
ippon POSITIVE TECHNOLOGY
DEPLOYING AN OPENSHIFT APPLICATION IN THE AWS GOVCLOUD

OVERVIEW

PART 01
CONTEXT

PART 02
SOLUTION

PART 03
FEDRAMP
HIGH & FIPS

FROM DEV TO PROD IN A COMPLETELY NEW ENVIRONMENT

Objective: Move a Java web application with microservices to a new Prod environment

- **Existing Application in Dev:**
  - **JHipster application** React, Java, Spring Boot
  - **Heavily open-source based** PostgreSQL, Kafka, Redis, etc.
  - **OKD (open source OpenShift)**
  - **Commercial AWS cloud environment**
- **Prod Environment:**
  - **FedRAMP High** Limited list of AWS services - no Kafka for example, FIPS compliance
  - **RedHat OpenShift**
  - **AWS GovCloud** Route53 is not available and only private hosted zones are available

ADDITIONAL REQUIREMENTS

- **Limit outbound connections from GovCloud** Minus some exceptions
  - **Nexus in GovCloud connects to Dev Nexus**
  - **Code repo must reside on GovCloud** CI/CD pipeline runs in GovCloud

COLLABORATION

DEV PLATFORM AND GOVCLOUD PLATFORM

GITLAB SETUP

TIP #1: Gitlab mirroring keeps *all* branches in sync from either direction. Make sure that is what you want!

“OFFLINE” ROOT CERTIFICATE AUTHORITY

Comply with security requirement to have the root certificate authority in a completely isolated and offline environment.

— **Needed a completely separate account from the application.**

— **More user management.** RBAC policies needed to be applied to both accounts

— **With stricter access requirements, engineers were not given access.** Training was required to launch an EC2 instance, connect via AWS Console, create a root CA, sign certificates from the application account correctly, change the CA certificate path length variable, and then package the root certificate in an encrypted WORM S3 bucket.

— **Management of the Root CA must be done manually to update the CRL or revoke certificates.**

— **Any changes to this component have impacts downstream.**

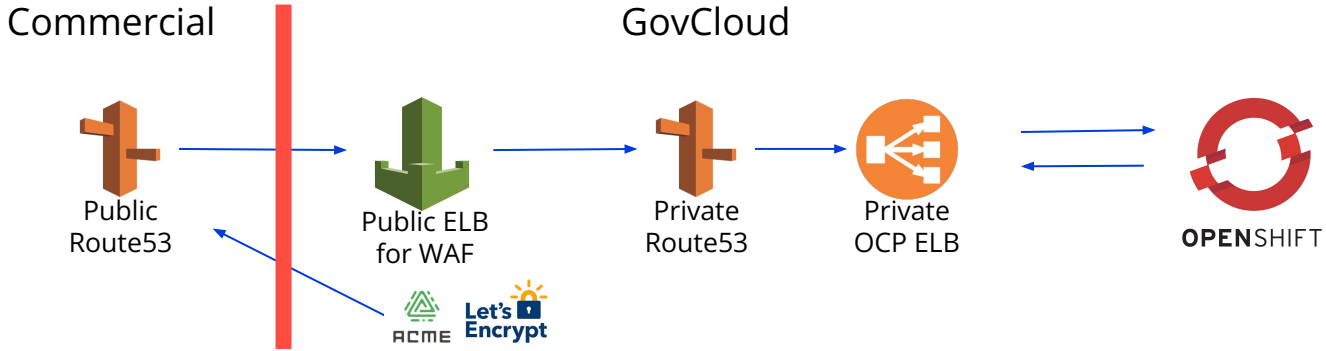
TIP #2: Create your root certificate account early so no delays are caused on the infrastructure setup due to missing certificates.

ROUTE53 IN GOVCLOUD

Public hosted zones are not allowed in the AWS GovCloud but are needed to do DNS validation for our ACME provider (i.e. Let's Encrypt).

— **A second commercial AWS account was required.**

— **More user management.** RBAC policies required



TIP #3: Create your public Route53 account early so no delays are caused on the infrastructure setup due to lack of DNS validation.

OPENSHIFT 4.X COREOS

RedHat strongly recommends running CoreOS as the "operating system" for OpenShift nodes.

— **CoreOS is NOT a fully functional operating system.** Many of the basic Linux libraries and commands do not work on CoreOS. Many of the typical agents (for security, logging, etc.) that can be installed on Linux systems cannot be installed on CoreOS.

— **For security purposes, we disabled root access.**

— **Operators must be installed to handle compliance/security tasks, log forwarding, etc.** Custom operators can be built if absolutely necessary.

TIP #4: If you need agents installed on your infrastructure nodes, verify operators exist with the needed functionality instead.

CUSTOM IMAGES

- **Custom images were required.** CIS hardened images exist but cost money and we were already paying for RedHat licenses.
- **Standard RedHat Enterprise Linux (RHEL) images were not hardened.**
  - **OpenSCAP** Validate and apply security changes
  - **FIPS module** More on this later
  - **AWS SSM** Our choice for terminal access
  - **SIEM vendor tools** TrendMicro, File/AuditBeats, Nessus

TIP #5: Use a tool like Packer to build custom images for auditability.

FEDRAMP HIGH

Several AWS services are not certified for FedRAMP High. For example, AWS Managed Kafka is not compliant but Kinesis is a compliant service. In our case, we were able to replace the Kafka functionality with Redis.

—

Partnering with a firm that specializes in FedRAMP compliance can save a lot of time. Some firms provide an environment that is ready for an Authority to Operate (ATO) assessment.

—

The security requirements are no joke!

TIP #6: Check to make sure the infrastructure and application dependencies you need are FedRAMP High certified if that is the security level required.

FIPS COMPLIANCE

FIPS COMPLIANCE

Federal Information Processing Standards (FIPS) is a mandatory standard for the protection of sensitive or valuable data within Federal systems.

One of the key challenges with FIPS is using compliant cryptographic modules throughout the application and infrastructure. This is particularly important for any encrypted communication between components.

— **Tools like GitLab are not fully compliant.** We had to get an exception to use GitLab. Since GitLab itself does not run the application, FIPS compliance was deemed less critical.

— **Many open source tools are not FIPS compliant.** For example, many tools use MD5 for hashing which is not FIPS compliant. Even very widely used open source products such as PostgreSQL are not compliant. We needed another exception to use PostgreSQL.

— **The AWS Elastic Load Balancer (ELB) SSL termination is not FIPS compliant.** There is a “beta” version that is compliant but has been in beta for several years. This can be used with a special request to AWS.

> **TIP #7: Verify *all* your components are FIPS compliant and start asking for exceptions early if tools you need are not compliant.**

TURN ON THE FIPS

Software that is FIPS compliant frequently requires a configuration change to turn on compliance.

— **Enable FIPS on RHEL.** Install and run dracut, update the grub config file and restart the machine. There are differences between RHEL7 and RHEL8 to enable FIPS.

— **The OpenShift installer can enable FIPS compliance in the installer configuration file.**

— **Turning on FIPS in Java requires much more than turning it on for the OS/container**

TIP #8: In many cases, the best approach was to turn on FIPS compliance and see what breaks. :)

FIPS WITH JAVA

Since Java runs in a virtual machine that is intended to run consistently everywhere, it does not rely on the underlying operating system for cryptographic libraries by default.

— **The default keystore and truststore are not FIPS compliant.** The stores must be switched to NONE which tells Java to use an operating system specific store (i.e. /etc/pki/nssdb/). The store type must be changed to PKCS11.

— **The Java security file must be updated to use FIPS compliant encryption schemes.** For example, the first security provider must be a FIPS compliant security provider (i.e. SunPKCS11).

— **The keystore must be setup in the init container because it is best practice to create it as read-only for the main application.**

— **The operating system must support an external keystore.**

— **We used the ubi8 JDK instead of the standard AdoptOpenJDK for FIPS compliance.**

TIP #9: PKCS11 cannot be used on a non-FIPS enabled machine.

TIP SUMMARY

- **Gitlab mirroring keeps *all* branches in sync from either direction. Make sure that is what you want!**
- **Create your root certificate account early so no delays are caused on the infrastructure setup due to missing certificates.**
- **Create your public Route53 account early so no delays are caused on the infrastructure setup due to lack of DNS validation.**
- **If you need agents installed on your OpenShift infrastructure nodes, verify operators exist with the needed functionality instead.**
- **Use a tool like Packer to build custom images for auditability.**
- **Check to make sure the infrastructure and application dependencies you need are FedRAMP High certified if that is the security level required.**
- **Verify *all* your components are FIPS compliant and start asking for exceptions early if tools you need are not compliant.**
- **In many cases, the best approach was to turn on FIPS compliance and see what breaks. :)**
- **PKCS11 cannot be used on a non-FIPS enabled machine.**

— QUESTIONS?