IntegrityMR: Integrity Assurance Framework for Big Data Analytics and Management Applications

Yongzhi Wang, Jinpeng Wei Florida International University
Mudhakar Srivatsa IBM T.J. Watson Research Center
Yucong Duan, Wencai Du Hainan University

Agenda

- Problem Statement
- MapReduce Task Layer Solution
- Application Layer Solution
- Conclusion & Future Work

Agenda

- Problem Statement
- MapReduce Task Layer Solution
- Application Layer Solution
- Conclusion & Future Work

Big Data Analytics & Cloud

Privacy?

Integrity?

Security?

Security Problem

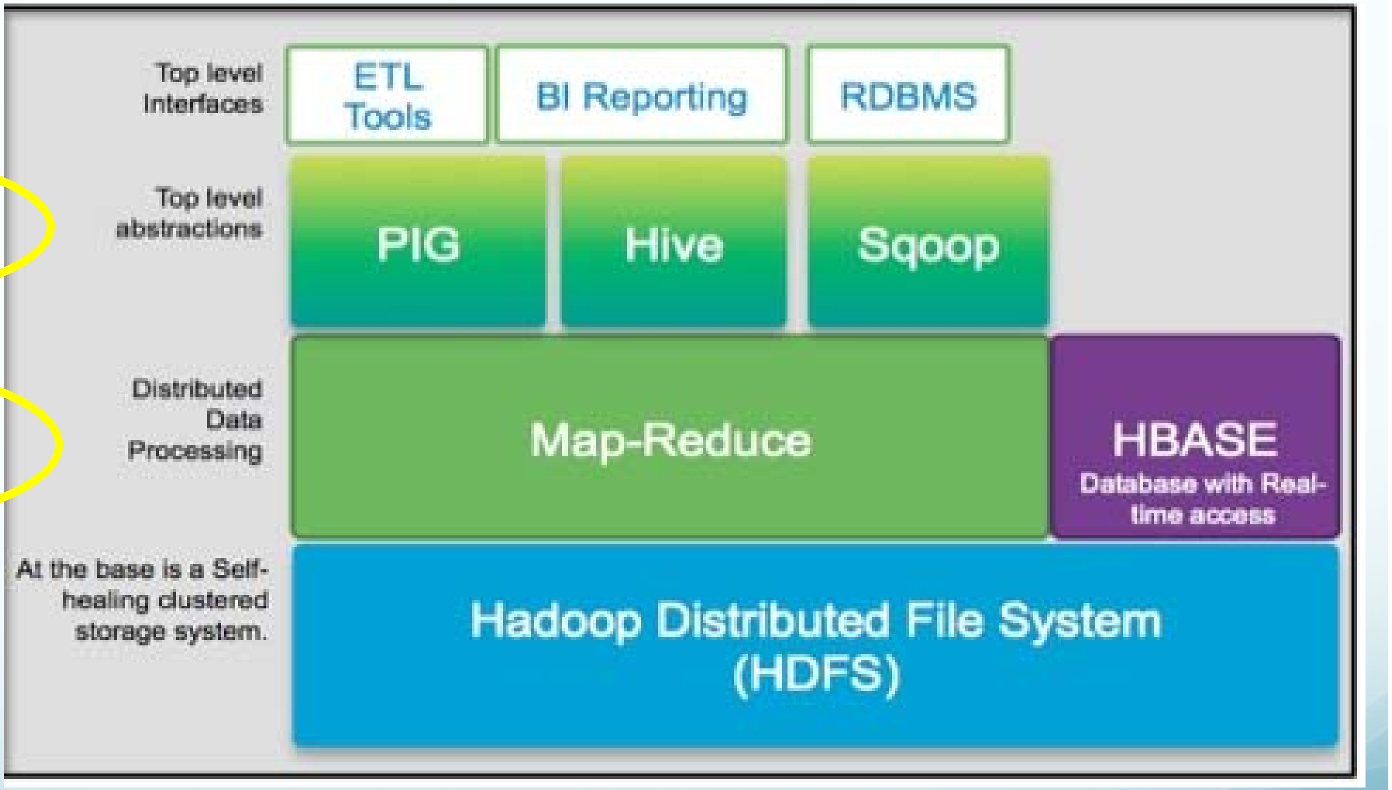
How do we construct big data analytics infrastructure on cloud that can provide high integrity assurance?

Big Data Infrastructure

Application Layer Integrity

MapReduce Task Layer Integrity

Storage Integrity: [5] [6]


Top level Interfaces
ETL Tools
BI Reporting
RDBMS
Top level abstractions
PIG
Hive
Sqoop
Distributed Data Processing
Map-Reduce
HBASE
Database with Real-time access
At the base is a Self-healing clustered storage system.
Hadoop Distributed File System (HDFS)

Agenda

- Problem Statement
- MapReduce Task Layer Solution
- Application Layer Solution
- Conclusion & Future Work

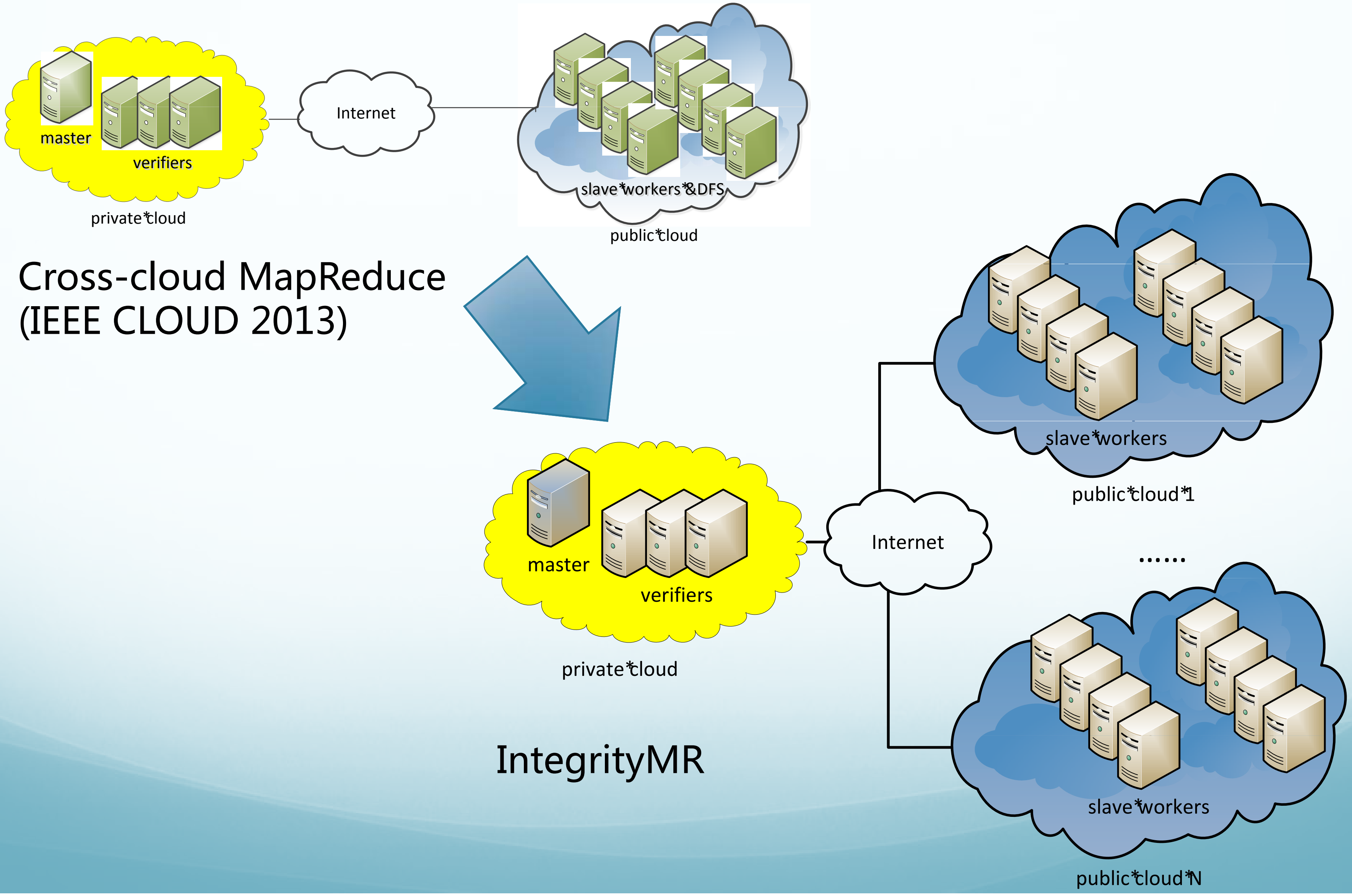
Related Works

- Wei Wei, Juan Du, Ting Yu, Xiaohui Gu, “SecureMR: A Service Integrity Assurance Framework for MapReduce” , in Proceedings of the 2009 Annual Computer Applications Conference.(ACSAC2009)

- Yongzhi Wang, Jinpeng Wei, “VIAF: Verification-based Integrity Assurance Framework for MapReduce” , in the 4thIEEE International Conference on Cloud Computing (CLOUD 2011).

- Yongzhi Wang, Jinpeng Wei, Mudhakar Srivatsa, “Result Integrity Check for MapReduce Computation on Hybrid Clouds” in the 6th IEEE International Conference on Cloud Computing (CLOUD 2013).

Architecture



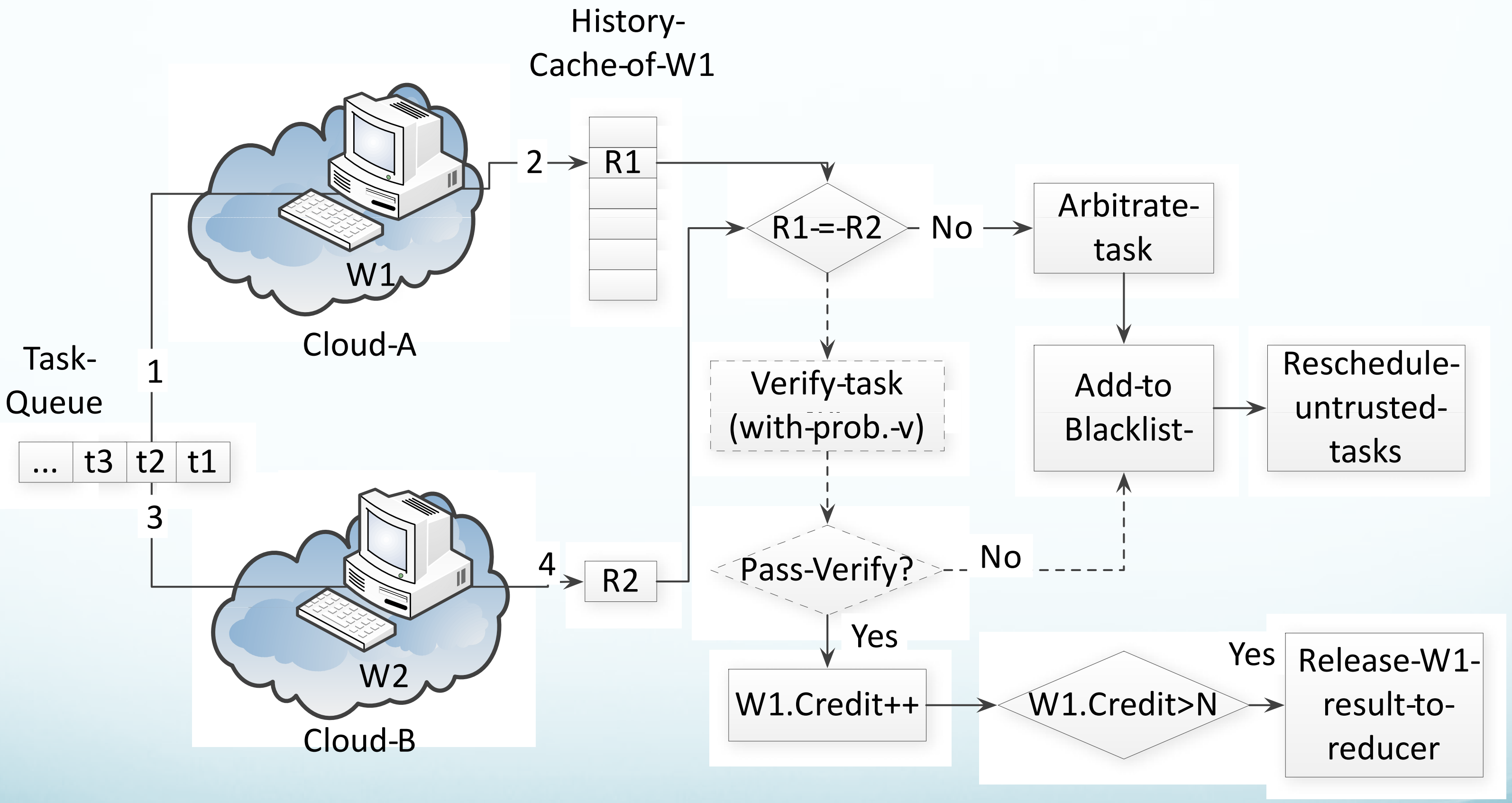
Cross-cloud MapReduce
(IEEE CLOUD 2013)

IntegrityMR

Architecture Design

- Trusted private cloud + Untrusted public clouds
- Trusted private cloud
  - Master controls the computation.
  - Verifier offers the trusted result verification.
- Untrusted public clouds
  - Offers the computation capacity.
  - Multiple clouds raise the bar for the attacker

Control Flow



Replication

Verification

Credit-based Management

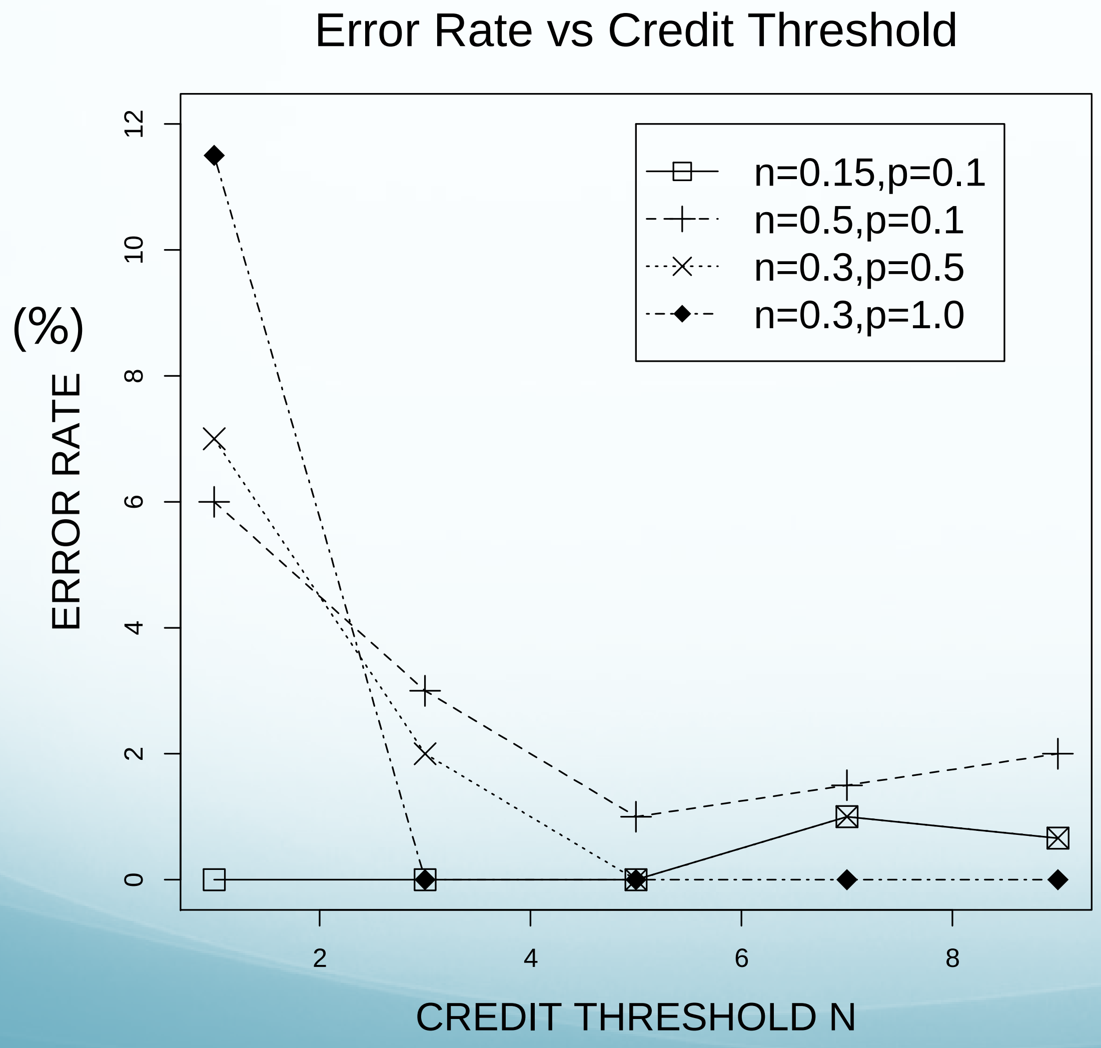
Experiment setup

- Environment
  - Private cloud:
    - a local Linux server (2.93GHz, 8-core Intel Xeon CPU, 16GB Ram)
  - Public clouds:
    - 6 Microsoft Azure extra small instances (1core @1GHz, 768MB Ram)
    - 6 Amazon EC2 small instances (1ECU, 1core, 1.7GB).
- Application
  - Word count (100 map task) for accuracy test
  - Mahout 20 Newsgroup Classification for performance test

Metrics of Accuracy and Overhead

- **Error rate**: The percentage of incorrect map task results accepted by the master in one job execution.
- **Worker overhead**: The percentage of extra number of map tasks executed on the workers on public cloud in one job execution.
- **Verifier overhead**: The percentage of map tasks executed by the verifiers on the private cloud in one job execution.

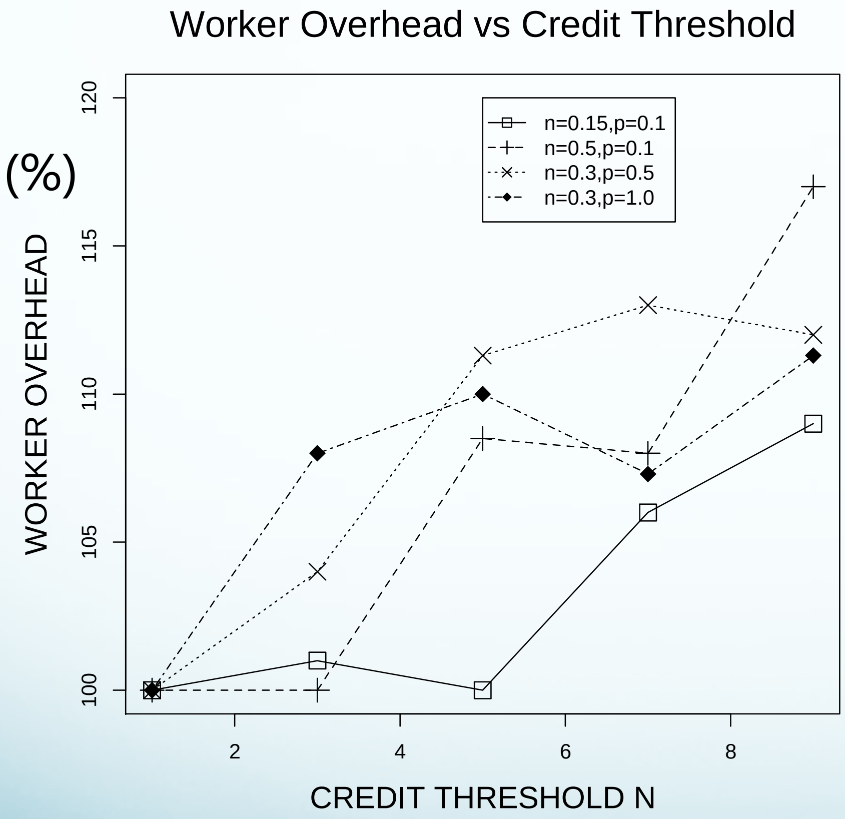
Accuracy

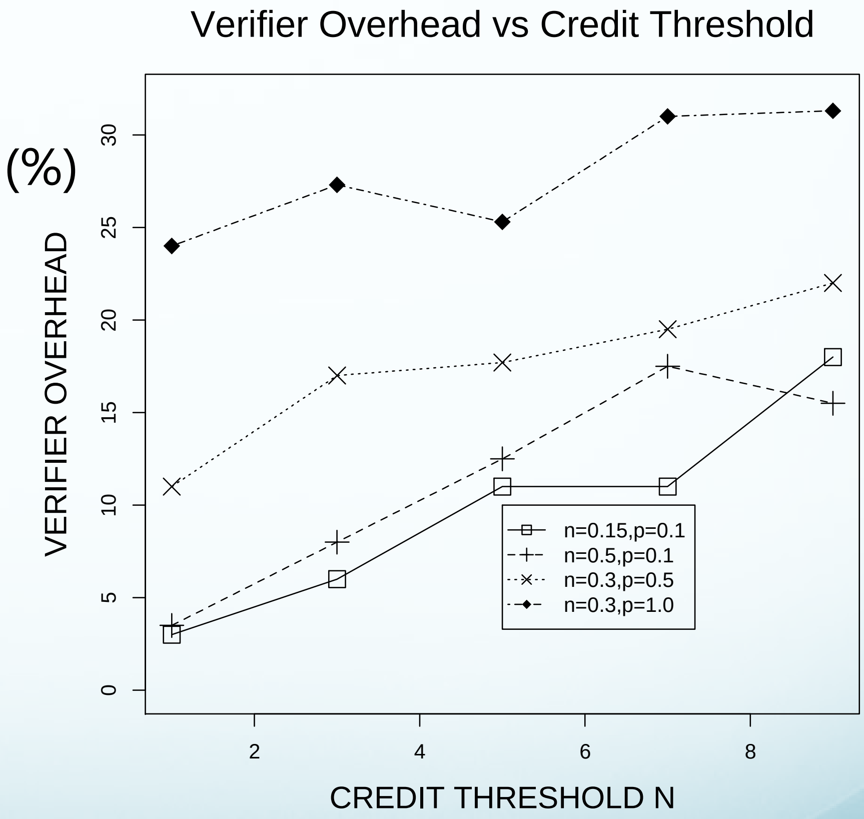


Error rate: The percentage of incorrect map task results accepted by the master in one job execution.

n: malicious node ratio
p: cheat probability
N: credit threshold

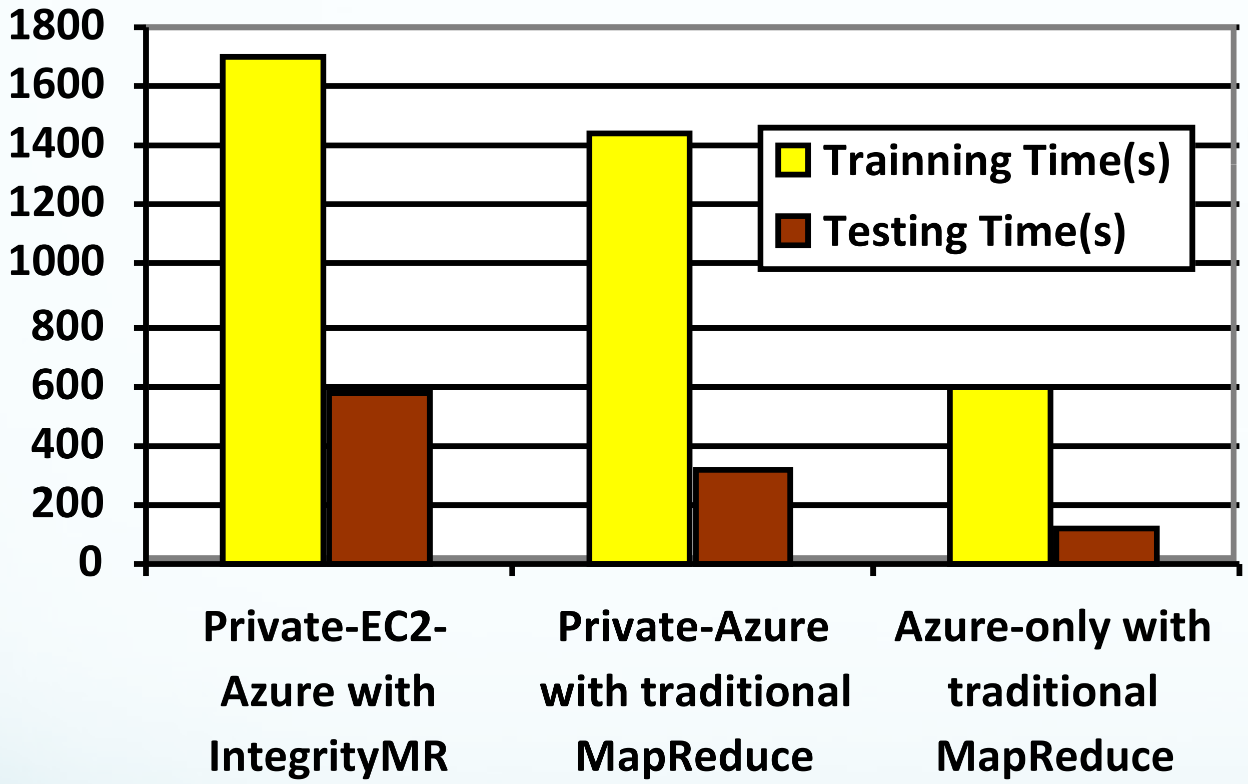
Overhead and Verifier Overhead





n: malicious node ratio
p: cheat probability
N: credit threshold

Execution time



Mahout
20 news group
Classification

N = 5
v = 0.15
Exec. Delay:
P-A compared to A-O: 145% and 177%
P-E-A compared to P-A: 18% and 82%

Name	Environment Composition	Cloud	Map Reduce
Private-EC2-Azure	Linux server on Private Cloud, 6 small instances on EC2, 6 extra small instances on Azure	Cross Cloud	IntegrityMR
Private-Azure	Linux server on Private Cloud, 6 extra small instances on Azure.	Cross Cloud	Map Reduce
Azure-only	6 extra small instances on Azure	Inside Cloud	Map Reduce

Agenda

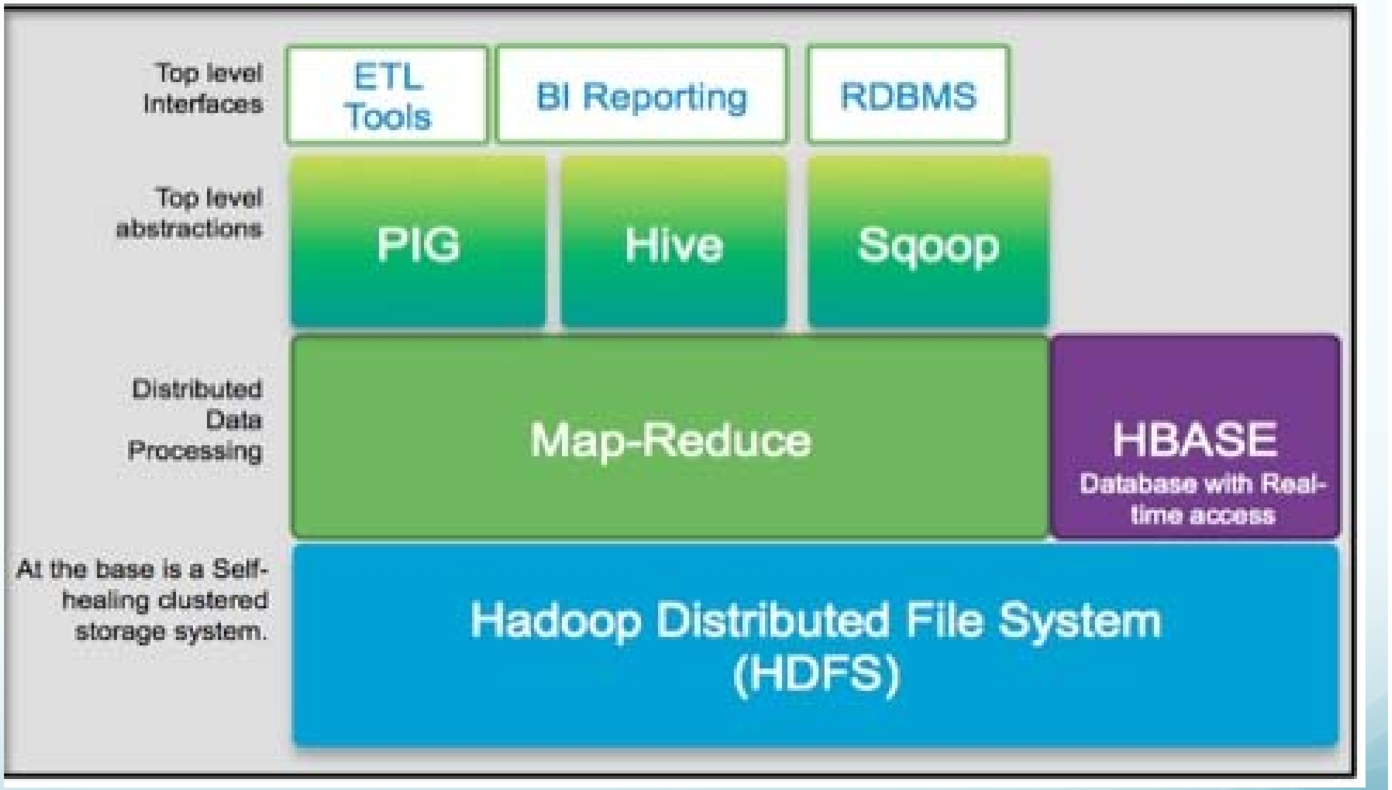
- Problem Statement
- MapReduce Task Layer Solution
- Application Layer Solution
- Conclusion & Future Work

Big Data Infrastructure

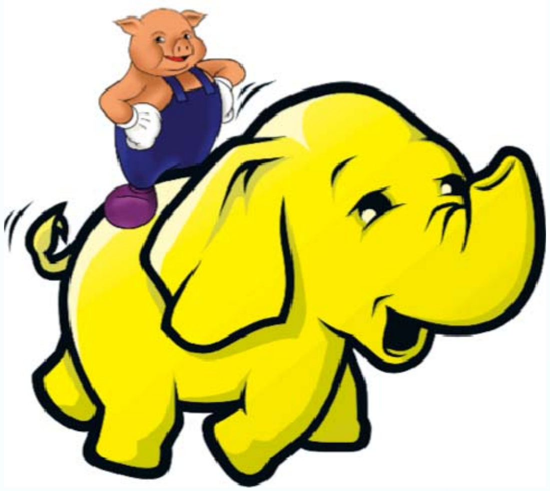
Application Layer Integrity

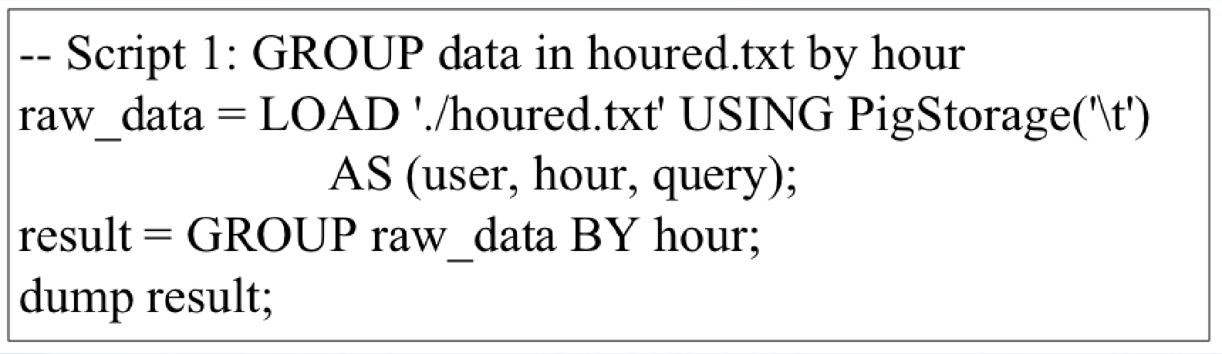
MapReduce Task Layer Integrity

Storage Integrity: [5] [6]

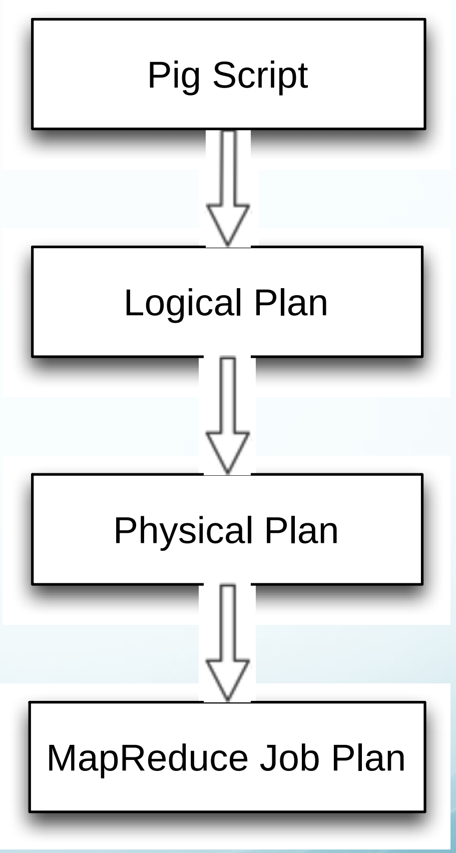

Top level Interfaces
ETL Tools
BI Reporting
RDBMS
Top level abstractions
PIG
Hive
Sqoop
Distributed Data Processing
Map-Reduce
HBASE
Database with Real-time access
At the base is a Self-healing clustered storage system.
Hadoop Distributed File System (HDFS)

Apache Pig





```
-- Script 1: GROUP data in houred.txt by hour
raw_data = LOAD './houred.txt' USING PigStorage('\t')
                    AS (user, hour, query);
result = GROUP raw_data BY hour;
dump result;
```



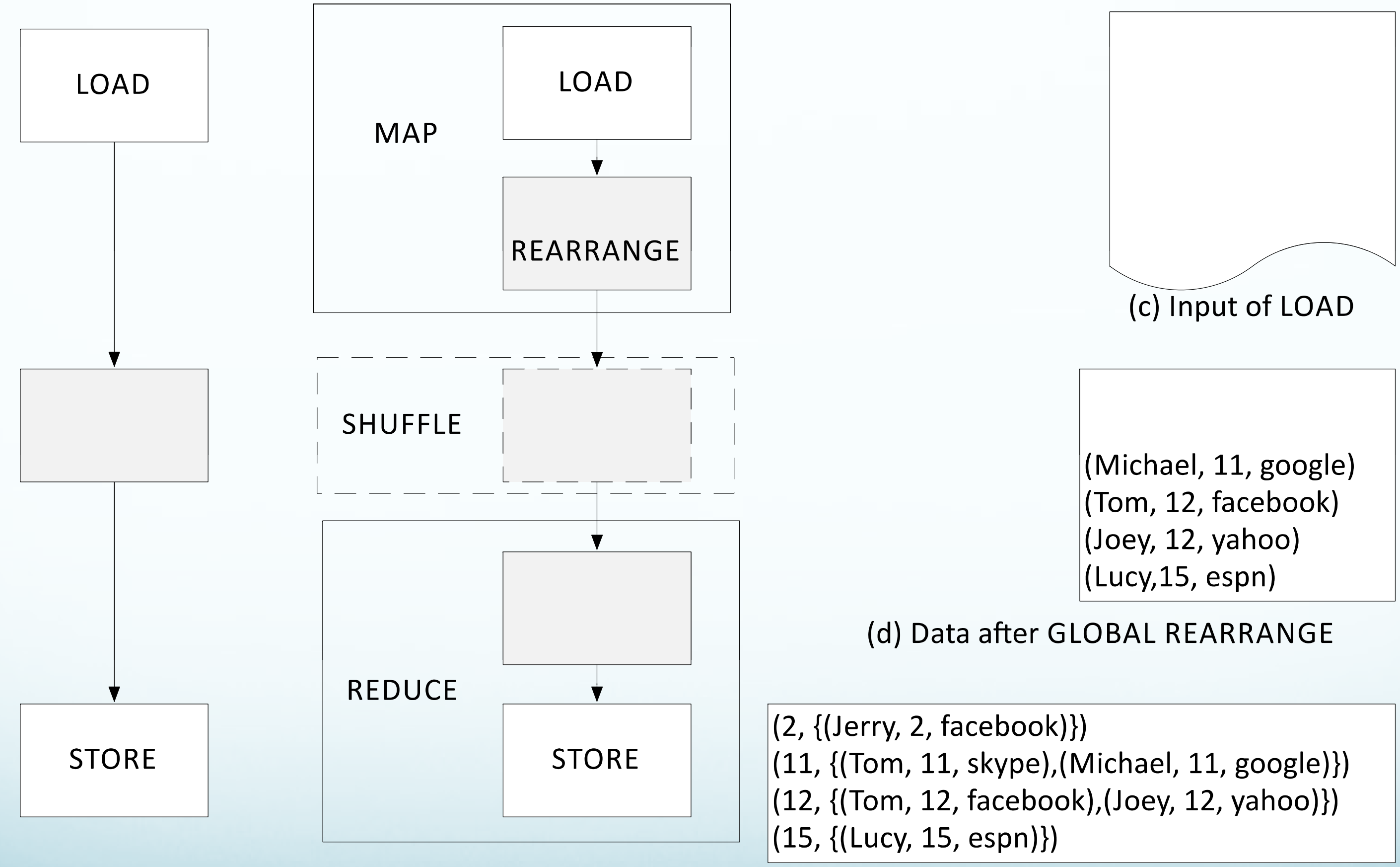
Pig Script

Logical Plan

Physical Plan

MapReduce Job Plan

How Pig Works



(c) Input of LOAD

(Michael, 11, google)
(Tom, 12, facebook)
(Joey, 12, yahoo)
(Lucy,15, espn)

(d) Data after GLOBAL REARRANGE

(2, {(Jerry, 2, facebook)})
(11, {(Tom, 11, skype),(Michael, 11, google)})
(12, {(Tom, 12, facebook),(Joey, 12, yahoo)})
(15, {(Lucy, 15, espn)})

(a) Logical Plan

(b) Map-Reduce Plan

(e) Data after PACKAGE

Intuition

- Transform the script so that to change the plan
  - Split the map task into two/more different tasks.
  - The output of different map tasks, although different, should obey the constructed invariant.
  - The reduce task is transformed to check the invariant.

Transformation Example

```
-- Script 1: GROUP data in houred.txt by hour
raw_data = LOAD './houred.txt' USING PigStorage('\t')
                    AS (user, hour, query);
result = GROUP raw_data BY hour;
dump result;
```

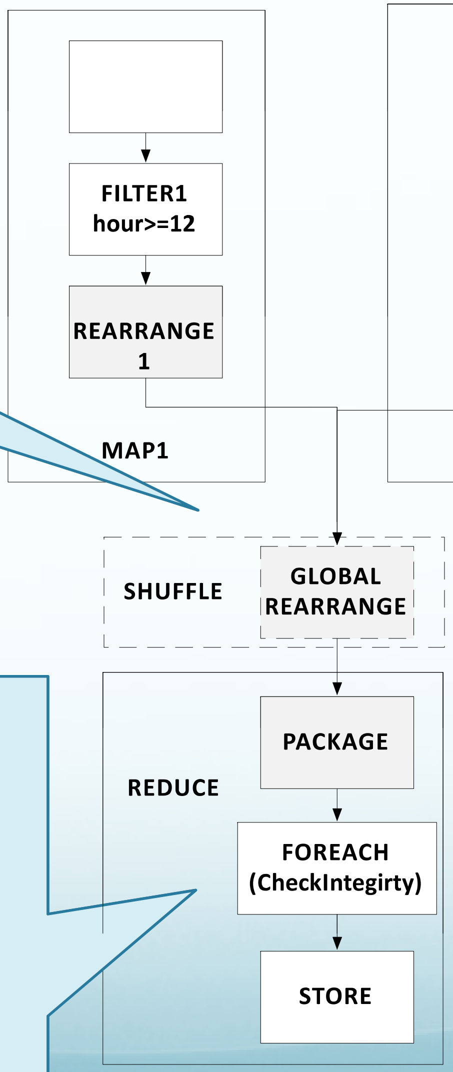
```
-- Script 2: invariant check is enforced
register ./tutorial.jar;
raw_data = LOAD './houred.txt' USING PigStorage('\t')
              AS (user, hour, query);
part1 = FILTER raw_data BY hour>=12;
part2 = FILTER raw_data BY hour<=12;
result = COGRUP part1 BY hour, part2 BY hour;
group_result=FOREACH result GENERATE
              group, org.apache.pig.tutorial.CheckInvariant($1,$2);
```

Split the map task into two/more different tasks, The output of different map tasks, although different, should obey the constructed invariant.

The reduce task is transformed to check the invariant

Plan Transformation

Split the map task.
The output of different map tasks obey the constructed invariant.



(a) Map-Reduce Plan

```
checkIntegrity (key, tuple1,
tuple2){
   If(key != 12)
       return true;
   else if(tuple 1 == tuple 2)
                return true;
          else
                return false;
}
```

(,   , g   g   )
(Joey, 12, yahoo)
(Lucy,15, espn)

(b) Input of LOAD1, LOAD2

(Tom, 12, facebook)
(Joey, 12, yahoo)
(Lucy,15, espn)

(c) Data after FILTER1

(Tom, 12, facebook)
(Tom, 11, skype)
(Jerry, 2, facebook)
(Michael, 11, google)
(Joey, 12, yahoo)

(d) Data after FILTER2

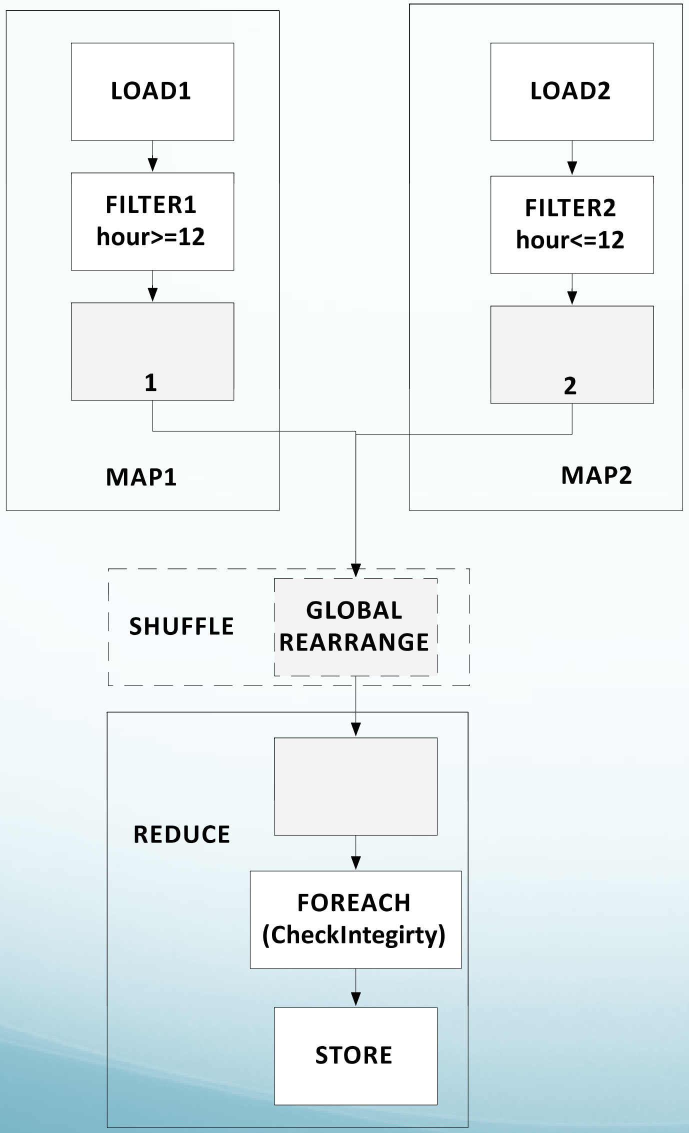
(12 , {(Tom, 12, facebook),(Joey, 12, yahoo)} ,
{(Tom, 12, facebook),(Joey, 12, yahoo)})
(15 , {(Lucy, 15, espn)} , {*null*})

(e) Data after PACKAGE

(2 , {(Jerry, 2, facebook)})
(11 , {(Tom, 11, skype),(Michael, 11, google)})
(12 , {(Tom, 12, facebook),(Joey, 12, yahoo)})
(15 , {(Lucy, 15, espn)})

(f) Data after FOREACH

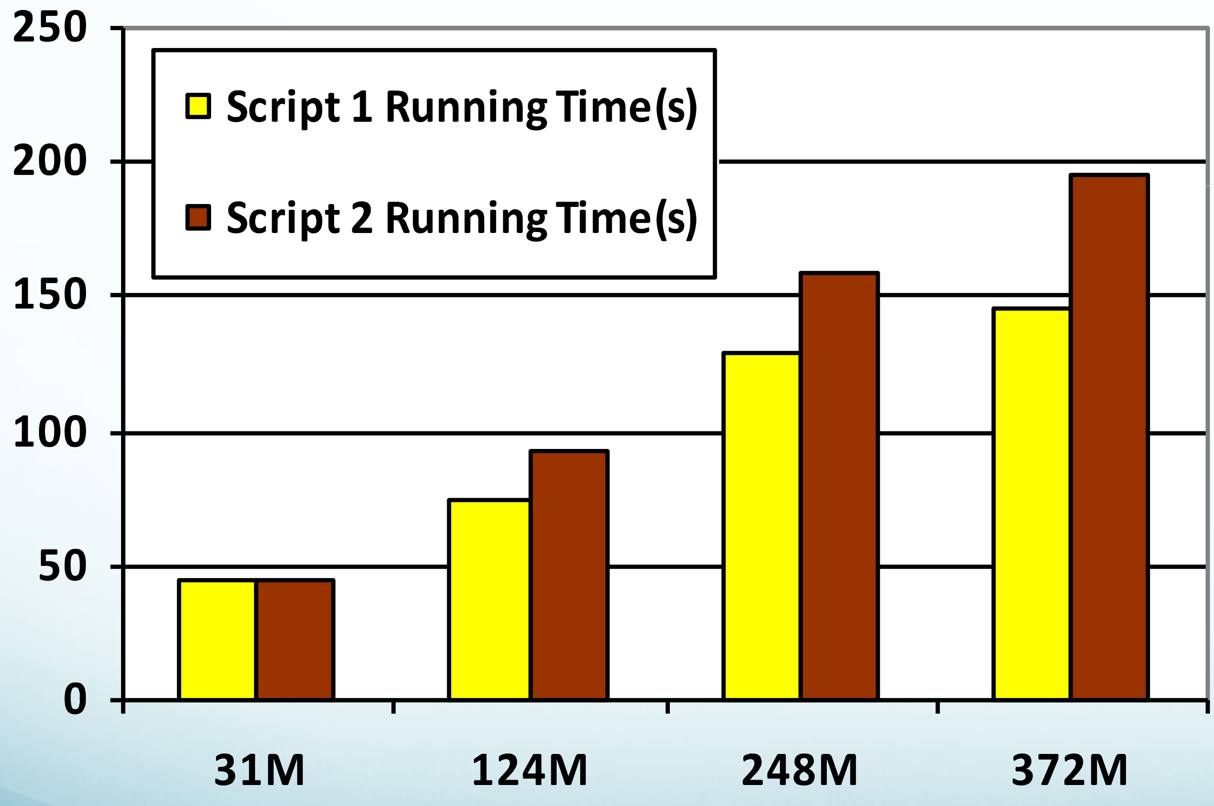
Security Argument



(a) Map-Reduce Plan

- Check is performed on reduce, which is executed by a trusted worker. The check logic cannot be leaked to the mapper.
- The map/reduce task can be obfuscated to hide the invariant.

Performance evaluation



3 virtual machines in local cluster:

- 1 as master and trusted worker.
- 2 as untrusted workers.

0-35% of slow down

Agenda

- Problem Statement
- MapReduce Task Layer Solution
- Application Layer Solution
- Conclusion & Future Work

Conclusion

- IntegrityMR explores Big Data analytic integrity from two alternative layers
  - Task layer:
    - Trusted private cloud + untrusted multiple public clouds architecture.
    - Replication, verification, credit-based management.
    - Experiment result: high integrity with non-negligible overhead
  - Application layer(Apache Pig):
    - Transform original script to introduce invariant in the map tasks
    - Check the invariant in the reduce task
    - Practice the idea by manually transform the script.

Future Works

- MapReduce task layer
  - Improve system performance by reducing cross-cloud communication and alleviate the DFS bottle neck.
- Application layer
  - Automating pig script transformation

Thank you!

Q&A