RECIPE

Hardware-Accelerated Replication Protocols

Dimitra Giantsidi, **Emmanouil (Manos) Giortamis**, Julian Pritzi,
Maurice Bailleu, Manos Kapritsos, Pramod Bhatotia

THE UNIVERSITY of EDINBURGH

Technische Universität München

Middleware 2025, Nashville, USA

Distributed systems power everything

ZippyDB

APACHE kafka

kubernetes

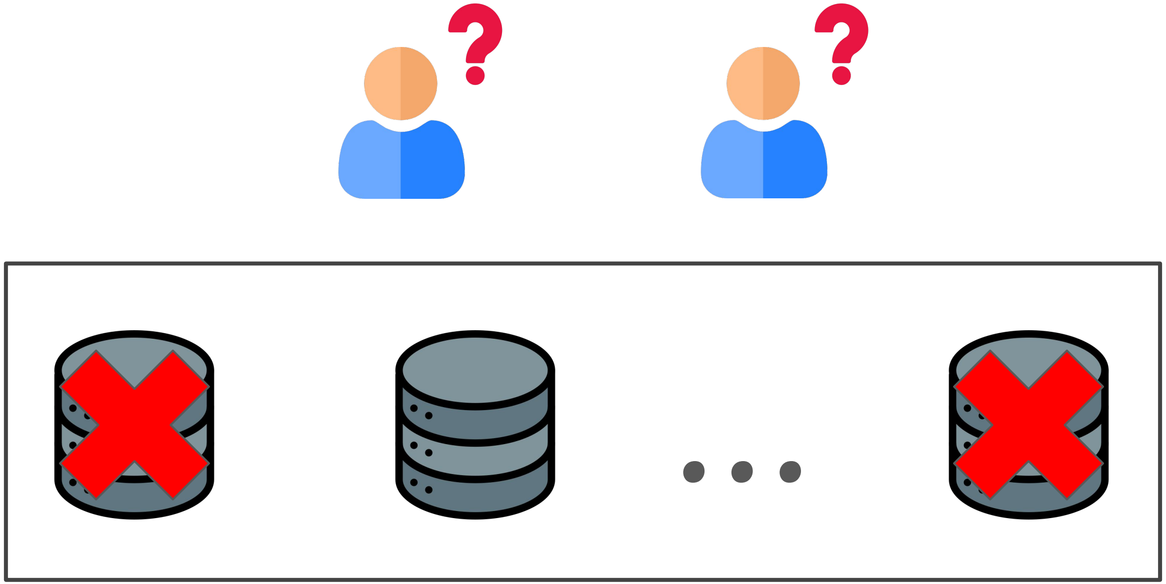
APACHE ZooKeeper

redis

HashiCorp Consul

Distributed systems are the foundation of modern cloud infrastructure

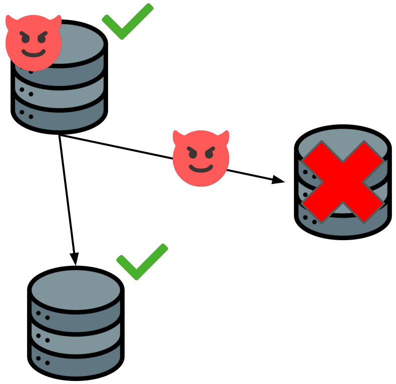
Distributed systems are prone to failures



How to make distributed systems fault-tolerant?

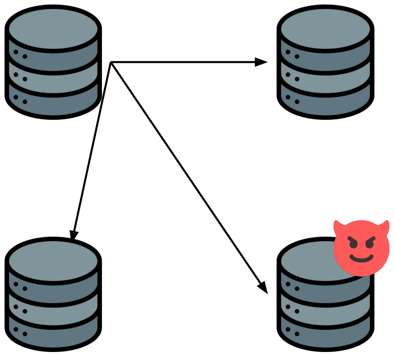
Fault tolerance models

Crash Fault Tolerance (CFT)



CFT system
2f+1 nodes handle ***f*** failures

Byzantine Fault Tolerance (BFT)



BFT system
3f+1 nodes handle ***f*** failures

Which fault tolerance model is the best?

CFT versus BFT

Crash Fault Tolerance (CFT)

- High performance
- Simplicity
- Vulnerability

Byzantine Fault Tolerance (BFT)

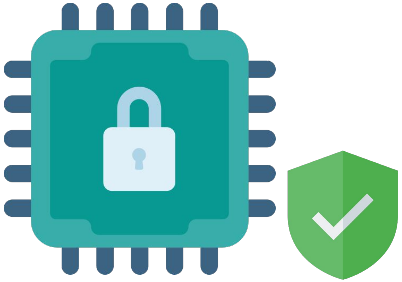
- Robustness
- Complexity
- Poor performance

In the modern untrusted cloud we need BFT guarantees, but they are expensive

Research question

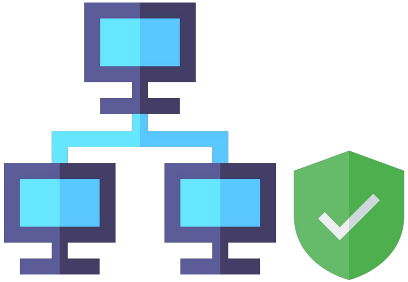
How do we <u>systematically</u> design **trustworthy distributed systems for Byzantine cloud environments** while <u>offering high performance and scalability</u>?

Key insight: Leverage modern hardware for BFT



Trusted computing

BFT robustness



Userspace networking

Performance

We can have CFT simplicity and performance with BFT robustness!

Trusted computing for BFT robustness

- CPU-based Trusted Execution Environments (TEEs)
- TEEs provide hardware isolation → protocol compliance
- BFT with TEEs requires 2f+1 nodes, same as CFT!

Trusted computing can make BFT systems **scalable**

Userspace I/O for BFT systems

- High-throughput and low-latency networking
- Kernel bypass → less system calls
- High performance for TEE-based systems

Userspace networking can make BFT systems **performant**

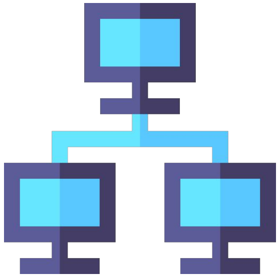
The complete RECIPE

CFT Protocol

- Simplicity 

TEEs

- Robustness 



Userspace I/O

- Performance 

Our proposal

RECIPE: Hardware-Accelerated Replication Protocols
Rethinking Crash Fault Tolerance Protocols for Untrusted Cloud Environments

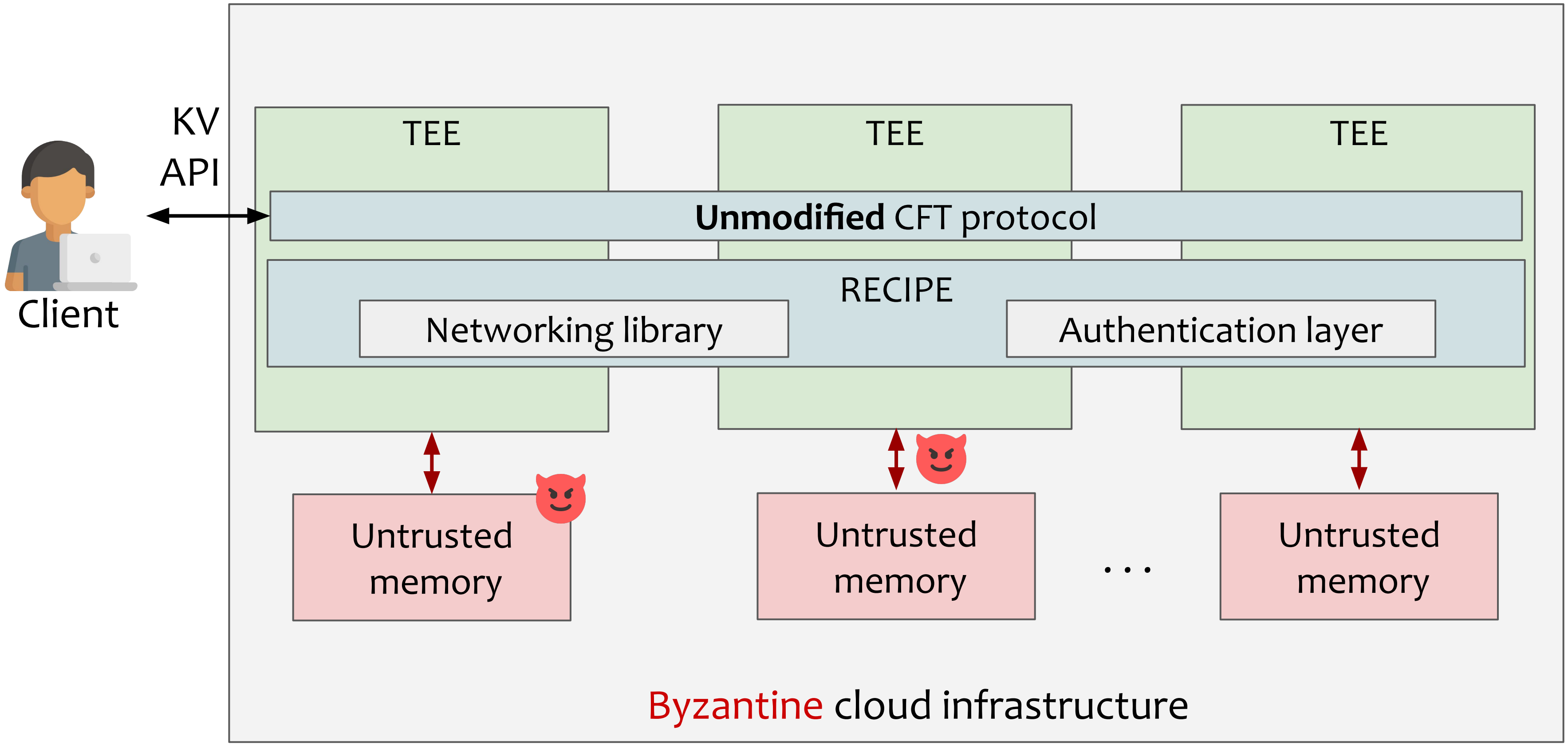
Properties:

- Robustness
  - can tolerate Byzantine actors in the cloud
- Generality
  - transparent application to a broad category of protocols
- Performance
  - scalability and high-throughput

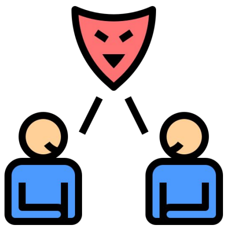
Outline

- ~~Motivation~~
- Overview
- System design
- Evaluation

RECIPE overview

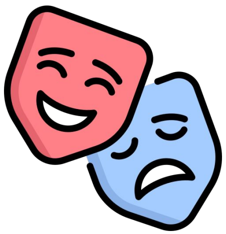


RECIPE ingredients for scalable BFT



#1: Non-equivocation

Do not make conflicting statements
to different nodes



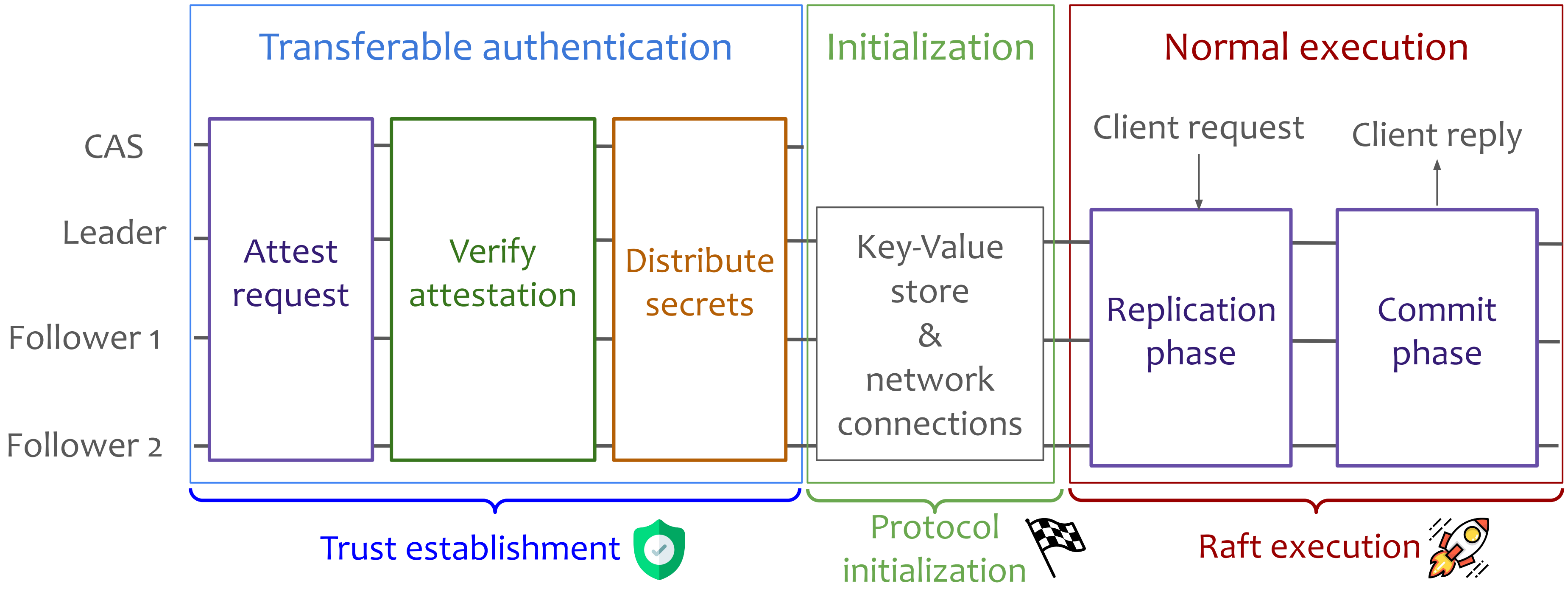
#2: Transferable authentication

Be capable of verifying
the original sender of the message

Allow systems to operate with **$2f+1$** nodes in Byzantine environments[1]

[1] On the (limited) power of non-equivocation, Clement et al., PODC'12.

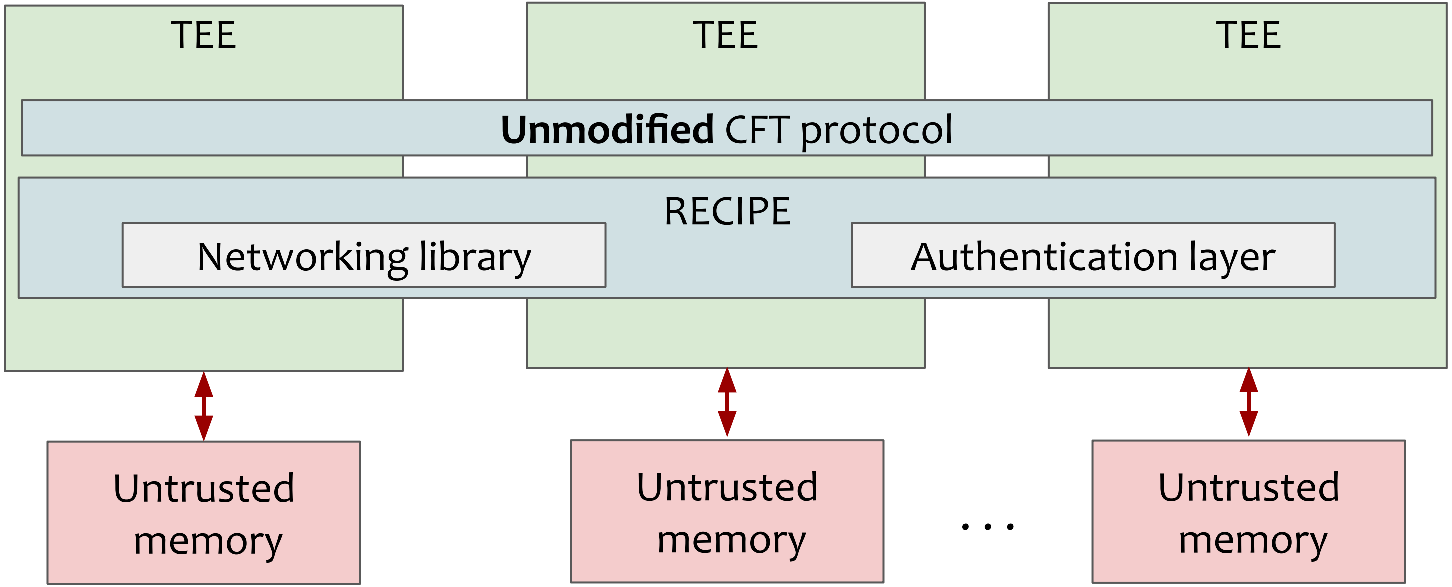
RECIPE protocol: Raft example



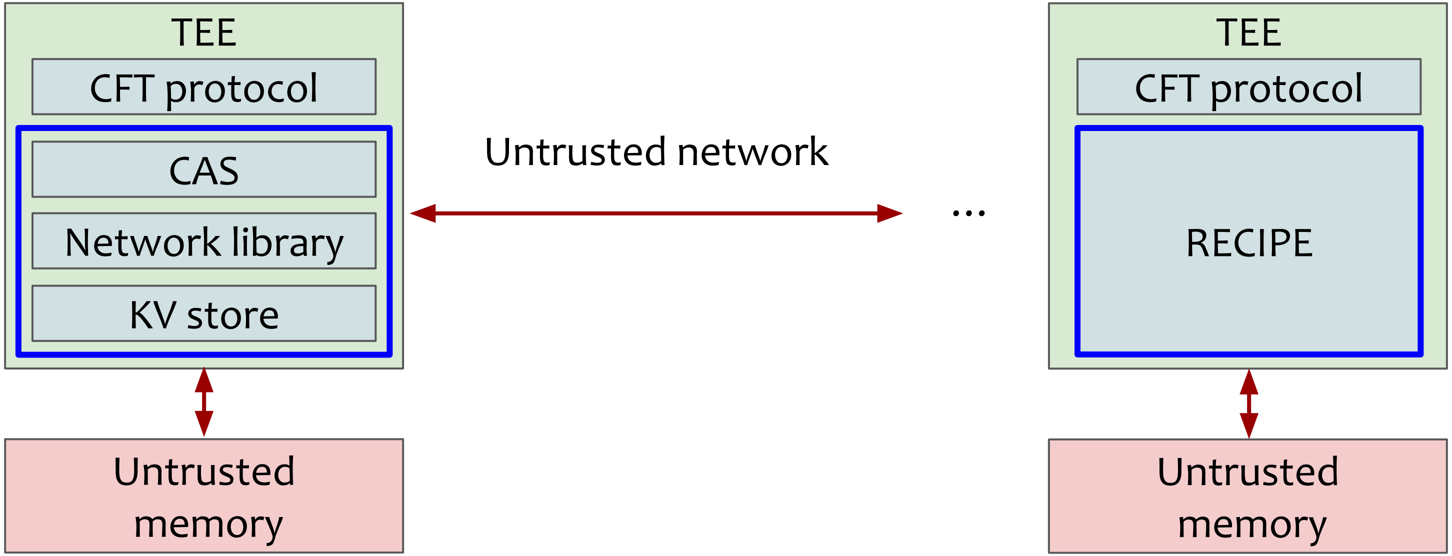
Outline

- ~~Motivation~~
- ~~Overview~~
- System design
- Evaluation

A RECIPE node: System stack

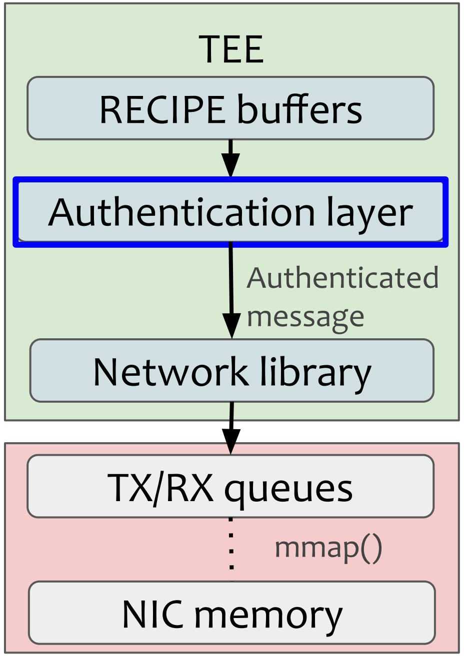


A RECIPE node: System stack



RECIPE network stack and APIs

- RECIPE authentication layer
  - transferable authentication, *i.e.*, MACs
  - non-equivocation, *i.e.*, sequence numbers
- RECIPE network library
  - userspace I/O within the TEEs
- RECIPE network APIs
  - authenticated message format
  - RPC-based generic APIs

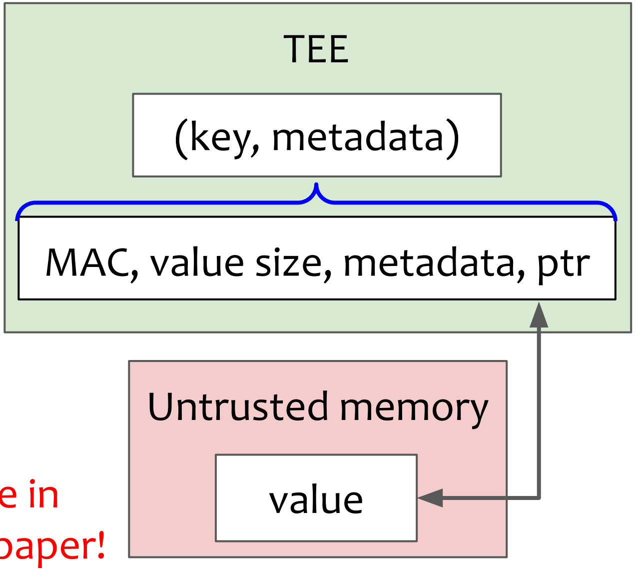


RECIPE implements user-space **trusted** networking with **generic** APIs

RECIPE Key-Value store

- Overcomes the limited trusted memory
  - splits keys and values
- Maintains the original CFT protocol semantics
  - increases the trust to individual nodes
  - e.g., local (linearizable) reads
- Also, offers confidentiality through encryption
  - not-provided by BFT

More in the paper!



RECIPE KV store offers **performance** while keeping **the protocol unchanged**

Outline

- ~~Motivation~~
- ~~Overview~~
- ~~System design~~
- Evaluation

Evaluation

Questions:

- What is the performance of RECIPE protocols w.r.t. the state-of-the-art BFT?
- How much overhead do TEEs have in RECIPE?
- What is the performance of RECIPE networking w.r.t. the state-of-the-art?

Experimental setup:

- Distributed system with 3x Intel i9-9900K @3.60GHz. 8 cores
- 40GbE QSFP+ network switch
- YCSB benchmarks, 10k keys, Zipfian distribution

RECIPE application on distributed systems

- We classify CFT protocols
- We select representative protocols
- We transform them with RECIPE

	Leader-based	Leader-less
Total order	**Raft [ATC'14]**	**AC [HPDC'17]**
Per-key order	**CR [OSDI'04]**	**ABD [PODC'90]**

BFT robustness with CFT performance!

R-Raft	R-CR	R-AC	R-ABD

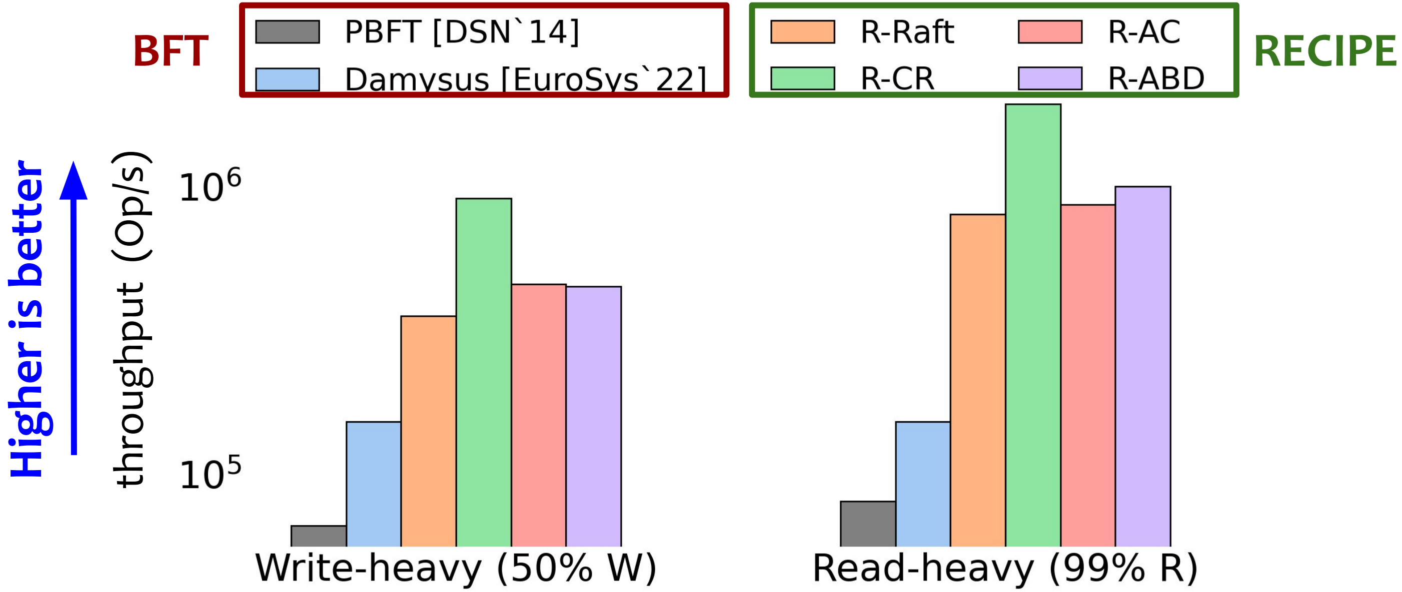
RECIPE is generic and applicable to *any* strongly consistent CFT protocol

Evaluation

Questions:

- What is the performance of RECIPE protocols w.r.t. the state-of-the-art BFT?
- How much overhead do TEEs have in RECIPE?
- What is the performance of RECIPE networking w.r.t. the state-of-the-art?

RQ1: RECIPE performance vs. BFT



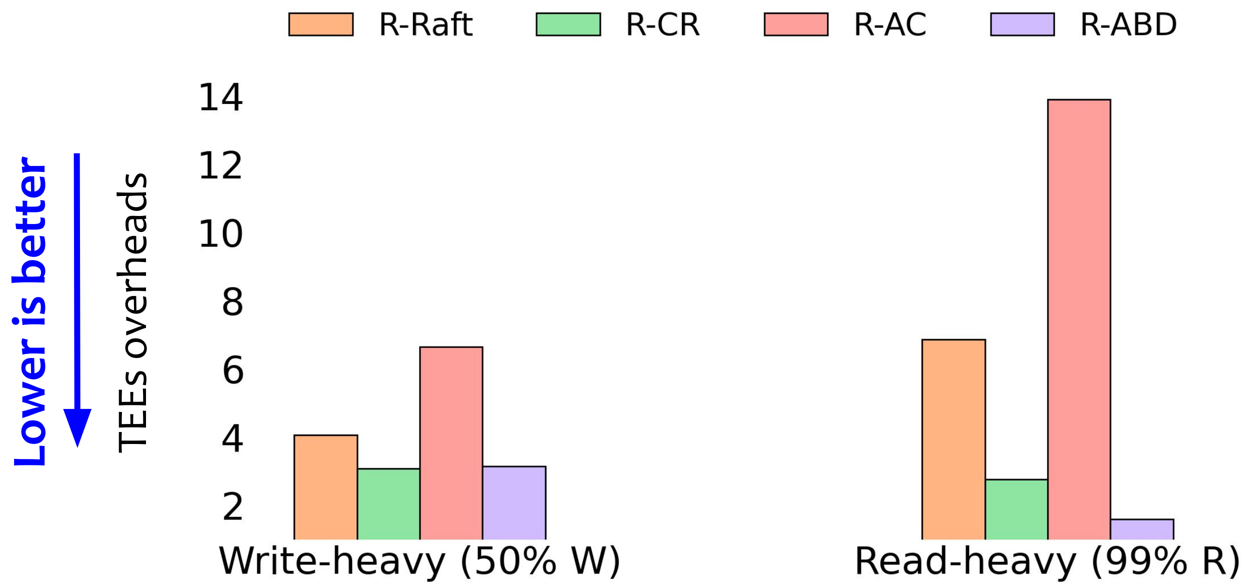
RECIPE achieves **5-20x better performance** compared to state-of-the-art BFT

Evaluation

Questions:

- What is the performance of RECIPE protocols w.r.t. the state-of-the-art BFT?
- How much overhead do TEEs have in RECIPE?
- What is the performance of RECIPE networking w.r.t. the state-of-the-art?

RQ2: TEE overheads



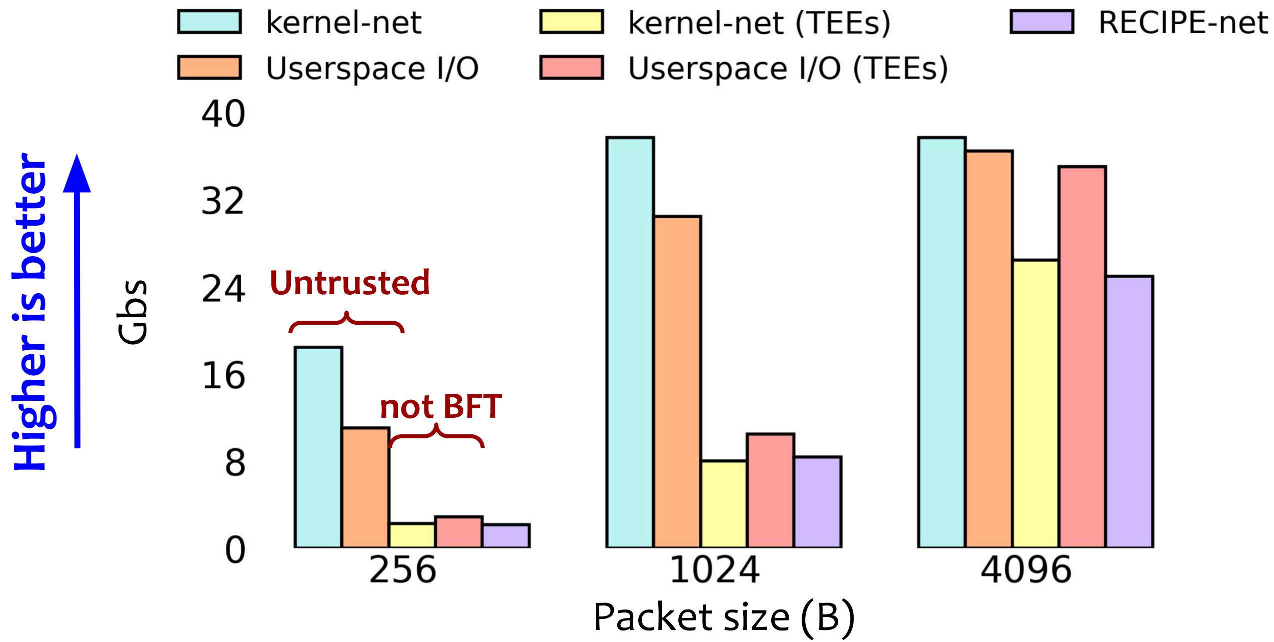
TEEs in RECIPE add **2-14x times slowdown** in throughput

Evaluation

Questions:

- What is the performance of RECIPE protocols w.r.t. the state-of-the-art BFT?
- How much overhead do TEEs have in RECIPE?
- What is the performance of RECIPE networking w.r.t. the state-of-the-art?

RQ3: RECIPE's networking performance



RECIPE networking offers **BFT guarantees** while maintaining performance

Summary

The CFT vs. BFT conundrum:

- CFT protocols are efficient but unsuitable for the untrusted cloud
- BFT protocols are robust but expensive and complex

RECIPE:

- BFT robustness with high performance and scalability
- Applicable to any strongly consistent CFT protocol

Paper **Code**