CNS-2130560
CNS-2145295

Fidelity of Cloud Emulators: The Imitation Game of Testing Cloud-based Software

Anna Mazhar, Saad Sher Alam, William X. Zheng, Yinfang Chen, Suman Nath, Tianyin Xu

CORNELL UNIVERSITY

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Microsoft

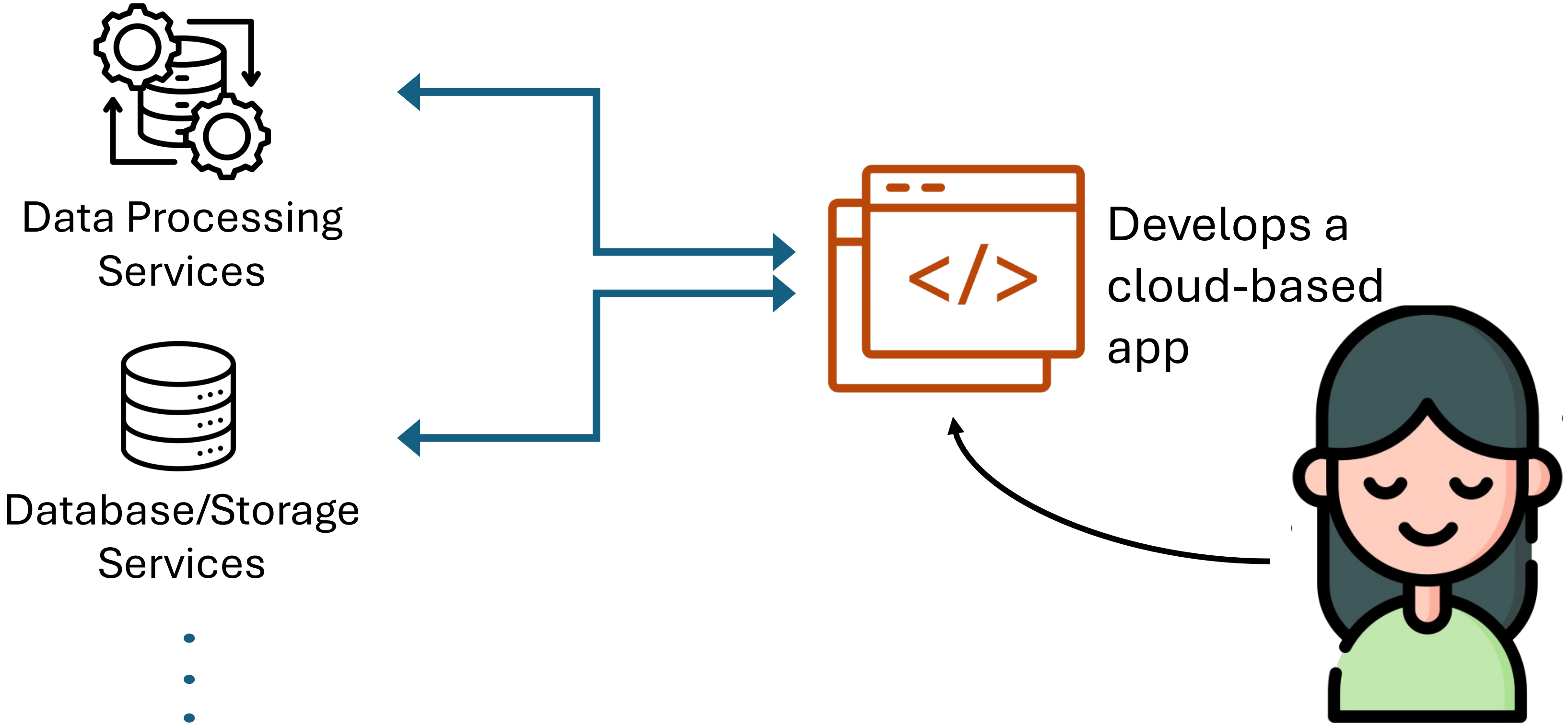
In a university far far away…

Develops a cloud-based app

Alice, a graduate student

How to test cloud-based software?

Cloud services



Data Processing Services

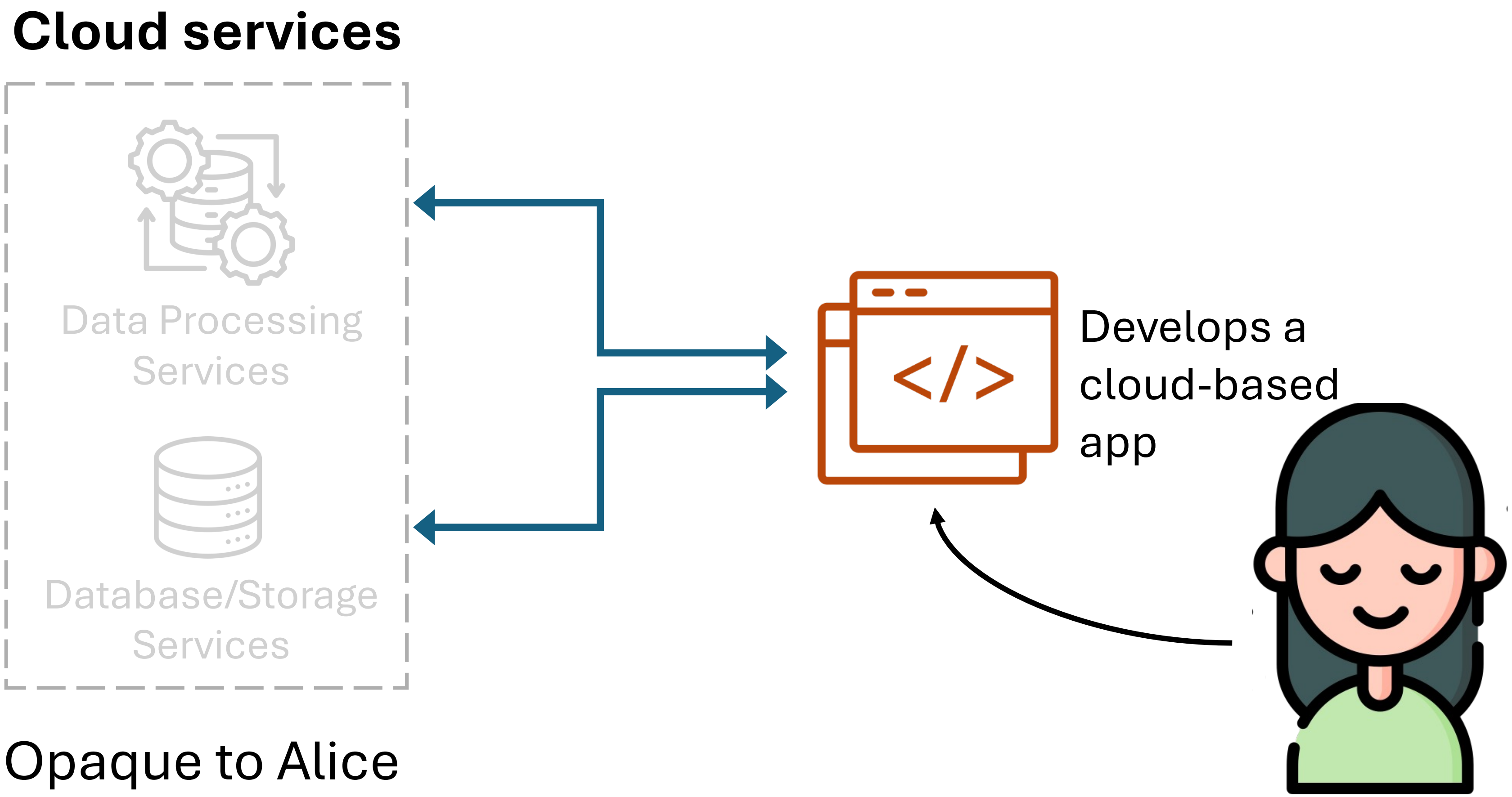
Database/Storage Services

⋮

Develops a cloud-based app

How to test cloud-based software?

Alice knows that rigorous testing of her app is critical!



Cloud services

Data Processing Services

Database/Storage Services

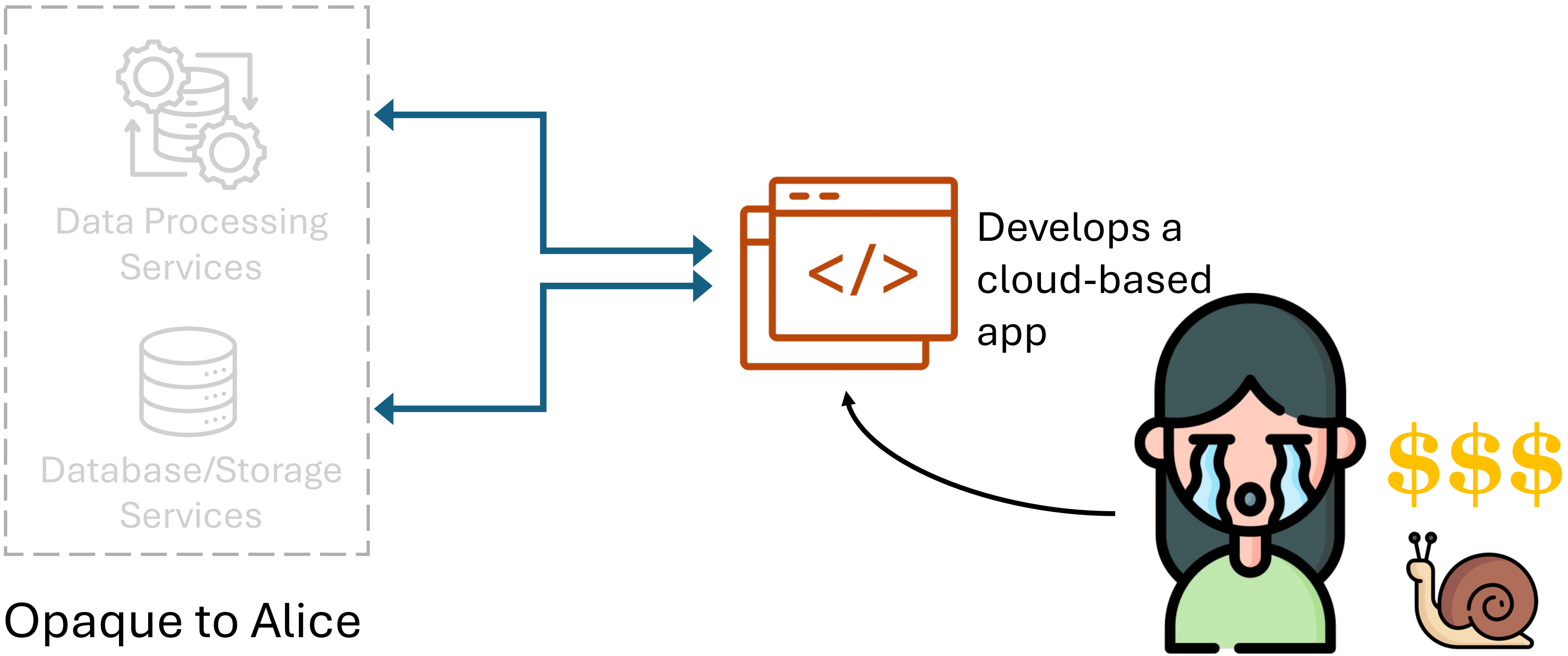
Opaque to Alice

Develops a cloud-based app

How to test cloud-based software?

Alice knows that rigorous testing of her app is critical!

Cloud services



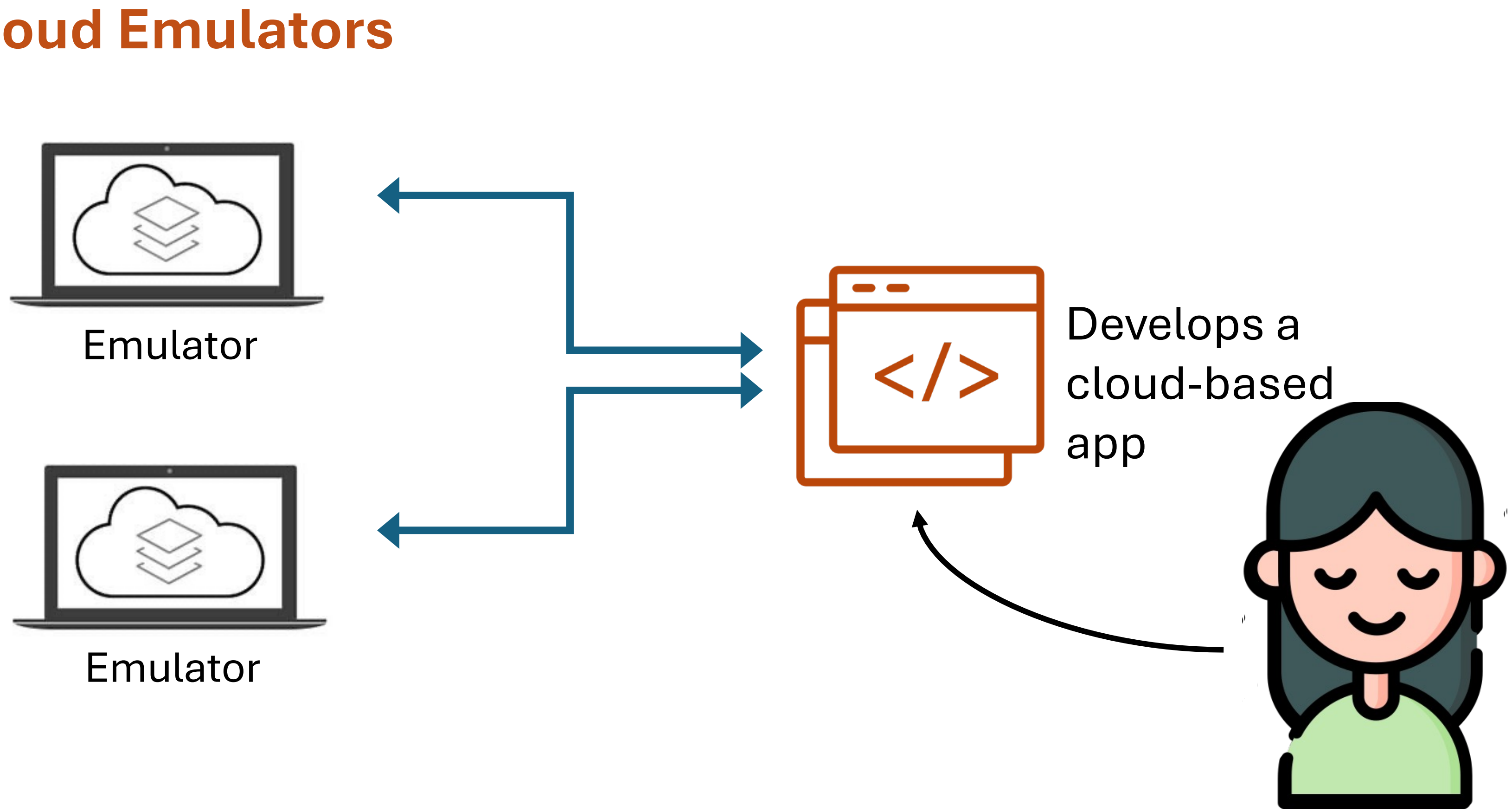
Data Processing Services

Database/Storage Services

Develops a cloud-based app

Opaque to Alice

How to test cloud-based software?



Dilemma: What shall Alice choose?

Emulator

- Cheap and fast
- May not accurately represent the cloud

Cloud

- Expensive
- High accuracy in terms of production behavior

Research Questions

- How much fidelity do existing cloud emulators provide?
- How important is it for effective software testing?
- How can we close the gaps and improve emulator fidelity?
- How can we minimize cost while using existing emulators?

Contributions

- **Problem: Challenges of testing cloud-based software**
  - Tradeoffs between cost, usability, and fidelity
- **Analysis: Discrepancies between real and emulated services**
  - A large-scale fuzzing-based measurement
  - Impacts on real-world software tests
  - Root cause analysis
- **Tooling: Hybrid cloud-emulator testing**
  - Adaptive selection between real and emulated services based on runtime monitoring
  - Can save up to 100% requests in CI/CD setup
- **Discussion: Directions to fundamentally close the gaps**
- **Artifacts**: https://github.com/xlab-uiuc/cloudtest

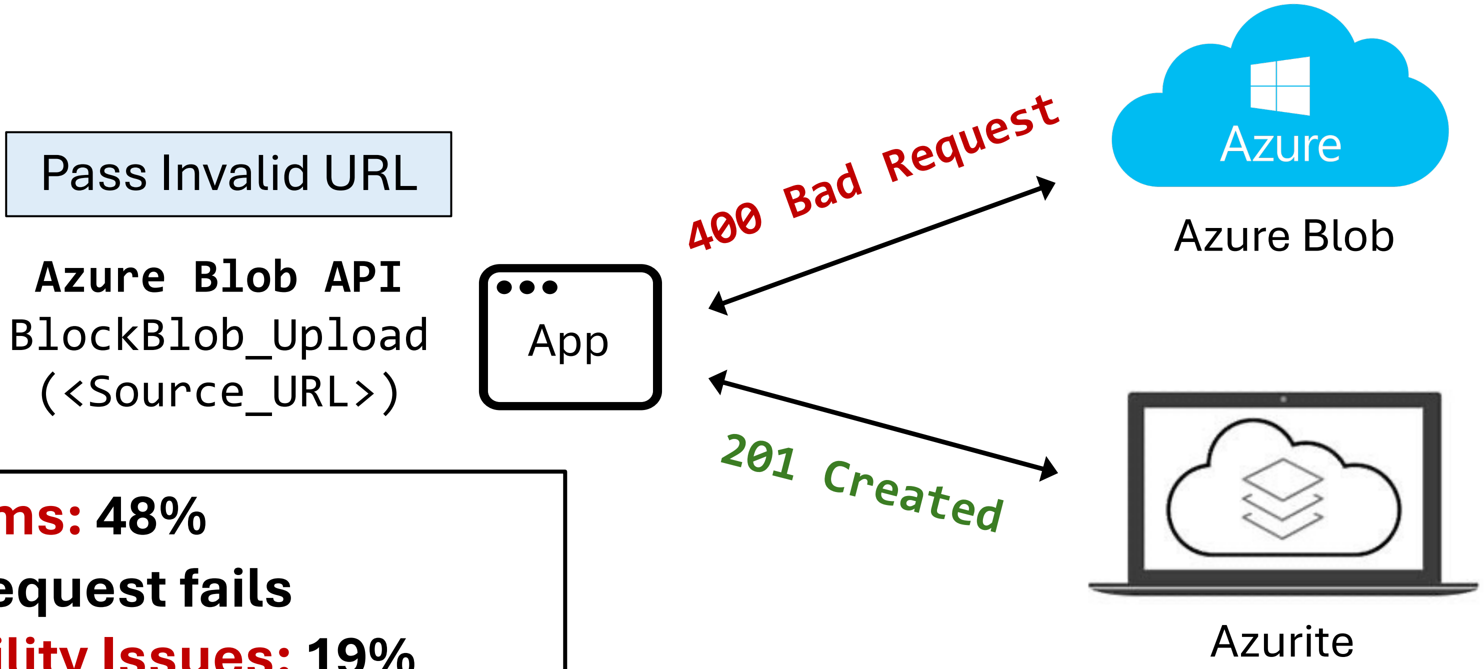
Methodology: Differential Analysis

- **Two *de facto* cloud emulators**
  - Azurite for Azure Storage
  - LocalStack for AWS
- **Five widely used cloud services**
  - Azure Blob
  - Azure Table
  - Azure Queue
  - AWS S3
  - AWS DynamoDB

- **Grammar-based SDK API fuzzing**
  - Generate diverse API sequences
    - OpenAPI specs as the grammar
  - Check response and side effects
- **Real-world software tests**
  - Tests from **10** open-source projects
    - Select tests that invoke cloud APIs
  - Check test results

Prevalence of Discrepant APIs

- Out of 255 APIs of Azure + AWS, **37% (94)** shows discrepant behavior.
    - 37% (35) of them may lead to **deployment safety violations**



- **False Alarms: 48%**
    - **Valid request fails**
- **Debuggability Issues: 19%**
    - **Both reject a request with different error messages**

Azurite lacks URL format validation check leading to a successful creation of resource

Discrepant APIs in Real-world Tests

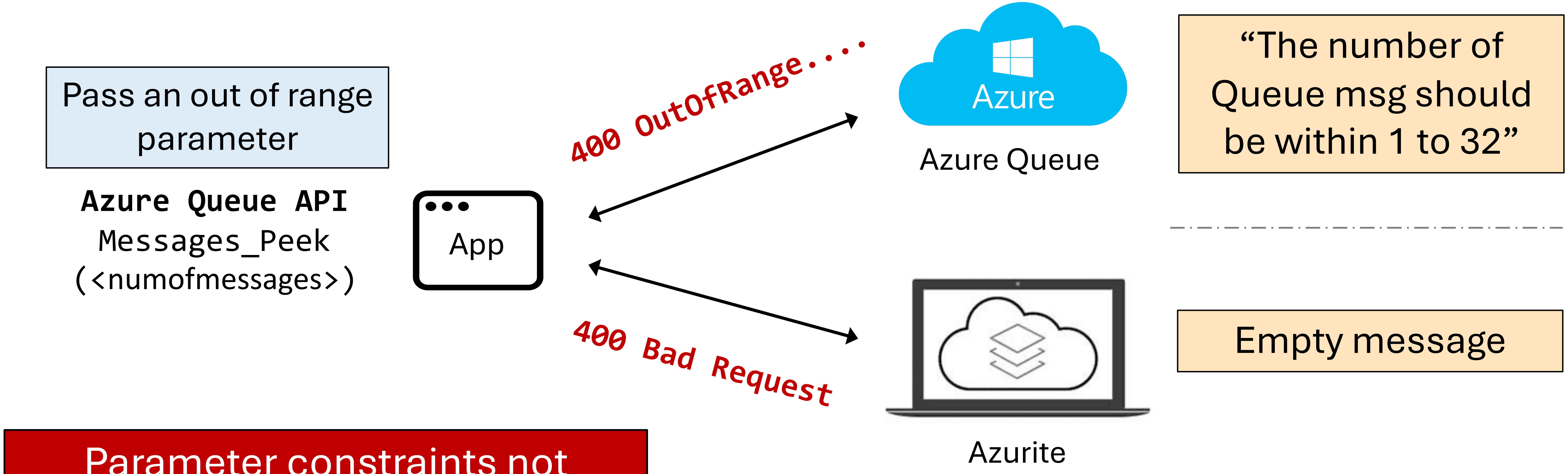
- Five out of ten evaluated projects had **discrepant test results**
  - Deployment Safety Violation was the major impact found in 96% of the tests

- From a total of **105** Azure Storage APIs, **43** APIs were invoked by tests
  - Out of **43** APIs, **12** discrepant APIs were invoked
  - Fixing 9 out of 12 can fix **100**% of discrepant results!

- Discrepant APIs does necessarily lead to discrepant results
  - Discrepancy manifest during testing when triggering parameters are used
  - Motivates our tool design

Root Cause Analysis (1)

Incomplete specs caused 56% of discovered discrepancies in Azure Storage

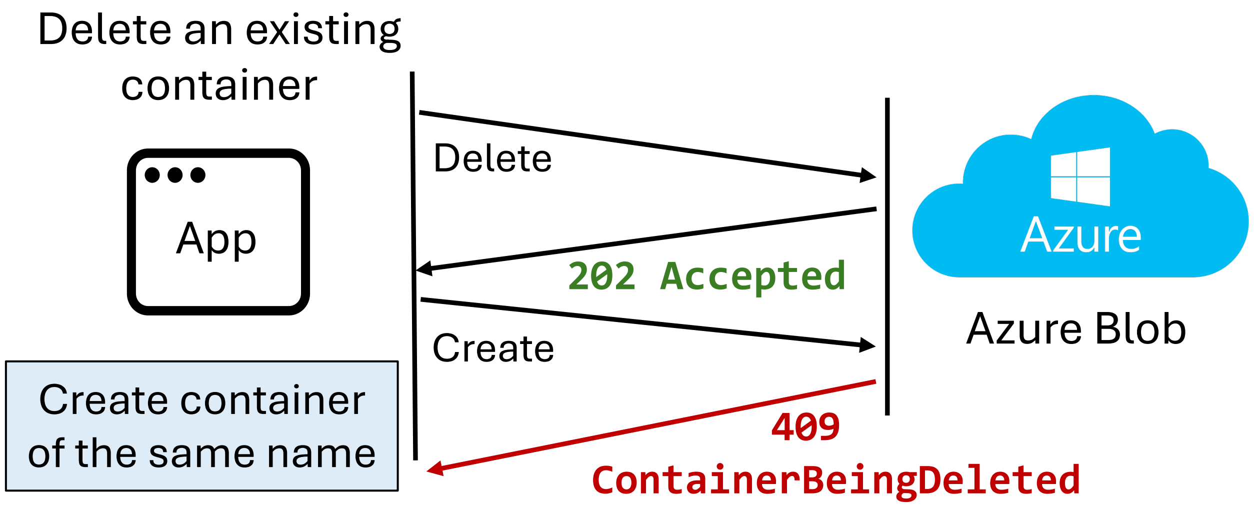


Parameter constraints not present in a machine checkable format in the API specifications

Root Cause Analysis (2)

More root causes in the paper!

Unspecified behavior caused 21% of total discrepancies



Delete an existing container

App

Delete

202 Accepted

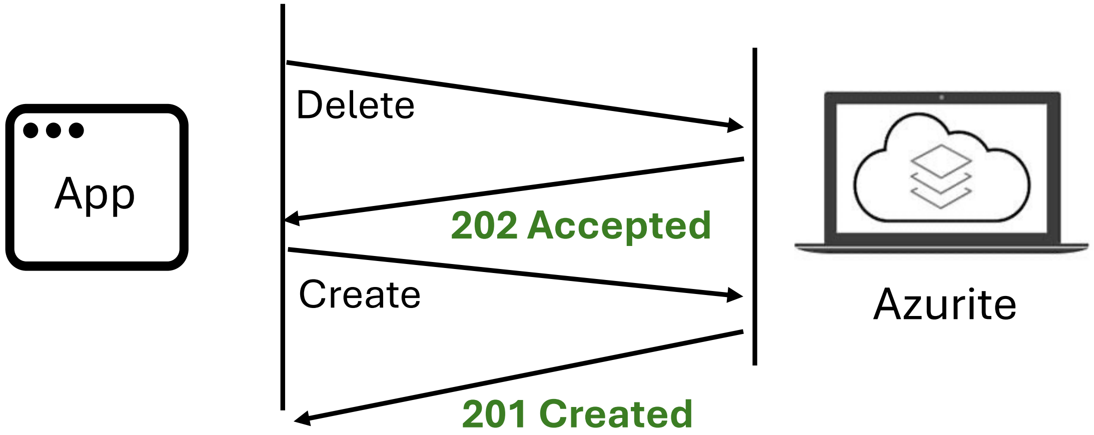
Create

409 ContainerBeingDeleted

Create container of the same name

Azure

Azure Blob



App

Delete

202 Accepted

Create

201 Created

Azurite

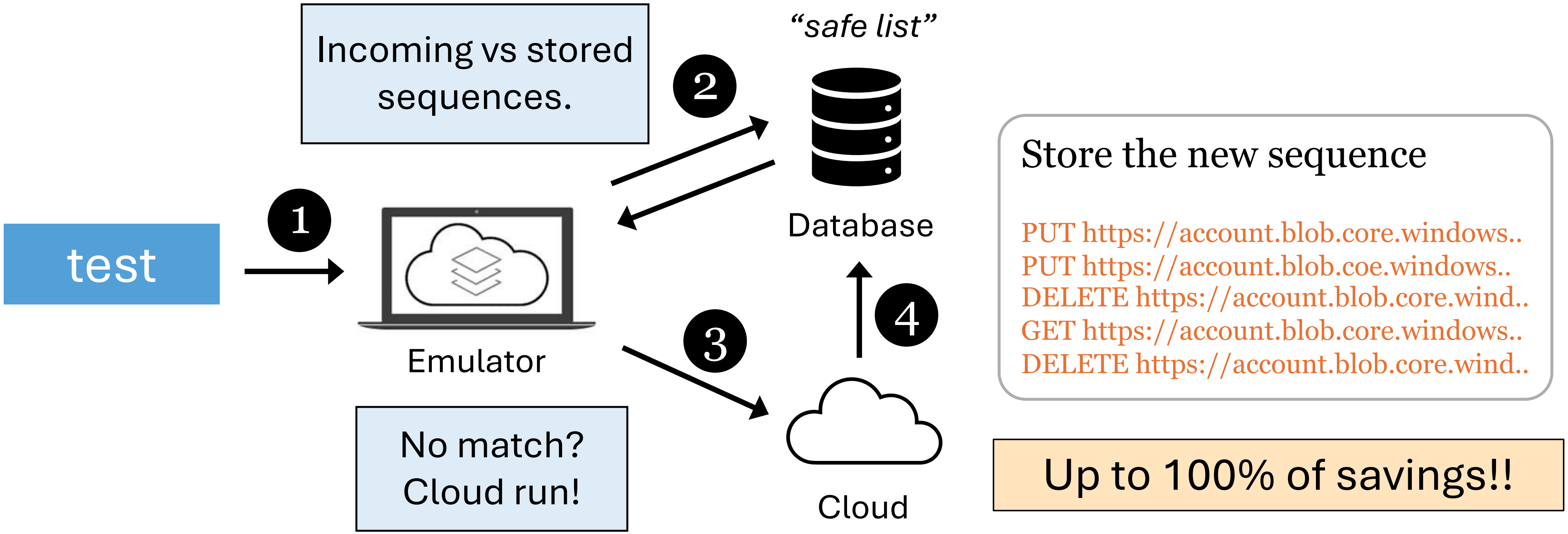
“The specified container is being deleted. Try operation later.”

Container is deleted instantly!

API synchrony is not specified hence not enforced in the emulator

ET : Hybrid Cloud-Emulator Testing

- **Principle**: Only run discrepant tests using real cloud services
- **Policies**: ***Three*** different polices based on test characteristics



Incoming vs stored sequences.

No match? Cloud run!

Store the new sequence

PUT https://account.blob.core.windows..
PUT https://account.blob.coe.windows..
DELETE https://account.blob.core.wind..
GET https://account.blob.core.windows..
DELETE https://account.blob.core.wind..

Up to 100% of savings!!

Discussion

- **Cloud service providers should play an active role.**
    - Already running continuous API fuzzing (e.g., RESTler).
    - Integrating the differential oracles

- **Executable formal models as emulators**
    - How to do it with large, complex APIs?

- **"POSIX" for cloud service APIs**
    - A lack of standard makes cloud API reliability even harder

- **Economical cloud services for software testing**
    - Spot instances is one good example

Conclusion

Much more meat in the paper!

- **Problem: Challenges of testing cloud-based software**
  - Tradeoffs between cost, usability, and fidelity
- **Analysis: Discrepancies between real & emulated services**
  - A large-scale fuzzing-based measurement
  - Impacts on real-world software tests
  - Root cause analysis
- **Tooling: Hybrid cloud-emulator testing**
  - Adaptive selection b/w real or emulated services via runtime monitoring
  - Can save up to 100% requests in CI/CD setup
- **Discussion: Directions to fundamentally close the gaps**

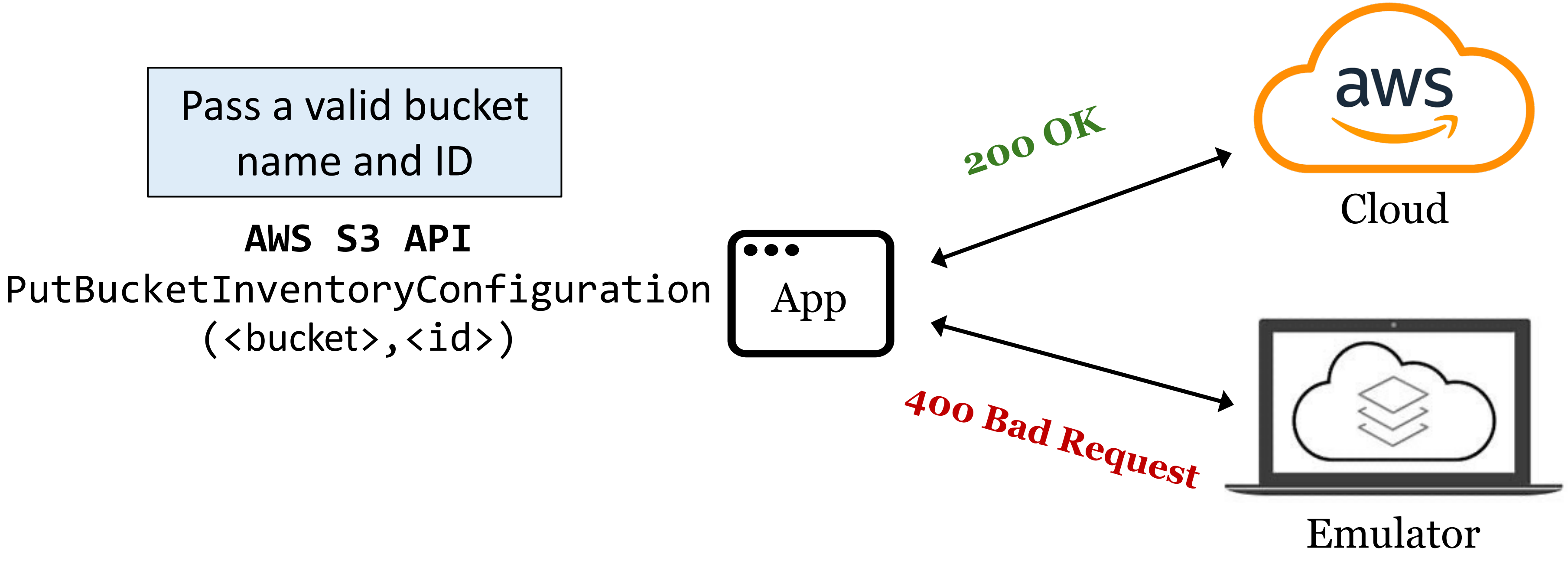
Artifacts

Connect w/ Anna

Backup Slides

Prevalence of discrepant APIs

- Out of 255 APIs of Azure + AWS, **37 %** shows discrepant behavior.
  - 48% of them may lead to **false alarms**



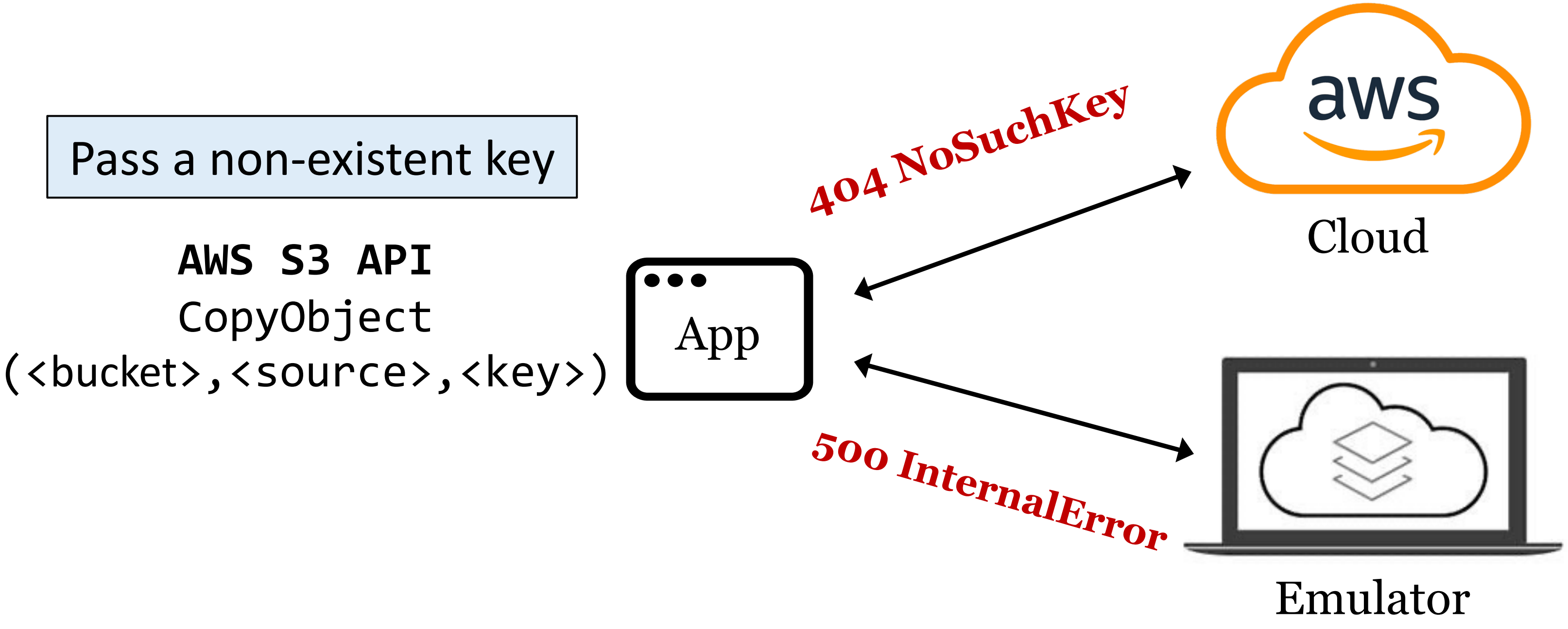
Pass a valid bucket name and ID

```
AWS S3 API
PutBucketInventoryConfiguration
(<bucket>,<id>)
```

Emulator inaccurately simulates S3's regional access causing error response to a valid request

Prevalence of discrepant APIs

- Out of 255 APIs of Azure + AWS, **37 %** shows discrepant behavior.
  - 19% of them may lead to **debuggability issues**



Pass a non-existent key

```
AWS S3 API
CopyObject
(<bucket>,<source>,<key>)
```

App

404 NoSuchKey

Cloud

500 InternalError

Emulator

Emulator fails to handle exceptions properly when invalid object keys are encountered.

Root Cause Analysis (3)

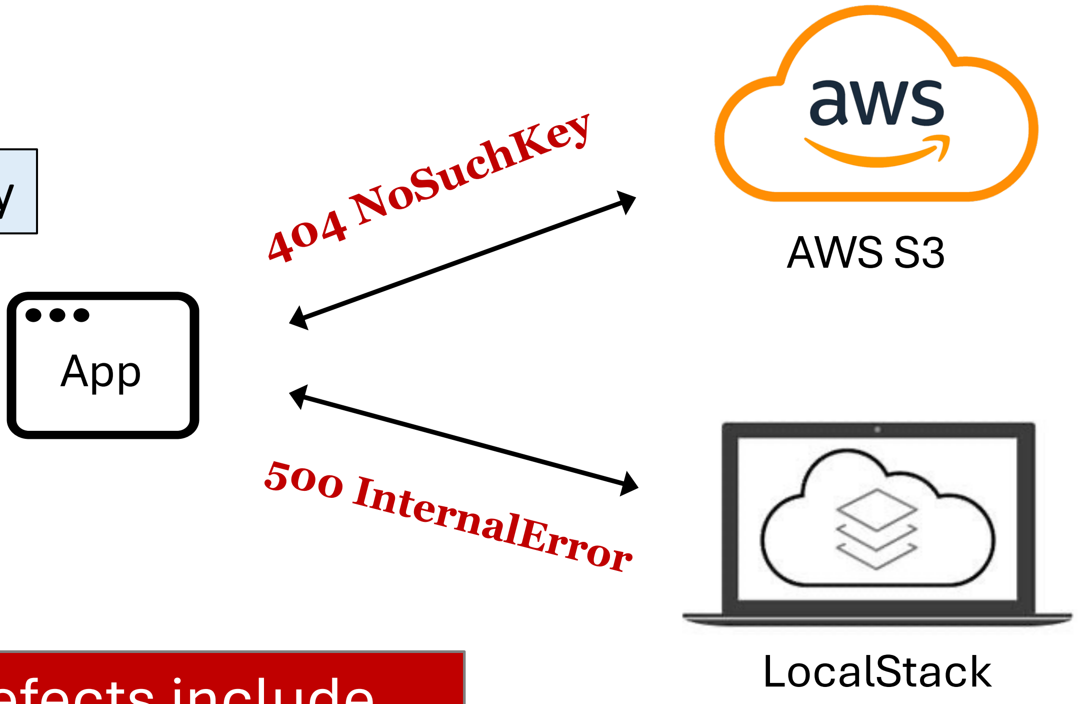
Implementation defects caused 46% of total discrepancies

(Five bugs we found have been fixed)

Pass a non-existent key

AWS S3 API

```
Restore_Object
(<bucket>,<key>)
```



“The specified key does not exist”

Invalid key not handled

Implementation defects include unimplemented features and bugs

Experimental Results

Project	Total Tests	Total Requests	# Saved Request		
			Policy A1	Policy A2	Policy A3
Orleans	189	117,905	29.4%	0.4%	29.6%
Insights	171	5,249	4.3%	47.2%	47.2%
Durabletask	101	79,654	0%	0%	0%
Streamstone	75	590	0%	99.2%	99.2%
IdentityAzureTable	51	9,860	100%	0.4%	100%

Savings vary across projects by policy

- A1: Savings with few discrepant APIs
- A2: Savings with deterministic API sequences
- A3: Achieves highest savings

Root Cause Analysis

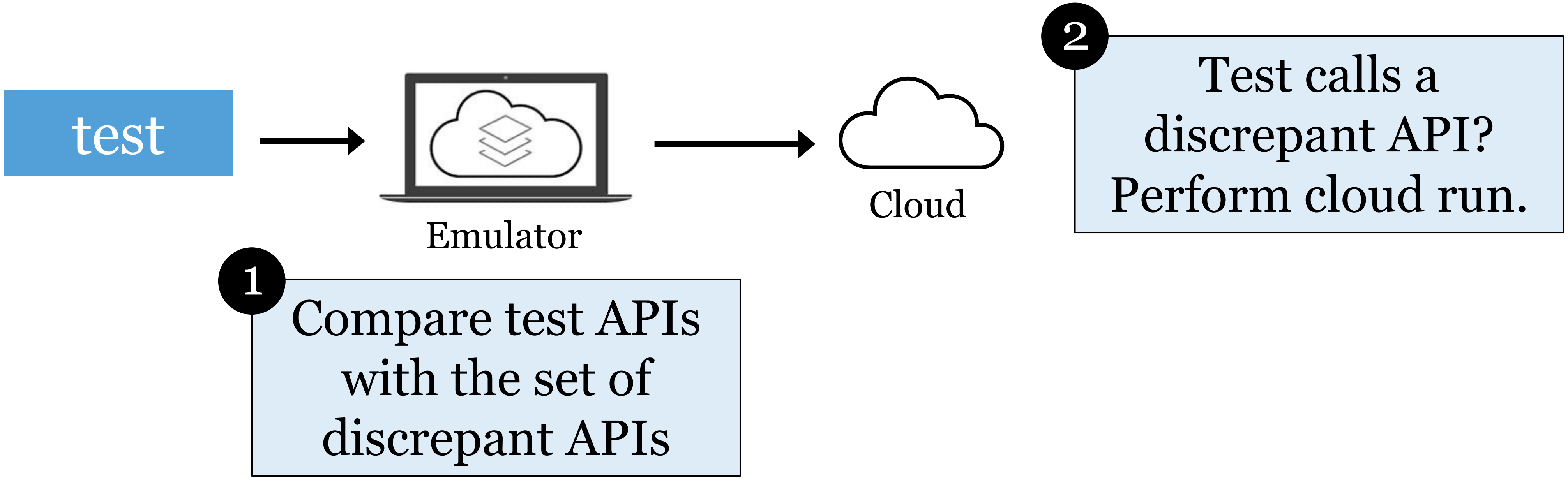
Service	Incomplete Specifications	Unspecified Behavior	Implementation Defects
Azure Blob	18 (58.1%)	1 (3.2%)	12 (38.7%)
Azure Queue	1 (50.0%)	1 (50.0%)	0 (0.0%)
Azure Table	0 (0.0%)	1 (100.0%)	0 (0.0%)
AWS S3	11 (29.7%)	14 (37.8%)	12 (32.4%)
AWS DynamoDB	2 (7.4%)	4 (14.8%)	21 (77.8%)

- Many discrepancies caused by incomplete specs
  - 58% for Azure Blob
- Defects in implementation are common
  - 78% for AWS DynamoDB and 39% Azure Blob

E_T : The Hybrid Testing Approach

API-based selection (Policy A1)

- Run test on the cloud in the presence of discrepant API.
  - API may or may not manifest into discrepancy



1 Compare test APIs with the set of discrepant APIs

2 Test calls a discrepant API? Perform cloud run.