WHITE PAPER

Security First

Security Features and Best Practices

Orchestral.ai

AI-Driven Orchestration

Introduction

An infrastructure orchestration platform is central to an enterprise deployment and by definition, it must have access to data and configurations throughout the network. Therefore the security of the system itself is of primary concern. It is essential that all components of the orchestration platform be both designed and deployed securely.

The security concepts of the Orchestral's Composer solution can be organized into four areas:

- **Server Security**
- **Application Security**
- **Communication Security**
- **Content Security**

Each of these layers need to be designed and deployed with security as a central goal for the overall system.

Server Security

Composer can be installed on RedHat/CentOS and Ubuntu Linux systems. Server administrators should follow enterprise best practices for Linux server hardening to ensure that the underlying operating systems are adequately secure. For more information on options and recommendations for secure installations refer to the documentation for RedHat/CentOS and Ubuntu.

Application Security

Authentication

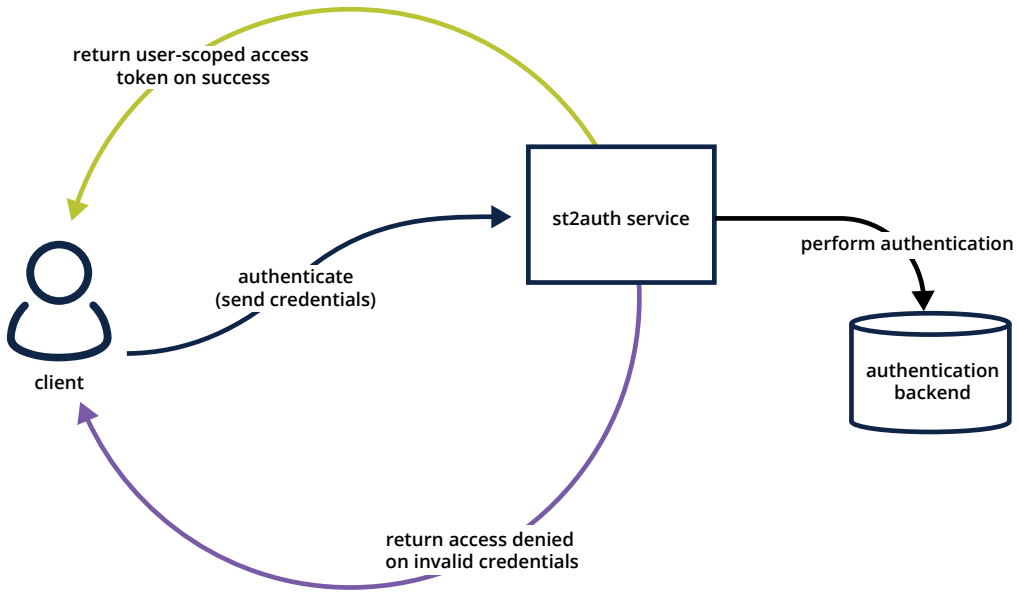
Composer includes an authentication service that is re-sponsible for handling user authentication and gener-ating time-limited access tokens. When authentication is enabled (the default), those access tokens are used to authenticate against the Composer REST APIs.

The service can be configured with different backends (i.e. PAM, LDAP, etc.) to handle the authentication. Dif-ferent backend packages are available from the Composer repo.

API Keys

Composer also supports API keys. These differ from tokens in that they do not expire. This makes them suitable for integrations with other applications, e.g. through webhooks.

For security purposes the <API_KEY_VALUE> is only shown at create time. Composer itself does not store this API Key value in its database, only a one-way hash is stored. It is not possible to retrieve an API Key after creation. In the event the API key is lost, there is no method available within Composer to read and



access the same key, the only alternative is to delete the old key and create a new key.

Role Based Access Control

Role Based Access Control (RBAC) allows system administrators to restrict users' access and limit the operations they can perform. For instance, a team is granted access to a limited set of contents such as a set of Composer packs or as granular as a number of Composer actions that the team can list, view, and/or execute.

A user represents an entity (person/system) which needs to be authenticated and interacts with Composer through the API. User permissions are represented as a union of permission grants which are assigned to all the user roles.

A role contains a set of permissions (permission grants) which apply to the resources. Permission grants are usually grouped together in a role using specific crite-ria (e.g. by project, location, team, responsibility, etc.). Permission can be granted at different levels of the ap-plication: platform level, pack level or action level.

By default, when a new Composer user is created, this user has no roles assigned to it, meaning the user has access to perform any API operation which is behind the RBAC wall. Roles are assigned to the users. Each user can have multiple roles assigned and each role can be assigned to multiple users.

Patch Schedule

Major releases are typically introduced every three to six months. However, any security vulnerabilities discovered in between releases are considered high priority and will be addressed in an immediate update.

The industry de facto standard of Responsible Disclosure for handling security issues is followed. This means the issue will only be disclosed after a fix for the issue has been developed and released and full credit will be attributed to the person who reported the issue.

It is recommended that users update as soon as practical whenever a new release is available.

Default Passwords

There is no default admin username or password for the platform, but those are required during installation. It is recommended to choose a unique admin username and strong password prior to or during the installation process.

In addition, when attached services, such as MongoDB, RabbitMQ, Redis, Corosync or Pacemaker are installed, they have authentication disabled or use a default static password. It is also essential to update these passwords. Note that if installed with the installation script, this is done automatically.

Communication Security

Composer interacts with the environment through actions and sensors. Sensors are a way to integrate ex-ternal systems and events with Composer. Sensors either periodically poll some external system, or passively wait for inbound events. These sensor events, when matched to conditions in rules, will trigger action execution in Composer.

Sensors must follow the Composer-defined sensor inter-face requirements. Actions are pieces of code that can perform arbitrary automation or remediation tasks in your environment.

Sensors and actions should be configured to interact with the infrastructure elements over encrypted TLS/SSL sessions. In addition to sensors, webhooks allow you to integrate external systems with Composer using HTTP webhooks. Unlike sensors which use a "pull" approach, webhooks use a "push" approach. They push triggers directly to the Composer API using HTTP POST requests. Webhook POSTs should also be encrypted with TLS/SSL.

Services

For MongoDB, RabbitMQ, Redis, Corosync and Pacemaker services, SSL/TLS should be enabled for encrypted communication. When deployed locally, they should be configured to only listen to the localhost. If deployed separately, then they should be deployed with internal IP addresses, and protected with an external firewall that only allows the platform server access to them. The clustering software necessary for High Availability deployment should communicate on a private internal network. These services do not need to be directly accessed by users.

Content Security

Contents include Composer packs where actions, work-flows, rules, sensors, and configuration are defined. Composer has a datastore service where configuration that is used by workflows and actions can be encrypt-ed. The goal of the datastore service is to allow users to store common parameters and their values within Composer for reuse in the definition of sensors, actions, and rules. The datastore service stores the data as a key-value pair.

The key-value store allows users to store encrypted values (secrets). Symmetric encryption using AES-256 is used to encrypt secrets. The Composer administrator is responsible for generating the symmetric key used for encryption/decryption. Note that the Composer operator and administrator (or anyone else who has access to the key) can decrypt the encrypted values.

By default, getting a key tagged as secret (via –encrypt) will always return encrypted values only.

Summary

The Orchestral Composer solution is designed to enhance the security of any enterprise in which it is deployed through a variety of tools and features. The first step of leveraging these advantages is to deploy the platform itself in a secure fashion. These best practices are a high level roadmap to secure deployments, and will be built upon as new features and capabilities are introduced.

Orchestral.ai
AI-Driven Orchestration

Orchestral.ai is a team of like-minded technology professionals possessing a combined experience of over 100 years in the IT industry.

Our team is uniquely versed in building commercial web scale clouds architectures, extensive knowledge and experience in data center operations and building data centers across the globe. We also pride ourselves on our expertise in the field of system modeling for capacity planning, scaling business applications, and our focus on the user experience. The combined expertise of our team at Orchestral has been leveraged to assemble the patented technologies aimed at alleviating the pains currently plaguing the IT industry.

Contact Us

For more information, please contact our Client Development Team at **info@orchestral.ai**