Trusted Hardware

- New threat model: adversary has physical control of computer
- General model:
  - We have a trusted chip that we assume hasn't been tampered with but everything else could be controlled
  - We generally use cryptography
- Problem with just having users input passwords
  - Users would need to enter passwords early in the boot process
  - Passwords tend to be weak, and adversaries can guess
- Problem we are generally trying to solve: how do we know the system is running the right software

Secure Boot

- Initial boot code comes hard-wired into the ROM
  - ROM code loads a boot loader and checks its signature
  - Boot loader loads OS kernel and checks signature
  - OS kernel checks integrity of all data it loads
  - Pass this, it is too costly to check all data
- One problem with this: we suppose we know exactly what key to sign the software, but how does hardware know which software is good / bad
- Instead:
  - System has some durably stored secret in hardware that is same across reboots
  - When system boots, it derives a secret key based on its hardware secret based on the boot loader / OS kernel / etc. that loaded
  - It uses this to decrypt the data on the drive / do authentication
  - Whenever a different software is loaded, this key changes
  - Therefore, for each combination of software that is loaded, the OS will generate a different key

Trusted Hardware

- This process relies on a separate trusted chip / device
- TPM (Trusted Platform Module)
  - Chip has an ephemeral set of registers and a key
  - PCR values get reset to zero only when computer is reset
  - CPU and TPM must reset together
- Code constantly modifies the PCR values
- This can also be used to prove to others over the network that you're running some software, but this relies on remote party to trust your TPM
- TPM can be used to seal / unseal data which basically uses the TPM key to encrypt / decrypt data

Bitlocker

- Bitlocker stores encryption key by sealing it with the TPM
- It only needs to unseal this in the BIOS
- On boot:
  - Disk has two partitions: first is BitLocker's bootstrapping code
  - Second is encrypted data
  - The first is measured at boot to be correct
  - The second is encrypted with the TPM and relies on getting the encrypted key out properly (so the bootstrap code and all code before it cannot be modified)
  - We can't do the same strategy of just measuring the second because it changes too often

- One problem: this relies on the fact that users cannot change ciphertext to reliably get certain chances in the plaintext
    - So far, this has shown to be true
    - Known as "poor-man's authentication"