EVALUATION OF THE CAPABILITIES OF WIRESHARK AS NETWORK INTRUSION SYSTEM

Alia Yahia and Eric Atwell
Sudan University of Science and Technology, Khartoum, Sudan
University of Leeds, Leeds, UK

Abstract: Network security professions learning network intrusion should be able to see attack signatures and learn the different techniques to detect them. Wireshark is an open source cross-platform protocol analyzer with a user-friendly interface. Wireshark has a protocol dissector that supports over 2000 protocols. In the paper we assume that Network Intrusion detection systems should have three components: a user interface, packet sniffer and a detection engine. The detection engine can either detect anomaly or signature based attacks but it must be automated: it should detect intrusions without human intervention. The paper shows that Wireshark can be considered a packet sniffer, protocol analyzer and trouble shooting tool but not a network intrusion detection system as it lacks the fundamental component which is an automated detection engine.

Keywords: Wireshark, network, intrusion system.

1. INTRODUCTION

The need to bridge the gap between theoretical and practical network security is vital as “black hats” are getting more aggressive by the day and the “white hats” must keep up with new attacks. A network intrusion detection system is software and network infrastructure that monitor network activities for any violation of the confidentiality, integrity and availability (CIA) of the system [1]. In a broad sense we can say intrusion detection systems are the systems that monitor any violation of the security policies. They can be classified into Anomaly detection model and Misuse detection (signature based) model.

1.1 Anomaly detection model

This model is based on building a profile of normal behavior; from this a baseline is created. Any deviation from this baseline will be flagged as an intrusion[1]. Anomaly detection’s main advantage is the ability to detect zero based attacks (unknown attacks). Its major disadvantage is the high percentage of false positives.

1.2 Misuse detection

This model looks for patterns in the packets (packet headers or payload), these patterns or signatures are compared to those found in a database. It can also search for known activities that violate the stated security policy. Misuse detection’s main advantage is that it has a low false alarm rate, but its major drawback is it cannot detect novel attacks.

Network intrusion detection systems consist of three major components:
1) Packet sniffer: capture packets from the network interface.
2) User interface: where the user can see alerts notifying them if an intrusion has occurred.
3) Detection engine: where the packets are flagged as benign or malicious according to a certain algorithm.

2. WIRESHARK

Wireshark is a network protocol analyzer that has tools and features which makes it easy for a user to analyze network traffic. Wireshark is used to view packet headers and information about the different TCP/IP layers, as shown in Figure 1.
2.1 Wireshark features and tools as a network intrusion detection system.

2.1.1 Display filter

There are two types of filters in Wireshark: capture filter and display filter. The main difference between a capture filter and a display filter is that a capture filter cannot be altered during capture, only before. Display filters are used to display packets in Wireshark according to a certain criteria. Figure 2 shows an example.

![Wireshark display filter: UDP scan filtered by icmp.type==3 and icmp.code==3](image)

2.1.2 I/O graph

Wireshark has the IO graph feature, which summarizes the packet flow; this is illustrated in Figure 3.

![Wireshark I/O Graph](image)

2.1.3 Protocol hierarchy statistics
In the Protocol hierarchy, if you see data where Wireshark does not have a protocol dissector for this protocol, this is suspicious. High percentage of insecure protocols like Internet Relay Chat or Tor is also suspicious.

![Figure 4: Wireshark Protocol hierarchy](image)

2.1.4 Network conversation window

A network conversation is the traffic between two specific endpoints; Figure 5 shows an example.

![Figure 5: Wireshark Network conversion window](image)

2.2 Coloring rules

Coloring rules that represent an intrusion can be applied. Coloring rules require a name, a string (based on the display filter format), a foreground
color and a background color. Figure 6 shows an example coloring rule in Wireshark.

![Example coloring rule in Wireshark](image)

**Figure 6:** Wireshark Coloring rule for HTTP errors

2.3 Expert information

The expert information is a log of the anomalies found by Wireshark in a capture file, as shown in Figure 7.

![Expert information log](image)

**Figure 7:** Expert info

There are some common groups of expert information. The following are currently implemented:

- **Checksum:** a checksum was invalid
- **Sequence:** protocol sequence suspicious, e.g. sequence wasn’t continuous or a retransmission was detected or …
- **Response Code:** problem with application response code, e.g. HTTP 404 page not found
- **Request Code:** an application request (e.g. File Handle == x), usually Chat level
• Undecoded: dissector incomplete or data can’t be decoded for other reasons
• Reassemble: problems while reassembling, e.g. not all fragments were available or an exception happened while reassembling
• Protocol: violation of protocol specs (e.g. invalid field values or illegal lengths), dissection of this packet is probably continued
• Malformed: malformed packet or dissector has a bug, dissection of this packet aborted

3. RELATED RESEARCH

Price [2] highlighted a number of points that make up a good Intrusion detection system. The system must be able to run without human supervision, it should be fault tolerant by having a knowledge-base to fall back to in case of an error, resist to subversion, minimal overhead as the number of packets increased the analysis capabilities of the intrusion detection system shouldn’t decrease, it must observe deviation from normal behavior, it can be tailored to detect different types of attacks and must cope with changing system behaviour. In our review, these points were not found in Wireshark. Gupta and Mamtora [3] outlined the tools and techniques found in Wireshark that can be used to detect TCP based attacks and Denial of service attacks. Gupta highlighted that if the expert information displays a serious error like malformed packet this is an indication of an attack but they did not highlight what type of attack. The user interface is implemented using the conversion tool with an additional field which will indicate if the packet is intrusive or not. The detection process has to be enhanced by using data-mining techniques, for example using Weka or Rapid Miner tool. Banerjee et al. [4] tested the presence of unauthorized access to a server. Display filters and I/O graphs are used to detect intrusions. Four nodes with fixed IP address were used, and the attacker machine has a fixed IP address so the prerequisite to use of the display filters is that the attacking machine IP address should be known. The technique used only detected attacks in the server machine, which is dangerous because a user can gain access to a node from within the network and then escalate his privileges. Kumar [5] used display filters to detect the different type of attacks such as ARP scan, s Null scan, Xmas scan, ARP poisoning and IP protocol. Wireshark conversion was used to detect stealth attack. Cheok [7] combined coloring rules and display filters as a way to highlight the packets that contain intrusions. Cheok concentrated reconnaissance based attacks and ftp based attacks. The coloring of the packet according to a rule just highlights the packet according to certain criteria but these criteria do not make it decisively an intrusion. Qadeer et al. [8] does not explain how Wireshark can be enhanced to be an intrusion detection system. It highlights the mechanism to detect the presence of a packet sniffer in the network using round trip time and SNMP.

4. EXPERIMENT

Intrusion detection starts with where to place the sensor that will capture the packets. We captured the packet from the client side located in 3 sites over a period of 1 year. We captured 4 Gigabyte of data. The dataset contained wired and wireless traffic. We started by listing the protocol found in the dataset. The data field appeared which is an indication that Wireshark does not have the protocol dissector for this layer. It appeared in the data link layer and the application layer. In the application layer it could be not malicious because new applications are developed each day each with a different protocol. The NetBIOS over IPX protocol, an obsolete protocol, appeared as shown in Figure 8, which should have been replaced with NetBIOS over TCP/IP.
The third step was to look up the malicious URL in the dataset by using an online tool to detect malicious web sites. Figure 9 shows the results of the address resolving tool.

Malicious websites were found, but we had to go through the list manually because Wireshark does not have a tool to flag if the website is malicious or not. There is a list of malicious ports found in anti-virus websites [6] but analysis has to be done manually or by using display filters. We implemented the display filters [5] to detect the reconnaissance based attacks and the attack types stated in Cheok [7], shown in Figure 10.

<table>
<thead>
<tr>
<th>Attack</th>
<th>Display filter</th>
<th>Found/Not Found</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© JGRCS 2018, All Rights Reserved
<table>
<thead>
<tr>
<th>Attack Type</th>
<th>Filter Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ping Sweep</td>
<td>icmp.type==8 or icmp.type==0/tcp.dstport == 7/udp.dstport == 7</td>
<td>Found/Not Found</td>
</tr>
<tr>
<td>ARP Sweep</td>
<td>arp</td>
<td>Found</td>
</tr>
<tr>
<td>Stealth</td>
<td>tcp</td>
<td>Found</td>
</tr>
<tr>
<td>Xmas Scan</td>
<td>tcp.flags==0X029</td>
<td>Not Found</td>
</tr>
<tr>
<td>Land attack</td>
<td>(ip.src == ip.dst) &amp;&amp; (tcp.srcport == tcp.dstport</td>
<td></td>
</tr>
<tr>
<td>Nmap Scan</td>
<td>(ip.flags == 0x00) &amp;&amp; (icmp.type == 8 &amp;&amp; icmp.code == 0) &amp;&amp; (icmp.seq == 0) &amp;&amp; (not data)</td>
<td>Not Found</td>
</tr>
<tr>
<td></td>
<td>(ip.flags == 0x00) &amp;&amp; (tcp.flags == 0x0002) &amp;&amp; (tcp.dstport == 443)</td>
<td>Not Found</td>
</tr>
<tr>
<td></td>
<td>(ip.flags == 0x00) &amp;&amp; (tcp.flags == 0x0010) &amp;&amp; (tcp.dstport == 80)</td>
<td>Not Found</td>
</tr>
<tr>
<td></td>
<td>(ip.flags == 0x00) &amp;&amp; (icmp.type == 13 &amp;&amp; icmp.code == 0) &amp;&amp; (icmp[8:4] == 00:00:00:00) &amp;&amp; (not data)</td>
<td>Not Found</td>
</tr>
<tr>
<td>TCP SYN Stealth</td>
<td>(ip.flags == 0x00) &amp;&amp; (tcp.flags == 0x0002) &amp;&amp; (tcp.option_kind == 2) &amp;&amp; (tcp.option_kind == 3</td>
<td></td>
</tr>
<tr>
<td>TCP Connect scan</td>
<td>(ip.flags == 0x02) &amp;&amp; (tcp.flags == 0x0002) &amp;&amp; (tcp.option_kind == 2) &amp;&amp; (tcp.option_kind == 3 &amp;&amp; tcp.option_kind == 4 &amp;&amp; tcp.option_kind == 8)</td>
<td>Not found</td>
</tr>
<tr>
<td>To detect Nikto scanner</td>
<td>http.user_agent contains &quot;Nikto&quot; and http.request.uri contains 2e:2e:2f:</td>
<td>Not found</td>
</tr>
</tbody>
</table>

Figure 10. Attack types found using Wireshark display filter

The expert information tool crashed several times due to the number of error messages. We had to analyze each original packet capture separately. We found malformed packets, checksum errors, cipher text length exceeded the value stated in the SSL protocol, a lot of duplicate Acknowledgement (which is normal if the packet get lost in transit or the connection speed is slow), Window size is 0, this an indication that the tcp window size determine is not large enough.
enough and will be filled by the second packet. It is clear all these different messages do not definitively predict if it is an attack or not. It all depends on further analysis by the Wireshark user.

5. CONCLUSION

Although Wireshark has capabilities useful for network intrusion detection, it lacks one of the major components of an automated intrusion detection system, which is the detection engine. The detection process is not automated, so to detect intrusions the user should have expert knowledge about how protocols work and how they are implemented. The user has to understand packet headers used and their values; and how hackers can manipulate these values to attack a network. Although you can use display filter to detect network intrusion signatures this can be very time consuming and it not viable with the large amount of attack signatures. Wireshark’s protocol dissector can help you understand network patterns. It can give you an indication that something is wrong with the network but it in most cases it cannot give you a decisive answer to whether there is an attack or not. Wireshark will not be able to generate an alarm or take a security action against the unauthorized access [3].

REFERENCES