SLATE Traceability

Version	Comment	Effective date
1.0	Initial version	March 11, 2022

Purpose

This document describes the traceability components of the SLATE platform. The Overview of SLATE Platform Internals and Security[1] provides the basis for the components and how they function together.

Description of traceability mechanisms

The SLATE platform has various points of traceability available. Most of these points utilize various types of logs. These logs differ on the types of events collected. As noted, some of the logs collect in a central repository while others are only available at the service where they are generated.

The following sections describe the types of events that the SLATE Platform logs, by service.

SLATE API server

The SLATE API server has a very complete logging service for all events that happen between the clusters and the API server itself. All user commands, mapping, monitoring and web portal access show in the SLATE API server logs.

Examples of SLATE API server logs:

- **User command:**
  user_oZJDuowj3bE (*FirstName LastName*) requested to list clusters from 127.0.0.1:55044

[1] Overview of SLATE Platform Internals and Security, request a copy by sending email to security@slateci.io

This document is a policy of the SLATE (Services Layer at the Edge) project, supported by the National Science Foundation Office of Advanced Cyberinfrastructure: "CIF21 DIBBs: EI: SLATE and the Mobility of Capability", award number OAC-1724821.

cluster listing completed in 0.103601 seconds

- **Map on website:**
  Querying database for locations associated with cluster cluster_HuMFzohYwDA

- **CheckMK Actions**
  Unable to contact cluster_QJaTB3nj5a8 (check-mk-test): Unable to connect to the server: x509: certificate signed by unknown…
  Unable to contact cluster_iqLa5Ov8oZE (cluster3): Unable to connect to the server: net/http: request canceled while waiting…

- **Web Portal Access to the API:**
  user_zKfSKAc3maY (WebPortal) requested to ping cluster cluster_u9iI0_EYe3o from 128.xxx.xxx.xxx:48328

Where logged:

The SLATE API server logs locally and to a central external syslog server at the University of Chicago. The SLATE API server logs all of the commands executed against the SLATE API server.

Events logged:

SLATE API functions and commands that produce logs. All logs record informational items unless otherwise noted as logs errors only.

Logs are sent to standard out and standard error and accessed through `journalctl`. These commands break into the following groups. For a full listing of each of the commands, please reference the SLATE API README[2] and the links therein for the client manual and the resources directory.

- User Commands
- Application Commands
- Application Instance Commands
- Cluster Commands
- DNS Manipulator
- Group Commands
- Monitoring Credential Commands
- Persistent Store (internal database commands)
- Secret Commands
- Volume Claim Commands
- SLATE Service

[2] SLATE API Server README: https://github.com/slateci/slate-client-server/blob/master/README.md

SLATE Web Portal

The SLATE web portal requires a login via the CILogon[3] interface using Globus Auth[4] as a backend. The SLATE web portal is an NGINX[5] application in front of a Flask[6] application. The web portal provides basic http access logging, user login information, and matching to SLATE groups. More detailed logs for the user regarding login, attributes, Globus IDs, oauth token, etc. reside at the respective federated institution. The Sirtfi[7] trust framework can be used to obtain these logs in connection with managing a security incident.

Each of the SLATE cluster commands which the SLATE portal executes show in the SLATE API Server logs.

Where logged:

- The SLATE Portal logs locally on the server and to the UChicago Maniac Lab syslog server.

Events logged:

- Basic HTTP access
- User login

SLATE Command Line Interface

The SLATE Command line interface executes locally on a user's device used by any of the SLATE roles.[8] The SLATE Command line executes commands at the SLATE API server, therefore, it does not log directly but the results of each command executed show in the SLATE API server logs.

Where logged:

- All SLATE commands logged on SLATE API server
- No local logs for SLATE commands by default.

---

[3] CILogon: An Integrated Identity and Access Management Platform for Science: https://www.cilogon.org/home
[4] Globus Auth: An Identity and Access Management Platform service: https://www.globus.org/platform/services/auth
[5] NGINX: https://nginx.org/en/
[6] Flask: https://flask.palletsprojects.com/en/2.0.x/
[7] Sirtfi Trust Framework: https://refeds.org/sirtfi
[8] SLATE Roles: https://slateci.io/docs/concepts/index.html

Events logged:

- Every SLATE command available (see SLATE command line online help for a reference list.)

Amazon Web Services

All Amazon Web Services that SLATE uses log events via Cloudtrail[9] and CloudWatch[10]. The SLATE team is working to develop the processes to actively monitor these events. The following AWS services are used:

- Identity and Access Management
- Route53
- S3
- DynamoDB (hosted)

Where logged:

- AWS Cloudwatch

Events logged:

- Access Control
- Reference CloudWatch documentation noted in footnote
- Reference CloudTrail documentation noted in footnote

Repositories and Registries

Dockerhub

The SLATE team utilizes the Dockerhub[11] community repository for hosting container images. Specifically, the images are under the community slateci[12] section. SLATE does not utilize any advanced login features of Dockerhub and therefore does not use any of its traceability or logging capabilities.

Where logged:

- Dockerhub central logging

---

[9] Cloudtrail: https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-user-guide.html
[10] AWS CloudWatch: https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/WhatIsCloudWatchLogs.html
[11] Dockerhub: https://hub.docker.com
[12] Dockerhub slateci publishing point: https://hub.docker.com/u/slateci

Events logged:

- Docker maintains its own set of logs
- SLATE does not utilize any Dockerhub logging functionality

GitHub and Git Container Registry

The SLATE team uses GitHub[13] for code version control and the Git Container Registry[14] for hosting container images. The SLATE team currently leverages Commit Logs for watching code revision events, Audit Logs for additional code events, Security Log for access control, Network graph for performance and branch management, and Contributors to watch those editing the code. SLATE is allowing pushes to the main branches of the different code repositories but are able to monitor via the logging mechanisms. SLATE is not supporting commit signing[15] or protected branches[16] at this time.

Where logged:

- GitHub Commit logs (Repository Level)
- GitHub Audit Logs and GitHub Dashboard for visualization (Organization Level)
- GitHub Security log (Member and Organization Levels)
- GitHub Network Graph (Repository Level)
- GitHub Contributors (Repository Level)

Events logged:

- GitHub Commit Logs (see GitHub GraphQL API[17] reference for a complete list)
- GitHub Security Log (see GitHub GraphQL API reference for a complete list)

SOTERIA[18]

The SLATE team utilizes the SOTERIA repository of curated science container images, which uses Harbor[19]. SOTERIA login access is protected by CI-Logon. The SOTERIA team manages the logging of the instance and captures all commits in the logs. This work is in collaboration with the OSG.

[13] GitHub: https://github.com/
[14] GitHub Container Registry: https://ghcr.io
[15] GitHub Commit Signing: https://docs.github.com/en/authentication/managing-commit-signature-verification/signing-commits
[16] GitHub Protected Branches: https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/defining-the-mergeability-of-pull-requests/about-protected-branches
[17] GitHub GraphQL API: https://docs.github.com/en/graphql
[18] SOTERIA is a Repository of known curated and scanned images - predominantly open source science projects
[19] Harbor is an open source trusted cloud native registry project: https://www.cncf.io/projects/harbor/

Where logged:

- SOTERIA Harbor site

Events logged:

- SOTERIA team manages the events logged
- authentication/authorization
- Commits of images

Continuous Integration/Continuous Deployment

Jenkins

The SLATE Jenkins server resides at the University of Chicago. The server logs all command line and web access locally to the server and to the University of Chicago MANIAC Lab syslog.

Where logged:

- Local to the Jenkins server
- University of Chicago MANIAC Lab syslog

Events logged:

- Authentication/Authorization to the Jenkins web interface
- Authentication/Authorization to the command line of the server
- Jenkins CI/CD actions

GitHub Actions

SLATE uses GitHub Actions for a portion of the CI/CD workflow. The logging of GitHub Actions falls under the same logging umbrella as GitHub. In addition to the authentication/authorization, commits, etc., GitHub Actions records a history of all the actions taken. This history is 90 days by default.

Where logged:

- GitHub Commit logs (Repository Level)
- GitHub Audit Logs and GitHub Dashboard for visualization (Organization Level)
- GitHub Security log (Member and Organization Levels)
- GitHub Network Graph (Repository Level)
- GitHub Contributors (Repository Level)

Events logged:

- GitHub Commit Logs (see GitHub GraphQL API[20] reference for a complete list)
- GitHub Security Log (see GitHub GraphQL API reference for a complete list)

Configuration Management

The SLATE Puppet[21] instance and the associated GitLab[22] instance that support all the Puppet configurations both reside at the University of Chicago. The SLATE team uses both of these together to support configuration management and credential management. The SLATE Puppet and GitLab instances log both to the local servers and to the University of Chicago MANIAC Lab syslog.

Where logged:

- Log local to respective virtual machines
- Log to University of Chicago MANIAC Lab syslog server

Events logged:

- Who did updates to the puppet configuration
- Who pushed the update to the puppet configuration
- When did the updates to the puppet configuration happen
- When did the updates push to the servers
- All sudo events and changes to syslog config

Monitoring systems

The SLATE Checkmk[23] system which provides alerting and logging is a distributed system. The SLATE Checkmk system has an instance at each SLATE core {Chicago, Michigan, Utah} location which logs locally and to the respective sites' syslog servers. The primary Checkmk instance resides at the University of Michigan and monitors the central SLATE components such as the SLATE API server.

Where logged:

- Locally to each respective server
- Each respective location's syslog server

---

[20] GitHub GraphQL API: https://docs.github.com/en/graphql
[21] Puppet Configuration Management: https://puppet.com/use-cases/continuous-configuration-automation/
[22] GitLab: https://about.gitlab.com/https://about.gitlab.com/
[23] Checkmk: https://checkmk.com/

Events logged:

- Authentication / authorization to the web interface
- All web interactions

Log Retention

The SLATE Platform retains syslogs for a rolling 1 year time period. For specific hosted services, SLATE retains logs for 90 days. The University of Chicago MANIAC Lab syslog server retains logs for 90 days.

SLATE Platform logging services support auto rollover at individual core site syslog servers with email alerts for full filesystems and logs approaching log/filesystem quotas.