Finding Heterogeneous-Unsafe Configuration Parameters in Cloud Systems

Sixiang Ma, Fang Zhou, Michael D. Bond, Yang Wang
The Ohio State University

EuroSys 2021

Results Reproduced acm V1.1

Artifacts Evaluated acm Functional V1.1

Artifacts Available acm V1.1

The Ohio State University
Department of
Computer Science and Engineering

Heterogeneous Configurations Are Prevalent

- Heterogeneous hardware calls for heterogeneous configuration

- Online reconfiguration, e.g., reconfig command, rolling restart
    - Consequence: short window of heterogeneous configuration

APACHE HBASE

hadoop YARN

APACHE kafka

hadoop HDFS

Apache Zookeeper

Heterogeneous Configuration Can Cause Error

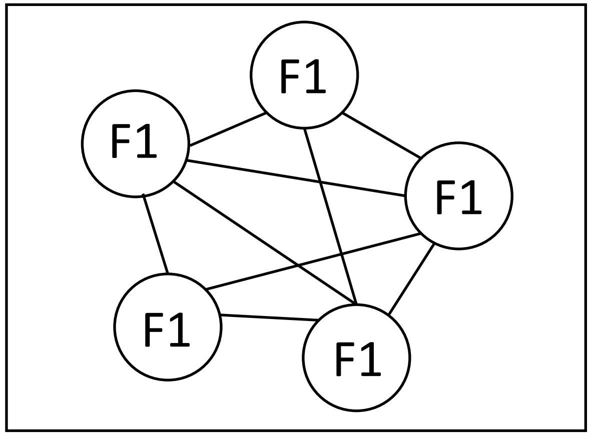
- Errors can happen even if each node has valid configuration locally.

F1

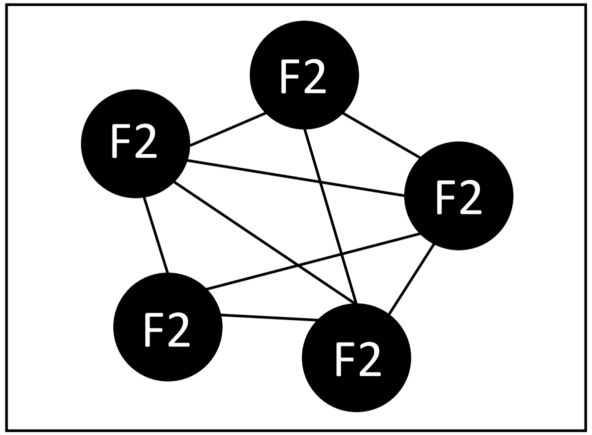
encrypt
=true
…

F2

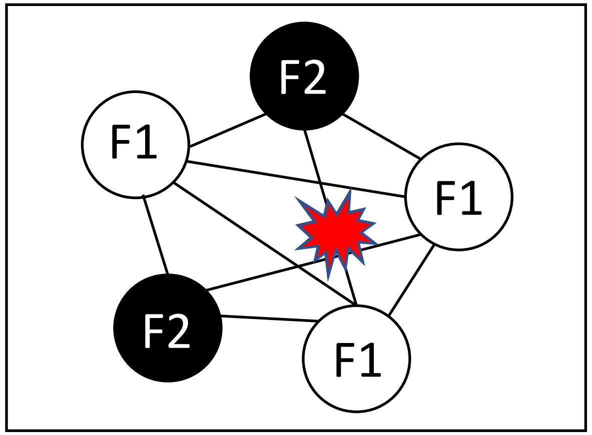
encrypt
=false
…



HomoConf(F1) is valid



HomoConf(F2) is valid



HeterConf(F1,F2) is invalid

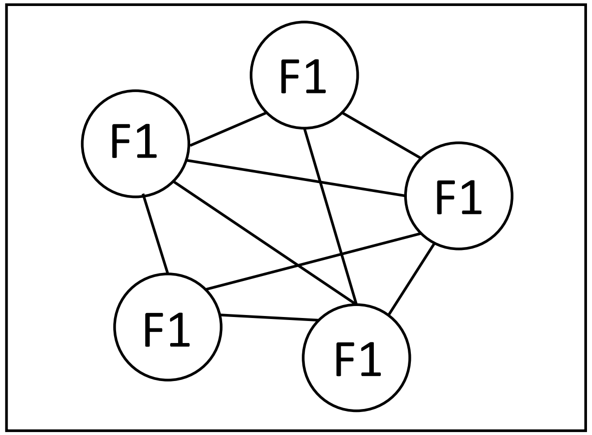
Heterogeneous Configuration Can Cause Error

- Errors
  valid c

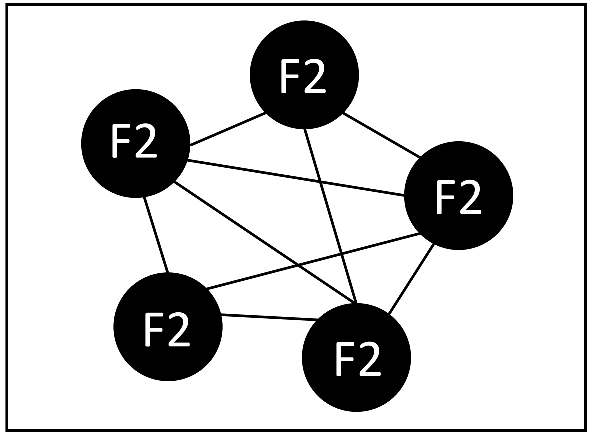
We call HeterConf(F1, F2) ***Invalid Heterogenous Configuration***, if it causes errors but HomoConf(F1) and HomoConf(F2) do not.

F2

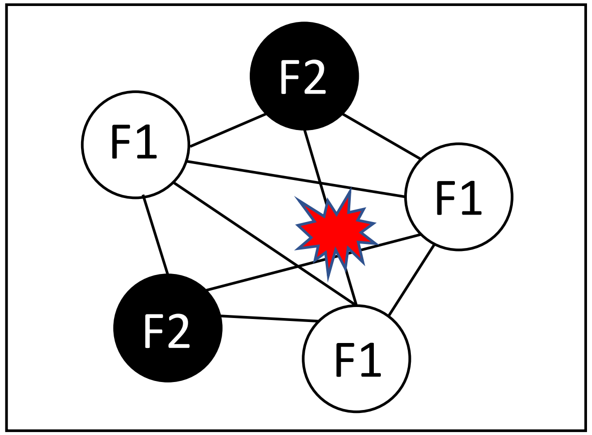
encrypt
=false
…



HomoConf(F1) is valid



HomoConf(F2) is valid



HeterConf(F1,F2) is invalid

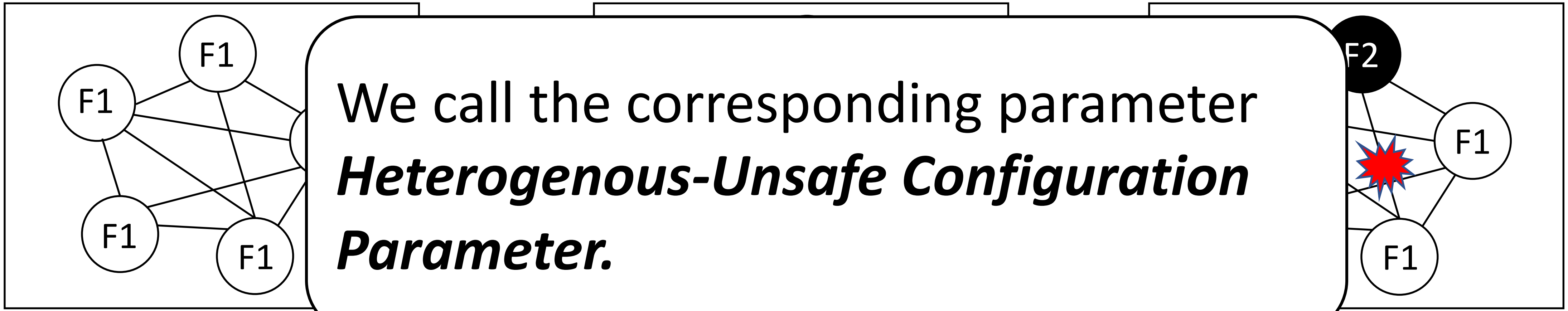
Heterogeneous Configuration Can Cause Error

- Errors
  valid c

We call HeterConf(F1, F2) ***Invalid Heterogenous Configuration***, if it causes errors but HomoConf(F1) and HomoConf(F2) do not.

F2

encrypt
=false
…



HomoConf(F1) is valid

HomoConf(F2) is valid

HeterConf(F1,F2) is invalid

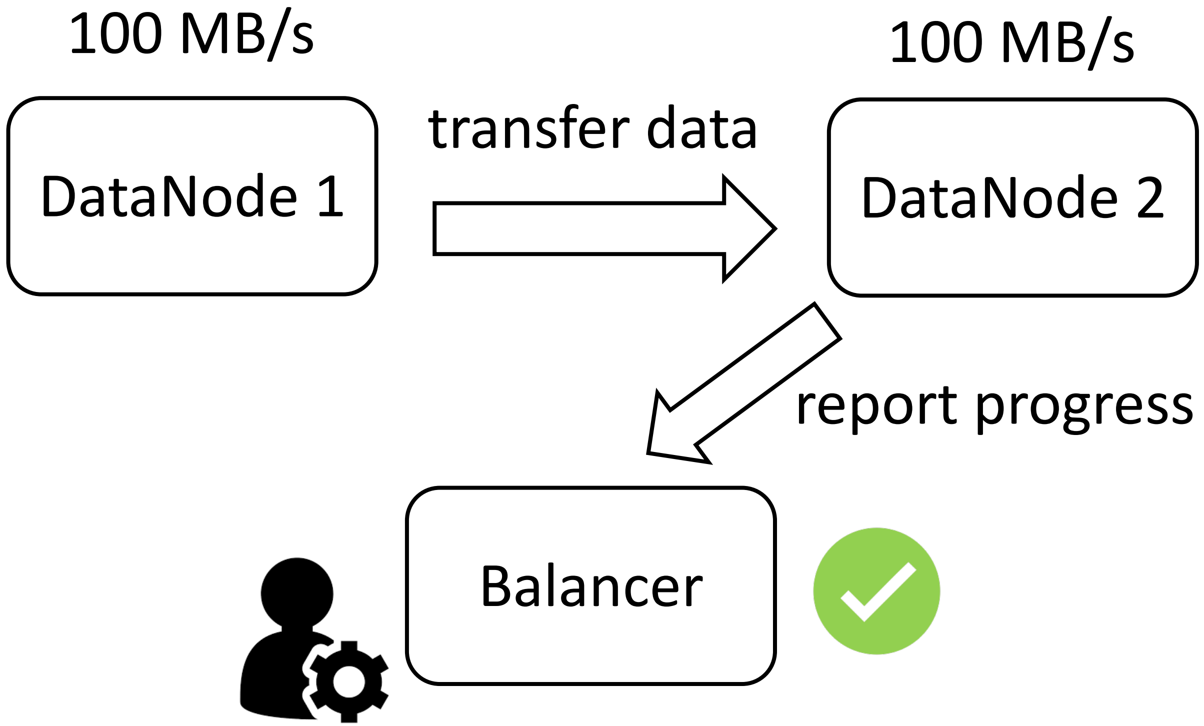
We call the corresponding parameter ***Heterogenous-Unsafe Configuration Parameter.***

Example: *dfs.datanode.balance.bandwidthPerSec*

- Specify the maximum amount of bandwidth that a HDFS DataNode can use for balancing purpose.

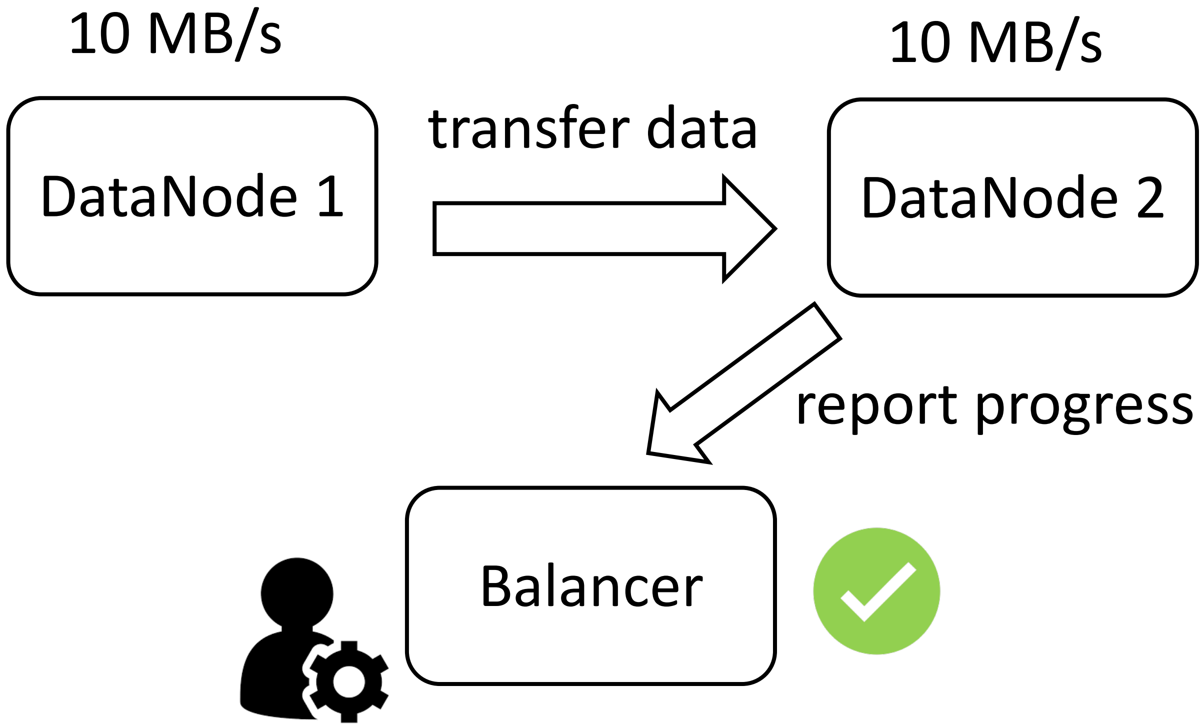
Example: *dfs.datanode.balance.bandwidthPerSec*

- Specify the maximum amount of bandwidth that a HDFS DataNode can use for balancing purpose.



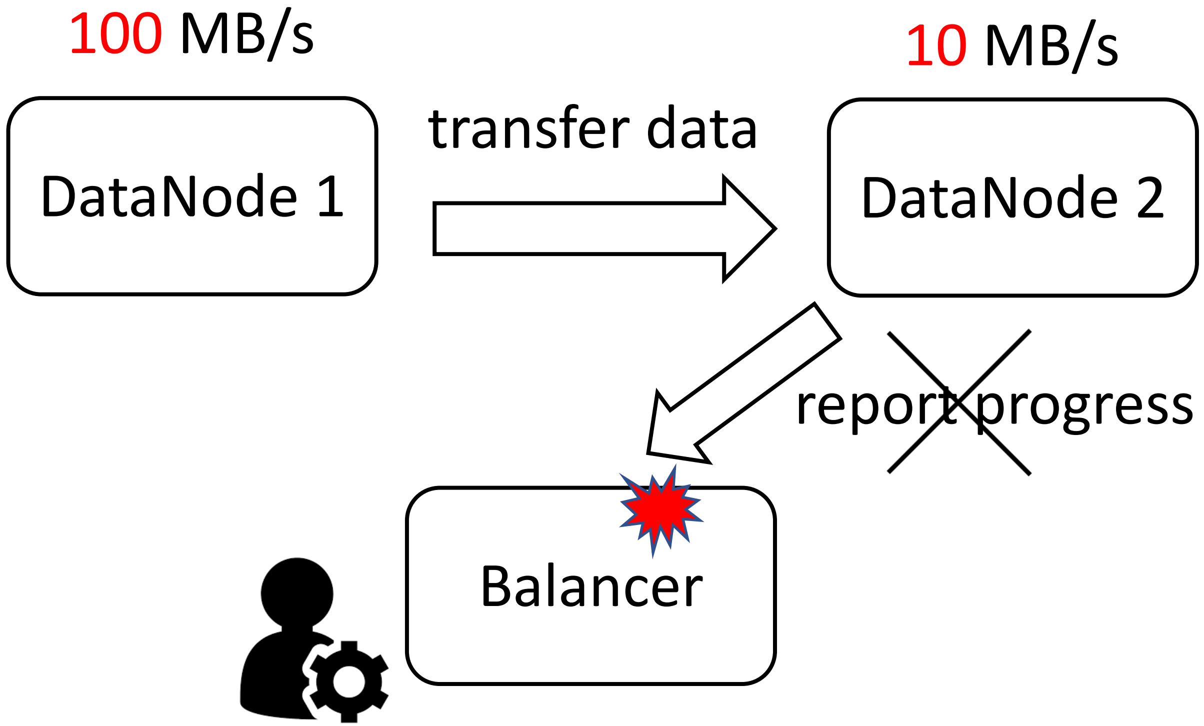
Example: *dfs.datanode.balance.bandwidthPerSec*

- Specify the maximum amount of bandwidth that a HDFS DataNode can use for balancing purpose.



Example: *dfs.datanode.balance.bandwidthPerSec*

- Specify the maximum amount of bandwidth that a HDFS DataNode can use for balancing purpose.



Related Work

This type of errors is different from the problem of erroneous configuration values [EnCore-ASPLOS’14, ConfValley-EuroSys’15, PCheck-OSDI’16, PracExtractor-ATC’20]

- Parameter values are valid.
- Errors happen when nodes communicate.

Overview

- Our goal: find heterogeneous-unsafe configuration parameters in cloud systems.

- ZebraConf: a testing framework that reuse existing unit tests

- It finds 41 true problems in HDFS, YARN, MR, HBase, Flink.

ZebraConf Uses Classic Software Testing Approach

- Challenge: some parameters may only take effect under specific workloads.

- Observation: mature cloud systems usually have rich unit tests.
  - High code coverage [Kairux-SOSP'19]
    - E.g., 90.1% statement coverage in HDFS
  - Many unit tests are using configuration
    - 3,628 unit tests in HDFS use config, covering 96.2% parameters

ZebraConf Uses Classic Software Testing Approach

- Challenge: some parameters may only take effect under specific workloads.

Reuse Existing Unit Tests for Our Purpose

    - E.g., 90.1% statement coverage in HDFS
  - Many unit tests are using configuration
    - 3,628 unit tests in HDFS use config, covering 96.2% parameters

ZebraConf: Major Challenges

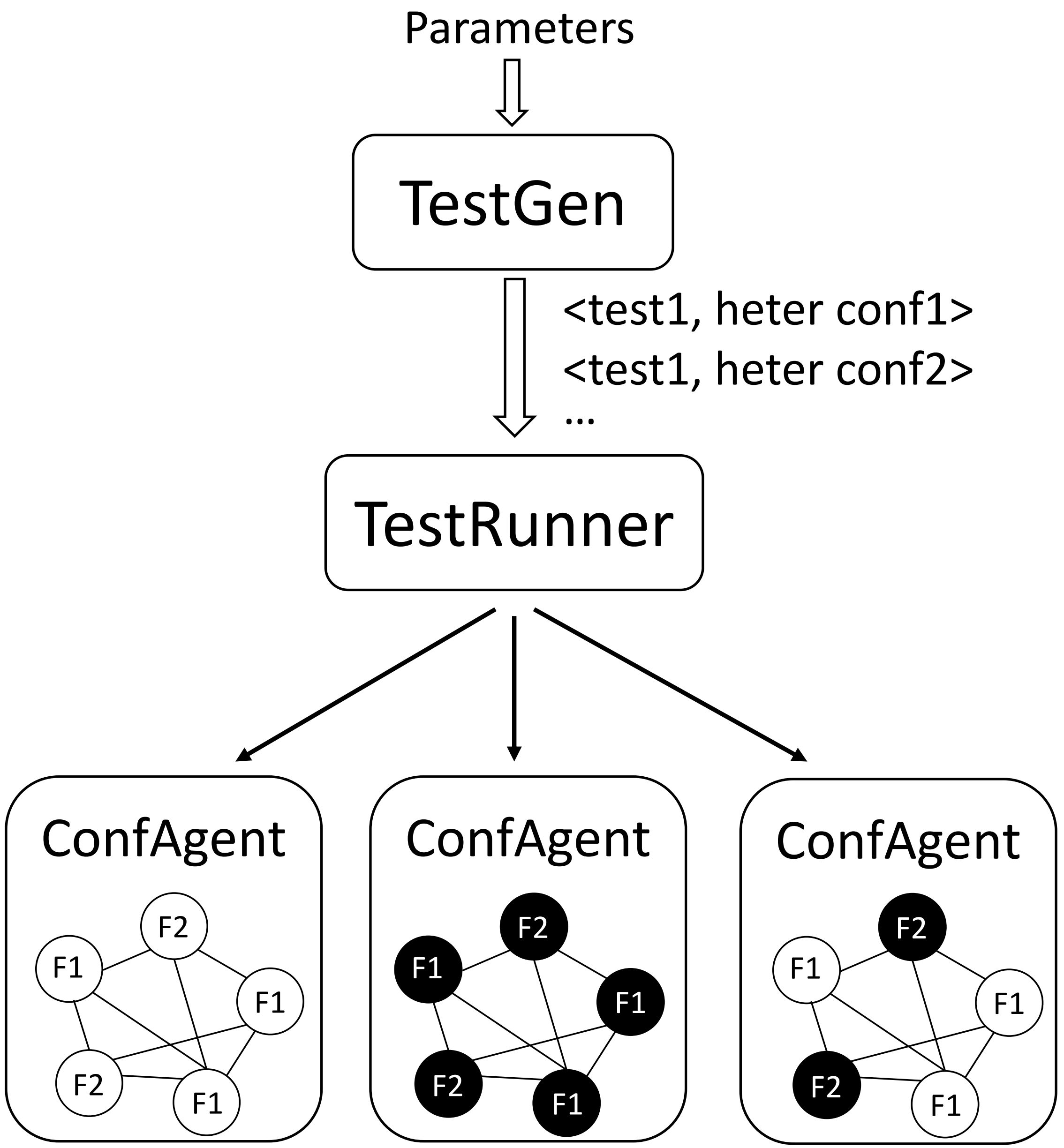
ZebraConf: Major Challenges

- C1: How to reduce testing time?
  - Apps can have 1000s of tests, 100s-1000s of parameters.
  - A test runs for several seconds to several minutes.

ZebraConf: Major Challenges

- C1: How to reduce testing time?
  - Apps can have 1000s of tests, 100s-1000s of parameters.
  - A test runs for several seconds to several minutes.
- C2: How to assign heterogeneous configuration in unit tests?
  - We can specify the config when starting a node as process.
    - E.g., hadoop-daemon.sh --config [CONFIG_PATH] start
  - However, this approach doesn’t work in unit tests.

ZebraConf Overview

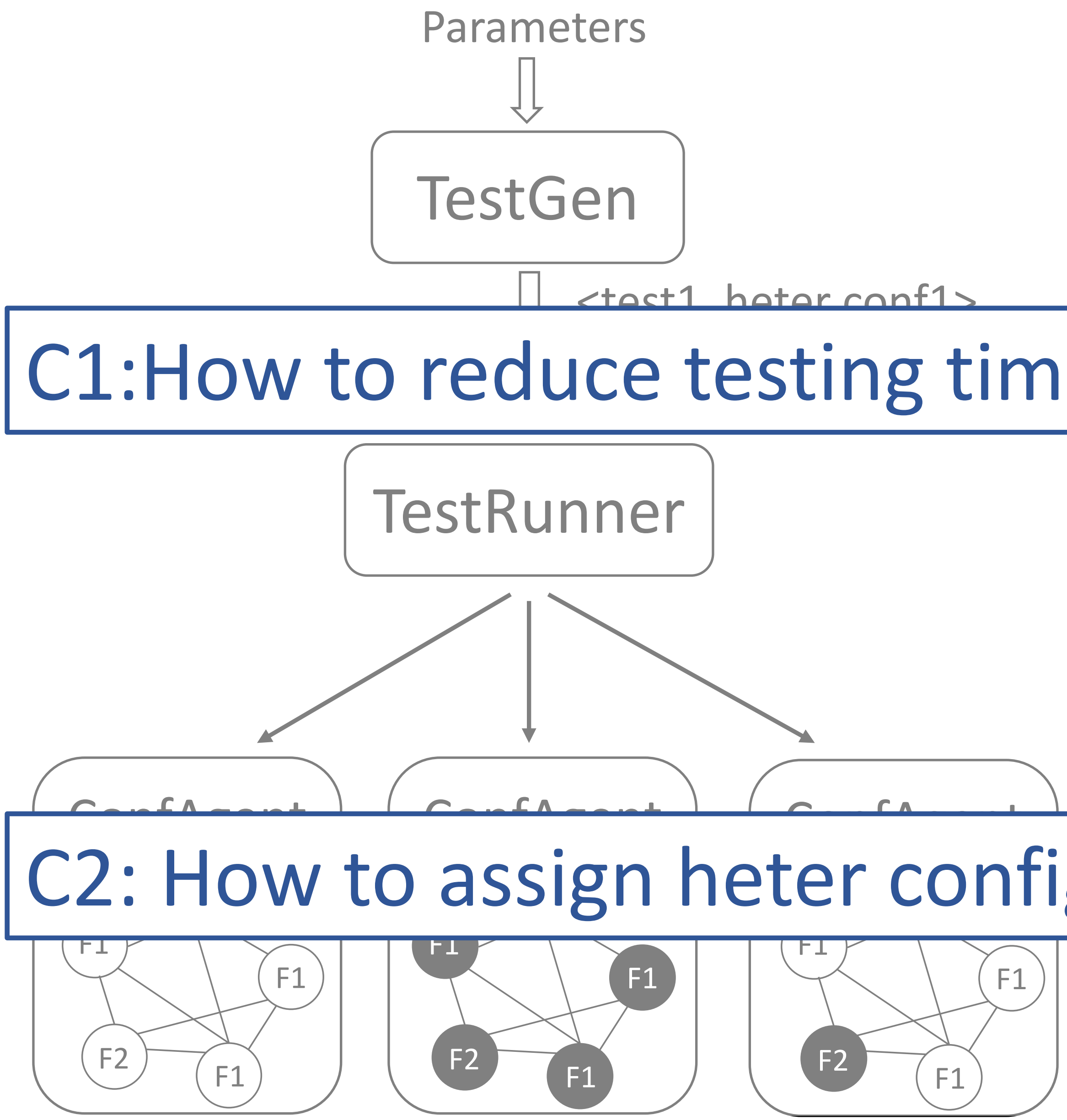


TestGen: generate test instances

TestRunner: conduct a test

ConfAgent: assign configs to nodes

ZebraConf Overview



C1:How to reduce testing time

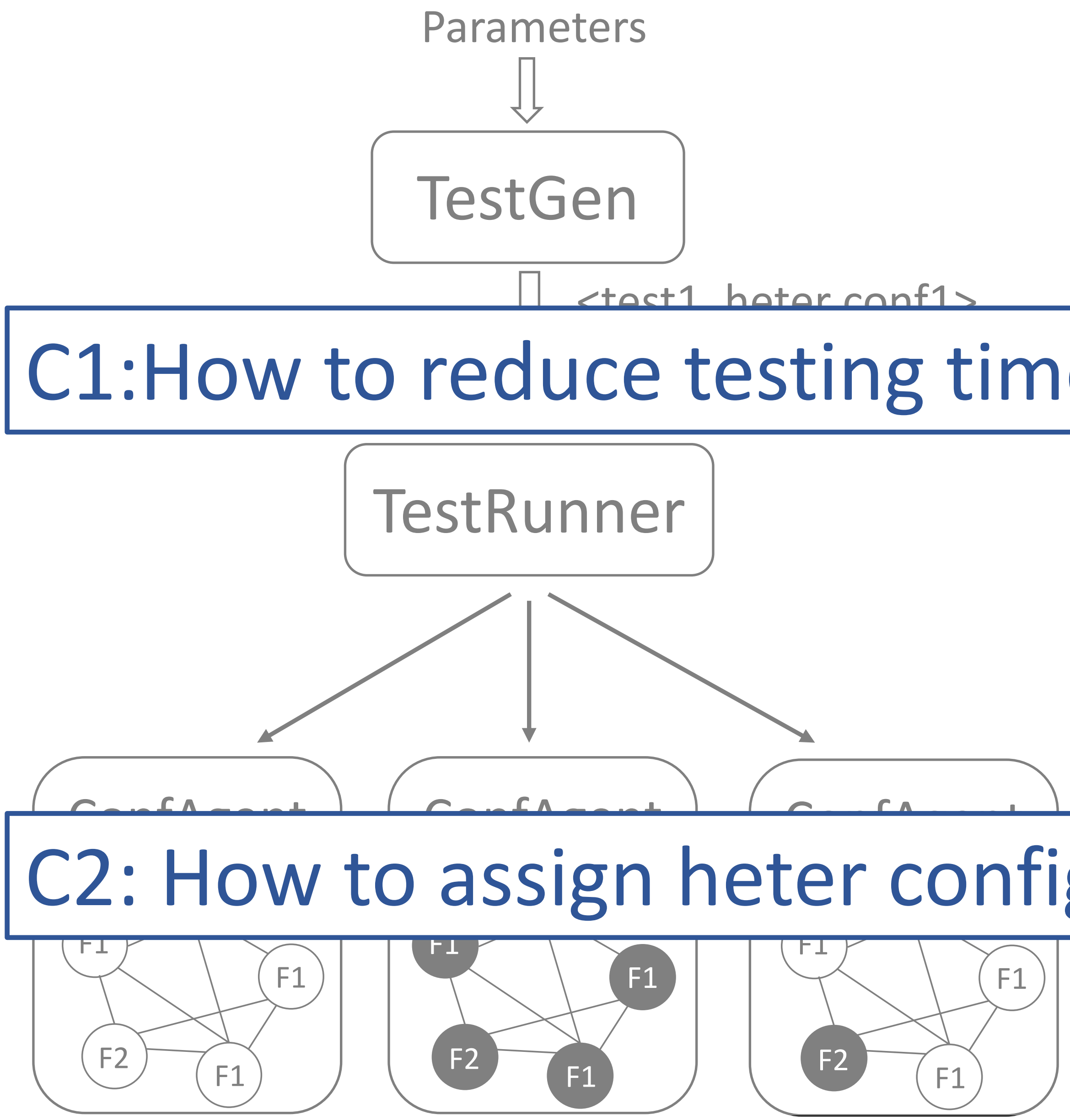
C2: How to assign heter config in unit tests

TestGen: generate test instances

TestRunner: conduct a test

ConfAgent: assign configs to nodes

ZebraConf Overview



C1:How to reduce testing time

C2: How to assign heter config in unit tests

TestGen: generate test instances

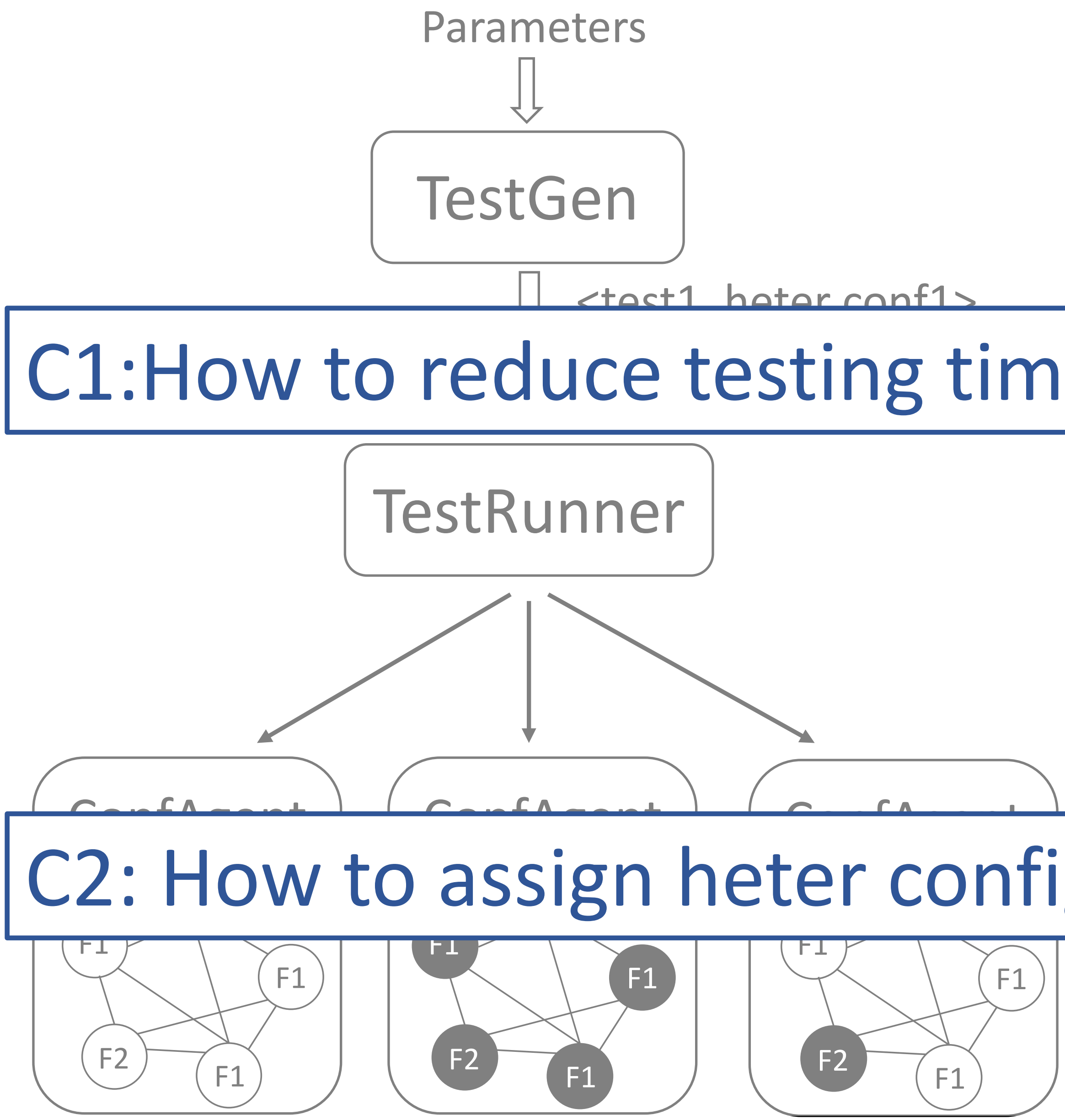
- Selective value assignment
- Pre-run profiling
- Pooled tests

TestRunner: conduct a test

- Supporting pooled testing
- Concurrent testing
- Hypothesis testing

ConfAgent: assign configs to nodes

ZebraConf Overview



C1:How to reduce testing time

C2: How to assign heter config in unit tests

TestGen: generate test instances

- Selective value assignment
- Pre-run profiling
- Pooled tests

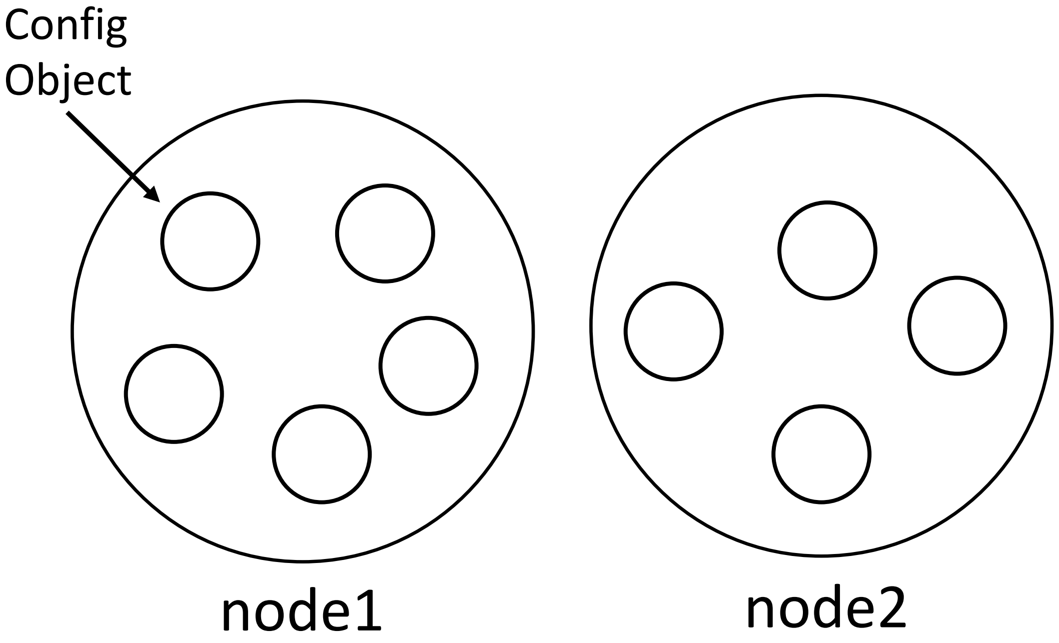
TestRunner: conduct a test

- Supporting pooled testing
- Concurrent testing
- Hypothesis testing

ConfAgent: assign configs to nodes

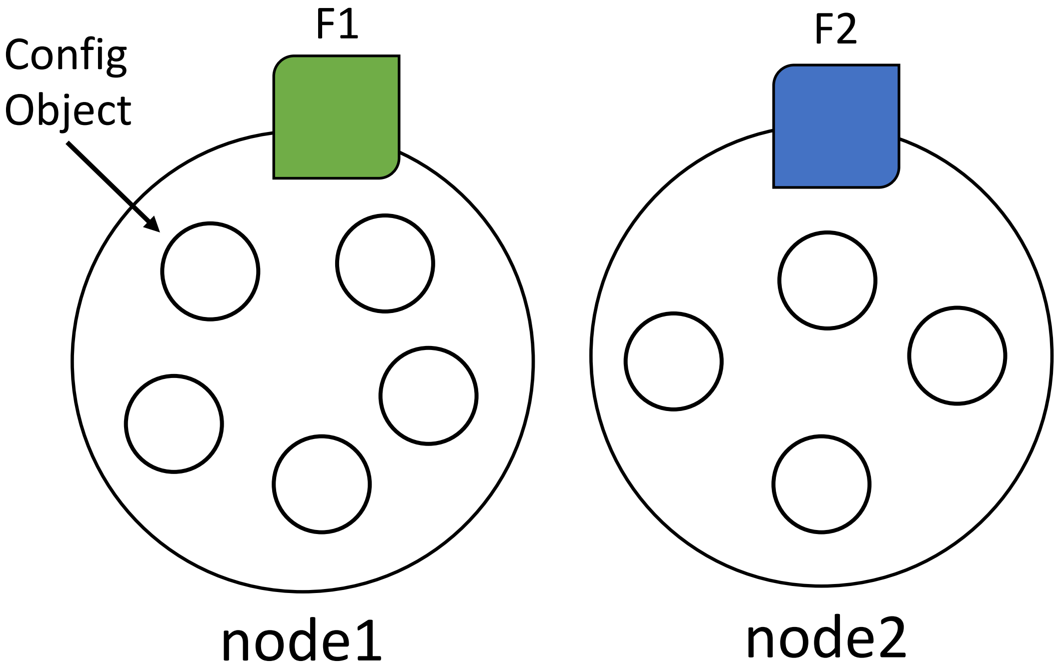
ConfAgent: Challenges

In distributed setting, we can specify the config file when starting a node as process.



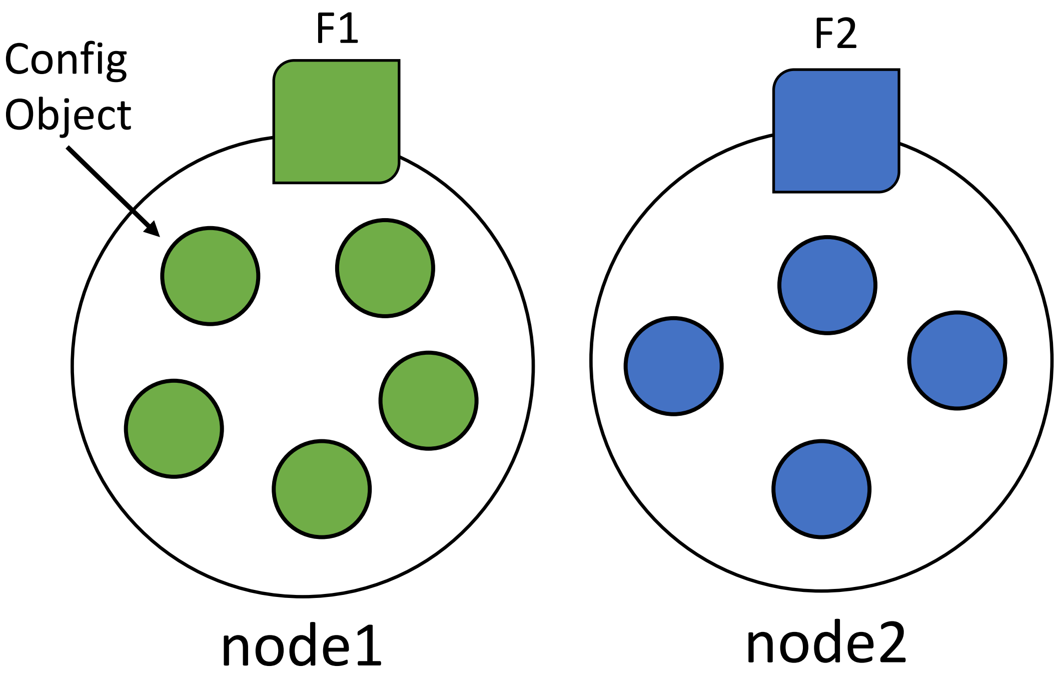
ConfAgent: Challenges

In distributed setting, we can specify the config file when starting a node as process.



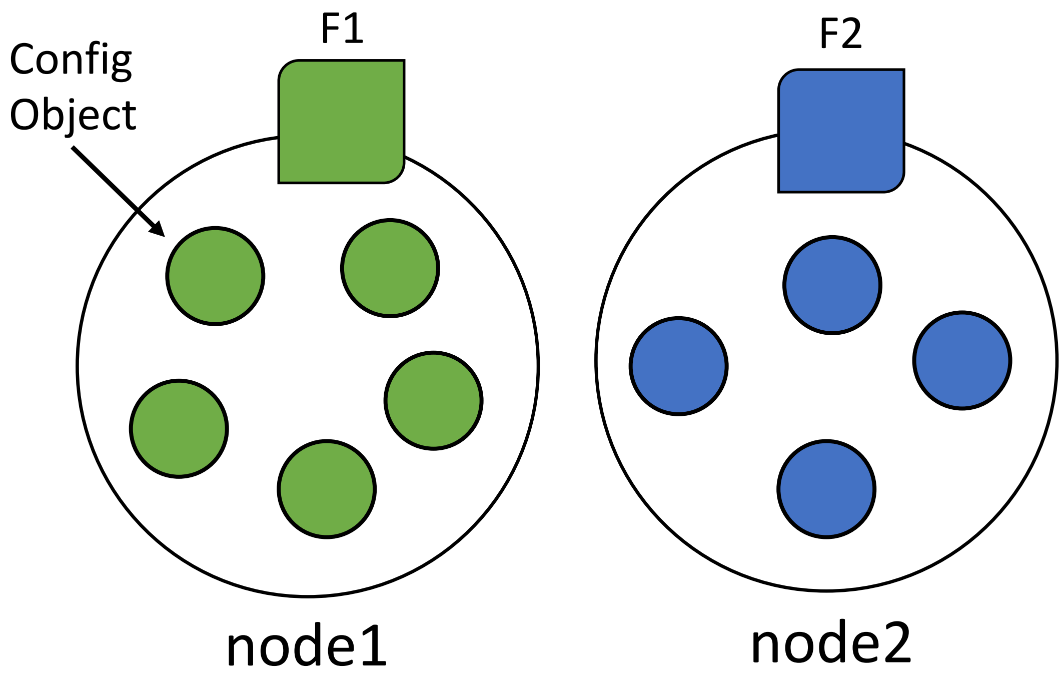
ConfAgent: Challenges

In distributed setting, we can specify the config file when starting a node as process.

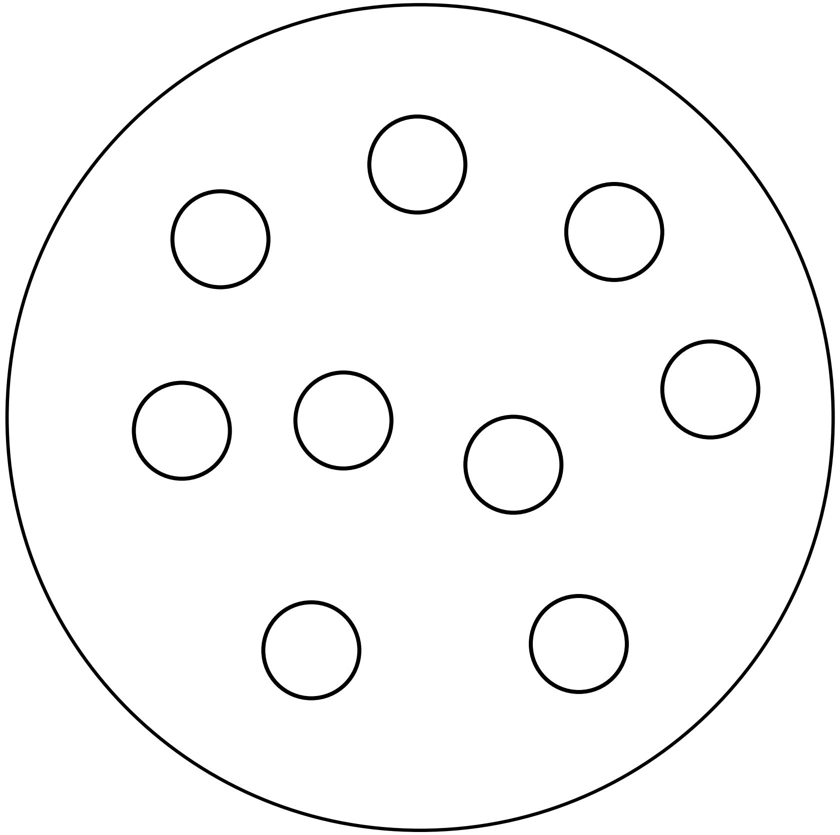


ConfAgent: Challenges

In distributed setting, we can specify the config file when starting a node as process.



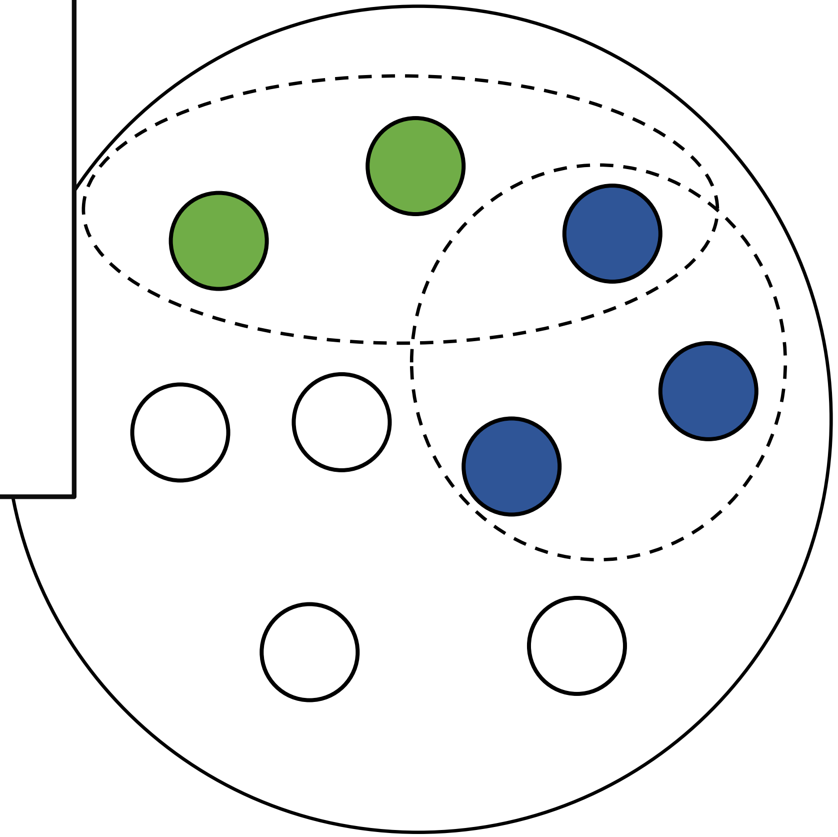
This doesn’t work in unit tests, as nodes are often created as threads in a single process (i.e., minicluster testing)



ConfAgent: Challenges

Key idea: attribute config objects to nodes & hack return values

t work in unit tests, as nodes are
ed as threads in a single process
uster testing)



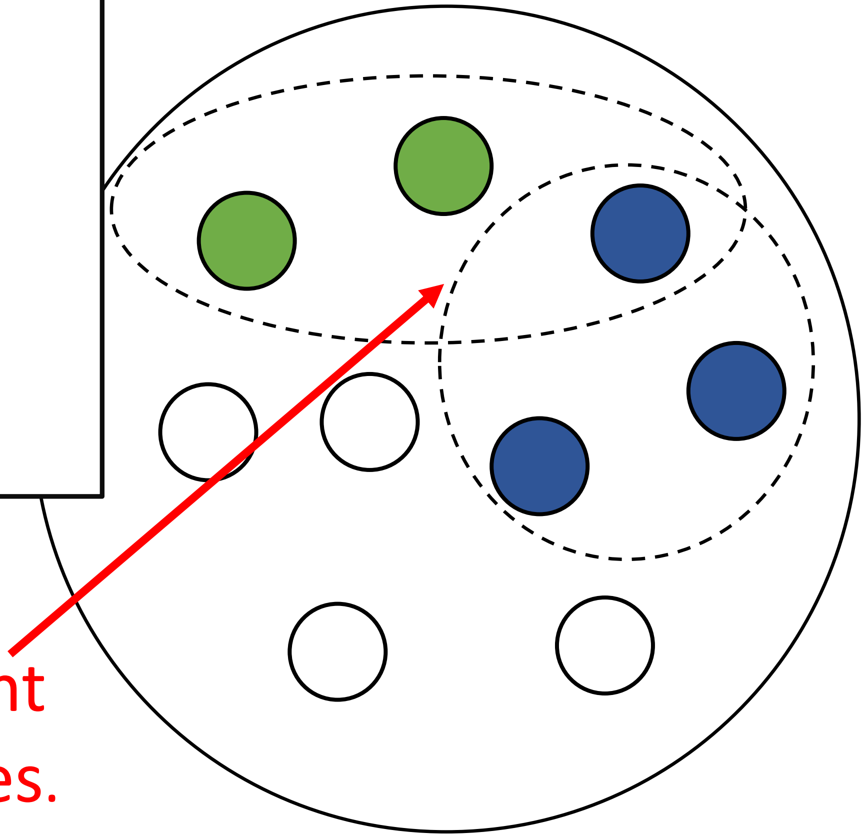
ConfAgent: Challenges

t work in unit tests, as nodes are
ed as threads in a single process
uster testing)

Key idea: attribute config objects to nodes & hack return values

Challenges:

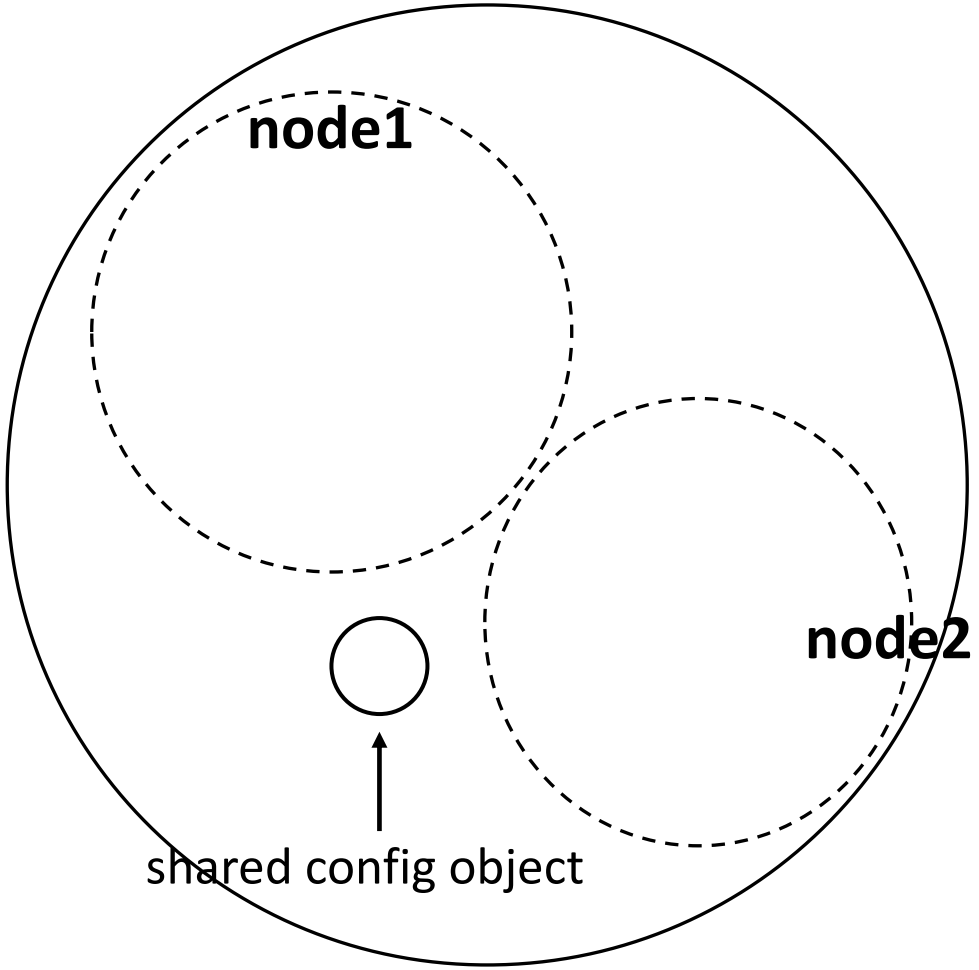
- Each node can have multiple config objects.
- Config objects can be shared among nodes.
  - Values in one config object seen by multiple nodes.



A node may read inconsistent values, causing false positives.

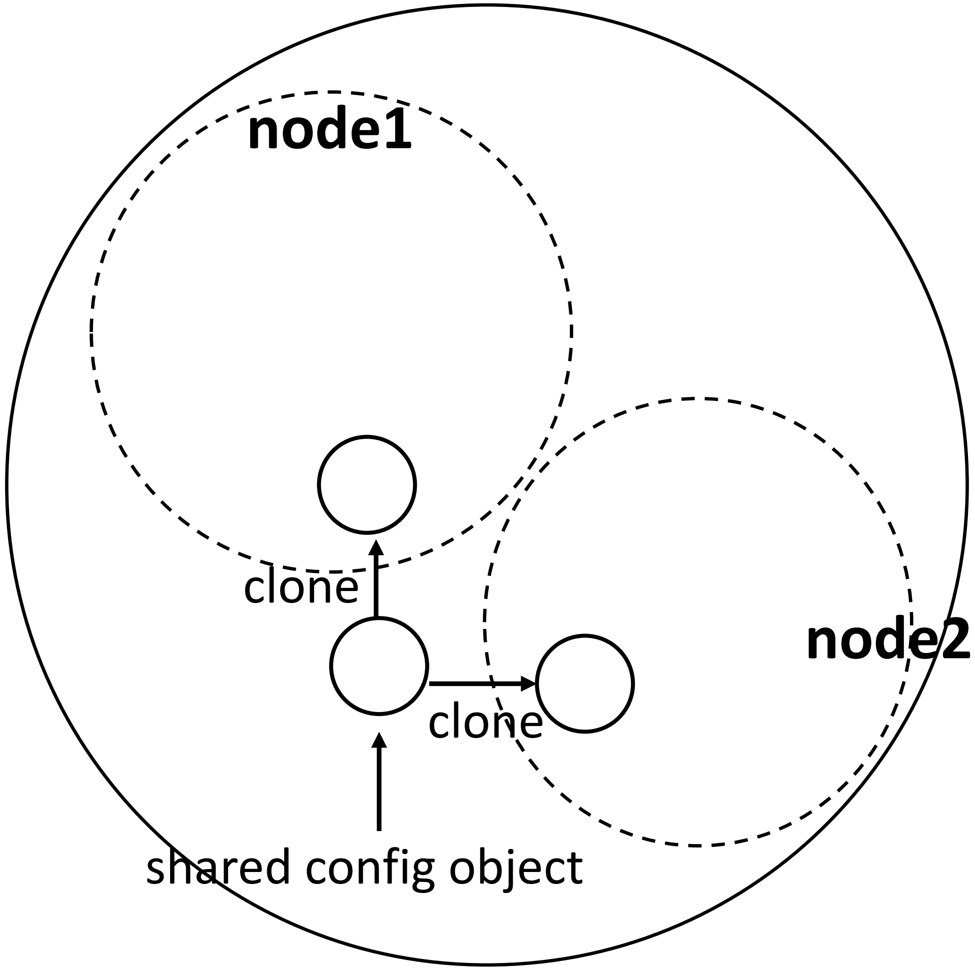
ConfAgent’s Solution

- Clone config objects that can be shared.



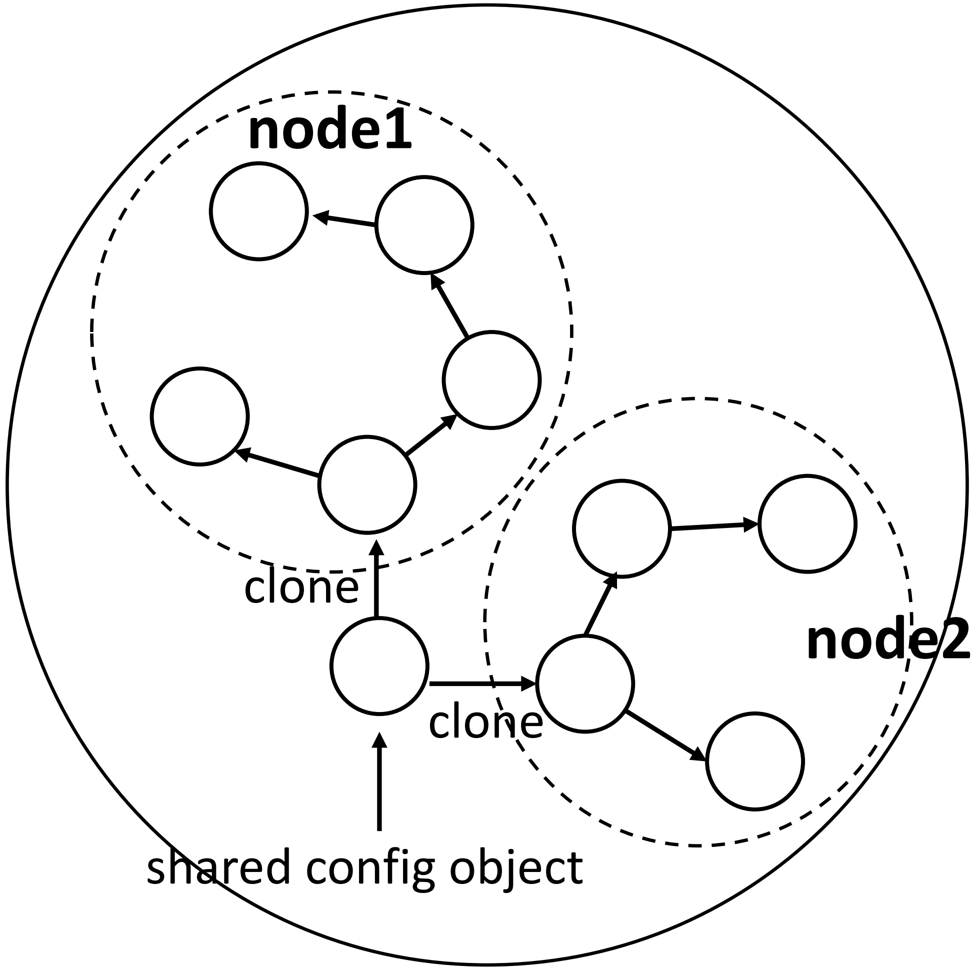
ConfAgent’s Solution

- Clone config objects that can be shared.



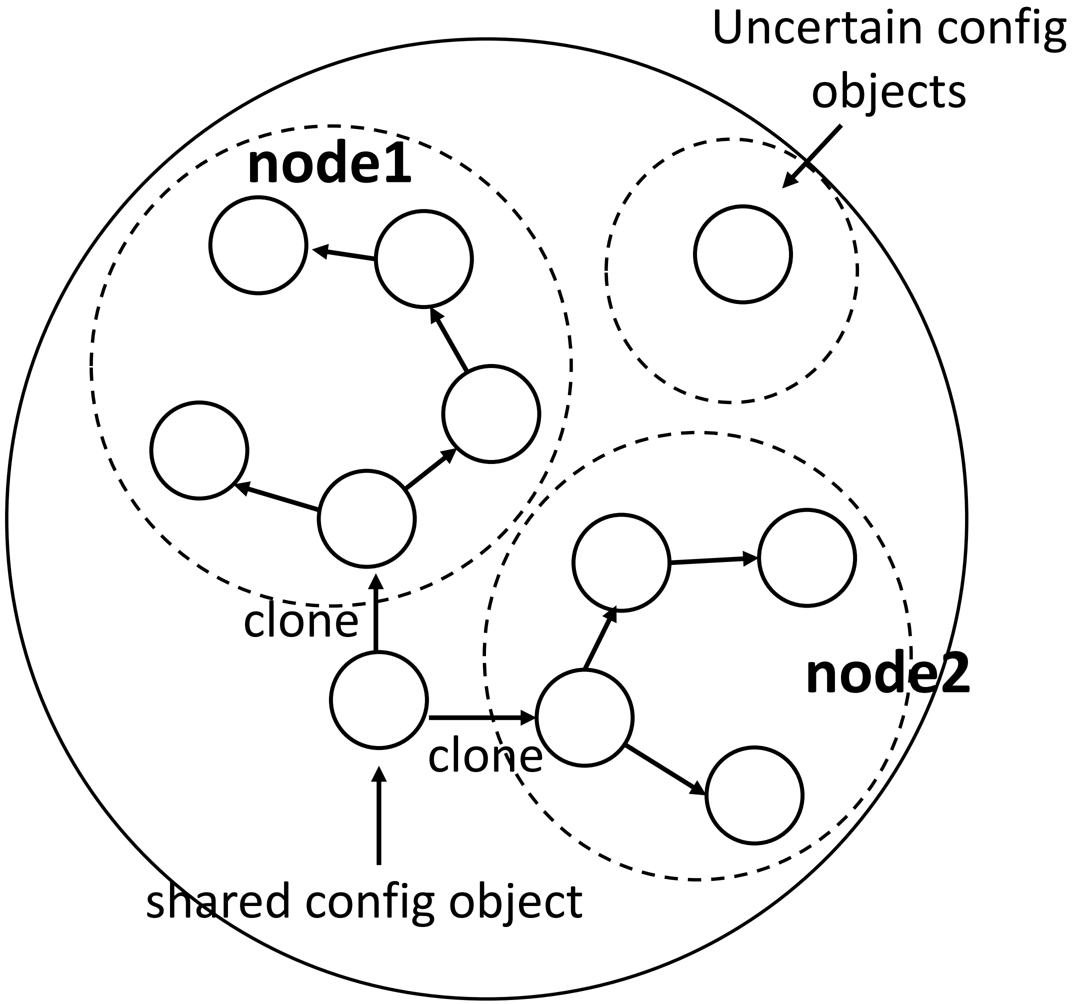
ConfAgent’s Solution

- Clone config objects that can be shared.
- Track config creation flow and attribute config objects to nodes by rules.
    - E.g., Conf b = new Conf(a) & a belongs to node1 → b belongs to node1



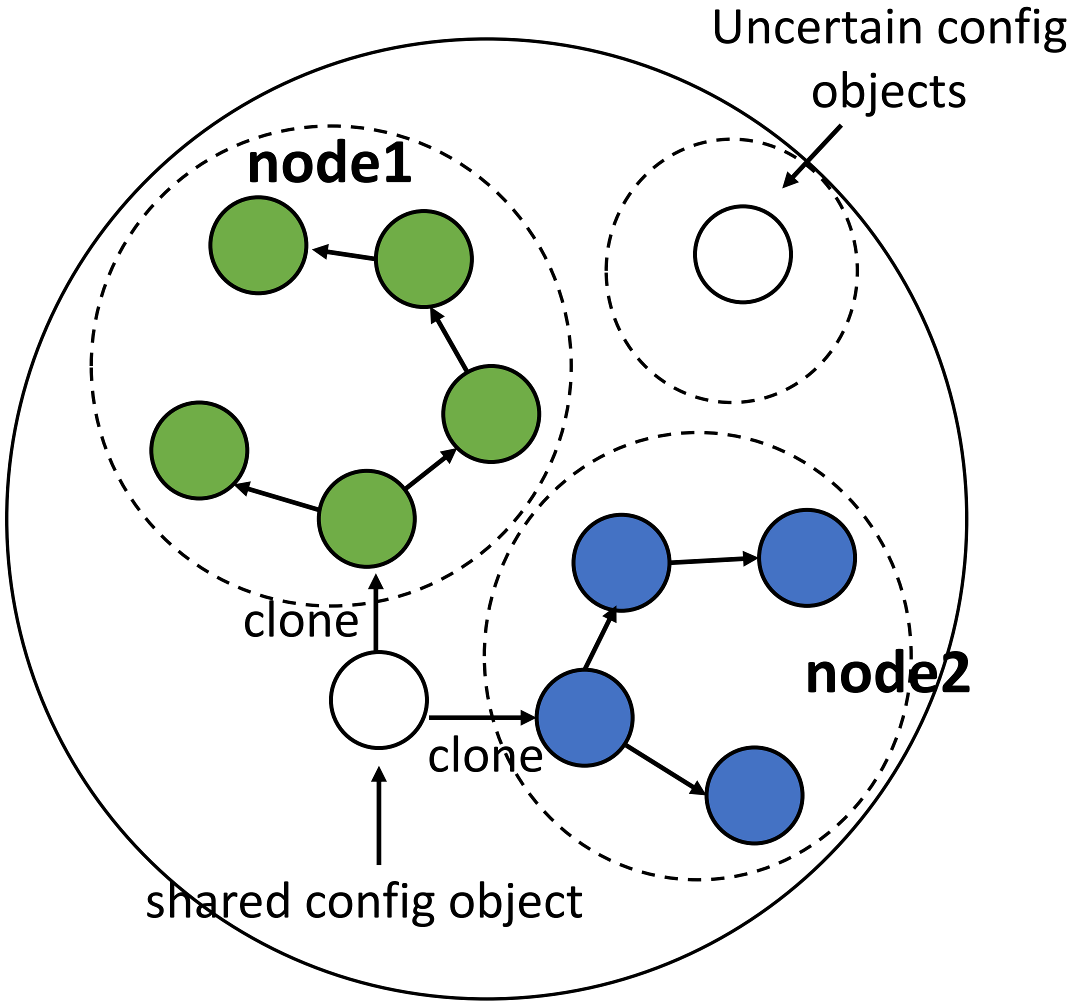
ConfAgent’s Solution

- Clone config objects that can be shared.
- Track config creation flow and attribute config objects to nodes by rules.
  - E.g., Conf b = new Conf(a) & a belongs to node1 → b belongs to node1
- Track uncertain config objects.
  - Avoid testing parameters read from them.



ConfAgent’s Solution

- Clone config objects that can be shared.
- Track config creation flow and attribute config objects to nodes by rules.
  - E.g., Conf b = new Conf(a) & a belongs to node1 → b belongs to node1
- Track uncertain config objects.
  - Avoid testing parameters read from them.
- Manipulate config parameter values.



Evaluation

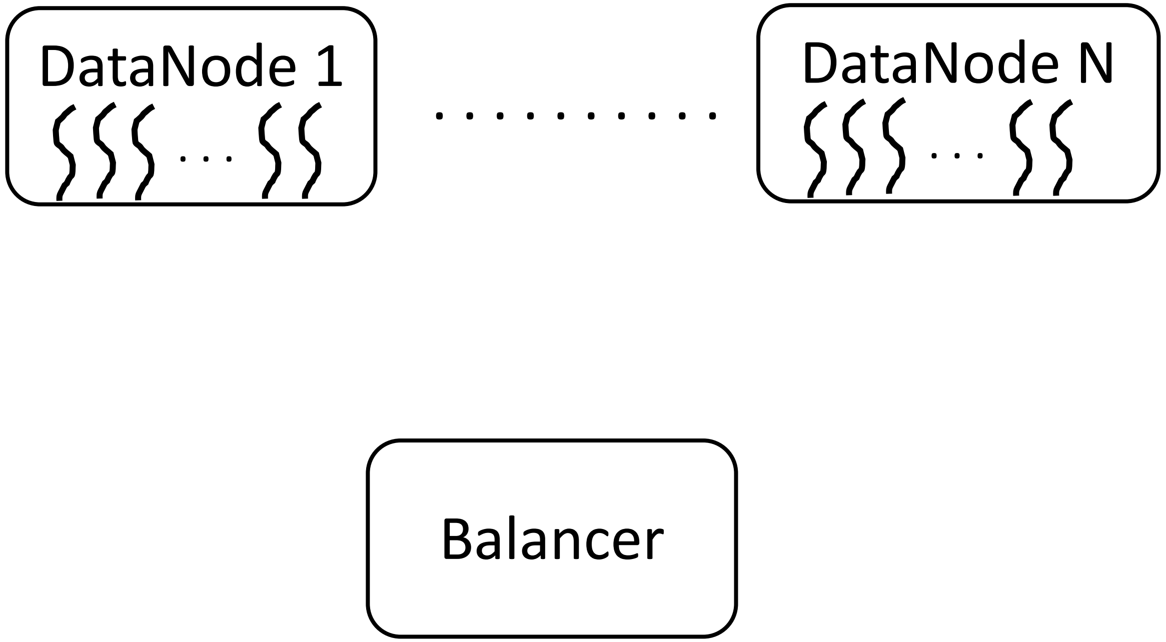
- Hardware setting
    - We run all the experiments on CloudLab
    - Intel Xeon 10-core CPUs, 192 GB DRAM, 480 GB SATA SSD
- Applications
    - Five app: HDFS, YARN, MR, HBase, Flink
    - Modification overhead: 18 to 38 LOC
    - Totally 4,652 machine hours with up to 100 physical machines, each running 20 Docker containers

Evaluation

- ZebraConf reports 57 heterogeneous-unsafe parameters.
- Our manual analysis finds 41 are true problems.
- Categories of these parameters:
  - Data transfer format related
  - Max limit related
  - Timeout related
  - Task numbers related
  - …
  - Unexpected ones

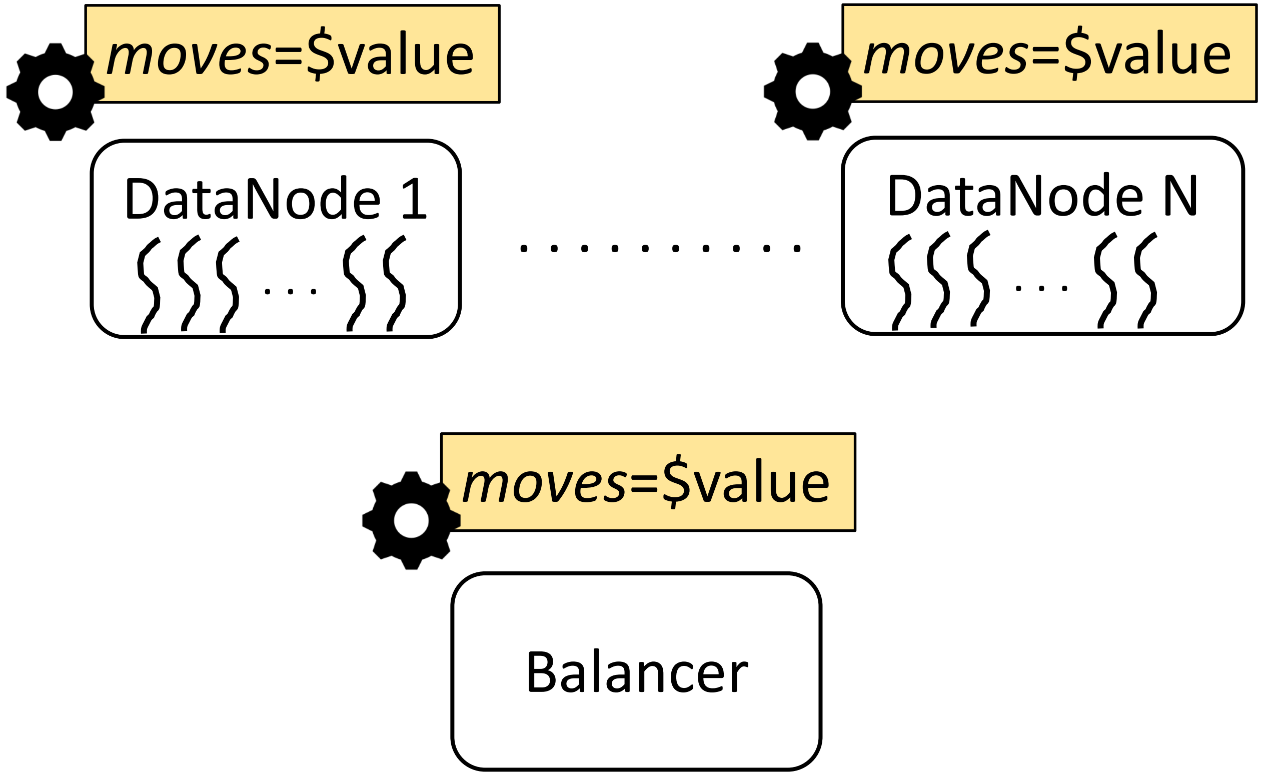
dfs.datanode.balance.max.concurrent.moves

- Limits the max number of threads that a DataNode can use for balancing.



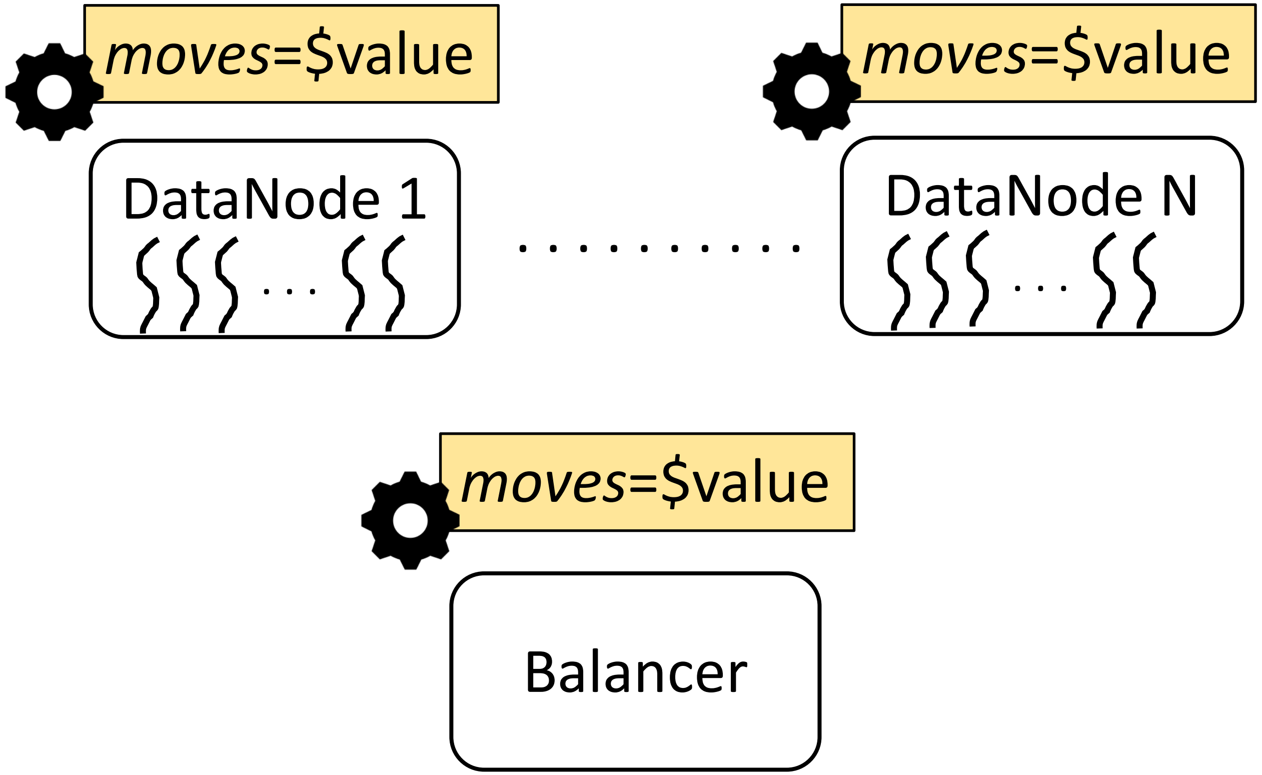
dfs.datanode.balance.max.concurrent.moves

- Limits the max number of threads that a DataNode can use for balancing.



dfs.datanode.balance.max.concurrent.moves

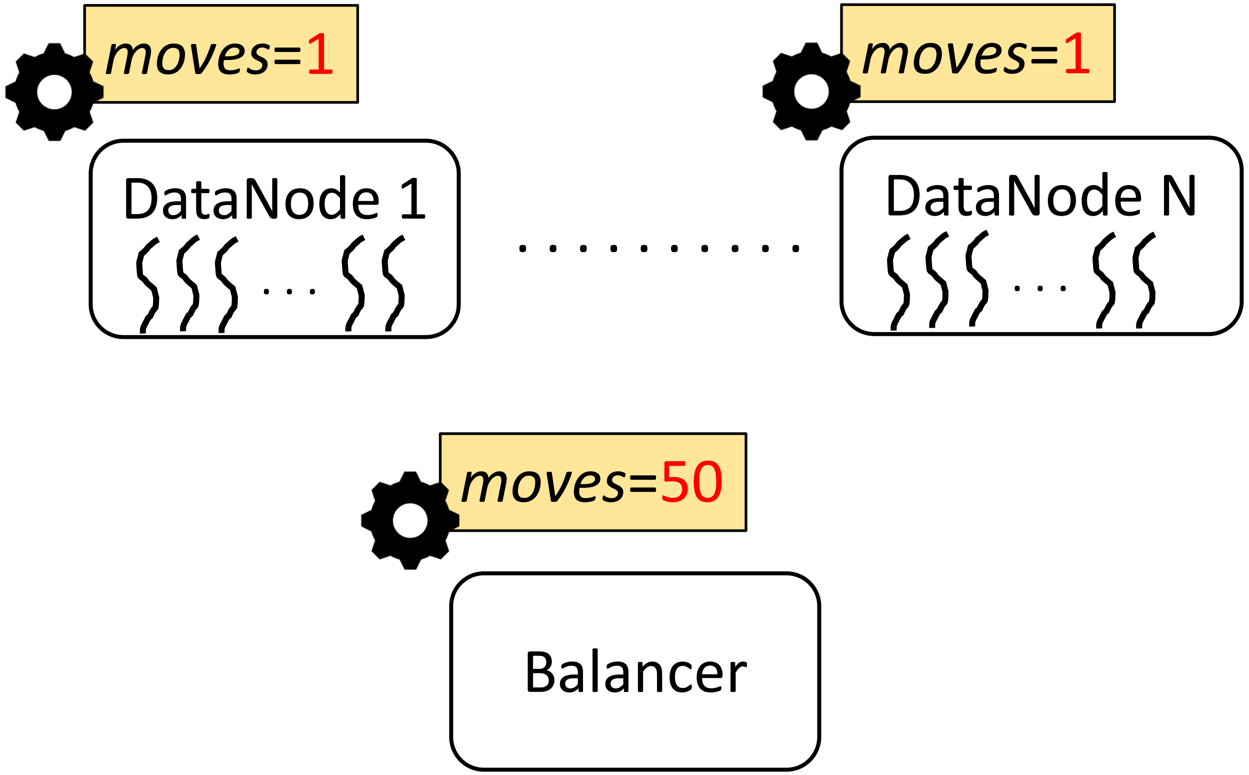
- Limits the max number of threads that a DataNode can use for balancing.



Configs	Balancing Time
Balancer: 50, DN: 50	14s
Balancer: 1, DN: 1	16.7s

dfs.datanode.balance.max.concurrent.moves

- Limits the max number of threads that a DataNode can use for balancing.

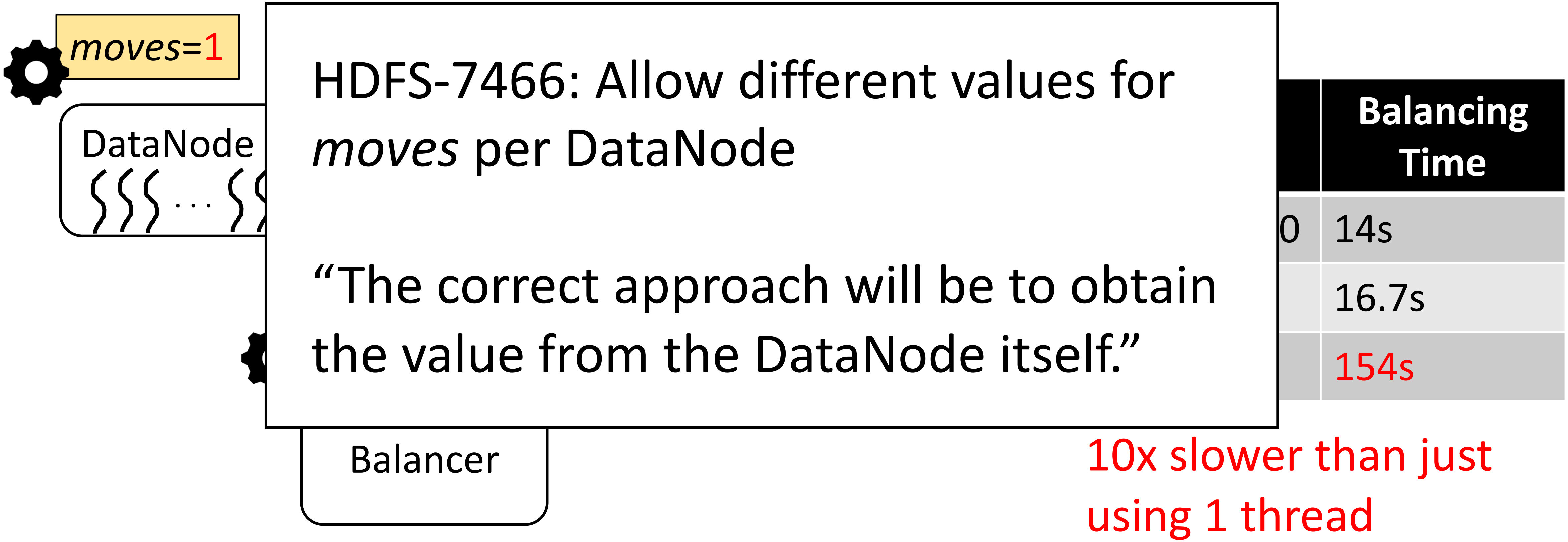


Configs	Balancing Time
Balancer: 50, DN: 50	14s
Balancer: 1, DN: 1	16.7s
Balancer: 50, DN: 1	154s

10x slower than just using 1 thread

dfs.datanode.balance.max.concurrent.moves

- Limits the max number of threads that a DataNode can use for balancing.



HDFS-7466: Allow different values for *moves* per DataNode

“The correct approach will be to obtain the value from the DataNode itself.”

	Balancing Time
0	14s
	16.7s
	154s

10x slower than just using 1 thread

Conclusion

- ZebraConf reuses existing unit tests to find unsafe parameters.
- We find 41 heterogeneous-unsafe parameters with ZebraConf.
- Need better support for heterogeneous configurations.
- We made ZebraConf publicly available: https://github.com/StarThinking/ZebraConf/