**Disk Encryption Attacks**

**Breaking Cryptoalgorithms and Brute Force Attacks**

A good cryptoalgorithm is one that currently cannot be broken; however, this doesn’t mean that in the future, there won’t be ways to crack it

* Good cryptoalgorithms only last 10-20 years

The only current attack that could work is Brute Force Attacks

Quantum computers are coming though which will half the strength of our current algorithms

* 128-bit can be broken with a quantum computer
* Brute forcing the entire 256-bit keyspace isn’t possible as it comes up against the limits of physics, so 256 is futureproofing

Meaning we would need to double the key size to get the same level of protection that we have today

First operational quantum computer is owned by GOOGLE

Make sure strong passwords are used to create the encryption keys too, only as strong as your weakest link

**Implementation Vulnerabilities**

* Weaknesses in the algorithm
* Open source is good as it can be publicly scrutinized

**Configuration issues**

* Some algorithms allow you to make configurations in a way that make them weaker

These can be deliberately introduced by adversaries

**Unknown errors and Attacks**

Need compensating controls for attacks you don’t know about yet

One could be that you never let your device out of your site or allow someone else on it

**Physical Attacks**

Don’t allow adversary to touch your device

Fully switched off and locked away

DMA attacks (direct Memory Attacks)

Memory can be directly accessed via the DMA port and the encryption key can be extracted

Tools like Inception can be used to do this

* Most DMA ports are now blocked unless the user is logged in

**Pre-boot Authentication**

The blocks where the OS is stored, must be decrypted before the OS can boot meaning the key must be available before there is any UI to ask for a password, this creates an attack surface

Hardware, such as a trusted platform module can be used to check for integrity, checking against system variables, can be used for authentication

With a pre-boot authentication environment, the key used to encrypt the data is not decrypted until an external key (password) is input into the system

**The Cold Boot Attack**

A dump of the machine’s volatile memory

* Encryption keys

Physically freeze the memory – to maintain integrity like DNA

Take it out, add to another machine, dump memory

**The Boot Key problem**

The boot partitions can’t be encrypted with only software encryptions

This can leave opportunity for an Evil Maid Attack

**Evil Maid**

Boots device using a different OSD (Live CD) and changes the bootloader to a hacked bootloader version

Next time the device is switched on and they key is entered, it’ll then be captured by the evil bootloader version

* Which can then be sent back to a destination or stored for later retrieval

If the bootloader is compromised, then the whole machine is essentially

* The bootloader can also be exploited via network malware

**Knowledgeable Evil Maid**

Someone who has access to device and also knows where certain data types and files are kept

* They could modify exact encrypted bits
* Modified in specific ways to compromise encryption
  + Insert code to download malware on next start-up, maybe a keylogger to capture the key next time its typed in

**Container, Volumes and Partitions**

Most of the same conditions apply as WDE but there is a bigger attack surface as there are files that aren’t encrypted

* Most OS files won’t be encrypted
  + Registry
  + Swap
  + Crash dump
  + Virtual memory

When you finally go to ‘end of life’ your machine, its easier when you have WDE

Advantages

* Separate containers for levels of importance, keep high level ones closed until needed
* Plausible deniability, some crypto systems don’t contain a signature of encryption
* Can combine both Container and WDE but this will slow the system down

SEDs

* SSD controller usually contains a full hardware encryption engine (126 or 256-bit AES)
* Encryption is hidden behind controller so cannot be scrutinised
* Randomly generates key, writes all data to drive using this key
  + Self-encrypting and self-decrypting which is helpful to no one
  + If a password can be entered to decrypt is what would be useful
* Even still, there are implementation vulnerabilities as they cant be relied upon as they are not open-source like software-based encryption and therefore untrustworthy