WHITE PAPER

RED HAT OPENSHIFT CONTAINER PLATFORM PRODUCT APPLICABILITY GUIDE FOR HIPAA SECURITY RULE

APPLICABILITY TO ASSIST CUSTOMERS IN HIPAA DEPLOYMENTS

JASON MACALLISTER
TERILYN FLOYD-CARNEY

VERSION 1.0

COALFIRE

North America | Europe
877.224.8077 | info@coalfire.com | Coalfire.com

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Red Hat, Inc. (“Red Hat”) delivers a comprehensive portfolio of products and services built from open-source software components using an affordable, predictable subscription and support model. Red Hat engaged Coalfire, a respected cybersecurity engineering, advisory, and assessment company, to conduct an independent technical assessment of Red Hat OpenShift Container Platform (OCP) 4 on Red Hat Enterprise Linux CoreOS (RHEL CoreOS). The purpose of this product applicability guide is to identify how OCP on RHEL CoreOS security capabilities, functions, and features align with Health Insurance Portability and Accountability Act of 1996 (HIPAA) Security Rule requirements.

OCP is a container platform that natively integrates open-source Linux container technologies, Kubernetes, and RHEL CoreOS. As RHEL CoreOS is deployed by default as part of OCP, note that further mentions of OCP assume the inclusion of RHEL CoreOS. OCP provides an application programming interface (API), web console, and command line interface (CLI) to manage the underlying container technologies and Kubernetes to allow users to orchestrate the creation and management of containers. OCP provides self-service build and deployment automation for containers in addition to operational container features including scaling, monitoring, and management capabilities. RHEL CoreOS is specifically designed as a pre-hardened operating system for running containerized applications from OCP. OCP and RHEL CoreOS include many built-in security capabilities for the protection of the platform and workloads running on the platform.

This product applicability guide may be useful for covered entities and business associates (CE&BAs) desiring to utilize container technologies within the framework of a HIPAA Security Rule program of compliance. The guide discusses the relevant OCP security capabilities that are applicable to supporting or addressing risks associated with containerization as well as the specifics of HIPAA Security Rule requirements. The focus of this paper is on the technical security features and capabilities of OpenShift as they align with HIPAA Security Rule requirements.

COALFIRE OPINION

Coalfire found that security controls, features, and functionalities that are built into OCP can support or address relevant HIPAA technical security standards of the Security Rule. Furthermore, OCP provides many advantages for the security of its orchestrated and hosted workloads.

INTRODUCING HIPAA SECURITY RULE

The HIPAA Security Rule specifically focuses on the safeguarding of electronic Protected Health Information (ePHI) through the implementation of administrative, physical, and technical safeguards. Compliance is mandated to all organizations defined by HIPAA as a CE&BA. These organizations are required to:

- Ensure the confidentiality, integrity, and availability of all ePHI that it creates, receives, maintains, or transmits.
- Protect against any reasonably anticipated threats or hazards to the security or integrity of such information.
- Protect against reasonably anticipated unauthorized uses or disclosures of protected health information.
- Ensure compliance by its workforce.

The requirements of the HIPAA Security Rule are organized according to safeguards, standards, and implementation specifications, the major sections include:

- Administrative safeguards
- Physical safeguards
- Technical safeguards
- Organizational requirements
- Policies and procedures
- Documentation requirements

While the administrative, physical, and technical requirements identified under HIPAA are mandatory, their implementation may differ based on the type of requirement. Under the HIPAA Security Rule, Standards and Implementation Specifications are classified as either required or addressable. It is important to note that neither of these classifications should be interpreted as optional. An explanation of each is provided below:

- **Required** – Implementation specifications identified as required must be fully implemented by the covered organization. Furthermore, all HIPAA Security Rule requirements identified as standards are classified as required.
- **Addressable** – The concept of an addressable implementation specification was developed to provide covered organizations flexibility with respect to how the requirement could be satisfied. To meet the requirements of an addressable specification, a covered organization must: (a) implement the addressable implementation specification as defined; (b) implement one or more alternative security measures to accomplish the same purpose; or (c) not implement an addressable implementation specification or an alternative. If the organization chooses an alternative control or determines that a reasonable and appropriate alternative is not available, the organization must fully document their decision and reasoning.

KEY REQUIREMENTS RELEVANT TO CONTAINERIZED WORKLOADS

While no technology solution or platform can be HIPAA compliant in and of themselves, it is important to understand security implications for the architecture, design, deployment, orchestration, and operation of containerized workloads that are involved with creating, processing, transmitting, or storing ePHI. Moreover, the HIPAA Security Rule does not specifically address the nuances of security for container orchestration platforms or their workloads; therefore, it can be useful to translate the security capabilities of new technologies to help reduce or address risk and facilitate a secure deployment and operation of workloads that are involved with regulated data.

ADDITIONAL USEFUL PUBLICATIONS

The National Institute of Standards and Technology (NIST) has also published Special Publication (SP) 800-190, *Application Container Security Guide*. The purpose of this document is to explain the security concerns associated with container technologies and make practical recommendations for addressing those concerns when planning for, implementing, and maintaining containers and container platforms. This publication is a guide that provides recommendations to help ensure the security of container technology implementations and usage. While NIST SP 800-190 is not specific to HIPAA, it can be useful guidance as it pertains to addressing risk associated with container platforms and containerized workloads.

Another NIST guide specific to HIPAA is SP 800-66 Rev. 1. This publication maps NIST SP 800-53 Rev. 4 requirements to HIPAA Security Rule safeguards and requirements and ties the HIPAA Security Rule to a framework for managing risk.

INTRODUCING OCP

OCP is a comprehensive, enterprise-grade application platform built for containers with Kubernetes. It is an integrated platform that can be used to run, orchestrate, monitor, and scale containers. OCP allows organizations to control, defend, and extend the application platform throughout an application's lifecycle. It enables a secure software supply chain to help make applications more secure without reducing developer productivity and provide a consistent operations and management experience across various infrastructures in support of multiple teams.

OCP ARCHITECTURE

The following is a list of components and roles that support OCP.

Operating System (OS) – OCP is packaged and deployed on RHEL CoreOS. Customers can create compute (i.e., worker) machines that use either RHEL CoreOS or Red Hat Enterprise Linux (RHEL) as their operating system (OS).

RHEL CoreOS is a single-purpose, container- and Kubernetes-optimized, minimal footprint OS technology powered by the same binaries as RHEL. This pre-hardened OS can assist organizations with meeting requirements for system hardening with least functionality through its lightweight, purpose-built nature, as it only includes the necessary features, functions, and services to host containers in an OCP environment. RHEL CoreOS is supported only as a component of OCP 4 for all OCP machines. including control plane (i.e., master) machines.

RHEL CoreOS is designed to be more tightly managed than a default RHEL installation. Management is performed remotely from the OCP cluster. When the OCP cluster is set up and RHEL CoreOS is deployed, customers can only modify a few system settings.

RHEL CoreOS incorporates the container runtime interface - open container initiative (CRI-O) container engine instead of the Docker container engine as the runtime interface for Kubernetes. This provides several benefits for the OCP. The CRI-O engine focuses on features needed by Kubernetes platforms, such as OCP, and offers specific compatibility with different Kubernetes versions. CRI-O also offers a smaller footprint and reduced attack surface than is possible with container engines that offer a larger feature set.

As part of the OCP offering, Red Hat CoreOS provides the capability to support transactional upgrades using the rpm -ostree upgrade system. Updates are delivered via container images and are part of the OpenShift update process. When deployed, the container image is pulled, extracted, and written to disk, and the bootloader is modified to boot into the new version. The machines in the cluster will reboot into the update in a rolling manner to ensure that cluster capacity is minimally impacted. RHEL CoreOS upgrades in OCP are performed during cluster updates.

RHEL and RHEL CoreOS have built-in security features and functionality that, as configured in an OCP installation, provide a secure platform for supporting compute machines that host workloads in containers that OCP orchestrates. While the option exits to utilize RHEL for worker nodes managed by the OCP, there are security benefits for using RHEL CoreOS, including reduced attack surface, pre-hardened OS, and automated updating among others. Customers should consider their own use cases when selecting their OCP architecture.

Operating Environment – OCP can be deployed on bare-metal physical hardware, on virtual infrastructure, or in the cloud. It can be deployed on private or certified public cloud environments, depending on the organization's specific use cases.

OCI Runtime – CRI-O is the only available container engine within OCP clusters.

Kubernetes – Kubernetes provides orchestration for complex multi-container services. Kubernetes also provides scheduling for services across a container host cluster. OCP adds developer- and operations-centric tools to Kubernetes that help enable rapid application development, easy deployment and scaling, and long-term lifecycle maintenance for applications and the platform itself. OCP also leverages integrated components from Kubernetes to automate application builds, deployments, scaling, health management, and more. Included in the automation capabilities of OCP is the ability to configure and deploy Kubernetes container host clusters.

Containers – End-user application instances, application components, or other services are run in Linux containers. The container only includes the necessary libraries, functions, elements, and code required to run the application.

Pods – While application components run in containers, OCP orchestrates and manages pods. A Kubernetes pod is a group of containers that are deployed together on the same host. A pod is an orchestrated unit in OCP made up of one or more containers. Generally, a pod should only contain a single function, such as an application server or web server, and should not include multiple functions. such as a database and application server.

OCP Components

The components of OCP are described below.

OpenShift Operators – An Operator is a method of packaging, deploying, and managing a Kubernetes-native application. Operators automate the lifecycle management of containerized applications within Kubernetes. A Kubernetes-native application is an application that is both deployed on Kubernetes and managed using the Kubernetes APIs and kubectl tooling. A controller is a core concept of Kubernetes and is implemented as a software loop that runs continuously on the Kubernetes master nodes, compares, and, if necessary, reconciles the expressed desired state and the current state of an object. Objects are well-known resources like Pods, Services, ConfigMaps, or PersistentVolumes. Operators apply the model of controller at the level of entire applications and are, in effect, application-specific custom controllers.

Since OpenShift itself is a fully containerized platform consisting of many different components, OCP takes advantage of Operators for driving the installation and upgrades of OpenShift and all its services. Operators are both the fundamental unit of the OCP 4 code base and a convenient way to deploy applications and software components for the OCP 4 customer's applications to use, including the Kubernetes core services along with monitoring (Prometheus, Grafana), logging (Elasticsearch, Fluentd, Kibana), software-defined networking, storage, registry, and other components that make up the OpenShift Kubernetes platform. Operators serve as the platform foundation that removes the need for manual upgrades of the OSs and OCP control plane applications. The Cluster Version Operator and Machine Config Operator allow simplified cluster-wide management of those critical components. Everything that makes up the platform is managed throughout its lifecycle with Operators.

Operator Lifecycle Manager – The Operator Hub provide facilities for storing and distributing Operators to people developing and deploying applications. The Operator Lifecycle Manager (OLM) is the backplane that facilitates management of Operators on a Kubernetes cluster. With OLM, administrators of the cluster can control which Operators are available in what namespaces and who can interact with the running Operators. The permissions of an Operator are configured automatically to follow a least-privilege approach. The OLM helps users install, update, and manage the lifecycle of all Operators and their associated services running across their clusters. The OLM runs by default in OCP 4, which aids administrators in installing, upgrading, and granting access to Operators running on their cluster.

OpenShift Nodes – Nodes are where end-user applications are ultimately run in containers. OpenShift nodes are instances of RHEL CoreOS or RHEL hosts with the OCP software deployed. Nodes will contain the necessary OpenShift node daemon, the Kubelet, the container runtime, and other necessary services to support the hosting of containers. Most of the software components that run above the OS (e.g., the software-defined network daemon) all run in containers themselves on the Node.

OpenShift Master – The OCP Masters are the control plane for OCP. The Master machines maintain and understand the state of the environment and orchestrate all activity that occurs on the Nodes. The Masters are run on RHEL CoreOS. Master machines are defined by a series of standalone machine API resources. Extra controls apply to Master machines to prevent customers from deleting all Master machines and breaking the cluster. Cluster Masters contain more than Kubernetes services for managing the OCP cluster; services that fall under the Kubernetes category on the Master include the API server, etcd, scheduler, controller manager server, and ingress controllers. Some of the services on the Master machines run as system services, while others run as static Pods. System services that are run on Master machines include sshd, CRI-O container engine, and Kubelet. The following are the four functions of the OpenShift Masters:

API and Authentication – The Master machines provide the single API that all tooling and systems interact with. Everything that interacts with OCP must go through this API. All API requests are Secure Sockets Layer (SSL) encrypted and must be authenticated. Authentication is handled by a built-in OAuth server. Red Hat recommends tying the Masters to an external identity and access management system using Lightweight Directory Access Protocol (LDAP), OpenID Connect, or other identity providers. Authorizations are handled by fine-grained role-based access control (RBAC). The Master evaluates requests for both authentication (AuthN) and authorization (AuthZ). Users of OCP who have been granted access can be authorized to work with specific projects.

Desired and Current State – The state of OCP is held in the OpenShift data store. The data store uses etcd, a distributed key-value store. The data store houses information about the OCP environment and pertaining to the OpenShift Master, including user account information and the RBAC rules; the OCP environment state, including application environment information and non-application user data; and important environment variables, secrets data, and other information.

Scheduler – The scheduler determines pod placement within the OCP. It uses a combination of configuration and environment state (e.g., CPU, memory, and other environmental factors) to determine the best fit for running pods across the Nodes in the environment. The scheduler is configured with a simple JSON file in combination with Node labels to carve up the OCP. This allows placement of pods within OCP to be based on the real-world topology, making use of concepts such as regions, zones, or other constructs relevant to the enterprise. These factors can contribute to the scheduled placement of pods in the environment and can ensure that pods run on appropriate Nodes associated with their function.

Health and Scaling – The OpenShift Master is also responsible for monitoring the health of pods and scaling the pods as desired to handle additional load. The OpenShift Master executes liveness and readiness tests using probes that are defined by users. The OpenShift Master can detect failed pods and remediate failures as they occur.

Service Layer – The OpenShift Service Layer allows for application components to easily communicate with one another. For instance, a front-end web service containing multiple web servers would connect to database instances by communication via the database service. OCP automatically and transparently handles load balancing across the services' instances. In conjunction with probes, the OpenShift Service Layer ensures that traffic is only directed toward healthy pods, which helps to maintain component availability.

Persistent Storage – Linux containers are natively ephemeral and only maintain data for as long as they are running. Applications or application components may require access to a long-term persistent storage repository, such as database engines. OCP provides the means to connect pods to external real-world storage, which allows for stateful applications to be used on the platform. Available persistent storage types include iSCSI, Fiber Channel, and NFS, as well as cloud-type object storage and software-defined storage options such as Red Hat OpenShift Container Storage. Persistent storage can be dynamically provisioned upon the user's request, provided the storage solution has an integration with OCP.

Ingress Controller – The Ingress Controller is commonly used to allow external access to an OCP cluster. The Ingress Operator manages Ingress Controllers. An Ingress Controller is configured to accept external requests and proxy them based on the configured routes. External requests are limited to HTTP and HTTPS using Server Name Indication (SNI) and Transport Layer Security (TLS) using SNI, which is sufficient for web applications and services that work over TLS with SNI. Ingress works in partnership with the service layer to provide automated load balancing to pods for external clients. The Ingress Controller uses the service endpoint information to determine where to route and load balance traffic; however, it does not route traffic through the Service Layer.

Cluster Network Operator – The Cluster Network operator deploys and manages the cluster network components, including the Container Network Interface (CNI) Software Defined Networking (SDN) plugin selected for the cluster during installation.

OpenShift SDN – The OpenShift software-defined network (SDN) is a unified cluster network that enables communication between pods across the OCP cluster. The OpenShift SDN configures an overlay network that uses Open vSwitch (OVS). Red Hat currently provides one SDN mode for the OCP pod network: the network-policy mode. The network policy mode provides fine-grained access control via user-defined rules. Network policy rules can be built in a "mandatory access control" style, where all traffic is denied by default unless a rule explicitly exists, even for pods or containers on the same host. The network policy mode is the default mode.

OpenShift Registry – The OpenShift Registry provides integrated storage and management for sharing container images, but OCP can utilize existing OCI-compliant container registries that are accessible to the Nodes and the OpenShift Master via the network.

Figure 1 below provides a high-level illustration of the OCP components:

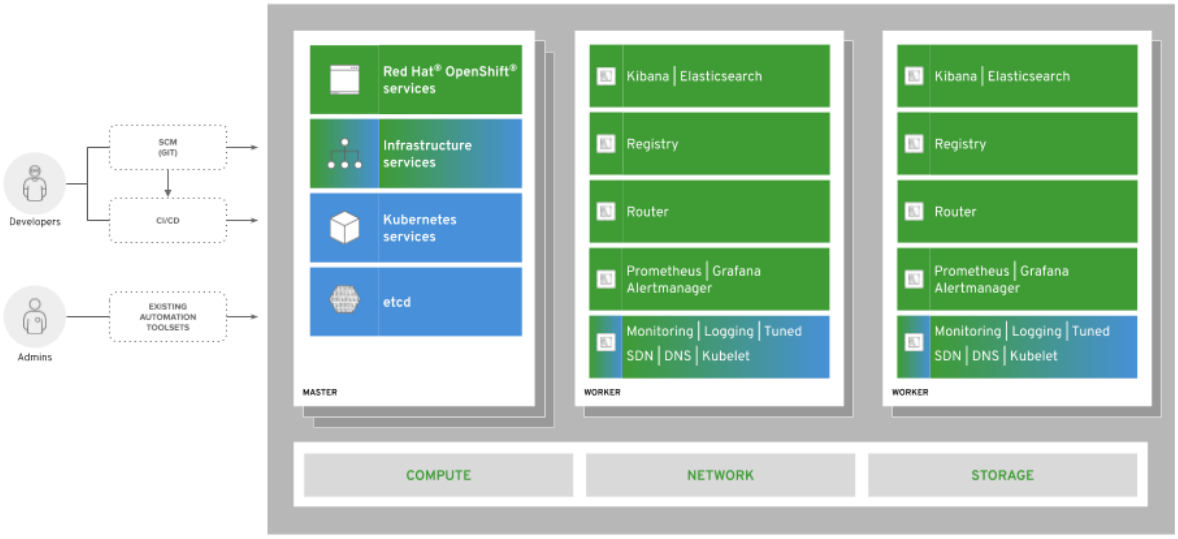

Figure 1: High-Level OpenShift Architecture

Users – User (e.g., operators, developers, application administrators) access to OCP is provided through standard interfaces including the web user interface (UI), CLI, and IDEs. These interfaces go through the authenticated and RBAC-controlled API. Users do not require system-level access to any of the OCP hosts or nodes, even for application debugging and troubleshooting tasks.

There are three types of users that can exist in an OCP environment: regular users, system users, and service accounts.

> Regular users are created automatically in the system upon first logon or via the API. Most interactive OCP users, including operators, developers, and application administrators, will be represented by this type of user account.
>
> Many of the System users are created automatically when the OCP infrastructure is defined, mainly for the purpose of enabling the infrastructure to securely interact with the API. System users include a cluster administrator (with access to everything in the OpenShift environment), a per-node user, users for use by ingress controllers and registries, and various others. There is also an anonymous system user that is used by default for unauthenticated requests. Note that even unauthenticated requests are managed by RBAC. Anonymous requests are assigned to the system:unauthenticated group, which is bound to cluster-scoped roles. Examples of these users are system:admin, system:openshift-registry, and system:node:node1.example.com.
>
> Service accounts are non-human system users, often associated with projects, used for API access in automation situations. Some default service accounts are created and associated when a project is first created. Project and cluster administrators can create additional service accounts for defining access to the contents of each project.

Projects – A project is a Kubernetes namespace with additional OCP annotations and metadata. It is the central vehicle by which access to resources for regular users is managed and is essentially the tenancy model of OCP. A project allows a community of users to organize and manage their content in isolation from other communities.

For more information on OCP concepts, features, and functions, please refer to Red Hat's product documentation.

OCP SECURITY

OCP enables continuous security with defense-in-depth capabilities and Red Hat's secured software supply chain. Security controls can be applied dynamically to the platform and the applications the platform supports. This allows security controls to keep up with the scale and agility of applications deployed in the platform. OpenShift runs on RHEL CoreOS and makes heavy use of the existing security features built into the OS. Red Hat manages the OS packages and provides trusted distribution of content. The security of OCP includes and utilizes hardened technologies such as SELinux; process, network, and storage separation; proactive monitoring of capacity limits (e.g., CPU, disk, memory); and encrypted communications for infrastructure support including SSH and SSL. Additionally, OCP provides integration with third-party identity management solutions to support secure authentication and authorization options in alignment with organization compliance requirements.

The following is a high-level list of OCP security features and capabilities:

Container Host and Platform Multitenancy – RHEL can manage multitenancy for the container runtime by using Linux namespaces, SELinux, CGroups, and Secure Computing Mode (seccomp) to isolate and protect containers. This can be useful for maintaining separation for workloads of different classifications.

Container Content – The Red Hat Container Catalog delivers validated application content from Red Hat and certified independent software vendor (ISV) partners.

Container Registries – OCP includes an integrated container registry that provides basic functionality supporting build and deployment automation within the cluster, tied into the OpenShift RBAC. Within the context of an organization needing to adhere to HIPAA Security Rule technical safeguard requirements, Red Hat Quay is an additional product that provides a registry with capabilities for both RBAC and vulnerability scanning of applications and software in images.

Building Containers – OCP integrates with Jenkins and Tekton and can be easily integrated with other continuous integration/continuous delivery (CI/CD) tools to manage builds and perform code inspection, code scanning, and validation.

Deploying Containers – OCP prevents containers from running as root or other specifically-named users by default. In addition, OCP enables granular deployment policies that allow operations, security, and compliance teams to enforce quotas, isolation, and access protections.

Container Orchestration – OCP integrates secure operational capabilities to support trust between users, applications, and security policies. OCP delivers container orchestration, automation of scheduling, and the running of application containers on clusters of physical or virtual machines through inclusion and extension of the open source Kubernetes project. It can be used to guide how and where containers can be deployed, deployment of containers based on availability of capacity, how containers can access each other based on least privilege, how access to shared resources and management is controlled, how container health is monitored, how the applications can be scaled automatically to meet need, and can enable developer self-service while maintaining and meeting designed security requirements.

Network Isolation – OCP uses an SDN approach to provide a unified cluster network that enables communication between pods across the OCP cluster. OCP uses the Multus CNI plugin to allow chaining

of CNI plugins. This Pod network is established and maintained by the OpenShift SDN, which configures an overlay network using Open vSwitch (OVS). There is a NetworkPolicy SDN mode available from Red Hat for the customer to configure network policies. Other third-party SDN solutions exist that are capable of being integrated into OpenShift as well.

Multus is a Multi CNI plugin to support the Multi Networking feature in Kubernetes using customer resource definition (CRD) based network objects in Kubernetes. During cluster installation, the customer configures the default Pod network. The default network handles all ordinary network traffic for the cluster. The customer can then define additional networks based on the available CNI plugins and attach one or more of these networks to their Pods. The customer can define more than one additional network for their cluster, depending on their needs. This gives the customer flexibility when configuring Pods that deliver network functionality, such as switching or routing.

The capability to create additional networks enables the customer to enhance network isolation, including data plane and control plane separation. The customer can send sensitive traffic onto a network plane that is managed specifically for security considerations and can separate private data that must not be shared between tenants or customers.

Secure the data – OCP provides access to and integration with a broad range of storage platforms and protocols, allowing applications to securely store and encrypt application data. OpenShift provides the option to encrypt sensitive data stored in etcd and to encrypt the RHEL CoreOS volumes.

Service Mesh – An optional feature of OCP is Red Hat OpenShift Service Mesh. This provides a platform for behavioral insights and operational control over the networked microservices in a service mesh. A service mesh is a network of microservices that make up applications in a distributed microservice architecture and the interactions between those microservices. When a service mesh grows in size and complexity it can become harder to understand and manage. With Red Hat Service Mesh, the customer can connect, secure, and monitor microservices in the OCP environment. Red Hat OpenShift Service Mesh adds a transparent layer on existing distributed applications without requiring any changes to the service code. The customer can add Red Hat OpenShift Service Mesh support to services by deploying a special sidecar proxy to relevant services in the mesh that intercepts all network communications between microservices. The service mesh is configured and managed using the control plane features.

The Red Hat OpenShift Service Mesh gives customers a way to create a network of deployed services to provide discovery, load balancing, service-to-service authentication, failure recovery, metrics, and monitoring capabilities. More complex operational functions provided by Red Hat OpenShift Service Mesh include A/B testing, canary releases, rate limiting, access control, and end-to-end authentication.

API management – OCP and Service Mesh can be integrated with the 3scale API Management platform to authenticate, secure, and rate-limit API access to applications and services.

SCOPE AND APPROACH FOR REVIEW

The understanding of OCP, RHEL CoreOS, RHEL, and their combined capabilities was gained through product specification, installation, configuration, administration, and integration documentation provided by Red Hat and generally made available from Red Hat's public-facing web site. Coalfire has further conducted interviews and engaged in live product demonstrations with Red Hat personnel and subject matter experts. For live product demonstration purposes, OCP was also deployed in a lab environment to provide hands-on demonstration, testing, and analysis of the system's capabilities to support compliance.

Coalfire's review of OCP began with a general alignment of the applicability of the technology against the high-level HIPAA Security Rule requirements objectives. This was further narrowed down to specific

requirements that OCP capabilities and features may support. An analysis of capability for the reviewed technology to address applicable requirements was then conducted. This analysis primarily focused on what an assessor might review when following HIPAA Security Rule testing guidance during an assessment.

The review of requirements was comprehensive and includes identification of gaps for which the technology may not be sufficient to address. This does not include requirements that are addressable through other means including, but not limited to, organizational procedures or the provision of additional third-party technologies.

SCOPE OF TECHNOLOGY AND SECURITY STANDARD TO REVIEW

Coalfire was tasked by Red Hat to review OCP 4. The focus on the review was on OCP and does not include other OpenShift offerings from Red Hat including, but not limited to, Red Hat OpenShift Dedicated, Microsoft Azure Red Hat OpenShift, or Red Hat OpenShift Online. The primary focus of the review included the components, features, and functionality of OCP along with the supporting underlying OS features and functionality as deployed on RHEL CoreOS. Coalfire did not include in the assessment worker or compute application pods or containers that an organization may deploy on OCP and the security that is likely necessary to be applied directly to those workloads. Containers that were deployed in the test environment were strictly used for the purposes of demonstrating the platform's orchestration, deployment, and management capabilities. Furthermore, Coalfire did not assess available image registries or repositories that may be used for acquiring applications, services, dependencies, or other elements to be hosted on or used within OCP.

For this review, Coalfire included requirements from the Health Insurance Portability and Accountability Act of 1996 and specifically the Health Insurance Reform: Security Standards; Final Rule publication dated February 20, 2003 from the Department of Health and Human Services Office of the Secretary available from https://www.hhs.gov. For a broader understanding of the requirements and their applicability to technical solution implementation, Coalfire also reviewed supporting documentation provided by NIST and HHS, including assessment guidance, the HIPAA Security Risk Assessment Tool, Security Rule Guidance Material, and Frequently Asked Questions. Applied understanding of the HIPAA security rule requirements and guidance was supplemented by documentation and guidance on relevant subjects including NIST SP 800-66 Rev. 1 and the NIST Publications Crosswalk. Coalfire applied the official crosswalk between HIPAA Security Rule and NIST Cybersecurity Framework released by the OCR. As OpenShift is a container platform, Coalfire also applied guidance from NIST SP 800-190, *Application Container Security Guide,* September 2017 publication.

COALFIRE EVALUATION METHODOLOGY

Coalfire initially examined the HIPAA Security Rule requirements and identified them as either procedural (organizational) or technical (implementation). Qualification of a requirement as procedural or technical was based on a review of the requirement narrative, testing procedures, and guidance.

Non-technical procedural requirements that include the definition and documentation of policies, procedures, and standards were not considered directly applicable to the technical solution. Likewise, non-technical requirements including operational procedures that describe manual processes were not assessed against the technology's capability. Examples of these types of non-technical requirements generally included maintenance of facility visitor logs, verification of an individual's identity prior to granting physical or logical access, performance of periodic physical asset inventories, or generation of network topology or flow diagrams.

Technical requirements were then assessed to determine applicability of the solution or solution components to address all are part of the standard. Where achievement of the requirement objectives was more likely to be met using a technology external to OCP or was not an inherent feature or capability of OCP, the assessed technology was determined to be not applicable or capable to address the safeguard or requirement. An example of security rule standard that Coalfire determined that OCP was not able to inherently address was protection from malicious software. This would more likely be addressed by the addition of a third party, possibly partner solution, that can be applied to or integrated with the platform, such as Aqua Security, Sysdig, or Twistlock. All HIPAA Security Rule Standards are considered pertinent and applicable to any system that stores, processes, or transmits ePHI. Therefore, additional consideration may be necessary when implementing container platforms for the hosting of ePHI-related workloads and data.

Coalfire further assessed the capability or the degree for which the solution could address the security rule standard. For applicable requirements, Coalfire designated a qualitative category of capability, including whether the solution was determined to fully support the requirement, partially support the requirement, or unable to support the requirement. In cases where the requirement was determined to be applicable but unsupported, additional thought for the use of third-party solutions should be considered.

OCP AND HIPAA

The narratives that follows details HIPAA Security Rule Technical Safeguards that OCP has applicability to address or support. This applicability applies either as a customer configurable item within OCP or as a native and default capability to address or support the safeguard. The OCP customer is also referred to as the CE&BA. The findings assume that OCP is used to host workloads that contain ePHI and therefore the OCP is considered in scope for application of the CE&BA's HIPAA-compliant security program. The findings are also specific to the OCP and are not inclusive of security control implementation that will be necessary for the underlying layers of the comprehensive environment, including hardware, additional software, or additional underlying platforms and components.

OCP APPLICABILITY DETAIL

The section provides details of OCP capabilities for alignment to address or support HIPAA Security Rule Technical Safeguard requirements. The requirements that are listed are the requirements that Coalfire determined that OCP may have some inherent technical security capability to support.

Access Control

164.312(a)(1) Access Control: Implement technical policies and procedures for electronic information systems that maintain electronic protected health information to allow access only to those persons or software programs that have been granted access rights as specified in § 164.308(a)(4).

General Guidance

The CE&BA should inventory and understand the technical access control capabilities available to users and systems for the OCP and their workloads. The CE&BA should also inventory and understand the administration, user, and system access methods and requirements for the OCP as well as hosted workloads. The CE&BA should then analyze the workloads and operations to identify access needs of all users and systems both to and from the OCP and the workloads. This will be the framework for assigning and providing specific and organizationally defined access to the OCP and the workloads.

The CE&BA should understand how RBAC and Mandatory Access Controls (MAC) are defined, assigned, and enforced with OCP and RHEL CoreOS. The CE&BA should also understand how network access

controls through the OpenShift SDN are capable of being defined to enforce specific network access between workloads and their components.

The CE&BA should ensure that all system users are assigned a unique identifier. Any default, generic, or group identifiers and associated credentials for users that cannot be tied back to a specific user identity should be disabled or vaulted. All activities recorded by users of OCP should be logged with enough storage space to support log retention standards. Audit logs should be sent to a third-party log aggregation solution or security information and event management (SIEM) solution. Users should be properly trained, including platform specific security awareness training that is aligned to the role and function that the user performs. The CE&BA should establish formal policies, processes, and procedures to guide access control for the organization, including access controls relative to the OCP.

As a part of access control procedures, the CE&BA should identify responsible parties for managing, implementation, and maintaining the access control procedures. Procedures will also be necessary for performing regular review of granted access as a matter of ongoing operations. Procedures should include steps for the assignment or updating of access that include initial access, increased access, and access to additional systems and applications. The CE&BA may evaluate procedures and standards for protecting data relative to access controls to identify any vectors for subverting technical guardrails that could inadvertently allow a user to gain additional unauthorized access or escalate their access rights. Finally, the CE&BA will want to establish procedures for the termination of access that accounts for all previously granted access assigned to the terminated user.

User and System Access

OCP provides a web console, which is a user interface accessible from a web browser. The web console can be used to visualize, browse, and manage the contents of projects. OCP also provides access through an API as well as a CLI. The OCP RESTful APIs are accessible via HTTP(S) on the OCP master servers. The REST APIs can be used to manage end-user applications, the cluster, and the users of the cluster. The OpenShift CLI exposes commands for managing applications, as well as lower-level tools to interact with each component of the system.

Interaction with OCP is associated with a user. An OCP user object represents an actor that may be granted permissions in the system by adding roles to the users or their groups. The users that can exist within OCP include regular users and service accounts. OCP is configured to use RBAC, allowing for granular determination of access for users or groups of users. RBAC objects determine whether a user can perform a given action on the system, based on the user's assigned roles. This allows platform administrators to use cluster roles and bindings to control who has various access levels to OCP itself and all contained OCP projects and resources. This also allows developers to use local roles and bindings to control who has access to their projects.

The authorization process is managed using rules, roles, and bindings. Rules are a set of permitted verbs (actions) on a set of objects, for example, whether something or someone can create pods. Roles are collection of rules. Users and groups can be associated with roles or bound to multiple roles at the same time. Bindings are associations between users and/or groups with a role.

OCP provides an abstraction layer from the underlying RHEL CoreOS. After the initial boot of RHEL CoreOS, systems are managed by the Machine Config Operator (MCO) that runs in the OCP cluster. Red Hat strongly discourages logging in to a RHEL CoreOS machine directly. Limited direct access to RHEL CoreOS machines in OCP can be used for debugging purposes. RHEL CoreOS has additional security features that support access control.

Coalfire recommends placing OpenShift on the CE&BA's internal network and ensuring that there is no direct Internet access to the OCP cluster or workloads without boundary controls at the edge of the CE&BA's network (e.g., firewalls, routers, application gateways, web proxies, web gateways, intrusion detection system [IDS], intrusion prevention system [IPS]). These are provided external to OpenShift and are part of the CE&BA's overall security architecture. The information flow control to and from the Internet or uncontrolled network is thus provided by the CE&BA's third-party solution.

It is recommended that administrator or privileged access be limited through defined control points such as jump hosts or bastion hosts. This approach limits direct access for privileged access to the OCP. At this point, additional access controls can be enabled and enforced such as idle session termination, session locks, system use notifications, and session termination.

Similarly, to how RBAC resources control user access, administrators can use Security Context Constraints (SCCs) to control permissions for pods. These permissions include actions that a pod (a collection of containers) can perform and what resources it can access. SCCs can be used to define a set of conditions that a pod must run with in order to be accepted in the system. SCCs allow administrators to control the following:

- Whether a pod can run privileged containers
- The capabilities that a container can request
- The use of host directories as volumes
- The SELinux context of the container
- The container user ID
- The use of host namespaces and networking
- The allocation of an FSGroup that owns the pod's volume
- The configuration of allowable supplemental groups
- Whether a container requires the use of a read only root file system
- The usage of volume types
- The configuration of allowable seccomp profiles.

OCP creates a cluster administrator, kubeadmin, after the installation process completes. This user has the cluster-admin role automatically applied and is treated as the root user for the cluster. After defining an identity provider and creating a new cluster-admin user, Red Hat recommended to remove the kubeadmin user to improve cluster security.

Audit Controls

164.312(b) Audit Controls: Implement hardware, software, and/or procedural mechanisms that record and examine activity in information systems that contain or use electronic protected health information

Guidance

Typically, the CE&BA's containerized workloads (services or applications) have direct access to the ePHI data. This application layer can be abstracted from the underlying platform with controls in place to limit the access between each layer to only that which is necessary for the application or services designed tasks. The CE&BA should certainly build their applications and services to be inherently resilient to support security of ePHI. At the same time, it will be important for the CE&BA to understand how each layer of the infrastructure, platform, and support systems can impact the security of ePHI as well.

The CE&BA should determine the activities that will be tracked and audited to include both activities specific to the functioning of applications hosted on the OCP as well as activities associated with the management and operation of the OCP itself. The discovered and defined activities the CE&BA selects for audit may likely include those that represent the greatest risk to the confidentiality, integrity, and availability of ePHI. The organization can identify the parts of the system, applications, or processes that have the potential to introduce vulnerability to the data for tampering, inappropriate uses, or disclosures. It will be important for the CE&BA to understand the actions and the context of actions that can be performed with OCP. The CE&BA should define what data should be captured to best represent an audit trail that can support recreation or retracing of actions or steps that were carried out as part of incident investigation. The CE&BA will need to understand what the audit records include.

The CE&BA should define policies, processes, procedures, and standards for audit controls that include who is responsible for the overall audit process and results, how often audits will take place, how often audit results will be analyzed, where the audit information will reside, and what actions will be taken for audit findings that reveal employee violations. Additional elements of audit controls that are important for consideration include defining processes and mechanisms for reporting, notification, incident response and review. Finally, the CE&BA will be responsible for ongoing assessment of the effectiveness of their audit process to detect and report on security incidents with a plan to revise as necessary to continuously improve the effectiveness of their audit program.

OCP has the means to generate audit logs that can be used to track the access and actions taken by users and services within the OCP. OpenShift is capable of supporting audit requirements by generating auditable events and logs that contain information to establish what type of event occurred, when the event occurred, where the event occurred, the source of the event, the outcome or result of the event or action taken, and the identity of the service, individual, or subjects associated with the event. OCP generates audit records and events pertaining to actions taken by users or system components against the API, CLI, or Web Console interface. OpenShift also generates log data and performance metrics relative to its normal operations. The CE&BA should be responsible for determining the capabilities of the information system for auditing events that are defined by the organization as important or necessary for identifying security incidents.

OCP Support for Audit Controls

Based on the CE&BA's definition of required logs and events to be including for audit control processes, the CE&BA must ensure that OCP is properly configured to generate the necessary audit control data. OCP can be configured for cluster logging to aggregate logs for a range of OCP services. The cluster logging components are based on Elasticsearch, Fluentd, and Kibana. The Cluster Logging and Elasticseach operators are used to deploy the logging stack. The Cluster Logging operator deploys the Fluentd collector to each node in the OCP cluster. It collects all node and container logs and writes them to Elasticsearch. Kibana is the centralized web UI where users and administrators can create visualizations and dashboards with the aggregated data. The logging provided with OCP is useful for detecting and troubleshooting issues with the OCP and may be limited in functionality to support the CE&BA's security requirements. The CE&BA can configure the log collector to send logs to an external log aggregator or syslog server where the logs can be consumed by the CE&BA's SIEM. API events and host auditd logs are stored separately and can also be forwarded.

OCP is dependent upon accurate time synchronization to support sensitive operations, such as log keeping and time stamps. OCP uses the system time of the underlying host. OpenShift provides guidance for enabling NTP synchronization for the underlying hosts in support of keeping consistent time among all components of OCP. The hosts can be configured for coordinated time synchronization in support of time

stamps using Coordinated Universal Time (UTC). OCP operations include etcd leader election and health checks for pods and some other issues and helps prevent time skew or drift problems.

Coalfire recommends that events and logs generated by OCP be sent to or consumed by a third-party SIEM solution external to OpenShift. This allows the logs generated by OCP to be correlated with other systems logs which may include logs generated directly by workloads that are hosted by the OCP. It also allows the CE&BA greater control over the integrity and availability of the log data should it be needed as part of the CE&BA's incident response processes, including forensic investigation as.

Integrity

164.312(c)(1) Integrity: Implement policies and procedures to protect electronic protected health information from improper alteration or destruction.

Guidance

The CE&BA should identify the users who have been authorized to access ePHI. The CE&BA should also provide a justifiable basis for establishing access to the ePHI for each user. The users should be trained on how to use the information in accordance with their job responsibilities. Finally, there should be an audit trail that is established for all access to ePHI that helps to confirm that all access to ePHI is in accordance with the defined authorization parameters.

Similarly, the CE&BA should understand the various ways that ePHI can be accessed and identify likely scenarios or unauthorized sources (e.g., hackers, insider threats, business competitors) that can jeopardize the information's integrity through some method of interception or unauthorized access. This should include understanding the role that underlying infrastructure, platforms, and applications play as in the lifecycle of ePHI. The organization should understand what should or can be done to protect the integrity (decrease or eliminate the likelihood of the data being altered) of the information and prevent the risk of compromise of the data as it is being processed, transmitted, or stored. The processes and procedures for the protection of ePHI should be formally documented with established standards including techniques, methodologies, or tooling that can be employed to protect the integrity of ePHI. These techniques, methodologies, and tooling should be implemented according to policies and standards that are defined by the CE&BA to ensure the quality of the protected data.

Principally, the protection of ePHI to ensure the integrity of the data should be handled at the application layer. Methods employed by the application could include digital signatures attached to the data, file integrity monitoring, and checksum technologies.

The organization should continually monitor and evaluate the effectiveness of controls to reduce the risk of compromise of ePHI.

OCP for Support of Integrity

Security for the integrity of ePHI will optimally be implemented at the application layer. OCP also includes security features and functions that can be useful to protect the workloads or applications at the platform level and reduce the likelihood or opportunity for compromise to occur with the workloads. OCP provides mechanisms that can be useful for isolation of workloads, detection of vulnerabilities in workloads, support for ensuring the immutability of containers in runtime, protection for the integrity of images held in the container registry from alternation and compromise, securing of network transmissions for management plane, control plane, and data plane communications, and support for security monitoring for the platform and workloads.

Starting at the OS layer, Red Hat has combined OCP and RHEL CoreOS. RHEL CoreOS uses the security features and functionality of Red Hat Enterprise Linux and Red Hat Enterprise Linux Atomic Host. This OS

is pre-hardened and designed specifically to include only necessary components to support containerization and reduce the surface area of attack. RHEL CoreOS includes security features such as Linux namespaces, SELinux, CGroups, and seccomp to help to isolate the container workloads and reduce or eliminate the likelihood of a rogue process from a compromised container from spreading to adjacent workloads and negatively impacting the entire host.

Linux namespaces enable creating an abstraction of a global system resource to make it appear as a separate instance to processes within a namespace. This allows multiple containers to utilize the same resource simultaneously without creating a conflict. SELinux provides an additional layer of security to keep containers isolated from each other and from the host. SELinux allows administrators to enforce mandatory access control for every user, application, process, and file. CGroups (control groups) limit, account for, and isolate the resource usage (e.g., CPU, memory, disk I/O, network) of a collection of processes. CGroups are used to ensure that containers on the same host are not impacted by each other. Seccomp (Secure Computing Mode) profiles can be associated with a container to restrict available system calls. For the protection of the OCP cluster and workloads, RHEL CoreOS employs security mechanisms to isolate processes and memory space such as SELinux and seccomp.

The CE&BA can select, implement, and utilize third-party tools to enable vulnerability scanning for containers, detection of unauthorized configuration changes in containers, and detect and respond to malware in containers. Red Hat provides multiple registry options where the CE&BA can store and manage container images in the context of their CI/CD pipeline and in accordance with their development, security, and operations (DevSecOps) program. Red Hat offers the Red Hat Registry registry.redhat.io, which requires authentication from whence partner images can be acquired from the Red Hat Container Catalog. Red Hat also has Quay.io available which offers several security capabilities including vulnerability scanning and support for managing the integrity of container images. Red Hat also has a private version of Quay.io called Red Hat Quay that includes security features that are useful for the protection of images. OCP also comes with its own private container registry that runs on the cluster.

All OCP traffic to the masters is secured using mutually authenticated HTTPS communication. Certificates for the mutual authentication are checked against a built-in OCP certificate authority (CA). OpenShift Service Mesh can be installed per project to automatically encrypt traffic between pods.

End-to-end encryption for applications in OCP can be supported with OpenShift routes, including features that can support use cases requiring two-way SSL authentication. There are three ways to perform TLS termination with secure routes in OCP, including edge, re-encryption, and passthrough. Edge and re-encrypt routes can be created with custom certificates

OCP also supports encryption of the Container Platform Cluster datastore to secure secrets data or any other resource data that is contained in the database. RHEL CoreOS volumes can also be encrypted.

For connected OCP implementations, Red Hat provides a secure mechanism to keep OCP and RHEL CoreOS updated. This allows the CE&BA to ensure that any vulnerabilities or security issues that should be discovered regarding the OCP are addressed in a timely manner.

Identity and Authentication

164.312(d) Person or Entity Authentication: Implement procedures to verify that a person or entity seeking access to electronic protected health information is the one claimed.

Guidance

The CE&BA should establish policies, procedures, and standards for person and entity authentication. These policies, procedure, and standards should include requirements and mechanisms for corroboration

of a person's identity, authentication of the validity of transmission sources, and verifying claims that a person or entity is authorized for access with specific privileges to the information and information systems. The choice of authentication and identification methods should be sufficiently strong to support nonrepudiation.

Identity and Authentication and OCP

Red Hat recommends using a supported third-party identity and authentication provider for centralized management of user identities and authenticators. OCP includes a built-in OAuth server. Requests to the OCP API are authenticated using OAuth Access Tokens or X.509 Client Certificates depending on access method. Users obtain OAuth access tokens to authenticate themselves to the API. When a person requests a new OAuth token, the OAuth server uses the configured identity provider to determine the identity of the person making the request. It then determines what user that identity maps to, creates an access token for that users, and returns the token for use.

All request for OAuth tokens involves a request to *<namespace_route/oauth/authorize>*. Most authentication integrations place an authenticating proxy in front of this endpoint or configure OCP to validate credentials against a backing identity provider. Red Hat provides guidance and requirements for integrating an identity and authentication provider for access to OCP. Multiple options are available as supported identity providers including HTPasswd, Keystone, LDAP, Basic Authentication, Request header, GitHub or GitHub Enterprise, GitLab, Google, and OpenID Connect. The CE&BA should choose an identity provider and authentication method that best fits their operational standards and security goals. The use of an external third party-integrated identity and authentication provider allows the CE&BA to also implement multi-factor authentication for further reduction in risk as necessary. Access to OCP should be limited to the CE&BA's defined access methods. Any vectors of access outside of these defined parameters should be well documented and tightly controlled as part of emergency break glass procedures in the event the primary access methods are not functioning.

OCP implements a user agent that can be used to prevent an application developer's CLI accessing the OCP API. If a client uses a library or binary file, they cannot access the OCP API. This allows platform administrators to prevent clients from access the API with the userAgentMatching configuration setting of a master configuration.

OCP make use of service accounts to allow components to directly access the API. Service accounts are API objects that exist within each project. Service accounts provide a flexible way to control API access without sharing a regular user's credentials. Each service account's name is derived from its project and name. Service accounts are members of two groups: system:serviceaccounts, and system:serviceaccounts:<project>. Service accounts automatically contain two secrets: An API token and Credentials for the OpenShift Container Registry. Service accounts can be granted roles in the same way that roles are granted to regular user accounts.

OCP provides administrators the ability to configure LDAP Sync to sync LDAP records from an LDAP provider with the OCP records. This allows the CE&BA to manage groups in one place.

Transmission Security

164.312(e)(1) Transmission Security: Implement technical security measures to guard against unauthorized access to electronic protected health information that is being transmitted over an electronic communications network.

Guidance

Coalfire recommends that the CE&BA identify any possible unauthorized sources that may be able to intercept or modify information while it is being transmitted either externally over the internet or internally

on the internal network. Understanding scenarios where data could be compromised while in transit should guide the CE&BA toward the selection and implementation of controls to minimize the risk to data compromise.

Based on an understanding of the risk to data being compromised during transit over a network, the CE&BA should develop policies, procedures, and standards that define how data will be secured while being transmitted. An understanding of the flow of data over transmission links should be useful to the CE&BA to identify where controls should be applied.

Methods for securing data in transit include encryption of data while in transit, isolation or segmentation of networks to minimize network access to only what is necessary, implementation of tools to inspect data in transit to ensure that the data is not compromised, and monitoring the networks for intrusion or compromise.

For communications between endpoints within the OCP, container network capabilities are provided to control the flow of information within the container environment and between pods. Kubernetes tells a node to attach a pod to a network. The network plugin then allocates to the pod an IP address from an internal SDN network. This ensures that all containers within the pod behaves as if they are on the same host and causes all containers within a pod to share network space. Giving each pod its own IP address means that pods can be treated from a network perspective similarly to physical hosts or virtual machines in terms of port allocation, networking, naming, service discovery, load balancing, application configuration, and migration.

OCP and Transmission Security

OpenShift uses an SDN approach to provide a unified cluster network that enables communication between pods across the OCP cluster. This pod network is established and maintained by the OpenShift SDN mode, which configures an overlay network using OVS. Almost all packet delivery decisions are performed with OpenFlow rules in the OVS bridge br0, which provides flexible routing. Depending on the deployed SDN plugin, network isolation can be enforceable and finely controlled.

In a cluster using the CNI plug-in that supports NetworkPolicy, network isolation is controlled entirely by NetworkPolicy objects. OpenShift SDN supports using NetworkPolicy in its default isolation mode. By default, all pods in a project are accessible from other pods and network endpoints. To isolate one or more Pods in a project, a NetworkPolicy object can be created in the project to indicate the allowed incoming connections. If a pod is matched by selectors in one or more NetworkPolicy objects, then the pod will accept only connections that are allowed by at least one of those NetworkPolicy objects. A pod that is not selected by any NetworkPolicy objects is fully accessible. A NetworkPolicy object can be created that matches all pods but accepts no traffic to deny all traffic by default. NetworkPolicy objects are additive which means that multiple NetworkPolicy objects can be combined to satisfy complex network requirements.

OCP also provides egress support for traffic leaving the OCP environment (e.g., a pod sending a request to an external database). The OpenShift SDN can be configured to assign one or more egress IP addresses to a project. All outgoing external connections from the specified project will share the same fixed source IP, allowing external resources to recognize the traffic based on the egress IP. The customer can setup firewall or routing rules on the physical network to control the traffic between the resource in OCP and the database given the source IP as the egress IP assigned to the resource.

Red Hat offers OpenShift Service Mesh as an optional component to be employed that provides several key capabilities uniformly across a network of services such as traffic management, service identity and security, policy enforcement, and telemetry. Traffic Management controls the flow of traffic and API calls between services, makes calls more reliable, and makes the network more robust in the face of adverse conditions. Service identity and security provides services in the mesh with a verifiable identity and provides

the ability to protect the service traffic as it flows over networks of varying degrees of trustworthiness. Policy enforcement allows the organization to apply policy to the interaction between services, ensure that the access policies are enforced, and resources are fairly distributed among consumers. Policy changes can be made by configuring the mesh, not by changing the application code. Telemetry allows the organization to gain understanding of the dependencies between services and the nature and flow of traffic between them. This allows the organization to quickly identify issues.

All OCP traffic from Nodes to the masters is secured using mutually authenticated HTTPS (TLS 1.2) communication. Internal OpenShift component to component traffic is also secured using mutually authenticated TLS 1.2 encryption. Certificates for the mutual authentication are checked against a built-in OCP certificate authority (CA).

CONCLUSION

OCP hosted on RHEL CoreOS, as reviewed by Coalfire, can be effective in providing support for the outlined objectives and requirements of the HIPAA security rule in support of a HIPAA compliance program. Through proper implementation and integration into the organization's greater technical infrastructure and information security management systems, OCP may be useable in a HIPAA-controlled environment. Care should be given for the implementation of OCP and the use of the platform for the deployment of containers in support of microservice architectures. The organization wishing to use OCP should consider the guidance provided by NIST SP 800-190 when designing their implementation.

Coalfire's opinion is based on observations and analysis of the provided documentation, interviews with Red Hat personnel, and hands-on engagement with a lab environment. The provided conclusions are based upon several underlying presumptions and caveats, including adherence to vendor best practices and hardening of configuration as supported by the system components. This solution should be implemented in alignment with the organization's mission, values, business objectives, general approach to security and security planning, and with respect to the overall organizational security and compliance program.

ADDITIONAL INFORMATION, RESOURCES, AND REFERENCES

https://openshift.com

https://docs.openshift.com/container-platform/4.1/welcome/index.html

https://docs.openshift.com/container-platform/4.2/networking/cluster-network-operator.html

https://docs.openshift.com/container-platform/4.2/networking/configuring-ingress-cluster-traffic/configuring-ingress-cluster-traffic-ingress-controller.html

https://blog.openshift.com/introducing-red-hat-openshift-4/

https://www.openshift.com/learn/topics/operators

https://docs.openshift.com/container-platform/4.2/architecture/architecture.html#architecture-key-features_architecture

https://www.redhat.com/cms/managed-files/cl-container-security-openshift-cloud-devops-tech-detail-f7530kc-201705-en.pdf

https://csrc.nist.gov/projects/risk-management/detailed-overview

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

https://csrc.nist.gov/Projects/Risk-Management/Risk-Management-Framework-Quick-Start-Guides/Step-0-Prepare

https://csrc.nist.gov/Projects/Risk-Management/Risk-Management-Framework-Quick-Start-Guides/Step-1-Categorize

https://csrc.nist.gov/Projects/Risk-Management/Risk-Management-Framework-Quick-Start-Guides/Step-2-Select

https://csrc.nist.gov/Projects/Risk-Management/Risk-Management-Framework-Quick-Start-Guides/Risk-Management-Framework-(RMF)-Step-6-Monitor

https://csrc.nist.gov/Projects/Risk-Management/Risk-Management-Framework-(RMF)-Overview/Security-Controls

https://csrc.nist.gov/publications/detail/fips/199/final

https://csrc.nist.gov/publications/detail/fips/200/final

https://csrc.nist.gov/publications/detail/sp/800-53/rev-4/final

https://csrc.nist.gov/publications/detail/sp/800-37/rev-1/final

https://csrc.nist.gov/publications/detail/sp/800-66/rev-1/final

https://www.hhs.gov/hipaa/index.html

https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/administrative/securityrule/securityrulepdf.pdf?language=es

ABOUT THE AUTHORS

Jason Macallister | Senior Consultant, Cyber Engineering, Coalfire
Jason Macallister joined Coalfire in 2015 and brings over 20 years of experience in architecture and engineering cloud technology. Jason consults on information security and regulatory compliance topics as they relate to advanced infrastructure and emerging products and solutions.

Terilyn Floyd-Carney | Senior Consultant, Healthcare & Life Sciences, Coalfire
Terilyn Floyd-Carney joined Coalfire in 2014, bringing over 20 years of experience in the information technology security field to the Healthcare team. She has led extensive consulting and assessment engagements against HIPAA Safeguards, the HITRUST Framework, DEA EPCS regulations, and FDA standards.

Published February

ABOUT COALFIRE

Coalfire is the cybersecurity advisor that helps private and public-sector organizations avert threats, close gaps, and effectively manage risk. By providing independent and tailored advice, assessments, technical testing, and cyber engineering services, we help clients develop scalable programs that improve their security posture, achieve their business objectives, and fuel their continued success. Coalfire has been a cybersecurity thought leader for more than 16 years and has offices throughout the United States and Europe. Coalfire.com