Oracle® Solaris 11.2의 보안 셸 액세스 관리

부품 번호: E53965-02
2014년 9월

ORACLE®

목차

이 설명서 사용

*Oracle® Solaris 11.2*에서 보안 셸 액세스 관리에서는 보안 원격 액세스를 위해 보안 셸 기능을 관리하고 사용하는 방법을 설명합니다.

- **개요** - Oracle Solaris의 보안 셸 사용에 대한 개념과 작업을 설명합니다.
- **대상** - 기업에서 보안을 구현해야 하는 시스템 관리자
- **필요한 지식** - 보안 개념과 용어에 익숙해야 합니다.

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 보안 셸 사용(작업)

Oracle Solaris의 보안 셸 기능을 사용하면 비보안 네트워크를 통해 원격 호스트에 안전하게 액세스할 수 있습니다. 셸은 원격 로그인, 원격 창 표시 및 원격 파일 전송을 위한 명령을 제공합니다. 이 장에서는 다음 내용을 다룹니다.

참조 정보는 2장. 보안 셸 참조를 참조하십시오.

보안 셸(개요)

보안 셸은 새로 설치된 Oracle Solaris 시스템의 기본 원격 액세스 프로토콜입니다. Oracle Solaris의 보안 셸은 Secure Sockets Layer 및 Transport Layer Security를 구현하는 오픈 소스 툴킷 OpenSSL을 기반으로 내장되었습니다.

이 툴킷은 Oracle Solaris에서 두 개의 고유 버전으로 제공됩니다.

- 보안 셸이 실행되는 기본 버전은 버전 1.0.0입니다.
- 버전 0.9.8은 암호화 모듈에 대한 미국 정보 컴퓨터 보안 표준인 FIPS-140FIPS 140을 구현합니다.

  FIPS 140 모드에서 보안 셸을 사용하는 방법에 대한 자세한 내용은 "보안 셸 및 FIPS 140" [10]을 참조하십시오.

보안 셸에서 인증은 암호, 공개 키 또는 둘 다 사용하여 제공됩니다. 모든 네트워크 트래픽은 암호화됩니다. 따라서 보안 셸은 침입자가 가로챈 통신을 읽지 못하도록 합니다. 또한 보안 셸은 침입자의 시스템 속임수를 방지합니다.

보안 셸을 필요 시 VPN(Virtual Private Network)으로 사용할 수도 있습니다. VPN은 X 윈도우 시스템 트래픽을 전달할 수도 있고, 암호화된 네트워크 링크를 통해 로컬 시스템과 원격 시스템 간에 개별 포트 번호를 연결할 수도 있습니다.

보안 셸을 통해 다음 작업을 수행할 수 있습니다.

- 비보안 네트워크를 통해 안전하게 다른 호스트에 로그인합니다.
- 두 호스트 간에 안전하게 파일을 복사합니다.
- 원격 호스트에서 안전하게 명령을 실행합니다.

서버측에서 보안 셸은 버전 2(v2)의 보안 셸 프로토콜을 지원합니다. 클라이언트측에서 v2 외에도 클라이언트가 버전 1(v1)을 지원합니다.

보안 셸 인증

보안 셸은 원격 호스트에 대한 연결에 사용할 공개 키 및 암호 인증 방법을 제공합니다. 개인 키는 네트워크를 통과하지 않으므로 공개 키 인증이 암호 인증보다 강력한 인증 방식입니다.

인증 방법은 다음 순서로 시도됩니다. 구성이 인증 방법을 충족하지 않을 경우 다음 방법이 시도됩니다.

- **GSS-API** - `mech_krb5`(Kerberos V), `mech_dh`(AUTH_DH) 등의 GSS-API 방식에서 자격 증명을 사용하여 클라이언트와 서버를 인증합니다. GSS-API에 대한 자세한 내용은 "Developer's Guide to Oracle Solaris 11 Security "의 "Introduction to GSS-API"를 참조하십시오.
- **호스트 기반 인증** - 호스트 키 및 `rhosts` 파일을 사용합니다. 클라이언트의 RSA 및 DSA 공개/개인 호스트 키를 사용하여 클라이언트를 인증합니다. `rhosts` 파일을 사용하여 사용자에 대해 클라이언트에 권한을 부여합니다.
- **공개 키 인증** - RSA 및 DSA 공개/개인 키로 사용자를 인증합니다.
- **암호 인증** - PAM을 사용하여 사용자를 인증합니다. v2의 키보드 인증 방법은 PAM의 임의적인 프롬프트를 허용합니다. 자세한 내용은 `sshd`(1M) 매뉴얼 페이지의 `SECURITY` 절을 참조하십시오.

다음 표에서는 원격 호스트에 로그인하려고 시도 중인 사용자를 인증하기 위한 요구 사항을 보여 줍니다. 사용자는 로컬 호스트인 클라이언트에 있습니다. 원격 호스트인 서버는 `sshd` 데몬을 실행 중입니다. 표에서는 보안 셸 인증 방법과 호스트 요구 사항을 보여줍니다.

표 1-1 보안 셸 인증 방법

인증 방법	로컬 호스트(클라이언트) 요구 사항	원격 호스트(서버) 요구 사항
GSS-API	GSS 방식에 대한 개시자 자격 증명입니다.	GSS 방식에 대한 승인자 자격 증명입니다. 자세한 내용은 "보안 셸에서 GSS 자격 증명 취득" [32]을 참조하십시오.

호스트 기반	사용자 계정
	
	`/etc/ssh/ssh_host_rsa_key` 또는 `/etc/ssh/ssh_host_dsa_key`의 로컬 호스트 개인 키
	
	`/etc/ssh/ssh_config`의 `Hostbased Authentication yes`
		사용자 계정
		
		`/etc/ssh/known_hosts` 또는 `~/.ssh/known_hosts`의 로컬 호스트 공개 키
		
		`/etc/ssh/sshd_config`의 `Hostbased Authentication yes`
		
		`/etc/ssh/sshd_config`의 `IgnoreRhosts no`

인증 방법	로컬 호스트(클라이언트) 요구 사항	원격 호스트(서버) 요구 사항
		/etc/ssh/shosts.equiv, /etc/hosts.equiv, ~/.rhosts 또는 ~/.shosts의 로컬 호스트 항목
암호 기반	사용자 계정	사용자 계정 PAM을 지원합니다.
서버에서만 RSA(v1)를 사용한 .rhosts	사용자 계정 /etc/ssh/ssh_host_rsa1_key의 로컬 호스트 공개 키	사용자 계정 /etc/ssh/ssh_known_hosts 또는 ~/.ssh/known_hosts의 로컬 호스트 공개 키 /etc/ssh/sshd_config의 IgnoreRhosts no /etc/ssh/shosts.equiv, /etc/hosts.equiv, ~/.shosts 또는 ~/.rhosts의 로컬 호스트 항목
RSA 또는 DSA 공개 키	사용자 계정 ~/.ssh/id_rsa 또는 ~/.ssh/id_dsa의 개인 키 ~/.ssh/id_rsa.pub 또는 ~/.ssh/id_dsa.pub의 사용자 공개 키	사용자 계정 ~/.ssh/authorized_keys의 사용자 공개 키

보안 셸 및 OpenSSH 프로젝트

보안 셸은 OpenSSH (http://www.openssh.com) 프로젝트의 포크입니다. OpenSSH의 후속 버전에서 발견된 위험성에 대한 보안 수정 사항은 개별 버그 수정 및 기능으로 보안 셸에 통합되었습니다. 2012년 9월 당시 Oracle Solaris의 보안 셸 버전은 2.0이었습니다. ssh -V 명령은 버전 번호를 표시합니다.

다음은 이 릴리스의 보안 셸에서 v2 프로토콜에 대해 구현된 기능입니다.

- ForceCommand 키워드 - 사용자가 명령줄에서 입력하는 내용에 관계없이 지정된 명령을 강제로 실행합니다. 이 키워드는 Match 블록에서 가장 유용합니다. sshd_config 구성 옵션은 $HOME/.ssh/authorized_keys의 command="..." 옵션과 유사합니다.
- AES-128 문장암호 보호 - ssh-keygen 명령으로 생성된 개인 키는 AES-128 알고리즘으로 보호됩니다. 이 알고리즘은 문장암호 변경 등으로 새로 생성된 키 및 다시 암호화된 키를 보호합니다.
- sftp-server 명령에 대한 -u 옵션 - 사용자가 파일 및 디렉토리에 대해 명시적 umask를 설정할 수 있도록 합니다. 이 옵션은 사용자의 기본 umask를 대체합니다. 예는 sshd_config(4) 매뉴얼 페이지의 Subsystem에 대한 설명을 참조하십시오.
- Match 블록에 대한 추가 키워드 - AuthorizedKeysFile, ForceCommand 및 HostbasedUsesNameFromPacketOnly가 Match 블록에서 지원됩니다. 기본적으로 AuthorizedKeysFile의 값은 $HOME/.ssh/authorized_keys이며 HostbasedUsesNameFromPacketOnly는 no입니다. Match 블록을 사용하려면 보안 셸 기본 값에 대한 사용자 및 호스트 예외를 만드는 방법 [15]을 참조하십시오.

Oracle Solaris 엔지니어가 OpenSSH 프로젝트의 버그 수정을 제공합니다. 또한 다음 Oracle Solaris 기능을 보안 셸 포크에 통합했습니다.

- PAM - 보안 셸에 PAM이 사용됩니다. OpenSSH `UsePAM` 구성 옵션은 지원되지 않습니다.
- 권한 구분 - 보안 셸에 OpenSSH 프로젝트의 권한 구분 코드가 사용되지 않습니다. 보안 셸은 감사, 레코드 보관 및 키 갱신에 대한 처리와 세션 프로토콜에 대한 처리를 구분합니다.

  보안 셸 권한 구분 코드는 항상 설정되어 있으며 해제할 수 없습니다. OpenSSH `UsePrivilegeSeparation` 옵션은 지원되지 않습니다.
- 로케일 - 보안 셸은 RFC 4253 *Secure Shell Transfer Protocol*에 정의된 언어 협상을 전체적으로 지원합니다. 사용자가 로그인한 후 사용자의 로그인 셸 프로파일이 보안 셸의 협상된 로케일 설정을 대체할 수 있습니다.
- 감사 - 보안 셸은 Oracle Solaris 감사 서비스에 완전히 통합되었습니다. 감사 서비스에 대한 자세한 내용은 "Oracle Solaris 11.2의 감사 관리 "를 참조하십시오.
- GSS-API 지원 - GSS-API는 사용자 인증 및 초기 키 교환에 사용할 수 있습니다. GSS-API는 RFC4462 *Generic Security Service Application Program Interface*에 정의되어 있습니다.
- 프록시 명령 - 보안 셸은 SOCKS5 및 HTTP 프로토콜에 대한 프록시 명령을 제공합니다. 예는 방화벽 외부의 호스트에 대한 기본 보안 셸 연결 설정 방법 [27]을 참조하십시오.

Oracle Solaris 릴리스에서 보안 셸은 OpenSSH 프로젝트의 `SSH_OLD_FORWARD_ADDR` 호환성 플래그를 다시 동기화합니다.

보안 셸 및 FIPS 140

보안 셸은 OpenSSL FIPS 140 모듈의 소비자입니다. Oracle Solaris는 서버측 및 클라이언트측에 대한 FIPS 140 옵션을 제공합니다. FIPS 140 요구 사항을 준수하기 위해 관리자는 FIPS 140 옵션을 구성하고 사용해야 합니다.

FIPS 모드(보안 셸이 OpenSSL의 FIPS 140 모드를 사용하는 것)는 기본값이 아닙니다. 관리자는 보안 셸이 FIPS 140 모드에서 실행하도록 명시적으로 사용으로 설정해야 합니다. `ssh -o "UseFIPS140 yes"` *remote-host* 명령으로 FIPS 140 모드를 호출할 수 있습니다. 또는 구성 파일에서 키워드를 설정할 수 있습니다.

간단히 말해서 이 구현은 다음과 같은 항목들로 구성됩니다.

- FIPS 140 승인 암호화 `aes128-cbc`, `aes192-cbc`, `aes256-cbc`는 서버 및 클라이언트측에서 사용할 수 있습니다.

  `3des-cbc`는 클라이언트측에 기본적으로 제공되지만 잠재적인 보안 위험 때문에 서버측 암호화로는 제공되지 않습니다.
- 다음과 같은 FIPS 140 승인 MAC(Message Authentication Code)를 사용할 수 있습니다.

  - hmac-sha1, hmac-sha1-96
  - hmac-sha2-256, hmac-sha2-256-96
  - hmac-sha2-512, hmac-sha2-512-96
- 지원되는 4개의 서버-클라이언트 구성은 다음과 같습니다.
  - 클라이언트 또는 서버측의 비FIPS 140 모드
  - 클라이언트 및 서버측 모두의 FIPS 140 모드
  - 서버측의 FIPS 140 모드, 클라이언트측의 비FIPS
  - 서버측의 비FIPS 140 모드, 클라이언트측의 FIPS 모드
- ssh-keygen 명령에는 FIPS 모드의 보안 셸 클라이언트에 필요한 PKCS #8 형식으로 사용자의 개인 키를 생성할 수 있는 옵션이 포함됩니다. 자세한 내용은 ssh-keygen(1) 매뉴얼 페이지를 참조하십시오.

FIPS 140에 대한 자세한 내용은 "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 "을 참조하십시오. 또한 sshd(1M), sshd_config(4), ssh(1), ssh_config(4) 매뉴얼 페이지를 참조하십시오.

보안 셸 작업에 Sun Crypto Accelerator 6000 카드를 사용하면 보안 셸은 레벨 3의 FIPS 140 지원으로 실행됩니다. 레벨 3 하드웨어는 물리적 변조를 방지하고, ID 기반 인증을 사용하고, 하드웨어의 기타 인터페이스에서 중요한 보안 매개변수를 처리하는 인터페이스를 격리할 수 있는 것으로 인증되었습니다.

보안 셸 구성(작업)

보안 셸은 설치 시에 구성됩니다. 기본값을 변경하려면 관리 개입이 필요합니다. 다음 작업은 일부 기본값을 변경하는 방법을 보여줍니다.

보안 셸 구성(작업 맵)

다음 작업 맵에서는 보안 셸 구성 절차에 대해 설명합니다. 보안 셸을 사용하려면 "보안 셸 사용(작업)" [18]을 참조하십시오.

작업	설명	수행 방법
호스트 기반 인증을 구성합니다.	클라이언트와 서버에서 호스트 기반 인증을 구성합니다.	보안 셸에 대한 호스트 기반 인증 설정 방법 [12]
연결 대기 시간을 처리할 수 있도록 버퍼 크기를 늘립니다.	고대역, 대기 시간이 높은 네트워크에 대해 TCP 등록 정보 recv_buf의 값을 높입니다.	"Oracle Solaris 11.2의 TCP/IP 네트워크, IPMP 및 IP 터널 관리 "의 "TCP 수신 버퍼 크기 변경"
포트 전달을 구성합니다.	사용자가 포트 전달을 사용할 수 있도록 합니다.	보안 셸에서 포트 전달을 구성하는 방법 [15]

작업	설명	수행 방법
보안 셸 시스템 기본값에 대한 예외를 구성합니다.	사용자, 호스트, 그룹 및 주소에 대해 시스템 기본값과 다른 보안 셸 값을 지정합니다.	보안 셸 기본값에 대한 사용자 및 호스트 예외를 만드는 방법 [15]
`sftp` 전송에 대한 `root` 환경을 격리시킵니다.	파일 전송을 위한 보호되는 디렉토리를 제공합니다.	`sftp` 파일에 대한 격리된 디렉토리를 만드는 방법 [16]

▼ 보안 셸에 대한 호스트 기반 인증 설정 방법

다음 절차에서는 서버에서 클라이언트의 개인 키가 인증에 사용되는 공개 키 시스템을 설정합니다. 사용자는 공개/개인 키 쌍을 만들어야 합니다.

절차에서 언급되는 클라이언트와 로컬 호스트라는 용어는 사용자가 `ssh` 명령을 입력한 시스템을 나타냅니다. 서버와 원격 호스트라는 용어는 클라이언트가 연결하려고 시도 중인 시스템을 나타냅니다.

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

1. **클라이언트에서 호스트 기반 인증을 사용으로 설정합니다.**

   클라이언트 구성 파일 `/etc/ssh/ssh_config`에서 다음 항목을 입력합니다.

   ```
   HostbasedAuthentication yes
   ```

   파일 구문은 `ssh_config`(4) 매뉴얼 페이지를 참조하십시오.

2. **서버에서 호스트 기반 인증을 사용으로 설정합니다.**

   서버 구성 파일 `/etc/ssh/sshd_config`에서 동일한 항목을 입력합니다.

   ```
   HostbasedAuthentication yes
   ```

   파일 구문은 `sshd_config`(4) 매뉴얼 페이지를 참조하십시오.

3. **서버에서 관리자나 사용자는 클라이언트가 신뢰할 수 있는 호스트로 인식될 수 있게 해주는 파일을 구성합니다.**

   자세한 내용은 `sshd`(1M) 매뉴얼 페이지의 `FILES` 절을 참조하십시오.

   - **관리자가 구성하는 경우 서버의 `/etc/ssh/shosts.equiv` 파일에 클라이언트를 항목으로 추가합니다.**

     client-host

   - **사용자가 구성하는 경우 서버의 `~/.shosts` 파일에 클라이언트에 대한 항목을 추가해야 합니다.**

client-host

4. **서버에서 `sshd` 데몬이 신뢰할 수 있는 호스트 목록에 액세스할 수 있는지 확인합니다.**
   `/etc/ssh/sshd_config` 파일에서 `IgnoreRhosts`를 `no`로 설정합니다.

   ```
   ## sshd_config
   IgnoreRhosts no
   ```

5. **사이트의 보안 셸 사용자에게 두 호스트에 대한 계정이 있는지 확인합니다.**

6. **다음 방법 중 하나를 사용하여 클라이언트의 공개 키를 서버에 배치합니다.**

   ■ **서버에서 `sshd_config` 파일을 수정한 후 사용자에게 `~/.ssh/known_hosts` 파일에 클라이언트의 공개 호스트 키를 추가하도록 합니다.**

   ```
   ## sshd_config
   IgnoreUserKnownHosts no
   ```

   사용자 지침은 보안 셸에서 사용할 공개/개인 키 쌍 생성 방법 [18]을 참조하십시오.

   ■ **서버에 클라이언트의 공개 키를 복사합니다.**
   호스트 키는 `/etc/ssh` 디렉토리에 저장되어 있습니다. 일반적으로 첫번째 부트 시 `sshd` 데몬이 키를 생성합니다.

   a. **서버에서 `/etc/ssh/ssh_known_hosts` 파일에 키를 추가합니다.**
      클라이언트에서 한 행에 백슬래시 없이 다음 명령을 입력합니다.

      ```
      # cat /etc/ssh/ssh_host_dsa_key.pub | ssh RemoteHost \
      'cat >> /etc/ssh/ssh_known_hosts && echo "Host key copied"'
      ```

참고 - 서버에 호스트 키가 없을 경우 보안 셸을 사용하면 다음과 같은 오류 메시지가 생성됩니다.

```
Client and server could not agree on a key exchange algorithm:
client "diffie-hellman-group-exchange-sha256,diffie-hellman-group-
exchange-sha1,diffie-hellman-group14-sha1,diffie-hellman-group1-sha1",
server "gss-group1-sha1-toWM5Slw5Ew8Mqkay+al2g==". Make sure host keys
are present and accessible by the server process. See sshd_config(4)
description of "HostKey" option.
```

---

   b. **프롬프트가 표시되면 로그인 암호를 제공합니다.**
      파일이 복사되면 "Host key copied" 메시지가 표시됩니다.
      `/etc/ssh/ssh_known_hosts` 파일의 각 행은 공백으로 구분된 필드로 구성됩니다.

      hostnames algorithm-name publickey comment

c. **`/etc/ssh/ssh_known_hosts` 파일을 편집하고 복사된 항목에 *RemoteHost*를 첫번째 필드로 추가합니다.**

```
## /etc/ssh/ssh_known_hosts File
RemoteHost  <copied entry>
```

예 1-1 호스트 기반 인증 설정

다음 예에서는 각 호스트가 서버와 클라이언트로 구성됩니다. 각 호스트의 사용자는 다른 호스트에 대한 `ssh` 연결을 시작할 수 있습니다. 다음은 각 호스트를 서버와 클라이언트로 만드는 구성입니다.

- 각 호스트에서 보안 셸 구성 파일에는 다음 항목이 포함되어 있습니다.

```
## /etc/ssh/ssh_config
HostBasedAuthentication yes
#
## /etc/ssh/sshd_config
HostBasedAuthentication yes
IgnoreRhosts no
```

- 각 호스트에서 `shosts.equiv` 파일에는 다른 호스트에 대한 항목이 포함되어 있습니다.

```
## /etc/ssh/shosts.equiv on machine2
machine1

## /etc/ssh/shosts.equiv on machine1
machine2
```

- 각 호스트의 공개 키는 다른 호스트의 `/etc/ssh/ssh_known_hosts` 파일에 있습니다.

```
## /etc/ssh/ssh_known_hosts on machine2
… machine1

## /etc/ssh/ssh_known_hosts on machine1
… machine2
```

- 사용자는 두 호스트 모두에 대한 계정을 가집니다. 예를 들어, 사용자 John Doe에 대해 다음 정보가 나타납니다.

```
## /etc/passwd on machine1
jdoe:x:3111:10:J Doe:/home/jdoe:/bin/sh

## /etc/passwd on machine2
jdoe:x:3111:10:J Doe:/home/jdoe:/bin/sh
```

▼ 보안 셸에서 포트 전달을 구성하는 방법

포트 전달은 로컬 포트가 원격 호스트로 전달될 수 있도록 합니다. 소켓이 로컬측의 포트를 수신 대기하도록 효과적으로 할당됩니다. 마찬가지로 원격측에서도 포트를 지정할 수 있습니다.

참고 - 보안 셸 포트 전달에는 TCP 연결이 사용되어야 합니다. 보안 셸은 포트 전달을 위해 UDP 연결을 지원하지 않습니다.

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

1. **포트 전달을 허용하도록 원격 서버의 보안 셸 설정을 구성합니다.**

   `/etc/ssh/sshd_config` 파일에서 `AllowTcpForwarding`의 값을 `yes`로 변경합니다.

   ```
   # Port forwarding
   AllowTcpForwarding yes
   ```

2. **보안 셸 서비스를 다시 시작합니다.**

   ```
   remoteHost# svcadm restart network/ssh:default
   ```

   지속 서비스 관리에 대한 자세한 내용은 "Oracle Solaris 11.2의 시스템 서비스 관리"의 1장, "서비스 관리 기능 소개" 및 `svcadm`(1M) 매뉴얼 페이지를 참조하십시오.

3. **포트 전달을 사용할 수 있는지 확인합니다.**

   ```
   remoteHost# /usr/bin/pgrep -lf sshd
   1296 ssh -L 2001:remoteHost:23 remoteHost
   ```

▼ 보안 셸 기본값에 대한 사용자 및 호스트 예외를 만드는 방법

이 절차에서는 `/etc/ssh/sshd_config` 파일의 전역 절 뒤에 조건부 `Match` 블록을 추가합니다. `Match` 블록 뒤의 키워드/값 쌍은 일치 항목으로 지정된 사용자, 그룹, 호스트 또는 주소에 대한 예외를 지정합니다.

시작하기 전에 `solaris.admin.edit/etc/ssh/sshd_config` 권한 부여가 지정된 관리자여야 합니다. 기본적으로 `root` 역할에는 이 권한 부여가 있습니다. 자세한 내용은 "Oracle Solaris 11.2의 사용자 및 프로세스 보안"의 "지정된 관리 권한 사용"을 참조하십시오.

1. **편집을 위해 `/etc/ssh/sshd_config` 파일을 엽니다.**

```
# pfedit /etc/ssh/sshd_config
```

2. **기본 설정 이외의 다른 보안 셸 설정을 사용하도록 사용자, 그룹, 호스트 또는 주소를 구성합니다.**

   `Match` 블록을 전역 설정 뒤에 배치합니다.

   참고 - 파일의 전역 섹션에 항상 기본 설정이 나열되는 것은 아닙니다. 기본값은 `sshd_config`(4) 매뉴얼 페이지를 참조하십시오.

   예를 들어, TCP 전달을 사용하도록 허용하지 않을 사용자가 있을 수도 있습니다. 다음 예에서는 `public` 그룹의 사용자 및 이름이 `test`로 시작하는 사용자가 TCP 전달을 사용할 수 없습니다.

```
## sshd_config file
## Global settings

# Example (reflects default settings):
#
# Host *
#   ForwardAgent no
#   ForwardX11 no
#   PubkeyAuthentication yes
#   PasswordAuthentication yes
#   FallBackToRsh no
#   UseRsh no
#   BatchMode no
#   CheckHostIP yes
#   StrictHostKeyChecking ask
#   EscapeChar ~
Match Group public
AllowTcpForwarding no
Match User test*
AllowTcpForwarding no
```

   `Match` 블록 구문에 대한 자세한 내용은 `sshd_config`(4) 매뉴얼 페이지를 참조하십시오.

▼ sftp 파일에 대한 격리된 디렉토리를 만드는 방법

이 절차에서는 특히 `sftp` 전송을 위해 만들어진 `sftponly` 디렉토리를 구성합니다. 사용자는 전송 디렉토리 외부의 파일 또는 디렉토리를 볼 수 없습니다.

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

1. **보안 셸 서버에서 격리된 디렉토리를 `chroot` 환경으로 만듭니다.**

```
# groupadd sftp
# useradd -m -G sftp -s /bin/false sftponly
# chown root:root /export/home/sftponly
# mkdir /export/home/sftponly/WWW
# chown sftponly:staff /export/home/sftponly/WWW
```

이 구성에서 /export/home/sftonly는 root 계정만 액세스할 수 있는 chroot 디렉토리입니다. 사용자에게 sftponly/WWW 하위 디렉토리에 대한 쓰기 권한이 있습니다.

2. **계속해서 서버에서 sftp 그룹에 대한 Match 블록을 구성합니다.**

   /etc/ssh/sshd_config 파일에서 sftp subsystem 항목을 찾고 파일을 다음과 같이 수정합니다.

```
# pfedit /etc/ssh/sshd_config
...
# sftp subsystem
#Subsystem       sftp    /usr/lib/ssh/sftp-server
Subsystem       sftp    internal-sftp
...
## Match Group for Subsystem
## At end of file, to follow all global options
Match Group sftp
ChrootDirectory %h
ForceCommand internal-sftp
AllowTcpForwarding no
```

   다음 변수를 사용하여 chroot 경로를 지정할 수 있습니다.

   - %h - 홈 디렉토리를 지정합니다.
   - %u - 인증된 사용자의 사용자 이름을 지정합니다.
   - %% - % 기호를 이스케이프 처리합니다.

3. **클라이언트에서 구성이 올바르게 작동하는지 확인합니다.**

   chroot 환경의 파일이 다를 수 있습니다.

```
root@client:~# ssh sftponly@server
This service allows sftp connections only.
Connection to server closed.       No shell access, sftp is enforced.
root@client:~# sftp sftponly@server
sftp> pwd       sftp access granted
Remote working directory: /      chroot directory looks like root directory
sftp> ls
WWW             local.cshrc     local.login     local.profile
sftp> get local.cshrc
Fetching /local.cshrc to local.cshrc
/local.cshrc    100%  166     0.2KB/s   00:00     user can read contents
sftp> put /etc/motd
Uploading /etc/motd to /motd
Couldn't get handle: Permission denied     user cannot write to / directory
sftp> cd WWW
```

```
sftp> put /etc/motd
Uploading /etc/motd to /WWW/motd
/etc/motd      100%  118     0.1KB/s   00:00      user can write to WWW directory
sftp> ls -l
-rw-r--r--    1 101  10    118 Jul 20 09:07 motd      successful transfer
sftp>
```

보안 셸 사용(작업)

이 절에서는 사용자가 보안 셸을 익힐 수 있는 절차가 제공됩니다.

보안 셸 사용(작업 맵)

다음 작업 맵에서는 사용자의 보안 셸 사용 절차에 대해 설명합니다.

작업	설명	수행 방법
공개/개인 키 쌍을 만듭니다.	공개 키 인증이 필요한 사이트의 보안 셸에 대한 액세스를 사용으로 설정합니다.	보안 셸에서 사용할 공개/개인 키 쌍 생성 방법 [18]
문장암호를 변경합니다.	개인 키를 인증하는 구문을 변경합니다.	보안 셸 개인 키에 대한 문장암호 변경 방법 [20]
보안 셸을 사용하여 로그인합니다.	원격으로 로그인할 때는 암호화된 보안 셸 통신을 제공합니다.	보안 셸을 사용하여 원격 호스트에 로그인하는 방법 [21]
암호를 입력하지 않고 보안 셸에 로그인합니다.	보안 셸에 사용자 암호를 제공하는 에이전트를 사용하여 로그인할 수 있도록 합니다.	보안 셸에서 암호 프롬프트를 줄이는 방법 [22]
보안 셸에 root로 로그인합니다.	ZFS send 및 receive 명령에 대해 root로 로그인을 사용으로 설정합니다.	보안 셸을 사용하여 ZFS를 원격으로 관리하는 방법 [23]
보안 셸에서 포트 전달을 사용합니다.	TCP를 통한 보안 셸 연결에서 사용할 로컬 포트 또는 원격 포트를 지정합니다.	보안 셸에서 포트 전달을 사용하는 방법 [25]
보안 셸을 사용하여 파일을 복사합니다.	호스트 간에 안전하게 파일을 복사합니다.	보안 셸을 사용하여 파일을 복사하는 방법 [26]
방화벽 내부의 호스트에서 방화벽 외부의 호스트에 안전하게 연결합니다.	HTTP 또는 SOCKS5와 호환되는 보안 셸 명령을 사용하여 방화벽으로 분리된 호스트를 연결합니다.	방화벽 외부의 호스트에 대한 기본 보안 셸 연결 설정 방법 [27]

▼ 보안 셸에서 사용할 공개/개인 키 쌍 생성 방법

사용자 사이트에서 호스트 기반 인증 또는 사용자 공개 키 인증을 구현한 경우 사용자는 공개/개인 키 쌍을 생성해야 합니다. 추가 옵션은 ssh-keygen(1) 매뉴얼 페이지를 참조하십시오.

시작하기 전에 호스트 기반 인증이 구성되었는지 여부를 시스템 관리자에게 문의합니다.

1. **키 생성 프로그램을 시작합니다.**

```
mySystem% ssh-keygen -t rsa
Generating public/private rsa key pair.
…
```

여기서 -t는 알고리즘 유형으로, rsa, dsa 또는 rsa1입니다.

2. **키를 보관할 파일의 경로를 지정합니다.**
기본적으로 RSA v2 키를 나타내는 파일 이름 id_rsa가 괄호 안에 표시됩니다. Return 키를 눌러 이 파일을 선택하거나 다른 파일 이름을 제공할 수 있습니다.

```
Enter file in which to save the key (/home/username/.ssh/id_rsa):     <Press Return>
```

.pub 문자열을 개인 키 파일 이름에 추가하면 공개 키의 파일 이름이 자동으로 만들어집니다.

3. **키를 사용하는 데 필요한 문장암호를 입력합니다.**
이 문장암호는 개인 키를 암호화하는 데 사용됩니다. 널 항목은 사용하지 않는 것이 좋습니다. 문장암호는 입력할 때 표시되지 않습니다.

```
Enter passphrase (empty for no passphrase):     <Type passphrase>
```

4. **확인용으로 문장암호를 다시 입력합니다.**

```
Enter same passphrase again:     <Type passphrase>
Your identification has been saved in /home/username/.ssh/id_rsa.
Your public key has been saved in /home/username/.ssh/id_rsa.pub.
The key fingerprint is:
0e:fb:3d:57:71:73:bf:58:b8:eb:f3:a3:aa:df:e0:d1 username@my

System
```

5. **키 파일의 경로가 올바른지 확인합니다.**

```
% ls ~/.ssh
id_rsa
id_rsa.pub
```

이 단계에서는 공개/개인 키 쌍이 만들어져 있습니다.

6. **네트워크의 인증 방법에 따라 적절한 옵션을 사용하여 원격 호스트에 로그인합니다.**

- **관리자가 호스트 기반 인증을 구성한 경우 로컬 호스트의 공개 키를 원격 호스트에 복사해야 할 수도 있습니다.**
이제 원격 호스트에 로그인할 수 있습니다. 자세한 내용은 보안 셸을 사용하여 원격 호스트에 로그인하는 방법 [21]을 참조하십시오.

   a. **한 행에 백슬래시 없이 다음 명령을 입력합니다.**

```
% cat /etc/ssh/ssh_host_dsa_key.pub | ssh RemoteHost \
'cat >> ~./ssh/known_hosts && echo "Host key copied"'
```

b. **프롬프트가 표시되면 로그인 암호를 제공합니다.**

```
Enter password:       <Type password>
Host key copied
%
```

■ **사이트에서 공개 키를 통한 사용자 인증을 사용하는 경우 원격 호스트에서 `authorized_keys` 파일을 채웁니다.**

a. **공개 키를 원격 호스트에 복사합니다.**

한 행에 백슬래시 없이 다음 명령을 입력합니다.

```
mySystem% cat $HOME/.ssh/id_rsa.pub | ssh myRemoteHost \
'cat >> .ssh/authorized_keys && echo "Key copied"'
```

파일이 복사되면 "Key copied" 메시지가 표시됩니다.

b. **프롬프트가 표시되면 로그인 암호를 제공합니다.**

```
Enter password:       Type login password
Key copied
mySystem%
```

7. **(옵션) 앞으로는 문장암호를 묻지 않습니다.**

보안 셸에서 암호 프롬프트를 줄이는 방법 [22]을 참조하십시오. 자세한 내용은 `ssh-agent`(1) 및 `ssh-add`(1) 매뉴얼 페이지를 참조하십시오.

▼ 보안 셸 개인 키에 대한 문장암호 변경 방법

다음 명령에서는 개인 키에 대한 인증 방식인 문장암호를 변경합니다. 자세한 내용은 `ssh-keygen`(1) 매뉴얼 페이지를 참조하십시오.

● **문장암호를 변경합니다.**

-p 옵션을 사용하여 `ssh-keygen` 명령을 입력하고 프롬프트에 응답합니다.

```
mySystem% ssh-keygen -p
Enter file which contains the private key
      (/home/username/.ssh/id_rsa):     <Press Return>
Enter passphrase
      (empty for no passphrase):      <Type passphrase>
Enter same passphrase again:        <Type passphrase>
```

여기서 -p는 개인 키 파일의 문장암호 변경을 요청합니다.

▼ 보안 셸을 사용하여 원격 호스트에 로그인하는 방법

1. **보안 셸 세션을 시작합니다.**
   ssh 명령을 입력하고 원격 호스트의 이름 및 로그인을 지정합니다.

   ```
   mySystem% ssh myRemoteHost -l username
   ```

2. **프롬프트가 표시되면 원격 호스트 키의 신뢰성을 확인합니다.**
   원격 호스트의 신뢰성을 묻는 질문이 표시될 수 있습니다.

   ```
   The authenticity of host 'myRemoteHost' can't be established.
   RSA key fingerprint in md5 is: 04:9f:bd:fc:3d:3e:d2:e7:49:fd:6e:18:4f:9c:26
   Are you sure you want to continue connecting(yes/no)?
   ```

   일반적으로 이 프롬프트는 원격 호스트에 대한 초기 연결에 표시됩니다.

   - **원격 호스트의 신뢰성을 확인할 수 없을 경우 no를 입력하고 시스템 관리자에게 문의하십시오.**

     ```
     Are you sure you want to continue connecting(yes/no)? no
     ```

     관리자가 전역 /etc/ssh/ssh_known_hosts 파일을 업데이트합니다. 업데이트된 ssh_known_hosts 파일은 이 프롬프트가 표시되지 않도록 합니다.

   - **원격 호스트의 신뢰성을 확인한 경우 프롬프트에 응답하고 다음 단계를 계속합니다.**

     ```
     Are you sure you want to continue connecting(yes/no)? yes
     ```

3. **보안 셸에 대해 자신을 인증합니다.**

   a. **프롬프트가 표시되면 문장암호를 입력합니다.**

   ```
   Enter passphrase for key '/home/username/.ssh/id_rsa':      <Type passphrase>
   ```

   b. **프롬프트가 표시되면 계정 암호를 입력합니다.**

   ```
   username@myRemoteHost's password:      <Type password>
   Last login: Wed Sep  7 09:07:49 2011 from myLocalHost
   Oracle Corporation      SunOS 5.11       September 2011
   myRemoteHost%
   ```

4. **원격 호스트에서 트랜잭션을 수행합니다.**
   보낸 명령이 암호화되고, 수신한 응답이 암호화됩니다.

5. **보안 셸 연결을 해제합니다.**

   완료되면 **exit**를 입력하거나 일반적인 셸 종료 방법을 사용합니다.

```
myRemoteHost% exit
myRemoteHost% logout
Connection to myRemoteHost closed
mySystem%
```

예 1-2 보안 셸에 원격 GUI 표시

이 예에서 jdoe는 두 시스템 모두에서 초기 사용자이며 Software Installation 권한 프로 파일이 지정됩니다. jdoe는 원격 시스템에서 패키지 관리자 GUI를 사용하려고 합니다. X11Forwarding 키워드의 기본값이 계속 yes이고 xauth 패키지가 원격 시스템에 설치됩니다.

```
% ssh -l jdoe -X myRemoteHost
jdoe@myRemoteHost's password: password
Last login: Wed Sep  7 09:07:49 2011 from myLocalHost
Oracle Corporation      SunOS 5.11       September 2011
myRemoteHost% packagemanager &
```

▼ 보안 셸에서 암호 프롬프트를 줄이는 방법

암호문 및 암호를 입력하지 않고 보안 셸을 사용하려는 경우 에이전트 데몬을 사용할 수 있습니다. 호스트마다 계정이 다른 경우 세션에 필요한 키를 추가합니다.

필요한 경우 다음 절차의 설명에 따라 수동으로 에이전트 데몬을 시작할 수 있습니다.

1. **에이전트 데몬을 시작합니다.**

```
mySystem% eval `ssh-agent`
Agent pid 9892
```

2. **에이전트 데몬이 시작되었는지 확인합니다.**

```
mySystem% pgrep ssh-agent
9892
```

3. **에이전트 데몬에 개인 키를 추가합니다.**

```
mySystem% ssh-add
Enter passphrase for /home/username/.ssh/id_rsa:      <Type passphrase>
Identity added: /home/username/.ssh/id_rsa(/home/username/.ssh/id_rsa)
mySystem%
```

4. **보안 셸 세션을 시작합니다.**

```
mySystem% ssh myRemoteHost -l username
```

문장암호 프롬프트가 표시되지 않습니다.

예 1-3 ssh-add 옵션 사용

이 예에서는 jdoe가 에이전트 데몬에 두 개의 키를 추가합니다. -l 옵션이 데몬에 저장된 모든 키를 나열하는 데 사용됩니다. 세션 종료 시 -D 옵션이 에이전트 데몬에서 모든 키를 제거하는 데 사용됩니다.

```
myLocalHost% ssh-agent
mySystem% ssh-add
Enter passphrase for /home/jdoe/.ssh/id_rsa:      <Type passphrase>
Identity added: /home/jdoe/.ssh/id_rsa(/home/jdoe/.ssh/id_rsa)
mySystem% ssh-add /home/jdoe/.ssh/id_dsa
Enter passphrase for /home/jdoe/.ssh/id_dsa:      <Type passphrase>
Identity added:
/home/jdoe/.ssh/id_dsa(/home/jdoe/.ssh/id_dsa)

mySystem% ssh-add -l
md5 1024 0e:fb:3d:53:71:77:bf:57:b8:eb:f7:a7:aa:df:e0:d1
/home/jdoe/.ssh/id_rsa(RSA)
md5 1024 c1:d3:21:5e:40:60:c5:73:d8:87:09:3a:fa:5f:32:53
/home/jdoe/.ssh/id_dsa(DSA)

    User conducts Oracle Solaris Secure Shell transactions

myLocalHost% ssh-add -D
Identity removed:
/home/jdoe/.ssh/id_rsa(/home/jdoe/.ssh/id_rsa.pub)
/home/jdoe/.ssh/id_dsa(DSA)
```

▼ 보안 셸을 사용하여 ZFS를 원격으로 관리하는 방법

기본적으로 root 역할은 보안 셸에 원격으로 로그인할 수 없습니다. 이전까지 루트는 원격 시스템의 저장소에 ZFS 풀 데이터를 보내는 것과 같은 중요한 작업에 보안 셸을 사용했습니다. 이 절차에서 root 역할은 원격 ZFS 관리자로 작업할 수 있는 사용자를 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. **두 시스템 모두에서 사용자를 만듭니다.**

   예를 들어, zfsroot 사용자를 만들고 암호를 제공합니다.

```
source # useradd -c "Remote ZFS Administrator" -u 1201 -d /home/zfsroot zfsroot
```

```
source # passwd zfsroot
Enter password:
Retype password:
#

dest # useradd -c "Remote ZFS Administrator" -u 1201 -d /home/zfsroot zfsroot
dest # passwd zfsroot
...
```

zfsroot 사용자는 두 시스템에서 모두 동일하게 정의되어야 합니다.

2. **보안 셸 인증에 대한 사용자 키 쌍을 만듭니다.**

   키 쌍은 소스 시스템에 만들어집니다. 그런 후 공개 키가 대상 시스템에서 zfsroot 사용자에 복사됩니다.

   a. **키 쌍을 생성하고 이를 id_migrate 파일에 배치합니다.**

   ```
   # ssh-keygen -t rsa -P "" -f ~/id_migrate
   Generating public/private rsa key pair.
   Your identification has been saved in /root/id_migrate.
   Your public key has been saved in /root/id_migrate.pub.
   The key fingerprint is:
   3c:7f:40:ef:ec:63:95:b9:23:a2:72:d5:ea:d1:61:f0 root@source
   ```

   b. **키 쌍의 공용 부분을 대상 시스템에 보냅니다.**

   ```
   # scp ~/id_migrate.pub zfsroot@dest:
   The authenticity of host 'dest (10.134.76.126)' can't be established.
   RSA key fingerprint is 44:37:ab:4e:b7:2f:2f:b8:5f:98:9d:e9:ed:6d:46:80.
   Are you sure you want to continue connecting (yes/no)? yes
   Warning: Permanently added 'dest,10.134.76.126' (RSA) to the list of known hosts.
   Password:
   id_migrate.pub 100% |*****************************| 399 00:00
   ```

3. **두 시스템 모두에서 ZFS 파일 관리 권한 프로파일을 zfsroot에 지정합니다.**

   ```
   source # usermod -P +'ZFS File System Management' -S files zfsroot
   dest # usermod -P +'ZFS File System Management' -S files zfsroot
   ```

4. **대상 시스템에 권한 프로파일이 지정되었는지 확인합니다.**

   ```
   dest # profiles zfsroot
   zfsroot:
   ZFS File System Management
   Basic Solaris User
   All
   ```

5. **대상 시스템에서 ,키 쌍의 공용 부분을 전용 /home/zfsroot/.ssh 디렉토리로 이동합니다.**

   ```
   root@dest # su - zfsroot
   Oracle Corporation      SunOS 5.11      11.1    May 2012
   zfsroot@dest $ mkdir -m 700 .ssh
   ```

```
zfsroot@dest $ cat id_migrate.pub >> .ssh/authorized_keys
```

6. **구성이 작동하는지 확인합니다.**

```
root@source# ssh -l zfsroot -i ~/id_migrate dest \
pfexec /usr/sbin/zfs snapshot zones@test
root@source# ssh -l zfsroot -i ~/id_migrate dest \
pfexec /usr/sbin/zfs destroy zones@test
```

7. **(옵션) 스냅샷을 만들고 데이터를 복제할 수 있는지 확인합니다.**

```
root@source# zfs snapshot -r rpool/zones@migrate-all
root@source#  zfs send -rc rpool/zones@migrate-all | \
ssh -l zfsroot -i ~/id_migrate dest pfexec /usr/sbin/zfs recv -F zones
```

8. **(옵션) ZFS 관리를 위해 `zfsroot` 계정을 사용할 수 있는 기능을 제거합니다.**

```
root@dest# usermod -P -'ZFS File System Management' zfsroot
root@dest#  su - zfsroot
zfsroot@dest#  cp .ssh/authorized_keys .ssh/authorized_keys.bak
zfsroot@dest#  grep -v root@source .ssh/authorized_keys.bak> .ssh/authorized_keys
```

▼ 보안 셸에서 포트 전달을 사용하는 방법

로컬 포트가 원격 호스트에 전달되도록 지정할 수 있습니다. 소켓이 로컬측의 포트를 수신 대기하도록 효과적으로 할당됩니다. 보안 채널을 통해 이 포트에서 원격 호스트로의 연결이 설정됩니다. 예를 들어, IMAP4를 사용하여 원격으로 전자 메일을 얻기 위해 포트 143을 지정할 수 있습니다. 마찬가지로 원격측에서도 포트를 지정할 수 있습니다.

시작하기 전에 포트 전달을 사용하려면 관리자가 원격 보안 셸 서버에서 포트 전달을 사용으로 설정했어야 합니다. 자세한 내용은 보안 셸에서 포트 전달을 구성하는 방법 [15]을 참조하십시오.

- **원격 포트에서 로컬 포트로 또는 로컬 포트에서 원격 포트로 보안 포트 전달을 설정합니다.**

  - **원격 포트에서 보안 통신을 수신하도록 로컬 포트를 설정하려면 두 포트를 지정합니다.**

    원격 통신을 수신 대기하는 로컬 포트를 지정합니다. 또한 통신을 전달하는 원격 호스트 및 원격 포트를 지정합니다.

    ```
    mySystem% ssh -L localPort:remoteHost:remotePort
    ```

  - **로컬 포트에서 보안 연결을 수신하도록 원격 포트를 설정하려면 두 포트를 지정합니다.**

    원격 통신을 수신 대기하는 원격 포트를 지정합니다. 또한 통신을 전달하는 로컬 호스트 및 로컬 포트를 지정합니다.

    ```
    mySystem% ssh -R remotePort:localhost:localPort
    ```

예 1-4 로컬 포트 전달을 사용하여 메일 수신

다음 예에서는 로컬 포트 전달을 사용하여 원격 서버에서 안전하게 메일을 수신할 수 있는 방법을 보여 줍니다.

```
myLocalHost% ssh -L 9143:myRemoteHost:143 myRemoteHost
```

이 명령은 myLocalHost의 포트 9143에서 포트 143으로 연결을 전달합니다. 포트 143은 myRemoteHost의 IMAP v2 서버 포트입니다. 메일 응용 프로그램을 실행한 사용자는 localhost:9143에서처럼 IMAP 서버에 대한 로컬 포트 번호를 지정합니다.

예 1-5 원격 포트 전달을 사용하여 방화벽 외부에서 통신

이 예에서는 기업 환경의 사용자가 외부 네트워크의 호스트에서 회사 방화벽 내부의 호스트로 연결을 전달할 수 있는 방법을 보여 줍니다.

```
myLocalHost% ssh -R 9022:myLocalHost:22myOutsideHost
```

이 명령은 myOutsideHost의 포트 9022에서 로컬 호스트의 포트 22 sshd 서버로 연결을 전달합니다.

```
myOutsideHost% ssh -p 9022 localhost
myLocalHost%
```

▼ 보안 셸을 사용하여 파일을 복사하는 방법

다음 절차에서는 scp 명령을 사용하여 호스트 간에 암호화된 파일을 복사하는 방법을 보여 줍니다. 로컬 호스트와 원격 호스트 간 또는 두 원격 호스트 간에 암호화된 파일을 복사할 수 있습니다. scp 명령은 인증 프롬프트를 표시합니다. 자세한 내용은 "Oracle Solaris 11.2의 원격 시스템 관리 "의 "scp 명령을 사용한 원격 복사" 및 scp(1) 매뉴얼 페이지를 참조하십시오.

또한 sftp 보안 파일 전송 프로그램을 사용할 수 있습니다. 자세한 내용은 sftp(1) 매뉴얼 페이지를 참조하십시오. 예는 예 1-6. "sftp 명령 사용 시 포트 지정" 및 "Oracle Solaris 11.2의 원격 시스템 관리 "의 "원격 시스템에 로그인하여 파일 복사(sftp)"을 참조하십시오.

참고 - 감사 서비스는 ft 감사 클래스를 통해 sftp 트랜잭션을 감사할 수 있습니다. scp의 경우 감사 서비스는 ssh 세션에 대한 액세스 및 종료를 감사할 수 있습니다. 자세한 내용은 "Oracle Solaris 11.2의 감사 관리 "의 "FTP 및 SFTP 파일 전송을 감사하는 방법"을 참조하십시오.

---

1. **보안 복사 프로그램을 시작합니다.**
   소스 파일, 원격 대상의 사용자 이름 및 대상 디렉토리를 지정합니다.

```
mySystem% scp myfile.1 username@myRemoteHost:~
```

2. **프롬프트가 표시되면 문장암호를 제공합니다.**

```
Enter passphrase for key '/home/username/.ssh/id_rsa':      <Type passphrase>
myfile.1       25% |*******                     |    640 KB  0:20 ETA
myfile.1
```

출력의 두번째 줄에 표시된 대로 문장암호를 입력하면 진행 상황이 표시됩니다. 다음과 같이 진행 상황이 표시됩니다.

- 파일 이름
- 파일 전송 백분율
- 파일 전송 백분율을 나타내는 일련의 별표
- 데이터 전송량
- 전체 파일의 예상 도착 시간(ETA), 즉 남은 시간

예 1-6 `sftp` 명령 사용 시 포트 지정

이 예에서는 사용자가 `sftp` 명령으로 특정 포트를 사용하려고 합니다. 사용자는 `-o` 옵션을 사용하여 포트를 지정합니다.

```
% sftp -o port=2222 guest@RemoteFileServer
```

▼ 방화벽 외부의 호스트에 대한 기본 보안 셸 연결 설정 방법

보안 셸을 사용하여 방화벽 내부의 호스트에서 방화벽 외부의 호스트로 연결을 설정할 수 있습니다. 이 작업을 수행하려면 구성 파일에서 또는 명령줄 옵션으로 `ssh`에 대한 프록시 명령을 지정합니다. 명령줄 옵션은 예 1-7. “보안 셸 명령줄에서 방화벽 외부의 호스트에 연결”을 참조하십시오.

개인의 고유한 구성 파일 `~/.ssh/config`를 통해 `ssh` 상호 작용을 사용자 정의하거나 관리 구성 파일 `/etc/ssh/ssh_config`의 설정을 사용할 수 있습니다.

두 가지 유형의 프록시 명령으로 파일을 사용자 정의할 수 있습니다. 프록시 명령 중 하나는 HTTP 연결에 사용되며, 나머지 하나는 SOCKS5 연결에 사용됩니다. 자세한 내용은 `ssh_config`(4) 매뉴얼 페이지를 참조하십시오.

1. **구성 파일에서 프록시 명령 및 호스트를 지정합니다.**

다음 구문을 사용하여 필요에 따라 행을 여러 개 추가합니다.

```
[Host outside-host]
```

```
ProxyCommand proxy-command [-h proxy-server] \
[-p proxy-port] outside-host|%h outside-port|%p
```

`Host` *outside-host*

명령줄에서 원격 호스트 이름이 지정된 경우 프록시 명령 지정을 인스턴스로 제한합니다. *outside-host*에 와일드카드를 사용하면 일련의 호스트에 프록시 명령 지정이 적용됩니다.

proxy-command

프록시 명령을 지정합니다.

명령은 다음 중 하나일 수 있습니다.

- HTTP 연결의 경우 `/usr/lib/ssh/ssh-http-proxy-connect`
- SOCKS5 연결의 경우 `/usr/lib/ssh/ssh-socks5-proxy-connect`

`-h` *proxy-server* 및 `-p` *proxy-port*

해당 옵션은 각각 프록시 서버와 프록시 포트를 지정합니다. 있을 경우 프록시는 프록시 서버 및 프록시 포트를 지정하는 환경 변수(예: `HTTPPROXY`, `HTTPPROXYPORT`, `SOCKS5_PORT`, `SOCKS5_SERVER` 및 `http_proxy`)를 대체합니다. `http_proxy` 변수는 URL을 지정합니다. 옵션이 사용되지 않을 경우 관련 환경 변수를 설정해야 합니다. 자세한 내용은 `ssh-socks5-proxy-connect`(1) 및 `ssh-http-proxy-connect`(1) 매뉴얼 페이지를 참조하십시오.

outside-host

연결할 특정 호스트를 지정합니다. 명령줄에서 호스트를 지정하려면 `%h` 대체 인수를 사용합니다.

outside-port

연결할 특정 포트를 지정합니다. 명령줄에서 포트를 지정하려면 `%p` 대체 인수를 사용합니다. `Host` *outside-host* 옵션을 사용하지 않고 `%h` 및 `%p`를 지정하면 `ssh` 명령이 호출될 때마다 호스트 인수에 프록시 명령이 적용됩니다.

2. **외부 호스트를 지정하여 보안 셸을 실행합니다.**
예를 들면 다음과 같습니다.

```
mySystem% ssh myOutsideHost
```

이 명령은 개인 구성 파일에서 `myOutsideHost`에 대한 프록시 명령 지정을 검색합니다. 지정을 찾을 수 없을 경우 명령은 시스템 차원의 구성 파일 `/etc/ssh/ssh_config` 에서 찾습니다. `ssh` 명령이 프록시 명령으로 대체됩니다.

예 1-7 보안 셸 명령줄에서 방화벽 외부의 호스트에 연결

구성 파일에서 프록시 명령을 지정하는 방법은 방화벽 외부의 호스트에 대한 기본 보안 셸 연결 설정 방법 [27]에서 설명됩니다. 이 예에서는 `ssh` 명령줄에서 프록시 명령이 지정됩니다.

```
% ssh -o'Proxycommand=/usr/lib/ssh/ssh-http-proxy-connect \
-h myProxyServer -p 8080 myOutsideHost 22' myOutsideHost
```

ssh 명령에 대한 -o 옵션은 명령줄에서 프록시 명령을 지정하는 방법을 제공합니다. 이 예의 명령은 다음을 수행합니다.

- ssh를 HTTP 프록시 명령으로 대체합니다.
- 포트 8080 및 myProxyServer를 프록시 서버로 사용합니다.
- myOutsideHost의 포트 22에 연결합니다.

♦ ♦ ♦ 2 장

2 보안 셸 참조

이 장에서는 Oracle Solaris 보안 셸 기능의 구성 옵션에 대해 설명하고 다음 항목을 다룹니다.

보안 셸 구성 절차는 1장. 보안 셸 사용(작업)을 참조하십시오.

일반 보안 셸 세션

일반적으로 보안 셸 데몬(sshd)은 부트 시 네트워크 서비스가 시작될 때 시작됩니다. 데몬은 클라이언트로부터의 연결을 수신 대기합니다. 보안 셸 세션은 사용자가 ssh, scp 또는 sftp 명령을 실행할 때 시작됩니다. 새 sshd 데몬은 각 수신 연결에 대해 포크됩니다. 포크된 데몬은 키 교환, 암호화, 인증, 명령 실행 및 클라이언트와의 데이터 교환을 처리합니다. 이러한 세션 특성은 클라이언트측 구성 파일 및 서버측 구성 파일로 결정됩니다. 명령줄 인수는 구성 파일의 설정을 대체할 수 있습니다.

클라이언트와 서버는 상호 간에 자체적으로 인증되어야 합니다. 인증에 성공하면 사용자가 원격으로 명령을 실행하고 호스트 간에 데이터를 복사할 수 있습니다.

보안 셸의 세션 특성

sshd 데몬의 서버측 동작은 /etc/ssh/sshd_config 파일의 키워드 설정으로 제어됩니다. 예를 들어, sshd_config 파일은 서버에 대한 액세스가 허용되는 인증 유형을 제어합니다. sshd 데몬이 시작된 경우 명령줄 옵션으로도 서버측 동작을 제어할 수 있습니다.

클라이언트측 동작은 다음 우선 순위에 따라 보안 셸 키워드로 제어됩니다.

- 명령줄 옵션

- 사용자의 구성 파일 `~/.ssh/config`
- 시스템 차원의 구성 파일 `/etc/ssh/ssh_config`

예를 들어, 사용자는 명령줄에서 `-c aes256-ctr,aes128-ctr,arcfour`를 지정하여 `aes128-ctr`보다 우선하는 시스템 차원의 구성 `Ciphers` 설정을 대체할 수 있습니다. 그러면 첫번째 암호 `aes256-ctr`이 우선합니다.

보안 셸의 인증 및 키 교환

보안 셸 프로토콜은 클라이언트 사용자/호스트 인증 및 서버 호스트 인증을 지원합니다. 보안 셸 세션을 보호하기 위해 암호화 키가 교환됩니다. 보안 셸은 다양한 인증 및 키 교환 방법을 제공합니다. 일부 방법은 선택 사항입니다. 클라이언트 인증 방식은 표 1-1, “보안 셸 인증 방법”에 나와 있습니다. 서버는 알려진 호스트 공개 키를 사용하여 인증됩니다.

인증의 경우 보안 셸은 사용자 인증 및 주로 암호가 사용되는 일반적인 대화식 인증을 지원합니다. 또한 보안 셸은 사용자 공개 키 및 신뢰할 수 있는 호스트 공개 키를 통한 인증을 지원합니다. 키는 RSA 또는 DSA일 수 있습니다. 세션 키는 서버 인증 단계에서 서명된 일시적인 Diffie-Hellman 키 교환으로 구성됩니다. 또한 보안 셸은 인증에 GSS 자격 증명을 사용할 수 있습니다.

보안 셸에서 GSS 자격 증명 취득

보안 셸에서 인증에 GSS-API를 사용하려면 서버에 GSS-API 승인자 자격 증명이 있고 클라이언트에 GSS-API 개시자 자격 증명이 있어야 합니다. `mech_dh` 및 `mech_krb5`에 대한 지원이 제공됩니다.

`mech_dh`의 경우 `root`가 `keylogin` 명령을 실행했으면 서버에 GSS-API 자격 증명이 있는 것입니다.

`mech_krb5`의 경우 서버에 해당하는 호스트 주체의 `/etc/krb5/krb5.keytab`에 유효한 항목이 있으면 서버에 GSS-API 승인자 자격 증명이 있는 것입니다.

다음 중 하나가 완료된 경우 클라이언트에 `mech_dh`에 대한 개시자 자격 증명이 있는 것입니다.

- `keylogin` 명령이 실행된 경우
- `pam_dhkeys` 모듈이 `pam.conf` 파일에서 사용된 경우

다음 중 하나가 완료된 경우 클라이언트에 `mech_krb5`에 대한 개시자 자격 증명이 있는 것입니다.

- `kinit` 명령이 실행된 경우
- `pam_krb5` 모듈이 `pam.conf` 파일에서 사용된 경우

보안 RPC에서 `mech_dh`를 사용하는 데 대한 자세한 내용은 “Oracle Solaris 11.2의 Kerberos 및 기타 인증 서비스 관리 ”의 10 장, “네트워크 서비스 인증 구성”을 참조하십시

오. mech_krb5 사용에 대한 자세한 내용은 "Oracle Solaris 11.2의 Kerberos 및 기타 인증 서비스 관리 "의 2 장, "Kerberos 서비스 정보"를 참조하십시오. 방식에 대한 자세한 내용은 mech(4) 및 mech_spnego(5) 매뉴얼 페이지를 참조하십시오.

보안 셸의 명령 실행 및 데이터 전달

인증이 완료되면 사용자가 일반적으로 셸을 요청하거나 명령을 실행하여 보안 셸을 사용할 수 있습니다. 사용자는 ssh 명령 옵션을 통해 요청을 생성할 수 있습니다. 예를 들어, 의사 TTY를 할당하거나 X11 연결 또는 TCP/IP 연결을 전달하거나 보안 연결을 통해 ssh-agent 인증 프로그램을 사용으로 설정하는 요청을 생성할 수 있습니다.

기본적인 사용자 세션 구성 요소는 다음과 같습니다.

1. 사용자가 세션 모드를 시작하는 셸 또는 명령 실행을 요청합니다.

   이 모드에서는 데이터가 클라이언트측 터미널을 통해 전송 또는 수신됩니다. 서버측에서는 데이터가 셸 또는 명령을 통해 전송됩니다.
2. 데이터 전송이 완료되면 사용자 프로그램이 종료됩니다.
3. 기존 연결을 제외하고 모든 X11 전달 및 TCP/IP 전달이 중지됩니다. 기존 X11 연결 및 TCP/IP 연결은 열린 상태로 유지됩니다.
4. 서버가 클라이언트로 종료 상태 메시지를 보냅니다. 열린 상태로 유지되었던 전달된 포트 등 모든 연결이 해제되면 클라이언트가 서버에 대한 연결을 해제합니다. 그런 다음 클라이언트가 종료됩니다.

보안 셸의 클라이언트 및 서버 구성

보안 셸 세션의 특성은 구성 파일로 제어됩니다. 구성 파일은 명령줄의 옵션에 의해 특정 수준으로 대체될 수 있습니다.

보안 셸의 클라이언트 구성

대부분의 경우 보안 셸 세션의 클라이언트측 특성은 시스템 차원의 구성 파일 /etc/ssh/ssh_config로 제어됩니다. ssh_config 파일의 설정은 사용자의 구성 파일 ~/.ssh/config로 대체할 수 있습니다. 사용자는 명령줄에서 두 구성 파일을 대체할 수도 있습니다.

서버의 /etc/ssh/sshd_config 파일 설정에 따라 서버가 허용하는 클라이언트 요청이 결정됩니다. 서버 구성 설정 목록은 "보안 셸의 키워드" [34]를 참조하십시오. 자세한 내용은 sshd_config(4) 매뉴얼 페이지를 참조하십시오.

클라이언트 구성 파일의 키워드는 "보안 셸의 키워드" [34]에서 나열됩니다. 키워드에 기본값이 있을 경우 값이 제공됩니다. 이 키워드에 대해서는 ssh(1), scp(1), sftp(1) 및

ssh_config(4) 매뉴얼 페이지에서 자세히 설명합니다. 영문자순 키워드 목록 및 동등한 명령줄 대체는 표 2-5. “보안 셸 키워드에 대한 명령줄 항목”을 참조하십시오.

보안 셸의 서버 구성

보안 셸 세션의 서버측 특성은 /etc/ssh/sshd_config 파일로 제어됩니다. 서버 구성 파일의 키워드는 “보안 셸의 키워드” [34]에서 나열됩니다. 키워드에 기본값이 있을 경우 값이 제공됩니다. 키워드에 대한 자세한 내용은 sshd_config(4) 매뉴얼 페이지를 참조하십시오.

보안 셸의 키워드

다음 표에서는 키워드 및 해당 기본값(있을 경우)을 나열합니다. 키워드는 영문자순으로 표시됩니다. 클라이언트에 적용되는 키워드는 ssh_config 파일에 있으며, 서버에 적용되는 키워드는 sshd_config 파일에 있습니다. 두 파일에서 설정되는 키워드도 있습니다. v1 프로토콜을 실행 중인 보안 셸 서버의 키워드는 표시됩니다.

표 2-1 보안 셸 구성 파일의 키워드

키워드	기본값	위치
AllowGroups		서버
AllowTcpForwarding	yes	서버
AllowUsers		서버
AuthorizedKeysFile	~/.ssh/authorized_keys	서버
Banner	/etc/issue	서버
Batchmode	no	클라이언트
BindAddress		클라이언트
CheckHostIP	yes	클라이언트
ChrootDirectory	no	서버
Cipher	blowfish, 3des	클라이언트
Ciphers	aes128-ctr, aes128-cbc, 3des-cbc, blowfish-cbc, arcfour	모두
ClearAllForwardings	no	클라이언트
ClientAliveCountMax	3	서버
ClientAliveInterval	0	서버
Compression	no	모두
CompressionLevel		클라이언트
ConnectionAttempts	1	클라이언트
ConnectTimeout	시스템 TCP 시간 초과	클라이언트

키워드	기본값	위치
DenyGroups		서버
DenyUsers		서버
DisableBanner	no	클라이언트
DynamicForward		클라이언트
EscapeChar	~	클라이언트
FallBackToRsh	no	클라이언트
ForwardAgent	no	클라이언트
ForwardX11	no	클라이언트
ForwardX11Trusted	yes	클라이언트
GatewayPorts	no	모두
GlobalKnownHostsFile	/etc/ssh/ssh_known_hosts	클라이언트
GSSAPIAuthentication	yes	모두
GSSAPIDelegateCredentials	no	클라이언트
GSSAPIKeyExchange	yes	모두
GSSAPIStoreDelegateCredentials	yes	서버
HashKnownHosts	no	클라이언트
Host	* 자세한 내용은 “보안 셸의 호스트 특정 매개변수” [37]를 참조하십시오.	클라이언트
HostbasedAuthentication	no	모두
HostbasedUsesNameFromPacketOnly	no	서버
HostKey (v1)	/etc/ssh/ssh_host_key	서버
HostKey (v2)	/etc/ssh/host_rsa_key, /etc/ssh/host_dsa_key	서버
HostKeyAlgorithms	ssh-rsa, ssh-dss	클라이언트
HostKeyAlias		클라이언트
HostName		클라이언트
IdentityFile	~/.ssh/id_dsa, ~/.ssh/id_rsa	클라이언트
IgnoreIfUnknown		클라이언트
IgnoreRhosts	yes	서버
IgnoreUserKnownHosts	yes	서버
KbdInteractiveAuthentication	yes	모두
KeepAlive	yes	모두
KeyRegenerationInterval	3600 (seconds)	서버
ListenAddress		서버
LocalForward		클라이언트
LoginGraceTime	120 (seconds)	서버
LogLevel	info	모두
LookupClientHostnames	yes	서버
MACs	hmac-sha1-*, hmac-md5-* 및 hmac-sha2-* 알고리즘	모두

키워드	기본값	위치
Match		서버
MaxStartups	10:30:60	서버
NoHostAuthenticationForLocalHost	no	클라이언트
NumberOfPasswordPrompts	3	클라이언트
PAMServiceName		서버
PAMServicePrefix		서버
PasswordAuthentication	yes	모두
PermitEmptyPasswords	no	서버
PermitRootLogin	no	서버
PermitUserEnvironment	no	서버
PidFile	/system/volatile/sshd.pid	서버
Port	22	모두
PreferredAuthentications	hostbased,publickey,keyboard-interactive,passwor	클라이언트
PreUserauthHook		서버
PrintLastLog	yes	서버
PrintMotd	no	서버
Protocol	2,1	모두
ProxyCommand		클라이언트
PubkeyAuthentication	yes	모두
RekeyLimit	1G~4G	클라이언트
RemoteForward		클라이언트
RhostsAuthentication	no	서버, v1
RhostsRSAAuthentication	no	서버, v1
RSAAuthentication	no	서버, v1
ServerAliveCountMax	3	클라이언트
ServerAliveInterval	0	클라이언트
ServerKeyBits	512~768	서버, v1
StrictHostKeyChecking	ask	클라이언트
StrictModes	yes	서버
Subsystem	sftp /usr/lib/ssh/sftp-server	서버
SyslogFacility	auth	서버
UseFIPS140	no	모두
UseOpenSSLEngine	yes	모두
UsePrivilegedPort	no	모두
User		클라이언트
UserKnownHostsFile	~/.ssh/known_hosts	클라이언트
UseRsh	no	클라이언트
VerifyReverseMapping	no	서버
X11DisplayOffset	10	서버

키워드	기본값	위치
X11Forwarding	yes	서버
X11UseLocalHost	yes	서버
XAuthLocation	/usr/bin/xauth	모두

보안 셸의 호스트 특정 매개변수

서로 다른 로컬 호스트에는 다른 보안 셸 특정을 지정하는 것이 유용할 수도 있습니다. 관리자는 Host 키워드로 파일의 항목을 그룹화하여 호스트나 정규 표현식에 따라 적용할 별도의 매개변수 세트를 /etc/ssh/ssh_config 파일에 정의할 수 있습니다. Host 키워드를 사용하지 않을 경우 사용자가 작업 중인 로컬 호스트에 클라이언트 구성 파일의 항목이 적용됩니다.

보안 셸 및 로그인 환경 변수

sshd_config 파일에 다음 보안 셸 키워드가 설정되지 않은 경우 /etc/default/login 파일에서 동등한 항목의 값을 가져옵니다.

/etc/default/login의 항목	sshd_config의 키워드 및 값
CONSOLE=*	PermitRootLogin=without-password
#CONSOLE=*	PermitRootLogin=yes
PASSREQ=YES	PermitEmptyPasswords=no
PASSREQ=NO	PermitEmptyPasswords=yes
#PASSREQ	PermitEmptyPasswords=no
TIMEOUT=*seconds*	LoginGraceTime=*seconds*
#TIMEOUT	LoginGraceTime=120
RETRIES 및 SYSLOG_FAILED_LOGINS	password 및 keyboard-interactive 인증 방법에만 적용됩니다.

사용자 로그인 셸의 초기화 스크립트에서 다음 변수가 설정되면 sshd 데몬에 해당 값이 사용됩니다. 변수가 설정되지 않으면 데몬에 기본값이 사용됩니다.

TIMEZONE — TZ 환경 변수의 설정을 제어합니다. 설정되지 않은 경우 데몬이 시작되었으면 sshd 데몬에 TZ 값이 사용됩니다.

ALTSHELL — SHELL 환경 변수의 설정을 제어합니다. 기본값은 ALTSHELL=YES이며, 이 경우 sshd 데몬에 사용자 셸의 값이 사용됩니다. ALTSHELL=NO인 경우 SHELL 값이 설정되지 않습니다.

PATH — PATH 환경 변수의 설정을 제어합니다. 값이 설정되지 않은 경우 기본 경로는 /usr/bin입니다.

SUPATH — root에 대한 PATH 환경 변수의 설정을 제어합니다. 값이 설정되지 않은 경우 기본 경로는 /usr/sbin:/usr/bin입니다.

자세한 내용은 login(1) 및 sshd(1M) 매뉴얼 페이지를 참조하십시오.

보안 셸의 알려진 호스트 유지 관리

다른 호스트와 안전하게 통신해야 할 각 호스트의 로컬 호스트 /etc/ssh/ssh_known_hosts 파일에는 서버의 공개 키가 저장되어 있어야 합니다. 스크립트를 사용하여 /etc/ssh/ssh_known_hosts 파일을 업데이트할 수 있지만 스크립트가 큰 보안 위험에 노출되므로 이 방법은 사용하지 않는 것이 좋습니다.

/etc/ssh/ssh_known_hosts 파일은 다음과 같이 보안 방식으로만 배포해야 합니다.

- 보안 연결(예: 보안 셸, IPsec 또는 알려진 신뢰할 수 있는 시스템의 Kerberos화된 ftp)을 통해 배포
- 시스템 설치 시 배포

침입자가 보거스 공개 키를 known_hosts 파일에 삽입하여 액세스 권한을 얻을 수 있는 가능성을 없애려면 ssh_known_hosts 파일의 알려진 신뢰할 수 있는 소스를 사용해야 합니다. ssh_known_hosts 파일은 설치 중 배포할 수 있습니다. 나중에 scp 명령을 사용하는 스크립트를 통해 최신 버전을 복사할 수 있습니다.

보안 셸 파일

다음 표에서는 중요한 보안 셸 파일 및 제안되는 파일 사용 권한을 보여줍니다.

표 2-2 보안 셸 파일

파일 이름	설명	제안되는 사용 권한 및 소유자
~/.rhosts	사용자가 암호 없이 로그인할 수 있는 호스트를 지정하는 호스트/사용자 이름 쌍을 포함합니다. rlogind 및 rshd 데몬에도 이 파일이 사용됩니다.	-rw-r--r-- *username*
~/.shosts	사용자가 암호 없이 로그인할 수 있는 호스트를 지정하는 호스트/사용자 이름 쌍을 포함합니다. 다른 유틸리티는 이 파일을 사용하지 않습니다. 자세한 내용은 sshd(1M) 매뉴얼 페이지의 FILES 절을 참조하십시오.	-rw-r--r-- *username*
~/.ssh/authorized_keys	사용자 계정에 로그인할 수 있는 사용자의 공개 키를 보관합니다.	-rw-r--r-- *username*
~/.ssh/config	시스템 설정을 대체하는 사용자 설정을 구성합니다.	-rw-r--r-- *username*
~/.ssh/environment	로그인 시 초기 지정을 포함합니다. 기본적으로 이 파일은 읽히지 않습니다. 이 파일을 읽으려면 sshd_config 파일에서 PermitUserEnvironment 키워드를 yes로 설정해야 합니다.	-rw-r--r-- *username*

파일 이름	설명	제안되는 사용 권한 및 소유자
/etc/hosts.equiv	.rhosts 인증에서 사용되는 호스트를 포함합니다. rlogind 및 rshd 데몬에도 이 파일이 사용됩니다.	-rw-r--r-- root
~/.ssh/known_hosts	클라이언트가 안전하게 통신할 수 있는 모든 호스트의 호스트 공개 키를 포함합니다. 파일은 자동으로 유지 관리됩니다. 사용자가 알 수 없는 호스트를 사용하여 연결할 때마다 원격 호스트 키가 파일에 추가됩니다.	-rw-r--r-- *username*
/etc/default/login	해당하는 sshd_config 매개변수가 설정되지 않은 경우 sshd 데몬에 대한 기본값을 제공합니다.	-r--r--r-- root
/etc/nologin	이 파일이 존재할 경우 sshd 데몬은 root만 로그인할 수 있도록 합니다. 로그인하려고 시도 중인 사용자에게 이 파일의 내용이 표시됩니다.	-rw-r--r-- root
~/.ssh/rc	사용자 셸이 시작되기 전에 실행되는 초기화 루틴을 포함합니다. 샘플 초기화 루틴에 대해서는 sshd(1M) 매뉴얼 페이지를 참조하십시오.	-rw-r--r-- *username*
/etc/ssh/shosts.equiv	호스트 기반 인증에서 사용되는 호스트를 포함합니다. 다른 유틸리티는 이 파일을 사용하지 않습니다.	-rw-r--r-- root
/etc/ssh/ssh_config	클라이언트 시스템에서 시스템 설정을 구성합니다.	-rw-r--r-- root
/etc/ssh/ssh_host_dsa_key 또는 /etc/ssh/ssh_host_rsa_key	호스트 개인 키를 포함합니다.	-rw------- root
/etc/ssh_host_key.pub 또는 /etc/ssh/ssh_host_dsa_key.pub 또는 /etc/ssh/ssh_host_rsa_key.pub	호스트 공개 키(예: /etc/ssh/ssh_host_rsa_key.pub)를 포함합니다. 로컬 known_hosts 파일에 호스트 키를 복사하는 데 사용됩니다.	-rw-r--r-- root
/etc/ssh/ssh_known_hosts	클라이언트가 안전하게 통신할 수 있는 모든 호스트의 호스트 공개 키를 포함합니다. 파일은 관리자가 채웁니다.	-rw-r--r-- root
/etc/ssh/sshd_config	보안 셸 데몬 sshd에 대한 구성 데이터를 포함합니다.	-rw-r--r-- root
/system/volatile/sshd.pid	보안 셸 데몬 sshd의 프로세스 ID를 포함합니다. 여러 데몬이 실행 중인 경우 파일에 마지막으로 시작된 데몬이 포함됩니다.	-rw-r--r-- root
/etc/ssh/sshrc	관리자가 지정한 호스트 특정 초기화 루틴을 포함합니다.	-rw-r--r-- root

참고 - sshd_config 파일은 사이트 사용자 정의 패키지의 파일로 대체할 수 있습니다. 자세한 내용은 pkg(5) 매뉴얼 페이지에서 overlay 파일 속성의 정의를 참조하십시오.

다음 표에서는 키워드 또는 명령 옵션이 대체할 수 있는 보안 셸 파일을 나열합니다.

표 2-3 보안 셸 파일 위치에 대한 대체

파일 이름	키워드 대체	명령줄 대체
/etc/ssh/ssh_config		ssh -F *config-file* scp -F *config-file*
~/.ssh/config		ssh -F *config-file*

파일 이름	키워드 대체	명령줄 대체
/etc/ssh/host_rsa_key /etc/ssh/host_dsa_key	HostKey	
~/.ssh/identity ~/.ssh/id_dsa, ~/.ssh/id_rsa	IdentityFile	ssh -i *ID-file* scp -i *ID-file*
~/.ssh/authorized_keys	AuthorizedKeysFile	
/etc/ssh/ssh_known_hosts	GlobalKnownHostsFile	
~/.ssh/known_hosts	UserKnownHostsFile IgnoreUserKnownHosts	

보안 셸 명령

다음 표에서는 주요 보안 셸 명령을 요약합니다.

표 2-4 보안 셸의 명령

명령 매뉴얼 페이지	설명
ssh(1)	사용자를 원격 시스템에 로그인하고 원격 시스템에서 안전하게 명령을 실행합니다. ssh 명령은 비보안 네트워크를 통해 신뢰할 수 없는 두 호스트 간에 암호화된 보안 통신을 가능하게 합니다. 또한 X11 연결 및 임의적 TCP/IP 포트는 보안 채널을 통해 전달될 수 있습니다.
sshd(1M)	보안 셸의 데몬 데몬은 클라이언트로부터의 연결을 수신 대기하며 비보안 네트워크를 통해 신뢰할 수 없는 두 호스트 간에 암호화된 보안 통신을 가능하게 합니다.
ssh-add(1)	인증 에이전트 ssh-agent에 RSA 또는 DSA ID를 추가합니다. ID를 키라고도 합니다.
ssh-agent(1)	공개 키 인증에 사용되는 개인 키를 보관합니다. ssh-agent 프로그램은 X-세션 또는 로그인 세션 시작 시 시작됩니다. 기타 모든 창 및 다른 프로그램은 ssh-agent 프로그램의 클라이언트로 시작됩니다. 환경 변수 사용을 통해 에이전트는 사용자가 ssh 명령을 사용하여 다른 시스템에 로그인할 때 배치되고 인증에 사용될 수 있습니다.
ssh-keygen(1)	보안 셸에 대한 인증 키를 생성 및 관리합니다.
ssh-keyscan(1)	다양한 보안 셸 호스트의 공개 키를 수집합니다. ssh_known_hosts 파일 작성 및 확인을 지원합니다.
ssh-keysign(1M)	ssh 명령이 로컬 호스트의 호스트 키에 액세스하는 데 사용합니다. 보안 셸 v2를 통한 호스트 기반 인증 중 필요한 디지털 서명을 생성합니다. 사용자가 아닌 ssh 명령이 이 명령을 호출합니다.
scp(1)	암호화된 ssh 전송을 통해 네트워크의 호스트 간에 안전하게 파일을 복사합니다. rcp 명령과 달리 scp 명령은 인증에 암호 정보가 필요한 경우 암호 또는 문장암호를 묻습니다.
sftp(1)	ftp 명령과 유사한 대화식 파일 전송 프로그램입니다. ftp 명령과 달리 sftp 명령은 암호화된 ssh 전송을 통해 모든 작업을 수행합니다. 명령은 연결 후 지정된 호스트 이름에 로그인하고 대화식 명령 모드를 시작합니다.

다음 표에서는 보안 셸 키워드를 대체하는 명령 옵션을 나열합니다. 키워드는 ssh_config 및 sshd_config 파일에서 지정됩니다.

표 2-5 보안 셸 키워드에 대한 명령줄 항목

키워드	ssh 명령줄 대체	scp 명령줄 대체
BatchMode		scp -B
BindAddress	ssh -b *bind-addr*	scp -a *bind-addr*
Cipher	ssh -c *cipher*	scp -c *cipher*
Ciphers	ssh -c *cipher-spec*	scp -c *cipher-spec*
Compression	ssh -C	scp -C
DynamicForward	ssh -D *SOCKS4-port*	
EscapeChar	ssh -e *escape-char*	
ForwardAgent	ssh -A(사용) ssh -a(사용 안함)	
ForwardX11	ssh -X(사용) ssh -x(사용 안함)	
GatewayPorts	ssh -g	
IPv4	ssh -4	scp -4
IPv6	ssh -6	scp -6
LocalForward	ssh -L *localport:remotehost: remoteport*	
MACS	ssh -m *MAC-spec*	
Port	ssh -p *port*	scp -P *port*
Protocol	ssh -2(v2 전용)	
RemoteForward	ssh -R *remoteport:localhost: localport*	

색인

ㅍ

ㅎ