Introducción a la seguridad en cloud computing

PID_00266615

Joan Caparrós Ramírez
Lorenzo Cubero Luque
Jordi Guijarro Olivares

Tiempo mínimo de dedicación recomendado: 4 horas

Universitat Oberta de Catalunya

Joan Caparrós Ramírez

Ingeniero superior en Informática por la UAB, máster en Seguridad de las TIC por la UOC y máster en Diseño y programación de aplicaciones móviles por La Salle. Desarrollador Fullstack web & mobile especializado en entornos de alto rendimiento y seguridad. Actualmente desarrolla funciones de técnico líder de proyectos en el Área de Cálculo y Aplicaciones en el Consorci de Serveis Universitaris de Catalunya (CSUC).

https://www.linkedin.com/in/joan-caparros

Lorenzo Cubero Luque

Ingeniero superior en Informática por la UPC y máster en Gestión de las tecnologías de la información por La Salle. Ha desarrollado proyectos de implantación de metodologías ágiles en equipos DevOps. Actualmente lidera el equipo responsable de los servicios TI de una multinacional suiza dedicada al marketing digital, Netcentric AG.

@lj_cubero

https://www.linkedin.com/in/lorenzocubero

Jordi Guijarro Olivares

Ingeniero en Informática por la UOC y máster en Gestión de tecnologías de la información por la URL. Coordinador de operaciones y ciberseguridad tecnológica en el CSUC, donde lidera las actividades técnicas en los ámbitos de servicios *cloud* y ciberseguridad del consorcio. También coordina el equipo de ciberseguridad CSUC-CSIRT de la red académica y de investigación catalana (Anella Científica), y colabora en proyectos a nivel europeo en grupos de trabajo como SIG-CISS y la TF-CSIRT de la red académica y de investigación europea GéANT.

@jordiguijarro

https://es.linkedin.com/in/jordiguijarro

La revisión de este recurso de aprendizaje UOC ha sido coordinada por el profesor: Josep Jorba Esteve (2019)

Segunda edición: septiembre 2019

Av. Tibidabo, 39-43, 08035 Barcelona
Realización editorial: FUOC

Índice

Introducción

El modelo de servicio TI basado en *cloud* y el constante aumento de las amenazas informáticas implican un cambio en el concepto de seguridad informática en las organizaciones, especialmente en la importancia estratégica que tiene la seguridad en *cloud computing*. Estamos involucrados en una transformación de paradigmas en las tecnologías de la información y la comunicación (TIC) en la que la transformación digital basada en modelos *cloud* es la tónica general. El crecimiento del *cloud* incorpora de manera nativa a la web, y este crecimiento deriva en grandes retos para la seguridad informática.

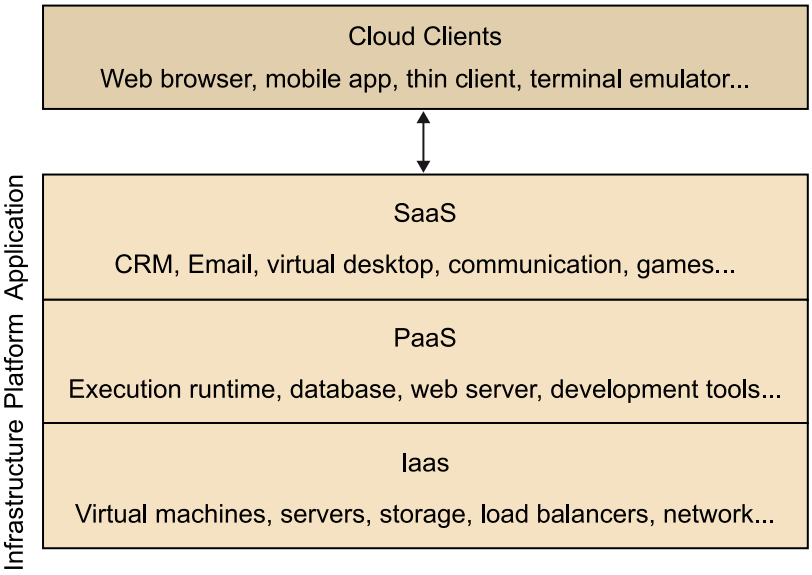
El cibercrimen es desarrollado por organizaciones internacionales que tienen como objetivo perjudicar a empresas y entidades gubernamentales. Por este motivo, las organizaciones necesitan tratar la seguridad en el *cloud* de una manera estructurada. En el momento en el que una organización decide confiar sus datos sensibles, como por ejemplo la información de sus clientes, ha de controlar en todo momento:

- La localización de la información.
- El proveedor y modelo del servicio *cloud*.
- Las niveles de servicio respecto a la integridad y la disponibilidad de los datos.

Además, hay que tener en cuenta el plan de continuidad de negocio, es decir, si este proveedor presenta vulnerabilidades en sus sistemas de seguridad, cuáles serían los potenciales riesgos en el negocio.

La gran mayoría de los proveedores de *cloud* aseguran que la seguridad, en última instancia, es responsabilidad del cliente. AWS, RackSpace, Google y otros solo se responsabilizan de sus centros de datos e infraestructura, su seguridad está certificada por las máximas autoridades en la materia. En particular, AWS lo llama «entorno de responsabilidad compartida». Como usuarios de diferentes servicios *cloud* somos responsables de los datos que alojamos en servidores y bases de datos de los proveedores de *cloud*. Recordemos entonces que la seguridad no solo es importante, sino que debemos trabajar en ella.

Figura 1. Modelos de servicio del *cloud computing*



Los proveedores de plataforma como servicio (PaaS) se encargan de la seguridad de la capa de infraestructura y plataforma, mientras que el cliente debe encargarse de la seguridad en la capa de aplicación.

En el caso de los proveedores infraestructura como servicio (IaaS), estos se encargan de la seguridad de la capa de infraestructura, mientras que el cliente debe encargarse de las otras dos.

En la capa de plataforma debemos usar todas las medidas necesarias para mantener nuestros servidores, bases de datos y copias de seguridad accesibles solo a personal autorizado; usando el principio de menor privilegio, otorgaremos a cada uno el mínimo privilegio necesario para realizar su trabajo.

Los analistas y programadores deben ocuparse de la capa de aplicación. No tiene mucho sentido blindar los datos y servidores si la aplicación permite accesos no legítimos a los datos o a funciones críticas de los servidores.

El *cloud computing* es una revolución en el mundo de las TIC. Trae en su desarrollo una atractiva perspectiva para las organizaciones, grandes o pequeñas, respecto a un mejor aprovechamiento de los recursos internos y a un modelo de gestión optimizado.

Este nuevo modelo ofrece a las organizaciones la posibilidad de tener un foco más claro en sus negocios a partir del momento en el que las tecnologías de información pasan a ser tratadas como un suministro más de sus cadenas productivas, y como todo nuevo concepto, el aspecto de la seguridad informática es siempre uno de los puntos más importantes que hemos de tener en cuenta, especialmente en momentos de intensa transformación.

En este capítulo dotaremos al alumno de la visión de los principales riesgos, como la fuga de información o las credenciales comprometidas y la suplantación en la autenticación; los principales patrones, y la mitigación proactiva de amenazas mediante los procedimientos operativos de seguridad para el control de proveedores.

También veremos la gestión de incidentes de seguridad en *cloud*: desde la política de seguridad de la información, hasta la respuesta a incidentes en entornos *cloud*.

Veremos en detalle el caso de uso de seguridad sobre el modelo de infraestructura como servicio pública basado en Amazon Web Services y DigitalOcean. En el apartado de Amazon Web Services veremos los servicios de seguridad que ofrece el proveedor y explicaremos en detalle cómo evitar ataques distribuidos de denegación de servicio. Mientras que en el caso de DigitalOcean haremos un repaso sobre prácticas de seguridad básica.

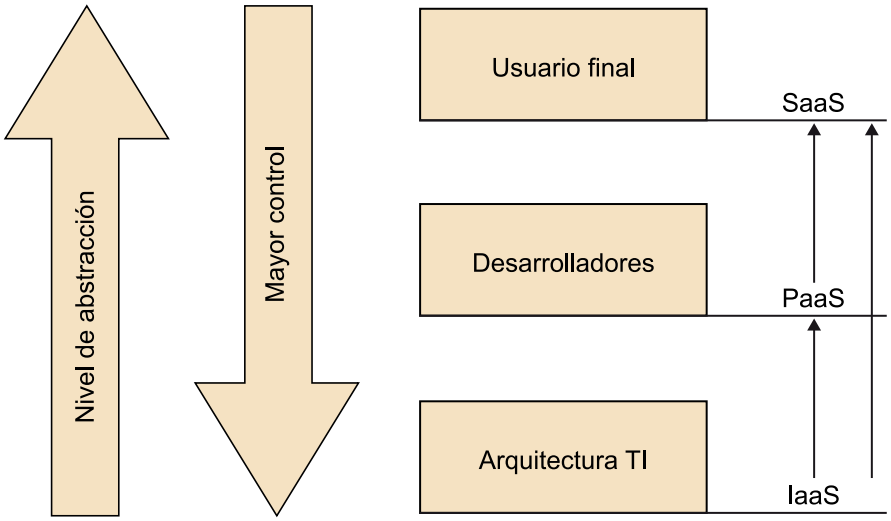
Analizaremos también el caso de uso sobre el modelo de plataforma como servicio privada basada en Docker: desde la seguridad del *host*, hasta módulos de computación segura. Al final de este apartado se proporciona al alumno un *checklist* de prácticas de seguridad en Docker.

Por último, haremos un repaso de las principales herramientas de seguridad: desde herramientas de gestión de acceso, hasta herramientas de gestión de vulnerabilidades de seguridad, pasando por los beneficios del uso de los estándares abiertos.

1. Principales riesgos, patrones y mitigación proactiva de amenazas

La naturaleza compartida en los catálogos de servicios en el campo de la computación en la nube introduce la aparición de nuevas brechas de seguridad que pueden amenazar los beneficios obtenidos en el cambio hacia tecnologías basadas en la nube. Organizaciones como la CSA (Cloud Security Alliance) advierten que es importante considerar que los servicios en la nube, por su naturaleza, permiten a los usuarios omitir las políticas de seguridad de toda la organización utilizando servicios externos, fenómeno que se conoce como «Shadow IT». De aquí la importancia de establecer nuevos controles para la detección, el control y la concienciación sobre los riesgos de esta nueva situación.

Figura 2. Modelos y áreas en el *cloud computing*



En términos generales, en función del tipo de servicio contratado, conforme va incrementándose el nivel de abstracción disminuye el control que el cliente tiene sobre la infraestructura. Del mismo modo, cuanto mayor control tiene la organización o el cliente sobre la infraestructura que proporciona el servicio, mayor nivel de seguridad y control puede aplicar sobre esta y, por tanto, sobre la información tratada.

1) Amenaza n.º 1: Fuga de información

Los entornos en la nube se enfrentan a muchas de las amenazas que ya encontramos en las redes corporativas tradicionales, pero estas se acentúan debido a la gran cantidad de datos almacenados en servidores en la nube, y los proveedores se convierten en un objetivo más atractivo. Si hablamos de datos/información, es muy importante tener en cuenta que la gravedad del daño depende en gran medida de la sensibilidad de los datos expuestos. La información

financiera expuesta tiende a salir en los titulares de la prensa generalista, pero las infracciones que involucran información de salud, secretos comerciales y propiedad intelectual son las más devastadoras.

Cuando se produce una fuga de datos, las empresas pueden incurrir en multas o pueden enfrentarse a demandas o incluso a cargos criminales. También se han de considerar las investigaciones de la infracción y las propias notificaciones hacia los clientes, que por daños de imagen pueden añadir costes muy significativos. Efectos indirectos, como daños a la marca y la pérdida de negocio, pueden afectar a las organizaciones durante varios años.

Al final, los proveedores de nube implementan controles de seguridad para proteger sus entornos, pero cabe recordar que en última instancia las organizaciones son responsables de proteger sus propios datos dentro de la organización y también en la nube. Es más que recomendable que las organizaciones utilicen mecanismos como la autenticación multifactor y el cifrado de datos para protegerse contra las fugas de información.

2) Amenaza n.º 2: Credenciales comprometidas y suplantación en la autenticación

Las brechas de datos y otros ataques frecuentemente resultan de un sistema de autenticación «pobre», como permitir contraseñas débiles y una mala administración de claves o certificados. A menudo, las organizaciones luchan con la gestión de identidades en tanto que tratan de asignar permisos adecuados a la función de trabajo del usuario. Pero en algunas ocasiones la falta de gestión correcta de las bajas provoca que no estas no sean efectivas en el momento en el que el puesto de trabajo cambia o el usuario abandona la organización.

Los sistemas de autenticación multifactor, como las contraseñas de un solo uso, la autenticación mediante dispositivos móviles y las tarjetas inteligentes protegen los servicios en la nube porque dificultan que los atacantes inicien sesión con contraseñas robadas. Existen multitud de ejemplos de organizaciones y servicios de internet muy conocidos que han expuesto millones de registros de clientes o usuarios como resultado de credenciales de usuario robadas. En la mayoría de los casos, asociado a fallos en el despliegue de un proceso de autenticación de tipo multifactor, así que una vez que los atacantes obtienen las credenciales, solo cabe esperar que el proceso que se encarga de la respuesta a incidentes sea efectivo.

Muchos desarrolladores cometen el error de incrustar credenciales y claves criptográficas en el código fuente y dejarlas en repositorios orientados al público, como GitHub. Las claves necesitan estar adecuadamente protegidas, y una infraestructura de clave pública bien asegurada es necesaria. También necesitan ser rotadas periódicamente para que a los atacantes les sea más difícil utilizar claves que han obtenido sin autorización.

Las organizaciones que planean federar la identidad con un proveedor de la nube necesitan entender las medidas de seguridad que el proveedor utiliza para proteger la plataforma de identidad. Centralizar la identidad en un solo repositorio tiene sus riesgos, y en cada caso se ha de valorar muy bien la protección y las salvaguardas asociadas.

3) Amenaza n.º 3: Interfaces y API *hackeadas*

Prácticamente todos los servicios y las aplicaciones en la nube ahora ofrecen API para facilitar tareas de automatización e interoperabilidad. Los equipos de TI utilizan interfaces y API para gestionar e interactuar con servicios en la nube, incluidos aquellos que ofrecen aprovisionamiento en autoservicio, gestión remota, orquestación, monitorización y supervisión en la nube.

La seguridad y la disponibilidad de los servicios en la nube, desde la autenticación y el control de acceso hasta el cifrado y la monitorización de actividades, dependen de la seguridad de la API. El riesgo aumenta con terceros que dependen de las API y se basan en estas interfaces, ya que las organizaciones pueden necesitar exponer más servicios y credenciales. Las interfaces débiles y las API exponen a las organizaciones a cuestiones de seguridad relacionadas con la confidencialidad, la integridad, la disponibilidad y la rendición de cuentas.

Las API y las interfaces tienden a ser la parte más expuesta de un sistema porque normalmente son accesibles desde internet. Se recomiendan controles adecuados como la primera línea de defensa y detección. Las aplicaciones y los sistemas de modelado de amenazas, incluidos los flujos de datos y la arquitectura/diseño, se convierten en partes importantes del ciclo de vida del desarrollo. También se recomiendan revisiones de código enfocadas en la seguridad y rigurosas pruebas de penetración.

4) Amenaza n.º 4: Vulnerabilidades

Las vulnerabilidades del sistema, o *bugs* explotables en los programas, no son nuevos, pero se han convertido en un problema mayor con la acentuación de los sistemas de información «multitenant» en el modelo de *cloud computing*. Las organizaciones comparten memoria, bases de datos y otros recursos en estrecha proximidad, lo que crea nuevas superficies de ataque.

Afortunadamente, los ataques a las vulnerabilidades del sistema pueden ser mitigados con «procesos de TI básicos». Las mejores prácticas incluyen la exploración regular de vulnerabilidades, la administración rápida de parches y un rápido seguimiento de las amenazas informadas.

Es importante considerar que los costes de mitigar las vulnerabilidades del sistema «son relativamente pequeños en comparación con otros gastos de TI». El coste de poner los procesos de TI en el lugar que les correspondes con el objetivo de controlar, detectar y reparar vulnerabilidades es pequeño en compara-

ción con el daño potencial. Las organizaciones necesitan parchear lo más rápido posible, preferiblemente como parte de un proceso automatizado y recurrente. Los procesos de control de cambios que tratan los parches de emergencia aseguran que las actividades relacionadas con el mantenimiento del software estén debidamente documentadas y revisadas por los equipos técnicos.

5) Amenaza n.º 5: Secuestro de cuentas

La pesca electrónica (*phishing*), el fraude y las explotaciones de código siguen teniendo éxito, y los servicios en la nube agregan una nueva dimensión a la amenaza debido a que los atacantes pueden interceptar actividad, manipular transacciones y modificar datos. A esto se debe añadir que los atacantes también pueden utilizar la aplicación en la nube para lanzar otros ataques de manera encadenada.

Las estrategias más comunes en la defensa en profundidad pueden contener los daños ocasionados por una infracción. Las organizaciones deben prohibir el intercambio de credenciales de cuenta entre usuarios y servicios, así como habilitar sistemas de autenticación multifactor cuando estén disponibles. Las cuentas, incluso las de servicio, deben ser monitorizadas para que cada transacción pueda ser rastreada e identificar un usuario unipersonal si fuere necesario. Una parte de la estrategia es proteger las credenciales de las cuentas de usuario para evitar que estas sean robadas.

6) Amenaza n.º 6: Intrusos maliciosos

Esta amenaza tiene muchas caras: un empleado o un antiguo empleado, un administrador de sistemas, un cliente o incluso un *partner*. La actividad maliciosa puede querer llevar a cabo el robo de datos o un acto de venganza. En un escenario en la nube, un actor privilegiado puede destruir infraestructuras enteras o manipular datos. Los sistemas que dependen únicamente de un proveedor de servicios en la nube, como podrían ser los de cifrado, corren sin duda un mayor riesgo.

Se recomienda que sean las propias organizaciones las que controlen el proceso de cifrado y las claves, segregando las tareas y minimizando el acceso dado a los usuarios. Las actividades de registro, monitorización y auditoría de los propios administradores también son consideradas a día de hoy como críticas.

En entornos de *cloud computing*, es fácil interpretar erróneamente un intento de ataque por parte de un proceso rutinario como actividad «malintencionada» sobre información privilegiada. Un ejemplo de esto sería un administrador que copia accidentalmente una base de datos confidencial de clientes a un servidor accesible públicamente. La formación y la prevención para prevenir tales errores se vuelven aspectos más críticos en la nube, debido a una mayor exposición de los sistemas y servicios.

7) Amenaza n.º 7: El parásito APT

La CSA llama acertadamente a las amenazas persistentes avanzadas (APT) formas «parásitas» de ataque. Las APT se infiltran en los sistemas para establecerse en un punto de apoyo, y luego exfolian furtivamente los datos y la propiedad intelectual durante un periodo prolongado de tiempo.

Las APT normalmente se mueven lateralmente a través de la red y se mezclan con el tráfico normal, por lo que son difíciles de detectar. Los principales proveedores de nube aplican técnicas avanzadas para evitar que las APT se infiltren en su infraestructura, pero los clientes deben ser tan diligentes en la detección de compromisos APT en las cuentas de la nube como lo harían en sus sistemas locales.

Entre los puntos de entrada comunes se incluyen la pesca electrónica, los ataques directos, unidades USB precargadas con *malware* y redes de terceros comprometidas. En particular, se recomienda entrenar a los usuarios para que reconozcan las técnicas de pesca electrónica o *phishing*.

Los programas de concienciación mantienen a los usuarios en alerta y lo hacen menos propensos a ser engañados para que una APT ingrese en la red. Por otra parte, los departamentos de TI deben mantenerse informados de los últimos ataques avanzados. Los controles de seguridad avanzados, la gestión de procesos, los planes de respuesta a incidentes y la capacitación del personal de TI conducen a mayores presupuestos de seguridad. Las organizaciones deben sopesar estos costes frente al posible daño económico infligido por los ataques exitosos de APT.

8) Amenaza n.º 8: Pérdida permanente de datos

A medida que la nube se ha madurado, los informes de pérdida permanente de datos debido al error del proveedor se han vuelto extremadamente raros. Sin embargo, se sabe que los *hackers* maliciosos eliminan de forma permanente los datos de la nube para dañar a las empresas, y los centros de datos en la nube son tan vulnerables a desastres naturales como cualquier instalación.

Los proveedores de nube recomiendan distribuir datos y aplicaciones a través de múltiples zonas para una mayor protección. Las medidas adecuadas de respaldo de datos son esenciales, así como la adhesión a las mejores prácticas en la continuidad del negocio y la recuperación de desastres. La copia de seguridad diaria de datos y el almacenamiento en otro lugar físico siguen siendo importantes con los entornos en la nube.

La carga de evitar la pérdida de datos no recae toda en el proveedor de servicios en la nube. Si un cliente cifra datos antes de subirlos a la nube, entonces ese cliente debe tener cuidado de proteger la clave de cifrado. Una vez que se pierde la clave, también se pierden los datos.

Las políticas de cumplimiento a menudo estipulan cuánto tiempo las organizaciones deben conservar los registros de auditoría y otros documentos. La pérdida de estos datos puede tener graves consecuencias regulatorias. Las nuevas normas de protección de datos de la UE también tratan la destrucción de datos y la corrupción de datos personales como infracciones de datos que requieren una notificación adecuada. Hay que conocer las reglas para evitar problemas.

9) Amenaza n.º 9: Inadecuada diligencia

Las organizaciones que abrazan la nube sin entender completamente el entorno y sus riesgos asociados pueden encontrarse con un «incremento de riesgos comerciales, financieros, técnicos, legales y de cumplimiento». El grado del riesgo depende de si la organización está intentando migrar a la nube o fusionarse (o trabajar) con otra compañía en la nube. Por ejemplo, las organizaciones que no examinan un contrato de servicio pueden no ser conscientes de la responsabilidad del proveedor en caso de pérdida de datos o algún tipo de incumplimiento.

Los problemas operacionales y de arquitectura surgen si el equipo de desarrollo de una empresa carece de familiaridad con las tecnologías en la nube, ya que las aplicaciones se despliegan en una nube en particular y no son habituales a día de hoy los modelos híbridos o de tipo *multicloud*.

10) Amenaza n.º 10: Abusos de los servicios en la nube

Los servicios en la nube pueden ser utilizados para apoyar actividades ilegales, como el uso de recursos de computación en la nube para romper una clave de cifrado para lanzar un ataque. Otros ejemplos incluyen el lanzamiento de ataques DDoS, el envío de correos electrónicos de *spam* y *phishing*, y el alojamiento de contenido malicioso.

Los proveedores deben reconocer estos tipos de abuso –como el análisis del tráfico para reconocer los ataques DDoS– y ofrecer herramientas para que los clientes puedan supervisar la salud de sus entornos *cloud*. Los clientes deben asegurarse de que los proveedores ofrezcan un mecanismo para notificar el abuso. Aunque los clientes no pueden ser víctimas directas de acciones maliciosas, el abuso en el servicio en la nube puede resultar en problemas de disponibilidad de servicios y pérdida de datos.

11) Amenaza n.º 11: Ataques DoS

Los ataques DoS han existido durante años, pero han ganado prominencia gracias a la computación en la nube, ya que a menudo afectan de manera destacada sobre la disponibilidad. Los sistemas pueden ralentizarse o estar fuera de juego. «Experimentar un ataque de denegación de servicio es como estar atrapado en un atasco de tráfico de hora punta; hay una manera de llegar a su destino y no hay nada que pueda hacer al respecto, excepto sentarse y esperar».

Los ataques de DoS consumen grandes cantidades de procesamiento, una factura que el cliente puede finalmente tener que pagar. Aunque los ataques DDoS de gran volumen no son muy comunes, las organizaciones deben ser conscientes de los ataques asimétricos a nivel de aplicación de DoS, que se dirigen a vulnerabilidades de servidor web y otros componentes críticos como son las bases de datos.

La norma habitual es que los proveedores de nube tienden a estar mejor preparados para manejar ataques de DoS que sus clientes. La clave es tener un plan para mitigar el ataque antes de que ocurra, por lo que los administradores tienen acceso a esos recursos cuando los necesitan.

12) Amenaza n.º 12: Tecnología compartida, peligros compartidos

Las vulnerabilidades en la tecnología compartida suponen una amenaza significativa para la computación en la nube. Los proveedores de servicios en la nube comparten infraestructura, plataformas y aplicaciones, y si surge una vulnerabilidad en cualquiera de estas capas, afecta a todos. «Una sola vulnerabilidad o mala configuración puede llevar a un compromiso a través de la nube de un proveedor entero».

Si un componente se ve comprometido –digamos, un hipervisor, un componente de plataforma compartida o una aplicación–, expone todo el entorno a un posible compromiso y violación.

En definitiva, es más que recomendable una **estrategia de defensa en profundidad**, incluyendo la **autenticación multifactor** en todos los *hosts*, **sistemas de detección de intrusos** tanto a nivel de *host* como en red, aplicando el concepto de **privilegios mínimos**, **segmentación de red** y **parche de recursos compartidos**.

1.1. Control de proveedores. Procedimientos operativos de seguridad

Algunas operaciones del servicio deben ser realizadas de manera conjunta por ambas partes, contratada y contratante, debiendo establecerse los roles, las responsabilidades (capacidad de autorizar y obligación de rendir cuentas) y los protocolos adecuados para llevarlas a cabo.

Cabe destacar las siguientes actividades, sin que sean las únicas que procedimentar:

- Mantenimiento y gestión de cambios.
- Gestión de incidentes.
- Continuidad - recuperación de desastres.
- Gestión del personal.
- Configuración de seguridad.
- Recuperación de datos de copias de seguridad.

1.1.1. Seguimiento del servicio

En los servicios prestados por terceros a la organización, tan importante es el acuerdo contractual con el proveedor de servicios como el seguimiento que se debe realizar sobre el servicio prestado. Para poder tener el control de los servicios, y por tanto también poder exigirle al proveedor el cumplimiento de cualesquiera medidas de seguridad aplicables, es necesaria una monitorización de estos.

- La medición del cumplimiento del servicio y el procedimiento para restaurar las desviaciones estipuladas contractualmente.

- El proceso de coordinación para el mantenimiento de los sistemas implicados.

- El proceso de coordinación ante incidentes o desastres.

En cuanto al primer aspecto y dadas las características de la prestación de servicios en la nube, en ocasiones con aspectos relevantes fuera del control de la organización cliente y también dada su forma de pago, en función del uso, es muy importante reflejar de un modo claro los términos del cumplimiento. Es necesario identificar en el contrato los derechos de la organización cliente para poder monitorizar el funcionamiento del servicio y de este modo poder comprobar el cumplimiento de las medidas de seguridad, los controles y las políticas que garantizan la integridad, confidencialidad y disponibilidad de los datos, y del mismo modo poder realizar la comprobación de que el nivel de prestación es el pactado. La definición y el control de los SLA son vitales para poder garantizar el cumplimiento de lo estipulado en el contrato.

La organización debe monitorizar de manera independiente el cumplimiento de los términos establecidos en el contrato, bien a través de controles técnicos propios o bien a través de la revisión y aprobación periódica de los informes de servicio proporcionados por el proveedor de servicios *cloud* (CSP).

Es necesario ejecutar esta monitorización sobre al menos los siguientes controles de seguridad y servicio con independencia de la categoría del sistema:

- Niveles de calidad, disponibilidad y capacidad del servicio ofrecido, incluyendo el cumplimiento de las obligaciones de servicio acordadas y la respuesta ofrecida por el proveedor ante desviaciones significativas.

- Gestión de incidentes de seguridad, incluyendo toda la información necesaria para determinar orígenes, objetivos, riesgos... asociados a cualquier incidente relevante.

- Controles de acceso a los servicios, incluyendo listado actualizado de usuarios autorizados para utilizar los servicios disponibles, y los privilegios asociados en cada caso.

- Cumplimiento normativo y legislativo entre el prestador y el cliente de los servicios, incluyendo los aspectos de cumplimiento de aplicación sobre el prestador que correspondan en cada caso, como auditorías LOPD, ISO, financieras...

- Situación actualizada de las medidas de protección de la información establecidas por el proveedor, incluyendo aspectos de seguridad física, protección contra software malicioso, seguridad del personal, copias de seguridad...

- Mecanismos de comprobación regular de los controles de seguridad por parte del proveedor y resultados de dichas comprobaciones.

Toda la información de los controles anteriores proporcionada periódicamente por el proveedor de servicios debe incluir de obligatoriamente cualesquiera anomalías o desviaciones significativas producidas durante el periodo, así como las acciones ejecutadas en cada caso como respuesta a estas situaciones susceptibles de introducir riesgos en la organización. Adicionalmente, debemos solicitar al proveedor de servicios la información de auditoría y no conformidades de aplicación en cada caso para poder verificar que las medidas de seguridad tomadas por este son las correctas y oportunas para solventar desviaciones halladas durante el proceso de auditoría.

1.1.2. Gestión de cambios

Otro aspecto que se debe tratar en la gestión diaria del servicio es el referente a la gestión y coordinación del mantenimiento de los sistemas. En este sentido, se deberá establecer contractualmente de acuerdo con los requisitos mínimos de normativas como el Esquema Nacional de Seguridad (ENS) la obligación de mantener actualizados los sistemas para garantizar su correcto funcionamiento, así como eliminar las posibles vulnerabilidades que pueden afectar a los sistemas.

Deberá definirse un procedimiento de coordinación en el mantenimiento de sistemas entre ambas partes para prevenir paradas o errores en la prestación del servicio; este procedimiento estará en línea con el proceso de gestión de cambio e incluirá la notificación con suficiente antelación de la realización de mantenimientos por parte del proveedor, identificando los tiempos en los que puede interrumpirse el servicio. La notificación se realizará previa y posteriormente al mantenimiento, y tras este se pedirá al cliente conformidad del correcto funcionamiento del servicio.

Por otra parte, siempre que el mantenimiento o la actualización implique un cambio mayor o pueda suponer el funcionamiento incorrecto de los sistemas de la organización cliente, el proveedor habilitará previamente un entorno actualizado para que el cliente puede verificar el correcto funcionamiento de sus sistemas en preproducción. Además, el proveedor deberá informar periódicamente de los mantenimientos y las actualizaciones realizados en los sistemas que albergan los sistemas del cliente.

1.1.3. Gestión de incidentes

De acuerdo con normativas vigentes y buenas prácticas, el proveedor deberá disponer de un procedimiento de gestión de incidentes. Se deberá informar a la organización cliente de:

- Procedimiento de notificación de incidentes.
- Tipología de incidentes incluidos en el servicio.
- Procedimientos específicos ante incidentes de seguridad.
- Tiempos de respuesta y resolución de incidentes.
- Mantenimiento y gestión del registro de incidentes.

Deberá definirse un procedimiento de coordinación ante incidentes que puedan afectar a los sistemas del cliente, procedimiento que deberá contemplar los flujos de información y las interacciones entre cliente y proveedor durante la gestión del incidente. A su vez, el proveedor deberá informar periódicamente de los incidentes que han afectado a los sistemas que soportan los sistemas del cliente.

En los niveles de servicio, en el caso de incidentes que afecten a la información o a los servicios del organismo contratante, el proveedor deberá facilitar toda la información forense necesaria para analizar el incidente y su gestión.

1.1.4. Respaldo y recuperación de datos

El proveedor deberá disponer de un procedimiento de copias de respaldo que garantice la restauración de la información como describe [mp.info.9]. El proveedor deberá informar a la organización cliente de:

- Política de copias de seguridad.

- Medidas de cifrado de información en respaldo.
- Procedimiento de solicitud de restauraciones de respaldo.
- Realización de pruebas de restauración.
- Alcance de los respaldos.
- Traslado de copias de seguridad (si procede).

1.1.5. Continuidad del servicio

De acuerdo con normativas como el ENS o normas como la ISO27001, los sistemas afectados deberán disponer de medidas para la continuidad del servicio. Si bien los niveles de disponibilidad, así como los tiempos de recuperación en la prestación de servicios, se encuentran recogidos contractualmente a través de los SLA, se deberá solicitar al proveedor evidencia de que existe un plan de continuidad de negocio que garantice la restauración de los servicios. El proveedor deberá informar a la organización cliente de:

- Existencia de plan de continuidad de negocio.
- Evidencia satisfactoria de la ejecución periódica de pruebas de continuidad.
- Análisis de impacto del servicio proporcionado.

Se considerarán los siguientes aspectos en función del nivel de disponibilidad requerido:

1) Se deberá requerir al proveedor evidencia de que existe un plan de continuidad de negocio cuyo alcance incluya los servicios objeto de la prestación.

2) Se exigirá constancia de que los tiempos de recuperación identificados en el análisis de impacto están alineados con los criterios definidos en los SLA en cuanto a tiempo de recuperación del servicio.

3) Deberá definirse un procedimiento de coordinación ante incidentes y desastres que puedan afectar a los sistemas del cliente, procedimiento que deberá contemplar los flujos de información y las interacciones entre cliente y proveedor durante la gestión tanto de incidentes como de desastres.

4) A su vez, el proveedor deberá informar periódicamente de los incidentes que han afectado a los sistemas que soportan los sistemas del cliente.

5) Se realizarán pruebas periódicas que involucren al organismo y a los diferentes proveedores y subproveedores para validar el correcto funcionamiento de los planes y el cumplimiento de los plazos y servicios mínimos previstos

1.1.6. Finalización

Salvo cuando la parte contratante disponga de una copia actualizada de su información, se deberán establecer los procedimientos para que esta recupere la información entregada al proveedor. En estos procedimientos se deben acotar formatos de datos y tiempos.

Salvo cuando la parte contratante no haya transferido información en claro al proveedor, se deberán establecer procedimientos para la eliminación de las copias en los equipos del proveedor. **Estos procedimientos deben incluir los mecanismos de borrado y las garantías de que han sido aplicados correctamente**. Los tiempos de destrucción de la información deberán tener en cuenta los requisitos legales de retención, si los hubiera.

Los procedimientos anteriores deben tener en el proveedor la parte correspondiente para prolongarlos a posibles terceros proveedores subcontratados.

1.1.7. Supervisión y auditoría

La organización cliente, como responsable última de los posibles riesgos que afecten tanto a la información como a los servicios prestados, deberá disponer de un determinado nivel de control sobre tales servicios. En este sentido y para garantizar el cumplimiento de las medidas de seguridad de aplicación en cada caso, la organización deberá evaluar la conveniencia de disponer del derecho de auditoría, con la profundidad correspondiente, sobre el proveedor de servicios o de solicitarle la siguiente documentación:

- Una declaración de aplicabilidad de las medidas que hay que aplicar.

- Una auditoría que verifique mediante evidencias el cumplimiento de las medidas de seguridad que sean de aplicación de acuerdo con el nivel del sistema.

- Auditorías de cumplimiento de normativa necesaria para satisfacer los requisitos de seguridad de la información del organismo contratante. Por ejemplo, LOPD, SAS70, PCI-DSS, etc.

El proveedor puede disponer de certificaciones o acreditaciones en materia de seguridad. Estas certificaciones pueden simplificar la auditoría completa del servicio prestado, en su condición de evidencias de cumplimiento que valorar por el equipo auditor. Por ejemplo:

- Auditorías recomendadas por ENISA para proveedores de servicios en la nube [ENISA-CCSL].

- Sistema de Gestión de la Seguridad de la Información (SGSI) [ISO/IEC 27001:2013].

- Sistema de Gestión de la Continuidad [ISO 22301:2012].

- Cloud Controls Matrix [CCM].

Figura 3. Control de medidas de seguridad sobre la base de la norma ISO 27001 y Cloud Control Matrix (CSA)

Medidas de seguridad		ISO 27000			CCM	
Org	**Marco organizativo**	Nivel	2005	2013	Nivel	v3
Org. 1	Política de seguridad	1	27001 4.2.1 5.1 27002 5.1.1 5.1.2 6.1.1 6.1.2 6.1.3 15.1.1	27001 4 5.2 5.3 27002 6.1.1 18.1.1	1	GRM-05 GRM-09

2. Gestión de riesgos e incidentes de seguridad en *cloud*

La manera de empezar que hemos considerado para esta sección no es en ningún caso la habitual, pero en el camino de la adopción del *cloud computing* y todo lo que implica es importante situar el punto en el que nos encontramos y el objetivo.

2.1. Gestión de riesgos: Cuestionario de evaluación

Las preguntas críticas que las organizaciones deben hacerse a sí mismas y a sus proveedores de la nube durante cada paso son las siguientes:

1) Asegurar eficazmente la gobernanza, el riesgo y los procesos de cumplimiento:

- ¿Qué normas de seguridad y privacidad de la información o qué regulaciones se aplican a la nube de dominio por el cliente?
- ¿El cliente tiene procesos de gobierno y de cumplimiento para el uso del servicio en la nube?
- De acuerdo con los requerimientos del cliente, ¿el proveedor tiene procesos apropiados de gobernanza y notificación?
- ¿Está claro qué controles legales y regulatorios se aplican a los servicios del proveedor?
- ¿Qué contemplan los acuerdos a nivel de servicio sobre la división de responsabilidades de seguridad entre proveedor y cliente?
- ¿Existe un riesgo relacionado con la ubicación de los datos?

2) Auditoría. Presentación de informes operacionales y procesos de negocio:

- ¿Existe alguna certificación demostrable por parte del proveedor? La información de auditoría se ajusta a una de las Normas para la auditoría de seguridad, como ISO 27001/27002.
- ¿Tiene el proveedor mecanismos para informar a los clientes tanto de aspectos rutinarios como de comportamiento excepcional relacionado con sus servicios?

- ¿Los controles de seguridad abarcan no solo los propios servicios en la nube, sino también las interfaces de gestión ofrecidas a los clientes?

- ¿Existe un proceso de notificación de incidentes y gestión de incidentes críticos?

3) Administrar a las personas, roles e identidades:

- ¿Ofrecen los servicios del proveedor un control de acceso de tipo granular?

- ¿Puede el proveedor proporcionar informes para supervisar el acceso de los usuarios?

- ¿Es posible integrar o federar sistemas de gestión de identidad de clientes con las instalaciones de gestión de identidad del proveedor?

4) Asegurar la correcta protección de datos e información:

- ¿Existe un catálogo de todos los activos de datos que se utilizarán o almacenarán en la nube?

- ¿Hay una descripción de los roles responsables?

- ¿Se ha tenido en cuenta el tratamiento de todas las formas de datos, en particular datos como vídeos e imágenes?

- ¿Existe una separación de entornos adecuada entre clientes?

- ¿Se han aplicado las medidas adecuadas de confidencialidad e integridad de los datos almacenados?

5) Políticas de privacidad:

- ¿Los servicios del proveedor cuentan con controles apropiados para manejar datos personales?

- ¿Existen restricciones de localización geográfica de los datos en el acuerdo de servicio?

- Si hay un incumplimiento sobre el tratamiento de los datos, el proveedor es responsable de informar de ello y resolverlo. ¿Están claras las prioridades y los plazos?

6) Evaluar la seguridad de las aplicaciones:

- Basado en el modelo que aplique (IaaS, PaaS, SaaS), ¿está claro quién tiene la responsabilidad de la seguridad de las aplicaciones (cliente o proveedor)?

- Si es el cliente, ¿tiene políticas y metodologías establecidas para asegurar los controles de seguridad apropiados para cada aplicación?

- Si es el proveedor, ¿el acuerdo de servicio en la nube hace que sus responsabilidades sean claras y que existan controles de seguridad específicos para ser aplicados a la aplicación?

- En cualquier caso, ¿la aplicación hace uso de técnicas de cifrado adecuadas para proteger los datos y las transacciones de los usuarios?

7) Asegurar la red:

- ¿Es posible la detección o inspección del tráfico de la red?

- ¿Qué capacidad tiene el proveedor para hacer frente a los ataques de denegación de servicio?

- ¿Tiene la red del proveedor la capacidad de detección y prevención de intrusiones?

- ¿Está el acceso a la red del cliente separado del acceso a la red del proveedor?

8) Controles de la infraestructura física:

- ¿Puede el proveedor de servicios en la nube demostrar los controles de seguridad apropiados aplicados a su infraestructura física e instalaciones?

- ¿Dispone el proveedor de servicios de instalaciones para garantizar la continuidad frente a amenazas o fallos en el equipamiento?

9) Términos del acuerdo sobre el servicio:

- ¿El acuerdo de servicio especifica las responsabilidades de seguridad del proveedor y del cliente?

- ¿El acuerdo de servicio requiere que todos los términos de seguridad también deben aplicar a cualquier proveedor de servicios utilizado por el proveedor?

- ¿El acuerdo de servicio tiene métricas para medir el desempeño y la eficacia de seguridad?

- ¿El contrato de servicio documenta explícitamente los procedimientos de notificación y manejo de incidentes de seguridad?

10) Requisitos de seguridad del proceso de salida o fin de contrato:

- ¿Existe un proceso documentado de salida como parte del servicio?

- ¿Está claro que todos los datos de los clientes del servicio en la nube se borran en el momento de finalizar el proceso de migración o salida?

- ¿Los datos de los clientes de servicios en la nube están protegidos contra pérdidas o incumplimientos durante la salida?

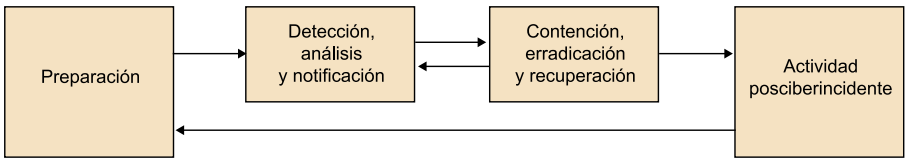
2.2. Respuesta a incidentes en entornos *cloud*

La labor del proveedor es básica en las actividades de respuesta ante la ocurrencia de algún incidente de seguridad. Esto incluye la verificación, el análisis del ataque, la contención, la recogida de evidencias, la aplicación de remedios y la restauración del servicio.

La colaboración entre los proveedores y los suscriptores para la detección y el reconocimiento de los incidentes es esencial para la seguridad y la privacidad en *cloud computing*, ya que la complejidad de los servicios puede dificultar la labor de la detección.

Se hace necesario entender y negociar los procedimientos de respuesta a incidentes antes de firmar un contrato de servicio. La localización de los datos es otro aspecto que puede impedir una investigación, por lo que es otro de los puntos que se deben negociar en los contratos.

Figura 4. Ciclo de vida de la respuesta a ciberincidentes



La solución que se negocie ha de tener la finalidad de mitigar el incidente en un tiempo que limite los daños y que mejore los tiempos de recuperación. Los equipos para la resolución deberían ser mixtos (proveedor y suscriptor), ya que la solución puede involucrar a alguna de las partes de forma individual o a ambas conjuntamente, y el incidente puede incluso afectar a otros suscriptores que comparten la infraestructura.

2.2.1. Eventos y ciberincidentes

Los ataques contra los sistemas de información son, cada día, no solo más numerosos y diversos, sino también más peligrosos o potencialmente dañinos. Aunque las acciones y medidas preventivas, adoptadas en función de los resultados obtenidos de los preceptivos análisis de riesgos a los que deben someterse todos los sistemas públicos, contribuyen sin lugar a dudas a reducir el número de ciberincidentes, la realidad nos muestra que, desafortunadamente, no todos pueden prevenirse.

Por tanto, se hace necesario disponer de la adecuada capacidad de respuesta a ciberincidentes, que detectando rápidamente ataques y amenazas minimice la pérdida o la destrucción de activos tecnológicos o de información, mitigue la explotación dañina de los puntos débiles de las infraestructuras y alcance la recuperación de los servicios a la mayor brevedad posible. Esta sección ofrece pautas para el manejo de ciberincidentes y para la determinación de la respuesta más adecuada a cada tipo, independientemente de la plataforma tecnológica subyacente, el hardware, los sistemas operativos o las aplicaciones.

Puesto que gestionar adecuadamente los ciberincidentes constituye una actividad compleja que contempla la adopción de métodos para recopilar y analizar datos y eventos, metodologías de seguimiento, procedimientos de tipificación de su peligrosidad y priorización, así como la determinación de canales de comunicación con otras unidades o entidades, propias o ajenas a la organización, la consecución de una capacidad de respuesta eficaz a ciberincidentes exige una planificación escrupulosa y la correspondiente asignación de recursos, adecuados y suficientes.

2.2.2. La respuesta a los ciberincidentes

Para las organizaciones, el beneficio más significativo de poseer una adecuada capacidad de respuesta a ciberincidentes es abordar su gestión de manera sistemática (es decir, siguiendo una metodología consistente y consolidada), lo que facilita la adopción de las medidas adecuadas.

Así, una correcta capacidad de respuesta a ciberincidentes ayuda a los equipos de seguridad responsables a minimizar la pérdida o exfiltración de información, o la interrupción de los servicios. Otro de sus beneficios es la posibilidad de utilizar la información obtenida durante la gestión del ciberincidente para preparar mejor la respuesta a incidentes de seguridad futuros y, en consecuencia, proporcionar una mayor y mejor protección a los sistemas.

2.2.3. Política de seguridad de la información y gestión de ciberincidentes

Las buenas prácticas en seguridad señalan la necesidad de una política de seguridad reconocida y conocida en la organización. **Normativas como el Esquema Nacional de Seguridad (ENS)** establecen los requisitos mínimos que debe contemplar toda política de seguridad, entre ellos, los incidentes de seguridad, para los que debe especificarse:

- La posición del **Equipo de Respuesta a Ciberincidentes** (ERI), sus competencias y autoridad, dentro de la estructura de la organización y la definición de los roles y responsabilidades de cada unidad.
- Normativa de seguridad.
- Definición de los ciberincidentes considerados a tenor del análisis de riesgos y los términos de referencia usados.
- Criterios para la comunicación de ciberincidentes y, en su caso, el intercambio de información, interna y externamente.
- Nivel de peligrosidad de los ciberincidentes.
- Procedimientos operativos de seguridad.
- Mecanismos para la notificación de informes de ciberincidentes.
- Formularios de notificación, comunicación e intercambio de información.
- Elementos del Plan de respuesta a ciberincidentes.

Los organismos del ámbito de aplicación del ENS deben poseer un Plan de respuesta a ciberincidentes que dé adecuada respuesta a sus requisititos específicos, atendiendo a la misión, el tamaño, la estructura y las funciones de la organización. El Plan debe, asimismo, determinar y asegurar que se dispone de los recursos humanos y materiales necesarios, y debe contar con el imprescindible apoyo por parte de la Dirección.

Una vez que la Dirección de la organización ha redactado y aprobado el Plan de respuesta a ciberincidentes, se iniciará su implantación. El Plan debería ser revisado, al menos, anualmente para asegurar que la organización está siguiendo adecuadamente la hoja de ruta para una mejora continua.

2.2.4. La gestión de los ciberincidentes

La gestión de ciberincidentes consta de varias fases:

1) La **fase inicial** contempla la creación y formación de un Equipo de Respuesta a Ciberincidentes (ERI) y la utilización de las herramientas y los recursos necesarios.

Durante esta fase de **preparación**, la organización, atendiendo a un previo análisis de riesgos, habrá identificado y desplegado un determinado conjunto de medidas de seguridad. La adecuada implantación de las medidas ayudará a detectar las posibles brechas de seguridad de los sistemas de información de la organización y a su análisis, en la fase de **detección**, **análisis y notificación**, lo que desencadenará los procesos de notificación pertinentes.

2) La organización, en la fase de **contención**, **erradicación y recuperación** del ciberincidente y atendiendo a su peligrosidad, deberá intentar, en primera instancia, mitigar su impacto, procediendo después a la eliminación de los sistemas afectados y tratando finalmente de recuperar el sistema al modo de funcionamiento normal. Durante esta fase será necesario, cíclicamente, persistir en el análisis de la amenaza, de cuyos resultados se desprenderán, paulatinamente, nuevos mecanismos de contención y erradicación.

3) Tras el incidente, en la fase de **actividad posciberincidente**, los responsables de la organización emitirán un Informe del ciberincidente, que detallará su causa originaria y su coste (especialmente, en términos de compromiso de información o de impacto en los servicios prestados), así como las medidas que la organización debe tomar para prevenir futuros ciberincidentes de naturaleza similar.

3. Caso de Uso I: Seguridad en entornos IaaS públicos

En este apartado veremos prácticas de seguridad aplicadas a entornos IaaS públicos, en concreto veremos el caso de Amazon Web Services y de Digital Ocean.

3.1. Caso Amazon Web Services

En el caso de **Amazon Web Services**, veremos brevemente los aspectos relativos a la seguridad más importantes que debemos tener en cuenta, para centrarnos después en detalle en cómo mitigar los ataques de tipo DDoS combinando diferentes servicios del proveedor de IaaS.

3.1.1. Seguridad de la infraestructura

Amazon Web Services, en adelante AWS, proporciona varias capacidades y servicios de seguridad para mejorar la privacidad y controlar el acceso de redes. Entre ellos, se incluyen los *firewalls* de red integrados en Amazon VPC, y las capacidades de *firewall* para aplicaciones web existentes en AWS WAF permiten crear redes privadas y controlar el acceso a las instancias y aplicaciones. También se incluye el cifrado en tránsito con TLS en todos los servicios. Y por último, existen las opciones de conectividad que permiten conexiones privadas o dedicadas desde la oficina o entorno *on-premise*.

Como podemos ver, AWS en sí ya implementa diferentes medidas de seguridad en cuanto a seguridad de la infraestructura, lo que permite a los usuarios de AWS centrarse en otros aspectos de la seguridad.

3.1.2. Cifrado de datos

AWS ofrece la posibilidad de añadir una capa de seguridad adicional a los datos en reposo en la nube, proporcionando características de cifrado en los servicios de base de datos y almacenamiento, como EBS, S3, Glacier, Oracle RDS, SQL Server RDS y Redshift.

También podemos utilizar servicios de administración de claves, como AWS Key Management Service, que permiten elegir si AWS administra las claves de cifrado o si mantenemos el control sobre las claves.

AWS también proporciona diversas API para que pueda integrar el cifrado y la protección de los datos con cualquiera de los servicios desarrollados en un entorno de AWS.

3.1.3. Trazabilidad de cambios

Utilizando herramientas de administración de inventario y configuración, como AWS Config, que identifican los diferentes recursos de AWS, se puede realizar el seguimiento de los cambios en esos recursos a lo largo del tiempo. Podemos así construir una trazabilidad sobre los cambios de configuración e incluso volver atrás en el tiempo hacia un escenario más seguro.

3.1.4. Escaneo de vulnerabilidades

AWS ofrece un servicio de evaluación de la seguridad, Amazon Inspector, que evalúa las aplicaciones automáticamente para detectar vulnerabilidades o desviaciones de las prácticas recomendadas. Dicho escaneo incluyes las redes, el sistema operativo y el almacenamiento.

3.1.5. Cuentas de usuario y API

Amazon ofrece herramientas para cuidar la seguridad. La autenticación por múltiples factores en las cuentas de AWS es muy recomendable mediante el servicio AWS Multi-Factor Authentication.

Se recomienda crear cuentas separadas para los usuarios de Amazon con el fin de no compartir contraseñas. Hay que asegurar también que no se utilicen cuentas *root* y que las cuentas solo tengan los privilegios necesarios, por ejemplo las cuentas de desarrollador. AWS Identity and Access Management (IAM) permite definir cuentas de usuarios individuales con permisos en los recursos de AWS.

Es recomendable utilizar las herramientas de Amazon para administrar claves privadas y almacenarlas posteriormente de forma segura. Por último, hay que supervisar las actividades sospechosas, como las llamadas inesperadas a la API e inicios de sesión inusuales.

3.1.6. Protección contra ataques DDoS

En el caso de un ataque DDoS (*distributed denial of service*), un atacante utiliza múltiples fuentes que pueden haber sido comprometidas o controladas por un colaborador para orquestar un ataque contra un objetivo. En un ataque DDoS, cada uno de los colaboradores o *hosts* comprometidos participa en el ataque, generando una inundación de paquetes o peticiones con el fin de que un servicio o recurso sea inaccesible a los usuarios legítimos.

Defensa de la capa de infraestructura

En un entorno de centro de datos tradicional, se puede proteger la capa de infraestructura frente a ataques DDoS utilizando técnicas como el sobreprovisionamiento, desplegando sistemas de mitigación de DDoS, o por medio la

limpieza de tráfico con la ayuda de servicios de mitigación de DDoS. En AWS existen opciones para diseñar una aplicación para poder escalar y absorber mayores volúmenes de tráfico sin recurrir a una complejidad innecesaria.

1) Tamaño de instancia

La capacidad de cómputos es un recurso escalable en Amazon EC2. Se puede escalar horizontalmente añadiendo instancias a la aplicación, según sea necesario. O también se puede escalar verticalmente usando instancias mayores. Algunos tipos de instancia constan de interfaces de red de 10 Gigabits, que mejoran la capacidad para manejar grandes volúmenes de tráfico. Esto ayuda a evitar la congestión de la interfaz para cualquier tráfico que llegue a la instancia de Amazon EC2. Estas instancias tienen mayor rendimiento de E/S, lo que implica una menor utilización de la CPU.

2) Elección de región

Muchos servicios de AWS, como Amazon EC2, están disponibles en múltiples ubicaciones en todo el mundo. Estas áreas geográficamente separadas se denominan Regiones AWS. En el momento de diseñar la arquitectura de la aplicación, se pueden elegir una o más regiones dependiendo de los requisitos. En cada región, AWS proporciona acceso a un conjunto único de conexiones a internet y relaciones de *peering* que permiten una latencia adecuada a los usuarios finales que se sitúan cerca de la región. También es importante considerar la elección de la región en términos de flexibilidad DDoS. Muchas regiones están más cerca de los grandes intercambios de internet. Muchos ataques DDoS se orquestan internacionalmente, por lo que es útil estar cerca de los puntos de intercambio, ya que esto ayuda a los usuarios finales a alcanzar la aplicación.

3) Balanceo de carga

Los ataques DDoS más grandes pueden superar el tamaño de una única instancia de Amazon EC2. Al mitigar estos ataques, hay que considerar opciones para balancear la carga. Con Elastic Load Balancing (ELB) se puede reducir el riesgo de sobrecarga de la aplicación mediante la distribución de tráfico entre muchas instancias de *backend*. ELB puede balancear automáticamente, lo que le permite gestionar grandes volúmenes de tráfico, como los volúmenes resultantes de los ataques DDoS.

ELB solo acepta conexiones TCP bien formadas. Esto significa que muchos de los ataques DDoS, como inundaciones SYN o ataques de reflexión UDP, no serán aceptados por ELB. Cuando ELB detecta estos tipos de ataques, escala automáticamente para absorber el tráfico adicional.

4) Entrega a escala mediante AWS Edge Locations

El acceso a conexiones de internet muy escaladas y diversas aumenta la capacidad para optimizar la latencia y el rendimiento a los usuarios finales, para absorber ataques de DDoS y para aislar las fallas mientras se minimiza el impacto de la disponibilidad. AWS Edge constituye una capa adicional de infraestructura de red que proporciona beneficios a las aplicaciones web que utilizan también Amazon CloudFront y Amazon Route53.

Con estos servicios, el contenido se sirve y las consultas DNS se resuelven desde ubicaciones que a menudo están más cerca de sus usuarios finales. Amazon CloudFront es un servicio de red de distribución de contenido (CDN) que se puede utilizar para entregar todo el sitio web, incluido contenido estático, dinámico, de transmisión e interactivo. Las conexiones TCP persistentes y el tiempo de vida variable (TTL) se pueden utilizar para acelerar la entrega de contenido, incluso si no se puede almacenar en caché. Esto permite usar Amazon CloudFront para proteger una aplicación web, incluso si no se está sirviendo contenido estático. Amazon CloudFront solo acepta conexiones bien formadas para evitar que muchos ataques DDoS comunes –como inundaciones SYN y ataques de reflexión UDP– lleguen a tu origen. Los ataques DDoS están geográficamente aislados cerca de la fuente, lo que evita que el tráfico afecte a otras ubicaciones. Estas capacidades pueden mejorar considerablemente la capacidad para seguir sirviendo tráfico a los usuarios finales durante ataques DDoS mayores. Se puede utilizar Amazon CloudFront para proteger un origen en AWS o en cualquier otro lugar de internet.

5) Resolución de nombres de dominio

Amazon Route 53 es un servicio de nombres de dominio (DNS) altamente disponible y escalable que se puede utilizar para dirigir tráfico a una aplicación web. Incluye muchas características avanzadas, como flujo de tráfico, enrutamiento basado en latencia, Geo DNS, comprobaciones de estado y supervisión. Estas características permiten controlar cómo responde el servicio a las peticiones DNS para optimizar la latencia. Amazon Route 53 usa Shuffle Sharding y Anycast Striping para que los usuarios finales puedan acceder a la aplicación, incluso si el servicio DNS es atacado por un ataque DDoS. Con Shuffle Sharding, cada servidor de nombres en su conjunto de delegaciones corresponde a un conjunto único de ubicaciones y rutas de internet. Esto proporciona una mayor tolerancia a fallos y minimiza la superposición entre clientes. Si un servidor de nombres en el conjunto de delegaciones no está disponible, los usuarios finales pueden volver a intentarlo y recibir una respuesta de otro servidor de nombres en una ubicación diferente. Anycast Striping se utiliza para que cada petición de DNS se sirva desde la ubicación más rápida. Esto tiene el efecto de extender la carga y reducir la latencia del DNS, lo que permite a los usuarios finales recibir una respuesta más rápida. Además, Amazon Route 53 puede detectar anomalías en la fuente y el volumen de las consultas DNS, y priorizar las peticiones de los usuarios que se sabe que son fiables.

Defensa de capa de aplicación

1) Detectar y filtrar peticiones web malintencionadas

Los cortafuegos de aplicaciones web (WAF) se utilizan para proteger aplicaciones web contra ataques que intentan explotar una vulnerabilidad en la aplicación. Los ejemplos más comunes incluyen la inyección SQL o el *cross-site scripting*. También se puede utilizar un WAF para detectar y mitigar los ataques DDoS de la capa de aplicación web.

En AWS, se puede utilizar Amazon CloudFront y AWS WAF para defender una aplicación contra estos ataques. Amazon CloudFront permite almacenar en caché estática el contenido y servirlo desde AWS Edge Locations. Además, Amazon CloudFront puede cerrar automáticamente conexiones de lectura lenta o ataques de escritura lenta. Se puede utilizar la restricción geográfica Amazon CloudFront para bloquear ataques que se originen desde ubicaciones geográficas donde no se espera servir a los usuarios finales.

Puede ser difícil identificar la firma de un ataque DDoS o identificar las direcciones IP que están participando en el ataque. Se puede utilizar la consola AWS WAF para ver una muestra de las peticiones que Amazon CloudFront ha enviado a AWS WAF. Algunos ataques consisten en tráfico web que se disfraza para parecer tráfico procedente de un usuario final. Para mitigar este tipo de ataque, se puede utilizar una función AWS Lambda para implementar listas negras basadas en una tasa. Si un *bot* o *crawler* excede este límite, se puede utilizar AWS WAF para bloquear automáticamente futuras peticiones.

2) Escala para absorber

Otra forma de hacer frente a los ataques en la capa de aplicación es escalar. En el caso de las aplicaciones web, se puede utilizar ELB para distribuir instancias EC2 sobreprovisionadas o configuradas para escalar automática con el objetivo de atender los aumentos de tráfico, que pueden ser resultado de un ataque DDoS en la capa de aplicación. Las alarmas de Amazon CloudWatch se utilizan para iniciar el Auto Scaling, que escalona automáticamente el tamaño de la flota de Amazon EC2 en respuesta a eventos previamente definidos. Esto protege la disponibilidad de las aplicaciones incluso cuando se está tratando un volumen inesperado de peticiones. Mediante el uso de Amazon CloudFront o ELB, la negociación SSL es manejada por el balanceador de carga, lo que puede impedir que sus instancias se vean afectadas por ataques que exploten vulnerabilidades SSL.

Ofuscación de los recursos AWS

Otra consideración importante al diseñar en AWS es limitar las oportunidades que un atacante pueda tener para aprender de la aplicación. Por ejemplo, si no se espera que un usuario final interactúe directamente con ciertos recursos, estos recursos no deben ser accesibles desde internet.

Del mismo modo, si no se espera que los usuarios finales o las aplicaciones externas se comuniquen con la aplicación en ciertos puertos o protocolos, ese tipo de tráfico debería no ser aceptado. Este concepto se conoce como reducción de la superficie de ataque.

Para muchas aplicaciones, los recursos AWS no necesitan estar completamente expuestos a internet. Por ejemplo, puede no ser necesario para las instancias de Amazon EC2 detrás de un ELB ser accesibles al público. En este escenario, se puede permitir a los usuarios finales acceder al ELB en ciertos puertos TCP y permitir que solo el ELB pueda comunicarse con las instancias de Amazon EC2. Esto puede lograrse configurando grupos de seguridad y listas de control de acceso a la red (NACL) dentro de Amazon Virtual Private Cloud (VPC). Amazon VPC le permite proveer una sección lógicamente aislada de la nube de AWS donde lanzar recursos de AWS en una red virtual que defina.

Los grupos de seguridad y las ACL de red son similares, ya que permiten controlar el acceso a los recursos de AWS dentro de una VPC. Los grupos de seguridad permiten controlar tráfico entrante y saliente a nivel de instancia, y la ACL de red ofrece capacidades similares, pero a nivel de subred VPC.

1) Grupos de seguridad

Se pueden especificar grupos de seguridad al iniciar una instancia. Todo el tráfico desde internet en un grupo de seguridad se niega implícitamente a menos que se cree una regla para permitir el tráfico. Por ejemplo, si consideramos una aplicación web que consiste en un ELB y varias instancias Amazon EC2, se podría crear un grupo de seguridad para el ELB (Grupo de seguridad ELB) y uno para las instancias (Grupo de seguridad servidor de aplicaciones web). A continuación, se pueden crear permisos para permitir el tráfico desde internet para el grupo de seguridad ELB y para permitir el tráfico del Grupo de seguridad ELB al servidor de aplicaciones web. Como resultado, el tráfico en internet no puede comunicarse directamente con sus instancias de Amazon EC2, lo que hace más difícil para un atacante aprender acerca de una aplicación.

2) Listas de control de acceso a la red

Esta herramienta es útil cuando se responde a ataques DDoS porque puede permitir crear reglas para mitigar el ataque si conoce la dirección IP origen. Con las ACL de red, se pueden especificar las reglas de permitir y denegar. Esto resulta útil en caso de que se deseen negar explícitamente ciertos tipos de

tráfico a la aplicación. Por ejemplo, puede definir direcciones IP (como rangos CIDR), protocolos y puertos de destino que se deben denegar para toda una subred. Si una aplicación utiliza solo tráfico TCP, se puede crear una regla para denegar todo el tráfico UDP o viceversa.

3) Protección basada en el origen de las peticiones

Si se está utilizando Amazon CloudFront con un origen que se encuentra dentro de una VPC, se puede utilizar una función AWS Lambda para actualizar automáticamente las reglas del grupo de seguridad asociado con el fin de permitir solo el tráfico de Amazon CloudFront. Esto obliga a las peticiones a ser procesadas por Amazon CloudFront y AWS WAF.

4) Protección de los puntos finales de API

Normalmente, cuando existe la necesidad de exponer una API al público, aparece el riesgo de que el interfaz API pueda ser atacado por un ataque DDoS. Amazon API Gateway es un servicio totalmente administrado que permite crear una API que actúa como una «puerta de entrada» a aplicaciones que se ejecutan en Amazon EC2, AWS Lambda o cualquier aplicación web.

Con Amazon API Gateway, no es necesario que ejecute sus propios servidores para la API, y se pueden ocultar otros componentes de la aplicación al acceso público. Esto ayuda a evitar que los recursos de AWS sean atacados mediante un DDoS. Amazon API Gateway está integrado con Amazon CloudFront, lo que permite beneficiarse de la capacidad adicional de DDoS que es inherente a ese servicio. También se puede proteger el *backend* del exceso de tráfico configurando límites de velocidad estándar o de ráfaga para cada método para las API de tipo REST.

3.2. Caso Digital Ocean

Digital Ocean es un proveedor estadounidense de servidores virtuales privados. La compañía alquila servidores de sus centros de datos, a los que llama *droplets*. Aprovechando que se trata de un proveedor muy simple, en este caso de uso se explicarán los conceptos de seguridad básica práctica aplicados al entorno de Digital Ocean.

3.2.1. Claves SSH

Las claves SSH son un par de claves criptográficas con las que se puede autenticar contra un servidor SSH como una alternativa a los *logins* basados en contraseña. Con anterioridad a la autentificación se han creado un par de claves, una privada y una clave pública. La clave privada se mantiene secreta y asegurada por el usuario, mientras que la clave pública puede ser compartida con cualquiera.

Para configurar la autenticación SSH, hay que colocar la clave pública del usuario en el servidor en un directorio específico. Cuando el usuario conecta al servidor, el servidor comprobará que el cliente tiene la clave privada asociada. El cliente SSH utilizará la clave privada para responder en una manera que pruebe la propiedad de la clave privada. Entonces, el servidor permite al cliente conectar sin el uso de una contraseña.

Mediante el uso de SSH, cualquier clase de autenticación, incluyendo autenticación bajo contraseña, es completamente encriptada. Aun así, cuando el servidor permite el uso de contraseñas, un usuario malicioso podría intentar acceder al servidor repetidamente. Con la capacidad actual de cómputo, es posible obtener acceso a un servidor mediante la automatización de estos intentos probando diferentes combinaciones de contraseñas hasta encontrar la correcta.

La instalación de la autenticación de claves SSH permite deshabilitar la autenticación basada en contraseñas. Generalmente, las claves SSH tienen muchos más bits de datos que una contraseña, lo que significa que hay muchas más combinaciones posibles que un atacante tendría que ejecutar para ganar acceso al sistema. Muchos algoritmos de clave SSH se consideran no *hackeables* por el hardware de computación actual simplemente porque requerirían demasiado tiempo para computar las diferentes posibilidades.

Las claves SSH son muy fáciles de instalar y es la forma recomendable de registrarse en cualquier entorno de servidor Linux o Unix de modo remoto. Un par de claves SSH se pueden generar en la máquina del usuario y puede transferirse la clave pública a sus servidores en unos minutos mediante un gestor de configuración.

En el caso de tener que utilizar autenticación mediante contraseña, existe la posibilidad de implementar una solución como fail2ban en los servidores para limitar los intentos de acceso fraudulentos.

3.2.2. Cortafuegos

Un cortafuegos es un dispositivo software que controla qué servicios están expuestos a la red. Esto significa bloquear o restringir el acceso a cada puerto, excepto aquellos que tengan que estar públicamente disponibles.

En un servidor típico, los servicios pueden ser categorizados en los siguientes grupos:

- Servicios públicos a los que puede acceder cualquier persona en internet, a menudo de forma anónima. Un buen ejemplo de esto es un servidor web.

- Servicios privados que solo tendrían que ser accedidos por un grupo de cuentas autorizadas o sitios seguros. Un ejemplo de esto puede ser un panel de base de datos.

- Servicios internos que tendrían que ser accesibles solo desde dentro del propio servidor, sin exponer el servicio al mundo exterior. Por ejemplo, puede tratarse de una base de datos que solo acepta conexiones locales.

Un cortafuegos puede garantizar que el acceso a un determinado servicio esté restringido según las categorías anteriores. Los servicios públicos pueden dejarse abiertos y estar disponibles para todos, y los servicios privados pueden restringirse basándose en criterios diferentes. Los servicios internos pueden ser completamente inaccesibles al mundo exterior. Para los puertos que no se utilizan, el acceso se bloquea completamente.

La configuración del cortafuegos es una parte esencial de la configuración de un servidor; incluso si los servicios instalados implementan características de seguridad o están restringidos a las interfaces de red correspondientes, un cortafuegos provee de una capa adicional de protección.

Un cortafuegos correctamente configurado restringirá el acceso a todo excepto a los servicios específicos que se necesita que permanezcan abiertos. Exponer solo unos pocos servicios reduce la superficie de ataque de un servidor, limitando los componentes vulnerables a la explotación.

Hay muchos cortafuegos disponibles para sistemas Linux, algunos de los cuales tienen una curva de aprendizaje más pronunciada que otros. Por lo general, la instalación del cortafuegos solo tardaría unos minutos y solo tendrá que pasar por un servidor de configuración inicial o cuando realice cambios en los servicios.

3.2.3. VPN y redes privadas

Las redes privadas son redes que solo son visibles dentro de la red que forman nuestros servidores.

Para conectar equipos remotos a una red privada, se utiliza una VPN, o red privada virtual, que es una forma de crear conexiones seguras entre equipos remotos y presentar la conexión como una red privada local. Esto proporciona una forma de configurar los servicios como en una red privada y de conectar servidores remotos a través de conexiones seguras.

El uso de una VPN es efectivamente una forma de mapear una red privada que solo los usuarios pueden ver. La comunicación será totalmente privada y segura. Otras aplicaciones se pueden configurar para pasar el tráfico a través

de la interfaz virtual expuesta por el software VPN. De esta manera, los únicos servicios se exponen públicamente son los servicios que están destinados a ser accedidos por los usuarios a través de internet.

El uso de redes privadas en un centro de datos que tenga esta capacidad es fácil: basta con habilitar la interfaz privada durante la creación de configuraciones de servidores y aplicaciones, y configurar el cortafuegos para usar la red privada.

En cuanto a VPN, la configuración inicial resulta un poco más complicada, pero la seguridad aumenta considerablemente, por lo que es la opción recomendada para la mayoría de los casos de uso. Cada servidor en una VPN necesita instalar y configurar los datos comparativos de seguridad y configuración necesarios para establecer la conexión segura. Después de que la VPN esté funcionando, las aplicaciones deben configurarse para usar el túnel VPN.

3.2.4. Infraestructura de clave pública y SSL/TLS encriptación

La infraestructura de clave pública, o PKI, se refiere a un sistema diseñado para crear, dirigir y validar certificados para identificar comunicaciones individuales y de cifrado. Los certificados SSL o TLS pueden ser auténticos y diferentes. Después de la autenticación, la comunicación establecida también puede encriptarse usando los mismos certificados.

Establecer una autoridad de certificación y los certificados de los administradores de los servidores de cada entidad dentro de la infraestructura sirve para validar la otra identidad de los miembros y cifrar el tráfico. Asimismo, puede evitar el ataque a las comunicaciones en el que un atacante imita ser un servidor de la infraestructura para interceptar tráfico.

Cada servidor puede configurarse para confiar en una autoridad de certificación centralizada. A continuación, se puede confiar implícitamente en cualquier certificado que señale la autoridad. Encriptación TLS / SSL es una manera de encriptar el sistema sin utilizar una VPN (que también SSL utiliza internamente).

Hay que configurar la autoridad de certificación e instalar el resto de la infraestructura. Además, dirigir los certificados crea una nueva necesidad de gestión de los certificados que deben crearse, firmarse o revocarse.

Implementar una infraestructura de clave pública tiene sentido en infraestructuras grandes. Asegurar las comunicaciones entre los componentes que usan VPN puede ser una buena estrategia hasta que alcance el punto en el que la PKI valga la pena los costes de administración adicionales.

3.2.5. Auditoría de archivo y archivo de sistemas de detección

La auditoría de intrusión es el proceso de comparar el sistema actual de un registro de los archivos y los archivos del sistema cuando se encontraban en buen estado. Por lo general, detecta cambios en el sistema que pueden no haber sido autorizados.

Esta estrategia es una manera de asegurarse de que el sistema de archivos no ha sido alterado por ningún usuario o proceso. A menudo, los intrusos quieren permanecer ocultos para seguir explotando el servidor durante un periodo prolongado de tiempo. Estos podrían reemplazar binarios con versiones comprometidas. Hacer una auditoría del sistema de archivos para saber si alguno de los archivos ha sido alterado asegura la integridad del entorno del servidor.

La realización de auditorías de archivo puede ser un proceso muy intenso. La configuración inicial implica la auditoría del sistema sobre cualquier cambio no estándar realizado en el servidor y la definición de rutas que deberían ser excluidas para crear una base de lectura.

También hace que las operaciones diarias sean más complicadas. Complica los procedimientos de actualización cuando sea necesario para volver a comprobar el sistema antes de ejecutar actualizaciones y reconstruir la *baseline* después de ejecutar la actualización. También es necesario descargar los informes a otra ubicación para que un atacante no pueda alterar la auditoría.

Si bien esto puede aumentar la carga de la administración, ser capaz de comprobar el sistema asegura que los archivos no han sido alterados. Algunos sistemas populares de detección son Tripwire y Aide.

3.2.6. Entornos de ejecución aislada

Aislar entornos de ejecución se refiere a cualquier método en el que componentes individuales se están ejecutando dentro de un espacio dedicado.

Esto puede significar separar los componentes de aplicación en servidores propios o puede referirse a configurar los servicios para operar en entornos *chroot* o contenedores. El nivel de aislamiento depende fuertemente de los requisitos de cada aplicación.

Aislar los procesos en entornos de ejecución individual aumenta la capacidad de aislar cualquier problema de seguridad que pueda surgir. Separar los componentes individuales puede limitar el acceso que un intruso tiene a otras piezas de la infraestructura.

Dependiendo del tipo de aislamiento, aislar una aplicación puede ser una tarea relativamente sencilla. Paquetizar los componentes individuales en contenedores es una manera, pero curiosamente Docker no considera su aislamiento una medida de seguridad.

Instalar un *chroot* en el entorno para cada pieza software también puede proporcionar un cierto nivel de aislamiento, pero cabe recordar que hay maneras de romper el *chroot* incluso para aplicaciones que se encuentra dentro de este tipo de aislamiento.

4. Caso de uso II: Seguridad en entornos PaaS privados basados en Docker

Docker es una plataforma que nos permite construir y ejecutar aplicaciones distribuidas basadas en contenedores. Esta nueva forma de trabajar está cambiando el paradigma del desarrollo de aplicaciones, ya que se pasa de aplicaciones monolíticas a aplicaciones basadas en microservicios. Muchas de las grandes empresas, como Google o Spotify, e infinidad de *start ups* recurren a esta arquitectura, afrontando nuevos retos en cuanto a seguridad se refiere.

Conceptualmente, cada aplicación bajo contenedores dockers debe aislarse de cualquier otra aplicación, limitando y controlando los recursos a los que estos pueden acceder mediante capacidades heredadas del *host*, como namespaces y cgroups.

Docker ofrece seguridad basada en capas; por defecto estos ya incorporan métodos de aislamiento, pero debemos recordar que Docker es una plataforma que puede ejecutarse en máquinas virtuales y añadir una capa suplementaria en el caso de requerir un aislamiento extra.

Para definir cómo securizar un entorno dockerizado, no existe un protocolo infalible, sino que procederemos a seguir un conjunto de recomendaciones para mitigar y dificultar cualquier ataque recibido.

Entre los ataques más comunes con los que deberemos tener especial atención encontraremos:

- Ataques **Conainer scape**, escalado de privilegios en el *host*.
- Ataque **Root/Kernel exploit**, *host* comprometido.
- Ataque **DOS** (*denial of service*), pérdida de disponibilidad.
- Ataque **Software & Crypto exploit**, pérdida de confidencialidad.

4.1. Seguridad en el *host*

Para definir la seguridad de nuestras aplicaciones dockerizadas, primero deberemos centrarnos en la raíz, donde estos se ejecutarán.

Seguiremos algunas recomendaciones que afectarán a distintos ámbitos del sistema anfitrión:

1) Uso de imágenes validadas

Durante la construcción de nuestros contenedores es común el uso de contenedores de terceros como punto de partida; la recomendación más clara es la de usar repositorios validados, ya que cualquier fuente no confiable puede exponer nuestro contenedor a numerosas vulnerabilidades.

2) Securización del *kernel* de Linux

Es recomendable aplicar los parches de Grsecurity (grsecurity.net), que ofrecen una mejora de seguridad para el *kernel* de Linux, incorporando entre otros mecanismos ante amenazas de seguridad un control de acceso inteligente y controles de prevención de vulnerabilidades basadas en corrupción de memorias.

3) Recursos del sistema con ulimit

Linux incorpora el comando ulimit, encargado de obtener y modificar los límites de los recursos del usuario; con él podremos limitar el número máximo de ficheros abiertos o el número máximo de procesos que los contenedores podrán manejar por usuario. Por otro lado, Docker también incorpora mecanismos para definir estos valores mediante el uso de parámetros con la misma nomenclatura (ulimit).

Con la limitación del número de recursos con ulimit estableceremos valores máximos para los contenedores, pero recordad que otras aplicaciones como Apache o Nginx disponen de parámetros extra para establecer distintos límites dentro de ellos.

Podremos definir distintos tipos de limitaciones, de manera global o local para contenedores. A continuación mostraremos distintos ejemplos de configuración de máximo de ficheros abiertos y de procesos activos:

a) Estableciendo límites durante el arranque del demonio de Docker:

```
docker daemon --default-ulimit nofile=100:200
```

El parámetro nofile define los valores máximos de la siguiente forma <soft limit>[:<hard limit>]. Diferenciaremos el soft limit y el soft limit: el soft limit es lo que realmente se aplica para una sesión o proceso, permitiendo al administrador (o usuario) establecer el límite máximo requerido; el hard limit es un campo opcional y define el límite máximo para el parámetro definido de soft limit.

b) Modificando el fichero /etc/security/limits.conf:

```
docker soft nofile <max_soft_limit>
docker hard nofile <max_hard_limit>
```

c) Mediante la sobreescritura de los valores de límites por defecto en el momento de la ejecución de un contenedor específico, limitando el número de ficheros máximos abiertos mediante el parámetro --ulimit nofile o estableciendo el número máximo de procesos abiertos por el contendor:

```
docker run --name <nombre container> --ulimit nofile=<soft limit>:<hard limit> ...

docker run --name <nombre container> --ulimit nproc=<soft limit>:<hard limit> ...
```

d) Estableciendo las opciones oportunas dentro del fichero de configuración de Docker /etc/sysconfig/docker:

```
OPTIONS="--ulimit nofile=<soft limit>:<hard limit> --ulimit nproc=<soft limit>:<hard limit>"
```

O en el fichero /etc/systemd/system/docker.service.d:

```
[Service]
Environment="OPTIONS=$OPTIONS \"--default-ulimit nofile=<soft limit>:<hard limit>\""
Environment="OPTIONS=$OPTIONS \"--default-ulimit nproc=<soft limit>:<hard limit>\""
```

Después de agregar o modificar un archivo deberemos ejecutar el comando systemctl daemon-reload para indicar a systemd que vuelva a cargar la configuración del servicio.

Nota

Los valores ulimit especificados para un contenedor sobrescriben los valores predeterminados que establecidos para el *daemon*.

4) Namespaces

Los espacios de nombres son una característica del *kernel* de Linux que aísla y virtualiza los recursos del sistema de una colección de procesos.

Los contenedores Docker son muy similares a los contenedores LXC y tienen características de seguridad similares. Cuando se inicia un contenedor con Docker se crean un conjunto de espacios de nombres y grupos de control para el contenedor.

Los espacios de nombres proporcionan la primera y más sencilla forma de aislamiento: los procesos que se ejecutan dentro de un contenedor no pueden ver los procesos que se ejecutan en otro contenedor o en el sistema *host*.

Para remapear nuestro usuario con otro usuario con su propio identificador dentro del contenedor, ejecutaremos:

```
docker daemon --userns-remap=<nombre_usuario>
```

Esta instrucción tomará el identificador del <usuario> dentro del *host* y lo remapeará dentro del contenedor siguiendo la fórmula:

```
FIRST_SUB_UID = 100000 + (UID - 1000) * 65536
```

Por ejemplo, si nuestro usuario en el *host* está definido por usuario::100000:65536, al ejecutar el remap podremos encontrar que dentro del contenedor corresponderá con usuario:1000:1

5) Capabilities

Las capabilities definen un sistema de seguridad que permite «fragmentar» los privilegios de *root* en distintos componentes, y estos podrán ser asignados de forma individual a procesos que requieran de privilegios especiales para recursos específicos sin necesidad de poseer privilegios de superusuario.

Docker utiliza esta granulación de privilegios para restringir los accesos a recursos, definiendo a qué capas podrá tener acceso el contenedor.

Para definir un acceso global a los recursos, aunque no recomendable, se utilizará la instrucción:

```
docker run --privileged=true...
```

A menudo es necesario exponer directamente los dispositivos a un contenedor. La opción que utilizaremos será --device, que permitirá vincular el contenedor con dispositivos de almacenamiento específicos o dispositivos de audio sin poseer privilegios de *root*:

```
# Acceso a disco en modo lectura/escritura
docker run --device=/dev/sdd -i -t ubuntu

# Acceso a la tarjeta de sonido
docker run --device=/dev/snd:/dev/snd ...
```

Por defecto, el contenedor podrá leer (*read*), escribir (*write*) y mknod sobre estos dispositivos. Estos privilegios pueden ser sobreescritos mediante las opciones :rwm para cada etiqueta *device*:

```
#Permisos de lectura/escritura esplicito

docker run --device=/dev/sda:/dev/xvdc:rw --rm -it ubuntu fdisk  /dev/xvdc
```

6) Módulos de seguridad (AppArmor)

Tanto AppArmor (Application Armor) como SeLinux son módulos de seguridad de Linux que protegen un sistema operativo y sus aplicaciones contra amenazas de seguridad.

Para utilizar AppArmor, un administrador del sistema asocia un perfil de seguridad AppArmor para cada programa. Así pues, Docker espera encontrar una política de AppArmor cargada y aplicada para limitar la aplicación dockerizada. El binario Docker instala un perfil predeterminado de AppArmor en el archivo /etc/apparmor.d/docker, y este perfil se utilizará en los contenedores, no en el Docker Daemon.

7) Computación segura o Seccomp

El modo de computación segura (Seccomp) es una característica del *kernel* de Linux. Se utiliza para restringir las acciones disponibles dentro del contenedor. La llamada al sistema seccomp () opera en el estado de la llamada seccomp, y se puede usar esta función para restringir el acceso de la aplicación.

Esta función solo está disponible si Docker se ha creado con seccomp y el núcleo está configurado con CONFIG_SECCOMP habilitado. Para comprobar si el *kernel* es compatible con seccomp, ejecutaremos:

```
cat /boot/config-`uname -r` | grep CONFIG_SECCOMP=
CONFIG_SECCOMP=y
```

El perfil de seccomp predeterminado deshabilita en torno a 44 llamadas de sistema de más de 300 disponibles. Es moderadamente protector mientras proporciona una amplia compatibilidad de aplicaciones. El perfil predeterminado de Docker tiene un diseño JSON y puede sobreescribirse mediante:

```
docker run --rm -it --security-opt seccomp=/path/to/seccomp/profile.json hello-world
```

Donde profile.json sería un fichero JSON siguiendo la siguiente estructura:

```
{
 "defaultAction": "SCMP_ACT_ERRNO",
 "architectures": [
  "SCMP_ARCH_X86_64",
  "SCMP_ARCH_X86",
  "SCMP_ARCH_X32"
 ],
 "syscalls": [
  {
   "name": "accept",
   "action": "SCMP_ACT_ALLOW",
   "args": []
  },
```

```
  {
   "name": "accept4",
   "action": "SCMP_ACT_ALLOW",
   "args": []
  },
  ...
 ]
}
```

Os recomendamos la lectura de la documentación oficial para la especificación de todas aquellas llamadas al sistema disponibles.

8) Limitación de CPU

En algunos entornos nos interesará la limitación del uso de la CPU del *host* para protegerlo frente a ataques que intenten vulnerar las capacidades del procesador anfitrión; para ello, Docker dispone de opciones de configuración específicas:

a) --cpu-shares <int>

Limitación relativa a la cantidad de procesadores disponibles, se especifica en porcentaje.

b) --cpu-quota <int>

Limitación del uso de la CPU en el contenedor. El valor 0 predeterminado permite que el contenedor tome el 100 % de un recurso de CPU (1 CPU). Para limitar el uso del procesador al 50 %, estableceremos un valor de cuota de 50000.

```
#Límite del uso de la CPU al 50%
docker run -ti --cpu-quota=50000 ubuntu:latest /bin/bash.
```

c) --cpu-period <int>

Esta opción se utiliza para establecer el periodo de CPU y limitar el uso de la CPU del contenedor. El periodo por defecto de CPU CFS es de 100 ms. Usualmente --cpu-period funciona junto con la opción --cpu-quota.

```
#Limitación del 50% de la CPU cada 50ms
docker run -it --cpu-period=50000 --cpu-quota=25000 ubuntu:14.04 /bin/bash
```

d) --cpuset-cpus <string>

Permite la restricción del uso de determinados CPU de nuestro *host*, que se especificarán en formato «1,3» en el caso de querer utilizar solo 2 CPU (número 1 y 3) o con el formato «0-2» para indicar que usaremos un rango (uso de CPU 0, 1 y 2).

9) Limitación de RAM

En Docker, o en cualquier otro sistema contenedor, la memoria es administrada por el *kernel* que ejecuta Docker y los contenedores. El proceso que se ejecuta en el contenedor sigue funcionando como un proceso normal pero en un cgroup diferente, donde cada cgroup posee sus propios límites, como la cantidad de memoria que el *kernel* entregará al espacio del usuario o, como ya hemos visto, qué dispositivos tendrá disponibles.

Para especificar las limitaciones oportunas, usaremos las opciones de ejecución de la familia --memory de docker:

a) --memory <int>[<unidad (b, k, m o g)>]

Para definir la cantidad de memoria máxima que el contenedor tendrá disponible se indicará en forma de número entero positivo la cantidad de memoria seguido de la unidad correspondiente (b, k, m o g). El mínimo de memoria es de 4 M.

```
#Limitación de la memoria a 1Gb
docker run --memory 1G ubuntu
```

b) --memory-reservation <int>[<unidad (b, k, m o g)>]

Se utiliza normalmente en conjunción con la instrucción --memory. En el caso de especificar ambos parámetros, la memoria (especificada con opción --memory) debe ser mayor que la memoria reservada. Al especificar una cantidad de memoria reservada, el contenedor inicia con el valor reservado; de lo contrario, utilizará el valor de la memoria asignada.

Por ejemplo, si un contenedor normalmente utiliza 256 Mb de memoria, pero ocasionalmente requiere 512 Mb de memoria por periodos cortos de tiempo, podemos establecer una reserva de memoria de 256 Mb y un límite de memoria de 600 Mb, estableciendo así los dos ámbitos donde la memoria del contenedor puede ubicarse.

El mínimo de memoria reservada es de 4 M.

```
#Reserva de memoria inicial de 256Mb con limite de 600Mb
docker run --memory 600M --memory-reservation 256M ubuntu
```

c) --memory-swap <int>[<unidad (b, k, m o g)>]

Los contenedores pueden hacer uso de la memoria SWAP, que amplía los límites de la memoria RAM establecida por el parámetro –memory. Se establece así el uso total de memoria como memoria + memoria SWAP asignada.

d) --memory-swappiness <int>

Existe la posibilidad de ajustar la frecuencia con la que se utilizan los ficheros de intercambio del contenedor con valores de 0 a 100. Los valores bajos indicarán que no se utilizará memoria SWAP a menos que sea totalmente necesario. Por otro lado, los valores altos indicarán un uso exhaustivo de este tipo de memoria.

10) Redes

Cada contenedor Docker tiene su propia pila de red, lo que significa que un contenedor no obtiene acceso privilegiado a los *sockets* o interfaces de otro contenedor, aunque esto es posible si el sistema *host* está configurado para ello, e incluso interactuar con *hosts* externos.

Cuando se especifica un puerto público para el contenedor o se utilizan vínculos, se permite el tráfico IP entre contenedores, es decir, pueden hacer *ping* unos a otros, enviar/recibir paquetes UDP y establecer conexiones TCP.

Desde el punto de vista de la arquitectura de red, todos los contenedores de un *host* Docker determinado están situados en interfaces de puente (*bridge*). Esto significa que son como máquinas físicas conectadas a través de un conmutador ethernet común.

4.2. Buenas prácticas de seguridad

La seguridad de los sistemas es una cuestión bien tratada por los propios contenedores, y es el propio Docker el que está centrando cada vez más esfuerzos en mejorar los sistemas integrados de seguridad de estos. No obstante, para comprobar que nada quede en el descuido, incluimos un listado de noventa buenas prácticas desarrolladas por el Center for Internet Security (CIS) para Docker, que se podrán seguir a modo de *checklist*.

1. Configuración del *host*

1.1. Crear una partición separada para los contenedores
1.2. Uso del *kernel* de Linux actualizado
1.3. No utilizar herramientas de desarrollo en producción
1.4. Securizar el *host* anfitrión del contenedor
1.5. Eliminar todos los servicios no esenciales del *host*
1.6. Mantener Docker actualizado
1.7. Solo permitir que los usuarios de confianza controlen el *daemon* de Docker
1.8. Aplicar técnicas de auditoría sobre el *daemon* Docker
1.9. Auditar archivos y directorios de Docker - /var/ lib/docker
1.10. Auditar archivos y directorios de Docker - /etc/docker
1.11. Auditar archivos y directorios de Docker - docker-registry.service
1.12. Auditar archivos y directorios de Docker - docker.service (marcado)
1.13. Auditar archivos y directorios de Docker - /var/run/docker.sock
1.14. Auditar archivos y directorios de Docker - /etc/sysconfig/docker
1.15. Auditar archivos y directorios de Docker - /etc/sysconfig/docker-network
1.16. Auditar archivos y directorios de Docker - /etc/sysconfig/docker-registry
1.17. Auditar archivos y directorios de Docker - /etc/sysconfig/docker-storage
1.18. Auditar archivos y directorios de Docker - /etc/default/docker

2. Configuración del *daemon* de Docker

2.1. No utilizar el controlador de ejecución lxc
2.2. Limitar el tráfico de red entre contenedores
2.3. Establecer el nivel adecuado de *logging*
2.4. Permitir que Docker realice cambios en *iptables*
2.5. No utilizar registros inseguros (sin TLS)
2.6. Configurar un espejo del registro local
2.7. No utilizar aufs como controlador de almacenamiento
2.8. No vincular Docker a otro puerto IP o un socket Unix
2.9. Configurar la autenticación TLS para el *daemon* Docker
2.10. Establecer el valor predeterminado ulimit según corresponda

3. Archivos de configuración del demonio Docker

3.1 Comprobar que la propiedad del archivo docker.service está establecida en root:root
3.2. Comprobar que los permisos de archivo docker.service están establecidos en 644 o más restrictivos
3.3. Comprobar que la propiedad del archivo docker-registry.service está establecida en root:root
3.4 Comprobar que los permisos de archivo docker-registry.service están establecidos en 644 o más restrictivos
3.5. Comprobar que la propiedad del archivo docker.socket está establecida en root:root
3.6. Comprobar que los permisos de archivo docker.socket se establecen en 644 o más restrictivos
3.7. Comprobar que la propiedad del archivo de entorno Docker está establecida en root:root
3.8. Comprobar que los permisos de archivo de entorno de Docker se establecen en 644 o más restrictivos
3.9. Comprobar que la propiedad del archivo del entorno docker-network está establecida en root:root
3.10. Comprobar que los permisos de archivos de entorno de red Docker están configurados en 644 o más restrictivos
3.11. Comprobar que la propiedad del archivo del entorno docker-registry está establecida en root:root
3.12. Comprobar que los permisos de archivo del entorno de registro de base de datos se establecen en 644 o más restrictivos
3.13. Comprobar que la propiedad del archivo del entorno de almacenamiento acoplador está establecida en root:root
3.14 Comprobar que los permisos de archivo del entorno de almacenamiento acoplador se establecen en 644 o más restrictivos
3.15. Verificar que la propiedad del directorio /etc/docker está establecida en root:root
3.16. Verificar que los permisos del directorio /etc/docker están configurados como 755 o más restrictivos
3.17. Comprobar que la propiedad del archivo del certificado del registry se establece en root:root
3.18. Comprobar que los permisos de los archivos del certificados del registry se establecen en 444 o más restrictivos
3.19. Comprobar que la propiedad del archivo del certificado de la CA de TLS está establecida en root:root
3.20. Comprobar que los permisos de archivo del certificados de CA de TLS se establecen en 444 o más restrictivos
3.21. Verificar que la propiedad del archivo del certificado del servidor Docker está establecida en root:root
3.22. Verificar que los permisos de los archivos de certificados del servidor Docker están establecidos en 444 o más restrictivos
3.23. Comprobar que la propiedad del archivo de clave de certificado del servidor Docker se establece en root:root
3.24. Verificar que los permisos de archivo de clave de certificado del servidor Docker están establecidos en 400
3.25. Verificar que la propiedad del archivo de socket Docker está establecida en root:docker
3.26. Comprobar que los permisos de archivo de zócalo de Docker están establecidos en 660 o más restrictivos

4. Imágenes de contenedor y archivo de generación

4.1. Crear un usuario para el contenedor
4.2. Utilizar imágenes de confianza para los contenedores
4.3. No instalar paquetes innecesarios en el contenedor
4.4. Regenerar las imágenes para incluir parches de seguridad

5. Tiempo de ejecución del contenedor

5.1. Verificar el perfil de AppArmor
5.2. Verificar las opciones de seguridad SELinux
5.3. Verificar que los contenedores solo ejecutan un único proceso principal
5.4. Restringir las capacidades del *kernel* de Linux dentro de los contenedores
5.5. No utilizar contenedores con privilegios
5.6. No montar directorios sensibles del *host* en los contenedores
5.7. No ejecutar ssh dentro de los contenedores
5.8. No asignar puertos privilegiados dentro de los contenedores
5.9. Abrir solo los puertos necesarios en el contenedor
5.10. No utilizar el modo «host network» en el contenedor
5.11. Limitar el uso de memoria para el contenedor
5.12. Establecer la prioridad de CPU del contenedor de manera adecuada
5.13. Montar el sistema de archivos raíz del contenedor como solo lectura
5.14. Vincular el tráfico de contenedores entrantes a una interfaz de *host* específica
5.15. Establecer la política de reinicio del contenedor en modo «on-failure»
5.16. No compartir el PID del procesos del *host* anfitrión con los contenedores
5.17. No compartir el espacio de nombres IPC del anfitrión
5.18. No exponer directamente los dispositivos del anfitrión con los contenedores
5.19. Sobreescribir el valor predeterminado ulimit en tiempo de ejecución solo si es necesario

6. Operaciones de seguridad de Docker

6.1. Realizar auditorías regulares de seguridad del sistema anfitrión y de los contenedores
6.2. Monitorizar el uso, el rendimiento y la métricas de los contenedores de Docker
6.3. Uso Endpoint protection platform (EPP) para contenedores
6.4. Hacer copias de respaldo de los contenedores
6.5. Utilizar un servicio de recolegida de registros centralizado y remoto
6.6. Evitar el almacenamiento de imágenes obsoletas o sin etiquetas correctas
6.7. Evitar el almacenamiento de contenedores obsoletos o sin etiquetas correctas

5. Herramientas de seguridad

En este apartado citaremos algunas de las herramientas esenciales para el control de acceso, el control de fuga de información y la gestión de vulnerabilidades. También veremos los beneficios del uso de los estándares abiertos.

5.1. Herramientas de control de acceso

Los denominados agentes de seguridad para el acceso a la nube (*cloud access security broker*) es una categoría de herramientas de seguridad que surgen como una necesidad para atender riesgos de seguridad considerados por los proveedores de software como servicio. Gartner los define como los «puntos que refuerzan las políticas de seguridad utilizadas en la nube, y que se ubican entre los usuarios y los proveedores de servicios *cloud*, para combinar e intercalar las políticas de seguridad de la organización, relacionadas con la manera en la que se accede a los recursos».

- Netskope.
- Paloalto Aperture.
- Cloudlock.

5.2. Herramientas contra la fuga de información

El software de prevención de pérdida de datos (DLP en sus siglas en inglés) está diseñado para detectar potenciales brechas de datos / transmisiones de datos exfiltración y prevenirlas mediante la monitorización, la detección y el bloqueo de datos confidenciales durante el uso (acciones de punto final), en movimiento (tráfico de red) y en reposo (almacenamiento de datos). En incidentes de fuga de datos, los datos confidenciales son revelados a personal no autorizado por intención malintencionada o por un error inadvertido. Tales datos confidenciales pueden venir en forma de información privada o de empresa, propiedad intelectual (IP), información financiera o de pacientes, datos de tarjetas de crédito y otra información dependiendo del negocio y la industria.

5.2.1. MyDLP Community

MyDLP es un software libre y de código abierto de prevención de pérdida de datos. Se puede instalar tanto en servidores de red como en estaciones de trabajo. Los canales de inspección de datos soportados incluyen web, correo, im, transferencia de archivos a dispositivos de almacenamiento extraíble e impresoras. MyDLP proyecto de desarrollo ha hecho su código fuente disponible bajo los términos de la GNU General Public License.

5.2.2. Ossec

OSSEC es un sistema de detección de intrusos basado en *host* de código abierto y libre (HIDS). Realiza análisis de registro, comprobación de integridad, supervisión del registro de Windows, detección de *rootkits*, alertas basadas en el tiempo y respuesta activa. Proporciona detección de intrusiones para la mayoría de los sistemas operativos, incluyendo Linux, OpenBSD, FreeBSD, OS X, Solaris y Windows. OSSEC tiene una arquitectura centralizada y multiplataforma que permite que múltiples sistemas sean fácilmente monitorizados y administrados.

5.3. Herramientas de gestión de vulnerabilidades

5.3.1. BDD-Security

BDD-Security es un marco de pruebas de seguridad que utiliza lenguaje natural en una sintaxis Given, When y Then para describir requisitos de seguridad como características. Esos mismos requisitos también son ejecutables como pruebas estándar de unidad/integración, lo que significa que pueden ejecutarse como parte del proceso de compilación/prueba/implementación durante el proceso de integración continua.

Prueba aplicaciones web y API desde un punto de vista externo y no requiere acceso al código fuente de destino.

5.3.2. OWASP ZAP

El OWASP Zed Attack Proxy (ZAP) es una de las herramientas de seguridad gratuitas más populares del mundo y es mantenida activamente por cientos de voluntarios internacionales. Puede ayudar a encontrar automáticamente vulnerabilidades de seguridad en sus aplicaciones web durante su desarrollo y prueba. También es una gran herramienta para *pentesters* experimentados para usar para pruebas de seguridad manuales.

El complemento oficial de OWASP ZAP Jenkins amplía la funcionalidad de la herramienta de seguridad ZAP en un entorno de CI:

- Administrar sesiones (cargar o persistir).
- Definir contexto (nombre, incluir URL y excluir URL).
- Contextos de ataque (Spider Scan, AJAX Spider, Active Scan).

5.3.3. Nessus

Nessus es un escáner de vulnerabilidades desarrollado por Tenable Network Security. Permite escanear los siguientes tipos de vulnerabilidades:

- Vulnerabilidades que permiten a un *hacker* remoto controlar o acceder a datos confidenciales en un sistema.

- Misconfiguración (por ejemplo, relé de correo abierto, parches que faltan, etc.).

- Contraseñas predeterminadas, algunas contraseñas comunes y contraseñas en blanco/ausentes en algunas cuentas del sistema. Nessus también puede llamar a Hydra (una herramienta externa) para lanzar un ataque de diccionario.

- Denegaciones de servicio contra la pila TCP/IP mediante el uso de paquetes malformados.

- Preparación para auditorías PCI DSS.

5.4. Beneficios del uso de los estándares abiertos

Cuando se trata de seguridad, los estándares abiertos aparecen para ofrecernos todos sus beneficios:

- Agilizar las implementaciones en la nube: una capa API bien conocida y estándar dará a los desarrolladores de la empresa la capacidad de aprovechar las funciones básicas de la nube rápidamente, acelerando así el ritmo de las implementaciones en la nube.

- Fomentar las innovaciones entre nubes: con las API de Cloud Security Open, los desarrolladores ahora tienen una forma de escribir funciones de nube cruzada sin tener que integrarse a la perfección con cada nube que toque. Esto puede abrir innovaciones en nuevos escenarios económicos, nuevas formas de hacer negocios tanto para los usuarios como para los proveedores de la nube.

- Extender servicios *cloud* a nuevas funcionalidades: Desde la perspectiva de un proveedor de servicios en la nube (CSP), las API Cloud Security Open permitirán a un conjunto mucho mayor de desarrolladores (que aquellos dentro de la propia compañía de CSP) aprovechar la base de datos y entregar funcionalidad adyacente. A veces este modelo puede conducir a experiencias de usuario totalmente nuevas e inesperadas y a avances tecnológicos, lo que puede hacer que el servicio sea mucho más atractivo para los usuarios finales.

Entre todas las iniciativas destaca la Open API de la Cloud Security Alliance.

Enlace de interés

Podéis encontrar más información sobre la Open API en el siguiente enlace: Open API Working Group.

5.5. Herramientas de monitorización de recursos *cloud* configurados de manera insegura

Existen herramientas para monitorizar y alertar en caso de encontrar recursos *cloud* configurados de manera insegura.

5.5.1. Security Monkey

Es una solución de monitorización desarrollada por el equipo de Netflix para analizar la seguridad de los servicios de Amazon Web Services, Google Compute Platform, OpenStack y organizaciones de GitHub.

Se trata de un desarrollo propio que ofrecen como *open source* y que en el caso de AWS nos permite monitorizar la configuración de los servicios EC2 (computación), RDS (base de datos), S3 (DNS) e IAM (identidad). Cualquier cambio que se realice sobre ellos quedará registrado para que tengamos el control de todo lo que sucede.

Security Monkey tiene una interfaz gráfica por defecto, donde se pueden consultar todas las cuentas, regiones y servicios *cloud*. La interfaz es suficientemente inteligente como para mostrar solo los últimos cambios.

La herramienta puede ser fácilmente extendida.

Link al proyecto: https://github.com/Netflix/security_monkey