Atom

Horizontally Scaling Strong Anonymity

Albert Kwon
MIT

Henry Corrigan-Gibbs
Stanford

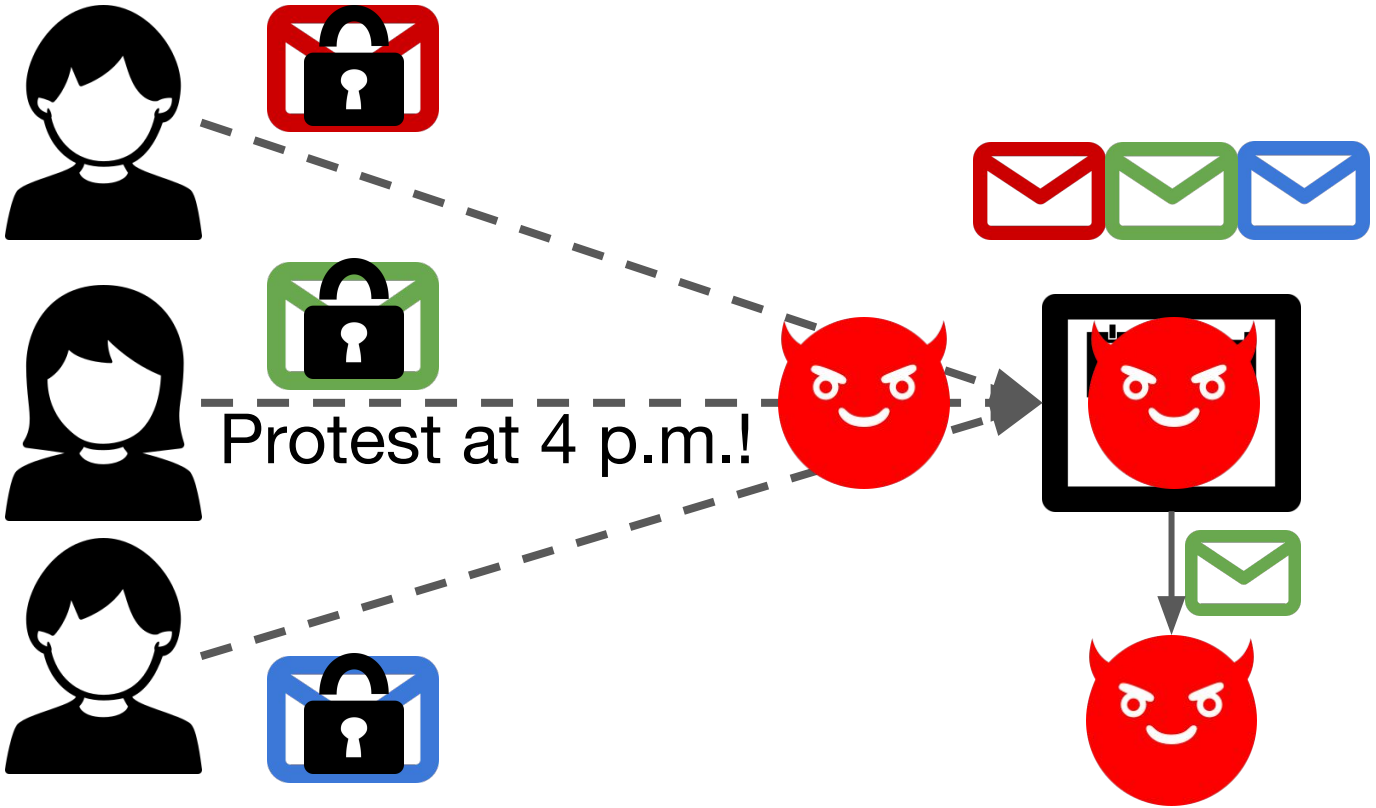
Srinivas Devadas
MIT

Bryan Ford
EPFL

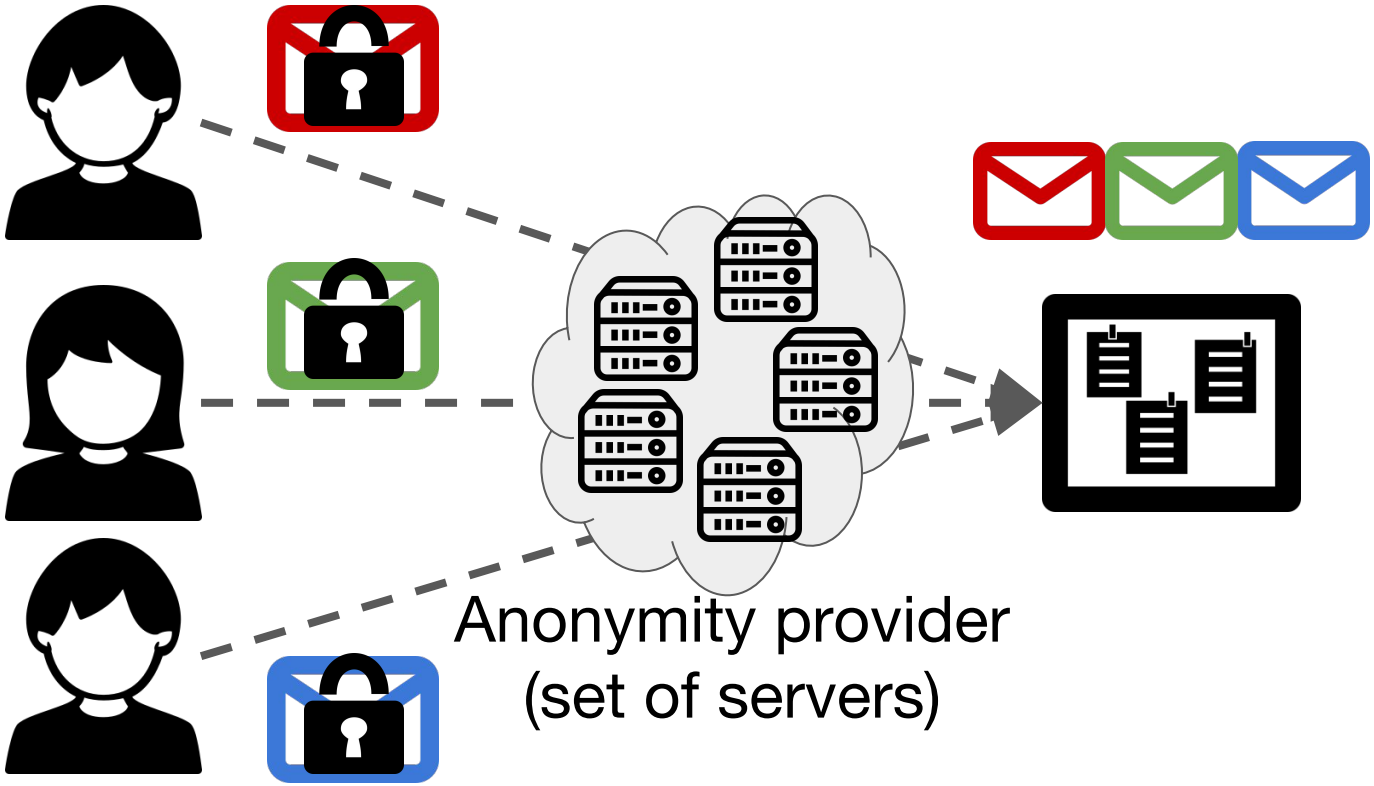
10/30/17, SOSP'17

Motivation

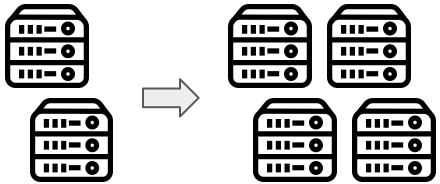
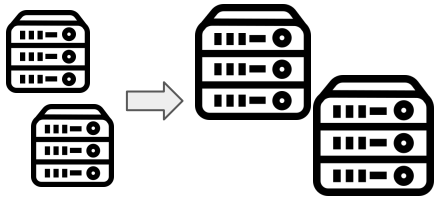
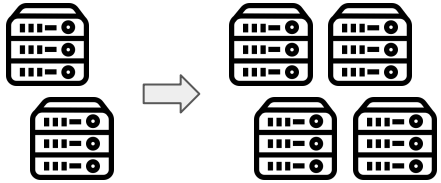
Anonymous bulletin board (broadcast) in the face of global adversary



Anonymous communication networks

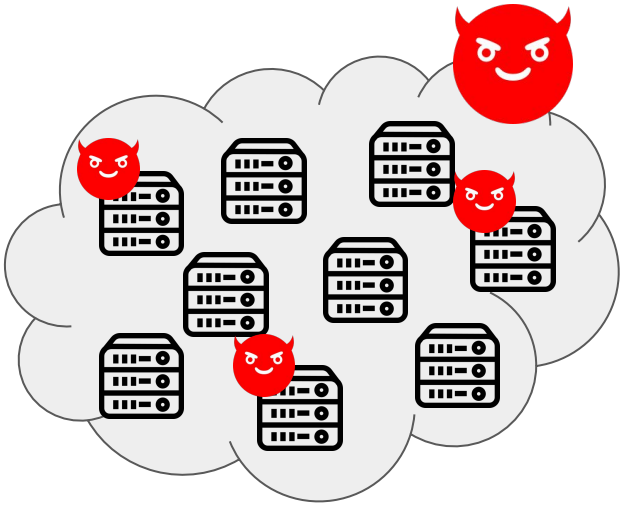


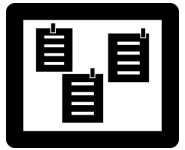
Existing systems vs. Atom

Properties	**Tor** [USENIX Sec'04]	**Riposte** [Oakland'15]	**Atom**
Scaling	Horizontal 	Vertical 	Horizontal 
Latency (1 million users)	< 10s	11 hrs	28min
Anonymity against global adversaries	Vulnerable	Secure	Secure

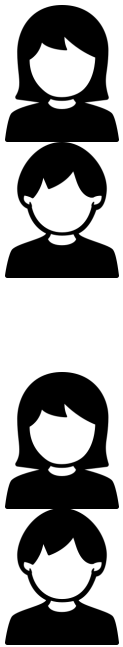
Deployment and threat model

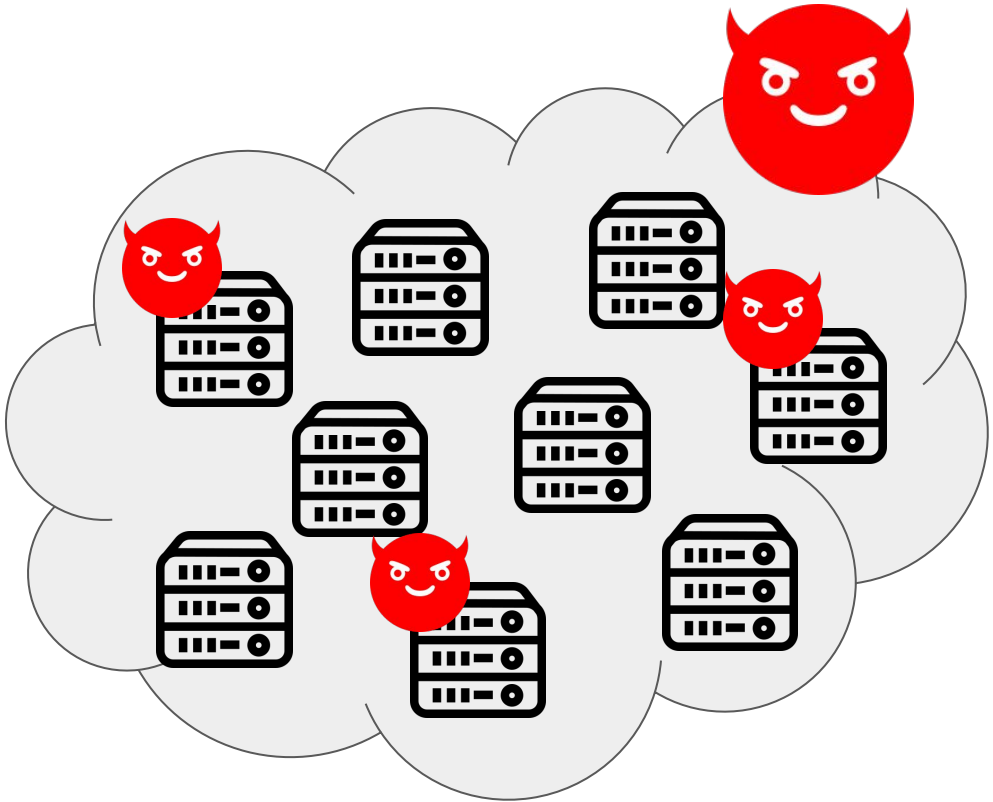
- Global network adversary
- A large number of users are malicious
- Constant fraction of the servers are malicious
  - 20%

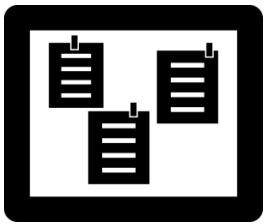




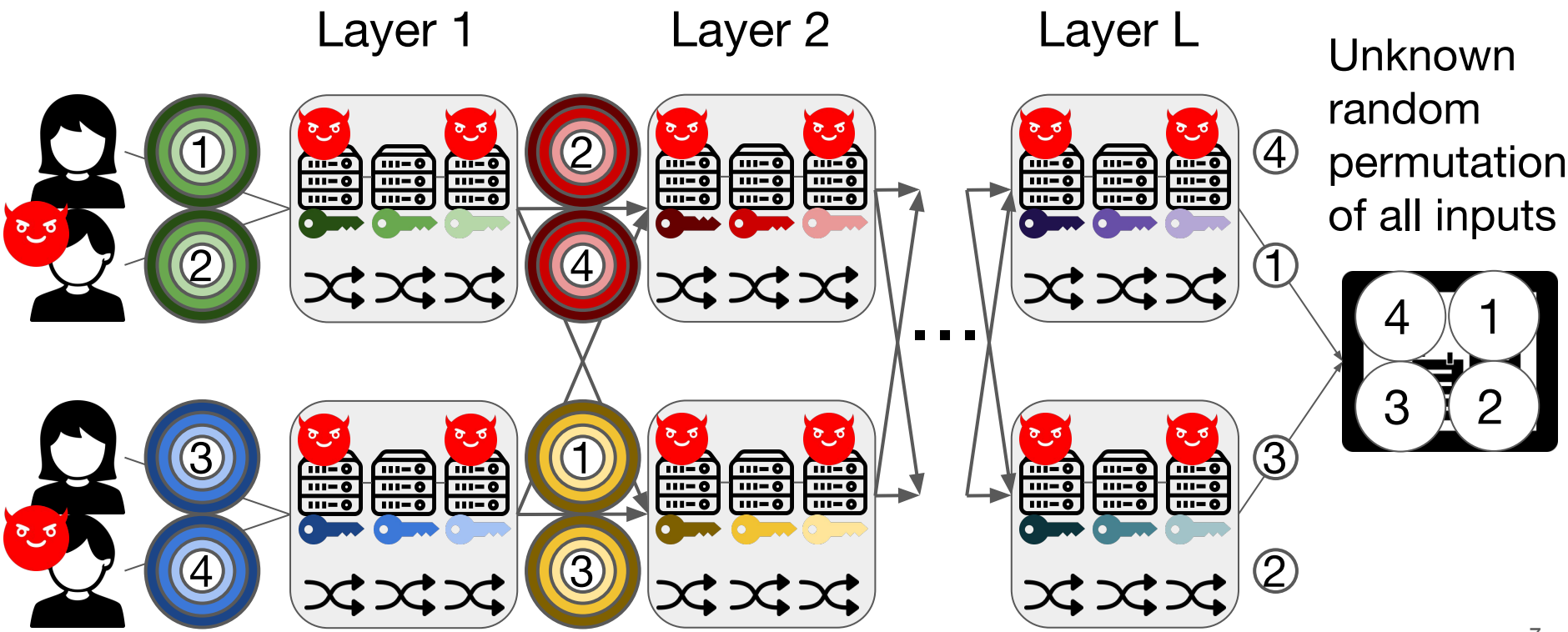
Atom overview







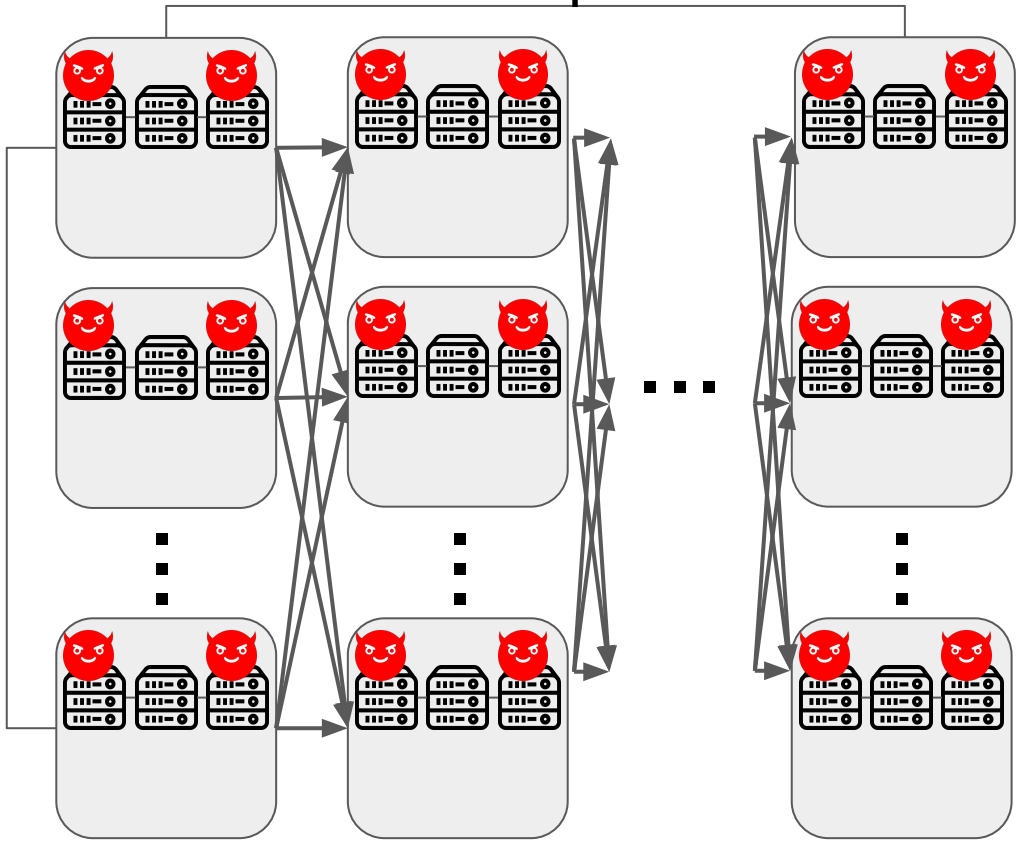
Atom overview



Horizontally scalability

Fixed
(Independent of the width)

Depth

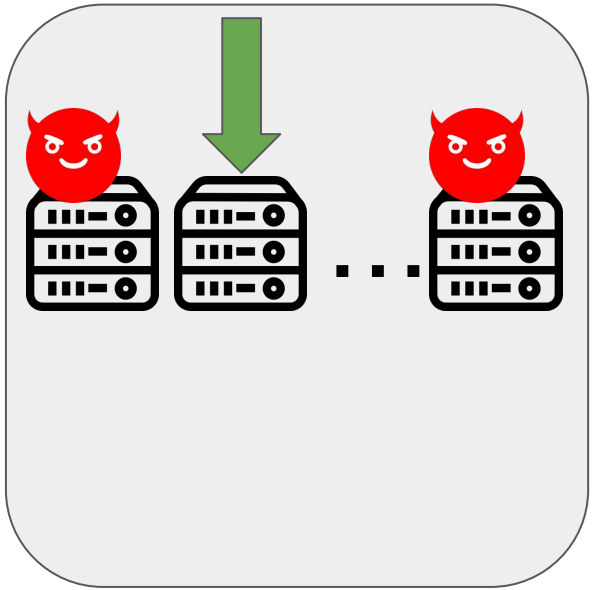


Width

More servers
=> Larger width

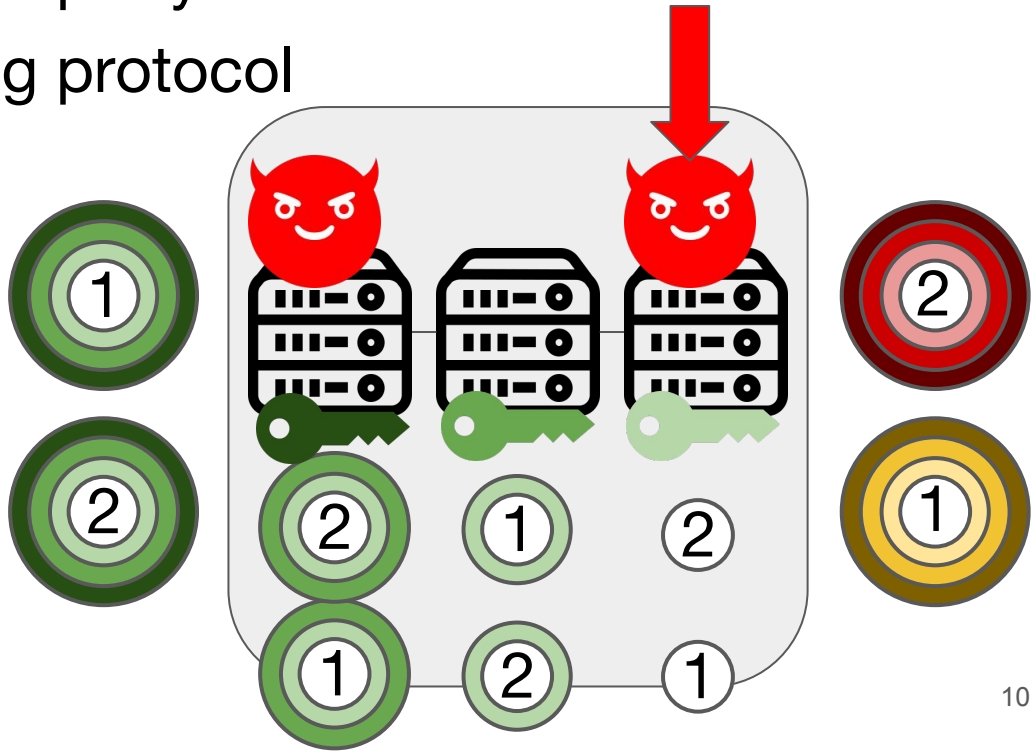
Challenges

1. Guaranteeing anytrust property



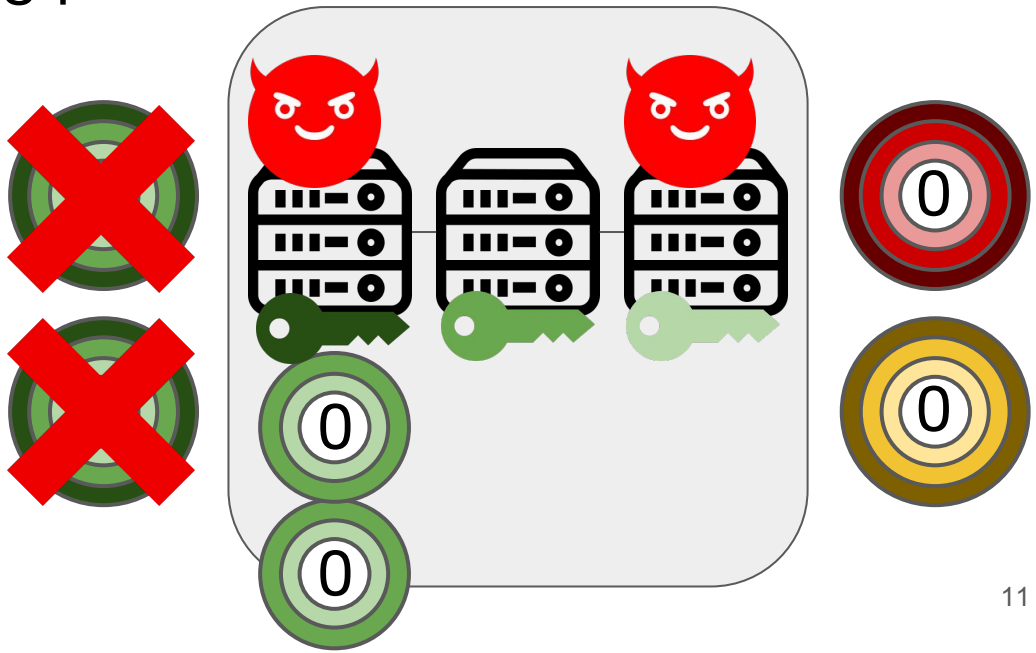
Challenges

1. Guaranteeing anytrust property
2. Group mixing and routing protocol

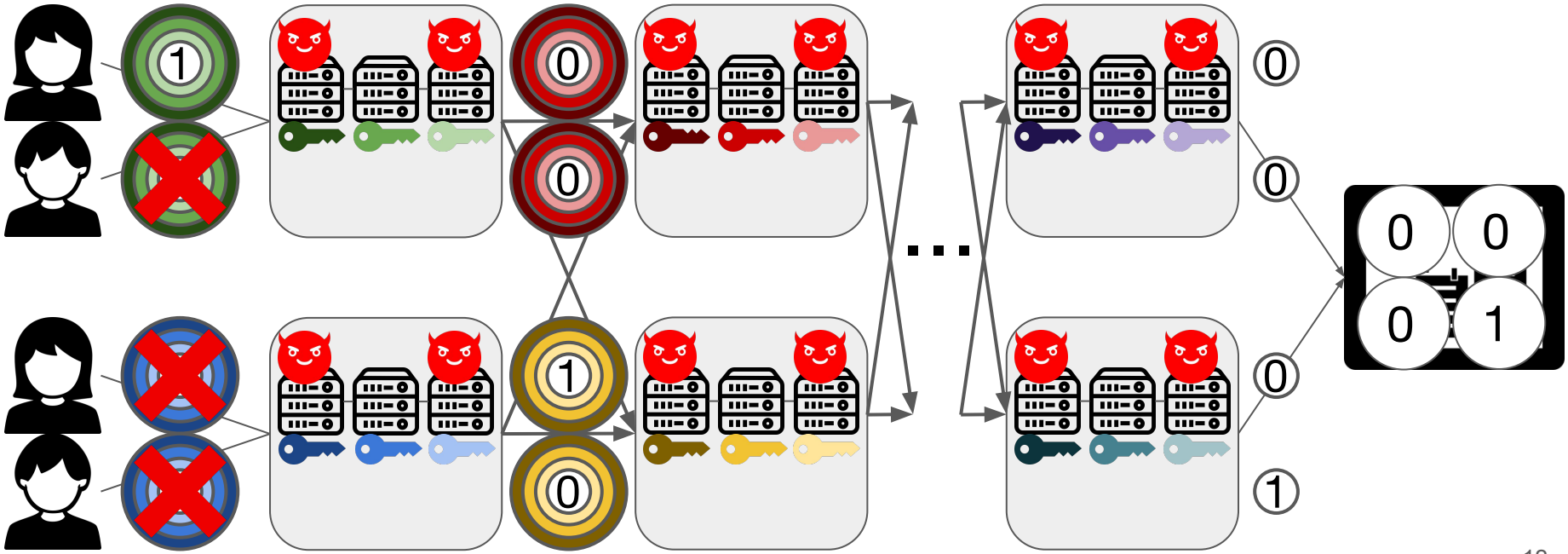


Challenges

1. Guaranteeing anytrust property
2. Group mixing and routing protocol
3. Active adversaries

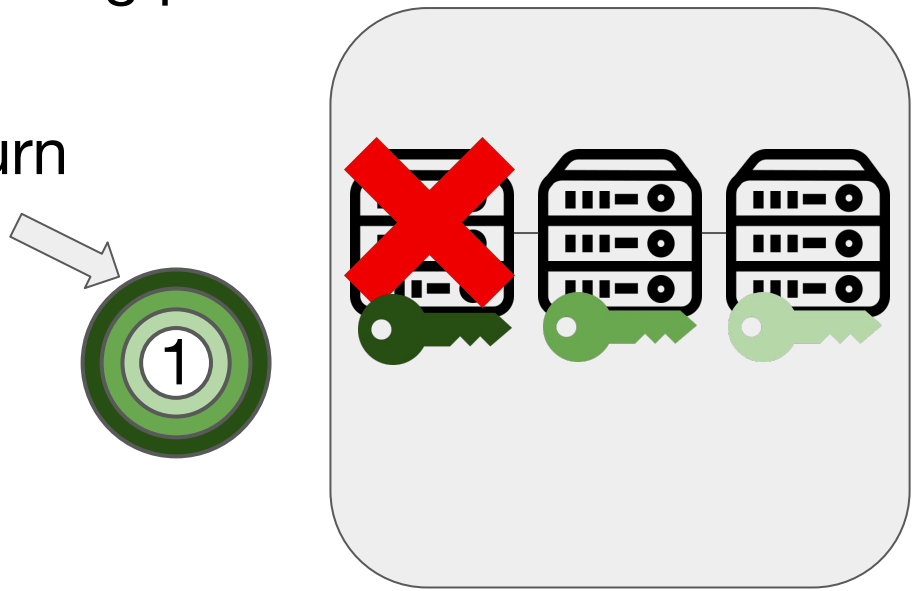


Active attacks



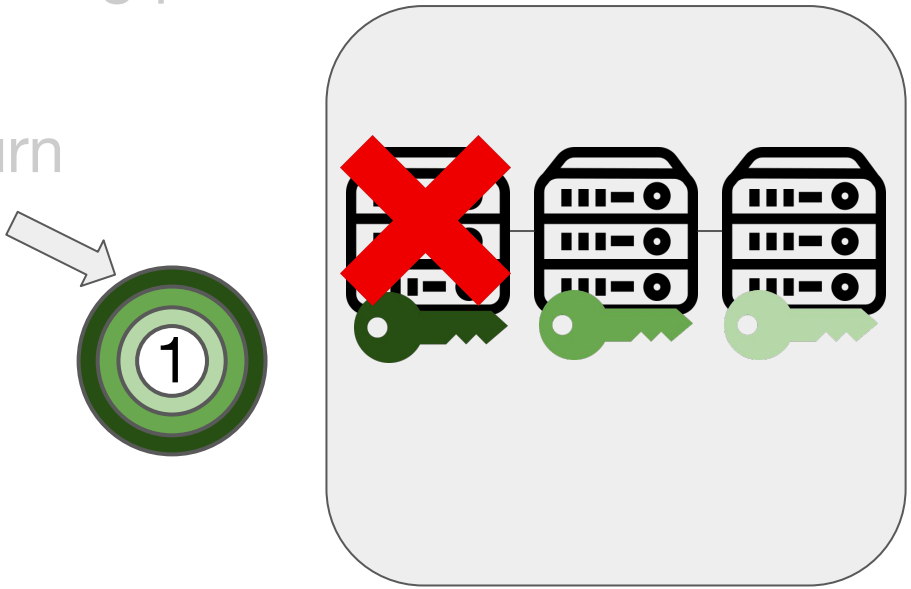
Challenges

1. Guaranteeing anytrust property
2. Group mixing and routing protocol
3. Active adversaries
4. Tolerating server churn

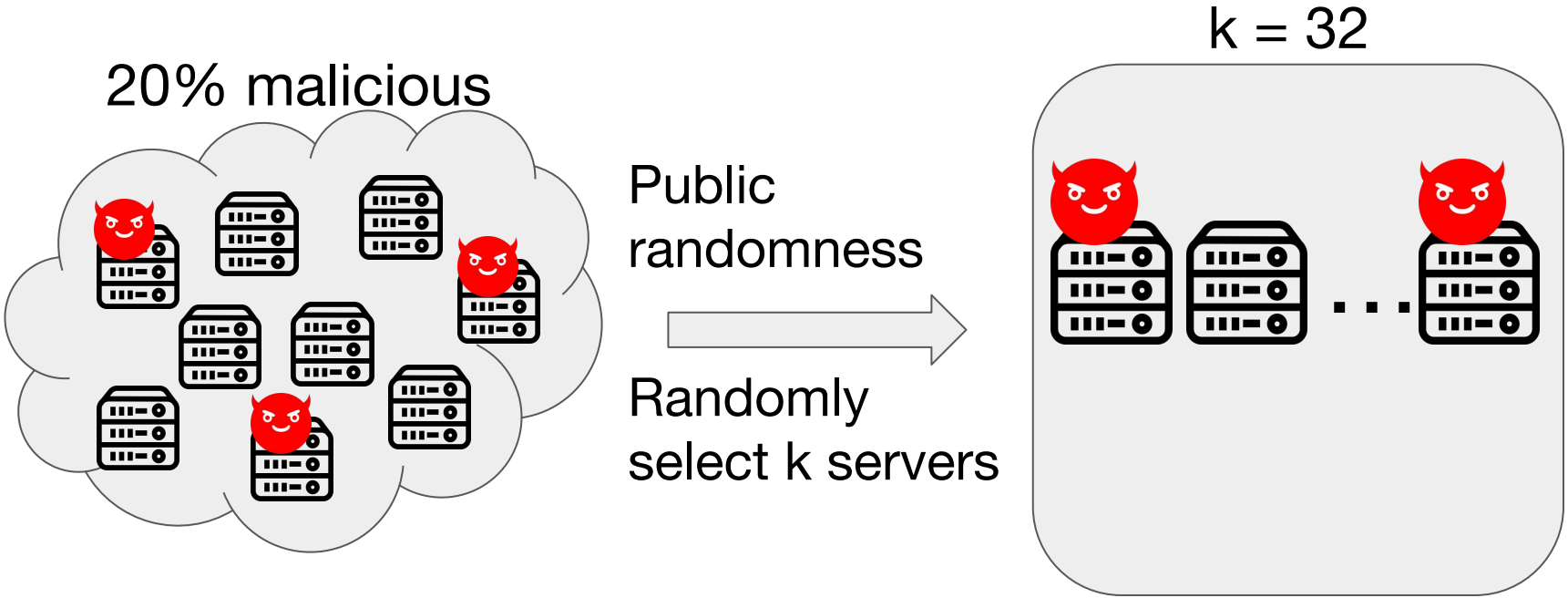


Challenges

1. Guaranteeing anytrust property
2. Group mixing and routing protocol
3. Active adversaries
4. Tolerating server churn



Generating anytrust groups



Pr[group is fully malicious] = 0.2^k

Pr[any group is fully malicious] < (# of groups) $\cdot$ 0.2^k < 2^{-64}

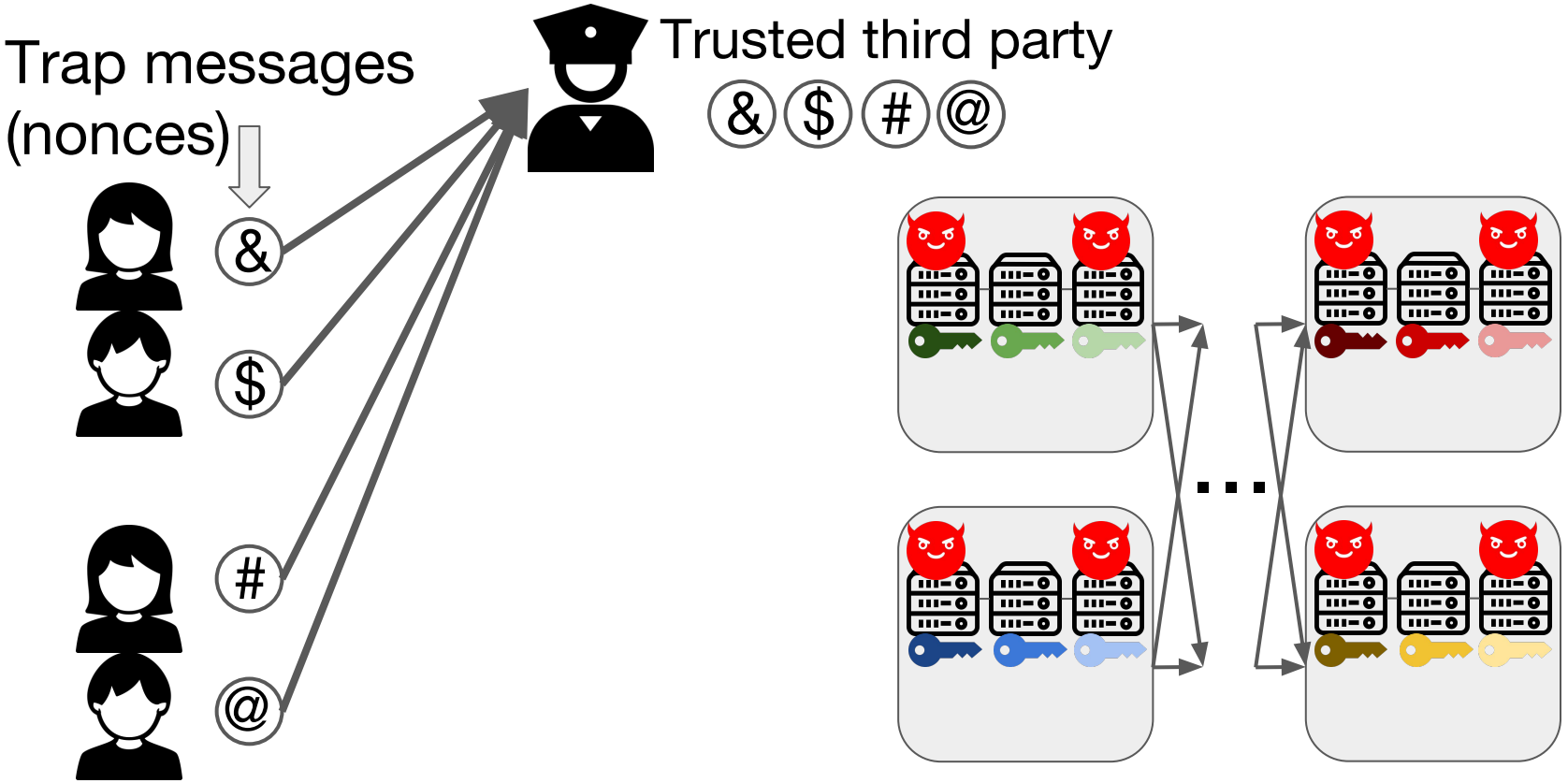
Handling actively malicious servers

Idea: use verifiable trap messages

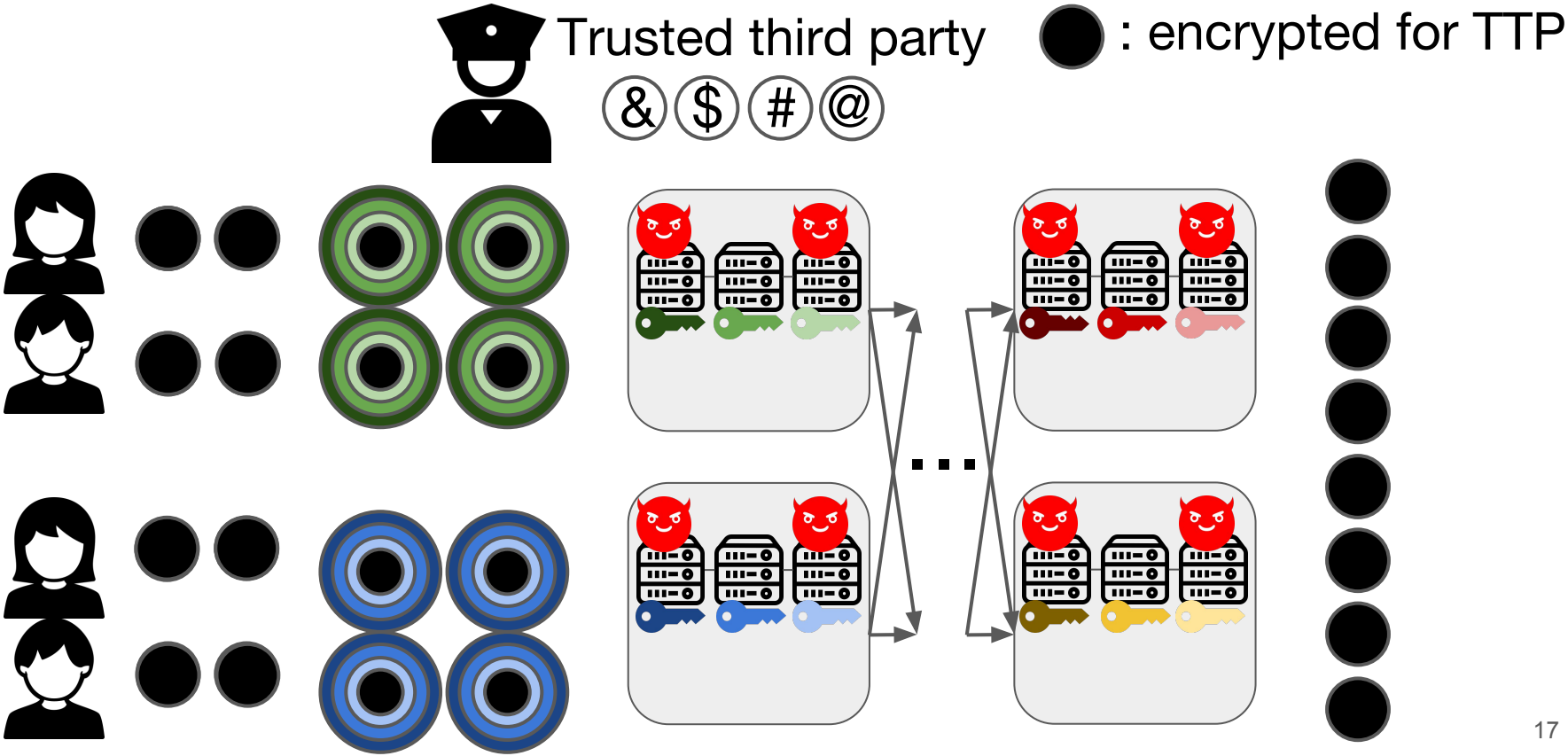
Trap messages (nonces)

Trusted third party

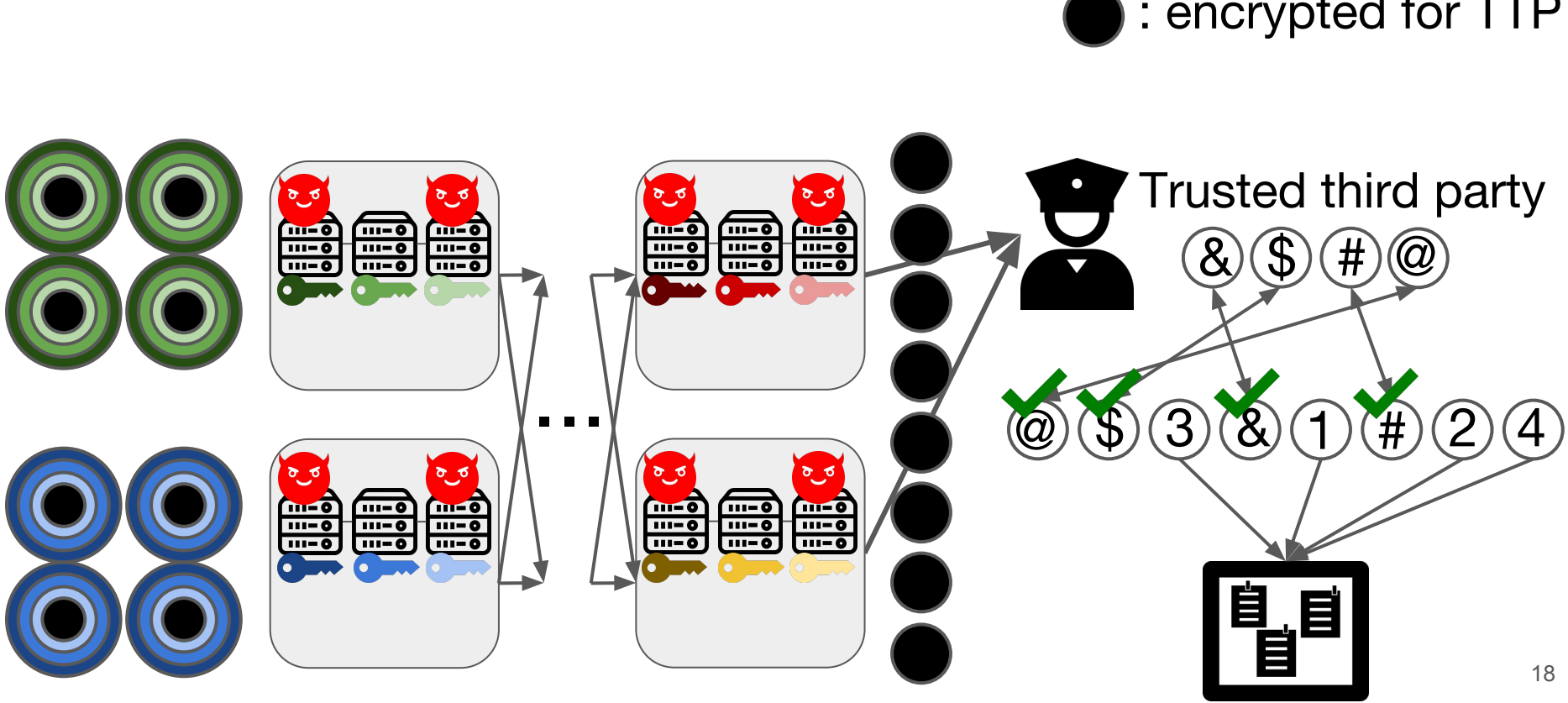
&$#@



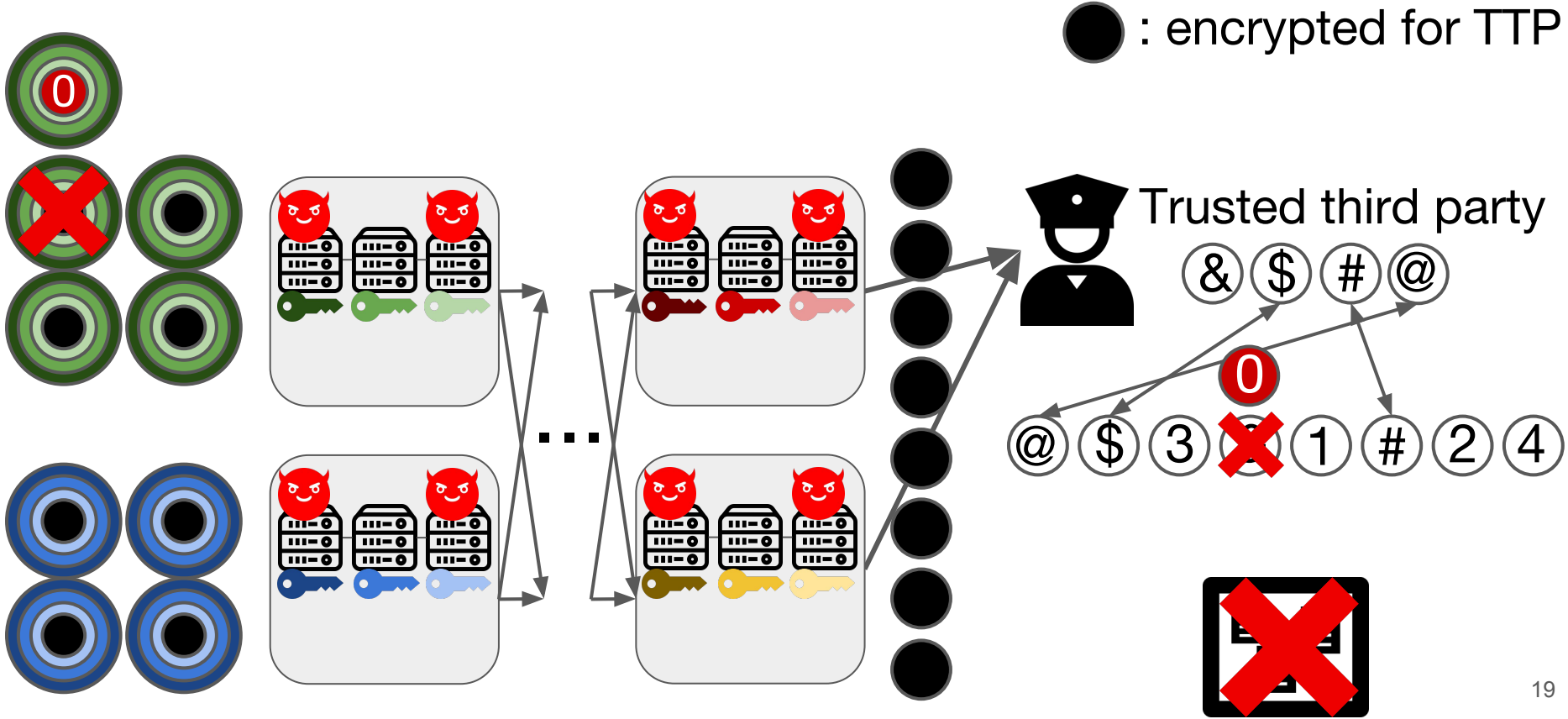
Send trap and real messages in a random order



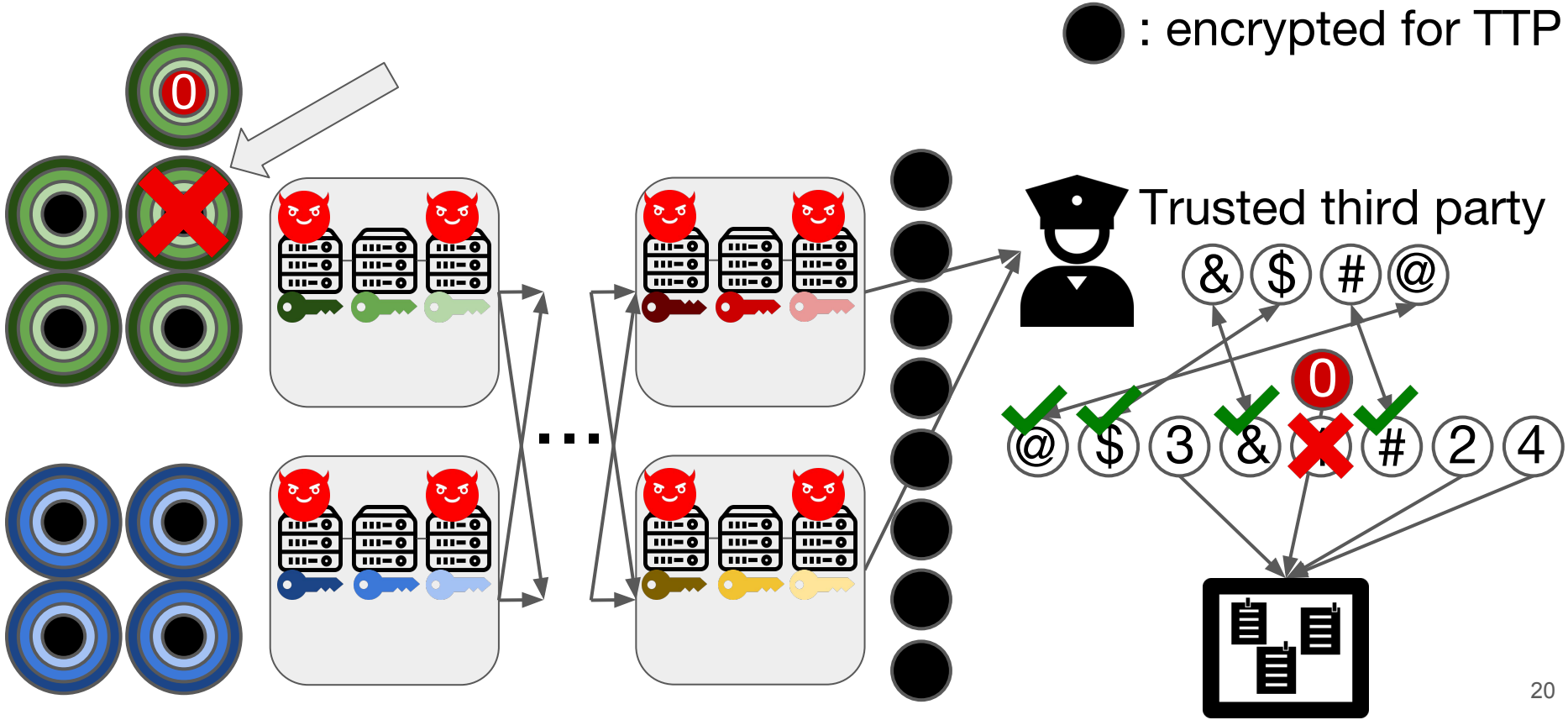
TTP checks for the traps



What happens when a trap message is dropped?



What happens when a real message is dropped?



Improving the trap messages

- Distributing the trust in the third party
- Distributing the trap verification and decryption

Properties of trap-based defense

- If the adversary tampers with any trap, then no plaintext revealed
- Can remove 1 message with probability ½
  - Remove *t* messages with probability 2^{-t}
  - Realistically remove < ~64 msgs
- Reactive

Two modes of operation

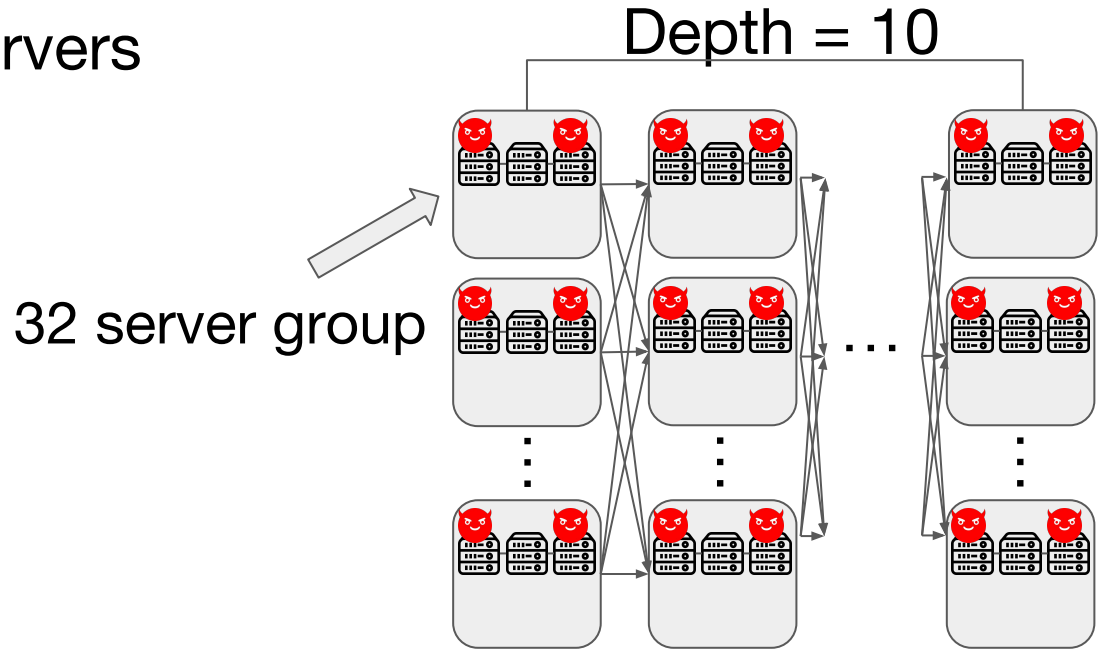
	Trap messages	**Zero-knowledge Proof**
Idea	Verify untamperable traps	Verify protocol with ZKP
Anonymity set size	N - *t*	N
Defense type	Reactive	Proactive
Latency	1x	4x

Implementation

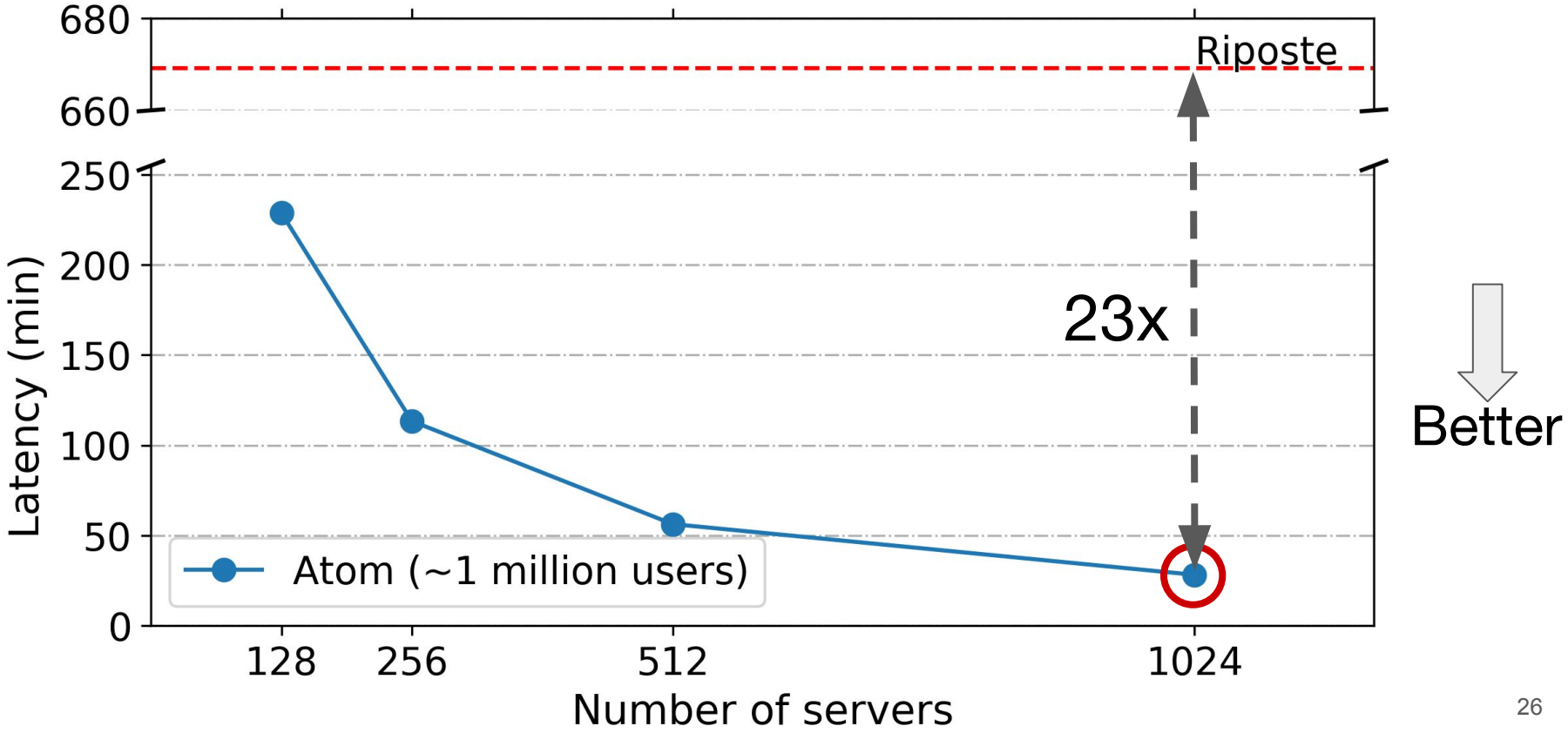
- ~4000 lines of Go
- Both trap and ZKP based defenses
- Code available at github.com/kwonalbert/atom

Evaluation setup

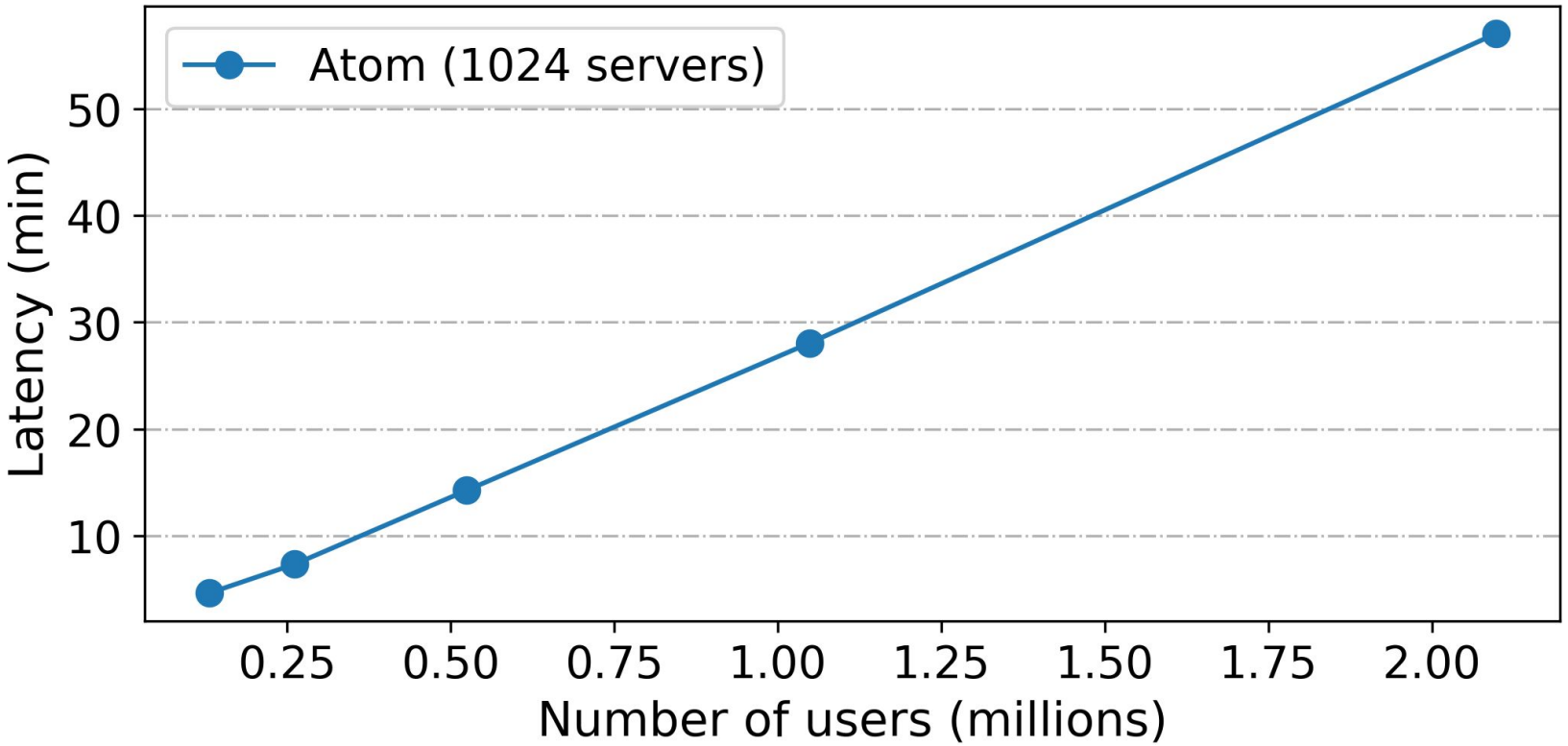
- Heterogenous set of 1024 EC2 servers
  - 80% of the servers were 4-core machines
- 20% malicious servers
- Trap messages
- 160-byte msgs



Latency is inversely proportional to the number of servers



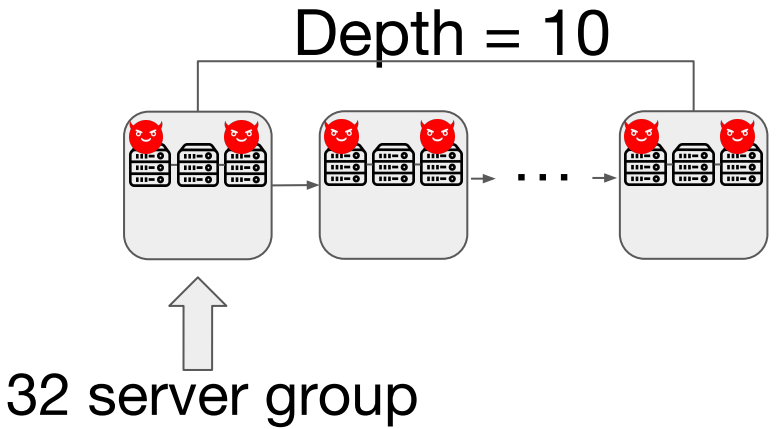
Latency scales linearly with the number of users



Better

Limitations

- Medium to high latency
- Denial-of-service



Related work

- Strong anonymity but veritically scaling
  - Dissent[OSDI'12], Riffle [PETS'16], Riposte [Oakland'15], ...
- Horizontally scaling systems but weaker anonymity
  - Crowds [ACM'99], Mixminion [Oakland'03], Tor [USENIX Sec'04], Aqua [SIGCOMM'13], Loopix [USENIX Sec'17], …
- Distributed mixing
  - Parallel mix-net [CCS'04], matrix shuffling [Håstad'06], random switching networks [SODA'99, CRYPTO'15], ...
- Private point-to-point messaging
  - Vuvuzela [SOSP'15], Pung [OSDI'16], **Stadium [SOSP'17]**

Conclusion

- Atom provides horizontally-scaling strong anonymity
  - Global anonymity set
  - Latency is inversely proportional to the number of servers
- Supports 1 million users with 160 byte msgs in 28min

github.com/kwonalbert/atom

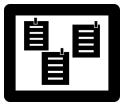
These icons were acquired from thenounprojcet.com, and are under CC BY 3.0 US

Created by H Alberto Gongora

Created by H Alberto Gongora

Created by Anil

Created by Andre Luiz Gollo



Created by Creative Stall