Homework 2: Computers are everywhere and they aren't very secure

José Antonio Álvarez Ocete

Main concepts presented

In the first lecture, we started by discussing some ideas about how **internet and making everything innovative could lead to worse outcomes for consumers in some situations**. We mentioned:

- Privacy violations due to aggressive and unnecessary data collection.
- Over-engineering simple things could lead to reliability losses.
- Potential insecurities due to connecting every element to the internet (IoT).

We explained some of the reasons for **how all of this is happening**. Mainly because of cheap and well-made hardware that almost everyone has access to and easy to made software. This leads to easy-to-make apps but increases complexity in large systems. We discussed the **security trade-off** that developers have to study upon developing an app, and how embracing security afterwards can help on this matter. Finally, we talked about **malware**: different types and how it affects where the malware is running, and the access it has.

During the second lecture, we talked a little bit about the law and how we should **ask for permission not forgiveness**. We also had some **discussions about ethics**, mainly about pre-auth at ride-sharing companies and about how banning is or is not ethical.

In the third lecture we talked about the different **phases of internet-scale attacks**:
1. Scan: Find vulnerabilities in machines (IoT devices, routers, computers…)
2. Infect: Create software to exploit those vulnerabilities and spread it through the internet.
3. Attack: launch "Command and control" signals through the channels to activate the attack.

We compared **Telnet vs SSH** and showed how Telnet doesn't encrypt its communications, being way less secure. However, this type of attack works exactly the same against SSH since it just tries combinations of users and passwords.

We also talked about the different types of DDoS attacks:
- Simple bandwidth attack: trying to overwhelm the server or an intermediate component by using a huge amount of bandwidth at the same time.
- Network-level SYN flood: Using how the TCP protocol works to overwhelm a usually low-usage buffer. This uses low bandwidth but can easily collapse the server.
- Attack the application itself: Asking for specific things to the application and trying to overwhelm the server by the management it has to compute behind the scenes.

Finally, we talked about several **monitoring techniques**:

- Network telescope listens on unused IP addresses: Using that botnets usually aim for random IPs and listen for messages that correspond to IPs that don't exist.
- Honeypot: Creating a fake machine using a VM to study logs and network activity to see how the malware develops in that machine.
- Trying to connect to botnets and understanding how they work.

Future Work

Looking at the first video, there are a few perspectives to talk about. Obviously, security needs to be improved within cars on these matters. A lot can be learnt here from desktop architectures and security measures: adding extra layers, don't give permissions for everything to every component… Also, when a component is compromised it should act quite different than what it happens in regular computer: non-indispensable components could simply stop working, while there are literally vital components, such as throttles and brakes that should have extra security and should never simply stop working.

On the other hand, assuming a car what be compromised and even self-driven, we could study how to identify these cars to prevent, or at least track, this kind of robberies that might happen in the future. Assuming the same algorithms can be used to steal a car and to self-drive a car with someone inside, two options come to my mind. Either to identify using computer vision techniques if there is someone in the car or a maybe more complex solution that keeps track of which cars are actually being self-driven at the moment and just checks the plate.

Tackling the Mirai video, there's plenty of work to be done. The first thing mentioned in this line of thought during the video is that they couldn't identify most of the devices that were infected. Although this provides lines of work for the research side, I think it brings up a more important question. Why are these devices not identified? Should they, at all? Should IoT manufacturers provide an easy way to identify their products? I guess the answer is no as long as you don't provide enough security for these devices. I mean, it is already pretty easy to access them right now, it would be even easier for attackers to do it if they knew more about the devices. However, if extra layers of security -including better default passwords- were used, then identifying devices could make it easier for research and security teams to study what happened afterwards and improve security in the long term.

But most companies wouldn't apply these measures unless we force them, right? Since it requires a lot of money and it doesn't provide a direct benefit for them. So, should we force to, by law? I think so. At least simple and basic measures that they are clearly not taking. An almost hard coded user and password is not security at all. And this is not only because the IoT devices could end up being part of a botnet, but also security for the final consumer that holds that device. The consumer should NOT be required to know how to protect the device and of course, companies shouldn't assume they do. If you are selling a complex device, maybe you should at least state so and explain that without the necessary technical background and effort, the device might not be secure.

Another line of future work was introduced by a question asked during the video: How old were the devices that were being studied? Although the research time didn't have enough data to answer this question (due to the obscurity of the devices discussed above), it leads to an interesting idea: a lot of IoT devices get created and almost instantly stop receiving updates. This is a huge issue, an exploit for these devices could be made public and the companies don't provide a solution for it anymore.