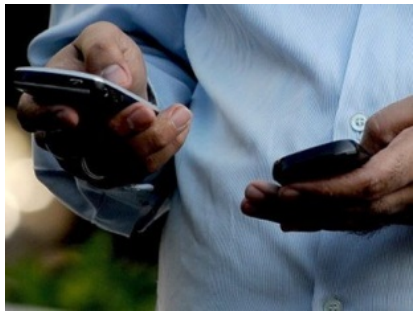
cTPM: A Cloud TPM for Cross-Device Trusted Applications

Chen Chen (ETH Zurich),
Himanshu Raj, Stefan Saroiu, Alec Wolman
(Microsoft Research)

Motivation

- People are using more than one mobile device





- Mobile devices have started to use trusted hardware

Industry



Chromebook





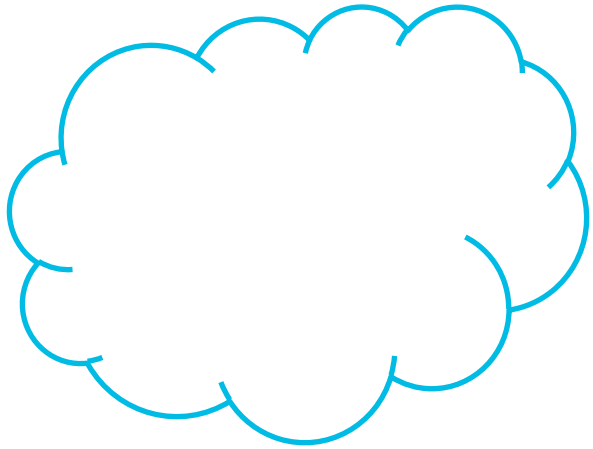
Research Prototypes

A few examples:

- Pasture, Kotla et al. [OSDI'12]
- Trusted sensors, Liu et al. [Mobisys'12]
- TLR, Santos et al. [ASPLOS'14]

Cross-device Data Sharing is Easy





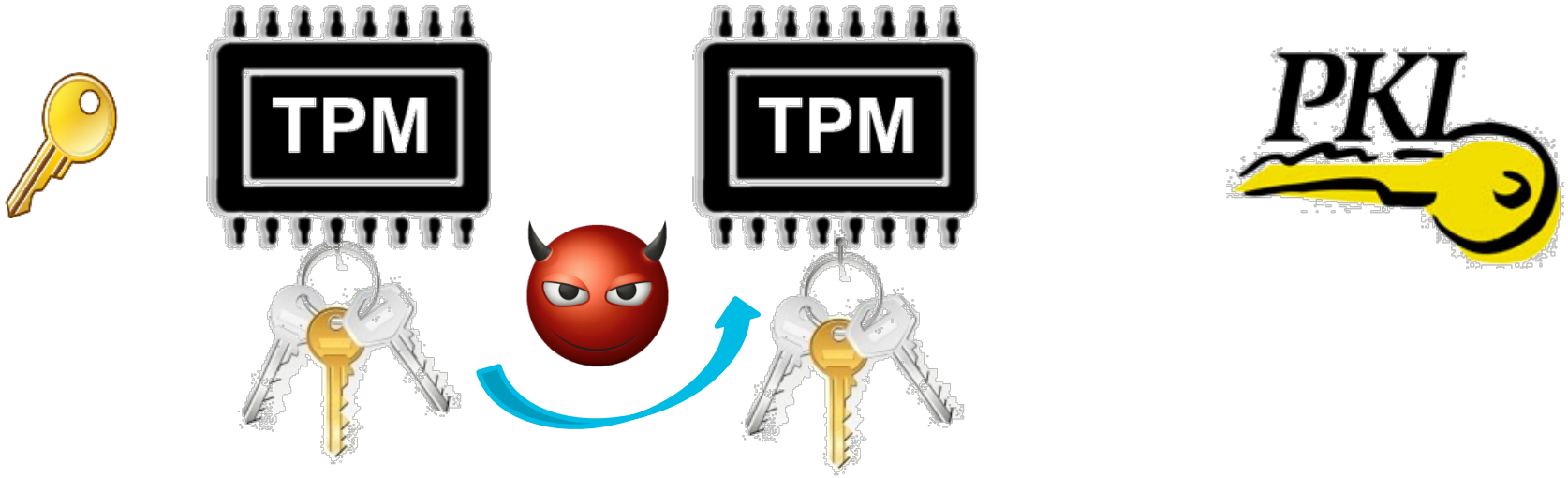


TPM-protected Data Sharing is Hard



Challenge: Sharing Keys across TPMs

- TPM root key never leaves the chip
- Key migration requires PKI + secure execution mode



Main Research Question



- What is the minimal change to the TPM design to enable data sharing across TPMs?

Our Solution: cTPM

- cTPM = TPM + additional root key pre-shared with cloud

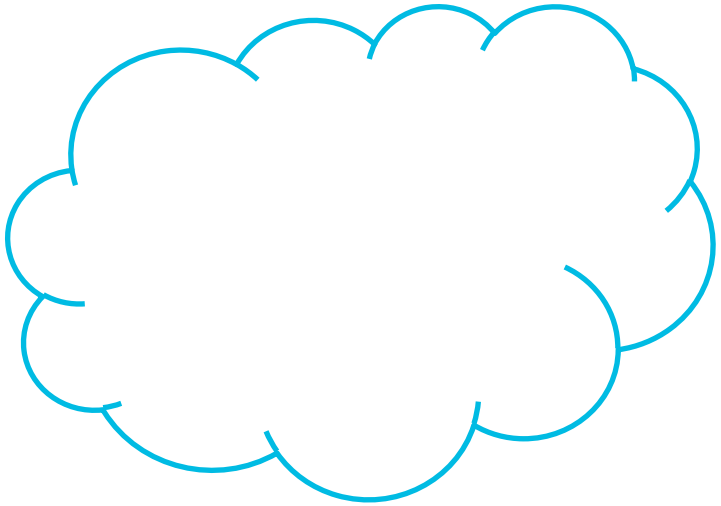


cTPM: Two Additional Benefits

1. Fast and large “remote” NVRAM storage





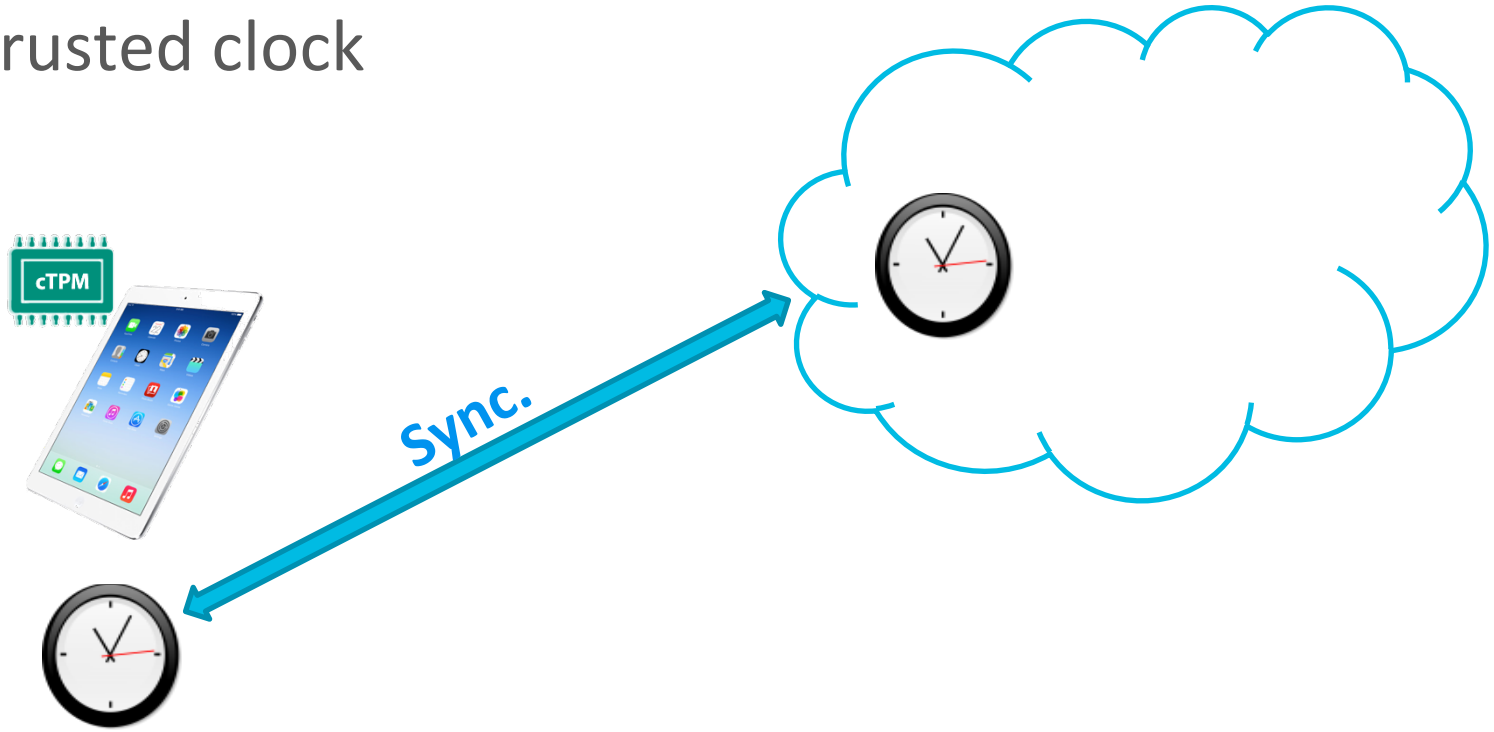






cTPM: Two Additional Benefits

1. Fast and large NVRAM storage
2. Trusted clock



Talk Outline



- Motivation
- Background & Design alternatives
- Design & Implementation
- Evaluation
- Conclusion

TPM Background

- Trusted Platform Module (TPM): secure co-processor
  - Crypto primitives: SHA1, RSA, …
  - Code measurement and attestation

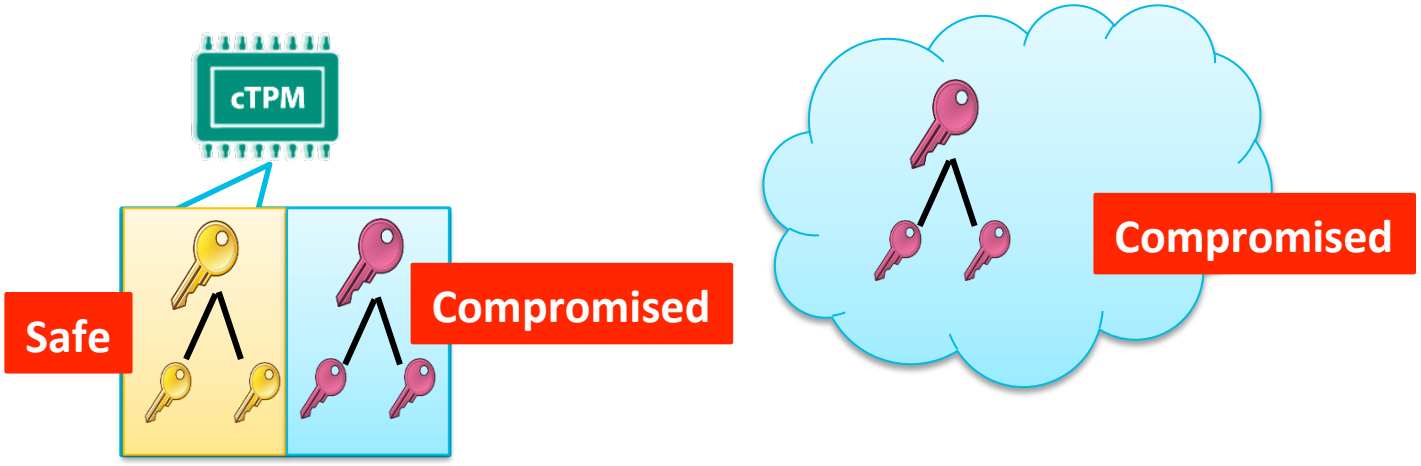


- Trust Computing Group (TCG) defines TPM specifications:
  - TPM 1.2: widely deployed today
  - TPM 2.0: emerging new standard to replace TPM 1.2

cTPM is based on TPM 2.0

Threat Model & Trust Assumptions

- Threat model
  - In-scope: software-based attacks: e.g., malware
  - Out-of-scope: physical hardware attacks on TPM
- Dual relationship with the cloud:
  - We trust cloud with cTPM shared root key
  - We do not trust the cloud with existent TPM root keys



- Future work: securing cloud-side of cTPM

SEM: TPM's extensibility mechanism



- SEM: Secure Execution Mode (aka. Intel TXT)
  - Implements a CPU-based reboot
  - Run trusted code in an isolated environment
- Implement additional TPM functionality with SEM:
  - Step 1: Suspend OS, Enter SEM, Run additional trusted code
  - Step 2: Cleanup, Exit SEM, Resume OS
- Very challenging to use SEM
  - Performance overhead, engineering issues
  - Lack of support on mobile devices
  - No production software today uses SEM

Talk Outline



- Motivation
- Background & Design alternatives
- Design & Implementation
- Evaluation
- Conclusion

cTPM Design Challenges



- Provisioning pre-shared root keys in cTPM
- Secure key sharing across cTPMs
- Secure communication channel
- Cloud-side NV storage
- Trusted clock

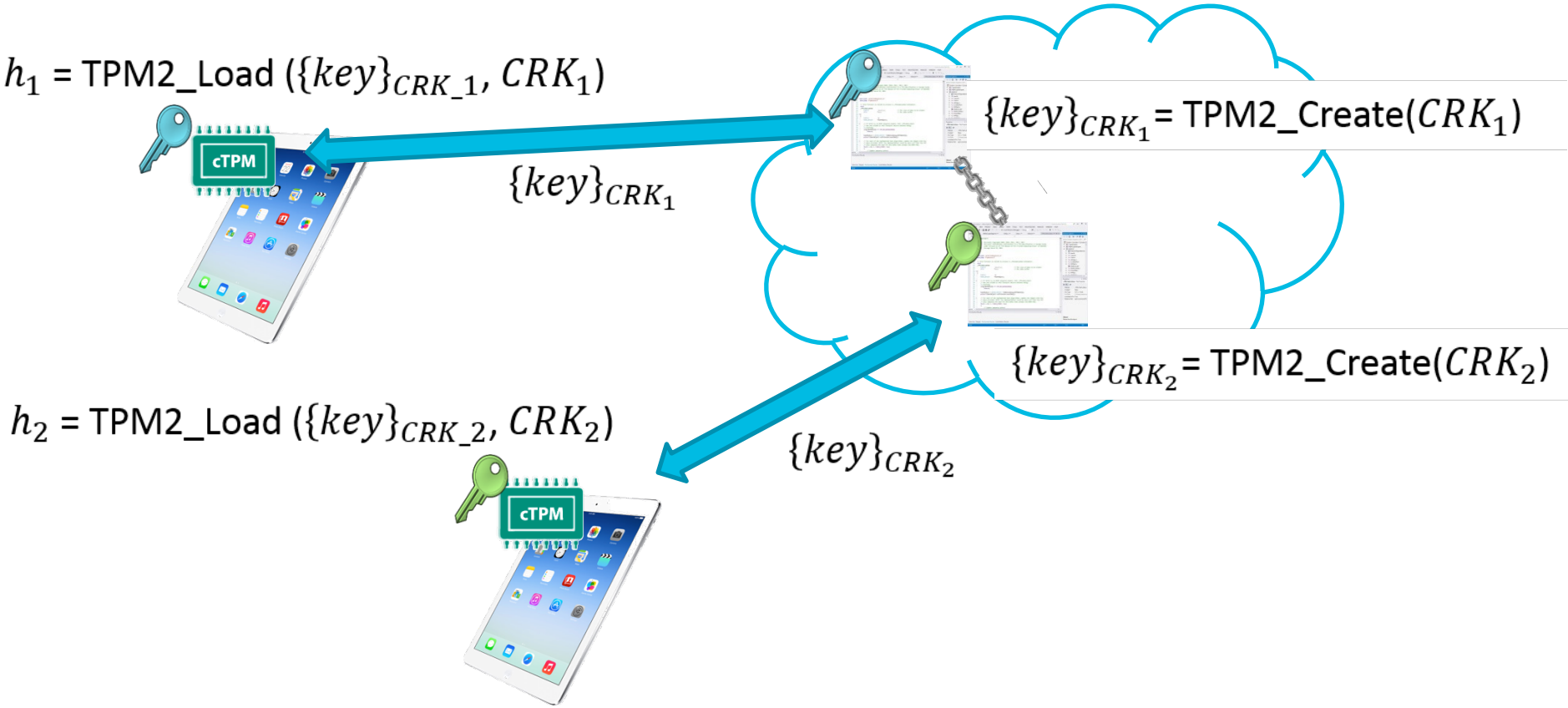
Provision cTPM Root-of-Trust



- We provision each cTPM with a cloud seed:
  Unique random value pre-shared with cloud
  - iPhones, iPads share their seeds with iCloud
  - Microsoft Surfaces share their seeds with Azure

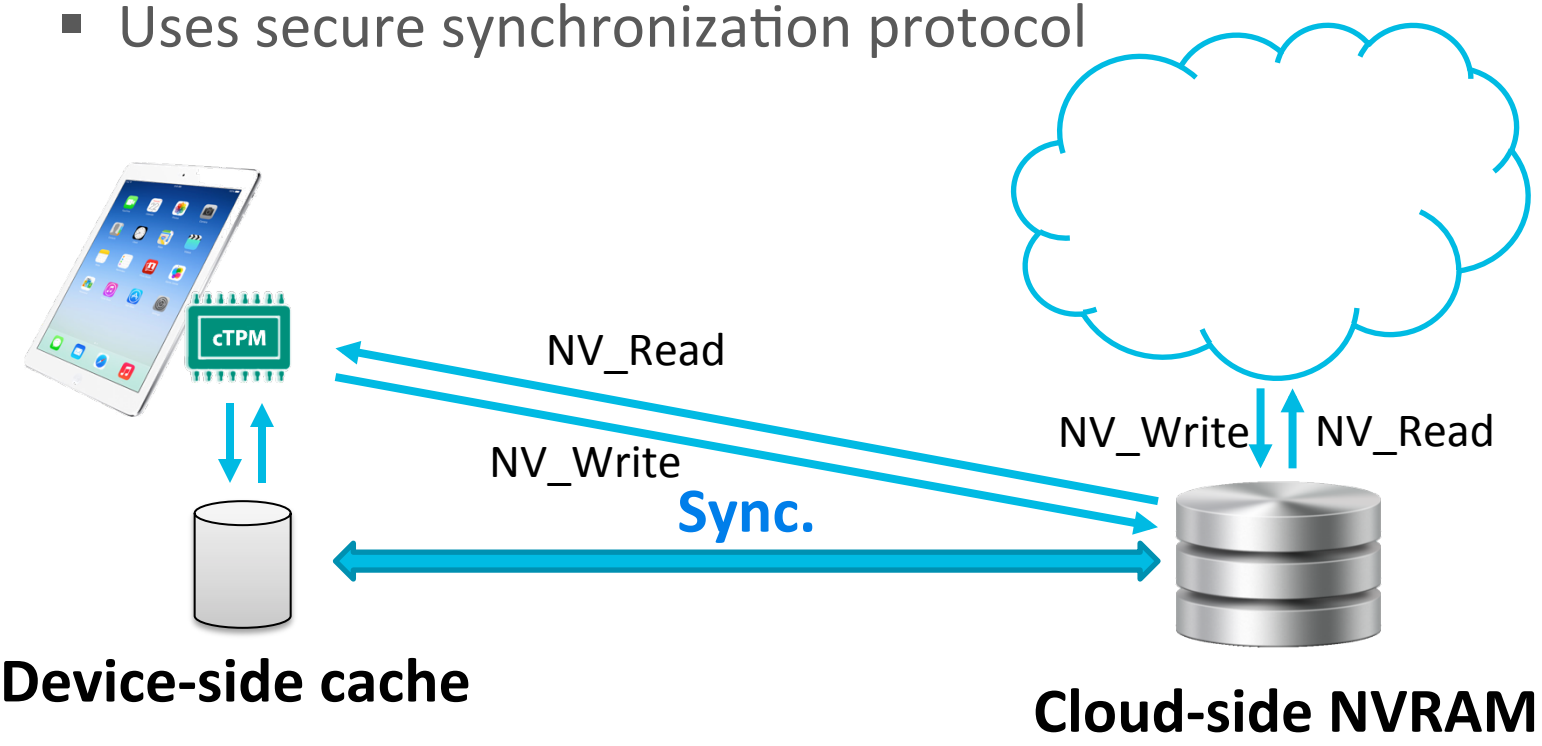
- On boot-up, cTPM deterministically generates two keys:
  - Cloud Root Key (CRK) protects all cross-device secrets
  - Cloud Communication Key (CCK) protects all communication

Secure Key Sharing



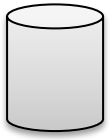
Communication Methodology

- Send/Recv = NV_Write/NV_Read
- Device-side cache masks transient connectivity loss
  - Uses secure synchronization protocol



Synchronization Protocol

- Pull



$@t_1\ \{nonce, NV_READ, NV_Sel\}_{CCK}$

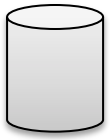


$@t_2\ \{nonce, RC_SUCCESS, NVs\}_{CCK}$

Device-side Cache **Cloud-side NVRAM**

Check $t_2 - t_1 <$ Timeout, otherwise abort

- Push



$\{nonce, ctr, NV_WRITE, NV_Sel, NVs\}_{CCK}$

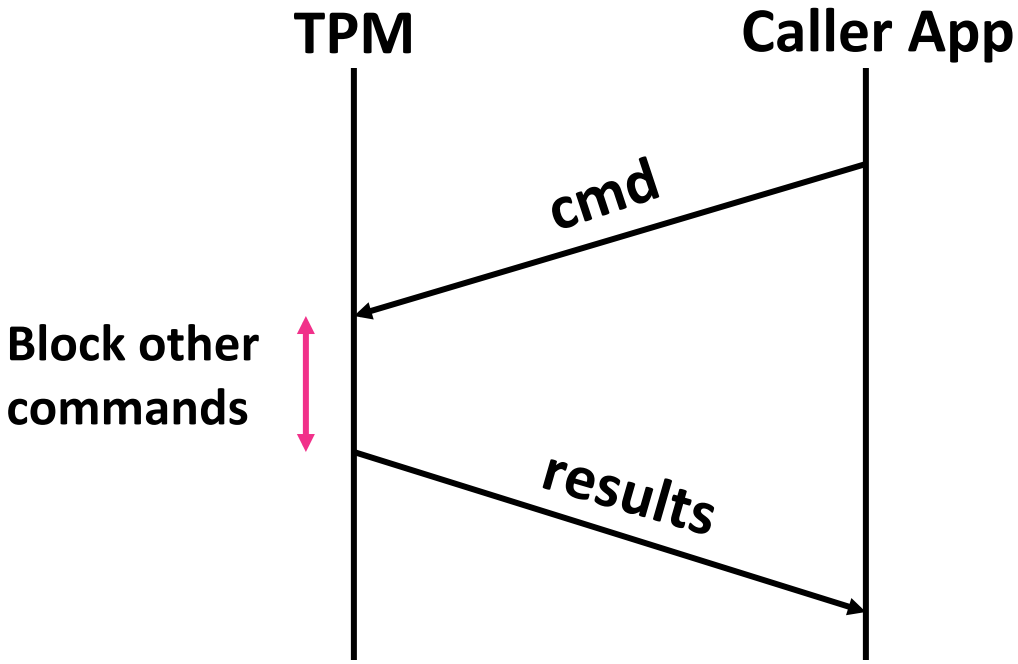


$\{nonce, ctr, RC_SUCCESS\}_{CCK}$

Device-side Cache **Cloud-side NVRAM**

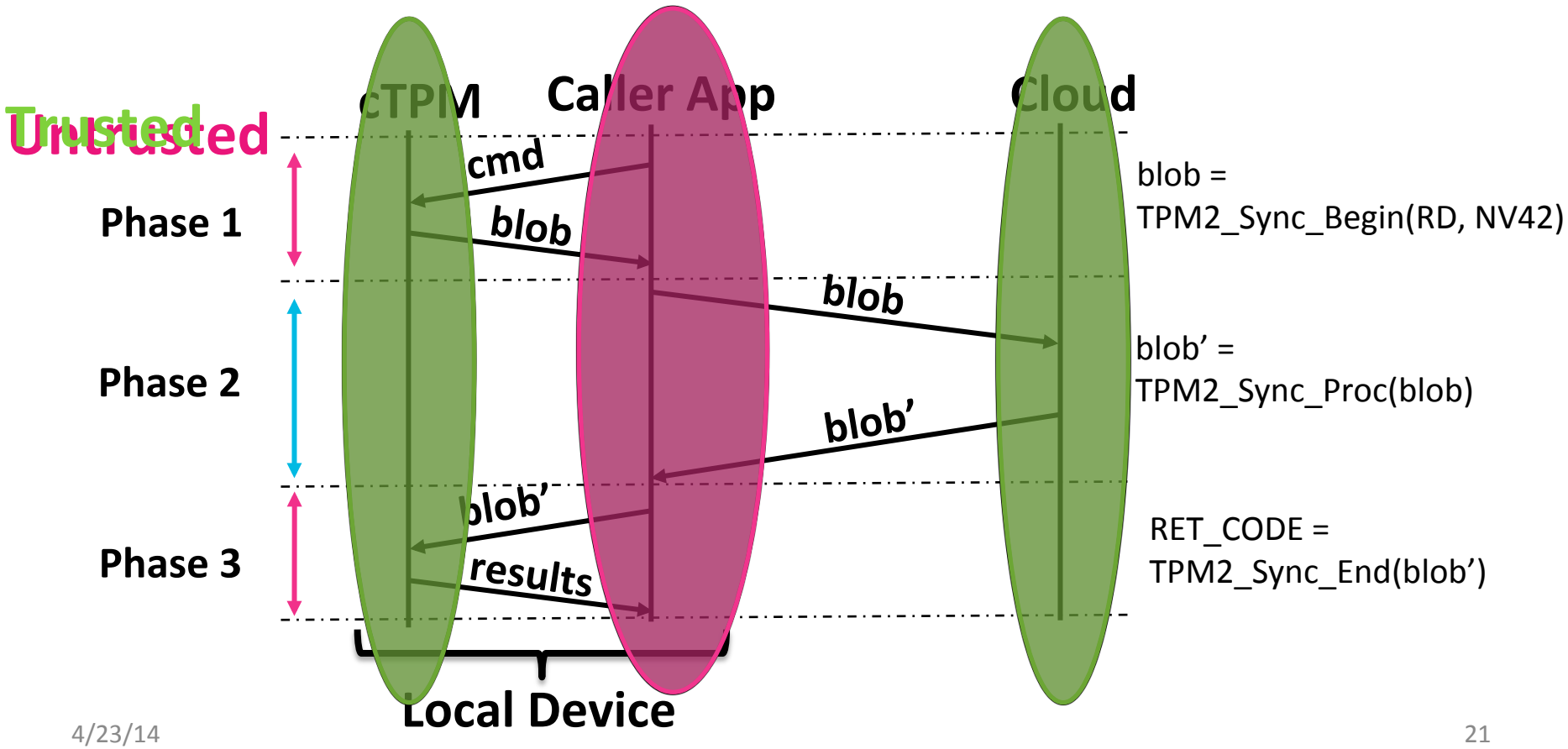
TPM Usage Model

- TPM alone can't establish connection with cloud



Secure Asynchronous Communication

- Design asynchronous commands: better availability and responsiveness
- Establish secure channel out of untrusted entities



Cloud-backed NV Storage



- Local cached NV entries have TTLs
  - Once TTL expires, NV entry in the cache becomes invalid
  - Apps can re-sync with cloud to refresh TTL
- cTPM timeout can abort pending cloud commands
  - Cloud can adjust this timeout, default value is 5 minutes
- Trusted clock: “special” NV entry updated by cloud
  - Clock accuracy controlled by separate clock timeout
  - Default timeout value is 1 second

Implementation Details



- 3 new commands (TPM 2.0 has 108 commands)
  - TPM2_Sync_Begin()
  - TPM2_Sync_End()
  - TPM2_Sync_Proc()

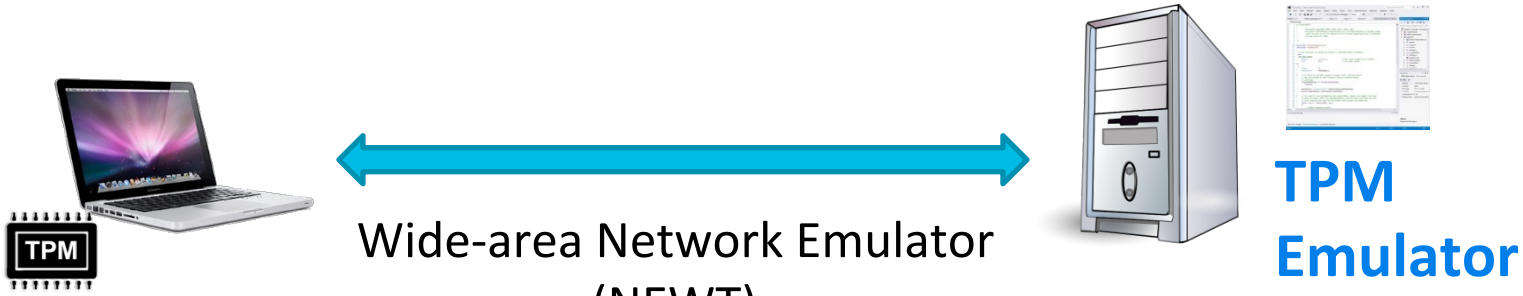
- cTPM is based on TPM2.0
  - 1,304 lines of code (TPM 2.0 is 23,163 LoC)
- cTPM versions of:
  - Pasture, Kotla et al. [OSDI'12]
  - TrInc, Levin et al. [NSDI'09]

Talk Outline



- Motivation
- Background & Design alternatives
- Design & Implementation
- Evaluation
- Conclusion

Evalution Setup



Wide-area Network Emulator (NEWT)
Used 3G and Wi-Fi typical RTTs
(reported by J. Huang et al. [Mobisys'12])

Evaluation Questions



- Are the cTPM protocols secure?
- What is the performance of crypto operations?

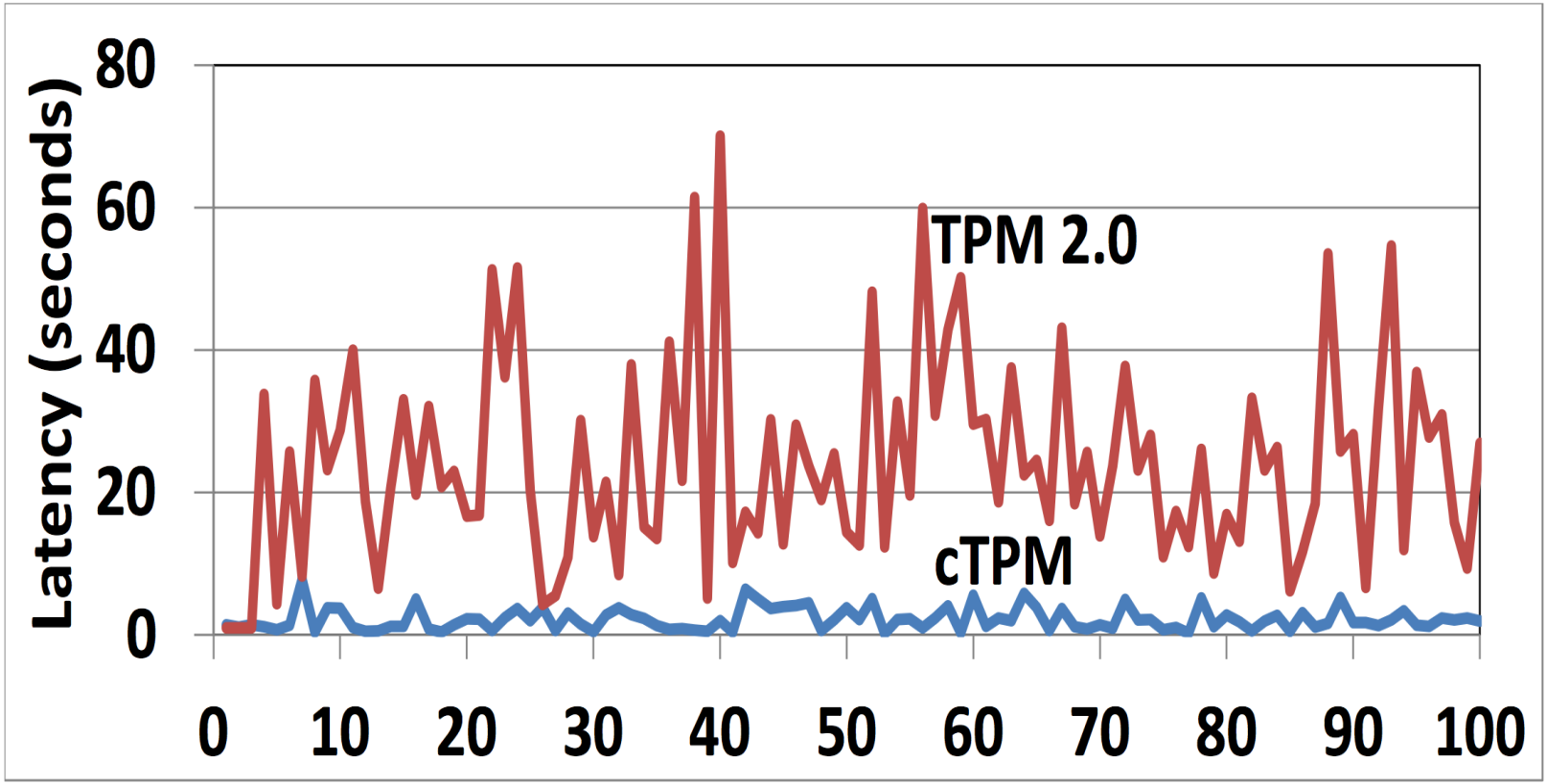
Protocol Verification



- Verified correctness of sync. protocols with ProVerif
- Attacker model:
  - Attacker has unrestricted access to OS, applications & network
- We did not verify the correctness of implementation

Latency of RSA-2048 key creation

- Creating RSA-2048 key in the cloud is 12X faster than on local device



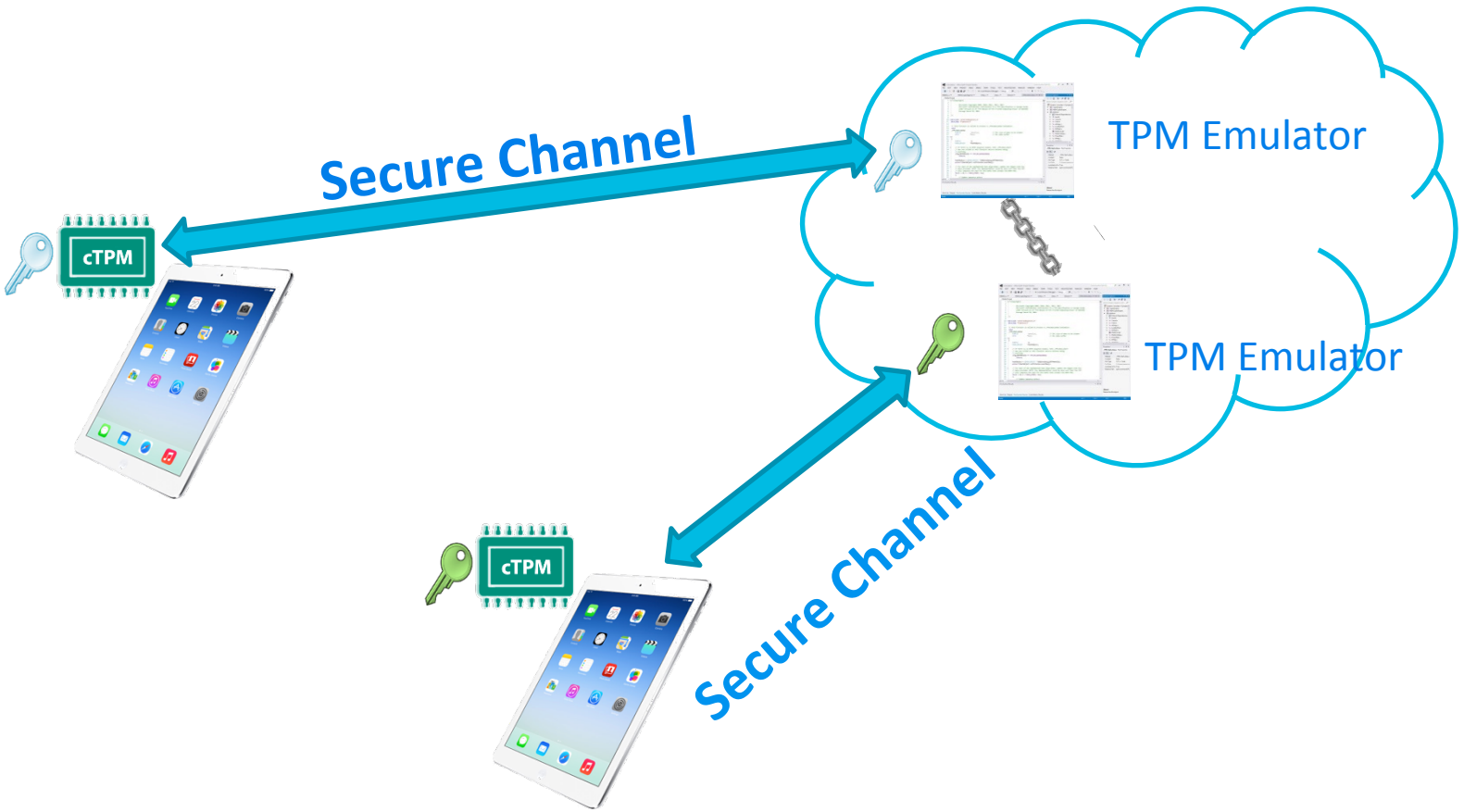
Conclusions



- cTPM: Shared cloud seed in TPM 2.0
  - Small design change to support cross-device scenarios
- Additional cTPM benefits:
  - High performance NV storage
  - Trusted clock
- Full implementation of cTPM in TPM 2.0
  - Implemented Pasture and TrInc on top of cTPM

Thank you

- Chen Chen (ETH Zurich): chen.chen@inf.ethz.ch



Backup #1: Related Work

- [Kotla, 2012] R. Kotla, T. Rodeheffer, I. Roy, P. Stuedi, and B. Wester. Pasture: Secure Offline Data Access Using Commodity Trusted Hardware. In *Proc. of 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Hollywoood, CA, 2012.

- [Liu, 2012] H. Liu, S. Saroiu, A. Wolman, and H. Raj. Software Abstractions for Trusted Sensors. In *Proc. of 10th International Conference on Mobile Systems, Ap- plications, and Services (MobiSys)*, Lake District, UK, 2012.

- [Santos, 2014] N. Santos, H. Raj, S. Saroiu, and A. Wolman. Trusted Language Runtime (TLR): Enabling Trusted Applications on Smartphones. In *Proc. of 12th Workshop on Mobile Computing Systems and Applications (HotMobile)*, Phoenix, AZ, 2011.

- [Huang, 2012] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A Close Examination of Performance Power Characteristics of 4G LTE Networks. In *Proc. of 10th International Conference on Mo- bile Systems, Applications, and Services (MobiSys)*, Lake District, UK, 2012.

- [Levin, 2009] D. Levin, J. R. Douceur, J. R. Lorch, and T. Mosci- broda. TrInc: Small trusted hardware for large dis- tributed systems. In *Proc. of 6th USENIX Sympo- sium on Networked Systems Design and Implemen- tation (NSDI)*, Boston, MA, 2009.

Backup #2: cTPM Owner Change



- When the mobile device is stolen
  - Ask the cloud to stop providing service to the device
- When the mobile device is sold to a new owner
  - Re-keying:
    - Hardware method
      - The merchant use a special device for re-keying
    - Software method
      - Implement a re-key protocol in cTPM

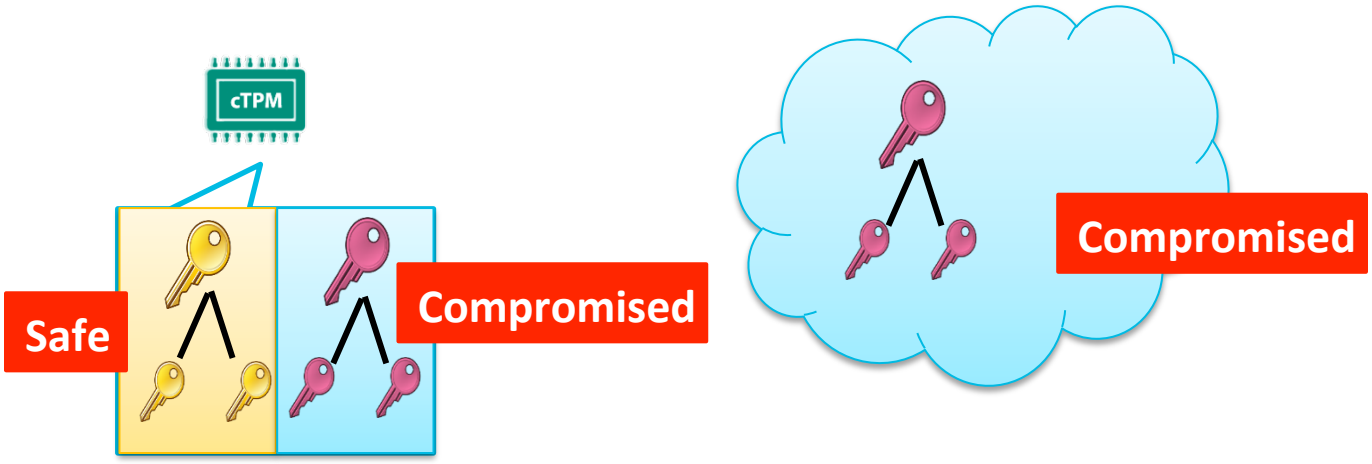
Backup #3: Why not a design alternative



- A shared key with the cloud in current domains
  - Problem: no domain to use
    - Privacy domain is for endorsement key
    - Owner domain is cleared when the owner takes ownership
    - Platform domain is used by the manufacturer for testing

Backup #4: Cloud in Trust Computing Base

- Dual relationship with the cloud



- Securing the cloud is an active area of research:
  - Emerging technology (Intel SGX) can help protect secrets in the cloud

Backup #5: Pasture with cTPM



- Semantics of Pasture with cTPM are complex

- Unmodified Pasture with multiple devices:
  - Hard to keep Pasture protocols in lockstep across multiple devices
- cTPM Pasture with multiple devices:
  - Decisions made on one device are automatically synced to all other devices

Backup #6: TrInc with cTPM



- TrInc requires a counter in NVRAM to solve equivocation problem for distributed system
  - Each counter in NVRAM only supports ~10k writes

- TrInc with cTPM
  - Provides high performance counter in “remote” NVRAM
  - Offers unlimited number of writes

Backup #7: TPM 2.0

- Offer Algorithm Agility
- Offer three control domains
- TPM 2.0 specification == reference implementation

Crypto Primitives: SHA1, SHA256, RSA, ECC, RC4, … …






```
/* Park-Miller "minimal standard" 31 bit
 * pseudo-random number generator, implemented
 * with David G. Carta's optimisation: with
 * 32 bit math and without division.
 */

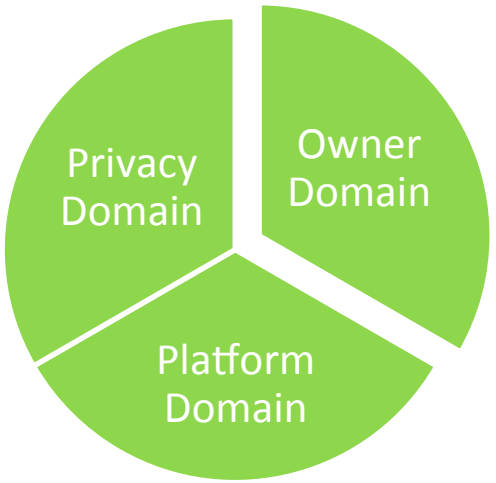
long unsigned int rand31_next()
{
    long unsigned int hi, lo;

    lo = 16807 * (seed & 0xFFFF);
    hi = 16807 * (seed >> 16);

    lo += (hi & 0x7FFF) << 16;
    lo += hi >> 15;

    if (lo > 0x7FFFFFFF) lo -= 0x7FFFFFFF;

    return ( seed = (long)lo );
}
```



Latency of cloud-side NVRAM accesses

- Cloud-side NV write is **3.5X faster** than local-side

