Artifacts Available acm V1.1

Artifacts Evaluated acm Reusable V1.1

Test Selection for Unified Regression Testing

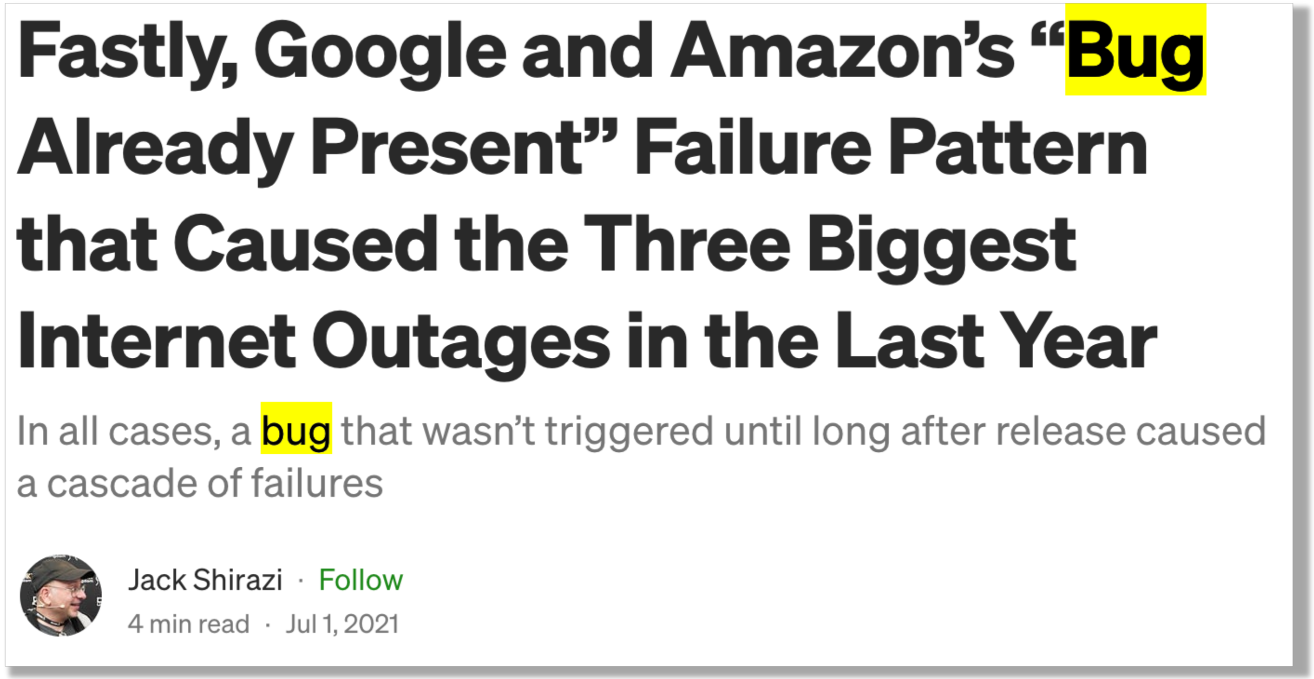
Shuai Wang, Xinyu Lian, Darko Marinov, Tianyin Xu

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

ICSE, Melbourne, Australia
05/18/2023

Code bugs and misconfigurations are two dominating root causes of software failures

Code bugs and misconfigurations are two dominating root causes of software failures



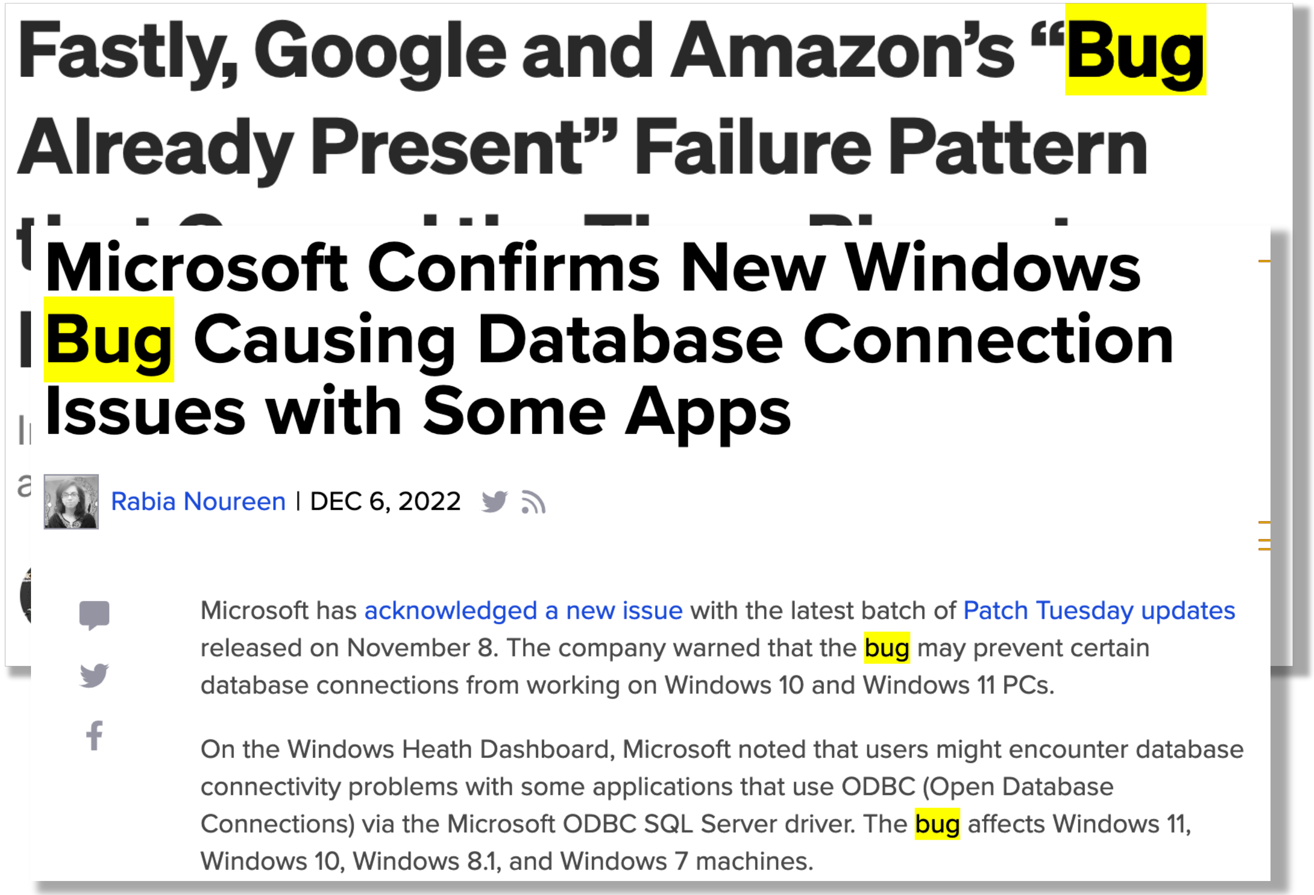
Fastly, Google and Amazon's "Bug Already Present" Failure Pattern that Caused the Three Biggest Internet Outages in the Last Year

In all cases, a bug that wasn't triggered until long after release caused a cascade of failures

Jack Shirazi · Follow

4 min read · Jul 1, 2021

Code bugs and misconfigurations are two dominating root causes of software failures



Fastly, Google and Amazon's "Bug Already Present" Failure Pattern

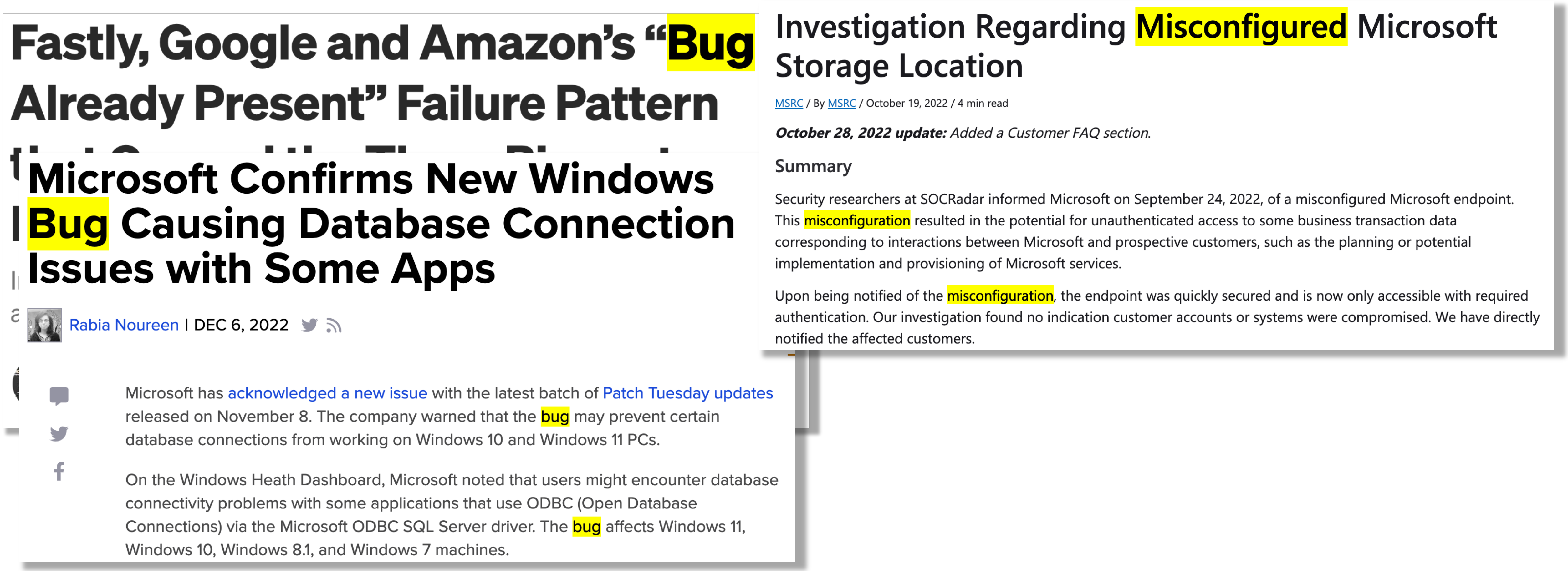
Microsoft Confirms New Windows Bug Causing Database Connection Issues with Some Apps

Rabia Noureen | DEC 6, 2022

Microsoft has acknowledged a new issue with the latest batch of Patch Tuesday updates released on November 8. The company warned that the bug may prevent certain database connections from working on Windows 10 and Windows 11 PCs.

On the Windows Heath Dashboard, Microsoft noted that users might encounter database connectivity problems with some applications that use ODBC (Open Database Connections) via the Microsoft ODBC SQL Server driver. The bug affects Windows 11, Windows 10, Windows 8.1, and Windows 7 machines.

Code bugs and misconfigurations are two dominating root causes of software failures



Fastly, Google and Amazon's "Bug Already Present" Failure Pattern

Microsoft Confirms New Windows Bug Causing Database Connection Issues with Some Apps

Rabia Noureen | DEC 6, 2022

Microsoft has acknowledged a new issue with the latest batch of Patch Tuesday updates released on November 8. The company warned that the bug may prevent certain database connections from working on Windows 10 and Windows 11 PCs.

On the Windows Heath Dashboard, Microsoft noted that users might encounter database connectivity problems with some applications that use ODBC (Open Database Connections) via the Microsoft ODBC SQL Server driver. The bug affects Windows 11, Windows 10, Windows 8.1, and Windows 7 machines.

Investigation Regarding Misconfigured Microsoft Storage Location

MSRC / By MSRC / October 19, 2022 / 4 min read

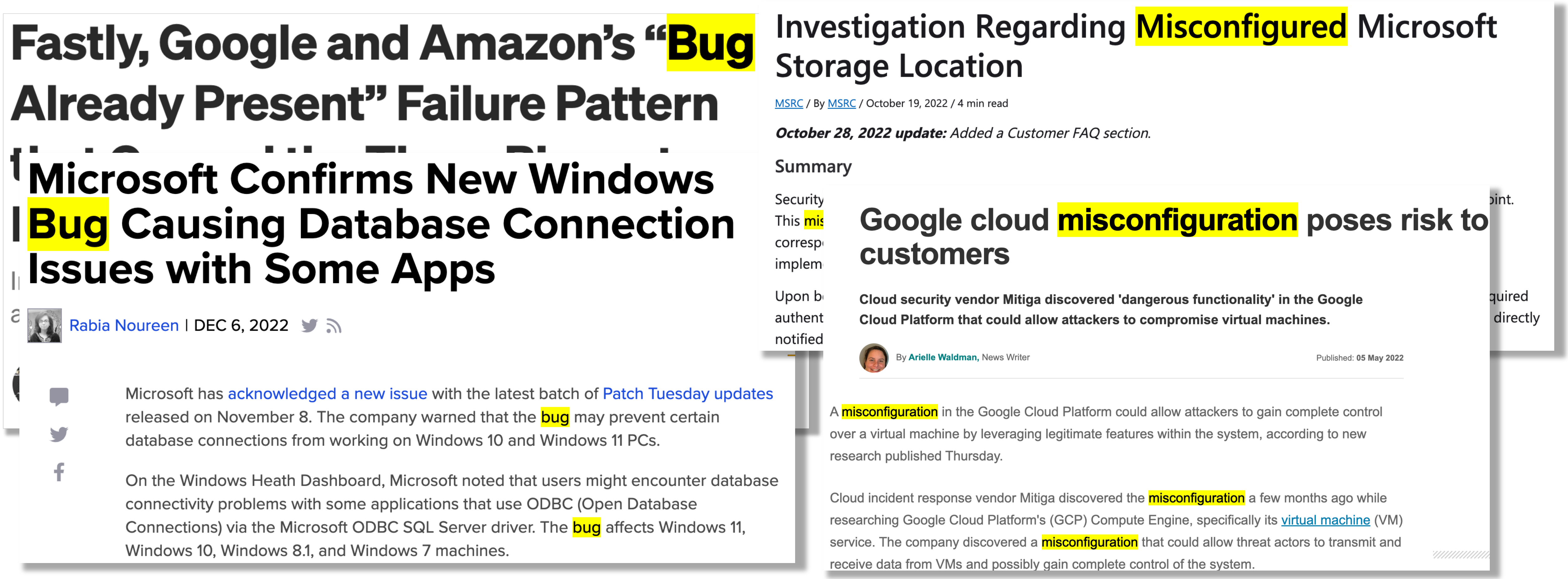
October 28, 2022 update: *Added a Customer FAQ section.*

Summary

Security researchers at SOCRadar informed Microsoft on September 24, 2022, of a misconfigured Microsoft endpoint. This misconfiguration resulted in the potential for unauthenticated access to some business transaction data corresponding to interactions between Microsoft and prospective customers, such as the planning or potential implementation and provisioning of Microsoft services.

Upon being notified of the misconfiguration, the endpoint was quickly secured and is now only accessible with required authentication. Our investigation found no indication customer accounts or systems were compromised. We have directly notified the affected customers.

Code bugs and misconfigurations are two dominating root causes of software failures



Fastly, Google and Amazon's "Bug Already Present" Failure Pattern

Microsoft Confirms New Windows Bug Causing Database Connection Issues with Some Apps

Rabia Noureen | DEC 6, 2022

Microsoft has acknowledged a new issue with the latest batch of Patch Tuesday updates released on November 8. The company warned that the bug may prevent certain database connections from working on Windows 10 and Windows 11 PCs.

On the Windows Heath Dashboard, Microsoft noted that users might encounter database connectivity problems with some applications that use ODBC (Open Database Connections) via the Microsoft ODBC SQL Server driver. The bug affects Windows 11, Windows 10, Windows 8.1, and Windows 7 machines.

Investigation Regarding Misconfigured Microsoft Storage Location

MSRC / By MSRC / October 19, 2022 / 4 min read

October 28, 2022 update: *Added a Customer FAQ section.*

Summary

Security
This mis
corresp
implem

oint.

Upon b
authent
notified

quired
directly

Google cloud misconfiguration poses risk to customers

Cloud security vendor Mitiga discovered 'dangerous functionality' in the Google Cloud Platform that could allow attackers to compromise virtual machines.

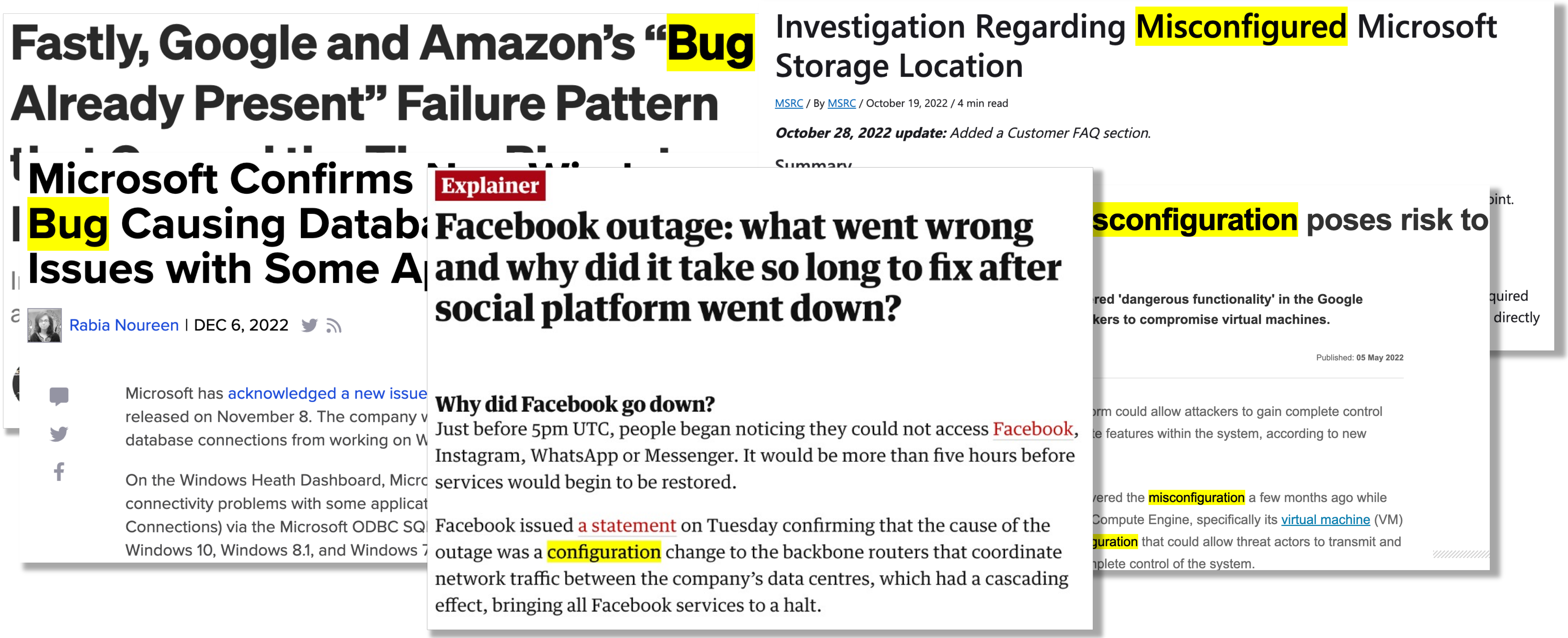
By **Arielle Waldman,** News Writer

Published: **05 May 2022**

A misconfiguration in the Google Cloud Platform could allow attackers to gain complete control over a virtual machine by leveraging legitimate features within the system, according to new research published Thursday.

Cloud incident response vendor Mitiga discovered the misconfiguration a few months ago while researching Google Cloud Platform's (GCP) Compute Engine, specifically its virtual machine (VM) service. The company discovered a misconfiguration that could allow threat actors to transmit and receive data from VMs and possibly gain complete control of the system.

Code bugs and misconfigurations are two dominating root causes of software failures



Fastly, Google and Amazon's "Bug Already Present" Failure Pattern

Microsoft Confirms Bug Causing Databa Issues with Some A

Rabia Noureen | DEC 6, 2022

Microsoft has acknowledged a new issue
released on November 8. The company w
database connections from working on W

On the Windows Heath Dashboard, Micro
connectivity problems with some applicat
Connections) via the Microsoft ODBC SQ
Windows 10, Windows 8.1, and Windows 7

Explainer

Facebook outage: what went wrong and why did it take so long to fix after social platform went down?

Why did Facebook go down?

Just before 5pm UTC, people began noticing they could not access Facebook, Instagram, WhatsApp or Messenger. It would be more than five hours before services would begin to be restored.

Facebook issued a statement on Tuesday confirming that the cause of the outage was a configuration change to the backbone routers that coordinate network traffic between the company's data centres, which had a cascading effect, bringing all Facebook services to a halt.

Investigation Regarding Misconfigured Microsoft Storage Location

MSRC / By MSRC / October 19, 2022 / 4 min read

October 28, 2022 update: Added a Customer FAQ section.

Summary

oint.

quired
directly

sconfiguration poses risk to

red 'dangerous functionality' in the Google
kers to compromise virtual machines.

Published: 05 May 2022

rm could allow attackers to gain complete control
e features within the system, according to new

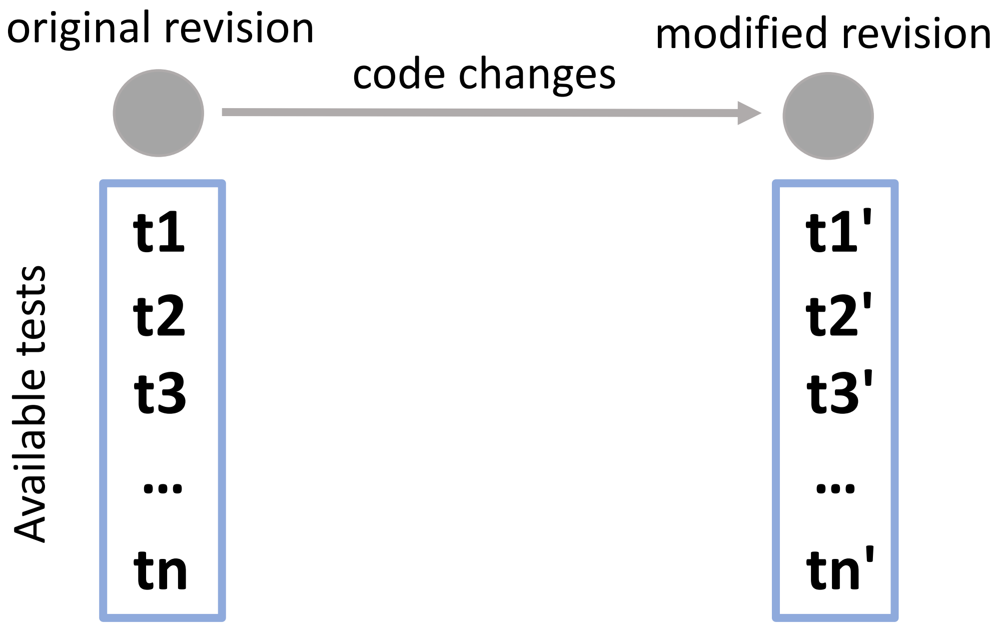
ered the misconfiguration a few months ago while
Compute Engine, specifically its virtual machine (VM)
guration that could allow threat actors to transmit and
plete control of the system.

Regression testing for code changes

- Checking that code changes do not break working functionality
- Widely used in modern CI/CD environments

Regression testing for code changes

- Checking that code changes do not break working functionality
- Widely used in modern CI/CD environments



original revision

code changes

modified revision

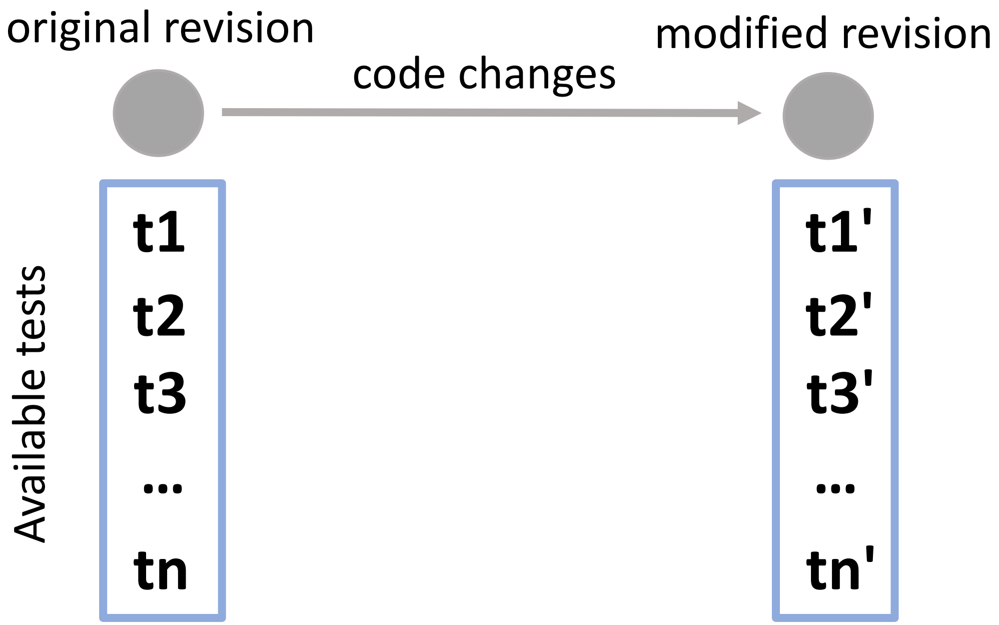
Available tests

t1
t2
t3
...
tn

t1'
t2'
t3'<br>...
tn'

Regression testing for code changes

- Checking that code changes do not break working functionality
- Widely used in modern CI/CD environments
- Mostly testing code under **default configuration**



original revision

code changes

modified revision

Available tests

t1
t2
t3
…
tn

t1'
t2'
t3'
…
tn'

Configuration testing for config changes

- Ctest: test production configuration changes together with code [1]

[1] Xudong Sun, et.al. Testing Configuration Changes in Context to Prevent Production Failures. OSDI 2020

Configuration testing for config changes

- Ctest: test production configuration changes together with code [1]

Configuration test

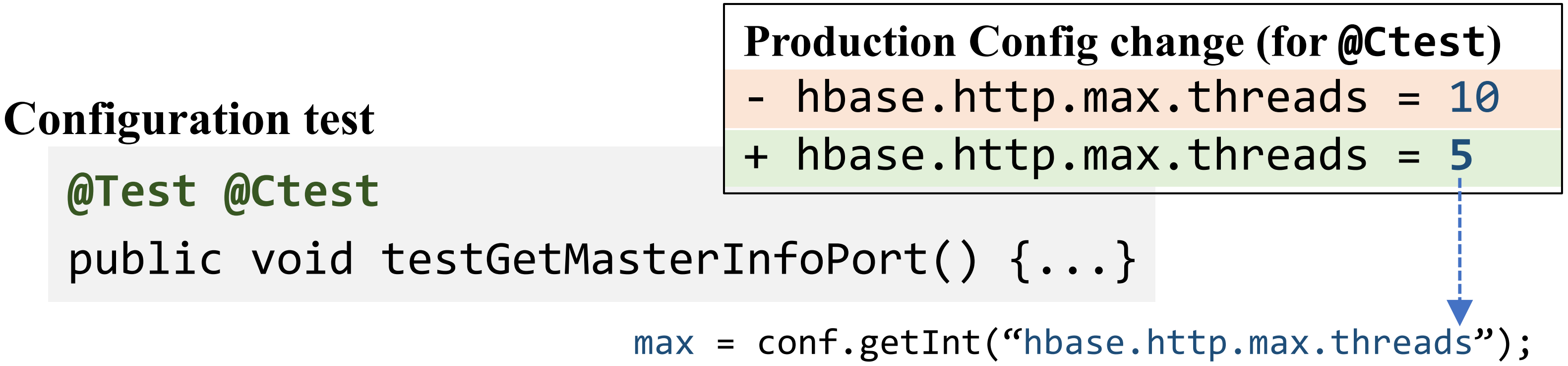
```
@Test @Ctest
public void testGetMasterInfoPort() {...}
```

```
max = conf.getInt("hbase.http.max.threads");
```

[1] Xudong Sun, et.al. Testing Configuration Changes in Context to Prevent Production Failures. OSDI 2020

Configuration testing for config changes

- Ctest: test production configuration changes together with code [1]



Production Config change (for `@Ctest`)

```
- hbase.http.max.threads = 10
+ hbase.http.max.threads = 5
```

Configuration test

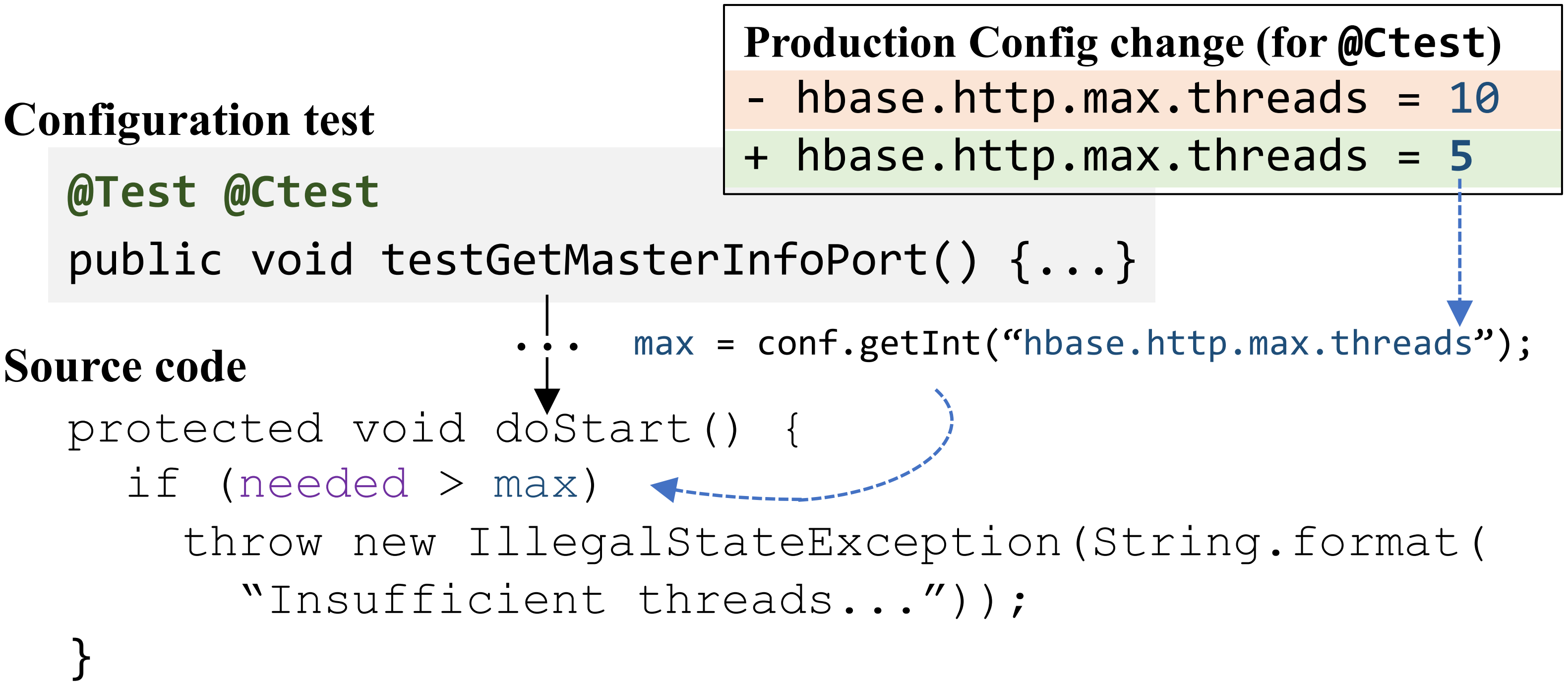
```
@Test @Ctest
public void testGetMasterInfoPort() {...}
```

```
max = conf.getInt(“hbase.http.max.threads”);
```

[1] Xudong Sun, et.al. Testing Configuration Changes in Context to Prevent Production Failures. OSDI 2020

Configuration testing for config changes

- Ctest: test production configuration changes together with code [1]



Production Config change (for @Ctest)

```
- hbase.http.max.threads = 10
+ hbase.http.max.threads = 5
```

Configuration test

```
@Test @Ctest
public void testGetMasterInfoPort() {...}
```

Source code

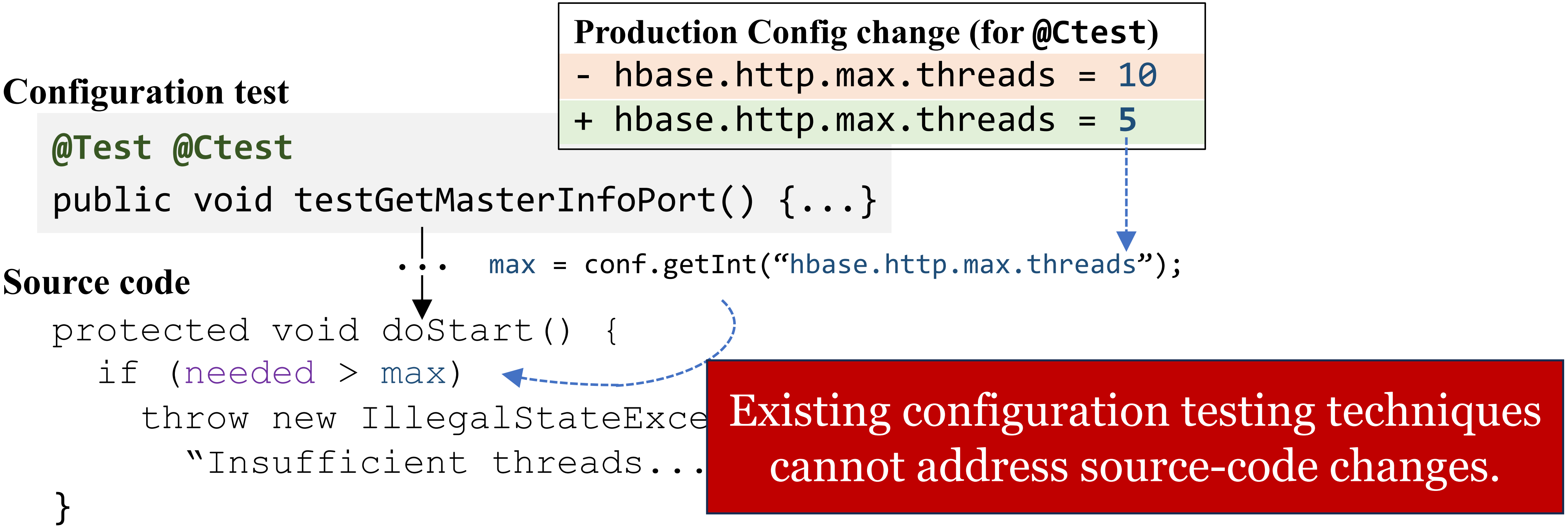
```
protected void doStart() {
  if (needed > max)
    throw new IllegalStateException(String.format(
      "Insufficient threads..."));
}
```

```
...  max = conf.getInt("hbase.http.max.threads");
```

[1] Xudong Sun, et.al. Testing Configuration Changes in Context to Prevent Production Failures. OSDI 2020

Configuration testing for config changes

- Ctest: test production configuration changes together with code [1]



Production Config change (for @Ctest)

```
- hbase.http.max.threads = 10
+ hbase.http.max.threads = 5
```

Configuration test

```
@Test @Ctest
public void testGetMasterInfoPort() {...}
```

Source code

```
...   max = conf.getInt(“hbase.http.max.threads”);
protected void doStart() {
  if (needed > max)
    throw new IllegalStateExce
      “Insufficient threads...
}
```

Existing configuration testing techniques cannot address source-code changes.

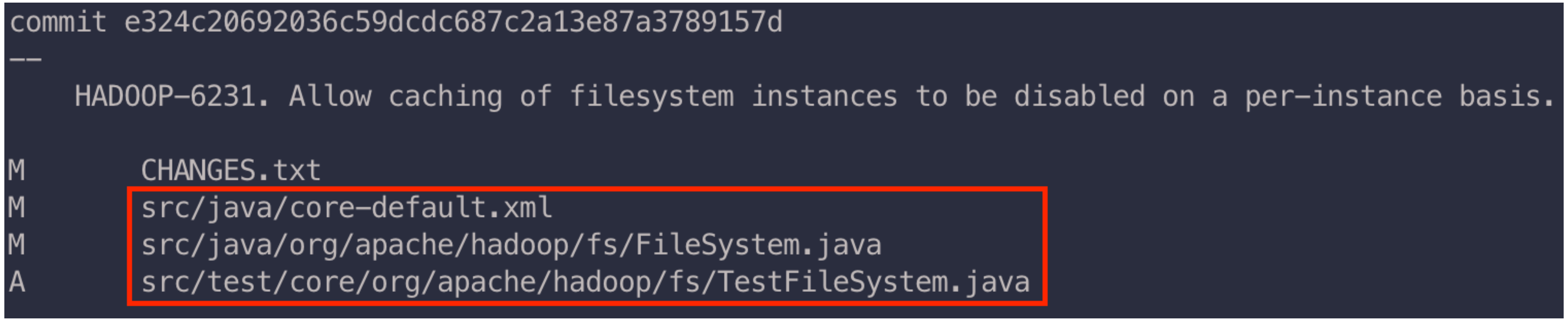
[1] Xudong Sun, et.al. Testing Configuration Changes in Context to Prevent Production Failures. OSDI 2020

The need for a *unified* testing technique

- Code and configuration are constantly changed together
- Existing testing techniques test code and configuration separately
  - Cannot address co-evolution of code and configurations
  - Cannot catch defects due to inconsistent code and configuration changes

The need for a *unified* testing technique

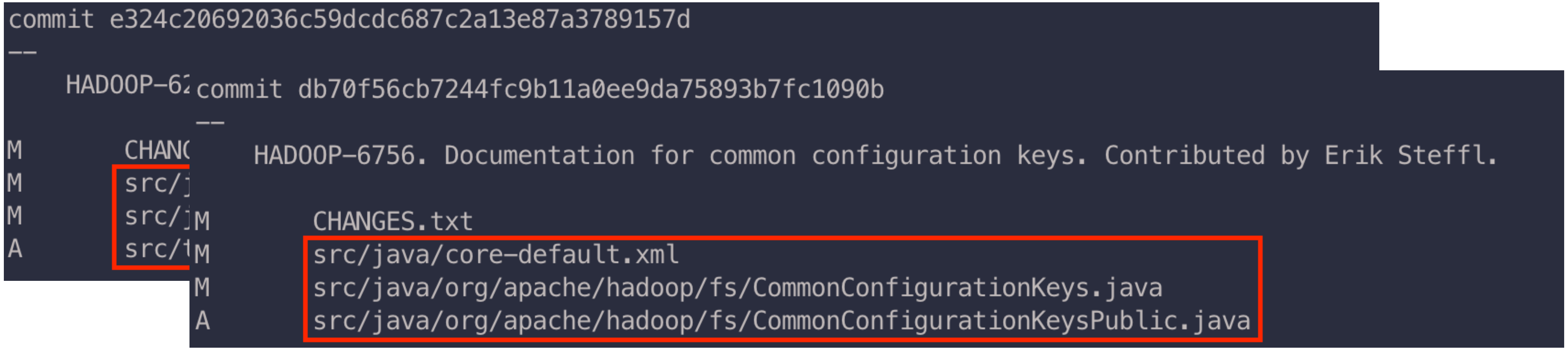
- Code and configuration are constantly changed together
- Existing testing techniques test code and configuration separately
  - Cannot address co-evolution of code and configurations
  - Cannot catch defects due to inconsistent code and configuration changes



```
commit e324c20692036c59dcdc687c2a13e87a3789157d
--
    HADOOP-6231. Allow caching of filesystem instances to be disabled on a per-instance basis.

M       CHANGES.txt
M       src/java/core-default.xml
M       src/java/org/apache/hadoop/fs/FileSystem.java
A       src/test/core/org/apache/hadoop/fs/TestFileSystem.java
```

The need for a *unified* testing technique

- Code and configuration are constantly changed together
- Existing testing techniques test code and configuration separately
    - Cannot address co-evolution of code and configurations
    - Cannot catch defects due to inconsistent code and configuration changes

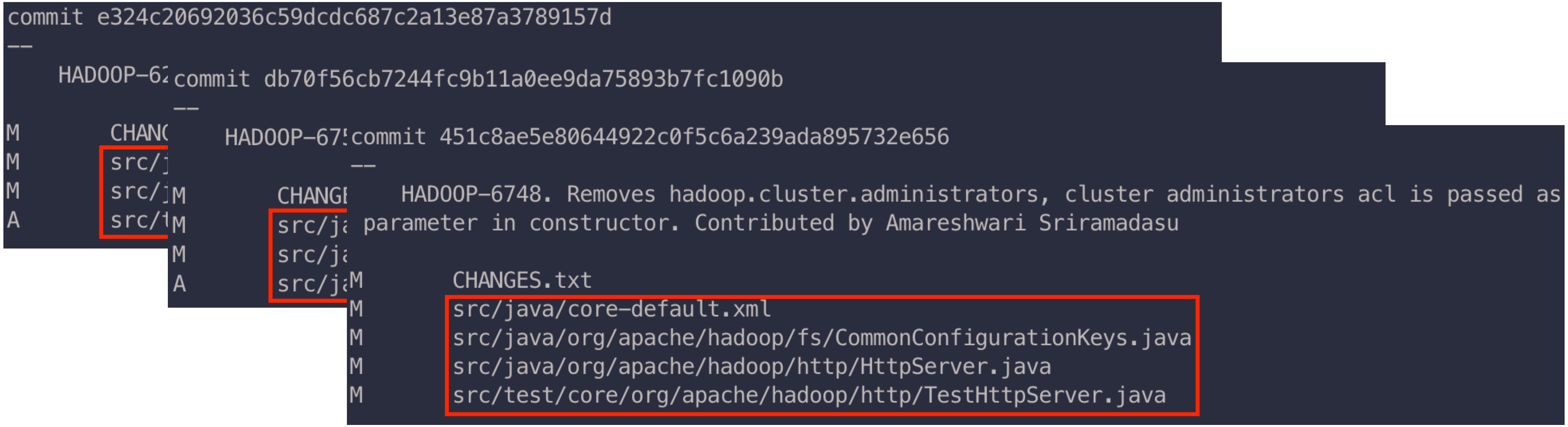


```
commit e324c20692036c59dcdc687c2a13e87a3789157d
--
    HADOOP-62

M       CHANC
M       src/j
M       src/j
A       src/i
```

```
commit db70f56cb7244fc9b11a0ee9da75893b7fc1090b
--
    HADOOP-6756. Documentation for common configuration keys. Contributed by Erik Steffl.

M       CHANGES.txt
M       src/java/core-default.xml
M       src/java/org/apache/hadoop/fs/CommonConfigurationKeys.java
A       src/java/org/apache/hadoop/fs/CommonConfigurationKeysPublic.java
```

The need for a *unified* testing technique

- Code and configuration are constantly changed together
- Existing testing techniques test code and configuration separately
    - Cannot address co-evolution of code and configurations
    - Cannot catch defects due to inconsistent code and configuration changes



```
commit e324c20692036c59dcdc687c2a13e87a3789157d
--
    HADOOP-62

M       CHANC
M       src/j
M       src/j
A       src/t
```

```
commit db70f56cb7244fc9b11a0ee9da75893b7fc1090b
--
    HADOOP-67

M       CHANGE
M       src/ja
M       src/ja
A       src/ja
```

```
commit 451c8ae5e80644922c0f5c6a239ada895732e656
--
    HADOOP-6748. Removes hadoop.cluster.administrators, cluster administrators acl is passed as
 parameter in constructor. Contributed by Amareshwari Sriramadasu

M       CHANGES.txt
M       src/java/core-default.xml
M       src/java/org/apache/hadoop/fs/CommonConfigurationKeys.java
M       src/java/org/apache/hadoop/http/HttpServer.java
M       src/test/core/org/apache/hadoop/http/TestHttpServer.java
```

Contributions

- **Concept: Unified Regression Testing (URT)** for checking both code and configuration changes
- **Algorithm: Unified Regression Test Selection (uRTS)** for speeding up URT, with the same safety guarantee
- **Implementation**: Implemented on top of Ekstazi and OpenCtest
- **Evaluation** with five large software projects
  - Hundreds of code revisions and dozens of configuration files
  - Largest RTS experiments performed on open-source projects
- **Data/code release:** https://github.com/xlab-uiuc/uRTS-ae

Unified regression testing is costly

- URT generalizes traditional regression testing and config testing
- Testing </> changes under not only default ⚙ but also production ⚙
- Testing ⚙ changes against the new </>
- Handling diffs that co-change both </> and ⚙
- May run each change multiple times under different ⚙
  - Existing regression testing (on default config) is already expensive!

</> **Source code**

⚙ **Configuration**

uRTS to the rescue

- Minimizing the number of tests to run for a given diff
    - Could change code only, config only, or both code and config

uRTS to the rescue

- Minimizing the number of tests to run for a given diff
  - Could change code only, config only, or both code and config

Change	Regular Test	Config Test
Only code change	Run	Run
Only default config	Run	Skip
Only production config	Skip	Run
Code + default config	Run	Run
Code + production config	Run	Run
Code + default and prod config	Run	Run

uRTS to the rescue

- Minimizing the number of tests to run for a given diff
  - Could change code only, config only, or both code and config

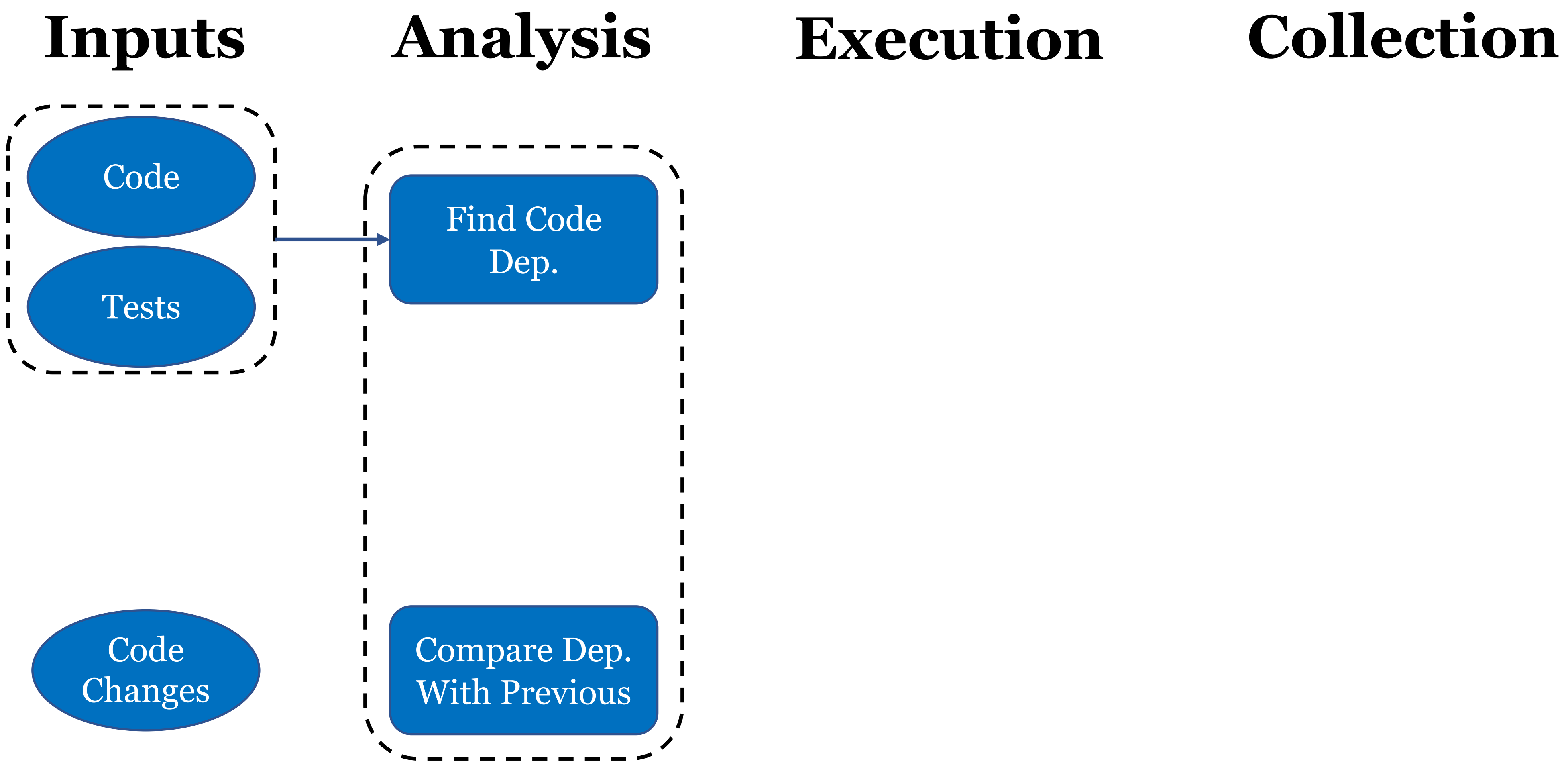
- **Key observation -- changes are typically small**
  - A diff changes a small piece of code, or a small number of config values
  - A production config changes only a small number of default config values
  - The production configs only differ in a small number of values

Traditional RTS

Traditional RTS

Analysis **Execution** **Collection**

Traditional RTS



Inputs

Analysis

Execution

Collection

Code

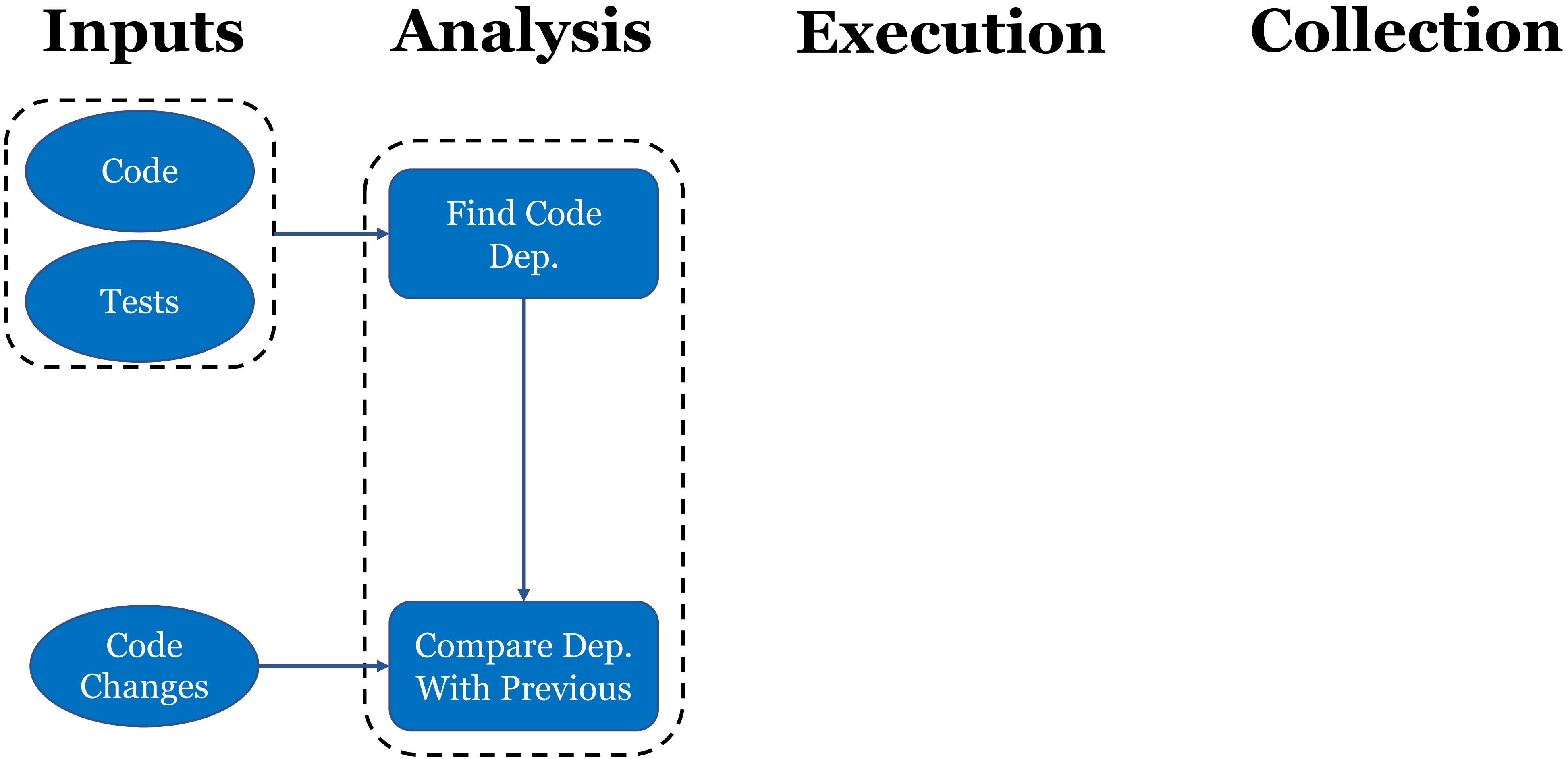
Tests

Find Code Dep.

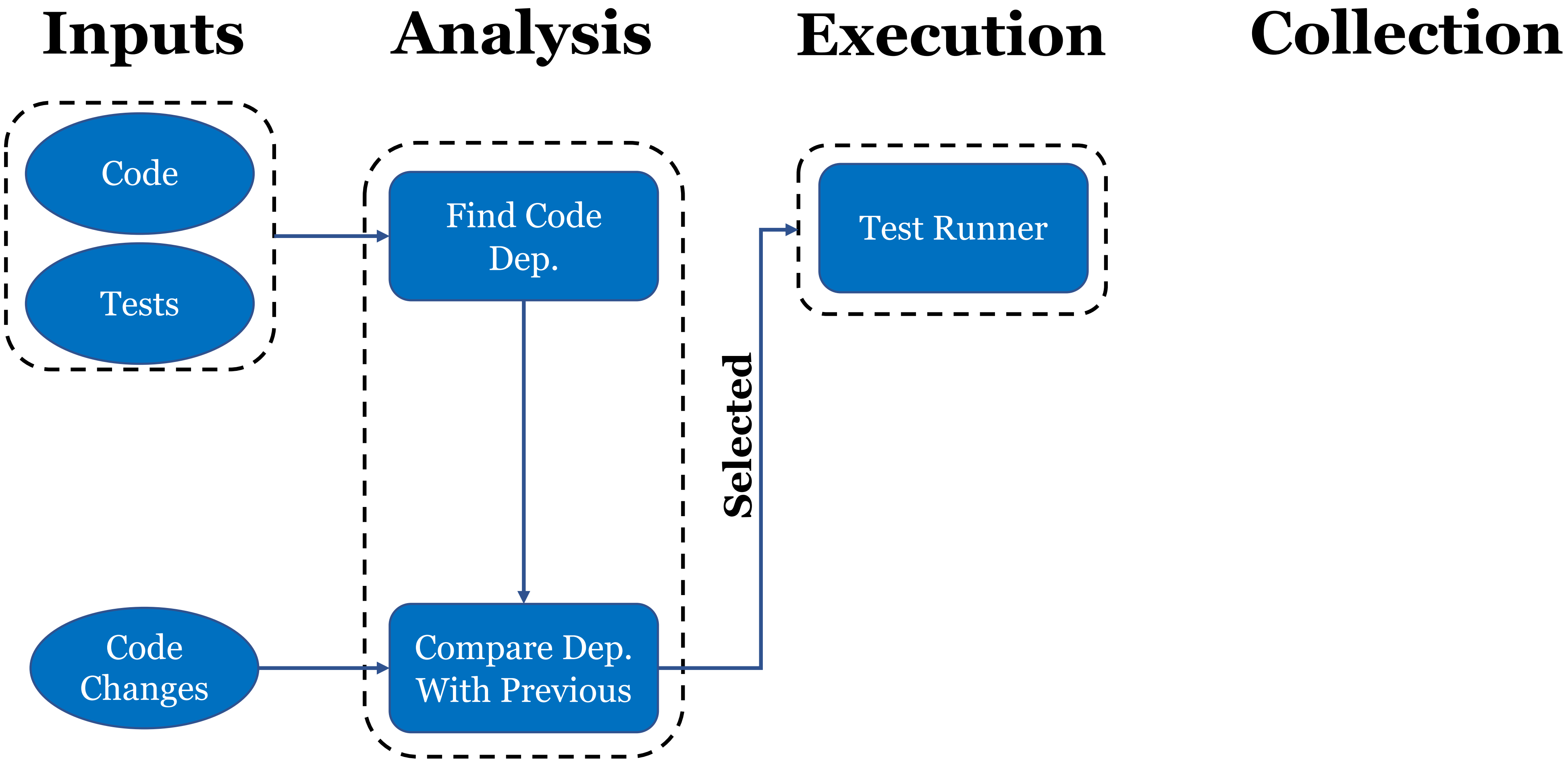
Code Changes

Compare Dep. With Previous

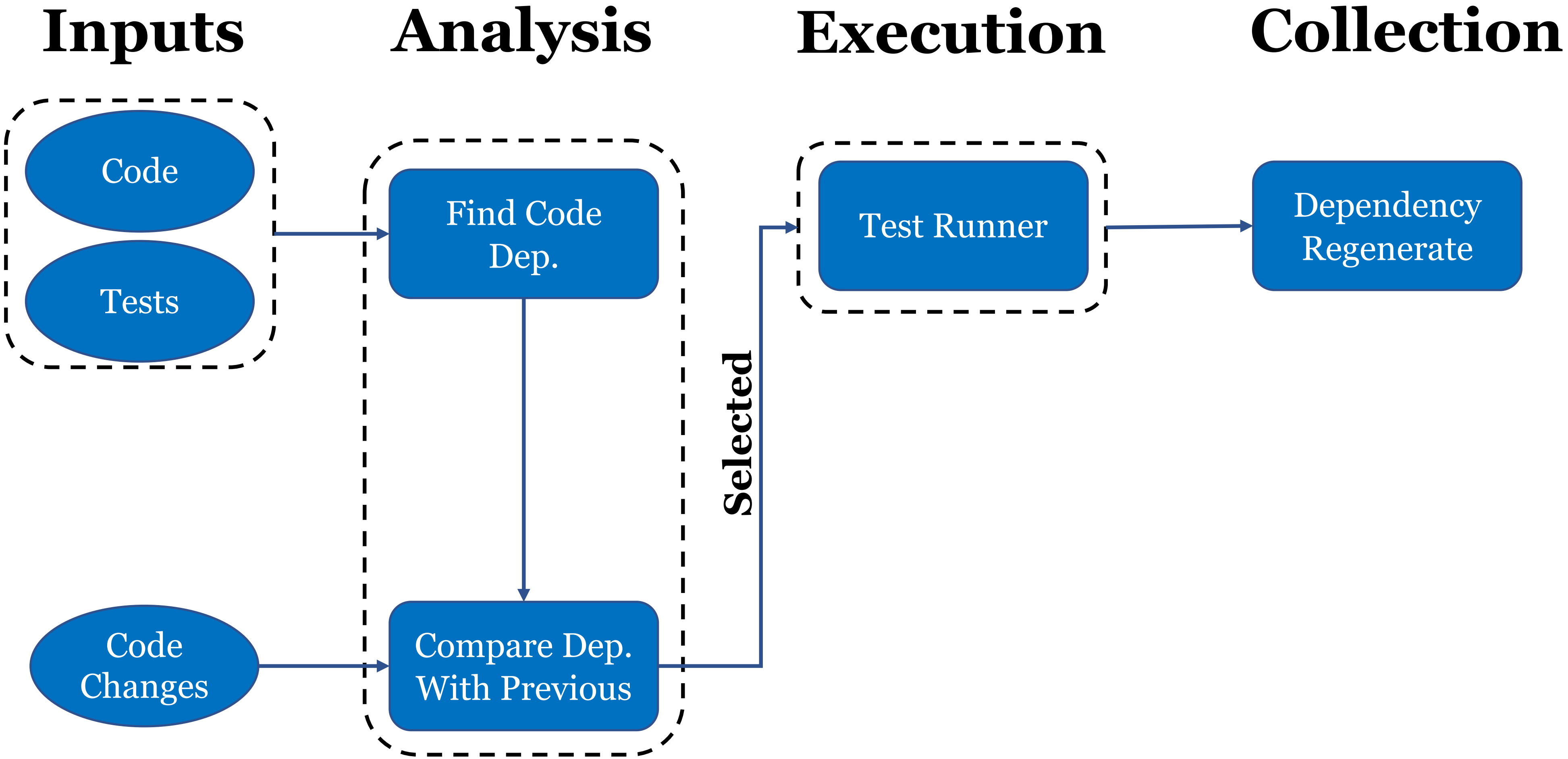
Traditional RTS



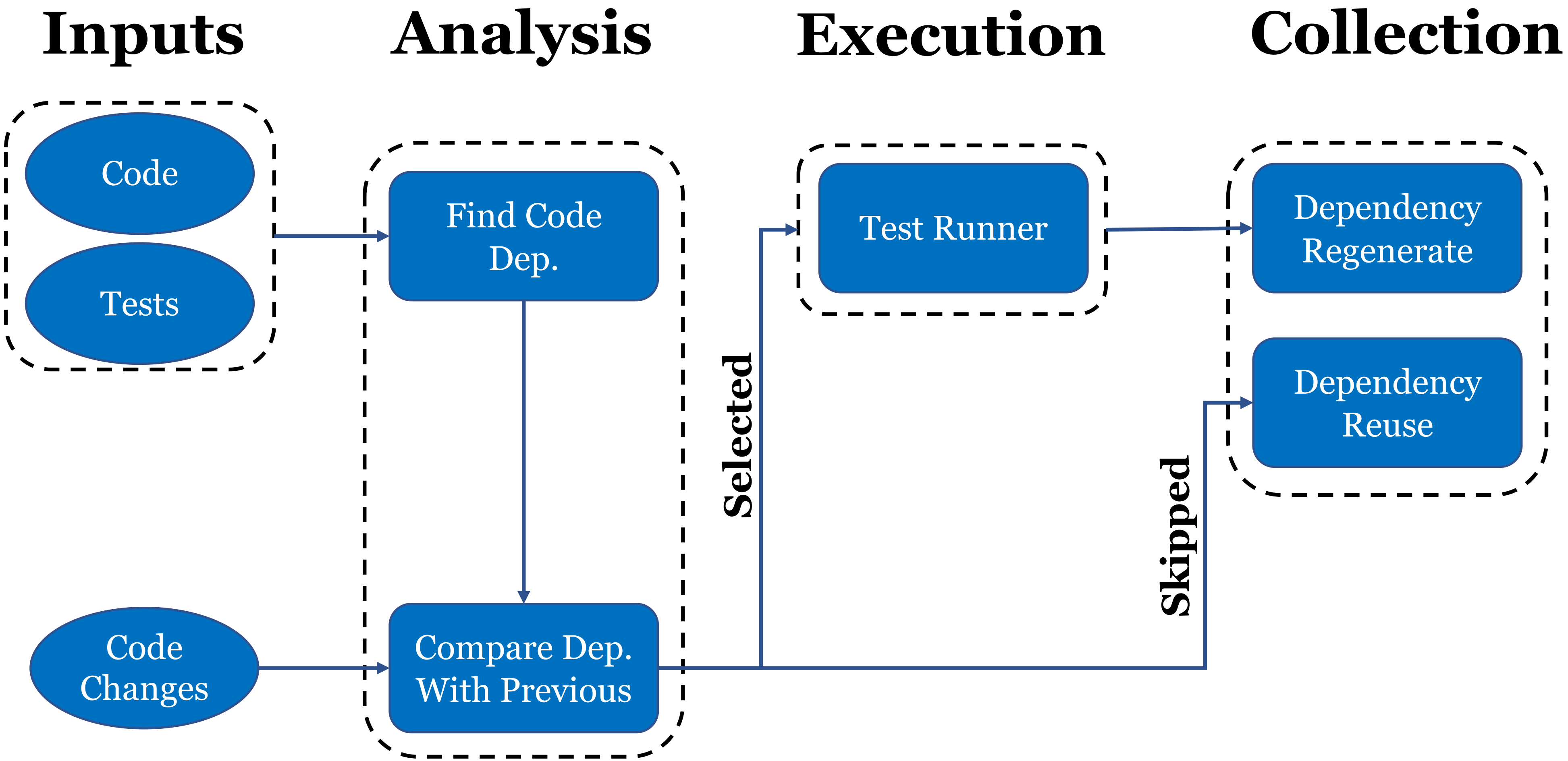
Traditional RTS



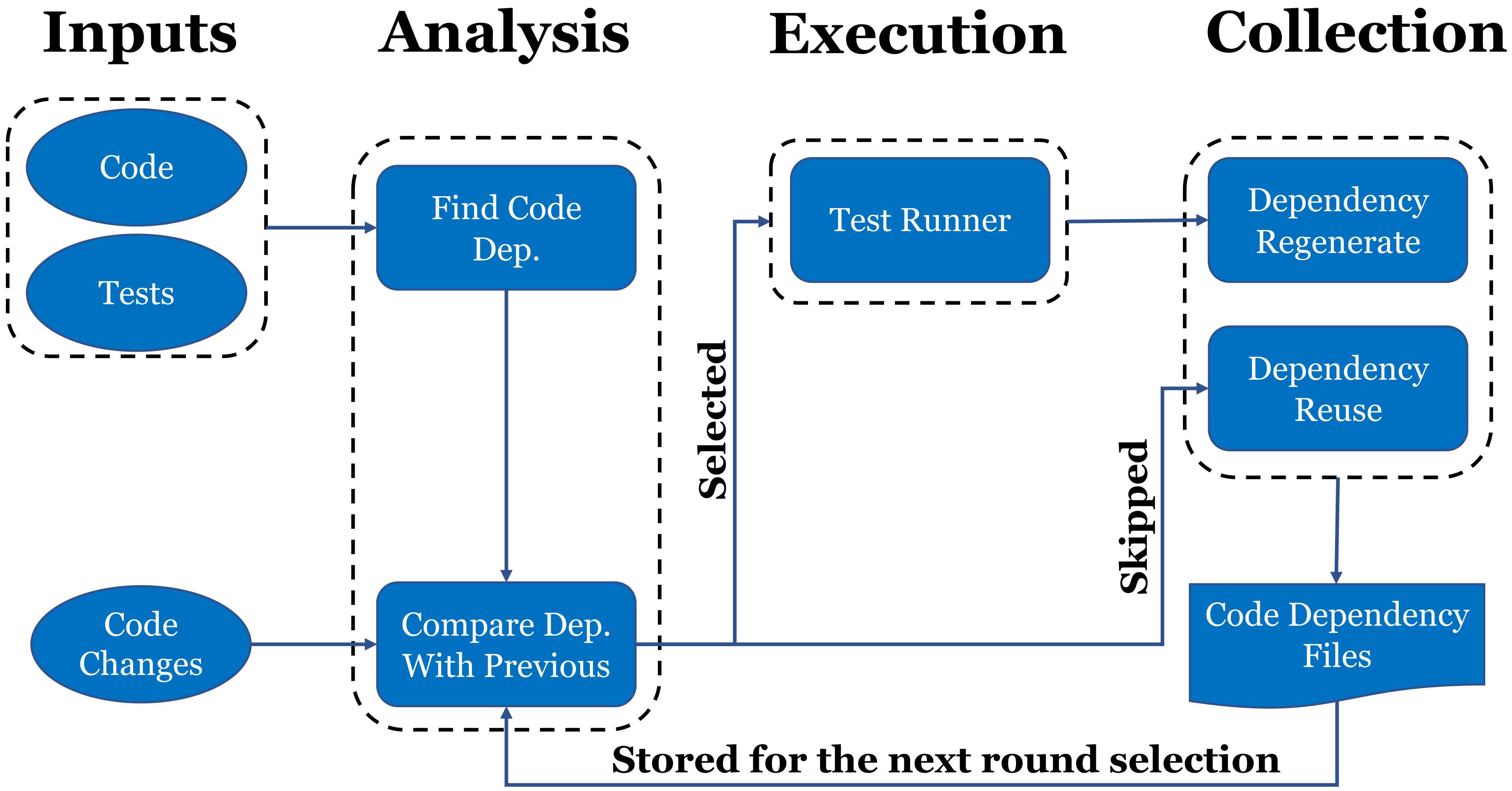
Traditional RTS



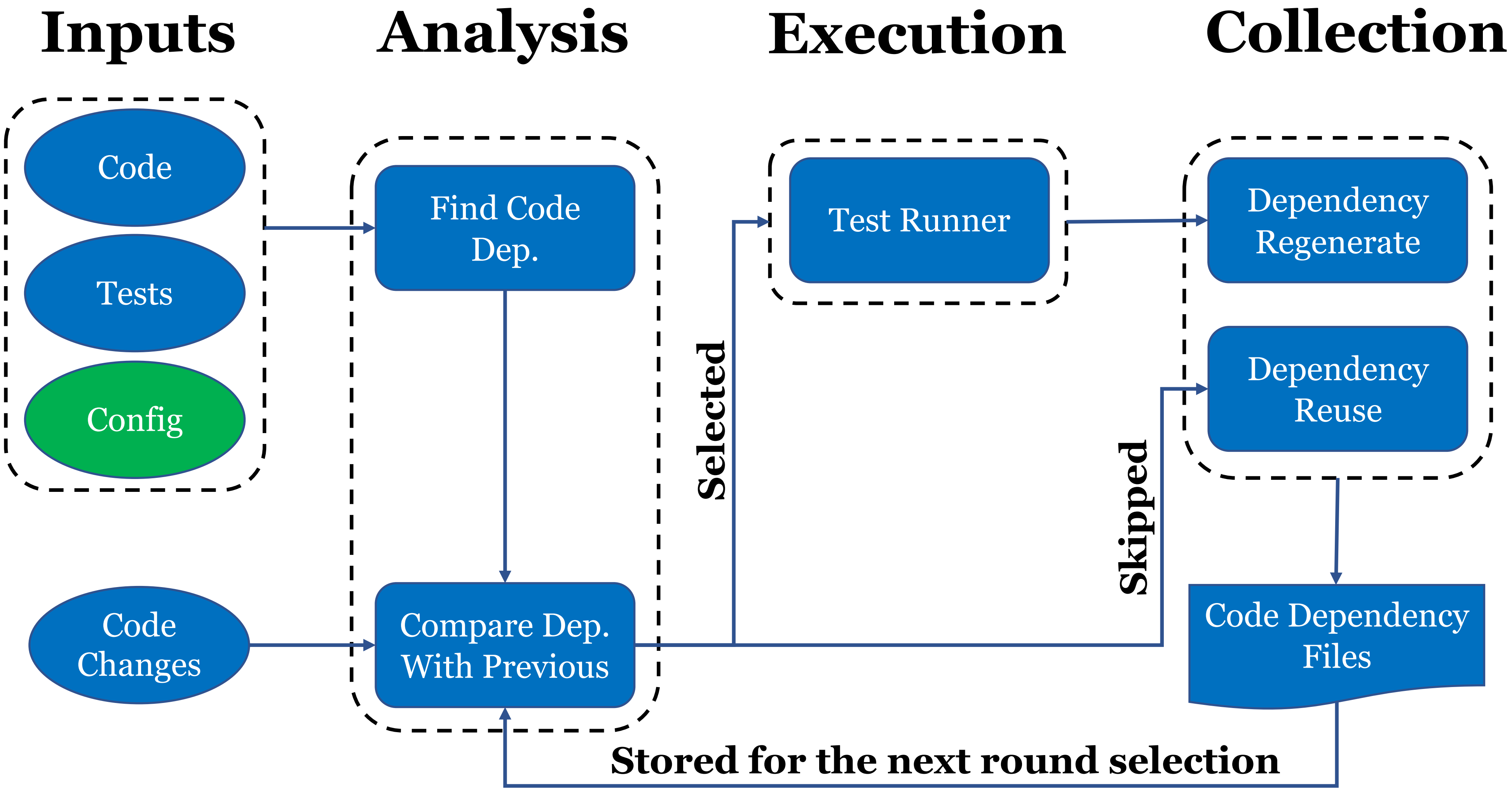
Traditional RTS



Traditional RTS

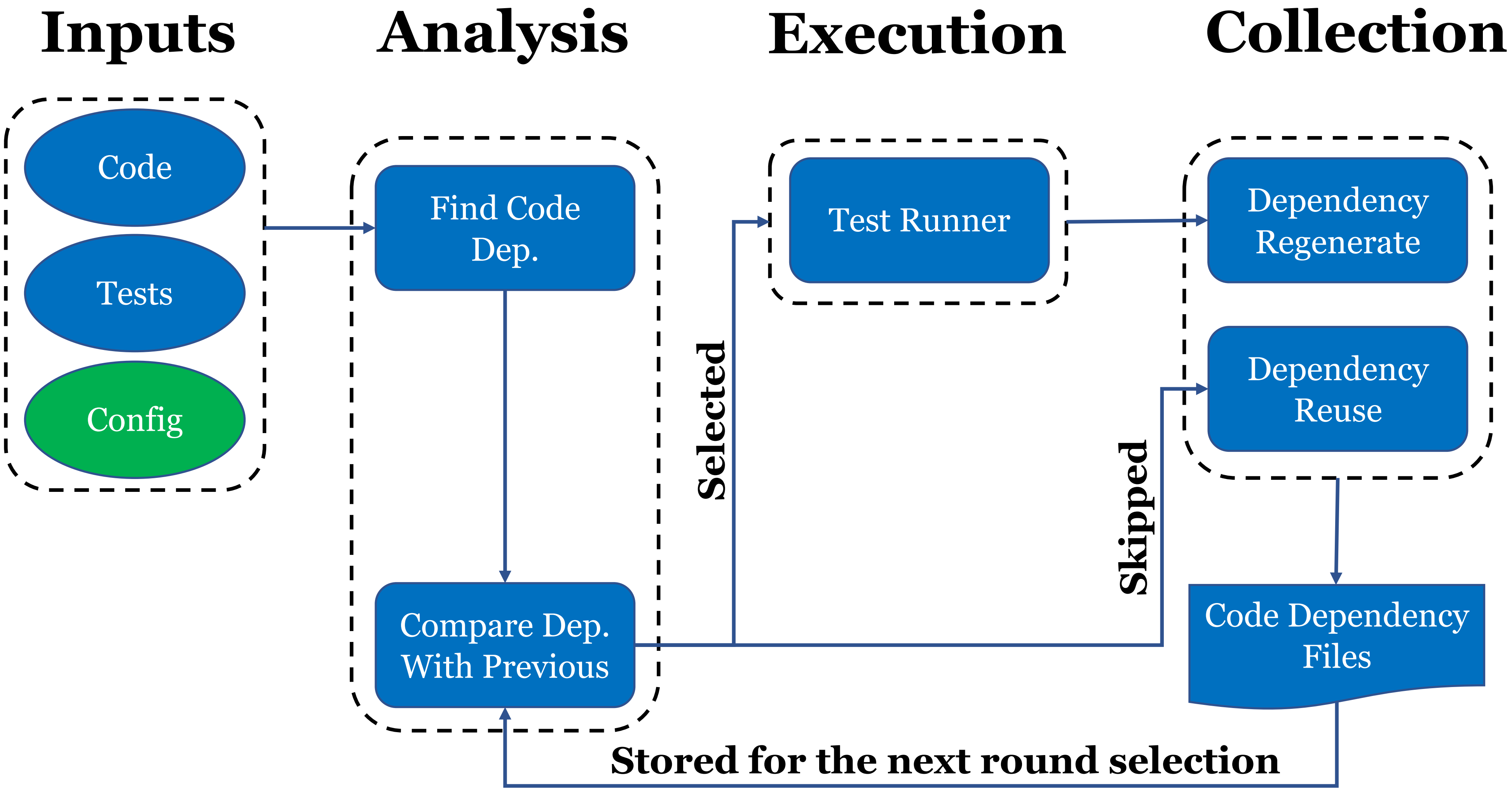


uRTS Implementation



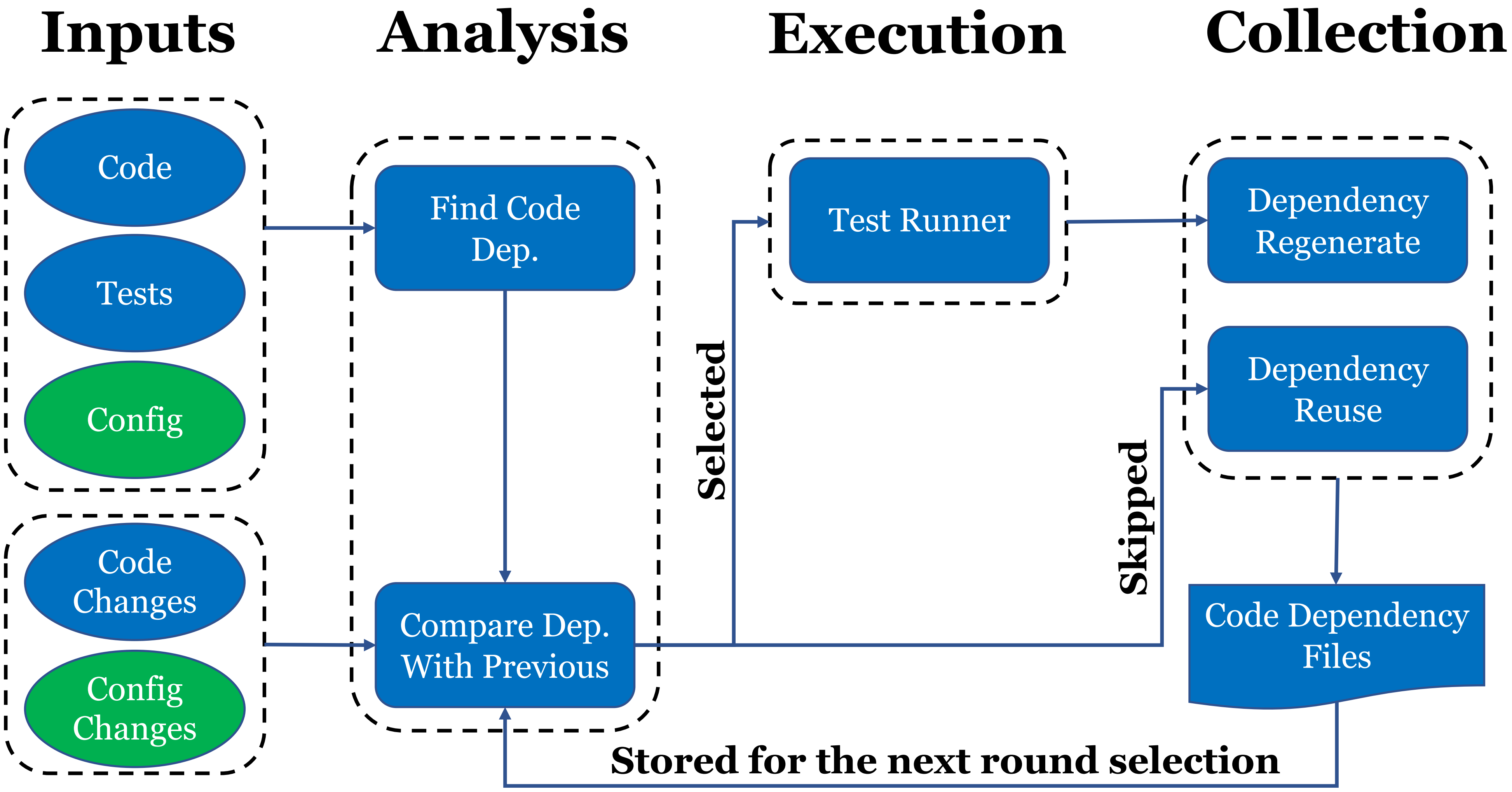
Green color represents the changes in uRTS compared with traditional RTS

uRTS Implementation



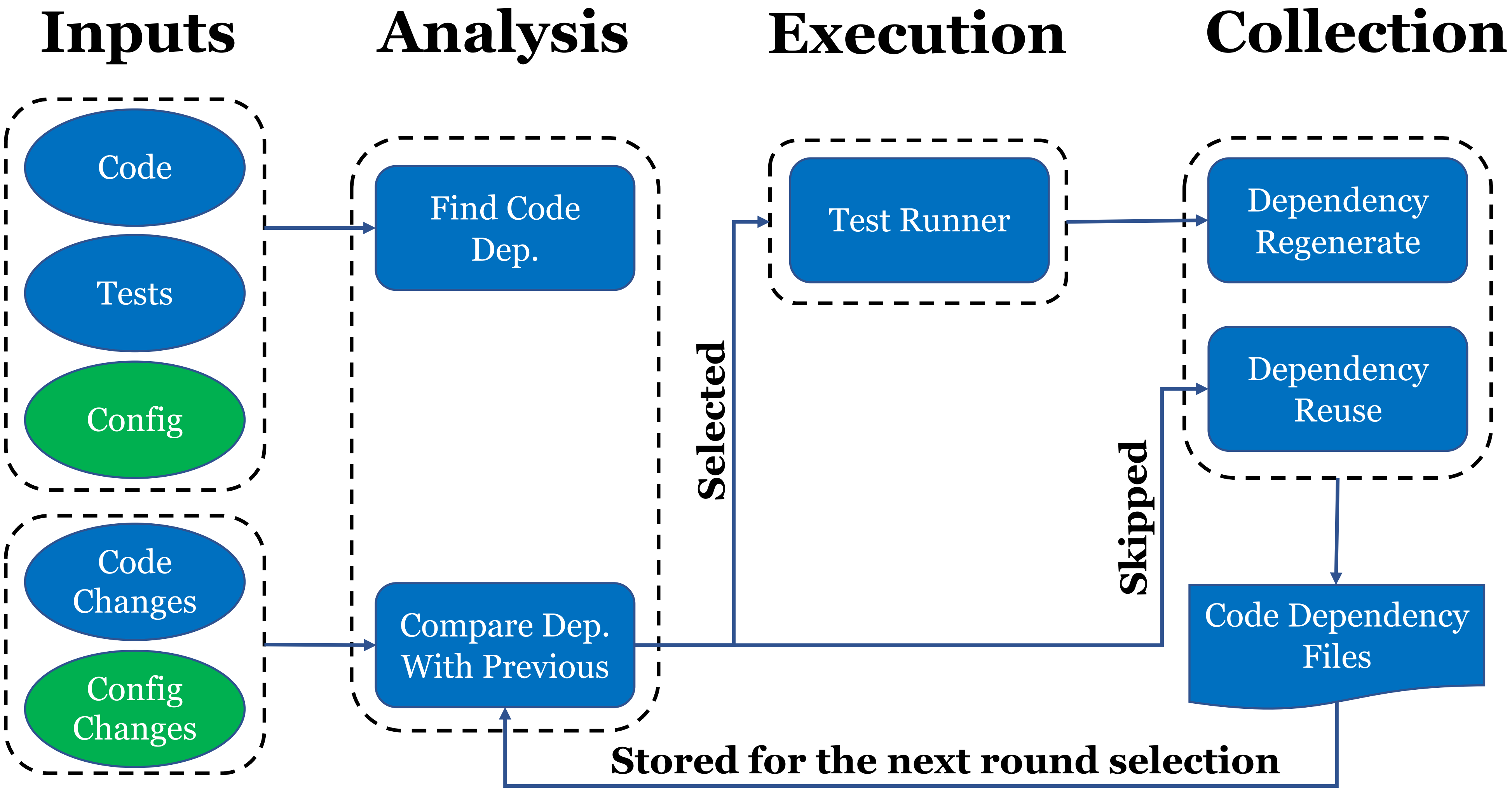
Green color represents the changes in uRTS compared with traditional RTS

uRTS Implementation



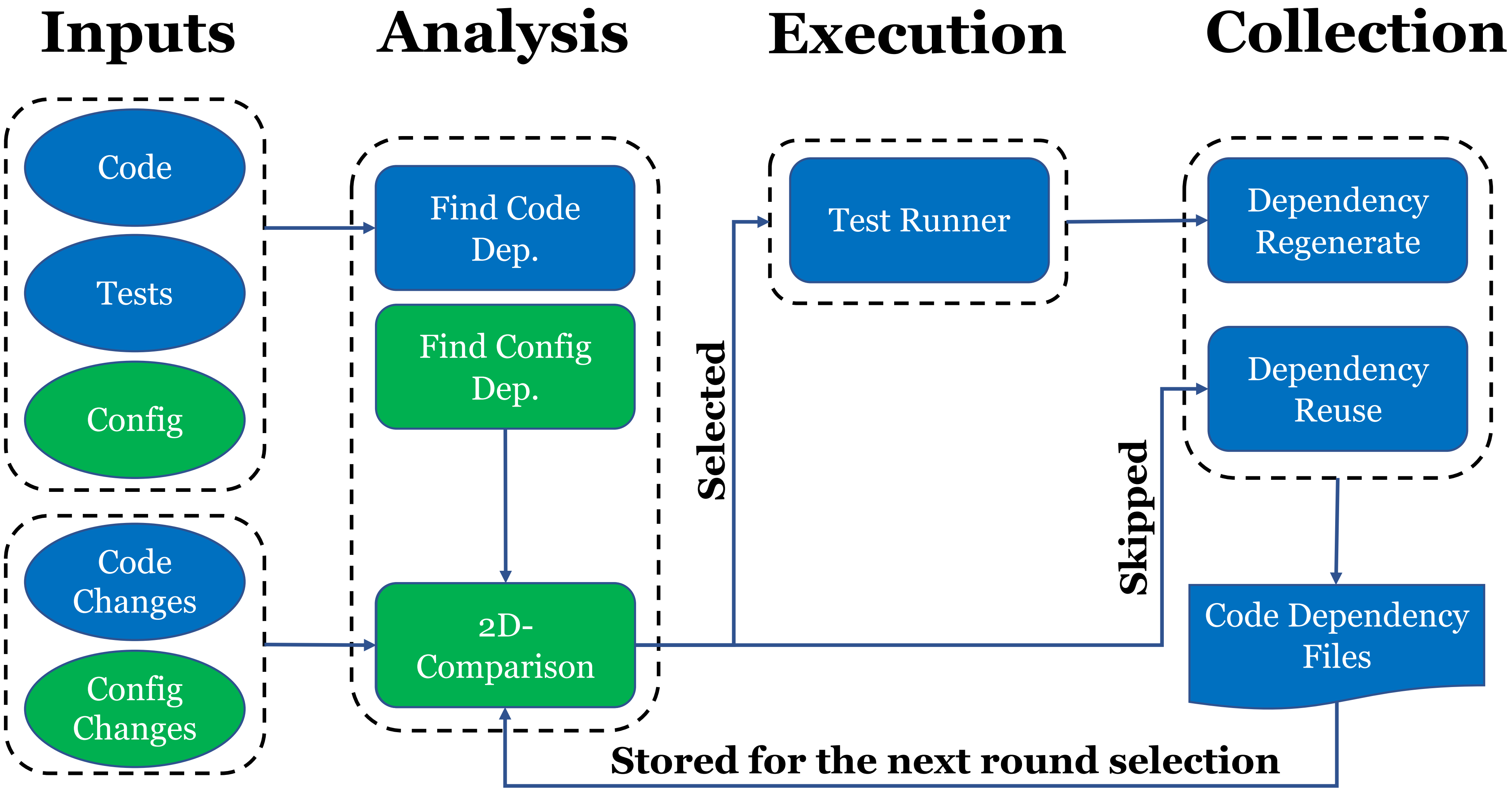
Green color represents the changes in uRTS compared with traditional RTS

uRTS Implementation



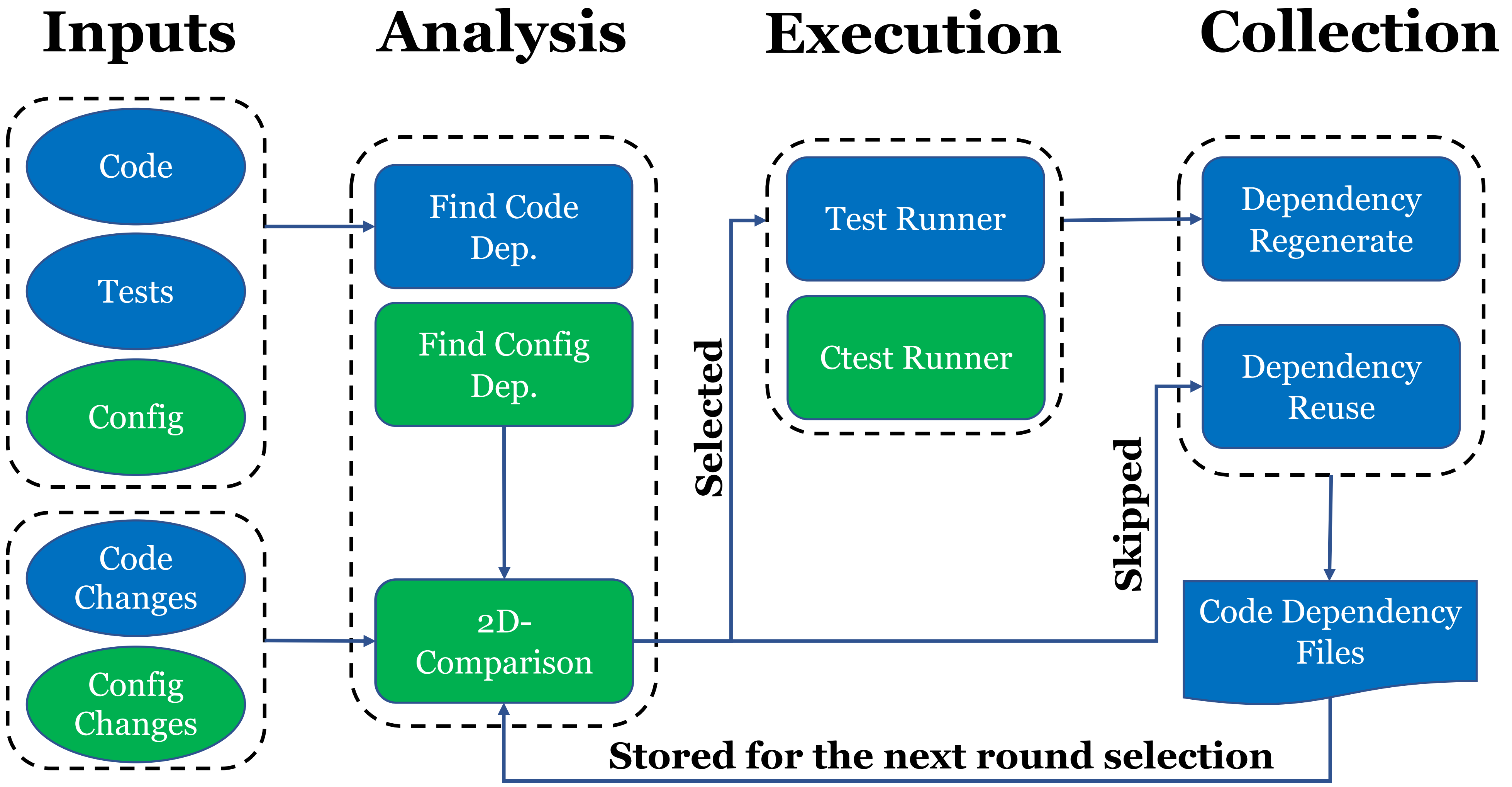
Green color represents the changes in uRTS compared with traditional RTS

uRTS Implementation



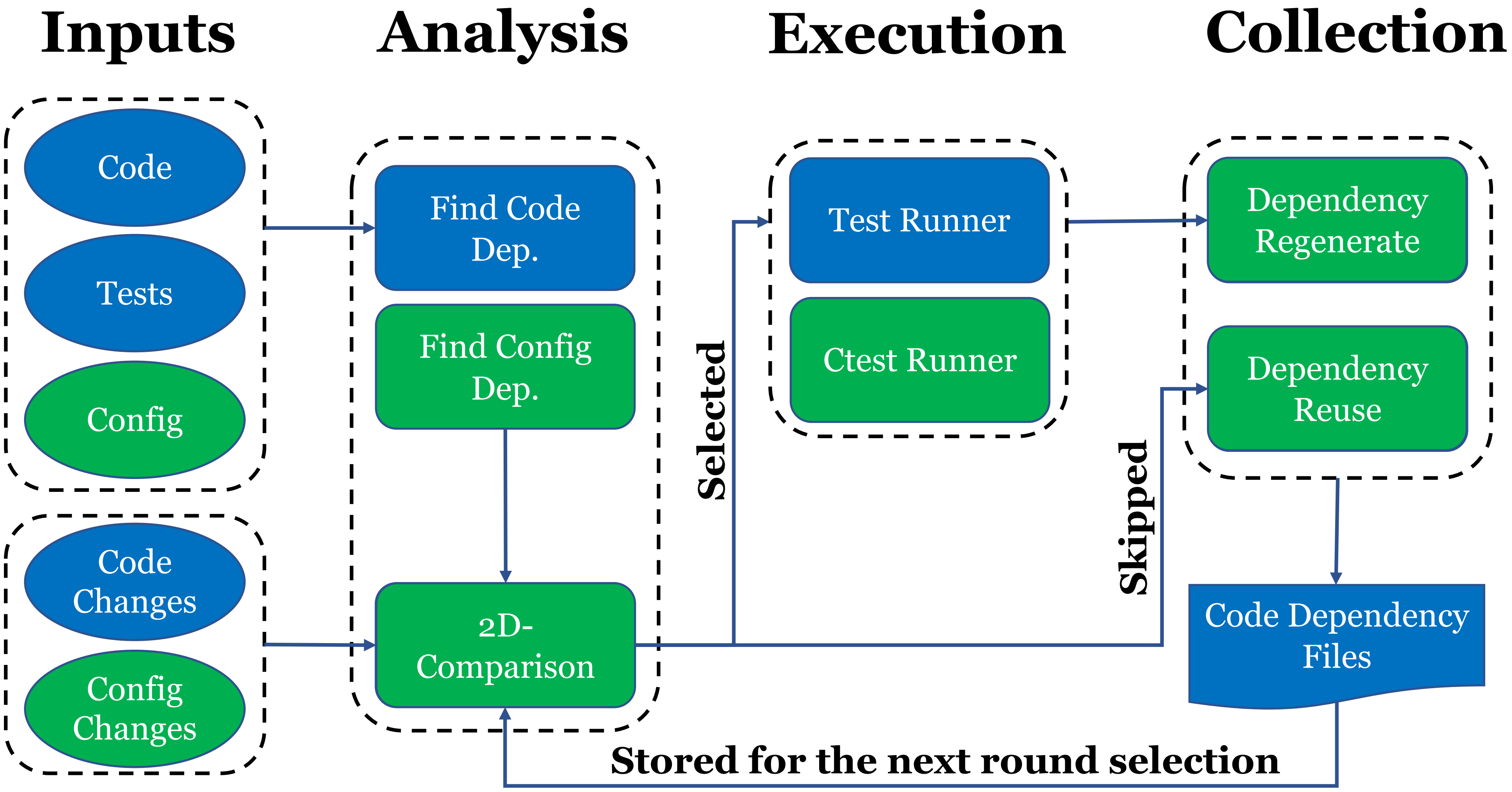
Green color represents the changes in uRTS compared with traditional RTS

uRTS Implementation



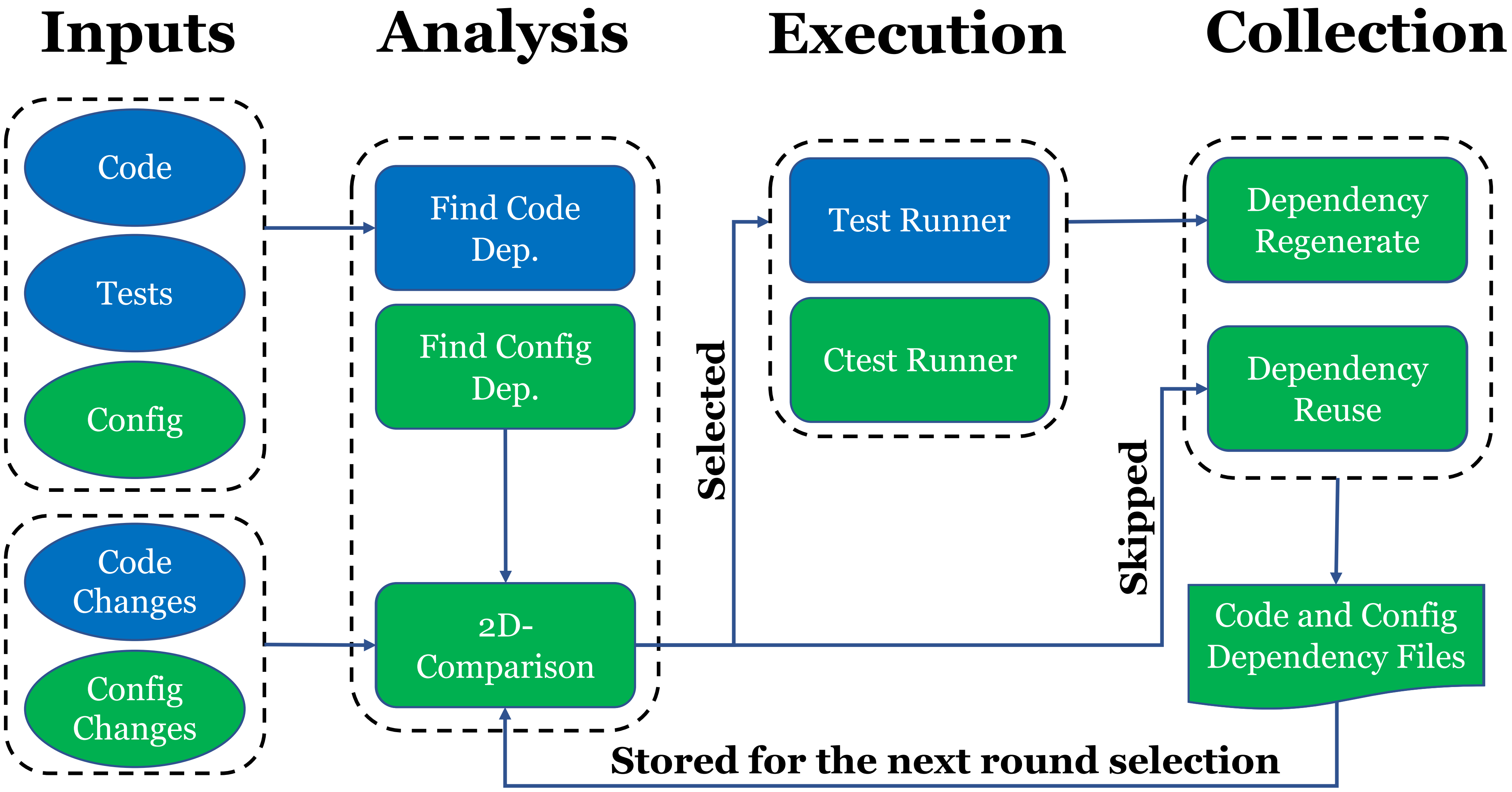
Green color represents the changes in uRTS compared with traditional RTS

uRTS Implementation



Green color represents the changes in uRTS compared with traditional RTS

uRTS Implementation



Green color represents the changes in uRTS compared with traditional RTS

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.

uRTS tracks code and config dependencies

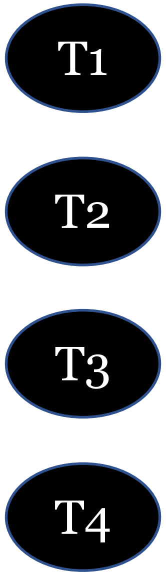
- Dependencies are entities that can affect test behavior.

Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.



T1

T2

T3

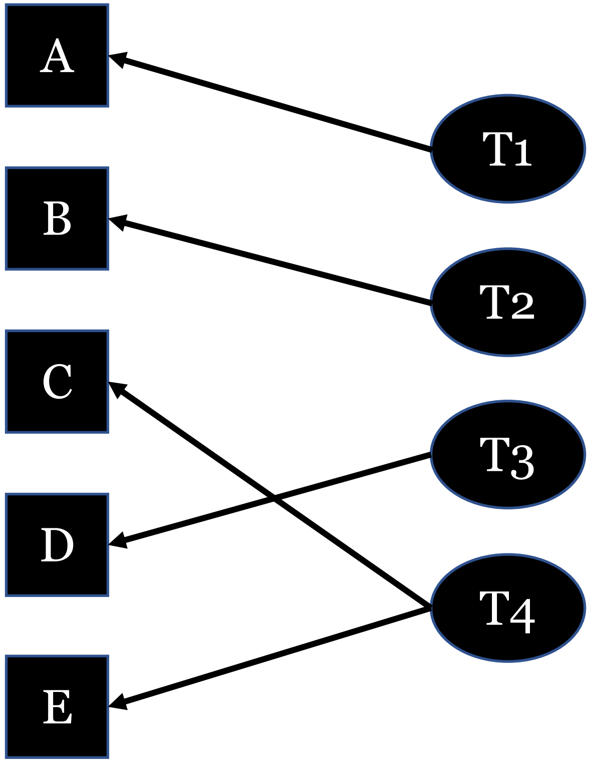
T4

Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.

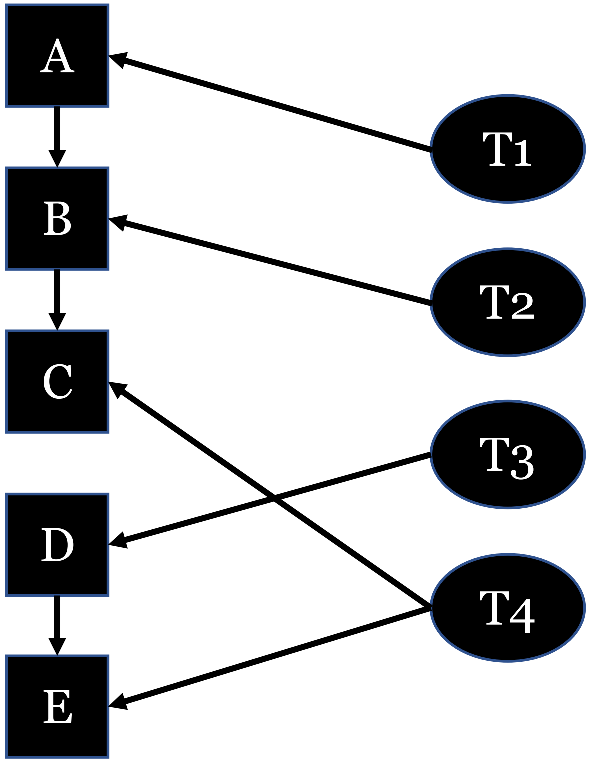


Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.

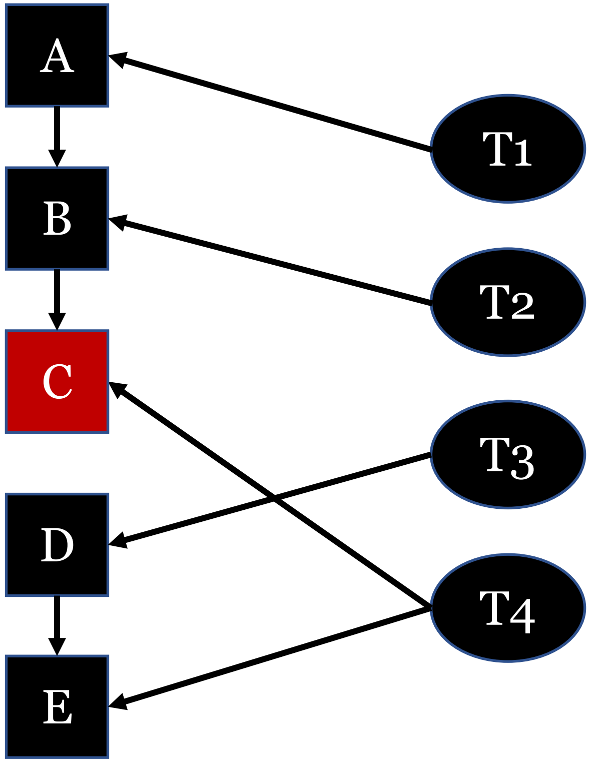


Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.

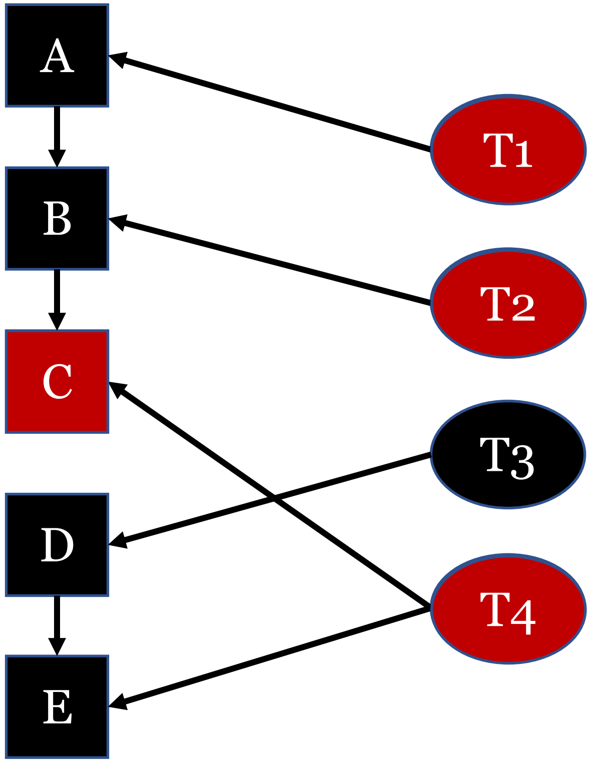


Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.

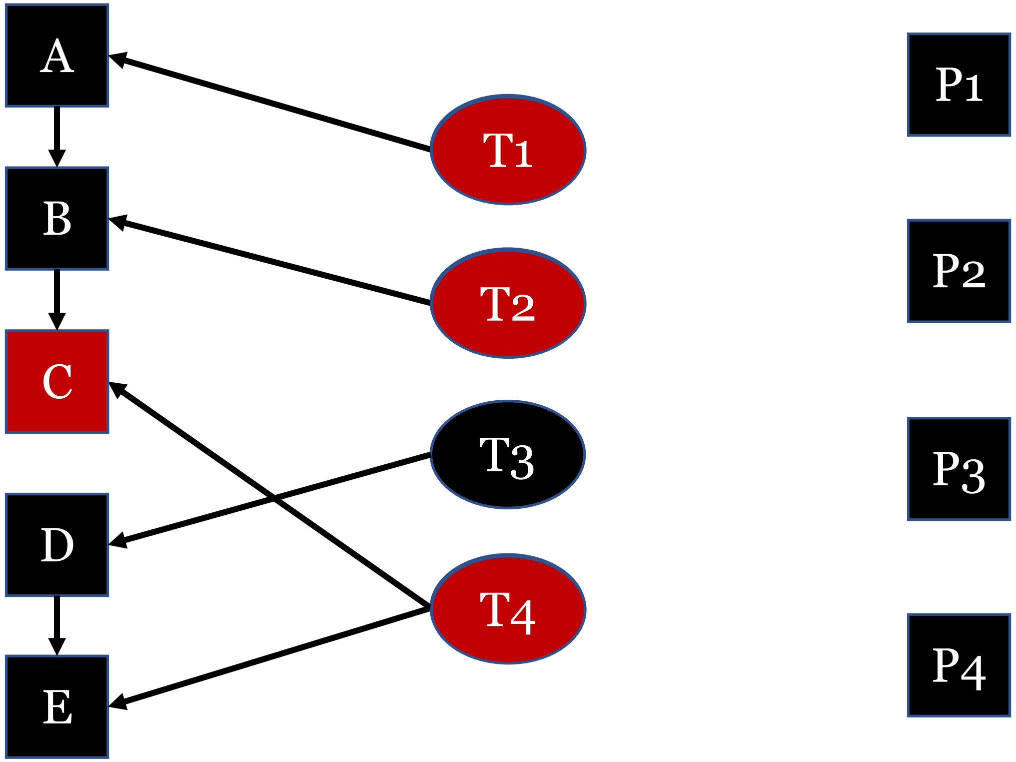


Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.

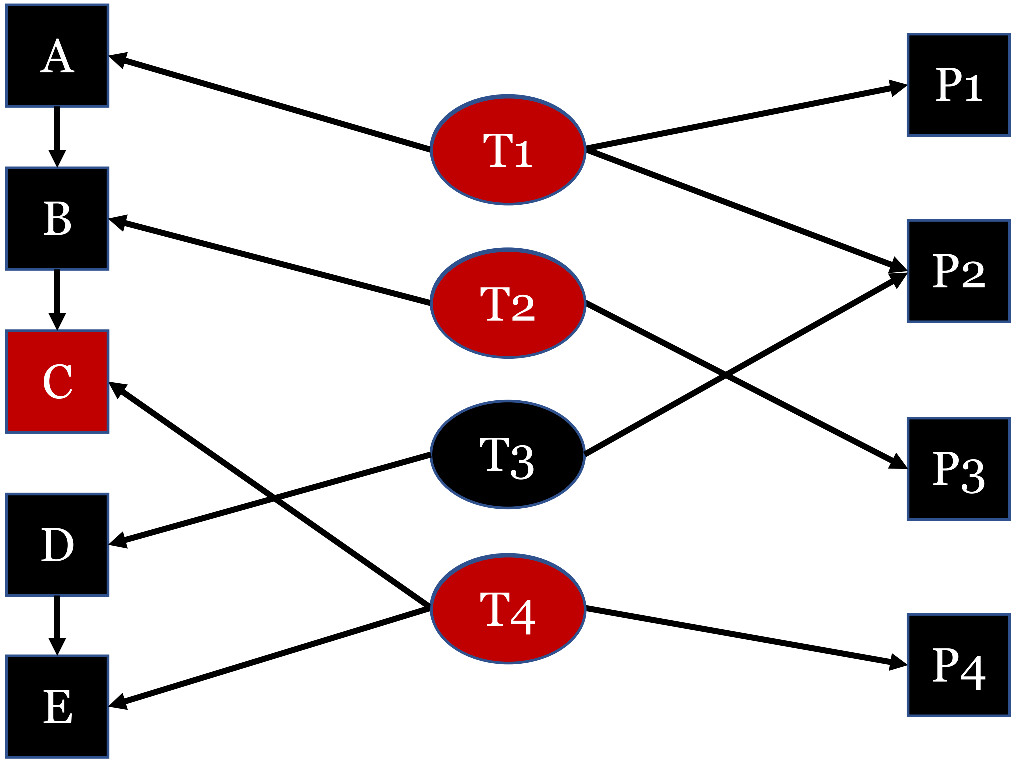


Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.

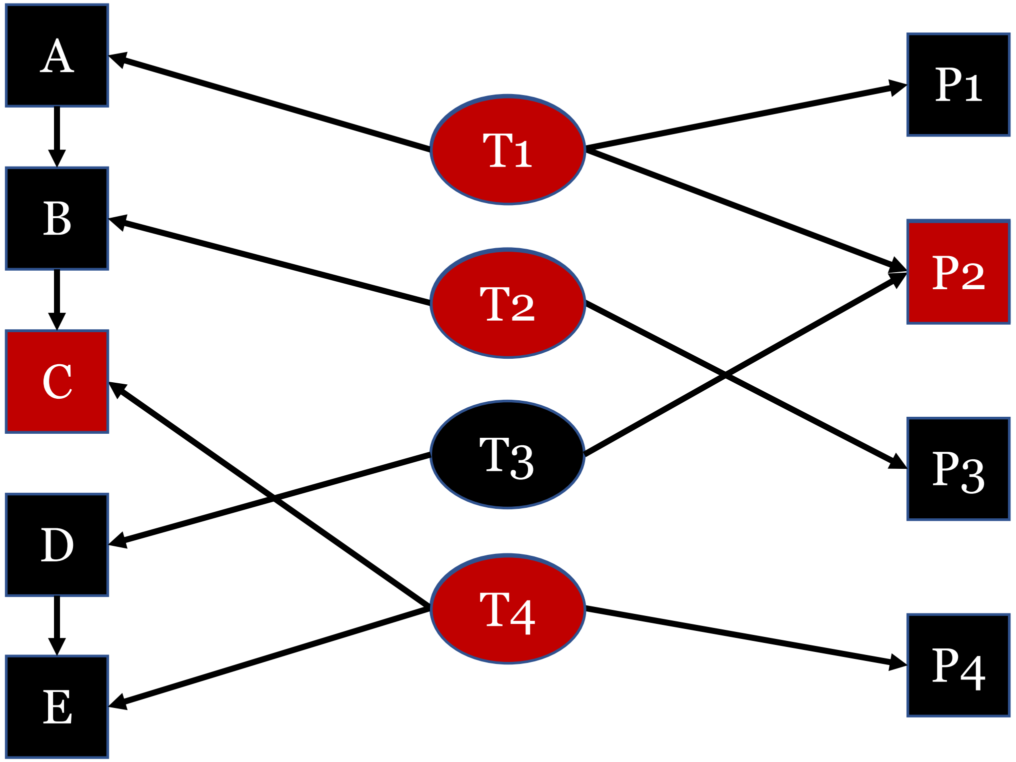


Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.

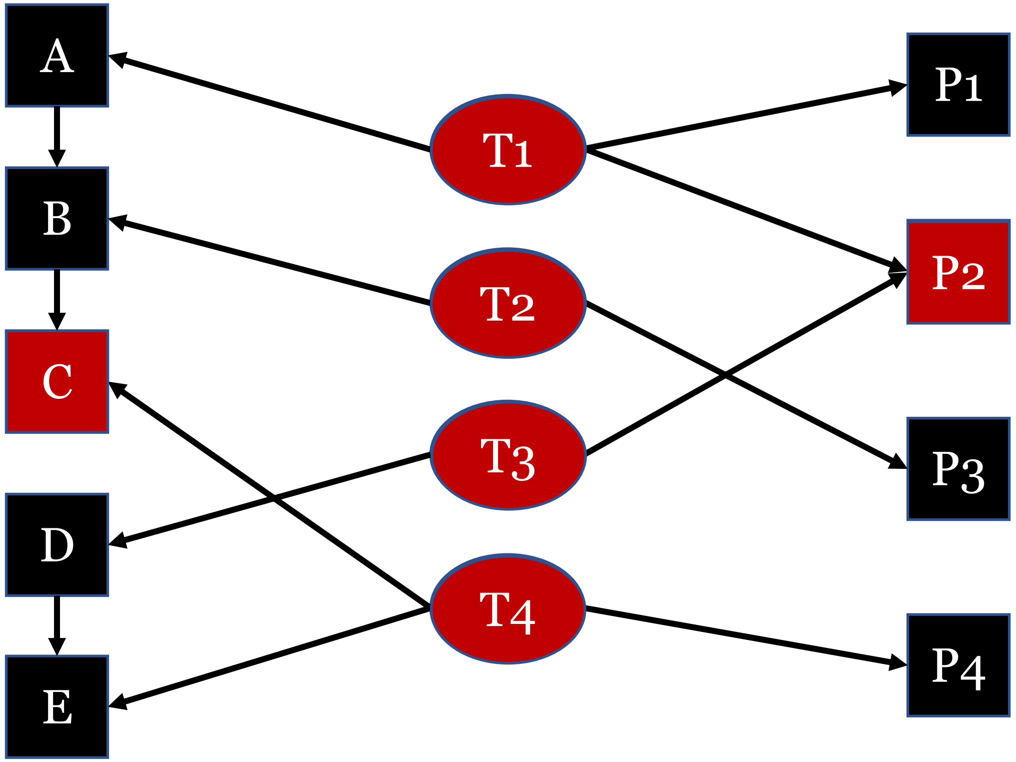


Code dependencies

Configuration dependencies

uRTS tracks code and config dependencies

- Dependencies are entities that can affect test behavior.



Code dependencies

Configuration dependencies

2D comparison for Ctest selection

- Goal: select as **few** tests as possible while keeping **safety**

2D comparison for Ctest selection

- Goal: select as **few** tests as possible while keeping **safety**

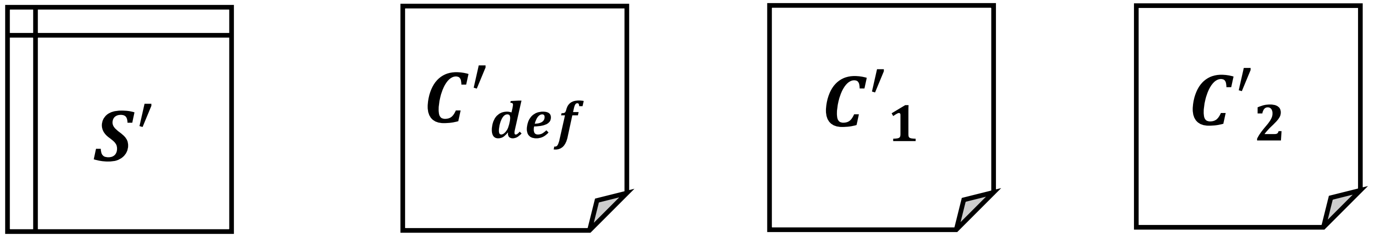
$\boldsymbol{ct\left(\widehat{P}\right)}$ **A configuration test ct parameterized by a set of configuration parameter $\boldsymbol{\widehat{P}}$**

2D comparison for Ctest selection

Code

Default config

Production configs



S'

C'_{def}

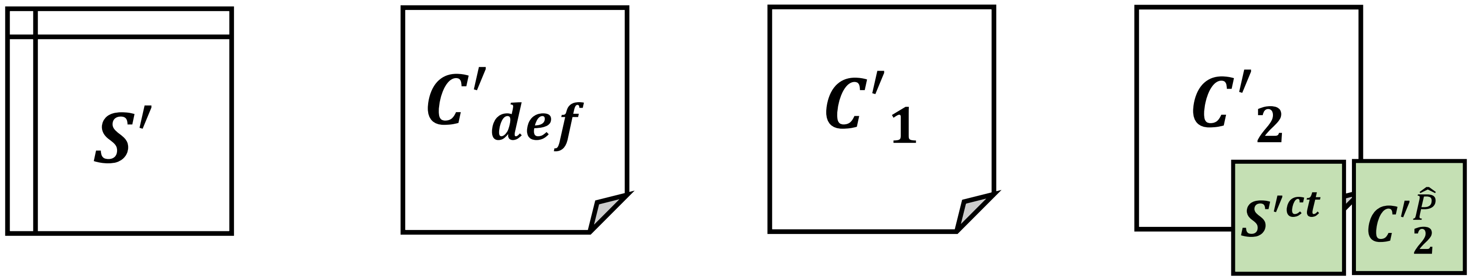
C'_1

C'_2

$ct(\widehat{P})$

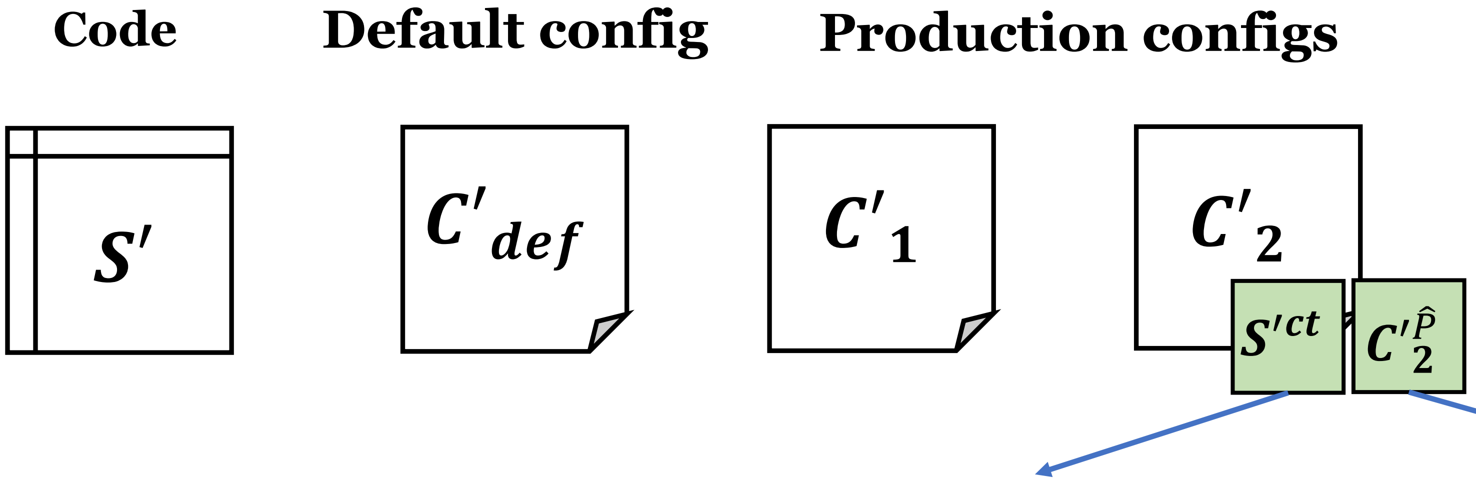
2D comparison for Ctest selection

Code **Default config** **Production configs**



$ct(\hat{P})$

2D comparison for Ctest selection

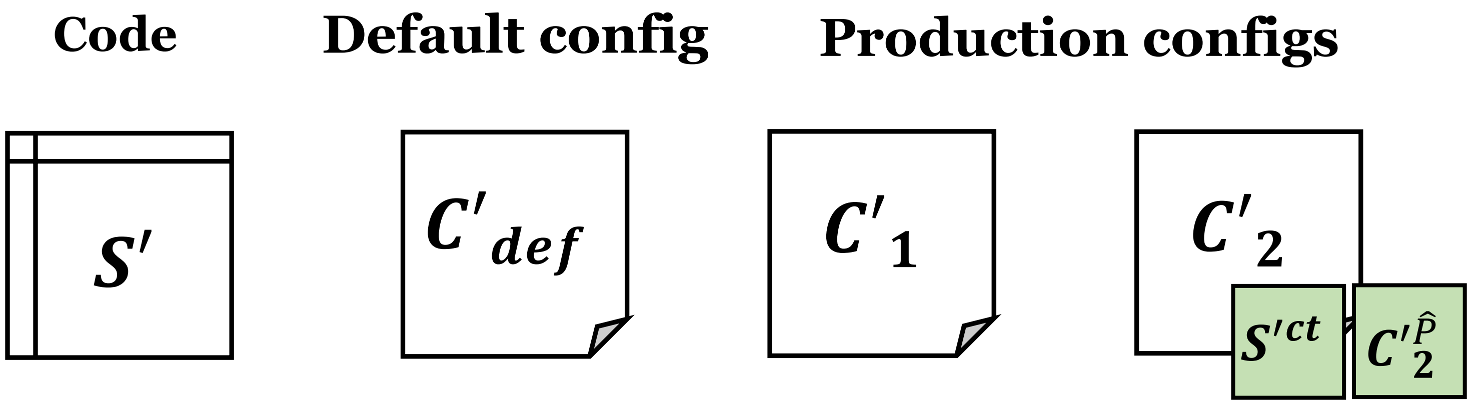


$ct(\hat{P})$

Code dependency for $ct(\hat{P})$ under code revision S’

Config dependency for $ct(\hat{P})$ with config set $\hat{P} \subseteq C'_2$

2D comparison for Ctest selection



Code

Default config

Production configs

S'

C'_{def}

C'_1

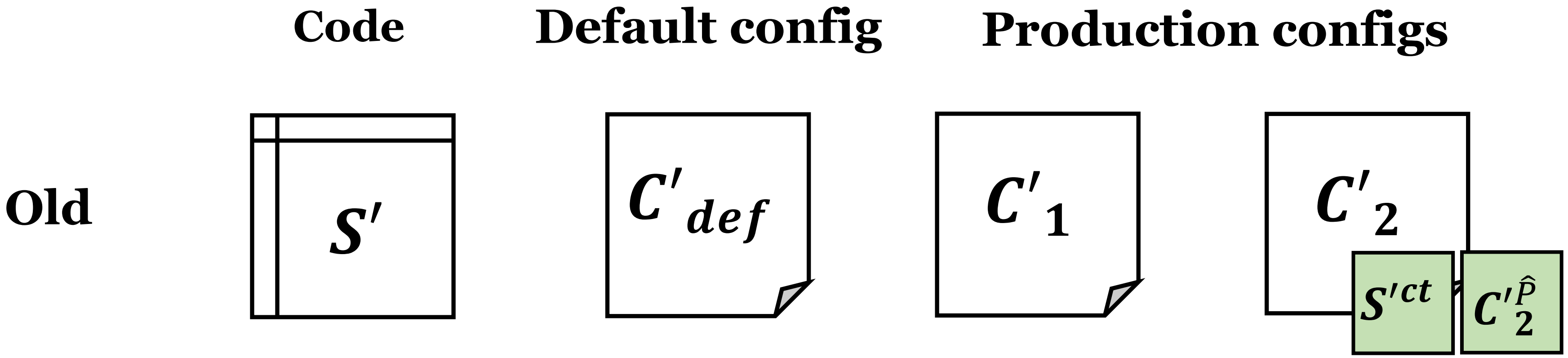
C'_2

S'^{ct}

$C'^{\hat{P}}_2$

$ct(\hat{P})$

2D comparison for Ctest selection



Code

Default config

Production configs

Old

S'

C'_{def}

C'_1

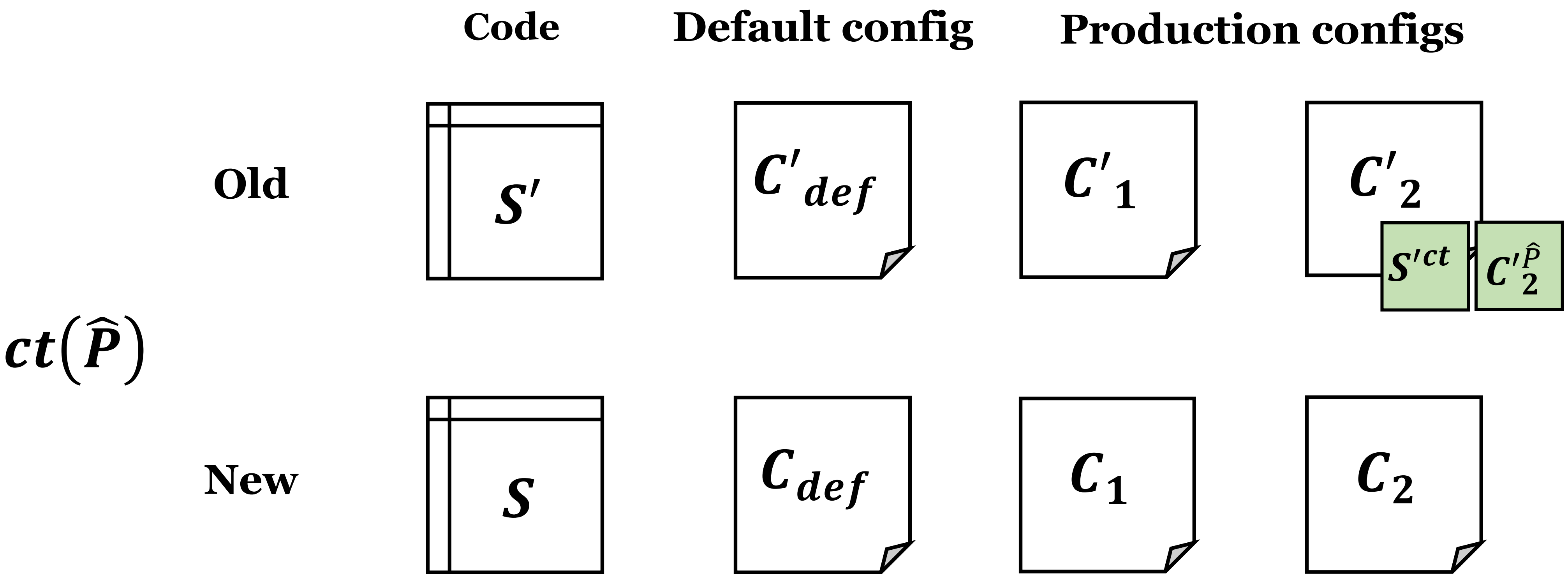
C'_2

S'^{ct}

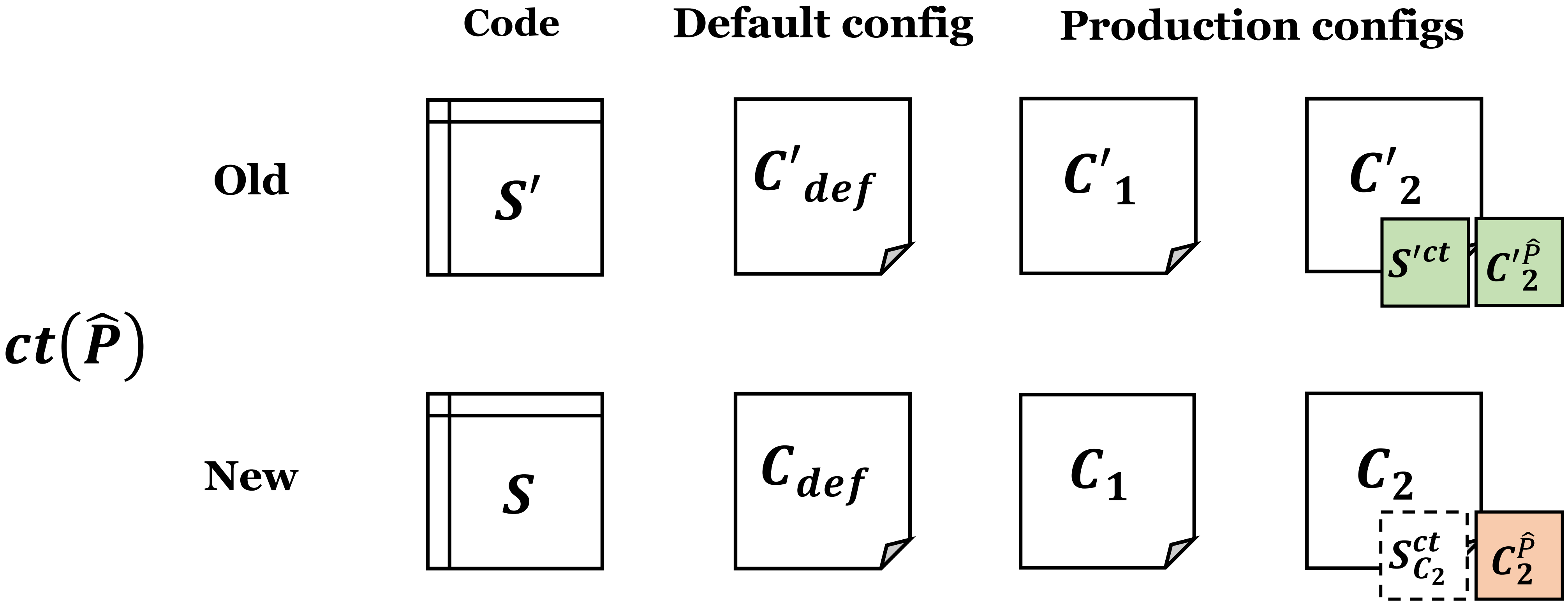
$C'^{\hat{P}}_2$

$ct(\hat{P})$

2D comparison for Ctest selection



2D comparison for Ctest selection



Code

Default config

Production configs

Old

S'

C'_{def}

C'_1

C'_2

S'^{ct}

$C'^{\hat{P}}_2$

$ct(\hat{P})$

New

S

C_{def}

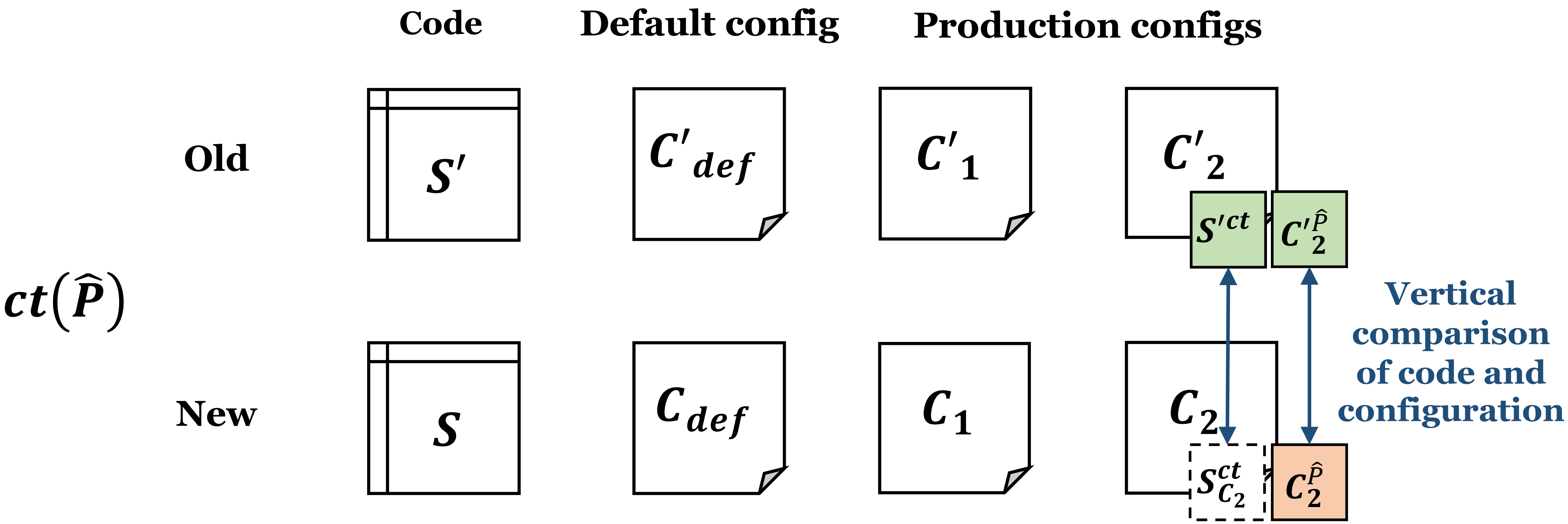
C_1

C_2

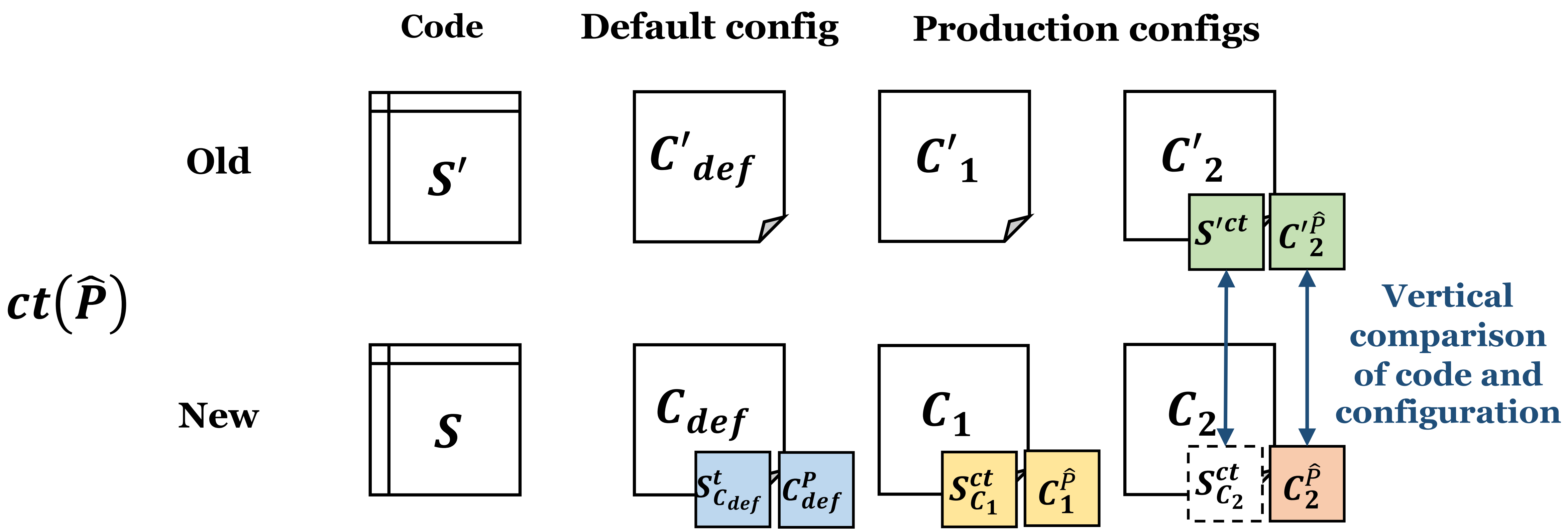
$S^{ct}_{C_2}$

$C^{\hat{P}}_2$

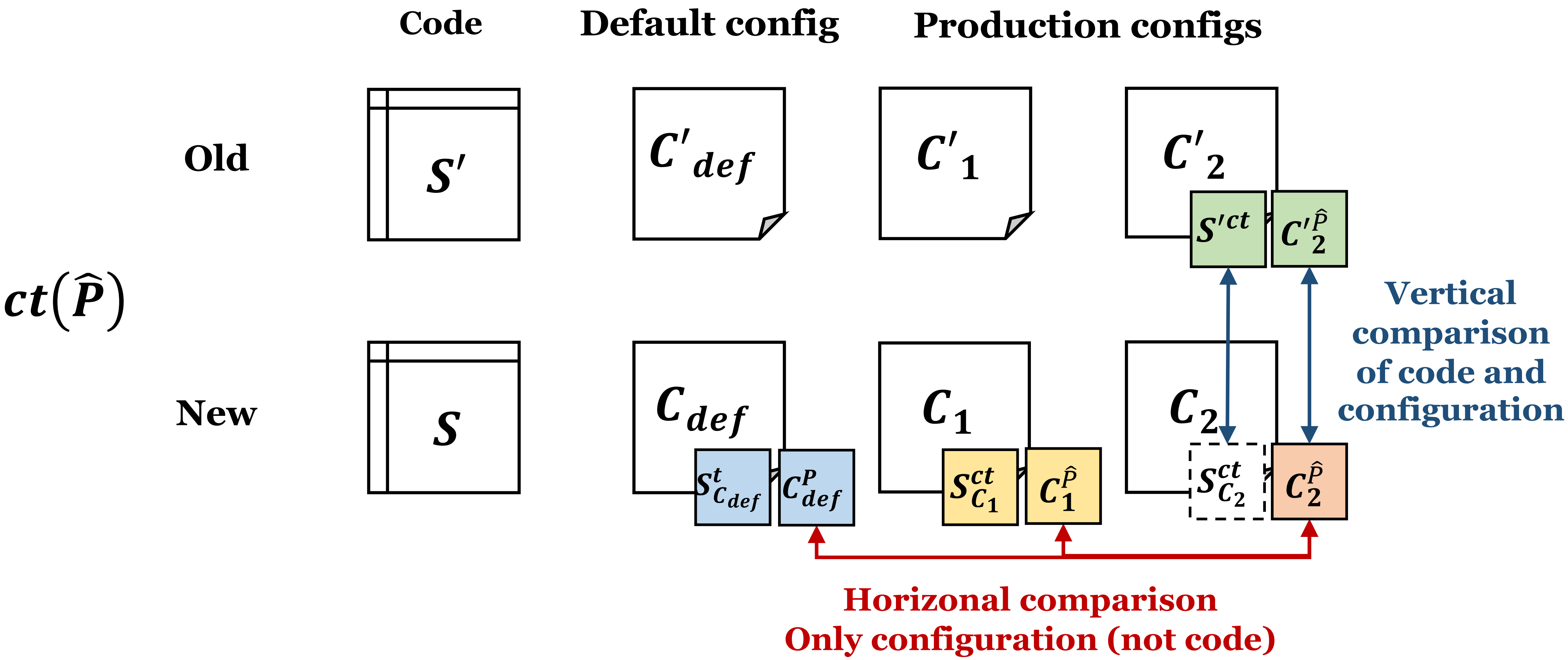
2D comparison for Ctest selection



2D comparison for Ctest selection



2D comparison for Ctest selection



Code

Default config

Production configs

Old

S'

C'_{def}

C'_1

C'_2

S'^{ct}

$C'^{\hat{P}}_2$

$ct(\hat{P})$

New

S

C_{def}

$S^t_{C_{def}}$

C^P_{def}

C_1

$S^{ct}_{C_1}$

$C^{\hat{P}}_1$

C_2

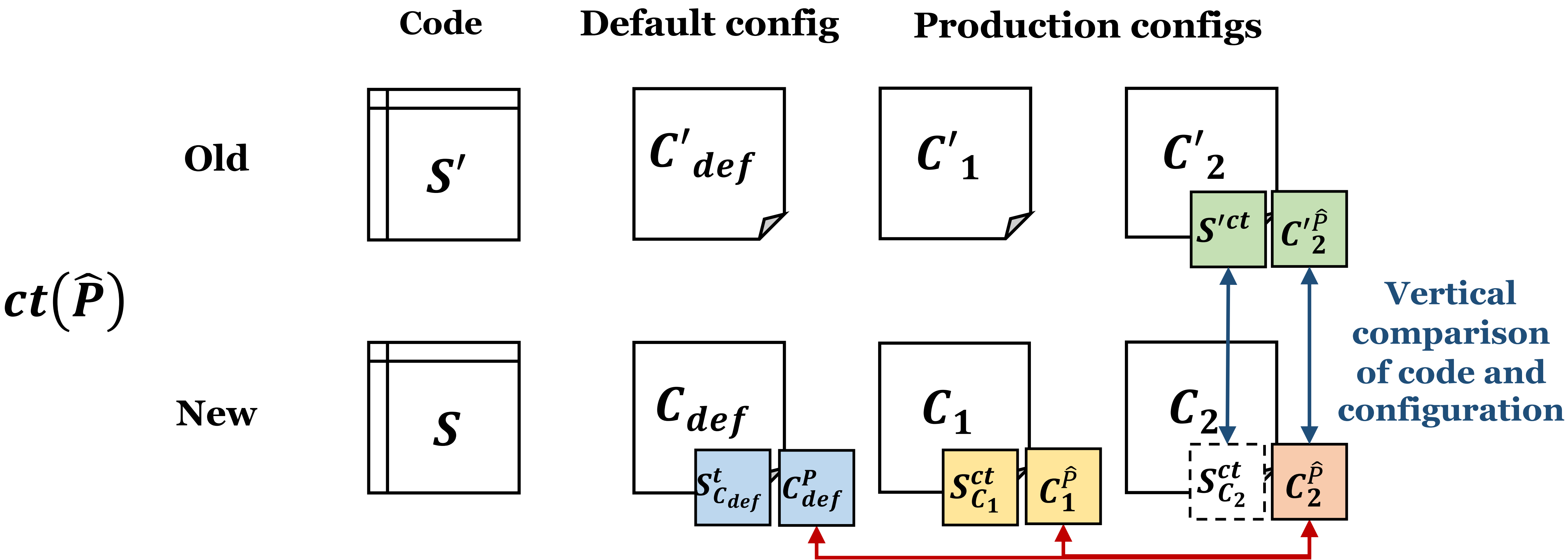
$S^{ct}_{C_2}$

$C^{\hat{P}}_2$

Vertical comparison of code and configuration

Horizonal comparison
Only configuration (not code)

2D comparison for Ctest selection



Code

Default config

Production configs

Old

S'

C'_{def}

C'_1

C'_2

S'^{ct}

$C'^{\hat{P}}_2$

$ct(\hat{P})$

Vertical comparison of code and configuration

New

S

C_{def}

$S^{t}_{C_{def}}$

C^{P}_{def}

C_1

$S^{ct}_{C_1}$

$C^{\hat{P}}_1$

C_2

$S^{ct}_{C_2}$

$C^{\hat{P}}_2$

Not select if either vertical or horizontal doesn’t change

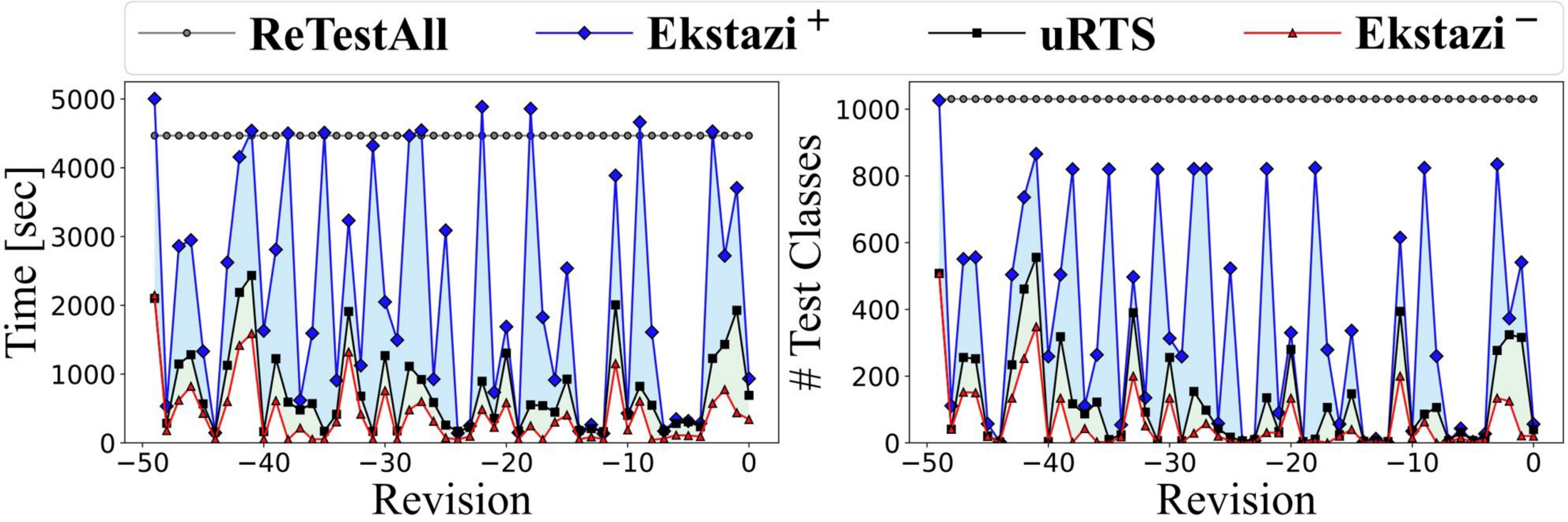
Horizonal comparison
Only configuration (not code)

Main Results

- Evaluated uRTS with 5 popular large, widely used open-source projects, 250 code revisions and 100 config changes
- Can uRTS effectively reduce the test overhead?
    - Reduce end-to-end timing by **3.64X** and **1.87X** over ReTestAll and *safe*-Ekastazi
    - Reduce number of tests by **8.92X** and **2.29X** over ReTestAll and *safe*-Ekastazi
- What is the overhead of URT with uRTS?
    - Take **1.93X** on end-to-end timing over *unsafe*-Ekastazi on three configurations

Main Results

- Evaluated uRTS with 5 popular large, widely used open-source projects,



(a) HCommon (Time)

(b) HCommon (# Tests)

Conclusion

- **Concept: Unified Regression Testing (URT)** for checking both code and configuration changes
- **Algorithm: Unified Regression Test Selection (uRTS)** for speeding up URT, with the same safety guarantee
- **Implementation**: Implemented on top of Ekstazi and OpenCtest
- **Evaluation** with five large software projects
    - Hundreds of code revisions and dozens of configuration files
    - Largest RTS experiments performed on open-source projects
- **Data/code release:** https://github.com/xlab-uiuc/uRTS-ae