Secured Boot:
A Practical Introduction

A lecture to get you ready for Assignment 1!

Demo files are on Canvas Announcement

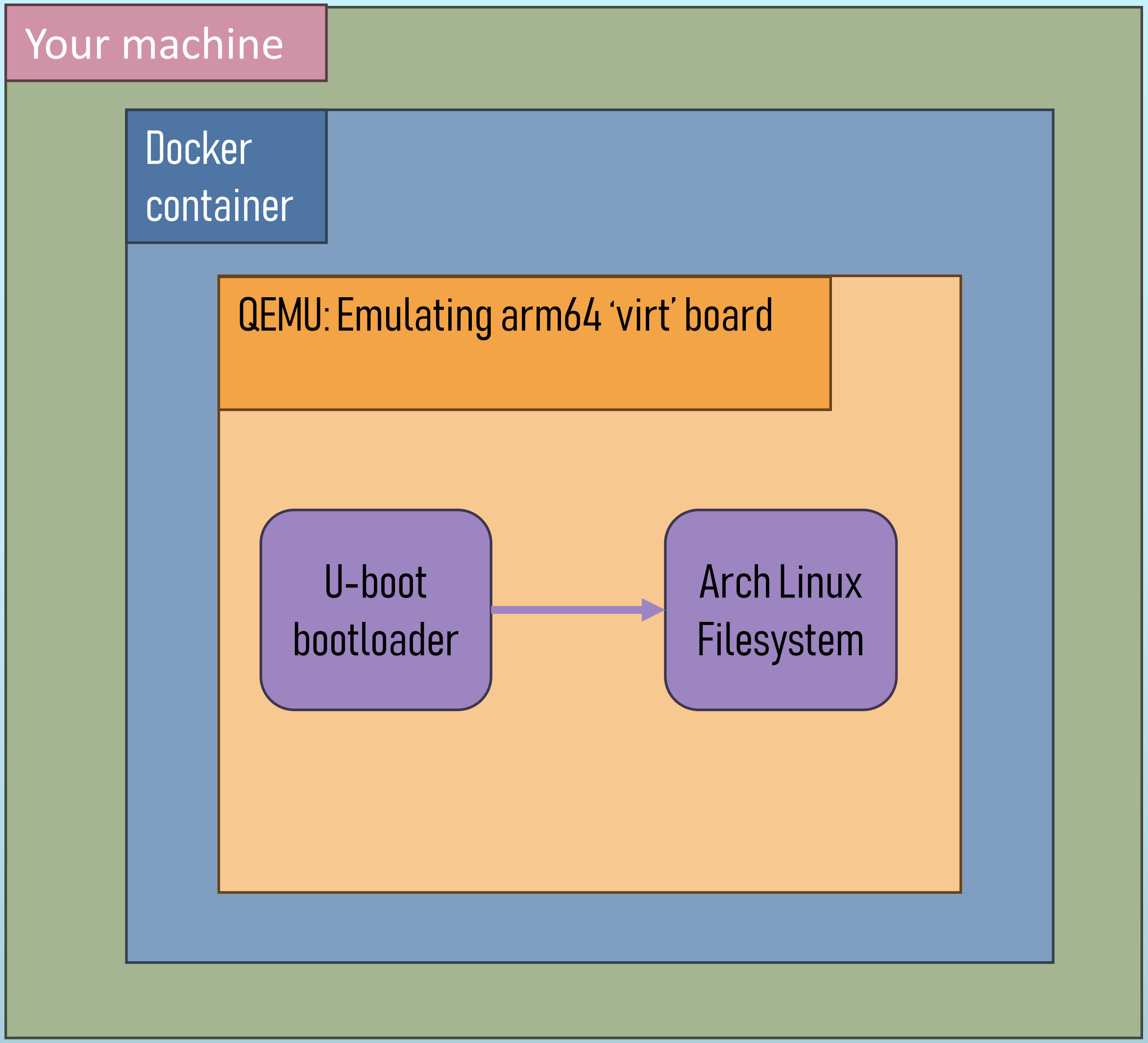
What we will cover

- Assignment Design Overview
- How to run the required tooling: Docker Introduction
- QEMU
- Device trees
- U-Boot
- Disk encryption

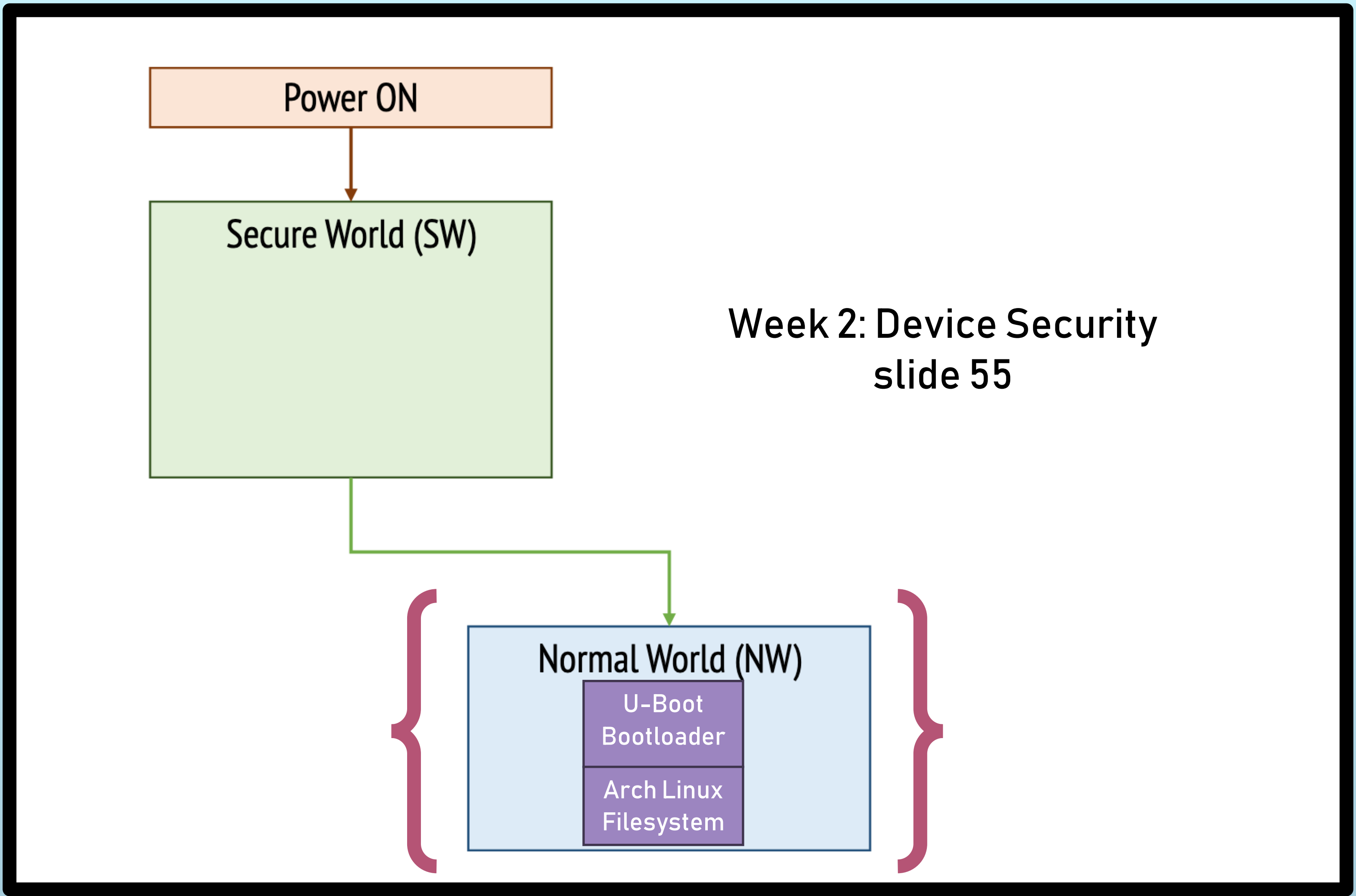
- Slides will be put up on Canvas

Assignment Setup

- You will be configuring an arm64 platform
    - Emulation of system: QEMU
    - Bootloader: U-Boot
    - OS: Arch Linux
- We have also have a Dockerfile for you to use
    - It will setup an Ubuntu Linux environment with all the required software already installed
    - You don't need to use this for the assignment, but it is recommended!



Boot flow in context



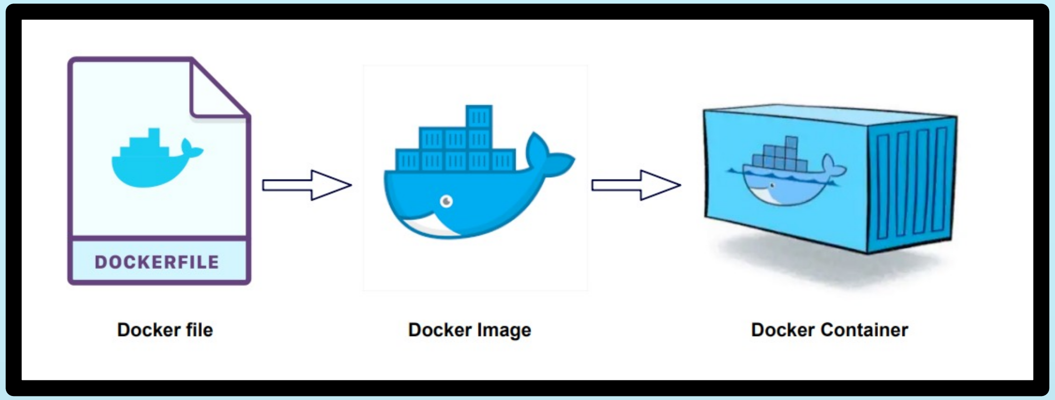
Week 2: Device Security
slide 55

Docker

Running the assignment

Using Docker

- Docker builds **Images,** from which one can run **containers**
  - **Containers**: a program based on an **Image**
- Docker uses **Dockerfiles** to build Images
  - Scripts that tell Docker how to build your image
  - Usually named 'Dockerfile'
- The Docker container is **not** part of the platform you are assessing
  - You don't have to use Docker, but you will need to install the required tools yourself



Dockerfiles **build** the Image, and then using the Image we **run** a container

DEMO:
Docker build & run

In the dmss_demo folder, there is a file called `Dockerfile`

1. `docker build` downloads dependencies and creates an image based on `Dockerfile`
2. `docker run` runs a container based on the created image

```
docker build -t demo .
```

Note: ‘-t’ flag here names the image ‘demo’

```
docker run -it demo
```

Note: ‘-it’ will allow us to run commands inside the container

QEMU

Emulation software

DEMO: QEMU

- A machine emulator
  - A good way to test in-development firmware and software
- Similar, but not equivalent to, virtualizing software
- Lots of options and command line parameters!
  - Storage drives
  - Peripheral devices

How to start QEMU emulating the `virt` board, with the bootloader image `u-boot.bin`

```
qemu-system-aarch64 –cpu cortex-a57 –smp 4 –m 5G
-machine virt,virtualization=on –bios u-boot.bin
–nographic
```

How to quit QEMU (when running with `–nographic`)

```
Ctrl-A  x
```

i.e. Press Control- A, and then press x

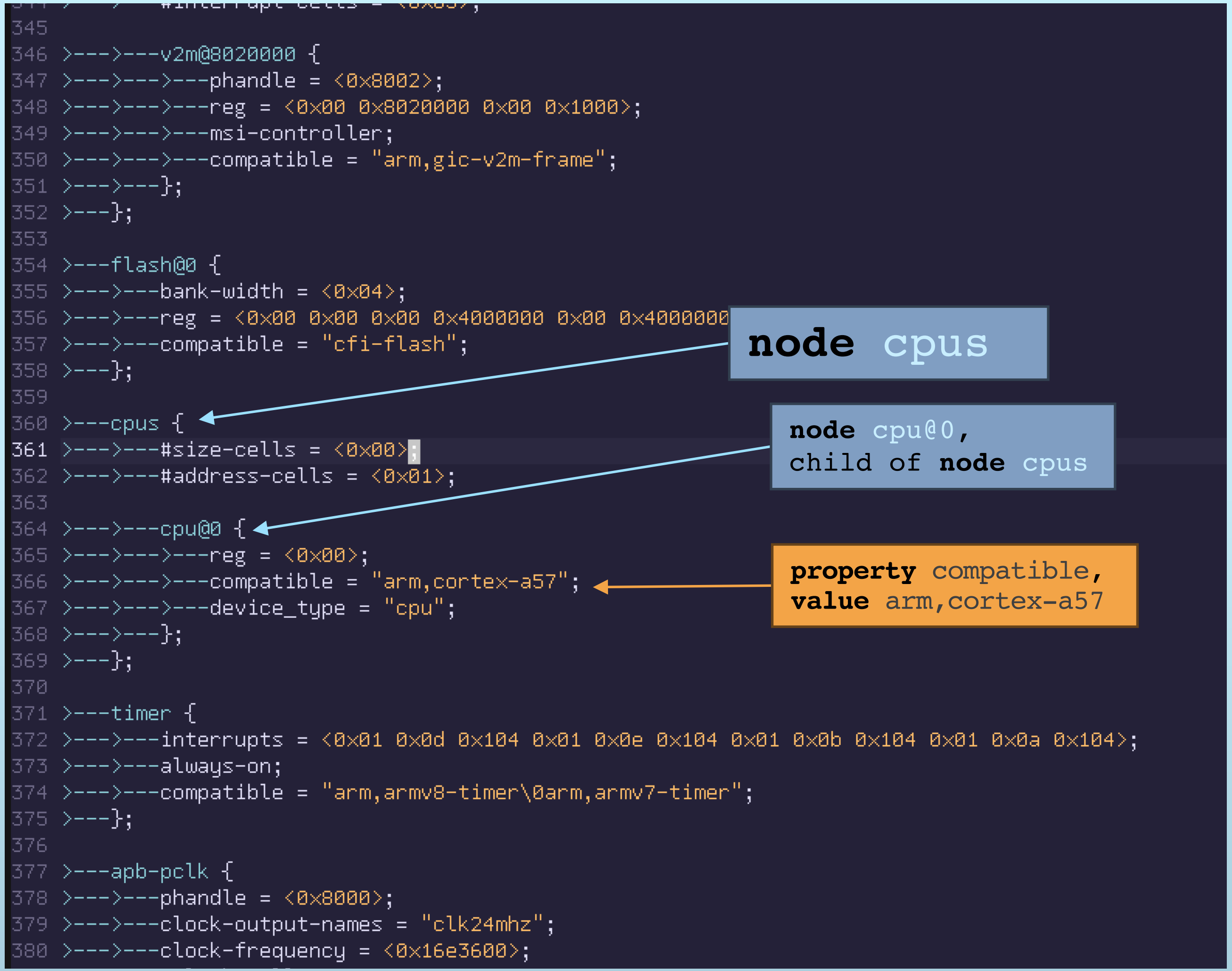
ERROR!!! ‘No valid **device tree binary** found at 0000xxx'

Device Trees

A way to describe hardware components of a computer system

Quick overview of device trees

- A data structure for describing computer platforms
  - Consist of **nodes** and **properties**
- Firmware and Operating Systems need to know what hardware is present on a system
  - Device trees provide this description
- Platform vendors often provide device trees for their platforms



```
>--->---v2m@8020000 {
>--->--->---phandle = <0x8002>;
>--->--->---reg = <0x00 0x8020000 0x00 0x1000>;
>--->--->---msi-controller;
>--->--->---compatible = "arm,gic-v2m-frame";
>--->---};
>---};

>---flash@0 {
>--->---bank-width = <0x04>;
>--->---reg = <0x00 0x00 0x00 0x4000000 0x00 0x4000000
>--->---compatible = "cfi-flash";
>---};

>---cpus {
>--->---#size-cells = <0x00>;
>--->---#address-cells = <0x01>;

>--->---cpu@0 {
>--->--->---reg = <0x00>;
>--->--->---compatible = "arm,cortex-a57";
>--->--->---device_type = "cpu";
>--->---};
>---};

>---timer {
>--->---interrupts = <0x01 0x0d 0x104 0x01 0x0e 0x104 0x01 0x0b 0x104 0x01 0x0a 0x104>;
>--->---always-on;
>--->---compatible = "arm,armv8-timer\0arm,armv7-timer";
>---};

>---apb-pclk {
>--->---phandle = <0x8000>;
>--->---clock-output-names = "clk24mhz";
>--->---clock-frequency = <0x16e3600>;
```

node cpus
node cpu@0, child of **node** cpus
property compatible, **value** arm,cortex-a57

Devicetree Compiler

- `dtc` in Linux
- Transforms devicetree source files (`.dts`) to devicetree binary (`.dtb`) files, and vice-versa

Transforming devicetree **source** file `board.dts` to a devicetree binary called `output.dtb`

```
dtc –I dts -O dtb board.dts -o output.dtb
```

Transforming devicetree **binary** file `output.dtb` to a devicetree binary called `output_src.dts`

```
dtc –I dtb -O dts output.dtb -o output_src.dts
```

Note: You may get a string of warnings when compiling `.dts` files, you can *mostly* safely ignore these.

Fixing the ’no valid device tree’ error

We need to pass in a .dtb to QEMU at the specified address

```
qemu-system-aarch64 –smp 4 –m 5G -machine virt,virtualization=on –cpu cortex-
a57 –bios u-boot.bin -device loader,file=output.dtb,addr=0x<addr> –nographic
```

In QEMU, the **Generic Loader** device option loads images or values at specific memory addresses on the system.

Syntax: `-device loader,file=<file>,addr=<addr>`

Note: No spaces between the different arguments to device!

You can have multiple loader devices in one QEMU invocation to load multiple images at different addresses

Once U-boot can find our devicetree, it should start successfully

U-Boot

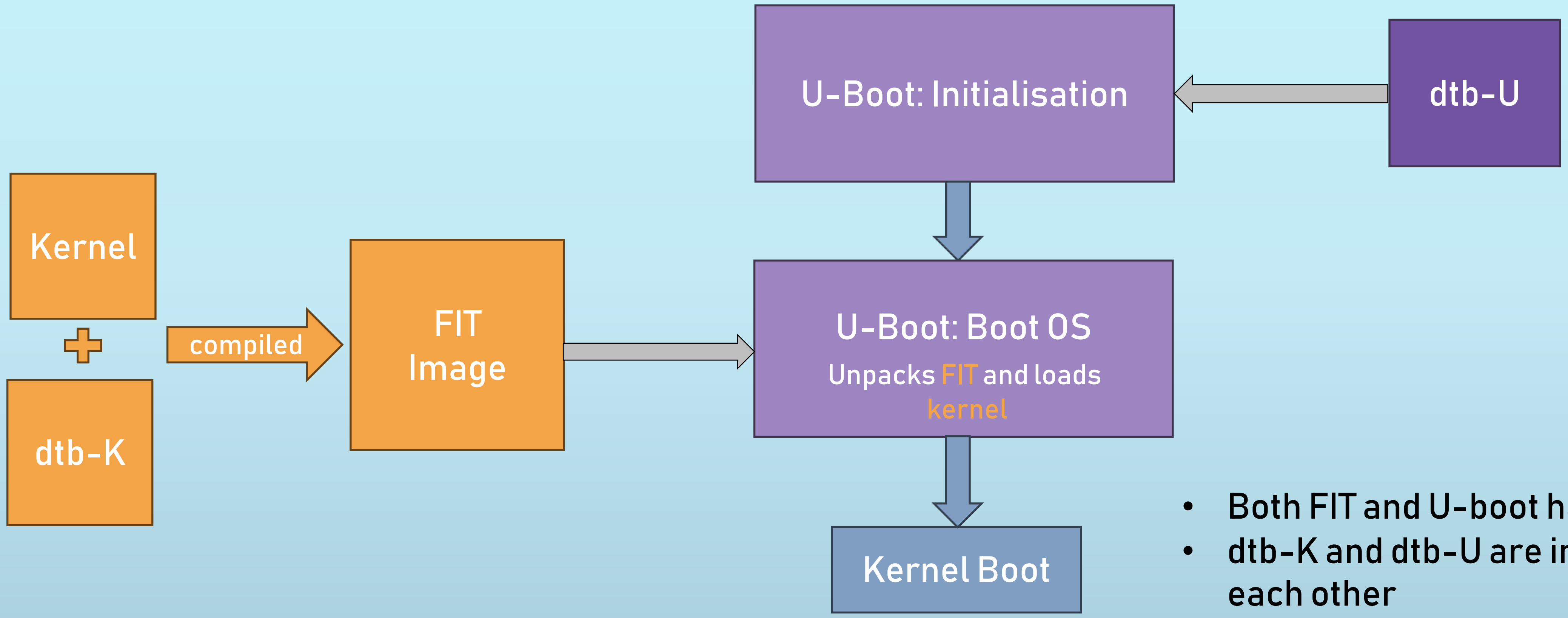
- Open-source bootloader targeting embedded platforms
- Lots of configuration options!
  - Boot methods: memory addresses, network
  - Scripting support
  - **Image Verification**
- You will be assessing a system using U-boot's **Verified Boot**

U-Boot FIT Images

- Flattened Image Tree
  - Source file extension: `.its`
  - Binary file extension: `.itb`
- Describes and packages multiple binaries into one image
- In this arm64 platform FIT, we have:
  - Kernel Image
  - Platform dtb file
  - Ramdisk image <optional>

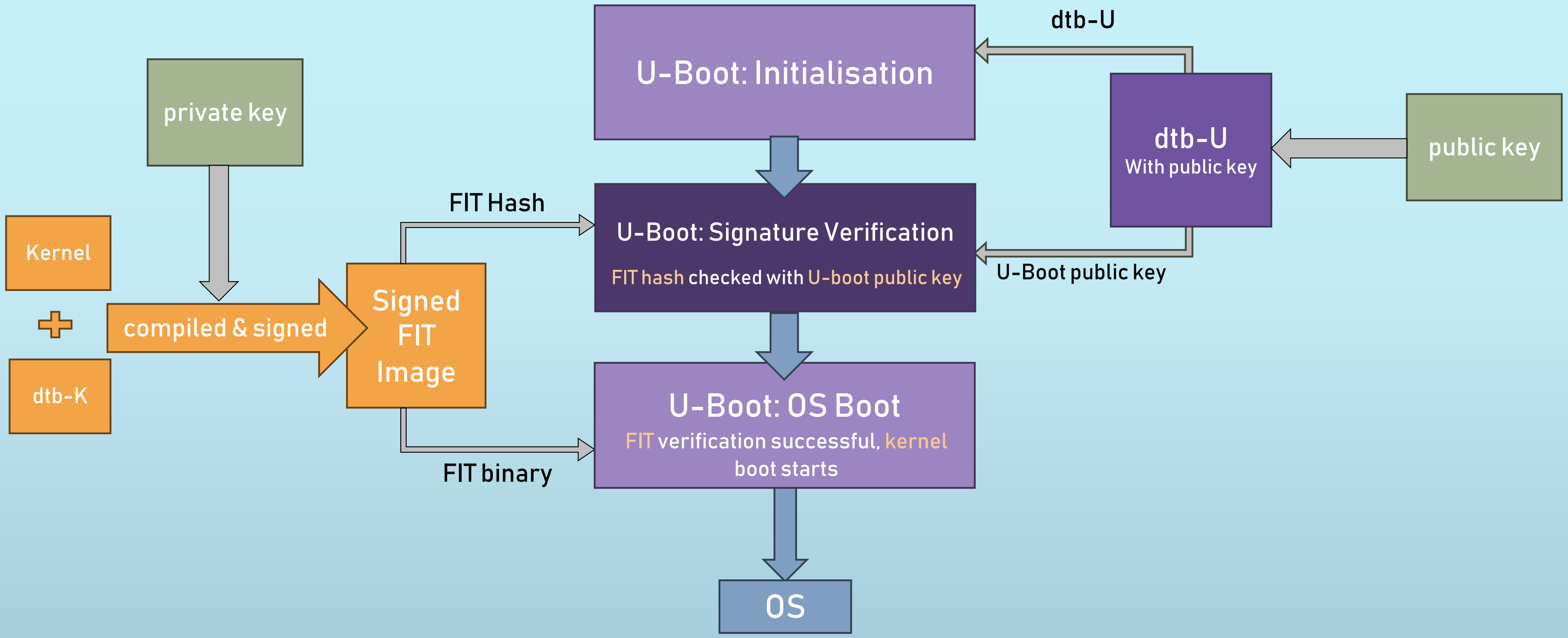
```
/dts-v1/;

/ {
description = "Simple image with single Linux kernel and FDT blob";
#address-cells = <1>;

images {
        kernel {
        description = "Vanilla Linux kernel";
        data = /incbin/("./vmlinux.bin.gz");
        type = "kernel";
        arch = "ppc";
        os = "linux";
        compression = "gzip";
        load = <00000000>;
        entry = <00000000>;
        hash-1 {
                algo = "crc32";
        };
        hash-2 {
                algo = "sha256";
        };
        };
        fdt-1 {
        description = "Flattened Device Tree blob";
        data = /incbin/("./target.dtb");
        type = "flat_dt";
        arch = "ppc";
        compression = "none";
        hash-1 {
                algo = "crc32";
        };
        hash-2 {
                algo = "sha256";
        };
        };
};

configurations {
        default = "conf-1";
        conf-1 {
        description = "Boot Linux kernel with FDT blob";
        kernel = "kernel";
        fdt = "fdt-1";
        };
};
};
```

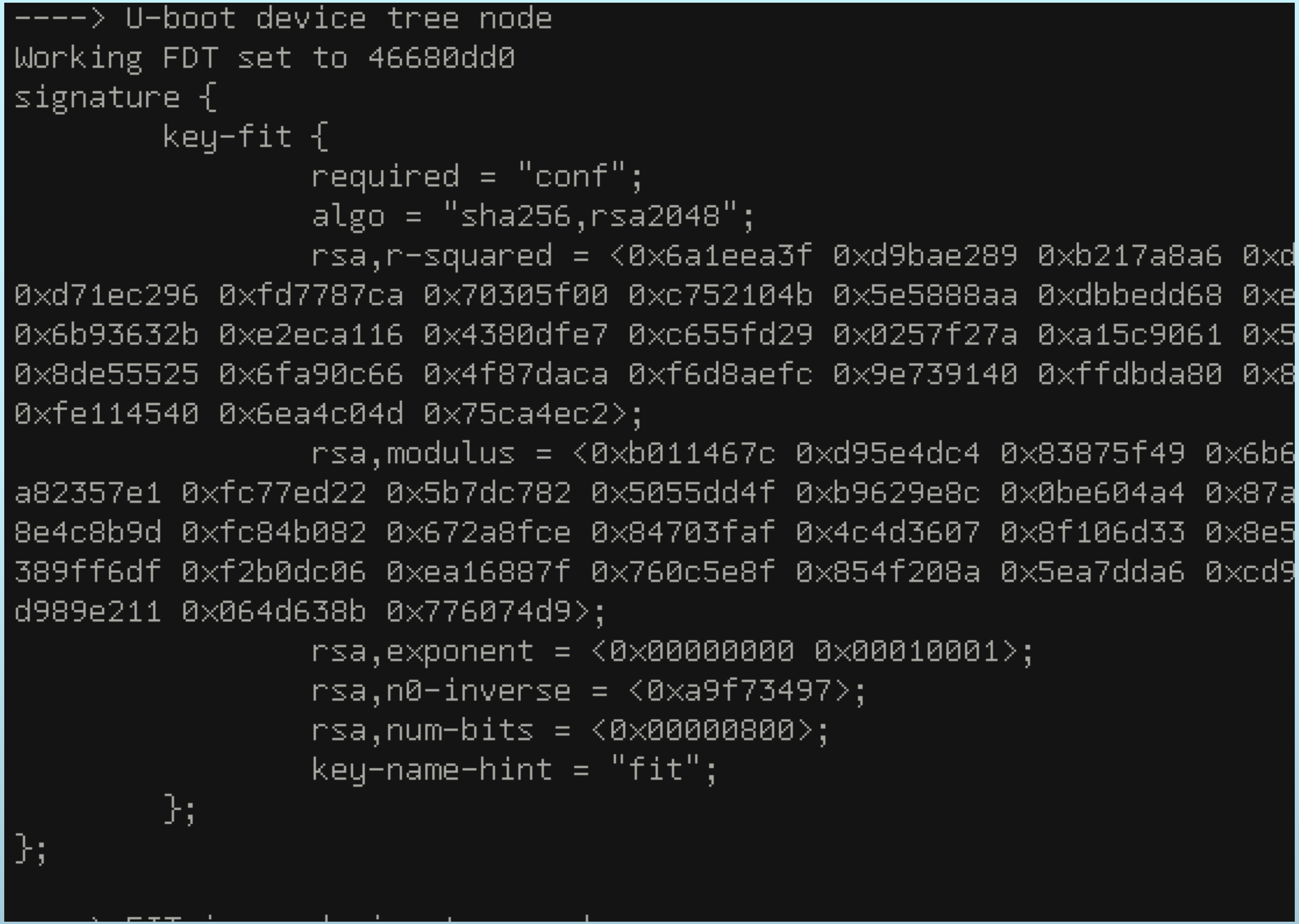
U-Boot: Booting a FIT image



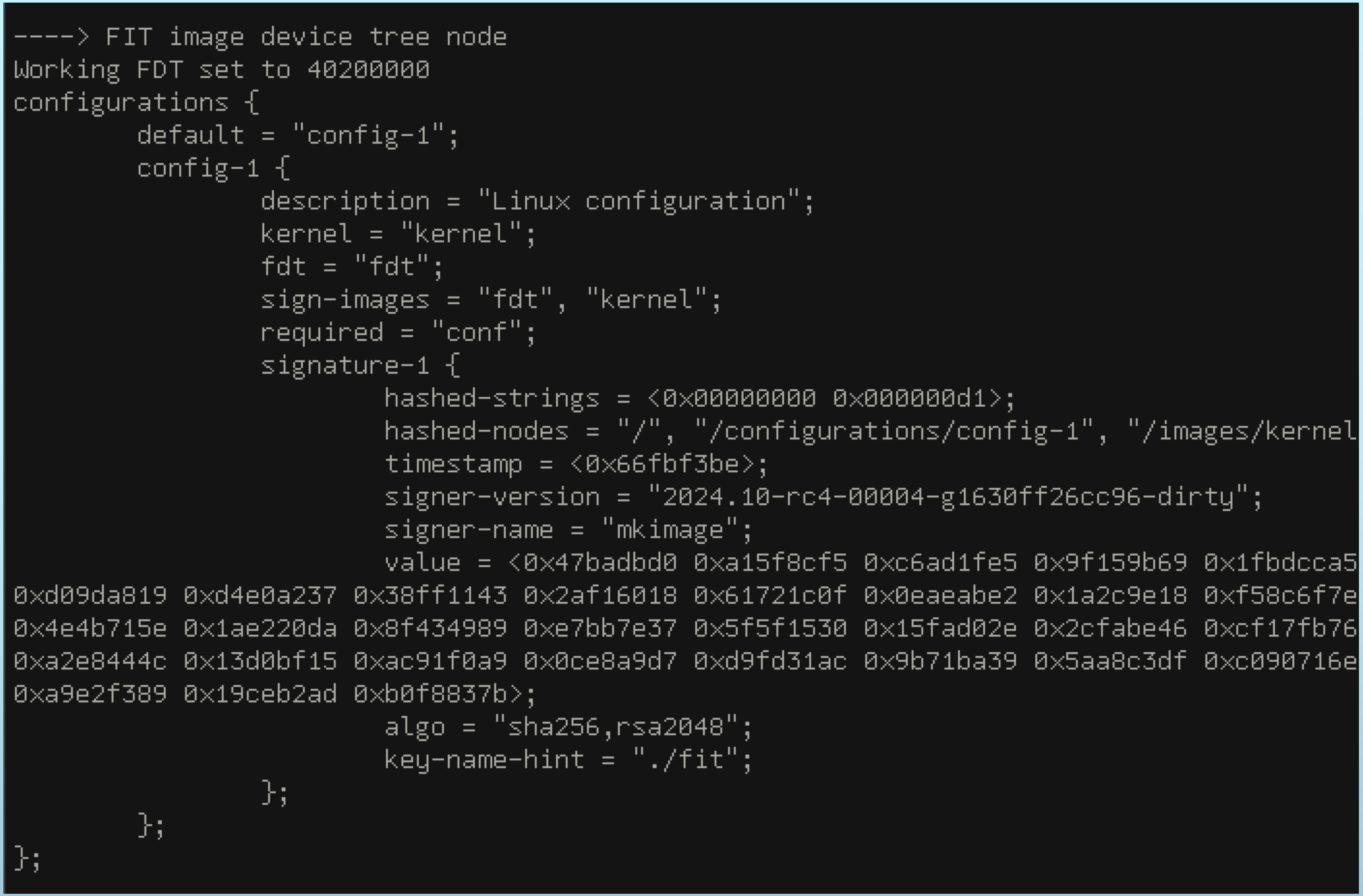
- Both FIT and U-boot have a dtb
- dtb-K and dtb-U are independent of each other

U-Boot: Booting & Verifying a FIT image





```
----> U-boot device tree node
Working FDT set to 46680dd0
signature {
        key-fit {
                required = "conf";
                algo = "sha256,rsa2048";
                rsa,r-squared = <0x6a1eea3f 0xd9bae289 0xb217a8a6 0xd
0xd71ec296 0xfd7787ca 0x70305f00 0xc752104b 0x5e5888aa 0xdbbedd68 0xe
0x6b93632b 0xe2eca116 0x4380dfe7 0xc655fd29 0x0257f27a 0xa15c9061 0x5
0x8de55525 0x6fa90c66 0x4f87daca 0xf6d8aefc 0x9e739140 0xffdbda80 0x8
0xfe114540 0x6ea4c04d 0x75ca4ec2>;
                rsa,modulus = <0xb011467c 0xd95e4dc4 0x83875f49 0x6b6
a82357e1 0xfc77ed22 0x5b7dc782 0x5055dd4f 0xb9629e8c 0x0be604a4 0x87a
8e4c8b9d 0xfc84b082 0x672a8fce 0x84703faf 0x4c4d3607 0x8f106d33 0x8e5
389ff6df 0xf2b0dc06 0xea16887f 0x760c5e8f 0x854f208a 0x5ea7dda6 0xcd9
d989e211 0x064d638b 0x776074d9>;
                rsa,exponent = <0x00000000 0x00010001>;
                rsa,n0-inverse = <0xa9f73497>;
                rsa,num-bits = <0x00000800>;
                key-name-hint = "fit";
        };
};
```



```
----> FIT image device tree node
Working FDT set to 40200000
configurations {
        default = "config-1";
        config-1 {
                description = "Linux configuration";
                kernel = "kernel";
                fdt = "fdt";
                sign-images = "fdt", "kernel";
                required = "conf";
                signature-1 {
                        hashed-strings = <0x00000000 0x000000d1>;
                        hashed-nodes = "/", "/configurations/config-1", "/images/kernel
                        timestamp = <0x66fbf3be>;
                        signer-version = "2024.10-rc4-00004-g1630ff26cc96-dirty";
                        signer-name = "mkimage";
                        value = <0x47badbd0 0xa15f8cf5 0xc6ad1fe5 0x9f159b69 0x1fbdcca5
0xd09da819 0xd4e0a237 0x38ff1143 0x2af16018 0x61721c0f 0x0eaeabe2 0x1a2c9e18 0xf58c6f7e
0x4e4b715e 0x1ae220da 0x8f434989 0xe7bb7e37 0x5f5f1530 0x15fad02e 0x2cfabe46 0xcf17fb76
0xa2e8444c 0x13d0bf15 0xac91f0a9 0x0ce8a9d7 0xd9fd31ac 0x9b71ba39 0x5aa8c3df 0xc090716e
0xa9e2f389 0x19ceb2ad 0xb0f8837b>;
                        algo = "sha256,rsa2048";
                        key-name-hint = "./fit";
                };
        };
};
```

Useful U-Boot commands

1. `help`	1. Prints out U-boot command list
2. `printenv`	2. Prints U-boot environment variables
• `printenv bootargs`	
3. `setenv`	3. Sets U-Boot environment variables
• `setenv bootargs …`	
• `setenv -a bootargs …`	
4. `fdt`	4. fdt utitlites: setting working fdt and printing out working fdt information
• `fdt addr $loadaddr`	
• `fdt addr $fdtcontroladdr`	
• `fdt list`	
5. `iminfo`	5. Prints out loaded image information
6. `boot`	6. Runs default boot command

Disk Encryption

Encrypting data in a Linux system

dm-crypt and Linux Unified Key Setup (LUKS)

https://wiki.archlinux.org/title/Dm-crypt/Device_encryption

- **dm-crypt**: Disk encryption system for the Linux Kernel
  - Can encrypt entire storage disks e.g. root file systems
- Encryption is secured with **keys**, which are either:
  - Passphrases
  - Keyfiles
- During boot, the encrypted volume will need to be decrypted
  - Enter passphrase
  - Put keyfile in initramfs - automatically decrypted
  - Need to set `root=…` kernel bootarg - see wiki!

View LUKS header/key information

```
cryptsetup luksDump <device>
```

OR

```
systemd-cryptenroll <device>
```

Further options in man pages online!

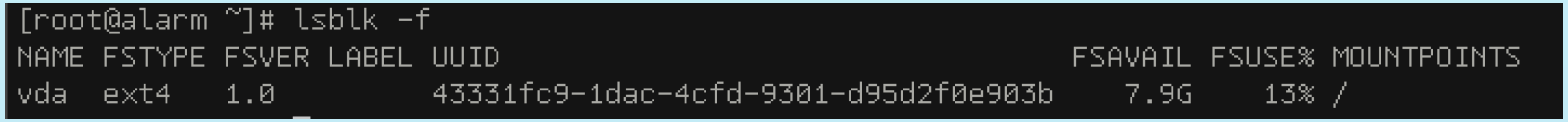
https://man.archlinux.org/man/core/cryptsetup/cryptsetup.8.en

https://man.archlinux.org/man/systemd-cryptenroll.1

dm-crypt comparison

- Without encryption

  `/dev/vda` Unencrypted volume, ext4 filesystem mounted at /

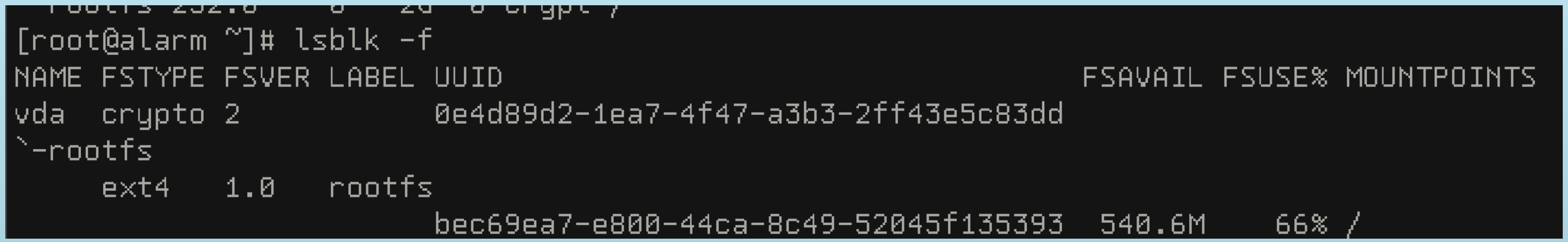


```
[root@alarm ~]# lsblk -f
NAME FSTYPE FSVER LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
vda  ext4   1.0         43331fc9-1dac-4cfd-9301-d95d2f0e903b    7.9G    13% /
```

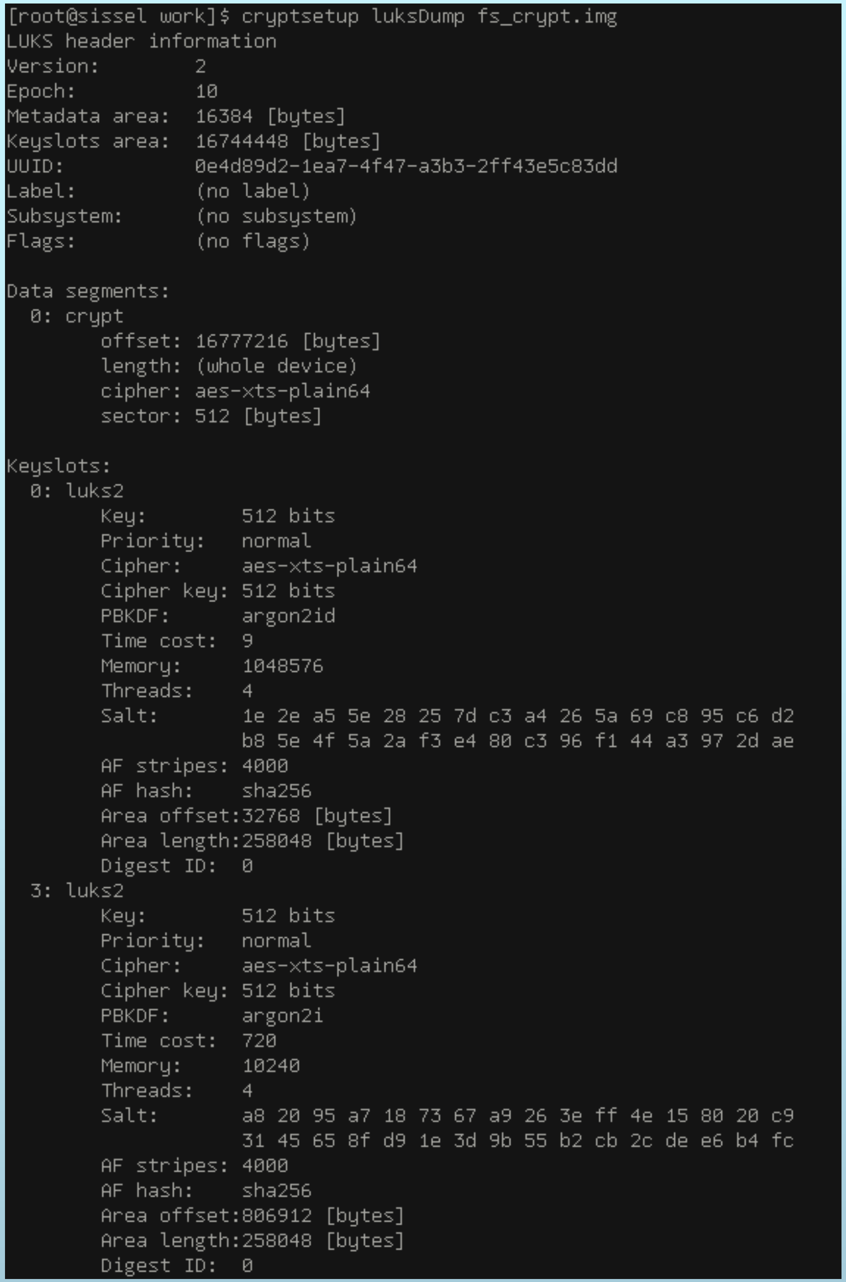
- With encryption

  `/dev/vda` LUKS encrypted volume, decrypted by kernel

  └→ `/dev/mapper/rootfs` Decrypted LUKS volume, ext4 filesystem mounted at /

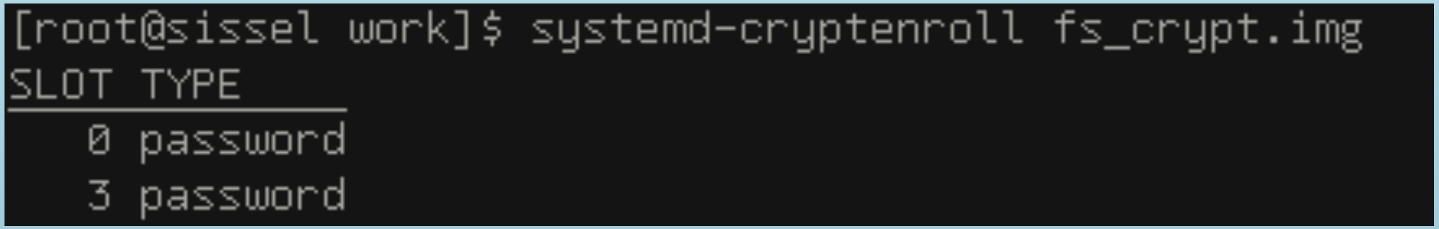


```
  rootfs 252:0    0   2G  0 crypt /
[root@alarm ~]# lsblk -f
NAME FSTYPE FSVER LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
vda  crypto 2           0e4d89d2-1ea7-4f47-a3b3-2ff43e5c83dd
`-rootfs
     ext4   1.0   rootfs
                        bec69ea7-e800-44ca-8c49-52045f135393  540.6M    66% /
```



```
[root@sissel work]$ cryptsetup luksDump fs_crypt.img
LUKS header information
Version:        2
Epoch:          10
Metadata area:  16384 [bytes]
Keyslots area:  16744448 [bytes]
UUID:           0e4d89d2-1ea7-4f47-a3b3-2ff43e5c83dd
Label:          (no label)
Subsystem:      (no subsystem)
Flags:          (no flags)

Data segments:
  0: crypt
        offset: 16777216 [bytes]
        length: (whole device)
        cipher: aes-xts-plain64
        sector: 512 [bytes]

Keyslots:
  0: luks2
        Key:        512 bits
        Priority:   normal
        Cipher:     aes-xts-plain64
        Cipher key: 512 bits
        PBKDF:      argon2id
        Time cost:  9
        Memory:     1048576
        Threads:    4
        Salt:       1e 2e a5 5e 28 25 7d c3 a4 26 5a 69 c8 95 c6 d2
                    b8 5e 4f 5a 2a f3 e4 80 c3 96 f1 44 a3 97 2d ae
        AF stripes: 4000
        AF hash:    sha256
        Area offset:32768 [bytes]
        Area length:258048 [bytes]
        Digest ID:  0
  3: luks2
        Key:        512 bits
        Priority:   normal
        Cipher:     aes-xts-plain64
        Cipher key: 512 bits
        PBKDF:      argon2i
        Time cost:  720
        Memory:     10240
        Threads:    4
        Salt:       a8 20 95 a7 18 73 67 a9 26 3e ff 4e 15 80 20 c9
                    31 45 65 8f d9 1e 3d 9b 55 b2 cb 2c de e6 b4 fc
        AF stripes: 4000
        AF hash:    sha256
        Area offset:806912 [bytes]
        Area length:258048 [bytes]
        Digest ID:  0
```

- These commands also work on live device volumes
  - i.e. /dev/vda
- Note: No indication whether a passphrase or a keyfile was used for each keyslot!



```
[root@sissel work]$ systemd-cryptenroll fs_crypt.img
SLOT TYPE
   0 password
   3 password
```

cryptsetup & systemd-cryptenroll

Adding keys (passphrase)

```
cryptsetup luksAddKey <device>
systemd-cryptenroll <device> --password
```

Adding keys (keyfile)

```
cryptsetup luksAddKey <device> keyfile.key
```

Will prompt for existing passphrase, or can authenticate with an existing keyfile with `-d existing.key`

Changing keys

```
cryptsetup luksChangeKey <device> <new key file>
```

cryptsetup & systemd-cryptenroll

Removing keys (passphrase)

```
cryptsetup luksRemoveKey <device>
systemd-cryptenroll <device> --wipe-slot=<SLOT>
```

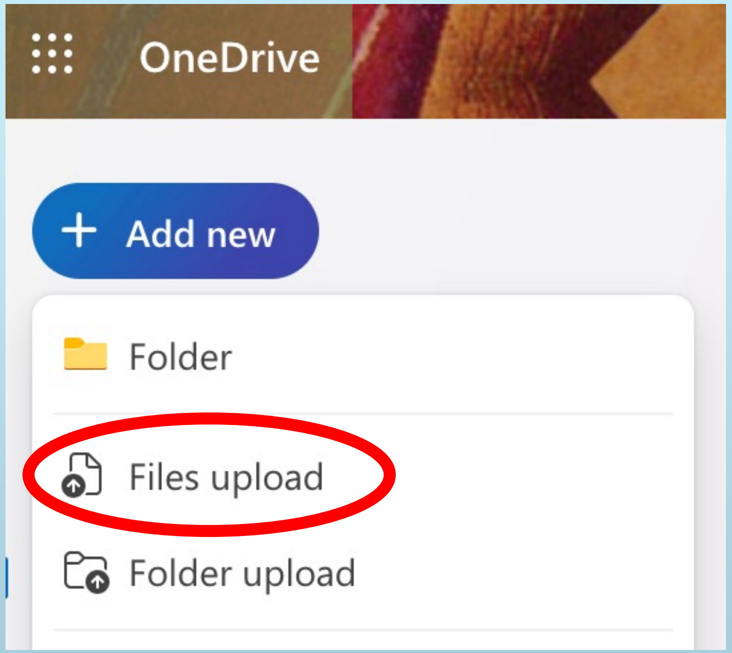
Removing keys (keyfile)

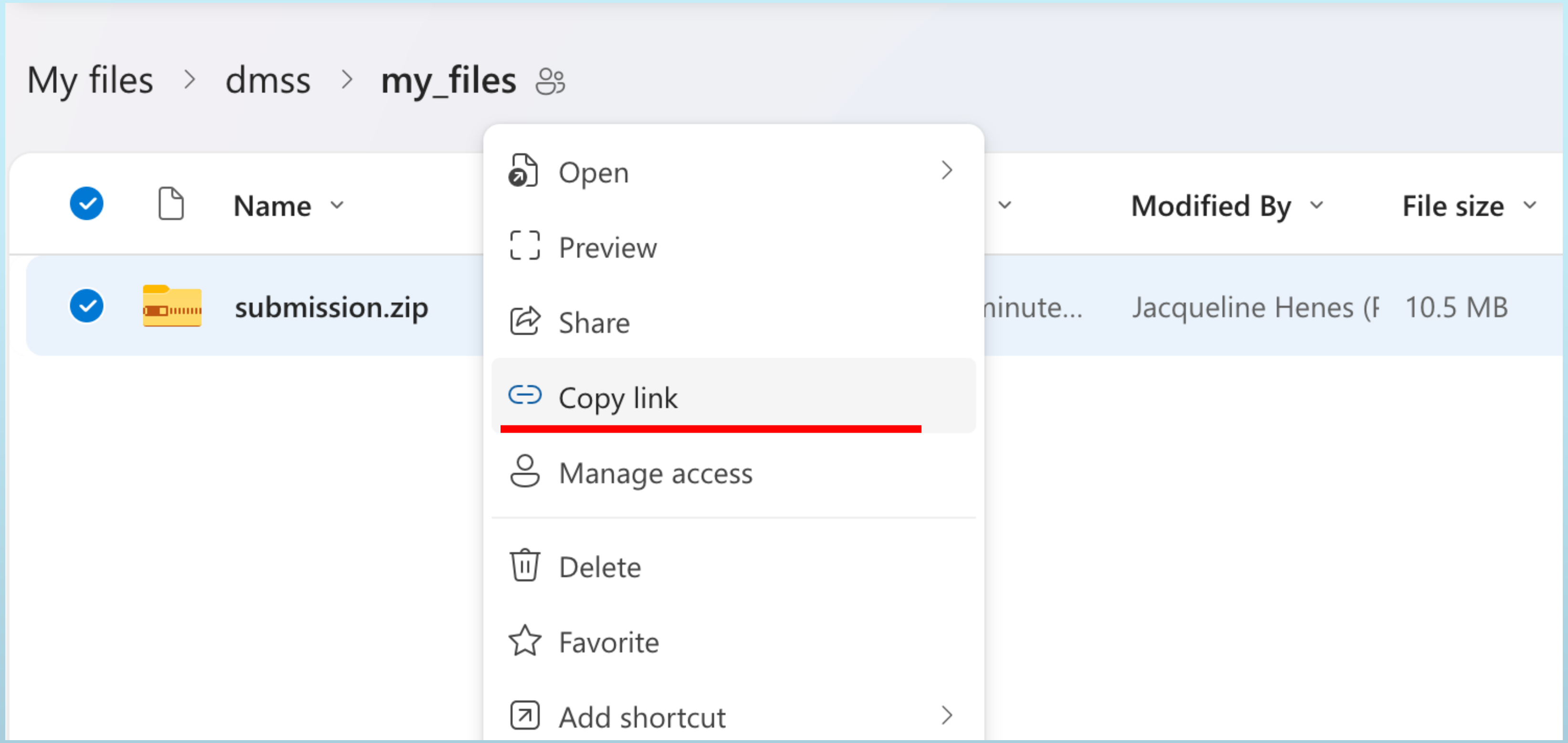
```
cryptsetup luksRemoveKey <device> keyfile.key
```

Will prompt for existing passphrase, or can authenticate with an existing keyfile with `-d existing.key`

Uploading files to OneDrive

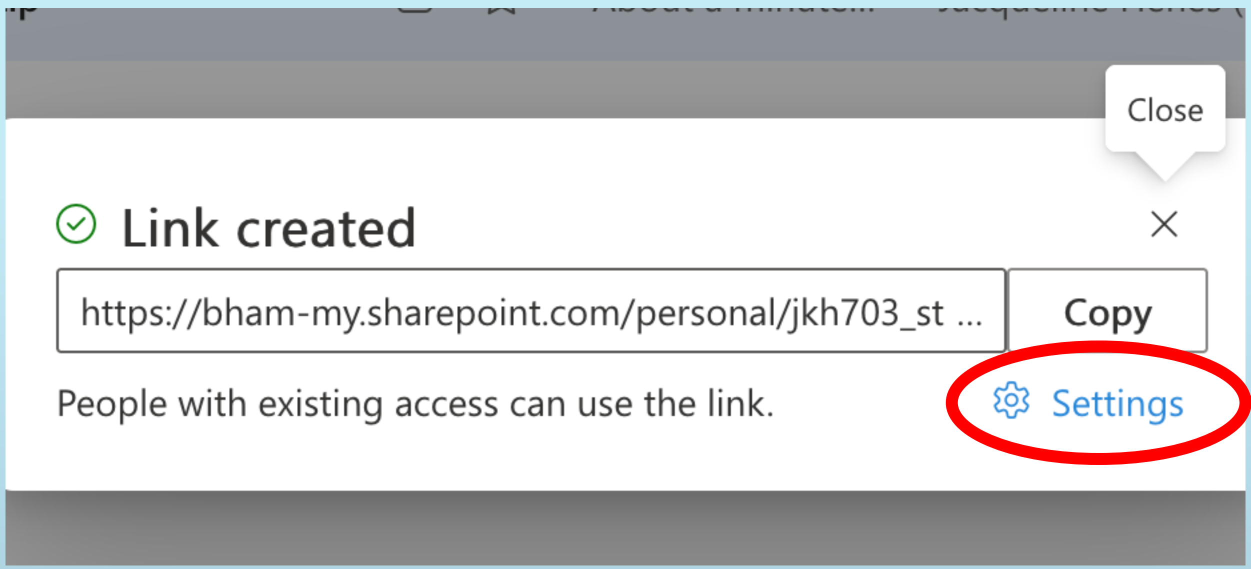
- https://bham-my.sharepoint.com/





Uploading files to OneDrive

- Put link to file in submission report!

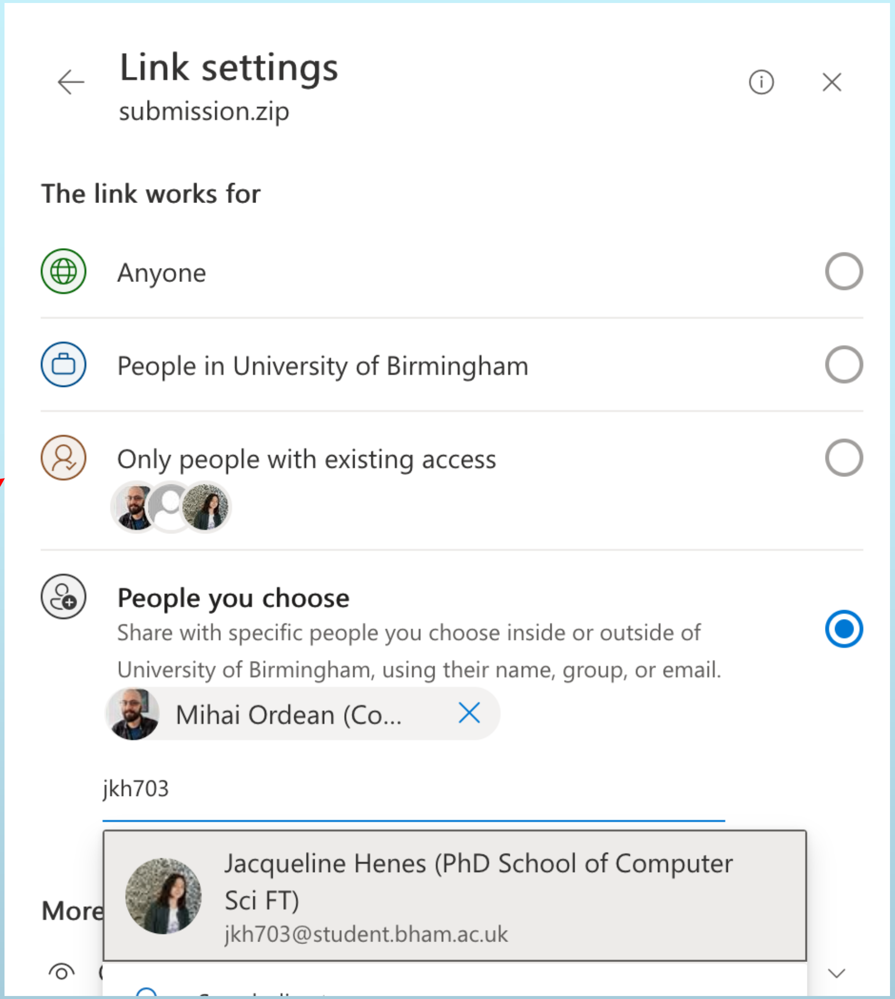


Close

Link created

https://bham-my.sharepoint.com/personal/jkh703_st ... Copy

People with existing access can use the link. Settings



Link settings
submission.zip

The link works for

- Anyone ( )
- People in University of Birmingham ( )
- Only people with existing access ( )
- People you choose (●)
  Share with specific people you choose inside or outside of University of Birmingham, using their name, group, or email.

Mihai Ordean (Co...

jkh703

Jacqueline Henes (PhD School of Computer Sci FT)
jkh703@student.bham.ac.uk

More

Last few notes

- assignment_faq.md has some useful information and guidance: Please read it!
- Looking for understanding
  - Concepts you have been learning about applied in practice
- The internet is a wonderful tool
  - Most (all?) tools used are open-source, with documentation