SLATE and Security

Chris Weaver
July 16, 2019

What SLATE is

- **S**ervices **L**ayer **A**t **T**he **E**dge
- A federated platform for containerized edge services, to help science teams collaborate with computing providers to deploy and manage software infrastructure
- Based on Kubernetes (and so Docker)

Running a Service on SLATE

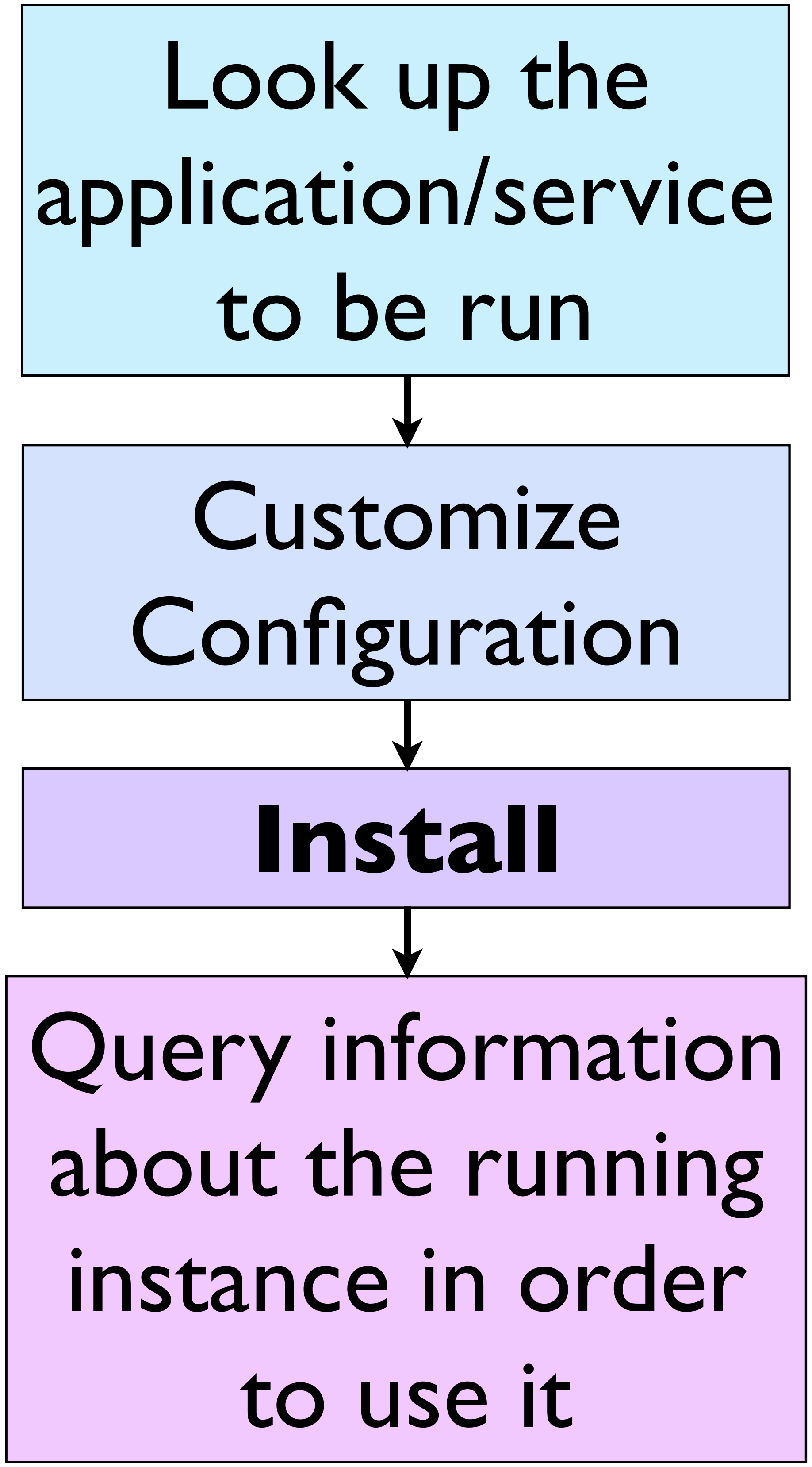
- To deploy services on SLATE one must:
  - Create an account (using institutional credentials)
  - Join or create a group
  - Get permission from the administrators of at least one cluster to run services there
  - (Optionally) download and customize a configuration file
  - Launch the service (with the configuration)

osg-frontier-squid config

```
# Instance to label use case of Frontier Squid deployment
# Generates app name as "osg-frontier-squid-[Instance]"
# Enables unique instances of Frontier Squid in one namespace
Instance: global

SquidConf:
  # The amount of memory (in MB) that Frontier Squid may use on the machine.
  # Per Frontier Squid, do not consume more than 1/8 of system memory with Frontier Squid
  CacheMem: 128
  # The amount of disk space (in MB) that Frontier Squid may use on the machine.
  # The default is 10000 MB (10 GB), but more is advisable if the system supports it.
  # Current limit is 999999 MB, a limit inherent to helm's number conversion system.
  CacheSize: 10000
  # The range of incoming IP addresses that will be allowed to use the proxy.
  # Multiple ranges can be provided, each separated by a space.
  # Example: 192.168.1.1/32 192.168.2.1/32
  # The default set of ranges are those defined in RFC 1918 and typically used
  # within kubernetes clusters.
  IPRange: 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
```

Deployment Workflow



Look up the application/service to be run

Customize Configuration

Install

Query information about the running instance in order to use it

Deploying perfSONAR

```
# Find the PerfSONAR testpoint application
$ slate app list | grep 'Name\|perfsonar'
Name                App Version Chart Version Description
perfsonar-testpoint 4.2.0       1.0.3         Perfsonar Testpoint Deployment
# Get the default configuration
$ slate app get-conf perfsonar-testpoint > ps.yaml
# Customize the configuration
$ vi ps.yaml
# Do the install
$ ./slate app install perfsonar-testpoint --cluster uchicago-prod --group slate-dev --conf ps.yaml
Successfully installed application perfsonar-testpoint as instance slate-dev-perfsonar-testpoint-cnw-test with ID instance_U-2KiIGqFKs

# Query instance information
$ ./slate instance info instance_U-2KiIGqFKs
Name                         Started                  Group     Cluster       ID
perfsonar-testpoint-cnw-test 2019-Jul-15 18:06:39 UTC slate-dev uchicago-prod instance_U-2KiIGqFKs

Pods:
  slate-dev-perfsonar-testpoint-cnw-test-84596d7c85-ns8xk
    Status: Running
    Created: 2019-07-15T18:06:44Z
    Host: sl-uc-xcache1.slateci.io
    Host IP: 192.170.227.137

# Run a test against the new endpoint
$ pscheduler task rtt --dest 192.170.227.137
Waiting for result...

1       192.170.227.137  64 Bytes  TTL 64  RTT   0.2690 ms
2       192.170.227.137  64 Bytes  TTL 64  RTT   0.1650 ms
3       192.170.227.137  64 Bytes  TTL 64  RTT   0.1170 ms
4       192.170.227.137  64 Bytes  TTL 64  RTT   0.1990 ms
5       192.170.227.137  64 Bytes  TTL 64  RTT   0.2020 ms

0% Packet Loss  RTT Min/Mean/Max/StdDev = 0.117000/0.190000/0.269000/0.051000 ms
```

SLATE User Authentication

- SLATE users create/login into accounts on a web portal (https://portal.slateci.io) using institutional credentials (which are never handled by SLATE)
- SLATE issues unique tokens which can be installed for use by the command line interface
    - Tokens can be revoked/replaced but do not otherwise expire
- All user actions are logged, which can be used to investigate security issues
    - Current system is very simple (just textual logs maintained by systemd) and has no special tools—further development may be needed

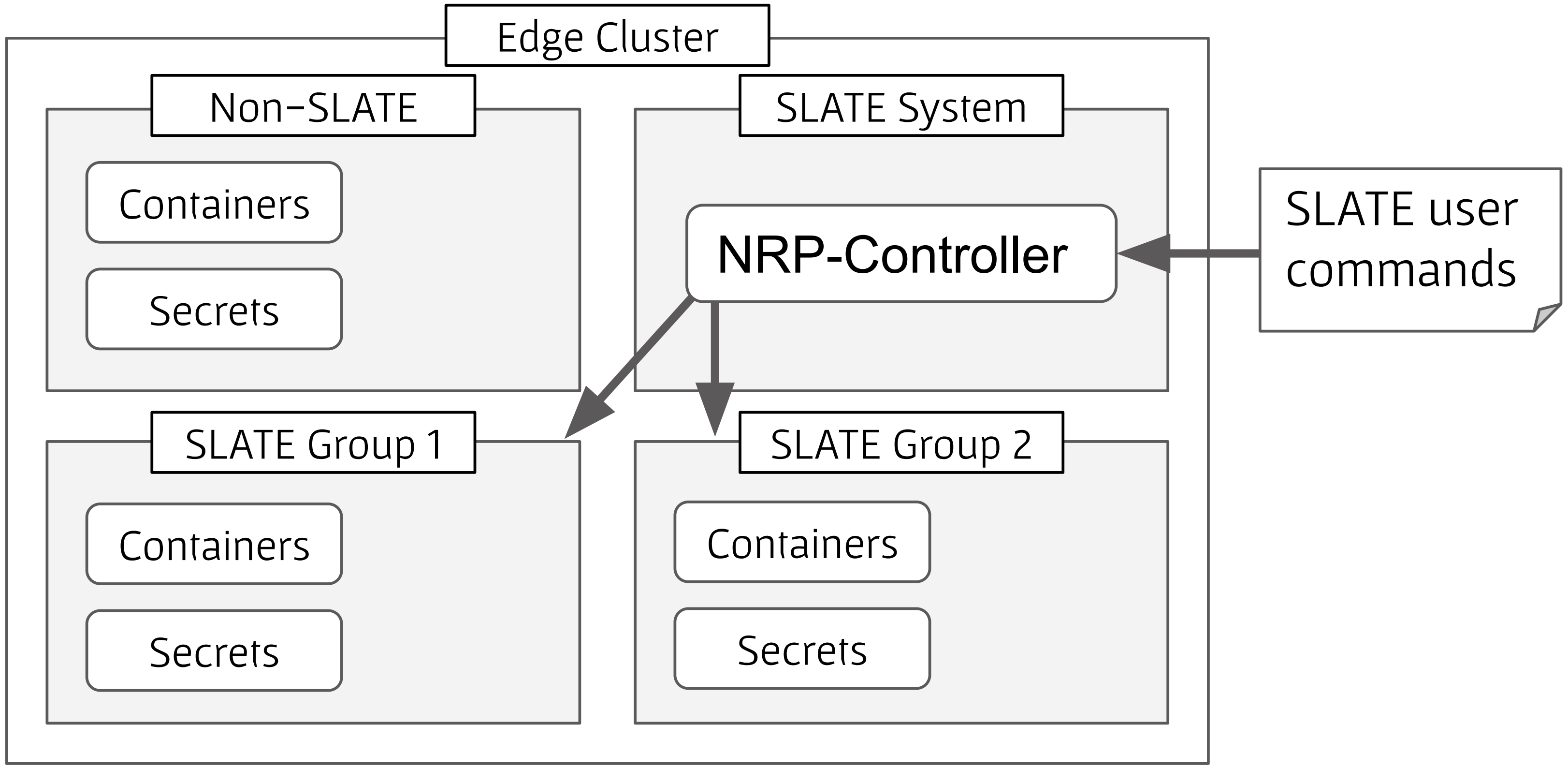
Security Considerations

- Sharing resources across sites requires a framework for trust
- Infrastructure services need to persist for long periods and often to accept incoming connections
- Site administrators do not want arbitrary software running in their data centers
- There must be suitable separation between different SLATE users and between SLATE and other uses of Kubernetes at the site

SLATE Privilege Separation

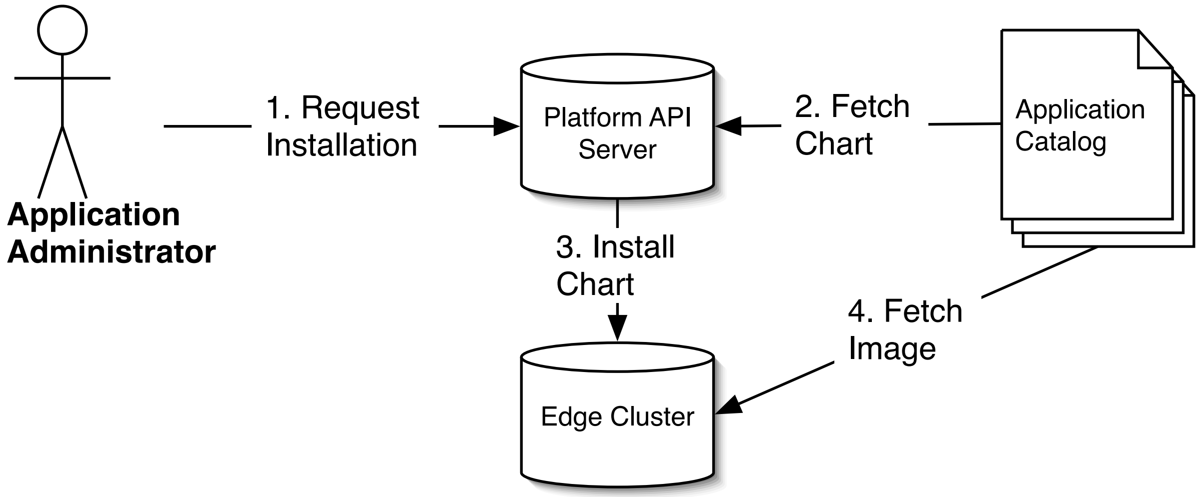
- Besides site administrators wanting guarantees that no application will misbehave on their network, users deploying applications want guarantees that other users will not interfere with their applications
- The site may also have other users of the same resources (Kubernetes cluster), and SLATE and its users should not disrupt them either

SLATE Privilege Separation



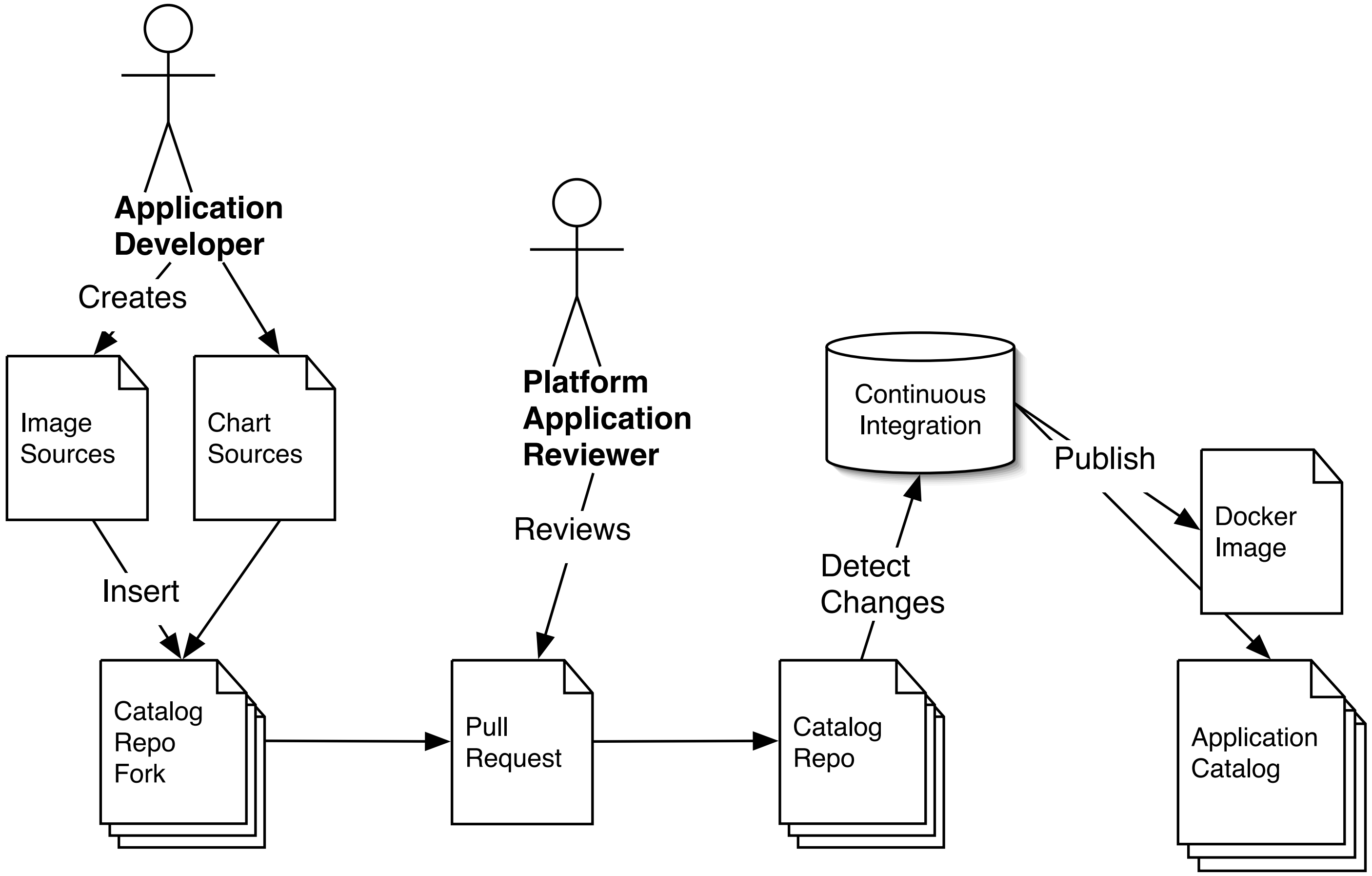
- Kubernetes provides 'namespaces' which can separate user containers and secrets
- SLATE uses an additional tool (the NRP-Controller) to mediate access only to its own namespaces

SLATE Application Security



- All SLATE applications are packaged using Helm
- Applications are registered centrally in a catalog, subject to an approval process
- Only approved applications can be launched
- Site admins can additionally whitelist which applications they authorize a given guest group to run

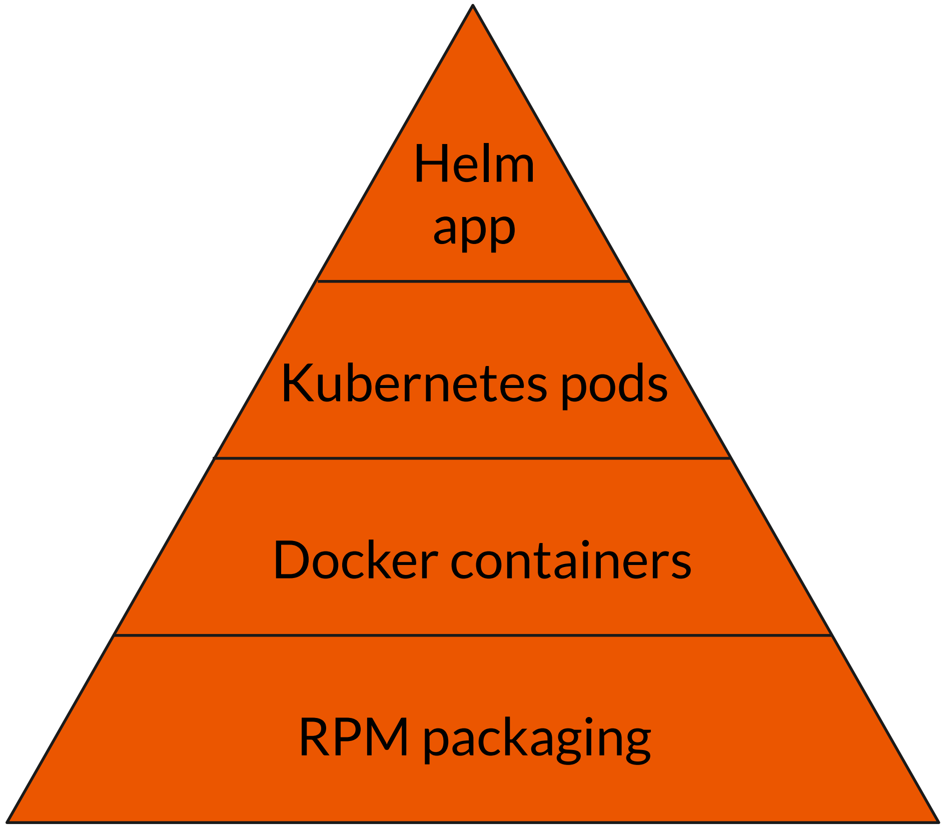
Application Approval



Before SLATE allows an application to be deployed, it must pass a review process, including both a short inspection by a human reviewer and automated checks by a Continuous Integration system

Container Image Security

- SLATE needs to extend trust to some upstream container images such as base OS images and Open Science Grid images
- This suggests that approach to image security should be developed and shared between these different layers of the grid services 'stack'

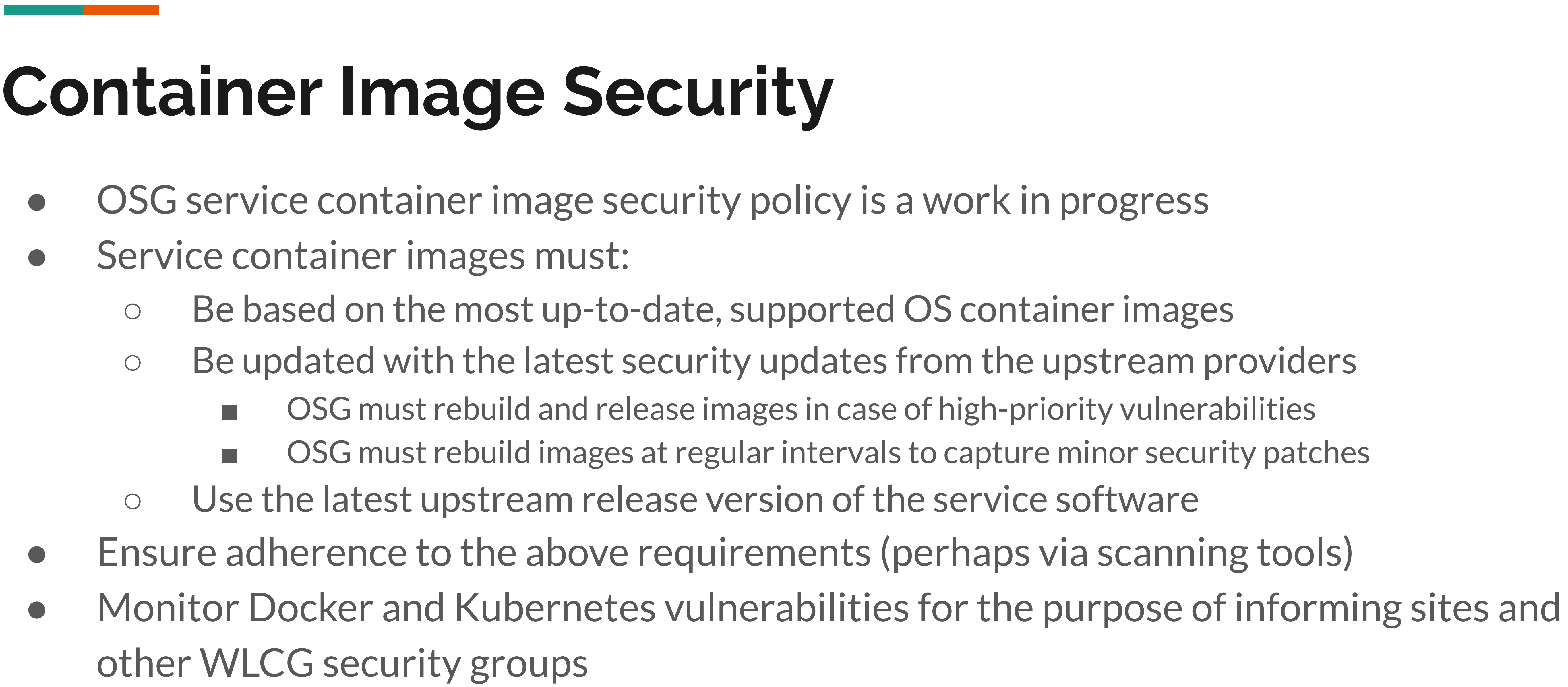


On-going Security Work

- The OSG Software Team is developing a service image security policy
- The SLATE team has just begun an engagement with the TrustedCI program to develop security policies
- We are generally interested to hear feedback from anyone in the community about the needs for the security of distributed, containerized services and how these needs might be met

Extra

(Brian Lin's Comments on OSG's Plans)



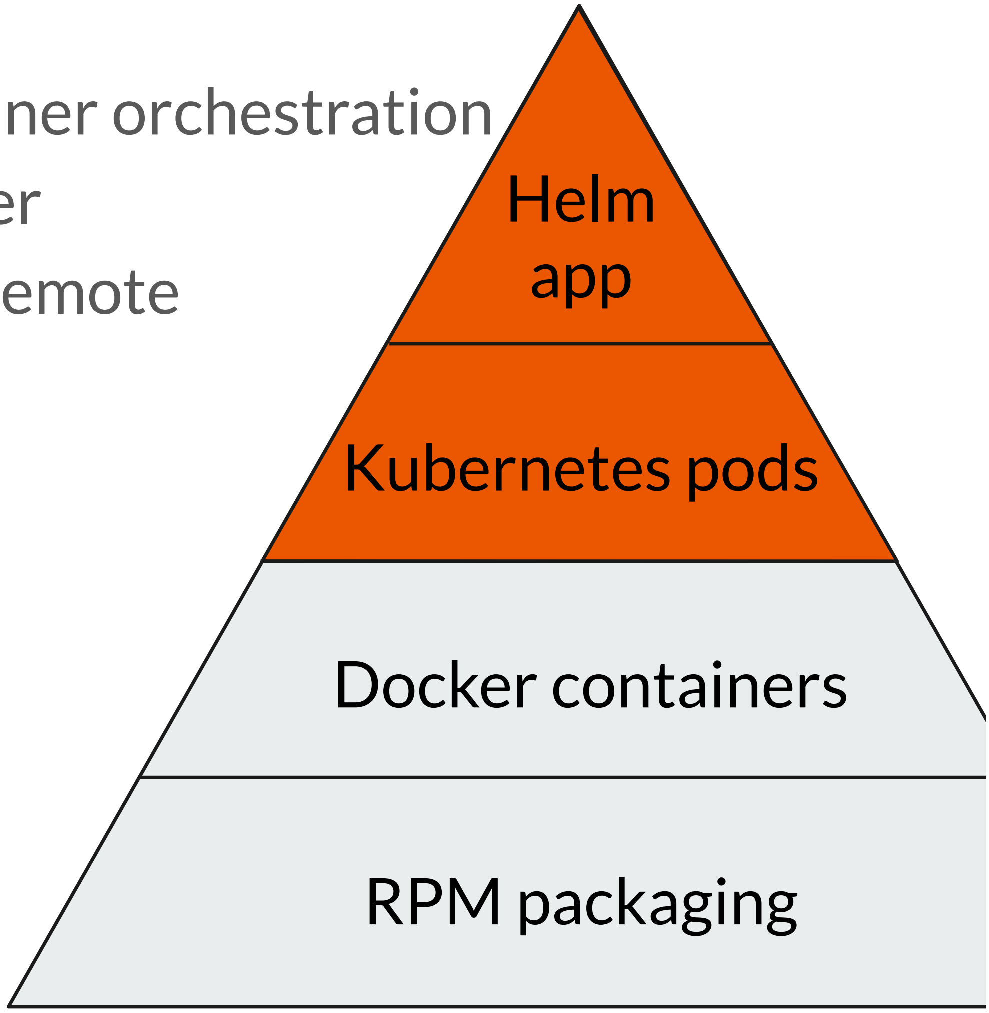
Container Image Security

- OSG service container image security policy is a work in progress
- Service container images must:
  - Be based on the most up-to-date, supported OS container images
  - Be updated with the latest security updates from the upstream providers
    - OSG must rebuild and release images in case of high-priority vulnerabilities
    - OSG must rebuild images at regular intervals to capture minor security patches
  - Use the latest upstream release version of the service software
- Ensure adherence to the above requirements (perhaps via scanning tools)
- Monitor Docker and Kubernetes vulnerabilities for the purpose of informing sites and other WLCG security groups

(Brian Lin's Comments on OSG's Plans)

Downstream Security

- Many consumers of OSG images (SLATE, PRP) use container orchestration tools instead of directly running the containers via Docker
- SLATE maintains a catalog of Helm apps that are run on remote Kubernetes clusters
  - Apps must pass a human review process before being added to the catalog
  - User separation between apps at a site is guaranteed by Kubernetes namespaces
  - Details described in an upcoming PEARC paper
  - Security posture undergoing review by Trusted CI
- OSG's Frontier Squid and ATLAS XCache images act as the base for their respective SLATE apps


Helm app
Kubernetes pods
Docker containers
RPM packaging