CSE 409, Fall 2011, Rob Johnson, http://www.cs.stonybrook.edu/~rob/teaching/cse409-fa11/
Alin Tomescu, December 9th, 2011

Incentives in security

"If you're designing a security system where one guy does all the work and another guy reaps all the benefits then that system might not fly so well." – Rob Johnson

DDoS and botnets

Your computer can be taken over by a **worm** written by a criminal. The attacker can end up controlling a lot of machines and building up what is called a **botnet**.

The attacker might flood *Amazon.com* with traffic using a **distributed denial of service (DDoS)** attack.

Using **social networking** he can amplify his DDoS attack. For instance, he can find a large image stored on *Amazon.com* and hot-link to it on social networking profiles. Every time one of these social networking profiles is displayed, a request will be sent to *Amazon.com* for that image. This can end up generating a lot of traffic.

Amazon.com might call ISPs and tell them it is being DDoS'd by certain IP address. However, the botnet machines can fake their source IP addresses.

Egress filtering

Every computer is connected to an ISP so **ISPs are in a good position to detect sent packets which have an invalid IP address** (an IP address that is not part of the ISP's network). This is called **egress filtering**.

Who benefits from it? **Someone else**! **Not the ISP!** As a result, egress filtering is not deployed by any ISPs.

From the point of view of performance costs and legal, ISPs have no excuse not to deploy egress filtering since ISPs already do "weird" stuff like deep packet inspection (DPI) and rewriting your HTML traffic to insert their own ads.

This is an example of **failure of incentives**: There exists a perfectly simple solution to the problem but since the people implementing the security feature do not get any benefits, they have no incentive to implement it.

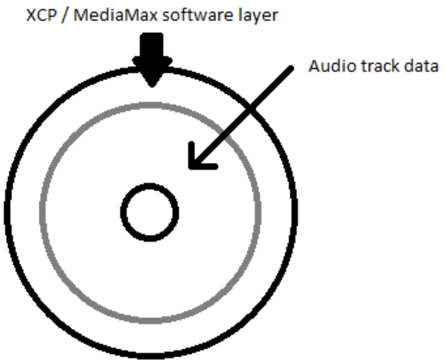
SONY DRM fiasco

Back in the day, SONY was releasing audio CDs and they wanted to **prevent piracy**. Specifically, in order to maximize their profits, SONY wanted to prevent users from copying its CDs.

There were two different vendors at the time who were building DRM software to protect audio CDs. The software solutions were:

- **XCP**
- **MediaMax** (two versions 3 and 5)

On audio CDs the information is recorded from the inside out (as opposed to LPs). Protected audio CDs had a data track on the outer edge of the CD that stored the XCP or Mediamax software.



If you played the CD on a regular player it would ignore the data track, but if you were inserting it into the CD drive of your computer it would automatically run the DRM software (assuming Windows platform and autorun enabled).

- If you played the CD on a Mac there's no autorun so SONY would hope you would click on the DRM software icon

A simple way to get past this DRM protection: Color the data track of protected CDs with a black marker to make it unreadable by CD-ROMs ☺

Normal **software asks for permission from the user to run/install** and display an **end-user license agreement (EULA).**

However, SONY needed to install **"temporary protection",** since before the user accepted the EULA he could start ripping the CD.

How the "temporary protection" worked:

1. If it found CD ripping programs running it would prompt the user to stop them. If the user did not stop it, the "temporary protection" software would eject the CD.
    a. To overcome this, you can use a ripping application that locks the CD tray.
    b. To overcome this, you can change the executable name of the CD ripping program, since the DRM protection only looked at the file name to identify the ripping programs.
2. Installed itself preemptively.
    a. There was a driver that was running and did all the CD ripping monitoring. The driver would also ensure that audio CD can only be played with the software that was on the data track on the audio CD.
        i. The driver was *temporary* since it was not marked to be restarted on reboot.
    b. The software had a bug and installed itself into a world writeable directory.
        i. If you had a multi-user system one user could mess with another user's software.
        ii. This opened a serious security hole.
        iii. On top of that, even if a user changed the permissions to be more restrictive, the permissions would be reset back to world writeable every time a protected CD was inserted.

To top it off, there was an incompatibility between version 3 and 5 of MediaMax that lead to permanent activation of the temporary protection system without user approval.

From the **business perspective**:

- SONY was a huge company, and did not want to annoy its users.
- The DRM software companies were startups and they were fighting for installation dominance, so it was in their interest to aggressively install their software on as many systems as they could

This became a **PR disaster** for SONY and the XCP and MediaMax companies because their software was aggressive, insecure, and impossible to uninstall. In the end, they allowed users to uninstall the system by providing them with an option of downloading an uninstaller.

Uninstaller disaster

The XCP and MediaMax companies did not want the uninstalling process to be convenient, so they made it a real pain. Eventually, it boiled down to installing an ActiveX control, which was a generic downloader & executer (downloaded and executed any program off the web) for the uninstall program.

However, the ActiveX control was left on after installation, so any website could use it, opening a serious security hole. Now it became a *"run on anything you want on my system"* ActiveX control for any website on the Internet.

Eventually, there was a class action lawsuit and they made a better uninstaller.

Recognizing protected CDs

XCP and MediaMax must **recognize protected CDs** when they are inserted in the CD-ROM. They have to ensure this since their goal is to have companies pay them to have their CDs protected.

They must mark the CDs with something that's **unforgeable** (this is their primary goal). Also the **mark must be indelible** (a secondary goal).

To achieve this, one company took a sequence of samples (16 bits) at around second 2 in the audio track and modified the three least significant bits in each sample to encode some information like a checksum.

However, their approach failed on both counts: forgeability and permanence.

Conclusion

Look at different companies' motives and incentives to evaluate whether a product is serving your interests or theirs.

As a developer, you have to think of the user's rights and watch out for the interest of your vendors.