PUSL3131 Lab

Intrusion Analysis: Network Traffic Forensic Analysis Answers

5. Server attack – Honeynet challenge

You might want to consider the following stages of Intrusion Analysis. Use the spaces below to write any commands, notes, BPF filters that you use in this process:

1. If your target is 192.168.1.102, which is the IP Address of the main attacker? Which other IP addresses are involved in the attack? How do you know that? Hint: Run Snort in named VM to generate alerts.

The main attacker is 207.35.251.172. Snort in Securityonion triggered approximately 37 alerts for the use of FTP SITE EXEC commands and FTP SITE overflow attempts (sid: 200001971, sid: 2100361, sid: 2000002417, sid: 2000001529, sid: 2101529). The payload of these packets suggests that buffer overflow attacks are indeed taking place against target 192.168.1.102. The source IP is attacking host 207.35.251.172.

To get an appreciation of the order of events, we sort alerts based on AlertID or Date/Time.

Apart from the main attacking host, the compromised server sent an email to IP address 64.4.49.71, containing the /etc/passwd password file. (alert 'ET ATTACK_RESPONSE Possible /etc/passwd via SMTP (linux style', sid: 2003149). The snort rule, seems to be looking for the content "root:x:0:0root:/:root:/" in any established sessions and requests to servers on destination port 25/tcp. Looking at the payload, it indeed looks like root@localhost sent an email from 192.168.1.102 to 64.4.49.71 (destination port 25, source port 1031), to hatcheryhatched@hotmail.com containing the password file. The payload contains the content that triggered the rule.

2. Which vulnerability did the intruder exploit? Which IDS alerts captured the vulnerability exploitation attacks?

As suggested previously, the attacker used FTP SITE EXEC commands, and the Buffer Overflow vulnerability to get access to the system. By locating and following TCP stream for session 207.35.251.172:2243-192.168.1.102:21, we can see that the attacker tried several exploits before one became successful. Snort in securityonion generated alerts with sid 200001971, 2100361, 2000002417, 2000001529, and 2101529 (as described in previous question, q1).

3. At which point do you know that the attack became successful? Hint:
   Look for 'Attack Response' alerts

After the FTP related alerts, snort reports 'GPL ATTACK_RESPONSE id check returned root' (sid:2100498). This could indicate a response to an attack, when the compromised server confirms escalation of privileges (in this case escalation from guest account to root). This is not the only snort alert that was triggered when the compromised server responded to indicate root privileges. The rule itself is very generic and prone to false positives, as it only looks for the content

"uid=0(root)" in IP packets. However, when seen within the context of this attack, it does indeed

suggest this is a true alert. All related alerts are responses from the compromised server to the main attacking host, from source port 21/tcp (same socket that was used for exploitation), right after several exploitation attempts. When following TCP stream on wireshark for the offending session (see point 2), we can indeed see they eventually managed to escalate their privileges, got root access and ran command 'w' to verify this. The FTP server returned answer 'uid=0(root) gid=0(root)', which suggests the current user is root (rather than a guest account).

1. Are there any other internal IP addresses involved or affected by this incident? How will you find out?
   Hint: Use a BPF filter, or use Wireshark Display Filters

Securityonion does not report any other attacking activity involving other internal hosts. We will need to use a bpd filter (or wireshark display filter) to identify any other internal IP addresses that communicated with the attacking hosts. Which filter would that be? We could also search for the presence of account names and passwords that were used in the attack, in other internal systems.

2. What happened after the successful vulnerability exploit?
   Hint: Use Wireshark 'Follow TCP Stream' of relevant sessions

The dataset contains a lot of post-exploit attacking activity. The answer identifies only the main post-exploit activity.
When following the stream in wireshark for socket 207.35.251.172:2243-192.168.1.102:21, we can see that the attacker issues commands, such as: w (who else is logged in at the time), navigating the file structure to discover user programs and data (/usr, /usr/local, /usr/local/bin, /usr/local/etc), devices and drives connected to the system (/dev), and available desktop environments (/etc/x11/applnk/Internet). The attacker then deletes the password for user account nobody. He then attempts to copy the modification access creation times of the password file /etc/passwd into a temporary file, with the intention of preserving this information and hiding the fact that /etc/passwd will end up being modified. Indeed, the attacker creates another user dns, deletes their password, and then tries to hide their tracks. After further navigating within the file system, the attacker prints the password file /etc/passwd.

When looking at Conversations in Wireshark, we can identify further suspicious sessions that need further investigation. Which other hosts did the compromised server communicate with after the attack?

3. The intruder downloaded rootkits. What were they called? Use Wireshark's "Edit -> Find Packet" menu option to locate the sessions where rootkits are exchanged.
   Hint 1: the rootkits are exchanged over Active FTP, where commands are sent over port 21/tcp ftp-data is exchanged over port 20/tcp
   Hint 2: Follow TCP Stream to recover the files, and save then in Raw format.

Looking at suspicious sessions through Conversations in Wireshark (sorting on volume of Bytes to identify the busiest sessions), we identify tcp sockets 192.168.1.102:1027 - 193.231.236.42:20 and 192.168.1.102:1026 – 193.231.236.42:20. These are ftp data sessions, which contain ftp files. To identify all ftp traffic between these 2 hosts, we can apply a display filter. The ftp commands session over 21/tcp (socket 192.168.1.102:1025 – 193.231.236.42:21) reveals the name of rootkits that are being requested and then transported over the ftp-data port 20/tcp. The rootkits are:

- Packet **#1178** file **Zer0. tar. gz**
- Packet **#1413** file **copy. tar. gz**
- Packet **#1768** file **oopy. tar. gz**

4. BONUS Question: The attacker also establishes a ssh session. Which IP address, server port number and which SSH client does the attacker use? Hint 1: Use 'Endpoints' or 'Conversations' from Wireshark to find out.
   Hint 2: Identify port numbers that are "close" to SSH server port 22/tcp

Using Endpoints or Conversations, we can see that there is no traffic over port 22/tcp (ssh). However, there is traffic over port 24/tcp. By displaying this traffic only, we can see that the compromised server accepts a session from 217.156.93.166 on this port. Following the TCP stream for this session shows that the attacker is using client SSH-1.5-PuTTY

Packet #23085 and tcp port 24
Packet #23089 and SSH-1.5PuTTY