CS7038 - Malware Analysis - Wk02.1

Attack Introduction

Coleman Kane
kaneca@mail.uc.edu

January 17, 2017

UNIVERSITY OF Cincinnati

Content

This lecture covers the following concepts:

- Setting up lab environments in VirtualBox
- Utilizing networking and shared folders features
- Using kali to build a very rudimentary attack
- Simulating delivery/targeting
- Exploitation
- Example of analysis of a malware sample

Why Learn Attack?

Understanding what an adversary wants to achieve, through intrusion, is important to inform your intuition and guide your conclusions about what you're analyzing.

Frequently, an attacker must take steps to self-analyze their own malware, in order to decide what modifications may need to be employed to side-step target-side mitigations.

Part of doing malware analysis is going to include attempting to recreate attacker actions. Becoming familiar with attacks and attack tools will be vital in validating analysis conclusions, as well as determining if an attack can be prevented and/or detected based upon analysis findings.

Kali Linux

You saw a brief experiment with Kali linux in the previous lecture. We will dive in further to this tool.
Distributor: Offensive Security - `https://www.kali.org`
OVA: Downloads->Download Kali Linux->KALI VIRTUAL IMAGES->VirtualBox

Pre-built environment predominantly used for penetration testing and vulnerability assessments

Tools included:

- Metasploit Framework - `https://www.metasploit.com/`
- Armitage - `https://www.fastandeasyhacking.com/`
- Powerfuzzer - `https://www.powerfuzzer.com/`
- OpenVAS - `https://www.openvas.org/`
- Burp Suite - `https://portswigger.net/burp`

Basic Remote-Shell

We will begin with a basic remote-shell attack. The objective is that we have identified a target and we would like to get some sort of interactive access to their system, so that we may gather more information about the target and possibly steal valuable information as well.

One common approach is a simple "reverse shell". Basically, get a payload onto the target's computer such that the computer will initiate a connection back to a server you control, and will provide you command-line access to the system.

Process:

- Set up your remote control *listen server*
- Configure a payload to connect back to your *listen server*
- Embed that payload in a *carrier* or *trojan* (in our case, a PDF)
- Deliver the *trojan* to your target somehow
- Social-engineer the target into opening the PDF with Acrobat reader
- Wait for connection

Key Concerns for Test Environment

In the real world, the research for a successful attack can take weeks or even months. It is unrealistic, for us, to attempt to successfully attack an up-to-date system within the scope of this class. Rather, we will analyze attacks that are already known and mitigated against. To achieve this, we "cheat" and configure VM lab environments that are comprised of known-vulnerable components.

In the real world, a number of mitigations are typically in place:

- Antivirus software
- OS-level environment randomization
- Newest versions of application software
- Latest operating-system updates, closing known bugs
- Many companies employ further network and endpoint analysis mitigations as well

With the right tooling and adequate time, attackers demonstrate that all fixes and mitigations are temporary.

Basic Attack Template

Borrowed content and example from two sources:

Building the payload and carrier. Simple reverse-shell using a PDF exploit against Acrobat 9.x and Windows XP or 7.

- `http://null-byte.wonderhowto.com/how-to/hack-like-pro-embed-backdoor-connection-innocent-looking-pdf-0140942/`

Configuring the listener for the reverse shell:

- `https://www.offensive-security.com/metasploit-unleashed/binary-payloads/`

An example might be to create the PDF in a manner that appears attractive to the recipient, and then send an unsolicited email to the recipient encouraging them to download and open the PDF using Acrobat reader.

Analyzing the Document

Kali comes with a neat and useful **malware analysis tool**:

- PDFparser: `https://blog.didierstevens.com/programs/pdf-tools/`

The tool can read the PDF content into native data structures, and grants you features to navigate the document objects and streams:

- Inventory of objects & streams
- Extraction tools
- Decoders for common PDF encoding methods
- Filtering
- Searching
- Integration with other powerful tools (yara, for example)