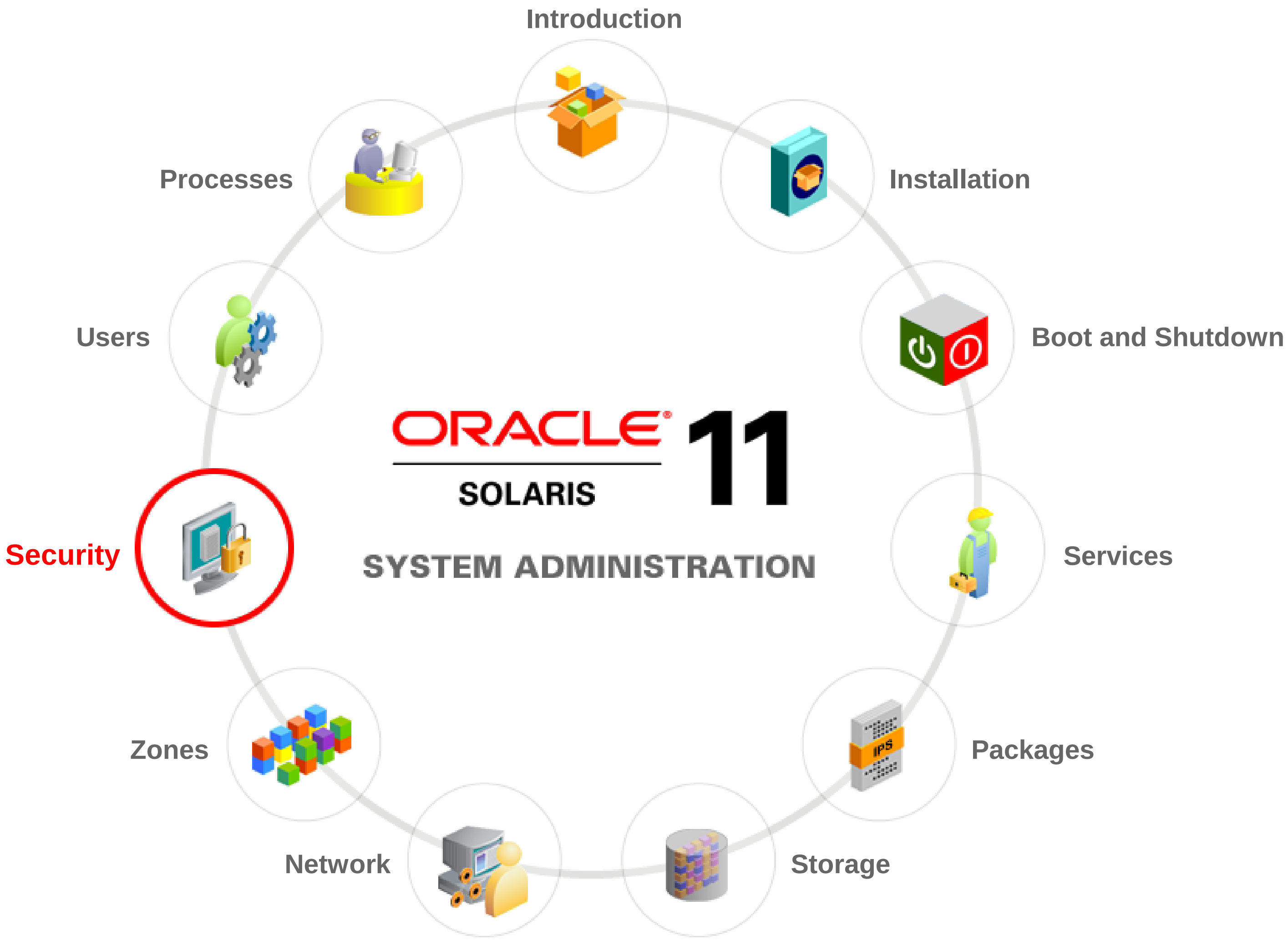
9

Controlling Access to Systems and Files

ORACLE

Workflow Orientation



Objectives

After completing this lesson, you should be able to:

- Establish system and file access control
- Control access to systems
- Control access to files
- Secure access to a remote host

Importance of System and File Access Control

It is important to control access to systems and files to prevent:

- Unauthorized user access
- Intruders gaining remote access

Implementing System and File Access Control

As part of implementing system and file access control, you will learn how to:

- Set up and test system and file access controls
- Verify that the controls are working
- Set up and test Secure Shell

Agenda

- **Controlling Access to Systems**
- Controlling Access to Files
- Securing Access to a Remote Host

Controlling Access to Systems

You can control a user’s access to a system by:

- Securing logins and passwords
- Changing the password algorithm

Login and Password Security

- Use login control and password assignment to prevent unauthorized logins to a system or the network.
- The `login` command:
  - Verifies the username and password
  - Denies access to the system if the username and/or password are incorrect
- Ensure that all the accounts on a system have a password.
- Passwords are kept secure through:
  - Encryption
  - Placement in a separate file from username and other information

Password Algorithms and the `/etc/security/policy.conf` File

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

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

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

/etc/security/crypt.conf File

```
#
#ident    "%Z%%M%   %I%      %E% SMI"
#
# The algorithm name __unix__ is reserved.

1         crypt_bsdmd5.so.1
2a        crypt_bsdbf.so.1
md5       crypt_sunmd5.so.1
5         crypt_sha256.so.1
6         crypt_sha512.so.1
```

Identifier	Description
1	MD5 algorithm
2a	Blowfish algorithm
md5	Sun MD5 algorithm
5	SHA256 algorithm
6	SHA512 algorithm
unix	Traditional UNIX encryption algorithm

Controlling and Monitoring System Activities

It is your responsibility to control and monitor system activity by performing the following:

- Setting limits on who can use what resources
- Logging resource use
- Monitoring who is using the resources

Note: The system tracks real and effective user and group ID logins. To determine the real UID, use `who am i`. To determine the effective UID, use `whoami`.

Securing Logins and Passwords

- Displaying a user’s login status
- Displaying users without passwords
- Disabling user logins temporarily
- Monitoring failed login attempts
- Monitoring all failed login attempts
- Changing the password algorithm
- Verifying the password algorithm change
- Monitoring who is using the `su` command

Displaying a User’s Login Statusq2

To display a user’s login status, use `logins -x -l` *`loginname`*.

```
# logins -x -l jjones
jjones          1003    itadmin           110      joe jones
                        /export/home/jjones
                        /usr/bin/bash
                        PS 010170 -1 -1 -1
```

To display a group’s status, use `logins -x -g` *`groupname`*.

```
# logins -x -g itadmin
jjones          1003    itadmin           110      joe jones
                        /export/home/jjones
                        /usr/bin/bash
                        PS 010170 -1 -1 -1
ppeter          1004    itadmin           110      pan peter
                        /export/home/ppeter
                        /usr/bin/ksh
                        PS 120570 2 5 10
```

Displaying Users Without Passwords

To display users without passwords, use `logins -p`.

```
# logins -p
omai           1016    staff           10      olin mai
mhatter        1009    staff           10      maddy hatter
tbone           501    other            1      terry bone

# grep omai /etc/shadow
omai::15310::::::
```

Booting the System to Single-User Mode

To temporarily block any non-administrative users from logging in to the system, run `init S`.

```
# init S
```

This is commonly done to facilitate system maintenance.

To enable general user login, run `init 3`.

```
# init 3
```

Monitoring Failed Login Attempts

1. Create the `loginlog` file in the `/var/adm` directory.
2. Set read and write permissions for the `root` user on the `loginlog` file.
3. Change group membership to `sys` on the `loginlog` file.
4. Verify that the log works.

```
# touch /var/adm/loginlog
# chmod 600 /var/adm/loginlog
# chgrp sys /var/adm/loginlog
# cat /var/adm/loginlog
jjones:/dev/pts/2:Mon Nov 11 23:21:10 2015
jjones:/dev/pts/2:Mon Nov 11 23:21:21 2015
jjones:/dev/pts/2:Mon Nov 11 23:21:30 2015
jjones:/dev/pts/2:Mon Nov 11 23:21:40 2015
jjones:/dev/pts/2:Mon Nov 11 23:21:49 2015
```

Monitoring All Failed Login Attempts

1. Edit the `/etc/default/login` file with `SYSLOG=YES` and `SYSLOG_FAILED_LOGINS=0`.
2. Create a file with the correct permissions to hold the logging information.
   a. Create the `authlog` file in the `/var/adm` directory.
   b. Set read and write permissions for the `root` user on the `authlog` file.
   c. Change group membership to `sys` on the `authlog` file.
3. Edit the `syslog.conf` file to log failed password attempts.
   a. Make the `auth.notice` entry in the `syslog.conf` file.
   b. Refresh the `system-log` service.
4. Verify that the log works.

Monitoring All Failed Login Attempts: Example

```
# vi /etc/default/login
# more /etc/default/login
…
SYSLOG=YES
…
SYSLOG_FAILED_LOGINS=0
…
# touch /var/adm/authlog
# chmod 600 /var/adm/authlog
# chgrp sys /var/adm/authlog
# vi /etc/syslog.conf
# grep auth.notice /etc/syslog.conf
*.err;kern.notice;auth.notice            /dev/sysmsg
auth.notice                  /var/adm/authlog
#auth.notice         ifdef(`LOGHOST', /var/log/authlog, @loghost)
# svcadm refresh system/system-log

<Test the entry by attempting to log in as user using an incorrect password>

# cat /var/adm/authlog
Dec 2 16:57:27 client1 su: [ID 810491 auth.crit] 'su jdoe' failed for oracle
on /dev/pts/1
```

Changing the Password Algorithm

1. View the available password-encrypting algorithms in the `/etc/security/crypt.conf` file and determine which algorithm you want to use.
2. Using a text editor, change the password algorithm in the `/etc/security/policy.conf` file by:
   a. Commenting out the current default entry
   b. Specifying a different encryption algorithm from the list of available algorithms

Changing the Password Algorithm: Example

```
# cat /etc/security/crypt.conf
#
#ident  "%Z%%M%  %I%      %E% SMI"
#
# The algorithm name __unix__ is reserved.

1       crypt_bsdmd5.so.1
2a      crypt_bsdbf.so.1
md5     crypt_sunmd5.so.1
5       crypt_sha256.so.1
6       crypt_sha512.so.1…
# vi /etc/security/policy.conf
#
CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6
#
# Passwords previously encrypted with SHA256 will be encrypted with
SHA512
# when users change their passwords.
#
#CRYPT_DEFAULT=5
CRYPT_DEFAULT=6
```

Verifying the Password Algorithm Change

```
# grep jjones /etc/shadow
jjones $5$ABL6xEPA$NZ6SOesHBOas7/kJPWsdUyMTzbBvWo4L6lmkqx4YX8B:15310:56:70:7:::

<Changed password algorithm in /etc/security/policy.conf>

# passwd jjones
New Password:
Re-enter new Password:
passwd: password successfully changed for jjones
# grep jjones /etc/shadow
jjones $5$ABL6xDJBA$NZ6SOesHBOas7/kABCsdUyMTzbBvWo4L6lmkqx4YX8B:15310:56:70:7:::

# passwd -d jjones
passwd: password information changed for jjones
# grep jjones /etc/shadow
jjones::15310:56:70:7:::

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

# grep jjones /etc/shadow
jjones $6$peJpli9l$N.lDkvtuNInL42iV2Y7Pno6MJiI.CPWXSvFvs.vynTQx22u9ivnb.cwpYSyncXATQia/pXwf
zwCn//LOTTw9n1:15310:56:70:7:::
```

Monitoring Who Is Using the `su` Command

- By default, `su` logging is enabled in `/var/adm/sulog`.
- The `SULOG=/var/adm/sulog` entry in `/etc/default/su` enables `su` logging.

To monitor `su` logging, use `more /var/adm/sulog`.

```
# more /var/adm/sulog
SU 12/01 10:26 - pts/0 jjones-root
SU 12/01 10:59 + pts/0 jjones-root
SU 12/02 11:11 + pts/0 root-omai
SU 12/02 14:56 - pts/0 jdoe-root
SU 12/02 14:57 + pts/0 jdoe-root
```

Quiz

In which file can you specify the password algorithms configuration?

a. `/etc/passwd`

b. `/etc/shadow`

c. `/etc/security/crypt.conf`

d. `/etc/security/policy.conf`

Quiz

In which file can you specify the password algorithms configuration?

a. `/etc/passwd`

b. `/etc/shadow`

c. `/etc/security/crypt.conf`

d. `/etc/security/policy.conf`

Agenda

- Controlling Access to Systems
- **Controlling Access to Files**
- Securing Access to a Remote Host

Controlling Access to Files

To secure files and directories in Oracle Solaris 11, you can use:

- UNIX file permissions
- Access control lists (ACLs)

Command	Description
`ls`	Lists the files in a directory and information about the files
`chown`	Changes the ownership of a file
`chgrp`	Changes the group ownership of a file
`chmod`	Changes permissions on a file. You can use either symbolic mode, which uses letters and symbols, or absolute mode, which uses octal numbers, to change the permissions on a file.

File Types

Symbol	Description
`b`	Block special file
`c`	Character special file
`d`	Directory
`l`	Symbolic link
`s`	Socket
`D`	Door. A door is a special file for inter-process communication between a client and server, currently implemented only in Solaris.
`P`	Named pipe
`-` (minus sign)	Regular text file or a program

UNIX File Permissions

Symbol	Permission	Object	Description
r	Read	File	Designated users can open and read the contents of a file.
		Directory	Designated users can list the files in the directory.
w	Write	File	Designated users can modify the contents of the file or delete the file.
		Directory	Designated users can add files or add links in the directory. They can also remove files or remove links in the directory.
x	Execute	File	Designated users can execute the file, if it is a program or shell script.
		Directory	Designated users can open files or execute files in the directory. Users can `cd` into the directory.
-	Denied	File and Directory	Designated users cannot read, write, or execute the file.

Interpreting File Permissions

Permissions	Interpretation
`-rwx------`	This file has read, write, and execute permissions set only for the file owner. Permissions for the class **group** and **other** are denied.
`dr-xr-x---`	This directory has read and execute permissions set only for the directory owner and the group.
`-rwxr-xr-x`	This file has read, write, and execute permissions set for the file owner. Read and execute permissions are set for the class **group** and **other**.

Special File Permissions

- The special permission types for executable files and public directories are:
  - **`setuid`:** Grants access to the files and directories that are normally available only to the owner
  - **`setgid`:** Grants access based on the permissions that are granted to a particular group
  - **sticky bit:** Protects the files within a directory
- When special permissions are used, a user who runs an executable file assumes the ID of the owner (or group) of the executable file.
- Special permissions present a security risk.
- The system should be monitored for any unauthorized use of the `setuid` and `setgid` permissions.

File Permission Modes

You use `chmod` to set permissions in either of two modes:

- **Symbolic Mode:** Combinations of letters and symbols are used to add permissions or remove permissions.
- **Absolute Mode:** Numbers are used to represent file permissions. This is the most commonly used method to set permissions.

Setting File Permissions in Symbolic Mode

Symbol	Function	Description
`u`	*who*	User (owner)
`g`	*who*	Group
`o`	*who*	Others
`a`	*who*	All
`=`	*operator*	Assign
`+`	*operator*	Add
`-`	*operator*	Remove
`r`	*permissions*	Read
`w`	*permissions*	Write

Setting File Permissions in Symbolic Mode

Symbol	Function	Description
`x`	*permissions*	Execute
`l`	*Permissions*	Mandatory locking; `setgid` bit is on; group execution bit is off.
`s`	*permissions*	`setuid` or `setgid` bit is on.
`t`	*permissions*	Sticky bit is on; execution bit for others is on.

Setting File Permissions in Absolute Mode

Octal Value	Interpretation	Permissions Description
0	`---`	No permissions
1	`--x`	Execute permission only
2	`-w-`	Write permission only
3	`-wx`	Write and execute permissions
4	`r--`	Read permission only
5	`r-x`	Read and execute permissions
6	`rw-`	Read and write permissions
7	`rwx`	Read, write, and execute permissions

Setting Special File Permissions in Symbolic or Absolute Mode

- To set special permissions on a file, you can use either the symbolic or absolute mode.
- To set or remove the `setuid` permission on a directory, you must use symbolic mode.
- To set special permissions in absolute mode, you add a new octal value.

Octal Value	Special File Permissions
1	Sticky bit
2	`setgid`
4	`setuid`

Protecting Files with Basic UNIX Permissions

- Displaying file permissions
- Changing file ownership
- Changing the group ownership of a file
- Changing file permissions in symbolic mode
- Changing file permissions in absolute mode
- Setting special file permissions in absolute mode

Displaying File Permissions

To display file permissions for all the files in a directory, use `ls -la`.

```
# cd /sbin
# ls -la
total 4960
drwxr-xr-x   4 root     bin        454  Oct 28  05:10 .
drwxr-xr-x  33 root     sys         45  Oct 27  10:00 ..
-r-xr-xr-x   1 root     bin      12772  Oct 19  20:55 autopush*
lrwxrwxrwx   1 root     root        10  Oct 27  10:00 accept -> cupsaccept
...
```

To display the permissions for a directory, use `ls -ld`.

```
# cd ..
# ls -ld sbin
lrwxrwxrwx   1 root     root        10  Oct 27  10:03 sbin -> ./usr/sbin
```

Changing File Ownership

1. Display the permissions on a file by using `ls -l` *`filename`*.
2. Change the owner of the file by using `chown` *`loginname`* *`filename`*.
3. Verify that the owner of the file has changed by using `ls -l` *`filename`*.

```
# ls -l test-file
-rw-r--r--   1 mhatter  staff   112640 Nov 2 10:49 test-file
# chown omai test-file
# ls -l test-file
-rw-r--r--   1 omai     staff   112640 Nov 2 08:50 test-file
```

Changing the Group Ownership of a File

1. Display the permissions on a file by using `ls -l` *`filename`*.
2. Change the group ownership of the file by using `chgrp groupname` *`filename`*.
3. Verify that the group ownership of the file has changed by using `ls -l` *`filename`*.

```
# ls -l test-file
-rw-r--r--   1 omai  staff    112640 Nov 6 08:50 test-file
# chgrp itadmin test-file
# ls -l test-file
-rw-r--r--   1 omai  itadmin  112640 Nov 6 08:50 test-file
```

Changing File Permissions in Symbolic Mode

1. Display the permissions on a file by using `ls -l` *filename*.
2. Change the file permissions by using `chmod` *who operator permissions filename*.
3. Verify that the permissions of the file have changed by using `ls -l` *filename*.

```
# ls -l test-file
-rw-r--r--   1 omai itadmin   112640 Nov 6 08:50 test-file
# chmod g+wx test-file
# ls -l test-file
-rw-rwxr--   1 omai itadmin   112640 Nov 6 09:00 test-file
# chmod u-w test-file
# ls -l test-file
-r--rwxr--   1 omai itadmin   112640 Nov 6 09:05 test-file
```

Changing File Permissions in Absolute Mode

1. Display the permissions on a file by using `ls -l` *`filename`*.
2. Change the file permissions by using `chmod` *`nnn`* *`filename`*.
3. Verify that the permissions of the file have changed by using `ls -l` *`filename`*.

```
# ls -l test-file
-rw-r--r--   1 omai itadmin   112640 Nov 7 08:50 test-file
# chmod 674 test-file
# ls -l test-file
-rw-rwxr--   1 omai itadmin   112640 Nov 7 09:10 test-file
# chmod 474 test-file
# ls -l test-file
-r--rwxr--   1 omai itadmin   112640 Nov 7 09:15 test-file
```

Setting Special File Permissions in Absolute Mode

1. Display the permissions on a file by using `ls -l` *`filename`*.
2. Change the special file permissions by using `chmod` *`nnnn`* *`filename`*.
3. Verify that the permissions of the file have changed by using `ls -l` *`filename`*.

```
# ls -l test-file
-rw-r--r--   1 omai itadmin   112640 Nov 8 09:50 test-file
# chmod 4655 test-file
# ls -l test-file
-rwsr--r--   1 omai itadmin   112640 Nov 8 10:10 test-file
```

Protecting Files from Accidental Deletion

You can protect files from accidentally being deleted by marking them with the `nounlink` attribute.

```
# mkdir files
# cd files
# chmod S+vnounlink .
# touch test-file
# echo “test” >> test-file
# cat test-file
test
# rm test-file
rm: test-file not removed: Not owner
# chmod S-vnounlink .
# rm test-file
```

Protecting Against Programs with Security Risk

- Finding files with special file permissions
- Disabling programs from using executable stacks

Finding Files with Special File Permissions

1. To find files with `setuid` permissions, use `find` *`directory`* `-user root -perm -4000 -exec ls -ldb {} \; > /tmp/`*`filename`*.
2. To display the results, use `more /tmp/`*`filename`*.

```
# find / -perm -4000 -exec ls -ld {} \; > /var/tmp/suidcheck
find: /proc/1476/fd/4: No such file or directory
# more /var/tmp/suidcheck
-r-sr-xr-x 1 omai itsupport 0 Sept 19 13:44 /home/omai/test-file
-rwsr-xr-x 1 root bin   64588 Sept 19 09:03 /sbin/wificonfig
-r-sr-xr-x 1 root bin  206676 Sept 19 09:02 /usr/lib/ssh/ssh-keysign
-r-sr-xr-x 1 root bin   19452 Sept 19 09:02 /usr/lib/fs/smbfs/mount
...
```

Disabling Programs from Using Executable Stacks

1. Save a copy of the `/etc/system` file.
2. Edit the `/etc/system` file and add the following system directives:
   `set noexec_user_stack=1`
   `set noexec_user_stack_log=0`
3. Reboot the system by using `init 6`.

```
# vi /etc/system
# cat /etc/system
set noexec_user_stack=1
set noexec_user_stack_log=0
# init 6
```

Quiz

Which command enables you to change permissions on a file that is owned by a group?

a. `chown`

b. `chgrp`

c. `chmod`

Quiz

The `chmod` command can be used only with the absolute mode.

a. True

b. False

Quiz

The `chmod` command can be used only with the absolute mode.

a. True

b. False

Quiz

Which permission gives the following?

This file has read, write, and execute permissions set for the file owner. Read and execute permissions are set for the group and other.

a. `-rwx------`

b. `dr-xr-x---`

c. `-rwxr-xr-x`

Quiz

Which permission gives the following?

This file has read, write, and execute permissions set for the file owner. Read and execute permissions are set for the group and other.

a. `-rwx------`

b. `dr-xr-x---`

c. `-rwxr-xr-x`

Quiz

The special permission types `setuid` and `setgid` constitute a risk.

a. True

b. False

Quiz

The special permission types `setuid` and `setgid` constitute a risk.

a. True

b. False

Agenda

- Controlling Access to Systems
- Controlling Access to Files
- **Securing Access to a Remote Host**

Oracle Solaris Authentication Services

Oracle Solaris offers the following authentication services:

Authentication Service	Description
Secure RPC	An authentication mechanism that protects NFS mounts and a naming service
Pluggable Authentication Module (PAM)	A framework that enables various authentication technologies to be plugged in to a system entry service without recompiling the service
Simple Authentication and Security Layer (SASL)	A framework that provides authentication and security services to network protocols
Secure Shell	A secure remote login and transfer protocol that encrypts communications over an unsecure network
Kerberos service	A client/server architecture that provides encryption with authentication

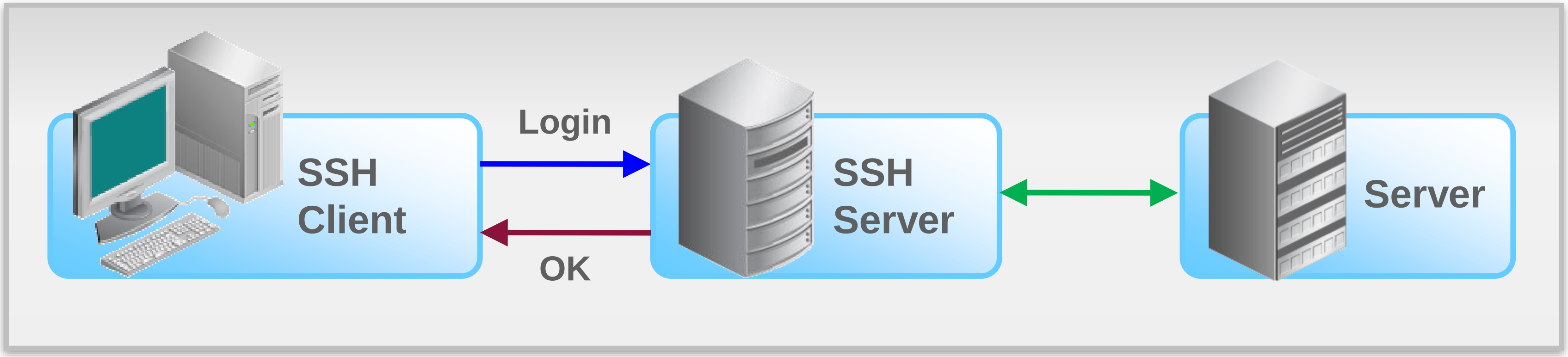
Secure Shell

- Is the default remote access control protocol on a newly installed Oracle Solaris 11 system
- Is a program for logging in to a remote system and executing commands on that system
- Enables users to securely access a remote host over an unsecured network
- Provides commands for remote login and remote file transfer
- Provides authentication by the use of passwords, public keys, or both
- Encrypts all network traffic

Secure Shell

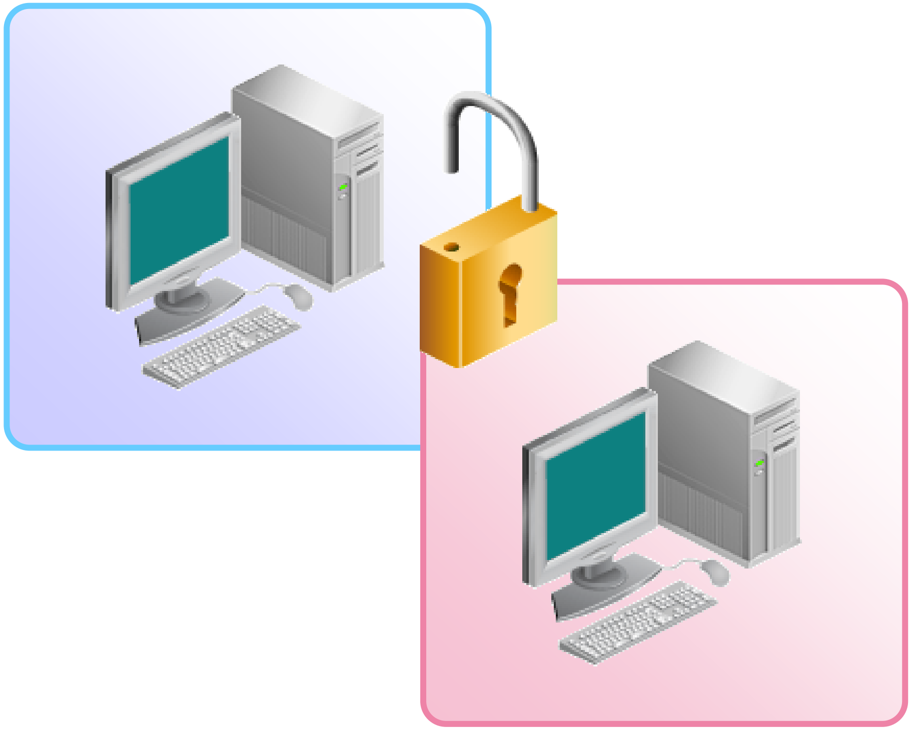
With Secure Shell, you can:

- Log in to another host securely over an unsecured network
- Copy files securely between the two hosts
- Run commands securely on the remote host



Secure Shell and the Secure Shell Protocol

- SSH supports both versions 1 and 2 of the Secure Shell protocol.
- Sites are encouraged to use only version 2.



Secure Shell Protocol Version 2: Parts

Protocol	Description
SSH Transfer Protocol	Is used for server authentication, algorithm negotiation, and key exchange. When this part of the SSH protocol completes, an encrypted communication channel is established between the server and the client.
SSH Authentication Protocol	Is used to verify the identity of the user that runs the `ssh` client. This protocol uses the established transfer protocol.
SSH Channel Protocol	Multiplexes the encrypted channel into logical connections. These connections can be used, for example, for user shell sessions, port forwarding, or X11 forwarding. This protocol uses the authentication protocol that the user established.

Secure Shell Authentication Methods

Method	Description
GSS-API	Uses credentials for GSS-API mechanisms
Public key authentication	Authenticates users with their RSA and DSA public/private keys
Password authentication	Uses PAM to authenticate users

Identifying the Secure Shell Defaults

- Only protocol version 2 is in effect.
- Port forwarding is disabled for the server and client sides.
- X11 forwarding is disabled on the server side.
- All authentication methods are enabled, including GSS-API (preferred authentication method).

Secure Shell sshd Daemon

- The sshd daemon is the daemon program for the secure shell client (ssh).
- ssh provides secure, encrypted communication between two untrusted hosts over an unsecure network.
- You can use the SMF to start, stop, or restart the sshd daemon.
- To notify the sshd daemon to read its configuration files again, use:

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

or

```
# svcadm restart ssh
```

Configuring Secure Shell

1. Verifying that users have access to both the client and the server
2. Logging in to a remote host with Secure Shell
3. Generating the public/private RSA key pair
4. Copying the RSA public key to the remote host
5. Verifying that the RSA public key is functioning
6. Generating the public/private DSA key pair
7. Copying the DSA public key to the remote host
8. Verifying the authentication process

Verifying That Users Have Access to Both the Client and the Server

Server side

```
# grep jjones /etc/passwd
jjones:x:1003:110:joe jones:/export/home/jjones:/usr/bin/bash
```

Client side

```
# grep jjones /etc/passwd
jjones:x:1003:110:joe jones:/export/home/jjones:/usr/bin/bash
```

Logging In to a Remote Host with Secure Shell

```
# su - jjones
Oracle Corporation      SunOS 5.11      11.3     September 2015
jjones@server1:$ ssh client1
The authenticity of host ‘client1 (192.168.0.111)' can't be
established. RSA key fingerprint is
38:d3:8a:bb:be:d4:b8:93:08:7a:b5:99:5d:7f:04:40.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added ‘client1,192.168.0.111' (RSA) to the
list of known hosts.
Password: <password>
Last login: Tue Jul 29 08:17:26 2015 from server1
Oracle Corporation      SunOS 5.11      11.3     September 2015
jjones@client1:~$ exit
Connection to client1 closed.
```

ORACLE

Generating the Public/Private RSA Key Pair

```
jjones@server1:$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key
(/export/home/jjones/.ssh/id_rsa): Press Enter Key
Enter passphrase (empty for no passphrase): <passphrase>
Enter same passphrase again: <passphrase>
Your identification has been saved in
/export/home/jjones/.ssh/id_rsa.
Your public key has been saved in
/export/home/jjones/.ssh/id_rsa.pub.
The key fingerprint is:
51:28:86:f9:3b:55:d3:bf:eb:a9:5d:af:0d:f5:2a:8f jjones@server1
jjones@server1:$ ls .ssh
id_rsa  id_rsa.pub
```

Copying the RSA Public Key to the Remote Host

```
jjones@server1:$ scp .ssh/id_rsa.pub jjones@client1:id_rsa.pub
Password: <password>
id_rsa.pub       100% |*****************************|   401
00:00
jjones@server1:$ ssh client1
Password: <password>
Last login: Tue July 29 08:19:04 2015 from server1
Oracle Corporation      SunOS 5.11      11.3     September 2015
jjones@client1:~$ ls
id_rsa.pub  local.cshrc  local.login  local.profile
jjones@client1:~$ mkdir -p .ssh
jjones@client1:~$ ls
id_rsa.pub  local.cshrc  local.login  local.profile
jjones@client1:~$ cat ./id_rsa.pub >> .ssh/authorized_keys
jjones@client1:~$ rm ./id_rsa.pub
```

Verifying That the RSA Public Key Is Functioning

```
jjones@client1:~$ exit
Connection to client1 closed.
jjones@server1:~$ ssh client1
Enter passphrase for key ‘/export/home/jjones/.ssh/id_rsa':
<passphrase>
Last login: Tue Jul 29 08:21:32 2015 from server1
jjones@client1:~$ exit
Connection to client1 closed.
```

Generating the Public/Private DSA Key Pair

```
jjones@server1:~$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key
(/export/home/jjones/.ssh/id_dsa): <Press Enter Key>
Enter passphrase (empty for no passphrase): <passphrase>
Enter same passphrase again: <passphrase>
Your identification has been saved in
/export/home/jjones/.ssh/id_dsa.
Your public key has been saved in
/export/home/jjones/.ssh/id_dsa.pub.
The key fingerprint is:
7a:b8:cb:f8:33:e5:fb:02:a5:c3:b2:53:cc:75:90:9e jjones@server1
jjones@server1:~$ ls -a .ssh
.      id_dsa        id_rsa        known_hosts
..     Id_dsa.pub    id_rsa.pub
```

Copying the DSA Public Key to the Remote Host

```
jjones@server1:~$ scp .ssh/id_dsa.pub
jjones@client1:id_dsa.pub
Enter passphrase for key '/export/home/jjones/.ssh/id_rsa':
<passphrase>
id_dsa.pub         100% |*****************************|    609
00:00
jjones@server1:~$ ssh client1
Enter passphrase for key ‘/export/home/jjones/.ssh/id_rsa':
<passphrase>
Last login: Tue Jul 29 08:23:05 2015 from server1
Oracle Corporation      SunOS 5.11      11.3    September 2015
jjones@client1:~$ ls
id_dsa.pub local.cshrc  local.login  local.profile
jjones@client1:~$ cat ./id_dsa.pub >> .ssh/authorized_keys
jjones@client1:~$ rm ./id_dsa.pub
jjones@client1:~$ exit
Connection to client1 closed.
```

Verifying the Authentication Process

```
jjones@server1:~$ ssh client1
Enter passphrase for key ‘/export/home/jjones/.ssh/id_rsa':
<Press Enter Key>
Enter passphrase for key ‘/export/home/jjones/.ssh/id_dsa':
<passphrase>
Last login: Tue Jul 29 08:25:16 2015 from server1
Oracle Corporation      SunOS 5.11      11.3     September 2015
jjones@server1:~$ exit
logout
Connection to client1 is closed.
```

Using the Secure Shell

- Reducing password prompts
- Locking and unlocking the authentication agent

Reducing Password Prompts

```
jjones@server1: ~$ eval `ssh-agent`
Agent pid 1886
jjones@server1: ~$ pgrep ssh-agent
1886
jjones@server1: ~$ env | grep SSH
SSH_AGENT_PID=1886
SSH_AUTH_SOCK=/tmp/ssh-XXXXJqaWVf/agent.1885
jjones@server1: ~$ ssh-add
Enter passphrase for /export/home/jjones/.ssh/id_rsa: <passphrase>
Identity added: /export/home/jjones/.ssh/id_rsa
(/export/home/jjones/.ssh/id_rsa)
Identity added: /export/home/jjones/.ssh/id_dsa
(/export/home/jjones/.ssh/id_dsa)
jjones@server1:~$ ssh-add -l
2048 51:28:86:f9:3b:55:d3:bf:eb:a9:5d:af:0d:f5:2a:8f
/export/home/jjones/.ssh/id_rsa (RSA)
1024 7a:b8:cb:f8:33:e5:fb:02:a5:c3:b2:53:cc:75:90:9e
/export/home/jjones/.ssh/id_dsa (DSA)
jjones@server1: ~$ ssh client1
Last login: Tue Jul 29 08:26:22 2015 from server1
Oracle Corporation      SunOS 5.11      11.3        September 2015
jjones@client1:~$ exit
Connection to client1 closed.
```

Locking and Unlocking the Authentication Agent

```
jjones@server1:~$ ssh-add -x
Enter lock password: <password>
Again: <password>
Agent locked.
jjones@server1:~$ ssh client1
Enter passphrase for key ‘/export/home/jjones/.ssh/id_rsa': <passphrase>
Last login: Tue Jul 29 08:27:14 2015 from server1
Oracle Corporation      SunOS 5.11      11.3       September 2015
jjones@server1:~$ exit
Connection to client1 closed.
```

```
jjones@server1:~$ ssh-add -X
Enter lock password: <password>
Agent unlocked.
jjones@server1:~$ ssh client1
Last login: Tue Jul 29 08:27:36 2015 from server1
Oracle Corporation      SunOS 5.11      11.3       September 2015
Connection to client1 closed.
```

Quiz

Secure Shell is an authentication service that ________.

a. Enables a user to securely access a remote host over an unsecure network
b. Provides authentication and security services to network protocols
c. Protects NFS mounts and a naming service

Quiz

Secure Shell is an authentication service that ________.

a. Enables a user to securely access a remote host over an unsecure network

b. Provides authentication and security services to network protocols

c. Protects NFS mounts and a naming service

Quiz

If you do not want to type your passphrase and your password to use Secure Shell, which of the following should you use?

a. `ssh-add`

b. `ssh-agent`

c. `ssh-keygen`

Quiz

If you do not want to type your passphrase and your password to use Secure Shell, which of the following should you use?

a. `ssh-add`

b. `ssh-agent`

c. `ssh-keygen`

Summary

In this lesson, you should have learned how to:

- Establish system and file access control
- Control access to systems
- Control access to files
- Secure access to a remote host

Practice 9: Overview

- 9-1: Controlling Access to Systems
- 9-2: Controlling Access to File Systems
- 9-3: Configuring Secure Shell