Towards Scalable Cluster Auditing through Grammatical Inference over Provenance Graphs

Wajih Ul Hassan, Mark Lemay, Nuraini Aguse,
Adam Bates, Thomas Moyer

NDSS Symposium 2018
Feb 20, 2018

ILLINOIS

BOSTON UNIVERSITY

UNC CHARLOTTE

Notable Data Breach in 2017

Forbes

Security / #CyberSecurity

SEP 7, 2017 @ 10:42 PM 41,538

Equifax Data Breach Impacts 143 Million Americans

EQUIFAX

Black Duck Blog

Equifax, Apache Struts, & CVE-2017-5638 Vulnerability

Written by Fred Bals | Senior Content Writer/Editor | Sep 15, 2017

Equifax Inc.
@Equifax

Follow

We recently discovered a cybersecurity incident involving consumer information. Once discovered, we acted immediately to stop the intrusion.

6:20 PM - 7 Sep 2017

Notable Data Breach in 2017

Forbes

Security / #C

SEP 7, 2017 @

Equifa

Ameri

Equifax Data Breach Timeline 2017

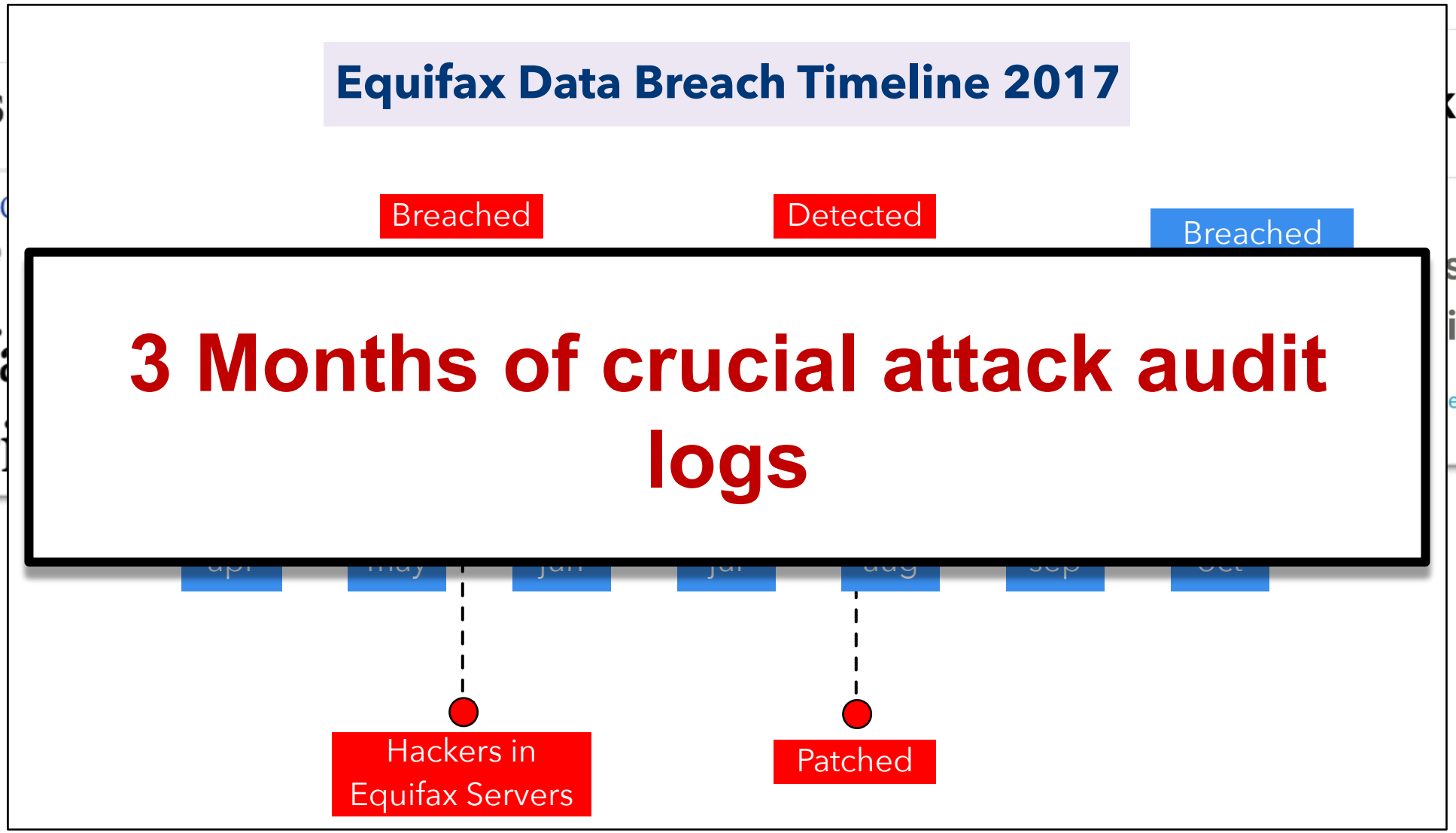
Blog

s, & CVE-

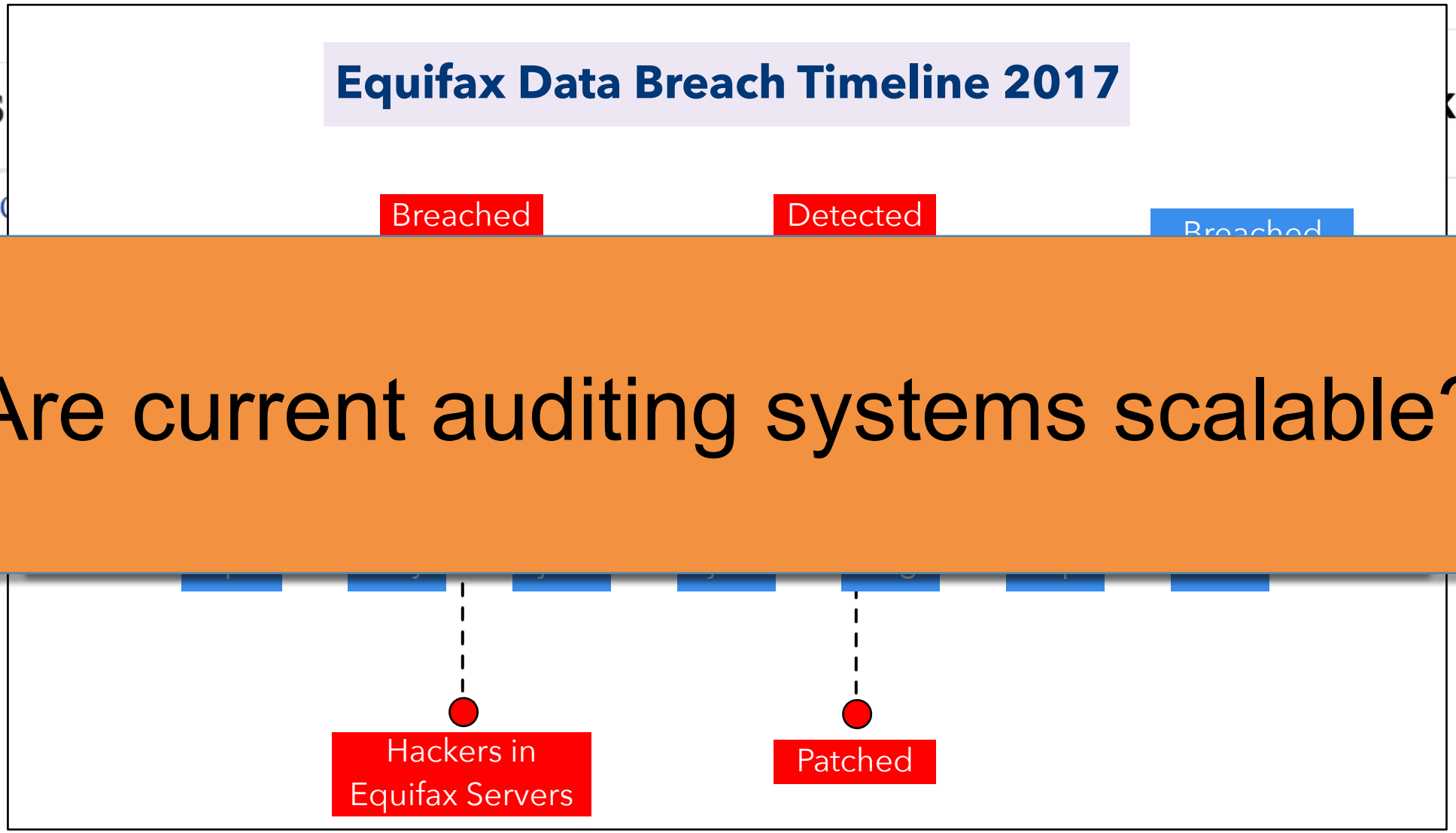
ity

er/Editor | Sep 15,

Notable Data Breach in 2017



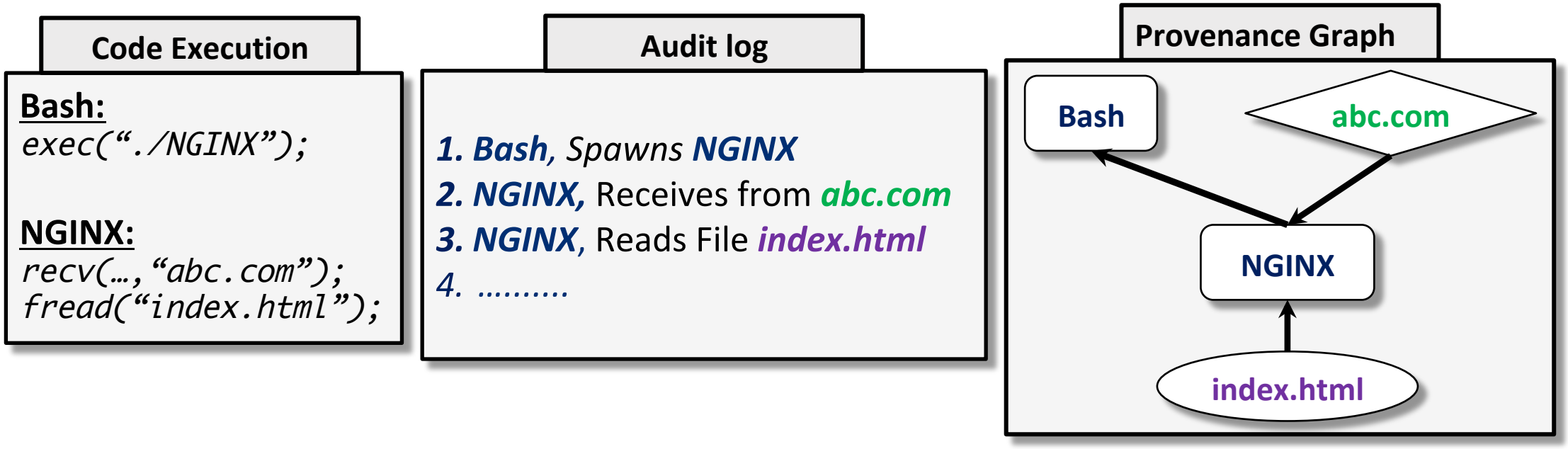
Notable Data Breach in 2017



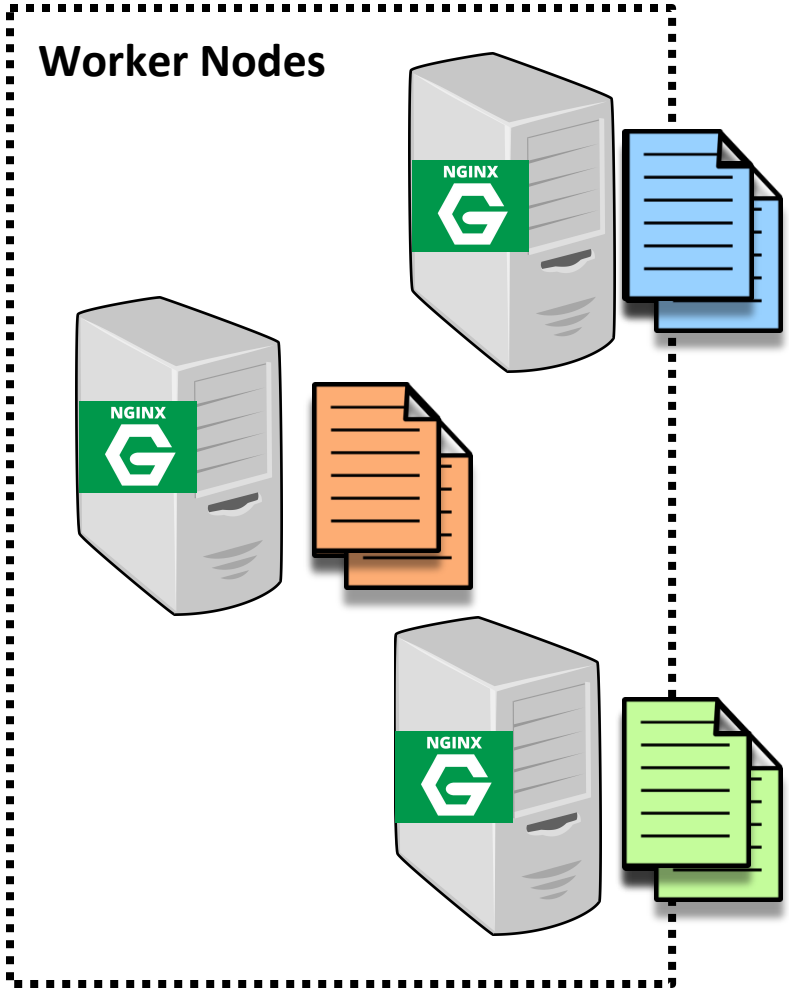
Are current auditing systems scalable?

Data Provenance aka Audit log

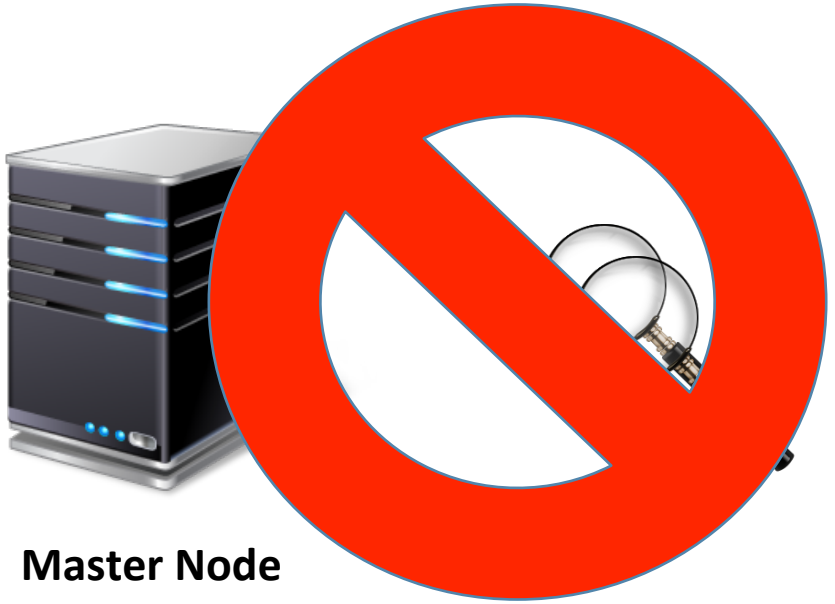
- Lineage of system activities
- Represented as Directed Acyclic Graph (DAG)
- Used for forensic analysis



Data Provenance in a Cluster



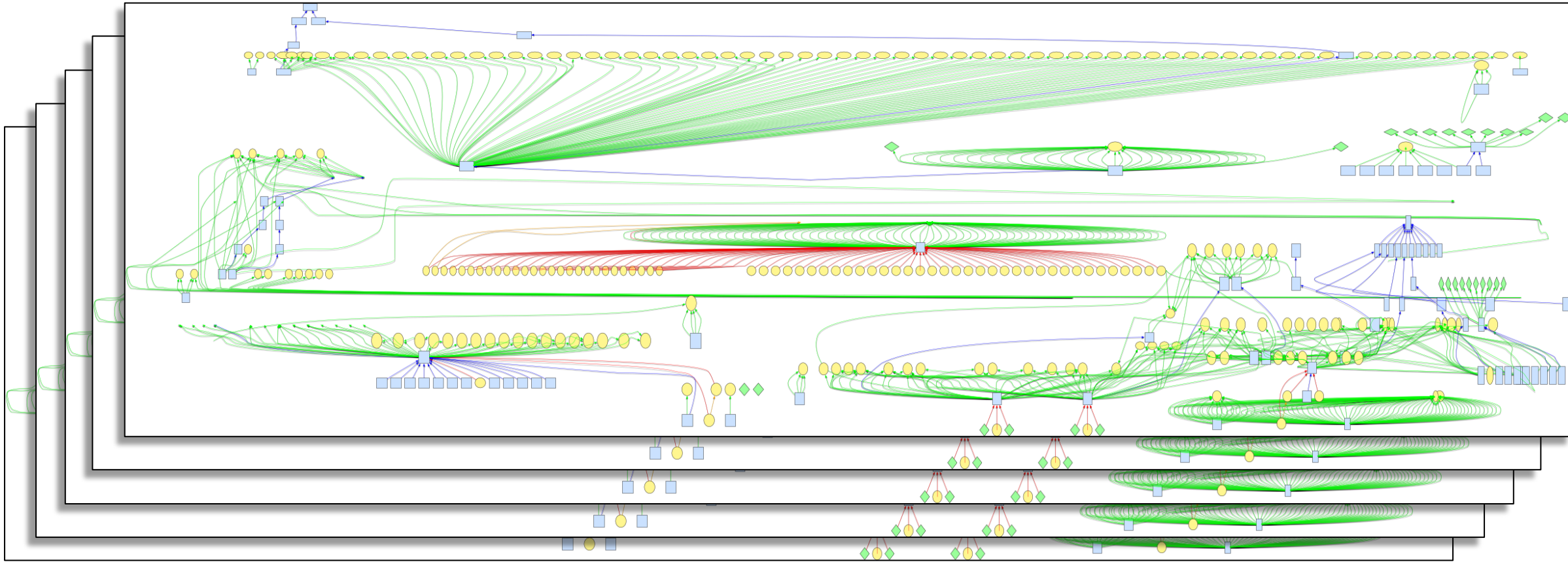
Centralized auditing not practical due to two limitations



Master Node

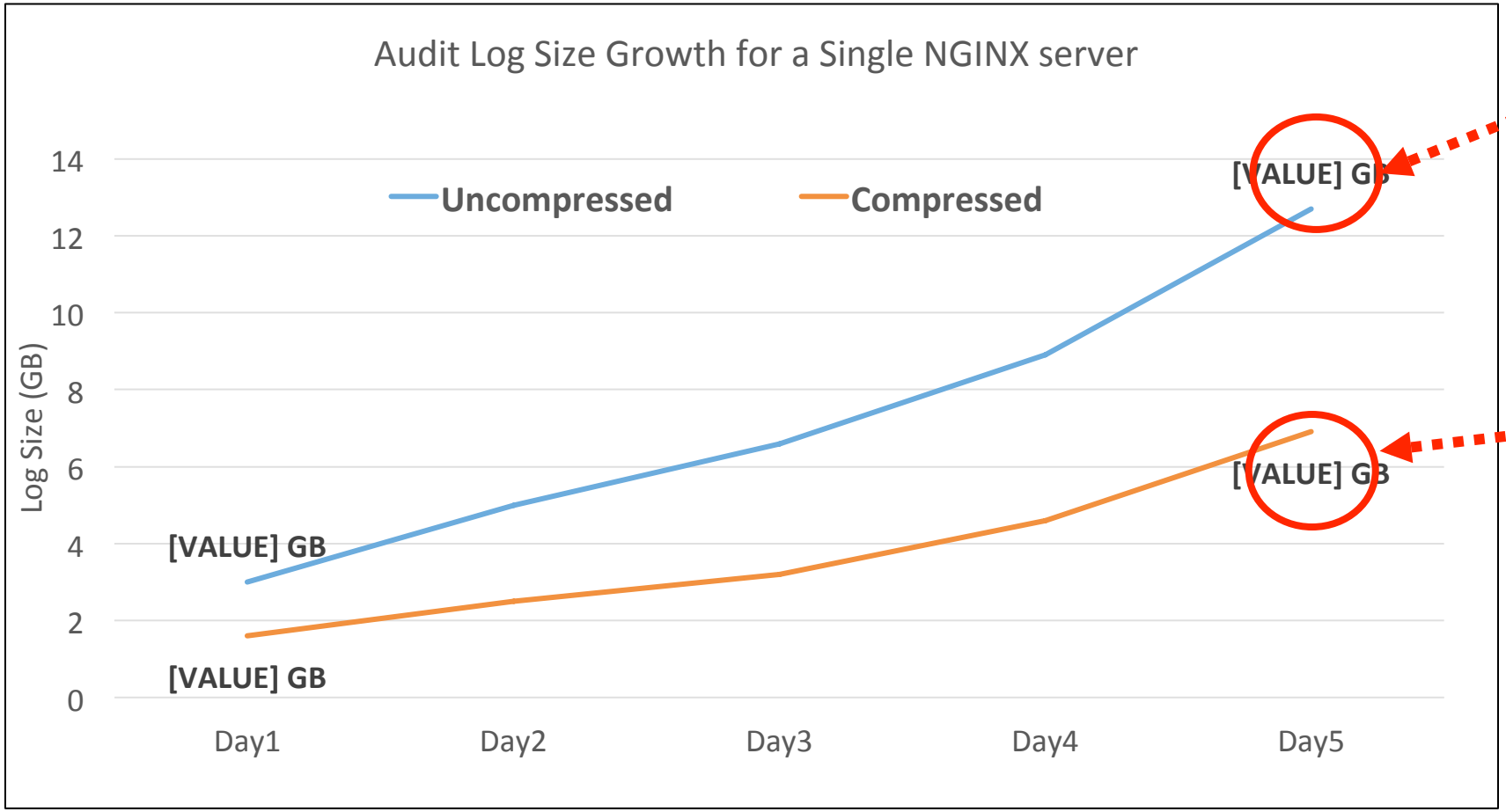
Limitation#1: Graph Complexity

- NGINX and MySQL running for 5 mins on a single machine



Limitation#2: Storage overhead

- Leads to network overhead as logs are transferred to master node

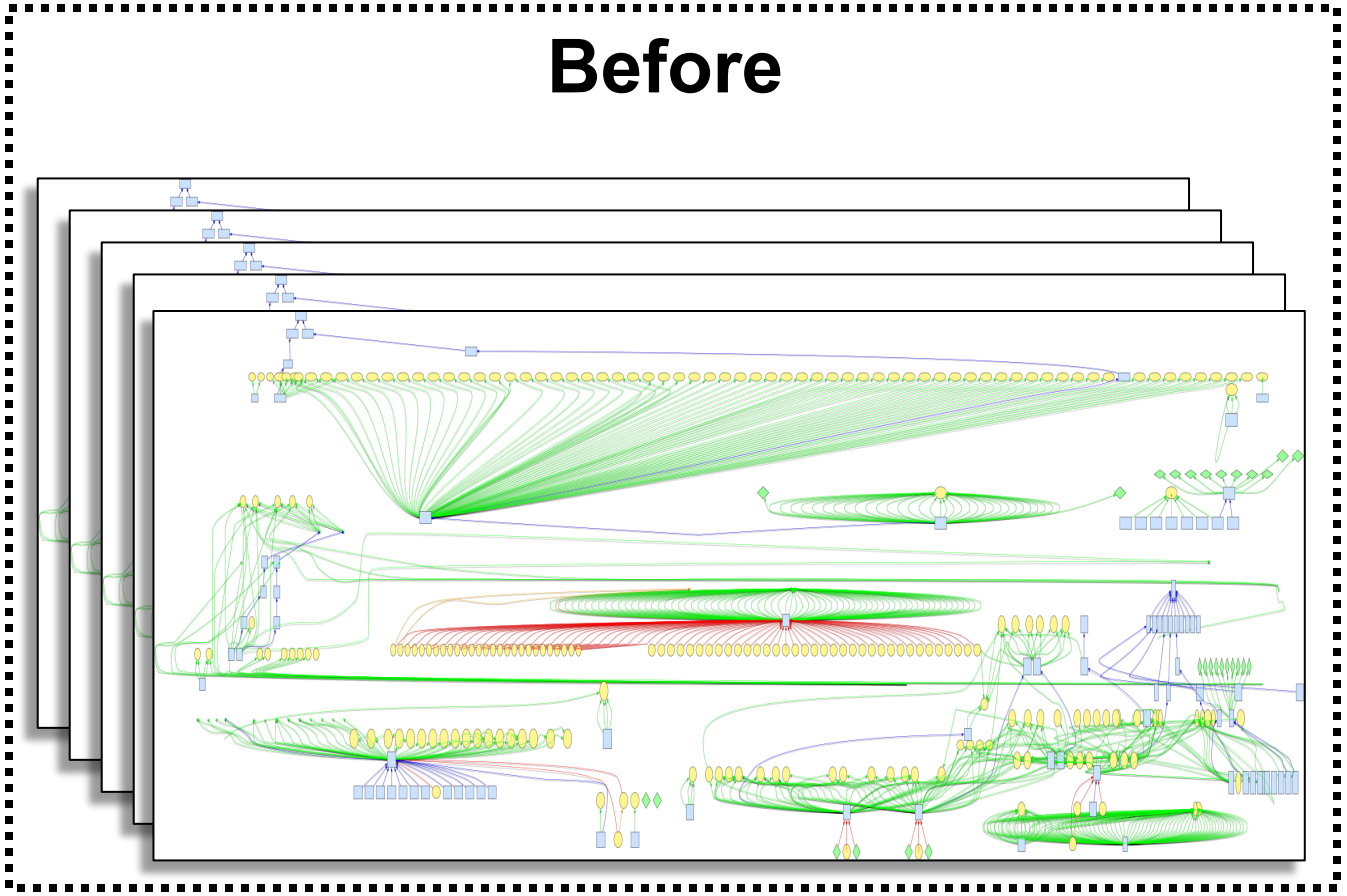


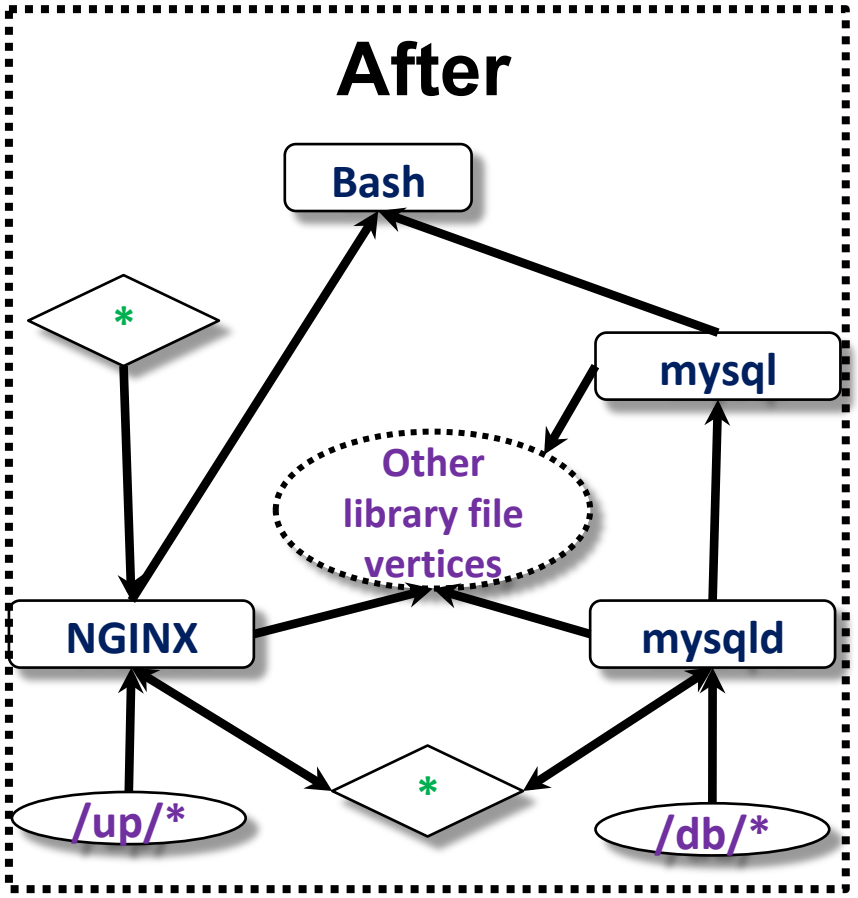
Winnower

- Cluster applications are replicated in accordance with microservice architecture principle
- Replicated apps produce highly homogeneous provenance graphs
  - core execution behaviour is similar

Key Idea:
Remove redundancy from provenance graphs across cluster before sending to **master node**

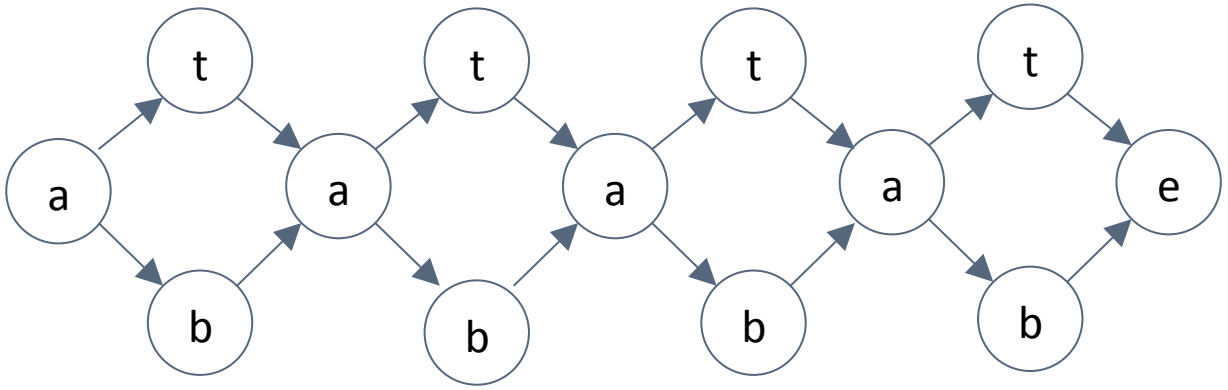
Master Node View with Winnower



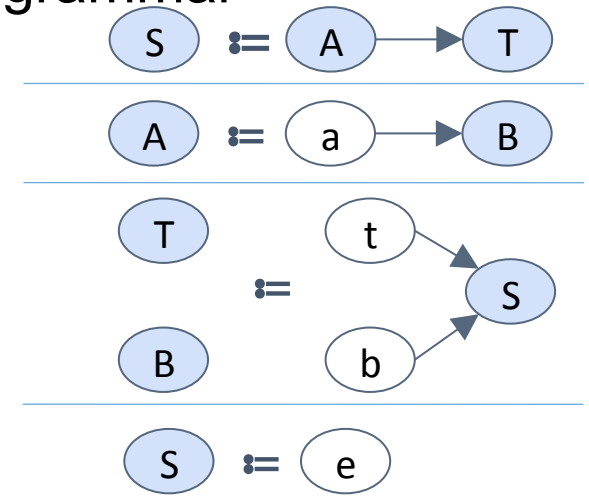


Winnower

- Build consensus model across cluster using graph grammars
- Like string grammar, graph grammars provide rule-based mechanisms
  - For generating, manipulating and analyzing graphs
  - *Induction* – produce grammar from a given graph
  - *Parsing* – membership test of a given graph is in a grammar

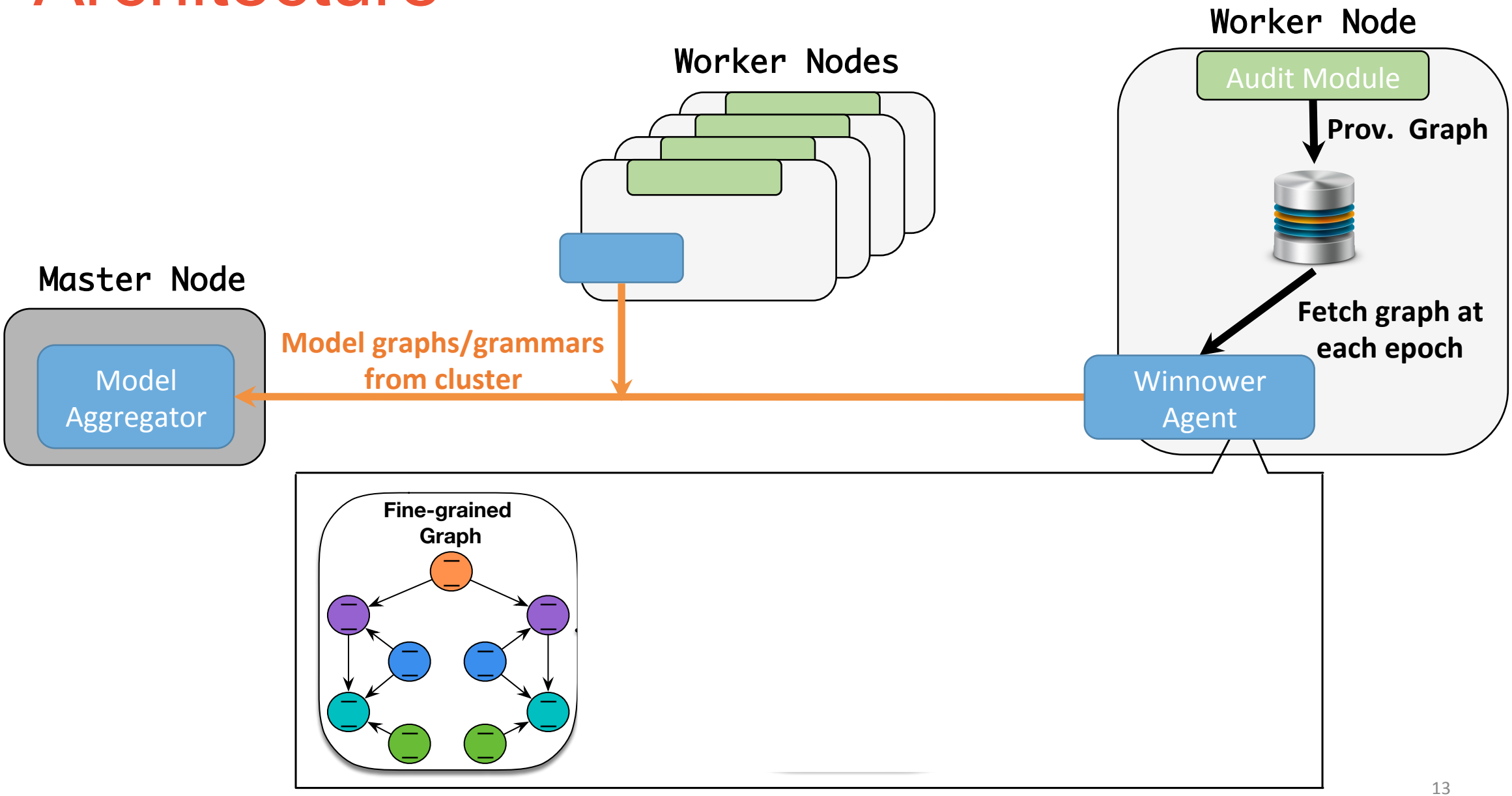


Graph

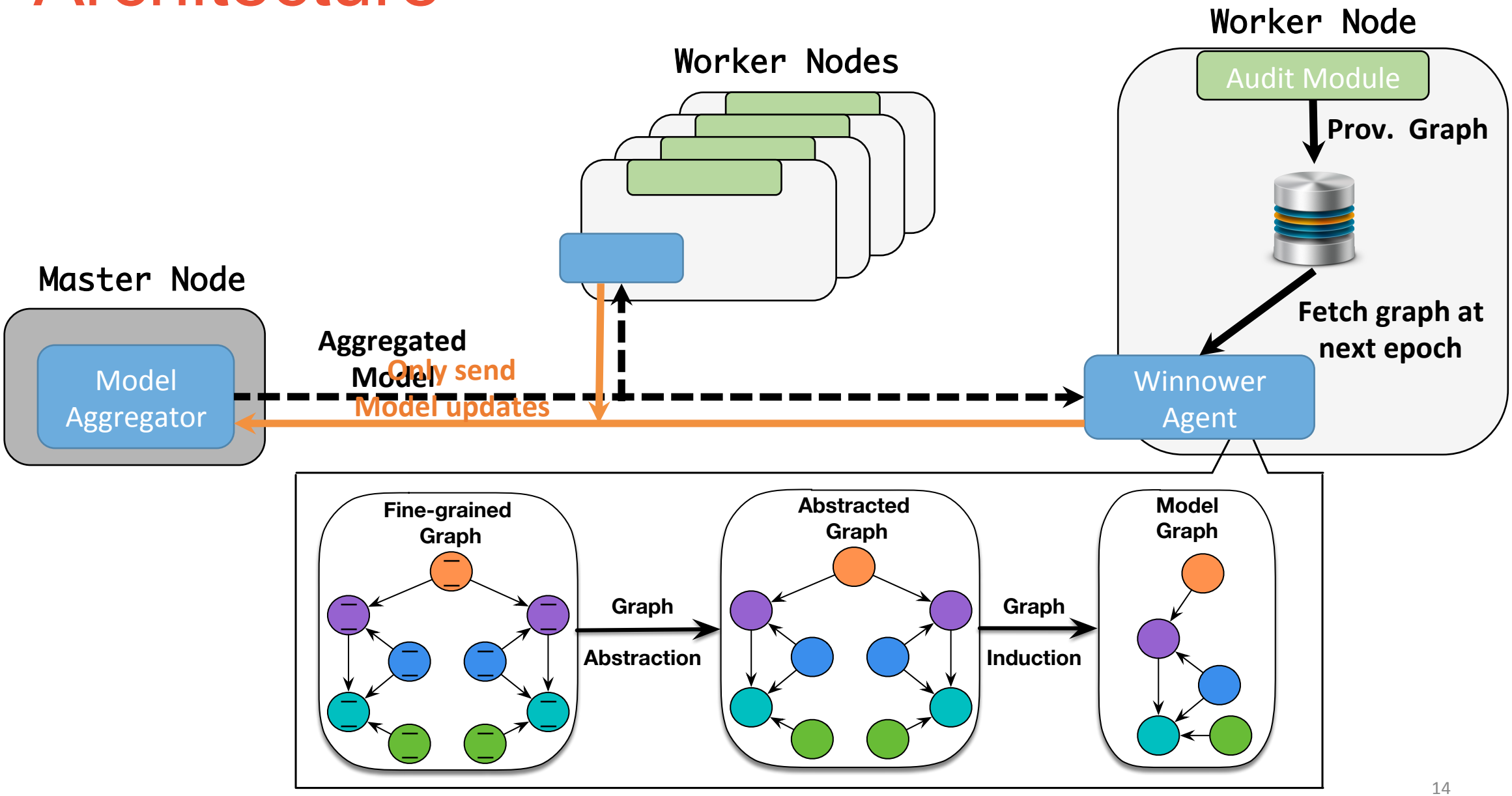


Graph Grammar

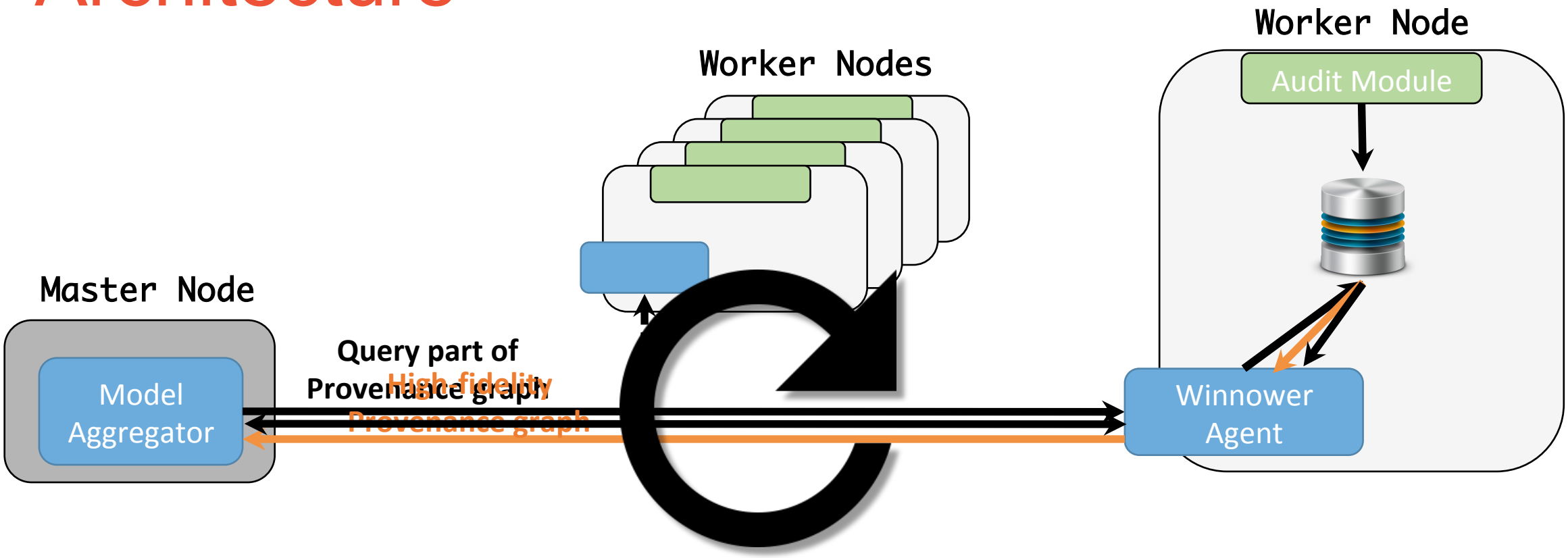
Architecture



Architecture

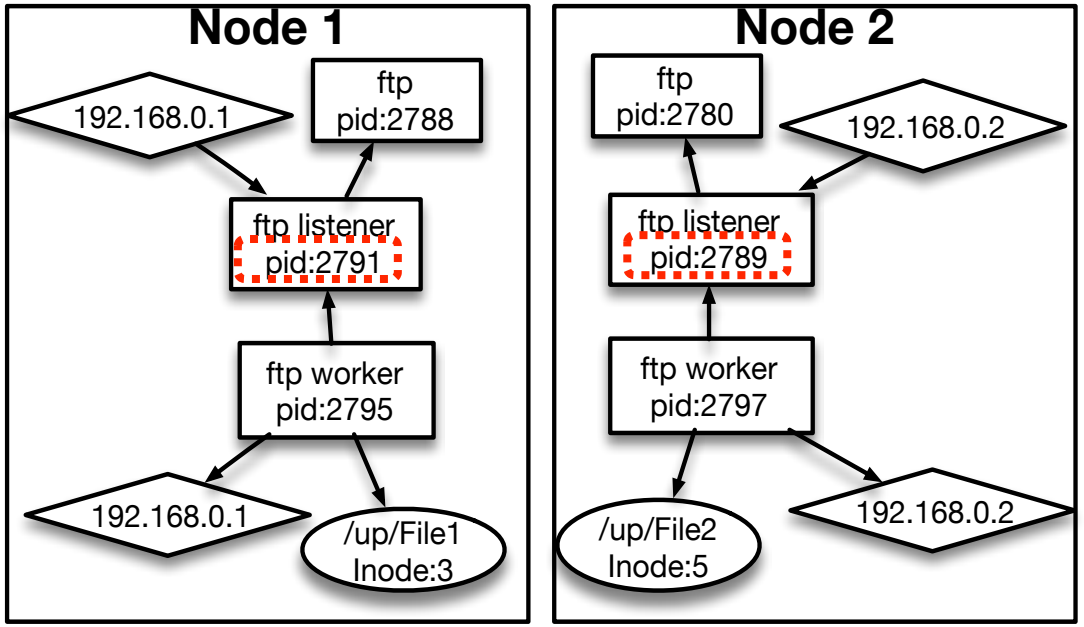


Architecture



Provenance Graph Abstraction

- Graph Induction process builds a model/grammar that concisely describe the whole graph
- However, instance-specific fields frustrate any attempts to build a generic application behaviour model

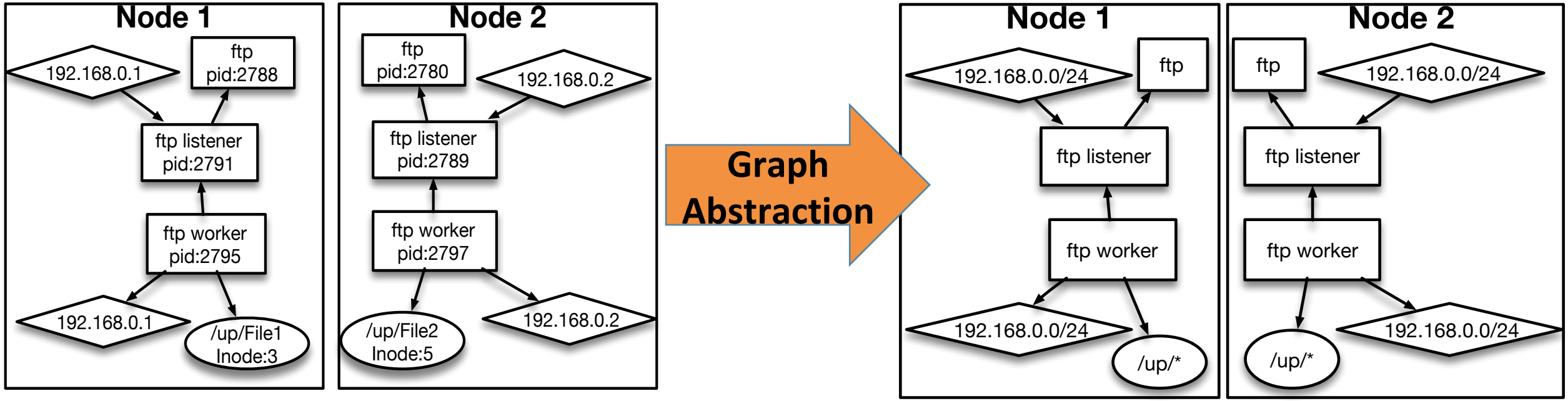


Graph Induction

No General model as instance specific information such PID is different among graphs

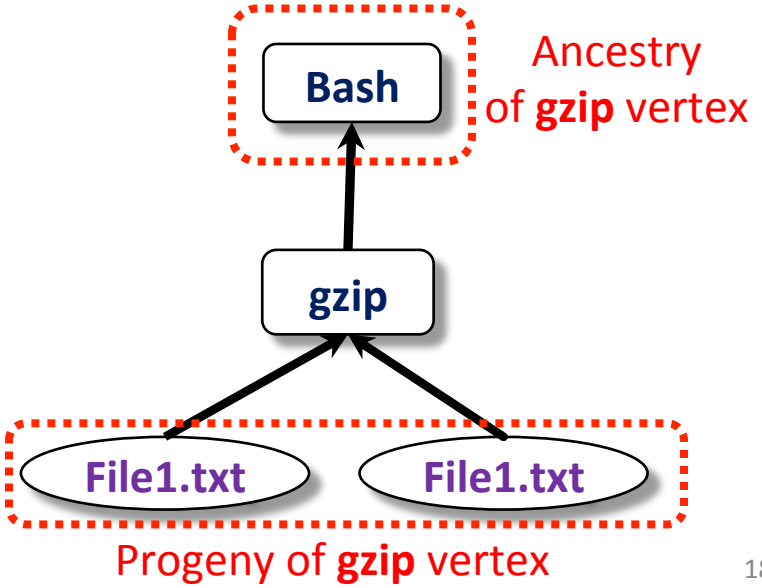
Provenance Graph Abstraction

- Provenance graph vertices have well defined fields
  - E.g. *pid:1234*, *FilePath:/etc/ld.so*
- Defined rules manually that remove or generalize these fields



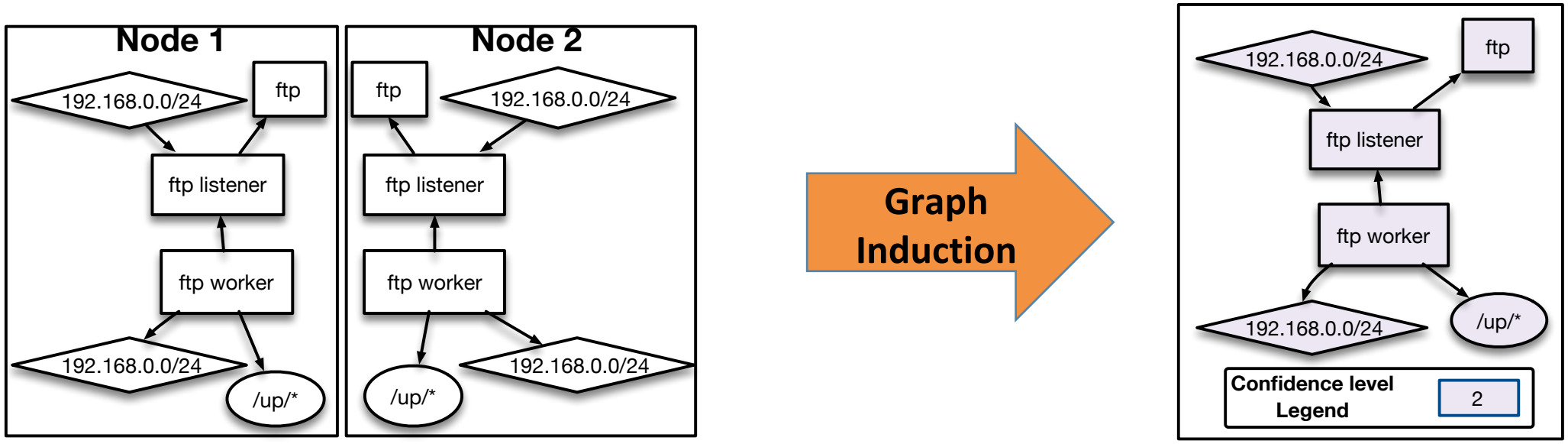
Provenance Graph Induction

- Deterministic Finite Automata (DFA) Learning to generate grammar
  - Encodes the causality in generated models
- In DFA learning the present state of a vertex includes the path taken to reach the vertex (provenance ancestry)
  - Winnower extends it to remember descendants (provenance progeny)
- State of each vertex consist of three items:
  1. Label
  2. Provenance ancestry
  3. Provenance progeny



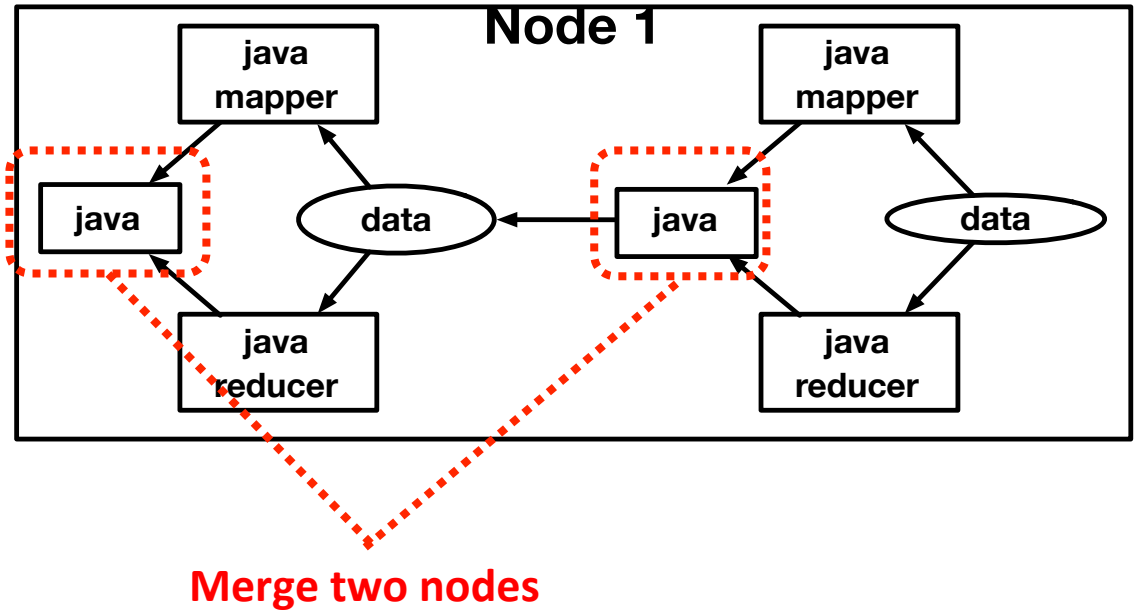
Provenance Graph Induction

- Finds repetitive patterns using standard implicit and explicit state merging algorithm
- Implicit state merging combines two subgraphs if states of each vertex are same in both subgraphs



Explicit State Merging

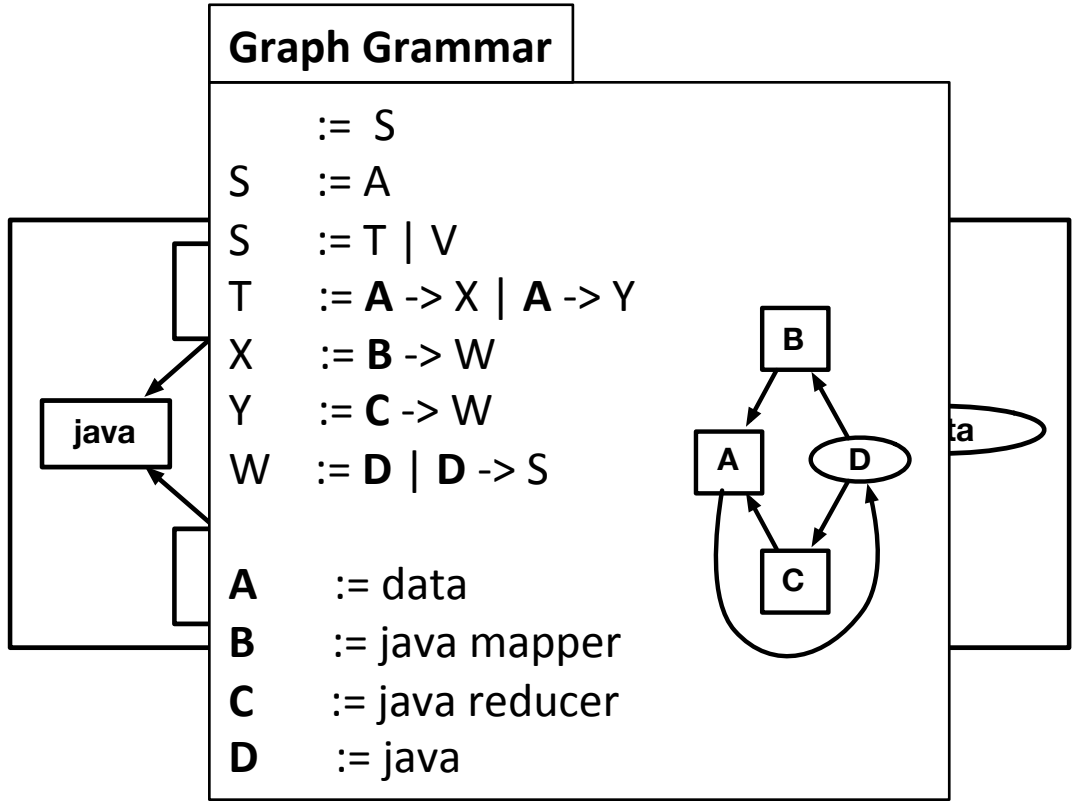
- At high-level explicit state merging
  - Picks two nodes and make their states same
  - Check if subgraph can be merged implicitly
- Consider a chained map reduce job



Graph Grammar

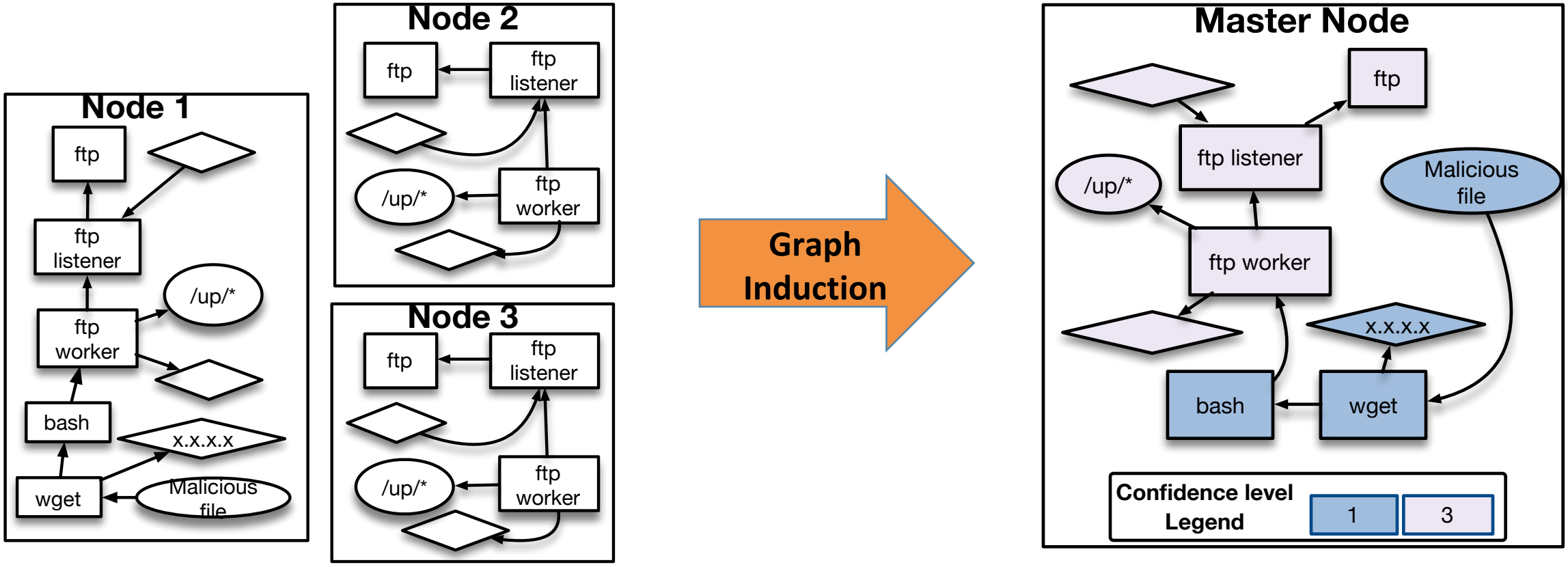
:= S
S := A
S := T | V
T := **A** -> X | **A** -> Y
X := **B** -> W
Y := **C** -> W
W := **D** | **D** -> S

A := data
B := java mapper
C := java reducer
D := java



Provenance Graph Induction

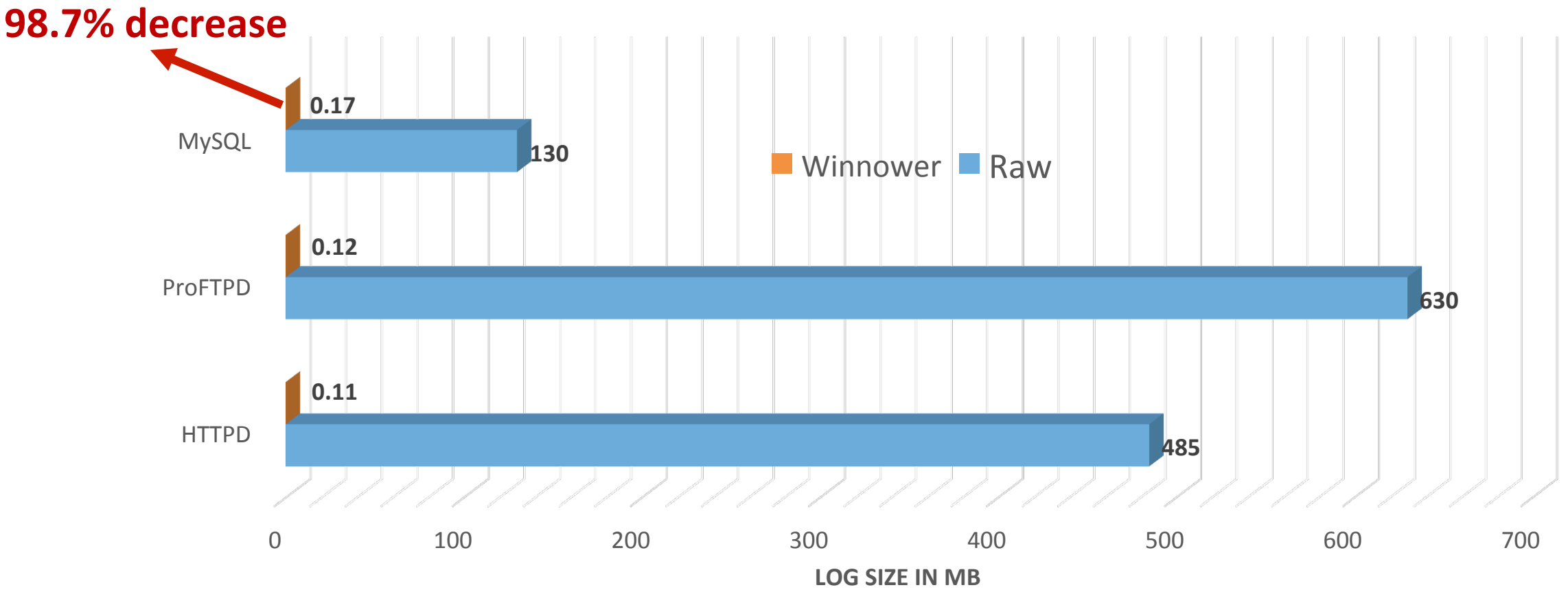
- Consider a graph with a malicious activity
- Malicious behavior is visible in the final model



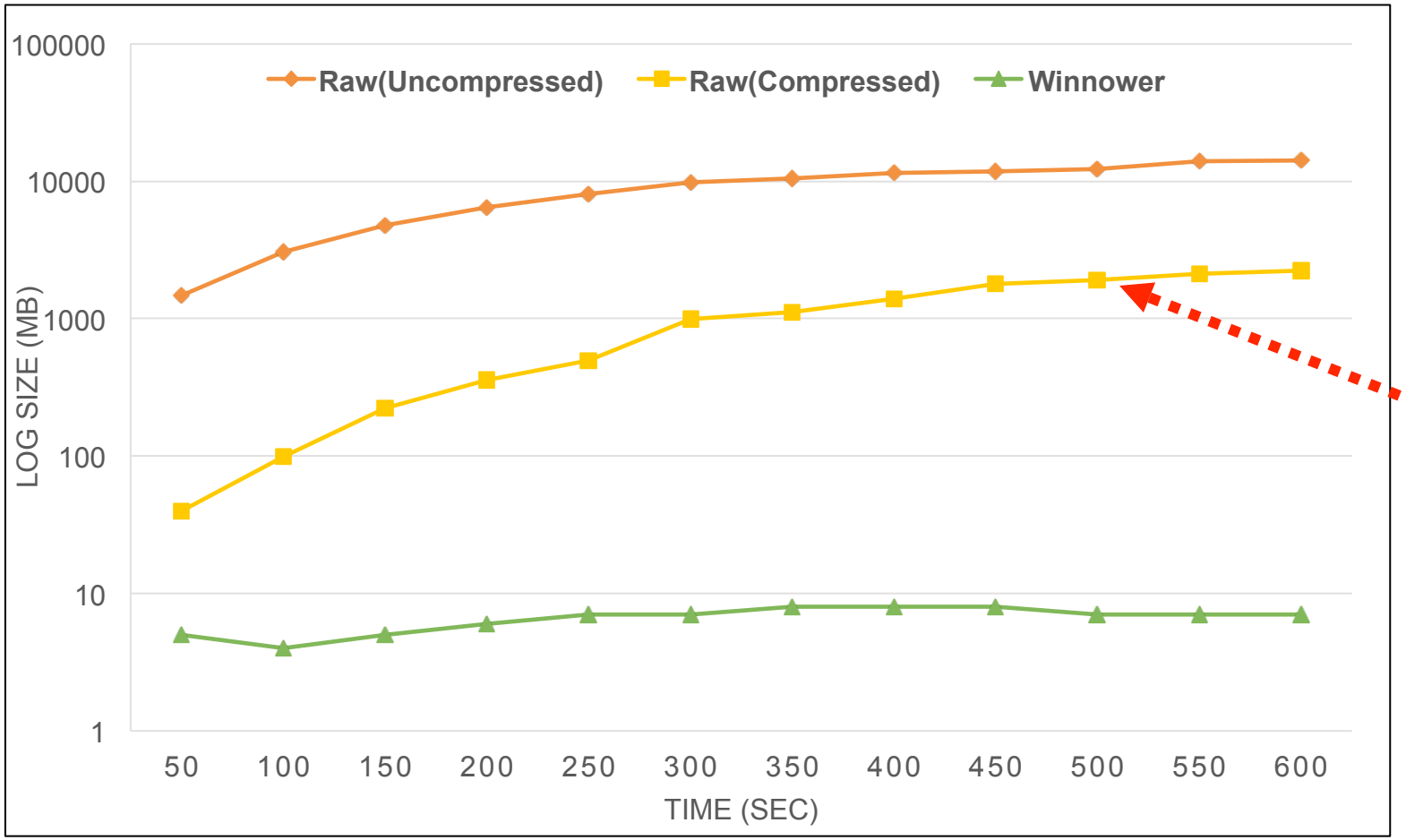
Evaluation Setup

- Setup
  - 1 VM as master node, 4 VMs as worker nodes
  - SPADE and Docker Swarm
  - Epoch size 50 sec
- Metrics
  - Storage Overhead
  - Computational Cost
  - Effectiveness

Storage Overhead on Master Node



Storage Reduction on Master Node

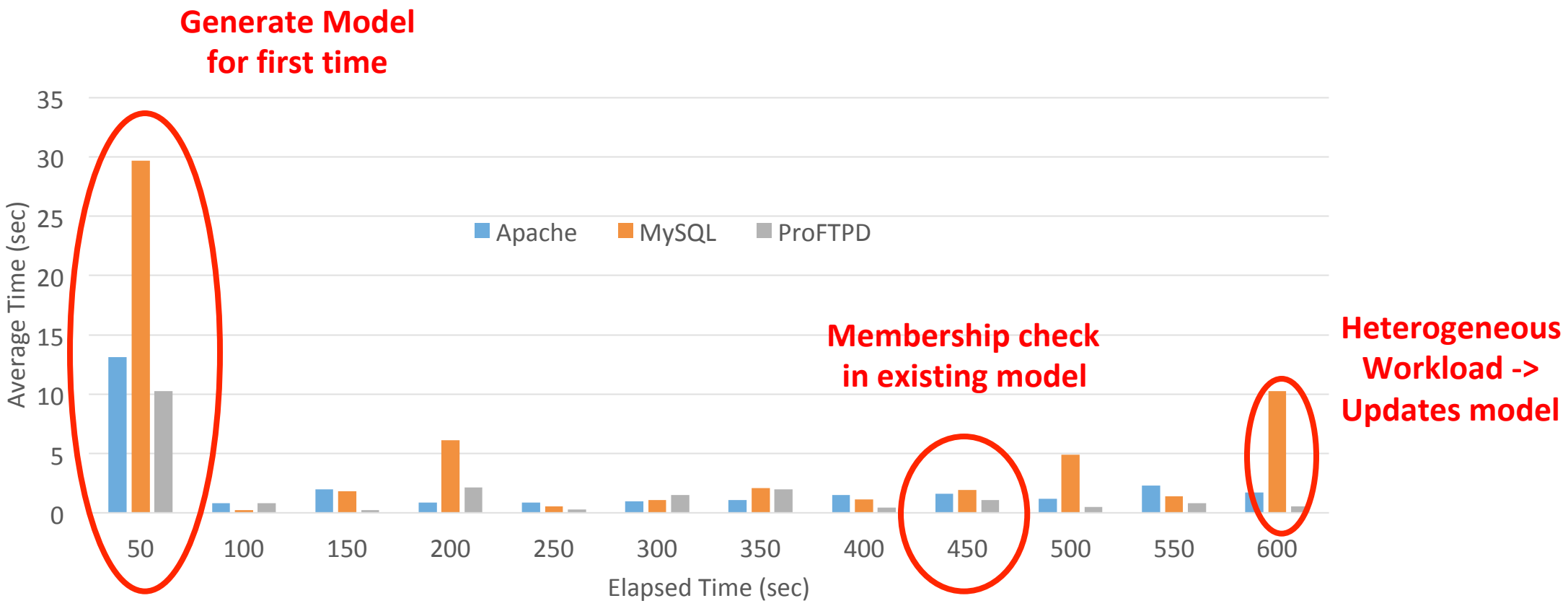


- Apache Webserver with moderate workload
- Note the log scale on y-axis

7z compression is not suitable:
- **No global view of cluster**
- **Oblivious to previous batch**

Evaluation: Computation Cost

- Average time spent in induction and membership test at each epoch

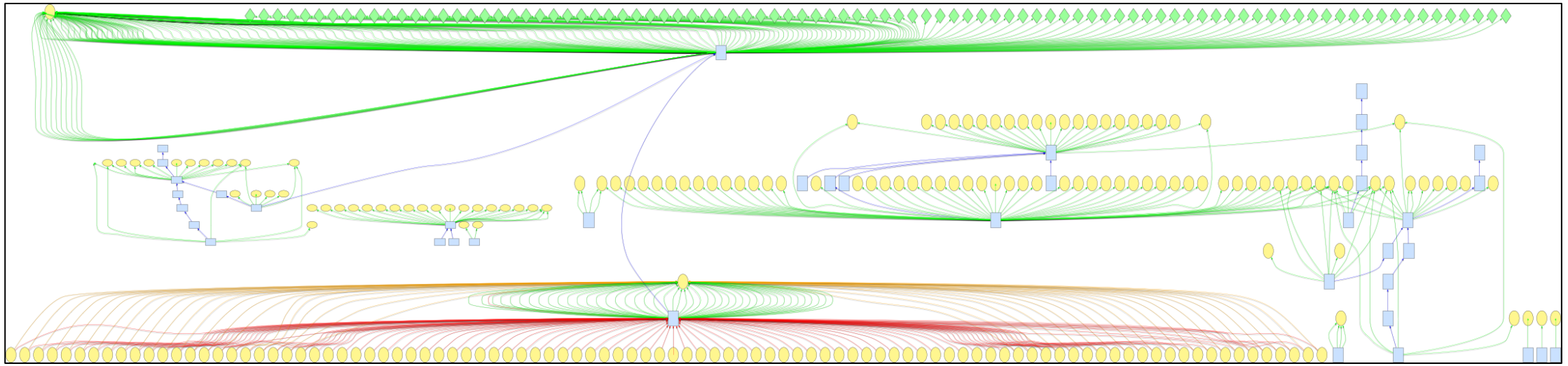


Case Study: Ransomware Attack

- Attacker exploits Redis database server vulnerability version < 3.2
- Vulnerability allows attacker to change SSH key and log in as Root
- Attacker deletes the database and left a note using vim to send bitcoins get database back

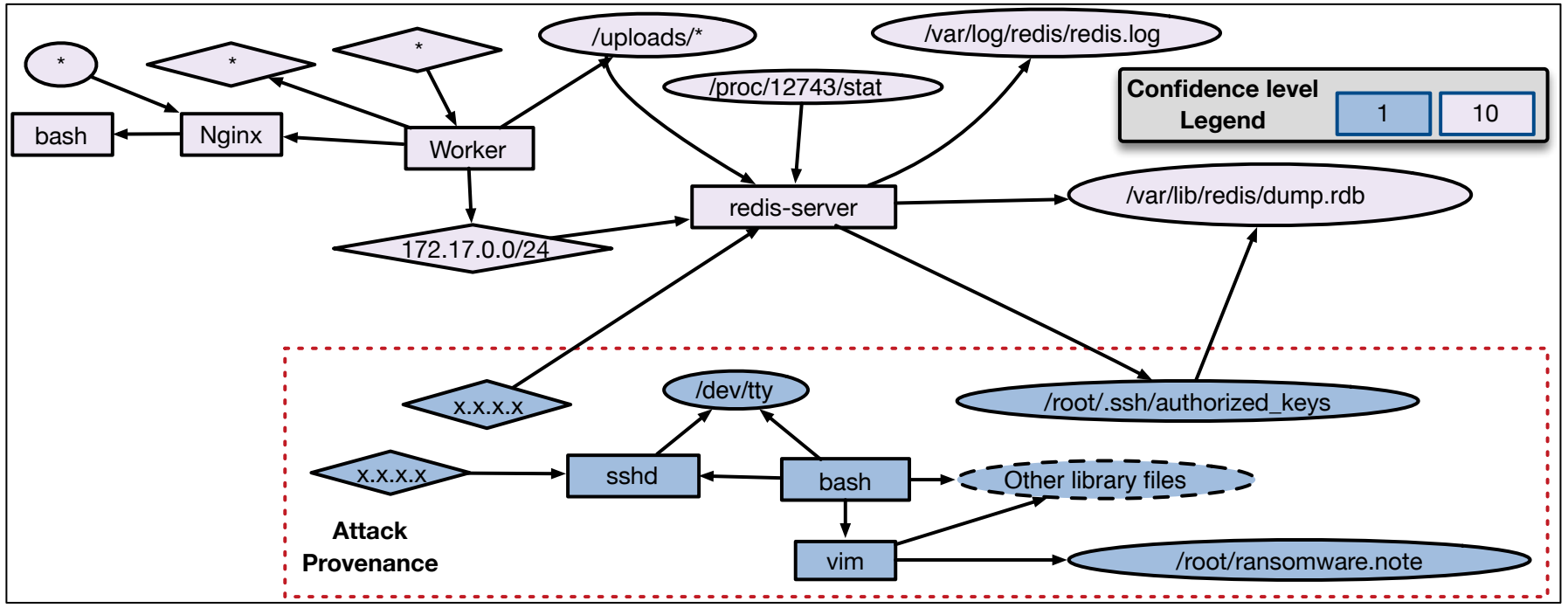
Traditional Graph of Attack

- 10 instances of redis running in the cluster
- ~80k vertices and ~83K edges with 161 MB size
- Part of provenance graph shown below



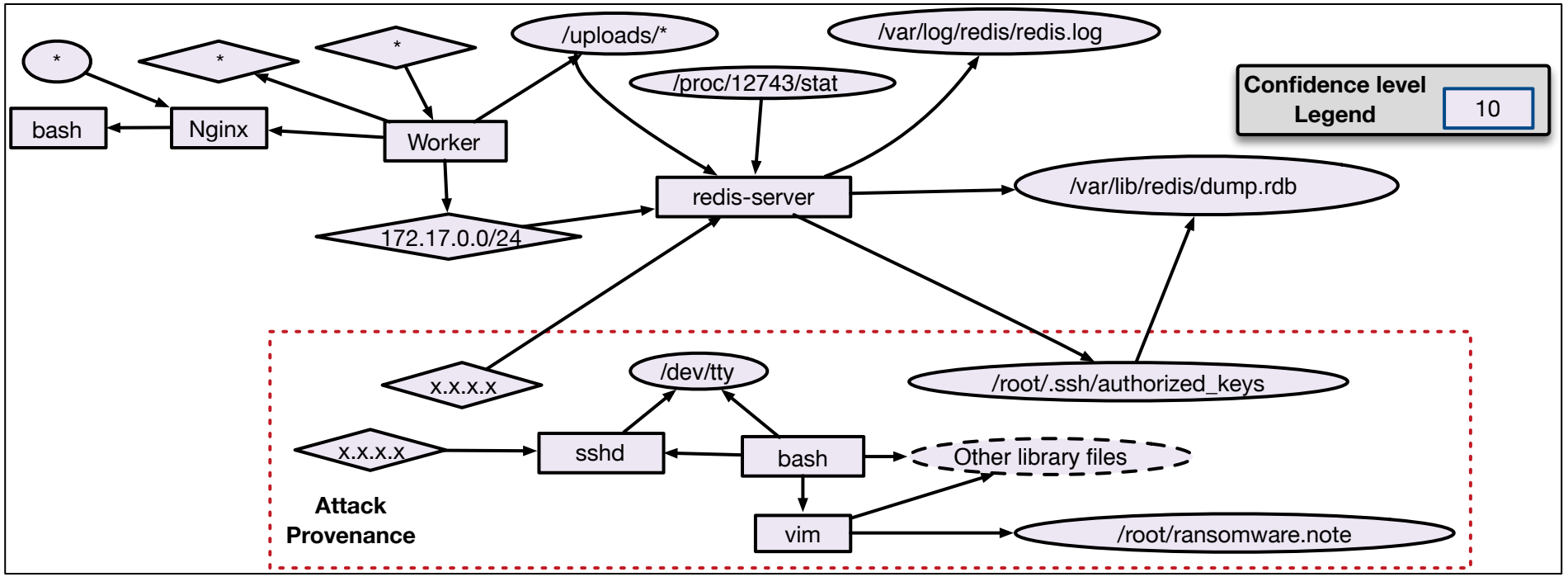
Winnower Generated Provenance graph

- 54 vertices and 68 edges with 0.7 MB size
- Part of graph is shown below:



Winnower Generated Provenance graph

- What happens if we attack all the nodes in the cluster



Conclusion

- Winnower is the first practical system for provenance-based auditing of clusters at scale with low overhead
- Winnower significantly improves attack identification and investigation in a large cluster

Questions

Thank you for your time.

whassan3@illinois.edu

Backup Slides

Threat model

- Assumptions
  - Winnower only tracks user-space attacks i.e. trusts the OS
  - Log integrity is maintained
- Attack surface
  - Distributed application replicated on Worker nodes
- Attacker’ motive
  - Gain control over worker node by exploiting a software vulnerability in the distributed application

Online Learning