Authenticated Storage Using Small Trusted Hardware

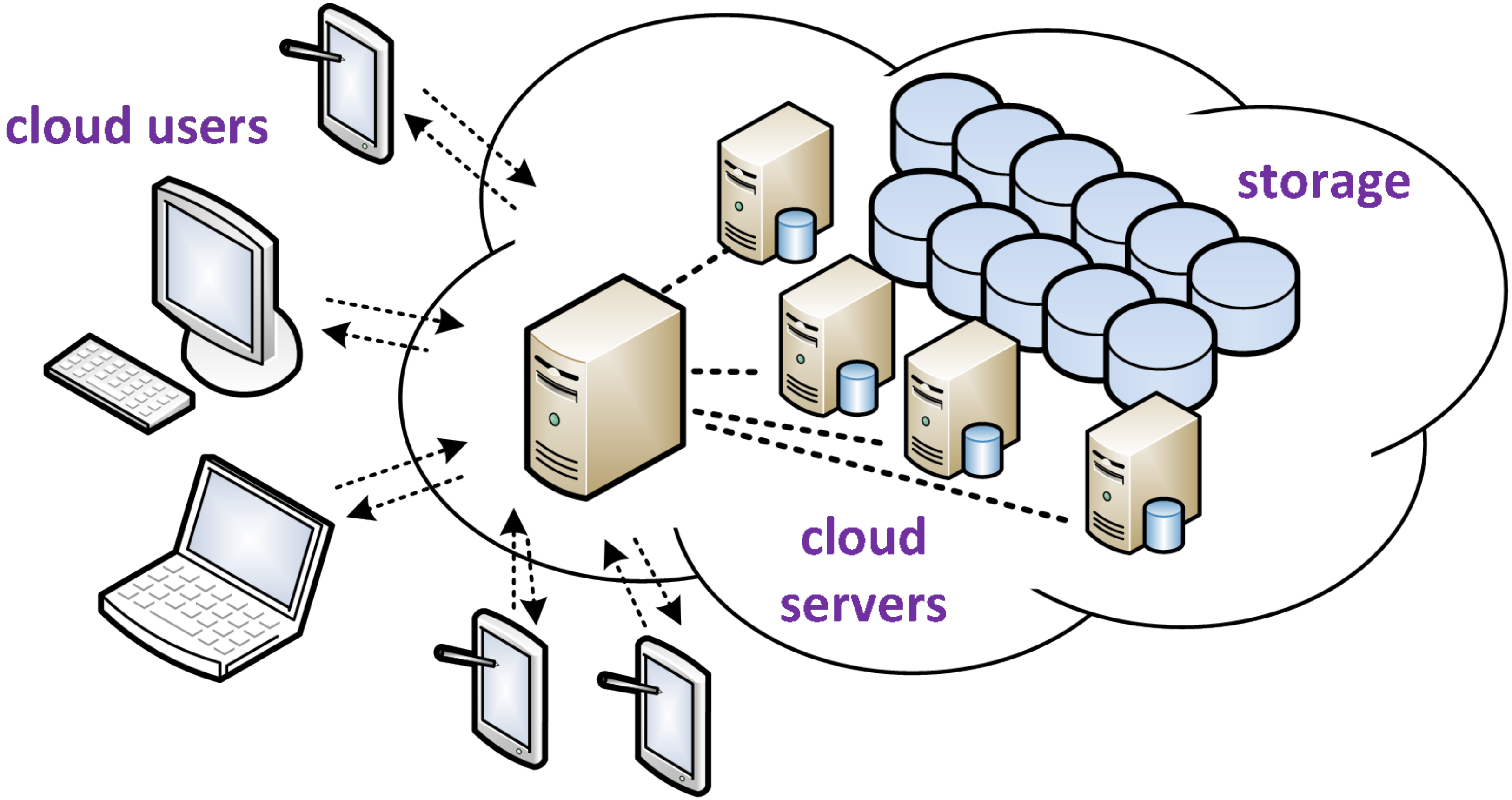
Hsin-Jung Yang, Victor Costan,

Nickolai Zeldovich, and Srini Devadas

Massachusetts Institute of Technology

November 8th, CCSW 2013

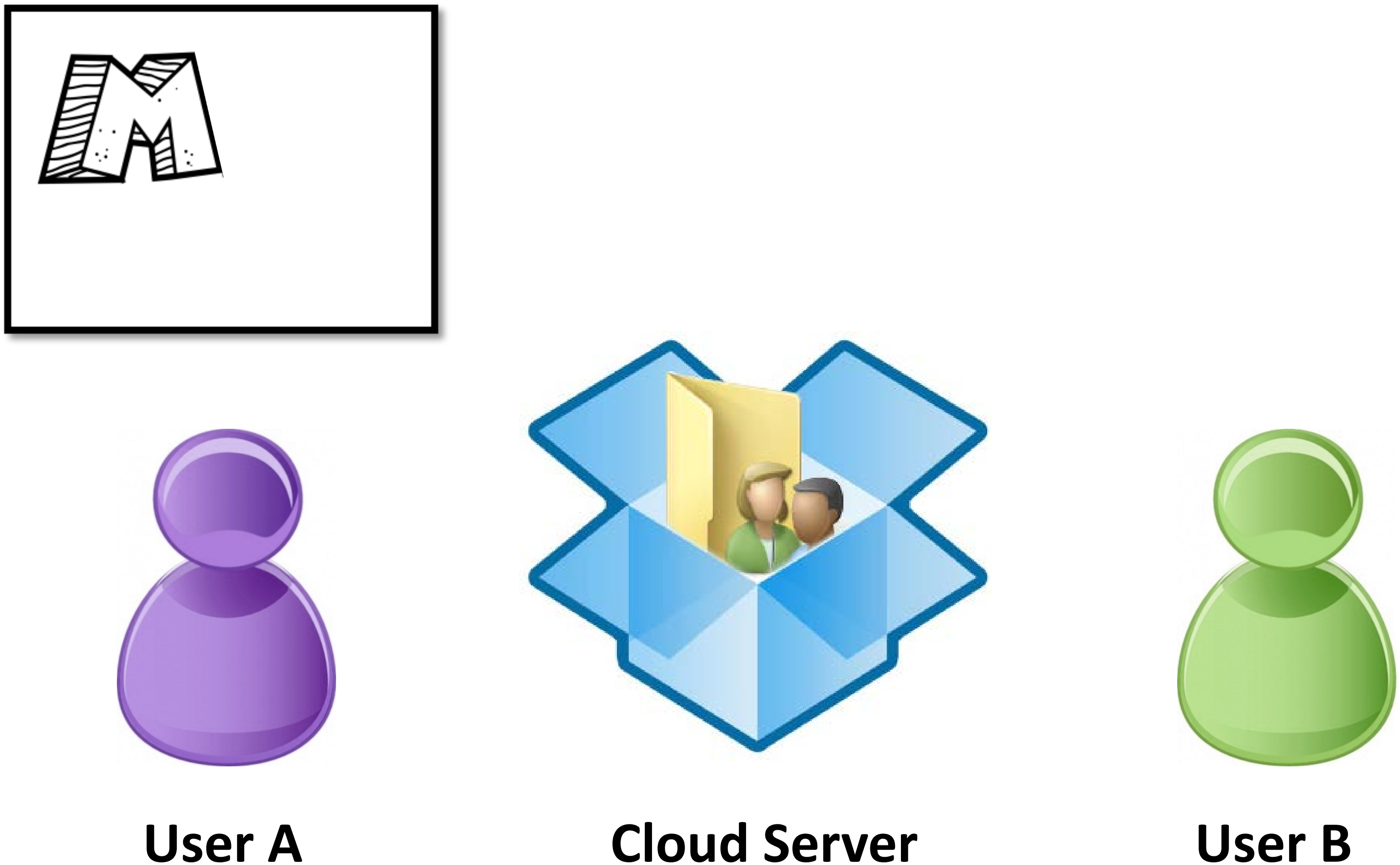
Cloud Storage Model



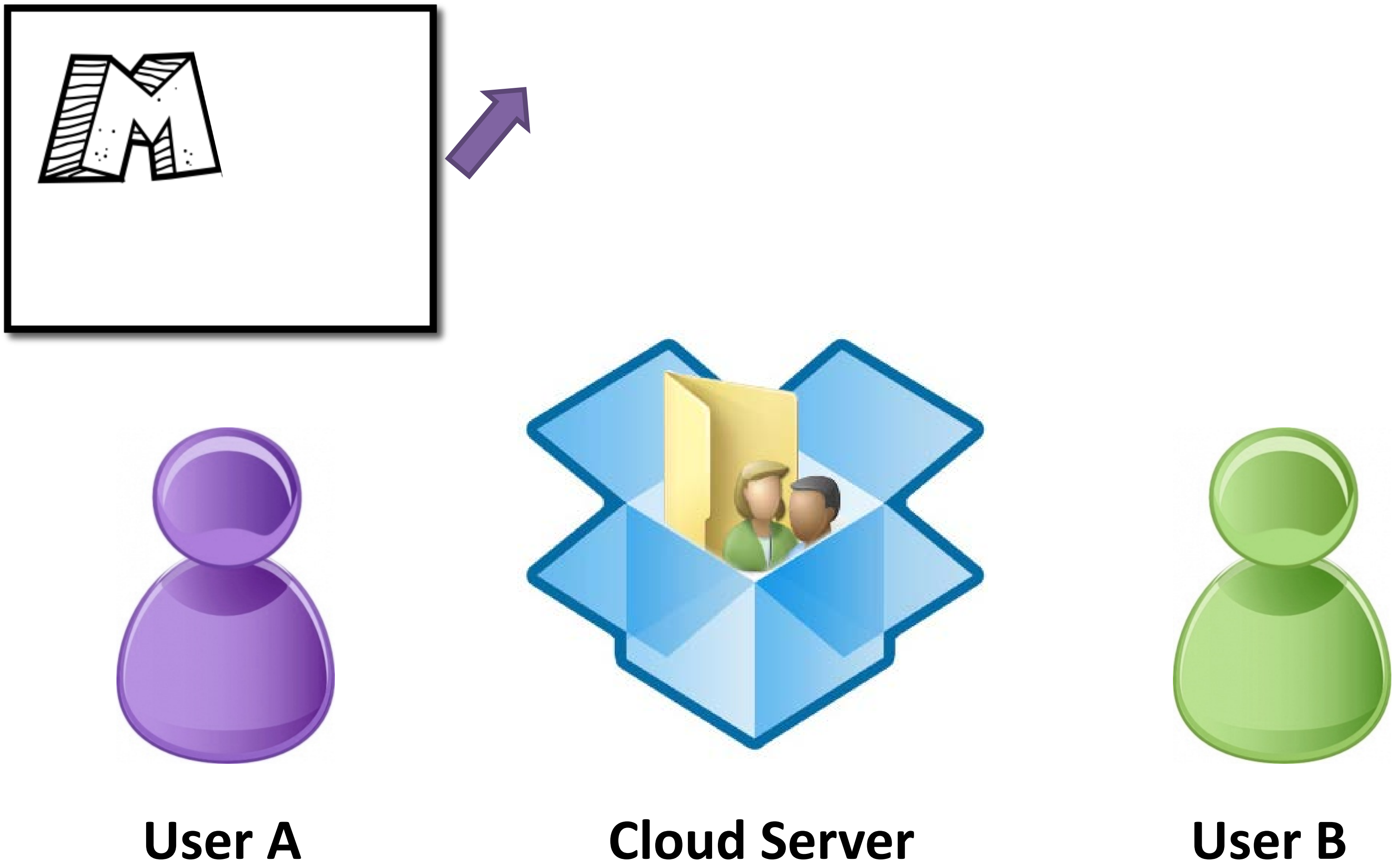
Cloud Storage Requirements

- **Privacy**
  - Sol: encryption at the client side
- **Availability**
  - Sol: appropriate data replication
- **Integrity**
  - Sol: digital signatures & message authentication codes
- **Freshness**
  - Hard to guarantee due to **replay attacks**

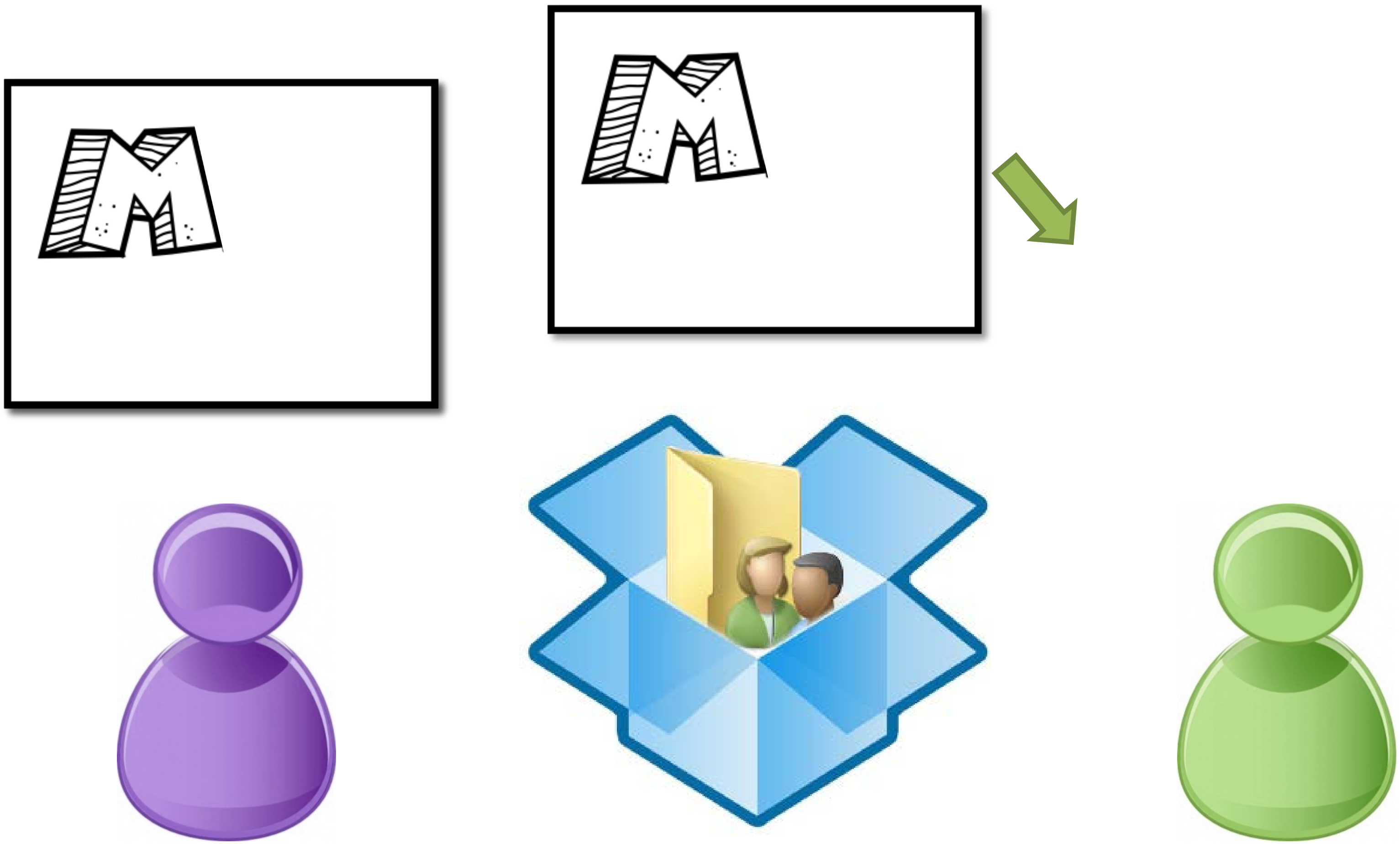
Cloud Storage: Replay Attack



Cloud Storage: Replay Attack

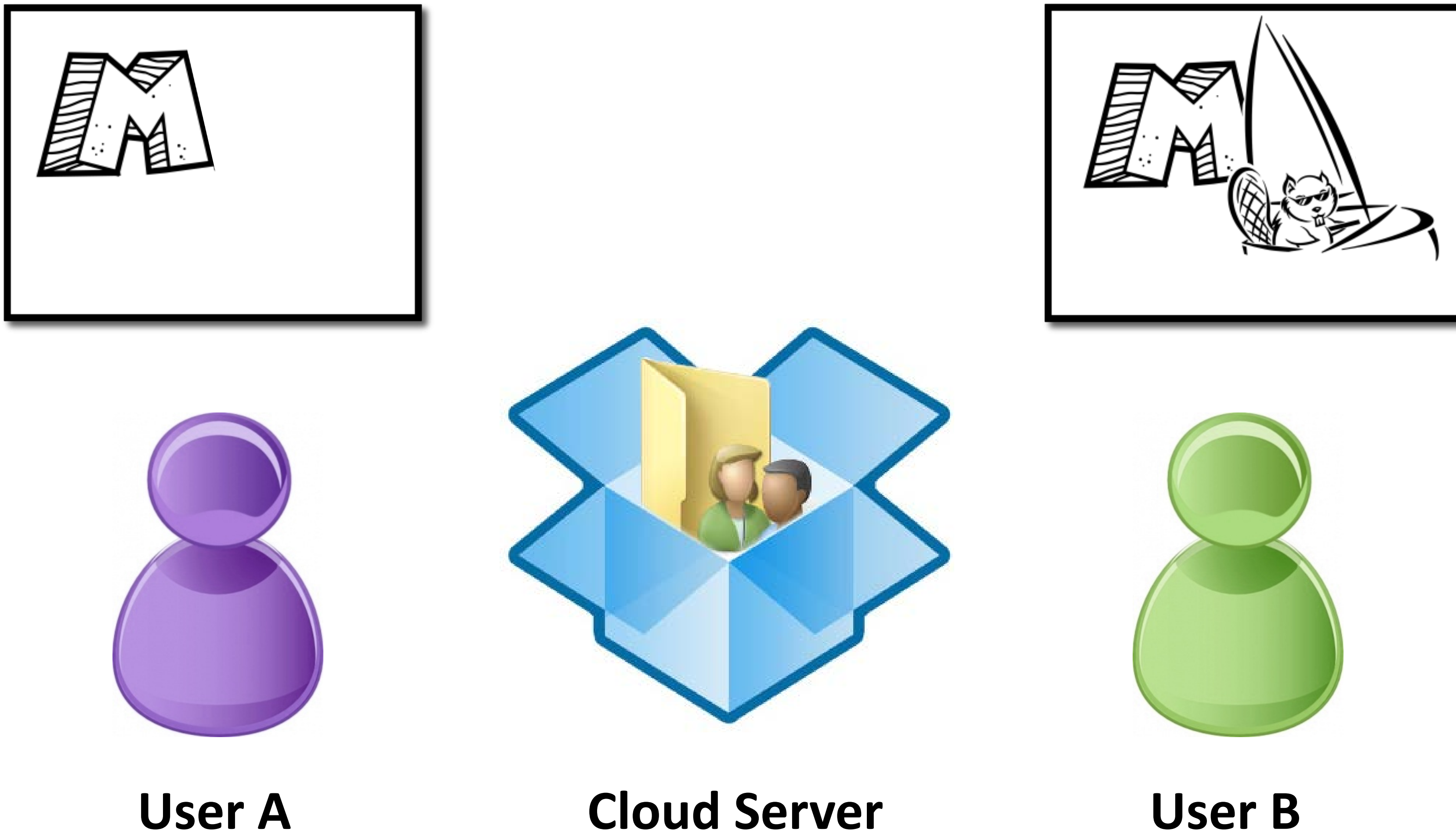


Cloud Storage: Replay Attack

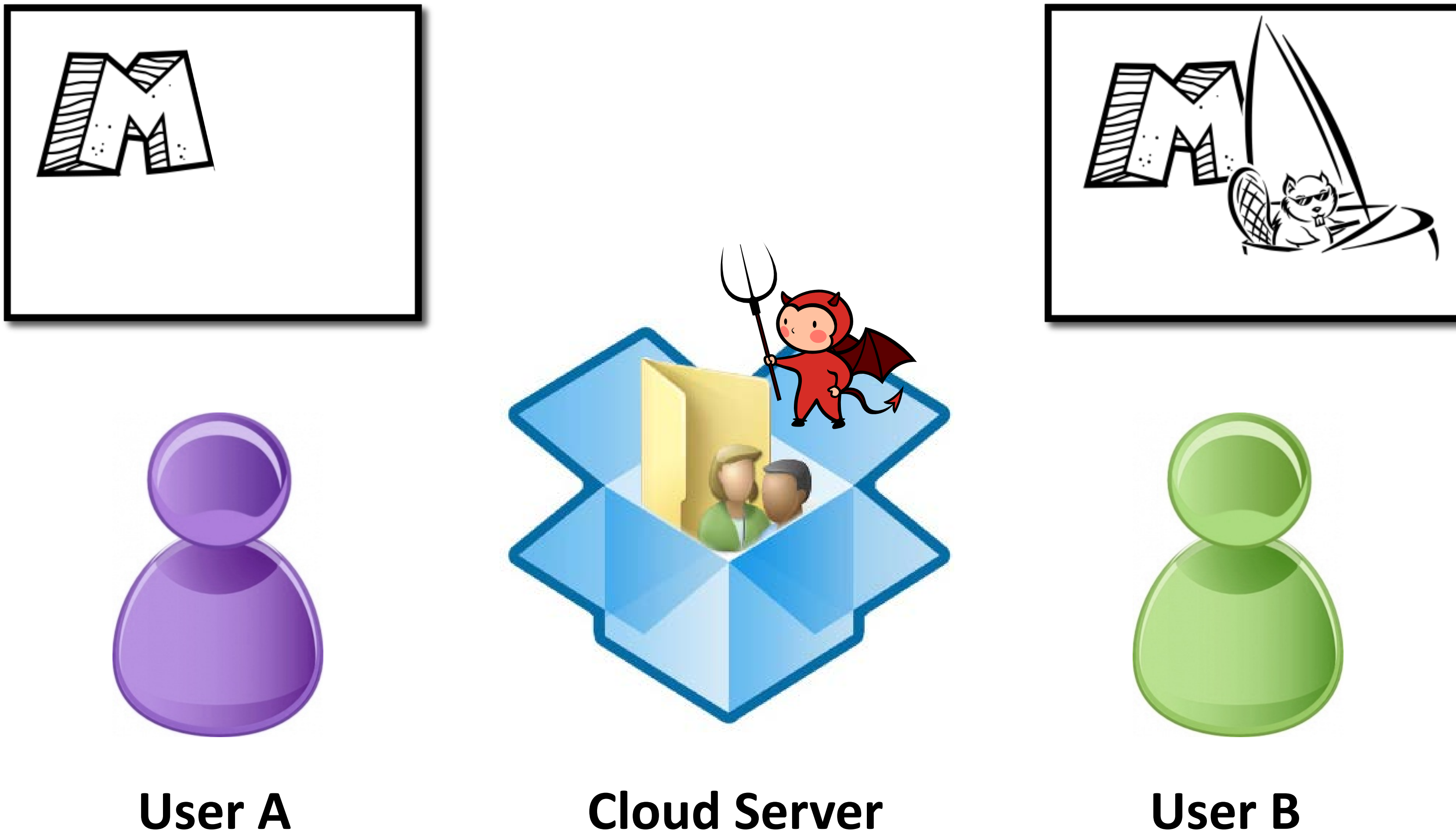


User A   Cloud Server   User B

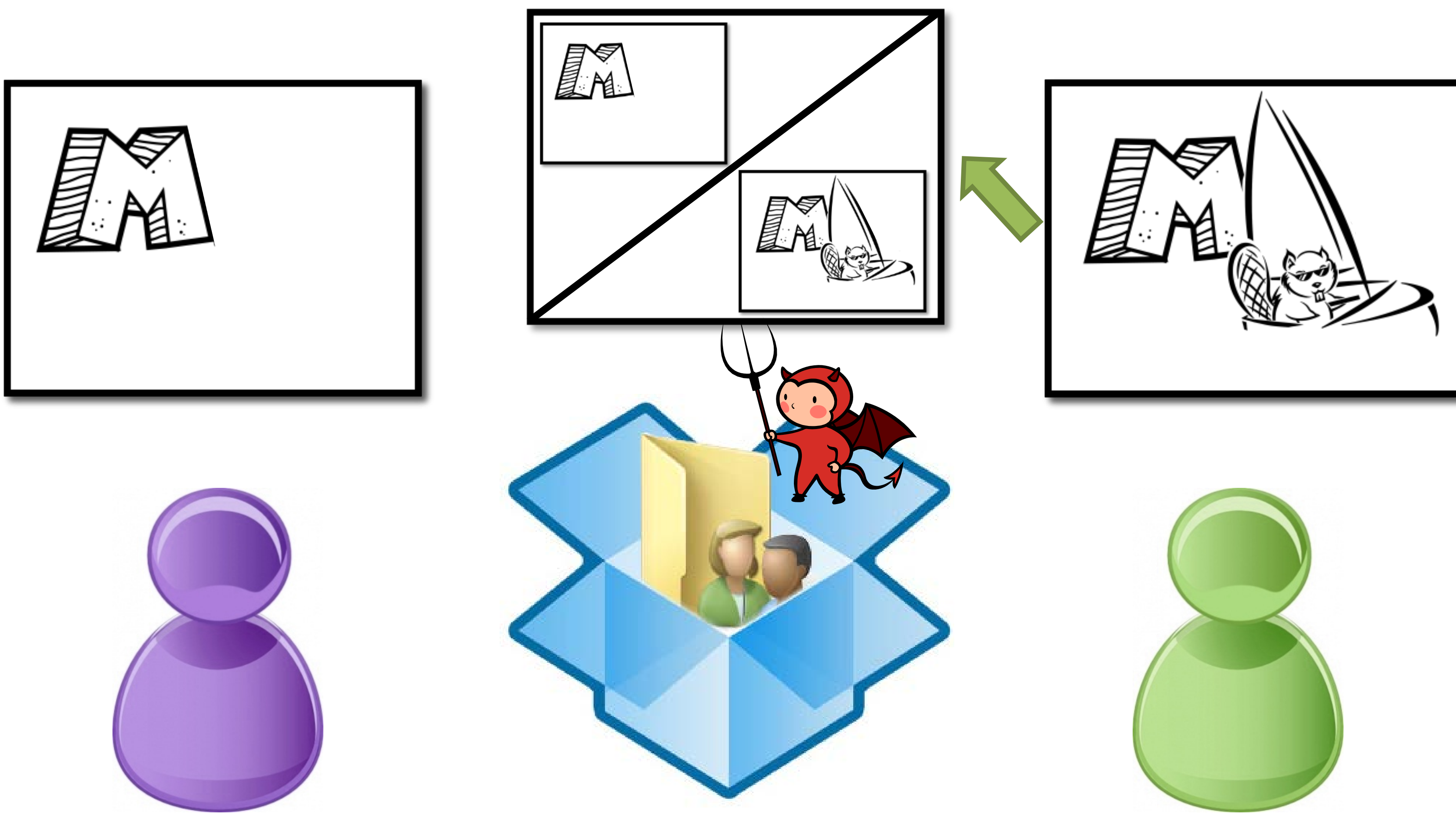
Cloud Storage: Replay Attack



Cloud Storage: Replay Attack

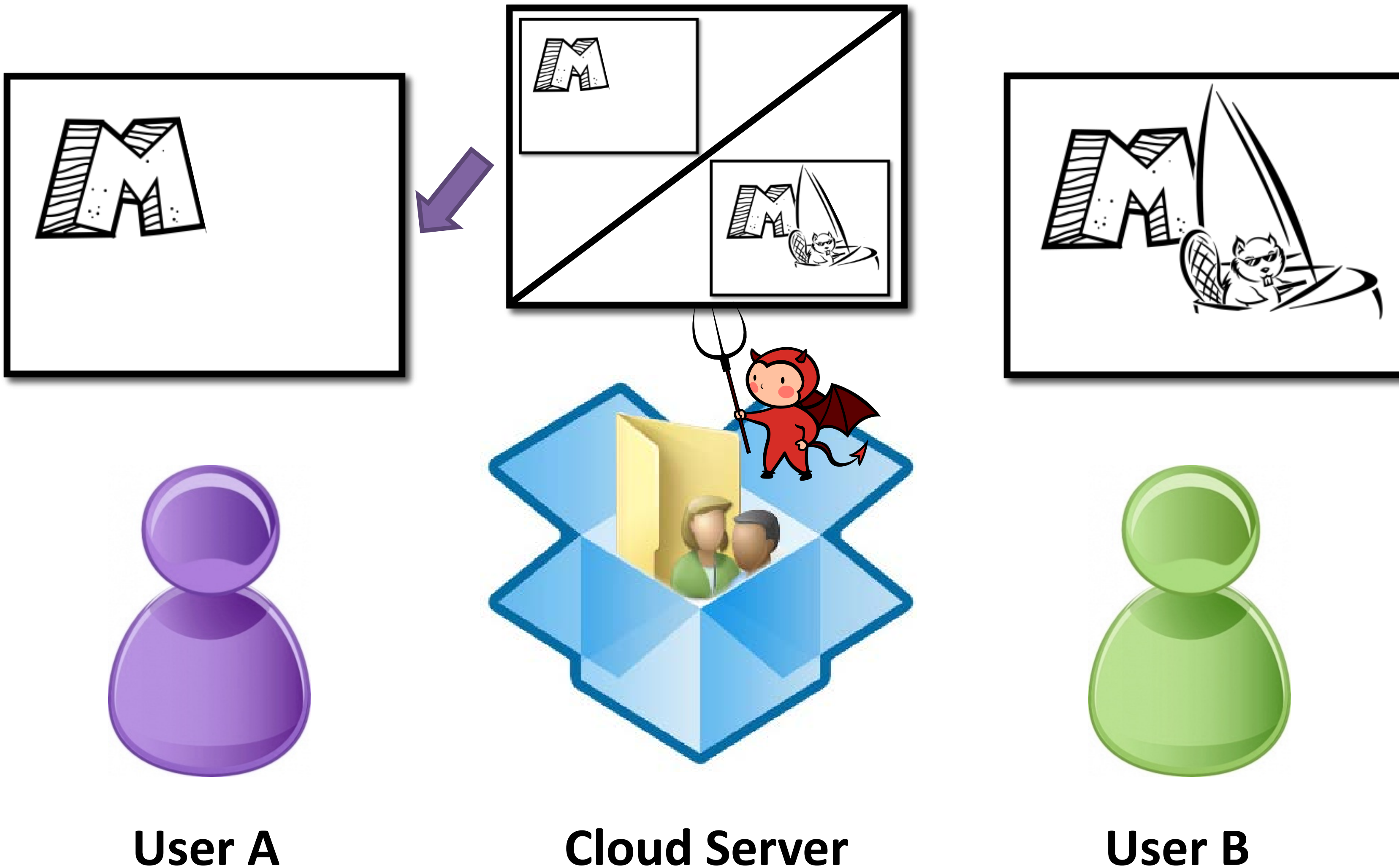


Cloud Storage: Replay Attack

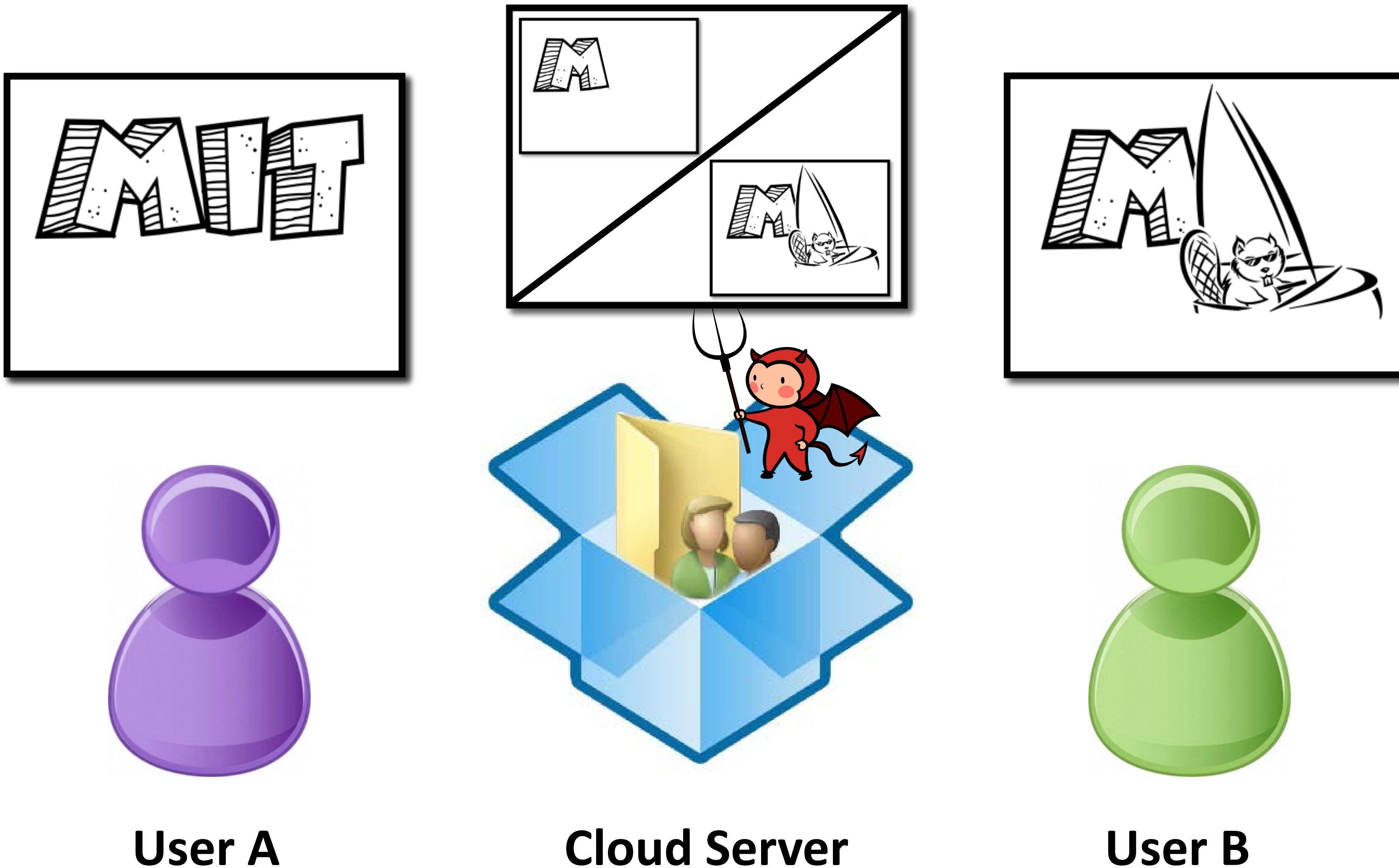


User A **Cloud Server** **User B**

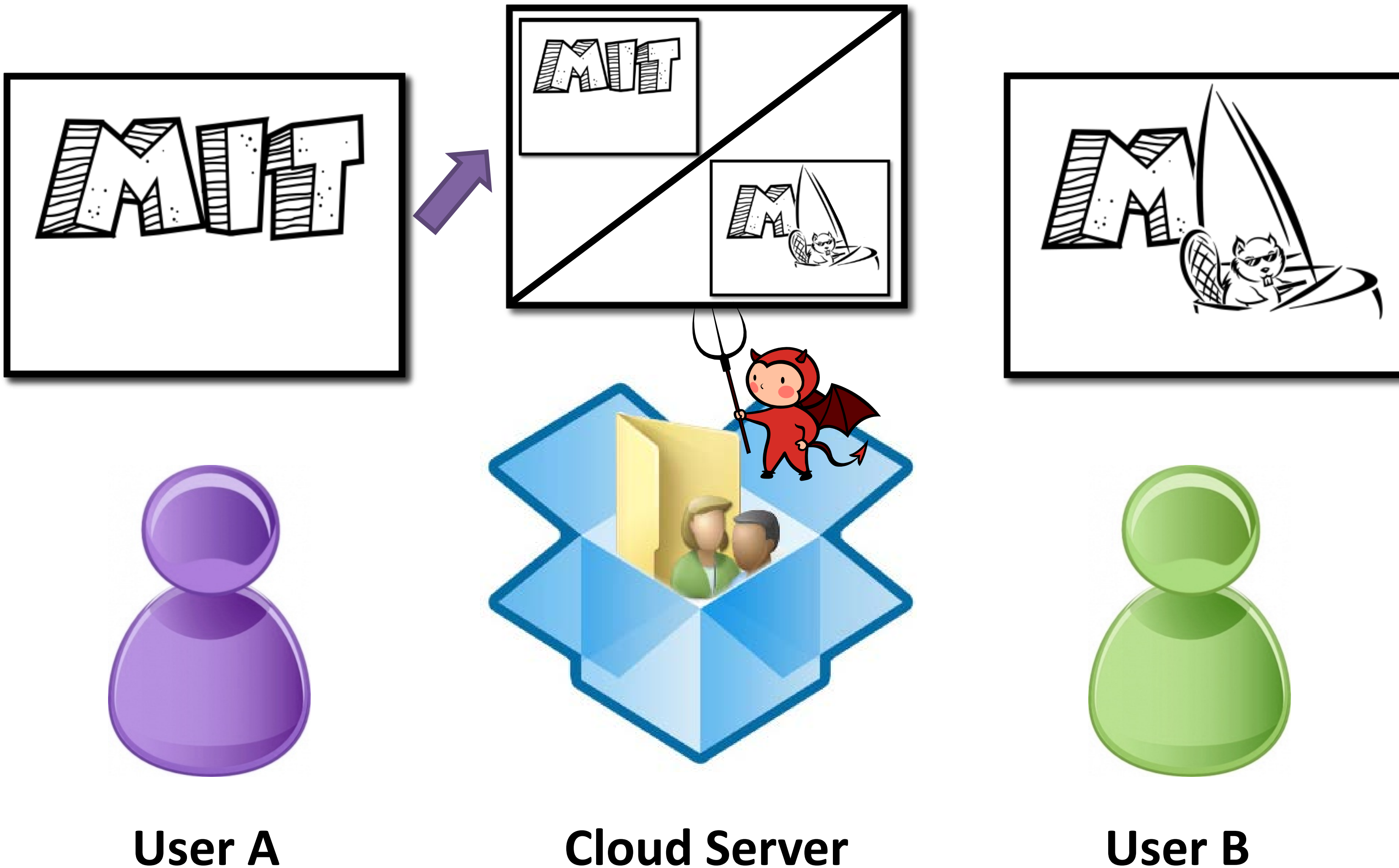
Cloud Storage: Replay Attack



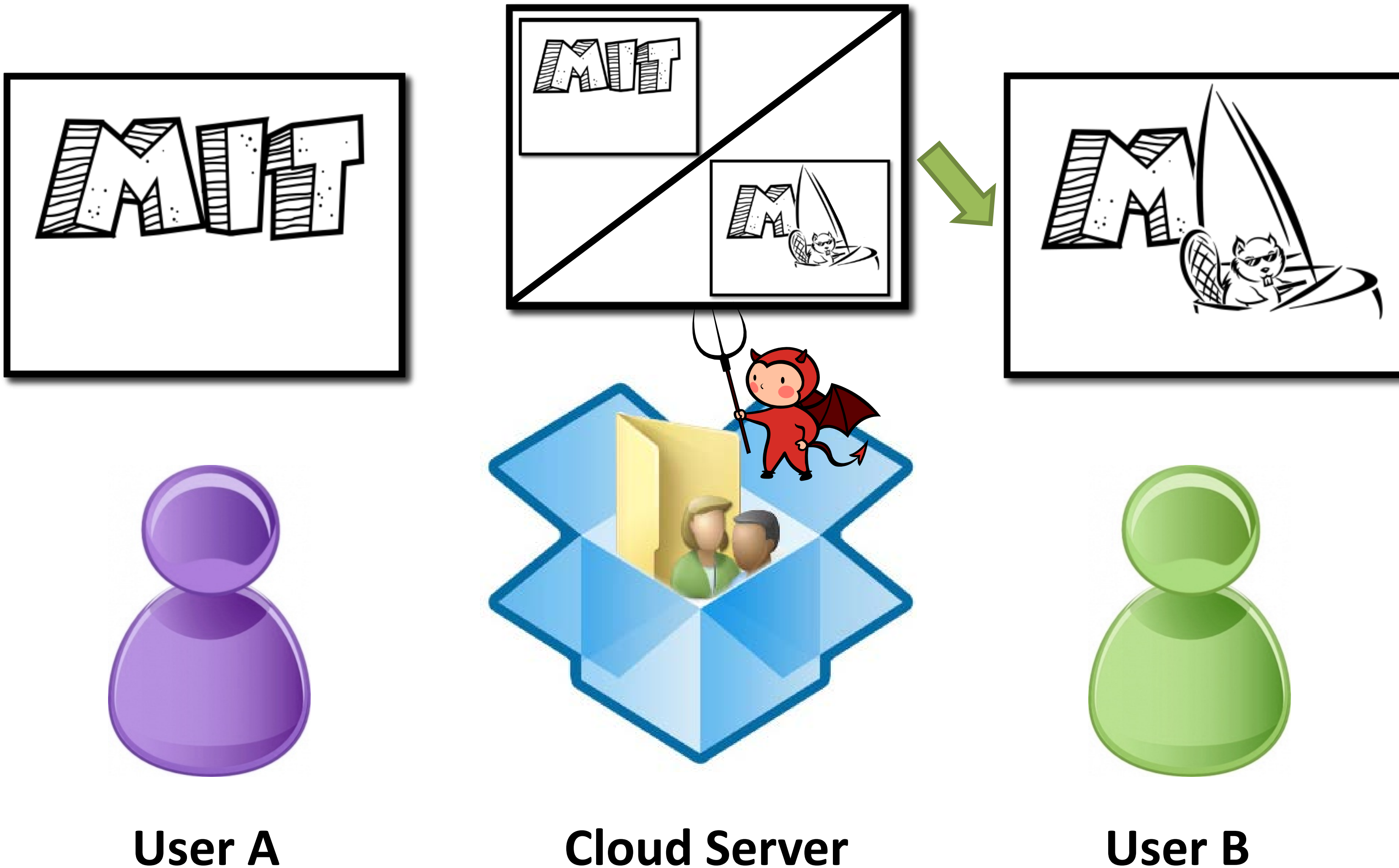
Cloud Storage: Replay Attack



Cloud Storage: Replay Attack

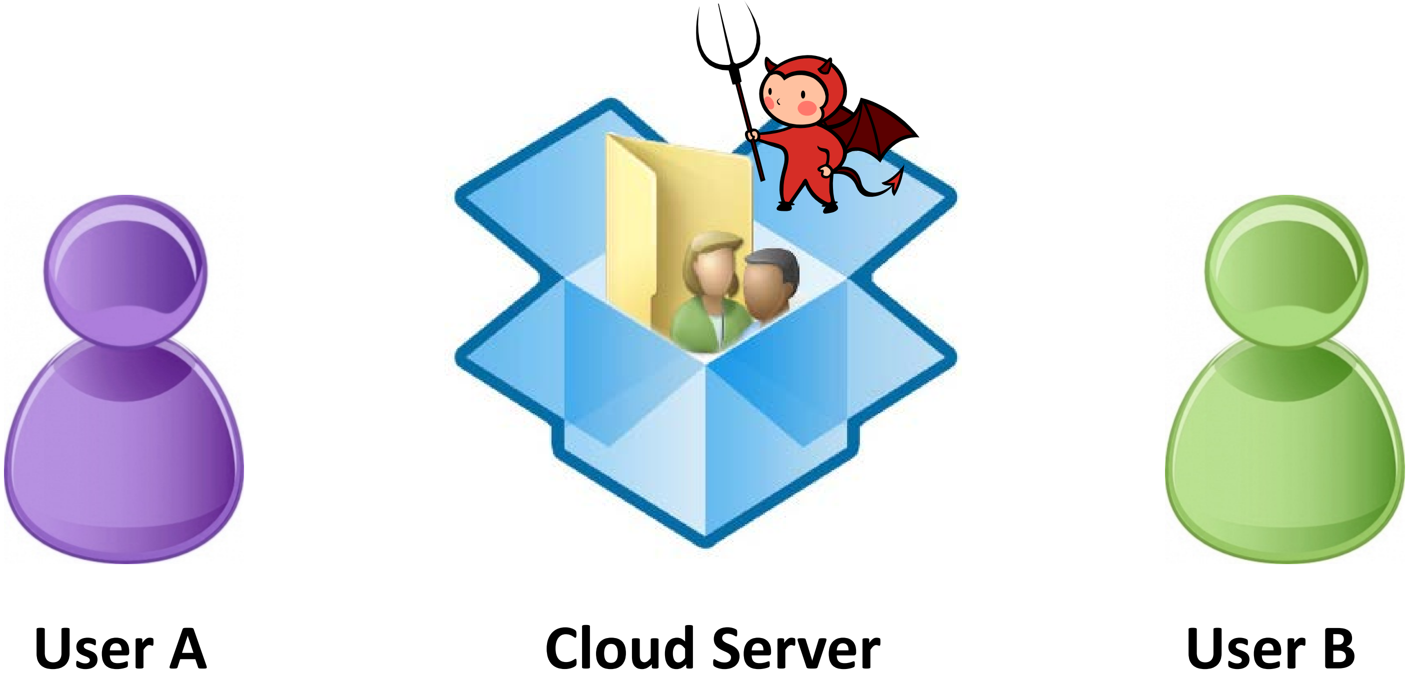


Cloud Storage: Replay Attack

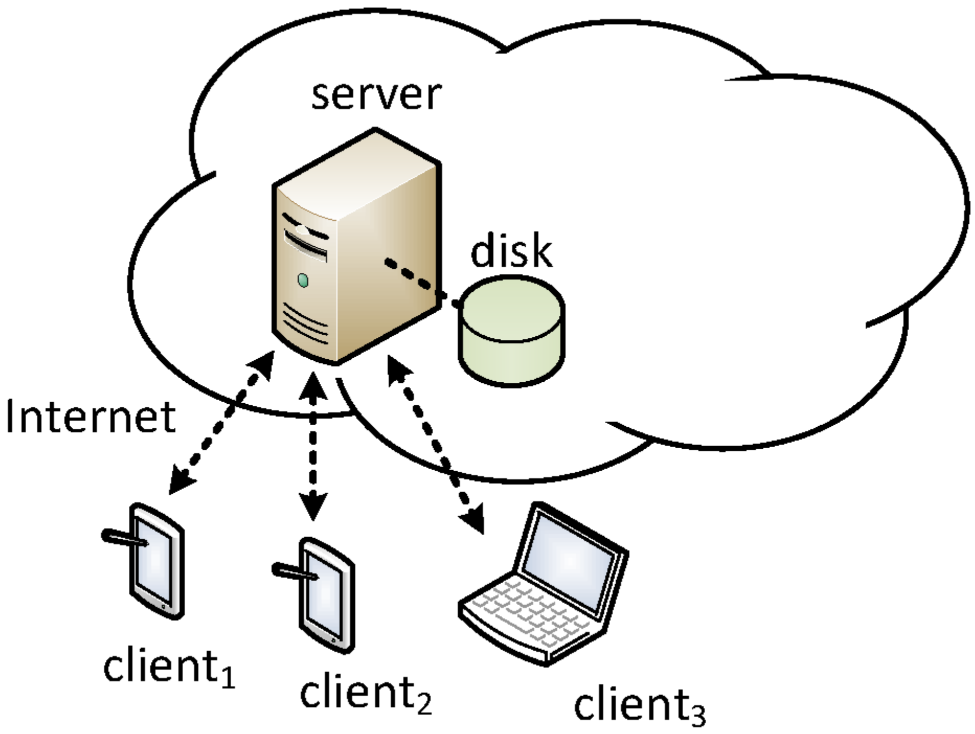


Cloud Storage: Replay Attack

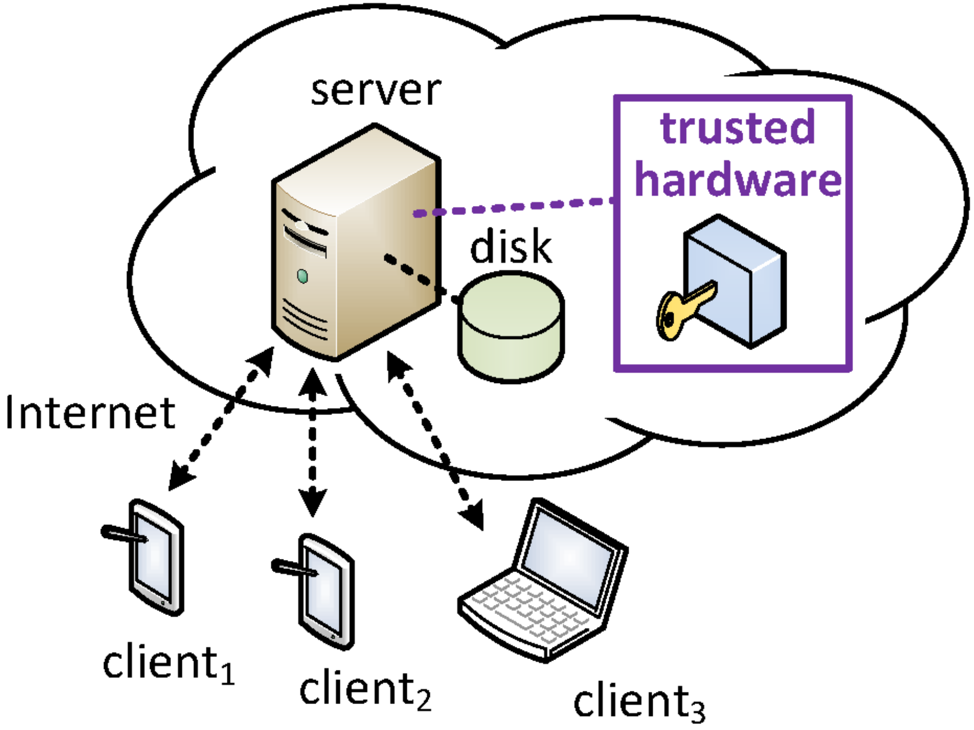
Software solution:
Two users contact with each other directly



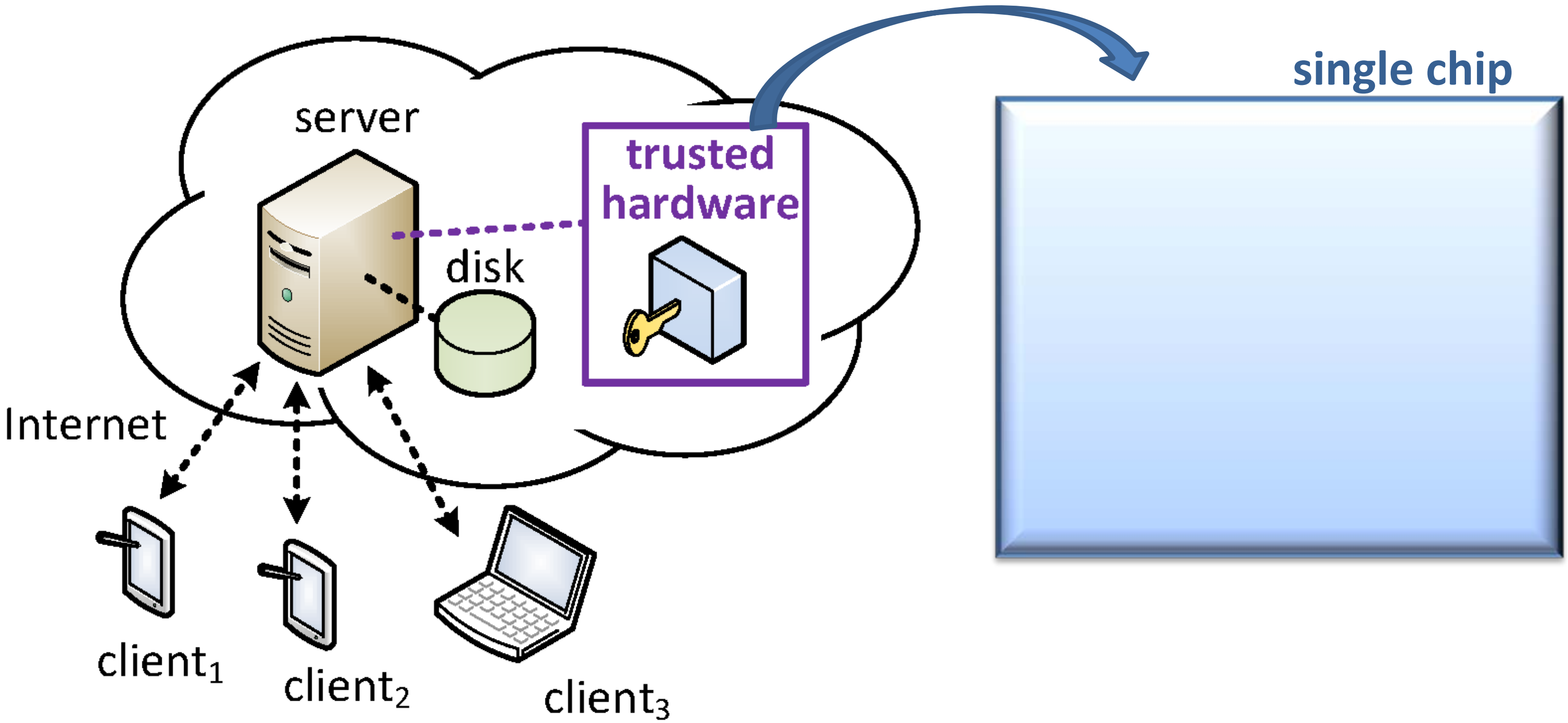
Solution: Adding Trusted Hardware



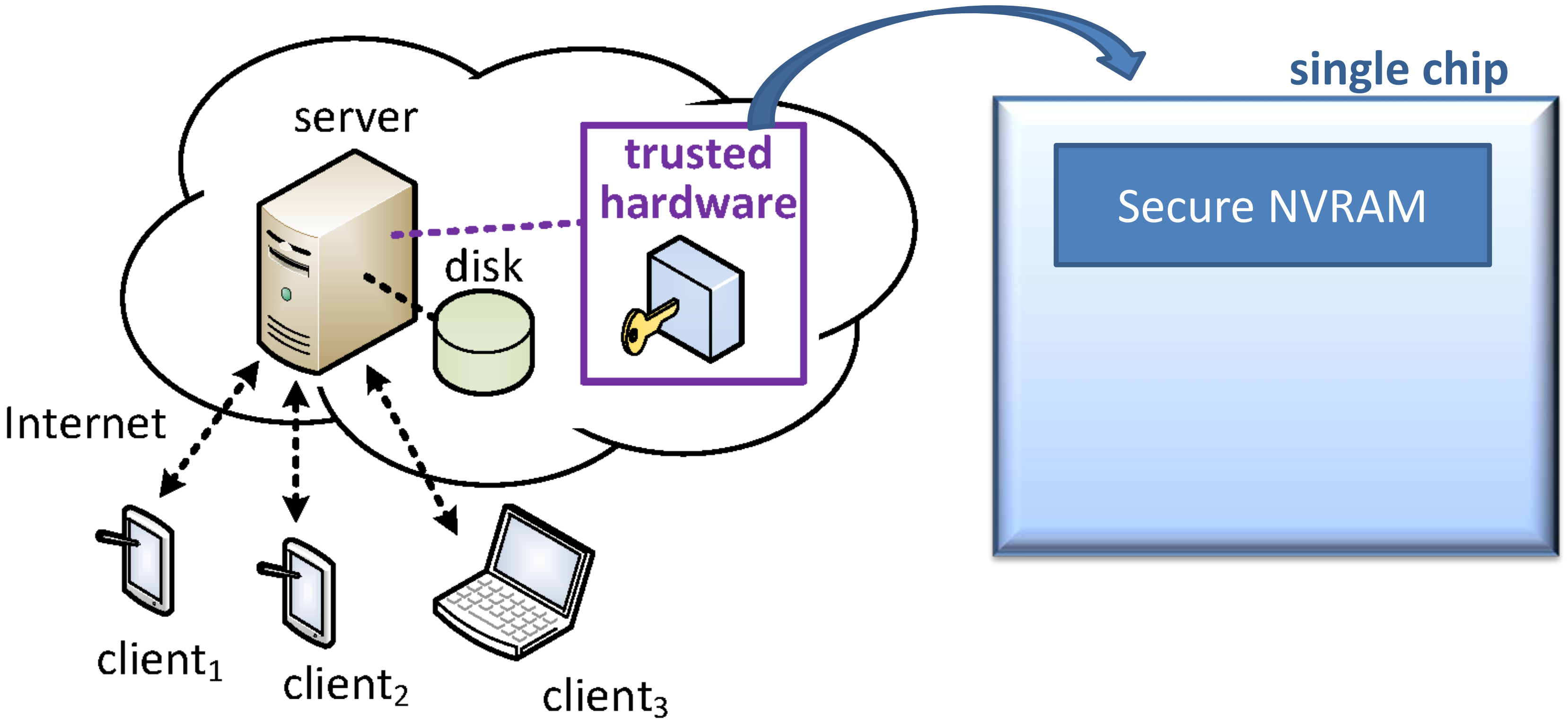
Solution: Adding Trusted Hardware



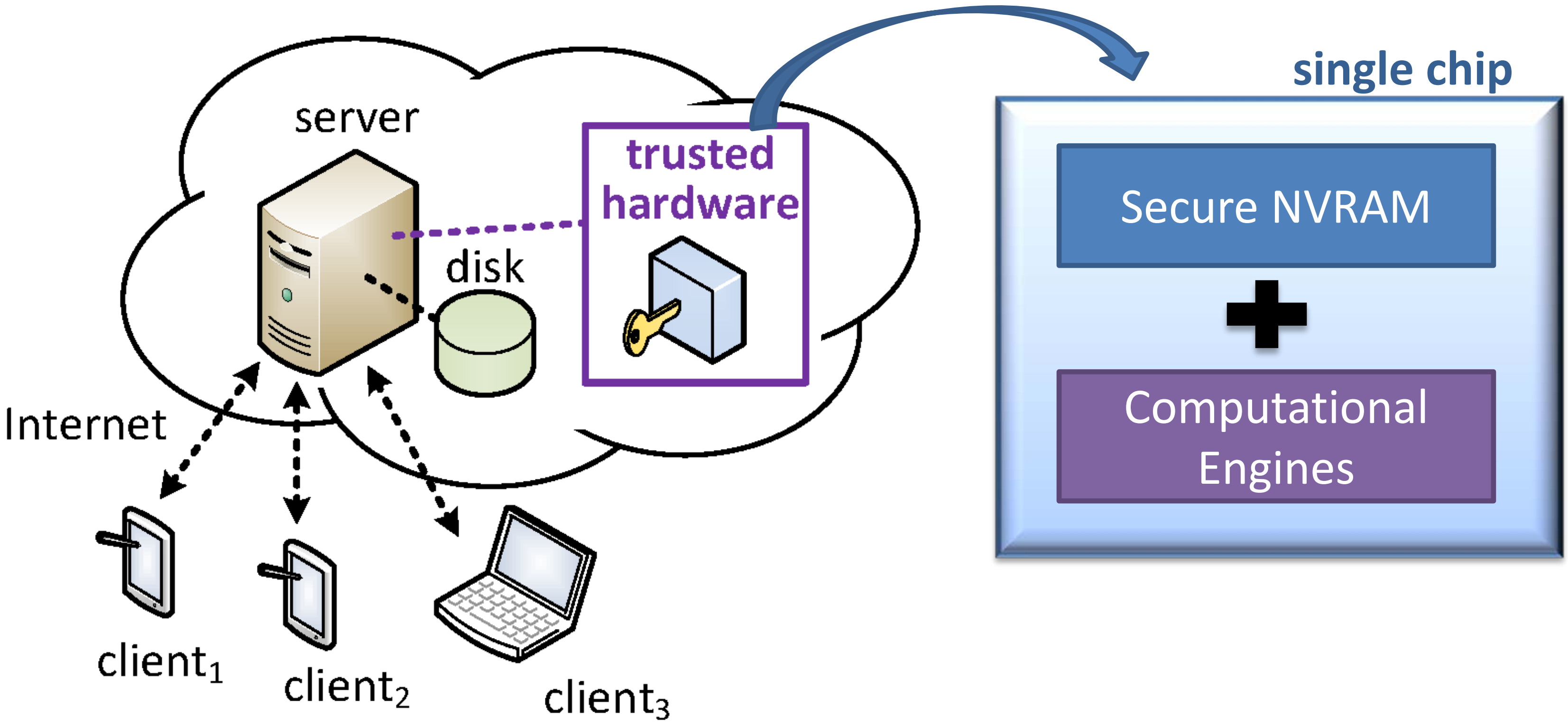
Solution: Adding Trusted Hardware



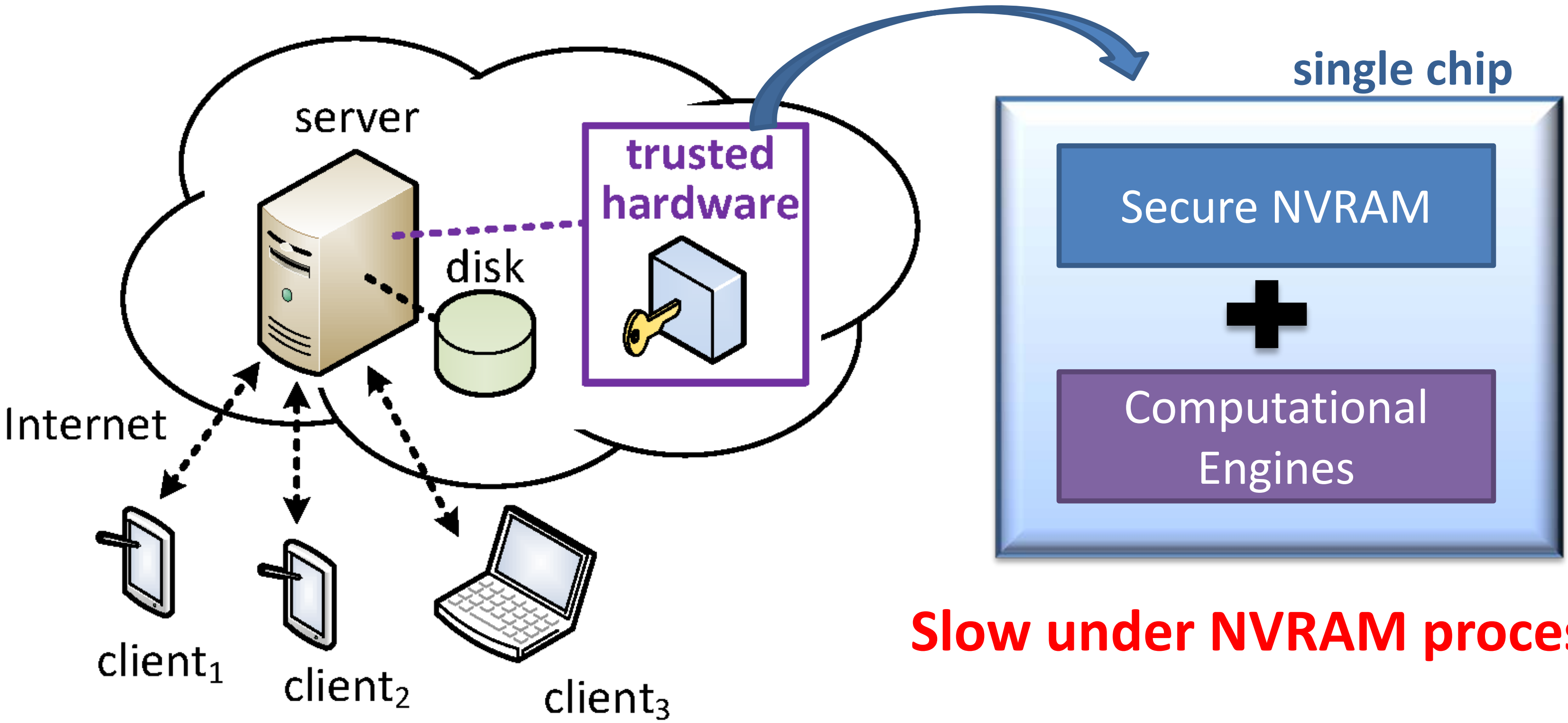
Solution: Adding Trusted Hardware



Solution: Adding Trusted Hardware

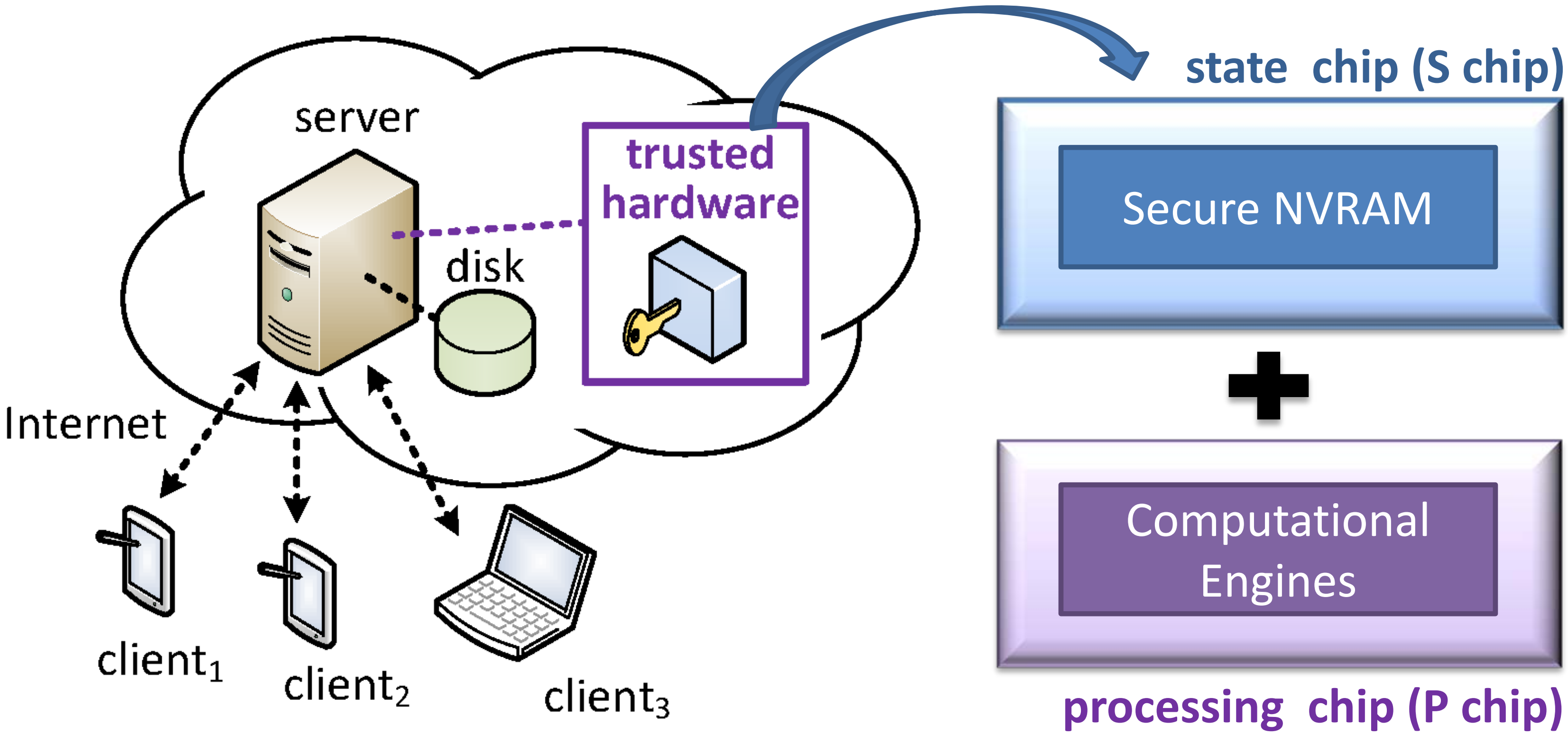


Solution: Adding Trusted Hardware

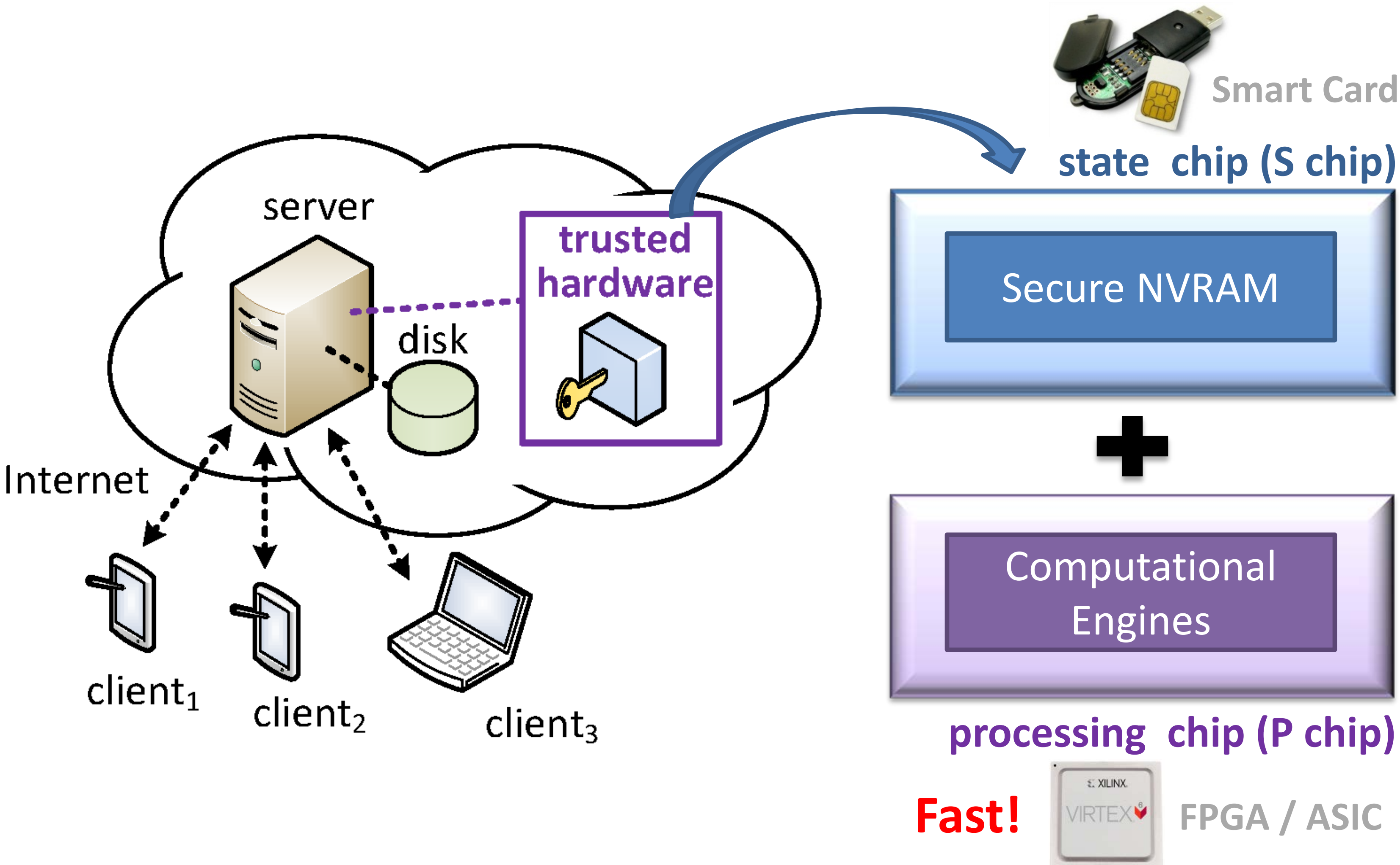


Slow under NVRAM process!

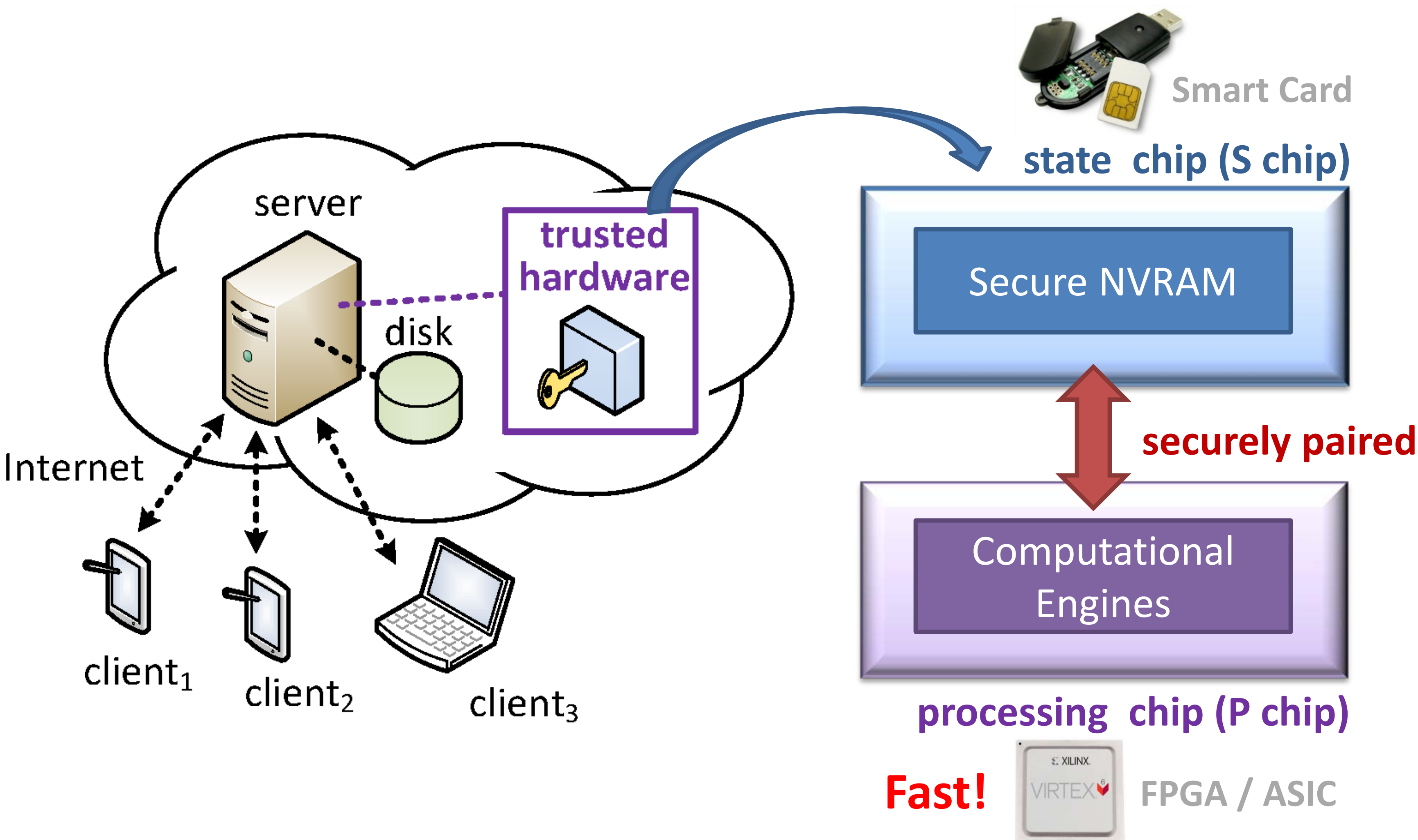
Solution: Adding Trusted Hardware



Solution: Adding Trusted Hardware

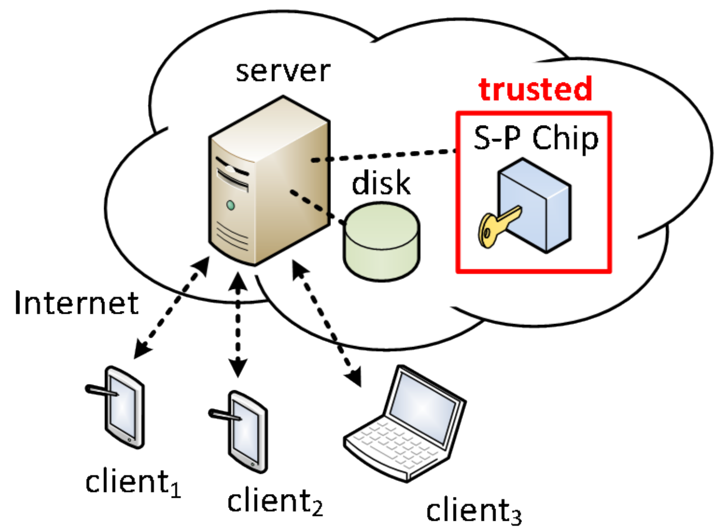


Solution: Adding Trusted Hardware

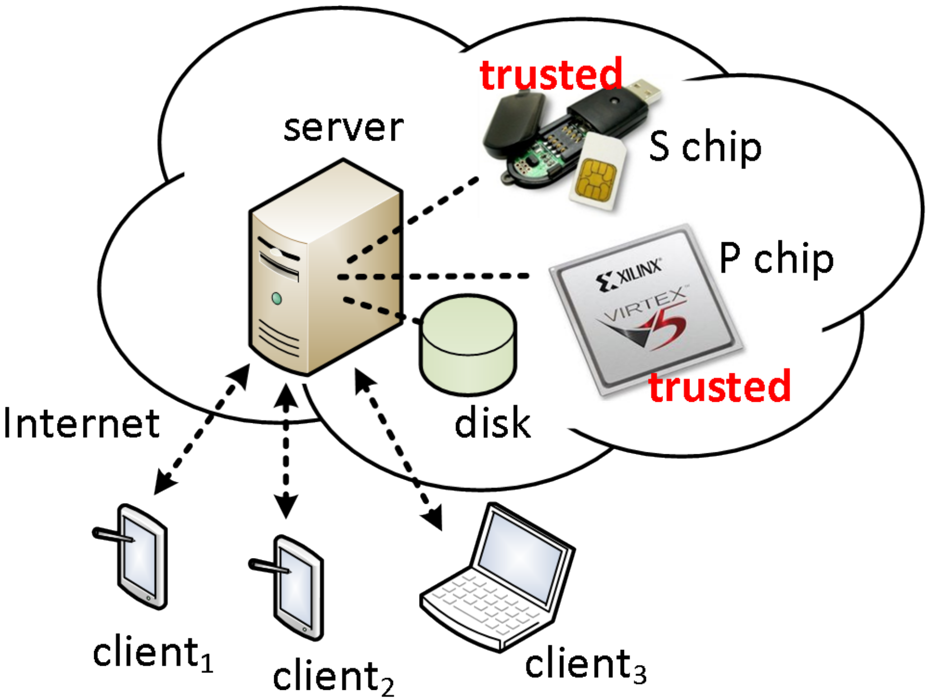


Outline

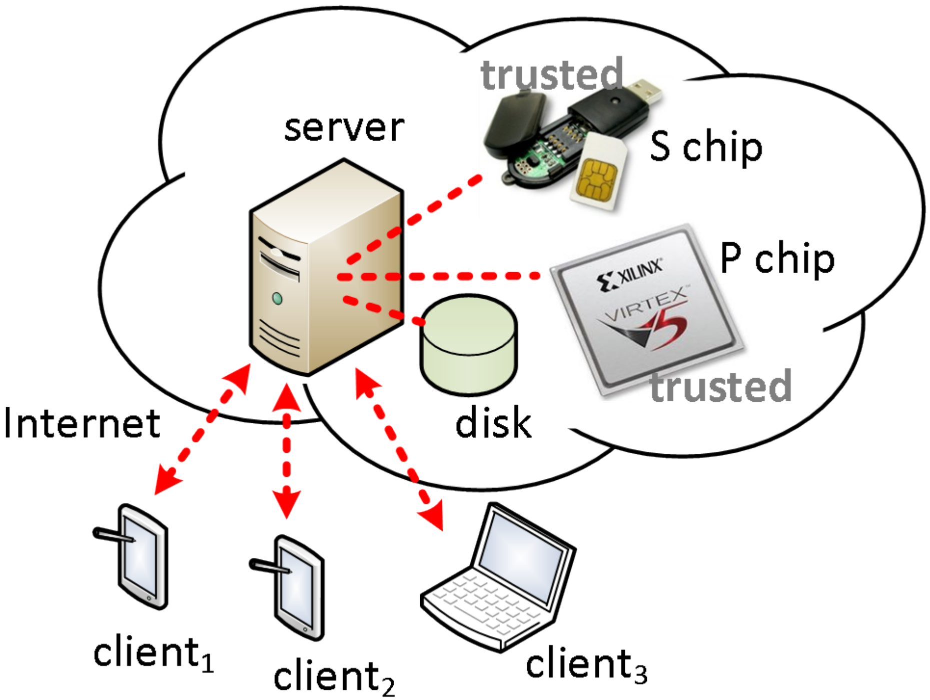
- Motivation: Cloud Storage and Security Challenges
- **System Design**
  - Threat Model & System Overview
  - Security Protocols
  - Crash Recovery Mechanism
- **Implementation**
- **Evaluation**
- **Conclusion**



Threat Model

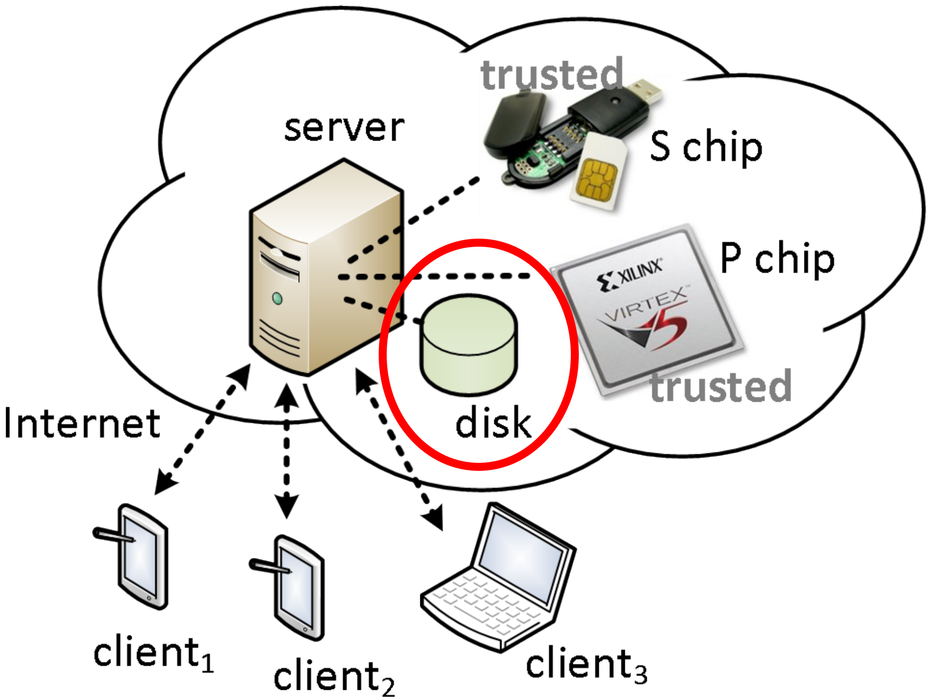


Threat Model



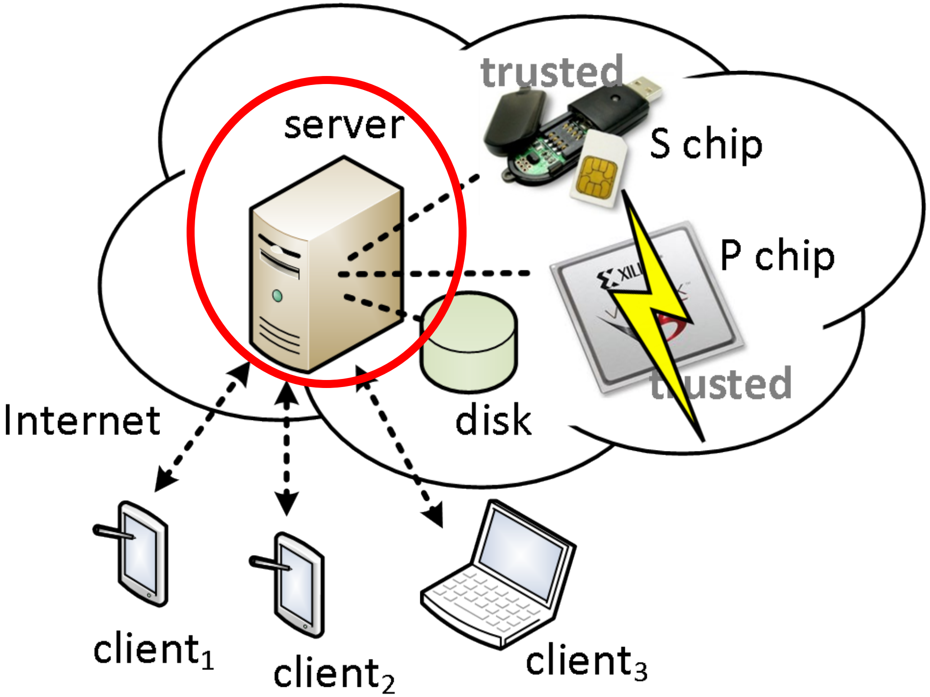
- Untrusted connections
- Disk attacks and hardware failures
- Untrusted server that may
  (1) send wrong response
  (2) pretend to be a client
  (3) maliciously crash
  (4) disrupt P chip’s power
- Clients may try to modify other’s data

Threat Model



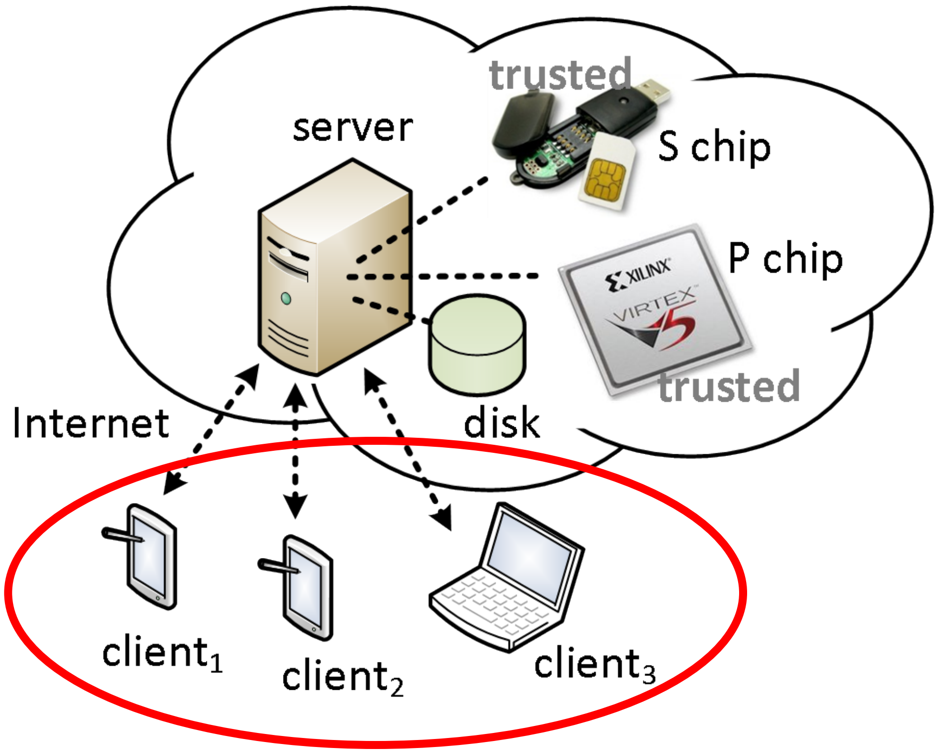
- Untrusted connections
- Disk attacks and hardware failures
- Untrusted server that may
  (1) send wrong response
  (2) pretend to be a client
  (3) maliciously crash
  (4) disrupt P chip’s power
- Clients may try to modify other’s data

Threat Model



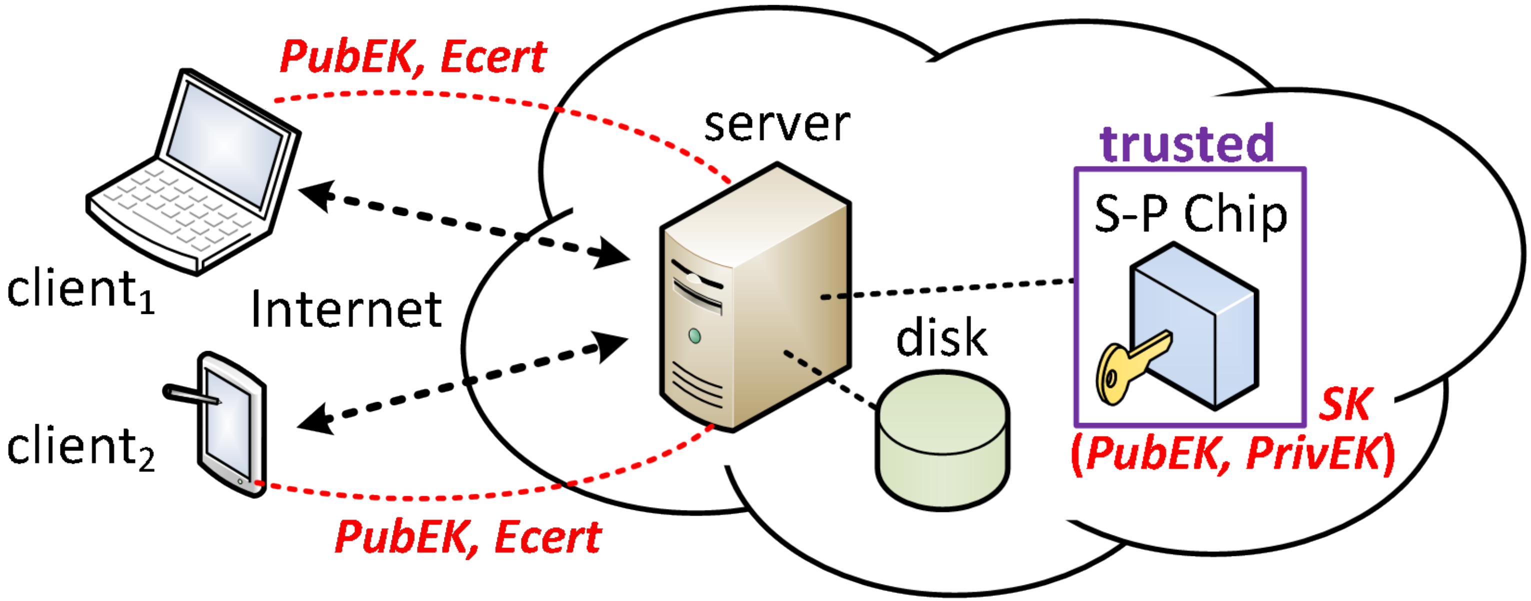
- Untrusted connections
- Disk attacks and hardware failures
- Untrusted server that may
  (1) send wrong response
  (2) pretend to be a client
  (3) maliciously crash
  (4) disrupt P chip’s power
- Clients may try to modify other’s data

Threat Model



- Untrusted connections
- Disk attacks and hardware failures
- Untrusted server that may
  (1) send wrong response
  (2) pretend to be a client
  (3) maliciously crash
  (4) disrupt P chip's power
- Clients may try to modify other's data

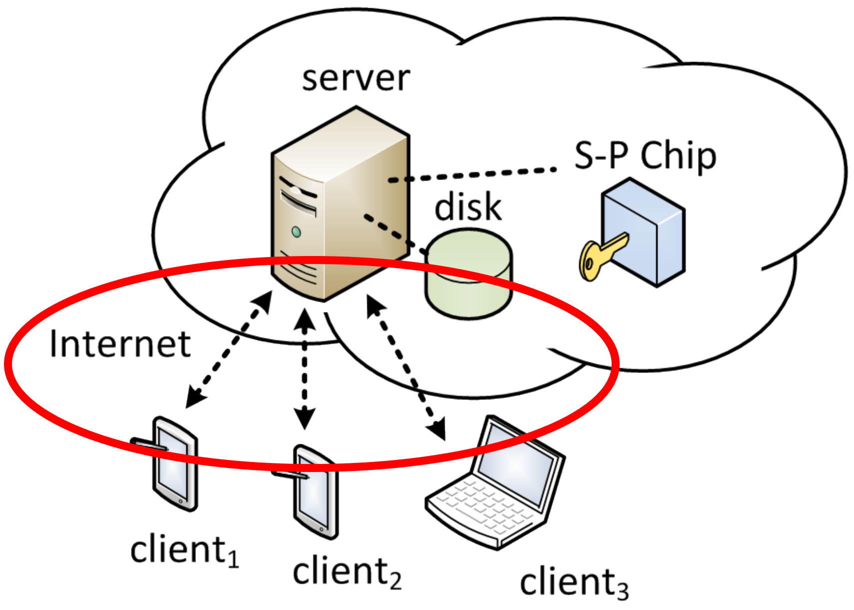
System Overview



- **Client <-> S-P chip:** HMAC key
- **S-P chip:** integrity/freshness checks,
  system state storage & updates
  sign responses
- **Server:** communication, scheduling, disk IO

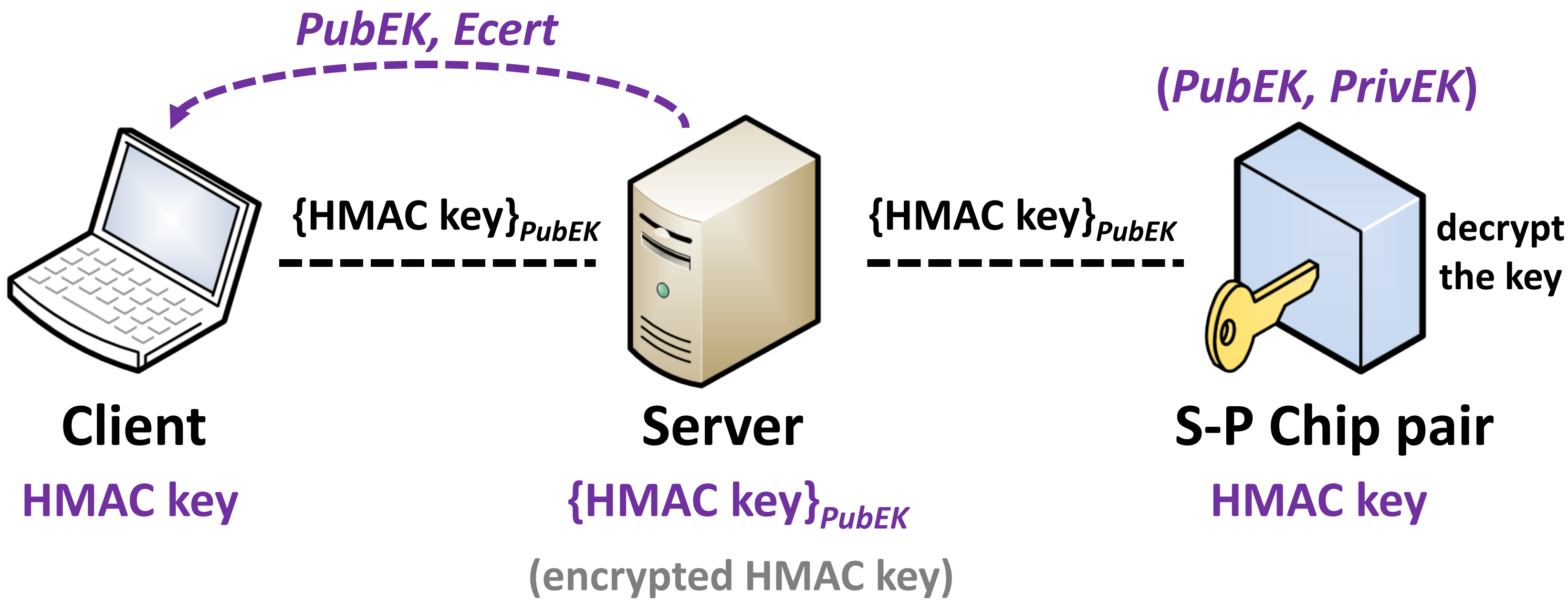
Security Protocols

- **Message Authentication**
- Memory Authentication
- Write Access Control
- System State Protection against Power Loss



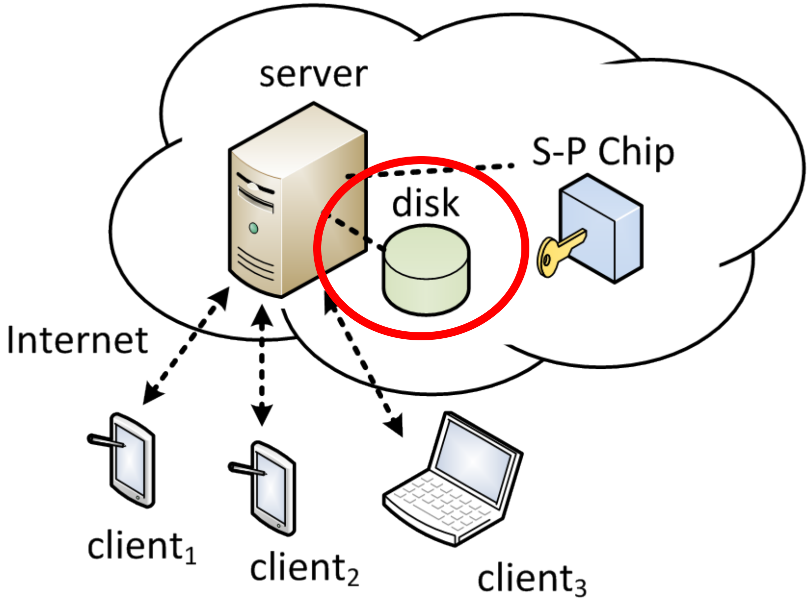
Design: Message Authentication

- **Untrusted network between client and server**
  - **Sol: HMAC Technique**
- **Session-based protocol (HMAC key)**



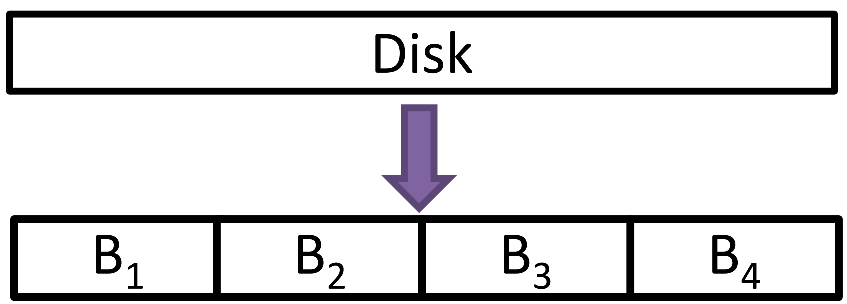
Security Protocols

- Message Authentication
- **Memory Authentication**
- Write Access Control
- System State Protection against Power Loss



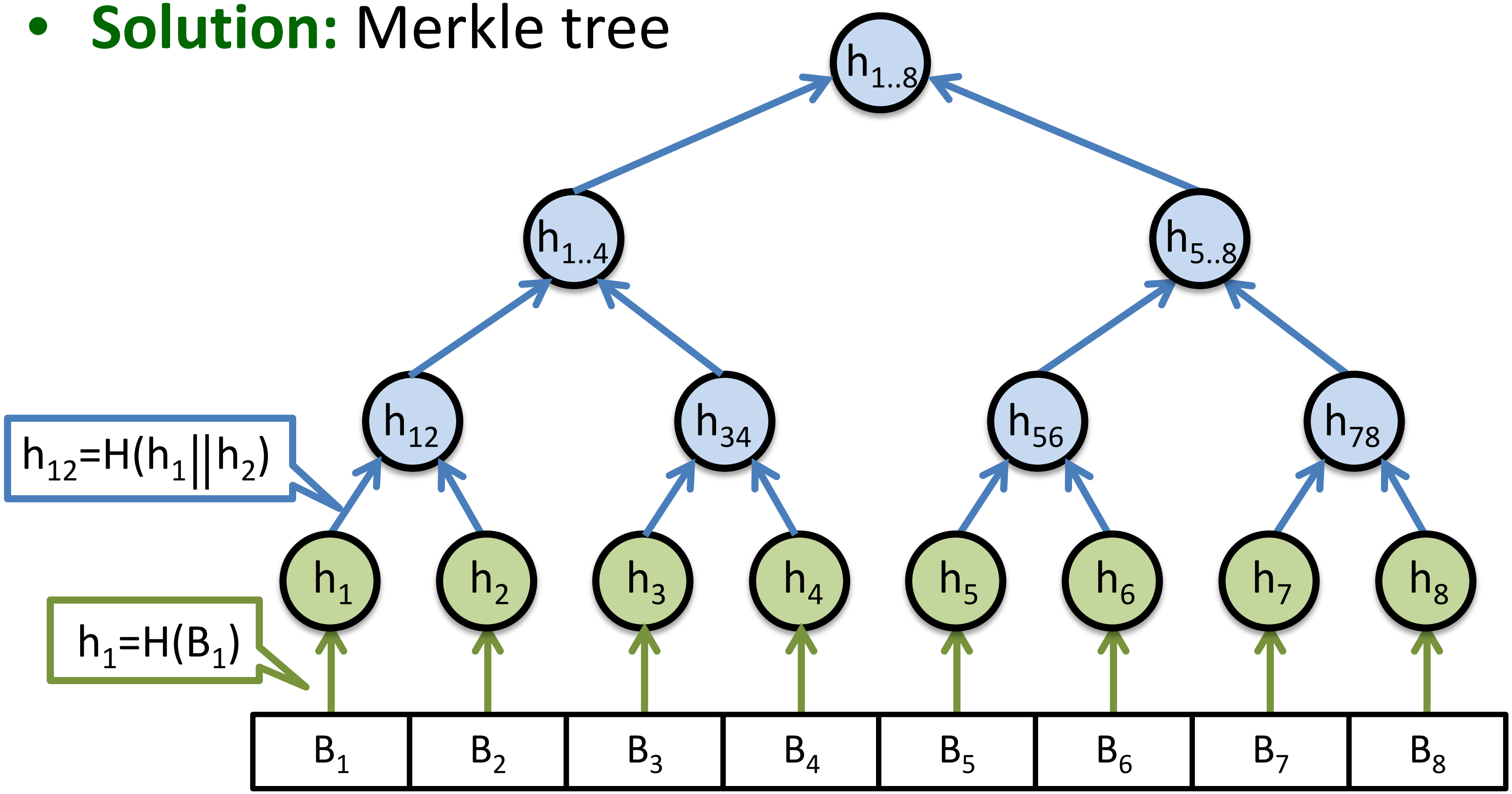
Design: Memory Authentication

- **Data protection against untrusted disk**
- **Block-based cloud storage API**
  - Fixed block size (1MB)
  - Write (block number, block)
  - Read (block number) → block
  - Easy to reason about the security



Design: Memory Authentication

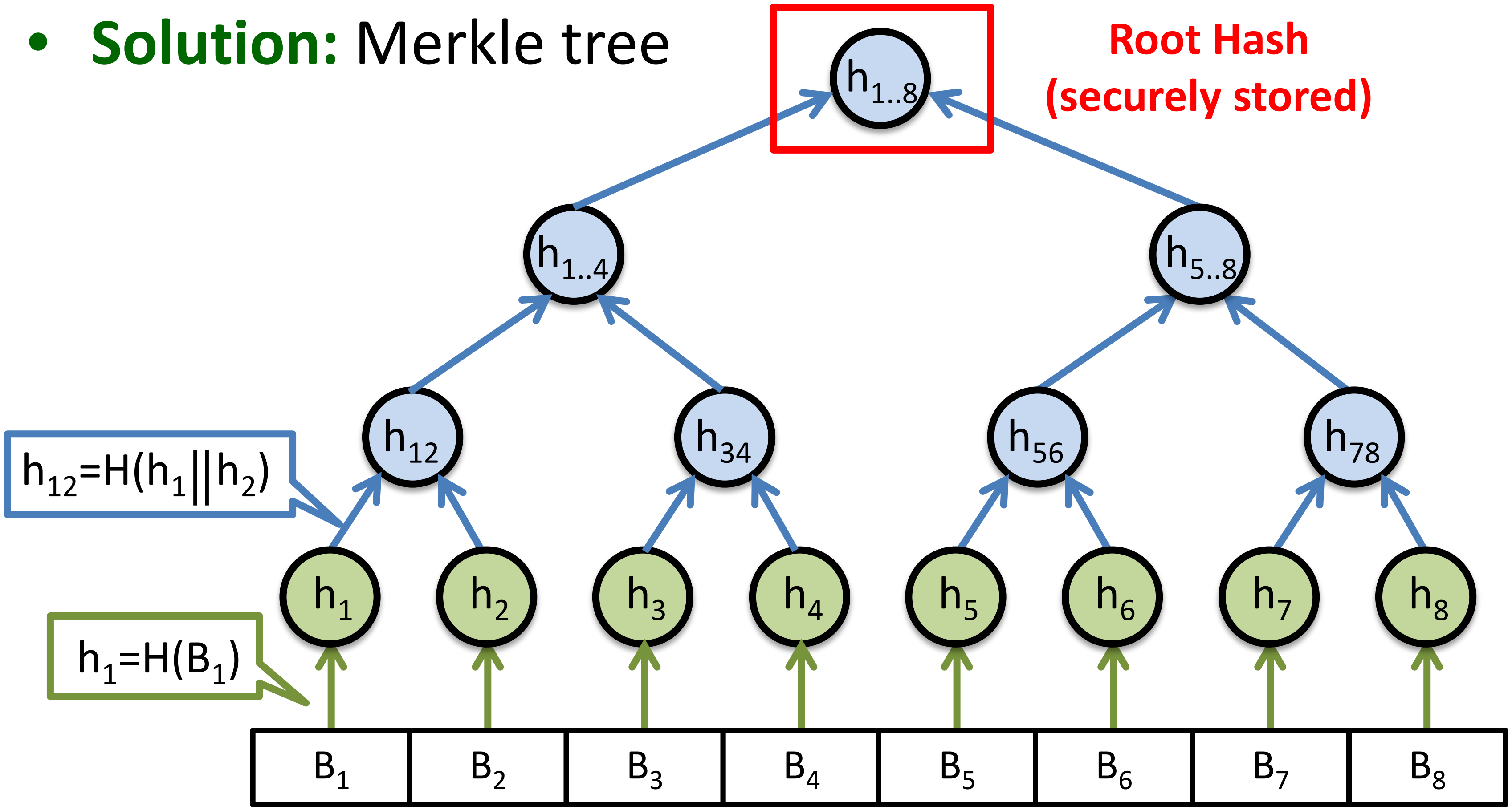
- **Solution:** Merkle tree



Disk is divided into many blocks

Design: Memory Authentication

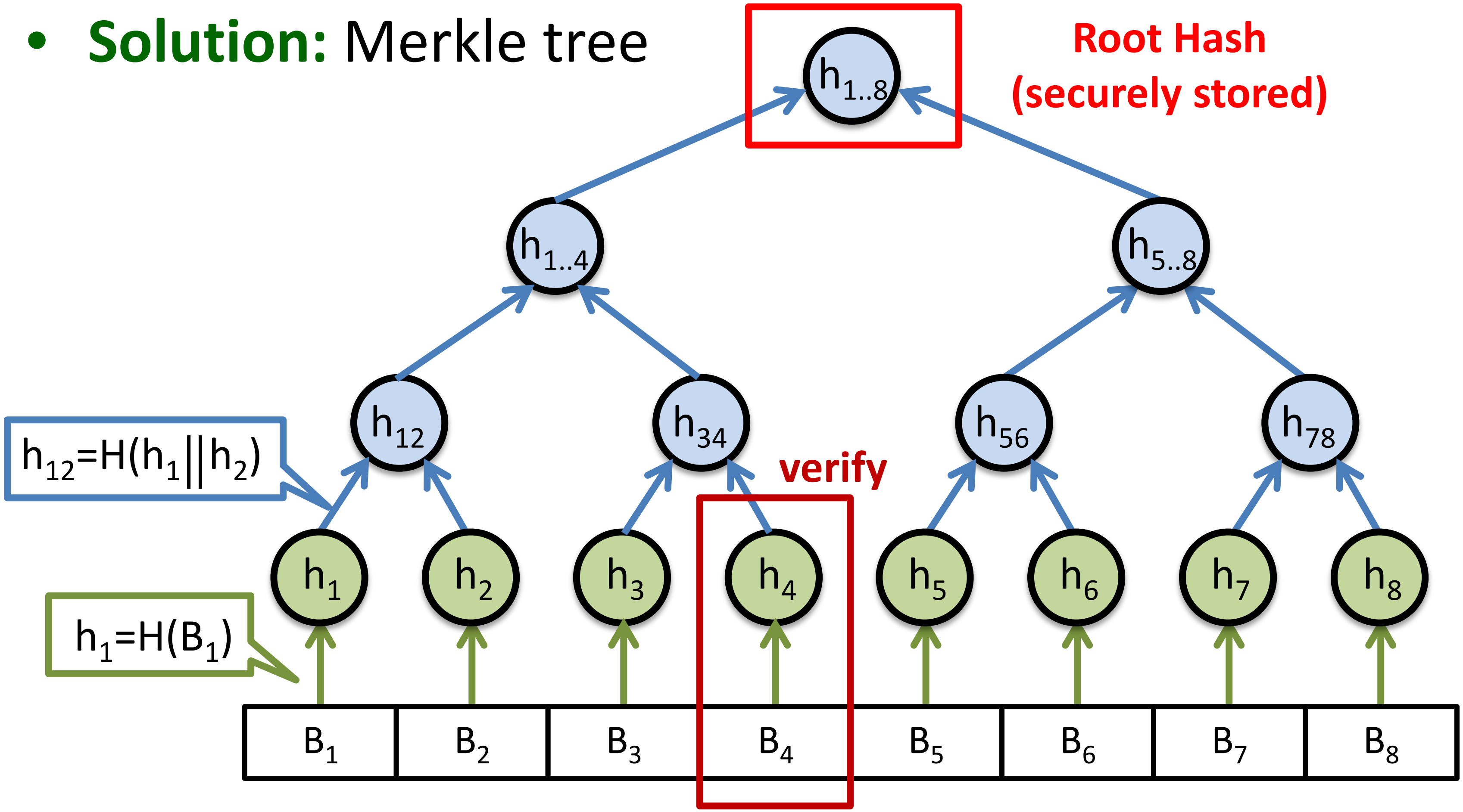
- **Solution:** Merkle tree



Disk is divided into many blocks

Design: Memory Authentication

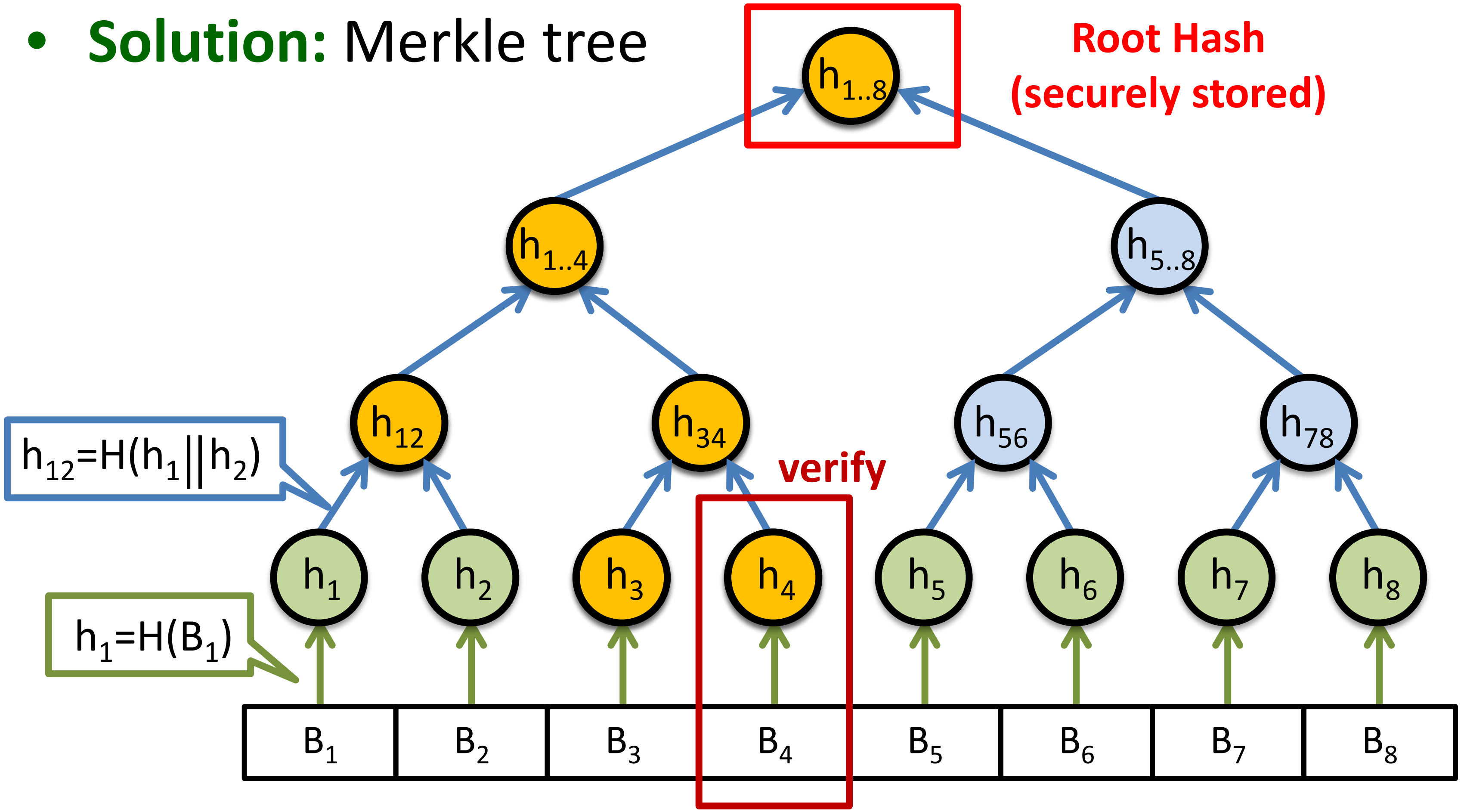
- **Solution:** Merkle tree



Disk is divided into many blocks

Design: Memory Authentication

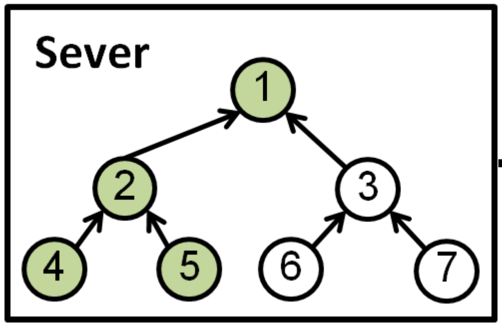
- **Solution:** Merkle tree



Disk is divided into many blocks

Merkle Tree Caching

- **Caching policy is controlled by the server**



Cache management commands:

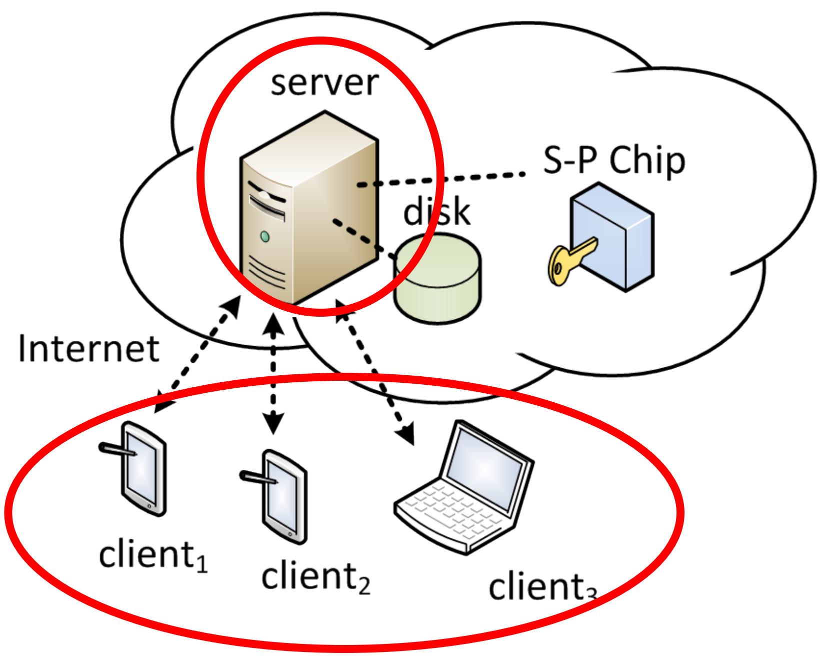
LOAD, VERIFY, UPDATE

P chip

Node #	Hash	Verified	Left child	Right child
1	fabe3c05d8ba995af93e	Y	Y	N
2	e6fc9bc13d624ace2394	Y	Y	Y
4	53a81fc2dcc53e4da819	Y	N	N
5	b2ce548dfa2f91d83ec6	Y	N	N

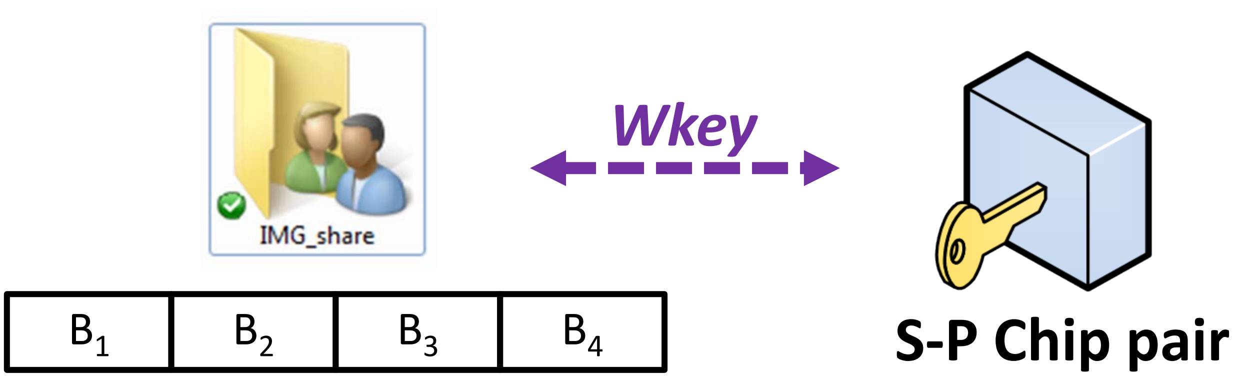
Security Protocols

- Message Authentication
- Memory Authentication
- **Write Access Control**
- System State Protection against Power Loss



Design: Write Access Control

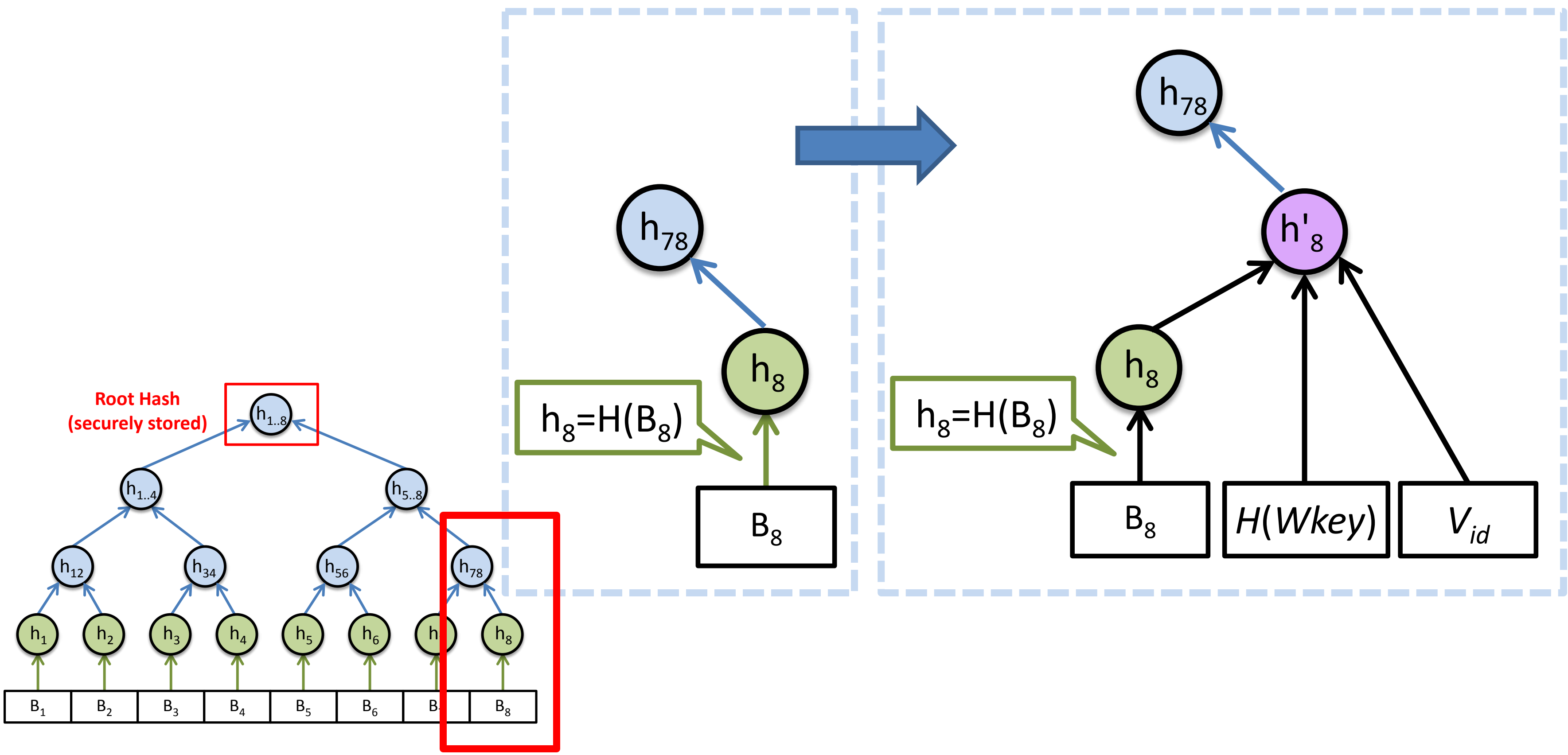
- **Goal: to ensure all writes are authorized and fresh**
- **Coherence model assumption:**
  - Clients should be aware of the latest update
- **Unique write access key (*Wkey*)**
  - Share between authorized writers and the S-P chip



- **Revision number (V_{id})**
  - Increase during each write operation

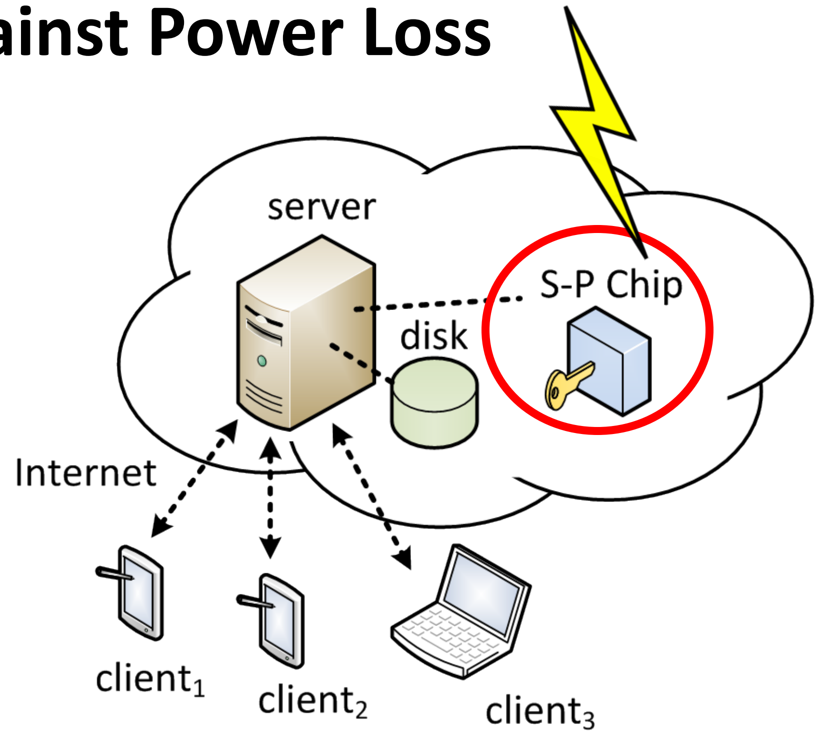
Design: Write Access Control

- **Protect *Wkey* and V_{id}**
  - Add another layer at the bottom of Merkle tree



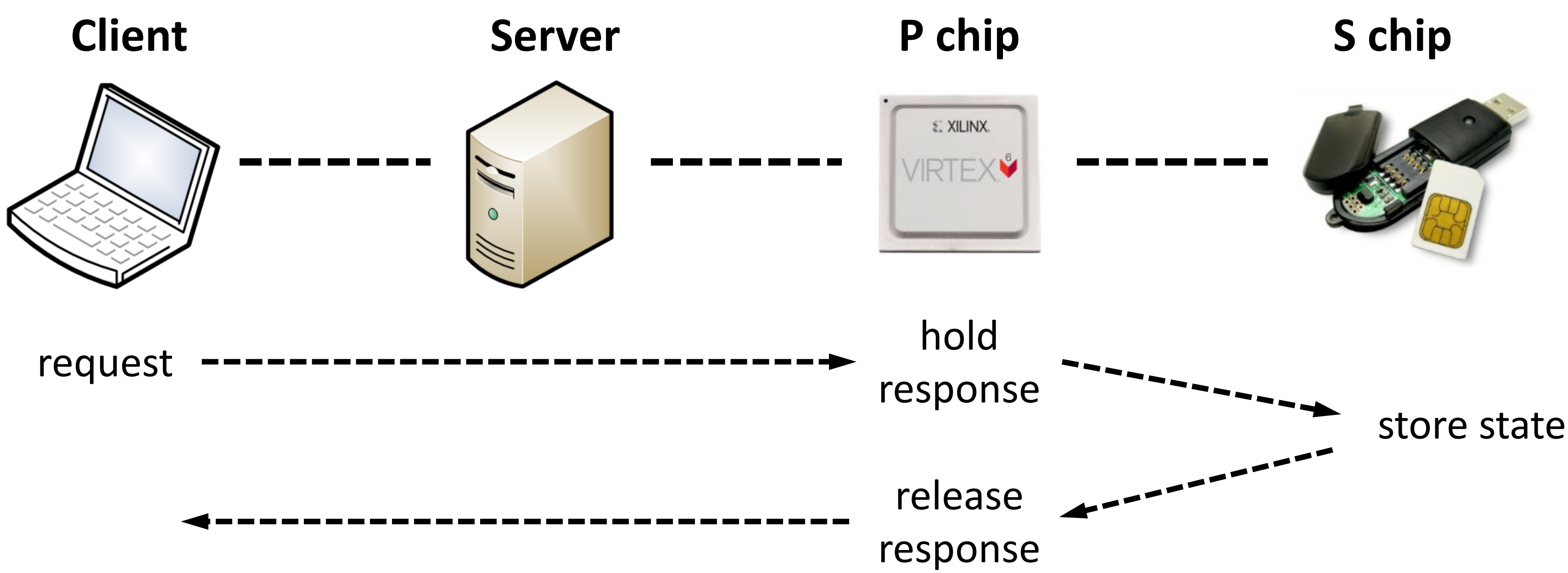
Security Protocols

- Message Authentication
- Memory Authentication
- Write Access Control
- **System State Protection against Power Loss**



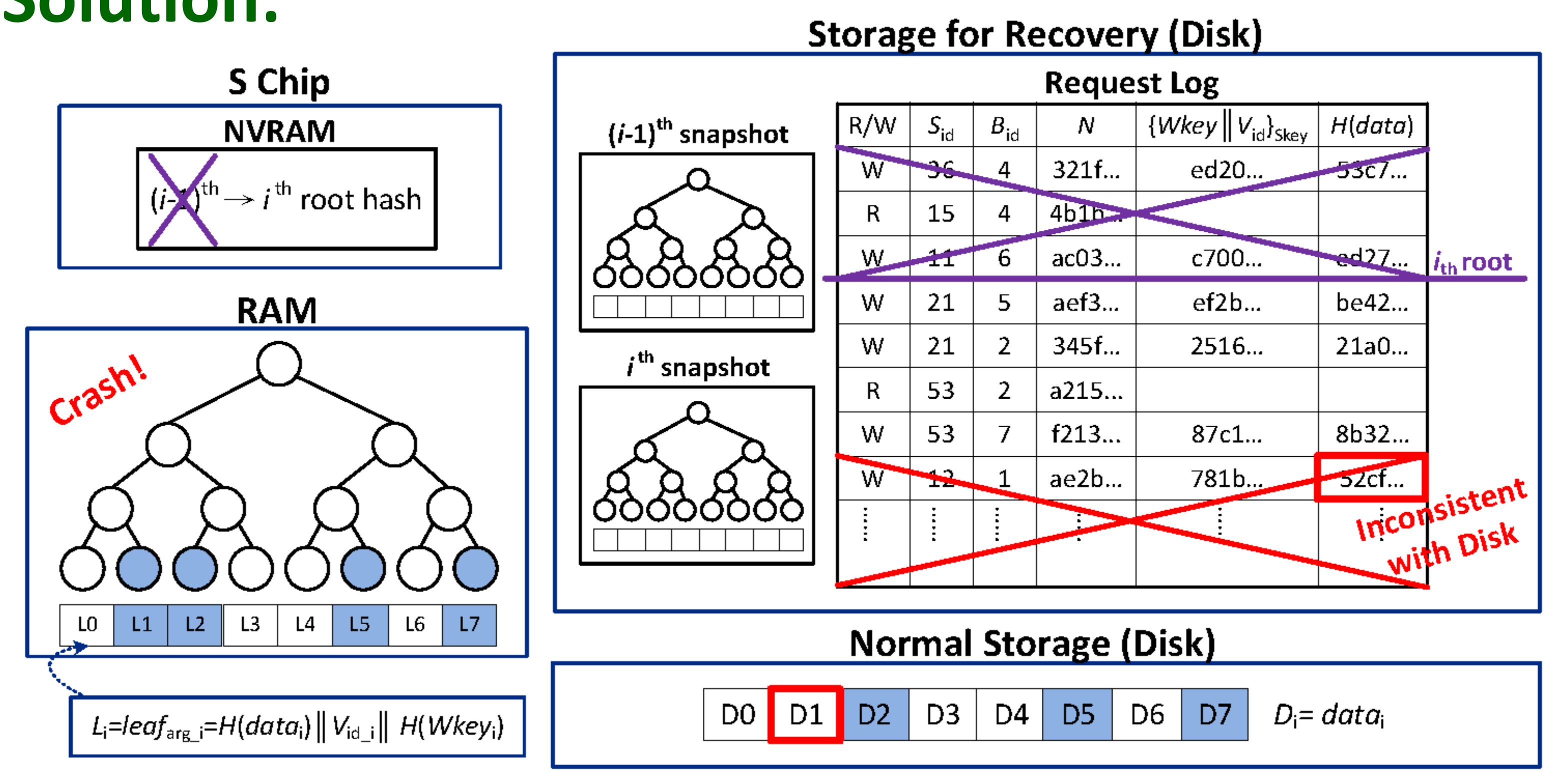
Design: System State Protection

- **Goal: to avoid losing the latest system state**
  - Server may interrupt the P chip's supply power
- **Solution: root hash storage protocol**



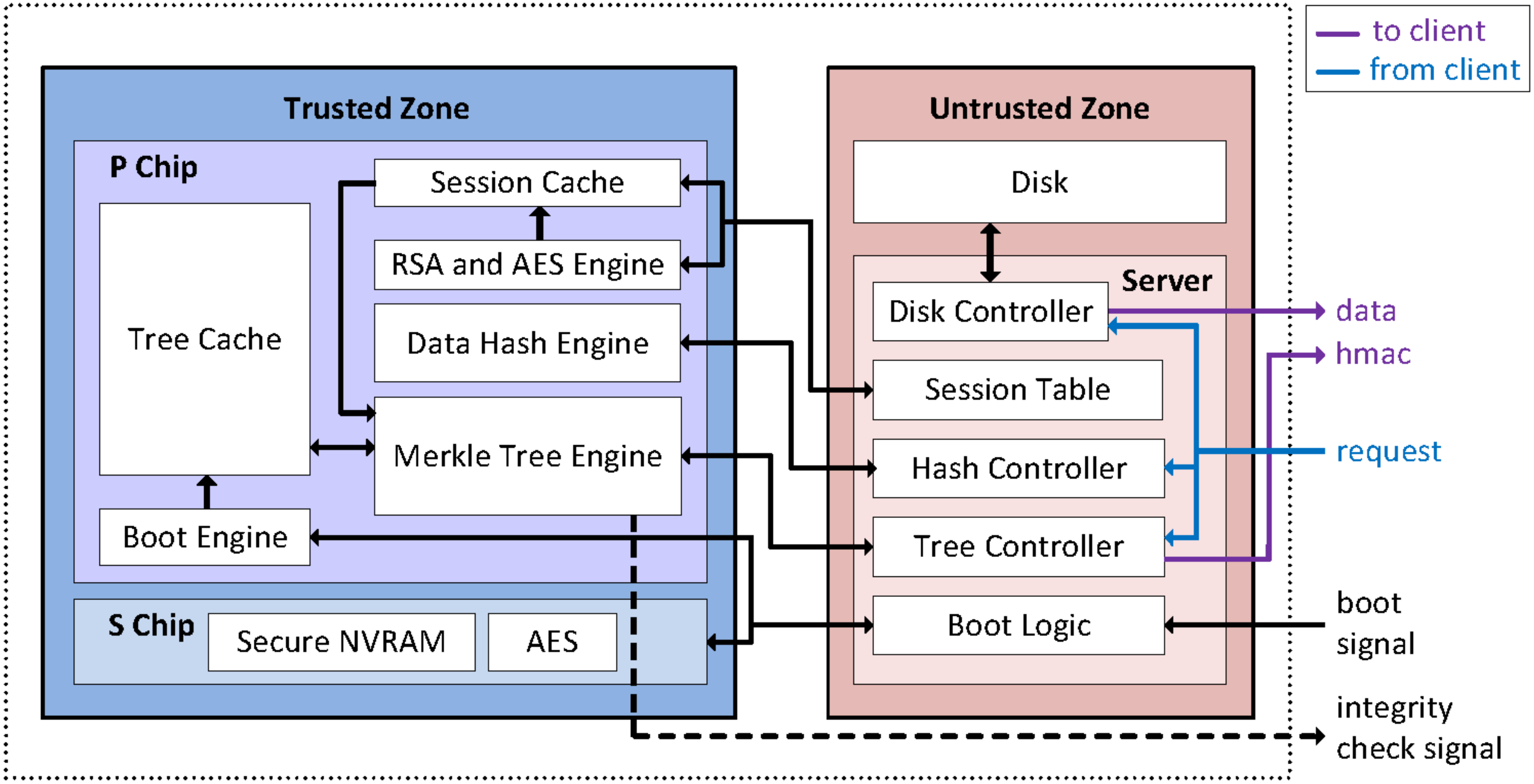
Design: Crash Recovery Mechanism

- **Goal: to recover the system from crashes**
  - Even if the server crashes, the disk can be recovered to be consistent with the root hash stored on the S chip
- **Solution:**



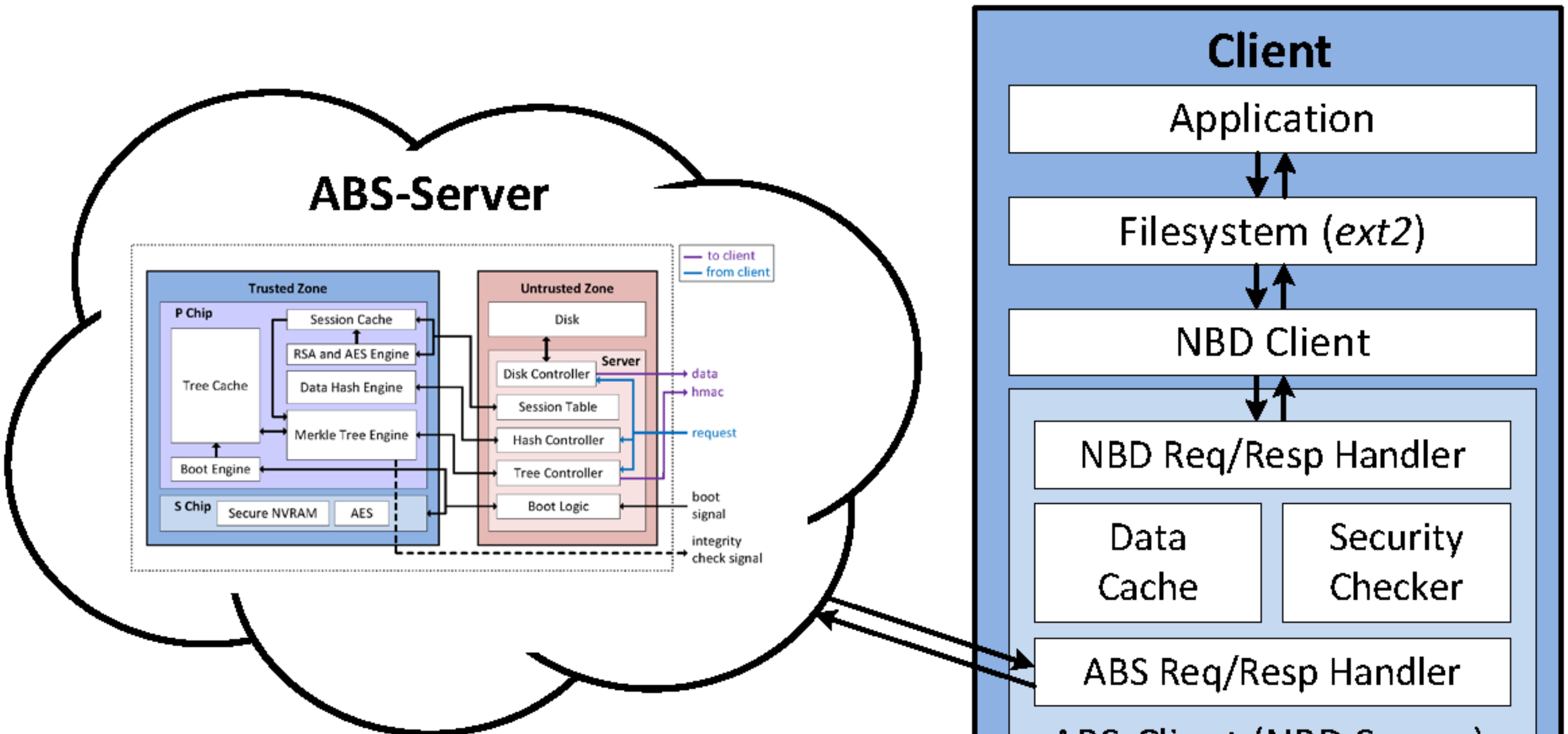
Implementation

- **ABS (authenticated block storage) server architecture**



Implementation

- **ABS client model**

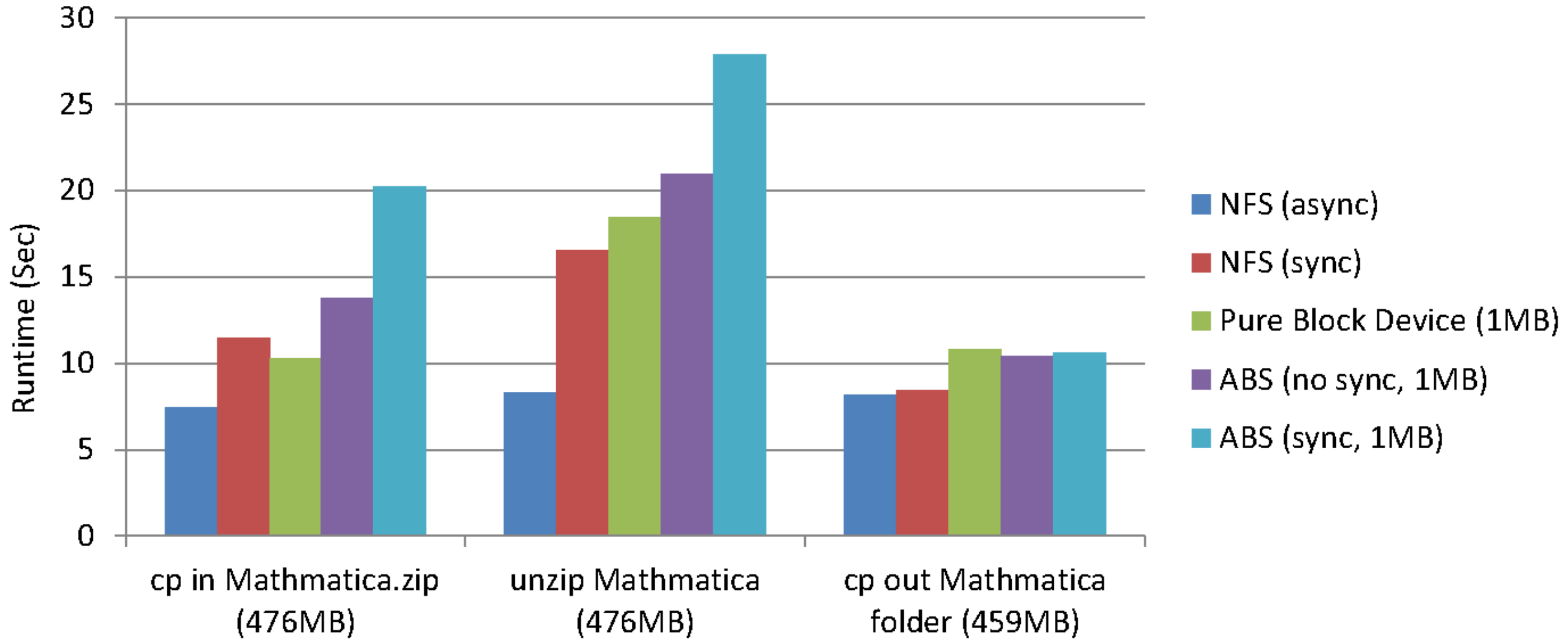


Performance Evaluation

- **Experiment configuration**
  - **Disk size:** 1TB
  - **Block size:** 1MB
  - **Server:** Intel Core i7-980X 3.33GHz 6-core processor
    with 12GB of DDR3-1333 RAM
  - **FPGA:** Xilinx Virtex-5 XC5VLX110T
  - **Client:** Intel Core i7-920X 2.67GHz 4-core processor
  - **FPGA-server connection:** Gigabit Ethernet
  - **Client-server connection:** Gigabit Ethernet

File System Benchmarks (Mathmatica)

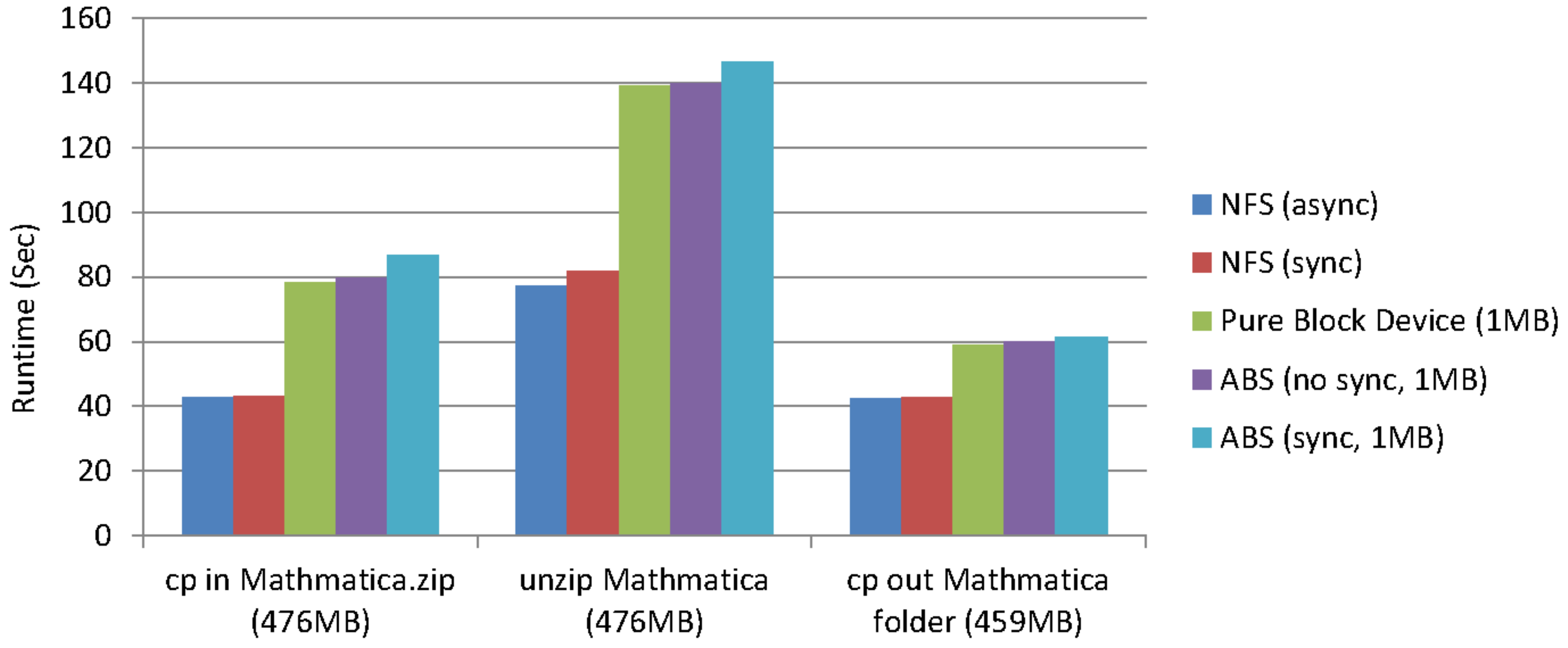
- **Fast network:**
  - Latency: 0.2ms
  - Bandwidth: 1Gbit/s



pure writes reads + writes pure reads

File System Benchmarks (Mathmatica)

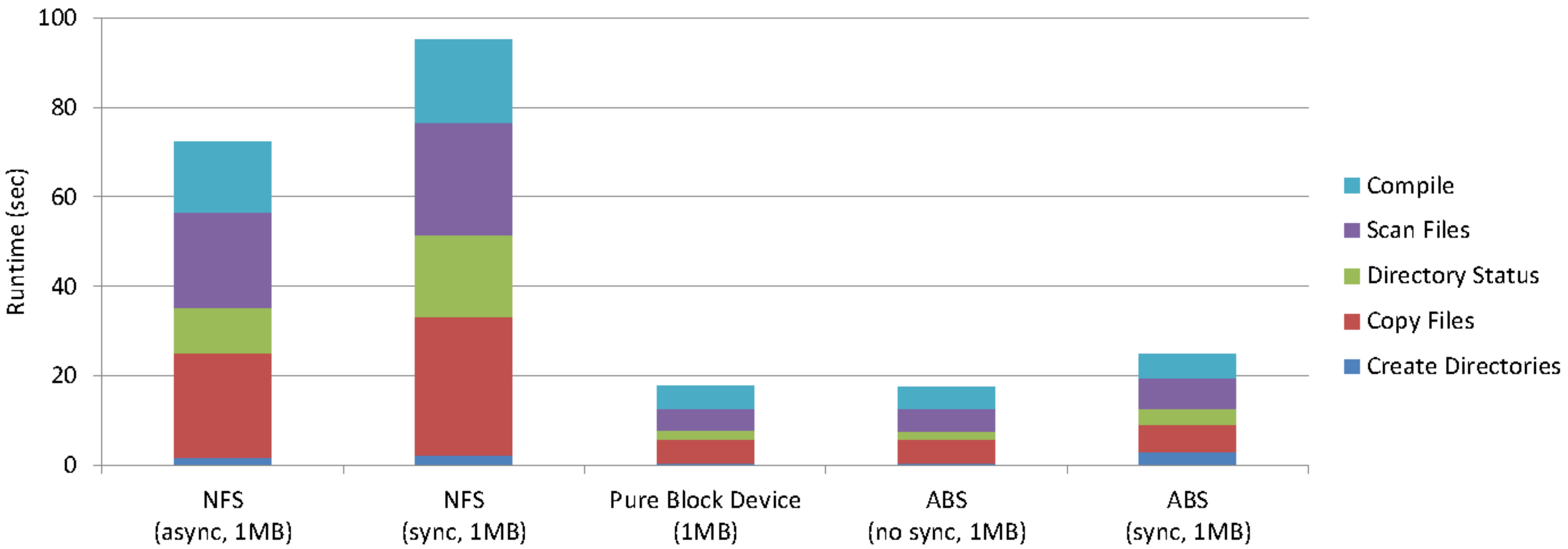
- **Slow network:**
  - Latency: 30.2ms
  - Bandwidth: 100Mbit/s



pure writes   reads + writes   pure reads

File System Benchmarks (Modified Andrew Benchmark)

- **Slow network:**
  - Latency: 30.2ms
  - Bandwidth: 100Mbit/s



Customized Solutions

- **Hardware requirements**

Demand Focused	Performance	Budget
Connection	PCIe x16 (P) / USB (S)	USB
Hash Engine	8 + 1 (Merkle)	0 + 1 (Merkle)
Tree Cache	large	none
Response Buffer	2 KB	300 B

- **Estimated performance**

Demand Focused		Performance	Budget
Randomly Write	Throughput	2.4 GB/s	377 MB/s
	Latency	12.3 ms + 32 ms	2.7 ms + 32 ms
Randomly Read	Throughput	2.4 GB/s	
	Latency	0.4 ms	
# HDDs supported		24	4

Customized Solutions

- **Hardware requirements**

Single chip!

Demand Focused	Performance	Budget
Connection	PCIe x16 (P) / USB (S)	USB
Hash Engine	8 + 1 (Merkle)	0 + 1 (Merkle)
Tree Cache	large	none
Response Buffer	2 KB	300 B

- **Estimated performance**

Demand Focused		Performance	Budget
Randomly Write	Throughput	2.4 GB/s	377 MB/s
	Latency	12.3 ms + 32 ms	2.7 ms + 32 ms
Randomly Read	Throughput	2.4 GB/s	
	Latency	0.4 ms	
# HDDs supported		24	4

Conclusion

- We build an authenticated storage system
  - Efficiently ensure data integrity and freshness
  - Prevent unauthorized/replayed writes
  - Can be recovered from accidentally/malicious crashes
- The system has 10% performance overhead on the network with 30 ms latency and 100 Mbit/s bandwidth
- We provide customized solutions
  - With limited resources: single-chip solution
  - With more hardware resources: two-chip solution

Thank You!