Table of Contents

* [OpenShift Certificate Authorities Redeployment Lab](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_openshift_certificate_authorities_redeployment_lab)
* [1. Configure OpenShift to Use Intermediate CA](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_configure_openshift_to_use_intermediate_ca)
  + [1.1. Prepare Environment](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_prepare_environment)
  + [1.2. Examine Existing Certificates](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_examine_existing_certificates)
  + [1.3. Deploy New Platform CA Files](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_deploy_new_platform_ca_files)
  + [1.4. Verify Intermediate Platform CA Certificates in Use](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_verify_intermediate_platform_ca_certificates_in_use)
* [2. Explore OpenShift CA Certificates](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_explore_openshift_ca_certificates)
  + [2.1. Explore CA Certificates in Master Configuration](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_explore_ca_certificates_in_master_configuration)
  + [2.2. Explore CA Certificates in Node Configurations](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_explore_ca_certificates_in_node_configurations)
* [3. Summary](https://www.opentlc.com/labs/ocp_operations/02_01_Redeploying_OpenShift_Certificate_Authorities_Lab.html#_summary)

OpenShift Certificate Authorities Redeployment Lab

**Use Case Scenario**

MitziCom has its own Public Key Infrastructure (PKI) to provide keys and certificates to secure communications. The PKI administrators keep their Root CA certificate secret, and have used it to sign an Intermediate CA certificate which is dedicated to the OpenShift project. As the OpenShift administrator, you use the **openshift-ansible** deployment and management system to configure OpenShift to use this Intermediate key and certificate to secure OpenShift’s communications. The **openshift-ansible**system uses the Intermediate certificate to generate all of the certificates necessary to run OpenShift. The **openshift-ansible**system can also be used to update certificates in a running OpenShift system.

1. Configure OpenShift to Use Intermediate CA

1.1. Prepare Environment

1. Log in to your OpenShift Bastion host and switch to the **root** user.

sudo -i

* + If this was a high availability environment, the OpenShift Master API service would be accessed through the load balancer.

1. Find the proper hostname and port of the Master API from the **openshift\_master\_cluster\_hostname** variable in the Ansible inventory file, and put it in a shell variable for ease of use:

MASTER\_API\_HOSTPORT=$(sed -n 's/openshift\_master\_cluster\_hostname=//p' /etc/ansible/hosts):443; echo $MASTER\_API\_HOSTPORT

1. Do the same for the router on the infrastructure node—the port number is **443**:

ROUTER=$(perl -ne 's/^(infranode1\..\*?\.internal).\*$/$1/ and print and last' /etc/ansible/hosts):443; echo ${ROUTER}

1. Set an environment variable with your environment’s GUID.

export GUID=$(hostname | cut -f2 -d.); echo $GUID

We’ve created some simple scripts to create the necessary certificates and keys for your environment. Use them throughout the GPTE DevOps certificate labs.

1. Generate the new Intermediate CA keys and special server keys on your Bastion host (if they do not already exist.)
2. cd /root/
3. git clone https://github.com/newgoliath/certs/
4. cd certs

./ca\_create.sh

1. If that is successful, execute the following to create server keys.

./create\_env\_certs.sh -g ${GUID}

1.2. Examine Existing Certificates

1. Examine the OpenShift Platform CA certificates on the Master server with Ansible and the OpenSSL tools:

ansible masters -m shell -a 'openssl x509 -text -in /etc/origin/master/ca.crt | grep Subject:'

**Sample Output**

master1.GUID.internal | SUCCESS | rc=0 >>

Subject: CN=openshift-signer@1504795784

* + This CA certificate is known as the *OpenShift Platform CA* because it is used to sign most of the certificates critical to the OpenShift system.

|  |  |
| --- | --- |
|  | The **Subject** field of the certificate is **/CN=openshift-signer@** followed by a UNIX epoch timestamp. That is the CA certificate created by default by the **openshift-ansible** deployer with the OpenShift **oc adm ca**commands when no other configuration is found. It is considered a *self-signed* certificate, because there is no other CA certificate that has signed it. In this lab, you replace the Platform CA’s self-signed certificate with the intermediate certificate, so that a common, externally generated CA can validate it and the certificates it signs. |

1. Examine the existing server certificates by getting the issuer name via a client connecting to the OpenShift API:

echo QUIT | openssl s\_client -connect ${MASTER\_API\_HOSTPORT} 2>&1 | more

* + The **Subject** field of the **ca.crt** certificate in the section above matches the certificates in the **Certificate chain** section below:

**Sample Output**

Thu Sep 7 11:11:49 2017

depth=1 CN = openshift-signer@1504795784

verify error:num=19:self signed certificate in certificate chain

verify return:0

CONNECTED(00000003)

---

Certificate chain

0 s:/CN=172.30.0.1

i:/CN=openshift-signer@1504795784

1 s:/CN=openshift-signer@1504795784

i:/CN=openshift-signer@1504795784

---

Server certificate

-----BEGIN CERTIFICATE-----

<OMITTED>

-----END CERTIFICATE-----

subject=/CN=172.30.0.1

issuer=/CN=openshift-signer@1504795784

---

Acceptable client certificate CA names

/CN=openshift-signer@1504795784

Server Temp Key: ECDH, prime256v1, 256 bits

|  |  |
| --- | --- |
|  | There is only one acceptable client certificate. This changes when you add the new CA certificate and key in a subsequent step. |

1. Examine the router’s certificates:

echo QUIT | openssl s\_client -connect ${ROUTER} 2>&1 | more

* + The router has a different CA certificate and does not express any client CA names:

**Sample Output**

Certificate chain

0 s:/CN=router.default.svc

i:/CN=openshift-service-serving-signer@1504795784

1 s:/CN=openshift-service-serving-signer@1504795784

i:/CN=openshift-service-serving-signer@1504795784

---

Server certificate

-----BEGIN CERTIFICATE-----

<OMITTED>

-----END CERTIFICATE-----

subject=/CN=router.default.svc

issuer=/CN=openshift-service-serving-signer@1504795784

---

No client certificate CA names sent

Server Temp Key: ECDH, prime256v1, 256 bits

1.3. Deploy New Platform CA Files

1. To include the new CA certificate and key, add the following lines to the **/etc/ansible/hosts** Ansible inventory file. Make sure to add the new lines in the **[OSEv3:vars]** section:
2. openshift\_master\_ca\_certificate={'certfile': '/root/certs/ca/intermediate/certs/intermediate.cert.pem', 'keyfile': '/root/certs/ca/intermediate/private/intermediate.key.pem'}
3. *#*
4. *# NOTE: CA certificate will not be replaced with existing clusters.*
5. *# This option may only be specified when creating a new cluster or*
6. *# when redeploying cluster certificates with the redeploy-certificates*

*# playbook.*

1. Run the **redeploy-openshift-ca.yml** Ansible Playbook to apply the new Intermediate CA certificate and key to the system and regenerate the necessary dependent certificates and keys:

time ansible-playbook -i /etc/ansible/hosts -f 20 /usr/share/ansible/openshift-ansible/playbooks/byo/openshift-cluster/redeploy-openshift-ca.yml

* + The update process takes about four minutes.

1.4. Verify Intermediate Platform CA Certificates in Use

When the **ansible-playbook** run is complete, you verify that the certificate and key were correctly distributed to your masters and then validate their contents.

1. Check that the proper files are in place by running Ansible to view the **Subject** fields of the Platform CAs:

ansible masters -m shell -a 'openssl x509 -text -in /etc/origin/master/ca.crt | grep Subject:'

**Sample Output**

master1.GUID.internal | SUCCESS | rc=0 >>

Subject: C=US, ST=North Carolina, O=Red Hat, Inc., OU=GPTE DevOps, CN=Red Hat OpenTLC Classroom Intermediate CA/emailAddress=gpte-devops-automation@redhat.com

1. Examine the new certificates of the OpenShift Master API:

echo QUIT | openssl s\_client -connect ${MASTER\_API\_HOSTPORT} 2>&1 | more

* + Note in this sample output fragment that neither the certificate chain nor the *issuer* of the server certificate changed:

**Sample Output**

Certificate chain

0 s:/CN=172.30.0.1

i:/CN=openshift-signer@1504795784

1 s:/CN=openshift-signer@1504795784

i:/CN=openshift-signer@1504795784

---

---

Server certificate

-----BEGIN CERTIFICATE-----

<OMITTED>

-----END CERTIFICATE-----

subject=/CN=172.30.0.1

issuer=/CN=openshift-signer@1504795784

---

* + Also note that the acceptable client certificates *have* changed. The output should contain certificates similar to the following:

**Sample Output**

Acceptable client certificate CA names

/C=US/ST=North Carolina/O=Red Hat, Inc./OU=GPTE DevOps/CN=Red Hat OpenTLC Classroom Intermediate CA/emailAddress=gpte-devops-automation@redhat.com

/CN=openshift-signer@1504795784

Server Temp Key: ECDH, prime256v1, 256 bits

* + Indicated here are the original **openshift-signer** certificate that was created upon OpenShift installation and the **/C=US/ST=North Carolina/O=Red Hat, Inc.** key that you added.

1. Examine the router certificates and note that neither the issuer nor the acceptable CAs changed for the router hosts:

echo QUIT | openssl s\_client -connect ${ROUTER} 2>&1 | more

**Expected Output**

*---*

Certificate chain

0 s:/CN=router.default.svc

i:/CN=openshift-service-serving-signer@1504795784

1 s:/CN=openshift-service-serving-signer@1504795784

i:/CN=openshift-service-serving-signer@1504795784

*---*

Server certificate

*-----BEGIN CERTIFICATE-----*

<OMITTED>

*-----END CERTIFICATE-----*

subject=/CN=router.default.svc

issuer=/CN=openshift-service-serving-signer@1504795784

*---*

No client certificate CA names sent

Server Temp Key: ECDH, prime256v1, 256 bits

2. Explore OpenShift CA Certificates

In this section, you validate the various certificates generated by the **openshift-ansible** installer and the OpenShift system. The primary goal of this section to familiarize you with the various security contexts implied by the certificate and key generation. In this section, you gain greater proficiency with Ansible and OpenSSL, and explore debugging certificates and TLS in the OpenShift environment.

2.1. Explore CA Certificates in Master Configuration

In this section, you explore which certificates and keys are created when you update the CA certificate with Ansible. You write a short script to examine all of the certificates in the master configuration directory to determine their **subject** and **issuer**.

1. Examine all of the certificates in the master configuration directory:
2. ansible masters -m shell -a 'for x in $(ls /etc/origin/master/\*.crt); do \

echo $x; openssl x509 -subject -issuer -in ${x} -noout; echo; done' | less

1. Examine the output:
   * Expect to see that *most*, but not all, of the certificates on the Master configuration are signed by the issuer of the certificate that you installed in the previous section:

**Sample Output**

"issuer= /C=US/ST=North Carolina/O=Red Hat, Inc./OU=GPTE DevOps/CN=Red Hat OpenTLC Classroom Intermediate CA/emailAddress=gpte-devops-automation@redhat.com"

* + Expect to see see some certificates that are *not* signed this way and appear as follows:

**Sample Output**

/etc/origin/master/master.etcd-ca.crt

subject= /CN=etcd-signer@1503514452

issuer= /CN=etcd-signer@1503514452

* + These certificates are associated with the system’s **etcd** and service **-signer** components.

|  |  |
| --- | --- |
|  | * There are different security contexts of those certificates signed by the CA provided, versus the certificates that are signed by a different CA. * The **.kube** directory was modified on your master hosts. The Ansible Playbook updated the certificates necessary for the **oc** command to function. |

2.2. Explore CA Certificates in Node Configurations

The nodes of the OpenShift system must also have access to certificates. In this section, you examine the difference between these certificates and those on the master hosts.

1. Examine all of the certificates in the node configuration directory:
2. ansible nodes -m shell -a 'for x in $(ls /etc/origin/node/\*.crt); do \

echo $x; openssl x509 -subject -issuer -in ${x} -noout; echo;done' | less

|  |  |
| --- | --- |
|  | * + Consider why all of the certificates associated with the **node** processes are signed by the CA certificate you introduced.   + Consider where the certificates for the router hosts are located. |

3. Summary

Redeploying the OpenShift Platform CA affects communication between only the master hosts, nodes, and Docker. It allows **etcd**, **router**, and **registry** to continue unchanged.

This strategy is especially useful because:

* It enhances the separation of concerns between platform administration and the rest of the components, protecting from privilege escalation.
* It allows the administrator to plan with the various teams separately for the very minor outages for the certificate replacement to occur in an orderly fashion.

Build Version: be32934b41da1e34c27e21e79bdf57d30c732a3e : Last updated 2018-01-19 05:59:04 EST