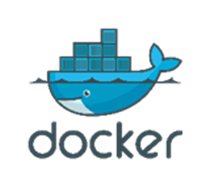
∞

Taking Cyber Security Seriously

**Expandable Defensive Cyber Operations Platform  
Configuration Guide – Node Servers (CentOS)**

****



****

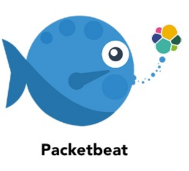












Sealing Technologies, Inc.

7134 Columbia Gateway Dr.

Suite 160

Columbia, MD 21046

|  |  |  |
| --- | --- | --- |
| Version | Author | Notes |
| 1.0 | Markus & Michael | Initial Draft. Program Reviews populated with instructions. |
|  |  |  |
|  |  |  |
|  |  |  |

Contents

[Introduction to the EDCOP 4](#_Toc493084953)

[Overview 4](#_Toc493084954)

[Basic Requirements 4](#_Toc493084955)

[Docker 4](#_Toc493084956)

[Installation 4](#_Toc493084957)

[Creating Docker Containers 6](#_Toc493084958)

[Environment Requirements 6](#_Toc493084959)

[Building the Containers 7](#_Toc493084960)

[Rancher 9](#_Toc493084961)

[Environment Requirements 9](#_Toc493084962)

[Adding Node Hosts 9](#_Toc493084963)

[Kubernetes 11](#_Toc493084964)

[Labeling the Node Server 11](#_Toc493084965)

[Running Node Applications 13](#_Toc493084966)

[Kibana 14](#_Toc493084967)

[Security Roles: Optional X-Pack Configuration 14](#_Toc493084968)

[Defining your Index Patterns 17](#_Toc493084969)

[Open vSwitch 18](#_Toc493084970)

[Installation 18](#_Toc493084971)

# Introduction to the EDCOP

This guide provides an overview of the Expandable Defense Cyber Operations Platform (EDCOP) concepts as well as configuration information, examples, and reference information pertaining to the tools utilized in the platform.

## Overview

The virtualized Network Security Monitor (NSM) supports security features that can protect your network against degradation as well upholding the confidentiality and integrity of the system through continuous monitoring.

The following components are covered in this configuration guide:

* Docker
* Rancher
* Kubernetes
* Elasticsearch
* Logstash
* Redis
* Filebeat
* Moloch Viewer/Sensor
* Packetbeat
* Suricata
* Bro Network Monitor
* Open vSwitch

For more information on the core applications, see the concept of operations (CONOPS) document.

## Basic Requirements

Before you begin this guide, it is assumed that you have already completed the installation of the master server by following the **EDCOP – Master Configuration Guide**. Before you begin to work on the node servers, you will need them to have CentOS 7, static IP addresses, **unique hostnames**, and the ability to communicate with the master server as well as the internet.

# Docker

The setup of Docker is the same as the master server since they require the same versions.

## Installation

In order to utilize the EDCOP, setup of Docker is necessary.

1. Initially, ensure there are no older versions of Docker, or the Docker-engine by uninstalling the software alongside the dependencies.

$ sudo yum remove docker \

docker-common \

container-selinux \

docker-selinux \

docker-engine

1. Enable the EPEL repository:

$ sudo yum -y install epel-release

1. Install the required packages.

$ sudo yum install -y yum-utils device-mapper-persistent-data lvm2 git wget net-tools python-pip python-wheel

1. Add the docker repository:

$ sudo yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

1. Update the *yum* package index

$ sudo yum makecache fast

1. Install version 17.06 of Docker.

$ sudo yum -y install docker-ce-17.06.0.ce-1.el7.centos.x86\_64

1. Edit the /etc/docker/daemon.json file. If it does not exist, simply create it.

$ sudo vi /etc/docker/daemon.json

The file should have the following contents:

{

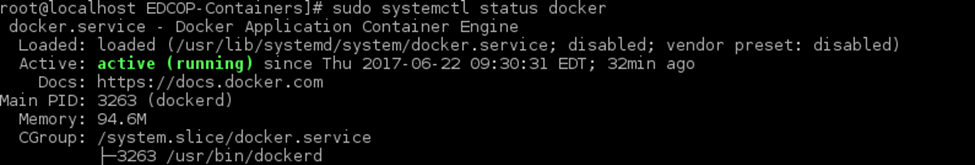
"storage-driver": "devicemapper"

}

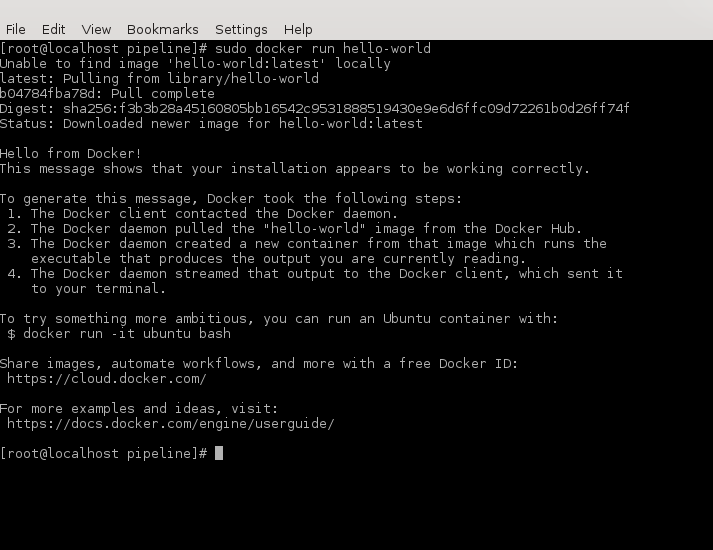
1. Start Docker

$ sudo systemctl start docker

1. Docker should now be up and running.

$ sudo systemctl status docker

1. To ensure that docker is correctly installed, run the hello-world image.

$ sudo docker run hello-world

For more information on Docker, please visit <https://doc.docker.com/engine/reference/commandline/docker>

# Creating Docker Containers

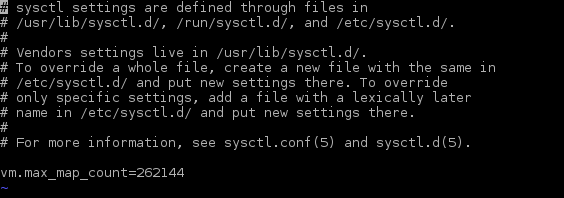
To begin setting up the node applications, we need to create containers for them to run inside.

## Environment Requirements

1. Before creating your containers, you will need to increase your virtual memory map count to run Elasticsearch. The bare minimum is 262144, but feel free to increase the value even further:

$ sudo sysctl -w vm.max\_map\_count=262144

\*Note: this change will be reset anytime you restart your system. To make the changes permanent, you will need to edit the */etc/sysctl.conf* file, add the following line:

$ vm.max\_map\_count=262144

1. You will also need to create multiple directories for your applications to store their data into. These directories will be the heart of our nodes, as everything will be formatted and stored here. Additionally, you will need to give these directories permissions.

$ sudo mkdir /esdata /data /data/moloch /data/moloch/raw /data/moloch/logs /data/suricata /data/bro

$ sudo chmod 757 /esdata /data/bro /data/suricata

$ sudo chmod a+rwx /data/moloch/raw /data/moloch/logs

## Building the Containers

For this section, we are going to be using the **current** node’s IP address for all *$ESURL* instances. Do not use the master’s IP for any of the changes below!

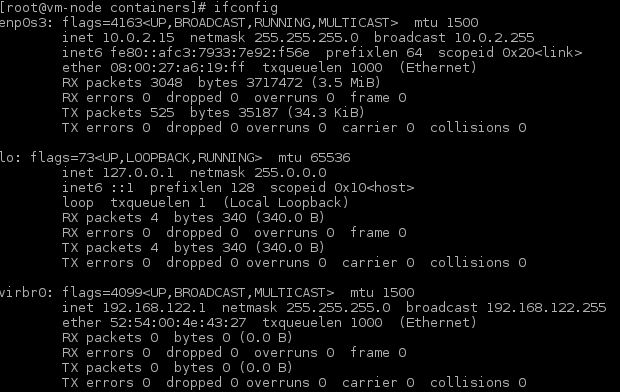
1. Now that we’ve made the preparations for running our containerized platform, you can clone the required files from our official GitHub page into an empty directory:

$ git clone https://github.com/sealingtech/edcop-cluster <$DIRECTORY>

1. Once the files finish copying, change directories into the newly created directory. Afterwards, change directories into the **“containers/minion”** directory , and then give the *buildcontainers.sh* script permissions to run as an executable:

$ sudo chmod +x buildcontainers.sh

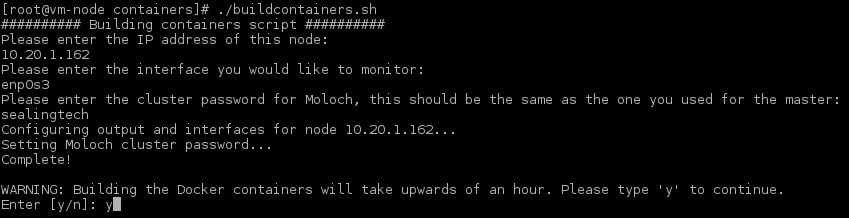
1. Now you’re going to need to know your node’s IP address, the password you used for Moloch, and which interface you’re going to monitor. You can see a list of interfaces by using the ***ifconfig*** command:

$ sudo ifconfig

1. Once you have all of the information you’re going to need, run the script using the following command from within the directory:

$ ./buildcontainers.sh

***\*Note: This will create container images for all the Master EDCOP applications that run within Docker. You may also use this script to make changes to the configuration files and then rebuild the images. However, this script does NOT run the containers because Kubernetes will ultimately be responsible for that in a production environment.***

1. Follow the on-screen instructions and then the script will build your worker node containers using the information you have provided.

# Rancher

## Environment Requirements

Before adding your nodes as Rancher Agents, you must have a **supported version** of Docker installed and are required to open ports on each host to allow traffic to pass through. To discover which versions of Docker are supported, please visit: <http://rancher.com/docs/rancher/v1.6/en/hosts/#supported-docker-versions>. Once you have obtained a supported version, **do not** yum update the entire system, as it will install the latest version of docker and break your environment!

1. Before adding the node to Rancher, expose the following ports and reload your firewall:

$ sudo firewall-cmd --zone=public --add-port=8080/tcp --permanent

$ sudo firewall-cmd --zone=public --add-port=8080/udp --permanent

$ sudo firewall-cmd --zone=public --add-port=8005/tcp --permanent

$ sudo firewall-cmd --zone=public --add-port=8005/udp –permanent

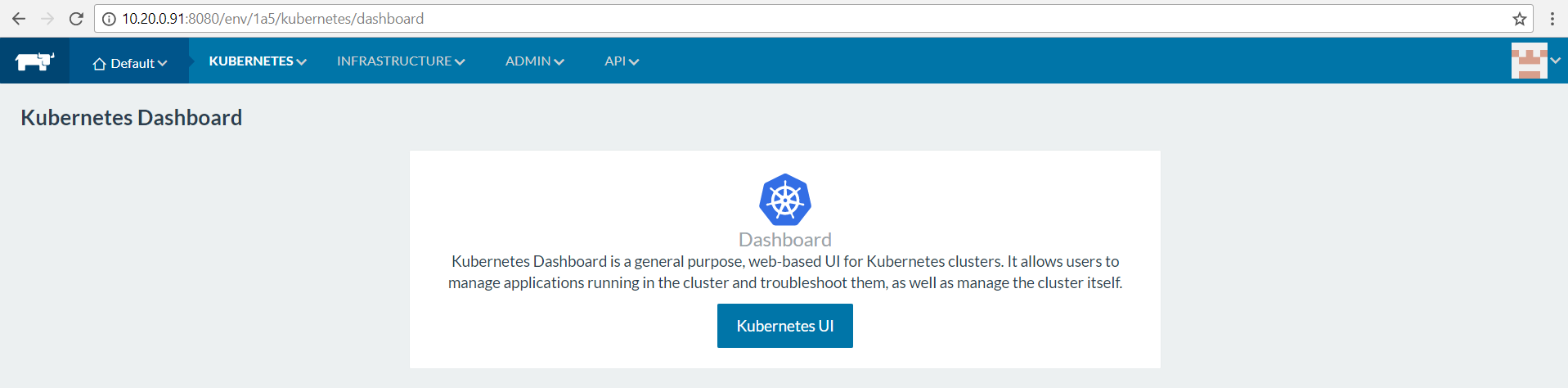
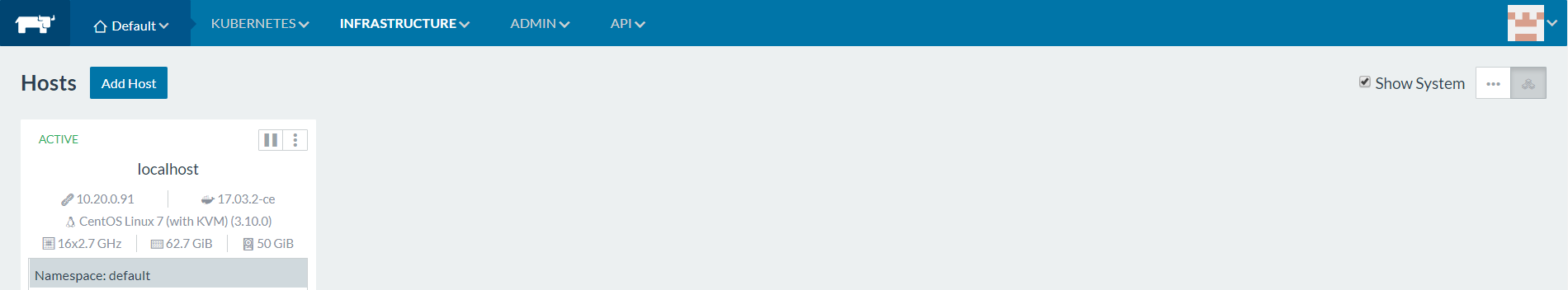
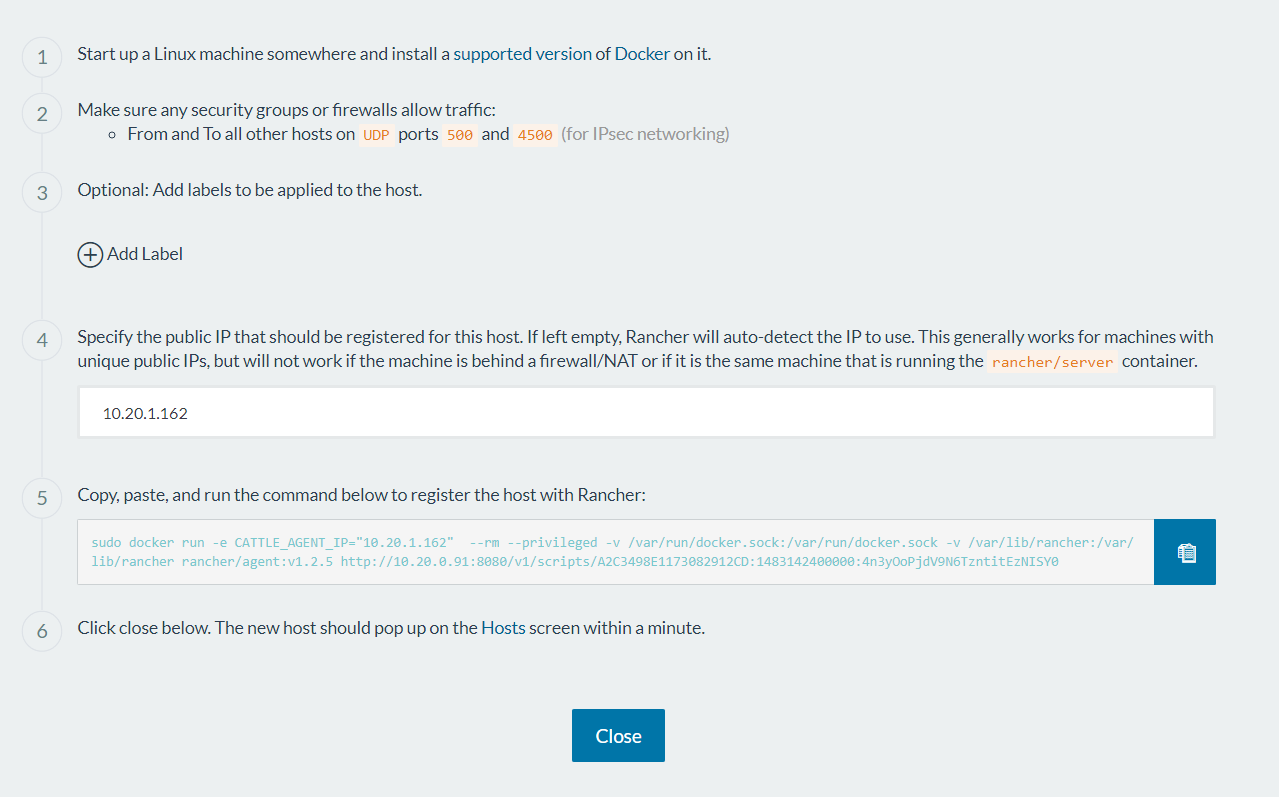
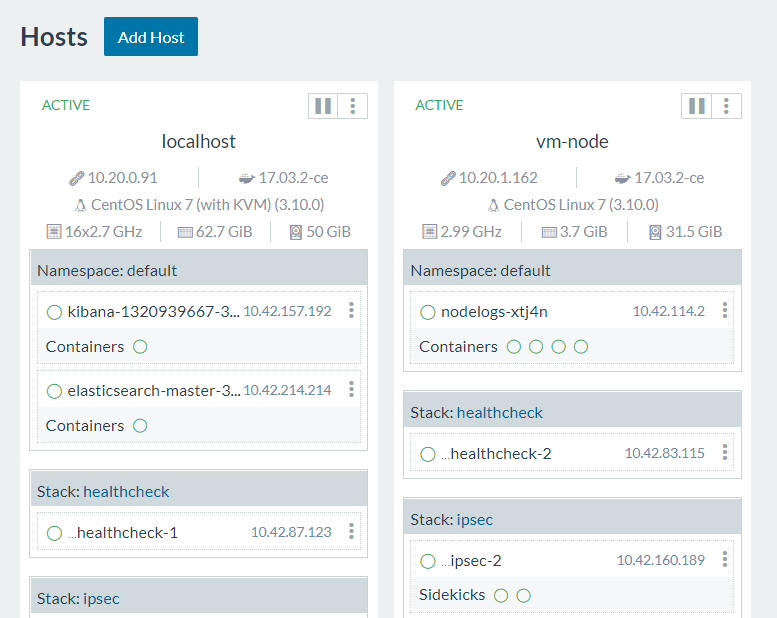
$ sudo firewall-cmd --zone=public --add-port=4500/udp --permanent

$ sudo firewall-cmd --zone=public --add-port=500/udp –permanent

$ sudo firewall-cmd --reload

## Adding Node Hosts

Before adding your nodes into Rancher, make sure they have **unique hostnames**, otherwise the automated setup of Kubernetes will fail.

1. From your internal network, navigate to Rancher’s web interface at http://<$IP-ADDRESS>:8080.
2. Navigate to the *Hosts* page from within the *Infrastructure* tab:
3. Click on *Add Host* and then input the IP address of the node you’re adding. Once it populates, copy the generated command and click close.
4. Switch back to the command line on the host you’re adding, and paste the command:
5. Wait for Rancher and Kubernetes to start up all of their agent/node services and then check the web interface to make sure the node has been added:
6. You can add other nodes running a supported version of Docker by repeating these steps, but make sure their environments are set up correctly and will allow connections through the firewall. **\*\*IMPORTANT: If you are adding multiple nodes to Kubernetes, their hostnames MUST be different or it will fail!**

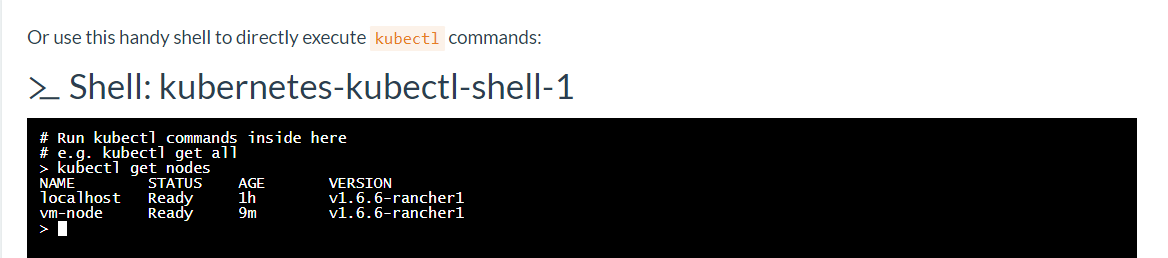
# Kubernetes

Since Kubernetes has already been set up, all we have to do is label our nodes and then run our applications and we’re good to go!

## Labeling the Node Server

In order to prevent master applications from running on our node servers, we need to label the nodes accordingly.

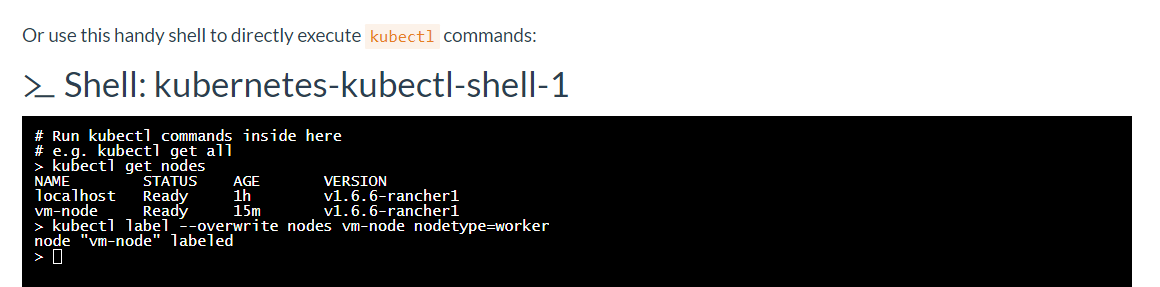
1. Navigate to the *CLI* page within the Kubernetes tab in order to use the shell:
2. Your node name should be the same as the name of the host, but to make sure begin by getting the information of the current nodes by typing:

$ kubectl get nodes

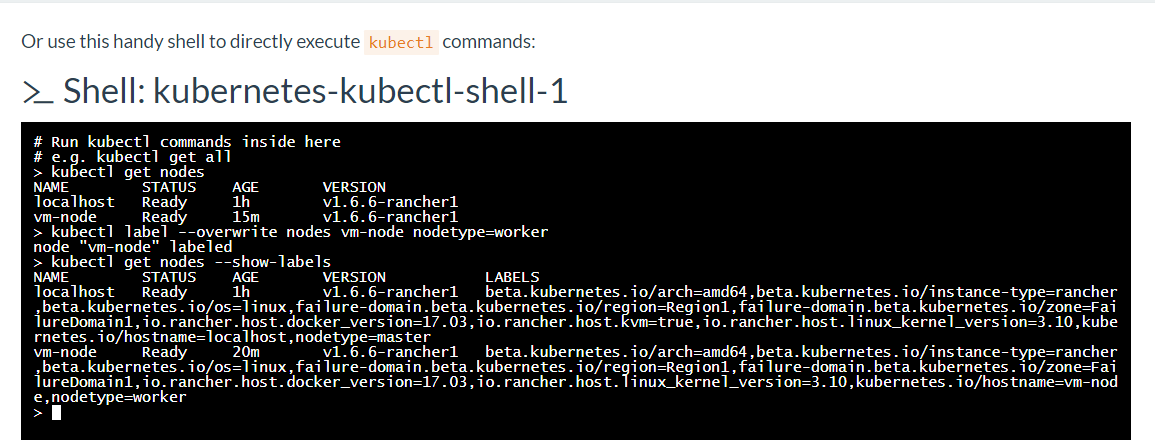
*\*Note: In a production environment, your host names* ***must*** *be unique, do not leave them as the default!*

1. Now label the new node as a worker by using the following:

$ kubectl label --overwrite nodes <$NODE-NAME> nodetype=worker



1. Make sure the label was applied correctly by checking the node information with the flag “show-labels”:

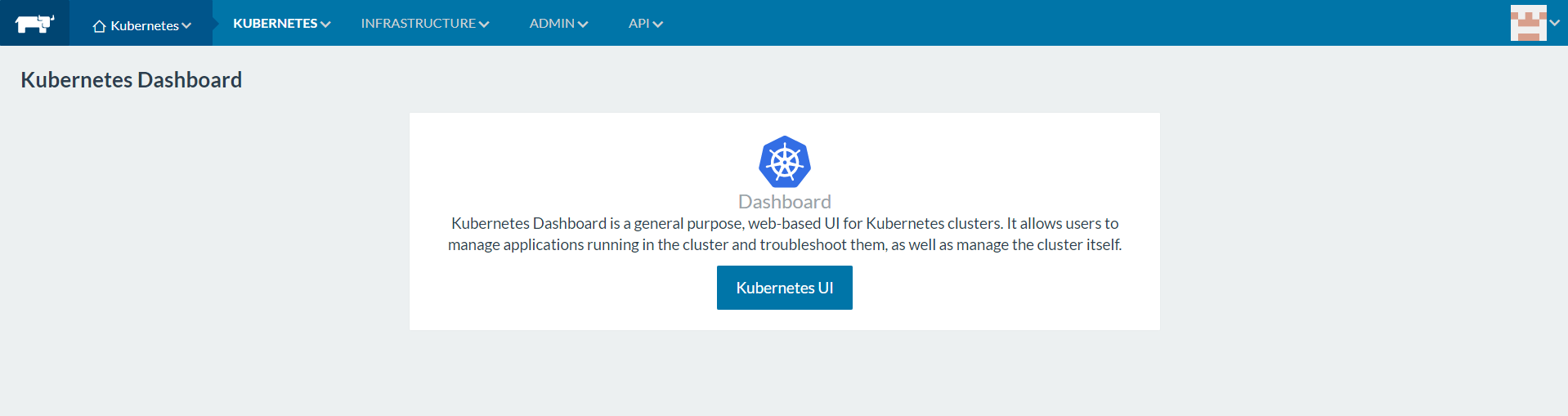
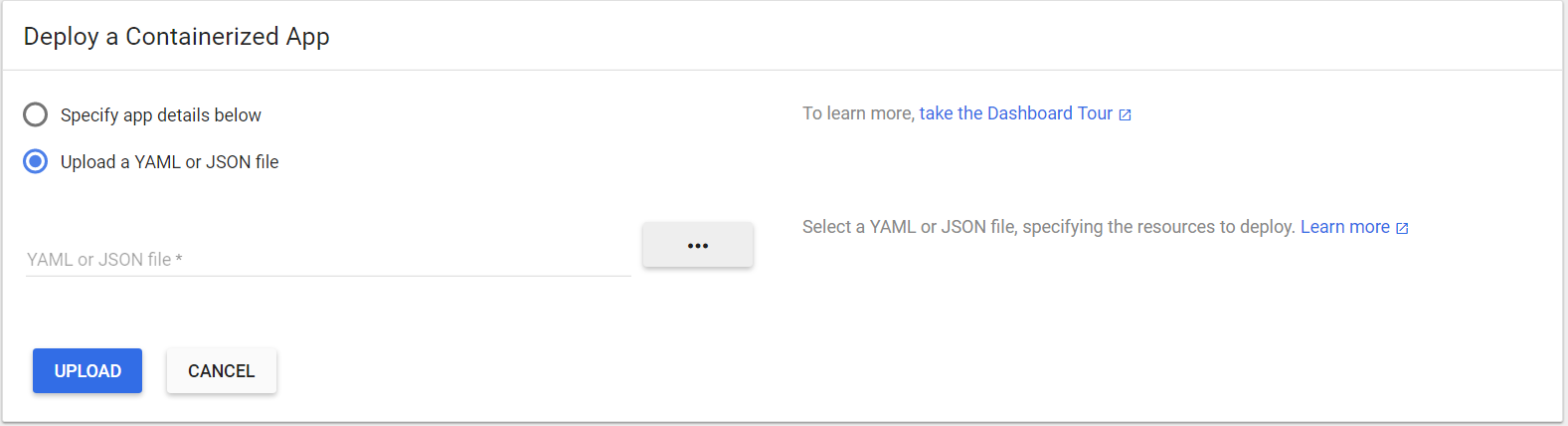
$ kubectl get nodes --show-labels

## Running Node Applications

Just like before, we’re going to be uploading the rest of our YAML files to deploy our applications.

1. If you’re accessing the Rancher + Kubernetes webpages from a different host than before, or just need a new copy of the files, you can clone them again using the following command:

$ git clone https://github.com/sealingtech/edcop-cluster <$DIRECTORY>

1. Open the Kubernetes web dashboard through Rancher’s Kubernetes tab:
2. Click on create in the upper right hand corner.
3. Select “Upload a YAML or JSON file” and then click on the three dots to search for your files.
4. Please upload the application files in the following order, and be sure to upload the service YAML file **before** its corresponding application file:
5. NodeLogs + service file
6. Suricata
7. Bro
8. Packetbeat
9. Moloch-Nodes

*\*Note: Most of your applications will not have service files, only the applications within the nodelogs pod need to accept traffic from external services.*

1. Reload the firewall from the current host’s command line now that Moloch has started:

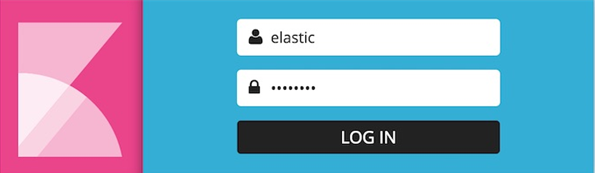
$ sudo firewall-cmd --reload

After you start all of your applications and services, you will be able to access Moloch at http://<HOST-IP>:8005 and the Kibana dashboard at http://<HOST-IP>:30002 .

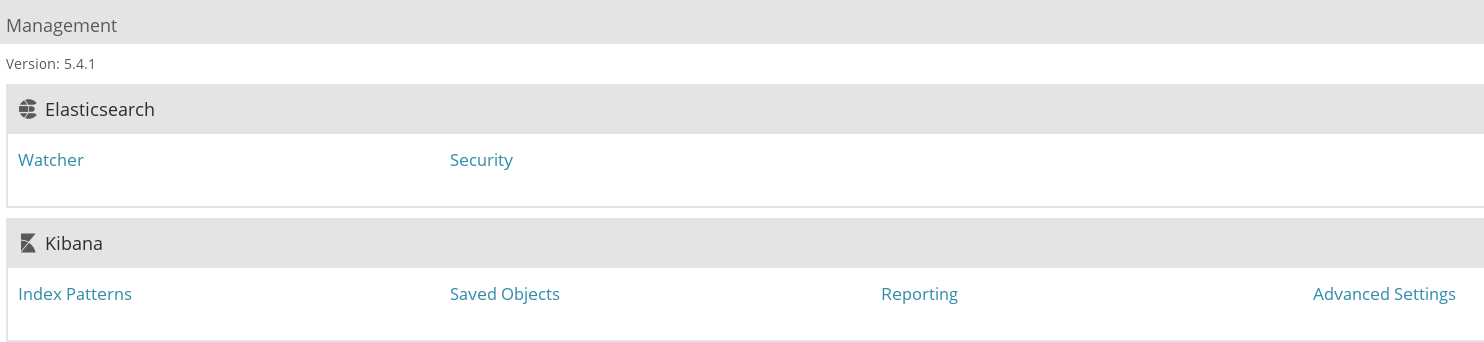
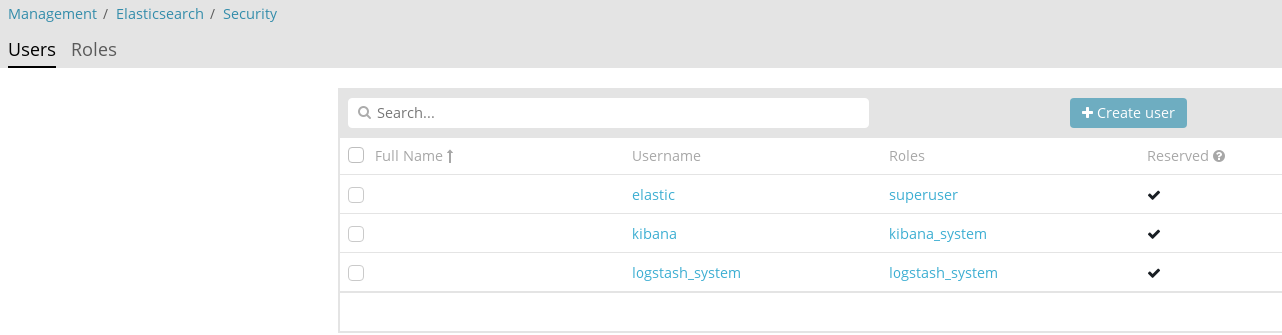
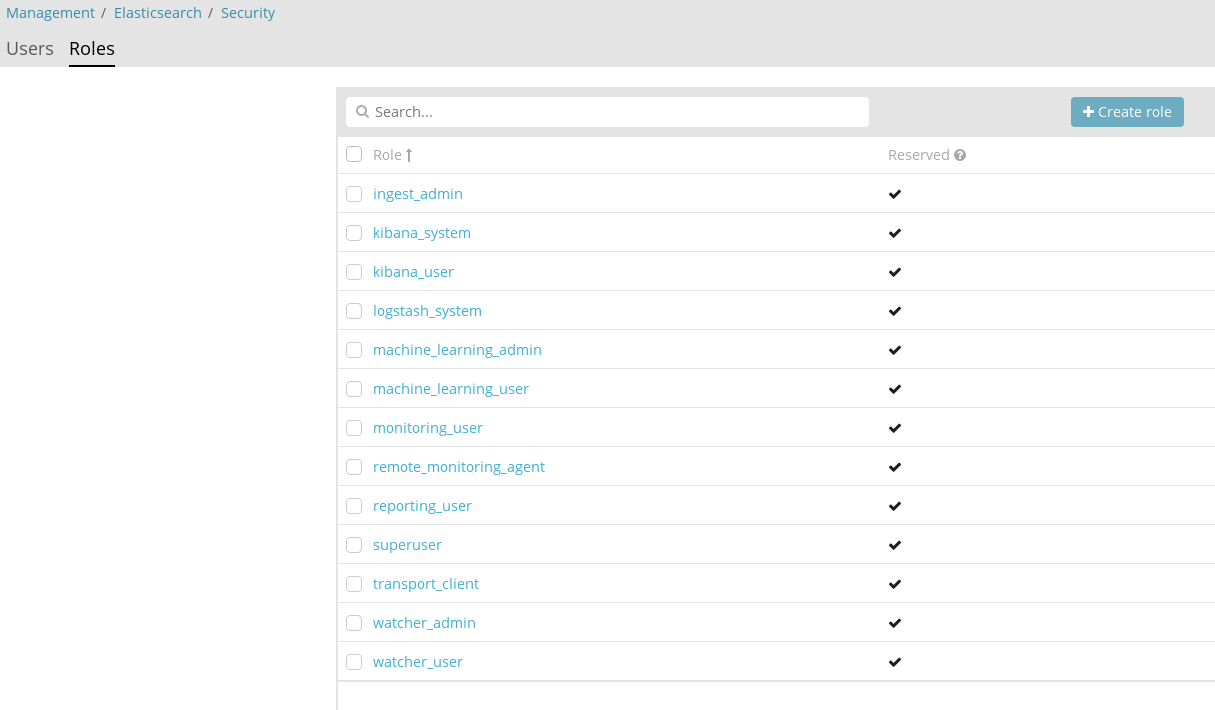
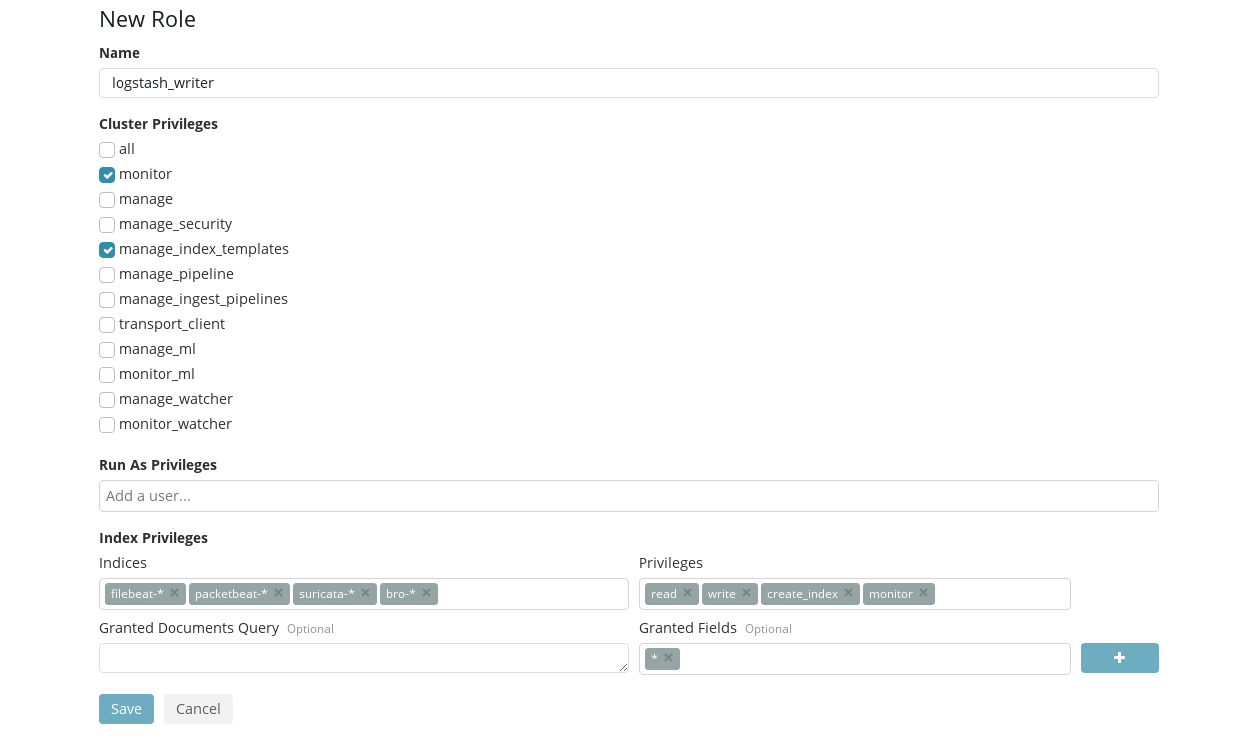
# Kibana

After running all of your applications within Kubernetes, you should be able to access Kibana at http://<$MASTER-IP>:30002. From there, you will be able to visual, query, and interact with all of the data you’ve collected. If you are planning on using Elastic’s security features suite (X-Pack), Kibana will use the following credentials:

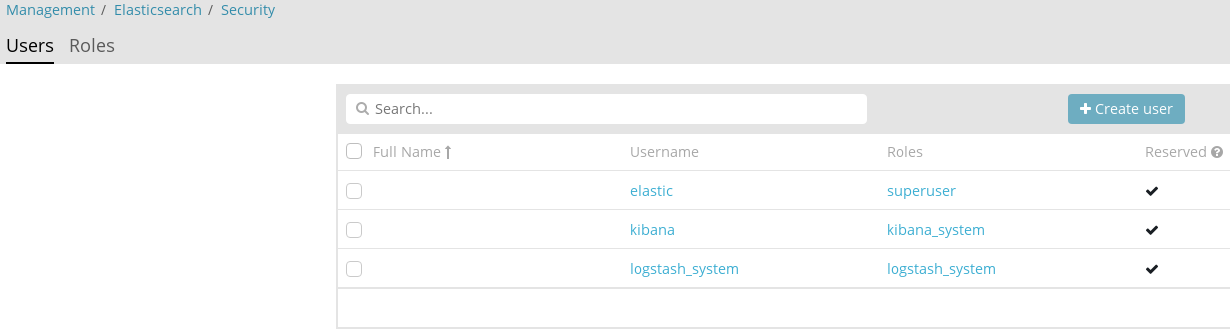
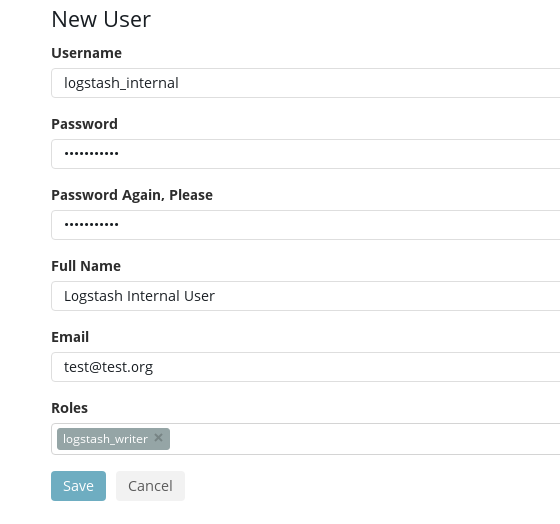
* Username: elastic
* Password: changeme



## Security Roles: Optional X-Pack Configuration

1. **Before** **you attempt to set this up, you will need to enable the disabled X-Pack settings in the Elasticsearch, Logstash, and Kibana yml files and uncomment the user/pass config in the Logstash pipeline files on ALL of your hosts.** These settings also require a paid elastic subscription service plan, which are available at <https://www.elastic.co/subscriptions>. To begin, you will need to log in to Kibana to set up a “writer” account for logstash. Currently, Logstash does not have the permissions to do its indexing job, and will fail to send any data to Elasticsearch for security reasons. After you log in, navigate to the **Management** tab towards the bottom of the services list.
2. Click on Security within the Elasticsearch options:
3. Switch to the Roles tab at the top of the page:
4. Now click on create role at the top right side of the page:
5. Create a new role named “*logstash\_writer*” and assign it *monitor* and *manage\_index\_templates* cluster privileges. Then create four separate indices, named *filebeat-\**, *packetbeat-\**, *suricata-\**, and *bro-\**. Assign Logstash *read*, *write*, *create\_index*, and *monitor* privileges for these indices, and finally click save when you’re finished.

***\*Note: The names of these settings are very important! Each of our services will be stored in a separate index for visualization purposes. The index names correspond to their respective service, and will allow Logstash to manage the indices effectively.***

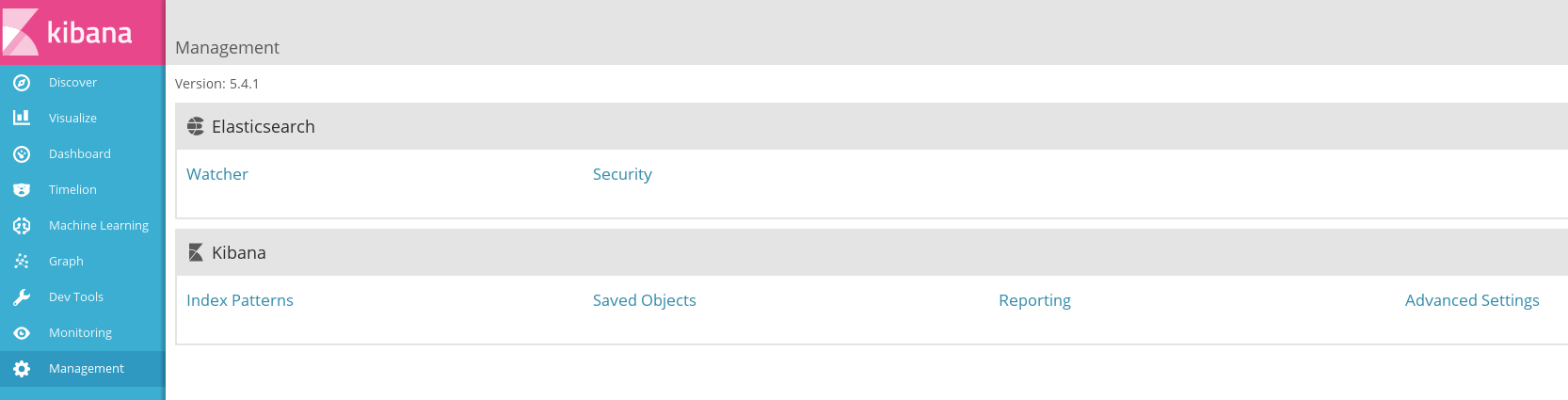
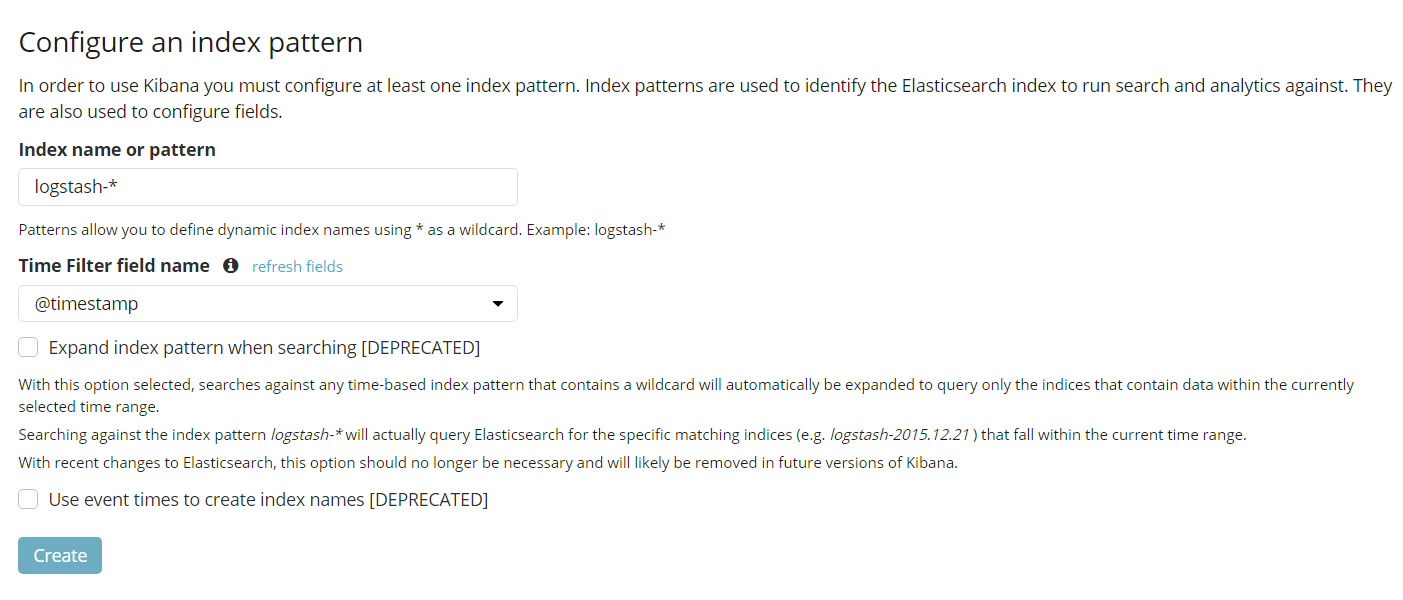
1. Switch back over to the Users tab, and create a new user:
2. Create a user named “*logstash\_internal*” and assign it the *logstash\_writer* role you just created. By default, the password used in the configuration files is “changeme”, but this can and should be changed within all the *~/logstash/pipeline* files.

***Note: The full name and email do not matter, but the username, password and roles must stay the same. If you change the username or password, you will need to edit the Logstash pipeline files to match these changes.***

To activate or check your X-Pack membership and license information, please visit: <https://www.elastic.co/guide/en/x-pack/current/installing-license.html>

For additional security information related to X-Pack, please refer to: <https://www.elastic.co/guide/en/x-pack/current/logstash.html>

## Defining your Index Patterns

1. To see your data, you must set up Kibana to point to the right indices. However, Kibana cannot access these indices if there’s no data in them! Make sure your applications are sending data to Elasticsearch before attempting to set up their respective indices or it will fail. To begin, log in to Kibana and then navigate to the Management tab.
2. Click on *Index Patterns* within the Kibana section. If this is your first time setting up an index, it will default to asking for a name. If you are adding another index, click on the plus sign at the top above your existing index names and it will bring you to the same screen.
3. All the index names should be structured *$APPNAME-\** so that Kibana can find all entries associated with that application or service. Once you enter a correct index name, the Time-field name option will populate and allow you to change it. Make sure the Time-field name is set to *@timestamp* and then click create to finish. Repeat for all of your different services.

***Indices: logstash-\*, packetbeat-\*, suricata-\*, bro-\****

# Open vSwitch

Open vSwitch will be an integral part of our Kubernetes networking because it allows us bridge our physical interfaces to allow traffic to flow in and out of the box. By bridging our interfaces, this also allows Suricata and Bro to accept and pass on incoming traffic.

## Installation

1. Install the required dependencies:

$ yum install -y make gcc openssl-devel autoconf automake rpm-build redhat-rpm-config python-devel openssl-devel kernel-devel kernel-debug-devel libtool wget python2-pip

1. Make a directory for building the RPM:

$ mkdir -p ~/rpmbuild/SOURCES

1. Download Open vSwitch:

$ wget http://openvswitch.org/releases/openvswitch-2.7.0.tar.gz

1. Copy the files into the directory we created:

$ cp openvswitch-2.7.0.tar.gz ~/rpmbuild/SOURCES/

1. Extract the files:

$ cd ~/rpmbuild/SOURCES

$ tar xfz openvswitch-2.7.0.tar.gz

1. File edits in order to build the RPM:

$ sed 's/openvswitch-kmod, //g' ~/rpmbuild/SOURCES/openvswitch-2.7.0/rhel/openvswitch.spec > ~/rpmbuild/SOURCES/openvswitch-2.7.0/rhel/openvswitch\_no\_kmod.spec

1. Build the RPM:

$ rpmbuild -bb --nocheck ~/rpmbuild/SOURCES/openvswitch-2.7.0/rhel/openvswitch\_no\_kmod.spec

1. Install the RPM:

$ yum localinstall ~/rpmbuild/RPMS/x86\_64/openvswitch-2.7.0-1.x86\_64.rpm

1. Start Open vSwitch:

$ systemctl start openvswitch.service

1. Enable it to start on boot:

$ chkconfig openvswitch on

1. Check the version to make sure it’s installed:

$ ovs-vsctl -V

1. Now that we have OVS installed and started, we can install the Open Virtual Networking Kubernetes plugin from their GitHub:

$ git clone https://github.com/openvswitch/ovn-kubernetes <$DIRECTORY>

1. Change directories into the newly created directory:

$ cd /$PATH/$TO/<$DIRECTORY>

1. Install the plugin:

$ pip install .