New Mexico Tech
Digital Forensics
Fall 2006

Lab VI: Timeline Analysis

Objectives

- Use MAC time information to generate a timeline of file activity
- Interpret timeline for finding evidence.

Procedures

Extract MAC Times for Allocated and Unallocated Files

Step 1

Five image files exist on */dev/hdb1*. These images are taken from a honeypot that was compromised. Each image file is a different partition from the honeypots main disk (*/dev/hda*).

Launch your "*Linux - Forensics*" virtual machine.

Mount */dev/hdb1* to */mnt/hacked*

View the *honeypot.sha1.txt* in */mnt/hacked/lab6/* to see how the disk was partitioned as well as the hash values for each partition.

This honeypot contained Red Hat release 6.2 and IDS logs show that the system was compromised on November 7, 2000. The honeypot's system time was set to Central Standard Time.

One of the most important features of the sleuthkit is the ability to create a timeline of file activity on the system. The timeline is created from the MAC times associated with every file and inode on the system.

> ***Question 1:*** *Knowing what you know about MAC times, will a timeline be able to show each and every time a specific file was accessed or modified? Why or why not?*
>
> *No. A timeline will only be able to show when a file was last last modified, accessed or changed. MAC times are not able to keep a complete history of when the files were accessed or changed. They can only be used to show the last time a file was modified or accessed.*

To create the timeline, we are interested in allocated and unallocated files and unallocated inodes.

> ***Question 2:*** *By looking at the MAC times of allocated files what can be learned about the person who comprimised the system? What sorts of files are you interested in viewing? (Hint: Think about more then just the*

files the hacker creates. What other files may be affected from his/her activity?)

If the hacker is sloppy and leaves any files he/she created or downloaded, you will see these file as allocated on the timeline, but the hacker affects more files then just the ones created by him/her. Any services used last by the hacker will be shown in the timeline such as an ftp client, ssh, tar, and other system commands at the command line. If the hacker was the last person to log into the system then the timeline will reflect when the hacker accessed the system by looking for the login daemon, bashrc, etc.

***Question 3:** Why should you be interested in looking at the MAC times of the unallocated files?*

The timeline should be able to show you what files were deleted and when. This will give you an idea of what file the hacker was interested in hiding to avoid detection. Deleted files may include log files, history files, and source code for malicious programs.

You've learned about the data-unit layer and the meta data layer. A third layer that sits above these two is the file or directory layer. A directory is simply a a special kind of file that cotains data representing the files and subdirectories that it contains. Some meta data will be stored at this level as well, such as the inode associated for each file or subdirectory.

To view the information contained at the file level, the sleuthkit tool *fls* can be used.

The inode of a directory you wish to view must be known to use *fls*. Recall from Lab 4 that inode 2 is associated with the root directory on linux systems.

Run *fls* on the root directory of the compromised system.

```
# fls -f linux-ext2 /mnt/hacked/lab6/honeypot.hda8.dd 2
```

You should see a list of the contents and their inode numbers under the root directory.

Find the inode number for the *etc* directory. Run *fls* on the *etc* directory.

```
# fls -f linux linux-ext2
/mnt/hacked/lab6/honeypot.hda8.dd etc-inode
```

You should now see the contents of the *etc* directory.

Find the inode numbers for the files *group* and *passwd*. Use *icat* to extract these files to your working directory to be used later on.

Fls can be used recursively to traverse down all directories with the *-r* option. The *-z* option is used to specify the time zone. The *-m* option prints the output in

machine format which is needed for a tool you use later on. An absolute path name is needed with the -m option. The machine time format includes more information about the files such as permissions, timestamps etc.

Use *fls* to extract information at the file layer about all allocated and unallocated files on the compromised system. By default *fls* does both. (Note: if no inode number is provided *fls* runs at the top most directory).

```
# fls -f linux-ext2 -z "CST6CDT" -r -m "/boot/"
/mnt/hacked/lab6/honeypot.hda1.dd >
/mnt/hacked/lab6/body.fls

# fls -f linux-ext2 -z "CST6CDT" -r -m "/usr/"
/mnt/hacked/lab6/honeypot.hda5.dd >>
/mnt/hacked/lab6/body.fls

# fls -f linux-ext2 -z "CST6CDT" -r -m "/home/"
/mnt/hacked/lab6/honeypot.hda6.dd >>
/mnt/hacked/lab6/body.fls

# fls -f linux-ext2 -z "CST6CDT" -r -m "/var/"
/mnt/hacked/lab6/honeypot.hda7.dd >>
/mnt/hacked/lab6/body.fls

# fls -f linux-ext2 -z "CST6CDT" -r -m "/"
/mnt/hacked/lab6/honeypot.hda8.dd >>
/mnt/hacked/lab6/body.fls
```

The 'body' file is just a naming convention that comes from an older tool called *grave-robber* (part of the Coroner's Toolkit) that could be used to collect information on allocated files because *fls* used to not have that feature.

Extract MAC Times for Unallocated Inodes

Step 2

Recall from Lab 4 that *ils* was used to list inode information for an entire image. By default *ils* will only list inodes that are unallocated.

The command below is an example of a small bash script at the command line that will allow you to run *ils* on all 5 images easily. (Note: Where you see a single carrot, enter a carriage return).

```
# for i in 1 5 6 7 8
> do
> ils -f linux-ext2 -m /mnt/hacked/lab6/honeypot.hda$i.dd
>> /mnt/hacked/lab6/body.ils
> done
```

The two body files should be concatinated into one.

```
# cat body.fls body.ils > body
```

Generating the Timeline with mactime

Step 3

The powerful sleuthkit tool *mactime*, will organize the data found in the body file based on their MAC times. By providing the *passwd* and *group* file, the *mactime* report will include actual usernames and groupnames instead of user and group ID numbers. A specific date range can be specified to control the size of the report. Remember, it is know that the system was compromised on November 7, 2000.

```
# mactime -z "CST6CDT" -p /mnt/hacked/lab6/passwd -g
/mnt/hacked/lab6/group -b /mnt/hacked/lab6/body
11/07/2000 > mactime.txt
```

The format of the output of *mactime* is as follows: time, filesize, which of the mac times is associated with the time shown, permissions, username, groupname, inode number, and filename. If the filename field is in carrot brackets then it means it came from the *ils* output.

Use less to view the *mactime.txt* file. Scroll down to the time November 8, 2000 08:25:53 where the first activity of the hacker begins. He starts by running the command *uptime*. Try running *uptime* on your own system if you don't know what it is.

Question 4: *Why might the attacker be interested in the information provided by uptime?*

Uptime will show how long the system has been up and running. This can be a good indication of what kind of system it is. If the system has been running for days, weeks or longer, it is probably a good indication that the system is some kind of server. Uptime also show how many user sessions are open, indicating there is other people using the system.

Question 5: *What did the attacker do with the host.deny file? What follows shortly after in the timeline that explains this activity?*

The MAC time field shows that the file was modified. The filesize is also showing 0. The attacker must have deleted the content of this file. The host.deny file is used to block connections with specified IP numbers. The timeline next shows that the attacker accessed the ftp client. It would make sense that the attacker would zero out the host.deny file so that his ftp session will not be blocked.

Find where the user *drosen* modifies the file that is now deleted. Given the information shown recover the file and determine the file type.

A good practice is to include the inode number in the filename of the data you extract. This will help you keep track of the evidence you find.

> ***Question 6:*** *How did you recover the file? How did you determine the type of file?*
>
> *The output from ils shows that there is unallocated inode 8133 that is on hda8. This file, before it was deleted, was modifed at 08:51:37 . icat can be used to recover the file.*
>
> ***# icat -f linux-ext2 /mnt/hacked/lab6/honeypot.hda8.dd 8133 > /mnt/hacked/lab6/i8133***
>
> *The file command can be used to determine the file type.*
>
> ***# file /mnt/hacked/lab6/i8133***
>
> *The file is a tar file.*

You should have been able to recover a tar file.

Untar the file to your working directory.

 # tar xvf /mnt/hacked/lab6/*recovered-file*

Now do a *ls -l* and look carefully.

> ***Question 7:*** *What is unusual about the tar file after it is extracted and what do you think the purpose is?*
>
> *The contents of the tar file were extracted to a directory whose name was a space character. This is a somewhat sneaky way to try and hide a directory.*

The file you recovered is a tar file containing the program eggdrop. Eggdrop is an IRC bot utility. IRC is a common program for hackers to use to communicate with each other and exchange programs.

> ***Question 8:*** *Looking at what comes next in the timeline at 08:51:53, what does the attacker appear to have done? What is the significance of the location of these files? What do you think these files are used for?*
>
> *It looks like the attacker unpacks another tar archive. A bunch of files in the same location are accessed within the just a couple of seconds. The files are unpacked to the directory /usr/man which is usually just used for holding the man pages. The directory is also named .Ci which means it will be a hidden directory. The files appear to be copies of common system utilities. This is most likely a root kit.*
>
> ***Question 9:*** *What does the attacker do at 08:52:09 and why?*

The attacker links /root/.bash_history to /dev/null. This disables the history file from logging the commands he uses at the command line.

This is only a fraction of all there is to discover on the compromised system. Feel free to explore as much as you want.

Using Autopsy to Create a Timeline

Step 4

You are now going to use autopsy to repeat the steps you did above.

Create the directory */mnt/hacked/lab6/autopsy*

Launch Autopsy

```
# autopsy /mnt/hacked/lab6/autopsy
```

Create a new case called lab6 and you as the investigator. The host is vmware-forensics and use the timzone of the compromised system, CST6CDT. Add each of the honeypot images being carefully to select the right file system and mount points.

From the window containing all the images, click the '*File Activity Time Lines*' button. Click the '*Create Data File*' button.

Make sure there is a check mark for all the images. Select all *data types to gather*. Name the output file *body* and click *OK*.

When Autopsy is done running the same command you ran earlier, click the '*Create Timeline*' button. Specify the starting date as Novemeber 7, 2000 and no end date. Save the output file as *mactime.txt* . Select the *honeypot.hda8.dd* as the image containing the *passwd* and *group* files.

View the timeline and verify it matches the timeline you created earlier.

Step 5

It is very important to do these next steps so that the lab is properly set up for the person who uses the computer after you.

Unmount any drives you mounted and shutdown the VMWare system.

In VMWare, revert '*Linux - Forensics*' back to the snapshot by clicking the '*Revert*' button.

From the *c:\vmware-images\Linux - Forensics* directory remove all files beginning with '*Linux – Forensics-Image*'.

Question 10: *What are your comments and suggestions for this lab?*