

TrInc :Small Trusted Hardware for Large Distributed Systems NSDI'09



DAVE LEVIN*
JACOB R. LORCH**

JOHN R. DOUCEUR**
THOMAS MOSCIBRODA**

PRESENTED BY:
GAURAV KUMAR MEHTA

* University of Maryland

**Microsoft Research

Faults in Distributed Systems



- Can generally be classified into *stopping faults* and *Byzantine faults*.
- *Stopping faults* cover cases of processors crashing or leaving.
- *Byzantine faults*: faults that result in the generation of incorrect result packets such as malicious use.

Byzantine Faults



- Tolerating f Byzantine faults :
 - Requires at least $3f+1$ participants as opposed to $2f+1$ required to tolerate stopping fault
- Property responsible for the above difference:
Equivocation
 - The ability to make conflicting statements to different participants.

Prior solutions to Equivocation



- Having an accountability layer
 - With Witnesses in PeerReview system
 - ✦ Eventual Fault detection & Additional communication overhead per message .
 - With Guards in Nysiad system
 - ✦ Additional communication overhead per message.

Trusted hardware Component



- Attested Append-only Memory(A2M) System by Chun et al.
- Proposed an abstraction of trusted log in an untrusted machine.
- Benefits:
 - Communication overhead is merely a constant factor
- Concerns with that:
 - A large trusted storage space and complexity
 - Applicability in different distributed protocols

TrInc



- Trusted Incrementer : Essentially a non-decreasing counter and a key
- Smaller in size and easier to deploy
- Applicable in Client Server architectures /peer-2-peer and a wide range of protocols.
- Other benefits:
 - Reduces communication overhead in addition to improving fault tolerance.
 - API with which it is easy to build distributed systems.

Overview of TrInc



- All participants have a trusted hardware, trinket
- States:
 - Global State
 - ✦ I : ID of the trinket
 - ✦ M: Meta Counter, keeping track of the number of counters created
 - Per Counter State
 - ✦ i: Identity of this counter , value of M when it was generated
 - ✦ c: current value of counter
 - ✦ K: key for attestations

Mallory sends a message m to Alice



Mallory (Message m)

Alice

$\text{Attest}(i, c', m)$



$\langle I, i, c, c', m \rangle_K$

Two types of attestations:

Advance attestation: Increases the counter value

Status attestation: Gives the current counter value

Each message is bound to a certain value of the counter and no other message will use the same value of the counter

Trinket



- Other API calls of trinket
 - CheckAttestation(a,i): to validate the attestation
 - CreateCounter(): to create a new counter and increment M
 - FreeCounter(i): to delete the counter i
 - ImportSymmetricKey(S,i): to add counter i to the current session with encrypted symmetric key S(can be decrypted only by trinket)
 - GetRecentAttestations(): Return Q (a limited size FIFO containing recent attestations to deal with power failures)

Designing systems with TrInc



- **A2M(append,advance,truncate,lookup):**
 - Basis of A2M resilience to equivocation is *append*, which binds a sequence number to the message.
 - For each log q , L_q and H_q
 - $s > H_q$ or $s < L_q$, implies failed lookups
- **TrInc-A2M**
 - Two counters per log for L_q and H_q
 - append- \rightarrow Advance attestation for H_q
 - Truncate- \rightarrow Advance attestation for L_q
 - Failed lookups- \rightarrow Status attestation

Properties of a TrInc-based A2M



- Can be applied to systems where A2M works.
- Stores logs in untrusted storage, thereby reducing the trusted storage demand.
- No need to truncate frequently to keep logs small.

BitTorrent and Piece Under-Reporting



- Bitfields represent which pieces of a file a peer has.
- A peer is interested in those peers who have pieces that it does not.
- Prolonged interest from others lead to lesser download times
- i receives q from peer j and now has $\{p,q\}$
- k and l want $\{p,q\}$, but i equivocates by saying it has only p , to keep them interested and hence increase his download speed

Applying TrInc



- Counter matches the bitfield size
- Peers attest their bitfield and the most recent piece received.
- No further pieces on equivocating or remaining silent.

Other applications



- PeerReview
- Elections
- Version Control systems
- DHT's
- And others where equivocation is a possible threat

Implementation



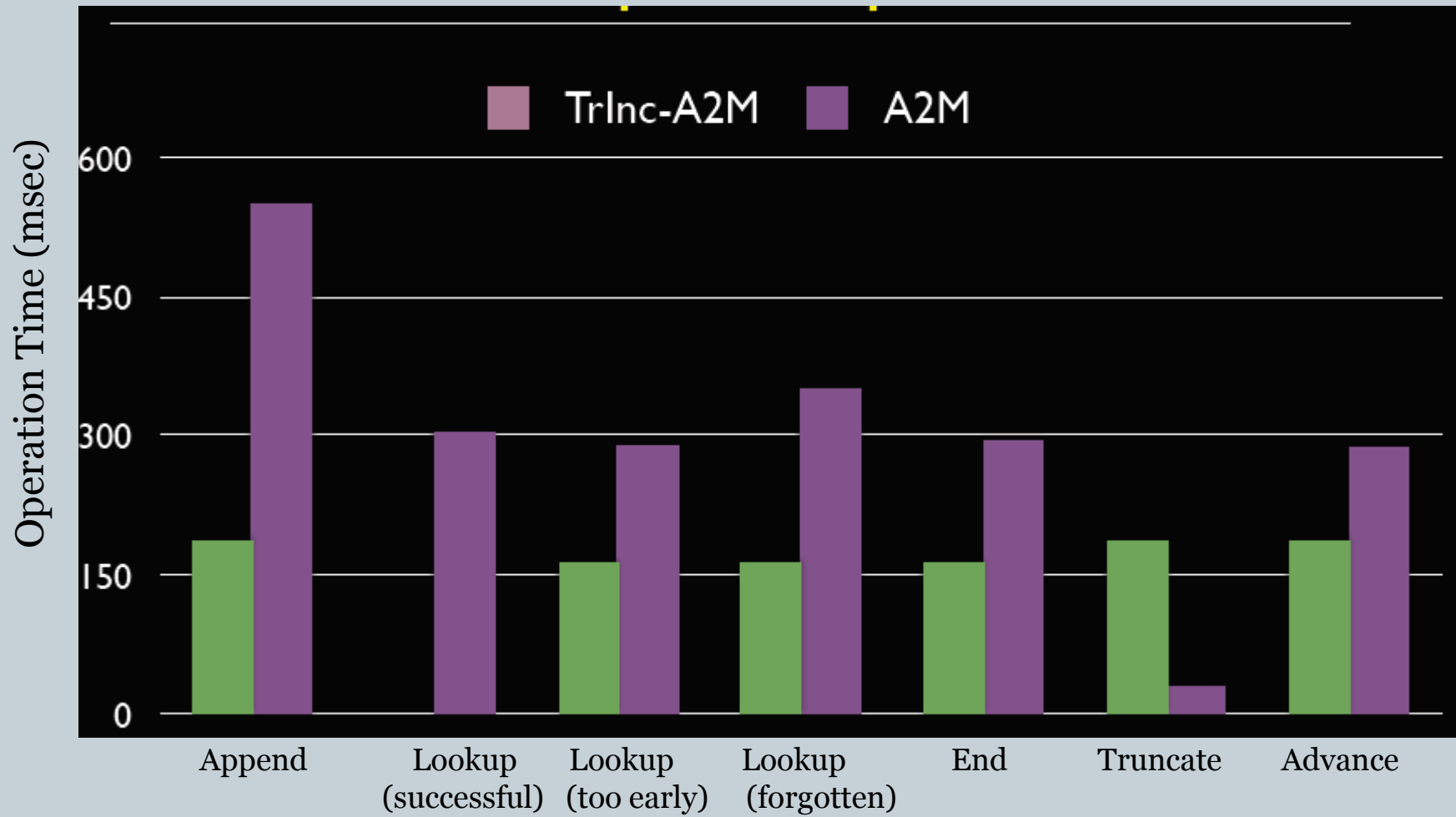
- Gemalto .NET Smartcard
 - Crypto unit (RSA & 3-DES)
 - 32-bit microcontroller
 - 80 KB persistent memory
- Case Studies:
 - TrInc-A2M
 - TrInc-PeerReview
 - TrInc-BitTorrent

Slow TrInc

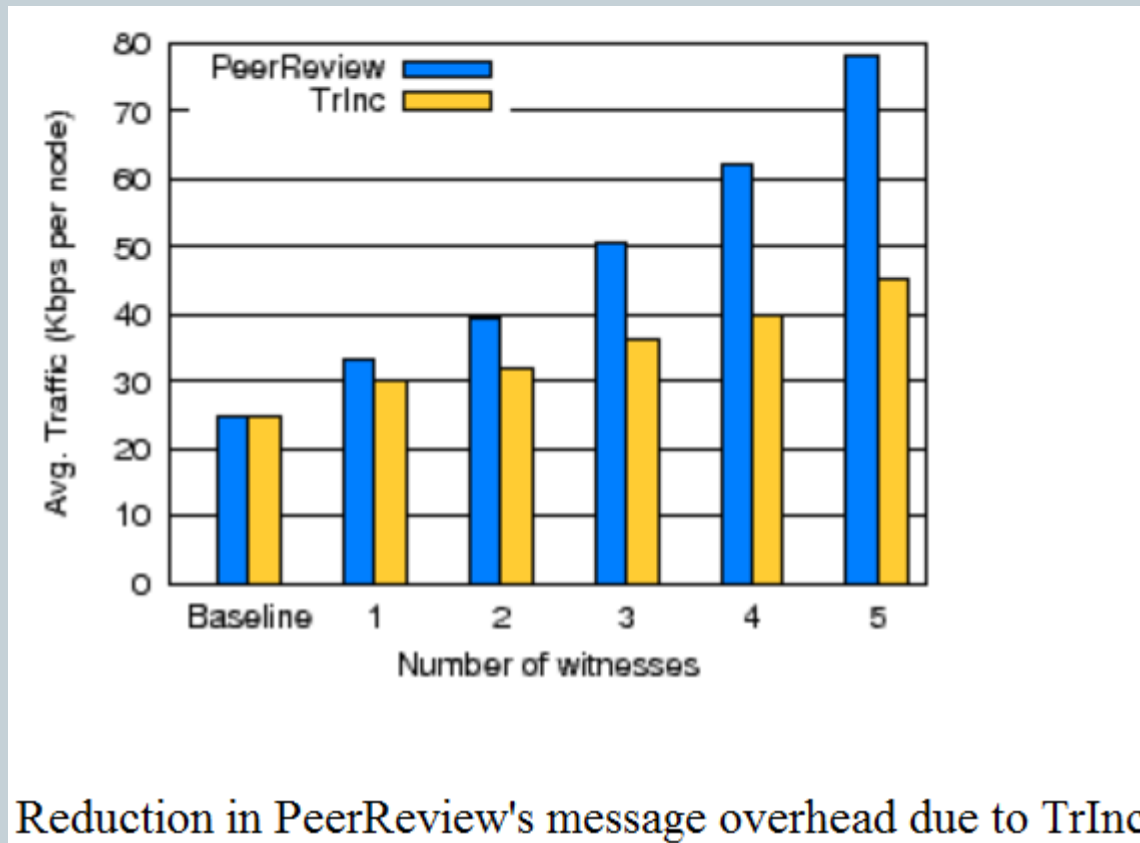


- On an untrusted PC, 3-DES took 0.017 ± 0.008 msec and RSA took 8.6 ± 0.67 msec as opposed to 100-200 msec on trusted hardware
- Difference between symmetric and asymmetric encryption was striking with symmetric being 2X times faster (500 times on PC).
- TrInc: Fundamentally new application for trusted hardware.

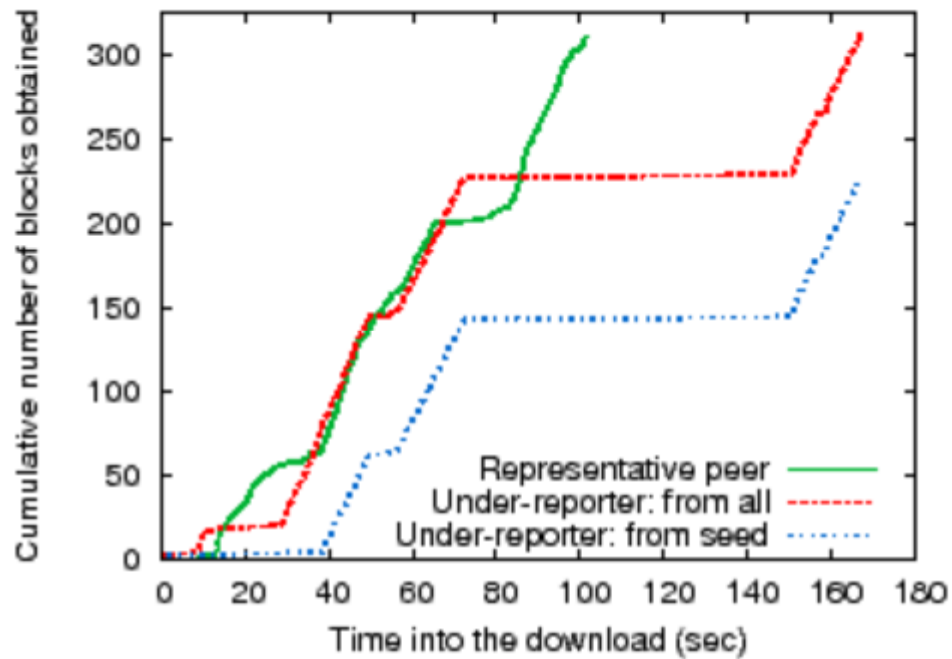
TrInc-A2M



TrInc-PeerReview



TrInc-BitTorrent



Rate of progress for various BitTorrent clients when TrInc is used.

Conclusion



- Equivocation can be avoided by using a trusted hardware component
- TrInc, a simple yet powerful abstraction for improving security in distributed systems.
- Implemented on real, currently available trusted hardware.
- Scope of improvement in the trusted hardware as TrInc is slow!



Thanks