Trusted Computing

Daniel Bosk

Department of Information and Communication Systems,
Mid Sweden University, Sundsvall.

24th April 2017

- The main purpose of DRM is to prevent piracy.
- This can be applied to all sorts of material; from photos, to films, to application programs, and all the way to operating systems.
- There are different approaches and purposes, e.g. to control piracy, but also to control the selling of used products.

- In the dawn of computing software was given away for free by the hardware (HW) vendors.
- This was one way to promote sales of HW, the users needed software to use the HW.
- This changed, and in the 1960's software was a significant cost.
- Now HW vendors charged extra for their OSes and there were third-party software vendors.

- In the 1970's software could be turned into general packages.
- I.e. software needed no longer be customised to the users' HW.
- Now problems with the ownership of code rose, what if one of your programmers left for a competitor and their program soon got some of your features.
- To determine if the programmer copied the source or reinvented it, software birthmarks could be used – i.e. analysing how the software is coded.

- Then came the 1980's, with these general purpose computer systems came attempts at copyright enforcement.
- Some approaches was to lock the software with an error message every few months, e.g. "Error X: Please call technical support", where X is a customer specific number.
- This worked for as long as users were technically unknowledgable and it didn't cross the limit what was considered reliable.
- Other apporaches was for the software to look at the processor's serial number.

- In summary, there was essentially three general approaches tried.
- First, to add uniqueness to the machine; e.g. a HW dongle.
- Second, to create uniqueness within it; e.g. install the software in a way that prevented naïve copying (cf. Adobe Photoshop which modified the boot loader and accidentally removes Grub).
- Generally people must be able to create a backup, but not copy those backups for sharing (copy generation control).
- And third, to use whatever uniqueness there already was; e.g. storing the characteristics of the computer, cards present, amount of memory, etc.
- This approach needs to handle HW upgrades though.

- One of the more modern approaches is to have the software connect to the vendor's servers to verify itself.
- This works as long as the software isn't needed offline.
- But even online it can be really annoying, cf. Ubisoft's Assassin's Creed DRM which required a constant connection.
- Another is to leave some critical part to be done by the vendor's servers.
- An example of this is Blizzard's Diablo 3 games, which lets the server handle the entire game (map generation, NPCs, etc.).

- However, the Blizzard approach might cause problems.
- For how long do you intend to support that product?
  - If I buy something, then I expect to be able to use it for as long as I like.
  - If you stop supporting it, and I need the product, I should be allowed to at least reverse engineer it and use that.

- Yet other approaches is to encrypt vital parts, e.g. some code or video.
- This can be used for both software and media, for which it is popular (DVD, BlueRay, streaming services).
- However, this must be decrypted before use ...
- But I can at least use the stuff for as long as I like (or have functioning equipment).

The idea

- What if a program running in a system could ascertain the integrity of the system?
- E.g. that we run a particular OS, that the OS is unmodified, that the program itself is unmodified.

Remote attestation

- We add a tamper-resistant hardware chip.
- This chip can query the rest of the hardware.
- It can then create a digitally signed summary of the hardware and attest that it is correct.
- We can even attest the running software.

Remote attestation

- We add a tamper-resistant hardware chip.
- This chip can query the rest of the hardware.
- It can then create a digitally signed summary of the hardware and attest that it is correct.
- We can even attest the running software.

Dangers

- This could be used to lock the user out of the hardware.
  - Run authentic Windows or don't use the hardware at all!
  - Linux?! Anything you create yourself?! If you're not a multimillion dollar company, who cares?

Sealed Storage

- Protects private data by binding it to the platform.
  - Use the hardware chip for encryption.
  - The chip includes the configuration as part of the key.
  - Only the chip has the key.

Example

- Encrypt your own data, no one can steal it and decrypt it elsewhere.
- If you change your hardware too much, then neither can you.

Example

- Encrypt media content with certain requirements.
- The hardware will only decrypt it if you run an unmodified version of a DRM-enforcing player.

Example

- Encrypt your own data, no one can steal it and decrypt it elsewhere.
- If you change your hardware too much, then neither can you.

Example

- Encrypt media content with certain requirements.
- The hardware will only decrypt it if you run an unmodified version of a DRM-enforcing player.

- The Trusted Platform Module (TPM) is an industry standard.
- It is maintained by the Trusted Computing Group (TCG).

- A different approach has to be taken for non-executable content, since this material cannot check itself.
- The approach here is watermarking using steganographic methods.
- However, these are also quite easily thwarted.