SecHadoop: End-to-End Privacy Preserving Hadoop

RK Shyamasundar
IIT Bombay
rkss@cse.iitb.ac.in
(joint work with
S. Swatish, Deepali Mittal, C. Aakash, NV Narendra Kumar)

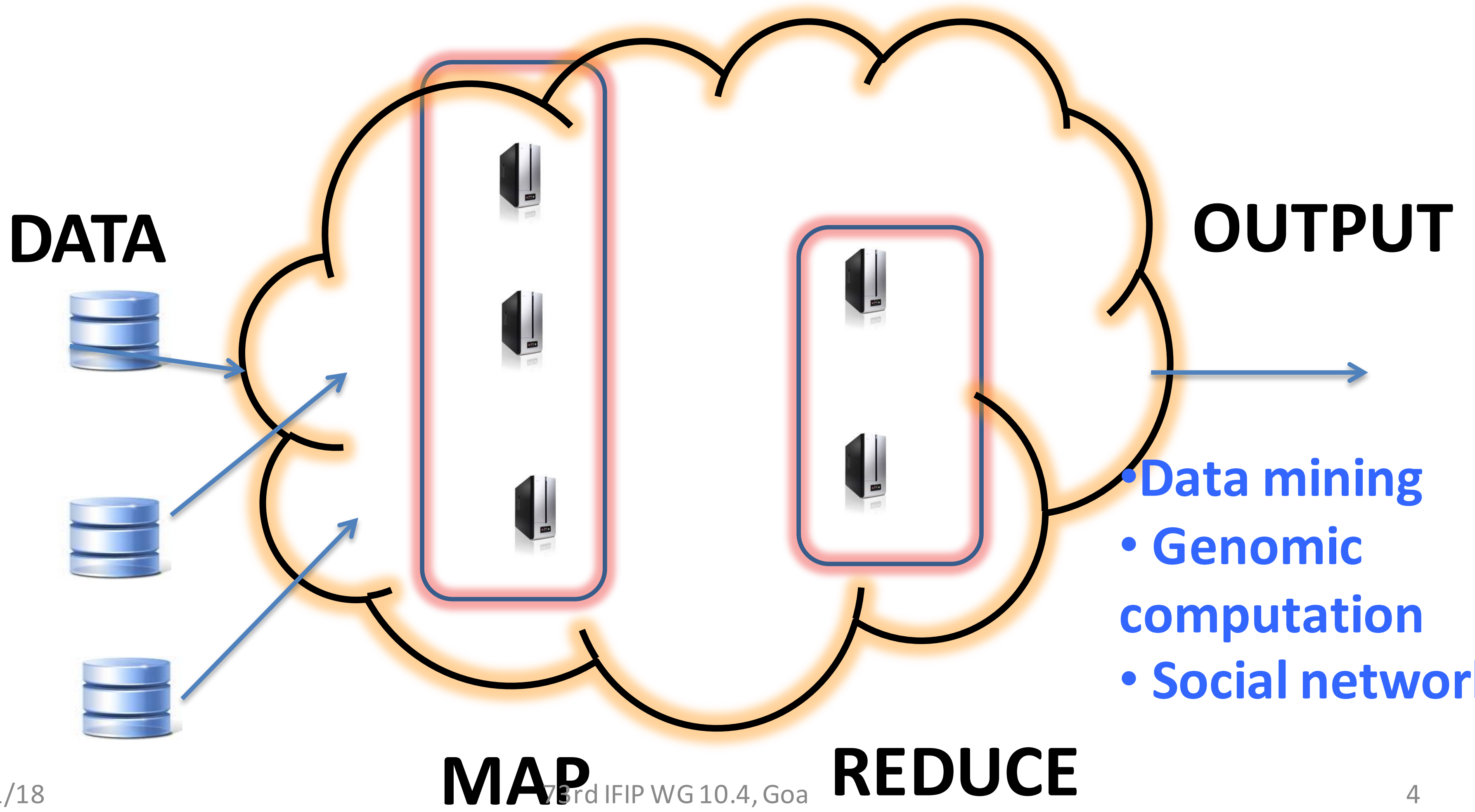
Agenda

- Need of Privacy preservation
- Problem
- Solution using RWFM Security Model
- Implementation Highlights
- Comparisons
- Ongoing Applications
- Conclusions

Motivation

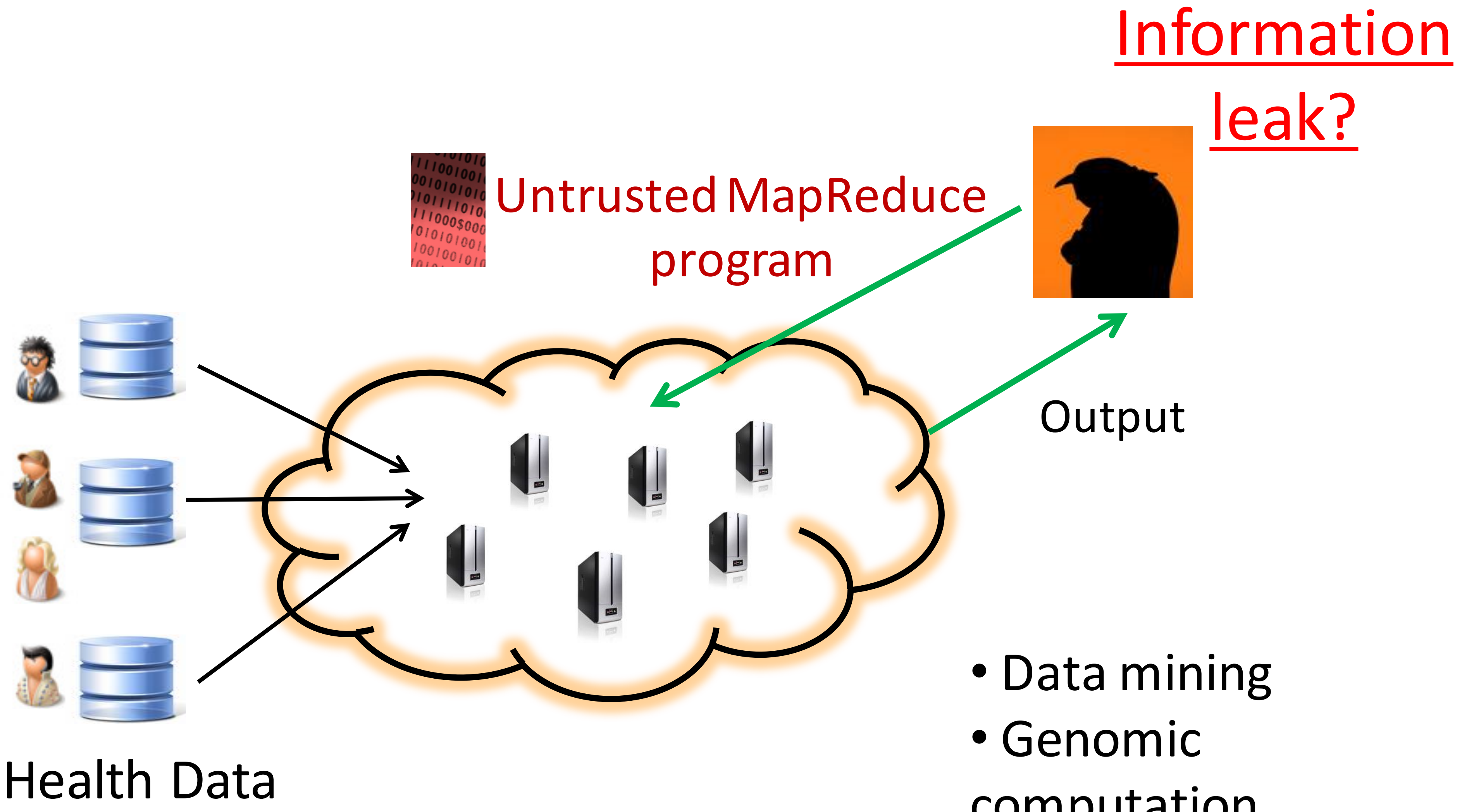
- Hadoop: Extensive scalable computations on massive data.
  - Achieves through Map-Reduce framework for computation and HDFS for distributed storage of the data.
- Privacy concerns arise due to
  - data divisions and intermediate data creations that is taking place while the computations are being carried out.
- User intervention in job execution in the form of Malware (or learning) in Hadoop can lead to privacy breaches.
  - Naturally requires a robust **decentralized information flow control**
- Secure end-to-end data flow in a Hadoop in a decentralized way preserving the original data privacy invariant
- Use such a scheme to protect against
  - Learning systems,
  - Desensitization of data without any leak,
  - Realize Provenance for Neurological data

MAP REDUCE



- Data mining
- Genomic computation
- Social networks

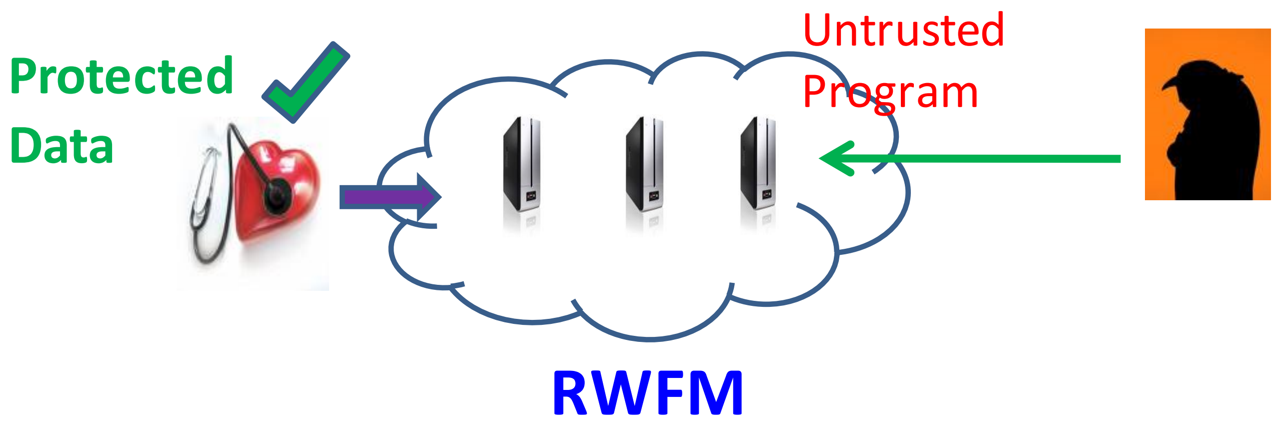
Impact of Untrusted Program



- Data mining
- Genomic computation
- Social networks

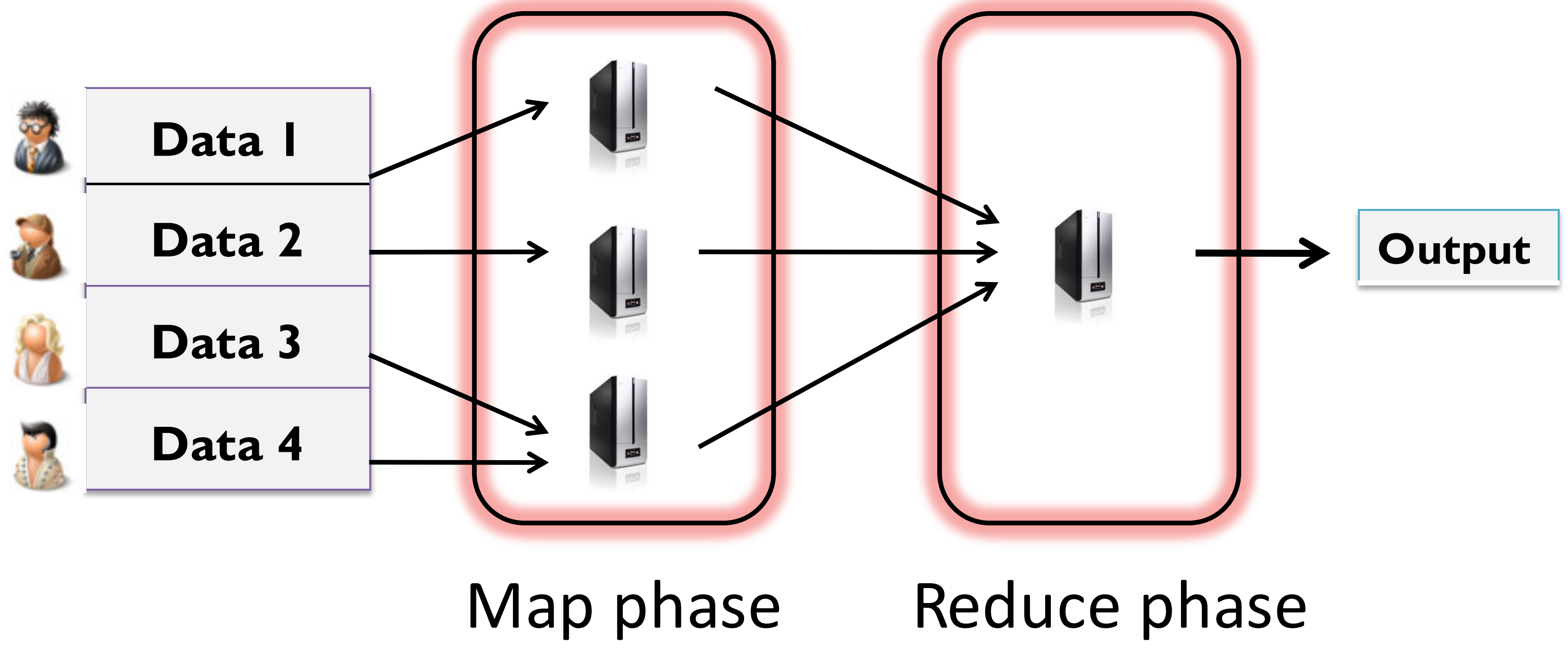
Realizing Privacy

Framework for privacy-preserving MapReduce computations with untrusted code.



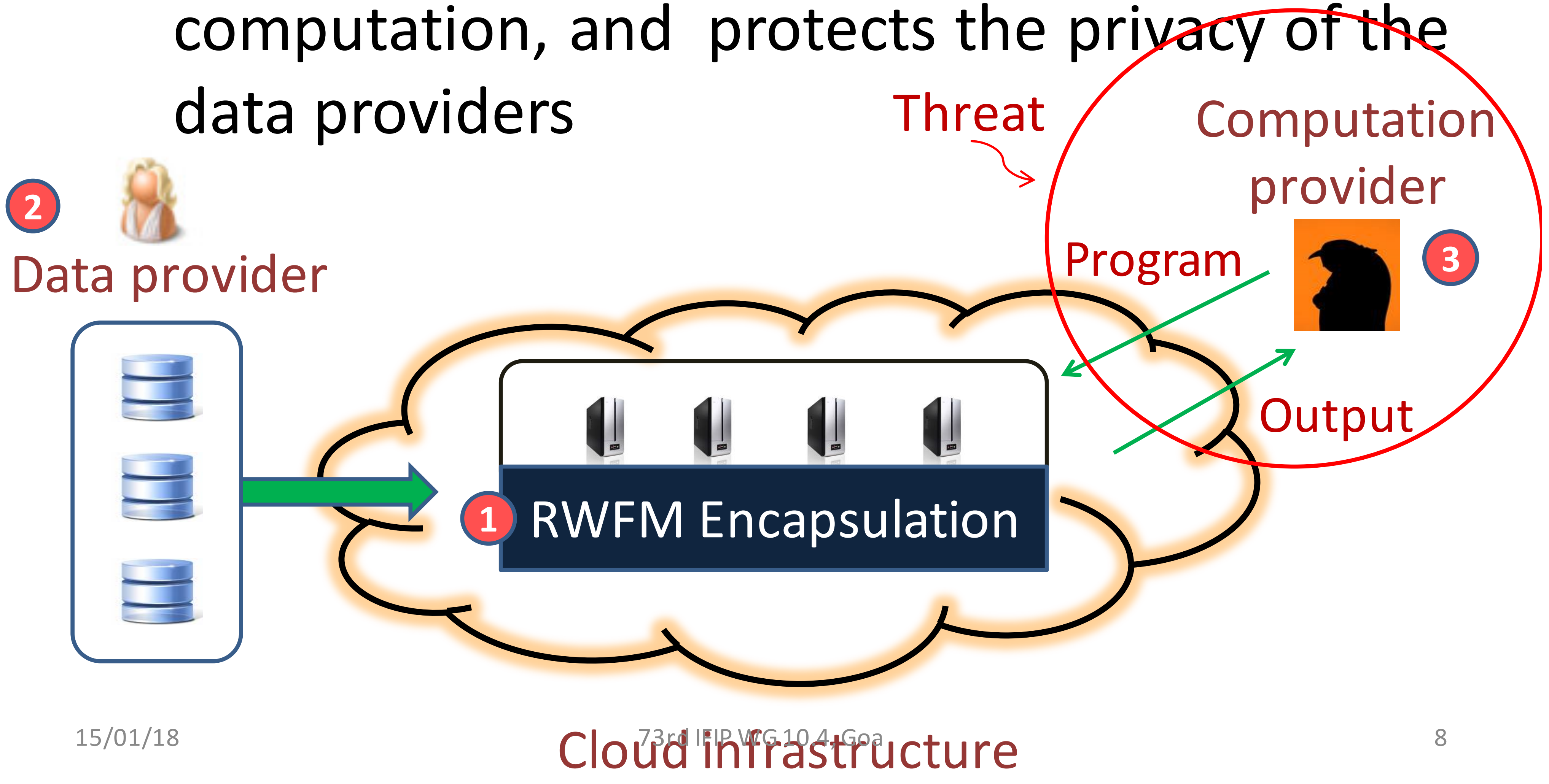
Background: MapReduce

$\text{map}(k_1,v_1) \rightarrow \text{list}(k_2,v_2)$
$\text{reduce}(k_2, \text{list}(v_2)) \rightarrow \text{list}(v_2)$



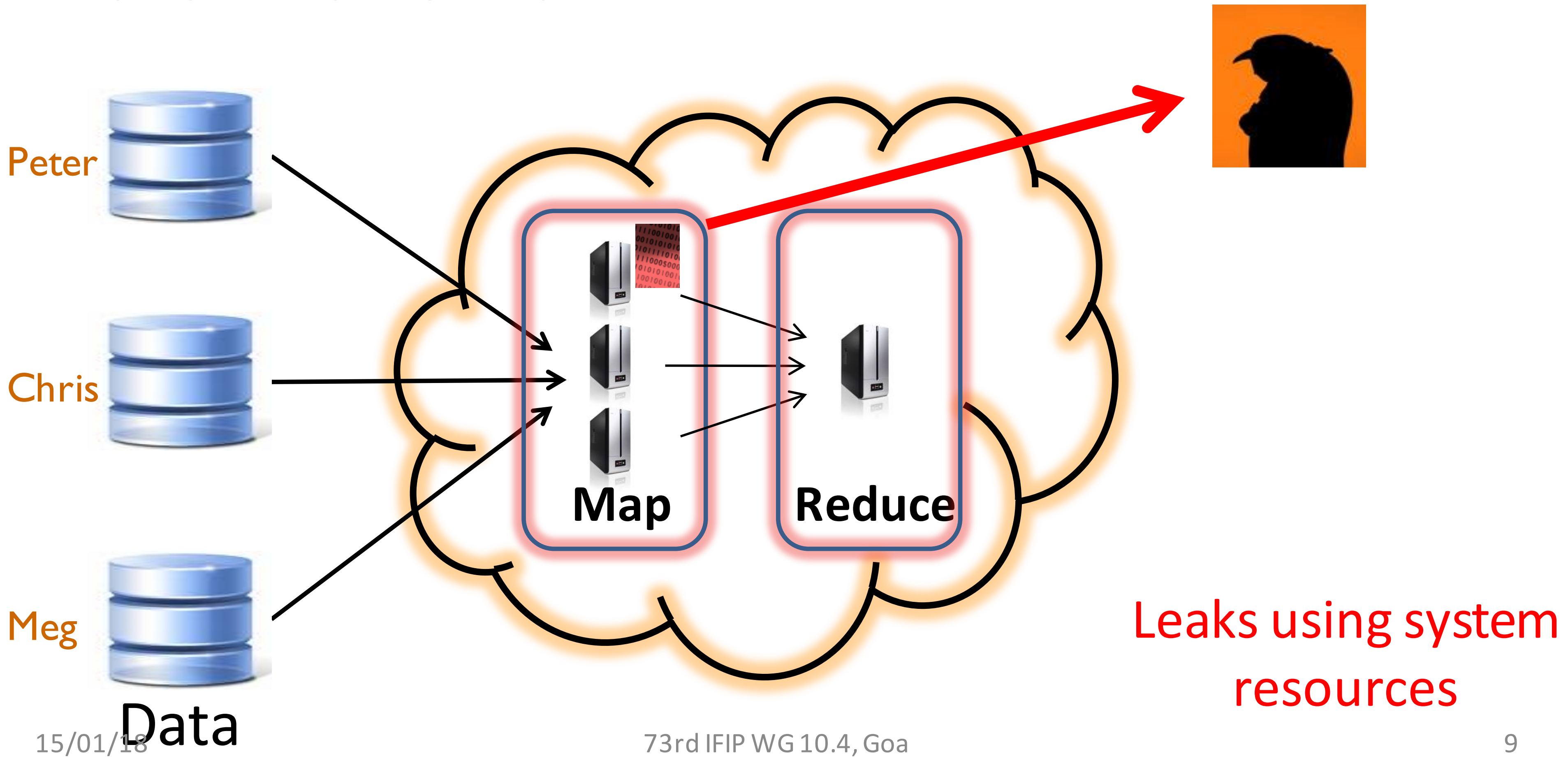
Threat model

- RWFM encapsulation monitors the computation, and protects the privacy of the data providers



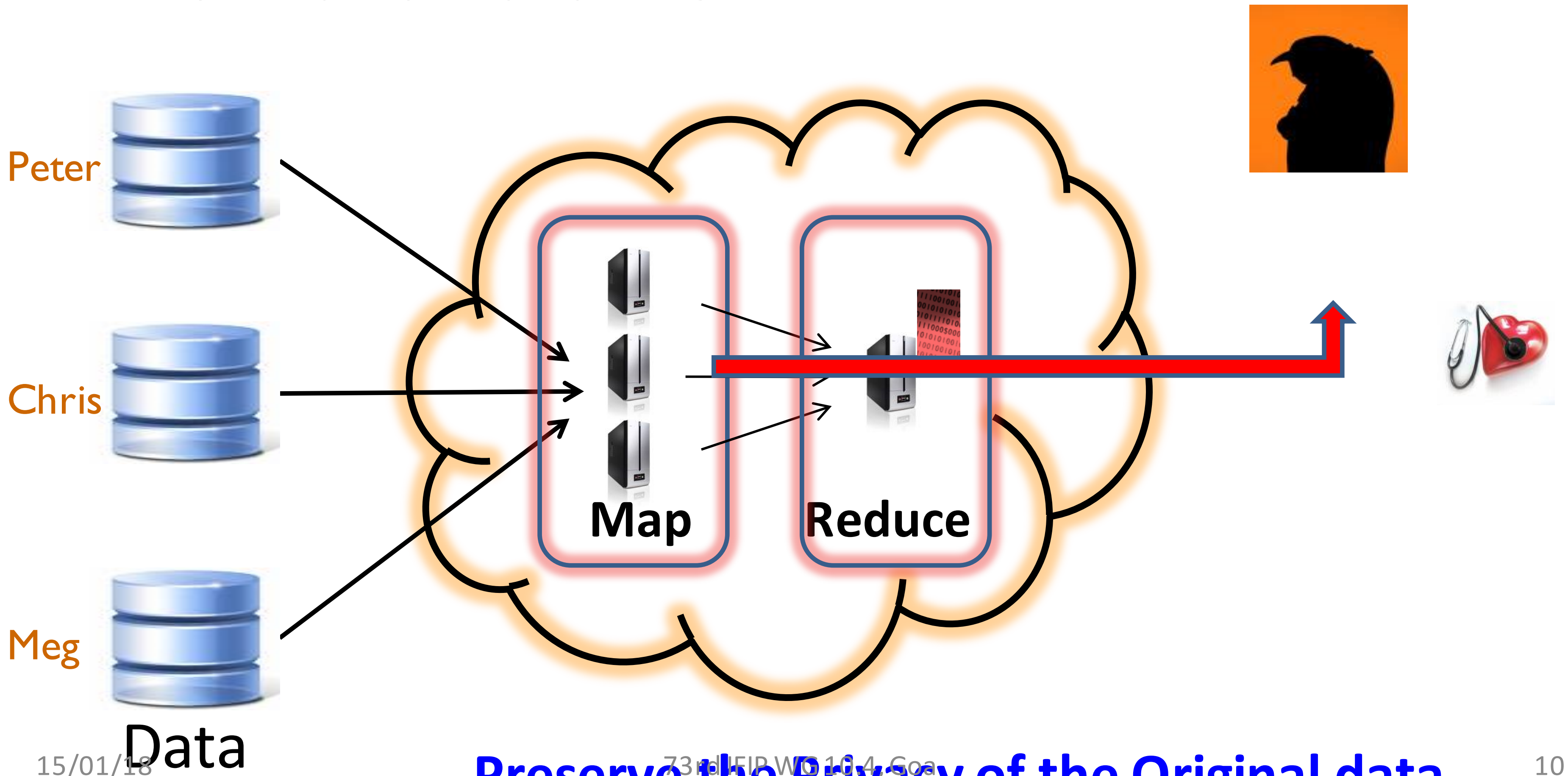
Challenge 1: Untrusted mapper

- Untrusted mapper code copies data, sends it over the network



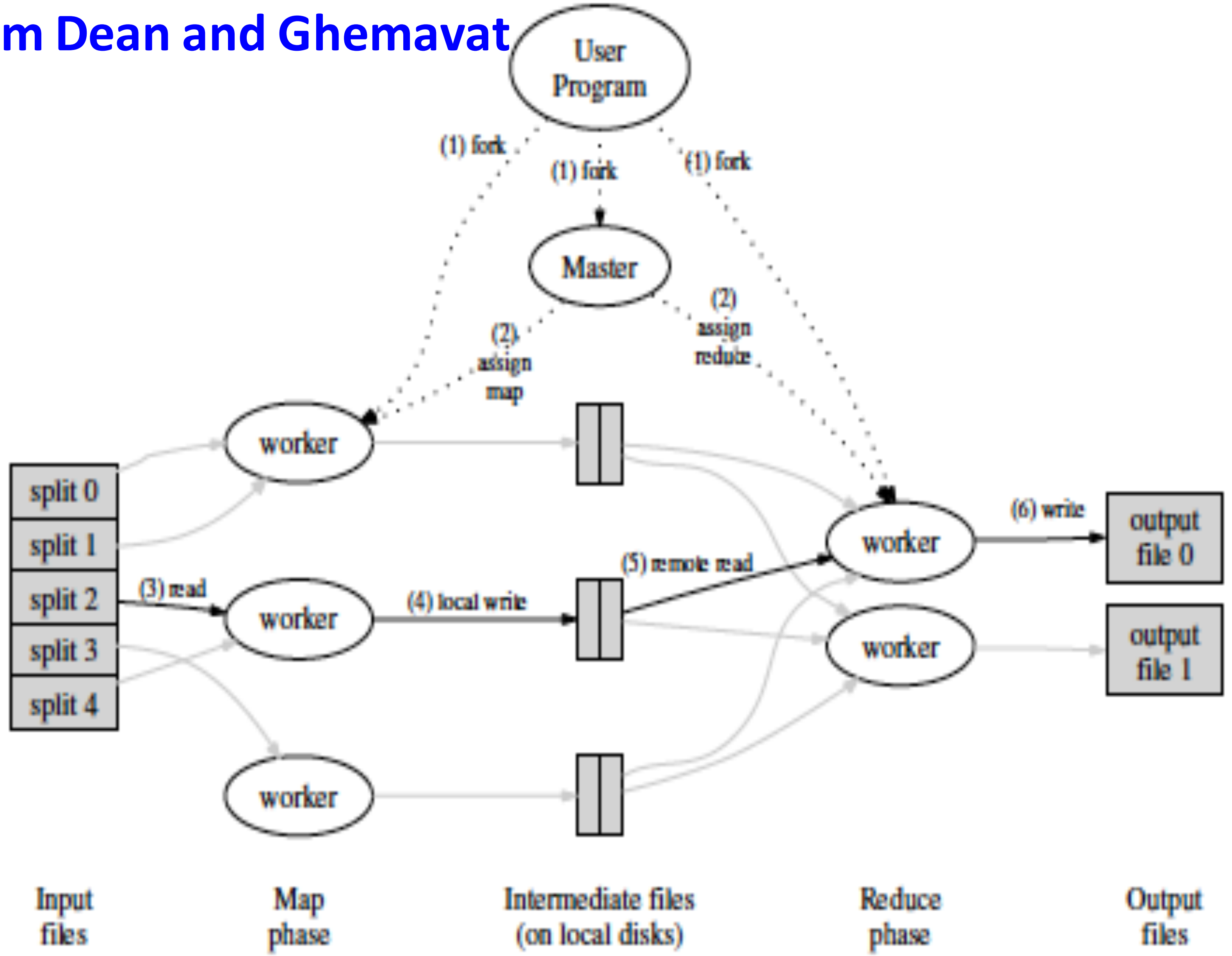
Challenge 2: Untrusted Reducer

- Output of the computation is also an information channel



Preserve the Privacy of the Original data

From Dean and Ghemavat



Status of Decentralized Information Flow Model

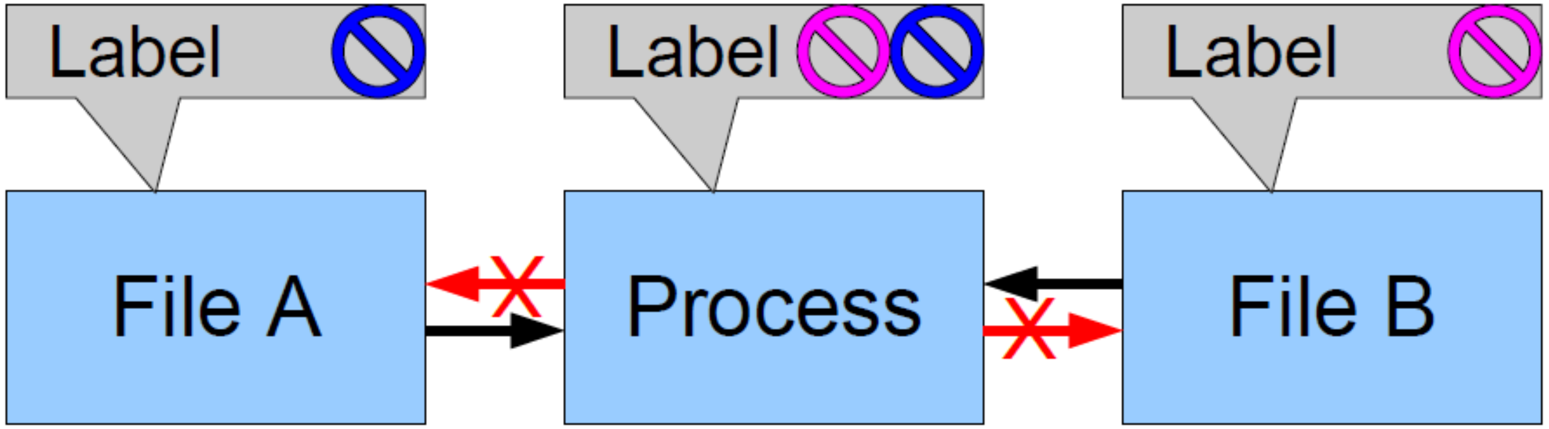
Decentralized Label Model

Myers and Liskov (2000)

- First model after **the seminal Lattice Information Flow model of Dorothy Denning (the lattice defines the flow)**
- Addresses the weaknesses of earlier approaches to the protection of confidentiality in a system containing untrusted code or users, even in situations of mutual distrust
- Allows users to control the flow of their information without imposing the rigid constraints of a traditional MLS
- Defines a set of rules that programs must follow in order to avoid leaks of private information
- Protects confidentiality for users and groups rather than for a monolithic organization
- Introduces a richer notion of declassification
  - in the earlier models it was done by a trusted subject; in this model principals can declassify their own data

Labels control information flow

- 🟦 Color is category of data (e.g. my files)
- 🛇 Blue data can flow only to other blue objects



Issues of State-of-the-art (a)

- 1985 Trusted Computer Systems Evaluation Criteria (Orange Book)
  - defines the security of a computer system by how well it **implements flow control** and how good its assurance is
- Despite huge efforts, systems developed had several drawbacks:
  - large TCB, slow, not easy to use, and very limited functionality

Issues of State-of-the-art (b)

- 2000 Myers & Liskov (DLM)
  - First Decentralized Label Model after 25 years (Myers and Liskov) – Cf. B Lampson
  - only readers for protecting confidentiality and only writers for protecting integrity
  - Issues: *for a proper tracking of any information flow property, it is important to control both reading and writing by subjects*

Issues of State-of-the-art (c)

- HiStar, Flume and Laminar systems
  - based on the product of Confidentiality and Integrity
  - **Issues**: *confidentiality and integrity are not orthogonal properties and issues of treating Declassification as a DAC*
  - Fred Schneider, in his book# chapter, clearly brings out the perils of combining confidentiality and integrity policies in this manner

# yet to be published,
available at http://www.cs.cornell.edu/fbs/publications/chptr.MAC.pdf

Issues of State-of-the-art (d)

- 2012 Mitchell et al. (DC labels)
  - not easy to derive consistent DC labels for modelling a given requirement
  - Flaw: *support for downgrading (discretionary control) is orthogonal to the IFC, thus, defeating the purpose of the mandatory controls*
- *New Robust decentralized Information Flow control model – RWFM (2016,2017) – Readers Writers Flow Model*

RWFM BASICS

RWFM Model

NV Narendra Kumar and RKs 2016, 2017

Readers-Writers Labels

- Security requirements of practical applications are often stated / easily understood in terms of who can read / write information
- Observations:
  - information readable by s_1 and $s_{2,}$ can-flow-to information readable only by s_1
  - information writable only by $s_{1,}$ can-flow-to information writable by s_1 and s_2
- Readers and writers can be used as labels!!

RWFM Label Format

- (owner/authority, readers, writers)
  - First component is a single subject denoting
    - *owner* in case of an object label
    - *authority* in case of a subject label
  - Second component is a set of subjects denoting
    - permissible readers in case of an object label
    - subjects who can read all the objects that this subject can read in case of a subject label
  - Third component is a set of subjects denoting
    - permissible writers in case of an object label
    - subjects who can write all the objects that this subject can write in case of a subject label

State of an Information System

- State of an information system is defined as the set of subjects and objects in the system together with their labels. Initial state
  - Objects and their labels as required for application
  - Each subject s starts with label $(s,*,\phi)$
- Whenever a subject tries to perform an operation on an object, it may lead to a state change and will have to be permitted only if deemed safe
  - Read
  - Write
  - Create
  - Downgrade
  - Relabel

State Transitions in RWFM

- Subject s with label (s_1,R_1,W_1) requests <u>*read*</u> access to an object o with label (s_2,
  - If $s_1 \in R_2$ then
    - relabel s to (s_1,$R_1 \cap R_2$,$W_1 \cup W_2$) and ALLOW access
  - Else
    - DENY access
- <u>POSSIBLE</u> state change (label of s may change)

s can

s has accessed information accessible only by

s is influenced by both W_1 and W_2

State Transitions in RWFM

- Subject s with label (s R W) requests *write*
  (s_2, 2 2)

all subjects can access

all subjects that have influenced the current information of s can also influence o

s can write o

  - If $s_1 \in W_2$ and $R_1 \supseteq R_2$ and $W_1 \subseteq W_2$ then
    - ALLOW access
  - Else
    - DENY access
- NO state change

State Transitions in RWFM

s, and all subjects that have influenced the current information of s have influenced o

accessed by s so far, car

- Subject s with label (s,R, uests *creation* of an object o
  - – create an object o and label it (s,R,W∪{s})

- DEFINITE state change (a new object is added to the system)

State Transitions in RWFM

- Subject s w
  an

$W_3)$

subjects that can access o can also access

subjects that could not access o but can access its downgraded version must have influenced information in o

all the subjects … s o can access its downgra… on also

  – If s_1 … and $s_1=s_2$ … and $W_1=W_2=W_3$ and $R_1=R_2$ and $R_3 \supseteq R_2$ and $R_3-R_2 \subseteq W_2$ then
    - ALLOW
  – Else
    - DENY
- POSSIBLE state change (label of o may change)

State Transitions in RWFM

s, and all subjects that influenced the current information of s have influenced the relabelling

all subjects that can access the relabelled object, could have accessed all the information that s has accessed so far, and the original object

- Subject s w
  - If $s_1 \in R_2$ ... $s_1=s_2=s_3$ and $W_2 \subseteq W_1$ and $W_3=W_1 \cup \{s\}$ and $R_2 \supseteq R_1 \supseteq R_3$ then
    - ALLOW
  - Else
    - DENY
- <u>POSSIBLE</u> state change (label of o may change)

Downgrading (Declassifying)

- For practical applications, adding readers (downgrading) to the result of a computation is essential for use by relevant parties
- Downgrading rules
  - only the owner of information may downgrade it
  - if a single source is responsible for the information, then readers that can be added is unrestricted
  - if multiple sources influenced the information, then only those who influenced it may be added as readers

RWFM permits intuitive specifications with simple access checks

- The above proposition simplifies the access check to s∈R(o) for subject s to read object o and s∈W(o) for subject s to write object o.

Example-1
WebTax

- Bob provides his tax-data to a professional tax preparer, who computes Bob's final tax form using a private database of rules for minimizing the tax payable and returns the final form to Bob
- Security requirements
  1. Bob requires that his tax-data remains confidential
  2. Preparer requires that his private database remains confidential

Example-1
WebTax

$$\begin{tikzcd}
\overset{1}{TD^{(B,\{B,P\},\{B\})}} \arrow[dr] & & \\
& \overset{1,2}{IR^{(P,\{P\},\{B,P\})}} \arrow[r, dashed] & \overset{1,2'}{FF^{(P,\{B,P\},\{B,P\})}} \\
\overset{2}{DB^{(P,\{P\},\{P\})}} \arrow[ur] & &
\end{tikzcd}$$

TD	Tax-data	IR	Intermediate results
DB	Database of tax optimization rules	FF	Final tax form
$\longrightarrow$	Flows-to	$\dashrightarrow$	Downgraded-to

Example-1
WebTax

	DLM	DC	RWFM
TD	{B: B}	(B, B)	(B, {B,P}, {B})
DB	{P: P}	(P, P)	(P, {P}, {P})
IR	{B: B; P: P}	(B∧P, B∨P)	(P, {P}, {B,P})
FF	{B: B}	(B, B∨P)	(P, {B,P}, {B,P})

- <u>DLM label format</u>: policies separated by ';', where each policy is of the form 'owner: readers'
- <u>DC label format</u>: 'readers, writers', where readers control confidentiality, writers control integrity
- <u>RWFM label format</u>: 'owner, readers, writers'

DLM, DC and RWFM Comparison

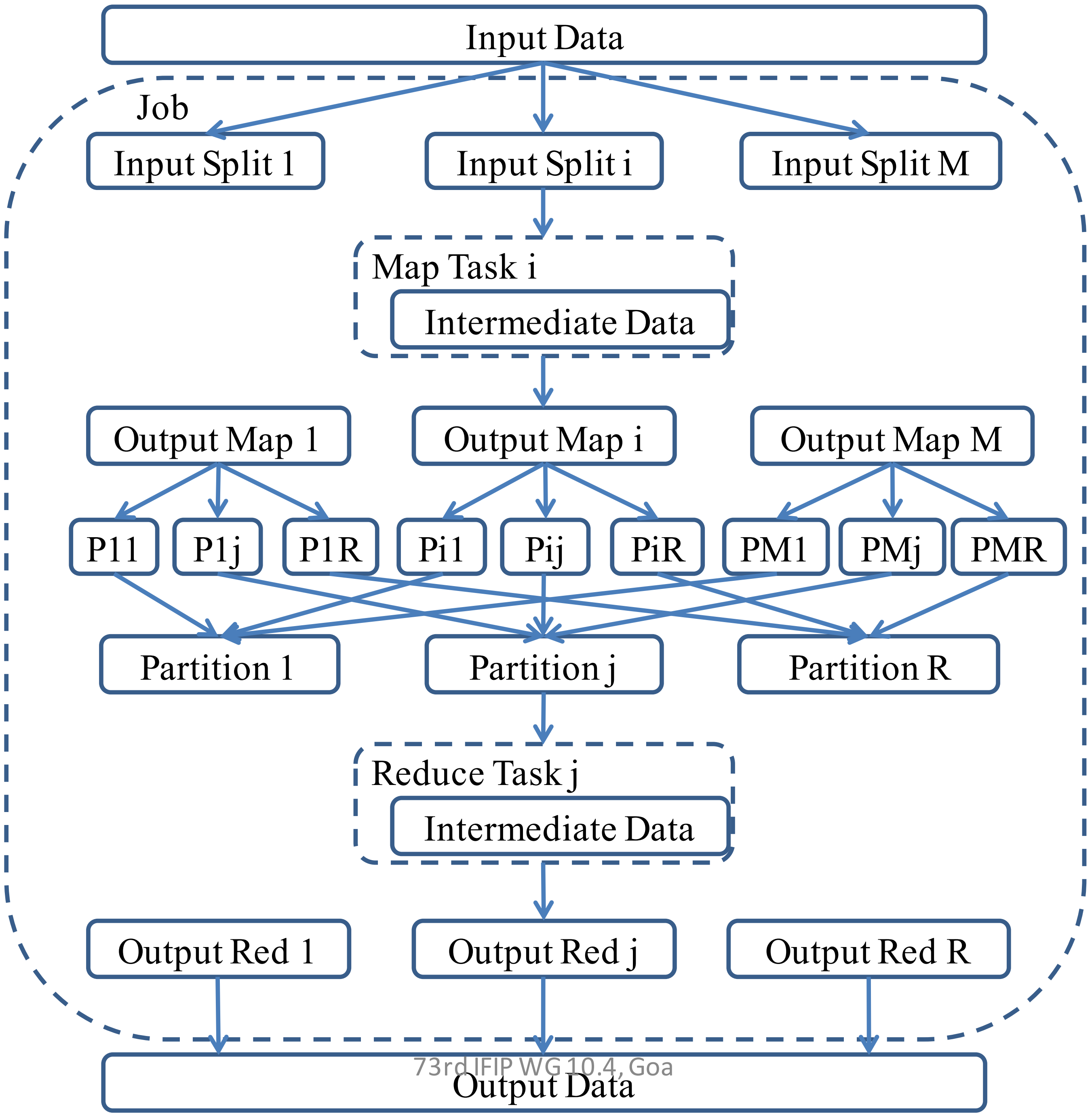
	DLM	DC	RWFM
Confidentiality	only Readers	only Readers	Readers and Writers
Integrity	only Writers	only Writers	Readers and Writers
Downgrading (DAC)	Purely discretionary	Purely discretionary	Consistent with IFC (MAC)
Ownership	Explicit	Implicit	Explicit
Authority	Orthogonal to the label	Orthogonal to the label	Explicit in the label

DLM, DC and RWFM Comparison

	DLM	DC	RWFM
Principal hierarchy and Delegation	Orthogonal to the label	Orthogonal to the label	Embedded in the label
Bi-directional flow	Difficult	Difficult	Simple and Accurate
Ease of use	Moderate	Moderate	Easy
Label size	Moderate to Large	Large	Small
No. of labels	Large	Large	Small (as required by the application)

Labelling Map Reduce Framework

Flow of a MapReduce Job



Flow of a MapReduce Job

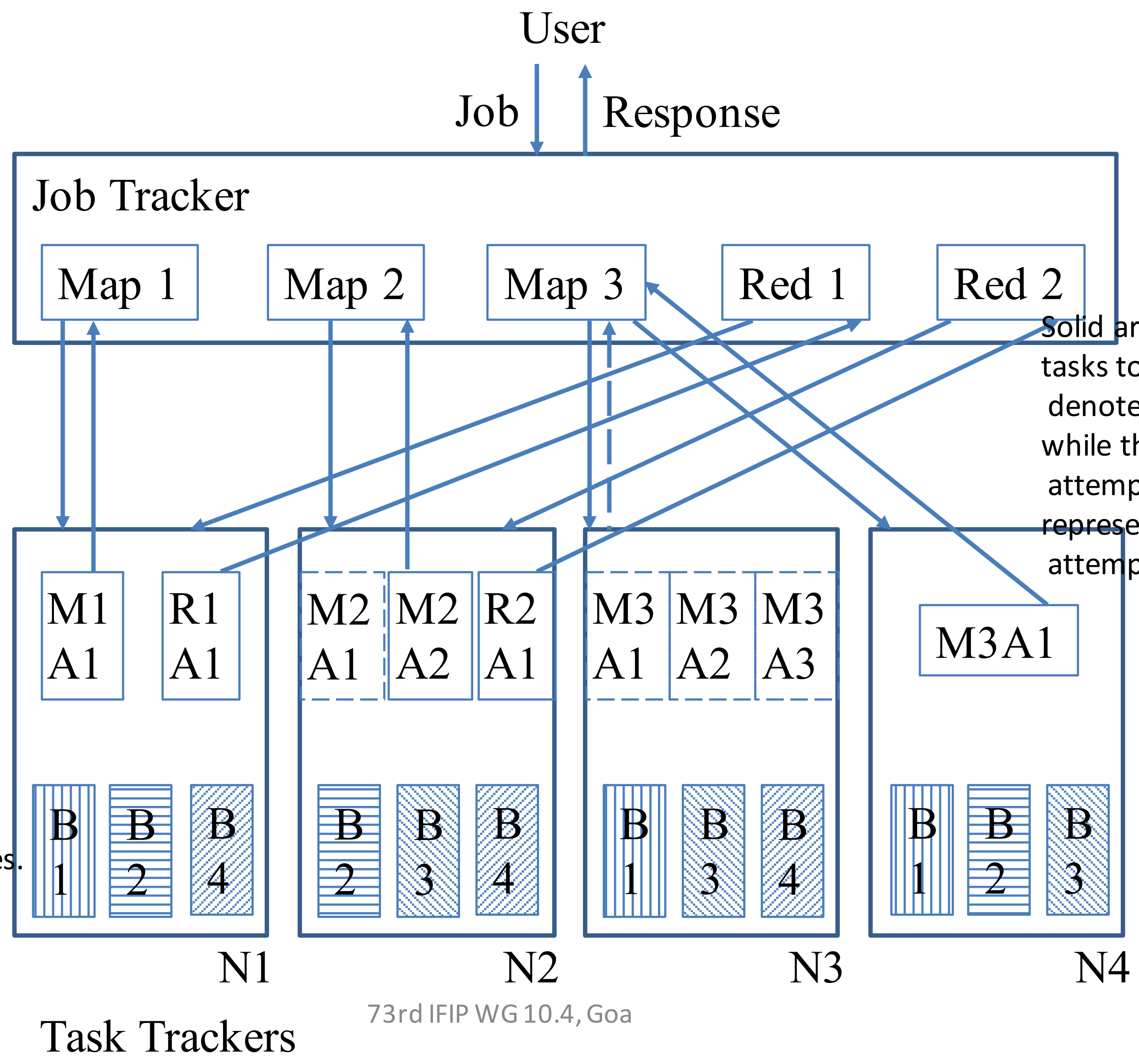
1. The job tracker splits input data, and creates and assigns map tasks
2. Map tasks execute on slave nodes to produce intermediate results
3. Job tracker partitions (shuffles and sorts) the intermediate results and assigns reduce tasks
4. Reduce tasks execute on slave nodes to produce final results
5. Job tracker aggregates the final results and produces the output for the user

Example Configuration of MapReduce

Dotted arrows from nodes to tasks represent failure of execution of the task on the node– happens due to data corruption

Boxes with dashes outlines represent failed attempts . E.g.,, attempt 1 of map 2 (M2A1) fails on N2, which then creates M2A2 which succeeds.

Shaded boxes represent data stored on the nodes. E.g., blocks 1, 2 and 4 are stored on node N1.

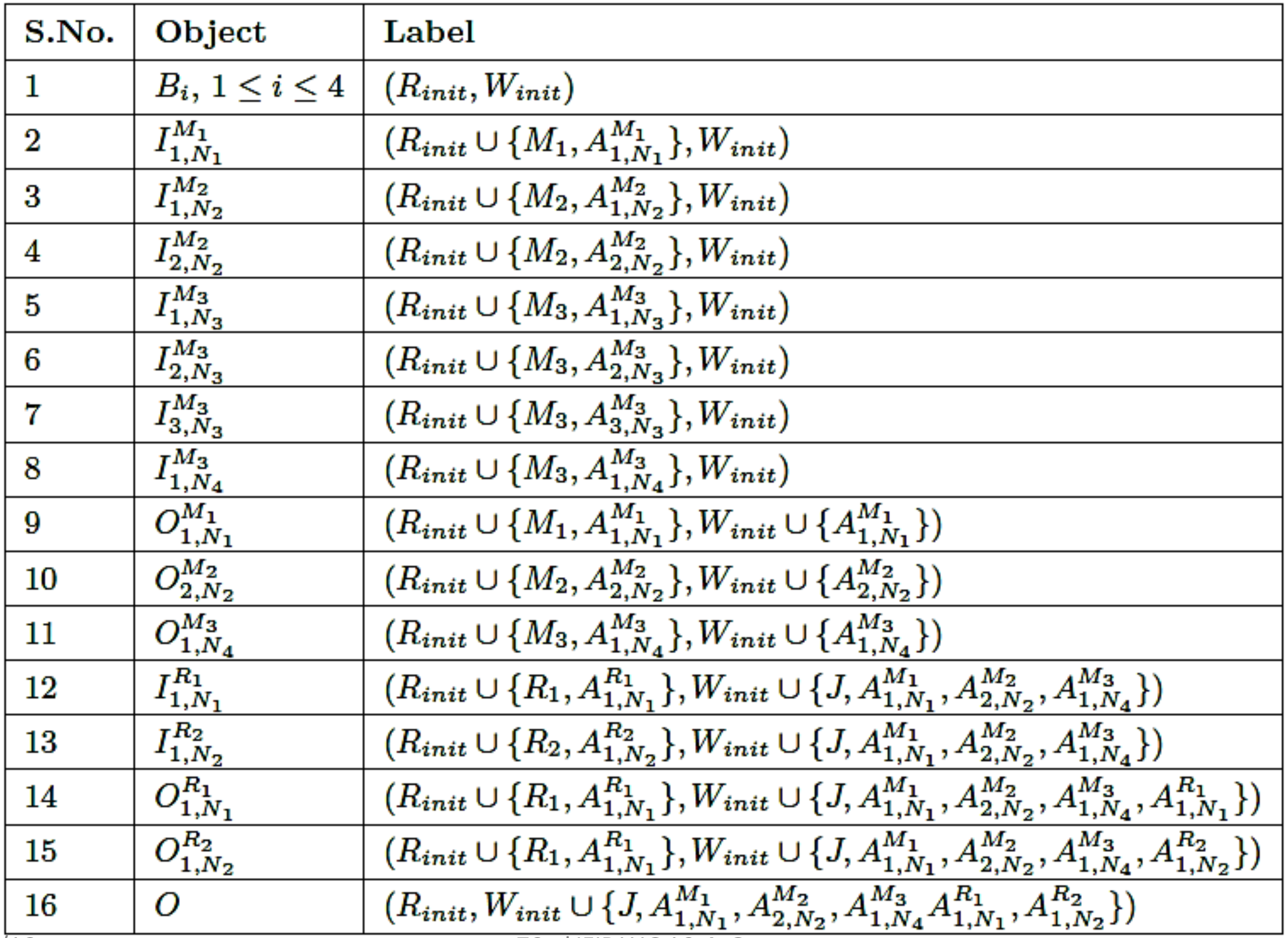


Solid arrows from tasks to nodes denote assignments, while those from attempts to tasks represent successful attempts.

Notation

1. Shaded boxes represent data stored on the nodes. For example, blocks 1, 2 and 4 are stored on node N1.
2. Boxes with dashes outlines represent failed attempts. For example, attempt 1 of map 2 (M2A1) fails on N2, which then creates M2A2 which succeeds.
3. Solid arrows from tasks to nodes denote assignments, while those from attempts to tasks represent successful attempts.
4. Dotted arrows from nodes to tasks represent failure of execution of the task on the node. This happens – potentially due to data corruption – when a threshold number of attempts of a task on a node fail. For example, task map 3 fails on node 3.

Labels for Example Configuration



S.No.	Object	Label
1	B_i, $1 \le i \le 4$	(R_{init}, W_{init})
2	$I^{M_1}_{1,N_1}$	$(R_{init} \cup \{M_1, A^{M_1}_{1,N_1}\}, W_{init})$
3	$I^{M_2}_{1,N_2}$	$(R_{init} \cup \{M_2, A^{M_2}_{1,N_2}\}, W_{init})$
4	$I^{M_2}_{2,N_2}$	$(R_{init} \cup \{M_2, A^{M_2}_{2,N_2}\}, W_{init})$
5	$I^{M_3}_{1,N_3}$	$(R_{init} \cup \{M_3, A^{M_3}_{1,N_3}\}, W_{init})$
6	$I^{M_3}_{2,N_3}$	$(R_{init} \cup \{M_3, A^{M_3}_{2,N_3}\}, W_{init})$
7	$I^{M_3}_{3,N_3}$	$(R_{init} \cup \{M_3, A^{M_3}_{3,N_3}\}, W_{init})$
8	$I^{M_3}_{1,N_4}$	$(R_{init} \cup \{M_3, A^{M_3}_{1,N_4}\}, W_{init})$
9	$O^{M_1}_{1,N_1}$	$(R_{init} \cup \{M_1, A^{M_1}_{1,N_1}\}, W_{init} \cup \{A^{M_1}_{1,N_1}\})$
10	$O^{M_2}_{2,N_2}$	$(R_{init} \cup \{M_2, A^{M_2}_{2,N_2}\}, W_{init} \cup \{A^{M_2}_{2,N_2}\})$
11	$O^{M_3}_{1,N_4}$	$(R_{init} \cup \{M_3, A^{M_3}_{1,N_4}\}, W_{init} \cup \{A^{M_3}_{1,N_4}\})$
12	$I^{R_1}_{1,N_1}$	$(R_{init} \cup \{R_1, A^{R_1}_{1,N_1}\}, W_{init} \cup \{J, A^{M_1}_{1,N_1}, A^{M_2}_{2,N_2}, A^{M_3}_{1,N_4}\})$
13	$I^{R_2}_{1,N_2}$	$(R_{init} \cup \{R_2, A^{R_2}_{1,N_2}\}, W_{init} \cup \{J, A^{M_1}_{1,N_1}, A^{M_2}_{2,N_2}, A^{M_3}_{1,N_4}\})$
14	$O^{R_1}_{1,N_1}$	$(R_{init} \cup \{R_1, A^{R_1}_{1,N_1}\}, W_{init} \cup \{J, A^{M_1}_{1,N_1}, A^{M_2}_{2,N_2}, A^{M_3}_{1,N_4}, A^{R_1}_{1,N_1}\})$
15	$O^{R_2}_{1,N_2}$	$(R_{init} \cup \{R_1, A^{R_1}_{1,N_1}\}, W_{init} \cup \{J, A^{M_1}_{1,N_1}, A^{M_2}_{2,N_2}, A^{M_3}_{1,N_4}, A^{R_2}_{1,N_2}\})$
16	O	$(R_{init}, W_{init} \cup \{J, A^{M_1}_{1,N_1}, A^{M_2}_{2,N_2}, A^{M_3}_{1,N_4} A^{R_1}_{1,N_1}, A^{R_2}_{1,N_2}\})$

RKS and NV Kumar, 2016)

Security Properties Assured by the Labelling

- Privacy Invariance: security and privacy reqs on the inputs are maintained as an invariant throughout the computation including the intermediate data that is produced in the process

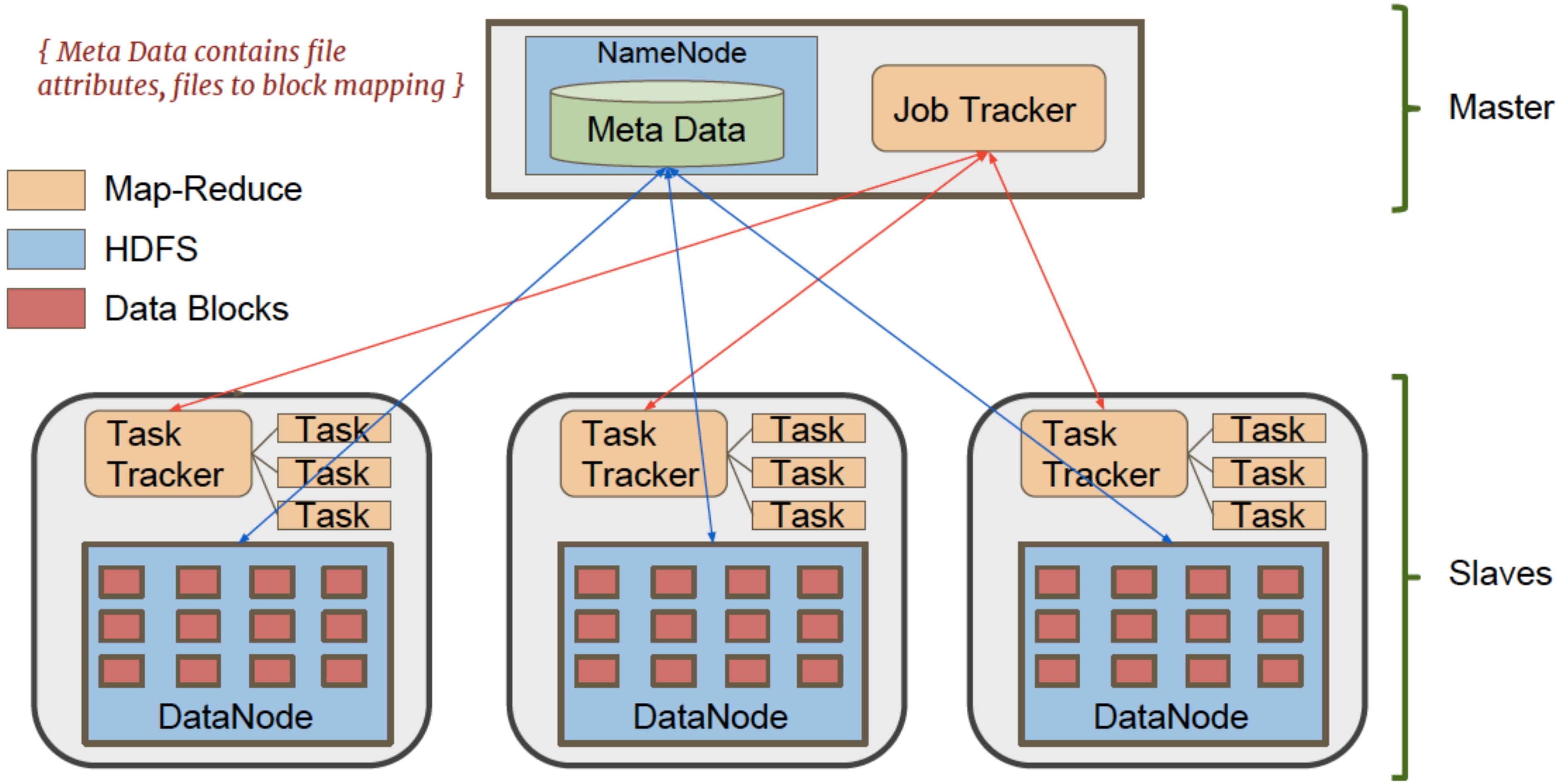
Security Properties Assured by the Labelling

- Protection from Malware: map and reduce are provided by the user and may be malicious, yet the attempt executing these tasks cannot access any data on the node other than the data provided as its input

Security Properties Assured by the Labelling

- Non-interference Free Execution: the attempts (could be of tasks of the same job or not) executing simultaneously on a give node are isolated due to labelling, and therefore cannot interfere with one another

Hadoop Architecture



Hadoop source is huge and complex which consists of 2.3 MLOC

Challenges in Implementation

CRUX

- Build an RWFM monitor to control the information flow
- Integrate DAC of HADOOP with the Information Flow labels of RWFM

Hadoop: Distributed Computing Infrastructure for Big Data Computations

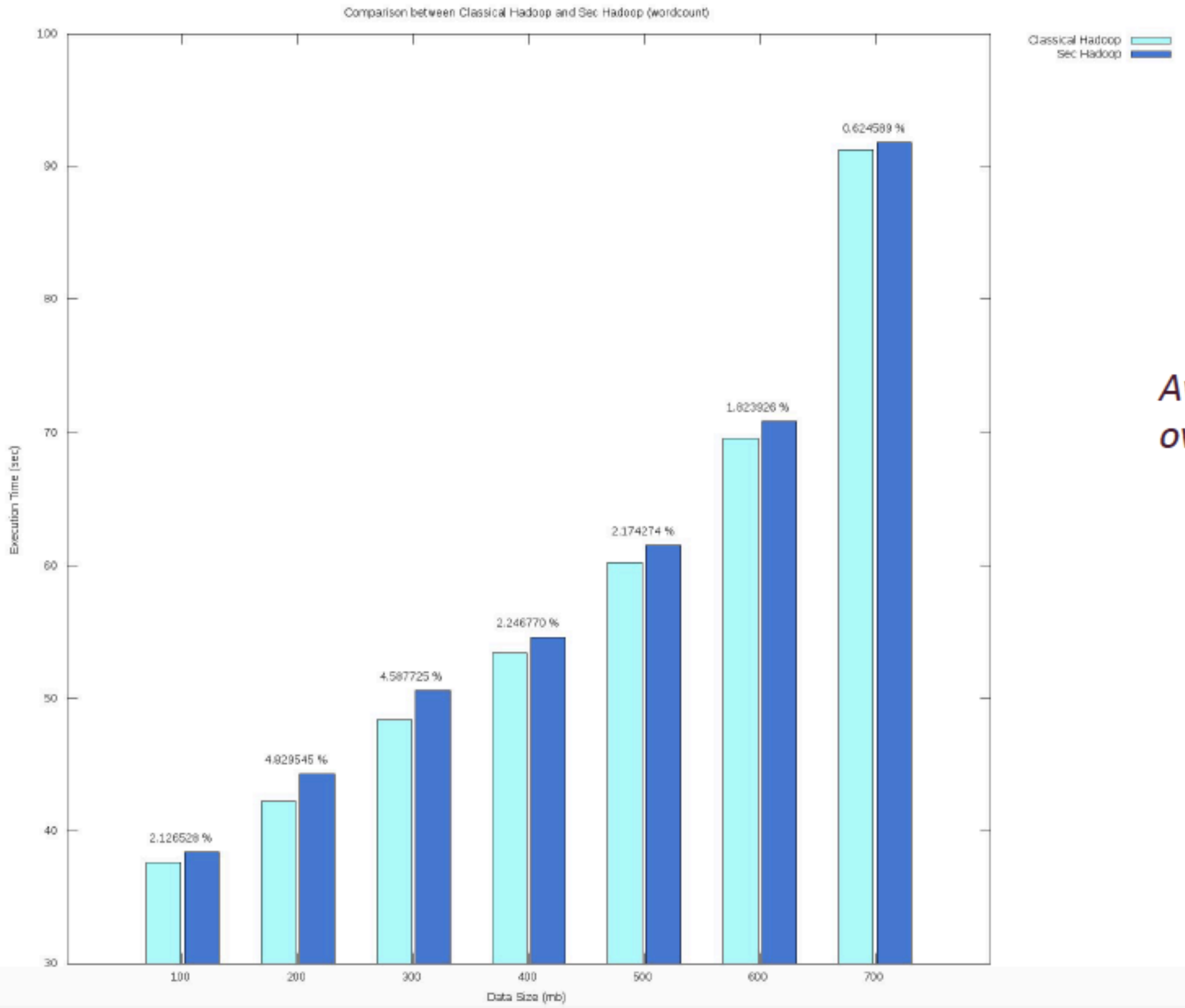
Hadoop Modules:

- Hadoop Distributed File System (HDFS)
  - Stores user data in files and provides redundancy for high availability.
- MapReduce Framework
  - Processes problems parallely on large data sets with large number of nodes.
  - Prefers locality of data, minimizes network congestion, increases overall throughput.
  - Advantages : Scalability, Fault Tolerance
- Identify possible points of leakage in the Hadoop System

Performance Results

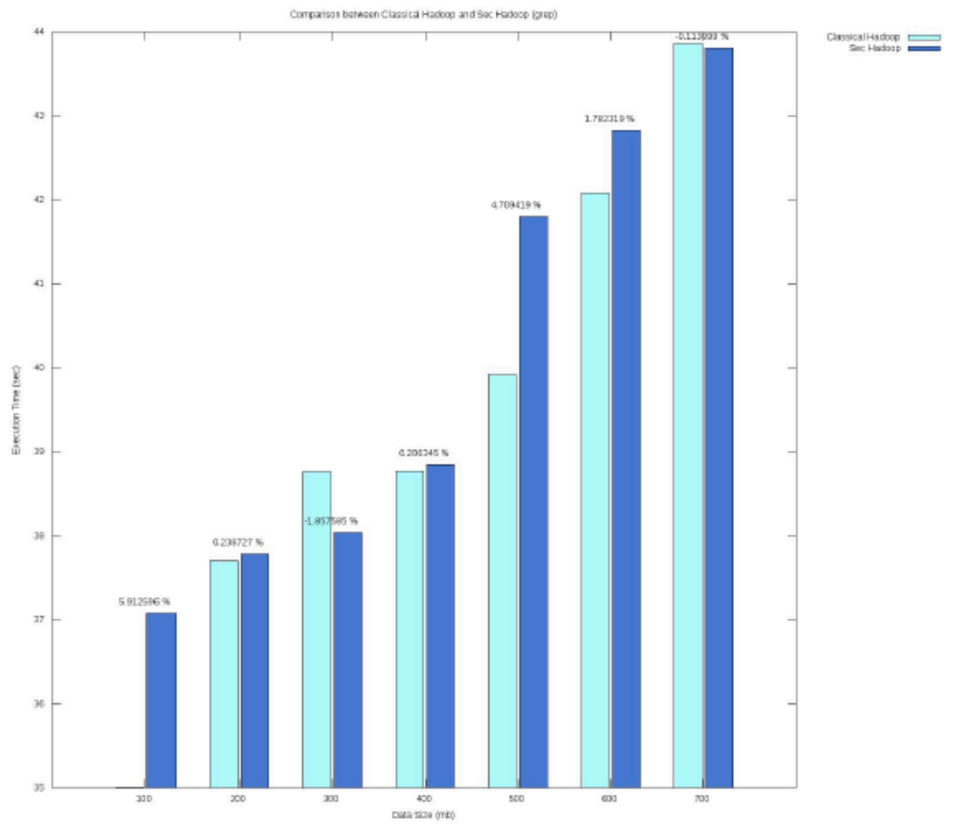
- Comparison between the performance of Classical Hadoop and SecHadoop by increasing input file size
- Performance overhead of SecHadoop is 2-5% more in comparison to Classical

Performance Comparison for WordCount Job



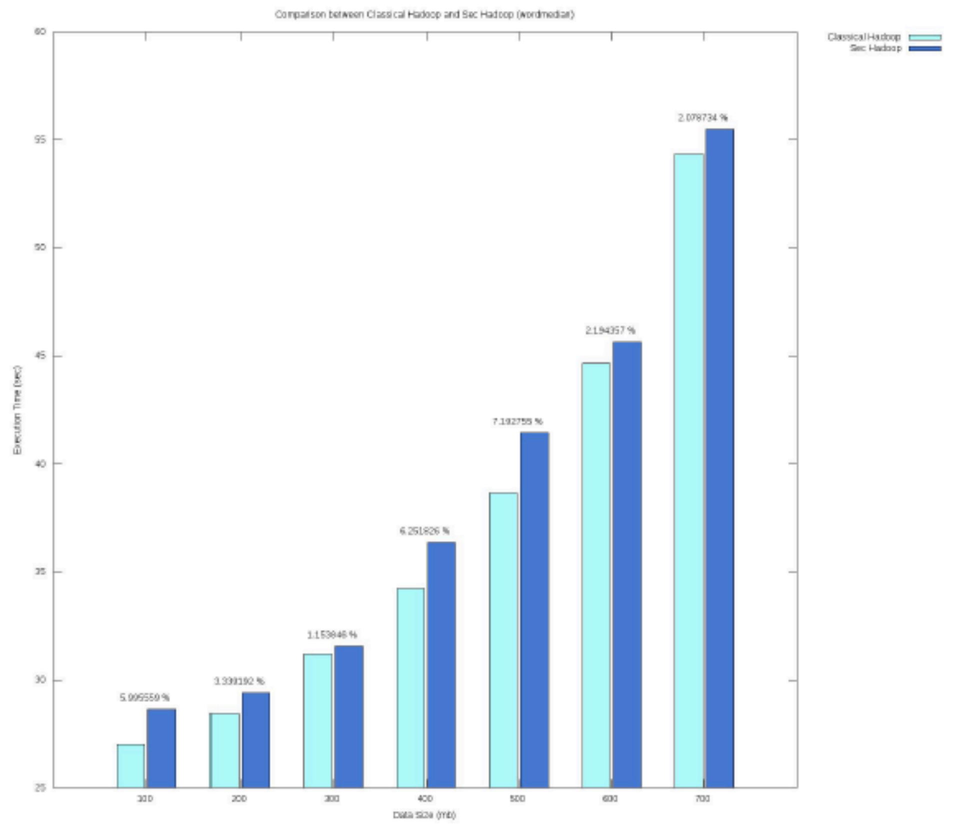
Average performance overhead : 2.62%

Performance Comparison for Grep Job



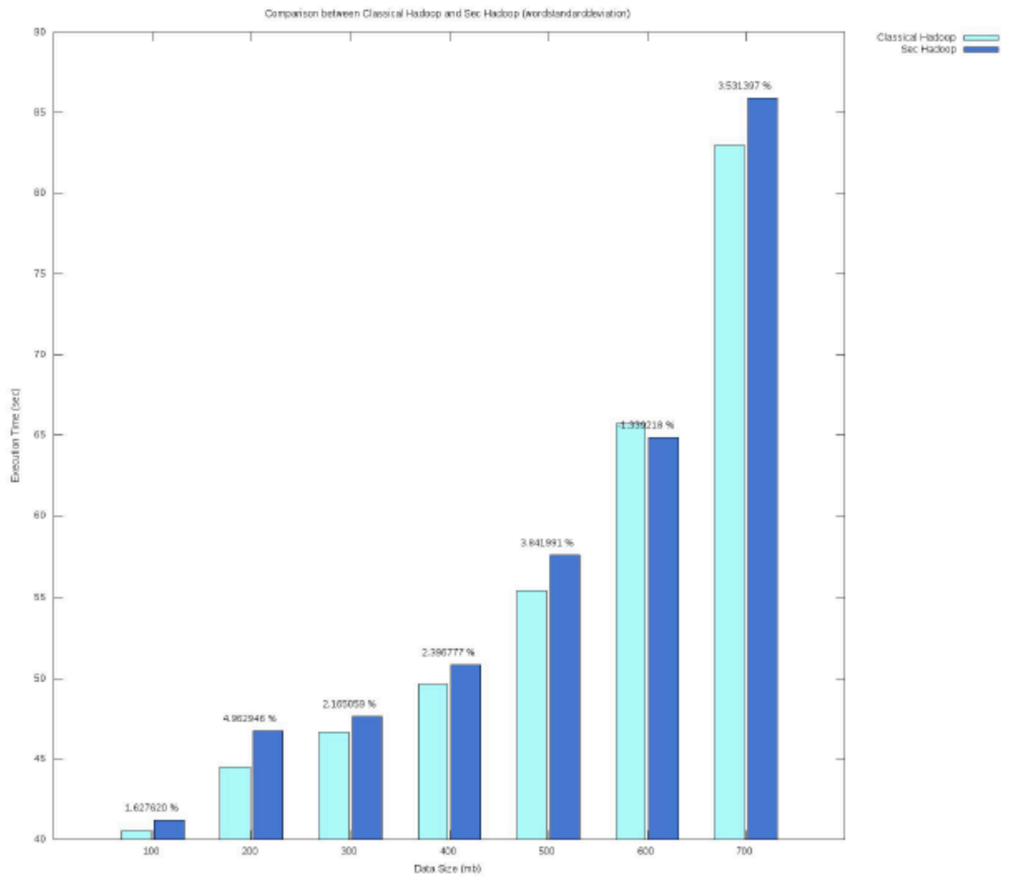
Average performance overhead : 1.55%

Performance Comparison for WordMedian Job



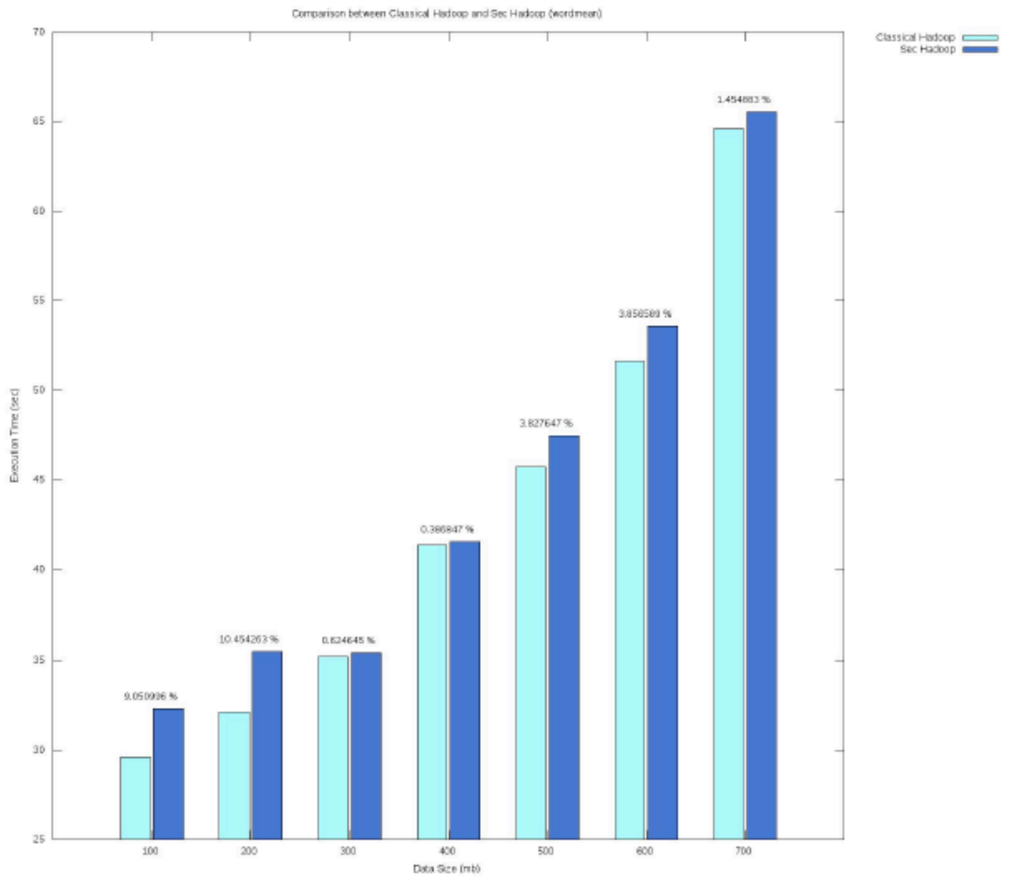
Average performance overhead : 4.02%

Performance Comparison for Word Standard deviation Job



Average performance overhead : 2.41%

Performance Comparison for WordMean Job



Average performance overhead : 4.23%

Comparison – Airavat*

- Airavat = SELinux (fixed set of syntactic labels) + "trusted" reducer + diff. privacy (add noise) -
- "reduce" provided by the user – difficult to trust
- SELinux $\neq$ full power of DIFC
- Differential policy will be difficult for dynamic evolving data
- RWFM = crisp combination of MAC (IFC) + DAC
  - Fine-grained labels preserve the privacy including that of the intermediate results without trust assumptions

I. Roy, S. T. V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel. **Airavat: Security and privacy for mapreduce**. In *7th USENIX NSDI, 2010*, pages 297–312.

Comparison – MLS MapReduce*

- MLS MapReduce = SELinux (fixed set of syntactic labels) + different HDFS name nodes (appropriately linked) for different labels
- Rigid data storage structure, inefficient solution
- RWFM labels more fine-grained – new lattice points generated as appropriate, particularly useful when combining information at different security levels

T. D. Nguyen, M. A. Gondree, J. Khosalim, and C. E. Irvine. **Towards a cross-domain mapreduce framework**. In IEEE *MILCOM, 2013,* pages 1436–1441.

Differential Privacy

- No need to use Differential Privacy which depends on noise introduction (hence difficult for evolving data) and other issues of privacy violation

Ease of Use

- Labels of initial objects (data) need to be provided for specifying the security and privacy requirements
- Zero-changes to the programming model – jar files for map and reduce
- Zero-overhead in terms of system usage – job submission and configurations
- Negligible performance overhead

Summary

- Preserves Privacy end-to-end
- Applicable for merging databases (for desensitization)
- Very Little Overhead
- Avoids “noises” required in differential privacy and also applicable for dynamic data

Ongoing work for Medical Data Sharing

Medical wisdom: Realized through a large number of experiments by a multiple parties. In the creation of such datasets, two properties are vital:

1. Privacy: very important as the medical information of the patient needs to be kept private by the individual and can be used for the purpose treatment and possible to gather data (or warnings) for the community.
2. provenance. important for re-constructing intermediate results or new experiments from intermediate ones and ownerships (IPRs)
3. For time, we need to integrated Attribute access control as well.

Thank You

Orange Book Standard

- Trusted Computer System Evaluation Criteria universally known as “the Orange Book”.
- B1 – Labeled Security Protection: the system must implement the Mandatory Access Control in which every subject and object of the system must maintain a security label, and every access to system resource (objects) by a subject must check for security labels and follow some defined rules.