TECHNISCHE UNIVERSITÄT BERLIN

Not Discrete Enough: On the Inherent Insecurity of dTPMs for Measured Boot

Cyber Security Experimentation and Test (CSET'25)

Christian Werling | **Tahmid Zahin** | Prof. Dr. Jean-Pierre Seifert

Honolulu, Hawaii, USA

08 December 2025

TPM

- **Trusted Platform Module (TPM)**: Hardware root of trust for cryptographic operations
  - Discrete TPM (dTPM): Separate chip, external communication bus
  - Firmware TPM (fTPM): Software implementation within CPU package

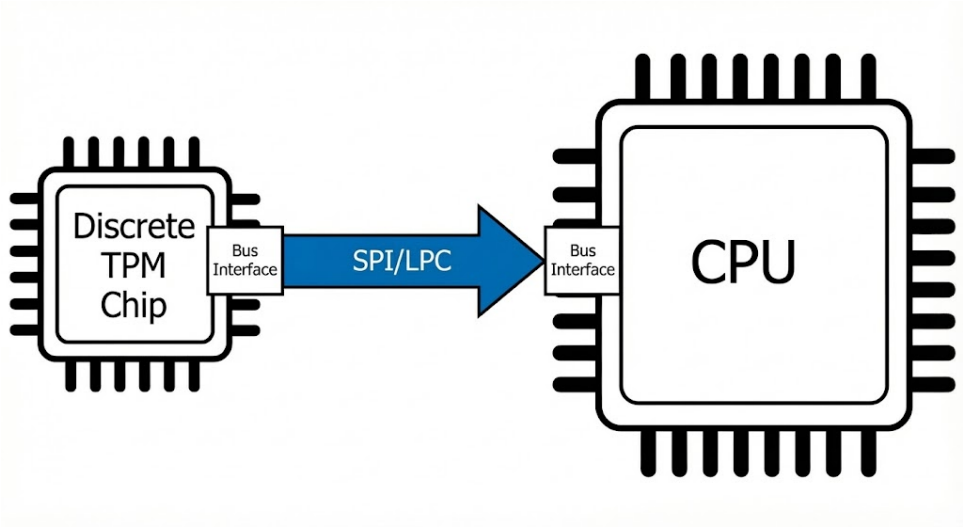


Fig: dTPM

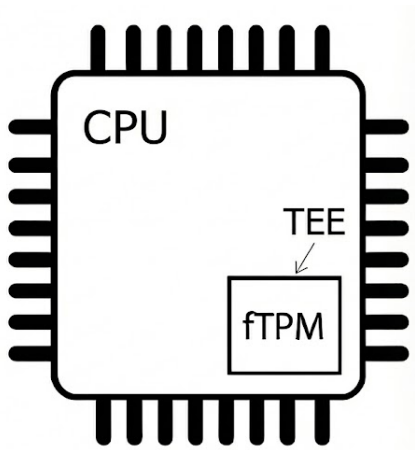


Fig: fTPM

Platform Configuration Registers & Full Disk Encryption

- **PCR:** Registers to store immutable states

- **Hash Chain**: Measuring system state with PCR_EXTEND
  - PCR_new = Hash(PCR_old || measurement)
  - Reset to Zero only with TPM Reset

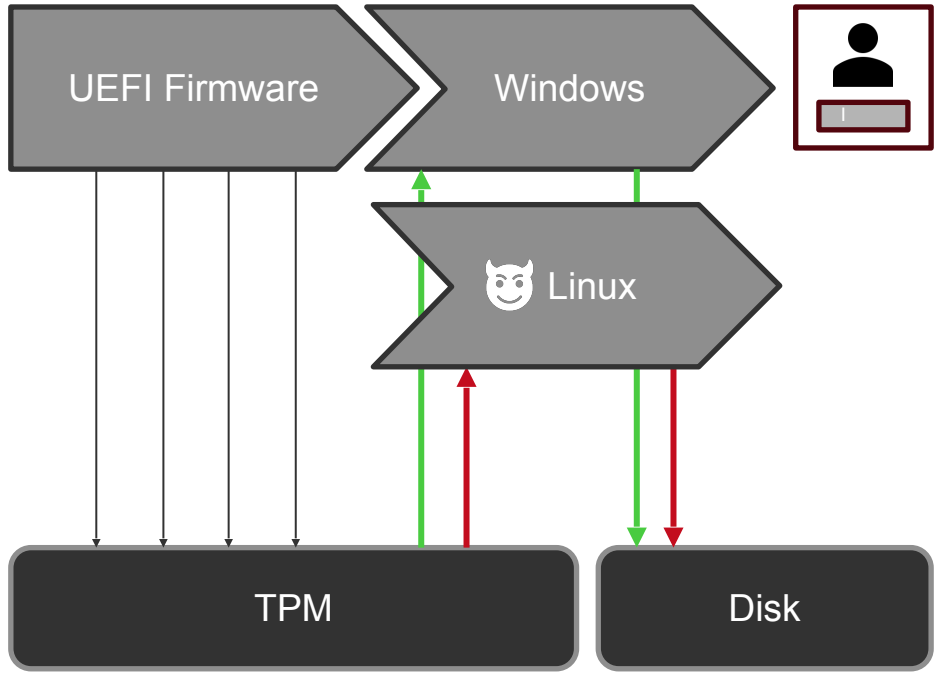
- **Full Disk Encryption**: BitLocker/LUKS seal keys to specific PCR states
  - PCR 7: Secure Boot state and authorities
  - PCR 11: Boot Manager "lock" mechanism (initially zeros, extended after unsealing)
  - TPM only protector

Attacks against discrete TPMs

- **Typical attack path**: Sniff bus communication of dTPMs to steal key
  - Could be mitigated through bus encryption techniques
- **Another security assumption**: TPM reset only occurs with platform reset
- **OSLO**: Bernhard Kauer performed a TPM (v1.2) reset attack 18 years ago (16th USENIX)
- **Validation**: 2 dTPM platforms (Asrock B850 & Lenovo ThinkPad T480 with Infineon SLB9670)
- **Research Questions**:
  - Can the dTPMs (v2.0) on modern hardware still be reset from an attacker-controlled environment?
  - Can event logs be extracted and replayed to reconstruct BitLocker's expected PCR states?

Threat Model

- Goal: Protect laptop when it’s stolen or lost
- Windows **Lockscreen** protects *running* system from unauthorized access
- However, Windows disk could be read from an attacker-controlled Live Linux
- With **TPM-backed** Full Disk Encryption and a key protected by the TPM:
  Live Linux will not be able to access the key (sealed by TPM) or disk (encrypted)



Attack Overview - End to End Process

Total Time: ~10 minutes • **Cost: $0–5 in basic tools**

UEFI Secure Boot vs. Measured Boot

1. Secure Boot

- UEFI Firmware enforces integrity ("Verify then Execute")
- **Chain of Trust:**
  - *Root (PK) → KEK → Signature DB (db) / Deny list (dbx)*
- **Outcome:** Blocks unauthorized bootloaders in UEFI
- **Commonly used certificates:**
  - Windows […] CA → signs Windows bootloaders
  - Microsoft […] CA → signs 3rd party bootloaders (e.g. Linux shim)

2. Measured Boot

- Firmware extends measurements of code and configuration into **TPM PCRs** before transferring control
- PCR7 records Secure Boot state and configuration
- **Outcome:** Immutable* evidence of the boot state

PCR7 Event Log Analysis

1. **EV_EFI_VARIABLE_DRIVER_CONFIG** `SecureBoot`
   Secure Boot status
   Inside: 1-byte state for `SecureBoot` variable (enabled=1, disabled=0), plus GUID/attributes.

2. **EV_EFI_VARIABLE_DRIVER_CONFIG** `PK`
   Platform Key
   Inside: root of trust public key material (X.509 in EFI signature lists) for Secure Boot.

3. **EV_EFI_VARIABLE_DRIVER_CONFIG** `KEK`
   Key Exchange Keys
   Inside: keys authorized to update `db` / `dbx` (typically OEM and Microsoft keys).

4. **EV_EFI_VARIABLE_DRIVER_CONFIG** `db`
   Signature Database
   Inside: authorized certificates and/or hashes for EFI images.

5. **EV_EFI_VARIABLE_DRIVER_CONFIG** `dbx`
   Forbidden Signature Database
   Inside: revoked certificates and blocked image hashes.

6. **EV_SEPARATOR** `0x00000000`
   Transition delimiter
   Inside: fixed 4-byte zero value marking the boundary before authority measurements.

7. **EV_EFI_VARIABLE_AUTHORITY** `db`
   Verification authority
   Inside: the actual certificate chain from `db` used to validate code (issuer/subject, etc.).

Secure Boot on or off?

Secure Boot configuration

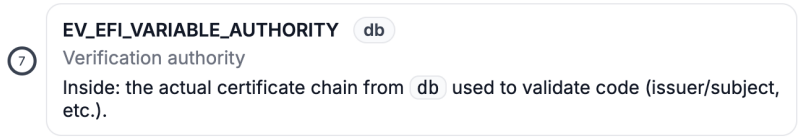
Who authorized the current boot loader?

(→ is it a Windows or a 3rd party boot loader?)

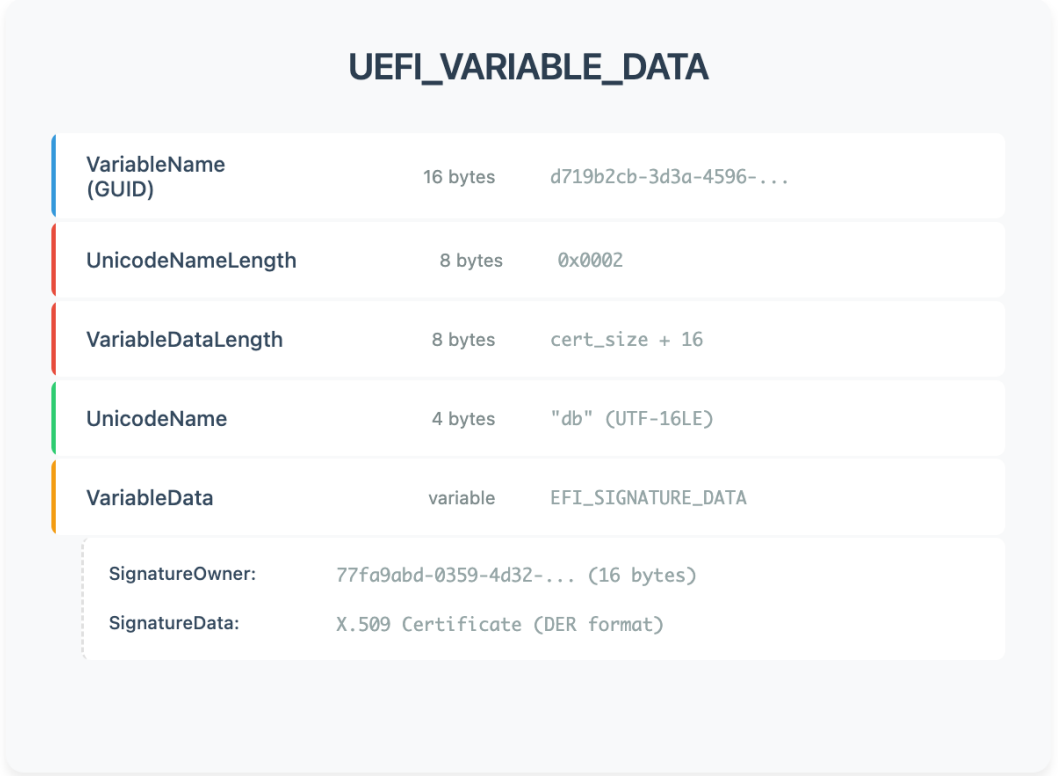
Event Digest Reconstruction Challenge

- **Why it can’t be replayed**:

  EV_EFI_VARIABLE_AUTHORITY event measures the Secure Boot certificate used to validate bootloader



- **Required Components**:
  - Signature Owner GUID: Globally unique based on authority Windows/Linux
  - Certificate: Microsoft Windows Production CA 2011
- Technical Solution:
  - Reverse-engineered firmware digest calculation



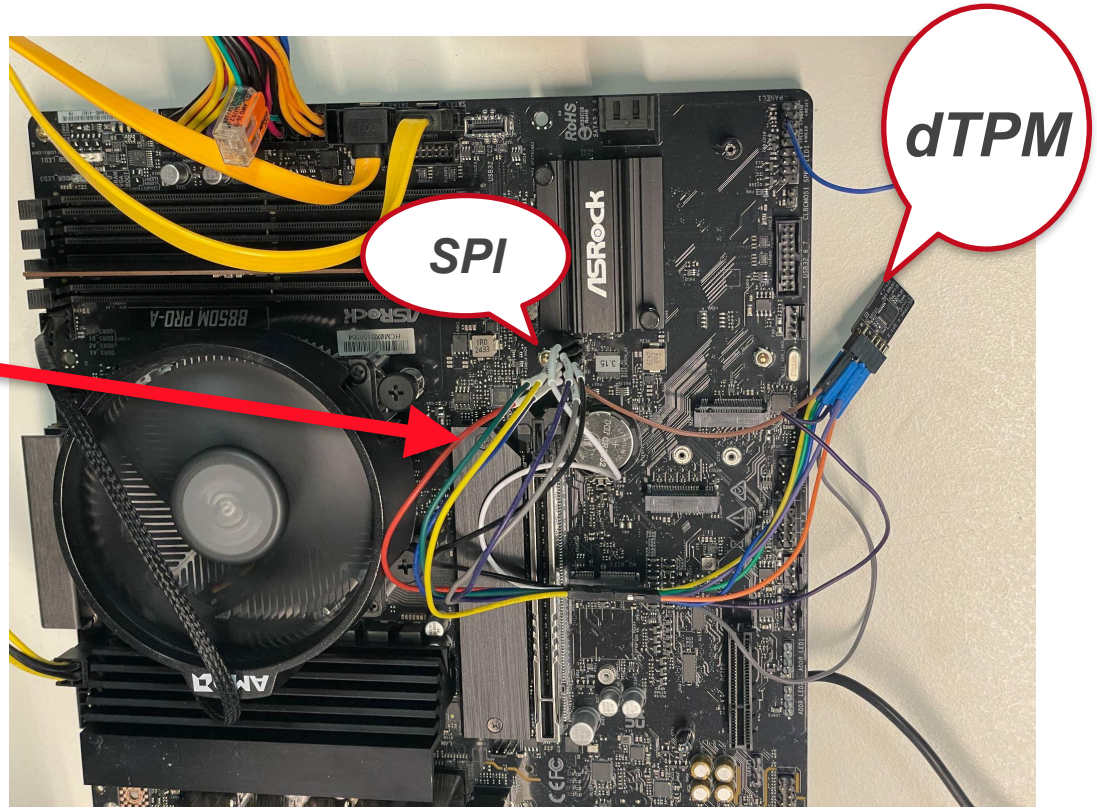
Complete PCR7 Replay Process

1. Parse event log into structured format
2. Replay PCR7 events up to EV_SEPARATOR (firmware/OS boundary)
3. Manual event digest calculation: Insert forged EV_EFI_VARIABLE_AUTHORITY Digest

Expected PCR 11 Digest is zero, so no replay required

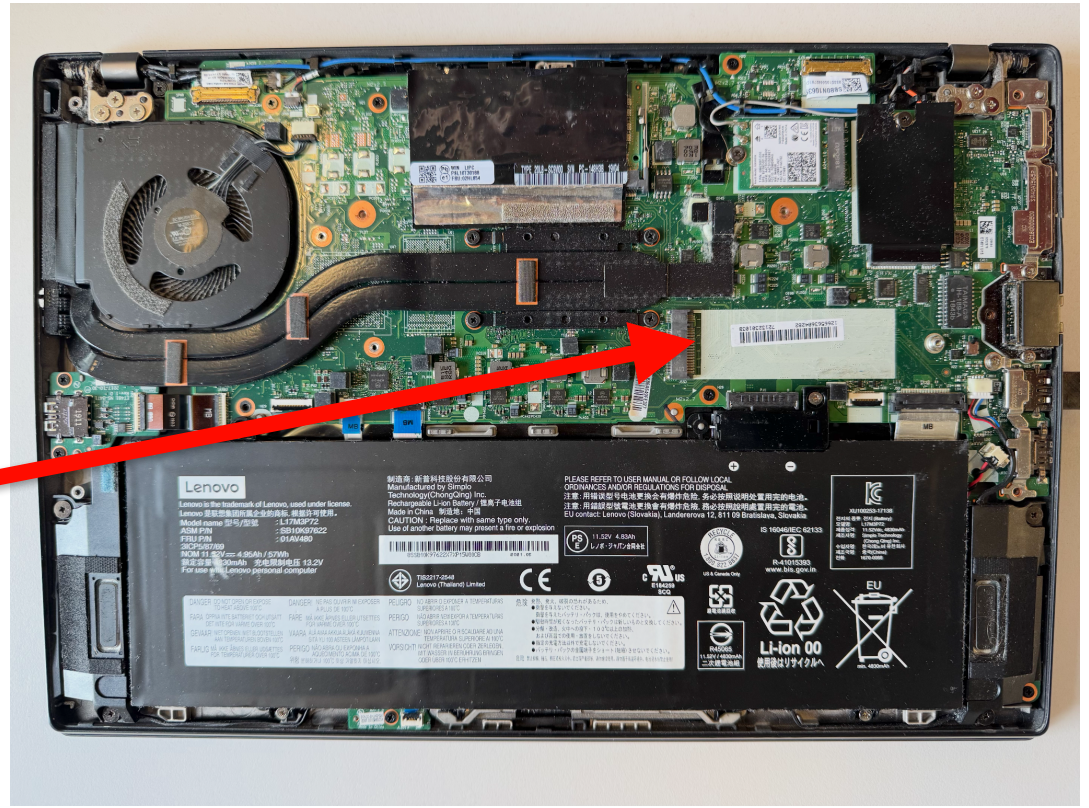
Hardware Reset - Asrock B850 + Infineon SLB9670

- **Challenge**: Reset TPM without host interruption
- **Platform Setup**: Dedicated dTPM module on development board
- **Reset Method**: VCC interruption
- **Process**: Driver unbind → VCC interrupt → Driver rebind → Verify PCRs = 0
- **Result**: Clean Startup(CLEAR) achieved without host reboot



Hardware Reset - Lenovo ThinkPad T480 + Infineon SLB9670

- **Real-World Challenge**: Built-in dTPM, no exposed headers
- **Physical Access**: M.2 slot provides reset mechanism
  - Remove bottom cover
  - Access M.2 SSD slot
  - Ground PERST# pin on M.2 connector
- No full motherboard disassembly required



Hardware Reset - Lenovo ThinkPad T480

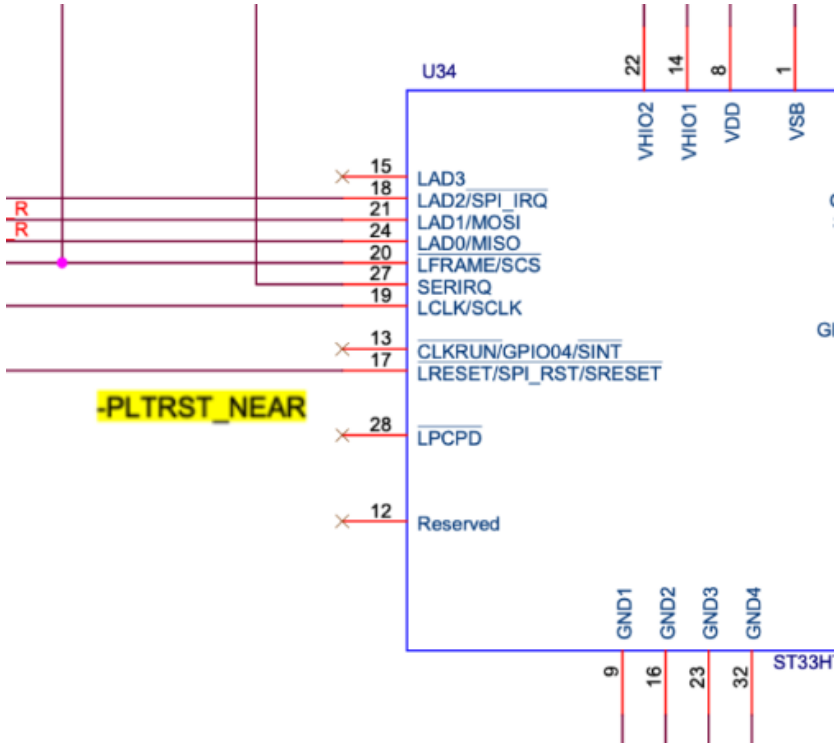
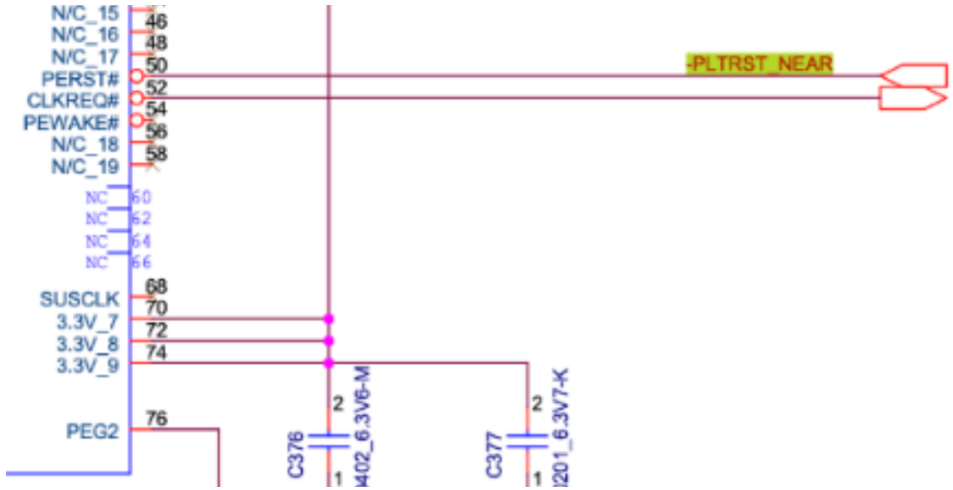


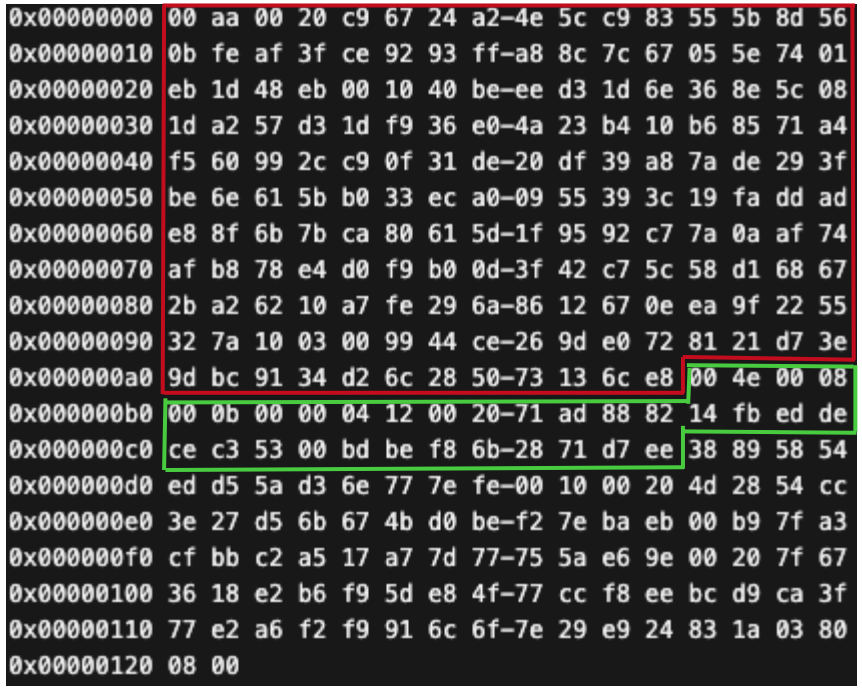
Fig: DTPM Schematic



M.2 Socket 3 (Key-M) for 2280 S3 SSD

Bitlocker VMK Unsealing

- **Sealed Object Extraction**:
  - Parse BitLocker metadata from volume header using dislocker
  - Locate Payload after line “`--> ENTRY TYPE VMK”
  - Extract TPM2B_PUBLIC and TPM2B_PRIVATE structures
- **Unsealing Process**:
  - Load sealed object
  - Start TPM policy session
  - Apply PolicyPCR for PCR 7,11 digest verification
  - Execute TPM2_Unseal command
  - Retrieve decrypted Volume Master Key



Results

- **Validation**: Attack successfully works on both dTPM platforms
- Total attack time ~10 minutes
- **Costs:** Significantly cheaper than bus sniffing

Method	Cost	Complexity
Bus Sniffing	$10-1,499	Medium
Reset/Replay	$0-5	Easy

Conclusion

- **Countermeasure Effectiveness**:
  - ✓ Strong: Firmware/Integrated TPM (eliminates external reset capability)
  - ✓ Strong: TPM+PIN (adds unreplayable human factor)
  - △ Limited: BIOS passwords
  - ✗ Ineffective: Bus encryption (protects parameters but not event log)
  - △ Effective but breaks Spec: PCR hiding. PCR transparency required for attestation

- **Recommendations**:
  - TPM+PIN must be minimum standard for dTPM systems
  - fTPM or other integrated TPM for new designs

Thank You

Not Discrete Enough: On the Inherent Insecurity of dTPMs for Measured Boot

Christian Werling
cwerling@sect.tu-berlin.de
SecT, TU Berlin
Berlin, Germany

Tahmid Zahin
zahin@campus.tu-berlin.de
TU Berlin
Berlin, Germany

Jean-Pierre Seifert
jean-pierre.seifert@tu-berlin.de
SecT, TU Berlin & Fraunhofer SIT
Berlin, Germany