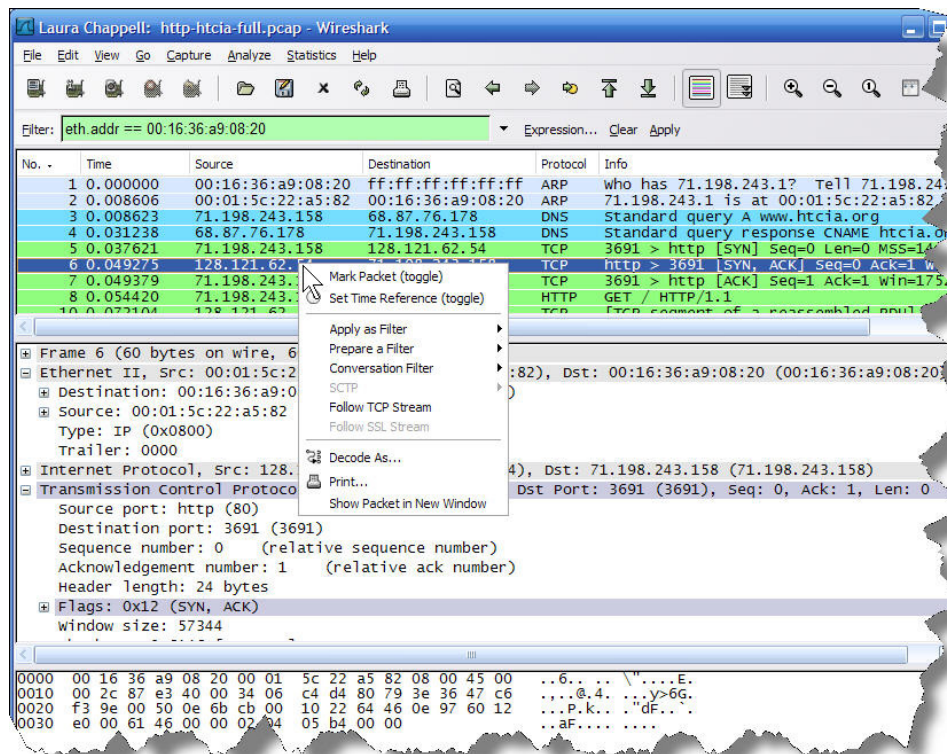


WSU04: Wireshark Network Forensics and Security

Appendix A: Wireshark Lab Exercises





Wireshark Lab Exercise

Covert FTP Connection

Trace File: extra01.pcap

The traffic is already ugly on this new system that just had BitTorrent loaded on it. Look in the trace for the background FTP connection. The user didn't have any idea this connection was being established.

Step 1. Open the trace file listed above.

Step 2. Examine this trace and focus on the FTP connection.

Question 1: What do you think is the purpose of this FTP connection?

Question 2: Was it successful?

Question 3: What is the name of the host running the FTP daemon?

Question 4: This process is running on a new HP desktop system. Does this process appear to be linked to HP?



Wireshark Lab Exercise

Dueling Honeypots

Trace File: extra02.pcap

It's a cat fight! Watch the change of direction in the scan process when one aggressive honeypot gets scanned by another aggressive honeypot.

Step 1. Open the trace file listed above.

Step 2. Examine this trace and differentiate between honeypot 1 and honeypot 2.

Question 1: What is the IP address of the first honeypot?

Question 2: What is the IP address of the second honeypot?

Question 3: At what point in the trace did the second honeypot begin its scan against the first honeypot?

Step 3. Create an **IO Graph** with two filters:

Black line: `ip.src==24.6.137.85 && tcp.flags == 0x02`

Red line: `ip.src==24.6.138.50 && tcp.flags == 0x02`

Green line: no filter



Wireshark Lab Exercise

Worm-Infected System

Trace File: extra03.pcap

A client system (172.16.1.10) is in trouble. After it boots up the CPU utilization climbs to 100% and the system locks up within 3 minutes. You can see many problems in the trace - incoming DCERPC communications and the client establishing TFTP and IRC communications to remote systems.

Step 1. Open the trace file listed above.

Step 2. Examine this trace and Follow TCP Stream on the IRC communication.

Question 1: What IRC network did the client connect to?

Question 2: What is the client's nickname?

Question 3: What files is the IRC client told to download?

Question 4: Which of these files did the actually client download?

Question 5: What was the download speed?

Question 6: Did the client download any other files? If so, what application was used for the download?

This client is infected with a variant of the sdbot worm, also referred to as HLLW.Donk (Symantec).

McAfee Information:

“These SDBot names vary considerably, but regularly try to look similar to other legitimate Windows executable names, so that a user viewing the Task Manager might assume that the names listed are valid.

The exact method of propagation will vary between variants. However, the following characteristics are typical:

The worm propagates via accessible or poorly-secured network shares, and some variants are intended to take advantage of high profile exploits:

- *DCOM RPC vulnerability (MS03-026) -*
<http://www.microsoft.com/technet/security/bulletin/MS03-026.aspx>
- *WEBDAV vulnerability (MS03-007) -*
<http://www.microsoft.com/technet/security/bulletin/MS03-007.aspx>
- *LSASS vulnerability (MS04-011) -*
<http://www.microsoft.com/technet/security/bulletin/MS04-011.aspx>
- *ASN.1 vulnerability (MS04-007) -*
<http://www.microsoft.com/technet/security/bulletin/MS04-007.aspx>
- *Workstation Service vulnerability (MS03-049) -*
<http://www.microsoft.com/technet/security/bulletin/MS03-049.aspx>
- *PNP vulnerability (MS05-039) -*
<http://www.microsoft.com/technet/security/bulletin/MS05-039.aspx>
- *Imail IMAPD LOGIN username vulnerability -*
http://www.osvdb.org/displayvuln.php?osvdb_id=16804
- *Cisco IOS HTTP Authorization Vulnerability -*
<http://www.cisco.com/warp/public/707/IOS-httplevel-pub.html>

There are now over 4000 variants of this threat, many of which were proactively detected, and this number continues to grow at a rapid rate.”

Question 7: How did this client get infected? How would you protect this client?



Wireshark Lab Exercise

Hidden Data

Trace File: extra04.pcap

ARP packets are minimum-sized packets and must be padded to meet the minimum 64-byte length for this Ethernet network. When we look at the padding on these packets, we see something strange.

Step 1. Open the trace file listed above.

Step 2. Examine this trace and look through the padding of the ARP packets.

Question 1: Which systems padded the ARP packets with data?

Question 2: Which systems padded the ARP packets properly?

Question 3: What do you think the *public* and *manager* strings may be used for?



Wireshark Lab Exercise

Checking for Poisoner

Trace File: extra05.pcap

This trace file shows someone running ettercap's 'Check for Poisoner' function. This process locates other systems that may be running ettercap on the network.

Step 1. Open the trace file listed above.

Step 2. The host at IP address 12.234.13.202 is checking for other poisoners. Examine this trace and answer the following questions:

Question 1: What application is being used to look for other for other poisoners?

Question 2: How many systems responded?

Question 3: What application is being used to look for other for other poisoners?

Step 3. The IP ID field of these ping packets contains the signature 0xe77e (eleet-speak for 'ette' which is short for ettercap). Systems that answer back with the same IP ID value are most likely running ettercap.

Build a filter for all systems sending a response with the IP ID value of 0xe77e. Your filter should not display the queries from 12.234.13.202.

Question 4: What filter did you create/apply?

Question 5: Which other systems appear to be running ettercap?

Question 6: Is the value 0xe77e used anywhere else in the query packets?

Question 7: How many responding systems use 0xe77e in that same location (as found in Question 6)?

Note: Ettercap can be downloaded from ettercap.sourceforge.net.



Wireshark Lab Exercise

Decrypt SSL Traffic with an RSA Key

Trace File: rsasnakeoil2.cap

Note: This trace file was downloaded from the wiki.wireshark.org/samplecaptures page. The file name has *not* been changed during course development. This trace file has been edited to show traffic to/from 127.0.0.1.

Wireshark can decryption SSL sessions that use RSA key exchange as long as the host private key is provided using the SSL Preferences setting.

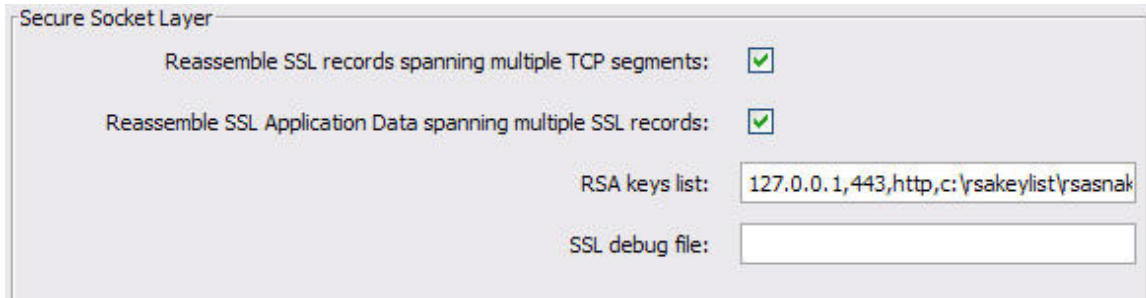
Excerpt from the Wireshark ssl-c dissector file:

```
* Notes:
*
* - Does not support dissection
*   of frames that would require state maintained between frames
*   (e.g., single ssl records spread across multiple tcp frames)
*
* - Identifies, but does not fully dissect the following messages:
*
*   - SSLv3/TLS (These need more state from previous handshake msgs)
*     - Server Key Exchange
*     - Client Key Exchange
*     - Certificate Verify
*
*   - SSLv2 (These don't appear in the clear)
*     - Error
*     - Client Finished
*     - Server Verify
*     - Server Finished
*     - Request Certificate
*     - Client Certificate
*
* - Decryption is supported only for session that use RSA key exchange,
*   if the host private key is provided via preference.
*
* - Decryption need to be performed 'sequentially', so it's done
*   at packet reception time. This may cause a significant packet capture
*   slow down. This also cause do dissect some ssl info that in previous
*   dissector version were dissected only when a proto_tree context was
*   available
```

Step 1. Open the trace file listed above.

Step 2. Examine this trace and note that the data is encrypted.

- Step 3.** Create an `c:\rsakeylist` directory on your system.
- Step 4.** Copy the `rsasnakeyoil2.key` file into the `c:\rsakeylist` directory.
- Step 5.** Select **Edit > Preferences > Protocols > SSL**.
- Step 6.** In the RSA Keys list field, enter the following value and click OK.
`127.0.0.1,443,http,c:\rsakeylist\rsasnakeyoil2.key`



Note: The syntax for the RSA keys list field is <ip>,<port>,<protocol>,<path and key filename>.

- Step 7.** Review the trace file again and answer the following questions.

Question 1: What files did the client request of the HTTP server (other than the index page as noted by the GET / HTTP/1.1 request in packet 11)?

1. _____
2. _____
3. _____
4. _____
5. _____
6. _____
7. _____
8. _____
9. _____