Homework 1
Writeup
CS 373
James Tyler Ball

Week 1:

As I begin the week, the first thing I am attempting in this course is the VM setup. I had attempted this earlier in the week, but encountered an 'unauthorized' page. As I try a second time days later it seems to work. I am not sure if something was fixed or if the change was on my end. So now I have access to the Virtual Windows 7 desktop. The next step is to alter a file and call it "evil.exe". This seems like a hint that I've reached a good time to step away from the VM and look at the Malware slides.

Prior to watching lectures:

Before I even watch the lectures, I'd like to establish where I'm coming from in terms of my knowledge of Malware. My understanding is that Malware comes in many forms such as browser addons, keystroke trackers, and viruses. Malware can be used to send your personal data to undesired third parties for purposes ranging from gathering massive data on users to stealing financial information.

In the past I've downloaded simple scripts for fun like wifi jammers and kali linux. I don't feel I'm well-versed with these tools, though.

Basics of Malware:

Our professor D Kevin McGrath works for McAfee, and the early videos focus on theory. The first thing that left an impression on me was a visualization of how anti-malware engineers operate. Fighting the spread of malware involves careful analytics with specialized tools and research in combination with quick responses to new threats and strict, safe practices.

I learned that the first malware was written by two Pakistanian brothers to protect the software they wrote from piracy and how the Morris' Worm spread. The lecture then went on to discuss reasons why malware persists today.

Primarily the motivation is financial gain, but malware is also written for research purposes, weaponization, and intelligence gathering. It also changed my perceptions to learn that not all malware authors are brilliant coders, many (likely most) offenders simply purchase malware kits.

My perceptions of malware were further shifted when we learned that more and more we see malware targeting mobile and apple software, platforms that have been considered relatively "safe" by many non-tech oriented people I know. We also were shown how disproportionate numbers of Americans were targeted. This could be due to the number of devices the average American uses, but can also be explained by our nation's lack of a firewall.

The lectures then proceeded to better describe the role of someone performing anti-malware research. Not only must they use safe and effective practices when taking countermeasures, but also be able to effectively communicate their findings to people who are unfamiliar with the more technical aspects of their work.

Something I hadn't considered is how it is in the best interests of often competing anti-malware organizations to work together and share their knowledge. Not only is there a shortage of cyber security experts, but the impression I got is that there are far more people spreading malicious code than there are working to stop such practices, and it is almost crucial to work together.

We then learned about the difficulties and approaches when researching malware. Viruses can be polymorphic, parasitic, and/or worm-type, trojans often are idle until they spot a 'target', and often malware tries to detect if it is on a VM, very little of malware code is original. This allows countermeasures to be written to fool malware and better determine what a piece of malicious code is trying to do.

Of course the greatest vulnerability (as reiterated in part 2) of a machine is the user. Sometimes it only takes one individual clicking a bad link or inserting a strange USB stick to spread malware onto a network.

Next, the lectures got a bit more technical. We were shown this script:

which is part of a python script used to display running processes. we also learned basic definitions like White/Black/Grey (type of file), goat (a sacrificial machine), and how hashing is used to safely identify malware.

The professor then stressed how false positives in anti-malware is a huge no-go area. This is part of why taking appropriate countermeasures to new malware can take so long: the engineers have to be sure of the potential threat, and the time from proof of concept to market can take 2-3 years.

We discussed trends in the industry: bootkits and other types of malware fade in and out of popularity. previously vulnerable targets like adobe and java are stepping up security. Old platforms like XP have exploits that will not be fixed.

Something that surprised me was how the professor stressed not to be quick to blame China for malicious attacks. Coming from a manufacturing background, I am aware of how common IP theft was in China, but there are plenty of ways to make it look like an attack is coming from China such as TOR or hacking Chinese servers.

We learned a few of the industry's common practices like naming conventions:

- **type**:**platform**/**family**.**variant**!**information**
- example Trojan:Win32/Reveton.T!Ink

We also talked about how to handle and transport malware ('inactive state' while zipped and use password "infected"). Development Environments need to have firewalls and isolation from sensitive networks.

We broke down the idea of Advanced Persistent Threats and characteristics. In doing so we discussed who the offenders might be, what their motives are, who they might target, and what the objective of such an attack might be. When discussing these goals, it was also stressed why an APT is so difficult to resist: the attackers emphasize stealth and non-detection in their methods. An ideal attack enters, completes the goal, and leaves a system undetected.

We touched again on the common human issues that might lead to system vulnerabilities. Humans leave sensitive data visible to attackers (metadata) and can haphazardly introduce

risks to a system by unintentionally. I am reminded of an article that discussed trying to achieve a balance between ease of use for customers while also trying to be as secure as possible (many users find 2-step authorization to be a hassle).

The professor showed us the APT-Kill-Chain, and the back-and-forth methods that might be used by attackers and defenders at each step. We also were shown some industry methods like sniffing or reversing clients to find passwords. Because malware file names so closely resemble actual system files, Windows has signed software (which malware authors are trying to imitate). We can determine if a piece of software is after credit card data by offering a bait target.

The video took us through a hypothetical situation in which we used context clues to identify that an attacker using USB extraction might likely be an insider. This led to an example of a case in which a hacker was caught because he forgot to enable his TOR proxy, emphasizing again how important it is for anti-malware engineers to be vigilant and carefully observe the data they acquire.

Finally we went over some tools and practices for analyzing malware code in an inactive state. IDA Pro is fantastic for reverse engineering. Hex codes are often swapped, ROT13 is a basic method to slow down detection. FileInsight can analyze naming conventions. UPD can decompress files that have been packed.

Special Note on Lab:

Something else that jumped out at me was all the elements of Computer Science I've missed in my coursework so far. I've found I need to better familiarize myself with windows source files and the windows cmd operations. I also struggled with the VM. I had frequent freezing issues, and I learned that I couldn't just blindly follow directions but had to run some trial and error tests to get the results I was after.