Reflections on Trusting Distributed Trust

Emma Dauterman     Vivian Fang     Natacha Crooks     Raluca Ada Popa

UC Berkeley

HotNets 2022
Bob is using end-to-end encrypted messaging.

Bob is pleased!
Bob broke his phone!

His secret key is gone.

Bob is displeased.
How to back up secret keys?

Problem: Signal has Bob’s secret key and can decrypt Bob’s messages.
How to back up secret keys?

Problem: Signal has Bob’s secret key and can decrypt Bob’s messages

No single trust domain can compromise Bob’s secret key
Design pattern: distributing trust for security

Distributed-trust applications:

- General secure multi-party computation
  [BGW88, GMW87, Yao82]
- Private queries
  [Popcorn, Checklist, Senate, DORY, Waldo]
- Private analytics
  [Prio, Poplar]
- Anonymous messaging
  [Tor, Riposte, Blinder, Dissent, Express]
- Cryptocurrency custody
  [Fireblocks, Curv, Unbound, Knox]
- Byzantine fault-tolerant consensus
  [Diem, HyperLedger, HotStuff]

… and many, many more!

Security when no more than $t$ trust domains are compromised at once.
Design pattern: distributing trust for security

How do we set up distributed trust?

- Cryptocurrency custody
  [Fireblocks, Curv, Unbound, Knox]
- Byzantine fault-tolerant consensus
  [Diem, HyperLedger, HotStuff]

... and many, many more!

Security when no more than $t$ trust domains are compromised at once.
Attempt: Developer deploys trust domains in different clouds

1. Developer sets up application
2. User interacts with application
Attempt: Developer deploys trust domains in different clouds

Problem: Developer becomes central point of attack!
Developer can:
- Inspect memory
- Push malicious update

1. Developer sets up application
2. User interacts with application
3. Developer is compromised
Solution today: cross-organization coordination

Problem: Human coordination across organizations takes time and money!

1. Different organizations agree to manage servers for application
2. User interacts with application
How can a developer set up a distributed-trust application on her own without becoming a central point of attack?

**Takeaway #1**: Setting up distributed-trust systems is a hard problem that needs further study.
How can a developer set up a distributed-trust application on her own without becoming a central point of attack?

How can we audit a distributed-trust deployment?

Inspiration: Certificate transparency
Our proposal: Audit distributed-trust deployment

Challenge: How does auditor know that the deployed code matches the published code?
Our proposal: Audit distributed-trust deployment

Challenge: How does auditor know that the deployed code matches the published code?

Solution: Secure hardware attests to deployed code.

Developer deploys and publishes code

Auditor inspects code
Our proposal: Audit distributed-trust deployment

**Challenge:** How to prevent a compromised developer from learning Bob’s secret?

- User splits secret across trust domains
- Developer is compromised
Our proposal: Audit distributed-trust deployment

Challenge: How to prevent a compromised developer from learning Bob’s secret?

Solution: Secure hardware locks developer out of application memory.

User splits secret across trust domains

Developer is compromised
Our proposal: Audit distributed-trust deployment

User splits secret across trust domains

Developer is compromised

Solution: Heterogeneous secure hardware.

Doesn’t secure hardware become a central point of attack?
Our proposal: Audit distributed-trust deployment

Takeaway #2: Developers can build an auditable distributed-trust deployment using secure hardware.

User splits secret across trust domains

Developer is compromised

See paper: Auditing updates
Looking forward: Cloud services for distributed trust

- Developer submits code to cloud service
- Cloud service attests to code that is running
- Developer is locked out of application memory

**Takeaway #3**: We need cloud services that help developers set up distributed-trust systems.
Conclusion

Takeaway #1: Setting up distributed-trust systems is a hard problem that needs further study.

Takeaway #2: Developers can build an auditable distributed-trust deployment using secure hardware.

Takeaway #3: We need cloud services that help developers set up distributed-trust systems.

Thanks!

Emma Dauterman
edauterman@berkeley.edu

Vivian Fang
v.fang@berkeley.edu