VIAF: Verification-based Integrity Assurance Framework for MapReduce

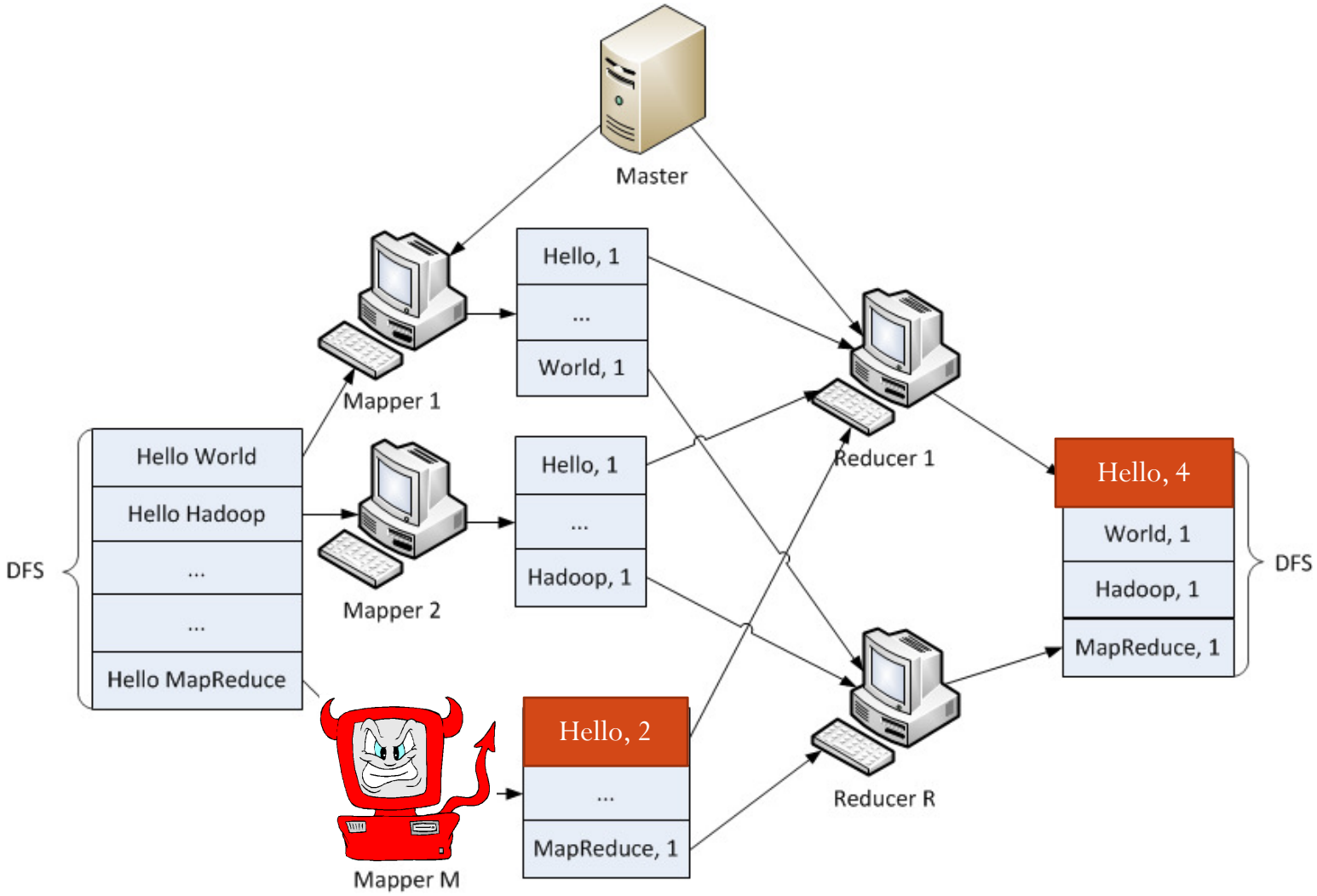
Yongzhi Wang, Jinpeng Wei

FIU FLORIDA INTERNATIONAL UNIVERSITY

MapReduce in Brief

- Satisfying the demand for large scale data processing
- It is a parallel programming model invented by Google in 2004 and become popular in recent years.
- Data is stored in DFS(Distributed File System)
- Computation entities consists of one Master and many Workers (Mapper and Reducer)
- Computation can be divided into two phases: Map and Reduce

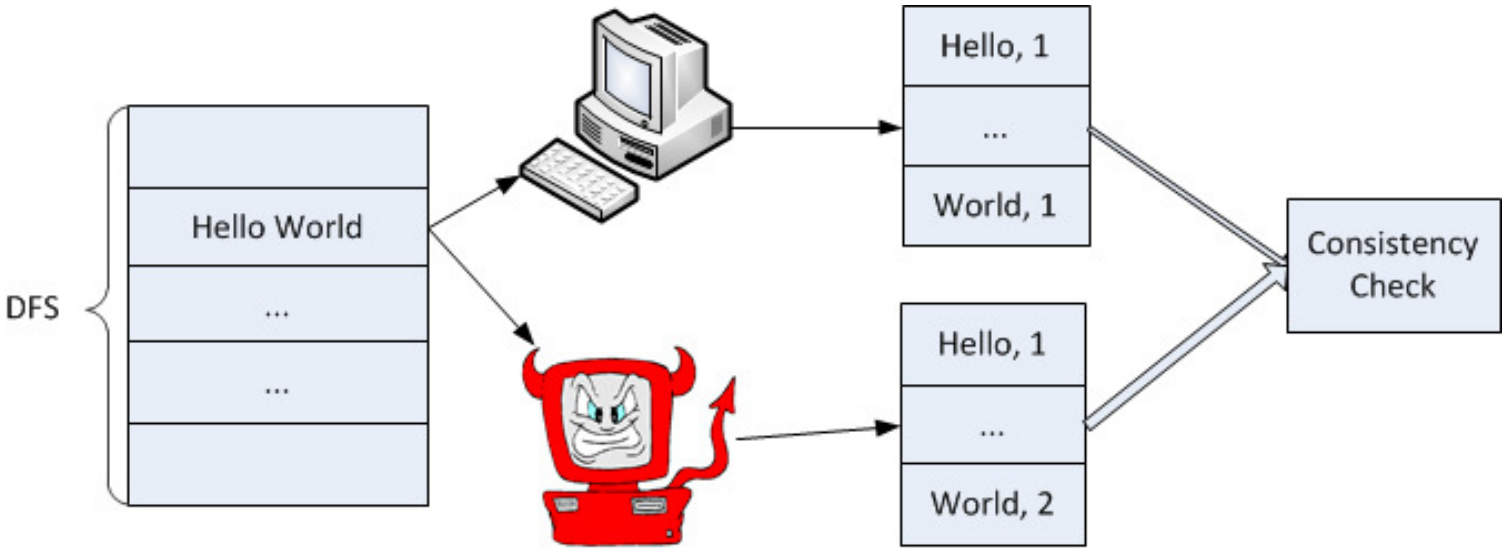
Word Count instance



Integrity Vulnerability

How to detect and eliminate malicious mappers to guarantee high computation integrity ?

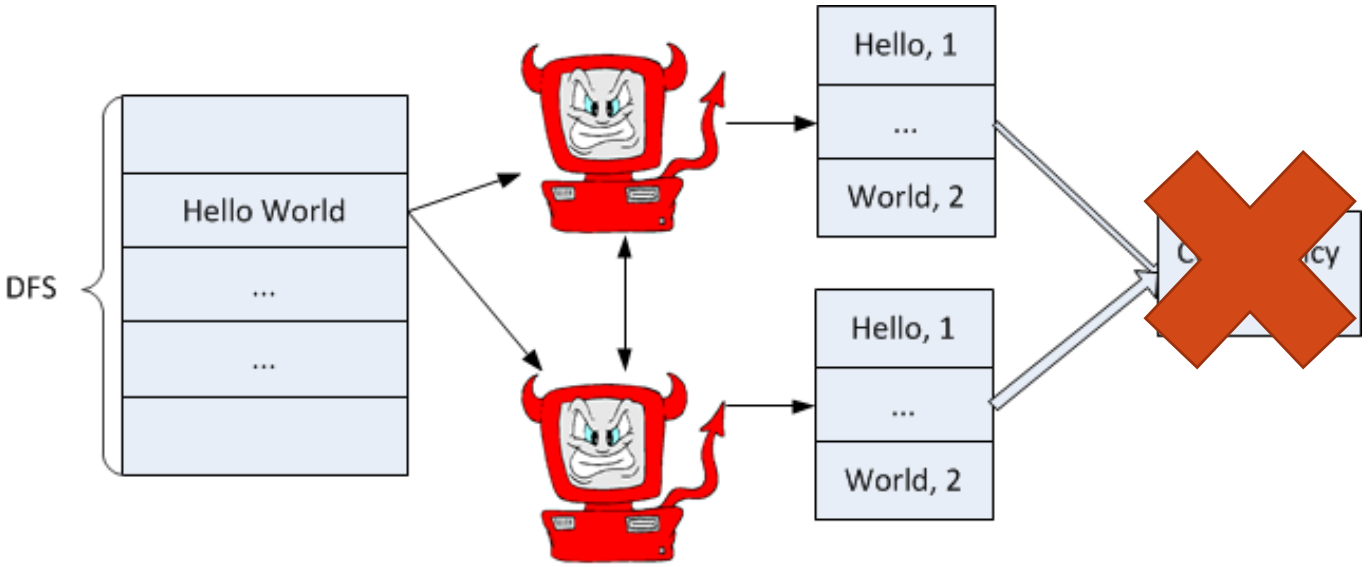
Straightforward approach: Duplication



Non-collusive worker

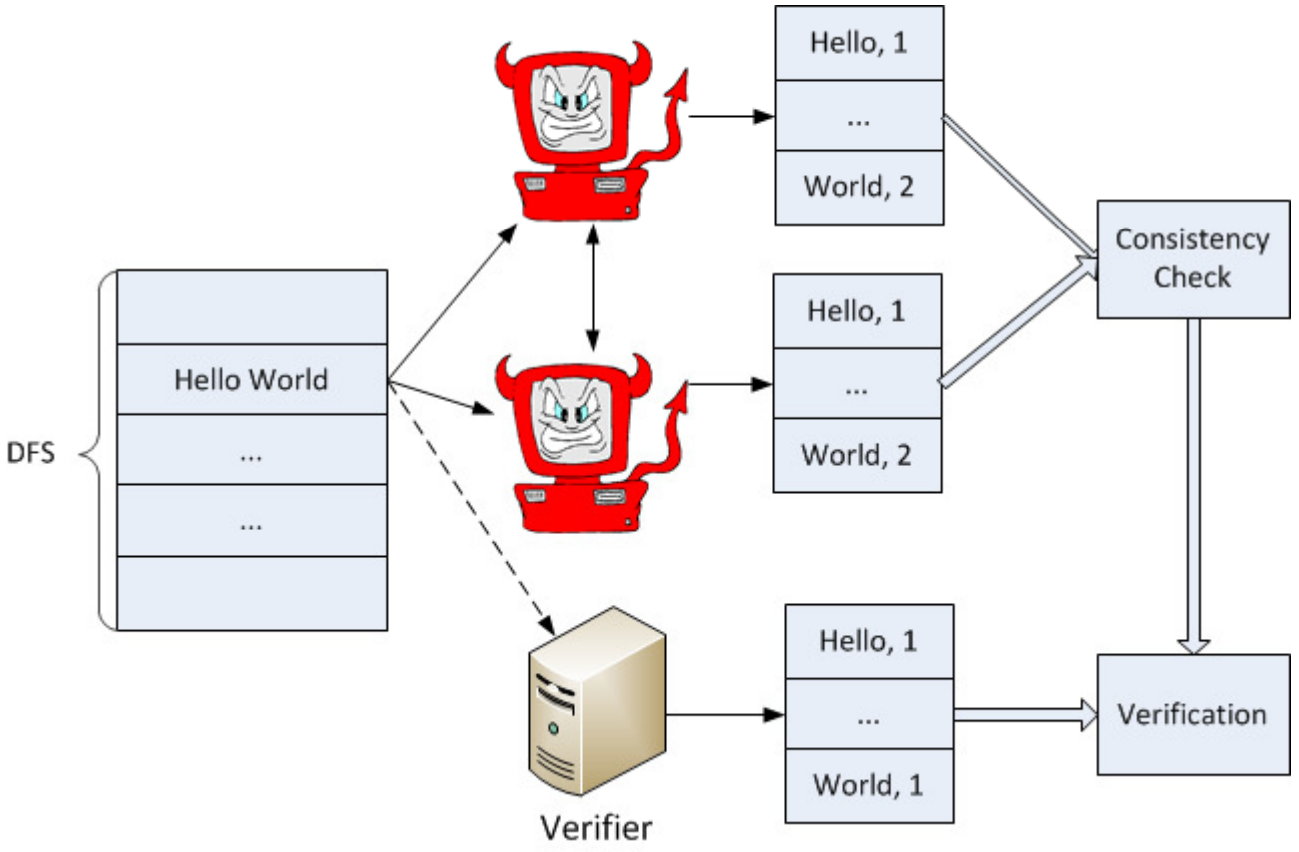
Integrity vulnerability

But how to deal with **collusive malicious** workers?



VIAF Solution

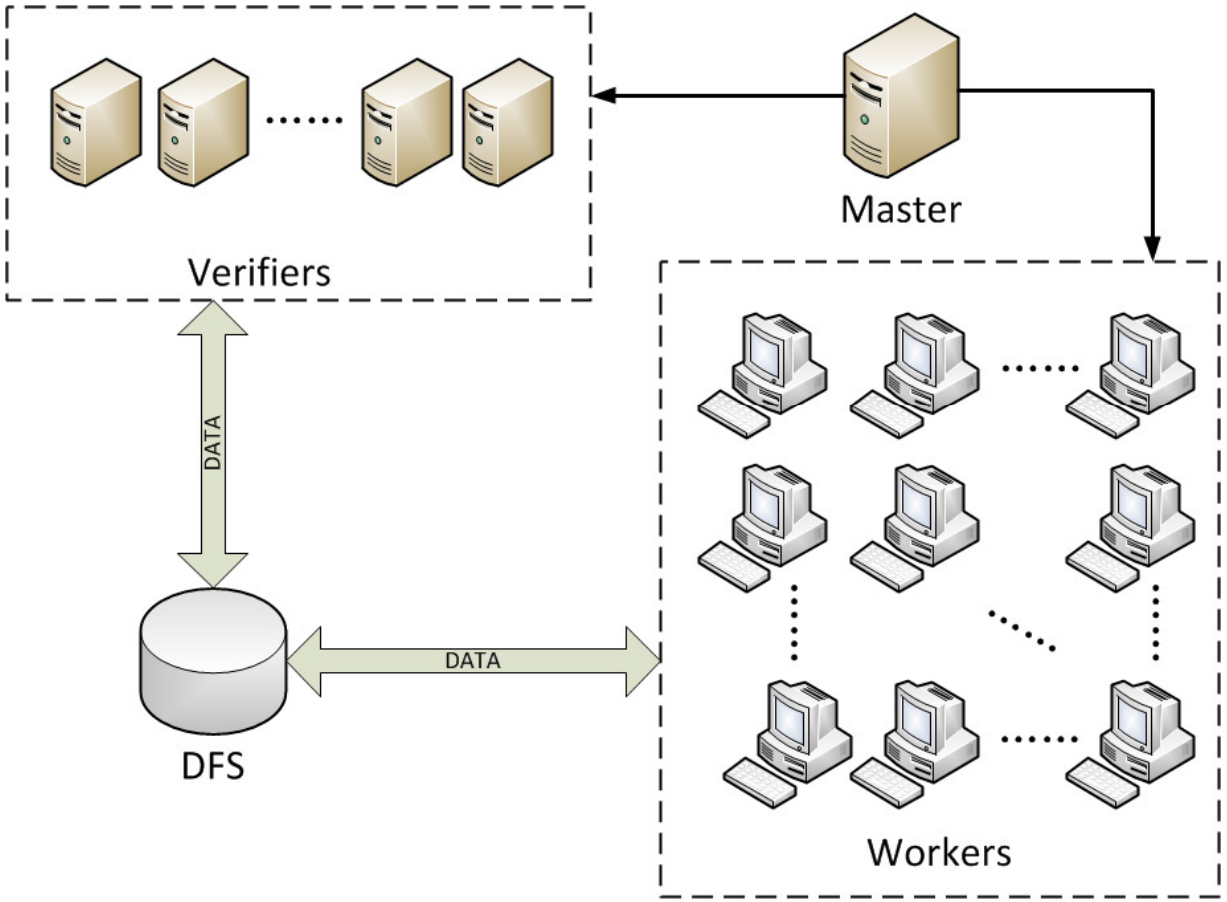
- **Duplication**
- **Introduce trusted entity to do random verification**



Outline

- Motivation
- System Design
- Analysis and Evaluation
- Related work
- Conclusion

Architecture



VIAF Architecture

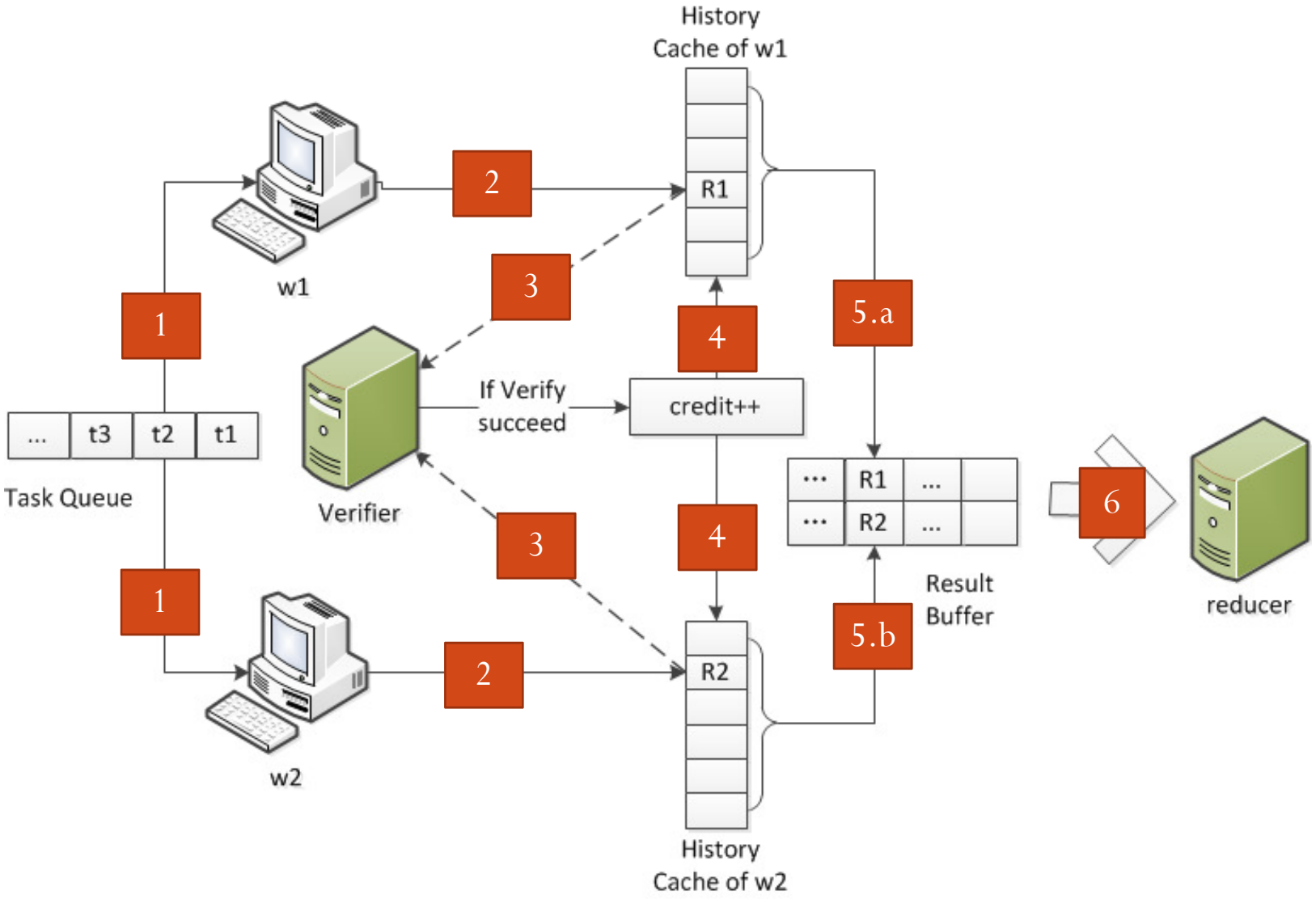
Assumption

- Trusted entities
  - Master
  - DFS
  - Reducers
  - Verifiers
- Untrusted entities
  - Mappers
- Information will not be tampered in network communication
- Mappers' result reported to the master should be consistent with its local storage (Commitment based protocol in SecureMR)

Solution

- Idea: Duplication + Verification
  - Deterministically duplicate each task to TWO mappers to discern the non-collusive mappers
  - Non-deterministically verify the consistent result to discern the collusive mappers
  - The credit of each mapper is accumulated by passing verification
  - A Mapper become trustable only when its credit achieves Quiz Threshold
- 100% accuracy in detecting non-collusive malicious mapper
- The more verification applied on a mapper, the higher accuracy to determine whether it is collusive malicious.

Data flow control



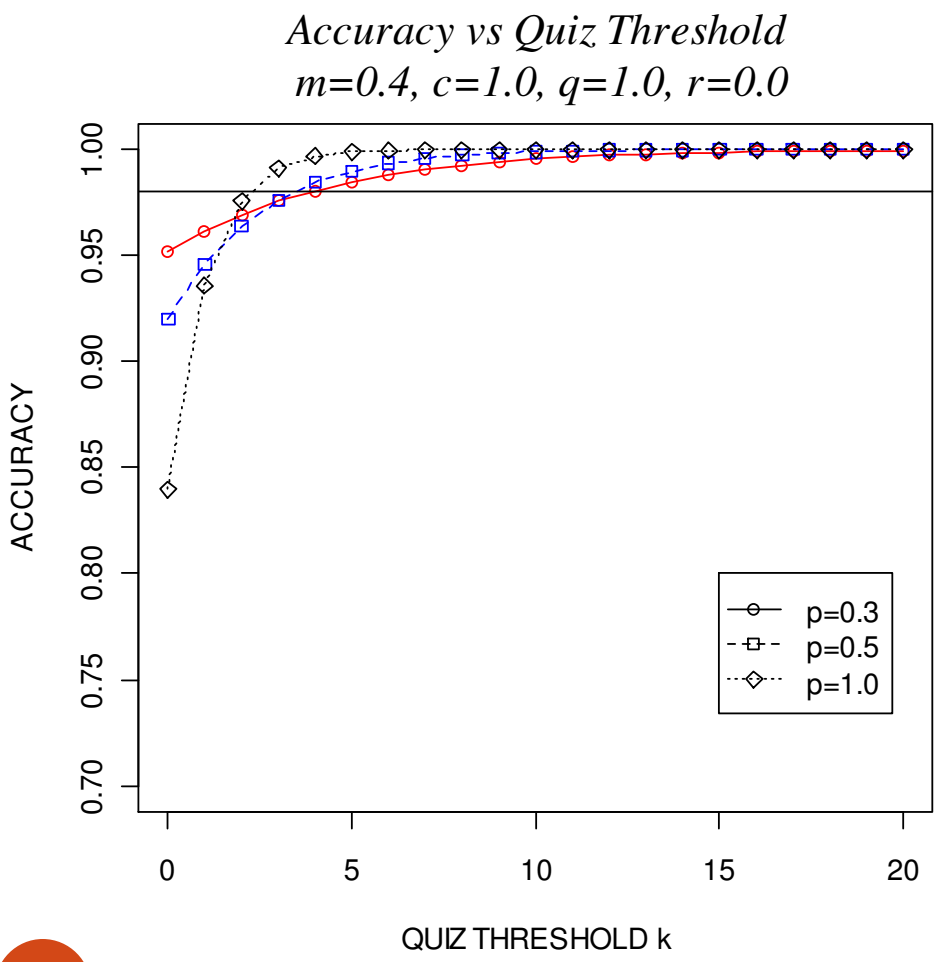
Outline

- Motivation
- System Design
- Analysis and Evaluation
- Related work
- Conclusion

Theoretical Analysis

- Measurement metric (for each task)
  - Accuracy -- The probability the reducer receive a good result from mapper
  - Mapping Overhead – The average number of execution launched by the mapper
  - Verification Overhead – The average number of execution launched by the verifier

Accuracy vs Quiz Threshold



Collusive mappers only, different p:

m – Malicious worker ratio

c – Collusive worker ratio

p – probability that two assigned collusive workers are in one collusion group and can commit a cheat

q – probability that two collusive workers commit a cheat

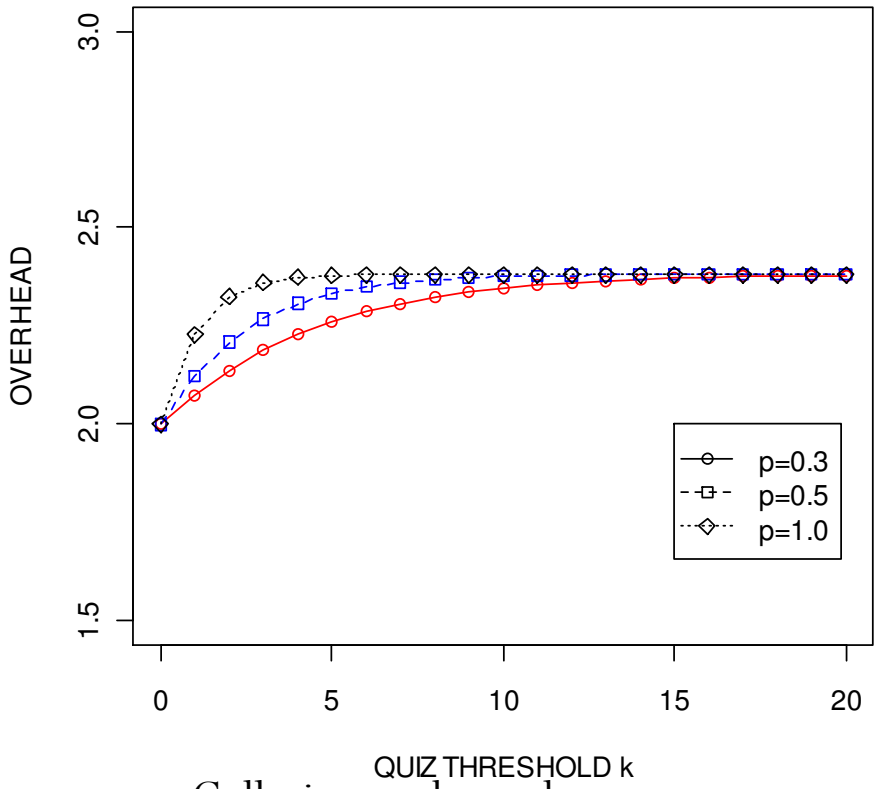
r – probability that a non collusive worker commit a cheat

v – Verification Probability

k – Quiz Threshold.

Overhead vs Quiz Threshold

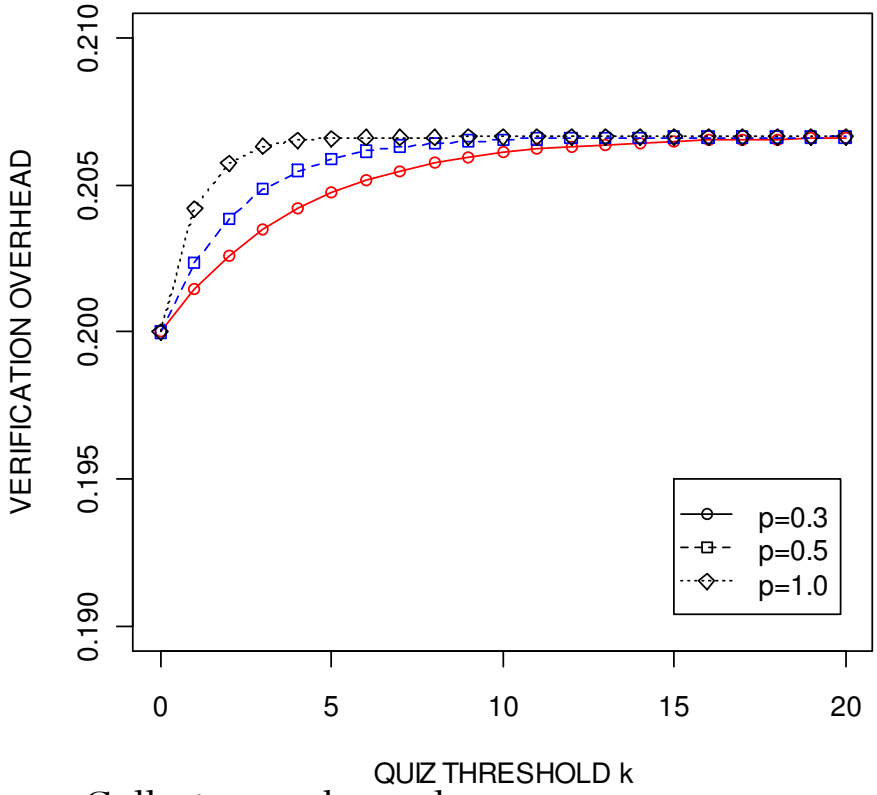
Mapper Overhead vs Quiz Threshold
m=0.4, c=1.0, q=1.0, r=0.0



Collusive worker only:

Overhead for each task stays between 2.0 to 2.4

Verification Overhead vs Quiz Threshold
m=0.4, c=1.0, q=1.0, r=0.0, v=0.2



Collusive worker only:

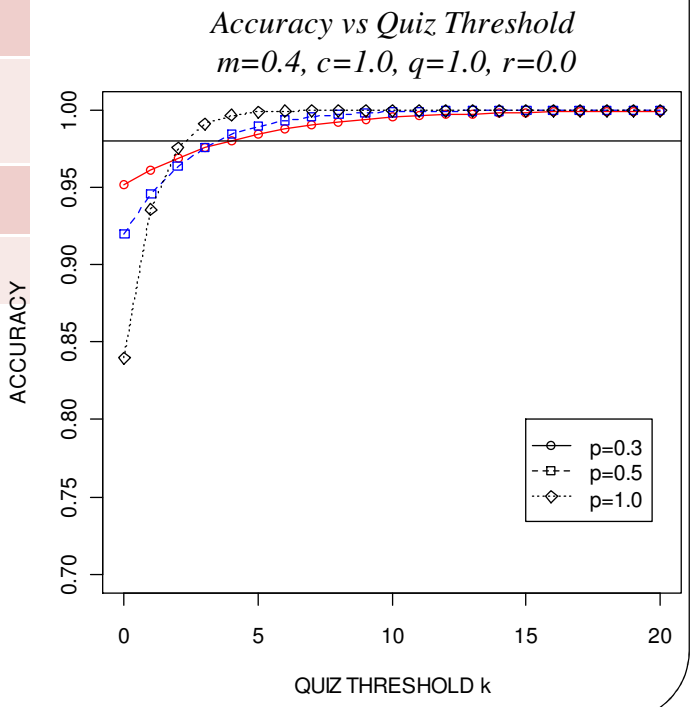
Verification Overhead for each task stays between 0.2 to 0.207 when v is 0.2

Experimental Evaluation

- Implementation based on Hadoop 0.21.0
- 11 virtual machines(512 MB of Ram, 40 GB disk each, Debian 5.0.6) deployed on a 2.93 GHz, 8-core Intel Xeon CPU with 16 GB of RAM
- word count application, 400 mapping tasks and 1 reduce task
- Out of 11 virtual hosts
  - 1 is both the Master and the benign worker
  - 1 is the verifier
  - 4 are collusive workers (malicious ratio is 40%)
  - 5 are benign workers

Evaluation result

Quiz Threshold	*Accuracy*	*Mapper Overhead*	*Verification Overhead*
0	87.20%	2.000	0
1	99.42%	2.045	22.00%
2	99.83%	2.074	23.58%
3	100%	2.053	23.00%
4	100%	2.162	23.58%
5	100%	2.046	21.75%
6	100%	2.111	22.58%
7	100%	2.027	19.83%



Where c is 1.0, p is 1.0, q is 1.0, **m is 0.4→0**

Outline

- Motivation
- System Design
- Analysis and Evaluation
- Related work
- Conclusion

Related work

- Replication, sampling, checkpoint-based solutions were proposed in several distributed computing contexts
  - Duplicate based for Cloud: SecureMR(Wei et al, ACSAC '09), Fault Tolerance Middleware for Cloud Computing(Wenbing et al, IEEE CLOUD '10)
  - Quiz based for P2P: Result verification and trust based scheduling in peer-to-peer grids (S. Zhao et al, P2P '05)
  - Sampling based for Grid: Uncheatable Grid Computing (Wenliang et al, ICDCS'04)
- Accountability in Cloud computing research
  - Hardware-based attestation: Seeding Clouds with Trust Anchors (Joshua et al, CCSW '10)
  - Logging and Auditing: Accountability as a Service for the Cloud(Jinhui, et al, ICSC '10)

Outline

- Motivation
- System Design
- Analysis and Evaluation
- Related work
- Conclusion

Conclusion

- Contributions
  - Proposed a Framework (VIAF) to defeat both collusive and non-collusive malicious mappers in MapReduce calculation.
  - Implemented the system and proved from theoretical analysis and experimental evaluation that VIAF can guarantee high accuracy while incurring acceptable overhead.
- Future work
  - Further exploration without the assumption of trust of reducer.
  - Reuse the cached result to better utilize the verifier resource.

THANKS!

Q&A