# All about Eve: Execute-Verify Replication for Multi-Core Servers[1]

*OSDI’12*

Presenters: Yubo Wang, Chongzhou Fang

## Outline

## Introduction

1. **execute-then-verify** vs. **agree-then-execute**

2. **deterministic execution** vs. **Nondeterministic interleaving of requests**

# Protocol Overview

Execution stage:

1. Batching
2. Mixing
3. Executing (in parallel)

Verification stage:

1. Agreement
2. Commit or rollback

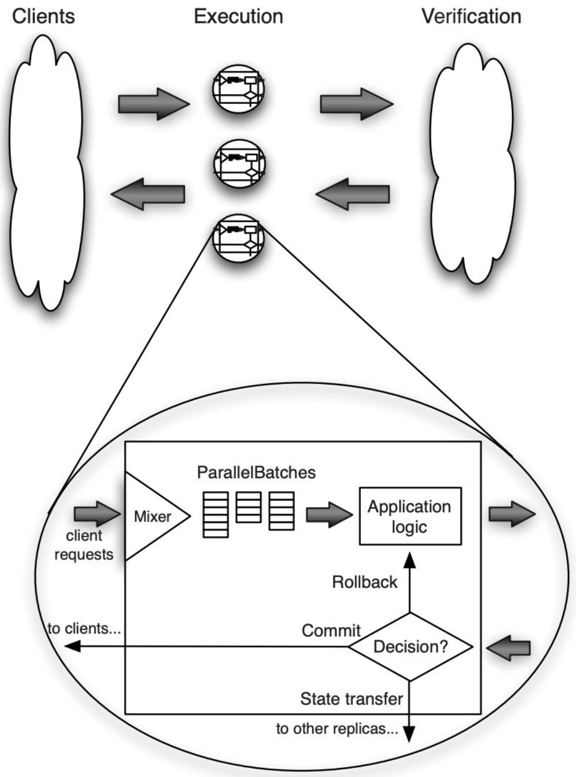
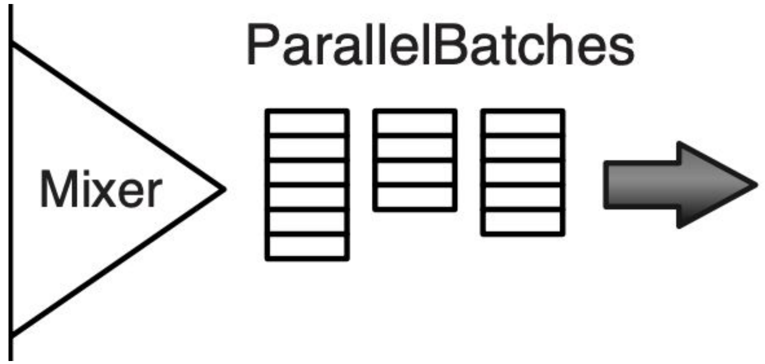


**Figure 1:** Overview of Eve.

# Execution Stage: Mixer Design

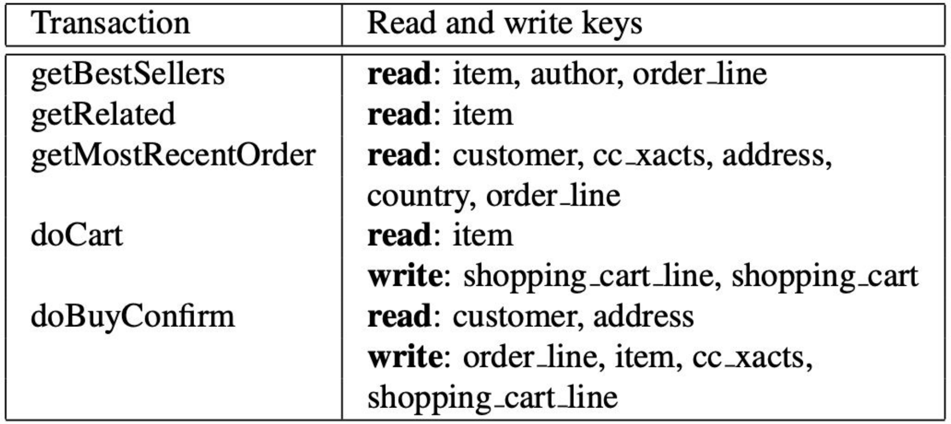
1. parallelBatchList
2. readSet
3. writeSet



## Execution Stage: Mixer Design

An example for mixer:

1. parallelBatchList
2. readSet
3. writeSet

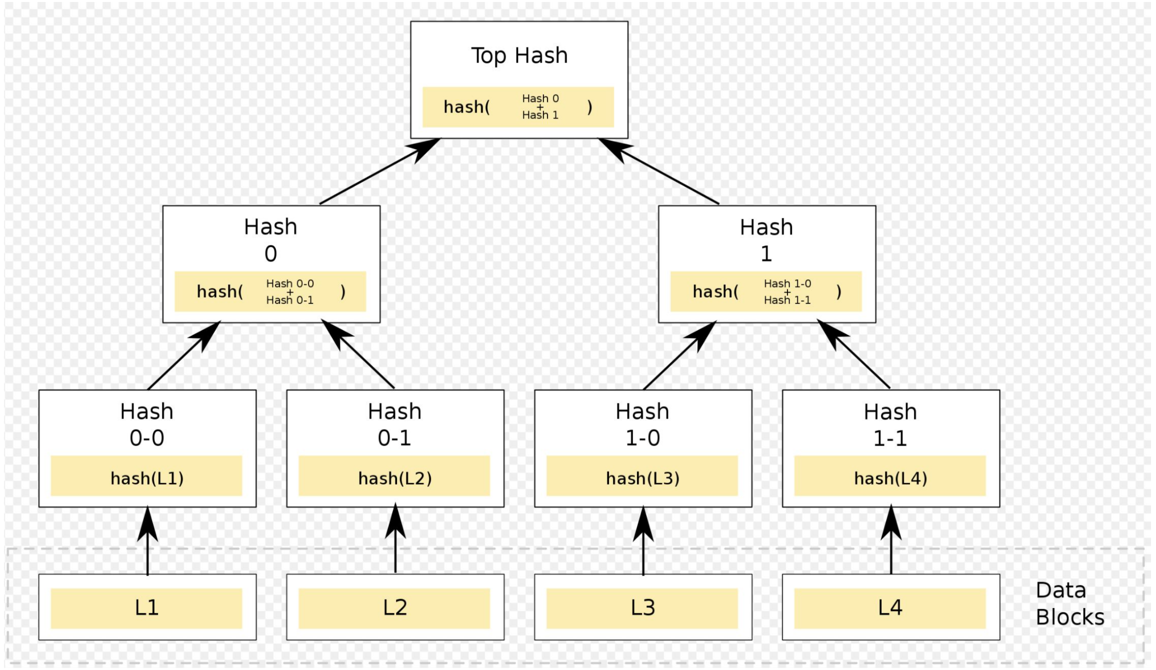


| Transaction | Read and write keys |
|---|---|
| getBestSellers | **read**: item, author, order_line |
| getRelated | **read**: item |
| getMostRecentOrder | **read**: customer, cc_xacts, address, country, order_line |
| doCart | **read**: item<br>**write**: shopping_cart_line, shopping_cart |
| doBuyConfirm | **read**: customer, address<br>**write**: order_line, item, cc_xacts, shopping_cart_line |

**Figure 2:** The keys used for the 5 most frequent transactions of the TPC-W workload.

# Execution Stage: Stage Management

Deterministic Merkle Tree[2]



## Verification Stage

- Goal:
  - Check whether tokens produced by execution replicas match

- Method:
  - Verification Protocol
  - Enough tokens match: Success, commit
  - Not enough: Divergence, roll-back

## Verification Stage

- Optimization for Read-Only Requests
  - **First** executed without involving verification stage
  - Enough replies match:
    - Clients receive values
  - Otherwise:
    - Reissued as regular requests

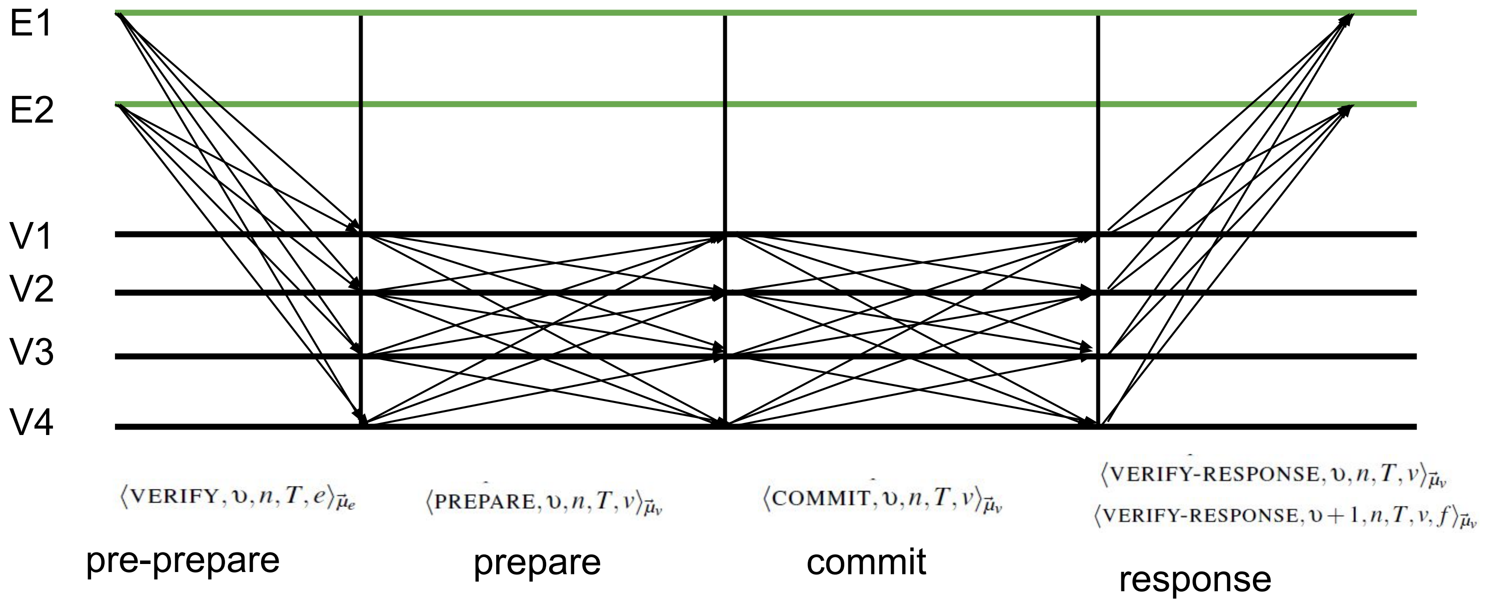
# Verification Stage - *Asynchronous BFT*

- Difference between PBFT and Verification Protocol

1. In PBFT: agree on the output of a single node
    - In Eve: agree on the behavior of execution replicas
2. In PBFT: agree on the inputs to the state machine
    - In Eve: agree on the outputs of the state machine

# Verification Stage - *Asynchronous BFT*

- Verification process
-


E1
E2
V1
V2
V3
V4
$\langle \text{VERIFY}, \upsilon, n, T, e \rangle_{\vec{\mu}_e}$
$\langle \text{PREPARE}, \upsilon, n, T, v \rangle_{\vec{\mu}_v}$
$\langle \text{COMMIT}, \upsilon, n, T, v \rangle_{\vec{\mu}_v}$
$\langle \text{VERIFY-RESPONSE}, \upsilon, n, T, v \rangle_{\vec{\mu}_v}$
$\langle \text{VERIFY-RESPONSE}, \upsilon+1, n, T, v, f \rangle_{\vec{\mu}_v}$
pre-prepare
prepare
commit
response

## Verification Stage - *Asynchronous BFT*

- Upon receiving Verify-Response message
  1. Commit
     - Condition: View number not increased, agreed-upon token matches previously sent one
     - Action: Mark sequence number stable, release requests to clients, etc.

## Verification Stage - *Asynchronous BFT*

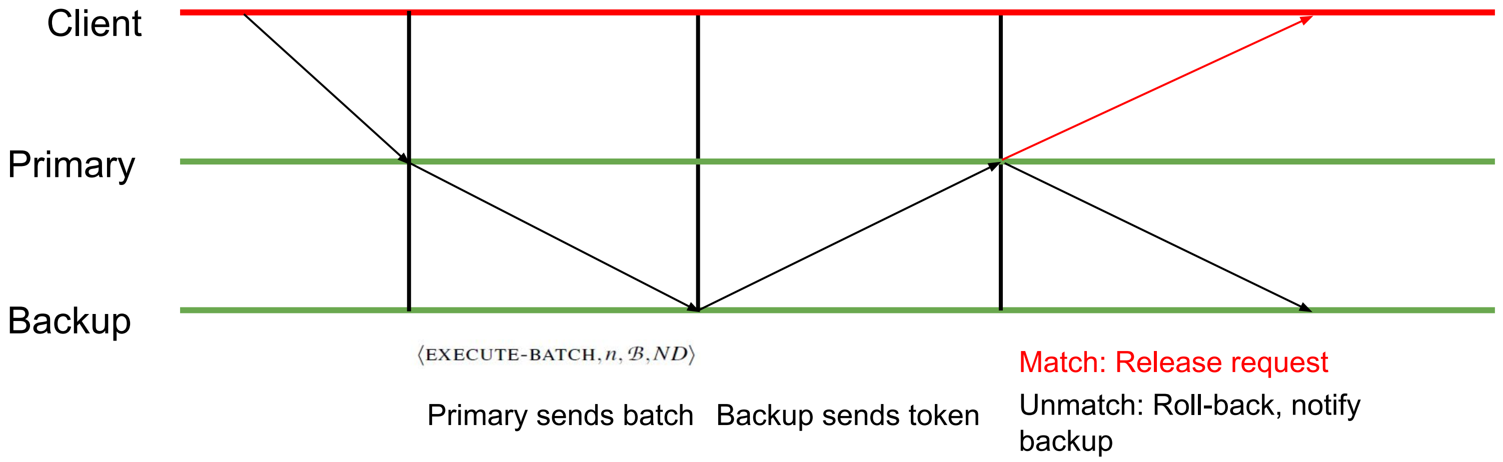
- Upon receiving Verify-Response message
  1. Commit
     - Condition: View number not increased, agreed-upon token matches previously sent one
     - Action: Make sequence number stable, release requests to clients, etc.
  2. State Transfer
     - Condition: View number not increased, tokens doesn’t match
     - Action: Issues a state-transfer request to other replicas

## Verification Stage - *Asynchronous BFT*

- Upon receiving Verify-Response message
  1. Commit
     - Condition: View number not increased, agreed-upon token matches previously sent one
     - Action: Make sequence number stable, release requests to clients, etc.
  2. State Transfer
     - Condition: View number not increased, tokens doesn’t match
     - Action: Issues a state-transfer request to other replicas
  3. Roll-back
     - Condition: View number increased
     - Action: Roll back state, execute batch sequentially, etc.

# Verification Stage - *Synchronous Primary-Backup*

- System Settings
-



## Verification Stage - *Tolerating Concurrency Bugs*

- Eve provides protection over concurrency bugs

- Fix concurrency faults: roll-back and sequential execution

## Verification Stage - *Tolerating Concurrency Bugs*

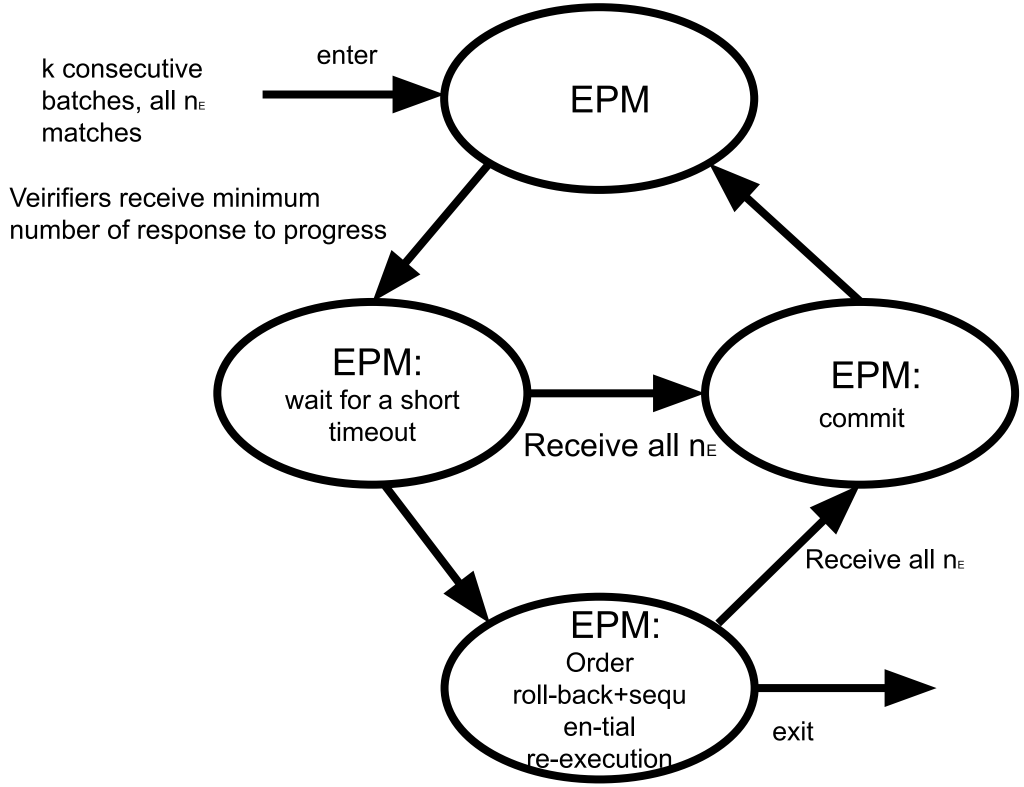
- Asynchronous Case
  - When configured with $n_{exec} = 2u+1$ and $r = 0$, asynchronous Eve is safe, live, and correct despite up to u concurrency or omission faults.

## Verification Stage - *Tolerating Concurrency Bugs*

- Asynchronous Case
    - When configured with $n_{exec} = 2u+1$ and $r = 0$, asynchronous Eve is safe, live, and correct despite up to u concurrency or omission faults.
- Synchronous Case
    - When configured with just u+1 execution replicas, Eve can continue to operate with 1 replica if u replicas fail by omission

## Verification Stage - *Tolerating Concurrency Bugs*

- Extra protection during good intervals



k consecutive batches, all $n_E$ matches
enter
EPM
Veirifiers receive minimum number of response to progress
EPM: wait for a short timeout
Receive all $n_E$
EPM: commit
EPM: Order roll-back+sequ en-tial re-execution
Receive all $n_E$
exit

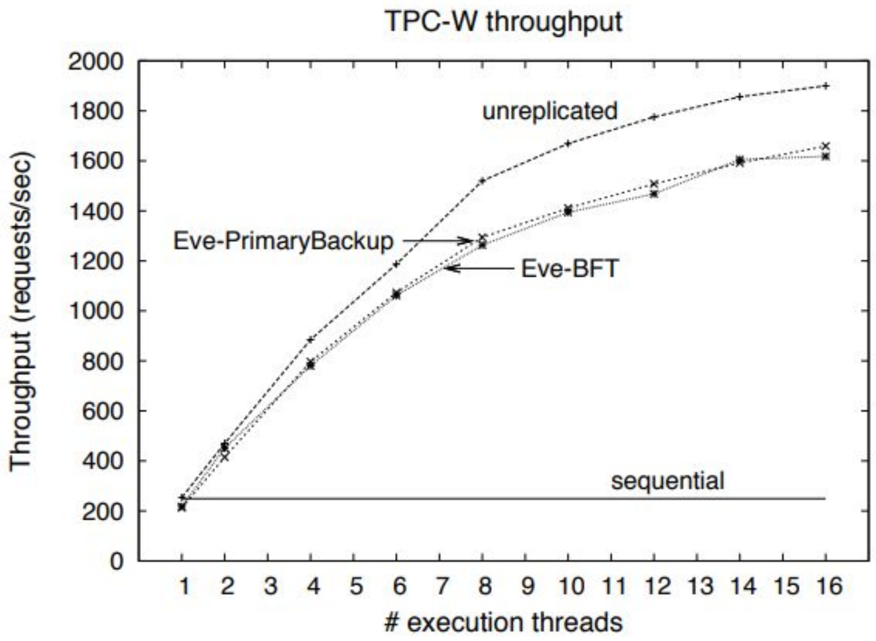
## Evaluation

1. Throughput gain
2. Influence of mixer
3. Currency bug mask

Key-value store application, H2 Database Engine

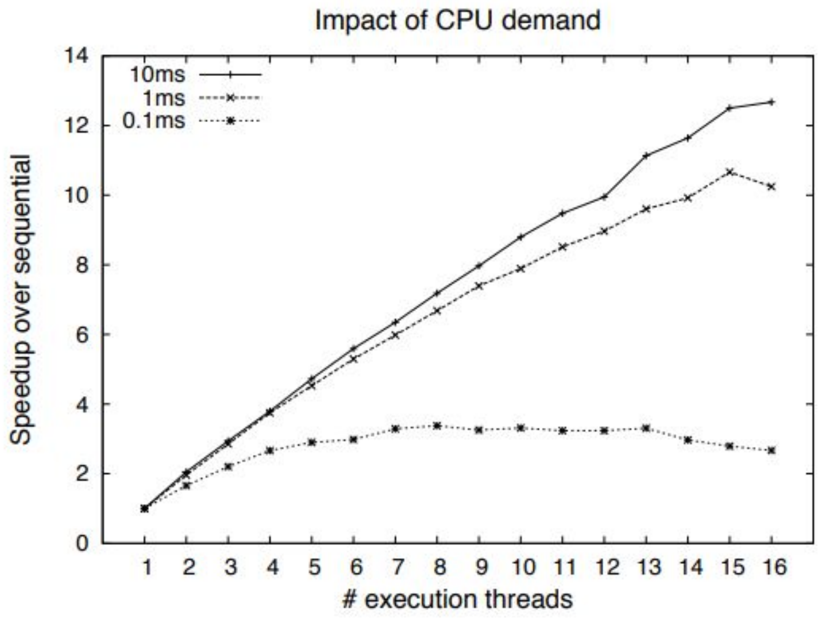
# Evaluation

- Throughput



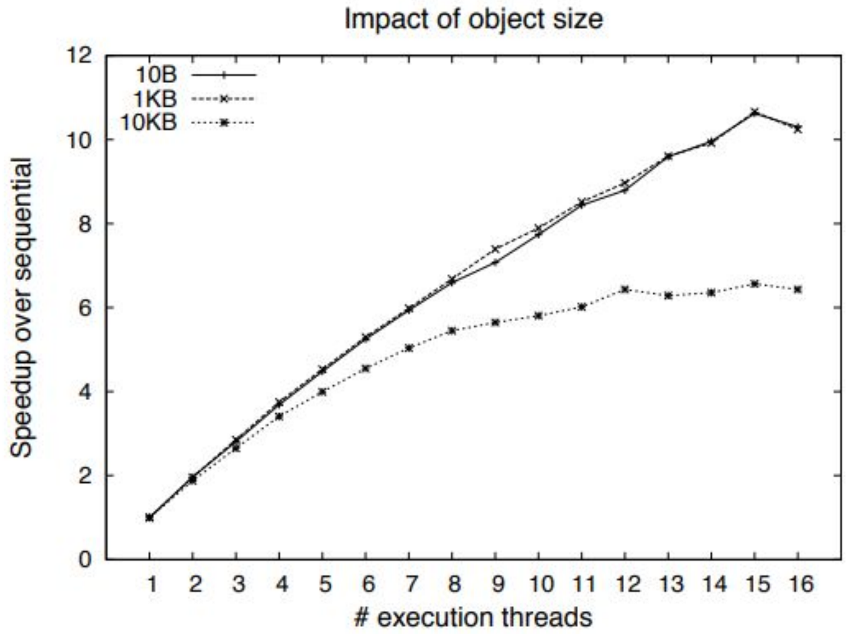
# Evaluation

- Varying CPU demand



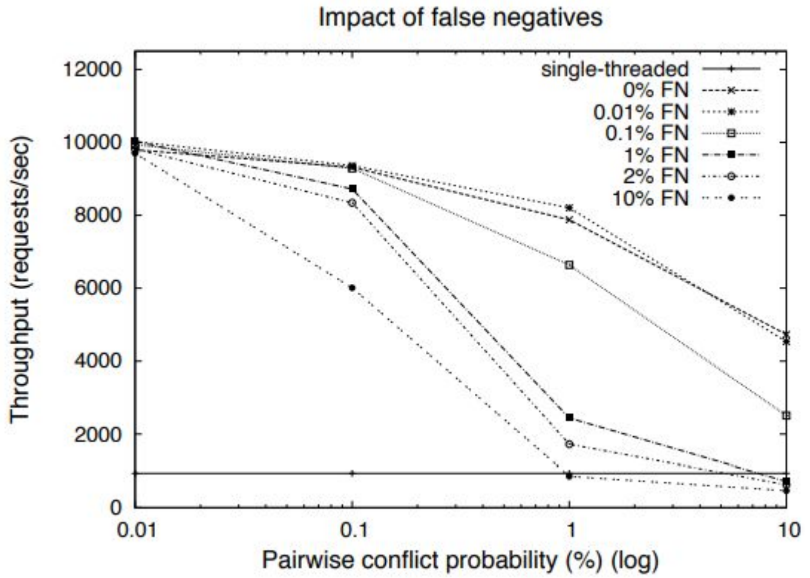
# Evaluation

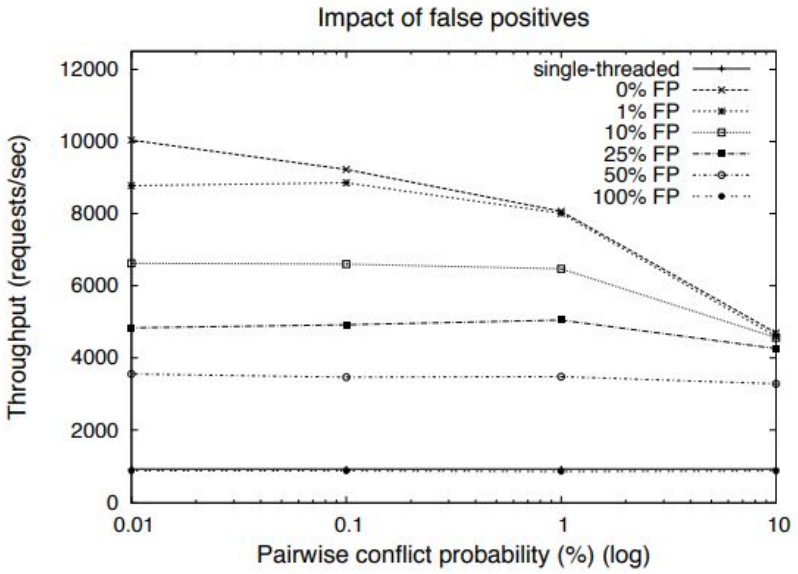
- Varying object size



# Evaluation

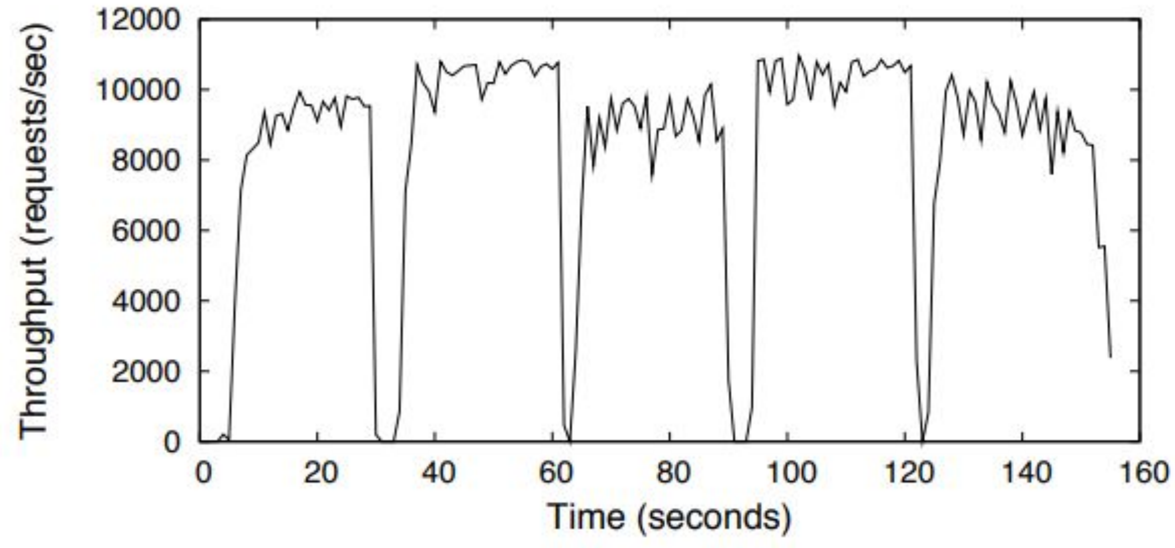
- Varying conflict probability





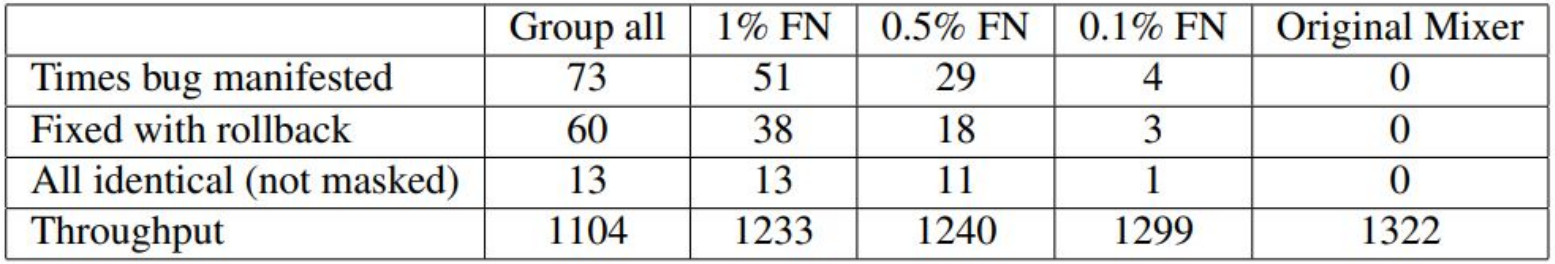
## Evaluation

- Failure and Recovery



## Evaluation

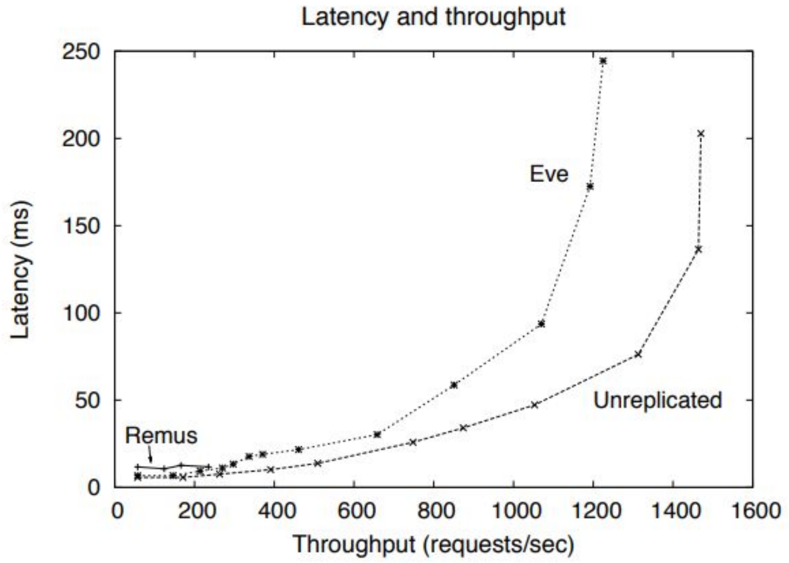
- Concurrency faults

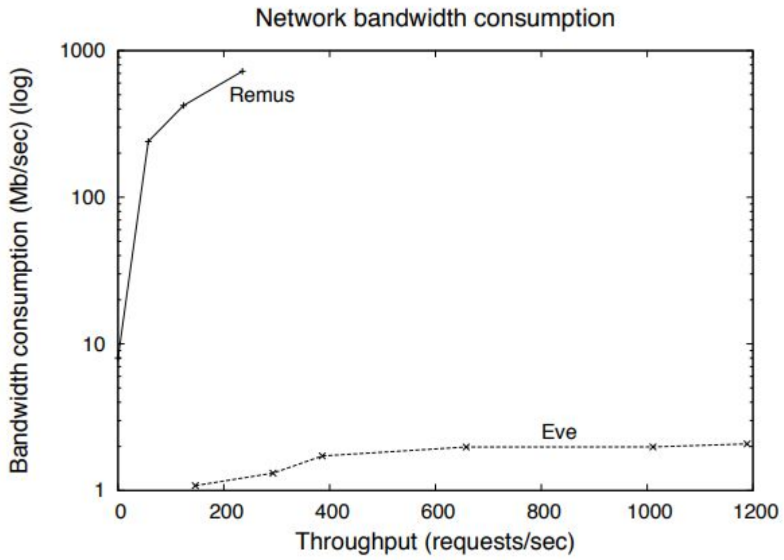


| | Group all | 1% FN | 0.5% FN | 0.1% FN | Original Mixer |
|---|---|---|---|---|---|
| Times bug manifested | 73 | 51 | 29 | 4 | 0 |
| Fixed with rollback | 60 | 38 | 18 | 3 | 0 |
| All identical (not masked) | 13 | 13 | 11 | 1 | 0 |
| Throughput | 1104 | 1233 | 1240 | 1299 | 1322 |

# Evaluation

- Comparison with Remus





## Conclusion

- Eve
    - New Execute-Verify architecture
    - Allow state machine replication to scale to multi-core servers
    - For the first time: allow interleaving requests nondeterministically and execute independently
    - Tolerate omission/commission faults in both asynchronous and synchronous
    - Protects against concurrency bugs

## Reference:

[1] Kapritsos, Manos, et al. "All about Eve: execute-verify replication for multi-core servers." Presented as part of the 10th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 12). 2012.

[2] Becker, Georg. "Merkle signature schemes, merkle trees and their cryptanalysis." Ruhr-University Bochum, Tech. Rep (2008).