1. Introduction

1.1 O² and the O²/FLP Project

The Online-Offline (O²) Project is the upgrade for the ALICE data acquisition system for Run 3 and 4. A core feature is to reduce the amount of data collected as quickly as possible. Integral to this are the First Level Processors (FLPs), who take data coming in from the detector links and perform initial selection and local processing of data.

1.2 Foreman

To maintain an inventory of the roughly 250 FLP servers and to perform life-cycle management we use Foreman. All machines in the FLP server farm are configured to boot from the Foreman instance via PXE. Foreman distributes a special discovery image which when booted registers the booting machine in Foreman. Foreman then sorts these machines into host groups. These host groups specify the purpose of the machine and come with associated Ansible roles. The machine is rebooted into a standard CERN CentOS 7 image and after installation is provisioned with the Ansible roles associated with the machine’s host group.

2. Contributions

2.1 Cockpit

Cockpit is a web interface to access diagnostic information of a machine. This is important to debug issues affecting the performance of servers in an easy way. However, for convenience reasons we would prefer to have this interface embedded into Foreman, to achieve an all-in-one solution to server management.

Fortunately there exists a plugin for Foreman that does exactly that. This plugin detects if Cockpit is running on a machine and then creates a drop-down menu in that host’s dashboard. Each item in the drop-down embeds a single tab of the Cockpit interface.

One hurdle is that Cockpit’s web interface is encrypted with self-signed certificates by default. When embedded in the Foreman dashboard, a certificate error is displayed and due to the method of embedding, no way to accept the certificate is available. We solve this issue in the next section.

2.2 Certificates

There are several reasons to use certificates signed by the CERN Certification Authority (CA): To enable the embedding of Cockpit, to avoid a certificate prompt when navigating to a new Foreman instance and to use the CERN Single-Sign-On mechanism.
CERN provides tools to make setting up certificates mostly painless. The website of the CA offers an interface for a server’s main user or responsible person to set it up for auto-enrollment. That means that a certificate requested from the CA will be renewed automatically regularly to prevent it from running out. The machine additionally needs to be configured for the CERN Kerberos environment. Finally, a simple executable delivered by the CERN CentOS repositories retrieves the certificate.

2.3 Single-Sign-On
Single-Sign-On (SSO) enables us to rely on the CERN authorization structure (user accounts and e-groups) instead of managing our own authentication.

The simplest way to do this is to use SAML2 via Apache's mod_auth_mellon. Foreman is configured to use an external authentication source and now expects to retrieve user information via environmental variables. A mapping is created between (external) e-groups and (internal) user groups to manage permissions for specific e-groups. The Apache module is installed and configured, once again with help of the CERN CentOS repositories. Only specific e-groups are whitelisted to use the SAML endpoint in order to provide an additional layer of security. Finally the application is registered in CERN’s SSO web interface.

2.4 Configuration
For operations during Run 3, a Foreman instance at Point 2 is required to manage the FLP server farm. As a test setup during detector commissioning, a vertical slice consisting of the Foreman instance and some FLPs and service machines is set up.

An test instance of Foreman was already running in the O² lab. However extracting all the configuration changes made to this machine was difficult. Thus it was important to us to move all configuration to Ansible, in order to have a record of changes and being able to reproduce the configuration easily. This has the added benefit that an installation of Foreman managed by Ansible is largely hands-off. Foreman offers both a REST API and also modules for Ansible that support its configuration. We rely on these modules as much as possible, as those adhere to Ansible’s descriptive and idempotent execution model the closest.

The final configuration features two host groups, one for FLPs and one for machines running other services. Autodiscovery rules are created to sort newly discovered machines into these host groups based on their hostname. Ansible roles are associated with those host groups and discovered machines are provisioned using those roles.

3 Future and Unfinished Work
During my thirteen week stay I started work on some tasks that I did not have enough time to complete. In this section, I’ll briefly touch on what tasks were left unfinished and what was blocking them.

3.1 Virtualization
Foreman understands the concept of "Compute Resources", which for our purposes can be seen as ways to allocate resources on a host. Relevant for us is the Libvirt provider type, which declares that a host is capable of creating and running virtual machines. After creating a Libvirt resource for a host, it is possible to deploy virtual machines from Foreman’s Create Host dialog.

Given the CERN network topology, we need to create virtual machines with specific MAC addresses to ensure that they get assigned the desired hostname. There are two problems with that in a default Foreman installation: On the one hand, we need to have a fixed desired-hostname-to-mac-address mapping that Foreman gives us no way of storing. On the other hand, if entering a MAC address in the
interface section of the Create Host dialog, that address is not passed to Libvirt on the host and instead a random one is assigned.

A potential way forward are hook scripts by using the `foreman-hooks` plugin. These are scripts written in Bash or Python that are executed by Foreman whenever certain tasks are executed. These would allow intercepting the virtual machine creation request and updating its definition with the desired parameter. The hook script would need to run partially on the Foreman instance and partially on the Libvirt host for this.

### 3.2 Autodiscovery

A central feature of the Foreman setup at Point 2 is the automatic discovery and provisioning of machines by PXE-booting a specific discovery image from Foreman. This image tells Foreman facts about the new system which Foreman then uses to categorize the machine into one of the host groups. The server is then rebooted into the default CERN CentOS installation image and a Kickstart file is provided on the command line to manage the remaining provisioning.

However retrieving the Kickstart file fails due to a complicated interplay between certificates and the proxy-and-server structure of Foreman. Without the Kickstart file, installation halts halfway through.

### 4 Summary

Aside from the major parts of my project, I also worked on several small tasks, such as quality-of-life improvements to the deployment playbooks, bug fixes and upgrading software versions. Overall, my project hopefully helped make the deployment of the Point 2 infrastructure a lot smoother.

On a personal level, my stay at CERN was a great opportunity for a look "behind the scenes". I very much enjoyed my time there and feel like I learned a lot about the management of IT infrastructure for a large physics experiment.