CCN-CERT

BP/27

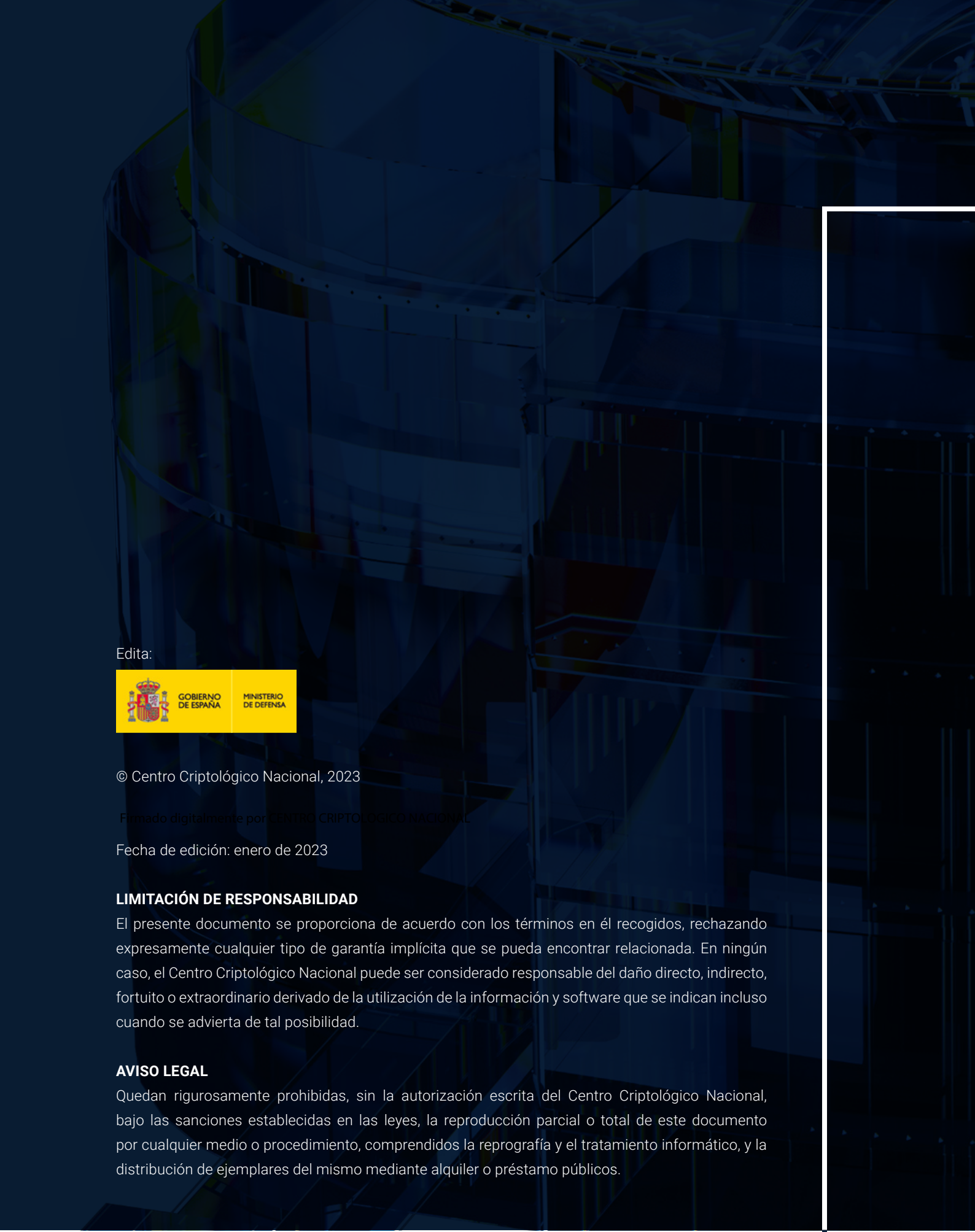
Recomendaciones de seguridad en Kubernetes

INFORME DE BUENAS PRÁCTICAS

SEPTIEMBRE 2022

ccn-cert centro criptológico nacional

CCN centro criptológico nacional



Edita:

GOBIERNO DE ESPAÑA MINISTERIO DE DEFENSA

Fecha de edición: enero de 2023

Índice

1. Introducción **4**
2. Kubernetes **5**
2.1 Versiones 5
2.2 Componentes 6
2.2.1 Plano de control de Kubernetes 6
2.2.2 Nodos trabajadores 8
2.3 Arquitectura 9
3. Seguridad en Kubernetes **11**
3.1 Seguridad en infraestructura 12
3.1.1 Virtualización 12
3.1.2 Protección basada en kernel 12
3.1.3 Políticas y segmentación de red 13
3.1.4 Namespaces de contenedores 15
3.1.5 Políticas de recursos 15
3.1.6 Fortificación del plano de control 17
3.1.7 Fortificación nodos trabajadores 18
3.1.8 Información sensible 20
3.1.9 Actualizaciones 20
3.2 Seguridad en contenedores y pods 22
3.2.1 Control de acceso a los pods 22
3.2.2 Control de acceso basado en roles 24
3.2.3 Contenedores inmutables 25
3.2.4 Creación de contenedores 25
3.2.5 Seguridad de los pods 26
3.2.6 Seguridad en los entornos de Kubernetes 27
3.3 Incidentes en los tiempos de ejecución 28
3.3.1 Contenedores no “root” 28
3.3.2 Seguridad por tokens en cuentas de servicio 28
3.3.3 Registros de auditoría 29
3.3.4 Logs del sistema 30
3.3.5 Registros de auditoría en Kubernetes 31
3.3.6 Registro de contenedores y nodos trabajadores 33
3.3.7 Auditoría por SECCOMP 34
3.3.8 Detección de amenazas 35
4. Lista de comprobación **36**
5. Decálogo de recomendaciones **39**
6. Glosario **41**

1. Introducción

Con la llegada de la tecnología de contenerización (Docker), se produjo una forma de crear, implementar y dimensionar rápidamente aplicaciones y servicios o realizar el despliegue de software.

Obteniendo el apoyo de grandes compañías como Google, RedHat o Microsoft entre otras, se ha convertido en una opción recurrente a la hora de aplicar soluciones por las empresas, ya sea de una forma local "on-premise" o con un alojamiento en la nube. Es una tecnología con una continua evolución, consiguiendo la necesidad de generar cada vez más microservicios críticos para el correcto funcionamiento de una empresa o entidad, llegando a realizar estructuras de alta disponibilidad con redundancia de recursos, y la necesidad de proteger los datos que manejan los servicios alojados.

Ante este progreso, y al forjar cada vez estructuras más complejas, surge la necesidad de generar una orquestación de todos los conjuntos de contenedores, emergiendo la figura de Kubernetes.

Kubernetes, es una palabra originaria del griego que significa "timonel" o "piloto", es el que se encarga de gestionar la coordinación para la realización de despliegues, la supervisión de los servicios, así como el escalado de recursos. En definitiva, la gestión de todos los servicios o microservicios generados por una arquitectura distribuida de los sistemas.

Kubernetes es una plataforma para orquestar contenedores, manejando la coordinación, supervisión y escalado de recursos en una arquitectura distribuida. La tecnología de contenerización (Docker) es recurrente en empresas.

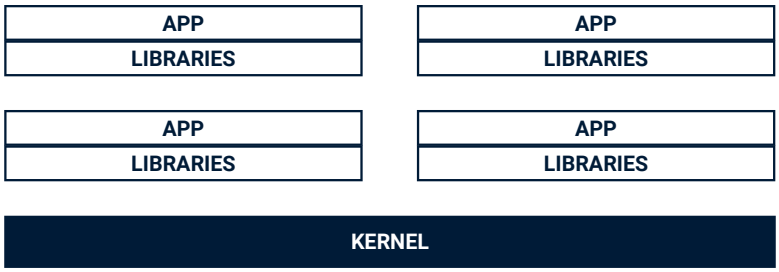


Ilustración 1 - Estructura de contenedores

2. Kubernetes

Kubernetes es una plataforma de código abierto diseñada para la administración de clústeres de servicios y aplicaciones implementados en contenedores, pudiendo estar alojados de forma “on-premise” o en la “nube”. Su funcionalidad principal es la administración centralizada y automatización de las cargas de trabajo.

Desarrollado por Google y los ingenieros Craig McLuckie, y Brendan Burns, siendo liberado el proyecto en el año 2014, está gestionado por la Fundación de computación nativa en la nube, de siglas CNCF, perteneciente como una sección de la Linux Foundation.

Kubernetes se suele denominar “k8s”, siendo una forma de abreviación obtenida del remplazo de las ocho letras de Kubernetes, comprendidas entre la “k” inicial y la “s” final.

El uso de esta tecnología, al soler ser implementada de una forma virtualizada, proporciona varias opciones de flexibilidad en su configuración y diferentes capas de seguridad.

Kubernetes es una plataforma para administrar clústeres de aplicaciones y servicios contenidos, ofrece flexibilidad y seguridad al usarse virtualizado.

2.1 Versiones

Kubernetes ofrece una cadencia de lanzamiento principal de tres (3) actualizaciones menores en ciclos anuales, siendo soportadas por parches de seguridad aproximadamente hasta un año después de su lanzamiento.

Las versiones de Kubernetes se expresan con una numerología dividida en tres bloques y separadas por números.

- **Primer bloque.** Es el indicativo de la versión principal o mayor de la versión de Kubernetes.
- **Segundo bloque.** Se muestra como la versión menor o secundaria.
- **Tercer bloque.** Indica el número de versión de parche aplicado al aplicativo de Kubernetes.

NOTA: Puede obtener información adicional sobre el ciclo de vida del producto a través del siguiente enlace:

https://Kubernetes.io/releases/release/

2.2 Componentes

Un clúster de Kubernetes se puede definir como un conjunto de distintos nodos, que pueden ser diferentes máquinas virtuales o físicas. Dichos nodos se clasifican en:

- Plano de control (Control Plane).
- Nodos trabajadores (worker).

2.2.1 Plano de control de Kubernetes

Se encarga de la administración del ecosistema de Kubernetes, y toma las decisiones globales del clúster, reaccionando a todos los eventos desencadenados por los diferentes nodos, como podría ser la respuesta a la creación de una réplica de tareas.

Los componentes del plano de control son los siguientes.

- **Kube-apiserver:** Se encarga de la interacción con los distintos usuarios del sistema a través de una interfaz de control de sistema, gestionando las peticiones al clúster de Kubernetes. El servidor de la API de Kubernetes, determina qué solicitudes son válidas y cómo procesarlas.
- **ETCD:** Es una clase de almacenamiento de datos no volátil, donde se almacena toda la información relevante a la configuración del clúster de Kubernetes.
- **Kube-sheduler:** Es donde se ordenan y planifican las prioridades en la gestión del clúster mediante el uso de APIs desde la línea de comandos.
- **Kube-controller-manager:** El encargado de realizar la revisión de los procesos independientes de los componentes del ecosistema de Kubernetes. Dichos componentes son:
  - **Control de replicación.**
  - **Controlador de puntos finales (servicios y pods).**
  - **Controlador de tokens y cuentas de servicio (namespaces).**
  - **Controlador de nodos.**
- **Cloud-controller-manager:** Se encarga de la ejecución de controladores que interactúan con la nube. Puede interactuar con balanceadores de carga, intermediar con actualizaciones o supresiones de distintos nodos trabajadores o configurar rutas de red.

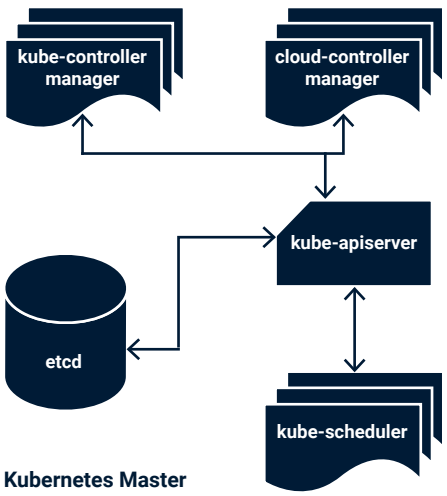


Ilustración 2 - Estructura de nodo maestro.

2.2.2 Nodos trabajadores

En los inicios de la creación de Kubernetes, al nodo trabajador (worker) se le denominó "minion", pudiendo estar formado por una o varias máquinas, tanto virtuales como máquinas físicas, todo ello dependiente del tipo de clúster.

Cada nodo trabajador está controlado por un "nodo maestro", que contiene todas las herramientas necesarias para la gestión de uno o varios "pods".

Un pod o pods, es una agrupación lógica usada por Kubernetes para realizar una gestión más uniforme de un conjunto de uno o más contenedores en los que alojan servicios o aplicaciones, que son ofrecidos por Kubernetes como la solución final del producto.

Los contenedores en el pod, comparten los recursos del sistema en cuanto a ubicación de espacio de disco y los nombres de red (espacios de nombre) que se les otorga para su identificación.

Si varios pods comparten una funcionalidad, para realizar una gestión eficaz y homogénea, Kubernetes agrupa todos esos pods en una unidad lógica denominada "servicios" facilitando así, por ejemplo, la asignación de IPs externas de acceso, o facilitando un balanceo de alta disponibilidad.

Los componentes de un nodo trabajador, son los enumerados a continuación:

- **Kubelet:** Cada nodo de un clúster de Kubernetes posee un agente, el cual tiene la competencia de verificar la ejecución de los pods, comunicándolo al plano de control.
- **Kube-proxy:** Agente encargado de las reglas de red, así como de los reenvíos de conexiones con el anfitrión que aloja el clúster.
- **Entorno de ejecución de contenedores:** Es la ejecución de los contenedores en sí en el entorno de Kubernetes, soportando diferentes tecnologías de contenerización como Docker, containerd, cri-o o rktlet. Básicamente ejecuta cualquier implementación de contenerización capaz de soportar una interfaz de runtime de Kubernetes (Kubernetes CRI).

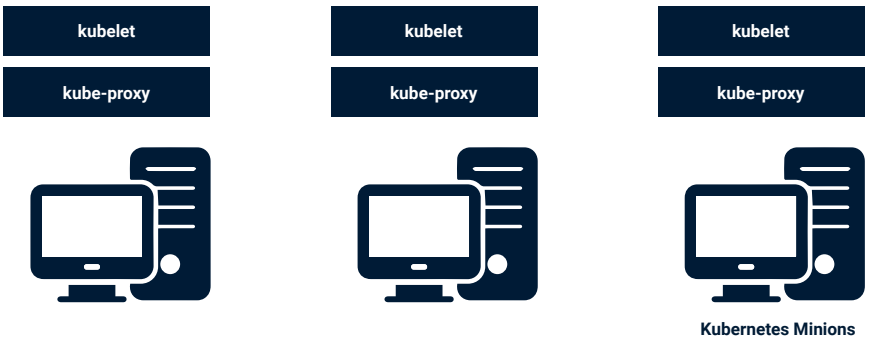


Ilustración 3 - Estructura de nodo worker.

2.3 Arquitectura

La arquitectura utilizada por Kubernetes, es una arquitectura de "maestro-esclavo" entre los nodos maestros y los nodos trabajadores.

Los servidores o equipos de un clúster, suelen comprender uno o varios "nodos", que a su vez pueden estar alojados en uno o varios servidores "on-premise" físicos o virtuales o en la nube.

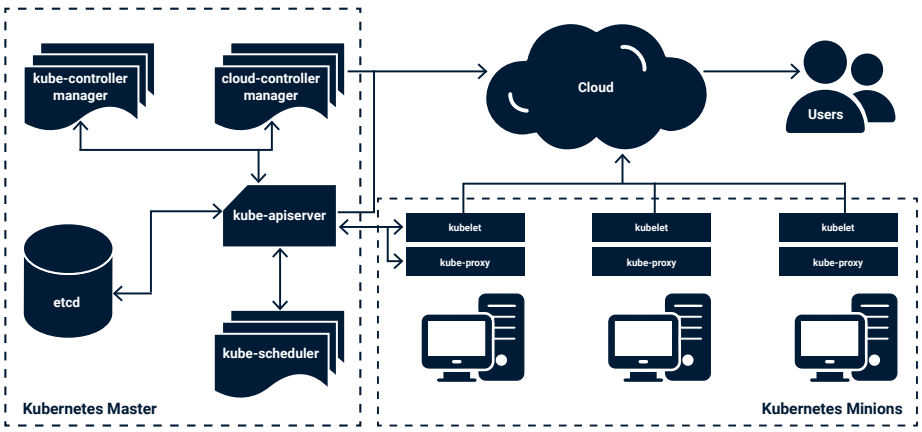


Ilustración 4 - Arquitectura clúster Kubernetes

2. Kubernetes

Los clústeres de Kubernetes, cuando son alojados en la nube, deben ser ubicados en proveedores de servicios certificados por Kubernetes, los cuales administrarán parte de la infraestructura del entorno de Kubernetes. Se debe tener en cuenta que la mayoría de las configuraciones usadas por defecto por parte de dichos proveedores de servicios, no suelen ir acompañadas de unas políticas de seguridad férreas, sino que se prioriza la funcionalidad y el acceso al servicio.

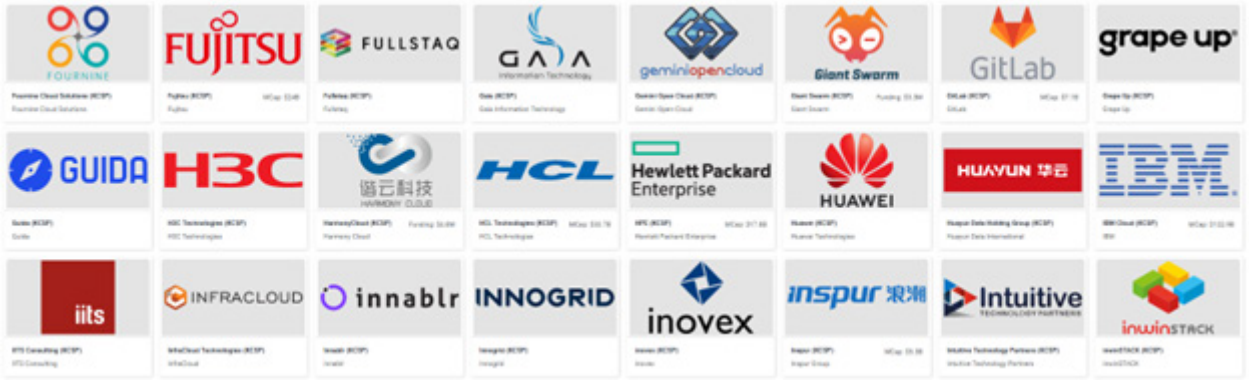


Ilustración 5 - Ejemplo de proveedores certificados de servicio.

NOTA: Si desea consultar el listado completo de CSP (proveedores certificados de servicio) puede encontrar más información en el enlace

https://Kubernetes.io/partners/#kcsp

Una organización que necesite tener funciones adicionales de seguridad y tener un control sobre ellas, puede elegir establecer los servicios de Kubernetes por medio de servidores locales “on-premise”.

3. Seguridad en Kubernetes

Una organización puede tener todos sus datos alojados en un clúster de Kubernetes, información que puede ser codiciada por ciberdelincuentes.

Así mismo, la consecución de la potencia de cálculo de los equipos que alojan los diferentes procesos y servicios de Kubernetes, puede ser usurpado para realizar uso de técnicas para la extracción de cripto monedas por métodos de minería.

Otro método de agresión por parte de un ciberdelincuente, es la denegación de los servicios públicos que puedan ofrecer las diferentes soluciones de Kubernetes por parte de una organización, provocando caídas de servicios y llegando a ocasionar pérdidas económicas.

La seguridad de Kubernetes se puede dividir en tres puntos:

- **Elementos que conforman el sistema y su infraestructura.**
- **Configuraciones de los componentes de Kubernetes.**
- **Incidentes en los tiempos de ejecución.**

La seguridad en Kubernetes se divide en tres puntos: protección de los datos, evitar el uso indebido de recursos y prevenir ataques de denegación de servicio.

3.1 Seguridad en infraestructura

3.1.1 Virtualización

El empleo de tecnología de virtualización para la ejecución de un sistema de contenedores, puede otorgar capas extras de seguridad para el funcionamiento de las diferentes soluciones y servicios. A dicho sistema se le otorga una "fracción" de los recursos totales del servidor físico, por lo cual no puede acceder a la totalidad del conjunto de recursos del host donde está alojado, evitando una denegación total del sistema. Se dota de un aislamiento de los contenedores.

La virtualización de los sistemas otorga la funcionalidad de realizar "puntos de restauración", en caso de necesidad de volver a un momento anterior a una configuración aplicada al sistema.

3.1.2 Protección basada en kernel

El uso de herramientas de kernel como "seccomp", se puede emplear para la limitación de las llamadas que realiza un contenedor en el sistema, reduciendo así la exposición ante un posible atacante.

Otra herramienta de protección por kernel del sistema, es SElinux. El uso del contexto de seguridad otorgado por SElinux, proporciona al contenedor una protección contra cambios no deseados y un nivel mayor de auditoría de seguridad.

3.1.3 Políticas y segmentación de red

Los contenedores dentro de un Pod comparten dirección IP y puerto, dichas direcciones IP deben de ser declaradas en los archivos de configuración "localhost" para poder realizar las conexiones.

Teniendo en cuenta que los intercambios de información entre los pods y los servicios se realizan por comunicaciones de red, es un punto a tener en cuenta la segmentación de redes, para realizar las conexiones entre los distintos elementos de una forma segura, aislando los contenedores y servicios que sean necesarios.

Se debe realizar una correcta política de red, delimitándola con el uso de cortafuegos y cifrado del tráfico permitido para la protección del tránsito de datos entre los diferentes puntos del sistema.

El cifrado de las comunicaciones se ha de efectuar mediante certificados TLS, en sus versiones TLS 1.2 o TLS 1.3, versiones de protocolos considerados seguros.

En la siguiente tabla se muestran los principales parámetros de configuración para las comunicaciones cifradas por certificados.

Fichero	Parámetro	Valor	Observaciones
kube-apiserver.yaml	etcd-certfile	[ruta]	Conexión cifrada entre APIserver y etcd
kube-apiserver.yaml	etcd-keyfile	[ruta]	Conexión cifrada entre APIserver y etcd
kube-apiserver.yaml	tls-cert-file	[ruta]	APIserver uso de tráfico cifrado
kube-apiserver.yaml	tls-private-key-file	[ruta]	APIserver uso de tráfico cifrado
etcd.yaml	auto-tls	false	Evitar uso certificados autofirmados
etcd.yaml	peer-client-file	[ruta]	Configuración TLS para etcd
etcd.yaml	peer-key-file	[ruta]	Configuración TLS para etcd
etcd.yaml	peer-client-cert-auth	true	Autentificación segura etcd
etcd.yaml	peer-auto-tls	false	Evitar uso certificados autofirmados

Fichero	Parámetro	Valor	Observaciones
etcd.yaml	trusted-ca-file	[ruta]	Uso de entidad certificadora
10-kubeadm.conf	tls-cert-file	[ruta]	Uso de tráfico cifrado para Nodo Trabajador
10-kubeadm.conf	tls-private-key-file	[ruta]	Uso de tráfico cifrado para Nodo Trabajador
10-kubeadm.conf	RotateKubeletServerCertificate	true	Rotación de certificados para servidor kubelet

Tabla 1 – Configuraciones de TLS.

Para la creación de políticas de red, es necesario un complemento de interfaz de contenedores (CNI), que admita APIs de NetworkPolicy.

Cada pod del conjunto de Kubernetes posee su propia IP privada en el clúster y se puede tratar igual que en un equipo real en el que se le asigna un puerto a la dirección ip (socket). Estos sockets, pueden alojar recursos y accesos a aplicativos de una organización, por lo cual debe existir un elemento que proporcione una estabilidad a los elementos de red entre los pods Kubernetes, puesto que, en un proceso, por ejemplo, de cambio o actualización de pods, estas direcciones IPs se pueden modificar dando como resultado una pérdida de conexión o perdida de acceso a un recurso.

Kubernetes soluciona estas posibles pérdidas de conexión por cambio de IP, unificando direcciones IP en conjuntos lógicos de Pods mediante servicios que son etiquetados.

La forma de acceder a estos servicios desde una fuente externa a Kubernetes, se suele hacer por “NodePorts”, otorgando de manera aleatoria un puerto a una IP, pudiéndose configurar de forma estática.

Para una correcta segmentación en la red, los complementos CNI de Kubernetes pueden usar para este cometido, los “nodeports” o cortafuegos, además de balanceadores de carga. De esta forma, al segmentar la red se limita la superficie de ataque a un ciberdelincuente.

3.1.4 Namespaces de contenedores

Los "espacios de nombres" son la forma que se tiene de delimitar los recursos de Kubernetes, asignando una etiqueta a un ámbito de recursos del clúster, y formando así una unidad en la que se pueden aplicar una serie de autorizaciones RBAC (control de acceso basado en roles) y políticas de red.

Hay tres espacios de nombres predeterminados:

- **Kube-system. Usado para los componentes de Kubernetes.**
- **Kube-public. Empleado para etiquetar los recursos públicos.**
- **Default. Son recursos de usuarios.**

3.1.5 Políticas de recursos

Una buena práctica en la gestión y para evitar un desbordamiento en los recursos del servidor de Kubernetes, es limitar el uso de los recursos físicos de hardware.

Las restricciones se pueden realizar por contenedores o por espacios de nombres. Las restricciones se realizan creando ficheros de políticas con la extensión "yaml" y aplicándolos mediante el comando "kubectl". A modo de ejemplo se muestra una creación de políticas de limitación de recursos y cómo aplicarlas en un clúster de Kubernetes.

Paso	Descripción
1.	Cree un espacio de nombres para que los recursos estén aislados del resto de su clúster. `$ kubectl create namespace nombre-quota-mem-cpu`

2.	Cree el fichero de configuración de políticas con la extensión "yaml" [política.yaml]:
	`apiVersion: v1`
	`kind: ResourceQuota`
	`metadata:`
	`  name: mem-cpu-demo`
	`spec:`
	`  hard:`
	`  requests.cpu: "1"`
	`  requests.memory: 1Gi`
	`  limits.cpu: "2"`
	`  limits.memory: 2Gi`

3.	Aplique la política de limitación de recursos con el siguiente comando. `$ kubectl apply -f [política.yaml] --namespace=nombre-quota-mem-cpu`
4.	Puede comprobar la aplicación de la política con el siguiente comando. `$ kubectl get quota --namespace=nombre-quota-mem-cpu`

3.1.6 Fortificación del plano de control

Es el núcleo de Kubernetes, gestionando los contenedores, las diversas tareas de mantenimiento, así como la gestión de la información sensible del clúster, por lo cual es un punto sensible en la fortificación de todo el ecosistema de Kubernetes.

Se deben usar comunicaciones cifradas por protocolos TLS (en sus versiones 1.2 y 1.3), poseer un control de acceso basado en roles (RBAC) y métodos de autenticación robustos con una suficiente complejidad en las contraseñas.

Otra medida es el control de acceso vía red de las comunicaciones que realiza la API de Kubernetes, limitando mediante cortafuegos los tipos de conexiones y direccionamiento de las mismas, para establecer un correcto uso de la API con la mayor fiabilidad posible.

Seguidamente se muestra una tabla con los principales puertos usados por la API de Kubernetes, así como sus protocolos.

Componente Nodo	Orientación tráfico	Protocolo	Puertos
Kube-controller-manager	Entrante	TCP	10257
Kube-scheduler (programador de tareas)	Entrante	TCP	10259
API Kubernetes	Entrante	TCP	6443
API etcd	Entrante	TCP	2379-2380
API kubelet	Entrante	TCP	10250

Tabla 2 – Tráfico de red API Kubernetes.

Los ficheros de configuración locales que gestionan los componentes del nodo de control han de tener una gestión de acceso por permisos y usuarios.

Fichero	Permisos	Propietario
kube-apiserver.yaml	644 o más restrictivos	root:root
kube-controller-manager.yaml	644 o más restrictivos	root:root
kube-scheduler.yaml	644 o más restrictivos	root:root
etcd.yaml	644 o más restrictivos	root:root
Configuraciones CNI	644 o más restrictivos	root:root
BBDD_etcd	644 o más restrictivos	root:root
admin.conf	644 o más restrictivos	root:root
scheduler.conf	644 o más restrictivos	root:root
controller-manager.conf	644 o más restrictivos	root:root
Certificados pki (extensión crt)	640	root:root
Llaves certificados pki (extensión key)	400	root:root

Tabla 3 – Membresía y permisos de ficheros del nodo de control.

NOTA: La ubicación por defecto de los ficheros "yaml" es "/etc/Kubernetes/manifests/". La ubicación por defecto de los ficheros "conf" es "/etc/Kubernetes/". Para más información visite la página

https://Kubernetes.io/docs

3.1.7 Fortificación nodos trabajadores

Del mismo modo que ocurre en un nodo maestro, se ha de tener un control de acceso a la red de cada nodo trabajador integrante del clúster de Kubernetes. Mediante la Identificación de la orientación del tráfico de red de cada nodo, se puede limitar el acceso a los nodos con el uso de un cortafuegos, configurándolos por medio de sus sockets.

Se muestran a continuación los puertos y servicios usados por los nodos trabajador de Kubernetes.

Componente Nodo	Orientación tráfico	Protocolo	Puertos
Servicios Nodo	Entrante	TCP	30000-32767
API kubelet	Entrante	TCP	10250

Tabla 4 – Tráfico de red nodo trabajador.

Los ficheros de configuración locales que gestionan los componentes del nodo de control han de tener una gestión de acceso por permisos y usuarios.

Fichero	Permisos	Propietario
10-kubeadm.conf	644 o más restrictivos	root:root
Archivo "kubeconfig" de kube-proxy	644 o más restrictivos	root:root
kubelet.conf	644 o más restrictivos	root:root

Tabla 5 – Membresía y permisos de ficheros del nodo trabajador.

NOTA: Para más información puede consultar los siguientes enlaces:

https://Kubernetes.io/docs/admin/kube-proxy/

https://Kubernetes.io/docs/admin/kubelet/

https://Kubernetes.io/docs/tasks/administer-cluster/kubelet-config-file/

3.1.8 Información sensible

La información sensible en Kubernetes (secrets o secretos), tales como contraseñas o certificados, se ubican para su uso en las configuraciones en los archivos "yaml", imágenes de contenedores o en variables de entorno, siendo más seguro utilizar su contenido en forma de archivo, y no en variable, puesto que en un fichero se puede regular su acceso por permisos y usuarios.

Esta información sensible es almacenada por Kubernetes sin cifrar, la codifica en base64 pudiendo ser cifrada, configurando su cifrado de datos en el servidor API.

A continuación, se muestra a modo de ejemplo la activación de la configuración de cifrado del servidor API.

Fichero	Permisos	Propietario
kube-apiserver.yaml	--encryption-provider-config	/[ubicación fichero]/secrets.yaml

Tabla 6 – Configuración apiserver.

NOTA: Para más información puede consultar el siguiente enlace.

https://Kubernetes.io/docs/tasks/administer-cluster/encrypt-data/.

3.1.9 Actualizaciones

Con el paso del tiempo, los programas y aplicaciones del entorno de Kubernetes son susceptibles de tener fallos de seguridad como consecuencia de la continua evolución de los ataques y aparición de vulnerabilidades y exposiciones de seguridad. Por este motivo, pueden necesitar ser actualizados y/o aplicarles parches de seguridad, independientemente de los medios en que estén implementados, que a su vez deberán aplicar las actualizaciones y/o parches de seguridad publicados por sus correspondientes fabricantes.

3. Seguridad en Kubernetes

Kubernetes ofrece una cadencia de lanzamiento principal de tres (3) actualizaciones menores en ciclos anuales, siendo soportadas por parches de seguridad aproximadamente durante un año después de su lanzamiento.

Las versiones de Kubernetes se expresan con una numerología dividida en tres bloques y separadas por números.

- **Primer bloque.** Es el indicativo de la versión principal o mayor de la versión de Kubernetes.
  .
- **Segundo bloque.** Se muestra como la versión menor o secundaria.
- **Tercer bloque.** Indica el número de versión de parche aplicado al aplicativo de Kubernetes.

Cuando una aplicación sufre una actualización, y sobre todo cuando es una actualización mayor de versión, suele ser necesario que, para una correcta aplicación de los cambios, sea necesario reiniciar el clúster, con lo cual se sufrirá un corte en los servicios desplegados por la solución de Kubernetes. Para evitar estos cortes de servicio, en ciertas ocasiones es necesario la instalación de un sistema de alta disponibilidad para así obtener una redundancia en los servicios.

Hay que destacar que, en un entorno de producción, la aplicación de cambios mayores en las aplicaciones a consecuencia de una actualización, puede provocar la incompatibilidad del aplicativo actualizado con la configuración del mismo antes de la actualización. Se deberán aplicar las actualizaciones previamente en un entorno de preproducción o pruebas, comprobando el correcto funcionamiento de los aplicativos desplegados, y una vez corroborado su correcto funcionamiento, aplicar las actualizaciones y cambios de configuración al entorno de producción.

3.2 Seguridad en contenedores y pods

3.2.1 Control de acceso a los pods

El acceso a un clúster de Kubernetes se realiza por medio de autenticación de usuario. Para evitar ataques de suplantación de identidad y poder así realizar llamadas no permitidas a la API de Kubernetes, la autenticación de usuario admite diferentes mecanismos de acceso que por defecto no se encuentran habilitados.

Hay que hacer una diferenciación en los usuarios de un sistema de Kubernetes, existiendo dos clases de cuentas:

- **Cuentas de servicio.**
- **Cuentas nominales para usuarios.**

Los pods realizan cada uno solicitudes y llamadas a la API de Kubernetes. Esta solicitud se realiza con una cuenta de servicio adjunta al pod, permitiendo realizar la llamada del "controlador de admisión" de "ServiceAccount". Cada espacio de nombres (namespace) debe tener asociada una cuenta de servicio, si esto no fuese así, el controlador de admisión adjuntaría una cuenta predeterminada al espacio de nombres.

Las cuentas de servicio se deben crear individualmente y otorgarles unos permisos específicos para poder delimitar sus funciones.

Las cuentas de servicio predeterminadas no deben usarse, por asignarles una serie de permisos por defecto. Así mismo, al crearse una cuenta por defecto, se le otorga un token para poder acceder a la API de una forma no controlada.

Para evitar la creación de tokens a cuentas de servicio predeterminadas y evitar la asignación de permisos, Kubernetes posee un mecanismo de control para tal fin.

Fichero	Parámetro a configurar	Valor
/var/run/secrets/Kubernetes.io/serviceaccount/token	automountServiceAccountToken	false

Tabla 7 – Configuración acceso pods.

Para usuarios nominales o administradores se debe aplicar un método de autenticación específico. Los métodos para realizar la autenticación en el sistema de Kubernetes, son los siguientes:

- **Certificados X509**
- **Bootstrap tokens**
- **OpenID tokens**

Se debe poseer una política de contraseñas y métodos de autenticación con una complejidad suficientemente elevada para evitar accesos no contemplados, evitando solicitudes o formas de acceso anónimas. En versiones de Kubernetes 1.6 y posteriores, se encuentran habilitadas por defecto las solicitudes anónimas en el sistema.

Se debe de configurar en el servidor API, la denegación de accesos anónimos.

Fichero	Parámetro a configurar	Valor
/etc/Kubernetes/manifests/kube-apiserver.yaml	anonymous-auth	false

Tabla 8 – Configuración acceso apiserver.

3.2.2 Control de acceso basado en roles

El control de acceso basado en roles (RBAC), se utiliza para la gestionar el acceso a los recursos del clúster por los usuarios.

Se debe de tener el control RBAC habilitado en la API, para poder controlar y gestionar de una manera correcta los recursos del clúster. Para inicializar el control de acceso basado en roles, ejecute el siguiente comando.

```
$ kubectl get quota --namespace=nombre-quota-mem-cpu
```

En una política deficiente de seguridad en los roles de acceso, aparecerán políticas de "AlwaysAllow". Tales parámetros de configuración deberán revisarse y establecer los márgenes de políticas de roles de accesos correctos a nivel de operatividad del usuario dentro del clúster, siempre imponiendo una política de mínimo privilegio en el acceso al sistema.

Se pueden establecer dos tipos de permisos:

- **Roles.** Establece permisos para un "nombres de espacio" en particular.
- **Clúster de Roles.** Establece permisos en todos los recursos del clúster, independientemente de los "nombres de espacio".

Los roles son usados para agregar permisos, no poseen reglas de denegación. El servidor de la API de Kubernetes denegará los permisos que no se permitan explícitamente.

Una vez creados los roles o clústeres de roles, para poder asociarlos a un usuario o grupo se utilizarán los "RoleBinding".

Los "RoleBinding" o "ClusterRoleBinding", son utilizados cuando se quieren gestionar una serie de permisos similares a diferentes espacios de nombre.

Los privilegios asignados a usuarios, grupos y cuentas de servicio, deben seguir el principio de mínimo privilegio, dando permisos únicamente para poder realizar una tarea, de igual forma ocurre con los roles.

3.2.3 Contenedores inmutables

En el caso de que uno de los contenedores esté comprometido y que un atacante hubiese tomado el control de este, podría borrar ficheros, crear scripts o incluso modificar los parámetros de configuración de un componente del contenedor.

Todo el escenario anterior se puede evitar bloqueando el sistema de archivos del contenedor por Kubernetes, poniendo el sistema en estado de solo lectura. No obstante, este modo puede entrañar un comportamiento anómalo en el funcionamiento del contenedor, por lo cual si se implementa se ha de realizar un estudio previo para obtener un resultado óptimo de funcionamiento.

Fichero	Parámetro a configurar	Valor
/etc/Kubernetes/manifests/kube-apiserver.yaml	readOnlyRootFilesystem	true

Tabla 9 – Configuración acceso apiserver.

3.2.4 Creación de contenedores

Una imagen de contenedor se puede crear desde su inicio, o generalmente se suele descargar de un repositorio y con la base de esa imagen realizar modificaciones.

Siempre que se descarguen imágenes de repositorios, estos repositorios tienen que ser de fuentes confiables.

Se puede configurar el entorno de Kubernetes para restringir la implementación en el clúster de imágenes de contenedores que no posean una firma digital de un origen confiable.

Cuando se inserta una imagen en el ecosistema de Kubernetes hay que realizar una comprobación en busca de bibliotecas obsoletas, vulnerabilidades conocidas, así como configuraciones que vulneren la seguridad de puertos o permisos inseguros.

3.2.5 Seguridad de los pods

Kubernetes posee dos mecanismos para otorgar de seguridad a los pods del clúster. Estos mecanismos son:

- **Pod Security Admission.**
- **Políticas de Seguridad de Pod (PSP).**

Las "Pod Security Admission", son unas políticas de seguridad implementadas por Kubernetes que vienen predeterminadas a partir de la versión 1.23. Los pods son clasificados (básicos, restringidos o privilegiados) proporcionando una implementación más simple que en PSP.

Con la característica de "Pod Security Admission" activada, se pueden configurar los espacios de nombres para definir un conjunto de modos de control de admisión para el uso de la seguridad en el pod y por cada espacio de nombres.

Kubernetes define un grupo de etiquetas para definir los niveles estandarizados de seguridad del pod.

Modo	Descripción
Enforce	Toda política que incurra en una infracción, se rechazará por el pod.
Audit	Las infracciones de políticas, se aplicarán, pero serán registradas por los registros de auditoría del sistema.
Warn	Las infracciones de políticas, se notificarán al usuario, pero se aplicarán los cambios en el sistema.

Tabla 10 – Etiquetas de políticas de seguridad.

En las políticas de seguridad de pod (PSP) ya en desuso, quedando obsoleta desde la versión 1.21, valida la creación de los pod y actualiza las solicitudes a la PAI del sistema utilizando un conjunto de reglas para su gestión.

3.2.6 Seguridad en los entornos de Kubernetes

Tan importante como la propia seguridad del clúster de Kubernetes es la seguridad del medio que aloja a la solución. La herramienta de gestión del clúster por línea de comandos es "kubectl", la cual se puede desplegar en sistemas operativos de Windows, linux y macOS.

NOTA: Es posible obtener información adicional sobre la instalación y soporte de sistemas operativos en el siguiente enlace:

https://Kubernetes.io/es/docs/tasks/tools/_print/

Si se usa una contenerización alojada en un hipervisor, la virtualización limita el uso del hardware, en vez de hacerlo el sistema operativo en el que se implementaría Kubernetes.

Un aislamiento de un hipervisor es más seguro que el aislamiento tradicional de los contenedores, pudiendo segregar de forma más amplia los diferentes nodos y limitando los recursos del sistema a los asignados por el hipervisor.

En una solución basada en el kernel de linux, se puede usar la herramienta seccomp para limitar las llamadas del sistema de contenedores, reduciendo la superficie de ataque al kernel. Herramientas similares en funcionalidad pueden ser Selinux o APParmor.

Algunas soluciones de motores de contenedores ofrecen un aislamiento entre aplicación de contenedores y el kernel del host denominados "SandBox", confinando los recursos en un espacio virtual aislado.

3.3 Incidentes en los tiempos de ejecución

3.3.1 Contenedores no “root”

De forma predeterminada, hay servicios en los contenedores que se ejecutan con el usuario “root”, usuario con acceso a todos los recursos del sistema. En muchas de esas tareas no es necesario la utilización de dicho usuario administrador, sino que lo puede realizar cualquier otro usuario con una correcta configuración de accesos a los recursos del clúster.

Un contenedor se puede configurar para que ciertas tareas no las realice el usuario “root”, esta configuración no predeterminada se suele establecer en el momento de creación de la imagen del contenedor.

Una alternativa en Kubernetes, es el uso de un pod con el parámetro de “SecurityContext” configurado en “runAsUser”, especificando que el identificador de usuario sea distinto de “0”, puesto de dicho identificador pertenece al usuario “root”.

3.3.2 Seguridad por tokens en cuentas de servicio

Por defecto, Kubernetes otorga un token “secreto” a una cuenta de servicio al crear un pod. Este token es otorgado dentro del pod en el tiempo de ejecución del servicio que desempeñe.

No todas las aplicaciones dentro de los contenedores requieren un acceso directo a las cuentas de servicio. Si una aplicación se ve comprometida por un atacante, pueden estar comprometidas las integridades y los accesos a los tokens, pudiendo ser usados para acceder a la API del sistema.

Se debe realizar un estudio de la necesidad de otorgar un token de manera automática a una cuenta de servicio. Para evitar la creación de tokens a cuentas de servicio predeterminadas y evitar la asignación de permisos, Kubernetes posee un mecanismo de control para tales fines.

Fichero	Parámetro a configurar	Valor
/var/run/secrets/Kubernetes.io/serviceaccount/token	automountServiceAccountToken	false

Tabla 11 – Configuración de accesos.

Si por necesidad no se puede implementar esta medida de seguridad, como por ejemplo en el aprovisionamiento de autenticaciones a servicios externos, se debe de aseverar el control por parte de las políticas de control de acceso basado en roles, minimizando los privilegios del pod dentro del Clúster.

3.3.3 Registros de auditoría

Los registros de auditoría de Kubernetes proporcionan una forma de rastrear información relevante para la seguridad en su sistema. Se generan entradas de registro para monitorizar la mayor cantidad posible de información sobre los eventos que suceden en el sistema. Esta información es crucial en entornos de misión crítica para determinar quién viola las políticas de seguridad y cuáles son las acciones se han realizado. La auditoría no proporciona seguridad adicional a su sistema, más bien se puede usar para descubrir infracciones de las políticas de seguridad utilizadas en el sistema.

Estas violaciones pueden evitarse con medidas de seguridad adicionales que, con el estudio de los eventos recolectados en los logs del sistema ayudan a la elaboración, como son las políticas de control de acceso basado en roles.

Los registros de auditoría monitorizan la actividad del clúster. Cronológicamente se realizan registros en cuestiones de seguridad y en las actividades realizadas por los usuarios del sistema y otros componentes, permitiendo así poder comprobar los hechos acontecidos en un evento en particular.

Los administradores del sistema deben aplicar un método de registro y monitorización de los recursos y de los accesos a estos por parte de cualquier usuario.

Algunos de los puntos más significativos para la monitorización de Kubernetes son:

- **Llamadas a la API.**
- **Métricas de rendimiento.**
- **Consumo de recursos.**
- **Tráfico de red.**
- **Modificación de imagen y contenedor.**
- **Cambios de privilegios.**
- **Creación y modificación del programador de tareas.**
- **Modificación en la jerarquía de ficheros.**

3.3.4 Logs del sistema

Las capturas de eventos por logs en las auditorías de seguridad, se deben realizar con una cierta periodicidad, teniendo en cuenta los históricos de logs recolectados con anterioridad y cotejar los cambios significativos entre dichos ficheros, así como realizar un análisis de porqué se han realizado cambios. Si se detecta un consumo excesivo en los recursos del sistema que sea un cambio significativo en relación de auditorías pasadas, por ejemplo, esto puede indicar un acceso no autorizado de un atacante para el uso del poder de cómputo del sistema.

Si la organización posee algún recolector de eventos centralizado (syslog), de gestión de eventos de seguridad o SIEM (en inglés, Security Information and Event Management), la transmisión de estos registros de seguridad podrá ayudar a la detección de anomalías del clúster, lo más cercano posible a los tiempos de ejecución.

Siempre que se realice un traslado de información, o de registros de log a un sistema centralizado, la conexión ha de realizarse de forma segura mediante certificados, como TLS en sus versiones 1.2 y 1.3, para poder garantizar la integridad de la información trasladada de un punto a otro.

Cuando se utiliza un servidor de registros externo a Kubernetes, se debe configurar el reenviador de registros con acceso de sólo anexar al almacenamiento externo. Esto protege los registros almacenados externamente para que no se eliminen o sobrescriban desde dentro del clúster.

3.3.5 Registros de auditoría en Kubernetes

En el clúster de Kubernetes, siempre que se hace una solicitud por parte del "Front-end" kubi-apiserver, se produce un registro del evento, ya sea por una solicitud de un usuario, una aplicación o por el plano de control. Cuando un evento se registra, la kube-apiserver busca la primera coincidencia en sus políticas de auditoría para realizar la clasificación del evento, no realizándose de forma predeterminada.

Los archivos de políticas de registro son almacenados en formato "yaml", y se utilizan para establecer las reglas y especificar qué grado de auditoría se debe registrar ante un evento que coincida con una política creada previamente.

```yaml
apiVersion: audit.k8s.io/v1 # Esto es obligatorio.
kind: Policy
# No generar eventos de auditoría para las peticiones en la etapa RequestReceived.
omitStages:
  - "RequestReceived"
rules:
  # Registrar los cambios del pod al nivel RequestResponse
  - level: RequestResponse
    resources:
    - group: ""
      # Los recursos "pods" no hacen coincidir las peticiones a cualquier sub-recurso de pods,
      # lo que es consistente con la regla RBAC.
      resources: ["pods"]
  # Registrar "pods/log", "pods/status" al nivel Metadata
  - level: Metadata
    resources:
    - group: ""
      resources: ["pods/log", "pods/status"]

  # No registrar peticiones al configmap denominado "controller-leader"
  - level: None
    resources:
    - group: ""
      resources: ["configmaps"]
      resourceNames: ["controller-leader"]
```

Ilustración 6 - Fragmento de política de registro de Kubernetes.

NOTA: Si desea extender la información sobre este punto, puede visitar el siguiente enlace.

https://Kubernetes.io/es/docs/tasks/debug-application-cluster/audit/

Para que una política o regla se considere válida, debe poseer alguno de los niveles de auditoría que gestiona el clúster de Kubernetes.

- **NONE.** No se registran los eventos que ejecutan la regla de registro.
- **METADATA.** Se registran los metadatos de la solicitud, como la marca de tiempo, pero no la petición en sí, ni la respuesta a esta.
- **RESQUEST.** Se registran los metadatos y la petición que dispara la ejecución del evento, pero no es registrada la respuesta. Esto no aplica para las peticiones que no son de recurso.
- **RESQUESTRESPONSE.** Se registra los metadatos, la petición y la respuesta. Esto no aplica para las peticiones que no son de recurso.

El nivel máximo de registros de auditoría es "RESQUESTRESPONSE", brindando el máximo de información disponible en caso de que ocurra un evento que tenga que ser monitorizado.

Los registros que poseen el nivel máximo de "RESQUESTRESPONSE" pueden incurrir en la captura de objetos tipo "secretos" donde puede ir almacenada información sensible, tales como contraseñas, en una codificación de base64. Además, genera una considerable cantidad de registros especialmente en clústeres que estén en producción, pudiendo provocar incidentes por falta de espacio.

Se deberá establecer un estudio, para una posterior implementación, de políticas en los registros de auditoría del clúster de Kubernetes de la organización, oscilando las políticas con los diferentes niveles de auditoría y haciendo especial seguimiento en los procesos o llamadas que sean críticas para el correcto funcionamiento de la aplicación.

Kube-apiserver, posee una serie de funcionalidades y parámetros de configuración para el manejo, almacenamiento y rotación de los logs o registros de auditoría generados por los eventos de las políticas de seguridad de Kubernetes. "Webhook" es un "backend" y un "endpoint" de http, al que Kubernetes invocará en el momento de ejecución de un evento, configurándose para enviar a una API de http externa los registros o información deseada.

NOTA: Si desea extender la información sobre este punto, puede visitar el siguiente enlace.

https://Kubernetes.io/docs/tasks/debug/debug-cluster/audit/

Fichero	Parámetro	Valor	Observaciones
kube-apiserver.yaml	audit-log-path	[ruta]	Ubicación de registros auditoría
kube-apiserver.yaml	audit-log-maxage	[Valor en días]	Rotación de registros de auditoría
kube-apiserver.yaml	audit-log-maxbackup	[Valor en días]	Conservación de registros de auditoría
kube-apiserver.yaml	audit-log-maxsize	[Valor en MegaBytes]	Tamaño máximo de registros de auditoría

Tabla 12 – Configuración de logs.

3.3.6 Registro de contenedores y nodos trabajadores

Kubelet realiza registros de auditoría en cada uno de los nodos de forma individualizada dentro de cada contenedor, gestionando los registros, almacenándolos y rotándolos según las configuraciones de las políticas.

Kubelet, es una herramienta de consola de comandos gestionada mediante el comando kubectl.

Comando	Parámetro	POD	Observaciones
kubectl	--namespace [nombre]	logs [POD]	Visualiza registros del contenedor

Tabla 13 – Comando kubectl.

3.3.7 Auditoría por SECCOMP

Todos los tipos de registros mencionados anteriormente pueden ser completados realizando registros de auditoría a las llamadas que se producen al kernel. Para tal fin se puede usar el aplicativo de "seccomp", pudiendo auditar así las llamadas de los contenedores en el ecosistema de Kubernetes.

Seccomp se encuentra deshabilitada de forma predeterminada, habilitándola y configurándola es capaz de limitar las llamadas al sistema por los contenedores, reduciendo la superficie de ataque de dicho sistema. Otra funcionalidad es la de grabar o registrar todas las llamadas realizadas al kernel.

Fichero	Parámetro	Valor	Observaciones
Kubelet-config.yaml	SeccompDefault	true	Activación seccomp

Tabla 14 – Comando kubectl.

Las configuraciones de seccomp se realizan por perfiles, permitiendo, denegando o registrando las llamadas. Deberá configurar los perfiles de seccomp, con los parámetros que mejor se adapten a las necesidades de su organización.

Fichero	Parámetro	Valor	Observaciones
/var/lib/kubelet/seccomp/[perfil]	[Configuración seccomp para un pod]	true	Activación seccomp y configuración

Tabla 15 – Configuración Seccomp.

Con la ayuda de seccomp, se puede identificar qué llamadas al sistema son necesarias para las operaciones normalizadas del clúster, y así utilizar patrones de operaciones de los pods, permitiendo identificar cualquier comportamiento anómalo de patrones e identificando actividades maliciosas.

3.3.8 Detección de amenazas

El uso de recolectores de logs, como puede ser el uso de journald, de syslog y rsyslog, agilizan la recopilación de información del sistema y ayudan a encontrar patrones para la detección de amenazas al sistema.

Una monitorización continua de los logs y de los recursos del sistema en los tiempos de ejecución, es una herramienta muy poderosa para comprender el funcionamiento de Kubernetes y detectar anomalías en los procesos, y así poder determinar si se ha atacado o si están atacando al sistema agentes maliciosos. Para tales cometidos, el uso de herramientas externas como puede ser un SIEM, puede resultar de gran ayuda para una organización, no solamente en cuestión de seguridad, sino también de operatividad del sistema, pudiéndose visualizar en tiempo real la gestión de los recursos por parte de Kubernetes.

4. Lista de comprobación

Criticidad	Descripción
Alta	El equipo host que aloja el aplicativo de Kubernetes debe tener su sistema actualizado, aplicados todos los parches de seguridad publicados por el fabricante del producto, y encontrarse en vigor su ciclo de vida.
Alta	Kubernetes debe tener instaladas las últimas actualizaciones y parches de seguridad publicados por el fabricante.
Alta	Las conexiones entre nodos y hacia los nodos se encuentran cifradas por certificados.
Alta	Las comunicaciones se establecen cifradas por protocolos TLS en sus versiones 1.2 o 1.3.
Alta	Los ficheros de configuración del “plano de control” solo podrán ser modificados por usuarios administradores.
Alta	Los certificados y sus llaves generadoras en Kubernetes son propiedad del usuario “root” y del grupo “root”.
Alta	Solo pueden acceder a los certificados y sus llaves generadoras de Kubernetes los usuarios administradores.
Alta	Los ficheros de configuración del “nodos trabajadores” solo podrán ser modificados por usuarios administradores.

4. Lista de comprobación

Criticidad	Descripción
Alta	El ingreso al entorno de Kubernetes solo se establece por usuarios autenticados.
Alta	Se encuentran configurados registros de auditoría en Kubernetes.
Alta	Los contenedores poseen un control de acceso basado en roles, y definidos los “namespaces” de los contenedores.
Media	Los ficheros de configuración del “plano de control”, pertenecen al usuario “root” y al grupo “root”.
Media	Los ficheros de configuración de los “nodos trabajadores”, deberán pertenecer al usuario “root” y al grupo “root”.
Media	La información sensible denominada “secrets” se encuentra cifrada.
Media	Las cuentas de servicio de los “pods” (ServiceAccount) están configuradas con permisos específicos para la realización de sus cometidos.
Media	Las políticas de seguridad de pop (PSP) se encuentran configuradas y activas.
Media	Las cuentas de servicio se encuentran configuradas para que no se les otorgue un token de autenticación de forma automática.
Media	Se encuentran configurados los registros de eventos en Kubernetes según niveles de auditoría.
Media	Se ha configurado una política de almacenaje de logs de auditoría.
Media	Se encuentran configurado en Kubernetes los registros a las llamadas del kernel.

Criticidad	Descripción
Baja	Los registros de auditoría se almacenan en un medio externo.
Baja	Los recursos del clúster de Kubernetes se configuran con una segmentación por cuotas.
Baja	En caso de que los contenedores no necesiten modificación, se encuentran como “inmutables”.

5. Decálogo de recomendaciones

A continuación, se indican diez recomendaciones de seguridad en el uso de Kubernetes.

Decálogo de recomendaciones para Kubernetes

1. Se recomienda **utilizar siempre la versión estable más actual** y con las últimas actualizaciones recomendados por el fabricante, eliminando así vectores de ataque con remediaciones aplicadas por el fabricante.

2. Se recomienda el **uso de algoritmos criptográficos seguros**. El uso de un algoritmo de cifrado robusto y confiable elimina la posibilidad de la interceptación de los mensajes del sistema.

3. Se recomienda la **utilización de fuentes oficiales y/o confiables** para realizar despliegues de imágenes de contenedores. El entorno de kubernetes administra contenedores que realizan ciertas funciones para una organización. Dichos contenedores pueden ser creados por la organización o descargados desde fuentes oficiales y confiables para su implementación en el sistema.

4. Se recomienda **utilizar siempre protocolos seguros (TLS),** para asegurar las comunicaciones de extremo a extremo, y evitar así la interceptación de la información.

5. Se recomienda el **uso de políticas de contraseñas robustas y complejas,** poseyendo una longitud mínima de caracteres, además del uso de mayúsculas, minúsculas símbolos y números, conjuntamente a un periodo de validez de la misma.

6. Se recomienda el **uso de imágenes mínimas y actualizadas.** Usando imágenes mínimas, se consigue un menor consumo de recursos y un menor vector de ataque.

7. Se recomienda **evaluar los privilegios utilizados por los contenedores.** Atendiendo al principio de mínima funcionalidad y mínimo privilegio.

8. Se recomienda **realizar una segregación física de la red.** Optimización de los recursos y segmentación de los elementos de la red, acotando los accesos a la información y evitando la propagación de incidentes de seguridad.

9. Se recomienda **realizar una segregación de roles y permisos en los usuarios.** Se definen para cada usuario las necesidades de ejecución dentro de kubernetes y la necesidad de acceso a la información que se gestiona.

10. Se recomienda **documentar la plataforma de kubernetes.** La infraestructura de kubernetes puede llegar a ser muy extensa, albergando varios nodos de control y trabajadores. Mantener el sistema documentado ayuda a identificar rápidamente los contenedores en desuso o con poca carga de actividad, posibilitando la optimización de recursos y consiguiendo una mejor gestión del sistema. Es aconsejable actualizar dicha documentación con cada cambio relevante realizado en el sistema.

CCN centro criptológico nacional

CCN-CERT centro criptológico nacional

6. Glosario

Término	Descripción
K8s	Abreviatura de Kubernetes.
Pod	Conjunto de uno o más contenedores, y unas especificaciones de cómo ejecutar dichos contenedores.
Cloud controller manager (CSP)	Componente opcional utilizado para implementaciones basadas en la nube. El controlador de la nube interactúa con el proveedor de servicios de la nube (CSP) para administrar los balanceadores de carga y las redes virtuales para el clúster.
ETCD	El almacén de respaldo persistente donde se guarda toda la información sobre el estado del clúster.
SELinux	Módulo de seguridad para el kernel Linux.
AppArmor	Módulo de seguridad para el kernel Linux.
RBAC	Control de acceso basado en roles.
Seccomp	Es una facilidad del kernel Linux que permite limitar las llamadas al sistema que un proceso puede realizar.
API	Interfaz de programación de aplicaciones.
CSP	Proveedor de servicios certificados.
CA	Entidad Certificadora.
TLS	Protocolo criptográfico empleado en redes. Seguridad en la Capa de Transporte.
HTTPS	Protocolo seguro de transferencia de hipertexto.
Docker o contenedor	Proyecto de código abierto que automatiza el despliegue de aplicaciones.
on-premise	Instalación de software o hardware de forma local.
Plano de control	Conjunto de componentes que se ejecutan en un único nodo del clúster de Kubernetes controlando varios de sus aspectos.
Nodos trabajadores	Conjunto de máquinas que realizan las tareas solicitadas que asigna el plano de control.
Kube-apiserver	Se encarga de la interacción con los distintos usuarios del sistema de Kubernetes.

6. Glosario

Término	Descripción
Kube-sheduler	Ordena y planifican las prioridades en la gestión del clúster.
Kube-controller-manager	Revisa los procesos de Kubernetes.
Pod	Conjunto de uno o más contenedores implementados en un solo nodo. Es el objeto más pequeño y simple de Kubernetes.
Kubectl	Interfaz de línea de comandos donde puede gestionar su clúster de Kubernetes.
Kubelet	Aplicación de cada nodo, que se comunica con el plano de control.
Kube-proxy	Proxy de red de Kubernetes.
Yaml	Formato de serialización de datos legible por humanos inspirado en diferentes lenguajes de programación.
NodePort	Acceso a través de un puerto a los nodos.
Namespace o espacios de nombres	Clústeres de tipo virtual que son respaldados por un mismo clúster físico.
Clúster	Sistemas distribuidos de cómputo unidos entre sí normalmente por una red de alta velocidad y que se comportan como si fuesen un único servidor.
Secrets o secretos	Objetos de tipo Secret en Kubernetes que permiten almacenar y administrar información confidencial, como contraseñas, tokens OAuth y llaves SSH.
hipervisor	Monitor de máquinas virtuales.
log	Es la grabación secuencial en un archivo o en una base de datos de todos los acontecimientos que afectan a un proceso particular.
Front-end	Interfaz de usuario.
Back-end	Servidor o motor de servicios.

CCN
centro criptológico nacional

CCN-CERT
centro criptológico nacional

www.ccn.cni.es

www.ccn-cert.cni.es

oc.ccn.cni.es