Auditing as a Service Part 4

CS528 - Cloud Computing

Mentors: Sahil Tikale - Naved Ansari

Ali Raza - Rushi Patel - Chenxi Li - Kevin Liang

Background

- **Bare Metal Clouds**
  - Bare metal machines instead of virtual machines
  - Current bare metal clouds also do provisioning
  - Tenant does not have the flexibility
  - Each tenants may have different requirements
 - How do we ensure different tenants can share the cloud?

Background

- ## Hardware Isolation Layer (HIL)*
    - Provides isolated network of nodes
    - Exokernel for the cloud
    - Provides as little abstractions as possible
    - Allows tenants to choose provisioning system and other services
    - Move resources between multiple clusters
    - Allows security sensitive applications to use public clouds

* Hennessey, Jason, et al. "HIL: designing an exokernel for the data center." *Proceedings of the Seventh ACM Symposium on Cloud Computing*. ACM, 2016.
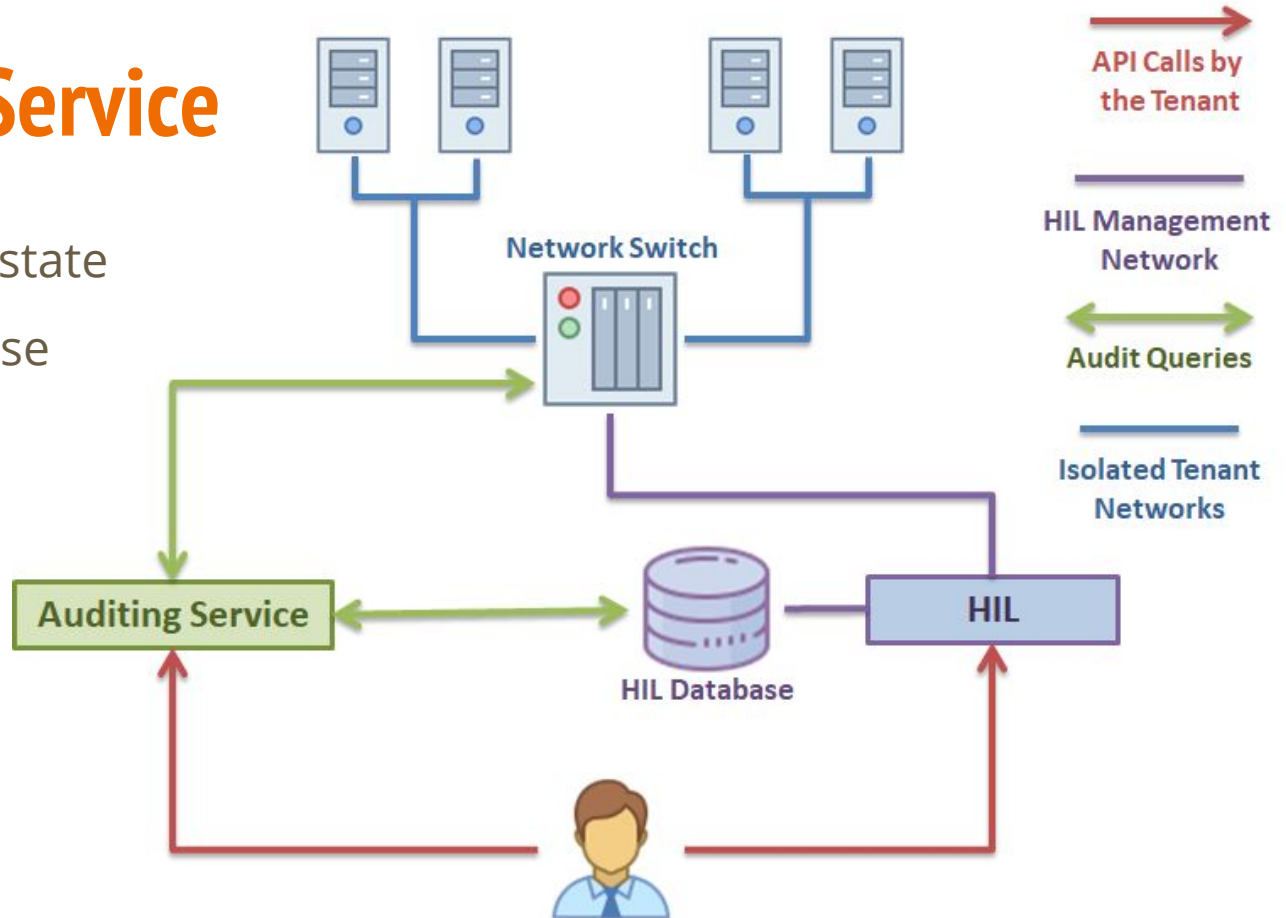
Motivation

- What if the switch state is changed without updating the HIL database?
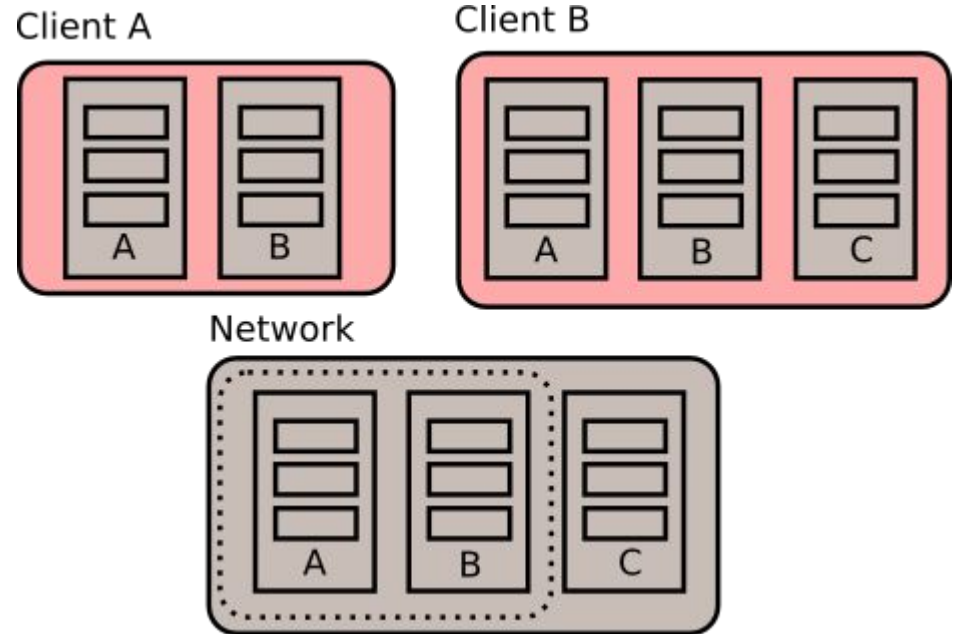
- What if the user does not trust the administrator?
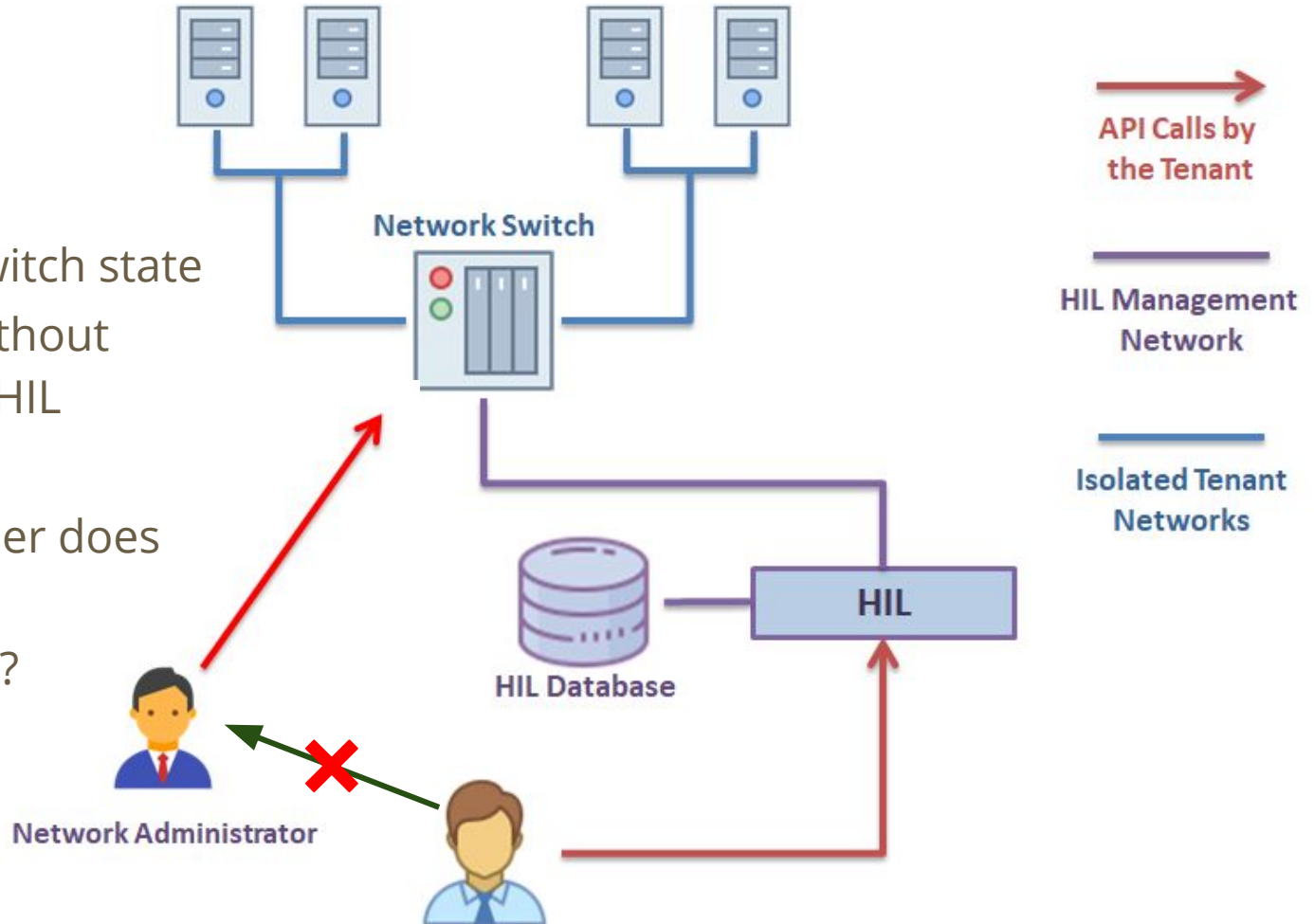
Auditing as a Service

- Query the switch state
- Query HIL database
- Check and report discrepancies

Security and Audit

- Administrator only access
- All information from HIL is available
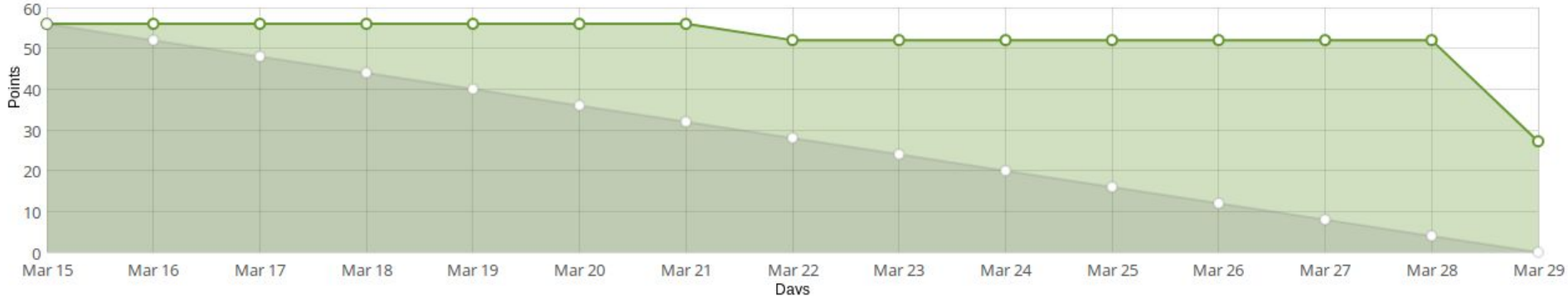- Must show detailed information for accurate usage of audit system

Current Epic

- **Epic 1**
  - Minimum Viable Product
  - Current state of HIL and network switches - Detailed info

Votes	Name	Project	Sprint	Assigned	Status	Progress	View options ⌄
▲ 0	#41 Audit system returns yes or no (Current State) `EPIC` ⌃				New ⌄		
▲ 0	#50 Audit system history and automated `EPIC` ⌃				New ⌄		

Fourth Sprint - MVP

- Complete switch communication requirements
- Make unit tests
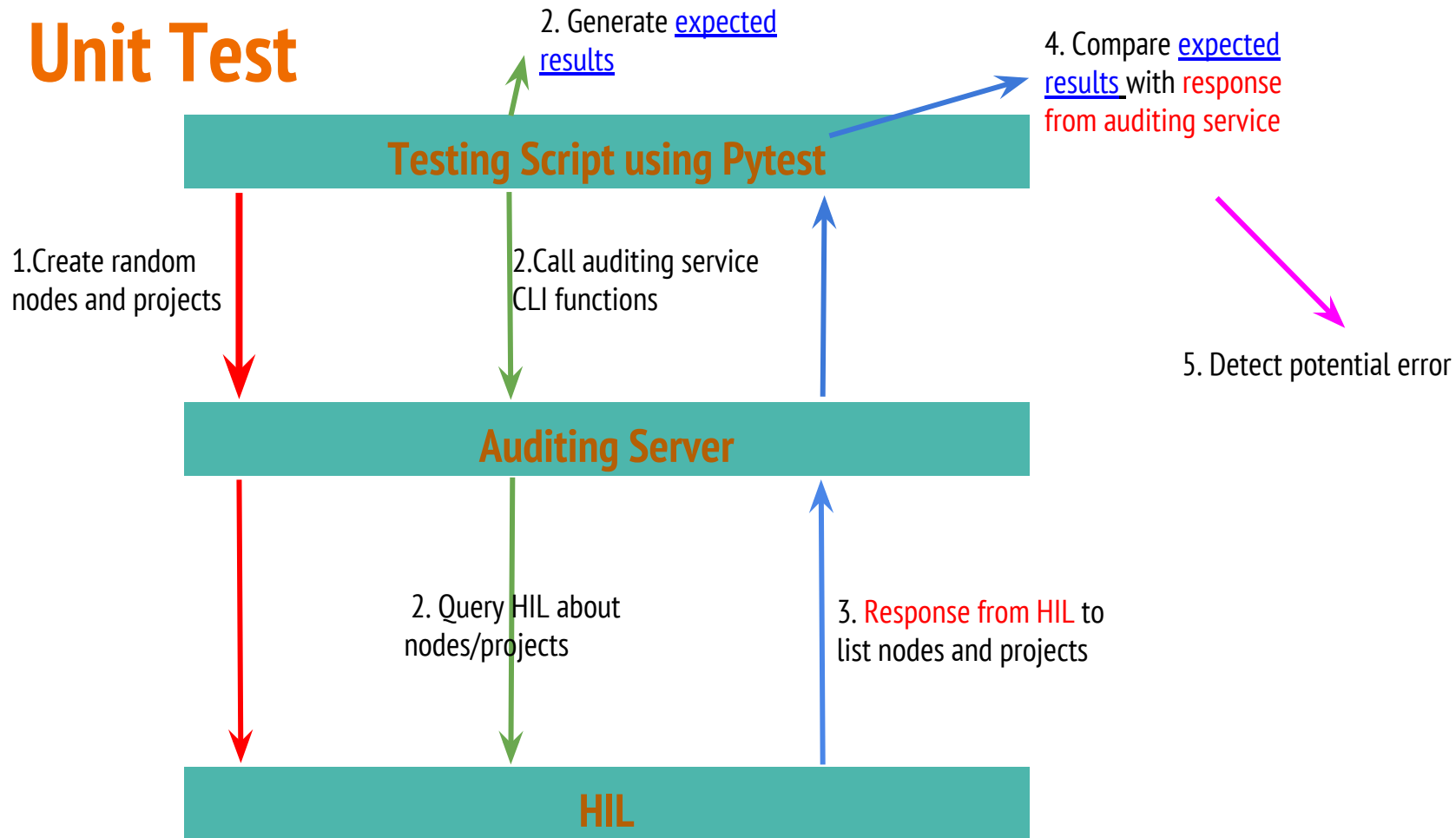- Start thinking about second epic functions and changes to system

Auditing Service Workflow
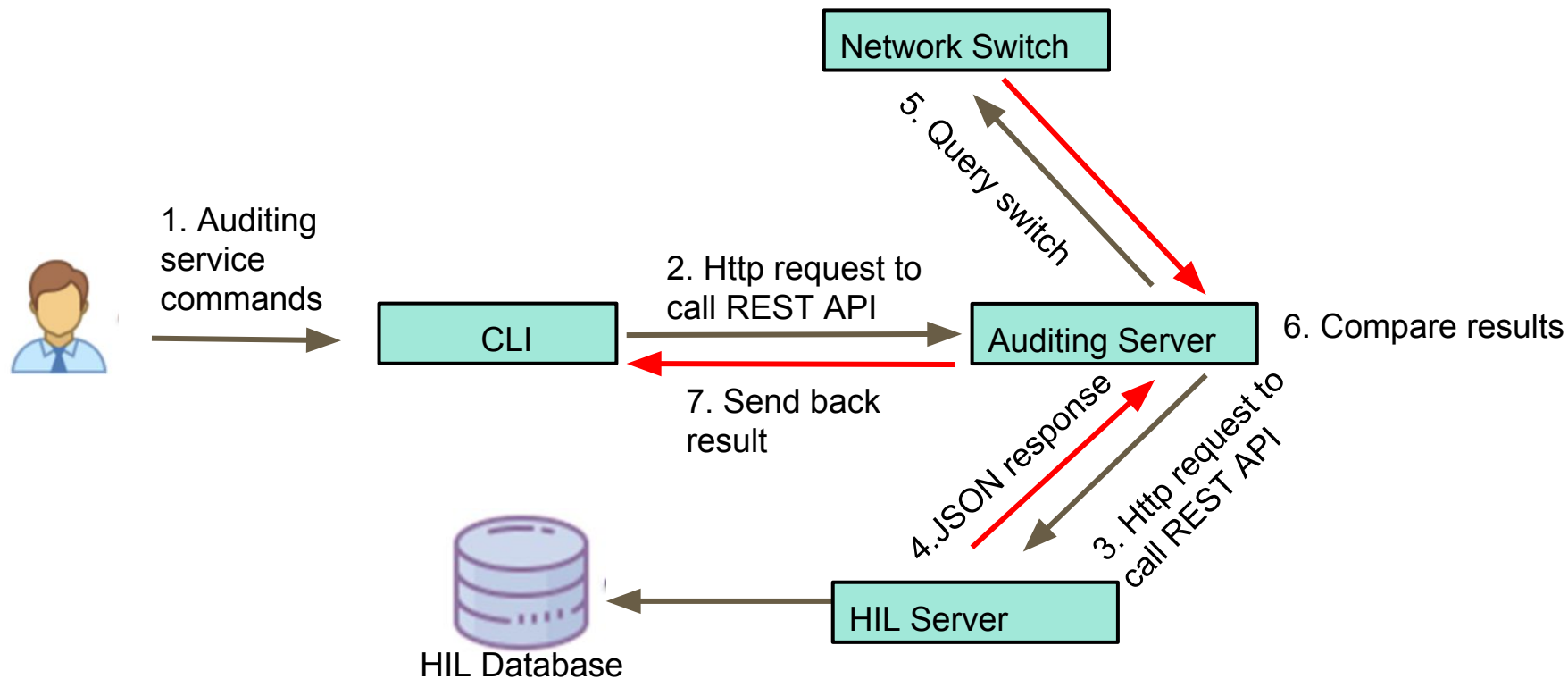
Unit Tests

- Created test functions for all auditing CLI functions
- Randomized creation of nodes and projects
- Automate the creation and deletion of nodes in HIL to test
- Simulate query to auditing server
- Use Pytest to detect potential error of auditing service

Unit Test

Testing Script using Pytest

2. Generate expected results

4. Compare expected results with response from auditing service

1.Create random nodes and projects

2.Call auditing service CLI functions

5. Detect potential error

Auditing Server

2. Query HIL about nodes/projects

3. Response from HIL to list nodes and projects

HIL

Demo

Next Sprint - Cleanup/Test and Epic 2

- **Epic 2**
  - Always on audit - history database of changes made to system
  - Poll switches for changes made directly to hil controlled nodes
  - Give a clear output that shows the history database to administrator
- **Cleanup/Test**
  - Complete unit tests of MVP
  - Finalize documentation for future users/developers

Thank you!