D-RTM on non-x86 architectures like ARM and POWER9

TPM.dev 2021 Conference

Piotr Król

3MDEB

whoami

Piotr Król
3mdeb Founder

- OSF and OSHW promotor interested in Trusted Computing
- Conference speaker and organizer
- TrenchBoot Steering Committee Core Member

- 12yrs in business
- 6yrs in Open Source Firmware and Trusted Computing integration
- C-level positions in

Who we are ?

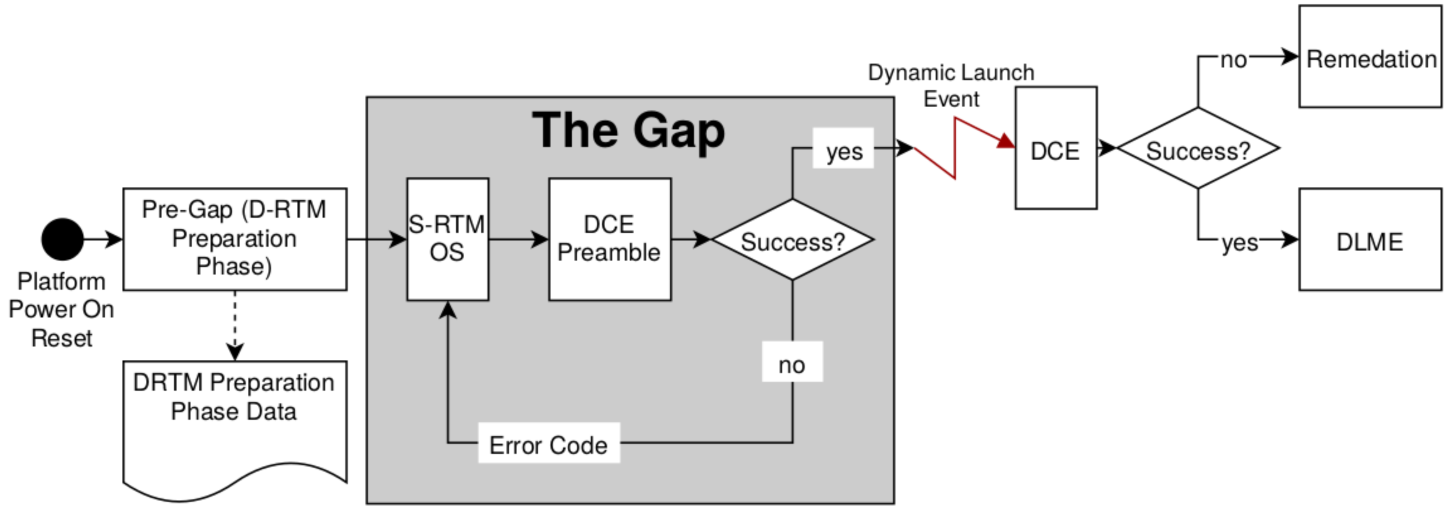
- coreboot licensed service providers since 2016 and leadership participants
- UEFI Adopters since 2018
- Yocto Participants and Embedded Linux experts since 2019
- Official consultants for Linux Foundation fwupd/LVFS project
- IBM OpenPOWER Foundation members

Kudos

- Krystian Hebel, Firmware Engineer @ 3mdeb, as main contributor to POWER9 status
- Stuart Yoder, System Architect @ ARM, for ARM section review

Agenda

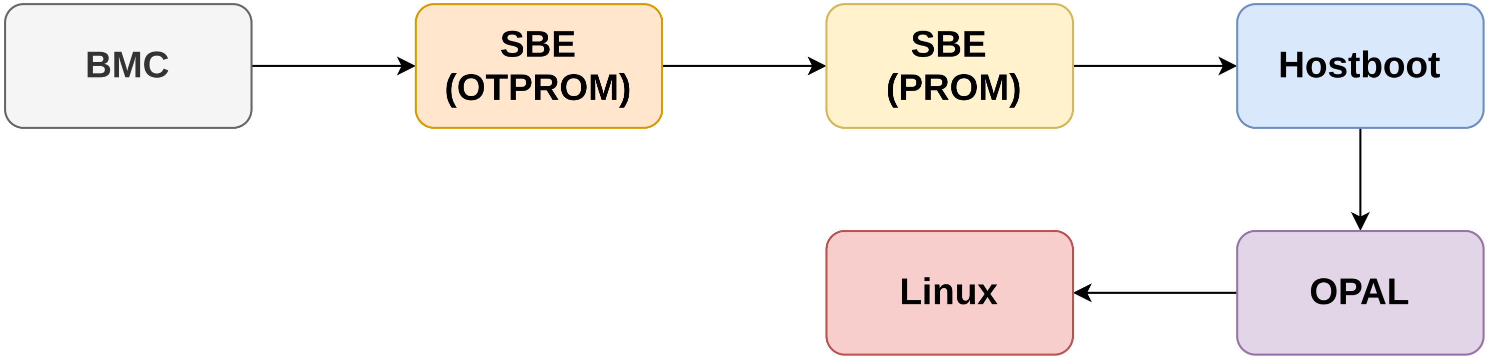
D-RTM



- DRTM start when Dynamic Launch event call executes
- DL Event controls PCRs 17-22, those are initialized with value -1
- DL Event change PCRs value to 0 and immediately extends with DCE hash
- Any attempt to reset TPM will set PCR[17] to -1 (TPM reset attack immunity)

TCG D-RTM Architecture v1.0.0

POWER9 boot process intro

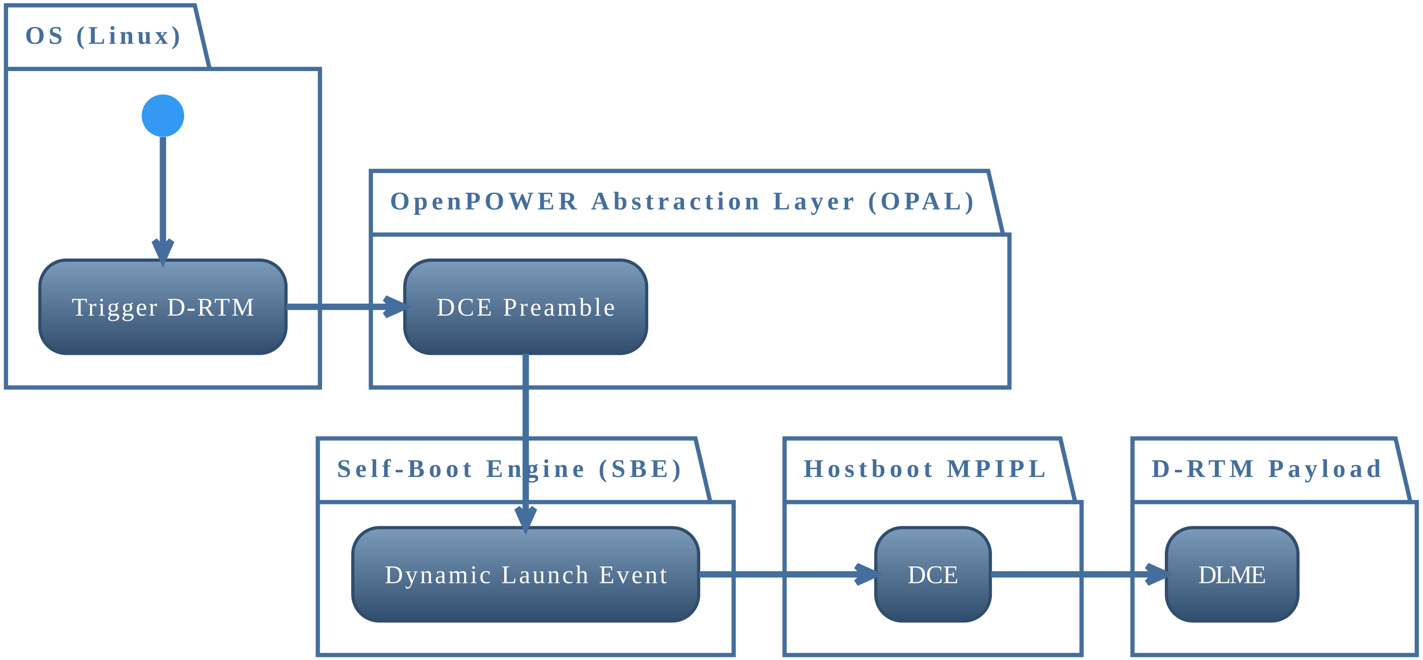


- **BMC** (Board Management Controller)
- **SBE** (Self-Boot Engine) - firmware and hardware (dedicated built-in small CPU), which configures and starts main CPU
- **Hostboot** - main CPU boot firmware (processor, bus and memory initialization)
- **OPAL** (OpenPOWER Abstraction Layer) - skiboot (runtime services) and skiroot (Linux kernel and initramfs, essentially part of bootloader, but included in OPAL image)

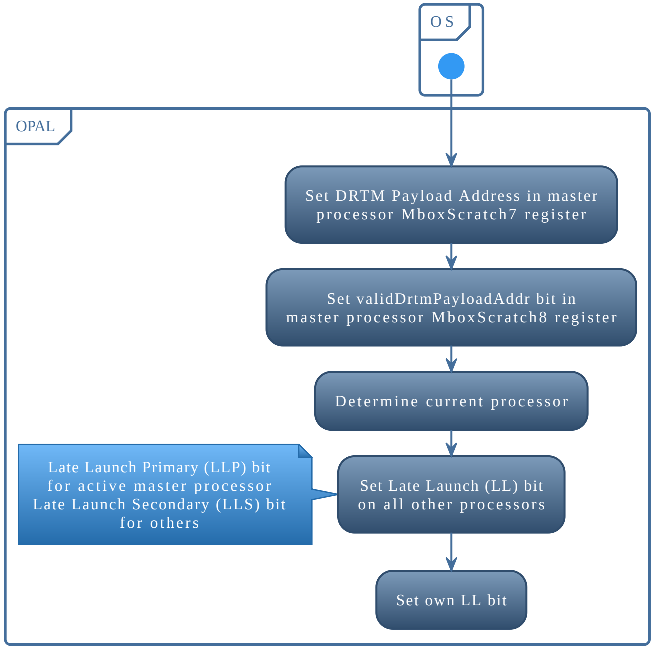
D-RTM status in POWER9

- There is only partial implementation of D-RTM support in OpenPOWER firmware, because of that D-RTM is not fully functional
- D-RTM on POWER9 is implemented in SBE and Hostboot firmware
    - this is different to x86 where special CPU instruction was introduced for such purpose
    - TPM locality 4 is enabled by SBE, code running on host CPU doesn't have the ability to do so. This is one of the requirements defined by TCG.
- SBE support was introduced in 2016, by following patches by Shakeeb Pasha
    - MPIPL Start Chipops and Mpipl istep implementation
    - Continue MPIPL implemntation
- Hostboot support was introduced in 2017, by following patches by Nick Bofferdinig
    - Support DRTM RIT protection
    - Add TPM device driver support for DRTM PCR reset sequence

Overview of D-RTM on POWER9

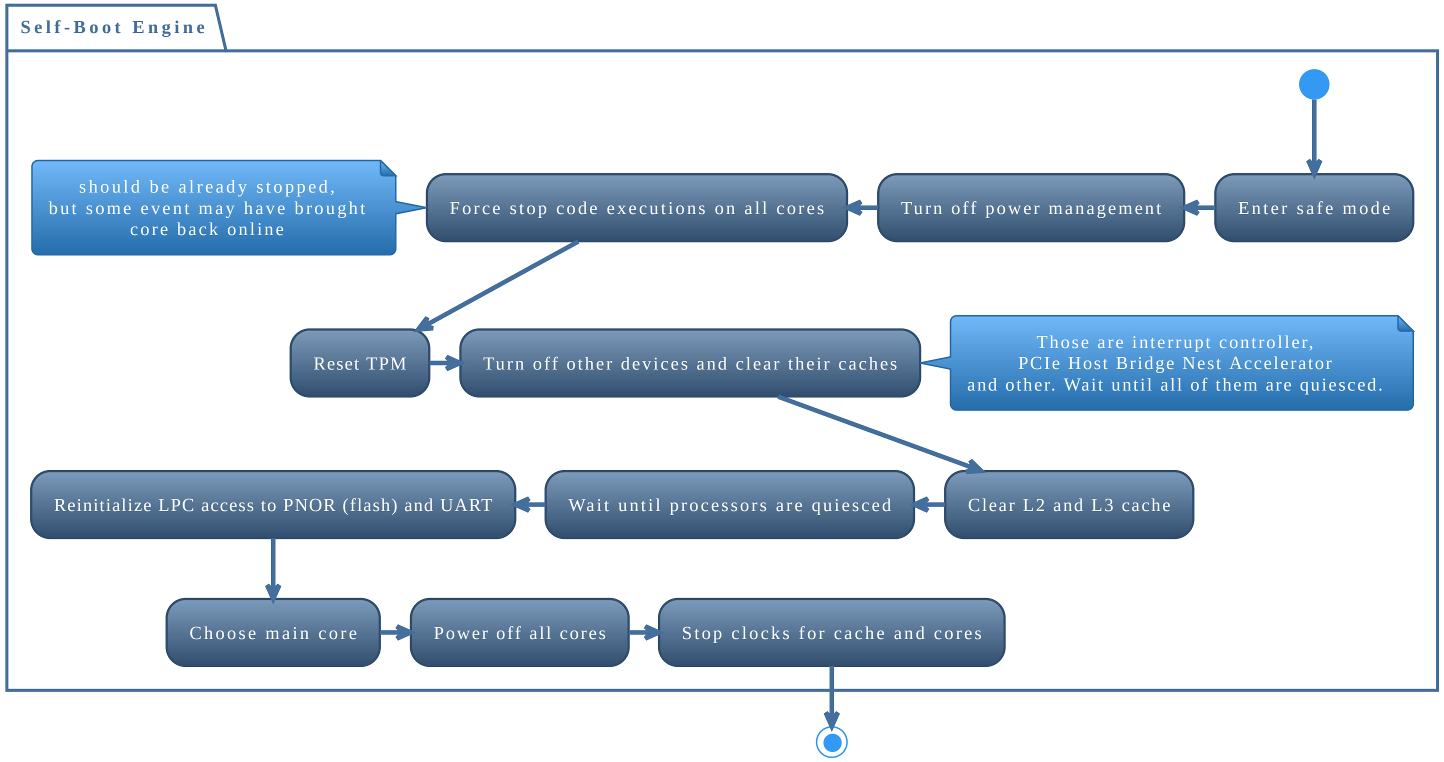


Triggering DRTM


OS
OPAL
Set DRTM Payload Address in master processor MboxScratch7 register
Set validDrtmPayloadAddr bit in master processor MboxScratch8 register
Determine current processor
Late Launch Primary (LLP) bit for active master processor Late Launch Secondary (LLS) bit for others
Set Late Launch (LL) bit on all other processors
Set own LL bit

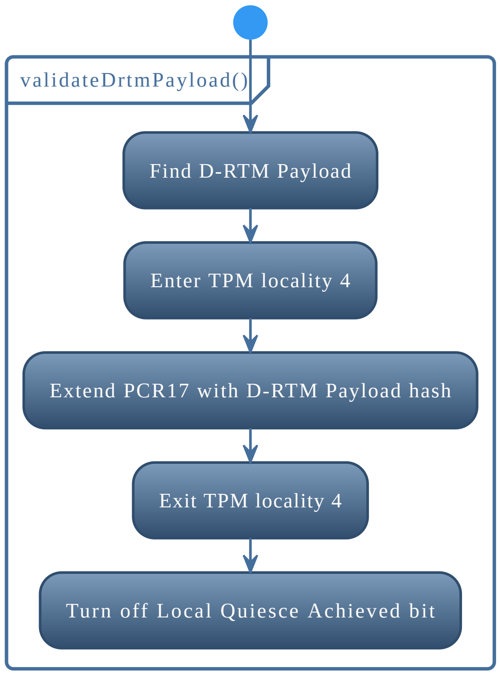
- Above diagram show how hypothetical D-RTM trigger from OS through OPAL could look like
- Documentation and code miss D-RTM Payload format description
- Setting LL cause SBE interrupt, which handles further processing

SBE side of things


Self-Boot Engine
Enter safe mode
Turn off power management
Force stop code executions on all cores
should be already stopped, but some event may have brought core back online
Reset TPM
Turn off other devices and clear their caches
Those are interrupt controller, PCIe Host Bridge Nest Accelerator and other. Wait until all of them are quiesced.
Clear L2 and L3 cache
Wait until processors are quiesced
Reinitialize LPC access to PNOR (flash) and UART
Choose main core
Power off all cores
Stop clocks for cache and cores

- After all those actions it seem all cores are prepared for waking interrupt
- Most of the system is down - source of waking interrupt is unknown

Back to Hostboot - MPIPL



- MPIPL (Memory Preserving Initial Program Loader) - part of OpenPOWER firmware which executes special boot path skipping memory training, ECC initial pattern writing and executes D-RTM Payload, if it was set.

Back to Hostboot - MPIPL

```
// Extend (arbitrary) measurement to PCR17
SHA512_t hash = {0};
memcpy(hash,DRTM_RIT_PAYLOAD,sizeof(DRTM_RIT_PAYLOAD));
pError = TRUSTEDBOOT::pcrExtend(TRUSTEDBOOT::PCR_DRTM_17,
                                TRUSTEDBOOT::EV_COMPACT_HASH,
                                hash,
                                sizeof(SHA512_t),
                                DRTM_RIT_LOG_TEXT,
                                sizeof(DRTM_RIT_LOG_TEXT));
```

- In existing implementation DRTM_RIT_PAYLOAD contains "DRTM" string
- PCR17 is extended with "DRTM\0\0\0..." up to the size of SHA512, there is no hash calculation
- SHA1 and SHA256 banks are extended with above "payload" trimmed to the side of appropriate hashes, so only first bytes are used for PCR extending

Back to Hostboot - MPIPL

The best description of current status of DRTM on POWER9:

```
#else

// TODO: RTC 167205: Securely verify the measured launch environment
// TODO: RTC 167205: Really measure+extend the payload

#endif
```

- Obviously existing implementation needs more work and looks more like unti testing then ready to use code.

Other quirks of current implementation

- Documentation says that SBE enables locality 4, but code doing so is nowhere to be found. Maybe done by hardware?
- Hostboot can only extend PCR17, every operation is hardcoded to use this PCR. This may have been done to easy convert SRTM to DRTM by changing code in one place instead of every operation separately, but it effectively blocked possibility of extending PCR18-PCR22, which are also reserved for DRTM.
- TPM DRTM reset is normally done by writing to HASH_START, writing all bytes of DCE one by one to HASH_DATA and finishing it with a write to HASH_END (everything in locality 4). On write to HASH_END TPM sets PCR17 to hash of bytes received through HASH_DATA (for all implemented banks). Hostboot skips HASH_DATA part, so PCR17's initial value is sha256sum /dev/null.
- Hostboot code only supports Nuvoton 65x and 75x Models at this time, I2C only. TPM header has also LPC, but LPC host controller doesn't expose TPM specific cycle types - only IO, memory and firmware memory cycles are available. Due to unavailability of those modules we were unable to test anything in practice.

POWER9 DRTM summary

- Probably the only path that has any chance of success with minimal modifications is:

  Hostboot -> DRTM -> Hostboot MPIPL -> OPAL -> Linux.

  - Obviously calling D-RTM from Hostboot is just a unit test in current code and probably the idea was to make it general mechnism that we can call from OS after making it work from Hostboot

- One of the reasons for using DRTM is lack of trust for firmware, but in this case that same firmware is (mostly) re-run during DRTM anyway.

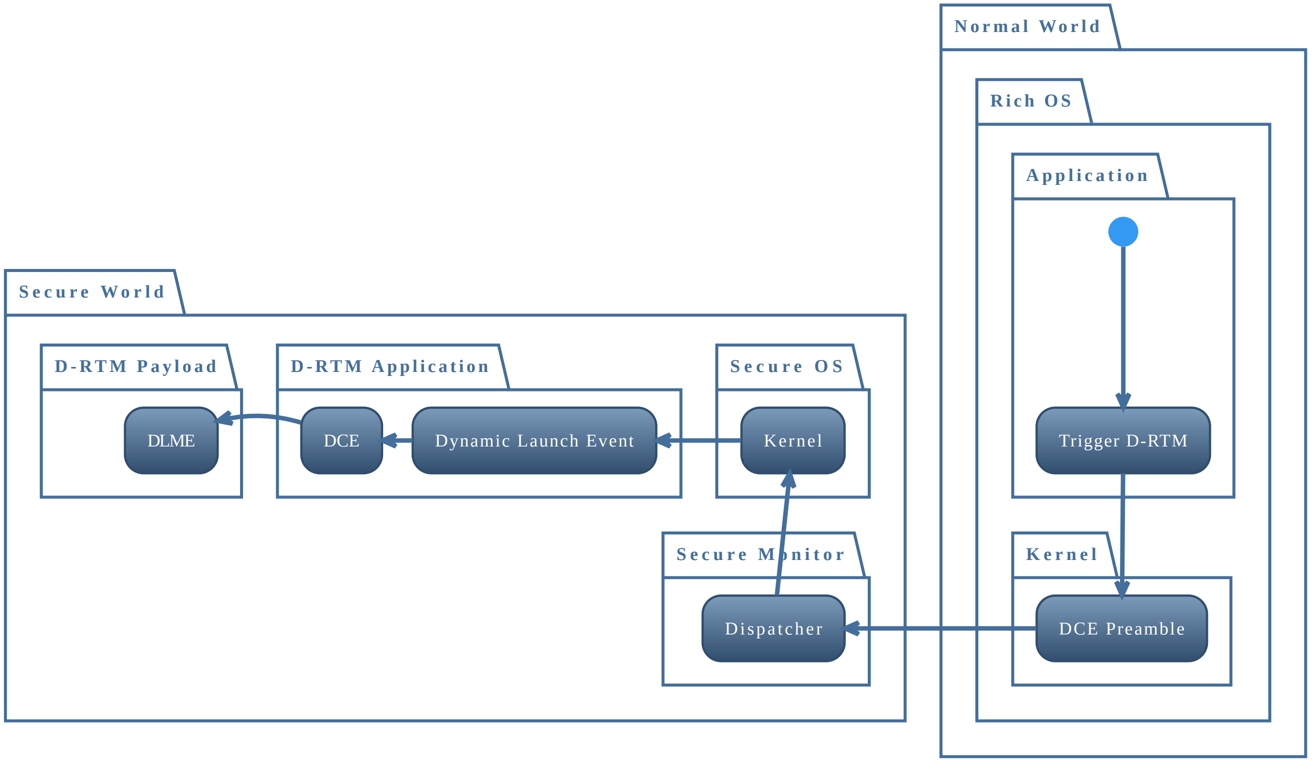
- Hardware components seem to be in place, implementation in software and user-level documentation is lacking.

D-RTM status in ARM

- ARM works on official D-RTM specification
  - publishing is planned in Fall 2021
  - TrenchBoot Steering Committee supported specification review process and provided feedback based on our past experience with Intel and AMD implementation
  - Unfortunately we are under NDA, so we cannot say more than you can find searching Internet
- There are some vendors, which seem to support D-RTM for quite long time (2018?)
  - Qualcomm Snapdragon 850 seem to support System Guard Secure launch
  - There are also some signs of NXP support in Project Mu
- We can find that on Qualcomm DRTM TrustZone application is implemented that supports SMC (Secure Monitor Call) memory protections

https://github.com/TrenchBoot/documentation/blob/master/steering-committee/Community-Meeting-June17-2021.md

Triggering D-RTM in ARM



- Above diagram is based on interpretation of Microsoft description of Qualcomm implementation and is only hypothetical

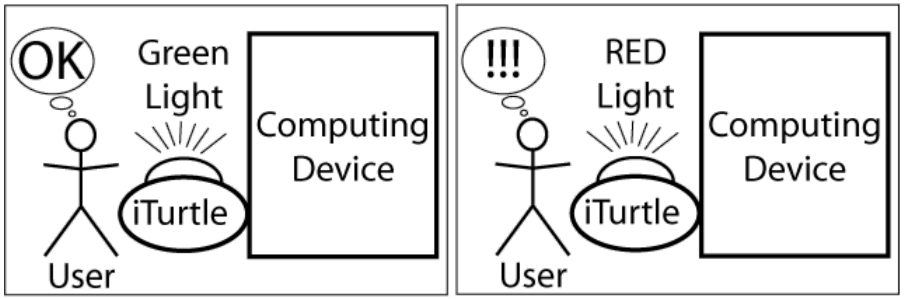
Q&A

TCG TPM related projects

We would like to thank NLNet Foundation for funding following projects

- **Fobnail** - OSS and OSHW USB token for attesting Dynamically Launched Measured Environments (DLME), granted for 3mdeb
- **lpnTPM** - OSS and OSHW Trusted Platform Module, granted for LPN Plant

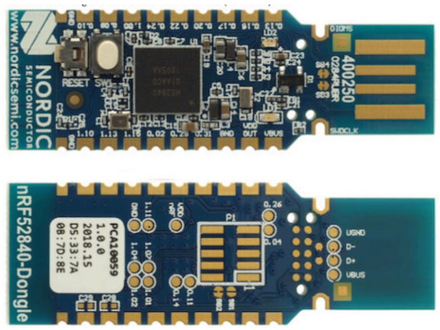
Fobnail



- The Fobnail project aims to provide a reference architecture for building offline integrity measurement servers on the USB device and clients running in Dynamically Launched Measured Environments (DLME)
    - In short it would be attestation server on USB token capable of attesting D-RTM payload
    - this is not new concept in principle, it was already announced by Jonathan McCune (Flicker author) at HotSec07 with the name iTurtle
- Project website: https://fobnail.3mdeb.com

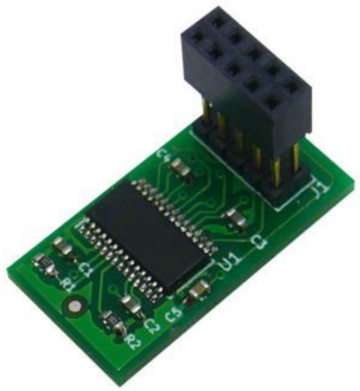
iTurtle: https://www.usenix.org/legacy/event/hotsec07/tech/full_papers/mccune/mccune_html/index.html

Fobnail - hw and sw stack



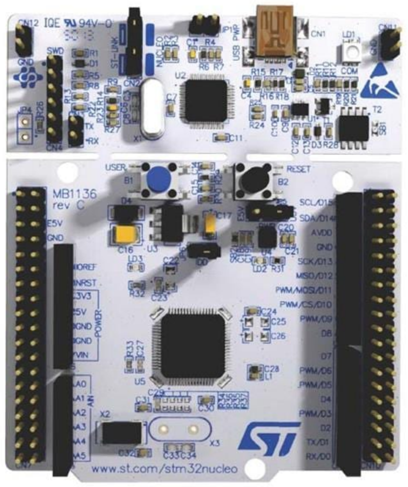
- HW and SW selection is at early stage
- We started development on Nordic nRF52840 development kit
  - we will consider use of recent version of Nitrokey tokens
  - future iterations will consider RISC-V FPGA implementation maybe even from European Processor Initiative
- Initial work around software stack would be based on Zephyr RTOS
  - we also consider Rust: TockOS and/or baremetal Trussed, or even RISC-V phone Precursor microkernel Xous
- We hope to create unified software stack on both Fobnail and DLME side

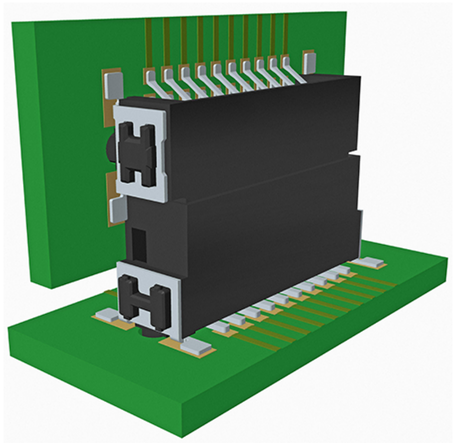
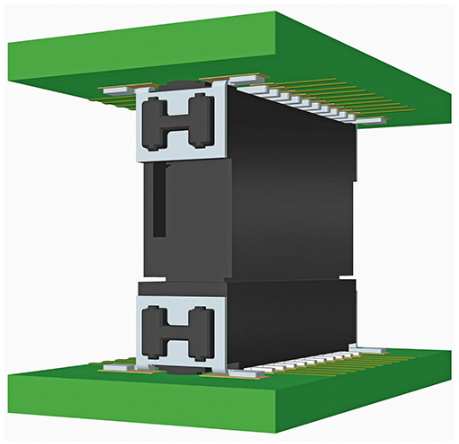
lpnTPM



- Problem statement:
  - there is lack of standardization around TPM modules pinout, connector and geometry
  - TPM firmware is not delivered in trustworthy way (closed source)
- lpnTPM will provide flexible OSHW design that can be adjusted to customer needs as well as OSS implementation based on Microsoft Reference Implementation

lpnTPM - hw and sw stack



- We plan to use Nucleo STML476RG as base for further development, it is already supported by Microsoft reference implementation, although nobody tried that for quite some time
- We are at connector prototyping phase - so above are just samples

Q&A