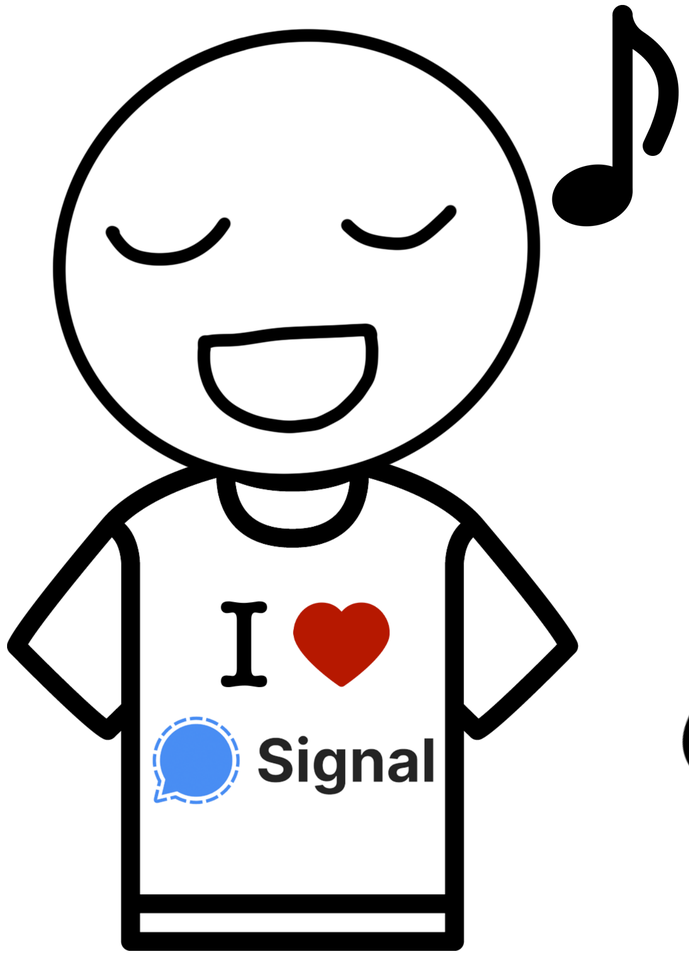
Reflections on Trusting Distributed Trust

Emma Dauterman **Vivian Fang** Natacha Crooks Raluca Ada Popa

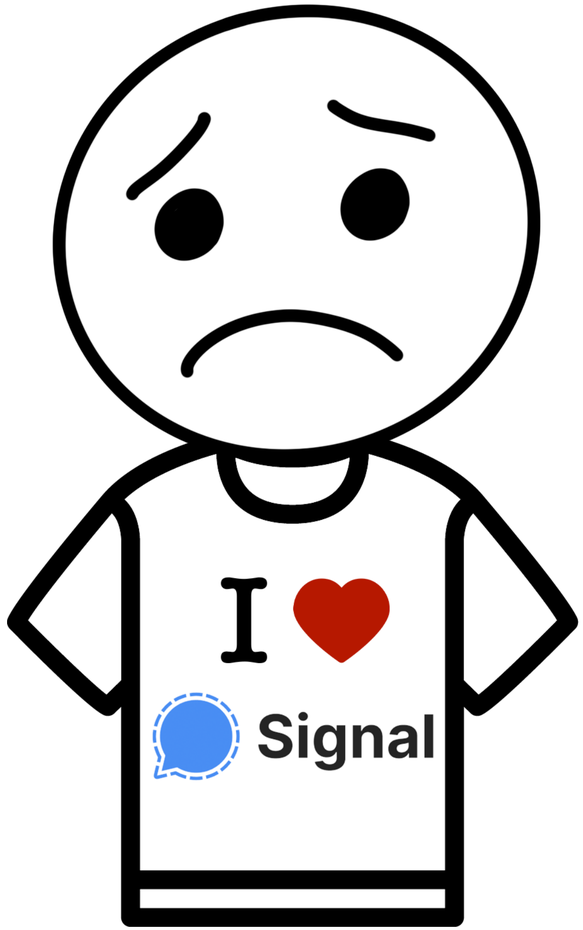
UC Berkeley

HotNets 2022



Bob is using end-to-end encrypted messaging.

Bob is pleased!

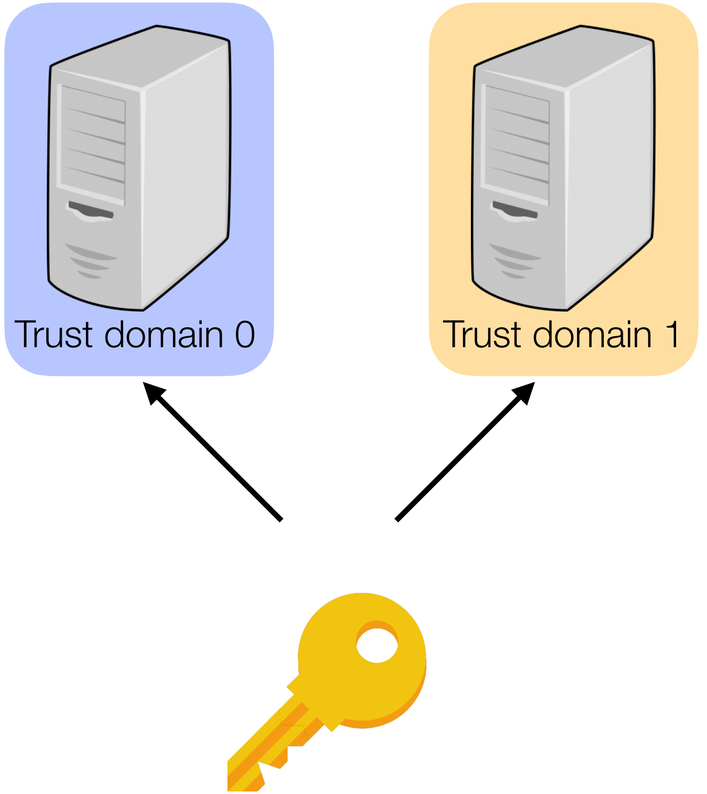


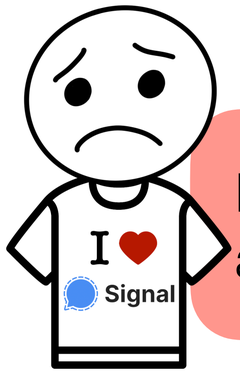
Bob broke his phone!

His secret key is gone.

Bob is displeased.

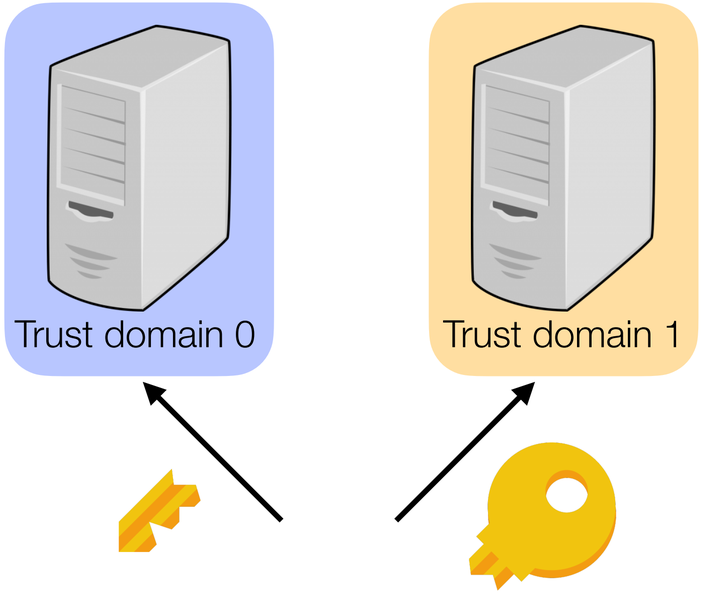
How to back up secret keys?

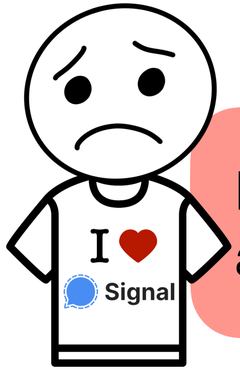




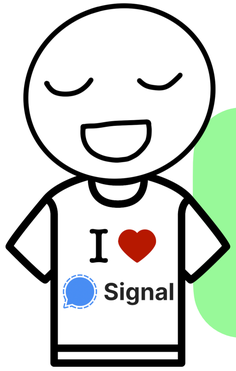
Problem: Signal has Bob's secret key and can decrypt Bob's messages

How to back up secret keys?



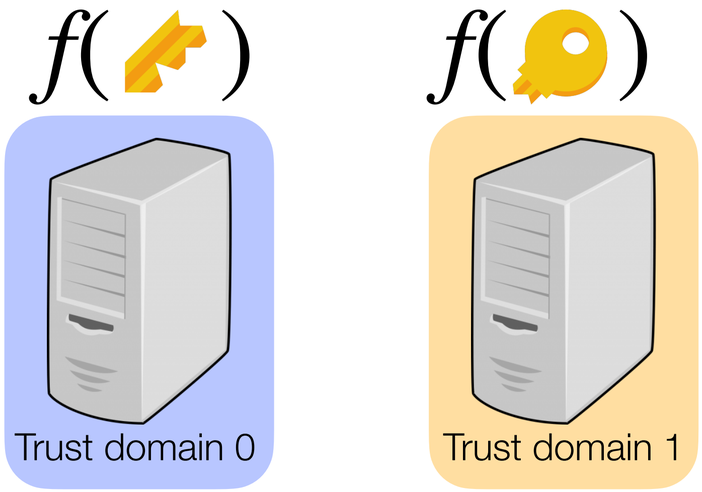


Problem: Signal has Bob's secret key and can decrypt Bob's messages



No single trust domain can compromise Bob's secret key

Design pattern: distributing trust for security



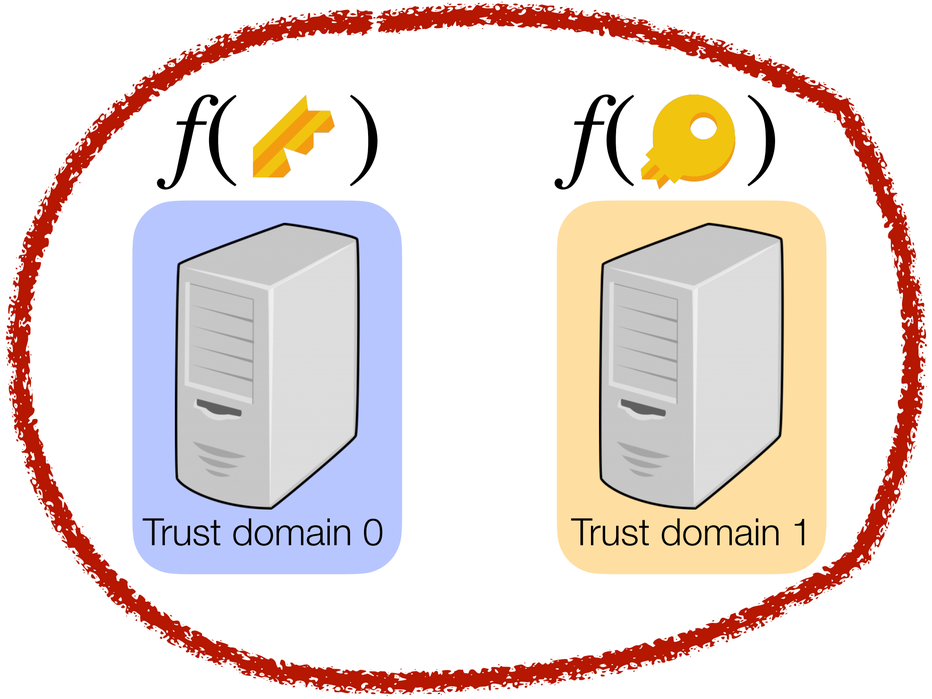
Security when no more than *t* trust domains are compromised at once.

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!

Design pattern: distributing trust for security



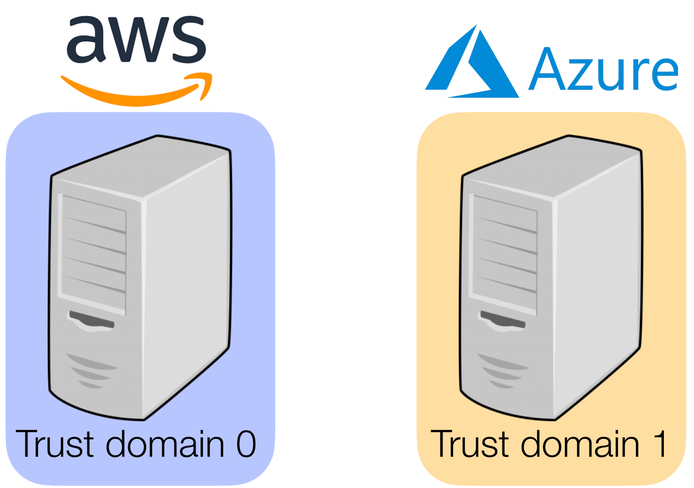
Security when no more than *t* trust domains are compromised at once.

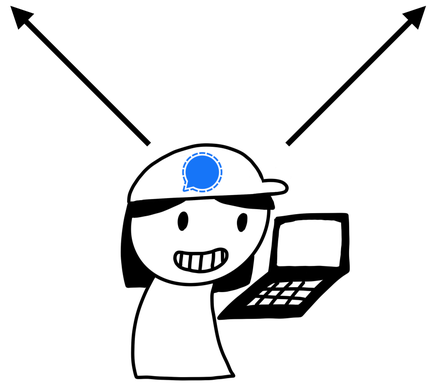
How do we set up distributed trust?

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

… and many, many more!

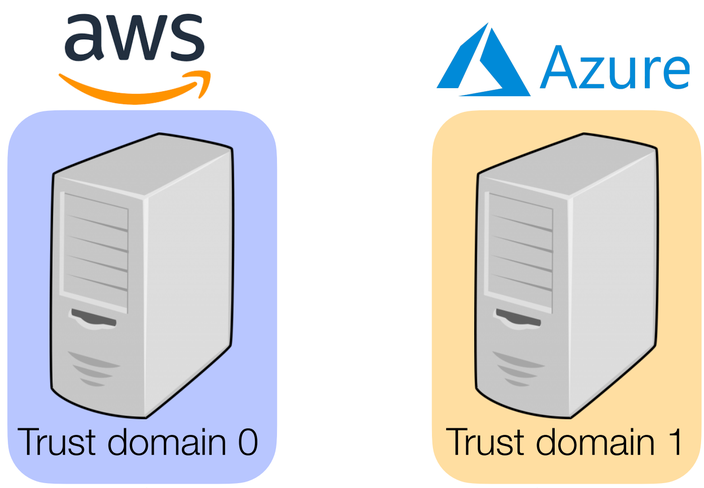
Attempt: Developer deploys trust domains in different clouds

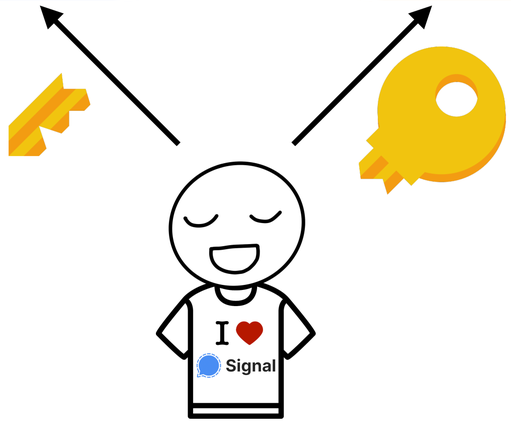




Developer

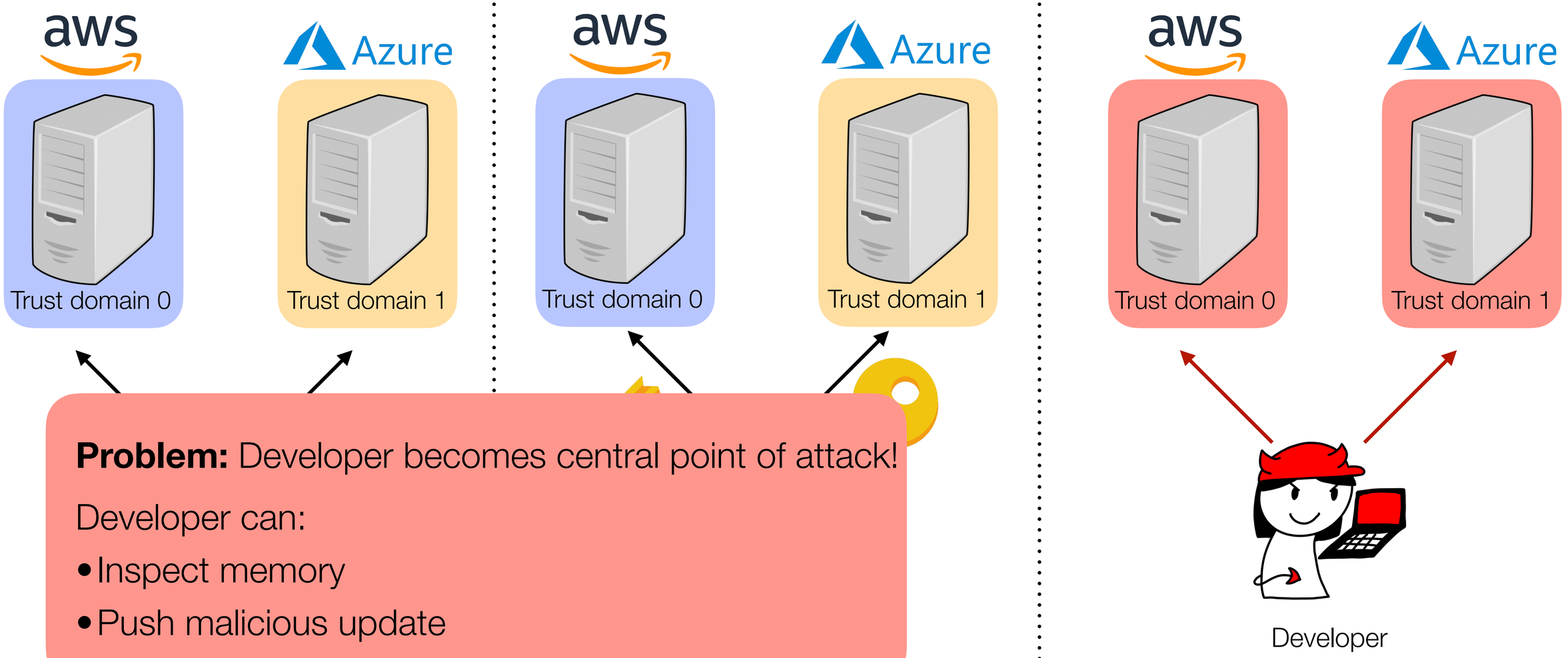
1 Developer sets up application





2 User interacts with application

Attempt: Developer deploys trust domains in different clouds



Problem: Developer becomes central point of attack!

Developer can:

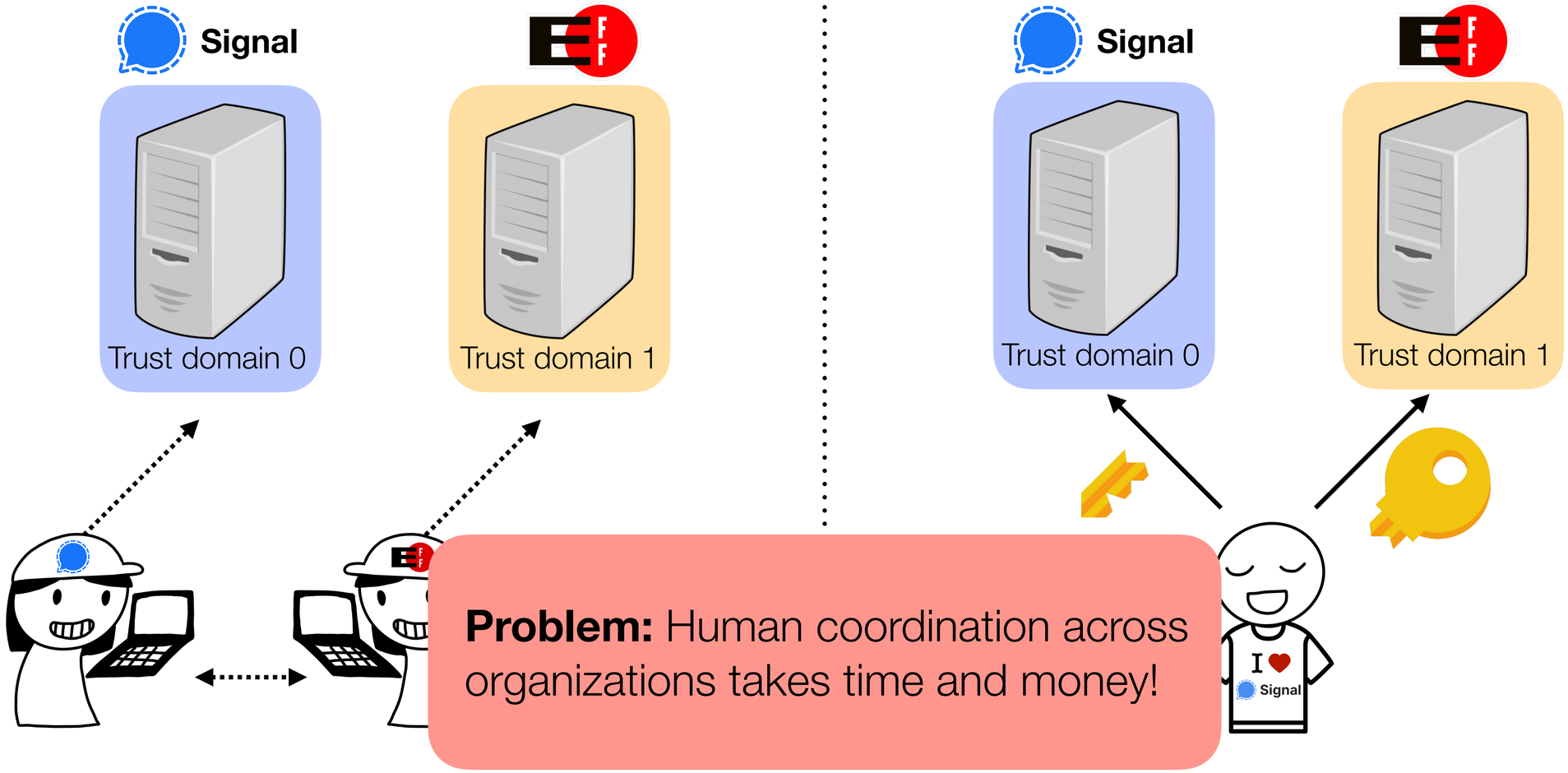
- Inspect memory
- Push malicious update

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!

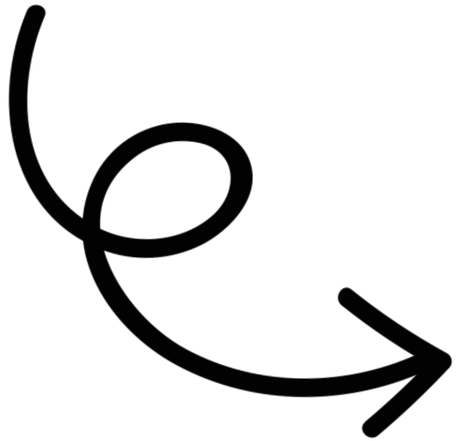
1 Different organizations agree to manage servers for application

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.

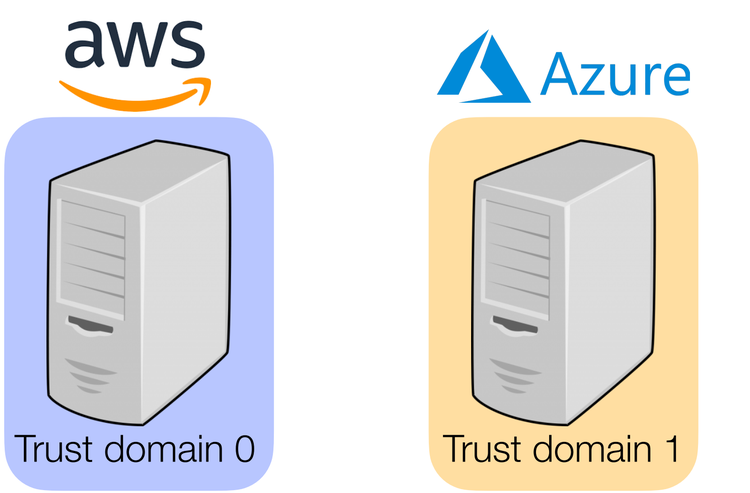
How can a developer set up a distributed-trust application on her own without becoming a central point of attack?

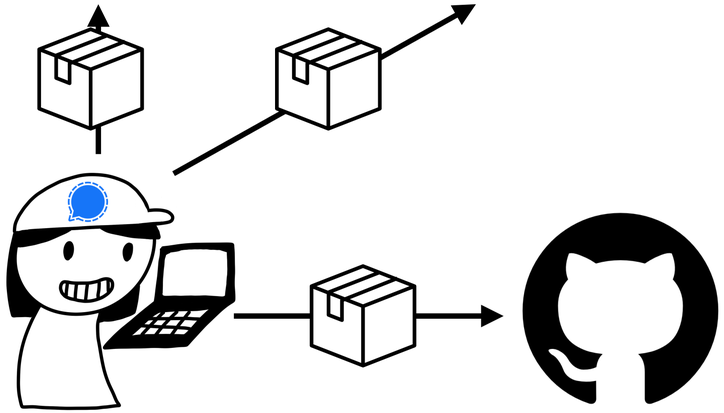


*How can we **audit** a distributed-trust deployment?*

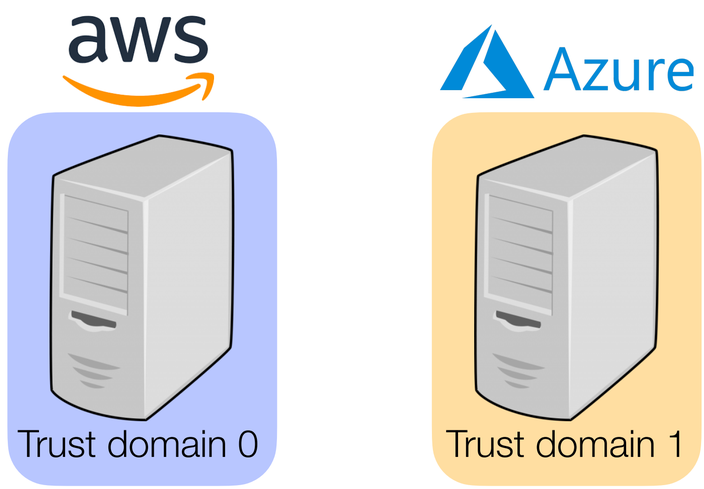
Inspiration: Certificate transparency

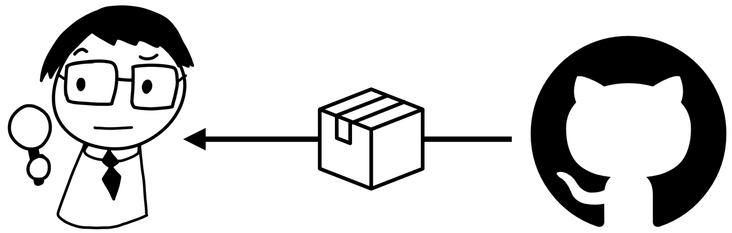
Our proposal: Audit distributed-trust deployment





Developer deploys and publishes code

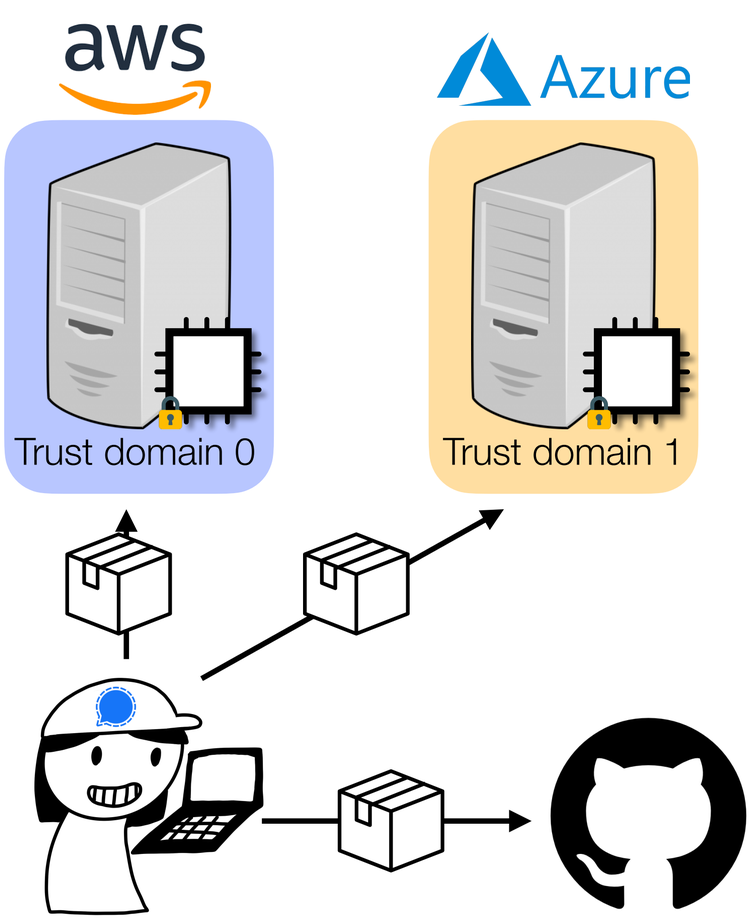




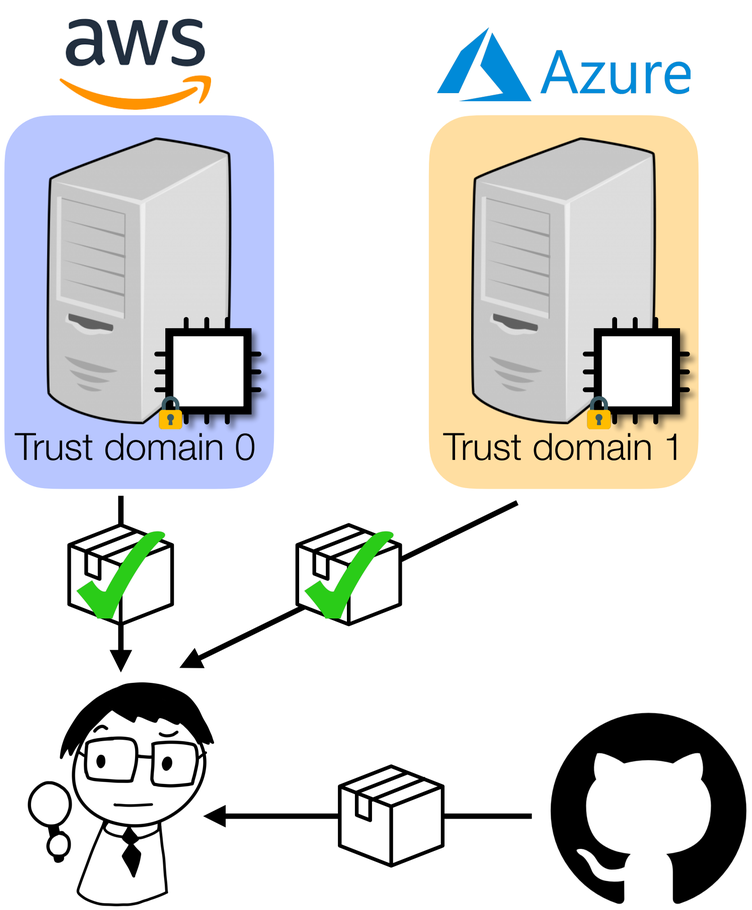
Auditor inspects code

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

Our proposal: Audit distributed-trust deployment



Developer deploys and publishes code

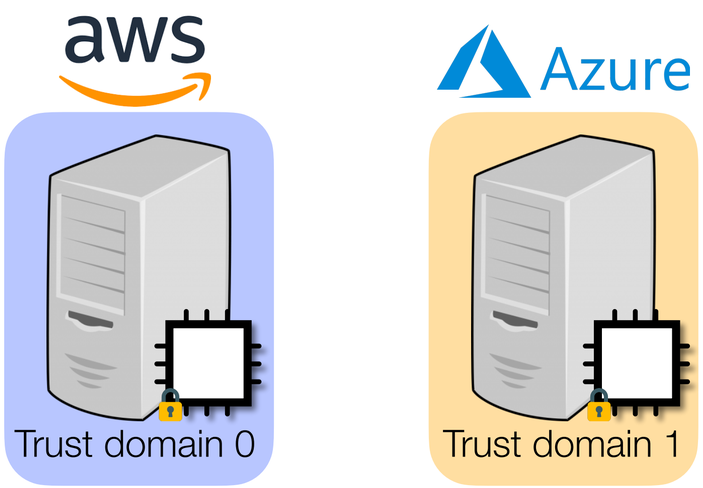


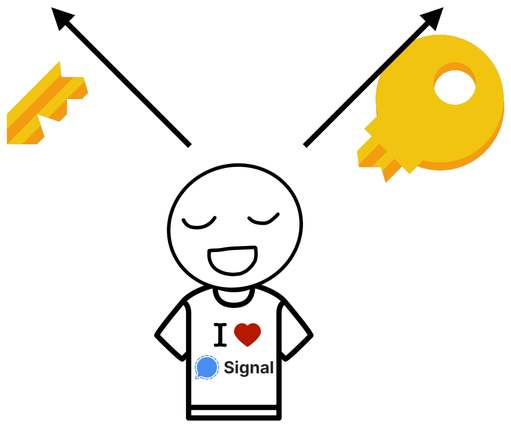
Auditor inspects code

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

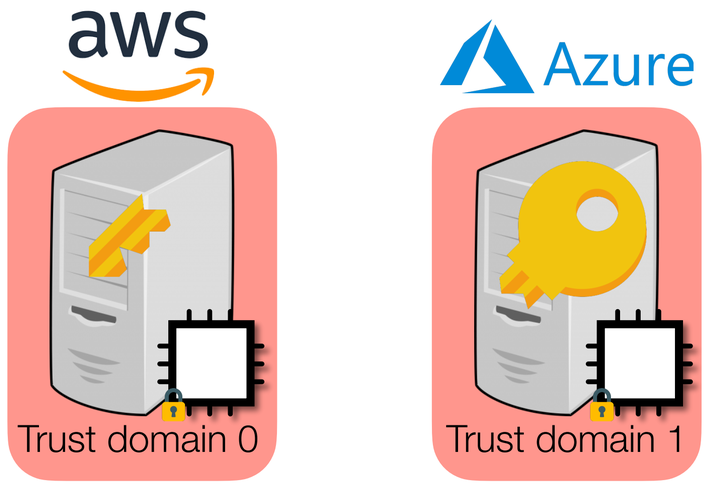
Solution: Secure hardware attests to deployed code.

Our proposal: Audit distributed-trust deployment





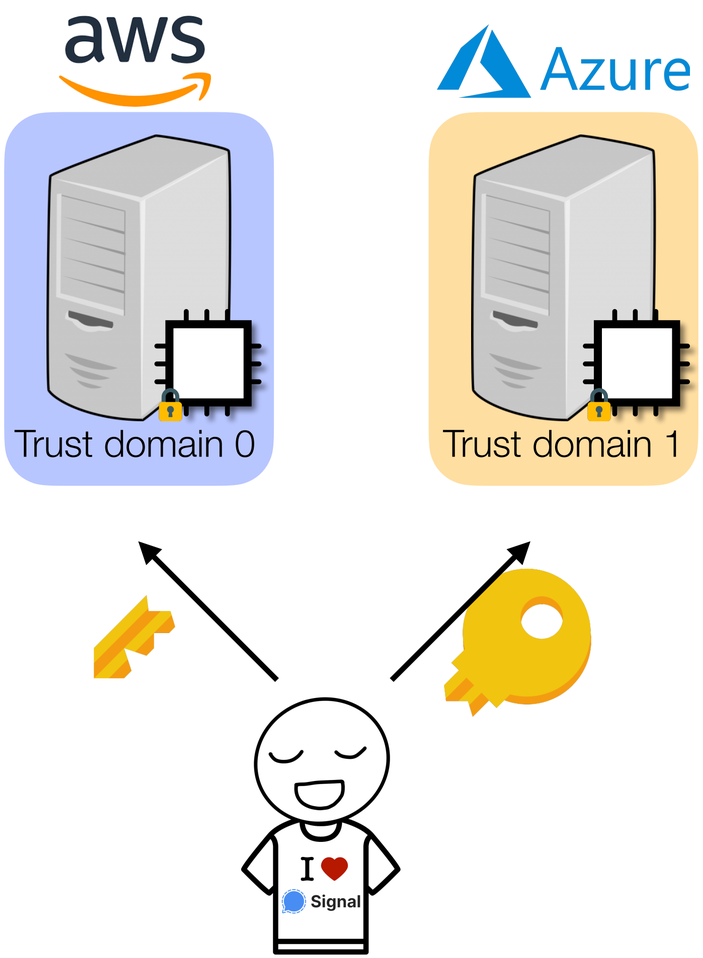
User splits secret across trust domains



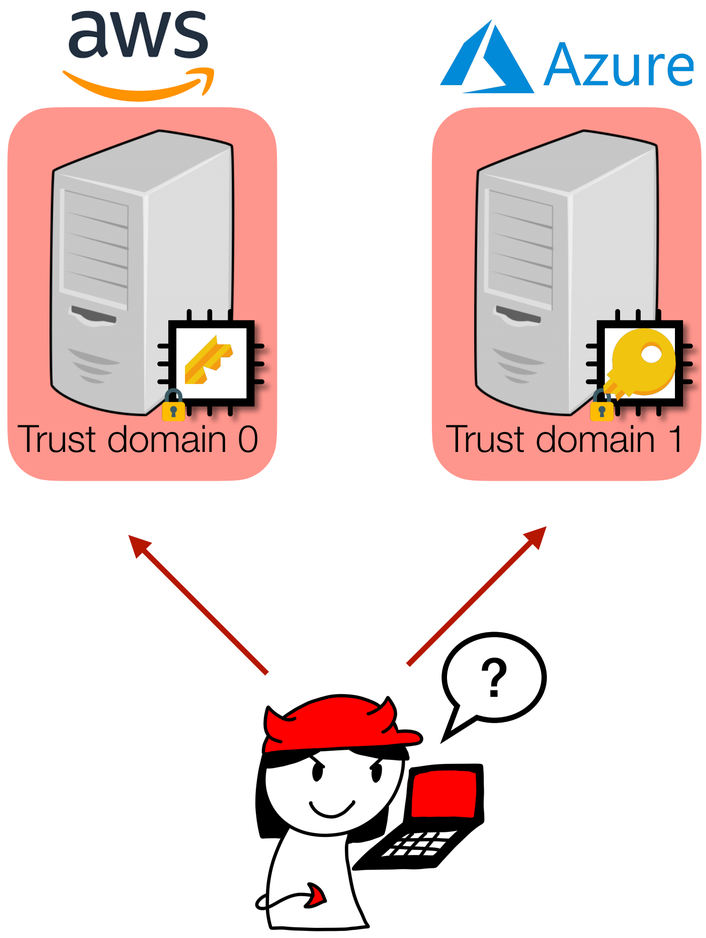
Developer is compromised

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

Our proposal: Audit distributed-trust deployment



User splits secret across trust domains

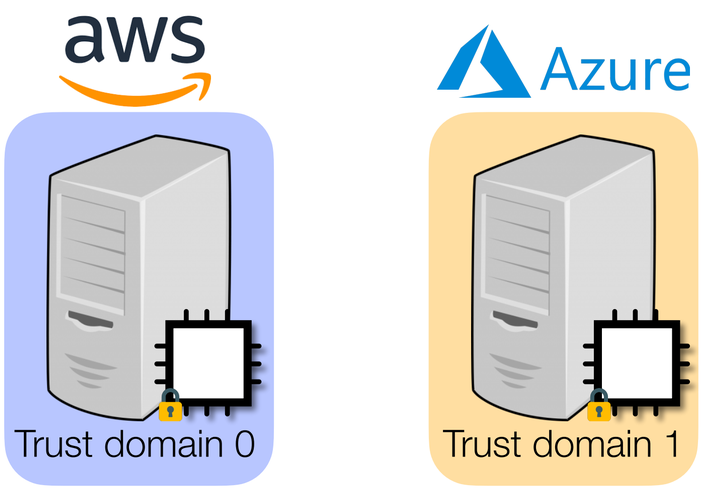


Developer is compromised

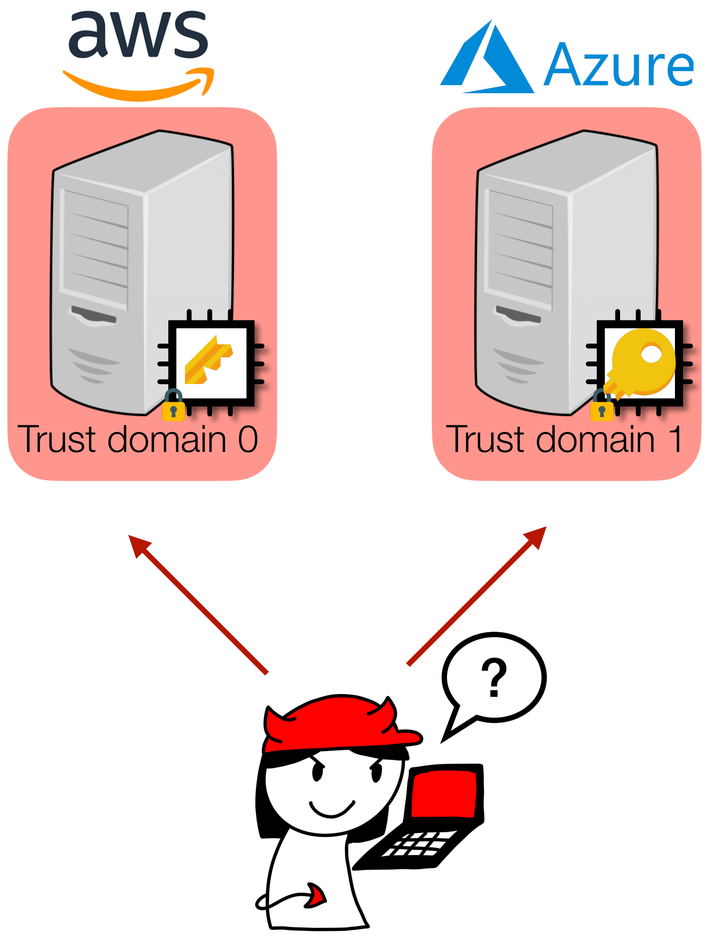
Challenge: How to prevent a compromised developer from learning Bob's secret?

Solution: Secure hardware locks developer out of application memory.

Our proposal: Audit distributed-trust deployment



User splits secret across trust domains

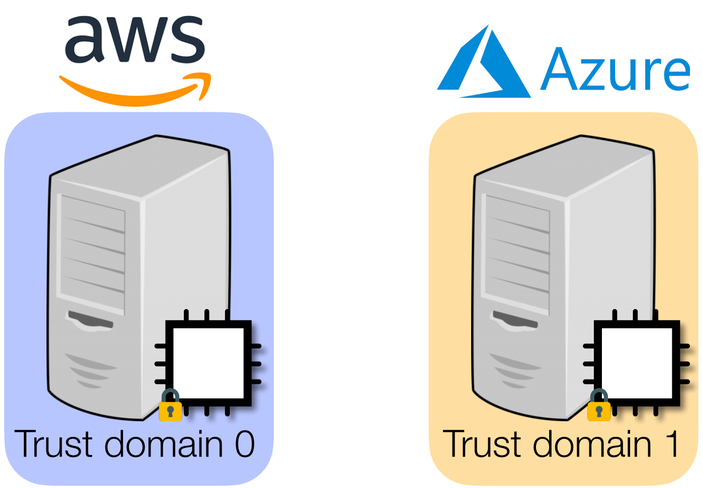


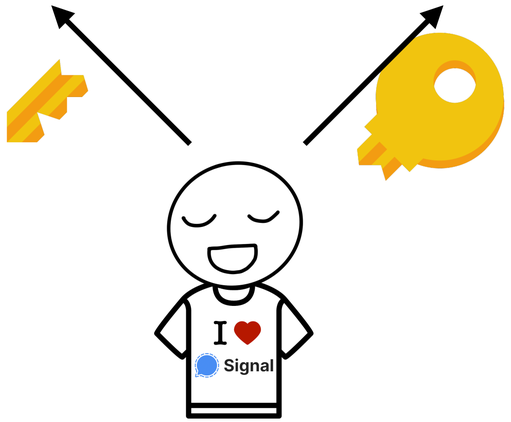
Developer is compromised

Doesn't secure hardware become a central point of attack?

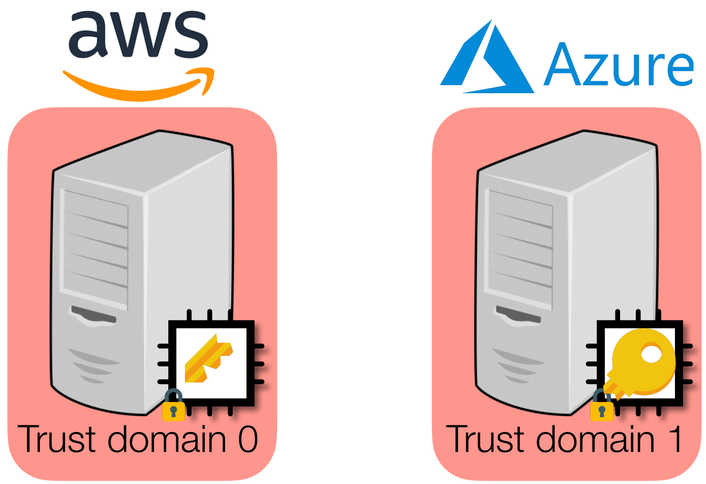
Solution: Heterogeneous secure hardware.

Our proposal: Audit distributed-trust deployment





User splits secret across trust domains



Developer is compromised

See paper:
Auditing updates

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

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

DT ~~FaaS~~

Distrubuted-trust ~~Function~~-as-a-service

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