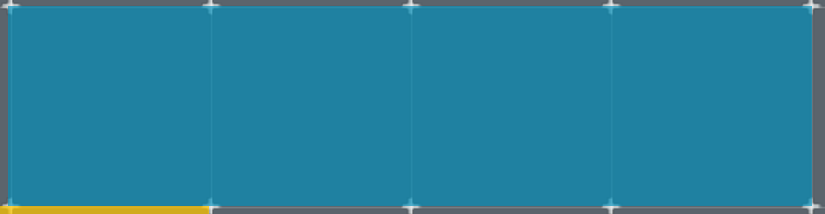
WISE-3610 Mbed Cloud Update

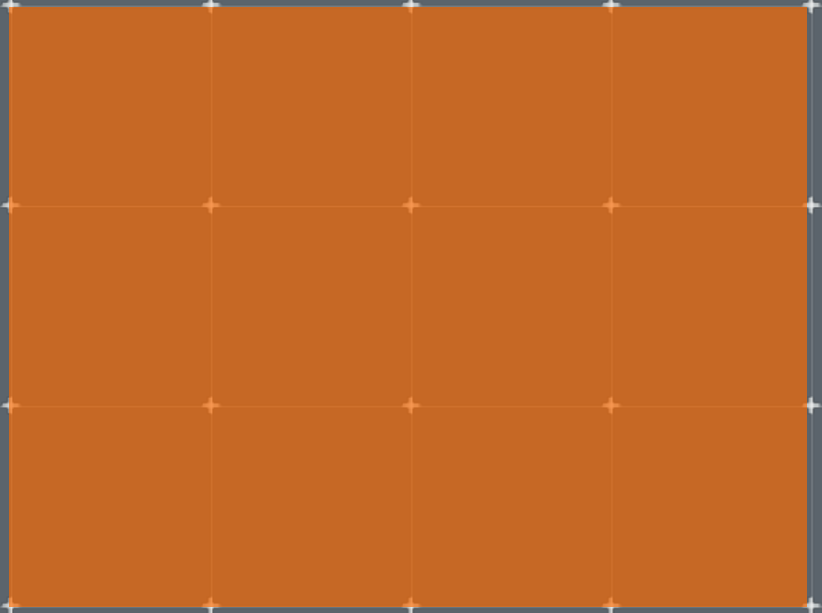
Dan Ros – Principal Architect

Marcus Chang – Staff Software Engineer

Leo Zhou – Software Engineer

Mbed Cloud Update – Product Overview





arm

The need for Update

IoT Devices can be around for a long time

IoT devices life time may be very long: 10-20 years

During that time they can be exposed to many security threats

Replacement is expensive!

Feature rollout enhances device value

"150,000 IoT Devices Behind 1Tbps DDoS attack on OVH"

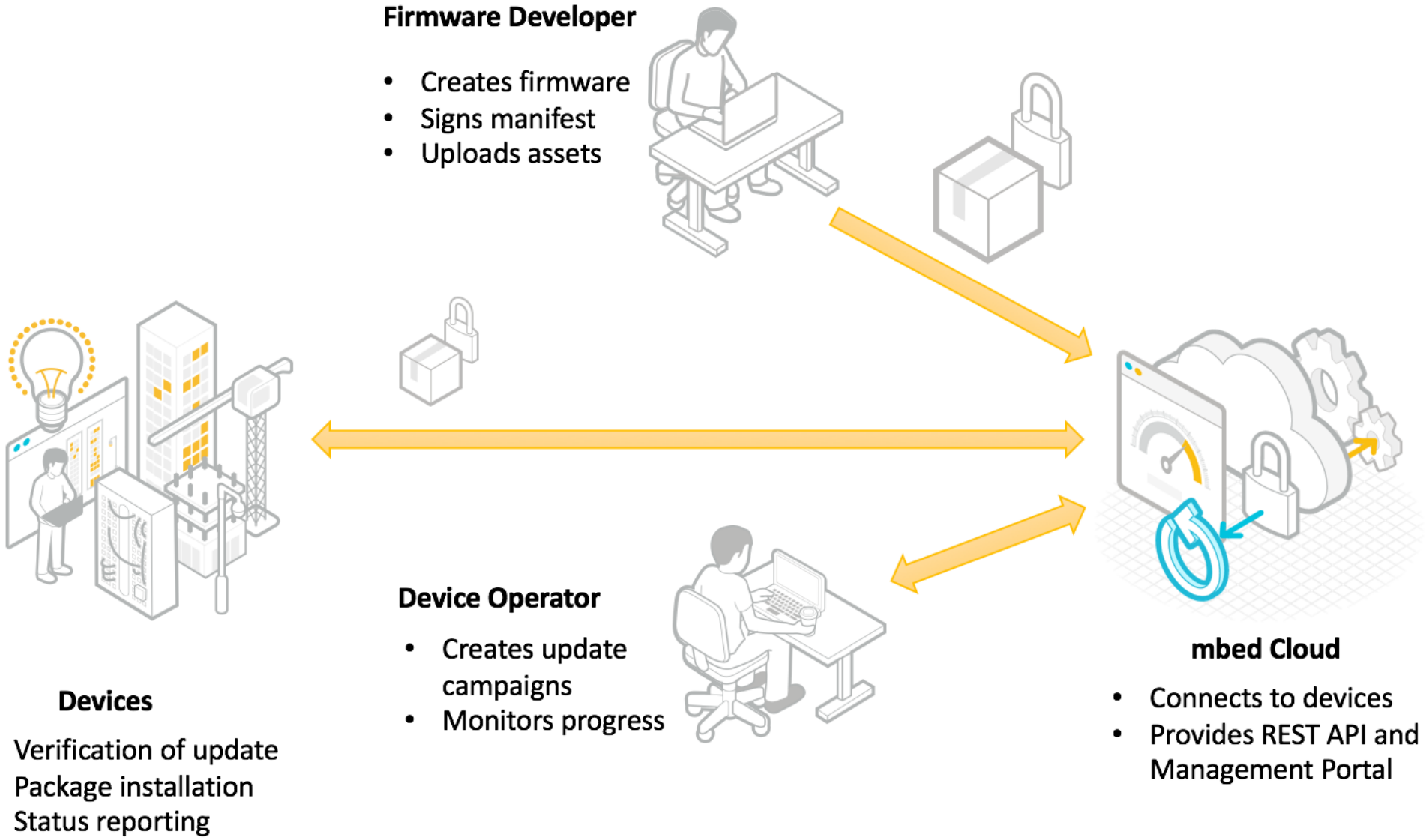
"12-Year-Old SSH Bug Exposes More than 2 Million IoT Devices"

"Hacked Cameras, DVRs Powered Todays Massive Internet Outage"

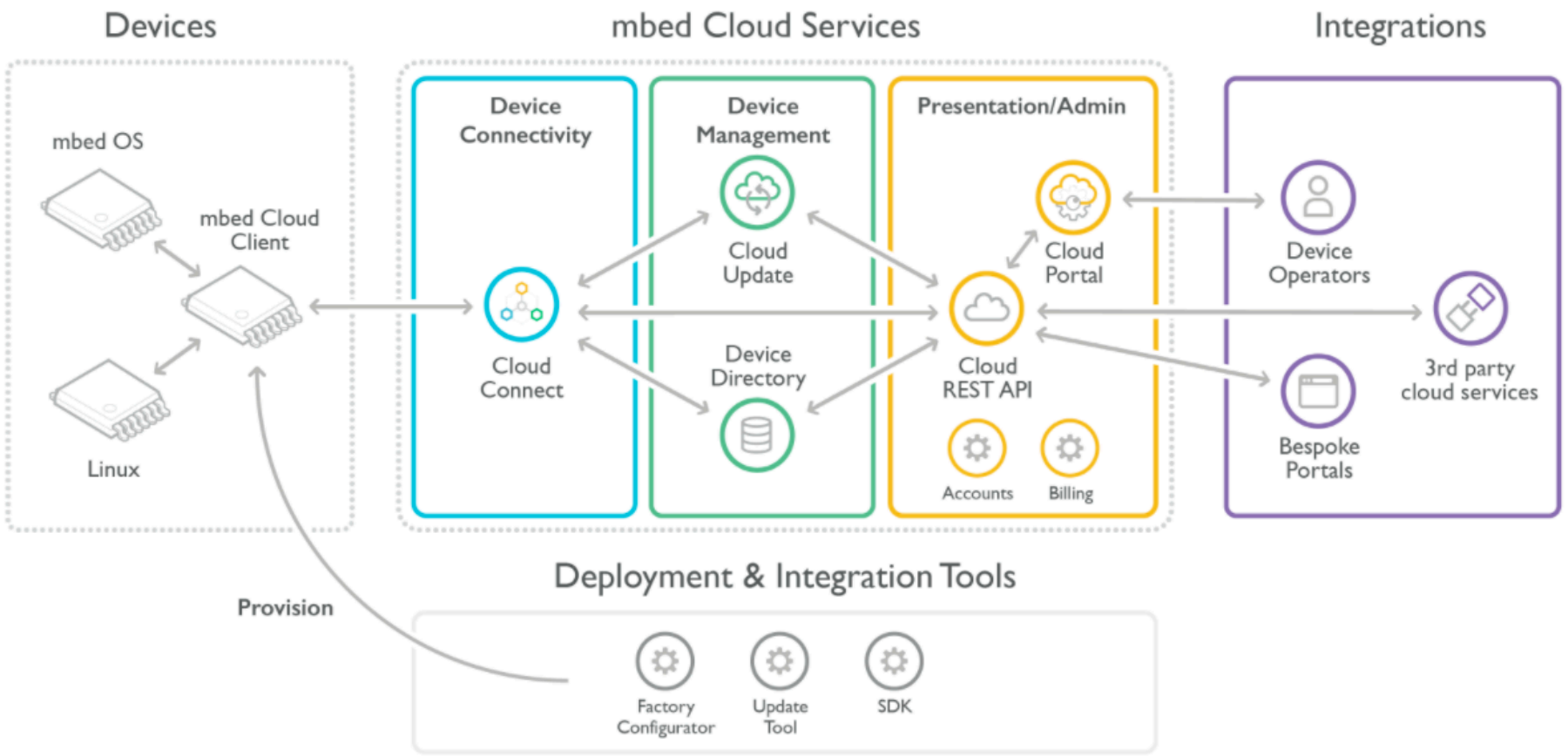
"Software Bug Could Affect Over 400,000 IoT devices"

arm

Introduction to Mbed Cloud Update



Update in Context



Feature Overview

Secure& Reliable with failsafe operation

- Reduce costs for rolling out updates

Flexible Workflow

- Simple but powerful APIs to manage and monitor devices
- Monitor theprogress of updates and status on all devices

Conditional Updates

- Support complex deployments andcritical device operations

Standards Support

- Efficient and Flexible delivery
- Network Efficient
- Broadcast and Mesh friendly

Security

Firmware update security is independent from transport protocol

- Delivery Network treated as untrustworthy
- Enables untrusted caches over unencrypted protocols

Update metadata is signed for authenticity and integrity

- Metadata is captured in a document called a Manifest

Rollback protection

- Prevents installation of potentially insecure/incompatible images
- Secure mechanism for authorised downgrade

Reliability

Device checks the update before download

- Checks for correct manufacturer/model, revision or other parameters as needed before download
- Checks authenticity of manifest

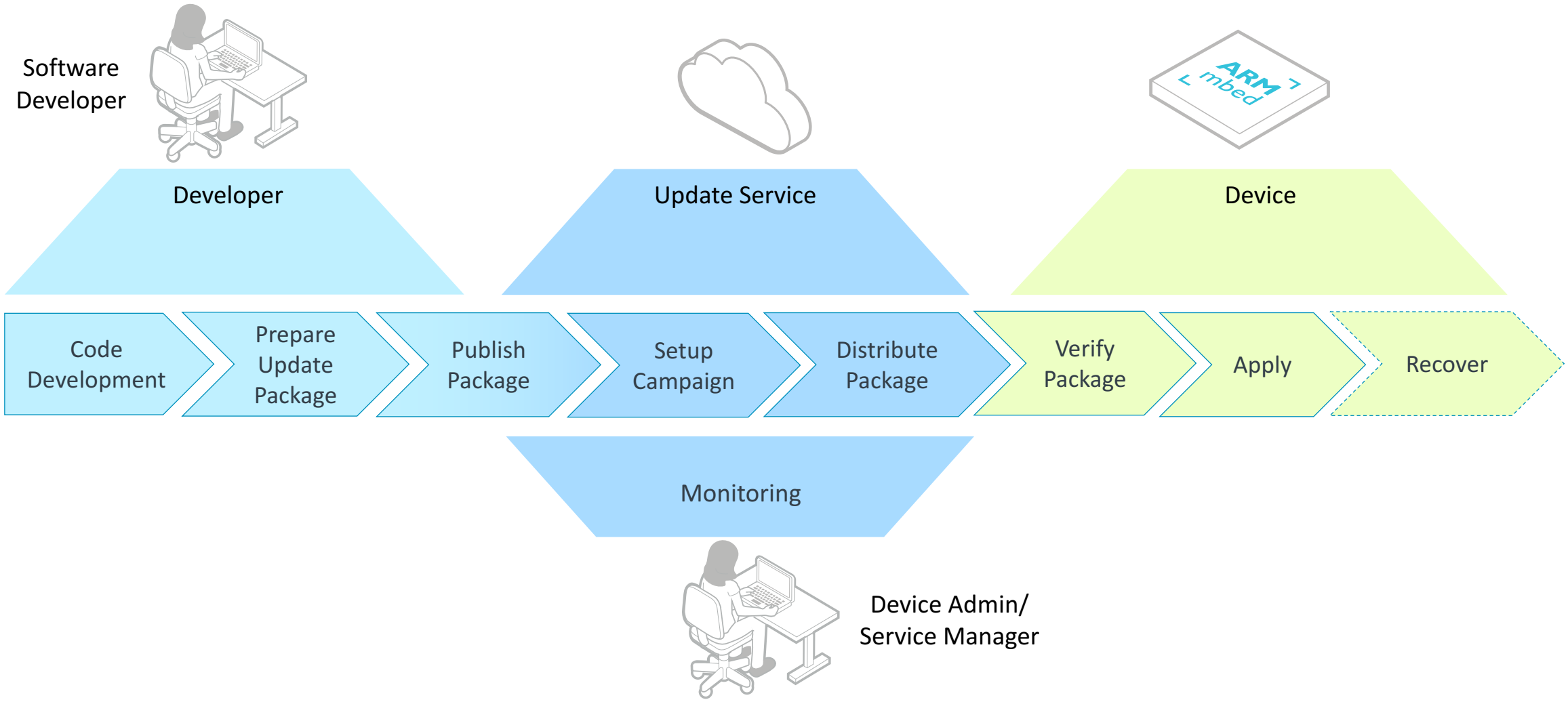
Device checks the update before application

- Integrity check of image before application
- Final client application check to delay application during critical device operations

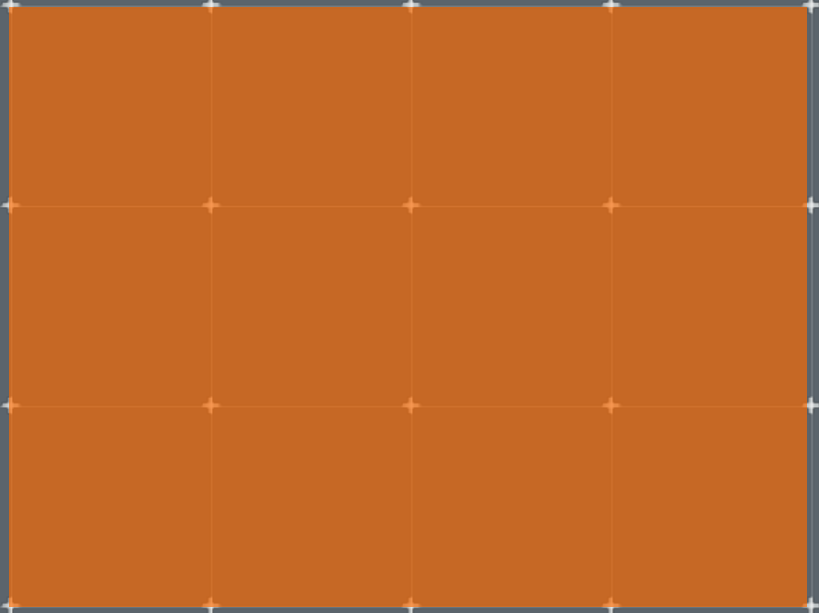
Failure Recovery

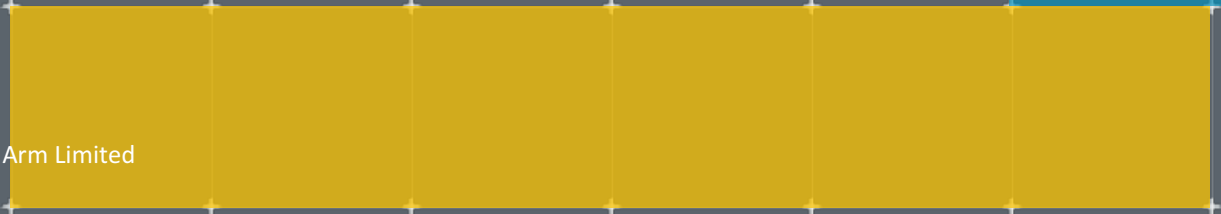
- Multiple points to detect errors and allow recovery
- Power failure at any time during the update will not “brick” the device*
- Checks for successful application

Update Workflow



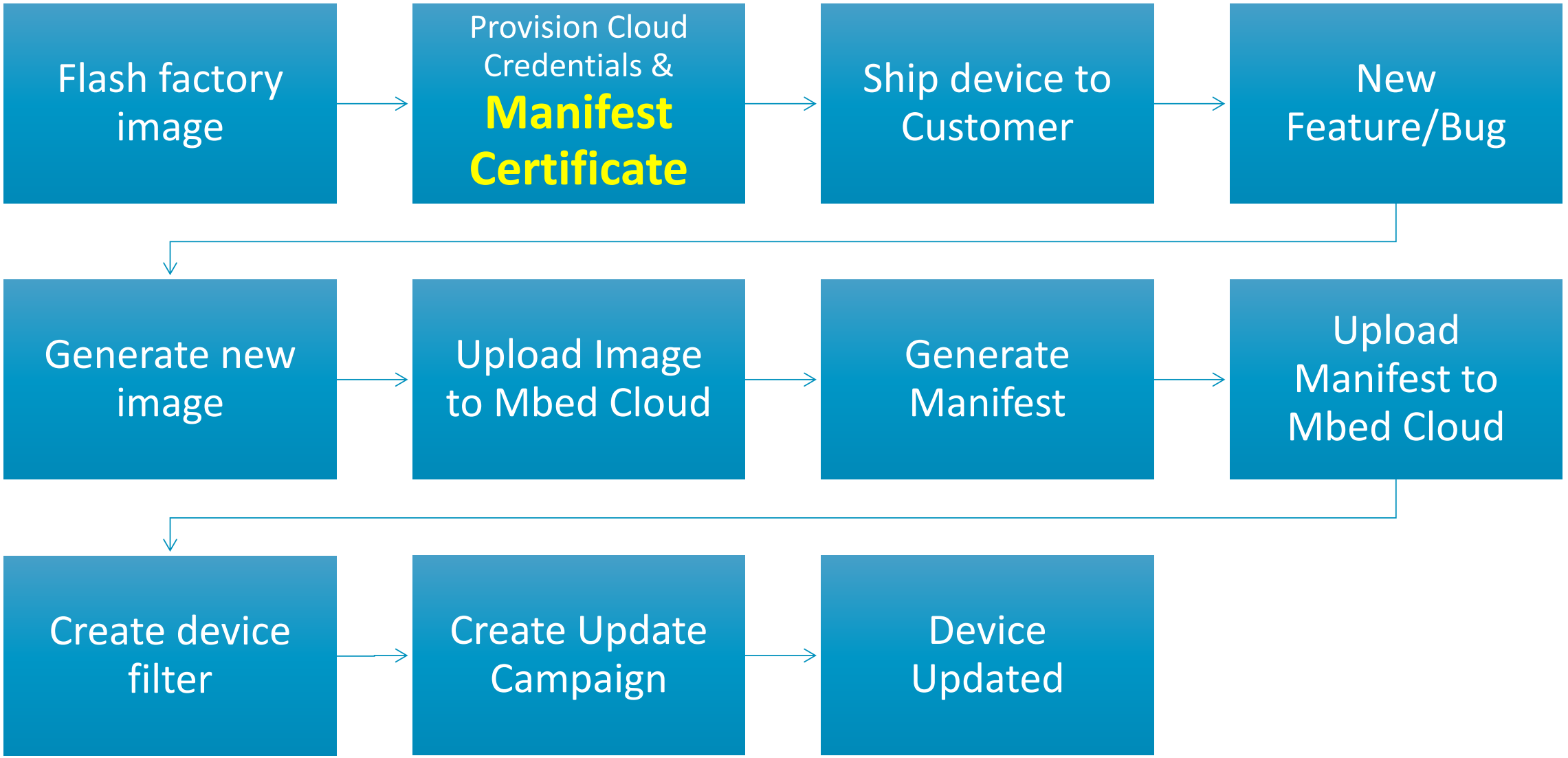
Firmware Update





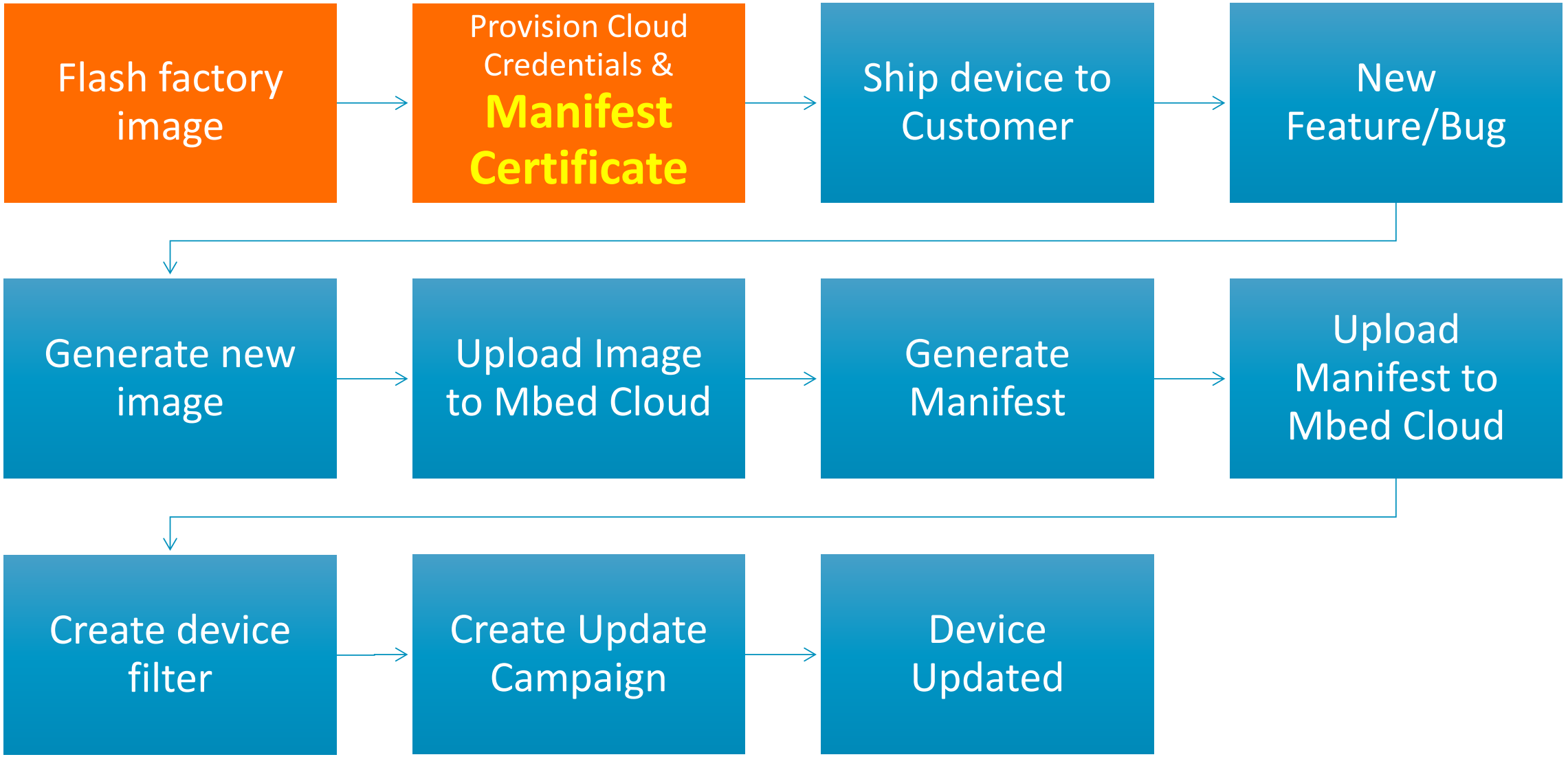
arm

Update life cycle



arm

Update life cycle



arm

Generate update certificate and private key

```
> manifest-tool init \
    -d "<company domain name>"
    -m "<product model identifier>"
```

1. update_default_resources.c
2. .update-certificates/
   - default.der
   - default.key.pem
3. .manifest_tool.json

arm

mbed cloud developer certificate

- Can be downloaded from portal.

Factory Image

Original Flash Layout

Address	Size	Name
0x00000000-0x00100000		0:SBL1
0x00100000-0x00200000		0:MIBIB
0x00200000-0x00300000		0:BOOTCONFIG
0x00300000-0x00400000		0:QSEE
0x00400000-0x00500000		0:QSEE_ALT
0x00500000-0x00580000		0:CDT
0x00580000-0x00600000		0:CDT_ALT
0x00600000-0x00680000		0:DDRPARAMS
0x00680000-0x00700000		0:APPSBLENV
0x00700000-0x00900000		0:APPSBL
0x00900000-0x00b00000		0:APPSBL_ALT
0x00b00000-0x00b80000		0:ART
0x00b80000-0x04c80000	65 MB	rootfs
0x04c80000-0x08000000	51MB	empty

arm

Factory Image

Change to Flash Layout

Original Layout

Address	Size	Name
0x00b80000-0x04c80000	65 MiB	rootfs
0x04c80000-0x08000000	51 MiB	empty

New Layout

Address	Size	Name
0x00B80000-0x00C00000	512 KiB	bootflags
0x00C00000-0x03C00000	48 MiB	root_a
0x03C00000-0x04400000	8 MiB	empty
0x04400000-0x07400000	48 MiB	root_b
0x07400000-0x07C00000	8 MiB	empty
0x07C00000-0x08000000	4 MiB	KCM

arm

Factory Image

Scripts

arm_initial_pack.sh

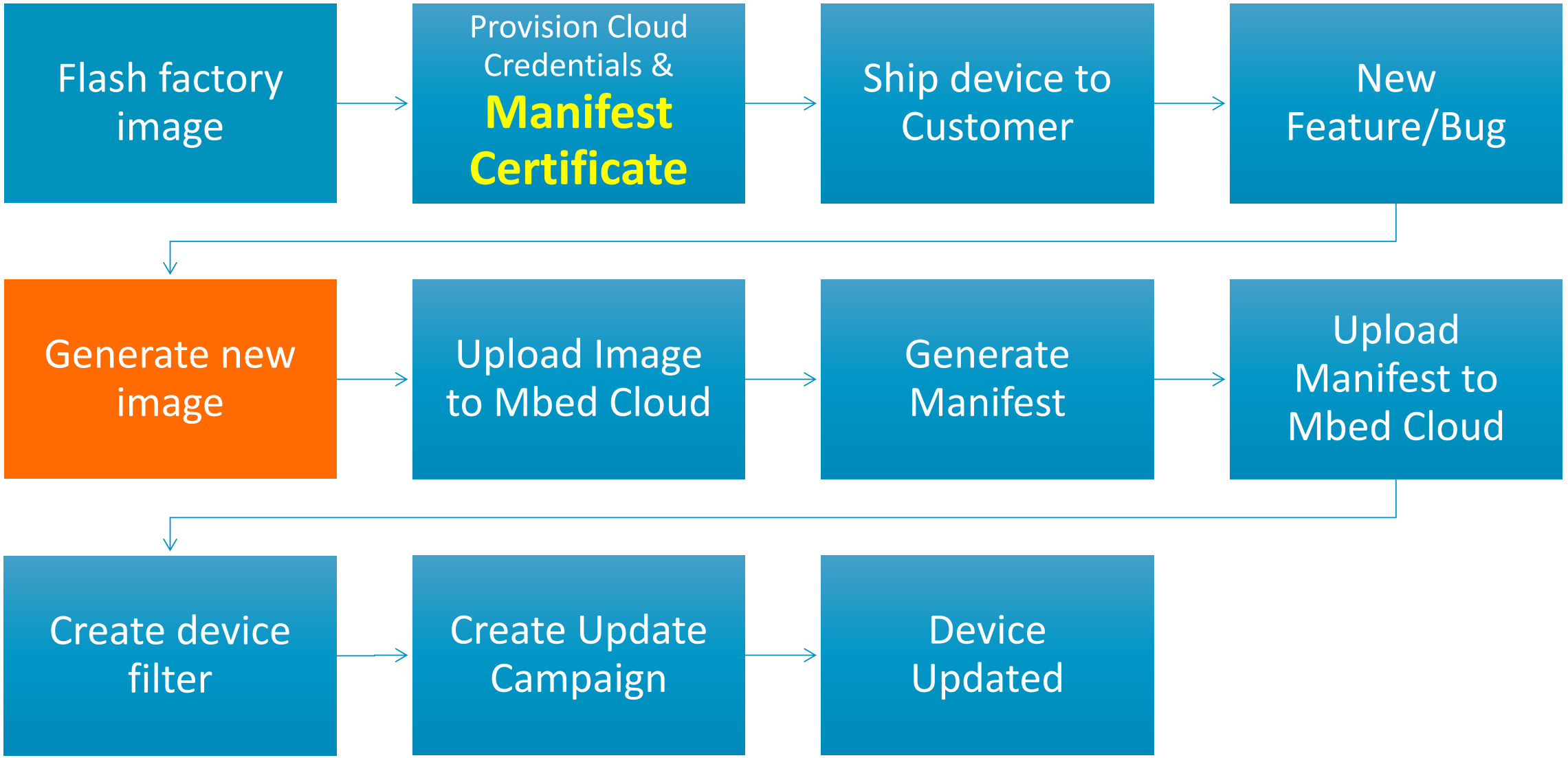
Just for dev, Don't use for production.

- Run **arm_initial_pack.sh** after sdk is built. It creates a image you can load onto your existing board to create all the required partitions.
- Load the image using `tftpboot`
- This image will run and reformat the flash to the desired layout

```
uboot> set ipaddr 192.168.1.1
uboot> set serverip 192.168.1.100
uboot> tftpboot nand-ipq4xx-single.img
```


arm

Update life cycle



arm

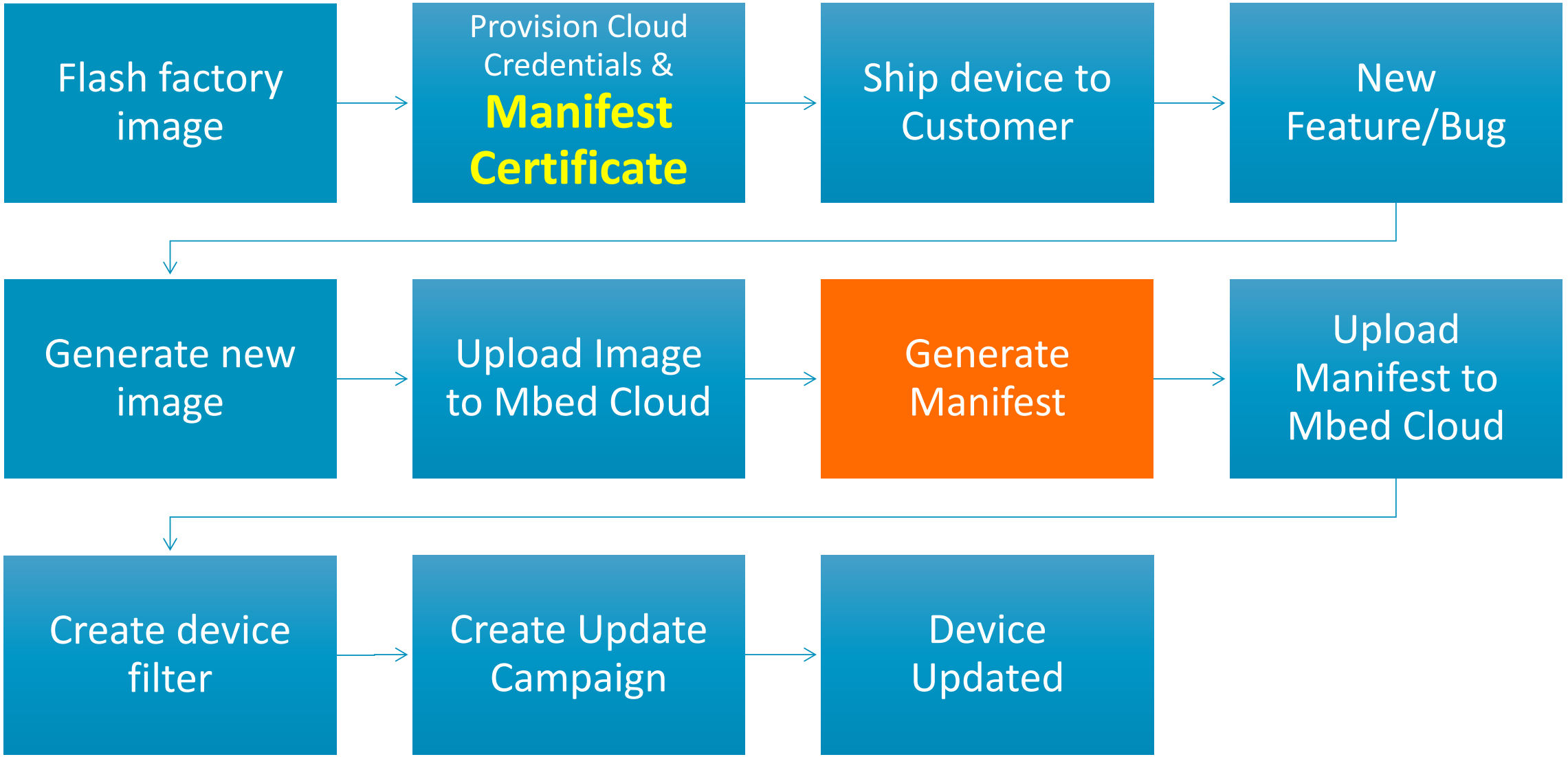
Create Update Image

Scripts

The build system creates a number of images. You will need to find the one that is only the root file system:

```
 > -rwxrwxr-x 1 adv adv 20578304 Sep 10 00:48
/home/adv/work/private/qca-networking-2016-spf-2-
0_qca_oem_standard.git/IPQ4019.ILQ.1.1.1.r2/common/build/ip
q/openwrt-ipq806x-ipq40xx-ubi-root.img
```

Update life cycle



arm

Generate new manifest

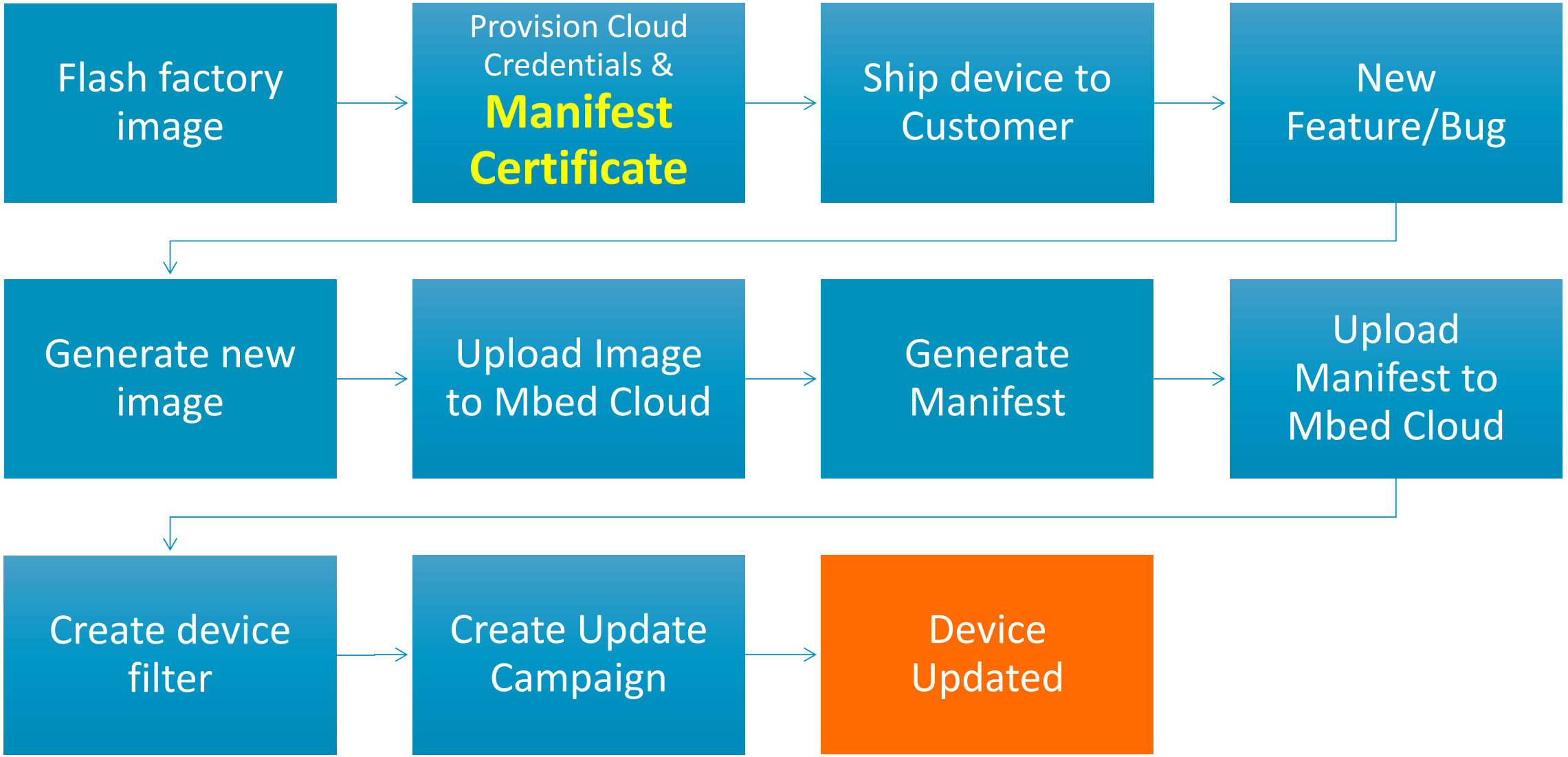
```
> manifest-tool create \
    --manifest-version 1 \
    --private-key      .update-certificates/default.key.pem \
    --input-file       input.json
    --output-file      output.manifest
```


arm

Generate new manifest

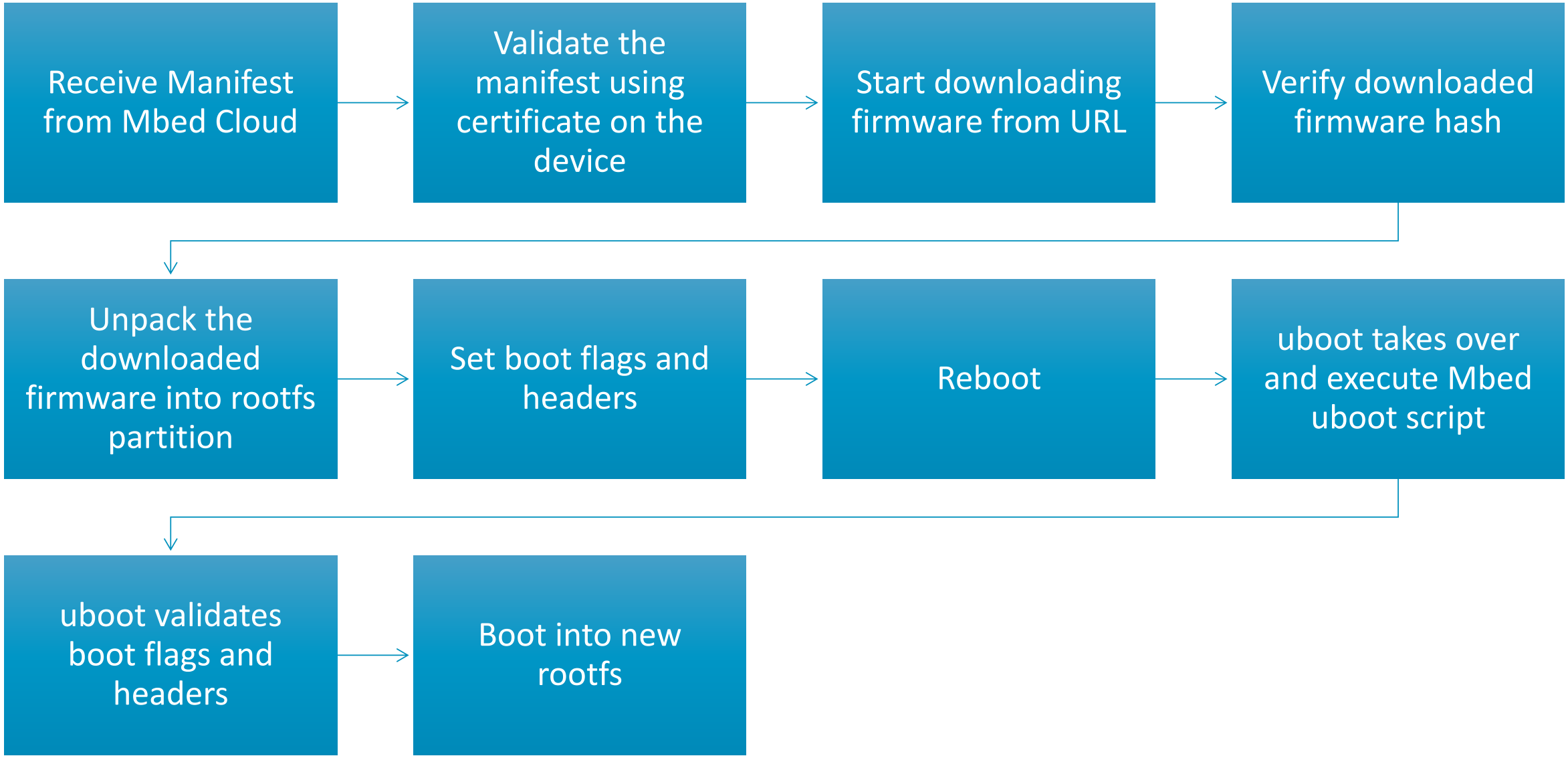
```
> cat input.json
{
    "encryptionMode" : "none-ecc-secp256r1-sha256",
    "vendorId"       : "<128-bit GUID in HEX>",
    "classId"        : "<128-bit GUID in HEX>",
    "payloadUri"     : "http://path.to/payload.bin",
    "payloadFile"    : "/path/to/payload.bin",
    "description"    : "Description of the update",
    "certificates"   : [
        {
            "uri"  : "http://path.to/certificate.der",
            "file" : "/path/to/certificate.der”
        }
    ]
}
```

Update life cycle



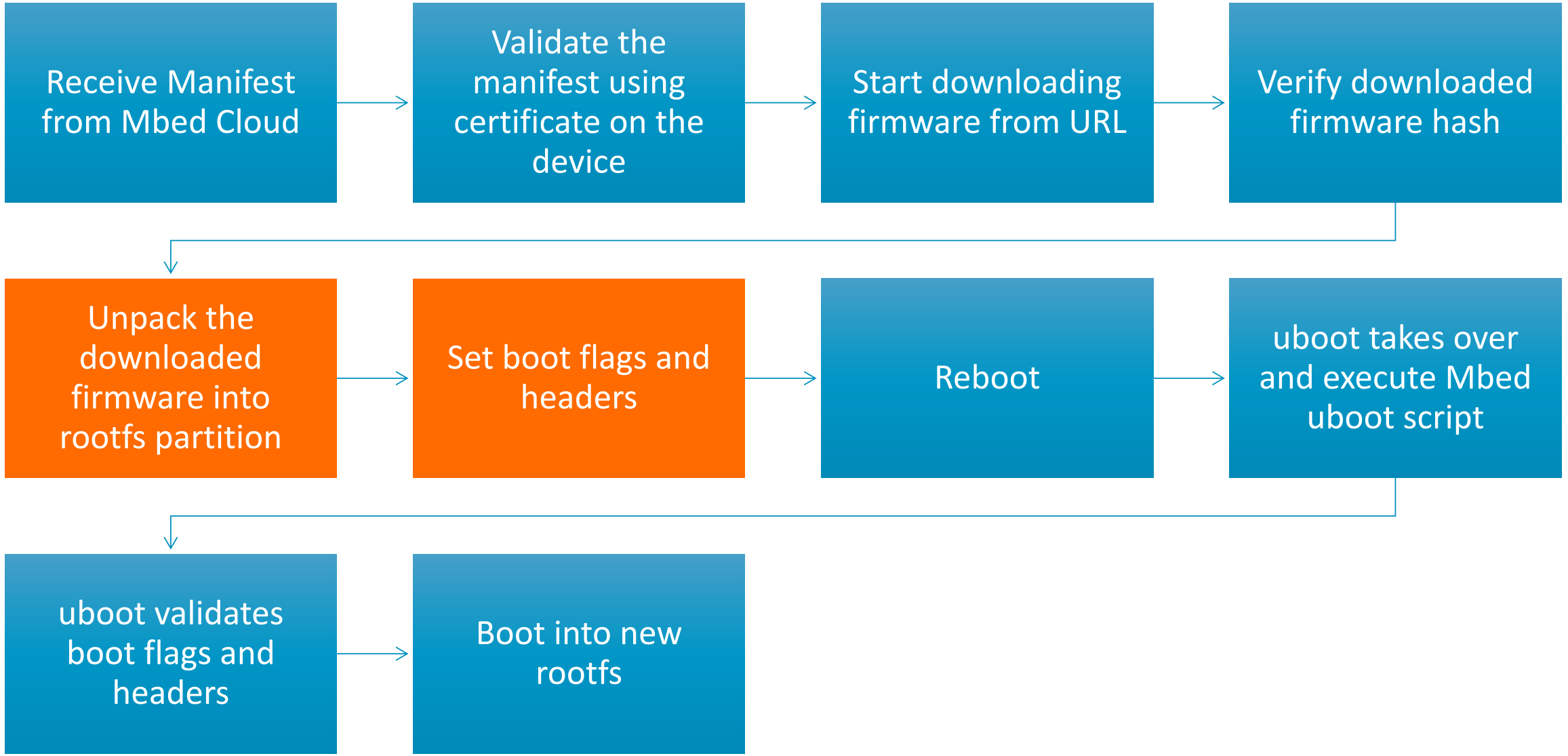
arm

What happens on the device



arm

What happens on the device



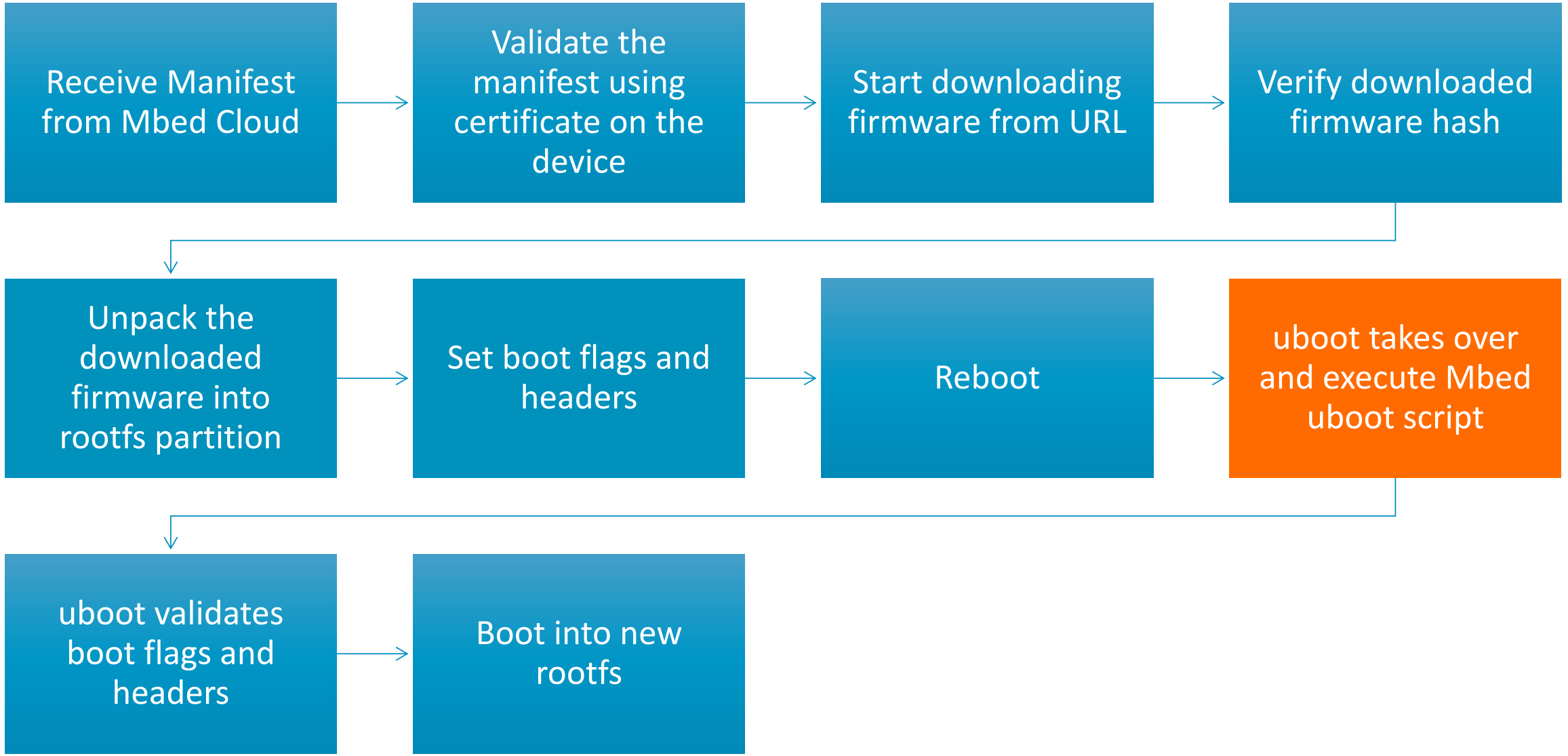
arm

The activation script

activate_script.sh

1. Copy firmware to partition
    1. Copy from the download location /tmp
    2. ubiformat /dev/${UPDATE_SLOT_MTD} -f $FIRMWARE –yes
2. Set boot flags and headers
    1. Write a header into flash
    2. Delete old header

What happens on the device



arm

Boot Process

The Usual Boot process on 3610

1. First stage bootloader is loaded from SBL1 partition which sets up the hardware (RAM, TrustZone, etc.).
2. bootloader reads the partition table stored in the MIBIB partition and sets up the hardware before passing control over to U-Boot in the APPSBL partition.
3. U-Boot runs custom **loadipq** command which deduces what memory is available and loads the kernel into RAM

Mbed boot process

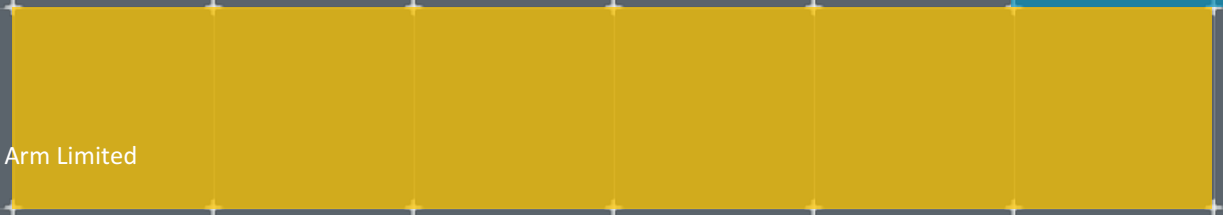
1. Modifies behaviour of uboot
2. Read header from flash
3. Validate the integrity of header
4. Boot into the partition corresponding to the header
5. Link

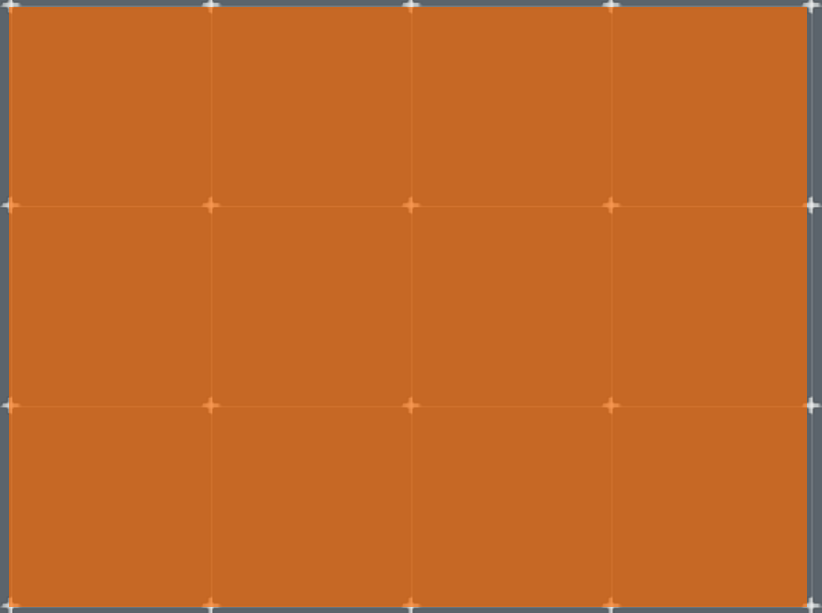
arm

Notes for advantech implementation

1. Mbed boot script does not do the same thing as loadipq. You need to make sure there is no side effects.
2. Secure boot is not enabled. If you require secure boot, you can add the related commands in the uboot script.

Real-time demo/hands-on



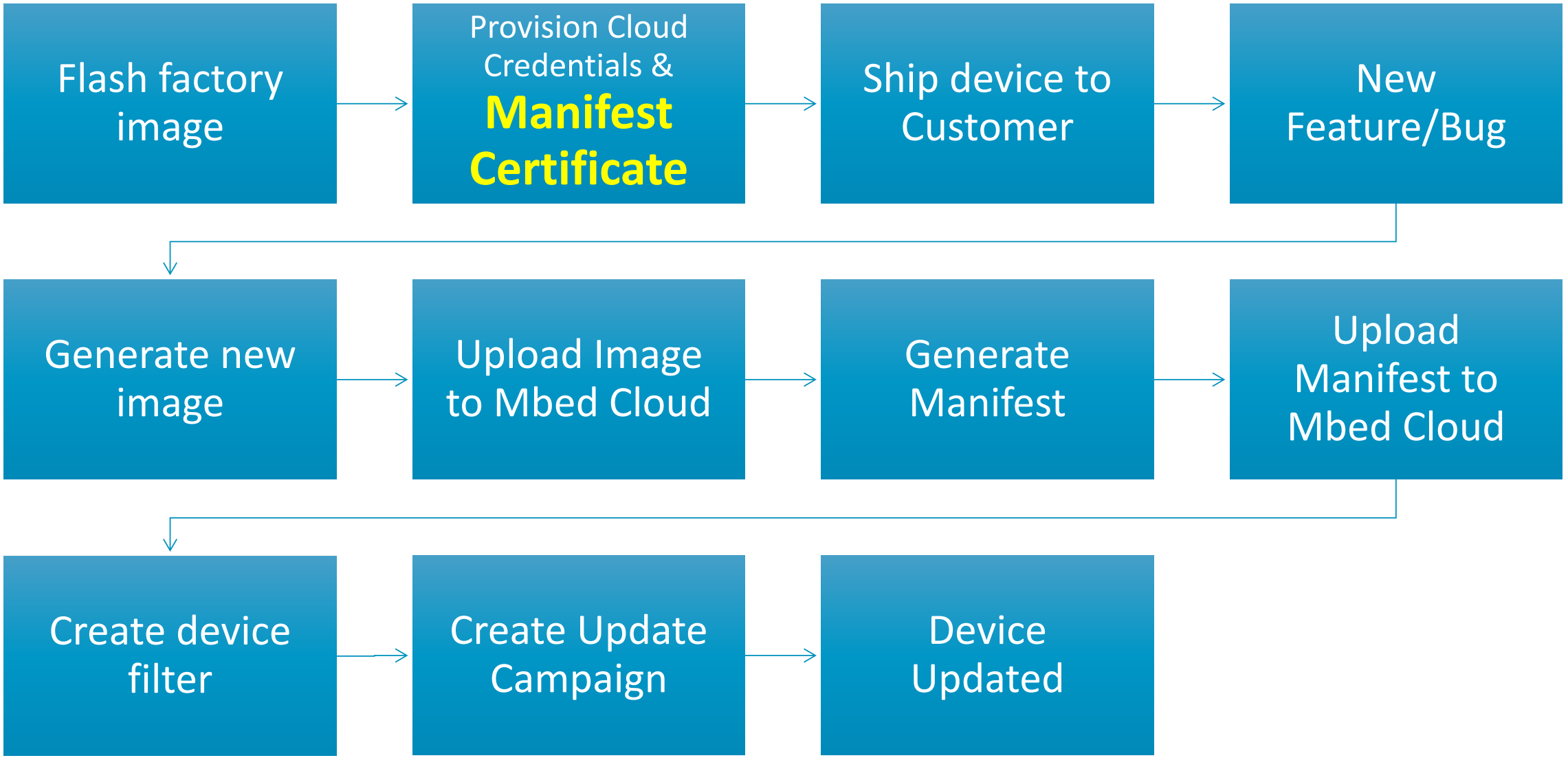


arm

Important Links

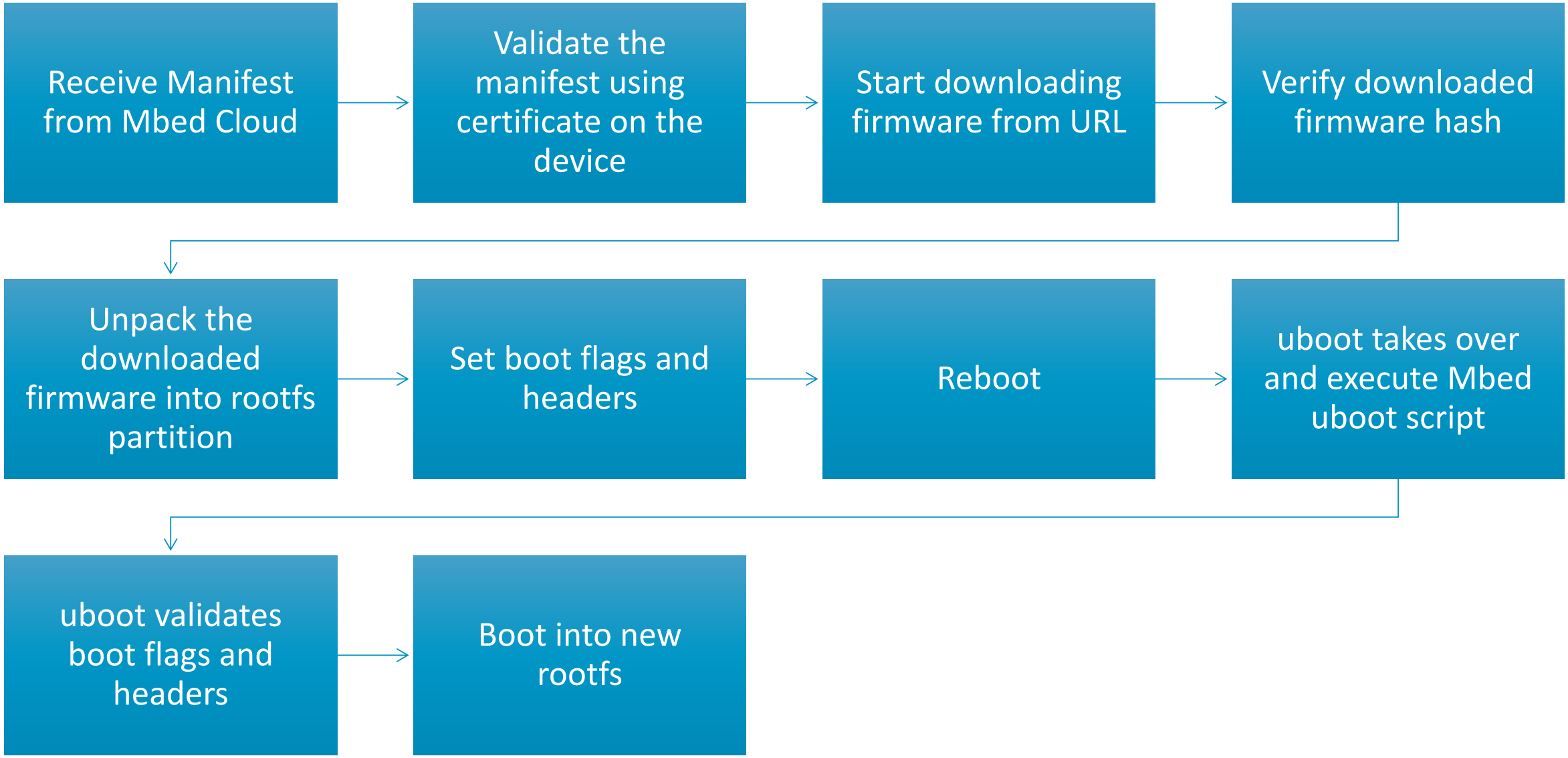
1. Cloud client: https://github.com/ARMmbed/mbed-cloud-client-example-wise-3610-confidential
2. Manifest-tool: https://github.com/ARMmbed/manifest-tool-restricted

Update life cycle



arm

What happens on the device



arm

Thank You!
Danke!
Merci!
谢谢!
ありがとう!
Gracias!
Kiitos!
감사합니다
धन्यवाद