LOYOLA UNIVERSITY CHICAGO
COMP 447-001 - INTRUSION DETECTION - FALL 2014

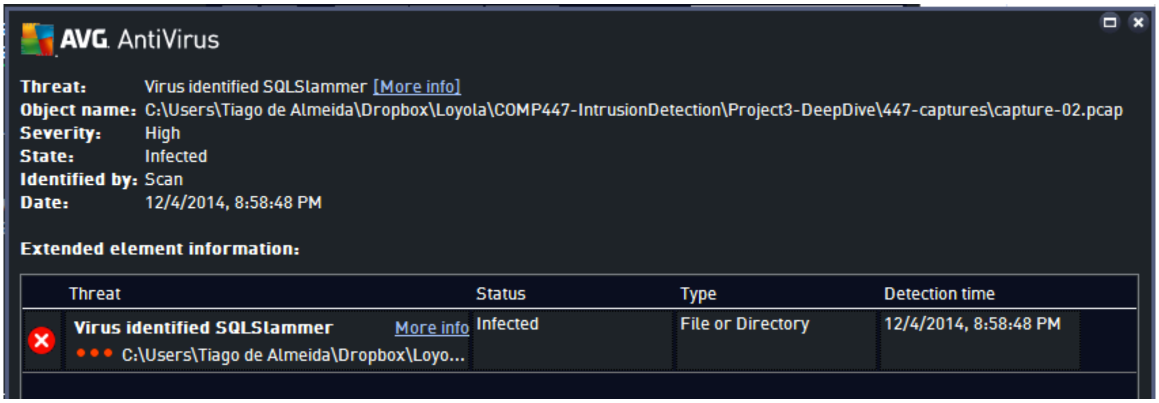
Tiago de Almeida - tdealmeida@luc.edu
1394611

Graduate Project 3
Deep Dive

Capture 2:

MS-SQL Slammer Worm

This packet capture contains a propagation attempt for the MS-SQL Slammer Worm. This worm exploits a buffer overflow vulnerability. It sends a 376 byte long UDP packet to port 1434. Infected systems will send identical packets, making the worm propagate quickly. The worm sends traffic to random IP addresses causing a DoS on the target network. [1]
It was detected by Snort, which identified the worm, as well as by AVG Antivirus, as shown in the screenshot below.



AVG AntiVirus

Threat: Virus identified SQLSlammer [More info]
Object name: C:\Users\Tiago de Almeida\Dropbox\Loyola\COMP447-IntrusionDetection\Project3-DeepDive\447-captures\capture-02.pcap
Severity: High
State: Infected
Identified by: Scan
Date: 12/4/2014, 8:58:48 PM

Extended element information:

Threat	Status	Type	Detection time
Virus identified SQLSlammer More info C:\Users\Tiago de Almeida\Dropbox\Loyo...	Infected	File or Directory	12/4/2014, 8:58:48 PM

Capture 3:

SYN Flood Attack

From [2]: “A SYN flood is a form of denial-of-service attack in which an attacker sends a succession of SYN requests to a target's system in an attempt to consume enough server resources to make the system unresponsive to legitimate traffic.” The idea here is to make the target reply with a SYN/ACK for each SYN sent, but never send ACK, therefore never completing the three-way handshake. As a consequence, the target will keep hundreds of hanging connections waiting for ACK and consuming resources.
Analyzing the packet capture on Wireshark we can see hundreds of SYN packets being sent from the same IP source address to the same IP destination address in an attempt of executing the SYN Flood attack.

Capture 4:
Heartbleed

This packet capture shows an exploitation of the TLS protocol version 1.1 known as Heartbleed. Analyzing the capture on Wireshark we can see the target returns more data that it should.
Snort also identified the exploitation as "TLSv1.1 Malicious Heartbleed RequestV2". In fact, I was able to find the blog post where this packet capture was originally posted: [3].

Capture 5:
SYN Scan

A port scan is an attempt to find out which ports are open (have a service listening on it) on a system. A SYN scanning executes by sending SYN packets to every port of interest in the target. Some ports may respond with a SYN/ACK packet, indicating they are willing to start a TCP connection. This scan is also known as "half-open scanning", because it never actually opens a full TCP connection. [6]
Analyzing this packet capture on Wireshark we can see the departure of hundreds of SYN packets from the same IP source to the same IP destination. Also, we can see that packets are sent from just two different ports on the source (36050 and 36051), but are sent to several different interesting ports on the destination. Most of the destination ports do not reply to the connection attempt. However, three interesting ports send a SYN/ACK back. They are 22 (SSH), 53 (DNS) and 80 (HTTP). The attacker can therefore see that there are services running on these ports.

Capture 6:
IP Fragmentation attack, using overlapped fragments

Analyzing the packet capture on Wireshark we can see hundreds of fragmented IP packets being sent from the same source IP address to the same destination IP address. The problem here is revealed when we check the offset of each fragment. Several packets are sent with the same offset, for the same ID. The goal here is to confuse the target receiver, which will not be able to re-assemble the fragments correctly. The target will consume resources and hang waiting for missing fragments and dealing with repeated fragments. This is considered a DoS attack. [7]
Snort was also able to identify the fragmentation overlap issue on this capture.

Capture 7:
Conficker B

Snort detects a network trojan called Conficker B. According to [4] "This worm makes changes to you PC and can disable important system services and security products, like antimalware or antivirus software. It spreads by infecting PCs on your network, removable drives (like USB flash drives), and weak passwords."
Wireshark shows us an interesting behavior: The Trojan sends ARP requests to all possible IPs in the local network with the hope of finding other targets to infect.

Capture 8:
UDP Scan

Analyzing the packet capture on Wireshark we can see hundreds of UDP packets being sent from the same IP source address and port towards the same IP destination, however to several different ports. This can be seen as a UDP scan. From [6]: "if a UDP packet is sent to a port that is not open, the system will respond with an ICMP port unreachable message. Most UDP port scanners use this scanning method, and use the absence of a response to infer that a port is open. However, if a port is blocked by a firewall, this method will falsely report that the port is open. If the port unreachable message is blocked, all ports will appear open. This method is also affected by ICMP rate limiting."
If the rate by which the packets are sent is too high, the target may not be able to handle the sending of ICMP port unreachable messages to other hosts. This way, this can be seen as a UDP Flood attack. Most operating systems mitigate this part of the attack by limiting the rate at which ICMP responses are sent.

Tentative analyses:

Capture 1:
The website www.virustotal.com provides a few scans and analyses. Besides identifying HTTP requests and DNS requests inside the capture, their Snort scan returned a few messages:

- BROWSER-IE Microsoft Internet Explorer Script Engine Stack Exhaustion Denial of Service attempt (Attempted Denial of Service)
- SENSITIVE-DATA Email Addresses (Sensitive Data was Transmitted Across the Network)
- (http_inspect) HTTP RESPONSE GZIP DECOMPRESSION FAILED (Unknown Traffic)

The first message is particularly interesting and seems to identify a Stack Exhaustion DoS attack.

Capture 9:
SMB attack

I have not been able to reach a reasonable conclusion for this packet capture. However, here is what I could notice.

Snort returns the following alert:

```
[**] [1:537:15] NETBIOS SMB IPC$ share access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
02/19-15:47:41.162553 192.168.0.93:2283 -> 192.168.0.84:139
TCP TTL:64 TOS:0x0 ID:50462 IpLen:20 DgmLen:124 DF
***AP*** Seq: 0x293D6066  Ack: 0x64B5A2B4  Win: 0xD80  TcpLen: 32
TCP Options (3) => NOP NOP TS: 236176 2672
```

Analyzing the packet capture on Wireshark we can see a TCP connection is established, and later a SMB session is negotiated. Then, in what looks like a DoS attack, the attacker sends consecutive 1 byte write requests to the server. The requests also seems to overwrite the same area after a few dozens requests. It looks like some kind of flood attack in order to take down the SMB service or even the whole system.

Capture 10:

Snort returned alert messages related to:
- ICMP PING undefined code
- SCAN nmap XMAS: Attempted Information Leak

The snort analyses from the website www.virustotal.com returned the following messages:
- POLICY-OTHER TCP packet with urgent flag attempt (Generic Protocol Command Decode)
- SERVER-OTHER Winnuke attack (Attempted Denial of Service)
- PROTOCOL-ICMP PING (Misc activity)
- PROTOCOL-ICMP Echo Reply (Misc activity)
- PROTOCOL-ICMP PING undefined code (Misc activity)
- INDICATOR-SHELLCODE x86 inc ebx NOOP (Executable Code was Detected)

On Wireshark we can see several packets with messages like the following:
- The acknowledgment number field is nonzero while the ACK flag is not set.
- The urgent pointer field is nonzero while the URG flag is not set.

The above reveal a non-consistent state of the analyzed packets. Some of them also have the urgent flag set, what is related to the Winnuke attack underlined in a Snort message mentioned above.

According to [5], Winnuke refers to a remote DoS attack that affected early versions of Microsoft Windows. The exploit sent a malicious TCP packet contained an Urgent Pointer (URG). Such field is rarely used in the TCP header and affected operating systems didn't handle it correctly.

References:

[1] Malware FAQ: MS-SQL Slammer
https://www.sans.org/security-resources/malwarefaq/ms-sql-exploit.php

[2] Wikipedia - SYN Flood
https://en.wikipedia.org/wiki/SYN_flood

[3] Didier Stevens Blog - Heartbleed: Packet Capture
http://blog.didierstevens.com/2014/04/09/heartbleed-packet-capture/

[4] Microsoft Malware Protection Center - Worm: Win32/Conficker.B
http://www.microsoft.com/security/portal/threat/encyclopedia/entry.aspx?Name=Worm%3aWin32%2fConficker.B

[5] Wikipedia - WinNuke
https://en.wikipedia.org/wiki/WinNuke

[6] Wikipedia - Port Scanner
https://en.wikipedia.org/wiki/Port_scanner

[7] Wikipedia - IP Fragmentation Attack
https://en.wikipedia.org/wiki/IP_fragmentation_attack