



Auditing UNIX, Linux and Oracle – An Interactive Session

Presented by:

Alvin Hau
Derek Koopowitz

Back to Business

Disclaimer

The slides and opinions expressed in this presentation are our own and may not reflect the official positions of the Federal Reserve Bank of San Francisco or the Board of Governors of the Federal Reserve System.

Agenda

- Goals and expectations of the session
- What tools will we need?
- Installing our tools
- Scanning our network to see what is out there

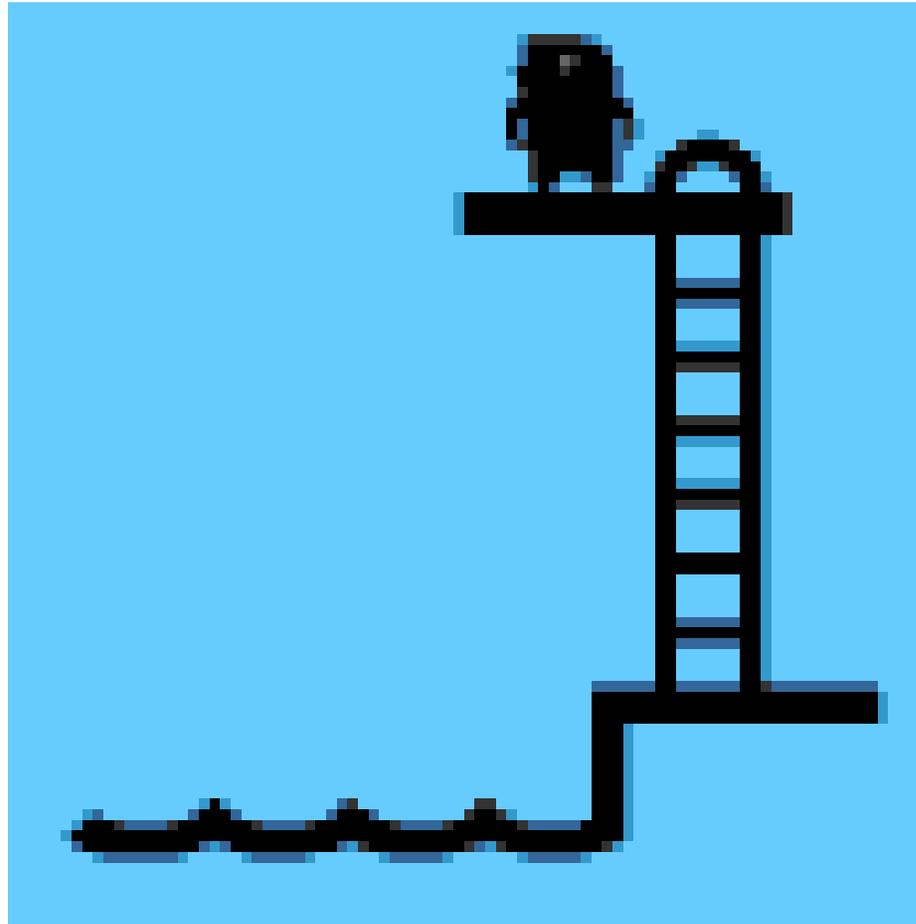
Lunch

- Auditing UNIX / Linux

Afternoon Break

- Auditing Oracle
- Q & A

Ok...Let's Dive Right In...



Goals And Expectations Of The Session

- What are you hoping to get out of this session?
- Any particular areas we should concentrate on?

- Let's meet on the other side...

What Tools Will We Need?

- Scanning the network
 - Nmap – open source network exploration and security auditing tool
 - Scanline – free command line port scanner from McAfee
- Vulnerability Scanning
 - Nessus – free for personal use and runs on Windows, UNIX and Linux
 - NOTE: We will not use this during the session – it will be demonstrated only
- Telnet/SSH Client
 - PuTTY – free SSH, Telnet and Rlogin client for Windows
- Password Crackers
 - John the Ripper – free and runs on Windows, DOS, UNIX, Linux, BeOS and OpenVMS
- Knowledge of UNIX / Linux command line commands
- Scripts to automate the information gathering
- Running SQL commands on Oracle
 - SQL*Plus

Installing Our Tools

Let's install our tools...

But first, connect up to the wireless network.

The one you need to connect up to is...

“FBI Surveillance”

The password is **“isaca2011”** – without the quotes.

Installing Our Tools

- *Disable (if you can) any anti-virus software you have running since several of the tools are detected as potentially harmful – such as scanlink.*
- Open up Windows Explorer
- Create a new folder on your C drive (or another drive) called “tools” – off the root of the drive
- Copy the contents of the “tools” folder on the USB drive to the folder created above

Installing Our Tools - continued

- Use Windows Explorer to browse to the “**tools\nmap**” folder
- Double click the executable “**vcredist_x86.exe**” – this will install some components that nmap needs

Installing Our Tools - continued

- Use Windows Explorer to browse to the **tools\putty** folder
- Right click on the **putty.exe** file and create a shortcut on the desktop
- Right click on the **psftp.exe** file and create a shortcut on the desktop

Installing Our Tools - continued

- Use Windows Explorer to browse to the **tools\sqlplus** folder
- Right click on the **sqlplus.exe** file and create a shortcut on the desktop

Set Up Our Tools

Setting up Putty – for Linux:

Double click the Putty shortcut on the desktop

Type the following for the host name and create a name for the session and save it.

The screenshot shows the PuTTY Configuration dialog box. The 'Category' list on the left is expanded to 'Session'. The 'Basic options for your PuTTY session' section is visible. The 'Host Name (or IP address)' field contains '192.168.1.10' and the 'Port' field contains '22'. The 'Connection type' section has 'SSH' selected. The 'Saved Sessions' list contains 'ISACA Ubuntu', 'ISACA Solaris', and 'Koopowitz (Internet)'. The 'Save' button is highlighted. Three callout boxes with red arrows point to these elements: '1. Type in the IP address' points to the Host Name field, '2. Type in a name for the session' points to the Saved Sessions list, and '3. Click the Save button' points to the Save button.

1. Type in the IP address

2. Type in a name for the session

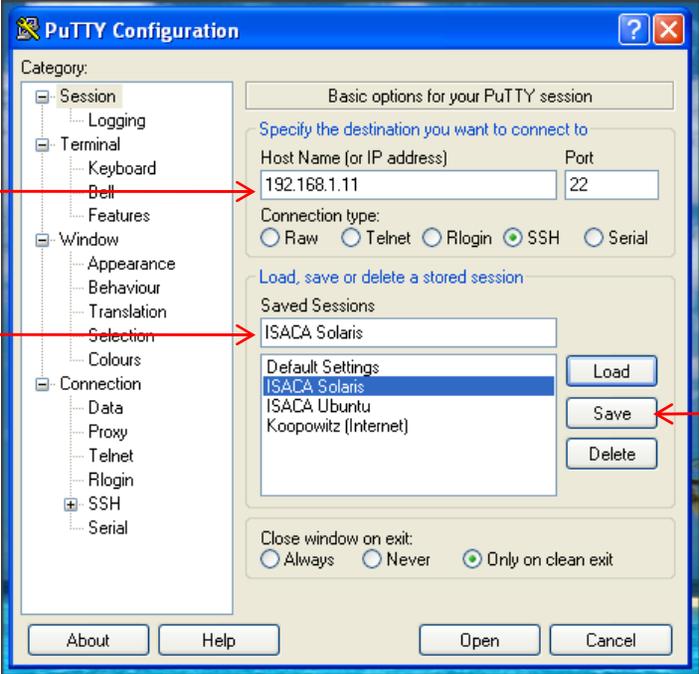
3. Click the Save button

Set Up Our Tools - continued

Setting up Putty – for UNIX:

Double click the Putty shortcut on the desktop

Type the following for the host name and create a name for the session and save it.



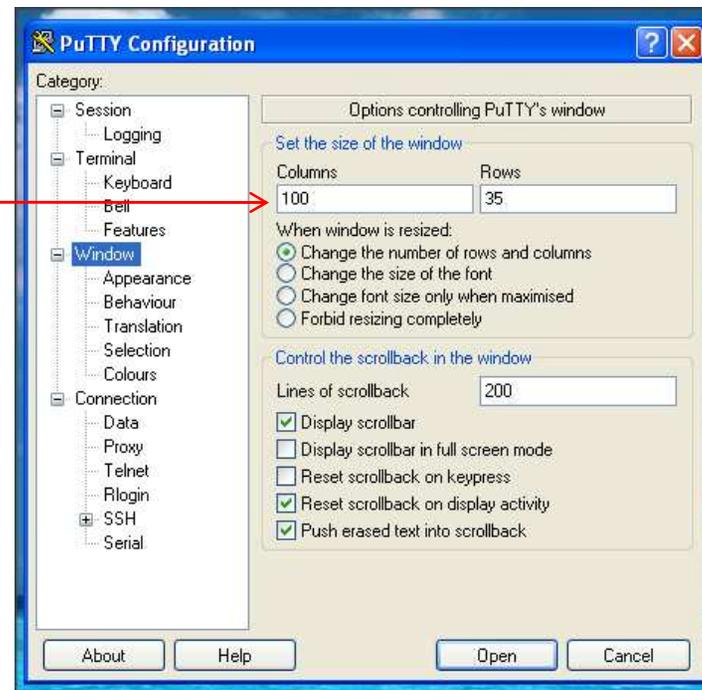
The screenshot shows the PuTTY Configuration dialog box. The 'Category' list on the left is expanded to 'SSH'. The 'Basic options for your PuTTY session' section is visible. The 'Host Name (or IP address)' field contains '192.168.1.11' and the 'Port' field contains '22'. The 'Connection type' section has 'SSH' selected. The 'Saved Sessions' list contains 'ISACA Solaris', 'Default Settings', 'ISACA Ubuntu', and 'Koopowitz (Internet)'. The 'Save' button is highlighted. Three callout boxes with red arrows point to these elements: '1. Type in the IP address' points to the Host Name field, '2. Type in a name for the session' points to the Saved Sessions list, and '3. Click the Save button' points to the Save button.

1. Type in the IP address
2. Type in a name for the session
3. Click the Save button

Set Up Our Tools - continued

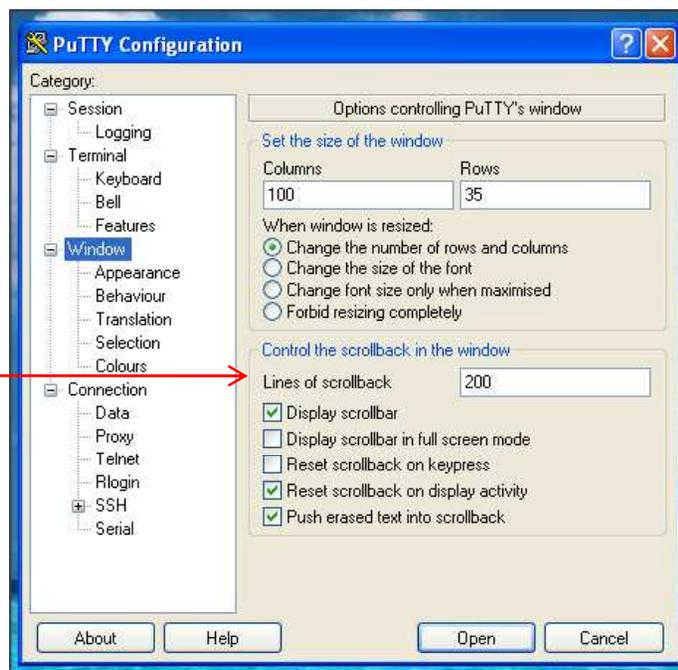
Additional settings that will help the display. Highlight the **Solaris** session you just saved and then click **Load**. Click on the **Window** category.

Modify these settings to suit your display



Set Up Our Tools - continued

Modify these settings to suit your display



When you are finished click on the “**Session**” category and then click “**Save**”. Repeat these two steps for the “**Ubuntu**” session you saved earlier.

Scanning Our Network

Why scan?

The process is essentially similar to a thief going through a neighborhood and checking every door and window on each house to find out which ones are open and which ones are locked.

Scanning explores the network so that one can build an inventory of hosts, services, and operating systems that are in use.

Scanning Our Network - continued

- Click the “**START**” button and then click “**Run...**”
- On the command line type **CMD** and then click **OK** – a **DOS** window should have opened up on your desktop.
- Change to the drive letter where you created the “**tools**” folder above – e.g. **c:**

Scanning Our Network - continued

Let's run **Scanline** now...

- Change to the new “**tools**” directory and then change to the **scanline** directory:

```
cd\tools
```

```
cd scanline
```

- Run the **scanline** program using the following parameters:

```
sl -f hostlist.txt -b -o isaca.txt -v
```

Scanning Our Network - continued

Let's review results of scan...

From the command line type the following command:

notepad isaca.txt

Scanning Our Network - continued

Run **NMAP** now...

From the command line...

- Change to the **nmap** directory:
`cd ..`
`cd nmap`
- Run the **NMAP** program using the following parameters:
`nmap -sV -sC -sS -T5 192.168.1.1/24 -oX myscan1.xml`
- Or you can use the following parameters:
`nmap -sV -sC -sS -T5 -iL hostlist.txt -oX myscan1.xml`

Scanning Our Network - continued

From Windows Explorer, browse to the `\tools\nmap` folder and double click on the file `myscan1.xml`. It will open up in Internet Explorer (or your default browser).

Scanning Our Network - continued

NMAP – continued

There are also canned scripts within **NMAP** that can be used. These are scripts that other users have written to automate a wide variety of tasks.

Auditing UNIX / Linux



Linux™



UNIX®

00011110 00011110 00011110 00011110 00011110 00011110 00011110 00011110

Setting Up To Audit UNIX / Linux

What do we need to get?

- We need a user account! A regular user account will do but we will need help from the Sysadmin in getting data from certain files that are only accessible using a privileged account.

Auditing UNIX / Linux

Where are the areas of risk in UNIX / Linux?

- User administration
- Network connectivity
- System administration
- File system
- Backups
- Physical security

The above areas are the most important and our focus will be on ensuring that controls are in place to mitigate risks in these areas.

- Any other risk areas that we are forgetting?

Auditing UNIX / Linux – User Administration

What should we focus on for user administration?

- Account authorization
- Password policy
- Strong passwords
- Unique user and group IDs
- Users sharing the same UID number
- Group membership
- System accounts
- Default umask settings for root, ftp user, and regular users
- Inactive accounts

Auditing **UNIX / Linux** – Running Our Scripts

Let's run **Putty** and log on to **UNIX** and **Linux**...

1. Double click the **Putty** shortcut icon you have on your desktop. Double click the **Solaris** line and this will connect you to the server. Type your user ID and password to log in.
2. Double click the **Putty** shortcut icon you have on your desktop. Double click the **Ubuntu** line this will connect you to the server. Type your user ID and password to log in.

Auditing **UNIX / Linux** - Commands

Let's familiarize ourselves with UNIX / Linux commands - common commands that are used to navigate/browse UNIX / Linux are:

- **passwd** – change a password
- **exit** – logs out of session
- **cat** – displays the contents of a file
- **head** – displays the first few lines of a file
- **tail** – displays the last few lines of a file

Auditing **UNIX** / **Linux** - Commands

Additional commands are:

- **more** – displays the contents of a file but pauses scrolling if more than one screen
- **ls -la** – displays a directory listing showing filenames and attributes
- **cd** – changes directory
- **rm** – deletes a file or directory
- **cp** – copy a file

Auditing **UNIX / Linux** - Commands cont'd

Additional commands are:

- **mv** – move a file (or rename it)
- **pwd** – displays current directory location
- **ps -ef** – displays information about processes/jobs running
- **grep -in** – search utility
- **find** – locate files
- **vi** – very basic editor

Auditing **UNIX / Linux** - Commands cont'd

Additional commands are:

- **nano** – more advanced editor in Linux
- **mkdir** – create a directory
- **chmod** – change permissions on a file
- **chown** – change ownership on a file
- **clear** – clears the screen
- **man** – displays help information for a command

Auditing **UNIX** / **Linux** - Permissions

File permissions – a VERY quick overview

File permissions give information about the file and what kind of file access (read, write or execute) is granted to users on the system.

```
-rw-r--r--  1 root root    724 2010-08-24 13:45 crontab
drwxr-xr-x  2 root root   4096 2011-01-28 19:07 cron.weekly
drwxr-xr-x  2 root root   4096 2011-01-30 22:00 dbconfig-common
drwxr-xr-x  3 root root   4096 2011-01-28 18:55 dbus-1
-rw-----  1 root root    143 2011-03-16 22:38 ddclient.conf
-rw-r--r--  1 root root   2969 2010-09-27 14:04 debconf.conf
-rw-r--r--  1 root root     12 2010-04-23 02:45 debian_version
```

Auditing **UNIX / Linux** - Permissions cont'd

The first character of the file's mode field indicates the type of file. Some modes are:

- - = indicates a plain file
- d = indicates a directory
- l = indicates a symbolic link

```
-rw-r--r-- 1 root root 724 2010-08-24 13:45 crontab
drwxr-xr-x 2 root root 4096 2011-01-28 19:07 cron.weekly
drwxr-xr-x 2 root root 4096 2011-01-30 22:00 dbconfig-common
drwxr-xr-x 3 root root 4096 2011-01-28 18:55 dbus-1
-rw----- 1 root root 143 2011-03-16 22:38 ddclient.conf
-rw-r--r-- 1 root root 2969 2010-09-27 14:04 debconf.conf
-rw-r--r-- 1 root root 12 2010-04-23 02:45 debian_version
```

Auditing **UNIX / Linux** - Permissions cont'd

The next nine characters taken in groups of 3 indicate who can do what with the file. There are three kinds of permissions:

- r = permission to read
- w = permission to write
- x = permission to execute

```
-rw-r--r-- 1 root root 724 2010-08-24 13:45 crontab
drwxr-xr-x 2 root root 4096 2011-01-28 19:07 cron.weekly
drwxr-xr-x 2 root root 4096 2011-01-30 22:00 dbconfig-common
drwxr-xr-x 3 root root 4096 2011-01-28 18:55 dbus-1
-rw----- 1 root root 143 2011-03-16 22:38 ddclient.conf
-rw-r--r-- 1 root root 2969 2010-09-27 14:04 debconf.conf
-rw-r--r-- 1 root root 12 2010-04-23 02:45 debian_version
```

Auditing **UNIX / Linux** - Permissions cont'd

Facts about permissions:

- One can have execute access without read. If one has read access but not execute access then one could copy the file and assign one's own execute permission – ownership will change if one does copy the file. Some versions of UNIX / Linux do require that an executable script must have read access as well.

```
-rw-r--r-- 1 root root 724 2010-08-24 13:45 crontab
drwxr-xr-x 2 root root 4096 2011-01-28 19:07 cron.weekly
drwxr-xr-x 2 root root 4096 2011-01-30 22:00 dbconfig-common
drwxr-xr-x 3 root root 4096 2011-01-28 18:55 dbus-1
-rw----- 1 root root 143 2011-03-16 22:38 ddclient.conf
-rw-r--r-- 1 root root 2969 2010-09-27 14:04 debconf.conf
-rw-r--r-- 1 root root 12 2010-04-23 02:45 debian_version
```

Auditing **UNIX** / **Linux** - Permissions cont'd

There are three classes of permissions as well:

- owner = file's owner
- group = users who are in the file's group
- other (or world) = everybody else on the system (except the superuser)

Most people think that file permissions are pretty basic, however, many systems have had security breaches because file permissions were not set correctly.

```
-rw-r--r--  1 root root    724 2010-08-24 13:45 crontab
drwxr-xr-x  2 root root   4096 2011-01-28 19:07 cron.weekly
drwxr-xr-x  2 root root   4096 2011-01-30 22:00 dbconfig-common
drwxr-xr-x  3 root root   4096 2011-01-28 18:55 dbus-1
-rw-----  1 root root    143 2011-03-16 22:38 ddclient.conf
-rw-r--r--  1 root root   2969 2010-09-27 14:04 debconf.conf
-rw-r--r--  1 root root     12 2010-04-23 02:45 debian_version
```

Auditing **UNIX / Linux** - Ownership

File Ownership and Access – a really QUICK overview

Only a superuser can change the ownership of a file. In earlier versions of UNIX / Linux, all users could change the ownership of a file that they owned – this allowed one to “give away” a file to someone else. *Unfortunately this is a big security risk.*

There are two attributes in files – the owner and the group that has access to the file. If one is the file owner then one can change the group of a file provided you are in the group to which you are trying to change the file. All superusers can change the group of a file.

```
-rw-r--r--  1 root root    724 2010-08-24 13:45 crontab
drwxr-xr-x  2 root root   4096 2011-01-28 19:07 cron.weekly
drwxr-xr-x  2 root root   4096 2011-01-30 22:00 dbconfig-common
drwxr-xr-x  3 root root   4096 2011-01-28 18:55 dbus-1
-rw-----  1 root root    143 2011-03-16 22:38 ddclient.conf
-rw-r--r--  1 root root   2969 2010-09-27 14:04 debconf.conf
-rw-r--r--  1 root root     12 2010-04-23 02:45 debian_version
```

Auditing **UNIX** / Linux – Running Our Scripts

Now we need to **FTP** the scripts to each server.

- Double click the **PSFTP** icon you created on your desktop
- At the **psftp>** prompt type the following command to connect to **UNIX**:

open 192.168.1.11

Auditing **UNIX** / Linux – Running Our Scripts

- You'll be prompted for your user ID and password.
- At the **psftp>** prompt type the following commands. Press enter after each line:

```
put c:\tools\unix-linux-scripts\audit.sh  
put c:\tools\unix-linux-scripts\audit1.sh  
put c:\tools\unix-linux-scripts\audit2.pl
```

- At the **psftp>** prompt type **quit** to end your FTP session and close the window.

Auditing **UNIX** / Linux – Running Our Scripts

Now we can execute the scripts – let's jump over to the **Putty** window that you started for **Solaris (192.168.1.11)**.

Type the command **ls -la** and you should see the 3 files in your directory. You'll also note that the permissions on the files do not allow them to be executed. We need to modify these permissions to **744**.

```
-rw-r--r-- 1 isacal other 2 2011-10-05 15:35 .rhosts
drwxr-xr-x 2 isacal other 5 2011-10-14 16:04 .vp
-rw-r--r-- 1 isacal other 3452 2011-10-14 16:04 .xsession-errors
-rw-r--r-- 1 isacal other 831 2011-11-01 10:40 audit.sh
-rw-r--r-- 1 isacal other 34132 2011-11-01 10:41 audit1.sh
-rw-r--r-- 1 isacal other 17244 2011-11-01 10:41 audit2.pl
drwxr-xr-x 2 isacal other 6 2011-10-14 16:04 Desktop
drwxr-xr-x 6 isacal other 6 2011-10-14 16:03 Documents
drwxr-xr-x 2 isacal other 2 2011-10-14 16:03 Downloads
-rw-r--r-- 1 isacal other 960 2011-09-02 15:09 local.cshrc
-rw-r--r-- 1 isacal other 988 2011-09-02 15:09 local.login
```

Auditing **UNIX** / Linux – Running Our Scripts

Before we modify the permissions, we'll need to **su** to **root** so that we can run the scripts since some of the files referenced in the scripts require root access.

- At the prompt type the following:

```
su root
```

- Enter the password of “**solaris1**” (without the quotes).
- Now let's modify the permissions. Enter the following command:

```
chmod 744 audit*
```

This will modify all the audit scripts it finds with the **744** permissions.

Auditing **UNIX** / Linux – Running Our Scripts

Now we can run the scripts...

- At the prompt type the following command and press enter:

```
./audit.sh
```

- When the script completes you should see 2 log files in the directory. Execute the command below to see the files:

```
ls -la
```

- We need to modify the permissions of the log files so that we can FTP them. Run the following command and press enter:

```
chmod 644 solaris*.log
```

Auditing **UNIX** / Linux – Running Our Scripts

Let's **FTP** the log files back to our laptop...

- Double click the **PSFTP** icon on the desktop to open the connection. Type:

open 192.168.1.11

- Press enter and then enter your login information. Type the following command and press enter:

get solaris.audit1.log

get solaris.audit2.*nnnn*.log

The ***nnnn*** above is unique and you can get this number if you list (**ls**) the files right after you connect up using **FTP**.

Auditing UNIX / Linux – Running Our Scripts

Now we need to **FTP** the scripts to each server.

- Double click the **PSFTP** icon you created on your desktop
- At the **psftp>** prompt type the following command to connect to **Linux**:

open 192.168.1.10

Auditing UNIX / Linux – Running Our Scripts

You'll be prompted for your user ID and password.

- At the **psftp>** prompt type the following commands. Press enter after each line:

```
put c:\tools\unix-linux-scripts\audit.sh
```

```
put c:\tools\unix-linux-scripts\audit1.sh
```

```
put c:\tools\unix-linux-scripts\audit2.pl
```

- At the **psftp>** prompt type **quit** to end your FTP session and close the window.

Auditing UNIX / Linux – Running Our Scripts

Now we can execute the scripts – let's jump over to the **Putty** window that you started for **Ubuntu (192.168.1.10)**.

Type the command **ls -la** and you should see the 3 files in your directory. You'll also note that the permissions on the files do not allow them to be executed. We need to modify these permissions to **744**.

```
isaca1@testubuntu:~$ ls -la
total 132
drwxr-xr-x  4 isaca1 isaca1 4096 2011-11-01 10:30 .
drwxr-xr-x 34 root    root    4096 2011-09-08 15:14 ..
-rw-r--r--  1 isaca1 isaca1 34132 2011-11-01 10:30 audit1.sh
-rw-r--r--  1 isaca1 isaca1 17244 2011-11-01 10:30 audit2.pl
-rw-r--r--  1 isaca1 isaca1  831 2011-11-01 10:30 audit.sh
-rw-r--r--  1 isaca1 isaca1 5333 2011-10-26 11:44 .bash_history
-rw-r--r--  1 isaca1 isaca1  220 2011-09-08 15:05 .bash_logout
-rw-r--r--  1 isaca1 isaca1 3499 2011-10-25 16:19 .bashrc
drwx----- 2 isaca1 isaca1 4096 2011-09-30 14:48 .cache
```

Auditing UNIX / Linux – Running Our Scripts

We'll need to use the **sudo** command to modify the permissions and run the scripts since some of the files referenced in the scripts require root access.

- Let's modify the permissions. Enter the following command:

```
sudo chmod 744 audit*
```

This will modify all the audit scripts it finds with the **744** permissions.

Auditing UNIX / Linux – Running Our Scripts

Now we can run the scripts...

- At the prompt type the following command and press enter:

```
sudo ./audit.sh
```

- When the script completes you should see 2 log files in the directory. Execute the command below to see the files:

```
ls -la
```

- We need to modify the permissions of the log files so that we can FTP them. Run the following command and press enter:

```
sudo chmod 644 testubuntu*.log
```

Auditing UNIX / Linux – Running Our Scripts

Let's **FTP** the log files back to our laptop...

- Double click the **PSFTP** icon on the desktop to open the connection. Type:

open 192.168.1.10

- Press enter and then enter your login information. Type the following command and press enter:

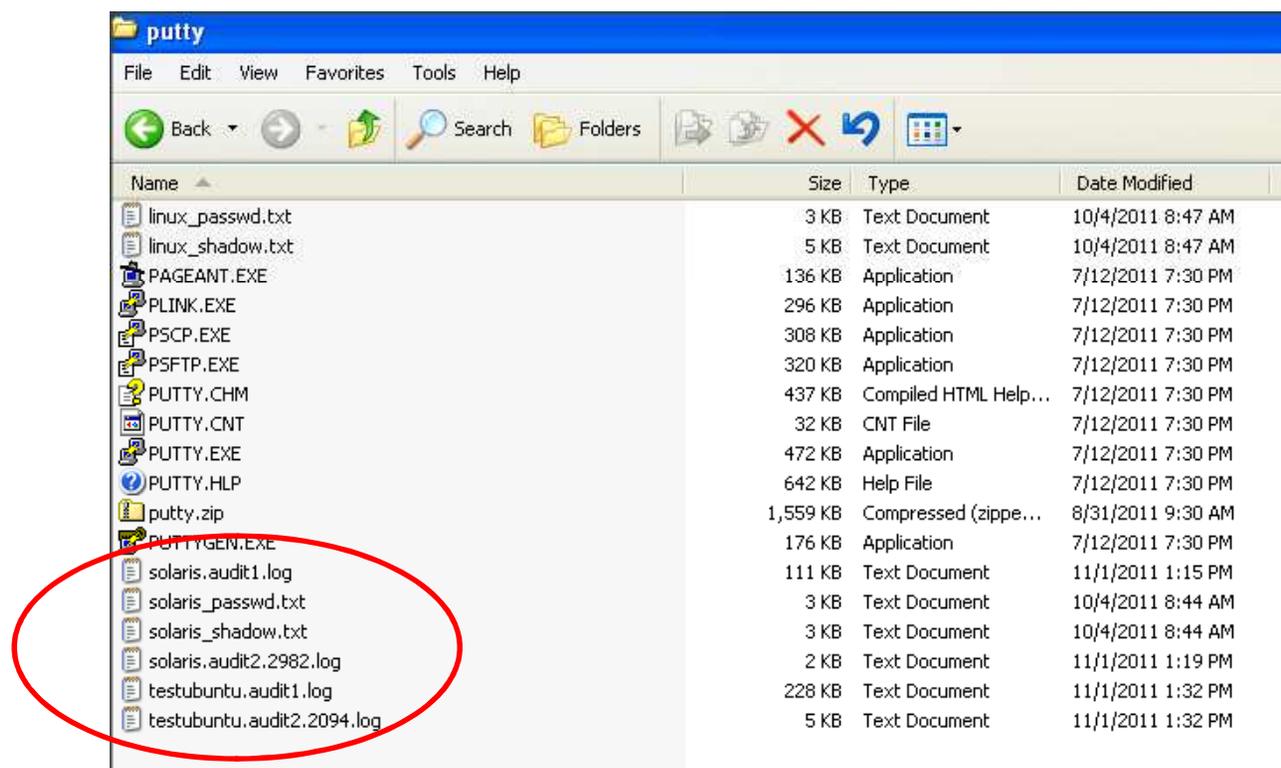
get testubuntu.audit1.log

get testubuntu.audit2.*nnnn*.log

The ***nnnn*** above is unique and you can get this number if you list (**ls**) the files right after you connect up using **FTP**.

Auditing **UNIX / Linux** – Running Our Scripts

Open up Windows Explorer and browse to the **\tools\putty** folder and you should see the log files.



Auditing **UNIX / Linux** – Account Authorization

The Sysadmins should document and enforce account management procedures. These should include the creation of new user accounts, moving a user to a new group or role, deletion of users, and handling of dormant or inactive accounts.

Auditing **UNIX / Linux** – Passwords

Password policy and strong passwords in place – how do we find out what the default policy is for all users?

UNIX

more /etc/default/passwd

Linux

more /etc/login.defs

Auditing **UNIX** / Linux – Passwords

What are in the files?

UNIX =

```
MAXWEEKS=  
MINWEEKS=  
PASSLENGTH=6  
  
# NAMECHECK enables/disables login name checking.  
# The default is to do login name checking.  
# Specifying a value of "NO" will disable login name checking.  
#  
#NAMECHECK=NO  
  
# HISTORY sets the number of prior password changes to keep and  
# check for a user when changing passwords. Setting the HISTORY  
# value to zero (0), or removing/commenting out the flag will  
# cause all users' prior password history to be discarded at the  
# next password change by any user. No password history will  
# be checked if the flag is not present or has zero value.  
# The maximum value of HISTORY is 26.  
#  
# This flag is only enforced for user accounts defined in the  
# local passwd(4)/shadow(4) files.  
#  
#HISTORY=0  
#  
# Password complexity tunables. The values listed are the defaults  
# which are compatible with previous releases of passwd.  
# See passwd(1) and pam_authtok_check(5) for use warnings and  
# discussion of the use of these options.  
#  
#MINDIFF=3  
#MINALPHA=2  
#MINNONALPHA=1  
#MINUPPER=0  
#MINLOWER=0  
#MAXREPEATS=0  
#MINSPECIAL=0  
#MINDIGIT=0  
#WHITESPACE=YES  
#
```

Sets defaults in terms of the number of weeks – even though actual values on user accounts are kept in days. If the policy is maximum days of 90 for a password then MAXWEEKS should be 13 (91 days). Suggested values are MAXWEEKS=13, MINWEEKS=1, WARNWEEKS=1, PASSLENGTH=8.

If your company policy is to keep a history of passwords used then this line should be uncommented and the value replaced. Suggested value is HISTORY=12.

Password complexity should be enabled by uncommenting these options. Suggested values are MINDDIFF=3, MINALPHA=3, MINUPPER=1, MINLOWER=1, MAXREPEATS=0, MINSPECIAL=1.

Auditing UNIX / Linux – Passwords

Linux =

```
# Password aging controls:
#
# PASS_MAX_DAYS Maximum number of days a password may be used.
# PASS_MIN_DAYS Minimum number of days allowed between password changes.
# PASS_WARN_AGE Number of days warning given before a password expires.
#
PASS_MAX_DAYS 99999
PASS_MIN_DAYS 0
PASS_WARN_AGE 7
#
# Min/max values for automatic uid selection in useradd
#
UID_MIN 1000
UID_MAX 60000
# System accounts
#SYS_UID_MIN 100
#SYS_UID_MAX 999
#
# Min/max values for automatic gid selection in groupadd
#
GID_MIN 1000
GID_MAX 60000
# System accounts
#SYS_GID_MIN 100
#SYS_GID_MAX 999
#
# Max number of login retries if password is bad. This will most likely be
# overridden by PAM, since the default pam_unix module has it's own built
# in of 3 retries. However, this is a safe fallback in case you are using
# an authentication module that does not enforce PAM_MAXTRIES.
#
LOGIN_RETRIES 5
#
# Max time in seconds for login
#
LOGIN_TIMEOUT 60
```

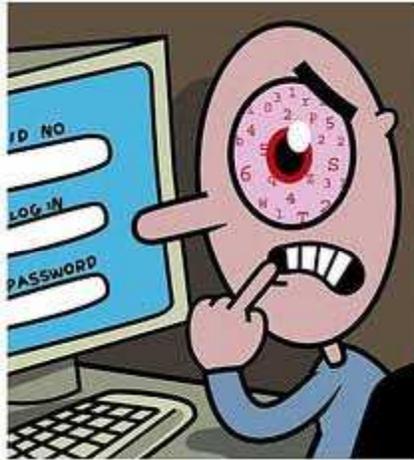
Set the default values to match your password policy. Suggested values are PASS_MAX_DAYS 90, PASS_MIN_DAYS 7, PASS_WARN_AGE 7.

If you are using the PAM authentication module then check the /etc/pam.d/common-password file. Add the minlen=8 to the end of the line below to force a minimum of an 8 character length password.

```
# here are the per-package modules (the "Primary" block)
password [success=1 default=ignore] pam_unix.so obscure sha512
# here's the fallback if no module succeeds
```

Auditing **UNIX / Linux** – Strong Passwords

How do we verify strong passwords are in place?



John the Ripper is our tool!

Auditing **UNIX** / Linux – Strong Passwords

Let's log on to UNIX first so that we can get the password file that contains the password hashes – this is the **/etc/shadow** file. We'll also copy the **/etc/passwd** file which contains user names and other useful information.

Auditing **UNIX** / Linux – Strong Passwords

Normal users do not (or should not) have ANY access to the **/etc/shadow** file, so in order for us to copy it, we'll need to have the Sysadmin copy it and send it to us, or if you have **SU** access (switch user), we can switch users to a privileged account that will allow us to copy the file from its current location to our home directory.

Auditing **UNIX** / Linux – Strong Passwords

Let's log in... use **Putty** for this... select the **Solaris** session you saved and click open. Type in your userid and password when prompted.

- At the **UNIX** prompt, type the following command:

```
cp /etc/shadow solaris_shadow.txt
```

What did you see?

Auditing **UNIX** / Linux – Strong Passwords

Since we don't have permissions to do this, have the Sysadmin do it for you, or if you have **su** access for a privileged account then use it.

Since we all have access to the root account for this class – let's go that route:

- At the UNIX prompt, type the following command:

su root

Auditing **UNIX** / Linux – Strong Passwords

- Now type the password for root which is “solaris1” (without the quotes).
- Type the following commands (one at a time):

```
cp /etc/passwd solaris_passwd.txt  
cp /etc/shadow solaris_shadow.txt
```

What did you see?

- Run the following command:

```
ls -la
```

Auditing **UNIX** / Linux – Strong Passwords

Because we used the root account to do the copy, our permissions on the **etc/shadow** file still default to the system permissions. In order for us to **FTP** this file back to our Windows machine we will need to change its permissions.

Auditing **UNIX** / Linux – Strong Passwords

- Type the following command to modify the permissions to read only for everyone:

```
chmod 444 solaris_shadow.txt
```

What permissions do you see now?

- Run the following command:

```
ls -la
```

Auditing UNIX / Linux – Strong Passwords

Let's log into Linux now... use Putty for this... select the Linux session you saved and click open. Type in your userid and password when prompted.

- At the Linux prompt, type the following command:

```
cp /etc/shadow linux_shadow.txt
```

What did you see?

- Type the following command:

```
ls -la
```

Auditing UNIX / Linux – Strong Passwords

Since we don't have permissions to do this, have the Sysadmin do it for you, or if you have **sudo** access for a privileged account then use it.

Since we all have SUDO access for this class – let's go that route:

- At the Linux prompt, type the following commands (one at a time):

```
cp /etc/passwd linux_passwd.txt  
sudo cp /etc/shadow linux_shadow.txt
```

- You'll be prompted to re-enter your “**sudo**” password at this point – this is your login password for the account you used.

What did you see?

- Type the following command:

```
ls -la
```

Auditing UNIX / Linux – Strong Passwords

Because we used the root account to do the copy of the **etc/shadow** file, our permissions on the file still default to the system permissions. In order for us to **FTP** this file back to our Windows machine we will need to change its permissions.

Auditing UNIX / **Linux** – Strong Passwords

- Type the following command to modify the permissions to read only for everyone:

```
sudo chmod 644 linux_shadow.txt
```

What permissions do you see now?

- Type the following command:

```
ls -la
```

Auditing **UNIX / Linux** – Strong Passwords

Running John the Ripper:

We need to run it from the actual server so log on to the Linux server using Putty.

- Combine the **passwd** and **shadow** files into one file at this time. Run the “**unshadow**” command using the following parameters:
- For Linux type the following command:

```
sudo unshadow linux_passwd.txt linux_shadow.txt > linux.txt
```

Auditing **UNIX / Linux** – Strong Passwords

Run John the Ripper using the following commands:

Linux:

```
sudo john linux.txt
```

What do you see?

Auditing **UNIX / Linux** – Strong Passwords

To show the cracked passwords at a later time, you can run the following command:

```
sudo john --show linux.txt
```

Auditing **UNIX / Linux** – Home Directories

Ensure that each user's home directory exists – if a home directory does not exist then that user will be placed in “/” and will not be able to write any files or have any local environment variable set.

In order to verify this, you will have to change to the home directory to see if it exists.

Ensure that the ownership of user's home directory belongs to the user since they will be accountable for all the files in their directory.

Auditing **UNIX / Linux** – Home Directories

- For example, in Linux, change to the **/home** directory using the command:

cd /home

- Then type the command **ls -la**.
- Compare the output of this listing to the **/etc/passwd** file to ensure that every user in the **passwd** file has a corresponding home directory and they own their home directories.

Auditing **UNIX / Linux** – Unique User / Group IDs

Verify that each user and group account has its own unique user identifier (**UID**) and does not match any system account **UIDs / GIDs**. Also make sure that all user/group names are unique and there aren't any duplicates and that each user is assigned a unique home directory.

If you type the command **cat /etc/passwd** you will see that each user account is assigned a user ID – this is the 3rd parameter on each line in the file. Home directories are the 6th parameter (or 2nd to last parameter).

Auditing **UNIX / Linux** – Unique User / Group IDs

You can **FTP** this file to your laptop and open the file in MS Excel – sort the file by the user name and **UID** columns and check to see if there are duplicates.

****NOTE:** Make sure that there aren't any other users in the file that have a UID of 0 (zero) – the only user that should have a UID of 0 is the “root” account.

Auditing **UNIX / Linux** – Group Membership

Verify that each group has appropriate membership and that each group account has its own unique group identifier (GID) and there are no duplicate group names.

If you type the command **cat /etc/group** you will see that each group account is assigned a GID – this is the 3rd parameter on each line in the file.

Auditing **UNIX / Linux** – Group Membership

You can **FTP** this file to your laptop and open the file in MS Excel – sort the file by the name column and the **GID** column and check to see if there are duplicates.

****NOTE:** Make sure that there aren't any other groups in the file that have a GID of 0 (zero). Make sure that the root group doesn't contain any users – question the Sysadmin if it does.

Auditing **UNIX / Linux** – Disable System Accounts

Ensure that all default system accounts are disabled (locked). Default system accounts are installed to manage applications and are usually disabled and do not allow anyone to log into these accounts or running an interactive shell.

Usually the password field is set to an invalid value to disable an account but another way is to set the shell field to “**false**”.

Auditing **UNIX / Linux** – Disable System Accounts

If you type the command **cat /etc/passwd** you will see that each user account is assigned a shell location - this is the last parameter on each line in the file. A value of “**/usr/bin/false**” should be in place for all system accounts that are not allowed to log in. Most user accounts will have **/bin/bash** or **/bin/ksh** as the shell.

Auditing **UNIX** / Linux – Inactive Accounts Locked

Generally, user accounts should automatically be locked if they haven't been used for a defined number of days. Some UNIX / Linux operating systems have the capability to automatically lock accounts if there has been no activity for a given number of days. If the file does not exist, then it should be created to establish default user settings when a new account is created.

For Solaris:

```
grep definact /usr/sadm/defadduser
```

Auditing **UNIX / Linux** – Inactive Accounts Locked

If an account is locked then the password field in the `/etc/shadow` file will contain the letters “*LK*” (UNIX) at the beginning of the password, and a “!” in the first position of the password field (Linux)... this essentially prevents the user from logging in. However, accounts with no activity should also be expired:

For UNIX, the following command will expire the account on the given date:

```
usermod -e 11/08/11 isaca1
```

For Linux, the following command will expire the account on the given date:

```
usermod -e 2011-11-08 isaca1
```

Check the `/etc/shadow` file to see the expiry date for locked accounts.

Auditing **UNIX / Linux** – Inactive Accounts Locked

In order to view the last login for a user account (UNIX or Linux) one needs to use the “**last**” command:

Display the last login information for user **isaca1**:

```
last isaca1
```

Displays the last login information for all users:

```
last
```

Auditing **UNIX / Linux** – Inactive Accounts Locked

One could also use the “**finger**” command (if installed) to display information about users:

Display information about user **isaca1**:

```
finger isaca1
```

Display information about all users:

```
finger
```

Auditing **UNIX / Linux** – Umask Settings

The **umask** (shorthand for "user file-creation mode mask") is a four-digit octal number that UNIX / Linux uses to determine the file permission for newly created files/directories. This value specifies the permissions you do NOT want to give to newly created files/directories.

By default, most UNIX / Linux versions specify a permission of **666** for new files and **777** for new directories. If a **umask** value is defined then this value is masked on the new file/directory by subtracting its value from the default permission.

Auditing **UNIX / Linux** – Umask Settings

For instance, if a **umask** of **022** is given to each user then all files created will have a permission of **644** and a directory permission of **755**.

UNIX and Linux tend to default to a **umask** setting of **022** – a more secure setting should be **077** which will ensure that all files/directories created will not be readable by any other user on the system.

Auditing **UNIX / Linux** – Umask Settings

To check the default **umask** setting, type the following command:

more /etc/default/login – UNIX

more /etc/profile – UNIX

more /etc/.login – UNIX

more /etc/profile – Linux

more /etc/login.defs – Linux

Auditing **UNIX / Linux** – Umask Settings

Unfortunately, each user has the capability to override the default **umask** setting by configuring their individual login settings. The Sysadmin should check the following files in each users home directory and if the **umask** is not **077** (or at a minimum **022**) then the Sysadmin should encourage the user to make it more restrictive.

more <users home path>/.**login**

more <users home path>/.**profile**

more <users home path>/**local.login**

Auditing **UNIX / Linux** – Umask Settings

FTP umask settings should also be checked since they do not inherit the users standard **umask** setting. Check the following files for the **umask** setting for **FTP**:

more /etc/ftpd/ftpaccess – UNIX

more /etc/vsftpd.conf – Linux

Auditing **UNIX** / **Linux** – Network Connectivity

What should we focus on for network connectivity?

- Minimize services
- Review trusted remote hosts and users
- Vendor access
- Warning banners

Auditing **UNIX / Linux** – Minimize Services

Disable all services that are not required for normal system operation. This will prevent the exploitation of vulnerabilities discovered at a later date, or are generally known today and remain unpatched on the system.

The latest versions of some UNIX systems allow the Sysadmin to harden the OS baseline with simple commands. For instance, in Solaris the “**netservices limited**” command will lock down the server. Most servers have to be manually locked down by disabling all services that are not required.

Auditing **UNIX / Linux** – Minimize Services

To see what services are running, run the following command:

```
netstat -an | grep LISTEN
```

Auditing **UNIX / Linux** – Minimize Services

Services to watch out for are:

- FTP – make sure it is configured correctly
- TFTP – shouldn't be used
- Sendmail (or any mail daemon) – shouldn't be running all the time and should only be used on a mail server
- NIS, NIS+ (LDAP should be used now)
- UUCP – replaced with better/faster/more secure protocols

Auditing **UNIX / Linux** – Trusts

What is a trusted host and user?

If one host trusts another host then any user that has the same user name on both hosts can log into the other host without typing a password.

A trusted user means that if a user on another host is designated as a trusted user on your current host, then that person can log into your host without providing a password.

Auditing **UNIX / Linux** – Trusts

rlogin allows one to easily jump from server to server, and **rsh** allows one to remotely run commands on a server without logging into that server. Trusts allow a user to only provide an initial password to log on and then springboard to other servers without providing a password to get on.

You can't always trust a host, and you can't trust the users on that host.

Auditing **UNIX / Linux** – Trusts

How does one form a trust relationship with another host or user?

/etc/hosts.equiv – system trusted hosts and users

~/.rhosts – user defined trusted hosts and users

Auditing **UNIX** / Linux – Trusts

In UNIX, one can disable **.rhosts** usage in the **/etc/pam.conf** file.
Run the following command:

```
grep "^#" /etc/pam.conf | grep "pam_rhosts_auth"
```

If the **.rhosts** usage has been disabled then one should see two lines displayed showing them commented out as follows:

```
#rlogin auth sufficient pam_rhosts_auth.so.1  
#rsh auth sufficient pam_rhosts_auth.so.1
```

Auditing UNIX / Linux – Trusts

In Linux, one can disable `.rhosts` usage in the files located in the `/etc/pam.d` directory. If this directory exists, then the system ignores the contents of the `/etc/pam.conf` file.

Auditing **UNIX / Linux** – Trusts

Verify that the `/etc/hosts.equiv` file does not exist, and if it does, then ensure that there are valid reasons to trust a particular host or hosts. Also ensure that a “+” by itself does not exist in this file as this means that any host out there is trusted.

Trusts are transitive as well – this means that if you trust a host then you also trust hosts that the other host trusts. This applies to users as well.

Auditing **UNIX / Linux** – Trusts

Verify that **.netrc** files do not exist in user home directories. A **.netrc** presents a significant risk since it stores unencrypted passwords for logging into remote hosts. If they do exist then they need to have restrictive permissions set so that other user's can't view the contents.

Sysadmins should monitor for the existence of **.netrc** files and validate with the user that they are in fact necessary and educate them on the importance of restrictive permissions.

Auditing **UNIX / Linux** – Vendor Access

Any vendor user accounts on the system should be disabled and only enabled if a vendor needs to do some work. One can assign an expiration date to the account so that it will automatically lock itself after that date has passed.

Unmonitored accounts, such as vendor accounts, are an invitation for someone to use them without authorization.

Auditing **UNIX / Linux** – Warning Banners

Displaying a warning banner prior to a normal user login may assist the prosecution of potential hackers/trespassers on the system. Changing some of the login banners also as the side effect of hiding OS version information and other detailed system information which could allow an attacker to research potential exploits for that OS version.

Auditing **UNIX** / **Linux** – Warning Banners

Inspect the following files to ensure appropriate information is being displayed:

/etc/motd

/etc/issue

/etc/ftpd/banner.msg – UNIX

/etc/vsftpd.conf – Linux

/etc/default/telnetd - UNIX

Auditing **UNIX / Linux** – System Administration

What should we focus on for system administration?

- Procedures in place to monitor the systems on a regular basis
- Procedures in place to apply latest patches
- Logging enabled for all services – preferably to a remote host used just for logging
- Procedures in place to review all system logs
- Root password (for UNIX and certain flavors of Linux)
- Root default group
- Root home directory
- Root path
- Root login (for UNIX and certain flavors of Linux)

Auditing **UNIX** / **Linux** – Logging

Logging should be enabled and monitored on a regular basis for all services on a server. It is also a good practice to copy all system logs to a secure, centralized log server. Some logs to monitor are:

/var/adm/messages – UNIX

/var/adm/lastlog – UNIX

/var/adm/sulog – UNIX

/var/adm/loginlog – UNIX

/var/cron/log - UNIX

/var/log/auth.log – Linux

/var/log/syslog – UNIX and Linux

/var/log/vsftpd.log - Linux

Auditing **UNIX / Linux** – Root Password

All UNIX systems have a superuser account which is normally called “**root**”. If one browses the **/etc/passwd** file the root account will have a UID of **0** (zero). The password for this account is generally called the “**root password**”. Always make sure that the **root** password is strong and that all password complexity rules are enforced for it.

Certain flavors of Linux do not have a superuser account called “**root**”, or if they do it is usually locked and no one can log on with it – one should ensure that the **/etc/passwd** file does not contain any other user or system accounts with a UID of **0**.

Auditing **UNIX / Linux** – Root Default Group

Ensure that the default **GID** for **root** is **0**. Older systems used to assign the root account to the “**other**” group which may be shared by other users on the system. All current UNIX / Linux flavors now assign the **root** account to the **root** group which has a **GID** of **0**.

Run the following command to verify that the **GID** for root is 0:

more /etc/passwd

Auditing **UNIX / Linux** – Root Home Directory

Some older versions of UNIX assigned a home directory of “/” to **root**. This doesn’t lend itself well to ensuring that **root’s** home directory is restricted to only **root**.

Ensure that the home directory for **root** is an actual directory (preferably **/root**) and is secured with file permission of **700** – read/write/execute by owner (**root**).

Run the following command to verify that the home directory for **root** is set to an actual directory:

```
more /etc/passwd
```

Auditing **UNIX / Linux** – Root Home Directory

Verify that the home directory is secured appropriately by using the command:

```
ls -la /
```

The command will display the contents of the / directory. Look for the **root** directory in this display and review the permissions on the directory.

Auditing **UNIX / Linux** – Root Path

Verify that path for the **root** user does not contain the current working directory (a “.”) or other writable directory in the executable path.

An attacker could gain superuser access by forcing an administrator operating as **root** to execute a trojan horse program.

Auditing **UNIX** / **Linux** – Root Path

To verify the path statement you will need to **su** to **root** (for UNIX):

```
su root  
echo $PATH
```

For Linux, run the following command:

```
sudo echo $PATH
```

Auditing **UNIX / Linux** – Root Login

The **root** account must be restricted from logging in from any location except for the console. This can be disabled by denying root logins in SSH.

Verify that the **/etc/ssh/sshd_config** has the “**PermitRootLogin no**” parameter set (UNIX and Linux).

Verify that the **/etc/default/login** file has the “**CONSOLE=/dev/console**” parameter configured (UNIX).

Auditing **UNIX** / **Linux** – File System

What should we focus on for the file system?

- Procedures in place to monitor files on a regular basis
- Verify world writeable files/directories are appropriate
- SUID and SGID files
- Sticky bits on world writeable directories
- Permissions on user home directories
- Permissions on user dot files

Auditing **UNIX / Linux** – World Writeable Files / Directories

What's wrong with world writeable files / directories?

Data in these files can be modified and compromised by any user on the system. World writeable files may also indicate an incorrectly written script or program that could potentially be the cause of a larger compromise to the system's integrity. Check with the Sysadmin about all the identified files to ensure they are legitimate.

Since world writeable files may reside in directories that a regular user may not have access to, these commands need to be run as **su** or **sudo**.

Auditing **UNIX / Linux** – World Writeable Files / Directories

The command to find world writeable files is (**su** to root for Solaris or use **sudo** for Linux):

```
find / -type f -perm -o+w -exec ls -l {} \;
```

The command to find world writeable directories is:

```
find / -type d -perm -o+w -exec ls -ld {} \;
```

Auditing **UNIX / Linux** – SUID and SGID Files

What exactly is an SUID and SGID file?

Processes executing these programs can assume another **UID** or **GID** when they're running. A program that changes its **UID** is called a **SUID** program (*set-UID*); a program that changes its **GID** is called a **SGID** program (*set-GID*). *A program can be both **SUID** and **SGID** at the same time.* The most common reason for a **SUID / SGID** program is to enable users to perform functions (such as changing their password) that require root privileges. Check with the Sysadmin about all the identified files to ensure they are legitimate.

Since **SUID / SGID** files may reside in directories that a regular user may not have access to, these commands need to be run as **su** or **sudo**.

Auditing **UNIX / Linux** – SUID and SGID Files

The command to find **SUID** files is (**su** to root for Solaris or **sudo** for Linux):

```
find / -type f -perm -04000 -exec ls -lg {} \;
```

The command to find **SGID** files is:

```
find / -type f -perm -02000 -exec ls -lg {} \;
```

Auditing **UNIX / Linux** – Sticky Bits

Why should the sticky bit be set on world writeable directories?

If a directory is world writeable then any user that has access to the directory can delete or overwrite ANY file in that directory. By setting the sticky bit on the directory then only the owners of a file can remove or overwrite the file. Check with the Sysadmin about all the identified directories to ensure they should not have the sticky bit set.

Auditing **UNIX / Linux** – Sticky Bits

Since world writeable directories may not be accessible by a regular user, these commands need to be run as **su** or **sudo**.

The command to find world writeable directories is:

```
find / -type d -perm -o+w -exec ls -ld {} \;
```

Auditing **UNIX / Linux** – User Home Directories

Why check user home directory permissions?

Group or world writeable user home directories may enable malicious users to steal or modify other users data or perhaps gain access to another user's system privileges.

Check with the Sysadmin about all the identified directories to ensure they communicate with the owners of the home directories about the risks of having their directories group or world writeable.

Auditing **UNIX / Linux** – User Home Directories

Since world writeable directories may not be accessible by a regular user, these commands need to be run as **su** or **sudo**.

The command to find world writeable directories is:

```
find / -type d -perm -o+w -exec ls -ld {} \;
```

Auditing **UNIX / Linux** – User dot Files

What are user dot files?

These are files that reside in a user's home directory that are generally configuration files that are used when a user runs a program or logs into the system. These are generally files that are modifiable by the user and so they should not be world or group writable. Having these files be world or group writable may enable a malicious user to steal or modify other users data or perhaps gain another user's system privileges.

Auditing **UNIX / Linux** – User dot Files

Check with the Sysadmin about all the identified files to ensure they communicate with the owners of the home directories about the risks of having their files group or world writeable.

Auditing **UNIX / Linux** – User dot Files

Since world or group writeable files may not be accessible by a regular user, these commands need to be run as **su** or **sudo**.

The command to find world writeable dot files is:

```
find / -name “.*” -perm -o+w -exec ls -la {} \;
```

The command to find group writeable dot files is:

```
find / -name “.*” -perm -g+w -exec ls -la {} \;
```

Auditing **UNIX / Linux** – Backups

What should we focus on for backups?

- Procedures in place to backup up the server on a regular basis
- Backups kept offsite

Auditing **UNIX / Linux** – Physical Security

What should we focus on for physical security?

- Access to the facility
- Smoke / fire alarms
- Backups
- Uninterruptible power supply
- Disaster recovery and business resumption plans in place
- And so on...

Auditing Oracle



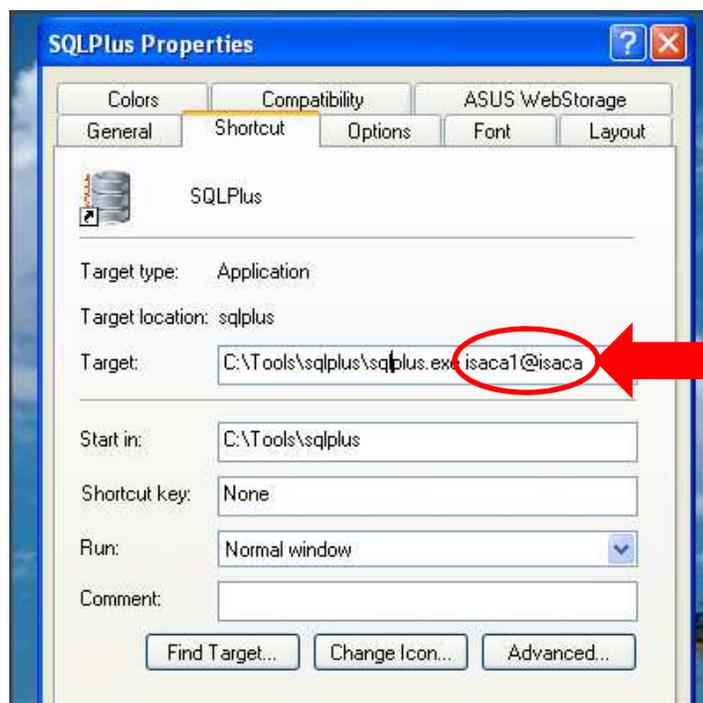
Setting Up To Audit Oracle

What do we need?

- Request access to the database environment – request a user ID on the sample databases with the following privileges:
 - **select any table**
 - **create session**
 - **select any dictionary**
- Request access to the UNIX / Linux server since you will need to check OS level settings. A regular user account will work. Have them add this user account to the **dba** group.

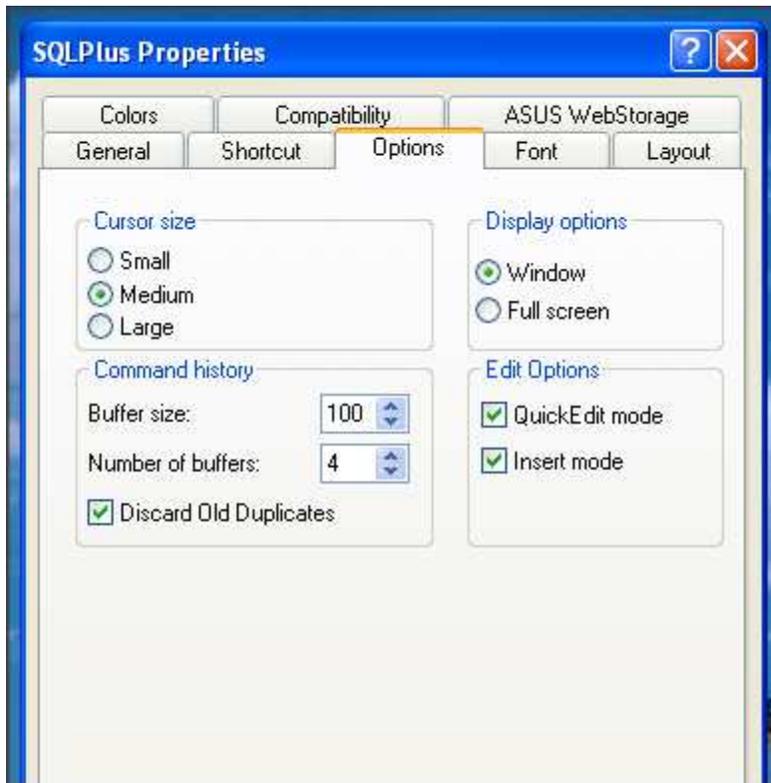
Running Oracle Scripts

Right click the **SQLPLUS** shortcut that was created on the desktop and select **properties**. Modify the settings on the following pages:



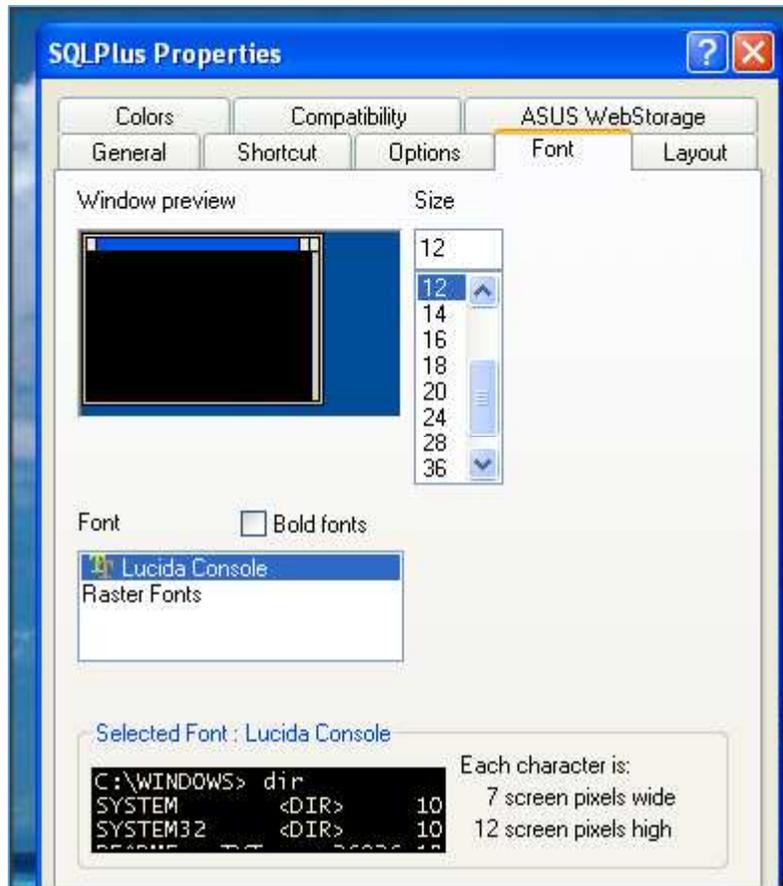
Modify the line to include your user ID and the name of the database. Your user ID should be **isaca***n* where the *n* is a number. The **@isaca** should come after the user ID.

Running Oracle Scripts – cont'd



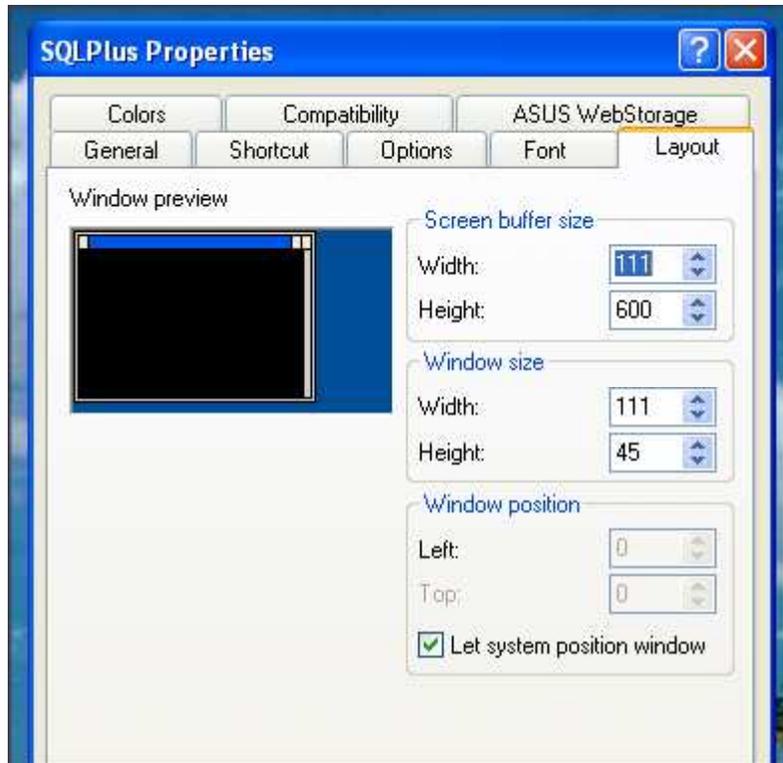
Modify your settings to match this screen.

Running Oracle Scripts – cont'd



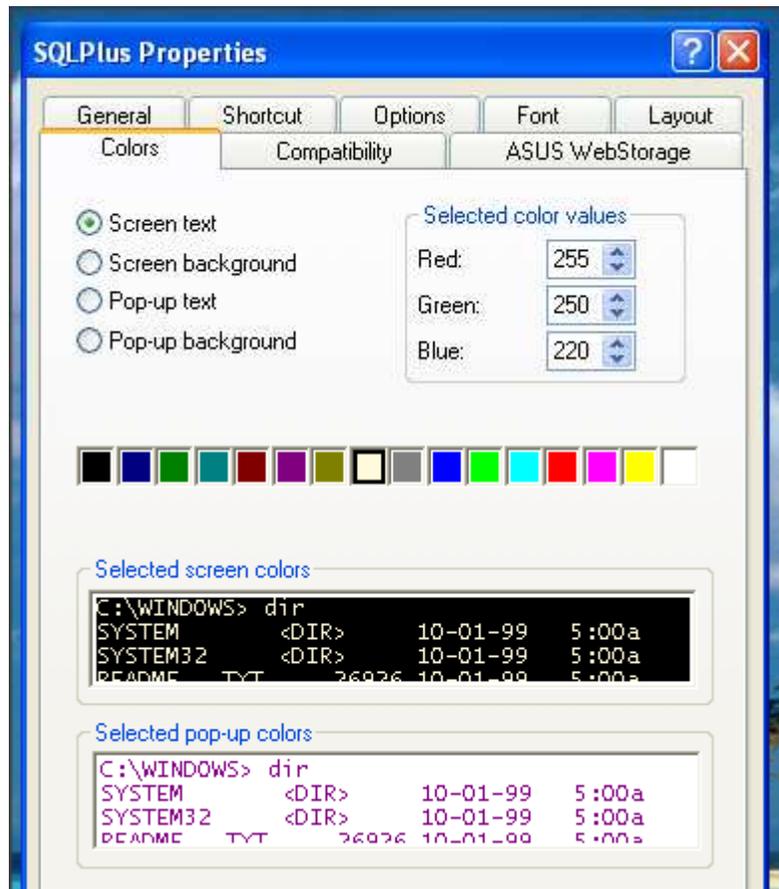
Modify your settings to match this screen.

Running Oracle Scripts – cont'd



Modify your settings to match this screen.

Running Oracle Scripts – cont'd



Modify your settings to match this screen.

Running Oracle Scripts – cont'd

Click **OK** to save all your changes to the shortcut.

Double click the shortcut to run **SQLPLUS**. You'll see a **DOS** box open up and a prompt asking you for your password. Type the password for your account and press enter.

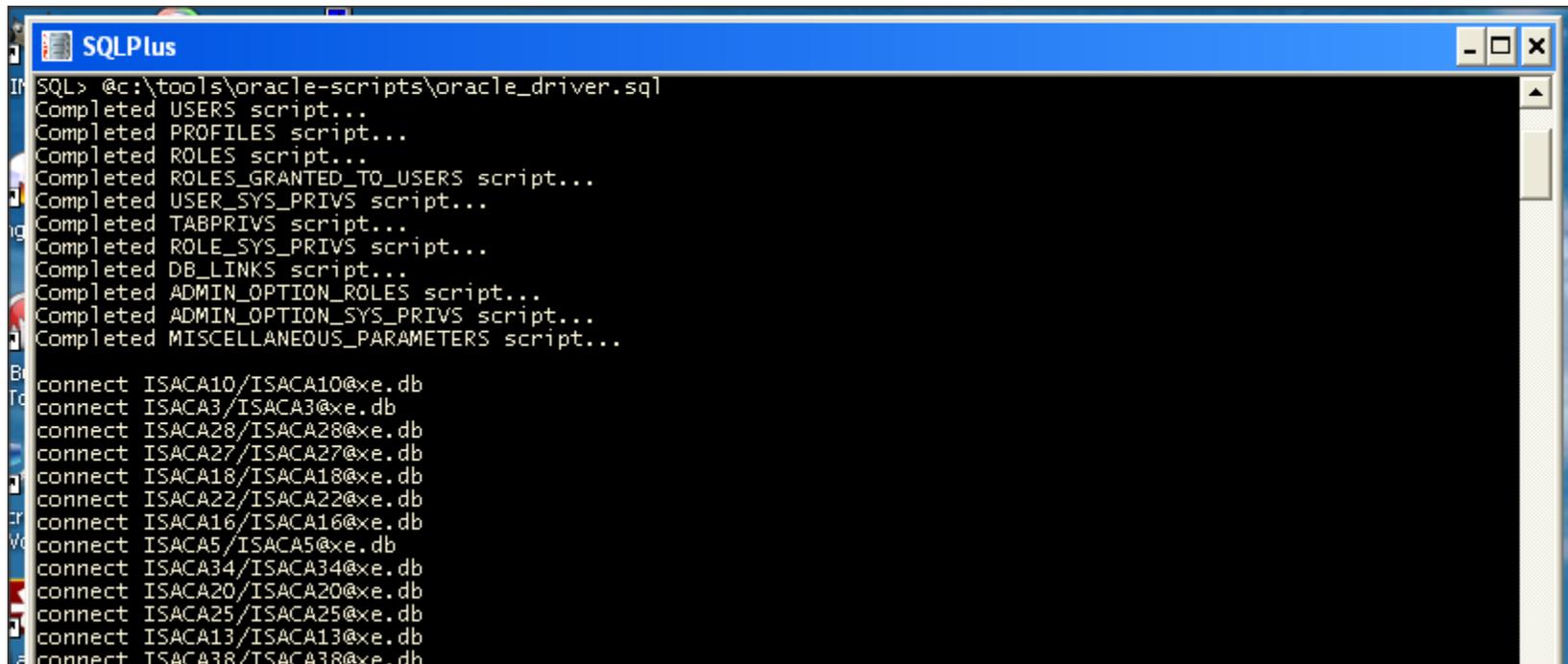
At the **SQL>** prompt type the following command and press enter:

```
@c:\tools\oracle-scripts\oracle_driver.sql
```

Substitute the location of your scripts if they are not located in the **c:\tools\oracle-scripts** directory.

Running Oracle Scripts – cont'd

You should see the following output displayed on your screen as the scripts run. The scripts will take a couple of minutes to complete.

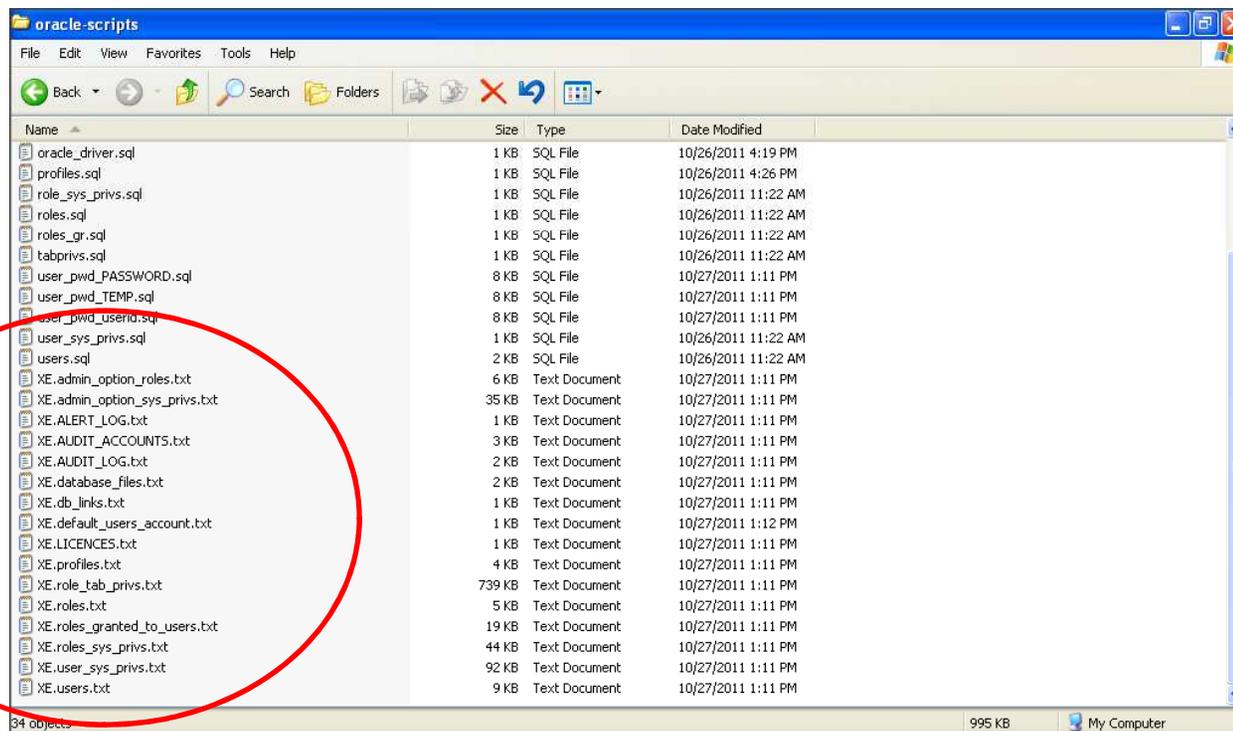


```
SQL> @c:\tools\oracle-scripts\oracle_driver.sql
Completed USERS script...
Completed PROFILES script...
Completed ROLES script...
Completed ROLES_GRANTED_TO_USERS script...
Completed USER_SYS_PRIVS script...
Completed TABPRIVS script...
Completed ROLE_SYS_PRIVS script...
Completed DB_LINKS script...
Completed ADMIN_OPTION_ROLES script...
Completed ADMIN_OPTION_SYS_PRIVS script...
Completed MISCELLANEOUS_PARAMETERS script...

connect ISACA10/ISACA10@xe.db
connect ISACA3/ISACA3@xe.db
connect ISACA28/ISACA28@xe.db
connect ISACA27/ISACA27@xe.db
connect ISACA18/ISACA18@xe.db
connect ISACA22/ISACA22@xe.db
connect ISACA16/ISACA16@xe.db
connect ISACA5/ISACA5@xe.db
connect ISACA34/ISACA34@xe.db
connect ISACA20/ISACA20@xe.db
connect ISACA25/ISACA25@xe.db
connect ISACA13/ISACA13@xe.db
connect ISACA38/ISACA38@xe.db
```

Running Oracle Scripts – cont'd

Use Windows Explorer to browse to the folder where your scripts are located. The output from the Oracle scripts are located in this folder – **c:\tools\oracle-scripts**.



Oracle Risks

Where are the areas of risk in Oracle?

- Database administration
- User administration
- Operating system level controls
- Database change control

The above areas are the most important and our focus will be on ensuring that controls are in place to mitigate risks in these areas.

- Any other risk areas that we are forgetting?

Database Administration

- Database administrators
- Patches
- Oracle account – OS level
- Listener security
- Default accounts
- Database links
- Control files

Database Administration - cont'd

- Alert log
- Auditing
- File locations
- Database backup and recovery

Database Administration – DBA's

How many DBAs are there? Is the number appropriate?

Review the results of the following:

- `role_sys_privs.sql` (`XE.roles_sys_privs.txt`)
- `user_sys_privs.sql` (`XE.user_sys_privs.txt`)
- `roles_gr.sql` (`XE.roles_granted_to_users.txt`)

Database Administration – Patches

Ensure that all the latest Oracle patches have been installed. The latest patch number for the version of Oracle that is running is part of the full version string. i.e. 10.2.0.1.0

Log into **sqlplus** to find out the database version number.

Check the Oracle website for the latest patch version number of the Oracle database that is installed at your location.

Database Administration – Lock Oracle Account

Lock the “**oracle**” OS level user account. This will deter attackers from trying to brute force the account. If the account cannot be locked, then ensure it has a VERY strong password.

```
grep -i oracle /etc/passwd
```

Database Administration – Listener

The listener is a named process that manages the network traffic between the Oracle database and the client.

Why protect it?

An attacker can easily remotely manage the listener and potentially take control of the server if it is not secured. The default installation of the Oracle database prior to Oracle 10g allowed a client to remotely administer a listener using the “**lsnrctl**” program or by issuing commands directly to the listener. Oracle 10.1 and above by default restricts all remote administration of the listener, unless security is explicitly disabled in the configuration file.

Database Administration – Listener Security

All of the following recommended changes to listener security should be made in the listener.ora file. To find the file run the command:

```
find / -name listener.ora
```

To view the contents of the file type the full path and name of the file:

```
more $ORACLE_HOME/network/admin/listener.ora
```

Database Administration – Listener Security

- Enable logging for all listeners – this will capture all listener commands and brute force attacks
- Modify the default install name of the listener. A distinct and unique name should be chosen.
- Apply all listener security patches
- Block SQL*Net traffic through firewalls unless absolutely necessary
- Secure the \$TNS_ADMIN directory – this is the location where the listener.ora file is stored

Database Administration – Listener Security

- Host names can be spoofed and could cause Oracle outages or man-in-the-middle attacks. Modify the host names to be IP addresses – host names are used as default.

Database Administration – Default Accounts

Ensure that Oracle default accounts are locked and expired. Oracle default account passwords are common and available on the Internet.

Database Administration – Database Links

Fixed user database links that have hard coded username and passwords must be avoided. Anyone with permissions or rights to view these usernames/passwords exposes that account to unnecessary risk.

Ask the DBA about all database links that are identified in the following:

- **db_links.sql (XE.db_links.txt)**

Database Administration – Control Files

Ensure that the database control files are secured with appropriate file permissions (at the OS level). The control files should only be accessible by **oracle** OS user and should not be world writeable. These control files should be archived on a regular basis.

Review the database control files that are identified in the following:

database_files.sql (XE.database_files.txt)

Logon to the server and view the files to check the permissions. For example:

```
ls -la /usr/lib/oracle/xe/oradata/XE/control.dbf
```

Database Administration – Alert Log

The Oracle alert log file must be reviewed regularly for errors. A periodic review may show errors, large numbers or exotic errors that can be an indicator of a system under attack. Check with the DBAs to ensure they review the alert log on a regular basis.

The location of the alert log is identified in the following file:

miscellaneous_parameters.sql (XE.ALERT_LOG.txt)

Logon to the server and browse the alert log for suspicious activity and question the DBA.

Database Administration – Auditing

It is ALWAYS a good idea to log the usage of highly privileged accounts such as **SYS**, and those users connecting with **SYSDBA** and **SYSOPER** privileges.

Check the **init.ora** or **spfileXE.ora** to see if the **AUDIT_SYS_OPERATIONS=TRUE**.

Find these files by issuing the command:

```
sudo find / -name *.ora
```

Database Administration – Auditing

Auditing should be enabled for all logons and logoffs and unsuccessful attempts by any SQL statement to access a table. Review of these files should be done in a timely manner.

There are several good 3rd party software packages that will provide insight into Oracle database activity by privileged users and any other type of anomaly that may show up.

Database Administration – File Locations

The redo, data and index files should always be distributed onto separate disks and controllers to ensure that recovery of a database will be easier. An Oracle database also should not be internet facing – it should be separated from the web server (or internet facing application) behind appropriate firewalling or other network protection systems.

Database Administration – File Locations

Review the locations of the redo logs (these are located under the **MEMBER** header) that are identified in the following file:

database_files.sql (XE.database_files.txt)

Check to see if they are located on a different drive (and controller).

Database Administration – Database Backups

Check with the DBAs to ensure that backups are being done and that the recovery process is tested. Failure to test that a recovery is possible could cause inability to recover data, leading to data loss.

User Administration

- Account management
- Password management
- Role administration
- System privileges
- Default accounts
- Default tablespace

User Administration – Account Management

The DBAs should document and enforce account management procedures. These should include the creation of new user accounts, moving a user to a new group or role, and handling of dormant or inactive accounts.

User Administration – Password Management

Password strength has been an issue with Oracle for a long time. Great strides have been made with Oracle since v8 (11g is the latest).

To ensure that complex passwords are used, the **SEC_CASE_SENSITIVE_LOGON=TRUE** should be set in the **init.ora** file.

User Administration – Password Management

Password strength can be managed through profiles – some things that can be restricted are:

- FAILED_LOGIN_ATTEMPTS
- PASSWORD_LIFE_TIME
- PASSWORD_REUSE_TIME
- PASSWORD_REUSE_MAX
- PASSWORD_LOCK_TIME
- PASSWORD_GRACE_TIME
- PASSWORD_VERIFY_FUNCTION

These profiles are then assigned to users.

User Administration – Password Management

Verify which profile is assigned to a user and then check the content of the profile to ensure that it is appropriate.

To check which profile is assigned to a user, review the following file:

- **users.sql (XE.users.txt)**

Review what resource names are assigned to the profile in the following file:

- **profiles.sql (XE.profiles.txt)**

User Administration – Role Administration

Verify that roles are being used to grant privileges to database resources and tables. Identify all roles defined and all the system and table privileges granted to those roles.

Verify that no users are granted predefined Oracle roles – application roles should be created to grant express privileges required by each user. Some predefined Oracle roles are **connect**, **resource**, **dba**, **exp_full_database** and **imp_full_database**.

User Administration – Role Administration

To check which roles are assigned to a user, review the following:

- `roles_gr.sql` (XE.roles_granted_to_users.txt)

Review the privileges that are assigned to the roles in the following:

- `role_sys_privs.sql` (XE.roles_sys_privs.txt)

User Administration – Role Administration

Verify what individual privileges are being assigned to users – these should not be assigned directly and should rather be assigned to a role and the role assigned to a user.

Verify that the **ADMIN OPTION** or **WITH GRANT** has not been given to roles and system privileges. If a user has the **ADMIN OPTION** or **WITH GRANT**, they can then grant that same privilege to another user.

User Administration – Role Administration

To check which roles have been granted to a user with the **ADMIN OPTION**, review the following:

- **admin_option_roles.sql**
(XE.admin_option_roles.txt)
- **admin_option_sys_privs.sql**
(XE.admin_option_sys_privs.txt)

User Administration – System Privileges

Identify users and roles that have been granted system privileges. Ensure that system privileges granted to users and roles are not excessive.

The following system privileges should not be granted to users or application roles:

- alter any “object”
- create any “object”
- insert any “object”
- delete any “object”
- drop any “object”
- update any table

User Administration – System Privileges

“object” could be **table, index, synonym, trigger, procedure, cluster, snapshot**, etc.

In addition, **select any** “object” where “object” could be **table, view, synonym, sequence**, etc.

The key word here is **“ANY”** ...

User Administration – System Privileges

To check which users have been granted privileges with the **ADMIN OPTION**, review the following:

- **user_sys_privs.sql (XE.user_sys_privs.txt)**

User Administration – Default Accounts

Ensure that any default installation passwords have been changed for all system accounts. Some of these accounts may be locked and expired so make sure they remain locked and expired. Use **sqlplus** to attempt to logon to these accounts. Accounts to check are:

SYS – change_on_install

SYSTEM - manager

DEMO - demo

SCOTT - tiger

DBSNMP - dbsnmp

OUTLN - outln

User Administration – Default Tablespace

Ensure that user accounts do not have their **DEFAULT_TABLESPACE** set to **SYSTEM**. Only user account **SYS** (or privileged user account) should have their default tablespace set to **SYSTEM**. This will prevent users from altering system objects.

To check the default tablespace of users, review the following:

- **user.sql (XE.users.txt)**

Operating System Level Controls

- Oracle directories and files
- Configuration files
- Oracle account and dba group

OS Level Controls – Oracle Directories And Files

- All files located in the **\$ORACLE_HOME/bin** directory must be owned by the OS level **Oracle** account.
- All file permissions in the **\$ORACLE_HOME/bin** directory should be set to **0755** or less.
- All file permissions in the **\$ORACLE_HOME** directories (excluding **\$ORACLE_HOME/bin**) should be set to **0750** or less.

OS Level Controls – Configuration Files

- All ***.ora** files should be restricted to the owner of the **Oracle** software and the **dba** group.
- All database datafiles should be restricted to the owner of the **Oracle** software and the **dba** group.

OS Level Controls – Oracle Account And Group

- Ensure that the use of the **oracle** account is restricted to the **DBAs**.
- Ensure that the **dba** group membership is appropriate – **more /etc/group**.

Database Change Control

Review the change control procedures. Determine who is responsible for initiating, testing, and installing changes. Determine if these procedures are adequately documented and followed.

Ensure that new software/upgrades are tested in a test environment.

THE END!! In Summary:

- Did we meet the goals and expectations of the session?
- Installed our tools
- Scanned our network to see what is out there
- Audited UNIX / Linux
- Audited Oracle

Q & A



Appendix

Protocols – OSI model and TCP/IP

Protocols

- Protocol Models
 - ISO's OSI Model
 - DARPA's TCP/IP Model

- Acronym's
 - ISO - International Organization for Standardization
 - OSI - Open Systems Interconnection model
 - DARPA - Defense Advanced Research Projects Agency
 - TCP - Transmission Control Protocol
 - IP - Internet Protocol
 - UDP – User Datagram Protocol

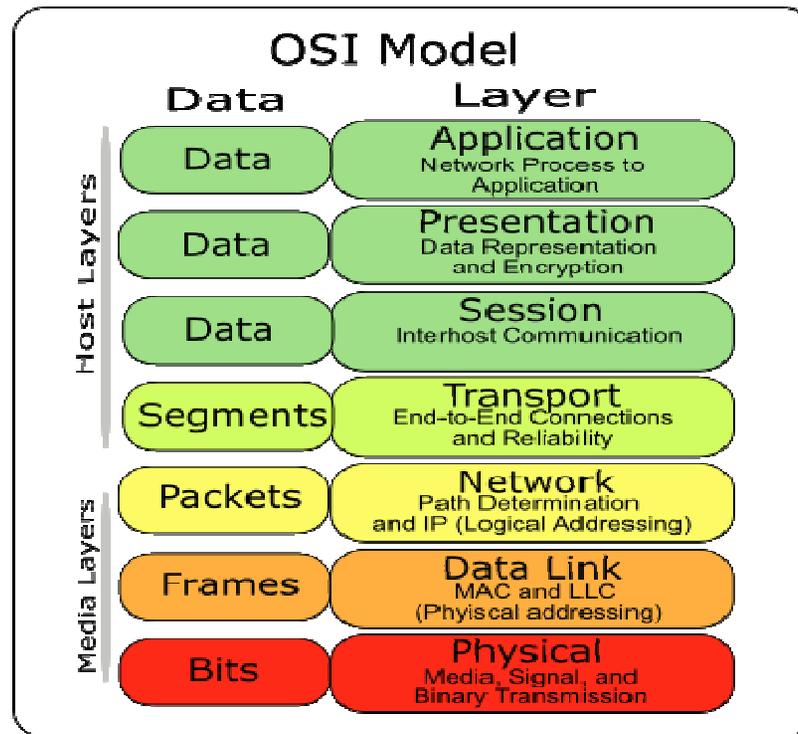
Protocols - continued

- OSI Model

OSI is a computer-communications architecture that uses layering. Each layer performs a related subset of the functions required to communicate with another system. It relies on the next lower layer to perform more primitive functions and to conceal the details of those functions.

Protocols - continued

- 7 Layers Of The ISO/OSI Model



Protocols - continued

- OSI 7 Layers And Function

- Physical - Deals with electrical and mechanical procedures (bits) to establish, maintain, deactivate physical links
- Data Link – Sends blocks (frames) of data across physical link providing flow control and error recovery
- Network – Provides upper layers with independence from data transmission and switching technologies
- Transport – Provides reliable, transparent transfer of data between end points, flow control and error recovery
- Session – Provides controls structure for communication between cooperating applications
- Presentation – Performs generally useful transformations on data to provide a standardized application interface and to provide common communications services; encryption, text, compression, reformatting
- Application – Provides services to users. Defines network applications to perform tasks such as file transfer, e-mail, network management

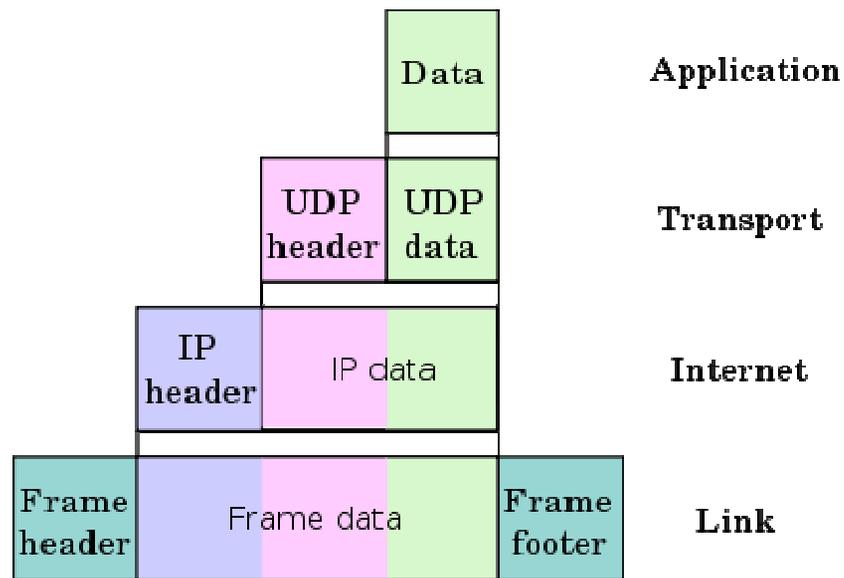
Protocols - continued

- TCP/IP Model

The TCP/IP model, or [Internet Protocol Suite](#), describes a set of general design guidelines and implementations of specific networking protocols to enable computers to communicate over a [network](#). TCP/IP provides end-to-end connectivity specifying how data should be formatted, addressed, transmitted, [routed](#) and received at the destination. Protocols exist for a variety of different types of communication services between computers. There are four abstraction layers.

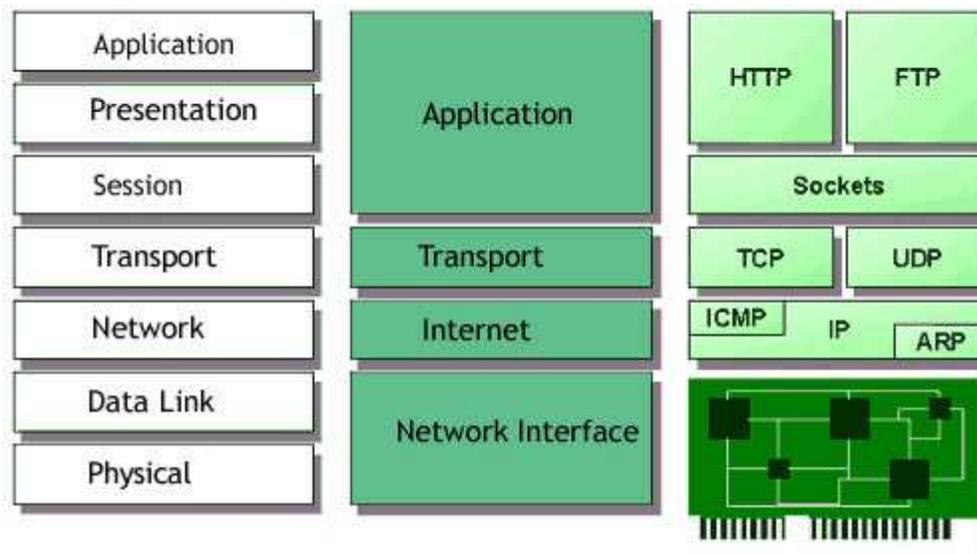
Protocols - continued

- Layers In The TCP/IP Model



Protocols - continued

- OSI Model mapped to the TCP/IP Model mapped to common protocols



OSI and TCP/IP

Protocols - continued

- TCP/IP Background
 - TCP/IP Protocol Suite resulted from research funded by DARPA
 - The protocol suite was designed to foster communication between computers:
 - With diverse hardware architectures
 - To accommodate multiple computer operating systems
 - Using any packet switched network

Protocols - continued

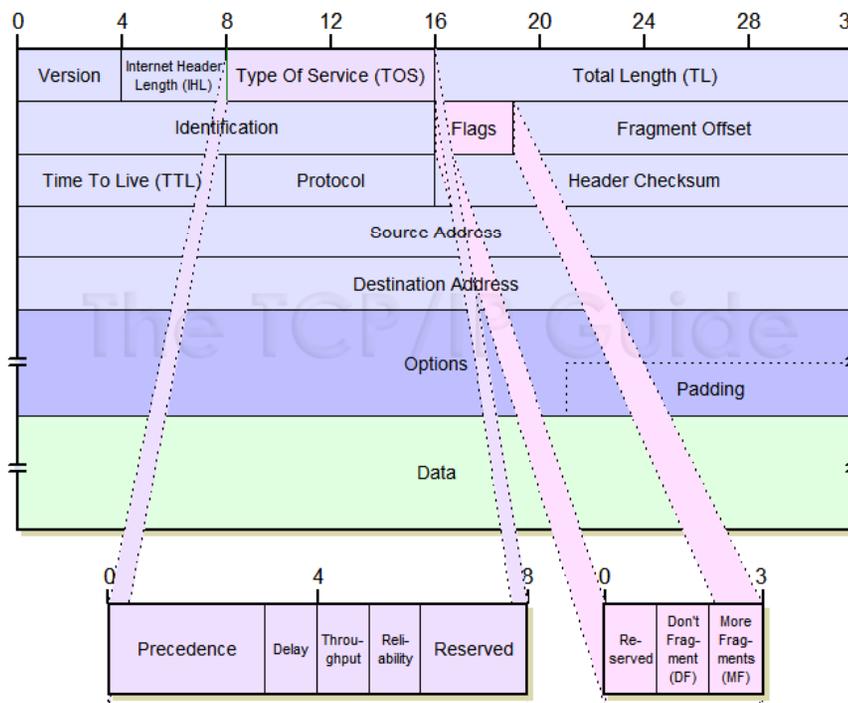
- TCP/IP Background
 - TCP/IP Protocol Suite Provides Three Conceptual Sets of Internet Services:
 - Application Services
 - Reliable Transport Service (TCP)
 - Connectionless Packet Delivery (UDP)

Protocols - continued

- TCP/IP's Basic Transfer Unit is an **IP Datagram**

Fields

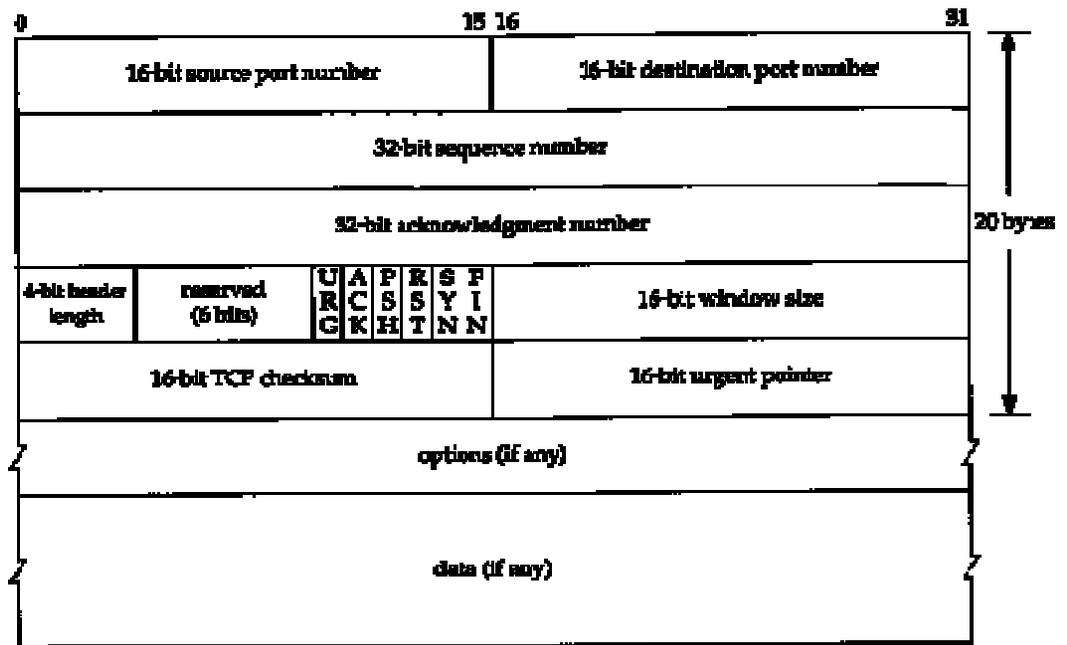
- VER - Version
- HLEN – Header Length
- Service Type
- Length
- ID, Flags, and Flags Offset
- TTL – Time To Live
- Protocol**
- Header Checksum
- Source IP Address**
- Destination IP Address**
- IP Options
- Padding
- Data



Protocols - continued

- TCP Packet Segment Format

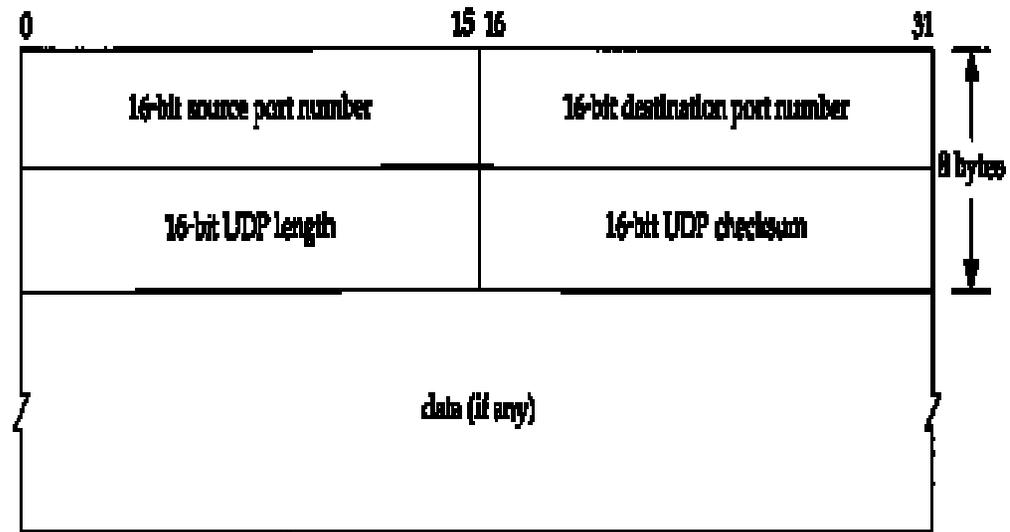
- Source Port
- Destination Port
- Sequence Number
- Acknowledgement Number
- HLEN
- Reserved
- Code Bits
- Window
- Checksum
- Urgent Pointer
- Options (IF ANY)
- Padding
- Data



Protocols - continued

- UDP Packet Segment Format

- Source Port
- Destination Port
- Length
- Checksum
- Data



Protocols - continued

- Communication Components
 - Packet - A physical message unit entity that passes through a network sending and receiving data between two computers:
 - It usually contains only a few hundred bytes of data
 - Carries identification that enables computers on the network to know whether it is destined for them or how to send it on to its correct destination
 - Networks – collection of computers and devices interconnected by communications channels that facilitate communications and allows sharing of resources and information. Typical networks are called packet-switched networks which route each data block called packets independently from all others. such as:
 - MPLS (multiprotocol label switching)
 - Ethernet
 - X.25
 - Frame Relay