A University Case Study on using the Metasploit to perform post exploitation in vulnerable networks

Sean Thorpe, Kedrian James, Rohan Malcolm, Julian Jarrett, Tyrone Grandison, Andre Stevens, Alex Thompson, Nickiesha Bonnick, Michael Williams, Omar Gordon
Questionable Assumptions

• Can we automate to detect a specific exploit on a software honey-pot by monitoring network traffic in our campus network?

• What is the best method to monitor the campus network traffic to and from the honey-pot to extract and match patterns of signature from an exploit?

• How successful are these methods?
Summary of the Assumptions

Exploit → Honeypot → Exploit Signature
Research Overview

"A honeypot is [...] a resource which is intended to be attacked and compromised to gain more information about the attacker and the used tools." (Baumann & Plattner, 2002)
Research Overview

An exploit is used to abuse a security vulnerability, leading to an attacker gaining unintended privileges. (Anley et al., 2011)
Research Overview

An exploit usually consists of two parts:
- First trigger the vulnerable application to execute custom code
- The "payload", containing the code to be executed
Research Overview

Metasploit:
- 1000+ different exploits
- Several hundred different payloads
- Metasploit encodes the payload, makes it hard to detect by signature
- Easy to use: choose an exploit, choose a payload to include, fire away!
Research Overview

• The Metasploit Framework is a penetration testing toolkit, exploit development platform, and research tool.

• The framework includes a lot of pre-verified exploits and auxiliary modules for handling penetration test.

• Different payloads, encoders, and handlers are also a part of the Metasploit penetration testing environment. We run Backtrack to support our test domain
Research Overview

I have an exploit for that vulnerable software!

I'm running vulnerable software!

Exploit
Payload  Encoder

...001101010010011010101100101...
I have an exploit for that vulnerable software!

I’m running vulnerable software!
Why is this important?

- The honeypot software itself may be a source of outdated vulnerabilities.

- Analysis of what happened requires manual analysis.

- Having signatures for the most-used penetration testing.

- An automated tool allows for valuable insight in attackers' activities.

- What we want is to automatically detect modern exploits and show which exploits were detected.
Exploits used within Metasploit

- FTP DNS server vulnerabilities
- Microsoft Based vulnerabilities
  Eg. Mso8
- We assume a large number of possible exploits
Post Exploitation Activities

- Copying Files using windows command shell
- Establishing Metrepeter sessions
  - Escalating priviliges
  - Dumping LM Hashes
- Modifying server Log files
Test Setup

Diagram:
- **Attacker System**
  - Virtual Machine
- **Honeypot**
  - Virtual Machine
  - Physical Machine
- **Firewall**
- **Internet**
Metasploit process

1. Executing Exploits against Honeypot
2. Capturing Exploit Traffic
3. Detecting Exploit Traffic
4. Extracting Signatures from Exploit Traffic
5. Matching Exploit Traffic against Signatures
Experimental Tracking

- We run an academic license version of honeybot as our test honeypot.

- Honeybot DB becomes available as a .csv file database with all the vulnerable exploits

- The .csv file logs save all packet traffic

- Then we pattern match to find suspicious traffic
Detecting Suspicious Traffic

- We run likelihood test ratio analysis as our naïve pattern match technique.

- We check each IP address based on timestamp for access logon frequency. Logon access of the passive attacker will be accepted by honeybot as “OK” as the decoy.

- We set a daily threshold (t) on the logon frequency to determine the p-value/critical value of high ratio traffic.

- Where the p-value > t flag traffic as a suspect.
Cumulative Traffic Frequency Ratios Over Six Weeks

Top Frequencies Over the period

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.20.44.75</td>
<td>3000</td>
</tr>
<tr>
<td>10.20.244.114</td>
<td>1500</td>
</tr>
<tr>
<td>10.20.28.188 factor(ip)</td>
<td>1000</td>
</tr>
<tr>
<td>10.20.108.114</td>
<td>800</td>
</tr>
<tr>
<td>10.20.32.152</td>
<td>700</td>
</tr>
</tbody>
</table>
Next Steps

• Extract signatures from the suspicious packet traffic

• Collect Multiple suspicious flows for the same exploit but different payloads

• Find the longest string shared by all suspicious flows using the Longest Common Substring (LCS) algorithm

• The resulting string will be used as signature compared with the LRT method we need to account for high false positives and negation of the null hypothesis

• This method depends on static parts in the exploit, regardless of the payload
Thank You

- Contact
  - sthorpe@utech.edu.jm
  - rohan.malcolm@utech.edu.jm
  - kjames@utech.edu.jm
Some Example Signatures

- Flow 1: eeddccacbefabcfdefbafcbafedfeaf
- Flow 2: aabcbeafeeddccafbdeaabcdefbcea
- Flow 3: feabcdefbfeacceafeabceaeccbeafabcaeddd
- The string "eeddcc" is the longest common substring in the
  - First 2 rows, but it does not occur in the 3rd row.