

Arm DRTM beyond the reference model: dynamic launch on real platforms and Linux

Linux Plumbers Conference 2026, System Boot and Security Microconference

Piotr Król, 3mdeb | Prague, 5 October 2026

Kudos

Upstream we build on

- Ross Philipson, Daniel P. Smith: x86 Secure Launch
- Jason Gunthorpe (NVIDIA): arm64 DRTM v2
- Raghupathy Krishnamurthy (NVIDIA): NV-Kernels DRTM
- Manish Badarkhe, Manish Pandey (Arm): TF-A DRTM

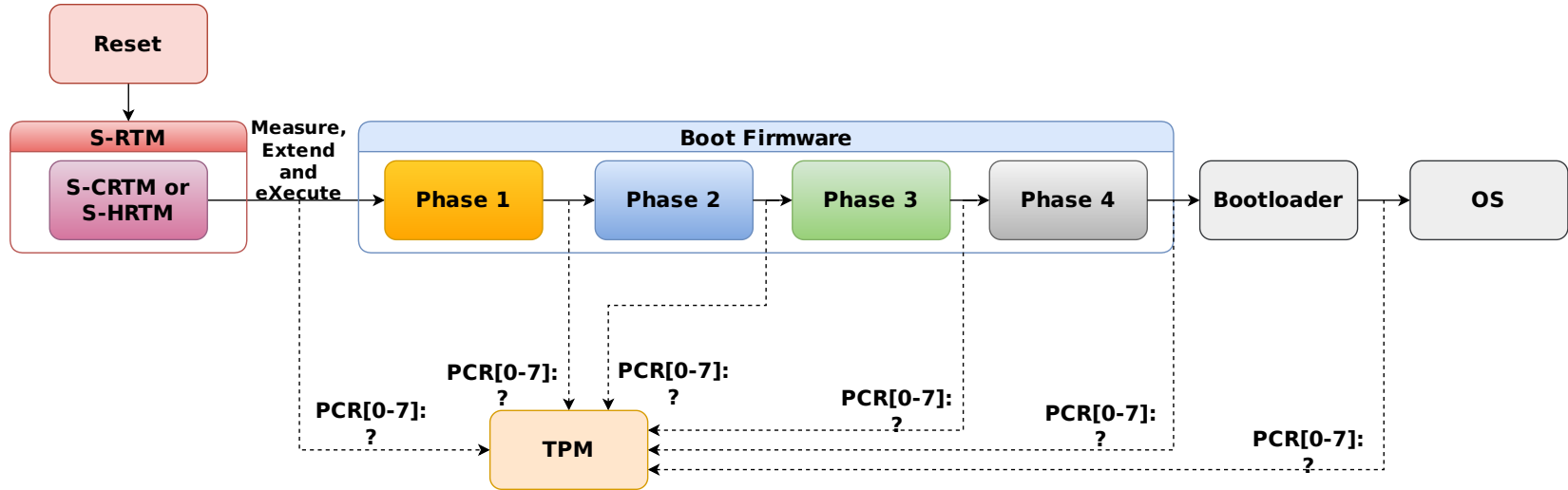
Review

- Stuart Yoder (Arm): TrenchBoot Arm blueprint review; TPM over FF-A
- Daniel P. Smith (Apertus), Richard Persaud (EdgeForm): TrenchBoot Arm blueprint
- Jason Gunthorpe (NVIDIA): feedback on this deck

3mdeb and alumni

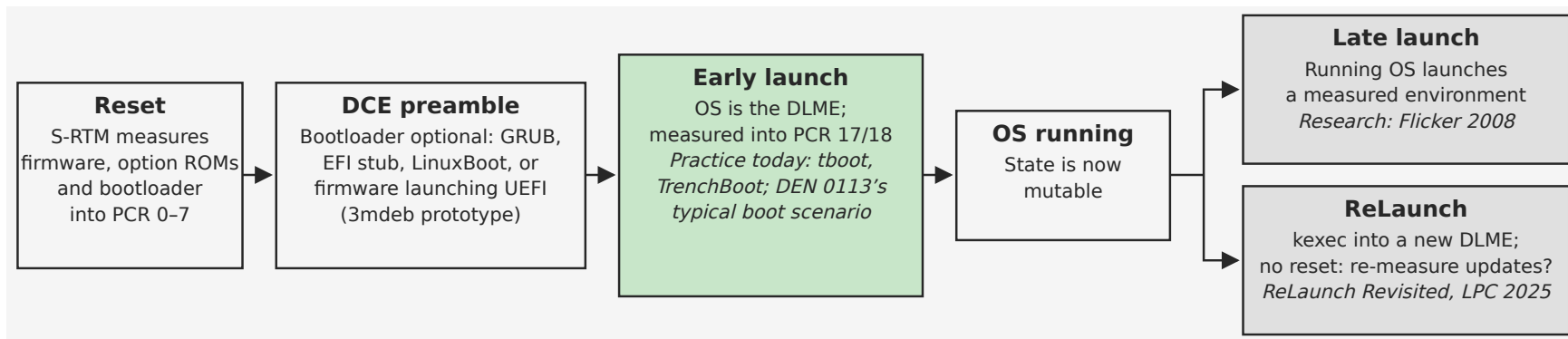
- Sergii Dmytruk: GRUB and Xen Secure Launch; Arm blueprint
- Maciej Pijanowski: Arm DRTM blueprint; Qualcomm IQ-9075 research
- Michał Żygowski: Arm lab on RD-V3, EDK2 + TF-A
- Krystian Hebel (formerly 3mdeb): TXT and parallel AP bring-up in Xen

Why trust from reset does not scale



- Everything since reset is trusted: hundreds of firmware drivers
- Golden values do not scale: every model, update and option ROM differs
- One flaw ships everywhere: LogoFAIL 2023, PKfail 2024
- AI era: more code, faster churn; 75% of Google's new code is AI-generated
- Regulation: EU CRA reporting since 11 September 2026

D-RTM measures only what is launched. It does not fix vulnerable firmware.

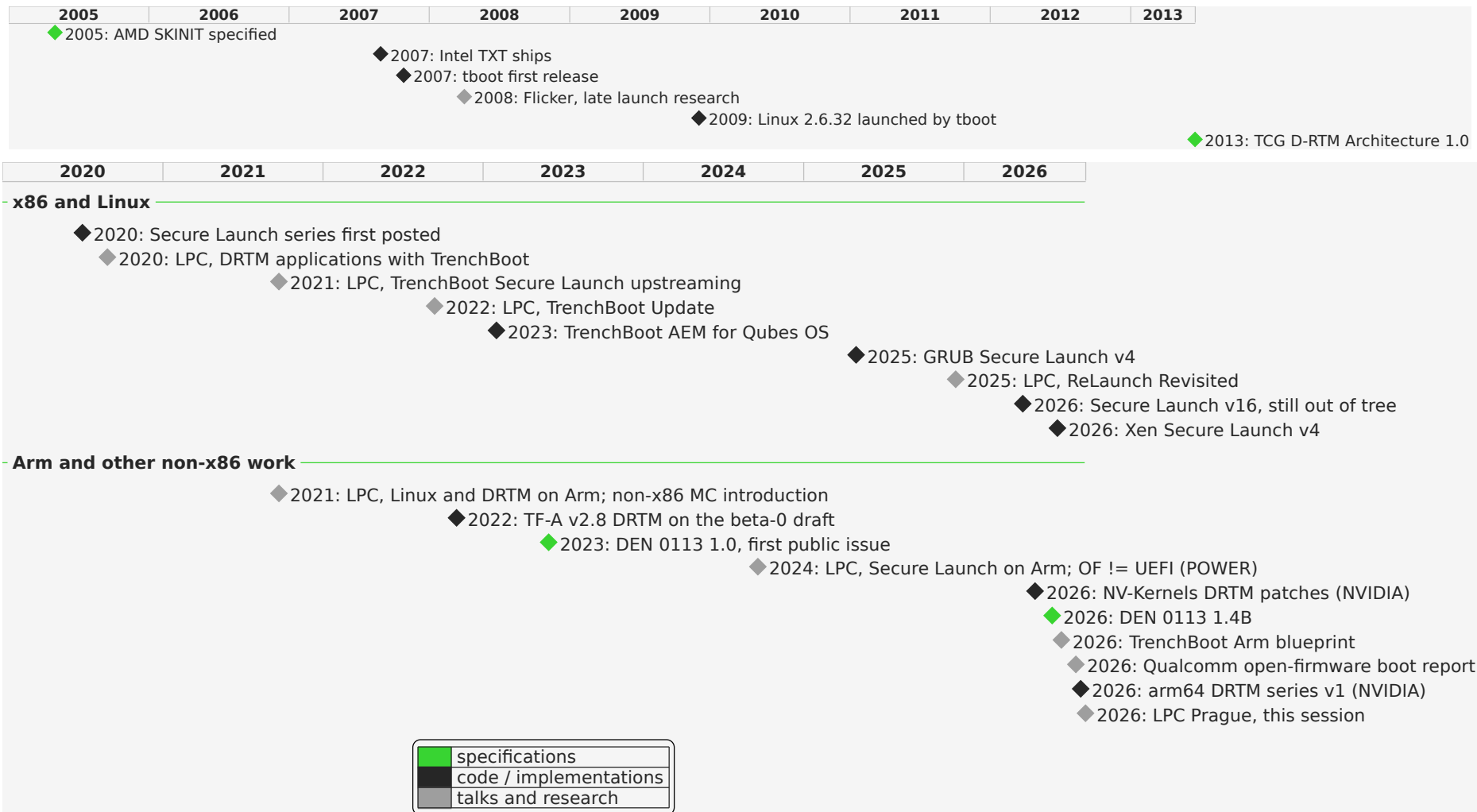


Terms: TCG defines dynamic launch "at an arbitrary time"; TrenchBoot splits early launch from late launch; Arm calls the boot-time launch the "typical boot scenario".

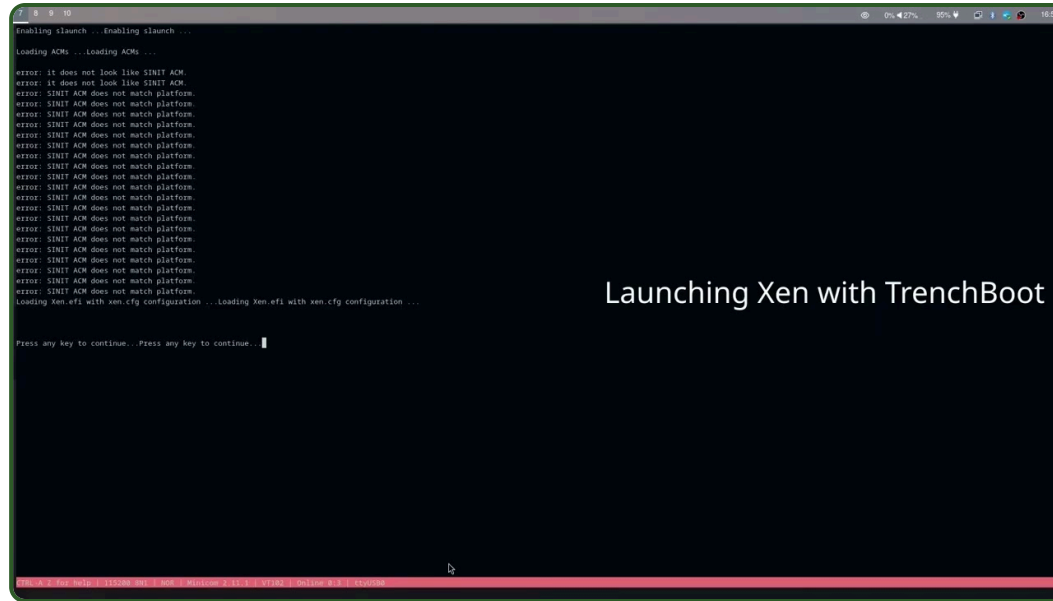
Why "any time" is hard: pre-launch state must be "validated or reinitialized"; UEFI runtime services are "a hole as big as a barn".

So we start with early launch, on Arm too: an SMC into EL3 firmware (TF-A) today; a hardware coprocessor can take over later behind the same interface, invisible to Linux.

Our goal: early launch on Arm, upstream; share with x86 what both need. ReLaunch next.



x86 today: TrenchBoot on Sapphire Rapids

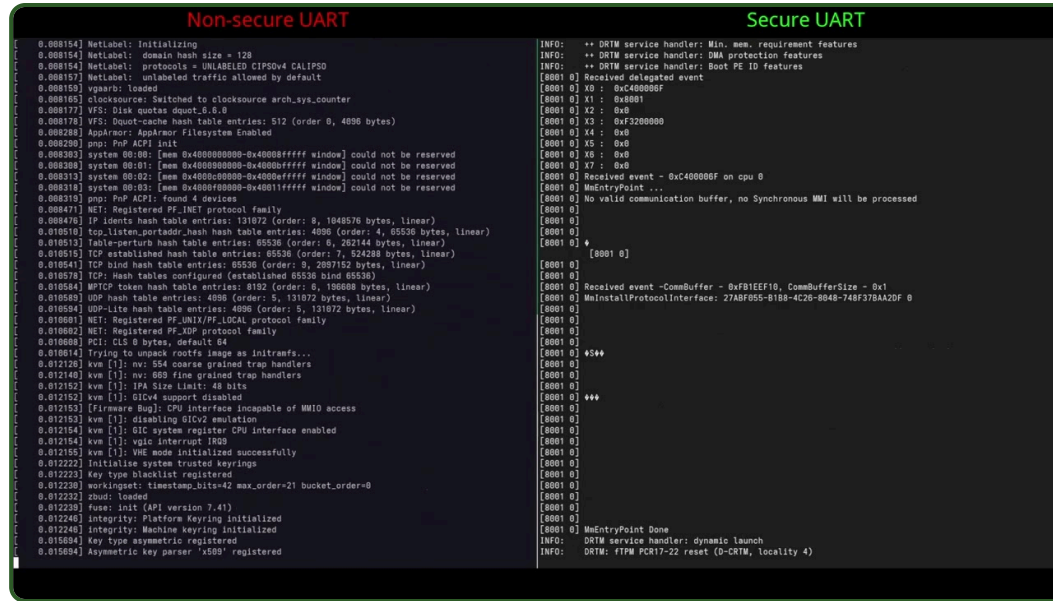


TrenchBoot on Sapphire Rapids

ASRock Rack SPC741D8-2L2T/BCM, Dasharo development build

youtu.be/GfOf6EYTClU

The demo: dynamic launch on the RD-V3 FVP



Arm DRTM on Neoverse RD V3: Dynamic Launch with fTPM and Linux

youtu.be/hR0izdbAI3M

The demo: dynamic launch on the RD-V3 FVP

EDK2

ACPI TPM2 + CRB/SMC interface; measured boot and a TCG2 event log.

TF-A at EL3

PoC RD-V3 DRTM hooks plus a BL31 fTPM; dynamic PCR reset and extend at locality 4.

Debian Linux

6.12.107 with an adapted backport of Jason Gunthorpe's v1 EFI-stub launch and `drtm_entry`.

Observed in the PoC: kernel reports `DRTM: launch completed`; Linux can read PCR 17/18 through TPM sysfs.

Model only, not secure: incomplete DMA/locality protection and volatile TPM state. No attestation or real-board validation is claimed.

Attestation (userspace)

Keylime, fwupd HSI
No dynamic-launch policy

Linux kernel

DLME entry
arm64 v2: posted
NV-Kernels: vendor tree
Interfaces differ from x86

PCR 17/18 + event log
NV kernel measurements: FIRME
D-CRTM coverage unverified

securityfs contract
x86: slaunch/eventlog
NV: slaunch/drtm_event_log

TPM localities
x86 Secure Launch v16:
TPM 1-10, locality 11-14

DCE preamble

GRUB
x86 Secure Launch v4,
no Arm code

EFI stub
arm64 v2 launches itself

LinuxBoot + kexec
NV-Kernels image-header
handoff

Firmware (EL3) and compliance

TF-A DRTM service
Merged since v2.8; Base FVP only
Interface 1.0 (spec 1.4);
arm64 v2 accepts 1.0

plat_drtm silicon port
None upstream;
NVIDIA reports
real HW (v2 cover)

Arm DRTM ACS
Alpha, spec 1.1,
runs from UEFI Shell

Qualcomm IQ-9075
TF-A BL2 + FIP; U-Boot UEFI
TPM measurement behavior:
not established here

 Merged upstream

 Posted, partial or vendor tree

 Missing or diverging

x86: Secure Launch v16 (TrenchBoot)

Prepare: platform launch state
TXT: TXT heap **and** SLRT handoff

Launch: GETSEC[SENDER] or SKINIT

DLME: sl_stub.S -> sl_main()
early TPM at locality 2, policy extends

Arm: DEN 0113 1.4 (B), arm64 v2

Prepare: EFI stub efi_drtm_*() fills
DRTM_PARAMETERS (88 bytes)

Launch: SMC 0xC4000114
D-CRTM in EL3 firmware or a coprocessor

DLME: drtm_entry -> primary_entry
no event log, no locality handling yet

Shared post-launch interfaces: proposal

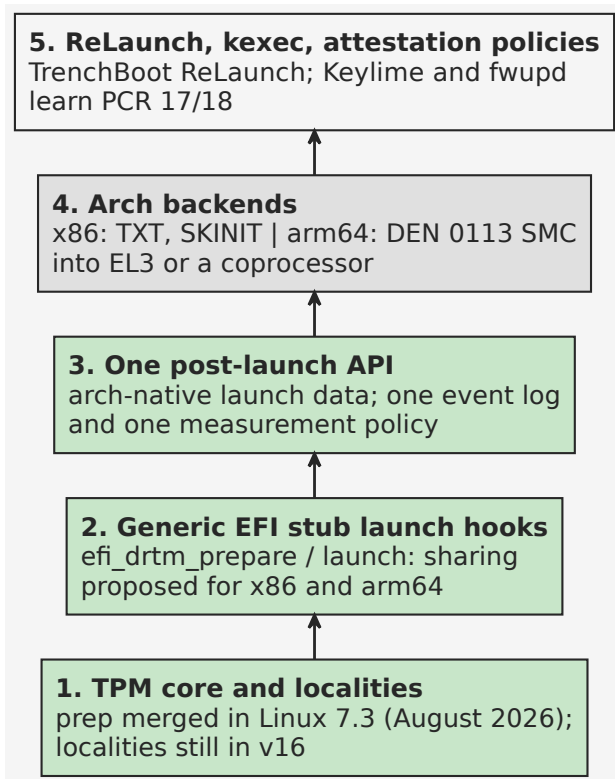
TPM core/localities; one securityfs event log;
measurement policy for cmdline, initrd and FDT.

Arm SLRT entry: open, not assumed

SLRT reserves an Arm tag. The arm64 v2 cover letter
argues for arch-native launch data and no loader ABI.

Kernel entry: DRTM starts Linux with the launch region in `x0`, not the DTB. `drtm_entry` checks the launch layout, takes the DTB from the EFI stub, then boots normally. Open: hardening that DTB hand-off.

What we ask: stack the work so it lands



Questions for this room

1. Which layer lands first, and through which tree: TPM, EFI, x86 or arm64?
2. Who owns an open TF-A DRTM port for silicon, and the PCR 17/18 + event-log gap?
3. One kernel API for launch data across arches, rather than one handoff format?
4. Where should launch and the kernel entry stub live, and which hand-off data must Linux validate?

3mdeb commits engineering to this stack: RD-V3 FVP first, then Rockchip RK3588 and Qualcomm IQ-9075 class boards.

Arm brings a second architecture to Linux DRTM. Shared post-launch hardening should serve the next one.

The background is a dark gray with decorative circuit-like lines in the corners. These lines are light gray and form various geometric shapes, including straight lines, right angles, and small circles, resembling a printed circuit board layout.

Q&A

Backup: launch request and post-launch record

```
/* x86: post-launch record
 * slr_table.h, Secure Launch v16 */
struct slr_table {
    u32 magic;          /* 0x4452544d */
    u16 revision;
    u16 architecture; /* TXT 1, SKINIT 2 */
                        /* Arm: none yet */
    u32 size;
    u32 max_size;
};
/* Reserved tag, no Arm struct yet */
#define SLR_ENTRY_ARM_INFO 0x0006
```

```
/* Arm: launch request, DEN 0113 1.4B
 * Table 9: 88 bytes, revision 2 */
struct drtm_parameters {
    u16 revision; u16 rsvd; u32 launch_features;
    u64 dlme_region_addr, dlme_region_size;
    u64 dlme_image_start, dlme_entry_point_offset;
    u64 dlme_image_size, dlme_data_offset;
    u64 nw_dce_addr, nw_dce_size;
    u64 memprot_table_addr, memprot_table_size;
};
/* DRTM_DYNAMIC_LAUNCH
 * SMC 0xC4000114, X1 = &params */
```

These structures have different jobs. Intel TXT also requires its TXT heap; SLRT alone is not a complete launch contract. Compare lifetimes and validation.

Backup: UEFI runtime services and variables

- "EFI runtime services are driving a hole as big as barn through DRTM" (Stuart Yoder, Arm, LPC 2021)
- NV-Kernels: "A DRTM launch must not call unmeasured pre-DRTM UEFI runtime services"
- TrenchBoot Xen also disables firmware runtime calls for Secure Launch (submitted implementation; mainline status not claimed)
- On Arm, UEFI is a Non-secure EL2 payload; variables may be served by a Secure world partition (Standalone MM over FF-A), outside the launched measurement
- Qualcomm's U-Boot/UKI path and NVIDIA's EFI/RT restrictions are implementation evidence, not a UEFI or DRTM certification claim
- The launch itself is an SMC into EL3, callable at runtime without UEFI runtime services; the open question is which UEFI runtime state a DLME may trust
- Firmware launching the UEFI payload as the DLME (3mdeb, coreboot and EDK2 on AMD, 2022) would bring the UEFI runtime under measurement; Arm is future work

References

Specifications and firmware

- [DEN 0113 DRTM Architecture for Arm, 1.4 \(B\)](#)
- [DEN 0113 1.4B PDF](#)
- [TF-A DRTM service, v2.15.0](#)
- [TF-A DRTM proof of concept](#)
- [TF-A v2.8 DRTM header \(beta0\)](#)
- [TF-A maintainers \(DRTM\)](#)
- [Arm sysarch-acis DRTM tests](#)

Platforms

- [meta-qcom PR 3258, open-firmware boot](#)
- [Qualcomm Dragonwing IQ-9075 EVK](#)
- [ODROID-M2, Rockchip RK3588S2](#)

Linux, bootloaders, hypervisors

- [x86 Secure Launch v16](#)
- [arm64: DRTM, boot portion v2 \(v1\)](#)
- [NV-Kernels: arm64 DRTM launch handoff for kexec loaders](#)
- [GRUB Secure Launch v4](#)
- [Xen Secure Launch v4](#)
- [Linux v2.6.32 tboot support](#)

TrenchBoot and 3mdeb

- [TrenchBoot DRTM on Arm blueprint](#)
- [Prospects of TrenchBoot on ARM](#)
- [Qualcomm DRTM Launch](#)
- [TrenchBoot AEM for Qubes OS](#)

Earlier talks

- [This session: Arm DRTM beyond the reference model, LPC 2026](#)
- DRTM at every LPC since 2017: next page

References: DRTM at Linux Plumbers

- 2017 [EFI + Intel TXT and TPM + Xen/Linux: how to make it work](#), Daniel Kiper
- 2019 [TrenchBoot: how to nicely boot system with Intel TXT and AMD SVM](#), Daniel Kiper
- 2020 [Advanced Applications of DRTM with TrenchBoot SecureLaunch for Linux](#), Daniel P. Smith
- 2020 [Passing and retrieving information from bootloader and firmware](#) (partly DRTM), Daniel Kiper, Michał Żygowski
- 2021 [Linux and DRTM on Arm](#), Stuart Yoder
- 2021 [TrenchBoot Secure Launch upstreaming](#), Ross Philipson, Daniel P. Smith
- 2021 [D-RTM on non-x86 architectures like ARM and POWER9](#), Piotr Król (deck at the MC introduction)
- 2022 [TrenchBoot Update](#), Daniel P. Smith, Ross Philipson
- 2024 [Secure Launch: DRTM solution on Arm platforms](#), Daniel P. Smith, Ross Philipson
- 2024 [Challenges in developing trustworthy Linux-based systems in an open-source way](#) (partly DRTM), Piotr Król
- 2025 [ReLaunch Revisited: a refresher on TrenchBoot late launch](#), Daniel Kiper, Daniel P. Smith
- 2025 [Android Boot, DRTM, UKIs](#) (partly DRTM), Dmitrii Merkurev, Leif Lindholm, Ram Muthiah

References: background

Launch terms and history

- [TCG D-RTM Architecture 1.0, 2013](#)
- [Intel TXT MLE Developer's Guide, rev 017.3](#)
- [tboot, first release 2007](#)
- [Flicker, EuroSys 2008](#)
- [TrenchBoot: early launch and late launch](#)
- [U. Steinberg, NOVA, BSMConf 2026](#)

Why trust from reset does not scale

- [Microsoft: hardware root of trust and SRTM](#)
- [LVFS: firmware SBOM, hundreds of DXE binaries](#)
- [CERT: LogoFAIL and PKfail](#)
- [Google: share of AI-generated code, 2026](#)
- [EU Cyber Resilience Act, 2024/2847](#)
- [EU Product Liability Directive, 2024/2853](#)

References: supporting sources 1

- [TrenchBoot Arm blueprint proposal](#)
- [DEN 0113 1.0](#)
- [Keylime measured boot](#)
- [systemd-cryptenroll](#)
- [Binarily PKfail](#)
- [DORA 2025](#)
- [v16 secure_launch.rst](#)
- [U. Steinberg, BSMConf 2026](#)
- [Linux and DRTM on Arm, LPC 2021](#)
- [TrenchBoot AMD coreboot DRTM payload blueprint](#)
- [NLnet: DRTM launch between coreboot and UEFI payload](#)
- [Dasharo coreboot PR 926](#)

References: supporting sources 2

- [RD-V3 FVP fTPM and DRTM PoC: implementation and limitations.](#)
- [TF-A v2.15.0 firmware design](#)
- [Secure Launch v16 patch 34/38](#)
- [AMD Pacifica specification, 33047 rev 3.01 \(2005\)](#)
- [Intel vPro launch, 2007-08-27](#)
- [LPC 2024, IBM POWER secure boot \(Nayna Jain, George Wilson\)](#)
- [TF-A v2.15.0 DRTM interface header](#)
- [RD-INFRA: successful DMA unprotect and failed-launch unwind](#)
- [TrenchBoot Xen: disable firmware runtime calls for Secure Launch](#)
- [fwupd HSI](#)
- [LPC 2024 Q&A, 27:19](#)
- [LPC 2024 Q&A: entry-interface discussion, 30:53](#)
- [TrenchBoot Secure Launch specification](#)
- [Michał Żygowski: Roadmapper timeline source](#)
- [Jonathan Cameron: first-pass review of the EFI-stub launch patch](#)
- [Jason Gunthorpe: version query and allocation reply](#)
- [Krystian Hebel: Xen parallel AP-stack preparation](#)
- [Krystian Hebel: Xen TXT AP bring-up](#)