberos 보안 모드를 주석 처리하는 "#"을 제거합니다.

```
# pfedit /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.

```
# svcadm enable network/dns/client:default
```

c. **gss 서비스를 다시 시작합니다.**

```
# svcadm restart network/rpc/gss
```

7. **(옵션) 클라이언트가 TGT를 자동으로 갱신하도록 하거나 사용자에게 Kerberos 티켓 만료를 경고하려면 /etc/krb5/warn.conf 파일에서 항목을 만듭니다.**

자세한 내용은 warn.conf(4) 매뉴얼 페이지 및 “모든 TGT(티켓 부여 티켓) 자동 갱신” [108]을 참조하십시오.

예 4-4 다중 마스터 KDC에서 작동하도록 Oracle Solaris 클라이언트 구성

Microsoft Active Directory(AD) Kerberos 서비스는 여러 마스터 서버에서 실행되는 KDC를 제공합니다. Oracle Solaris 클라이언트가 정보를 업데이트하도록 하려면 /etc/krb5/krb5.conf 파일의 admin_server 또는 kpasswd_server 선언에서 모든 서버를 나열해야 합니다. 이 예에서는 kdc1과 kdc2 사이에 공유되는 KDC 정보를 클라이언트가 업데이트하도록 허용하는 방법을 보여줍니다.

```
[realms]
EXAMPLE.COM = {
kdc = kdc1.example.com
kdc = kdc2.example.com
admin_server = kdc1.example.com
admin_server = kdc2.example.com
}
```

예 4-5 비Oracle Solaris KDC에 대해 Kerberos 클라이언트 구성

/etc/krb5/krb5.conf 파일의 realms 섹션에 라인을 추가하여 비Oracle Solaris KDC에서 작동하도록 Kerberos 클라이언트를 설정할 수 있습니다. 이 행은 클라이언트가 Kerberos 암호 변경 서버와 통신하는 동안 사용되는 프로토콜을 변경합니다. 다음 발췌 부분은 이 라인의 형식을 보여 줍니다.

```
[realms]
EXAMPLE.COM = {
kdc = kdc1.example.com
kdc = kdc2.example.com
admin_server = kdc1.example.com
kpasswd_protocol = SET_CHANGE
}
```

예 4-6 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 4-7 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 0 464 kdc1.example.com
_kpasswd._udp.EXAMPLE.COM           IN      SRV 0 0 464 kdc1.example.com
```

TGT(티켓 부여 티켓) 확인을 사용 안함으로 설정

기본적으로 Kerberos는 로컬 `/etc/krb5/krb5.keytab` 파일에 저장된 호스트 주체의 KDC가 TGT(티켓 부여 티켓)를 발급한 KDC인지 검사합니다. 이 검사를 `verify_ap_req_nofail`이라고 하며 DNS 스푸핑 공격을 방지합니다.

하지만 호스트 주체를 사용할 수 없는 클라이언트 구성에 대해서는 이 검사를 사용 안함으로 설정해야 합니다. 다음은 이 검사를 사용 안함으로 설정해야 하는 구성입니다.

- 클라이언트 IP 주소가 동적으로 지정됩니다(예: DHCP 클라이언트).
- 클라이언트는 서비스를 호스트하도록 구성되지 않으므로 만들어진 host 주체가 없습니다.
- 호스트 키는 클라이언트에 저장되지 않습니다.

TGT 검사를 사용 안함으로 설정하려면 `krb5.conf` 파일의 `verify_ap_req_nofail` 옵션을 `false`로 설정합니다. `verify_ap_req_nofail` 옵션은 `krb5.conf` 파일의 `[libdefaults]` 또는 `[realms]` 섹션에 입력할 수 있습니다. `[libdefaults]` 섹션의 설정은 모든 영역에 사용됩니다.

```
client # pfedit /etc/krb5/krb5.conf
[libdefaults]
default_realm = EXAMPLE.COM
verify_ap_req_nofail = false
...
```

`[realms]` 섹션에 옵션을 입력할 경우 정의된 영역에만 설정이 적용됩니다. 이 옵션에 대한 자세한 내용은 `krb5.conf`(4) 매뉴얼 페이지를 참조하십시오.

▼ 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.example.com 등의 옵션과 공유되는 경우 이 절차를 수행하십시오.

1. **kadmin 명령을 실행합니다.**

```
denver # /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
kadmin:
```

2. **NFS 클라이언트에 대한 root 주체를 만듭니다.**

   이 주체는 Kerberos 인증이 필요한 NFS 마운트된 파일 시스템에 대해 root와 동등한 액세스를 제공하는 데 사용됩니다. root 주체는 두 개의 구성 요소로 이루어진 주체여야 합니다. 영역 차원의 root 주체가 생성되지 않도록 두번째 구성 요소는 Kerberos 클라이언트 시스템의 호스트 이름이어야 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:
```

3. **서버의 keytab 파일에 root 주체를 추가합니다.**

   클라이언트가 NFS 서비스를 사용하여 마운트된 파일 시스템에 대해 root 액세스 권한을 갖도록 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.
kadmin:
```

4. **kadmin을 종료합니다.**

```
kadmin: quit
```

▼ Kerberos 영역에서 사용자의 자동 마이그레이션을 구성하는 방법

Kerberos 주체가 없는 사용자는 PAM을 사용하여 기존 Kerberos 영역으로 자동 마이그레이션될 수 있습니다. Kerberos 영역에서의 재인증 및 UNIX 자격 증명 인증을 처리할 수 있도록 마이그레이션 서버 및 마스터 서버에서 시스템별 PAM 구성 파일을 사용자 정의합니다.

PAM에 대한 자세한 내용은 1장. 플러그 가능한 인증 모듈 사용 및 pam.conf(4) 매뉴얼 페이지를 참조하십시오.

이 절차에서는 자동 마이그레이션을 사용하도록 로그인 서비스 이름을 구성합니다. 이 예에서는 다음 구성 매개변수를 사용합니다.

- 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 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **server1에 대한 호스트 서비스 주체가 있는지 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
   ...
   ```

   명령의 옵션에 대한 자세한 내용은 klist(1) 매뉴얼 페이지를 참조하십시오.

2. **server1이 나열되지 않으면 EXAMPLE.COM 영역의 Kerberos 클라이언트로 구성합니다.**

   단계는 "Kerberos 클라이언트 구성" [91]의 예를 참조하십시오.

3. **server1에 대한 PAM 정책을 수정합니다.**

   자세한 내용은 "사용자별 PAM 정책 지정" [19]을 참조하십시오.

   a. **server1에서 사용 중인 Kerberos 정책을 확인합니다.**

      ```
      % grep PAM_POLICY /etc/security/policy.conf
      # PAM_POLICY specifies the system-wide PAM policy (see pam_user_policy(5))
      ...
      PAM_POLICY=krb5_first
      ```

b. **해당 PAM 정책 파일을 복사한 다음 복사한 새 정책 파일을 수정하여 각 인증 스택에 `pam_krb5_migrate.so.1` 모듈을 추가합니다.**

```
server1 # cd /etc/security/pam_policy/; cp krb5_first krb5_firstmigrate
server1 # pfedit /etc/security/pam_policy/krb5_firstmigrate.
# login service (explicit because of pam_dial_auth)
#
login auth requisite    pam_authtok_get.so.1
...
login auth required  pam_unix_auth.so.1
login    auth optional  pam_krb5_migrate.so.1
#
# rlogin service (explicit because of pam_rhost_auth)
#
rlogin auth sufficient  pam_rhosts_auth.so.1
...
rlogin auth required    pam_unix_auth.so.1
rlogin   auth optional        pam_krb5_migrate.so.1
#
# Kerberized rlogin service
#
krlogin auth required  pam_unix_cred.so.1
krlogin auth required  pam_krb5.so.1
krlogin auth optional  pam_krb5_migrate.so.1
#
# rsh service (explicit because of pam_rhost_auth)
#
rsh auth sufficient  pam_rhosts_auth.so.1
rsh auth required    pam_unix_cred.so.1
rsh auth optional    pam_krb5_migrate.so.1
#
# Kerberized rsh service
#
krsh auth required  pam_unix_cred.so.1
krsh auth required  pam_krb5.so.1
krsh auth optional  pam_krb5_migrate.so.1
#
# Kerberized telnet service
#
ktelnet auth required  pam_unix_cred.so.1
ktelnet auth required  pam_krb5.so.1
ktelnet auth optional  pam_krb5_migrate.so.1
#
# PPP service (explicit because of pam_dial_auth)
#
ppp auth requisite    pam_authtok_get.so.1
...
ppp auth required  pam_unix_auth.so.1
ppp auth optional  pam_krb5_migrate.so.1
#
# GDM Autologin (explicit because of pam_allow).  These need to be
# here as there is no mechanism for packages to amend pam.conf as
```

```
# they are installed.
#
gdm-autologin auth  required    pam_unix_cred.so.1
gdm-autologin auth  sufficient  pam_allow.so.1
gdm-autologin auth optional    pam_krb5_migrate.so.1
#
# Default definitions for Authentication management
# Used when service name is not explicitly mentioned for authentication
#
OTHER auth requisite    pam_authtok_get.so.1
...
OTHER auth required  pam_unix_auth.so.1
OTHER auth optional  pam_krb5_migrate.so.1
#
# passwd command (explicit because of a different authentication module)
#
passwd auth required  pam_passwd_auth.so.1
#
# cron service (explicit because of non-usage of pam_roles.so.1)
#
cron account required pam_unix_account.so.1
#
# cups service (explicit because of non-usage of pam_roles.so.1)
#
cups account required pam_unix_account.so.1
#
# GDM Autologin (explicit because of pam_allow) This needs to be here
# as there is no mechanism for packages to amend pam.conf as they are
# installed.
#modified
gdm-autologin account  sufficient  pam_allow.so.1
#
.
.
.
```

c. **(옵션) 강제로 암호를 즉시 변경합니다.**

새로 만든 Kerberos 계정의 경우 `pam_krb5_migrate` 항목에 `expire_pw` 옵션을 추가하여 암호 만료 시간을 현재 시간으로 설정합니다. 자세한 내용은 pam_krb5_migrate(5) 매뉴얼 페이지를 참조하십시오.

service-name `auth optional     pam_krb5_migrate.so.1` **`expire_pw`**

d. **이 구성 파일에서, Kerberos 암호가 만료된 경우 액세스를 차단하도록 `OTHER` 계정 스택을 수정합니다.**

```
# Definition for Account management
# Used when service name is not explicitly mentioned for account management
# Re-ordered pam_krb5 causes password expiration in Kerberos to block access
#
OTHER account requisite   pam_roles.so.1
```

```
OTHER account required pam_krb5.so.1
OTHER account required pam_unix_account.so.1
OTHER account required pam_tsol_account.so.1
# OTHER account required pam_krb5.so.1
#
.
.
.
```

e. **수정된 구성 파일을 사용하도록 `policy.conf` 파일의 `PAM_POLICY` 항목을 변경합니다.**

```
server1 # pfedit /etc/security/policy.conf
...
# PAM_POLICY=krb5_first
PAM_POLICY=krb5_firstmigrate
```

자세한 내용은 `policy.conf` 파일을 읽어 보십시오.

4. **마스터 KDC에서 `kadm5.acl` 액세스 제어 파일을 업데이트합니다.**

다음 항목은 `root` 사용자를 제외한 모든 사용자에 대한 `host/server1.example.com` 서비스 주체에 마이그레이션 및 조회 권한을 부여합니다. `U` 권한을 사용하여 마이그레이션하지 않아야 할 사용자를 나열합니다. 이러한 항목은 전체 또는 `ui` 허가 항목 앞에 와야 합니다. 자세한 내용은 `kadm5.acl`(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # pfedit /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 `kadmind` 데몬이 `k5migrate` PAM 서비스를 사용할 수 있도록 설정합니다.**

`k5migrate` 서비스 파일이 `/etc/pam.d` 디렉토리에 없는 경우 서비스 파일을 디렉토리에 추가합니다. 자세한 내용은 `pam.d`(4) 매뉴얼 페이지를 참조하십시오.

이렇게 수정하면 마이그레이션이 필요한 계정에 대해 UNIX 사용자 암호 검증이 사용으로 설정됩니다.

```
kdc1 # pfedit /etc/pam.d/k5migrate
...
# Permits validation of migrated UNIX accounts
auth    required        pam_unix_auth.so.1
account required        pam_unix_account.so.1
```

참고 - `k5migrate`는 PAM 서비스의 이름입니다. 이 파일은 이름이 `k5migrate`여야 합니다.

6. **생산 환경에서 구성을 사용하기 전에 테스트합니다.**

- **일반 사용자로 수정한 각 PAM 서비스를 테스트합니다.**
- **`root`로 수정한 각 PAM 서비스를 테스트합니다.**

- **암호 변경을 강제로 적용한 다음 수정된 PAM 서비스를 테스트합니다.**

모든 TGT(티켓 부여 티켓) 자동 갱신

관리상 편의를 위해 TGT(티켓 부여 티켓) 만료에 대한 티켓 갱신 및 경고 메시지를 구성할 수 있습니다. 관리자는 모든 사용자에 대한 경고를 설정할 수 있고 사용자는 각자의 경고를 사용자 정의할 수 있습니다. 자세한 내용은 `warn.conf`(4) 및 `ktkt_warnd`(1M) 매뉴얼 페이지를 참조하십시오.

참고 - `ktkt_warn` 서비스가 기본적으로 사용 안함으로 설정되어 있습니다. 기존 Kerberos 클라이언트에서 서비스를 사용으로 설정하려면 `svcadm enable ktkt_warn` 명령을 실행하십시오.

예 4-8 모든 사용자에 대한 TGT 만료 메시지 구성

이 예에서는 TGT의 갱신 및 메시지 시스템을 구성하는 몇 가지 방법을 보여줍니다.

```
# pfedit /etc/krb5/warn.conf
##
## renew the TGT 30 minutes before expiration and send message to users terminal
##
mre@EXAMPLE.COM renew:log terminal 30m
##
## send a warning message to a specific email address 20 minutes before TGT expiration
##
mre@EXAMPLE.COM mail 20m mre@example2.com
##
# renew the TGT 20 minutes before expiration and send an email message on failure
##
bricker@EXAMPLE.COM renew:log-failure mail 20m -
##
## catch-all: any principal not matched above will get an email warning
* mail 20m -
```

메시지를 구성한 후 새 클라이언트에서 `kclient` 명령을 실행합니다.

```
client# /usr/sbin/kclient -p /net/denver.example.com/export/install/kcprofile
```

기존 클라이언트에서 서비스를 사용으로 설정합니다.

```
# svcadm enable network/security/ktkt_warn
```

예 4-9 사용자에 대한 TGT 만료 메시지 구성

각 사용자는 개별 `warnd` 구성 파일(`/var/user/`*`$USER`*`/krb-warn.conf`)을 구성할 수 있습니다. 이 파일은 관리자 파일을 읽지 못하도록 합니다.

```
% pfedit /var/user/mre/krb-warn.conf
mre@EXAMPLE.COM renew:log mail 25m &
```

TGT는 만료되기 25분 전에 갱신되며 갱신이 기록되고 이때 Kerberos 사용자 mre에게는 메일이 전송됩니다.

Kerberos 네트워크 애플리케이션 서버 구성

네트워크 애플리케이션 서버는 ftp, rcp, rlogin, rsh, ssh 및 telnet 네트워크 애플리케이션 중 하나 이상을 사용하여 액세스를 제공하는 호스트입니다. 몇 단계만으로 서버에서 이러한 응용 프로그램의 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 마스터 KDC를 구성합니다. "KDC와 Kerberos 클라이언트 간의 클럭 동기화" [121]에 설명된 대로 클럭을 동기화합니다. 프로세스를 완전히 테스트하려면 여러 클라이언트가 필요합니다.

애플리케이션 서버에서 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **새 서버에 대한 호스트 주체가 있는지 확인합니다.**
   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 반환하면 작업이 완료된 것입니다. 명령이 주체를 반환하지 않을 경우 다음 단계에 따라 새 주체를 만듭니다.

2. **마스터 KDC를 구성할 때 만든 admin 주체 이름 중 하나를 사용하여 서버에 로그인합니다.**

```
boston # /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
```

```
kadmin:
```

3. **서버의 `host` 주체를 만듭니다.**

```
kadmin: addprinc -randkey host/boston.example.com
Principal "host/boston.example.com" created.
kadmin:
```

`host` 주체의 용도는 다음과 같습니다.

- 원격 명령(예: `rsh` 및 `ssh`) 사용 시 트래픽 인증
- `pam_krb5`가 `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. 이 기능은 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.

4. **서버의 keytab 파일에 서버의 `host` 주체를 추가합니다.**

`kadmin` 명령이 실행되고 있지 않을 경우 `/usr/sbin/kadmin -p` *kws*`/admin`과 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.
kadmin:
```

5. **`kadmin`을 종료합니다.**

```
kadmin: quit
```

▼ FTP 실행 시 Kerberos를 통한 일반 보안 서비스 사용 방법

Kerberos 네트워크 애플리케이션은 인증, 무결성 및 프라이버시를 위해 일반 보안 서비스(GSS)를 사용할 수 있습니다. 다음 단계에서는 ProFTPD에 대해 GSS 서비스를 사용으로 설정하는 방법을 보여 줍니다.

시작하기 전에 FTP 서버에서 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **FTP 서버에 대한 주체를 추가하고 FTP 서버의 keytab 파일을 만듭니다.**
   변경 사항이 이전에 적용된 경우 이러한 단계가 필요하지 않을 수도 있습니다.

   a. **`kadmin` 명령을 시작합니다.**

   ```
   ftpserver1 # /usr/sbin/kadmin -p kws/admin
   Enter password: xxxxxxxx
   kadmin:
   ```

   b. **FTP 서버에 대한 `ftp` 서비스 주체를 추가합니다.**

   ```
   kadmin: addprinc -randkey ftp/ftpserver1.example.com
   ```

   c. **`ftp` 서비스 주체를 새 keytab 파일에 추가합니다.**
   새 keytab 파일을 만들면 서버의 keytab 파일에 있는 모든 정보를 노출하지 않고도 ftp 서비스에 이 정보를 제공할 수 있습니다.

   ```
   kadmin: ktadd -k /etc/krb5/ftp.keytab ftp/ftpserver1.example.com
   ```

   자세한 내용은 `ktadd` command in the `kadmin`(1M) 매뉴얼 페이지를 참조하십시오.

2. **새 keytab 파일의 소유권을 변경합니다.**

   ```
   ftpserver1 # chown ftp:ftp /etc/krb5/ftp.keytab
   ```

3. **FTP 서버에 대해 GSS를 사용으로 설정합니다.**
   `/etc/proftpd.conf` 파일을 다음과 같이 변경합니다.

   ```
   # pfedit /etc/proftpd.conf
   LoadModule      mod_gss.c

   GSSEngine       on
   GSSKeytab       /etc/krb5/ftp.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 내

용은 “UNIX 자격 증명과 GSS 자격 증명 간 매핑” [64]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 4-6 Kerberos NFS 서버 구성 작업 맵

작업	설명	지침
Kerberos NFS 서버를 구성합니다.	서버가 Kerberos 인증을 필요로 하는 파일 시스템을 공유할 수 있도록 합니다.	Kerberos NFS 서버 구성 방법 [112]
자격 증명 테이블을 만들고 수정합니다.	기본 매핑으로 충분하지 않은 경우 GSS 자격 증명을 UNIX UID에 매핑하는 데 사용할 자격 증명 테이블을 만든 다음 항목을 추가합니다.	자격 증명 테이블을 만들고 수정하는 방법 [113]
다른 영역의 사용자 자격 증명을 UNIX UID에 매핑합니다.	자격 증명 테이블에서 정보를 업데이트합니다.	예 4-10. “다른 도메인의 주체를 Kerberos 자격 증명 테이블에 추가”
두 개의 유사 영역 간에 자격 증명 매핑을 만듭니다.	영역이 암호 파일을 공유하는 경우 영역 간에 UID를 매핑합니다.	영역 간 자격 증명 매핑 제공 방법 [114]
Kerberos 인증과 파일 시스템을 공유합니다.	Kerberos 인증이 필요하도록 보안 모드와 파일 시스템을 공유합니다.	Kerberos 보안 모드가 여러 개인 보안 NFS 환경 설정 방법 [115]

▼ Kerberos NFS 서버 구성 방법

이 절차에서는 다음 구성 매개변수를 사용합니다.

- 영역 이름 = `EXAMPLE.COM`
- DNS 도메인 이름 = `example.com`
- NFS 서버 = `denver.example.com`
- `admin` 주체 = `kws/admin`

시작하기 전에 NFS 서버에서 `root` 역할을 맡아야 합니다. 자세한 내용은 “Oracle Solaris 11.2의 사용자 및 프로세스 보안 ”의 “지정된 관리 권한 사용”을 참조하십시오.

마스터 KDC를 구성합니다. “KDC와 Kerberos 클라이언트 간의 클럭 동기화” [121]에 설명된 대로 클럭을 동기화합니다. 프로세스를 완전히 테스트하려면 여러 클라이언트가 필요합니다.

1. **NFS 서버를 Kerberos 클라이언트로 구성합니다.**

   “Kerberos 클라이언트 구성” [91]의 지침을 따릅니다.

2. **NFS 서비스 주체를 추가합니다.**

   `kadmin` 명령을 사용합니다.

   ```
   denver # /usr/sbin/kadmin -p kws/admin
   Enter password: xxxxxxxx
   ```

```
kadmin:
```

a. **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.**

2.a단계에서 만든 각각의 고유한 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.
kadmin:
```

c. **kadmin을 종료합니다.**

```
kadmin: quit
```

3. **필요한 경우 특수한 GSS 자격 증명 맵을 만듭니다.**

일반적으로 Kerberos 서비스는 GSS 자격 증명과 UNIX UID 간에 적절한 맵을 생성합니다. 기본 매핑에 대해서는 “UNIX 자격 증명과 GSS 자격 증명 간 매핑” [64]을 참조하십시오. 기본 매핑으로 충분하지 않을 경우 자세한 내용은 자격 증명 테이블을 만들고 수정하는 방법 [113]을 참조하십시오.

4. **Kerberos 보안 모드와 NFS 파일 시스템을 공유합니다.**

자세한 내용은 Kerberos 보안 모드가 여러 개인 보안 NFS 환경 설정 방법 [115]을 참조하십시오.

▼ 자격 증명 테이블을 만들고 수정하는 방법

gsscred 자격 증명 테이블은 NFS 서버가 Kerberos 자격 증명을 UNIX UID에 매핑하는 데 사용합니다. 기본적으로 주체 이름의 주요 부분이 UNIX 로그인 이름과 일치됩니다. 이 테이블은 기본 매핑이 충분하지 않은 경우 만듭니다.

시작하기 전에 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **/etc/gss/gsscred.conf에 표시된 보안 방식이 files인지 확인합니다.**

```
# cat /etc/gss/gsscred.conf
...
#
files
#
#
# Syslog (auth.debug) a message for GSS cred to Unix cred mapping
#SYSLOG_UID_MAPPING=yes
```

2. **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 제거할 수 있습니다. 자세한 내용은 gsscred(1M) 매뉴얼 페이지를 참조하십시오.

3. **(옵션) 항목을 자격 증명 테이블에 추가합니다.**

예를 들어, NFS 서버에서 root 역할로 sandy/admin 주체를 UID 3736에 매핑하는 항목을 추가합니다. -a 옵션은 항목을 자격 증명 테이블에 추가합니다.

```
# gsscred -m kerberos_v5 -n sandy/admin -u 3736 -a
```

예 4-10 다른 도메인의 주체를 Kerberos 자격 증명 테이블에 추가

이 예에서는 FQDN(정규화된 도메인 이름)을 사용하여 다른 도메인의 주체를 지정합니다.

```
# 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. *username*@CORP.EXAMPLE.COM 및 *username*@SALES.EXAMPLE.COM에 대한 자격 증명이 동일한 UID에 매핑됩니다.

시작하기 전에 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

- **클라이언트 시스템에서 default_realm 및 auth_to_local_realm 항목을 krb5.conf 파일에 추가합니다.**

```
# pfedit /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 문제를 해결하는 방법에 대한 도움말은 "GSS 자격 증명에서 UNIX 자격 증명으로 매핑" [183]을 참조하십시오.

▼ Kerberos 보안 모드가 여러 개인 보안 NFS 환경 설정 방법

이 절차에서는 NFS 서버가 여러 보안 보드를 사용하여 보안 NFS 액세스를 제공할 수 있도록 합니다. 클라이언트가 NFS 서버와 보안 모드를 협상하는 경우 클라이언트는 서버에서 제공하는 첫번째 모드를 사용합니다. 이 모드는 서버가 공유하는 파일 시스템의 모든 후속 클라이언트 요청에 사용됩니다.

시작하기 전에 NFS 서버에서 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **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 Kerberos NFS 서버 구성 방법 [112]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
   ```

   자세한 내용은 `klist`(1) 매뉴얼 페이지를 참조하십시오.

2. **`/etc/nfssec.conf` 파일에서 Kerberos 보안 모드를 사용으로 설정합니다.**

   `/etc/nfssec.conf` 파일에서 Kerberos 보안 모드를 주석 처리하는 "#"을 제거합니다.

   ```
   # pfedit /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
```

3. **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.

4. **(옵션) 기본 보안 모드가 아닌 보안 모드를 사용하여 파일 시스템을 마운트합니다.**

   기본 보안 모드를 그대로 적용할 수 있는 경우 이 절차를 수행하지 마십시오.

   - **자동 마운트가 사용 중인 경우 기본 모드가 아닌 다른 보안 모드로 들어가도록 `auto_master` 데이터베이스를 편집합니다.**

     file-system `auto_home -nosuid,sec=`*mode*

   - **수동으로 `mount` 명령을 실행하여 기본 모드가 아닌 다른 모드를 통해 파일 시스템에 액세스합니다.**

     # mount -F nfs -o sec=*mode file-system*

예 4-11 하나의 Kerberos 보안 모드와 파일 시스템 공유

이 예에서는 `krb5` 보안 모드를 사용한 인증이 성공해야만 NFS 서비스를 통해 파일에 액세스할 수 있습니다.

```
# share -F nfs -o sec=krb5 /export/home
```

예 4-12 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 서비스에 대한 액세스를 위해 지연된 실행 구성

기본 Kerberos 환경에서는 제한된 시간이 지나면 자격 증명이 만료됩니다. cron 및 at과 같이 아무 때나 실행될 수 있는 프로세스의 경우 시간 제한이 있으면 문제가 발생합니다. 이 절차에서는 Kerberos를 통해 인증된 서비스를 사용해야 하는 지연된 실행 프로세스를 지원하도록 Kerberos 환경을 구성하는 방법을 설명합니다. Oracle Solaris에서는 PAM 모듈을 제공하고, 서비스 키를 사용하며, kclient 구성 옵션을 사용하여 Kerberos 인증을 사용한 지연 실행이 가능하도록 그리고 대체 솔루션보다 더 안전하도록 설정합니다.

참고 - cron 서버가 손상된 경우 공격자가 사용자를 가장하여 cron 서버에 대해 구성된 대상 서비스에 액세스할 수 있습니다. 따라서 이 절차에서 구성된 cron 호스트는 사용자에 대해 중간 서비스를 제공하므로 더 민감한 시스템입니다.

▼ Kerberos 서비스에 액세스할 수 있도록 cron 호스트를 구성하는 방법

이 절차에서는 다음 구성 매개변수를 사용합니다.

- cron 호스트 = host1.example.com
- NFS 서버 = host2.example.com
- LDAP 서버 = host3.example.com

1. **Kerberos를 지원하도록 cron 서비스를 구성합니다.**

   - **cron 호스트에 대해 Kerberos가 구성되지 않은 경우에는 시스템에서 kclient 명령을 실행합니다.**

     자세한 내용은 kclient(1M) 매뉴얼 페이지를 참조하십시오.

     예를 들어, 다음 명령은 EXAMPLE.COM 영역에 클라이언트를 구성합니다. 이 명령은 include 방식을 사용하여 /etc/pam.d/cron 서비스 파일에 pam_gss_s4u 파일을 포함합니다.

     ```
     # kclient -s cron:optional -R EXAMPLE.COM
     ```

   - **cron 호스트에 대해 Kerberos가 이미 구성된 경우에는 해당 호스트에서 cron 서비스에 대한 PAM 구성을 수동으로 수정해야 합니다.**

     cron 서비스에 대한 PAM 구성에 pam_gss_s4u 파일이 포함되어 있는지 확인합니다.

     ```
     # cd /etc/pam.d ; cp cron cron.orig
     # pfedit cron
           # PAM include file for optional set credentials
           # through Kerberos keytab and GSS-API S4U support
           auth include          pam_gss_s4u
     ```

2. **cron 호스트가 위임으로 작동하도록 설정합니다.**
   예를 들면 다음과 같습니다.

```
# kadmin -p kws/admin
Enter password: xxxxxxxx
kadmin: modprinc +ok_as_delegate host/host1.example.com@EXAMPLE.COM
Principal “host/host1.example.com@EXAMPLE.COM” modified.
```

3. **cron 작업을 만든 사용자를 대신해 cron 호스트가 자신에 대한 티켓을 요청하도록 설정합니다.**

```
kadmin: modprinc +ok_to_auth_as_delegate host/host1.example.com@EXAMPLE.COM
Principal “host/host1.example.com@EXAMPLE.COM” modified.
kadmin: quit
```

4. **LDAP에서 cron 호스트가 위임으로 사용하는 서비스를 지정하도록 호스트를 구성합니다.**
   예를 들어, cron 호스트가 Kerberos화된 NFS 서버인 host2에 있는 사용자의 홈 디렉토리에 액세스할 수 있도록 하려면 cron 서버의 LDAP 정의에 있는 krbAllowedToDelegateTo 매개변수에 NFS 호스트를 추가합니다.

   a. **위임 지정을 만듭니다.**

```
# pfedit /tmp/delghost.ldif
dn: krbprincipalname=host/
host1.example.com@EXAMPLE.COM,cn=EXAMPLE.COM,cn=krbcontainer,dc=example,dc=com
changetype: modify
krbAllowedToDelegateTo: nfs/host2.example.com@EXAMPLE.COM
```

   b. **지정을 LDAP에 추가합니다.**

```
# ldapmodify -h host3 -D "cn=directory manager" -f delghost.ldif
```

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. 자세한 내용은 “Kerberos 영역 계층 구조” [58]를 참조하십시오.

▼ 계층 영역 간 인증 설정 방법

이 절차의 예에서는 CORP.EAST.EXAMPLE.COM과 EAST.EXAMPLE.COM 간에 양방향 영역 간 인증을 설정합니다. 이 절차는 두 영역의 마스터 KDC에서 수행해야 합니다.

시작하기 전에 각 영역에 대한 마스터 KDC를 구성합니다. 인증 프로세스를 완전히 테스트하려면 여러 클라이언트가 필요합니다.

두 KDC 서버에서 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **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: xxxxxxxx
kadmin: addprinc krbtgt/CORP.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM
Enter password for principal krbtgt/CORP.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM:     /** Type strong
 password **/
kadmin: addprinc krbtgt/EAST.EXAMPLE.COM@CORP.EAST.EXAMPLE.COM
Enter password for principal krbtgt/EAST.EXAMPLE.COM@CORP.EAST.EXAMPLE.COM:     /** Type strong
 password **/
kadmin: quit
```

   참고 - 두 암호를 안전한 위치에 저장하고 보관하십시오.

2. **Kerberos 구성 파일에 모든 영역에 대한 도메인 이름을 정의할 항목을 추가합니다.**

```
# pfedit /etc/krb5/krb5.conf
[libdefaults]
.
.
[domain_realm]
.corp.east.example.com = CORP.EAST.EXAMPLE.COM
.east.example.com = EAST.EXAMPLE.COM
```

   이 예에서는 `CORP.EAST.EXAMPLE.COM` 및 `EAST.EXAMPLE.COM` 영역에 대한 도메인 이름이 정의됩니다. 파일은 하향식으로 검색되므로 이 파일에서 하위 도메인이 도메인 이름 앞에 나와야 합니다.

3. **Kerberos 구성 파일을 이 영역의 모든 클라이언트에 복사합니다.**

   영역 간 인증이 작동하려면 모든 시스템(슬레이브 KDC 및 기타 서버 포함)이 `/etc/krb5/krb5.conf`의 마스터 KDC 버전을 사용해야 합니다.

4. **두번째 영역에서 이 절차를 반복합니다.**

   참고 - 각 서비스 주체에 대해 지정된 암호는 두 KDC에서 같아야 합니다. 따라서 서비스 주체 `krbtgt/CORP.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM`에 대한 암호는 두 영역에서 동일해야 합니다.

▼ 직접 영역 간 인증 설정 방법

이 절차의 예에서는 두 개의 영역(CORP.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 구성합니다. 인증 프로세스를 완전히 테스트하려면 여러 클라이언트가 필요합니다.

두 KDC 서버에서 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **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: xxxxxxxx
kadmin: addprinc krbtgt/CORP.EAST.EXAMPLE.COM@SALES.WEST.EXAMPLE.COM
Enter password for principal
krbtgt/CORP.EAST.EXAMPLE.COM@SALES.WEST.EXAMPLE.COM:      /** Type strong password **/
kadmin: addprinc krbtgt/SALES.WEST.EXAMPLE.COM@CORP.EAST.EXAMPLE.COM
Enter password for principal
krbtgt/SALES.WEST.EXAMPLE.COM@CORP.EAST.EXAMPLE.COM:      /** Type strong password **/
kadmin: quit
```

2. **Kerberos 구성 파일에서 원격 영역에 대한 직접 경로를 정의할 항목을 추가합니다.**

   이 예에서는 CORP.EAST.EXAMPLE.COM 영역의 클라이언트를 보여 줍니다. SALES.WEST.EXAMPLE.COM 영역의 적절한 정의를 추가하려면 영역 이름을 교체합니다.

```
# pfedit /etc/krb5/krb5.conf
[libdefaults]
.
.
[capaths]
CORP.EAST.EXAMPLE.COM = {
SALES.WEST.EXAMPLE.COM = .
}

SALES.WEST.EXAMPLE.COM = {
CORP.EAST.EXAMPLE.COM = .
}
```

3. **Kerberos 구성 파일을 현재 영역의 모든 클라이언트에 복사합니다.**

   영역 간 인증이 작동하려면 모든 시스템(슬레이브 KDC 및 기타 서버 포함)이 Kerberos 구성 파일(/etc/krb5/krb5.conf)의 새 버전을 사용해야 합니다.

4. **두번째 영역에 대해 이 절차를 반복합니다.**

KDC와 Kerberos 클라이언트 간의 클럭 동기화

Kerberos 인증 시스템에 참여하는 모든 호스트의 내부 클럭은 지정된 최대 시간 범위(클럭 불균형이라고 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 Kerberos 프로토콜 메시지를 추적해야 할 기간을 결정합니다. 따라서 클럭 불균형 값이 클수록 애플리케이션 서버가 수집해야 할 정보가 많아집니다.

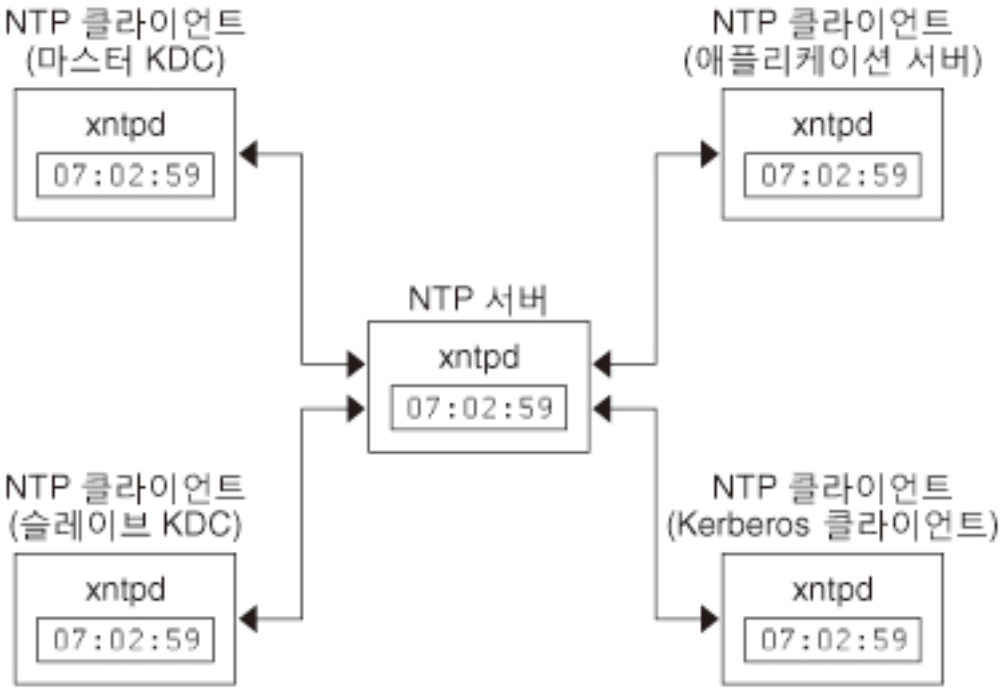
최대 클럭 불균형의 기본값은 300초(5분)입니다. `krb5.conf` 파일의 `libdefaults` 절에서 이 기본값을 변경할 수 있습니다.

참고 - 보안상 클럭 불균형을 300초 이상으로 늘리지 마십시오.

KDC와 Kerberos 클라이언트 간의 클럭은 동기화 상태로 유지되어야 하므로 동기화를 위해 NTP(Network Time Protocol) 소프트웨어를 사용하십시오. Oracle Solaris 소프트웨어에는 University of Delaware의 NTP 공용 도메인 소프트웨어가 포함되어 있습니다. 설명서는 NTP Documentation에서 제공합니다.

NTP를 통해 네트워크 환경에서 정확한 시간 또는 네트워크 클럭 동기화를 관리하거나 모두 관리할 수 있습니다. NTP는 서버-클라이언트 프로토콜입니다. 한 시스템은 마스터 클럭, 즉 NTP 서버입니다. 기타 모든 시스템은 시스템 클럭을 마스터 클럭과 동기화하는 NTP 클라이언트입니다. 클럭을 동기화하기 위해 NTP는 인터넷 표준 시간 서버와의 계약에 따라 UNIX 시스템 시간을 설정 및 유지 관리하는 `xntpd` 데몬을 사용합니다. 다음 그림은 이 서버-클라이언트 NTP 구현의 예를 보여 줍니다.

그림 4-1 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.
2. 네트워크에서 KDC 및 Kerberos 클라이언트를 구성하면 NTP 서버의 NTP 클라이언트가 되도록 설정됩니다. NTP 클라이언트로 구성하려면 마스터 KDC로 돌아가십시오.
3. 모든 시스템에서 NTP 서비스를 사용으로 설정합니다.

마스터 KDC와 슬레이브 KDC 교체

이 절의 절차에서는 마스터 KDC와 슬레이브 KDC를 간편하게 교체합니다. 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 설정하려고 할 슬레이브 KDC의 영역(증분 전파를 사용하는 영역)에서 이 절차를 수행합니다.

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

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 KDC를 설치합니다.**
   교체에 앞서 이 서버가 영역의 기타 모든 슬레이브 KDC처럼 작동해야 합니다. 자세한 내용은 수동으로 슬레이브 KDC를 구성하는 방법 [79]을 참조하십시오.

3. **설치가 완료되면 마스터 KDC 명령을 이동합니다.**
   이 슬레이브 KDC에서 마스터 KDC 명령을 실행하면 안됩니다.

```
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 교체 가능한 슬레이브 KDC 구성 방법 [123] 절차를 완료합니다.

`root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **새 마스터 KDC에서 `kadmin`을 시작합니다.**

```
kdc4 # /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
kadmin:
```

2. **`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@EXAMPLE.COM" created.
kadmin: addprinc -randkey -allow_tgs_req -clearpolicy kadmin/kdc4.example.com
Principal "kadmin/kdc4.example.com@EXAMPLE.COM" created.
kadmin:
```

3. **kadmin을 종료합니다.**

```
kadmin: quit
```

4. **새 마스터 KDC에서 강제로 동기화를 수행합니다.**
   다음 단계에서는 슬레이브 서버에서 강제로 전체 KDC 업데이트를 수행합니다.

   a. **krb5kdc 서비스를 사용 안함으로 설정하고 해당 로그 파일을 제거합니다.**

```
kdc4 # svcadm disable network/security/krb5kdc
kdc4 # rm /var/krb5/principal.ulog
```

   b. **업데이트가 완료되었는지 확인합니다.**

```
kdc4 # /usr/sbin/kproplog -h
```

   c. **KDC 서비스를 다시 시작합니다.**

```
kdc4 # svcadm enable -r network/security/krb5kdc
```

   d. **새 마스터 KDC 서버에 대한 업데이트 로그를 다시 초기화합니다.**

```
kdc4 # svcadm disable network/security/krb5kdc
kdc4 # rm /var/krb5/principal.ulog
```

5. **이전 마스터 KDC에서 kadmin 및 krb5kdc 서비스를 종료합니다.**
   kadmin 서비스를 종료하면 KDC 데이터베이스를 변경할 수 없게 됩니다.

```
kdc1 # svcadm disable network/security/kadmin
kdc1 # svcadm disable network/security/krb5kdc
```

6. **이전 마스터 KDC에서 전파 요청 폴링 시간을 지정합니다.**
   /etc/krb5/kdc.conf에서 sunw_dbprop_master_ulogsize 항목을 주석 처리하고 슬레이브의 폴링 간격을 정의하는 항목을 추가합니다. 이 항목은 폴링 시간을 2분으로 설정합니다.

```
kdc1 # pfedit /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
}
```

7. **이전 마스터 KDC에서 마스터 KDC 명령 및 `kadm5.acl` 파일을 이동합니다.**

   이전 마스터 KDC에서 마스터 KDC 명령을 실행하면 안됩니다.

```
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
```

8. **DNS 서버에서 마스터 KDC에 대한 별칭 이름을 변경합니다.**

   서버를 변경하려면 `example.com` 영역 파일을 편집하고 `masterkdc`에 대한 항목을 변경합니다.

```
masterkdc IN CNAME kdc4
```

9. **DNS 서버에서 새 별칭 정보를 다시 로드합니다.**

```
# svcadm refresh network/dns/server
```

10. **새 마스터 KDC에서 마스터 KDC 명령 및 슬레이브 `kpropd.acl` 파일을 이동합니다.**

    3단계 of 교체 가능한 슬레이브 KDC 구성 방법 [123]에서 마스터 KDC 명령을 이동했습니다.

```
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
```

11. **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 생성할 수 있는 슬레이브를 모두 나열해야 합니다. 자세한 내용은 `kadm5.acl`(4) 매뉴얼 페이지를 참조하십시오.

```
kdc4 # pfedit /etc/krb5/kadm5.acl
kws/admin@EXAMPLE.COM   *
kiprop/kdc1.example.com@EXAMPLE.COM p
```

12. **새 마스터 KDC에서 `kdc.conf` 파일에 업데이트 로그 크기를 지정합니다.**

    `sunw_dbprop_slave_poll` 항목을 주석 처리하고 `sunw_dbprop_master_ulogsize`를 정의하는 항목을 추가합니다. 이 항목은 로그 크기를 1000개 항목으로 설정합니다.

```
kdc1 # pfedit /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
}
```

13. **새 마스터 KDC에서 `kadmin` 및 `krb5kdc` 서비스를 사용으로 설정합니다.**

```
kdc4 # svcadm enable -r network/security/krb5kdc
kdc4 # svcadm enable -r network/security/kadmin
```

14. **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 principal kws/admin@EXAMPLE.COM with password.
Enter password: xxxxxxxx
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.
kadmin: quit
```

15. **이전 마스터 KDC에서 `krb5.conf`에 나열된 각 KDC에 대한 항목을 전파 구성 파일에 추가합니다.**

```
kdc1 # pfedit /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
```

16. **이전 마스터 KDC에서 `kpropd` 및 `krb5kdc` 서비스를 사용으로 설정합니다.**

```
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)를 위한 몇 가지 절차를 제공합니다. 데이터베이스 초기 설정 단계는 수동으로 마스터 KDC를 구성하는 방법 [74]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. 이 경우 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.

전파가 충분히 자주 실행되도록 하거나 Kerberos 데이터베이스 변경 빈도에 따라 서버에서 증분 전파를 구성해야 합니다. 다른 전파 방법보다 증분 전파가 기본적으로 사용됩니다. 데이터베이스를 수동으로 전파할 경우 관리 오버헤드가 늘어나고 전체 전파는 효율성이 떨어집니다.

주의 - 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 설명서에 설명된 Kerberos 설치 및 후속 구성 단계에서는 마스터 KDC와 슬레이브 KDC에 동일한 `kpropd.acl` 파일을 사용하도록 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. 모든 KDC에서 동일한 복사본을 사용하면 유지 관리가 간편해집니다.

`kprop_script` 명령

`kprop_script` 명령은 `kprop` 명령을 사용하여 Kerberos 데이터베이스를 다른 KDC에 전파합니다. `kprop_script` 명령이 슬레이브 KDC에서 실행되면 Kerberos 데이터베이스의 슬레이

브 KDC 복사본을 다른 KDC에 전파합니다. kprop_script는 전파할 KDC를 나타내는 인수에 대해 공백으로 구분된 호스트 이름 목록을 허용합니다.

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

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. 다음 번 데이터베이스 자동 전파 시 복사본이 다시 설치되므로 슬레이브 KDC에서 데이터가 손상된 경우 문제가 되지 않습니다. 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 덮어씁니다.

이 시나리오를 방지하려면 주기적으로 slave_datatrans 덤프 파일이 다른 위치에 복사되거나 kdb5_util의 dump 명령을 사용하여 별도의 다른 백업 복사본이 만들어지도록 cron 작업을 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 사용하여 데이터베이스 덤프 파일을 전파하십시오. 이 명령은 전송되는 데이터를 암호화합니다. 또한 kprop는 슬레이브 KDC로만 데이터를 전파하므로 허용되지 않은 호스트로 데이터베이스 덤프 파일이 잘못 전송될 가능성이 최소화됩니다.

예 4-13 Kerberos 데이터베이스 수동 백업

kdb5_util 명령의 dump 명령을 사용하여 데이터베이스를 백업합니다. root가 소유한 디렉토리에서 이 명령을 실행합니다.

```
# /usr/sbin/kdb5_util dump
```

다음 예에서는 이름이 dumpfile인 파일에 Kerberos 데이터베이스가 백업됩니다. -상세 정보 표시 옵션이 지정되었으므로 각 주체가 백업된 대로 인쇄됩니다. 주체를 지정하지 않았으므로 전체 데이터베이스가 백업됩니다.

```
# kdb5_util dump -verbose /var/user/kadmin/dumpfile
kadmin/kdc1.corp.example.com@CORP.EXAMPLE.COM
krbtgt/CORP.EXAMPLE.COM@CORP.EXAMPLE.COM
kadmin/history@CORP.EXAMPLE.COM
pak/admin@CORP.EXAMPLE.COM
pak@CORP.EXAMPLE.COM
changepw/kdc1.corp.example.com@CORP.EXAMPLE.COM
```

다음 예에서는 pak 및 pak/admin만 덤프에 포함됩니다.

```
# kdb5_util dump -verbose pakfile pak/admin@CORP.EXAMPLE.COM pak@CORP.EXAMPLE.COM
pak/admin@CORP.EXAMPLE.COM
pak@CORP.EXAMPLE.COM
```

자세한 내용은 kdb5_util(1M) 매뉴얼 페이지를 참조하십시오.

▼ Kerberos 데이터베이스 백업 복원 방법

시작하기 전에 KDC 마스터에서 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **마스터에서 KDC 데몬을 중지합니다.**

```
kdc1 # svcadm disable network/security/krb5kdc
kdc1 # svcadm disable network/security/kadmin
```

2. **kdb_util 명령의 load 명령을 사용하여 Kerberos 데이터베이스를 복원합니다.**
예를 들어 예 4-13. "Kerberos 데이터베이스 수동 백업"의 dumpfile 백업을 로드합니다.

```
# /usr/sbin/kdb5_util load /var/user/kadmin/dumpfile
```

3. **KDC 데몬을 시작합니다.**

```
kdc1 # svcadm enable -r network/security/krb5kdc
kdc1 # svcadm enable -r network/security/kadmin
```

예 4-14 Kerberos 데이터베이스 복원

다음 예에서는 이름이 database1인 데이터베이스가 dumpfile 파일의 현재 디렉토리에 복원됩니다. -update 옵션이 지정되지 않았으므로 새 데이터베이스가 만들어집니다.

```
# kdb5_util load -d database1 dumpfile
```

▼ 서버 업그레이드 후 Kerberos 데이터베이스 변환 방법

KDC 데이터베이스가 이전 릴리스를 실행하는 서버에 만들어진 경우 데이터베이스를 변환하면 향상된 데이터베이스 형식을 사용할 수 있습니다.

시작하기 전에 데이터베이스가 이전 형식을 사용하고 있는 경우에만 이 절차를 수행합니다.

KDC 마스터에서 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

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 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 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **`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 # pfedit /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
}
```

2. **kiprop 주체를 만듭니다.**
   kiprop 주체는 마스터 KDC 서버를 인증하고 마스터 KDC로부터의 업데이트를 허가하는 데 사용됩니다.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
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:
```

3. **마스터 KDC에서 kadm5.acl에 kiprop 항목을 추가합니다.**
   이 항목은 마스터 KDC가 kdc2 서버의 증분 전파 요청을 수신할 수 있도록 합니다.

```
kdc1 # pfedit /etc/krb5/kadm5.acl
*/admin@EXAMPLE.COM *
kiprop/kdc2.example.com@EXAMPLE.COM p
```

4. **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/krb5/kprop_script kdc2.example.com
```

5. **kadmind를 다시 시작합니다.**

```
kdc1 # svcadm restart network/security/kadmin
```

6. **증분 전파를 사용하는 모든 슬레이브 KDC 서버를 재구성합니다.**
   전체 지침은 증분 전파를 사용하도록 슬레이브 KDC를 재구성하는 방법 [132]을 참조하십시오.

▼ 증분 전파를 사용하도록 슬레이브 KDC를 재구성하는 방법

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **`kdc.conf`에 항목을 추가합니다.**
   첫번째 새 항목은 증분 전파를 사용으로 설정합니다. 두번째 새 항목은 폴링 시간을 2분으로 설정합니다.

```
kdc2 # pfedit /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
}
```

2. **`krb5.keytab` 파일에 `kiprop` 주체를 추가합니다.**

```
kdc2 # /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
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.
kadmin: quit
```

3. **`kpropd`를 다시 시작합니다.**

```
kdc2 # svcadm restart network/security/krb5_prop
```

▼ KDC 서버 동기화 여부 확인 방법

증분 전파가 구성된 경우 이 절차는 슬레이브 KDC의 정보가 업데이트되었는지 확인합니다.

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **KDC 마스터 서버에서 `kproplog` 명령을 실행합니다.**

```
kdc1 # /usr/sbin/kproplog -h
```

2. **KDC 슬레이브 서버에서 `kproplog` 명령을 실행합니다.**

```
kdc2 # /usr/sbin/kproplog -h
```

3. **최종 일련 번호 값과 최종 시간 기록 값이 일치하는지 확인합니다.**

예 4-15 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: Wed Dec 4 00:59:27 2013
Last time stamp: Wed Dec 4 01:06:13 2013
```

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: Wed Dec 4 01:06:13 2013
```

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 데이터베이스 전파

일반적으로 `cron` 작업은 Kerberos 데이터베이스를 슬레이브 KDC에 전파합니다. 주기적 `cron` 작업 이외에 슬레이브 KDC를 마스터 KDC와 동기화해야 할 경우 `/usr/lib/krb5/kprop_script` 명령과 `kprop` 명령의 두 가지 옵션이 있습니다. 자세한 내용은 스크립트 및 `kprop`(1M) 매뉴얼 페이지를 검토하십시오.

주의 - 슬레이브 KDC에서 증분 전파가 사용으로 설정된 경우에는 이러한 명령을 사용하지 마십시오.

▼ 수동으로 슬레이브 KDC에 Kerberos 데이터베이스를 전파하는 방법

1. **슬레이브 KDC에서 증분 전파가 사용으로 설정되지 않았는지 확인합니다.**

```
slave# grep sunw_dbprop_enable /etc/krb5/kdc.conf
sunw_dbprop_enable = true
```

2. **값이 `true`인 경우 증분 전파를 사용 안함으로 설정하고 `krb5_prop` 서비스를 다시 시작합니다.**

```
slave# cp /etc/krb5/kdc.conf /etc/krb5/kdc.conf.sav
slave# pfedit /etc/krb5/kdc.conf
...
sunw_dbprop_enable = false
...

slave# svcadm restart krb5_prop
```

3. **마스터 KDC에서 다음 명령 중 하나를 사용하여 마스터 KDC 데이터베이스를 슬레이브 KDC에 전파합니다.**

   - **`kprop_script` 명령은 슬레이브 KDC를 동기화하기 전에 데이터베이스를 백업합니다.**

     ```
     master# /usr/lib/krb5/kprop_script slave-KDC
     ```

   - **`kprop` 명령은 먼저 Kerberos 데이터베이스의 새 백업을 만들지 않고 현재 데이터베이스 백업을 전파합니다.**

     ```
     master# /usr/lib/krb5/kprop -f /var/krb5/slave_datatrans slave-KDC
     ```

4. **(옵션) 수동 전파가 완료되면 원래 `krb5.conf` 파일을 복원합니다.**

```
slave# mv /etc/krb5/kdc.conf.sav /etc/krb5/kdc.conf
```

Kerberos에 대해 병렬 전파 설정

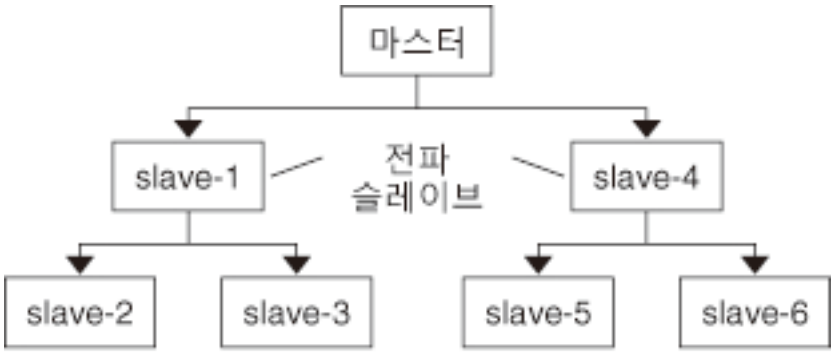
대부분의 경우 마스터 KDC는 슬레이브 KDC에 Kerberos 데이터베이스를 전파하는 데만 사용됩니다. 하지만 사이트에 KDC가 여러 개 있을 경우 전파 프로세스를 공유하는 로드(병렬 전파라고 함)를 사용할 수 있습니다.

주의 - 증분 전파를 사용 중인 경우에는 병렬 전파를 구성하지 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 4-2 “Kerberos의 병렬 전파 구성 예” 참조) 여기서 `slave-1`에서 `slave-3`까지는 하나의 논리적 그룹화로 구성되고 `slave-4`에서 `slave-6`까지는 다른 논리적 그룹화로 구성된다고 가정합니다. 이 경우 병렬 전파를 설정하려면 마스터 KDC가 데이터베이스를 `slave-1` 및 `slave-4`에 전파하도록 할 수 있습니다. 이런 방법으로 해당 KDC 슬레이브는 데이터베이스를 그룹 내 KDC 슬레이브에 전파할 수 있습니다.

그림 4-2 Kerberos의 병렬 전파 구성 예



병렬 전파 설정을 위한 구성 단계

병렬 전파를 사용으로 설정하기 위한 개략적인 구성 단계는 다음과 같습니다.

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.

참고 - 전파 슬레이브를 전파하는 데 걸리는 시간은 네트워크 대역폭, Kerberos 데이터베이스 크기 등의 인자에 따라 다릅니다.

---

3. 각 슬레이브 KDC에서 `kpropd.acl` 파일에 전파하는 KDC의 호스트 주체 이름을 추가하여 적절한 권한이 전파되도록 설정합니다.

예 4-16 Kerberos에서 병렬 전파 설정

그림 4-2. "Kerberos의 병렬 전파 구성 예"의 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-1`의 `kprop_script` 항목은 다음과 같이 표시됩니다.

```
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
```

Kerberos 데이터베이스의 stash 파일 관리

stash 파일에는 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 수동으로 제거해야 합니다.

stash 파일을 제거하려면 `rm` 명령을 사용합니다.

```
# rm stash-file
```

여기서 *stash-file*은 stash 파일의 경로입니다. 기본적으로 stash 파일은 `/var/krb5/.k5.`*realm*에 있습니다.

참고 - stash 파일을 다시 만들어야 할 경우 `kdb5_util` 명령의 `-f` 옵션을 사용할 수 있습니다.

▼ Kerberos 데이터베이스에 대한 키를 새로 생성, 사용 및 저장하는 방법

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **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 strong password **/
Re-enter KDC database master key to verify: xxxxxxxx
```

2. **새 마스터 키가 존재하는지 확인합니다.**

```
# kdb5_util list_mkeys
Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-256 CTS mode with 96-bit SHA-1 HMAC, No activate time set
KNVO: 1, Enctype: AES-128 CTS mode with 96-bit SHA-1 HMAC, Active on: Fri Dec 31 18:00:00 CST
 2011 *
```

   이 출력의 별표는 현재 활성 상태의 마스터 키를 식별합니다.

3. **새로 만들어진 마스터 키가 활성화될 시간을 설정합니다.**

```
# date
Fri Jul 11 17:57:00 CDT 2014
# kdb5_util use_mkey 2 'now+2days'
# kdb5_util list_mkeys
Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-256 CTS mode with 96-bit SHA-1 HMAC,
Active on: Sun Jul 13 17:57:15 CDT 2014
KNVO: 1, Enctype: AES-128 CTS mode with 96-bit SHA-1 HMAC,
Active on: Fri Dec 31 18:00:00 CST 2011 *
```

   이 예에서는 새 마스터 키가 모든 KDC에 전파되도록 충분한 시간을 확보하기 위해 날짜가 2일로 설정됩니다. 환경에 맞게 날짜를 조정합니다.

4. **(옵션) 새 주체를 만든 후 새 마스터 키가 사용되고 있는지 확인합니다.**

```
# kadmin.local -q 'getprinc tamiko' |egrep 'Principal|MKey'
Authenticating as principal root/admin@EXAMPLE.COM with password.
Principal: tamiko@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.

5. **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:   tamiko@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
```

6. **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.
```

7. **이전 마스터 키가 제거되었는지 확인합니다.**

```
# kdb5_util list_mkeys
Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-256 CTS mode with 96-bit SHA-1 HMAC,
Active on: Sun Jul 13 17:57:15 CDT 2014 *
```

8. **stash 파일을 업데이트합니다.**

```
# kdb5_util stash
Using existing stashed keys to update stash file.
```

9. **stash 파일이 업데이트되었는지 확인합니다.**

```
# klist -kt /var/krb5/.k5.EXAMPLE.COM
Keytab name: FILE:.k5.EXAMPLE.COM
KVNO Timestamp             Principal
---- --------------- ------------------------------------------------------------
```

```
2 05/11/2014 18:03 K/M@EXAMPLE.COM
```

Kerberos 서버에서 보안 수준 향상

이 절에서는 Kerberos 애플리케이션 서버 및 KDC 서버에서 보안 수준을 향상시키는 방법에 대한 유용한 정보를 제공합니다.

KDC 서버에 대한 액세스 제한

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.

- KDC를 지원하는 하드웨어에 대한 물리적 액세스를 제한합니다.

  KDC 서버 및 해당 모니터가 보안 설비에 있는지 확인합니다. 일반 사용자는 어떤 방법으로도 이 서버에 액세스할 수 없어야 합니다.
- 로컬 디스크 또는 KDC 슬레이브에서 KDC 데이터베이스 백업을 저장합니다.

  테이프가 안전하게 저장된 경우에만 KDC의 테이프 백업을 수행합니다. keytab 파일의 복사본에 대해 동일한 단계를 수행합니다.

  권장되는 방법은 이러한 파일을 다른 시스템과 공유되지 않는 로컬 파일 시스템에 저장하는 것입니다. 저장소 파일 시스템은 마스터 KDC 서버 또는 슬레이브 KDC에 있을 수 있습니다.

사전 파일을 사용하여 암호 보안 수준 향상

Kerberos 서비스는 사전 파일을 사용하여 사전에 있는 단어가 새 자격 증명의 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.

서비스에 사전 파일을 사용하도록 알리는 라인을 KDC 구성 파일(`kdc.conf`)에 추가해야 합니다. 이 예에서는 관리자가 `spell` 유틸리티를 통해 포함한 사전을 사용한 다음 Kerberos 서비스를 다시 시작합니다. 구성 파일에 대한 자세한 설명은 `kdc.conf`(4) 매뉴얼 페이지를 참조하십시오.

```
kdc1 # pfedit /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
}
kdc1 #
kdc1 # svcadm restart -r network/security/krb5kdc
kdc1 # svcadm restart -r network/security/kadmin
```

♦ ♦ ♦ 5 장

5 Kerberos 주체 및 정책 관리

이 장은 주체 및 주체와 관련된 정책을 관리하는 절차를 제공합니다. 또한 호스트의 keytab 파일을 관리하는 방법에 대해서도 설명합니다.

이 장에서는 다음 내용을 다룹니다.

주체 및 정책에 대한 배경 정보는 2장. Kerberos 서비스 정보 및 3장. Kerberos 서비스 계획을 참조하십시오.

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. 주체를 자동으로 만드는 스크립트를 실행할 수도 있습니다. `kadmin` 명령은 로컬 버전과 원격 버전이 있습니다.

- `kadmin` - 네트워크의 어떤 위치에서도 안전하게 작동하도록 Kerberos 인증을 사용합니다.
- `kadmin.local` - 마스터 KDC에서 직접 실행해야 합니다.

두 버전의 기능은 동일합니다. 원격 버전을 사용할 수 있으려면 로컬 버전을 사용하여 데이터베이스를 충분히 구성해야 합니다.

또한 Oracle Solaris에서는 대화식 GUI(그래픽 사용자 인터페이스)인 `gkadmin`을 제공합니다.

이 절에서는 `kadmin.local` 명령의 스크립팅 기능에 대해 설명하고 명령줄과 GUI 인터페이스를 비교합니다.

자동으로 새 Kerberos 주체 만들기

스크립트에 kadmin.local 명령을 사용하여 새 Kerberos 주체를 자동으로 만들 수 있습니다. 많은 새 주체를 데이터베이스에 추가하려는 경우에 자동화가 유용합니다.

다음 셸 스크립트 행은 새 주체를 자동으로 만드는 방법을 보여 줍니다.

```
awk '{ print "ank +needchange -pw", $2, $1 }' < /tmp/princnames |
time /usr/sbin/kadmin.local> /dev/null
```

위 예제는 쉽게 읽을 수 있도록 두 개의 행으로 분할되어 있습니다.

- 이 스크립트는 princnames라는 파일을 읽습니다. 이 파일에는 주체 이름 및 해당 암호가 포함되어 있으며 이 파일은 이러한 정보를 Kerberos 데이터베이스에 추가합니다.

  하나 이상의 공백으로 구분된 각 행에 주체 이름과 해당 암호를 포함하는 princnames 파일을 만들어야 합니다.
- ank 명령은 새 키를 추가합니다. ank는 add_principal 명령의 별칭입니다.
- +needchange 옵션은 처음 로그인하는 경우 새 암호를 입력하라는 프롬프트를 사용자(주체)에게 표시하도록 주체를 구성합니다.

  암호 변경을 요구하도록 설정하면 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.

gkadmin GUI

Kerberos gkadmin GUI는 CLI의 기능 대부분을 제공하며 온라인 도움말을 포함합니다. 다음 표에서는 CLI와 GUI의 차이점을 보여 줍니다.

표 5-1 gkadmin GUI 및 해당하는 명령줄 명령

gkadmin GUI 절차	해당하는 kadmin 명령
주체 목록 보기	list_principals 또는 get_principals
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

gkadmin GUI 절차	해당하는 kadmin 명령
정책 속성 보기	get_policy
새 정책 만들기	add_policy
정책 복제	해당하는 명령줄 명령 없음
정책 수정	modify_policy
정책 삭제	delete_policy

Kerberos gkadmin GUI는 상황에 맞는 도움말을 제공합니다. 브라우저에서 액세스하려는 경우 Oracle URL (http://docs.oracle.com/cd/E23824_01/html/821-1456/aadmin-23.html)을 사용하십시오. 이 파일을 사이트 URL에 복사할 수 있습니다. gkadmin을 사용하도록 호스트를 구성할 때 krb5.conf 파일에 온라인 도움말 URL을 지정합니다.

Kerberos 주체 관리

이 절에서는 kadmin 명령 및 gkadmin GUI를 사용하여 주체를 관리하는 예를 제공합니다. 자세한 내용은 kadmin(1M) 및 gkadmin(1M) 매뉴얼 페이지를 참조하십시오.

Kerberos 주체 및 해당 속성 보기

다음 예에서는 주체 및 해당 속성을 나열하는 방법을 보여 줍니다. 와일드카드를 사용하여 목록을 생성할 수 있습니다. 사용할 수 있는 와일드카드에 대한 자세한 내용은 kadmin(1M) 매뉴얼 페이지에서 expression에 대한 정의를 검토하십시오.

예 5-1 Kerberos 주체 보기

이 예에서는 list_principals 하위 명령을 사용하여 kadmin*와 일치하는 모든 주체를 나열합니다. 인수를 지정하지 않을 경우 list_principals는 Kerberos 데이터베이스에 정의된 모든 주체를 나열합니다.

```
# /usr/sbin/kadmin
kadmin: list_principals kadmin*
kadmin/changepw@EXAMPLE.COM
kadmin/kdc1.example.com@EXAMPLE.COM
kadmin/history@EXAMPLE.COM
```

예 5-2 Kerberos 주체의 속성 보기

다음 예에서는 jdb/admin 주체의 속성을 표시합니다.

```
kadmin: get_principal jdb/admin
Principal: jdb/admin@EXAMPLE.COM

Expiration date: [never]
Last password change: [never]

Password expiration date: Fri Sep 13 11:50:10 PDT 2013
Maximum ticket life: 1 day 16:00:00
Maximum renewable life: 1 day 16:00:00
Last modified: Thu Aug 15 13:30:30 PST 2013 (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
Attributes: REQUIRES_HW_AUTH
Policy: [none]
kadmin: quit
```

예 5-3 `gkadmin` GUI를 사용하여 Kerberos 주체 나열 및 주체에 대한 기본값 설정

이 예에서는 관리자가 새 관리자에게 주체 및 해당 속성 목록을 보여 주려고 하므로 `gkadmin` GUI를 사용합니다. 또한 새로 만들 주체에 대한 새 기본값을 설정합니다.

```
# /usr/sbin/gkadmin
```

창에 주체 이름, 암호, 영역 및 마스터 KDC 필드가 표시됩니다.

관리자가 모든 주체 이름 목록으로 이동한 다음 새 관리자에게 대소문자 구분 필터를 사용하는 방법을 보여 줍니다.

그런 다음 Edit(편집) 메뉴를 누르고 Properties(등록 정보)를 선택합니다. Require Password Change(암호 변경 필요)를 누른 후 변경 사항을 적용합니다.

현재 주체의 속성을 보기 위해 관리자가 주체 목록으로 이동하여 목록에서 주체를 선택합니다. 첫번째 대화 상자에 기본 속성이 표시됩니다. Next(다음) 버튼을 눌러 모든 속성을 표시합니다.

새 Kerberos 주체 만들기

새 주체에 새 정책이 필요한 경우 주체를 만들기 전에 새 정책을 만들어야 합니다. 정책을 만드는 방법은 예 5-10. “새 Kerberos 암호 정책 만들기”을 참조하십시오. 대부분의 Kerberos 정책은 암호 요구 사항을 지정합니다.

예 5-4 새 Kerberos 주체 만들기

다음 예에서는 pak라는 새 주체를 만들고 주체의 정책을 testuser로 설정합니다. 암호화 유형 같은 기타 필수 값에는 기본값이 사용됩니다.

```
# /usr/sbin/kadmin
kadmin: add_principal -policy testuser pak
Enter password for principal "pak@EXAMPLE.COM": xxxxxxxx
Re-enter password for principal "pak@EXAMPLE.COM": xxxxxxxx
Principal "pak@EXAMPLE.COM" created.
kadmin: quit
```

일반적으로 Kerberos 데이터베이스 관리 권한을 가진 사용자는 소수에 불과합니다. 새로 만든 주체에 관리 권한이 필요한 경우 "주체의 Kerberos 관리 권한 수정" [146]을 진행합니다.

Kerberos 주체 수정

다음 예에서는 Kerberos 주체의 암호 속성을 수정하는 방법을 보여 줍니다.

예 5-5 Kerberos 주체의 최대 암호 재시도 횟수 수정

```
# /usr/sbin/kadmin
kadmin: modify_principal jdb
kadmin: maxtries=5
kadmin: quit
```

예 5-6 Kerberos 주체의 암호 수정

```
# /usr/sbin/kadmin
kadmin: change_password jdb
Enter password for principal "jdb": xxxxxxxx
Re-enter password for principal "jdb": xxxxxxxx
Password for "jdb@EXAMPLE.COM" changed.
kadmin: quit
```

Kerberos 주체 삭제

다음 예에서는 주체를 삭제하는 방법을 보여 줍니다. 계속하기 전에 온라인 메시지에 응답합니다.

```
# /usr/sbin/kadmin
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
```

이 주체를 모든 ACL에서 제거하려면 "주체의 Kerberos 관리 권한 수정" [146]을 참조하십시오.

gkadmin GUI를 사용하여 Kerberos 주체 복제

gkadmin GUI를 사용하여 주체를 복제할 수 있습니다. 이 절차의 경우 해당하는 명령줄 명령은 없습니다.

1. /usr/sbin/gkadmin 명령을 사용하여 GUI를 시작한 후 Principals(주체) 탭을 누르고 복제할 주체를 선택합니다.
2. Duplicate(복제) 버튼을 누릅니다. 이 작업은 주체 이름 및 암호를 제외하고 선택한 주체의 모든 속성을 복제합니다.
3. 새 이름 및 암호를 지정한 다음 Save(저장)를 눌러 정확한 복제본을 만듭니다.
4. 복제본을 수정하려면 주체의 속성 값을 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.

일반적으로 Kerberos 데이터베이스 관리 권한을 가진 사용자는 소수에 불과합니다. 새로 만든 주체에 관리 권한이 필요한 경우 "주체의 Kerberos 관리 권한 수정" [146]을 진행합니다.

주체의 Kerberos 관리 권한 수정

Kerberos 데이터베이스를 관리할 수 있는 권한을 가진 소수의 사용자는 액세스 제어 목록(ACL)에 지정되어 있습니다. 이 목록은 /etc/krb5/kadm5.acl 파일에서 항목으로 유지 관리됩니다. 자세한 내용은 kadm5.acl(4) 매뉴얼 페이지를 참조하십시오.

항목을 kadm5.acl 파일에 추가하려면 pfedit 명령을 사용합니다.

```
# pfedit /usr/krb5/kadm5.acl
```

kadm5.acl 파일에 있는 항목의 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 jdb에게는 관리용 주체인 jdb/admin이 있을 수 있습니다. 주체가 둘이므로 사용자 jdb는 관리 권한이 필요한 경우에만 jdb/admin 티켓을 얻게 됩니다.

- *privileges* - 주체가 수행하거나 수행할 수 없는 작업을 지정합니다. 이 필드는 다음 문자 목록으로 된 문자열로 구성됩니다. 문자가 대문자이거나 지정되지 않은 경우 작업이 허용되지 않습니다. 소문자일 경우 작업이 허용됩니다.
  - [A]a - 주체 또는 정책의 추가를 허용하거나 허용하지 않습니다.
  - [C]c - 주체 암호 변경을 허용하거나 허용하지 않습니다.
  - [D]d - 주체 또는 정책의 삭제를 허용하거나 허용하지 않습니다.
  - [I]i - erberos 데이터베이스에 대한 조회를 허용하거나 허용하지 않습니다.
  - [L]l - 주체 또는 정책의 삭제를 허용하거나 허용하지 않습니다.
  - [M]m - 주체 또는 정책의 수정을 허용하거나 허용하지 않습니다.
  - x or * - 모든 권한을 허용합니다(admcil).
- *principal-target* - 이 필드에 주체가 지정된 경우 주체의 권한이 해당 주체에만 적용됩니다. 주체 그룹에 권한을 지정하려면 *principal-target*에 '*' 와일드카드를 사용합니다.

예 5-7 Kerberos 주체의 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 나열하고 이 주체에 대해 조회할 수 있는 권한을 jdb@EXAMPLE.COM 주체에 부여합니다.

```
jdb@EXAMPLE.COM li */root@EXAMPLE.COM
```

Kerberos 정책 관리

이 절에서는 kadmin 명령 및 gkadmin GUI를 사용하여 Kerberos 정책을 관리하는 예를 제공합니다. 대부분의 Kerberos 정책은 암호 요구 사항을 지정합니다.

정책을 관리하는 단계는 주체를 관리하는 단계와 비슷합니다. 자세한 내용은 kadmin(1M) 및 gkadmin(1M) 매뉴얼 페이지를 참조하십시오.

예 5-8 Kerberos 정책 목록 보기

이 예에서는 list_policies 하위 명령을 사용하여 *user*와 일치하는 모든 정책을 나열합니다. 인수를 지정하지 않을 경우 list_policies는 Kerberos 데이터베이스에 정의된 모든 정책을 나열합니다.

```
# kadmin
kadmin: list_policies *user*
testuser
financeuser
kadmin: quit
```

예 5-9 Kerberos 정책의 속성 보기

이 예에서는 `get_policy` 하위 명령을 사용하여 `financeuser` 정책의 속성을 봅니다.

```
# /usr/sbin/kadmin.local
kadmin.local: get_policy financeuser
Policy: financeuser
Maximum password life: 13050000
Minimum password life: 10886400
Minimum password length: 8
Minimum number of password character classes: 2
Number of old keys kept: 3
Reference count: 8
Maximum password failures before lockout: 5
Password failure count reset interval: 200
Password lockout duration: 300
kadmin: quit
```

참조 수는 이 정책이 지정된 주체 수를 나타냅니다.

예 5-10 새 Kerberos 암호 정책 만들기

이 예에서는 `add_policy` 하위 명령을 사용하여 `build11` 정책을 만듭니다. 이 정책을 사용하려면 암호에 적어도 3개의 문자 클래스가 필요합니다.

```
# kadmin
kadmin: add_policy -minclasses 3 build11
kadmin: quit
```

예 5-11 Kerberos 계정 잠금 정책 처리

이 예에서는 `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
```

15분이 되기 전에 잠금을 해제하려면 관리 작업을 수행해야 합니다.

```
# /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
kadmin: modify_principal -unlock principal
```

예 5-12 Kerberos 정책 수정

이 예에서는 `modify_policy` 하위 명령을 사용하여 `build11` 정책에 대해 최소 암호 길이를 8자로 변경합니다.

```
# kadmin
kadmin: modify_policy -minlength 8 build11
kadmin: quit
```

예 5-13 Kerberos 정책 삭제

이 예에서는 `delete_policy` 하위 명령을 사용하여 `build11` 정책을 삭제합니다.

1. 관리자가 정책을 사용하는 모든 주체에서 정책을 제거합니다.

   ```
   # kadmin
   kadmin: modify_principal -policy build11 *admin*
   ```

2. 그런 다음 정책을 삭제합니다.

   ```
   kadmin: delete_policy build11
   Are you sure you want to delete the policy "build11"? (yes/no): yes
   kadmin: quit
   ```

정책이 주체에 지정되어 있으면 `delete_policy` 명령이 실패합니다.

예 5-14 `gkadmin` GUI를 사용하여 Kerberos 정책 복제

`gkadmin` GUI에서 Duplicate(복제) 버튼을 눌러 선택한 정책을 복제할 수 있습니다. Policy Name(정책 이름) 필드에서 새 정책의 이름을 지정합니다. 복제한 정책 속성을 수정할 수도 있습니다. 단계는 "`gkadmin` GUI를 사용하여 Kerberos 주체 복제" [146]의 단계와 비슷합니다.

Keytab 파일 관리

서비스를 제공하는 모든 호스트에는 keytab file(keytab 파일)("키 테이블"의 줄임말)이라는 로컬 파일이 있습니다. keytab에는 서비스 키라고 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 주체에 대한 서비스 키를 포함하는 keytab 파일이 있어야 합니다.

서비스 키를 keytab 파일에 추가하려면 `kadmin` 프로세스의 `ktadd` 명령을 사용하여 적절한 서비스 주체를 호스트의 keytab 파일에 추가합니다. 서비스 주체를 keytab 파일에 추가하는 것이기 때문에 주체가 이미 Kerberos 데이터베이스에 있어야 합니다. Kerberos화된 서비스를 제공하는 애플리케이션 서버의 경우 keytab 파일은 기본적으로 `/etc/krb5/krb5.keytab`입니다.

keytab은 사용자 암호와 비슷합니다. 사용자가 자신의 암호를 보호해야 하듯이, 애플리케이션 서버도 해당 keytab 파일을 보호해야 합니다. 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 있을 수 있습니다. Kerberos 클라이언트의 사용자가 Kerberos화된 NFS 파일 시스템을 마운트하려는데 이때 root와 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 root로 사용하여 클라이언트의 root 주체에 대한 자격 증명을 얻어야 합니다.

주의 - root로 NFS 서버를 마운트하는 작업은 보안 위험이 따릅니다.

ktutil 명령을 사용하여 keytab 파일을 관리할 수도 있습니다. 이 대화식 명령은 kadmin과 마찬가지로 Kerberos 데이터베이스와 상호 작용하지 않기 때문에 이 명령을 사용하면 Kerberos 관리 권한 없이도 로컬 호스트의 keytab 파일을 관리할 수 있습니다. 주체가 keytab 파일에 추가되면 ktutil을 사용하여 keytab 파일의 키 목록을 확인하거나 일시적으로 서비스에 대한 인증을 사용 안함으로 설정할 수 있습니다.

참고 - kadmin의 ktadd 명령을 사용하여 keytab 파일에서 주체를 변경하면 새 키가 생성되어 keytab 파일에 추가됩니다.

Keytab 파일에 Kerberos 서비스 주체 추가

주체가 Kerberos 데이터베이스에 있는지 확인한 후 주체를 keytab 파일에 추가할 수 있습니다. 자세한 내용은 "Kerberos 주체 및 해당 속성 보기" [143]을 참조하십시오.

주체를 해당 keytab 파일에 추가해야 하는 호스트에서 kadmin 프로세스의 ktadd 명령을 실행합니다. 자세한 내용은 kadmin(1M) 매뉴얼 페이지를 참조하십시오.

```
# /usr/sbin/kadmin
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.keytab이 사용됩니다.
-q	간단한 정보를 표시합니다.
principal	keytab 파일에 추가될 주체를 지정합니다. host, root, nfs 및 ftp를 서비스 주체로 추가할 수 있습니다.

-glob *principal-exp* 주체 표현식을 지정합니다. *principal-exp*와 일치하는 모든 주체가 keytab 파일에 추가됩니다. 주체 표현식 규칙은 kadmin의 list_principals 명령에 대한 규칙과 같습니다. 사용할 수 있는 표현식에 대한 자세한 내용은 kadmin(1M) 매뉴얼 페이지에서 expression에 대한 정의를 검토하십시오.

예 5-15 Keytab 파일에 서비스 주체 추가

이 예에서는 KDC가 denver의 네트워크 서비스를 인증할 수 있도록 denver의 host 주체가 denver의 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.
kadmin: quit
```

Keytab 파일에서 서비스 주체 제거

keytab 파일에서 주체를 제거할 수 있습니다. 주체를 keytab 파일에서 제거해야 하는 호스트에서 먼저 주체 목록을 확인합니다. "Keytab 파일의 주체 표시" [152]를 참조하십시오.

그런 다음 kadmin 프로세스의 ktadd 명령을 실행합니다. 자세한 내용은 kadmin(1M) 매뉴얼 페이지를 참조하십시오.

```
# /usr/sbin/kadmin
kadmin: ktremove [-k keytab] [-q] principal [kvno | all | old ]
```

-k *keytab* keytab 파일을 지정합니다. 기본적으로 /etc/krb5/krb5.keytab이 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.

예 5-16 Keytab 파일에서 서비스 주체 제거

이 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 파일의 주체 표시

keytab 파일이 있는 호스트에서 `ktutil` 명령을 실행하고 keytab을 읽은 다음 주체를 나열합니다. 주체를 키 목록이라고도 합니다.

참고 - 다른 사용자 소유의 keytab 파일을 만들 수는 있지만, keytab 파일의 기본 위치를 사용하려면 `root` 소유권이 필요합니다.

예 5-17 Keytab 파일에 키 목록(주체) 표시

다음 예에서는 `denver` 호스트의 `/etc/krb5/krb5.keytab` 파일에 있는 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 Kerberos 서비스를 사용 안함으로 설정

때때로 네트워크 애플리케이션 서버에서 일시적으로 `ssh` 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.

참고 - 기본적으로 대부분의 서비스는 인증이 필요합니다. 서비스에 인증이 필요하지 않은 경우에는 암호를 사용 안함으로 설정해도 소용이 없습니다. 서비스는 계속 사용할 수 있습니다.

▼ 호스트에서 일시적으로 Kerberos 서비스에 대한 인증을 사용 안함으로 설정하는 방법

`ktutil` 명령을 사용하면 `kadmin` 권한이 없는 사용자가 서비스를 사용 안함으로 설정할 수 있습니다. 사용자가 서비스를 복원할 수도 있습니다. 자세한 내용은 `ktutil`(1) 매뉴얼 페이지를 참조하십시오.

시작하기 전에 `root` 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **현재 keytab 파일을 임시 파일에 저장합니다.**
   9단계에서 이 임시 파일을 사용하여 인증을 다시 사용으로 설정합니다.

2. **keytab 파일이 있는 호스트에서 `ktutil` 명령을 시작합니다.**

   참고 - 다른 사용자 소유의 keytab 파일을 만들 수는 있지만, keytab 파일의 기본 위치를 사용하려면 `root` 소유권이 필요합니다.

   ```
   # /usr/bin/ktutil
   ```

3. **keytab 파일을 키 목록 버퍼로 읽어 들입니다.**

   ```
   ktutil: read_kt keytab
   ```

4. **키 목록 버퍼를 표시합니다.**

   ```
   ktutil: list
   ```

   현재 키 목록 버퍼가 표시됩니다. 사용 안함으로 설정하려는 서비스에 대한 슬롯 번호를 메모해 둡니다.

5. **특정 서비스 주체를 키 목록 버퍼에서 제거하여 호스트 서비스를 일시적으로 사용 안함으로 설정합니다.**

   ```
   ktutil: delete_entry slot-number
   ```

   여기서 *slot-number*는 `list` 출력에서 삭제될 서비스 주체의 슬롯 번호를 지정합니다.

6. **키 목록 버퍼를 새 keytab 파일에 씁니다.**

   ```
   ktutil: write_kt new-keytab
   ```

7. **`ktutil` 명령을 종료합니다.**

   ```
   ktutil: quit
   ```

8. **새 keytab 파일을 사용하여 주체의 인증을 사용 안함으로 설정합니다.**

```
# mv new-keytab keytab
```

9. **(옵션) 서비스를 다시 사용으로 설정하려는 경우 임시 keytab 파일을 다시 원본 위치로 복사합니다.**

```
# cp original-keytab keytab
```

예 5-18 Kerberos 호스트를 일시적으로 사용 안함으로 설정

이 예에서는 denver 호스트의 host 서비스가 일시적으로 사용 안함으로 설정됩니다. denver의 호스트 서비스를 다시 사용으로 설정하기 위해 관리자는 저장된 keytab 파일을 원본 위치로 복사합니다.

```
denver # cp /etc/krb5/krb5.keytab /etc/krb5/krb5.keytab.save
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/nodenverhost.krb5.keytab
ktutil: quit
denver # cp /etc/krb5/nodenverhost.krb5.keytab /etc/krb5/krb5.keytab
```

사용자가 저장된 파일을 다시 원본 위치로 복사할 때까지 호스트를 사용할 수 없습니다.

```
denver # cp /etc/krb5/krb5.keytab.save /etc/krb5/krb5.keytab
```

♦ ♦ ♦ 6 장

6 Kerberos 응용 프로그램 사용

이 장에서는 Oracle Solaris에서 제공하는 Kerberos 기반 명령 및 서비스를 사용하는 방법에 대해 설명합니다. 이 장의 내용을 읽기 전에 명령의 원래 버전에 대해 잘 알고 있어야 합니다.

이 장은 응용 프로그램 사용자를 대상으로 하기 때문에 티켓 획득, 확인 및 삭제 등 티켓에 대한 내용을 다룹니다. 또한 Kerberos 암호 선택 또는 변경에 대한 내용도 다룹니다.

이 장에서는 다음 내용을 다룹니다.

Kerberos에 대한 개요는 2장. Kerberos 서비스 정보를 참조하십시오.

Kerberos 티켓 관리

이 절에서는 티켓 획득, 확인 및 삭제 방법에 대해 설명합니다. 티켓에 대한 개요는 "Kerberos 서비스의 작동 방식" [40]을 참조하십시오.

Oracle Solaris 릴리스에서 Kerberos는 `login` 명령에 내장됩니다. 하지만 티켓을 자동으로 가져오려면 적용 가능한 로그인 서비스에 대해 PAM 서비스를 구성해야 합니다. 자세한 내용은 `pam_krb5`(5) 매뉴얼 페이지를 참조하십시오.

티켓 복사본을 다른 호스트로 전달하도록 `ssh` 명령을 설정할 수 있습니다. 이렇게 하면 해당 호스트에 액세스하기 위해 명시적으로 티켓을 요청하지 않아도 됩니다. 보안상의 이유로 관리자가 이렇게 할 수 없도록 설정했을 수 있습니다. 자세한 내용은 `ssh`(1) 매뉴얼 페이지에서 에이전트 전달에 대한 설명을 참조하십시오.

티켓 수명에 대한 자세한 내용은 "티켓 수명" [167]을 참조하십시오.

Kerberos 티켓 만들기

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 들어ssh -l을 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 6-1 Kerberos 티켓 만들기

이 예에서는 사용자 kdoe가 자신의 시스템에 티켓을 만드는 방법을 보여 줍니다.

```
% kinit
Password for kdoe@CORP.EXAMPLE.COM: xxxxxxxx
```

이 예에서 사용자 kdoe는 -l 옵션을 사용하여 세 시간 동안 유효한 티켓을 만듭니다.

```
% kinit -l 3h kdoe@EXAMPLE.ORG
Password for kdoe@EXAMPLE.ORG: xxxxxxxx
```

Kerberos 티켓 확인

모든 티켓이 비슷한 것은 아닙니다. 예를 들면 어떤 티켓은 전달 가능 티켓이고 다른 티켓은 후일자 티켓이며 또 다른 티켓은 전달 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 klist에 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

“티켓의 유형” [166]에서는 티켓의 여러 가지 속성에 대해 설명합니다.

예 6-2 Kerberos 티켓 확인

이 예에서는 사용자 kdoe가 보유한 초기 티켓이 전달 가능(F)한 후일자(d) 티켓이지만, 아직 검증되지 않았음(i)을 보여 줍니다.

```
% /usr/bin/klist -f
Ticket cache: /tmp/krb5cc_74287
Default principal: kdoe@EXAMPLE.COM

Valid starting                Expires                Service principal
09 Feb 14 15:09:51  09 Feb 14 21:09:51  nfs/EXAMPLE.COM@EXAMPLE.COM
renew until 10 Feb 14 15:12:51, Flags: Fdi
```

다음 예에서는 사용자 kdoe가 보유한 두 티켓이 다른 호스트에서 사용자의 호스트로 전달되었음(f)을 보여 줍니다. 이 티켓은 전달 가능(F) 티켓이기도 합니다.

```
% klist -f
Ticket cache: /tmp/krb5cc_74287
Default principal: kdoe@EXAMPLE.COM

Valid starting                Expires                Service principal
07 Feb 14 06:09:51  09 Feb 14 23:33:51  host/EXAMPLE.COM@EXAMPLE.COM
renew until 10 Feb 14 17:09:51, Flags: fF

Valid starting                Expires                Service principal
08 Feb 14 08:09:51  09 Feb 14 12:54:51  nfs/EXAMPLE.COM@EXAMPLE.COM
renew until 10 Feb 14 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: kdoe@EXAMPLE.COM

Valid starting                Expires                Service principal
```

```
07 Feb 14 06:09:51  09 Feb 14 23:33:51  krbtgt/EXAMPLE.COM@EXAMPLE.COM
renew until 10 Feb 14 17:09:51, Flags: FRIA
Etype(skey, tkt): AES-256 CTS mode with 96-bit SHA-1 HMAC
Addresses: client.example.com
```

Kerberos 티켓 삭제

현재 세션 중에 획득한 모든 Kerberos 티켓을 삭제하려면 kdestroy 명령을 사용하십시오. 이 명령은 자격 증명 캐시를 삭제하여 모든 자격 증명과 티켓을 삭제합니다. 이 삭제 작업은 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.

자리를 비워야 할 경우 kdestroy 명령을 사용하거나 화면 보호기로 화면을 잠가야 합니다.

Kerberos 암호 관리

Kerberos 서비스가 구성된 경우 일반 Oracle Solaris 암호와 Kerberos 암호, 두 개의 암호가 생깁니다. 두 암호를 동일하게 설정하거나 다르게 설정할 수 있습니다.

암호 변경

PAM이 제대로 구성된 경우 Kerberos 암호를 두 가지 방법으로 변경할 수 있습니다.

- passwd 명령을 사용합니다. Kerberos 서비스가 구성된 경우 passwd 명령이 자동으로 새 Kerberos 암호를 묻는 메시지를 표시합니다.

  passwd를 사용하면 UNIX 암호와 Kerberos 암호를 동시에 설정할 수 있습니다. 하지만 passwd로 한 암호만 변경할 수 있으며 나머지 암호는 그대로 유지됩니다.

참고 - 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 명령을 사용합니다. kpasswd는 Kerberos 암호만 변경합니다. UNIX 암호를 변경하려는 경우에는 passwd를 사용해야 합니다.

kpasswd는 주로 유효한 UNIX 사용자가 아닌 Kerberos 주체에 대한 암호를 변경하는 데 사용됩니다. 예를 들어 jdoe/admin은 Kerberos 주체이기는 하지만 실제 UNIX 사용자는 아니므로, 암호를 변경하려면 kpasswd를 사용해야 합니다.

암호를 변경한 후 네트워크 전체에 암호를 전파해야 합니다. Kerberos 네트워크의 크기에 따라 전파하는 데 걸리는 시간은 몇 분에서 몇 시간이 될 수도 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.

Kerberos 정책은 암호에 대한 기준을 정의합니다. 각 사용자에 대해 정책을 설정할 수도 있고 기본 정책을 적용할 수도 있습니다. 정책에 대한 자세한 내용은 "Kerberos 정책 관리" [147]를 참조하십시오. Kerberos 암호에 대해 설정할 수 있는 기준을 보여 주는 예는 예 5-9. "Kerberos 정책의 속성 보기"를 참조하십시오. 암호 문자 클래스는 소문자, 대문자, 숫자, 문장 부호를 비롯한 기타 모든 문자입니다.

Kerberos의 원격 로그인

Oracle Solaris와 같이 Kerberos는 원격 액세스를 위해 ssh 명령을 제공합니다. ssh 명령은 설치 후 네트워크 요청을 수락하는 유일한 네트워크 서비스입니다. 따라서 Kerberos에 맞게 수정된 다른 네트워크 서비스(예: rlogin 및 telnet)는 사용 안함으로 설정됩니다. 자세한 내용은 "Kerberos 네트워크 프로그램" [45] 및 "Kerberos 사용자 명령" [159]을 참조하십시오.

Kerberos 사용자 명령

Kerberos V5 제품은 *Single Sign-On* 시스템이므로, 네트워크 응용 프로그램을 사용할 때 자신의 암호를 한 번만 입력하면 됩니다. Kerveros V5 응용 프로그램은 인증 및 선택적으로 암호화를 자동으로 수행하는데, 기존의 친숙한 네트워크 응용 프로그램 각각에 Kerberos가 내장되어 있기 때문입니다. Kerberos V5 응용 프로그램은 기존 UNIX 네트워크 응용 프로그램에 Kerberos 기능이 추가된 버전입니다.

참고 - 일반적으로 ssh 응용 프로그램만으로도 원격 로그인 요구 사항이 충족되어야 합니다. ssh(1) 매뉴얼 페이지를 참조하십시오. 제거된 네트워크 응용 프로그램을 사용으로 설정하는 방법에 대한 자세한 내용은 "Kerberos 네트워크 프로그램" [45]을 참조하십시오.

기존 네트워크 응용 프로그램의 Kerberos 기능에 대해서는 다음 매뉴얼 페이지와 OPTIONS 절을 참조하십시오.

- ftp(1)
- rcp(1)
- rlogin(1)

- rsh(1)
- telnet(1)

Kerberos화된 응용 프로그램을 사용하여 원격 시스템에 연결할 경우, 응용 프로그램, KDC 및 원격 시스템은 일련의 신속한 협상을 수행합니다. 이러한 협상이 완료되고 응용 프로그램이 사용자를 대신해 원격 시스템에 대해 사용자의 ID를 검증하면 원격 호스트는 사용자에게 액세스 권한을 부여합니다.

더 이상 사용되지 않는 원격 로그인 명령을 사용하는 예를 보려면 *Oracle Solaris 11.1 Administration: Security Services*의 "Kerberos 응용 프로그램 사용(작업)"을 참조하십시오.

♦ ♦ ♦ 7 장

7 Kerberos 서비스 참조

이 장에서는 Kerberos 제품에 포함된 여러 가지 파일, 명령 및 데몬에 대해 설명합니다.
이 장에는 다음 참조 정보가 포함되어 있습니다.

Kerberos 파일

다음은 Kerberos 서비스에서 사용하는 파일입니다.

`~/.gkadmin`	Kerberos 관리 GUI에서 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.
`/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 애플리케이션 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 모든 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 slave에 대한 전체 업데이트를 수행할 때 생성되는 임시 덤프 파일입니다.
`/var/user/`*$USER*`/krb-warn.conf`	사용자별 티켓 만료 경고 및 자동 갱신 구성 파일

Kerberos 명령

다음은 Kerberos 제품에 포함된 명령입니다. 명령은 매뉴얼 페이지별로 나열되어 있습니다.

ftp(1)	FTP(File Transfer Protocol) 응용 프로그램입니다.
kdestroy(1)	Kerberos 티켓을 삭제합니다.
kinit(1)	Kerberos TGT(티켓 부여 티켓)를 얻어 캐시에 저장합니다.
klist(1)	현재 Kerberos 티켓을 표시합니다.
kpasswd(1)	Kerberos 암호를 변경합니다.
ktutil(1)	Kerberos Keytab 파일을 관리합니다.
kvno(1)	Kerberos 주체의 키 버전 번호를 나열합니다.
rcp(1)	원격 파일 복사 응용 프로그램입니다.
rlogin(1)	원격 로그인 프로그램입니다.
rsh(1)	원격 셸 응용 프로그램입니다.
scp(1)	보안 원격 파일 복사 응용 프로그램입니다.
sftp(1)	보안 파일 전송 응용 프로그램입니다.
ssh(1)	원격 로그인을 위한 보안 셸입니다.
telnet(1)	Kerberos화된 `telnet` 응용 프로그램입니다.
kprop(1M)	Kerberos 데이터베이스 전파 프로그램입니다.
gkadmin(1M)	주체 및 정책을 관리하는 데 사용되는 Kerberos 데이터베이스 관리 GUI 프로그램입니다.
gsscred(1M)	`gsscred` 테이블 항목을 관리합니다.
kadmin(1M)	주체, 정책 및 Keytab 파일을 관리하는 데 사용되는 원격 Kerberos 데이터베이스 관리 프로그램입니다(Kerberos 인증이 필요함).
kadmin.local(1M)	주체, 정책 및 keytab 파일을 관리하는 데 사용되는 로컬 Kerberos 데이터베이스 관리 프로그램으로, 마스터 KDC에서 실행됩니다.
kclient(1M)	설치 프로파일과 함께 사용되거나 설치 프로파일 없이 사용되는 Kerberos 클라이언트 설치 스크립트입니다.
kdb5_ldap_util(1M)	Kerberos 데이터베이스용 LDAP 컨테이너를 만듭니다.

kdb5_util(1M)	Kerberos 데이터베이스 및 stash 파일을 만듭니다.
kdcmgr(1M)	Kerberos 마스터 및 슬레이브 KDC를 구성합니다.
kproplog(1M)	업데이트 로그에 업데이트 항목에 대한 요약 정보를 나열합니다.

Kerberos 데몬

다음은 Kerberos 제품에서 사용하는 데몬입니다.

/usr/lib/inet/proftpd	보안 파일 전송 프로토콜 데몬입니다.
/usr/lib/ssh/sshd	보안 셸 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

다음은 Kerberos 설명서 전체에서 사용되는 Kerberos 용어입니다. 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 파일에는 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 서비스라는 용어는 대개 서로 바꿔서 사용됩니다. 명확히 하자면, 서버라는 용어는 Kerberos 소프트웨어가 실행 중인 물리적 시스템을 정의하는 데 사용됩니다. 서비스는 서버에서 지원되는 특정 기능(예: `ftp` 또는 `nfs`)에 해당합니다. 설명서에서는 서버를 서비스의 일부로 설명하는 경우가 많은데, 이는 이러한 용어의 의미를 흐리게 합니다. 따라서 서버는 물리적 시스템을 나타내고, 서비스라는 용어는 소프트웨어를 나타냅니다.

Kerberos 제품은 두 가지 유형의 키를 사용합니다. 암호에서 파생된 키의 한 유형으로, 각 사용자 주체에게 제공되며 해당 사용자와 KDC만 알 수 있습니다. 다른 유형의 키는 암호와 관련이 없는 모든 키입니다. 따라서 사용자 주체가 사용하기에 적합하지 않습니다. 모든 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 모든 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 자격 증명에 포함되어 있는 티켓을 발행합니다. 티켓은 발행된 후 만료될 때까지 재사용할 수 있습니다.

자격 증명은 티켓 및 일치하는 세션 키를 포함하는 정보 패킷으로, 자격 증명은 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 대개 서비스 액세스를 요청할 때 한번만 사용됩니다. 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. “Kerberos 티켓 확인” [156]을 참조하십시오.

티켓은 다음 용어 중 하나 이상으로 설명할 수 있습니다.

Forwardable/ forwarded
: 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, `david`는 새 티켓 없이도 자신의 시스템에 로그인할 수 있으므로 다시 인증할 필요가 없습니다. 전달 가능 티켓에 대한 예는 예 6-1. “Kerberos 티켓 만들기”을 참조하십시오.

초기
: 초기 티켓이란 TGT(티켓 부여 티켓)를 기반으로 하지 않고 직접 발행되는 티켓입니다. 암호가 달라지는 응용 프로그램과 같은 일부 서비스의 경우 티켓을 "초기"로 표시해야 클라이언트에 보안 키가 있음을 입증할 수 있습니다. 초기 티켓은 클라이언트가 더 오래되었을 수 있는 TGT(티켓 부여 티켓)에 의존하지 않고 최근에 인증되었음을 나타냅니다.

잘못됨
: 잘못된 티켓은 아직 사용 가능하지 않은 후일자 티켓으로, 검증될 때까지 애플리케이션 서버에서 거부됩니다. 티켓을 검증하려면 티켓 시

작 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 Solaris에서는 프록시 가능 또는 프록시 티켓을 지원하지 않습니다.

  프록시 가능 티켓은 전달 가능 티켓과 비슷하지만, 단일 서비스에 대해서만 유효한 반면 전달 가능 티켓은 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 첫 번째 만료가 발생하기 전에 티켓을 갱신합니다. 예를 들어 모든 티켓의 최대 수명이 10시간일 경우 한 티켓은 한 시간 동안 유효할 수 있다고 가정해 보십시오. 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 방법에 대한 자세한 내용은 “Kerberos 티켓 확인” [156]을 참조하십시오.

티켓 수명

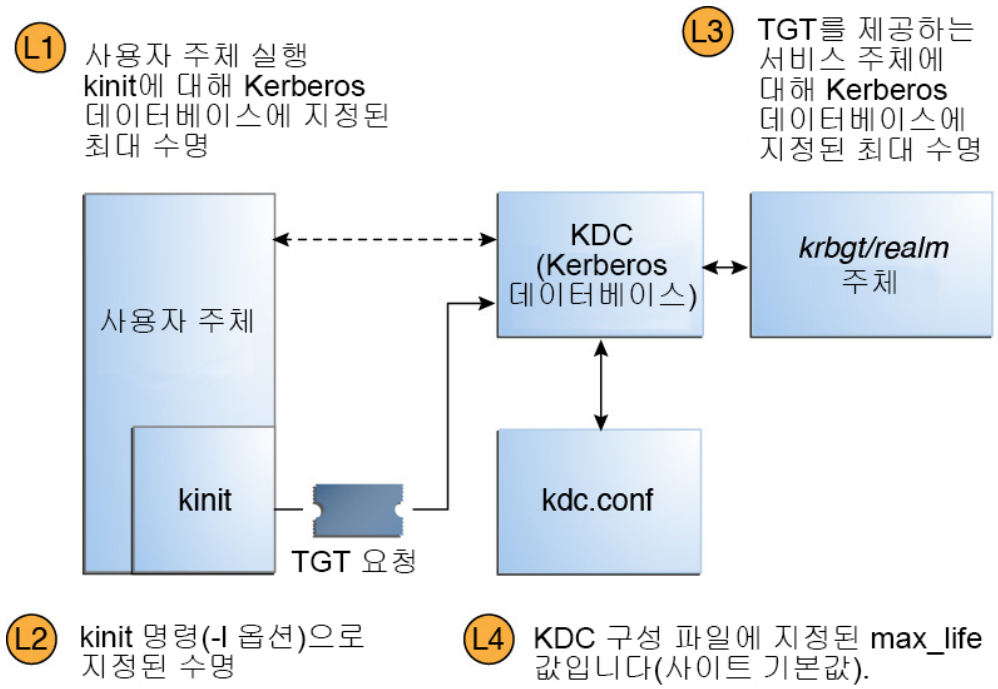
주체가 TGT(티켓 부여 티켓)를 포함한 티켓을 얻으면 티켓의 수명은 다음 수명 값 중 가장 작은 값으로 설정됩니다.

- `kinit`의 `-l` 옵션에 지정된 수명 값(`kinit`를 사용하여 티켓을 얻은 경우). 기본적으로 `kinit`는 최대 수명 값을 사용합니다.
- `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

다음 그림은 TGT의 수명이 결정되는 방식과 네 가지 수명 값이 비롯되는 위치를 보여 줍니다. 주체가 티켓을 얻으면 비슷한 방식으로 티켓의 수명이 결정됩니다. 단, `kinit`가 수명 값을 제공하지 않고, 티켓을 제공하는 서비스 주체가 `krbtgt/`*realm* 주체 대신 최대 수명 값을 제공한다는 두 가지 점만 다릅니다.

그림 7-1 TGT의 수명이 결정되는 방식



갱신 가능 티켓의 수명도 다음과 같이 네 갱신 가능 값 중 가장 작은 값에서 결정됩니다.

- `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. 다음 예에서는 일반 주체 이름을 보여 줍니다.

`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.

♦ ♦ ♦ 8 장

8 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 장에서는 다음 내용을 다룹니다.

Kerberos 오류 메시지

이 절에서는 각 오류가 발생하는 이유와 해결 방법을 비롯하여 Kerberos 오류 메시지에 대한 정보를 제공합니다.

gkadmin GUI 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 * failed
```

원인: `gkadmin` GUI에서 사용하는 Java Native Interfac에 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.

`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 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.

`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. `root`는 이 파일을 쓸 수 있어야 하며 그 외의 다른 사용자는 읽을 수 있어야 합니다.

`Client '`*principal*`' pre-authentication failed`

원인: 주체에 대한 사전 인증이 실패했습니다.

해결책: 사용자가 올바른 암호를 사용하고 있는지 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.

`Communication failure with server while initializing kadmin interface`

원인: 마스터 KDC에 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` 명령을 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.

`failed to obtain credentials cache`

원인: `kadmin` 초기화 중 `kadmin`이 `admin` 주체에 대한 자격 증명을 얻으려고 시도했는데 오류가 발생했습니다.

해결책: `kadmin` 명령을 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.

`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 절에 대한 쌍에 대괄호가 있는지도 확인하십시오.

`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` 명령을 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.

`kadmin: Bad encryption type while changing host/`*FQDN*`'s key`

원인: 새 릴리스에서는 기본 릴리스에 더 많은 기본 암호화 유형이 포함됩니다. 이전 버전의 소프트웨어에서 실행 중인 KDC에서는 지원되지 않는 암호화 유형을 클라이언트가 요청할 수 있습니다.

해결책: 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 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 암호를 지정하여 Kerberos 인증을 완료해야 합니다. 사용자가 자신의 원래 암호를 잊어버렸을 수 있습니다.

`Key version` *number* `is not available for principal` *principal*

원인: 키 버전이 애플리케이션 서버의 키 버전과 일치하지 않습니다.

해결책: `klist -k` 명령을 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 `login`에 대한 PAM 구성 파일에 `pam_krb5.so.1`의 올바른 경로가 포함되어 있는지 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 체크섬이 일치하지 않습니다. 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, 티켓을 삭제하고 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. 읽을 수 없는 경우에는 `kinit` 명령을 다시 실행하십시오.

`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` 주체에 `kadm5.acl` 파일에 있는 적절한 권한이 지정되지 않았습니다.

해결책: 적합한 권한이 있는 주체를 사용하십시오. 또는 사용 중인 주체가 적절한 권한을 받도록 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.

`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.

`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 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.

`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 do not match.`

원인: 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: kprop가 아닌 다른 응용 프로그램을 실행할 때 이 오류가 발생하는 경우 서버의 Keytab 파일이 올바른지 확인하십시오.

```
Truncated input file detected
```

원인: 작업에 사용된 데이터베이스 덤프 파일이 완전한 덤프 파일이 아닙니다.

해결책: 덤프 파일을 다시 만들거나, 다른 데이터베이스 덤프 파일을 사용하십시오.

Kerberos 문제 해결

이 절에서는 Kerberos 소프트웨어에 대한 문제 해결 정보를 제공합니다.

키 버전 번호 관련 문제

KDC에서 사용하는 키 버전 번호(KVNO)와 시스템에 호스트된 서비스의 /etc/krb5/krb5.keytab에 저장된 서비스 주체 키가 일치하지 않는 경우가 있습니다. keytab 파일을 새 키로 업데이트하지 않은 상태에서 KDC에 새 키 세트가 생성된 경우 KVNO가 동기화되지 않을 수 있습니다. 문제를 진단한 후 krb5.keytab 파일을 새로 고칩니다.

1. keytab 항목을 나열합니다.
   각 항목의 첫번째 항목은 각 주체의 KVNO입니다.

   ```
   # 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
```

형식이 올바르지 않을 경우 연관된 서비스가 공격 대상이 될 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 -x를 실행하십시오.

KDC에 기본 보안이 구성된 경우 rlogin 명령은 사용 안함으로 설정되므로 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 설정하십시오.

```
# svcadm disable svc:/network/login:eklogin
```

원격 액세스가 작동하지 않는 경우, KDC의 keytab 파일 때문에 문제가 발생했을 수 있습니다. 원격 액세스가 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 invalid argument 또는 bad directory 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 있습

니다. 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.

root 사용자로 인증 관련 문제

시스템에서 root가 되려고 시도하면서 호스트의 keytab 파일에 root 주체를 이미 추가했는데 인증에 실패할 경우, 두 영역을 검토해야 합니다. 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
```

그런 다음 gssd 서비스가 /etc/gss/gsscred.conf 파일을 읽도록 설정합니다.

```
# pkill -HUP gssd
```

이제 gssd가 자격 증명 매핑을 요청하므로 자격 증명 매핑을 모니터링할 수 있습니다. syslog.conf 파일이 debug 보안 레벨을 사용하여 auth 시스템 기능에 대해 구성된 경우, 매핑이 syslog 데몬에 의해 기록됩니다.

참고 - rsyslog 서비스 인스턴스가 사용으로 설정되어 있으면 매핑이 rsyslog 데몬에 의해 기록됩니다.

---

[1] Ken Hornstein, “Kerberos FAQ,” [http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html#kerbdns], accessed 10 March 2010.

Kerberos 서비스에서 DTrace 사용

Kerberos 방식은 다양한 프로토콜 메시지를 디코딩하기 위해 여러 가지 DTrace 프로브를 지원합니다. 목록은 부록 A. Kerberos용 DTrace 프로브를 참조하십시오. DTrace 프로브는 권한 있는 사용자가 암호화되지 않은 Kerberos 및 응용 프로그램 데이터를 쉽게 찾아볼 수 있도록 허용함으로써 다른 프로토콜 검사자보다 확실한 장점을 제공합니다.

다음 예에서는 Kerberos DTrace 프로브를 사용하여 볼 수 있는 정보를 보여 줍니다.

예 8-1 DTrace를 사용하여 Kerberos 메시지 추적

다음 스크립트는 DTrace 프로브를 사용하여 시스템에서 전송하고 수신한 Kerberos 메시지에 대한 세부 정보를 표시합니다. 표시되는 필드는 `krb_message-recv` 및 `krb_message-send` 프로브에 지정된 구조에 따라 달라집니다. 자세한 내용은 "Kerberos DTrace 프로브의 정의" [201]를 참조하십시오.

```
kerberos$target:::krb_message-recv
{
    printf("<- krb message recved: %s\n", args[0]->krb_message_type);
    printf("<- krb message remote addr: %s\n", args[1]->kconn_remote);
    printf("<- krb message ports: local %d remote %d\n",
            args[1]->kconn_localport, args[1]->kconn_remoteport);
    printf("<- krb message protocol: %s transport: %s\n",
            args[1]->kconn_protocol, args[1]->kconn_type);
}

kerberos$target:::krb_message-send
{
    printf("-> krb message sent: %s\n", args[0]->krb_message_type);
    printf("-> krb message remote addr: %s\n", args[1]->kconn_remote);
    printf("-> krb message ports: local %d remote %d\n",
            args[1]->kconn_localport, args[1]->kconn_remoteport);
    printf("-> krb message protocol: %s transport: %s\n",
            args[1]->kconn_protocol, args[1]->kconn_type);
    printf("\n");
}

kerberos$target:::krb_error-read
{
    printf("<- krb error code: %s\n", args[1]->kerror_error_code);
    printf("<- krb error client: %s server: %s\n", args[1]->kerror_client,
            args[1]->kerror_server);
    printf("<- krb error e-text: %s\n", args[1]->kerror_e_text);
    printf("\n");
}
```

위의 스크립트를 명령줄에서 호출하거나 `krb5kdc` 데몬을 사용하여 호출할 수 있습니다. 다음은 명령줄에서 `krb-dtrace.d`라는 스크립트를 호출하는 방법을 보여 주는 예입니다. 이 명령은 Kerberos가 메시지를 전송 및 수신하도록 합니다. `LD_NOLAZYLOAD=1`은 Kerberos DTrace 프로브가 포함된 Kerberos `mech_krb5.so` 라이브러리를 로드하는 데 필요합니다.

```
# LD_NOLAZYLOAD=1 dtrace -s ./krb\-dtrace.d -c kinit
dtrace: script './krb-dtrace' matched 4 probes
kinit: Client 'root@DEV.ORACLE.COM' not found in Kerberos database while g
etting initial credentials
dtrace: pid 3750 has exited
 CPU     ID                    FUNCTION:NAME
   2  74782 k5_trace_message_send:krb_message-send -> krb message sent: KR
B_AS_REQ(10)
-> krb message remote addr: 10.229.168.163
-> krb message ports: local 62029 remote 88
-> krb message protocol: ipv4 transport: udp


   2  74781 k5_trace_message_recv:krb_message-recv <- krb message recved:
KRB_ERROR(30)
<- krb message remote addr: 10.229.168.163
<- krb message ports: local 62029 remote 88
<- krb message protocol: ipv4 transport: udp

   2  74776     krb5_rd_error:krb_error-read <- krb error code: KDC_ERR_C_
PRINCIPAL_UNKNOWN(6)
<- krb error client: root@DEV.ORACLE.COM server: krbtgt/DEV.ORACLE.COM@DEV
.ORACLE.COM
<- krb error e-text: CLIENT_NOT_FOUND
```

krb5kdc 데몬에 스크립트를 사용하려면 svc:/network/security/krb5kdc:default 서비스가 사용으로 설정되고 온라인 상태여야 합니다. 다음 명령은 LD_NOLAZYLOAD=1을 사용하지 않는데 mech_krb5.so 라이브러리가 krb5kdc 데몬을 로드하기 때문입니다.

```
# dtrace -s ./krb\-dtrace.d -p $(pgrep -x krb5kdc)
```

예 8-2 DTrace를 사용하여 Kerberos 사전 인증 유형 보기

다음 예에서는 클라이언트가 선택한 사전 인증을 보여줍니다. 첫번째 단계는 다음과 같은 DTrace 스크립트를 만드는 것입니다.

```
cat krbtrace.d
kerberos$target:::krb_message-recv
{
printf("<- krb message recved: %s\n", args[0]->krb_message_type);
printf("<- krb message remote addr: %s\n", args[1]->kconn_remote);
printf("<- krb message ports: local %d remote %d\n",
args[1]->kconn_localport, args[1]->kconn_remoteport);
printf("<- krb message protocol: %s transport: %s\n",
args[1]->kconn_protocol, args[1]->kconn_type);
}

kerberos$target:::krb_message-send
{
printf("-> krb message sent: %s\n", args[0]->krb_message_type);
printf("-> krb message remote addr: %s\n", args[1]->kconn_remote);
printf("-> krb message ports: local %d remote %d\n",
args[1]->kconn_localport, args[1]->kconn_remoteport);
```

```
printf("-> krb message protocol: %s transport: %s\n",
args[1]->kconn_protocol, args[1]->kconn_type);
printf("\n");
}

kerberos$target:::krb_kdc_req-make
{
printf("-> krb kdc_req make msg type: %s\n", args[0]->krb_message_type);
printf("-> krb kdc_req make pre-auths: %s\n", args[1]->kdcreq_padata_types);
printf("-> krb kdc_req make auth data: %s\n", args[1]->kdcreq_authorization_data);
printf("-> krb kdc_req make client: %s server: %s\n", args[1]->kdcreq_client,
args[1]->kdcreq_server );
}

kerberos$target:::krb_kdc_req-read
{
/* printf("<- krb kdc_req msg type: %s\n", args[0]->krb_message_type); */
printf("<- krb kdc_req client: %s server: %s\n", args[1]->kdcreq_client,
args[1]->kdcreq_server );
printf("\n");
}

kerberos$target:::krb_kdc_rep-read
{
/* printf("<- krb kdc_rep msg type: %s\n", args[0]->krb_message_type); */
printf("<- krb kdc_rep client: %s server: %s\n", args[1]->kdcrep_client,
args[1]->kdcrep_enc_server );
printf("\n");
}

kerberos$target:::krb_ap_req-make
{
printf("-> krb ap_req make server: %s client: %s\n", args[2]->kticket_server,
args[2]->kticket_enc_client );
}

kerberos$target:::krb_error-read
{
printf("<- krb error code: %s\n", args[1]->kerror_error_code);
printf("<- krb error client: %s server: %s\n", args[1]->kerror_client,
args[1]->kerror_server);
printf("<- krb error e-text: %s\n", args[1]->kerror_e_text);
printf("\n");
}
```

그런 다음 다음 명령을 입력하여 Kerberos 시스템에서 권한 있는 사용자로 `krbtrace.d` 스크립트를 실행합니다.

```
# LD_BIND_NOW=1 dtrace -qs krbtrace.d -c "kinit -k"
.
.
-> krb kdc_req make pre-auths: FX_COOKIE(133) ENC_TIMESTAMP(2) REQ_ENC_PA_REP(149)
```

사전 인증 유형이 출력 결과에 표시됩니다. 다양한 사전 인증 유형에 대한 자세한 내용은 RFC 4120을 참조하십시오.

예 8-3 DTrace를 사용하여 Kerberos 오류 메시지 덤프

```
# dtrace -n 'krb_error-make {
printf("\n{");
printf("\n\tctime = %Y", (uint64_t)(args[1]->kerror_ctime * 1000000000));
printf("\n\tcusec = %d", args[1]->kerror_cusec);
printf("\n\tstime = %Y", (uint64_t)(args[1]->kerror_stime * 1000000000));
printf("\n\tsusec = %d", args[1]->kerror_susec);
printf("\n\terror_code = %s", args[1]->kerror_error_code);
printf("\n\tclient = %s", args[1]->kerror_client);
printf("\n\tserver = %s", args[1]->kerror_server);
printf("\n\te_text = %s", args[1]->kerror_e_text);
printf("\n\te_data = %s", "");
printf("\n}");
}'
dtrace: description 'krb_error-make ' matched 1 probe
CPU     ID                    FUNCTION:NAME
0  78307     krb5_mk_error:krb_error-make
{
ctime = 2012 May 10 12:10:20
cusec = 0
stime = 2012 May 10 12:10:20
susec = 319090
error_code = KDC_ERR_C_PRINCIPAL_UNKNOWN(6)
client = testuser@EXAMPLE.COM
server = krbtgt/EXAMPLE.COM@EXAMPLE.COM
e_text = CLIENT_NOT_FOUND
e_data =
}
```

예 8-4 DTrace를 사용하여 SSH 서버에 대한 서비스 티켓 보기

```
# LD_PRELOAD_32=/usr/lib/gss/mech_krb5.so.1 dtrace -q -n '
kerberos$target:::krb_kdc_req-make {
printf("kdcreq_server: %s",args[1]->kdcreq_server);
}' -c "ssh local@four.example.com" -o dtrace.out
Last login: Wed Sep 10 10:10:20 2014
Oracle Solaris 11 X86    July 2014
$ ^D
# cat dtrace.out
kdcreq_server: host/four.example.com@EXAMPLE.COM
```

예 8-5 초기 TGT를 요청할 때 DTrace를 사용하여 사용할 수 없는 KDC의 주소 및 포트 보기

```
# LD_BIND_NOW=1 dtrace -q -n '
kerberos$target:::krb_message-send {
```

```
printf("%s:%d\n",args[1]->kconn_remote, args[1]->kconn_remoteport)
}' -c "kinit local4"

10.10.10.14:88
10.10.10.14:750
10.10.10.14:88
10.10.10.14:750
10.10.10.14:88
10.10.10.14:750
kinit(v5): Cannot contact any KDC for realm 'EXAMPLE.COM'
while getting initial credentials
```

예 8-6 DTrace를 사용하여 Kerberos 주체의 요청 보기

```
# LD_BIND_NOW=1 dtrace -qs /opt/kdebug/mykdtrace.d \
-c 'kadmin -p kdc/admin -w test123 -q listprincs'
Authenticating as principal kdc/admin with password.
krb kdc_req msg type: KRB_AS_REQ(10)
krb kdc_req make client: kdc/admin@TEST.NET server:
kadmin/interop1.example.com@TEST.NET
krb message sent: KRB_AS_REQ(10)
krb message recved: KRB_ERROR(30)
Err code: KDC_ERR_PREAUTH_REQUIRED(25)
Err msg client: kdc/admin@TEST.NET server: kadmin/interop1.example.com@TEST.NET
Err e-text: NEEDED_PREAUTH
krb kdc_req msg type: KRB_AS_REQ(10)
krb kdc_req make client: kdc/admin@TEST.NET server:
kadmin/interop1.example.com@TEST.NET
krb message sent: KRB_AS_REQ(10)
krb message recved: KRB_AS_REP(11)
kadmin: Database error! Required KADM5 principal missing while
initializing kadmin interface
krb kdc_req msg type: KRB_AS_REQ(10)
krb kdc_req make client: kdc/admin@TEST.NET server:
kadmin/interop2.example.com@TEST.NET
krb message sent: KRB_AS_REQ(10)
krb message recved: KRB_ERROR(30)
Err code: KDC_ERR_S_PRINCIPAL_UNKNOWN(7)
Err msg client: kdc/admin@TEST.NET server: kadmin/interop2.example.com@TEST.NET
Err e-text: SERVER_NOT_FOUND
krb kdc_req msg type: KRB_AS_REQ(10)
krb kdc_req make client: kdc/admin@TEST.NET server:
kadmin/interop2.example.com@TEST.NET
```

다음은 위의 출력을 생성하기 위해 사용된 스크립트입니다.

```
kerberos$target:::krb_message-recv
{
printf("krb message recved: %s\n", args[0]->krb_message_type);
}

kerberos$target:::krb_message-send
{
```

```
printf("krb message sent: %s\n", args[0]->krb_message_type);
}

kerberos$target:::krb_kdc_req-make
{
printf("krb kdc_req msg type: %s\n", args[0]->krb_message_type);
printf("krb kdc_req make client: %s server: %s\n", args[1]->kdcreq_client,
args[1]->kdcreq_server );
}

kerberos$target:::krb_ap_req-make
{
printf("krb ap_req make server: %s client: %s\n", args[2]->kticket_server,
args[2]->kticket_enc_client );
}

kerberos$target:::krb_error-read
{
printf("Err code: %s\n", args[1]->kerror_error_code);
printf("Err msg client: %s server: %s\n", args[1]->kerror_client,
args[1]->kerror_server);
printf("Err e-text: %s\n", args[1]->kerror_e_text);
}
```

♦ ♦ ♦ 9 장

9 단순 인증 및 보안 계층 사용

이 장에서는 SASL(Simple Authentication and Security Layer)에 대한 정보를 제공합니다.

SASL 정보

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/sasl`에 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.1`로 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. 현재까지는 `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 사용 안함으로 설정됩니다.

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.

♦ ♦ ♦ 10 장

10 네트워크 서비스 인증 구성

이 장에서는 보안 RPC를 사용하여 NFS 마운트에서 호스트 및 사용자를 인증하는 방법에 대한 정보를 제공하고 다음 항목을 다룹니다.

보안 RPC 정보

보안 RPC(원격 프로세저 호출)는 인증 방식을 통해 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 사용하면 여러 호스트가 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 권한 부여 방식 `AUTH_DH`에서는 DES 암호화를 사용하여 허용된 액세스를 확인합니다. `AUTH_DH` 방식은 `AUTH_DES`라고 부르기도 합니다. 자세한 내용은 다음을 참조하십시오.

- 보안 NFS를 설정하고 관리하려면 "Oracle Solaris 11.2의 네트워크 파일 시스템 관리"의 "보안 NFS 시스템 관리"를 참조하십시오.

Kerberos 인증

Kerberos는 MIT에서 개발된 인증 시스템입니다. 이 릴리스에는 RPCSEC_GSS를 사용하는 Kerberos V5의 클라이언트측 및 서버측 구현이 포함되어 있습니다. 자세한 내용은 Kerberos NFS 서버 구성 방법 [112]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.

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 키라고도 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. NTP(Network Time Protocol)를 사용하여 클럭을 동기화할 수 있습니다. Oracle Solaris 소프트웨어에는 University of Delaware의 NTP 공용 도메인 소프트웨어가 포함되어 있습니다. 설명서는 NTP Documentation 웹 사이트에서 제공합니다.

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.

보안 RPC를 사용하여 인증 관리

마운트된 NFS 파일 시스템에 사용할 인증을 획득하여 네트워크 보안을 강화할 수 있습니다.

다음 작업 맵에서는 NIS 및 NFS에 대해 보안 RPC를 구성하는 절차를 안내합니다.

표 10-1 보안 RPC를 사용하여 인증 관리 작업 맵

작업	설명	지침
1. 키 서버를 시작합니다.	키를 생성하여 사용자를 인증할 수 있도록 합니다.	보안 RPC 키 서버를 다시 시작하는 방법 [197]
2. NIS 호스트에서 자격 증명을 설정합니다.	호스트의 root 사용자를 NIS 환경에서 인증할 수 있도록 합니다.	NIS 호스트에 대한 Diffie-Hellman 키를 설정하는 방법 [197]
3. NIS 사용자에게 키를 제공합니다.	사용자를 NIS 환경에서 인증할 수 있도록 합니다.	NIS 사용자에 대한 Diffie-Hellman 키를 설정하는 방법 [198]
4. 인증을 사용하여 NFS 파일을 공유합니다.	NFS 서버가 인증을 사용하여 공유 파일 시스템을 안전하게 보호할 수 있도록 합니다.	Diffie-Hellman 인증을 사용하여 NFS 파일을 공유하는 방법 [199]

▼ 보안 RPC 키 서버를 다시 시작하는 방법

시작하기 전에 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

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

NIS 도메인의 모든 호스트에서 이 절차를 수행합니다.

시작하기 전에 root 역할을 맡아야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

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 -s name-service/switch setprop config/publickey = astring: "nis"
# svccfg -s name-service/switch:default refresh
```

c. **publickey 값을 확인합니다.**

```
# 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 10-1 NIS 클라이언트에서 root에 대한 새 키 설정

다음 예에서는 Name Service Security 권한 프로파일이 있는 관리자가 earth를 보안 NIS 클라이언트로 설정합니다.

```
# newkey -h earth
Adding new key for unix.earth@example.com
New Password: xxxxxxxx
Retype password: xxxxxxxx
Please wait for the database to get updated...
Your new key has been successfully stored away.
#
```

▼ NIS 사용자에 대한 Diffie-Hellman 키를 설정하는 방법

NIS 도메인의 모든 사용자에 대해 이 절차를 수행합니다.

시작하기 전에 사용자에 대한 새 키를 생성하려면 NIS 마스터 서버에 로그인된 상태여야 합니다. Name Service Security 권한 프로파일이 지정되어 있어야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **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.

2. **사용자에게 로그인하고 `chkey -p` 명령을 입력하도록 합니다.**

   이 명령을 통해 사용자는 사용자만 알 수 있는 암호로 개인 키를 다시 암호화할 수 있습니다.

참고 - `chkey` 명령은 사용자에 대한 새 키 쌍을 만드는 데 사용할 수 있습니다.

예 10-2 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: xxxxxxxx
Retype password: xxxxxxxx
Please wait for the database to get updated...
Your new key has been successfully stored away.
#
```

그런 다음 사용자 `jdoe`는 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: xxxxxxxx
Please enter the login password for jdoe: xxxxxxxx
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 NIS 호스트에 대한 Diffie-Hellman 키를 설정하는 방법 [197]을 완료하십시오.

이 작업을 수행하려면 System Management 권한 프로파일이 지정된 관리자여야 합니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

1. **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.

2. **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.

♦ ♦ ♦ 부 록 A

Kerberos용 DTrace 프로브

이 부록에서는 DTrace 프로브 및 인수 구조에 대해 설명합니다. 사용 예제를 보려면 “Kerberos 서비스에서 DTrace 사용” [184]을 참조하십시오.

Kerberos의 DTrace 프로브

프로브는 DTrace가 일련의 작업을 수행하기 위해 요청을 바인딩할 수 있는 프로그램 위치 또는 작업입니다. 프로브는 공급자가 정의하고 구현합니다. 공급자는 커널에서 로드 가능한 모듈로, 프로브가 데이터를 추적할 수 있도록 해줍니다.

이러한 프로브는 USDT(User Statically Defined Tracing)용입니다. USDT 프로브는 userland의 Kerberos 프로토콜을 검사하도록 설계되었습니다. STD용 커널 프로브는 제공되지 않습니다.

적절한 DTrace 프로브가 원하는 정보(예: 스택 추적, 시간 기록, 또는 함수 인수)를 기록하도록 하는 스크립트를 만듭니다. 프로브가 실행되면 DTrace는 프로브에서 데이터를 수집하여 사용자에게 보고합니다. 프로브가 수행할 작업이 지정되지 않은 경우 DTrace는 프로브가 실행된 시점 및 CPU를 기록합니다.

Kerberos DTrace 프로브는 RFC4120: The Kerberos Network Authentication Service (V5) (http://www.ietf.org/rfc/rfc4120.txt)에 설명된 Kerberos 메시지 유형을 기반으로 모델링되었습니다. `libgss`를 통해 `mech_krb5`를 사용하는 응용 프로그램을 비롯하여 `libkrb5/mech_krb5`의 소비자는 프로브를 사용할 수 있습니다. 프로브는 메시지 생성과 소비 그리고 메시지 송신과 수신으로 나뉩니다. `libgss`에 대한 자세한 내용은 `libgss`(3LIB) 매뉴얼 페이지를 참조하십시오.

프로브를 사용하려면 `kerberos` 공급자, 프로브 이름(예: `krb_message-recv`) 및 인수를 지정합니다. 예제를 보려면 “Kerberos 서비스에서 DTrace 사용” [184]을 참조하십시오.

Kerberos DTrace 프로브의 정의

`KRB_AP_REP`용 프로브

```
kerberos$pid:::krb_ap_rep-make
kerberos$pid:::krb_ap_rep-read

args[0]        krbinfo_t *
args[1]        kaprepinfo_t *
```

KRB_AP_REQ용 프로브

```
kerberos$pid:::krb_ap_req-make
kerberos$pid:::krb_ap_req-read

args[0]        krbinfo_t *
args[1]        kapreqinfo_t *
args[2]        kticketinfo_t *
args[3]        kauthenticatorinfo_t *
```

KRB_KDC_REP용 프로브

```
kerberos$pid:::krb_kdc_rep-make
kerberos$pid:::krb_kdc_rep-read

args[0]        krbinfo_t *
args[1]        kdcrepinfo_t *
args[2]        kticketinfo_t *
```

KRB_KDC_REQ용 프로브

```
kerberos$pid:::krb_kdc_req-make
kerberos$pid:::krb_kdc_req-read

args[0]        krbinfo_t *
args[1]        kdcreqinfo_t *
```

KRB_CRED용 프로브

```
kerberos$pid:::krb_cred-make
kerberos$pid:::krb_cred-read

args[0]        krbinfo_t *
args[1]        kcredinfo_t *
```

KRB_ERROR용 프로브

```
kerberos$pid:::krb_error-make
kerberos$pid:::krb_error-read

args[0]        krbinfo_t *
args[1]        kerrorinfo_t *
```

KRB_PRIV용 프로브

```
kerberos$pid:::krb_priv-make
kerberos$pid:::krb_priv-read

args[0]        krbinfo_t *
args[1]        kprivinfo_t *
```

KRB_SAFE용 프로브

```
kerberos$pid:::krb_safe-make
kerberos$pid:::krb_safe-read

args[0]        krbinfo_t *
args[1]        ksafeinfo_t *
```

메시지 송/수신용 프로브

```
kerberos$pid:::krb_message-recv
kerberos$pid:::krb_message-send

args[0]        krbinfo_t *
args[1]        kconninfo_t *
```

Kerberos의 DTrace 인수 구조

특정 상황에서는 일부 인수의 값이 0이거나 비어 있을 수 있습니다. Kerberos 인수 구조는 일반적으로 RFC4120: The Kerberos Network Authentication Service (V5) (http://www.ietf.org/rfc/rfc4120.txt)와 일치하도록 설계되었습니다.

DTrace의 Kerberos 메시지 정보

```
typedef struct krbinfo {
uint8_t krb_version;                /* protocol version number (5) */
string krb_message_type;            /* Message type (AS_REQ(10), ...) */
uint64_t krb_message_id;            /* message identifier */
uint32_t krb_message_length;        /* message length */
uintptr_t krb_message;              /* raw ASN.1 encoded message */
} krbinfo_t;
```

참고 - Kerberos 프로토콜에는 메시지 식별자가 없습니다. krb_message_id 식별자는 Kerberos 공급자와 관련되며 make/read와 send/recv 프로브 간에 메시지를 연결하도록 설계되었습니다.

DTrace의 Kerberos 연결 정보

```
typedef struct kconninfo {
string kconn_remote;                /* remote host address */
string kconn_local;                 /* local host address */
uint16_t kconn_localport;           /* local port */
uint16_t kconn_remoteport;          /* remote port */
string kconn_protocol;              /* protocol (ipv4, ipv6) */
```

```
string kconn_type;                /* transport type (udp, tcp) */
} kconninfo_t;
```

DTrace의 Kerberos 인증자 정보

```
typedef struct kauthenticatorinfo {
string kauth_client;              /* client principal identifier */
string kauth_cksum_type;          /* type of checksum (des-cbc, ...) */
uint32_t kauth_cksum_length;      /* length of checksum */
uintptr_t kauth_cksum_value;      /* raw checksum data */
uint32_t kauth_cusec;             /* client time, microseconds */
uint32_t kauth_ctime;             /* client time in seconds */
string kauth_subkey_type;         /* sub-key type (des3-cbc-sha1, ...) */
uint32_t kauth_subkey_length;     /* sub-key length */
uintptr_t kauth_subkey_value;     /* sub-key data */
uint32_t kauth_seq_number;        /* sequence number */
string kauth_authorization_data;  /* top-level authorization types
(AD-IF-RELEVANT, ... ) */
} kauthenticatorinfo_t;

typedef struct kticketinfo_t {
string kticket_server;            /* service principal identifier */
uint32_t kticket_enc_part_kvno;   /* key version number */
string kticket_enc_part_etype;    /* enc type of encrypted ticket */
string kticket_enc_flags;         /* ticket flags (forwardable, ...) */
string kticket_enc_key_type;      /* key type (des3-cbc-sha1, ...) */
uint32_t kticket_enc_key_length;  /* key length */
uintptr_t kticket_enc_key_value;  /* key data */
string kticket_enc_client;        /* client principal identifier */
string kticket_enc_transited;     /* list of transited Kerberos realms */
string kticket_enc_transited_type;  /* encoding type */
uint32_t kticket_enc_authtime;    /* time of initial authentication */
uint32_t kticket_enc_starttime;   /* ticket start time in seconds */
uint32_t kticket_enc_endtime;     /* ticket end time in seconds */
uint32_t kticket_enc_renew_till;  /* ticket renewal time in seconds */
string kticket_enc_addresses;     /* addresses associated with ticket */
string kticket_enc_authorization_data;  /* list of top-level auth types */
} kticketinfo_t;

typedef struct kdcreqinfo {
string kdcreq_padata_types;       /* list of pre-auth types */
string kdcreq_kdc_options;        /* requested ticket flags */
string kdcreq_client;             /* client principal identifier */
string kdcreq_server;             /* server principal identifier */
uint32_t kdcreq_from;             /* requested start time in seconds */
uint32_t kdcreq_till;             /* requested end time in seconds */
uint32_t kdcreq_rtime;            /* requested renewal time in seconds */
uint32_t kdcreq_nonce;            /* nonce for replay detection */
string kdcreq_etype;              /* preferred encryption types */
string kdcreq_addresses;          /* list of requested ticket addresses */
string kdcreq_authorization_data;  /* list of top-level auth types */
```

```
uint32_t kdcreq_num_additional_tickets; /* number of additional tickets */
} kdcreqinfo_t;

typedef struct kdcrepinfo {
string kdcrep_padata_types;         /* list of pre-auth types */
string kdcrep_client;               /* client principal identifier */
uint32_t kdcrep_enc_part_kvno;      /* key version number */
string kdcrep_enc_part_etype;       /* enc type of encrypted KDC reply */
string kdcrep_enc_key_type;         /* key type (des3-cbc-sha1, ...) */
uint32_t kdcrep_enc_key_length;     /* key length */
uintptr_t kdcrep_enc_key_value;     /* key data */
string kdcrep_enc_last_req;         /* times of last request of principal */
uint32_t kdcrep_enc_nonce;          /* nonce for replay detection */
uint32_t kdcrep_enc_key_expiration; /* expiration time of client's key */
string kdcrep_enc_flags;            /* ticket flags */
uint32_t kdcrep_enc_authtime;       /* time of authentication of ticket */
uint32_t kdcrep_enc_starttime;      /* ticket start time in seconds */
uint32_t kdcrep_enc_endtime;        /* ticket end time in seconds */
uint32_t kdcrep_enc_renew_till;       /* ticket renewal time in seconds*/
string kdcrep_enc_server;           /* server principal identifier */
string kdcrep_enc_caddr;            /* zero or more client addresses */
} kdcrepinfo_t;

typedef struct kapreqinfo {
string kapreq_ap_options;           /* options (use-session-key,... ) */
uint32_t kapreq_authenticator_kvno; /* key version number */
string kapreq_authenticator_etype;  /* enc type of authenticator */
} kapreqinfo_t;

typedef struct kaprepinfo {
uint32_t kaprep_enc_part_kvno;      /* key version number */
string kaprep_enc_part_etype;       /* enc type of encrypted AP reply */
uint32_t kaprep_enc_ctime;          /* client time in seconds */
uint32_t kaprep_enc_cusec;          /* client time, microseconds portion */
string kaprep_enc_subkey_type;      /* sub-key type */
uint32_t kaprep_enc_subkey_length;  /* sub-key length */
uintptr_t kaprep_enc_subkey_value;  /* sub-key data */
uint32_t kaprep_enc_seq_number;     /* sequence number */
} kaprepinfo_t;

typedef struct kerrorinfo {
uint32_t kerror_ctime;              /* client time in seconds */
uint32_t kerror_cusec;              /* client time, microseconds */
uint32_t kerror_stime;              /* server time in seconds */
uint32_t kerror_susec;              /* server time, microseconds */
string kerror_error_code;           /* error code (KRB_AP_ERR_SKEW, ...) */
string kerror_client;               /* client principal identifier */
string kerror_server;               /* server principal identifier */
string kerror_e_text;               /* additional error text */
string kerror_e_data;               /* additional error data */
} kerrorinfo_t;

typedef struct ksafeinfo {
uintptr_t ksafe_user_data;          /* raw application specific data */
```

```
uint32_t ksafe_timestamp;          /* time of sender in seconds */
uint32_t ksafe_usec;               /* time of sender, microseconds */
uint32_t ksafe_seq_number;         /* sequence number */
string ksafe_s_address;            /* sender's address */
string ksafe_r_address;            /* recipient's address */
string ksafe_cksum_type;           /* checksum type (des-cbc, ...) */
uint32_t ksafe_cksum_length;       /* length of checksum */
uintptr_t ksafe_cksum_value;       /* raw checksum data */
} ksafeinfo_t;

typedef struct kprivinfo {
uint32_t kpriv_enc_part_kvno;      /* key version number */
string kpriv_enc_part_etype;       /* enc type of encrypted message */
uintptr_t kpriv_enc_user_data;     /* raw application specific data */
uint32_t kpriv_enc_timestamp;      /* time of sender in seconds */
uint32_t kpriv_enc_usec;           /* time of sender, microseconds */
uint32_t kpriv_enc_seq_number;     /* sequence number */
string kpriv_enc_s_address;        /* sender's address */
string kpriv_enc_r_address;        /* recipient's address */
} kprivinfo_t;

typedef struct kcredinfo {
uint32_t kcred_enc_part_kvno;      /* key version number */
string kcred_enc_part_etype;       /* enc type of encrypted message */
uint32_t kcred_tickets;            /* number of tickets */
uint32_t kcred_enc_nonce;          /* nonce for replay detection */
uint32_t kcred_enc_timestamp;      /* time of sender in seconds */
uint32_t kcred_enc_usec;           /* time of sender, microseconds */
string kcred_enc_s_address;        /* sender's address */
string kcred_enc_r_address;        /* recipient's address */
} kcredinfo_t;
```

보안 용어

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. private-key encryption(개인 키 암호화)도 참조하십시오.

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. principal name(주체 이름), user principal(사용자 주체)도 참조하십시오.

기본 주체 이름의 첫번째 부분입니다. 인스턴스, principal name(주체 이름), realm(영역)도 참조하십시오.

네트워크 애플리케이션 서버 `ftp`와 같은 네트워크 응용 프로그램을 제공하는 서버입니다. 영역에 여러 네트워크 애플리케이션 서버를 포함할 수 있습니다.

네트워크 정책 네트워크 트래픽을 보호하기 위해 네트워크 유틸리티가 구성하는 설정입니다. 네트워크 보안에 대한 자세한 내용은 "Oracle Solaris 11.2의 네트워크 보안 "을 참조하십시오.

다이제스트 message digest(메시지 다이제스트)를 참조하십시오.

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.

무결성 사용자 인증과 더불어, 암호화 체크섬을 통해 전송된 데이터의 유효성을 제공하는 보안 서비스입니다. authentication (인증), privacy(프라이버시)도 참조하십시오.

문장암호 개인 키를 문장암호 사용자가 만들었는지 확인하는 데 사용되는 문구입니다. 좋은 문장암호는 10-30자 길이로, 영문자와 숫자를 섞어서 만들고 단순한 문구와 단순한 이름을 피합니다. 통신을 암호화 및 해독하기 위해 개인 키 사용을 인증하려면 문장암호를 묻는 메시지가 나타납니다.

보안 서비스 서비스를 참조하십시오.

보안 정책 policy(정책)을 참조하십시오.

상속 가능한 세트 exec의 호출에서 프로세스가 상속할 수 있는 권한 세트입니다.

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 privacy(프라이버시)를 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. host principal(호스트 주체), user principal(사용자 주체)도 참조하십시오.

서비스 키 서비스 주체 및 KDC에서 공유되고 시스템 한도 밖에서 배포되는 암호화 키입니다. key(키)를 참조하십시오.

수퍼 유저 모델 컴퓨터 시스템의 전형적인 UNIX 보안 모델입니다. 수퍼 유저 모델에서 관리자는 all-or-nothing 방식으로 시스템을 제어합니다. 일반적으로 시스템을 관리하려는 경우 사용자는 수퍼 유저(`root`)로 로그인하여 모든 관리 작업을 수행할 수 있습니다.

알고리즘 암호화 알고리즘입니다. 입력을 암호화하거나 해시하는 확립된 순환적 계산 프로시저입니다.

암호 정책 암호를 생성하는 데 사용할 수 있는 암호화 알고리즘입니다. 암호 변경 주기, 암호 시도 허용 회수, 기타 보안 고려 사항 등 암호와 관련한 일반적인 사안이라고 할 수 있습니다. 보안 정책에 암호가 필요합니다. 암호 정책에서는 암호를 AES 알고리즘으로 암호화해야 하고, 추가로 암호 강도와 관련된 요구 사항이 필요할 수 있습니다.

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.

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.

유효 세트 현재 프로세스에 발효 중인 권한 세트입니다.

이름 서비스 범위 역할이 작동하도록 허가된 범위입니다. 즉, NIS LDAP와 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. 기본, principal name(주체 이름), 서비스 주체, user principal(사용자 주체)도 참조하십시오.

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. postdated ticket(후일자 티켓)도 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. policy(정책)을 참조하십시오.

장치 할당 사용자 레벨의 장치 보호입니다. 장치 할당은 하나의 장치를 한번에 한 사용자만 배타적으로 사용하도록 합니다. 장치 재사용 전에 장치 데이터를 비웁니다. 권한 부여를 사용하여 장치 할당이 허가된 사용자를 제한할 수 있습니다.

제한 세트 프로세스와 그 자식에 사용 가능한 권한에 대한 외부 제한입니다.

주체 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. 다신 말해서, 주체는 Kerberos가 티켓을 지정할 수 있는 고유한 개체입니다. principal name(주체 이름), 서비스 주체, user principal(사용자 주체)도 참조하십시오.

2. (RPCSEC_GSS API) 클라이언트 주체, 서버 주체를 참조하십시오.

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.

최소 권한의 원칙 최소한의 특권을 참조하십시오.

최소한의 특권 지정된 프로세스를 일부 수퍼 유저 권한에만 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 수퍼 유저 권한(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. authenticator (인증자), credential(자격 증명), 서비스, session key(세션 키)를 참조하십시오.

티켓 파일 credential cache(자격 증명 캐시)를 참조하십시오.

AES Advanced Encryption Standard의 머리글자어로, 고급 암호화 표준입니다. 대칭 128비트 블록 데이터 암호화 기술입니다. 미국 정부는 2000년 10월 알고리즘의 Rijndael 변형을 암호화 표준으로 채택했습니다. AES가 정부 표준으로 user principal(사용자 주체) 암호화를 대체합니다.

application server(애플리케이션 서버) 네트워크 애플리케이션 서버를 참조하십시오.

asynchronous audit event(비동기 감사 이벤트) 비동기 이벤트는 시스템 이벤트의 소수를 차지합니다. 이러한 이벤트는 어떤 프로세스와 연관되지 않으므로 프로세스를 차단했다가 나중에 깨울 수 없습니다. 초기 시스템 부트 및 PROM 진입/종료 이벤트가 비동기 이벤트의 예입니다.

audit files(감사 파일) 이진 감사 로그. 감사 파일은 감사 파일 시스템에 별도로 저장됩니다.

audit policy(감사 정책) 어떤 감사 이벤트를 기록할지 결정하는 전역 사용자별 설정입니다. 감사 서비스에 적용되는 전역 설정은 일반적으로 감사 추적에 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.

audit trail(감사 추적) 모든 호스트의 모든 감사 파일 모음입니다.

authenticated rights profile(인증된 권한 프로파일) 지정된 사용자나 역할이 프로파일에서 작업을 실행하기 전에 암호를 입력해야 하는 rights profile(권한 프로파일)입니다. 이 동작은 `sudo` 동작과 비슷합니다. 암호가 유효하며 구성 가능한 기간입니다.

authentication(인증) 주체의 제시된 신원을 확인하는 프로세스입니다.

authenticator(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.

authorization(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. 사용자 권한 관리에서는 보안 정책으로 금지된 일련의 작업을 수행하기 위해 역할 또는 사용자에게 지정할 수 있는(또는 권한 프로파일에 포함할 수 있는) 권한입니다. 권한 부여는 커널에서가 아니라 사용자 응용 프로그램 레벨에서 실행됩니다.

basic set(기본 세트) 로그인 시 사용자 프로세스에 지정되는 권한 세트입니다. 수정되지 않은 시스템에서 각 사용자의 초기 상속 가능한 세트는 로그인 시 기본 세트와 같습니다.

Blowfish 32-448비트의 가변 길이 키를 사용하는 대칭 블록 암호화 알고리즘입니다. 저작자인 Bruce Schneier에 따르면, Blowfish는 키를 자주 바꾸지 않는 응용 프로그램에 최적화되어 있습니다.

confidentiality(기밀성) privacy(프라이버시)를 참조하십시오.

consumer(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 provider(공급자)를 참조하십시오.

credential cache(자격 증명 캐시) KDC로부터 받은 자격 증명을 포함하는 저장 공간(대개 파일)입니다.

credential(자격 증명) 티켓 및 일치하는 세션 키를 포함하는 정보 패키지입니다. 주체의 신원을 인증하는 데 사용됩니다. 티켓, session key(세션 키)도 참조하십시오.

DES Data Encryption Standard의 머리글자어로, 데이터 암호화 표준입니다. 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 Algorithm의 머리글자어로, 디지털 서명 알고리즘입니다. 512-4096비트의 가변 키 크기를 사용하는 공개 키 알고리즘입니다. 미국 정부 표준인 DSS는 1024비트까지 지원합니다. DSA는 입력에 SHA1을 사용합니다.

ECDSA Elliptic Curve Digital Signature Algorithm의 머리글자어로, 타원 곡선 디지털 서명 알고리즘입니다. 타원 곡선 수학을 기반으로 하는 공개 키 알고리즘입니다. ECDSA 키 크기는 동일한 길이의 서명을 생성하는 데 필요한 DSA 공개 키의 크기보다 많이 작습니다.

flavor(종류) 전통적으로 보안 종류와 인증 종류는 인증 유형(AUTH_UNIX, AUTH_DES, AUTH_KERB)을 지칭한 종류로서, 동일한 의미입니다. RPCSEC_GSS는 인증과 더불어 무결성과 프라이버시 서비스를 제공하지만 역시 보안 종류입니다.

forwardable ticket(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 `david`는 새 티켓을 얻지 않고도(다시 인증 받을 필요 없이) 자신의 시스템에 로그인할 수 있습니다. proxiable ticket(프록시 가능 티켓)도 참조하십시오.

FQDN 정규화된 도메인 이름입니다. 간단한 `denver`와 대조되는 `central.example.com`을 예로 들 수 있습니다.

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. authentication (인증), 무결성, privacy(프라이버시)를 참조하십시오.

hardening(강화) 호스트에 내재된 보안 취약성을 제거하도록 운영 체제의 기본 구성을 수정한 것입니다.

hardware provider(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. provider(공급자)도 참조하십시오.

host principal(호스트 주체) `ftp`, `rcp` 또는 `rlogin`과 같은 다양한 네트워크 서비스를 제공하기 위해 주체(기본 이름 `host`로 서명됨)가 설정되는 특정 인스턴스의 서비스 주체입니다. 호스트 주체의 예는 `host/central.example.com@EXAMPLE.COM`입니다. 서버 주체도 참조하십시오.

host(호스트) 네트워크를 통해 액세스 가능한 시스템입니다.

KDC Key Distribution Center의 머리글자어로, 키 배포 센터입니다. 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.

Kerberos V5 구현에 가장 근접한 Oracle Solaris의 Kerberos 구현입니다.

"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 policy (Kerberos 정책) Kerberos 서비스에서 암호 사용을 통제하는 규칙 세트입니다. 정책을 통해 주체의 액세스나 티켓 수명 매개변수를 규제할 수 있습니다.

key(키) 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 file(keytab 파일)도 참조하십시오.

3. Kerberos에서 암호화 키로 사용되며 다음 세 가지 유형이 있습니다.

- 개인 키 - 주체 및 KDC에서 공유되고 시스템 한도 밖에서 배포되는 암호화 키입니다. private key(개인 키)도 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. session key(세션 키)도 참조하십시오.

keystore(키 저장소) 키 저장소는 응용 프로그램별로 검색하기 위한 암호, 문장암호, 인증서 및 기타 인증 객체를 저장합니다. 키 저장소는 일부 응용 프로그램이 사용하는 기술 또는 위치에 따라 달라질 수 있습니다.

keytab file(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(모든 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 보증하고 데이터 발신을 인증합니다. MAC는 도청에 대해 보호되지 않습니다.

MD5 디지털 서명을 포함하여 메시지 인증용으로 사용되는 반복적인 암호화 해시 함수입니다. 이 기능은 1991년 Rivest가 개발했습니다. 이 기술은 더 이상 사용되지 않습니다.

mechanism(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.

message digest(메시지 다이제스트) 메시지 다이제스트는 메시지에서 계산된 해시 값입니다. 해시 값은 메시지를 거의 고유하게 식별합니다. 다이제스트는 파일 무결성 확인에 유용합니다.

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. PAM은 로그인 시 Kerberos 세션 초기화를 사용으로 설정합니다.

permitted set(허가된 세트) 프로세스에서 사용할 수 있는 권한 세트입니다.

policy(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 사용할 장치를 할당하며 암호를 6주마다 변경하도록 보안 정책을 수립할 수 있습니다.

Oracle Solaris OS의 특정 영역에서 정책을 구현하는 방법은 audit policy(감사 정책), 암호화 프레임워크의 정책, 장치 정책, Kerberos policy (Kerberos 정책), 암호 정책 및 rights policy(권한 정책)을 참조하십시오.

postdated ticket(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, renewable ticket(갱신 가능 티켓)도 참조하십시오.

principal name(주체 이름) 1. *primary/instance@REALM* 형식의 주체 이름입니다. 인스턴스, 기본, realm(영역)도 참조하십시오.

2. (RPCSEC_GSS API) 클라이언트 주체, 서버 주체를 참조하십시오.

privacy(프라이버시) 전송된 데이터를 보내기 전에 암호화하는 보안 서비스입니다. 프라이버시에는 데이터 무결성과 사용자 인증도 포함됩니다. authentication (인증), 무결성, 서비스를 참조하십시오.

private key(개인 키) 각 사용자 주체에 제공되며 주체의 사용자와 KDC에만 알려진 키입니다. 사용자 주체의 경우 키는 사용자 암호를 기반으로 합니다. key(키)를 참조하십시오.

private-key encryption(개인 키 암호화) 개인 키 암호화에서 발신자와 수신자는 암호화에 동일한 키를 사용합니다. 공개 키 암호화도 참조하십시오.

privilege escalation(권한 에스컬레이션) 지정된 권한(기본값 대체 권한 포함)이 허용하는 리소스 범위 밖에 있는 리소스에 대한 액세스 권한을 얻는 것입니다. 그 결과, 프로세스가 권한이 없는 작업을 수행할 수 있습니다.

privilege model(권한 모델) 수퍼 유저 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.

privilege set(권한 세트) 권한 모음입니다. 각 프로세스에는 프로세스가 특정 권한을 사용할 수 있는지 여부를 결정하는 4개의 권한 세트가 있습니다. 제한 세트, 유효 세트, permitted set(허가된 세트), 상속 가능한 세트를 참조하십시오.

또한 권한의 basic set(기본 세트)는 로그인 시 사용자의 프로세스에 지정된 권한 모음입니다.

privilege-aware(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.

privilege(권한)

1. 일반적인 의미는 컴퓨터 시스템에서 일반 사용자의 권한을 넘는 작업을 수행할 수 있는 능력입니다. 수퍼 유저 권한은 수퍼 유저에게 부여된 모든 rights(권한)입니다. 권한 있는 사용자 또는 권한 있는 응용 프로그램은 추가 권한이 부여된 사용자 또는 응용 프로그램입니다.

2. Oracle Solaris 시스템에서 프로세스에 대한 별개의 권한입니다. 권한은 `root`인 프로세스를 좀 더 세부적으로 제어합니다. 권한은 커널에서 정의되고 시행됩니다. 권한을 프로세스 권한 또는 커널 권한이라고도 합니다. 권한에 대한 자세한 설명은 `privileges`(5) 매뉴얼 페이지를 참조하십시오.

privileged application(권한 있는 응용 프로그램)

시스템 컨트롤을 대체할 수 있는 응용 프로그램입니다. 응용 프로그램이 특정 UID, GID, 권한 부여 또는 권한과 같은 보안 속성을 검사합니다.

privileged user(권한 있는 사용자)

컴퓨터 시스템에 대해 일반 사용자의 권한 밖에 있는 권한이 지정된 사용자입니다. trusted users(신뢰할 수 있는 사용자)도 참조하십시오.

profile shell(프로파일 셸)

권한 관리에서 역할(또는 사용자)이 권한 프로파일에 지정된 권한 있는 응용 프로그램을 명령줄에서 실행할 수 있는 셸입니다. 프로파일 셸 버전은 시스템에서 사용할 수 있는 셸(예: `bash`의 `pfbash` 버전)에 해당합니다.

provider(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 플러그인이라고도 합니다. 소비자의 예는 consumer(소비자)를 참조하십시오.

proxiable ticket(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. forwardable ticket(전달 가능 티켓)도 참조하십시오.

public object(공용 객체)

`root` 사용자가 소유하고 어디서든 읽을 수 있는 파일입니다(예: `/etc` 디렉토리의 파일).

QOP Quality of Protection의 약자. 무결성 서비스나 프라이버시 서비스와 함께 사용되는 암호화 알고리즘을 선택할 수 있는 매개변수입니다.

RBAC Oracle Solaris의 사용자 권한 관리 기능의 일종인 역할 기반 액세스 제어입니다. rights(권한)을 참조하십시오.

RBAC policy(RBAC 정책) rights policy(권한 정책)을 참조하십시오.

realm(영역) 1. 단일 Kerberos 데이터베이스와 일련의 KDC(키 배포 센터)에 의해 제공된 논리적 네트워크입니다.

2. 주체 이름의 세번째 부분입니다. 주체 이름 `jdoe/admin@CORP.EXAMPLE.COM`의 경우 영역은 `CORP.EXAMPLE.COM`입니다. principal name(주체 이름)도 참조하십시오.

reauthentication(재인증) 컴퓨터 작업을 수행하기 위해 암호를 제공하도록 요구하는 것입니다. 일반적으로 `sudo` 작업에 재인증이 필요합니다. 인증된 권한 프로파일에 재인증을 요구하는 명령을 포함할 수 있습니다. authenticated rights profile(인증된 권한 프로파일)을 참조하십시오.

relation(관계) `kdc.conf` 또는 `krb5.conf` 파일에 정의된 구성 변수 또는 관계입니다.

renewable ticket(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.

rights policy(권한 정책) 명령과 연관된 보안 정책입니다. 현재 Oracle Solaris의 유효한 정책은 `solaris`입니다. `solaris` 정책은 권한 및 확장된 권한 정책, 권한 부여 및 `setuid` 보안 속성을 인식합니다.

rights profile(권한 프로파일) 프로파일이라고도 합니다. 역할 또는 사용자에 지정할 수 있는 보안 대체 모음입니다. 권한 프로파일에는 권한 부여, 권한, 보안 속성이 포함된 명령 및 보충 프로파일이라고 하는 기타 권한 프로파일이 포함될 수 있습니다.

rights(권한) all-or-nothing 수퍼 유저 모델의 대안입니다. 사용자 권한 관리 및 프로세스 권한 관리를 통해 조직은 수퍼 유저의 권한을 분담하고 사용자 또는 역할에 이를 지정할 수 있습니다. Oracle Solaris의 권한은 커널 권한, 권한 부여 및 프로세스를 UID 또는 GID로 실행하는 기능으로 구현됩니다. rights profile(권한 프로파일) 및 role(역할)에 권한을 수집할 수 있습니다.

role(역할) 지정된 사용자만 맡을 수 있는 권한 있는 응용 프로그램을 실행하기 위한 특수한 신원입니다.

RSA 디지털 서명 및 공개 키 암호화 체계를 얻기 위한 방법입니다. 1978년에 개발자 Rivest, Shamir, Adleman이 처음 기술했습니다.

scan engine(검사 엔진) 파일에 알려진 바이러스가 있는지 조사하는, 외부 호스트에 상주하는 타사 응용 프로그램입니다.

SEAM Solaris 시스템의 Kerberos 초기 버전에 대한 제품 이름입니다. 이 제품은 MIT(Massachusetts Institute of Technology)에서 개발된 Kerberos V5 기술을 기반으로 합니다. 이제 SEAM을 Kerberos 서비스라고 부릅니다. MIT 버전과는 약간 다릅니다.

secret key(보안 키) private key(개인 키)를 참조하십시오.

Secure Shell(보안 셸) 비보안 네트워크를 통해 보안 원격 로그인 및 기타 보안 네트워크 서비스를 제공하기 위한 특수한 프로토콜입니다.

security attributes(보안 속성) 수퍼 유저가 아닌 일반 사용자가 관리 명령을 실행할 때 명령을 성공하게 해주는 보안 정책의 대체입니다. 수퍼 유저 모델에서는 `setuid root` 및 `setgid` 프로그램이 보안 속성입니다. 이러한 속성을 명령에 적용할 때 누가 명령을 실행하든지 관계없이 명령을 성공합니다. privilege model(권한 모델)에서 커널 권한과 기타 rights(권한)이 보안 속성으로서 `setuid root` 프로그램을 대체합니다. 권한 모델은 `setuid` 및 `setgid` 프로그램을 보안 속성으로 인식하므로 수퍼 유저 모델과 호환됩니다.

security flavor(보안 종류) flavor(종류)를 참조하십시오.

security mechanism(보안 방식) mechanism(방식)을 참조하십시오.

seed(시드) 무작위 수를 생성하기 위한 숫자 스타터입니다. 스타터가 무작위 소스에서 기원할 때 시드를 무작위 시드라고 합니다.

separation of duty(책임 구분) 최소한의 특권 개념의 일부입니다. 책임 구분을 통해 한 사용자가 트랜잭션을 완성하는 모든 작업을 수행하거나 승인할 수 없도록 합니다. 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.

server(서버) 네트워크 클라이언트에 리소스를 제공하는 주체입니다. 예를 들어, `central.example.com` 시스템에 `ssh`를 제공하면 해당 시스템은 `ssh` 서비스를 제공하는 서버입니다. 서비스 주체도 참조하십시오.

session key(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. key(키)를 참조하십시오.

SHA1 보안 해시 알고리즘(Secure Hashing Algorithm)의 약자. 이 알고리즘은 2^{64} 미만의 입력 길이에서 작동하여 메시지 다이제스트를 생성합니다. SHA1 알고리즘은 DSA에 입력됩니다.

single-system image(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.

slave KDC(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.

software provider(소프트웨어 공급자) Oracle Solaris의 암호화 프레임워크 기능에서 암호화 서비스를 제공하는 커널 소프트웨어 모듈 또는 PKCS #11 라이브러리입니다. provider(공급자)도 참조하십시오.

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.

trusted users(신뢰할 수 있는 사용자) 결정된 사용자는 특정 신뢰 레벨에서 관리 작업을 수행할 수 있습니다. 대개 관리자가 신뢰할 수 있는 사용자의 로그인을 먼저 만들고 사용자의 신뢰 및 능력 레벨에 맞는 관리 권한을 지정합니다. 이러한 사용자는 시스템 구성 및 유지 관리 작업을 수행하는 데 도움이 됩니다. 권한 있는 사용자라고도 합니다.

user principal(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.

VPN(가상 사설망) 암호화를 사용하고 공용 네트워크를 통한 사용자 연결을 터널링하여 보안 통신을 제공하는 네트워크입니다.

색인

ㅈ