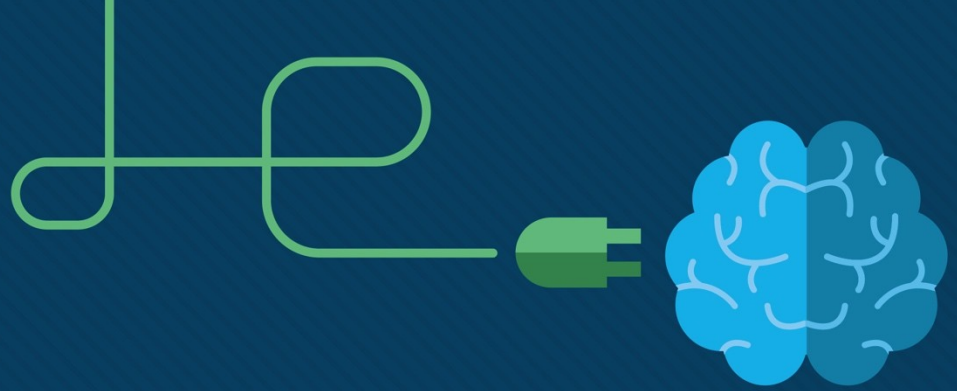


Modulo 6: Implementación de Aplicaciones y Seguridad

DevNet Associate v1.0

Objetivos del módulo

Título del módulo: Implementación de Aplicaciones y Seguridad

Objetivo del módulo: Usar la tecnología actual para implementar y brindar seguridad a las aplicaciones y datos en un entorno de nube.

Título del tema	Objetivo del tema
Entender las opciones de implementación con diferentes modelos	Explicar modelos de implementación de nube comunes.
Crear e Implementar una Aplicación de Muestra	Usar tecnología de contenedor para implementar una aplicación simple.
Integración continua/Implementación continua (CI/CD)	Explicar el uso de la integración continua/implementación continua (CI/CD) en la implementación de aplicaciones.
Redes para Desarrollo de Aplicaciones y Seguridad	Explicar la tecnología de red necesaria para el desarrollo de aplicaciones en un entorno de nube.
Protegiendo Aplicaciones	Utilizar técnicas comunes de seguridad de las aplicaciones para proteger los datos.

6.1 Comprensión de las opciones de implementación con diferentes modelos

Introducción a las opciones de implementación

- Los desarrolladores necesitan hacer algo más que entregar código de aplicación: deben preocuparse por cómo se implementan(implanta/despliega), protegen, operan, supervisan, escalan y mantienen las aplicaciones.
- La infraestructura física y virtual y las plataformas en las que se desarrollan e implementan aplicaciones están evolucionando rápidamente.
- Los desarrolladores se enfrentan a una creciente pila de opciones de plataforma, todas alojadas en infraestructuras y marcos (frameworks) de mayor flexibilidad y complejidad.

Entornos de Desarrollo

- Una pieza de código, antes de llegar al usuario, pasa a través de una serie de entornos que conduce a un aumento en su calidad y fiabilidad. Estos entornos son autónomos y imitan el entorno definitivo en el que vivirá el código.
- Normalmente, las organizaciones grandes utilizan una estructura de cuatro niveles:

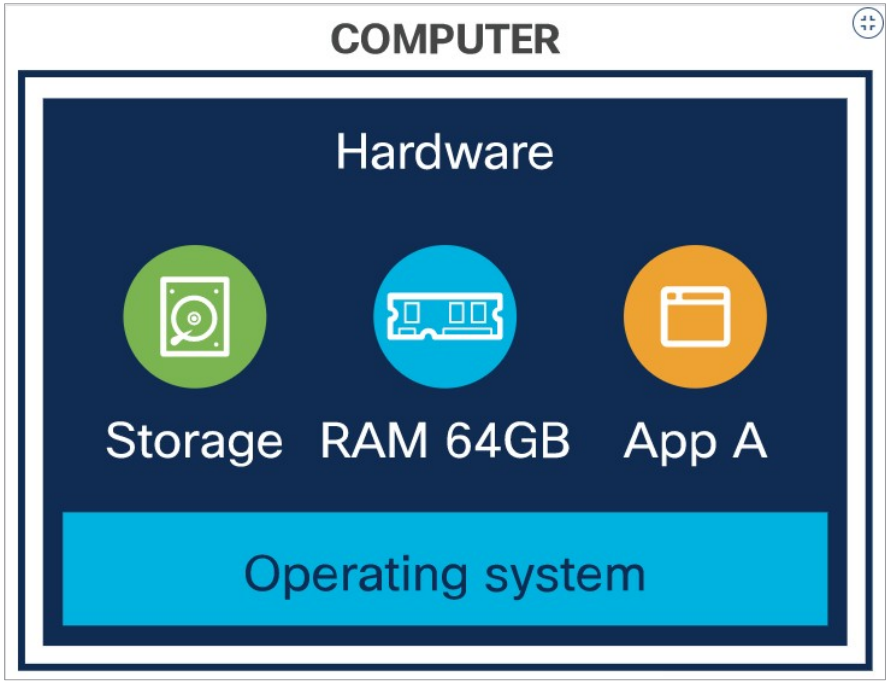
Desarrollo	Prueba	Etapas preliminares	Producción
Este entorno se utiliza para la codificación. También se utiliza para gestionar aspectos fundamentales de la infraestructura, como contenedores o redes en la nube.	Este entorno se utiliza para probar el código. Debe ser estructuralmente similar al entorno de producción final, a una escala mucho menor. A menudo incluye herramientas de prueba automatizadas, así como la integración con un sistema de control de versiones.	Este entorno se utiliza para las pruebas de aceptación final en un entorno realista. Una vez probado el código, se mueve al entorno de ensayo.	Este entorno se utiliza para implementar el código final para la interacción del usuario final. Debe ser dimensionado y construido para manejar el tráfico esperado, incluyendo sobretensiones que pueden venir estacionalmente o con un evento en particular.

Modelos de Implementación

Existen varios modelos de implementación que se pueden utilizar para implementar un software. Estos incluyen Bare Metal, Máquinas Virtuales, Infraestructura basada en contenedores y Computación sin servidor.

Bare Metal (metal básico) o Instalación desprovista de software

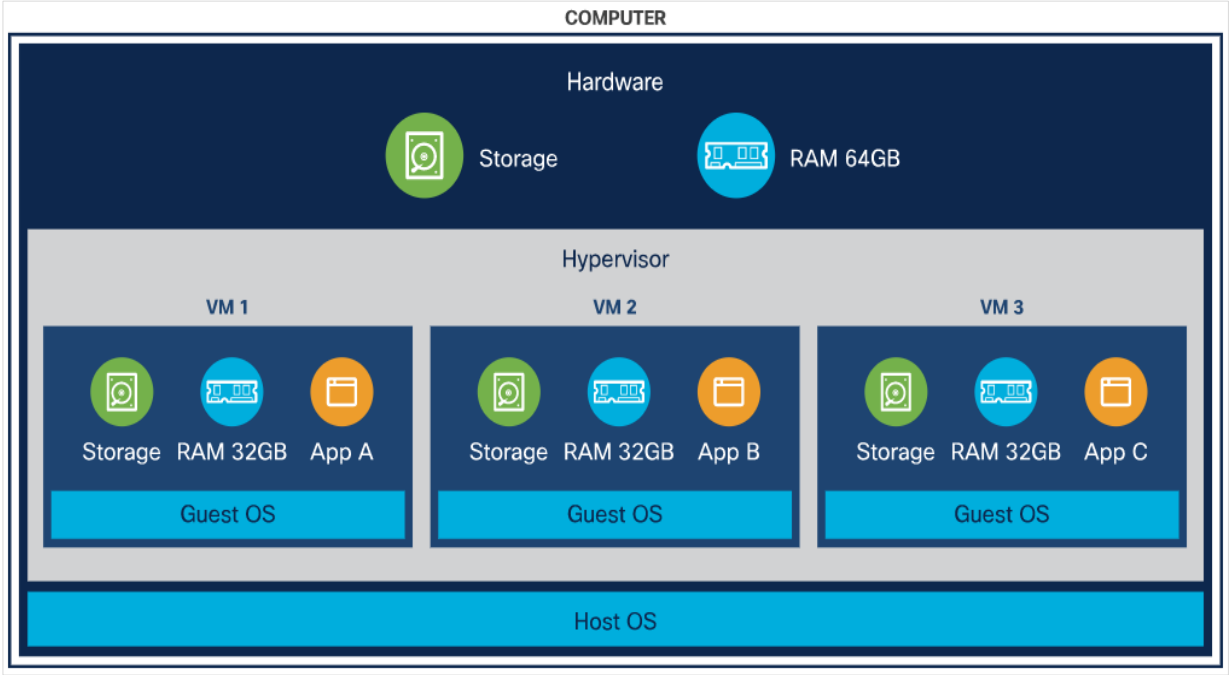
- Una implementación básica se está implementando esencialmente en un equipo real. Se utiliza para instalar un software directamente en el equipo de destino.
- En este método, el software puede acceder directamente al sistema operativo y al hardware.
- Es útil para situaciones que requieren acceso a hardware especializado o para aplicaciones de computación de alto rendimiento (High Perfomance Computing, HPC).
- Ahora se utiliza como infraestructura para alojar entornos de virtualización y nube.



Modelos de Implementación (Cont.)

Máquinas virtuales (VM)

- Las máquinas virtuales comparten los recursos del host. Es como una computadora dentro de la computadora y tiene su propia potencia informática, interfaces de red y almacenamiento.
- Hypervisor es un software que crea y administra máquinas virtuales.
- Las máquinas virtuales se ejecutan encima de un hipervisor que proporciona a las máquinas virtuales hardware simulado o acceso controlado al hardware físico subyacente.



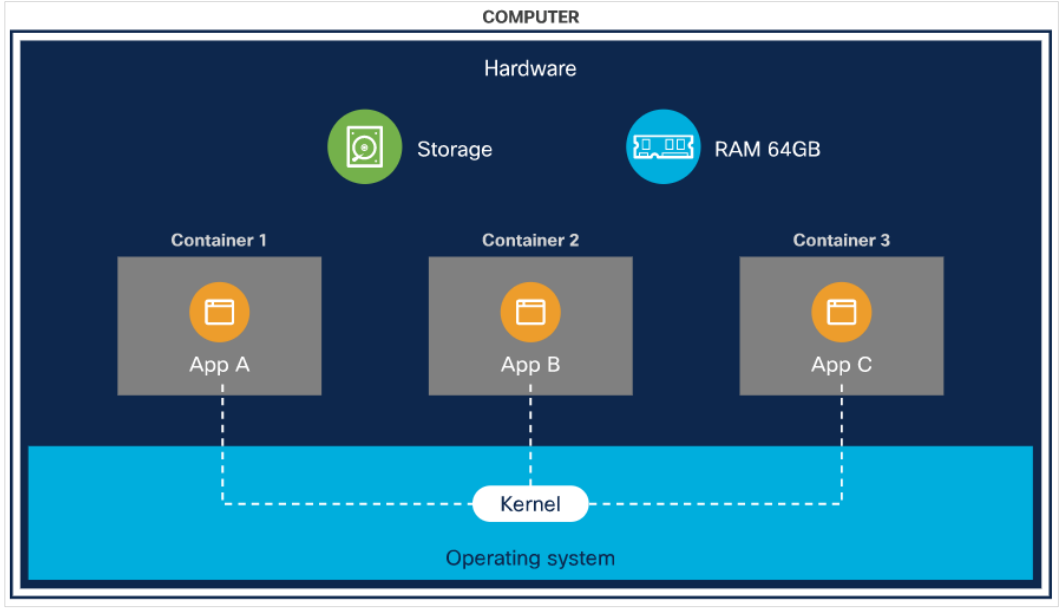
Modelos de Implementación (Cont.)

Infraestructura basada en contenedores

- Los contenedores se diseñaron para proporcionar las mismas ventajas que las máquinas virtuales, como el aislamiento de cargas de trabajo y la capacidad de ejecutar varias cargas de trabajo en una sola máquina, pero están diseñados para iniciarse rápidamente.

- Los contenedores comparten los recursos del host, incluido el kernel.

- Un contenedor comparte el sistema operativo de la máquina host y utiliza binarios y bibliotecas específicos del contenedor.



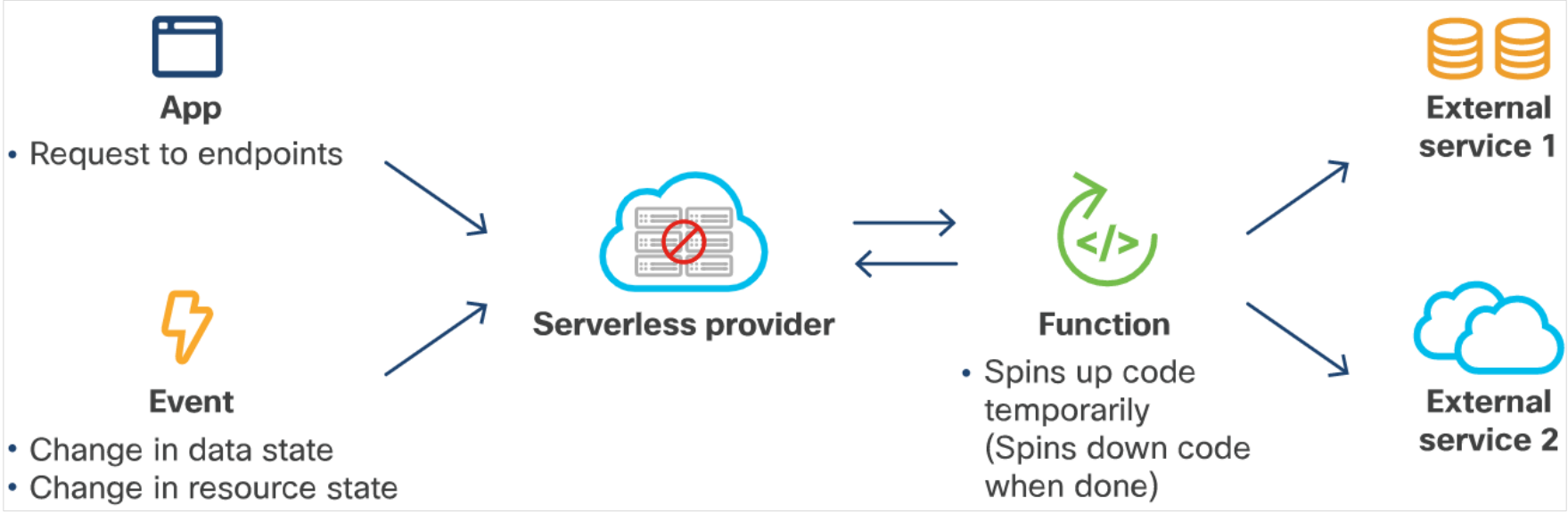
Modelos de Implementación (Cont.)

Informática sin Servidor

- La informática sin servidor aprovecha una tendencia moderna hacia las **aplicaciones que se construyen entorno a los servicios**. La aplicación realiza una llamada a otro programa o carga de trabajo para realizar una tarea en particular, para crear un entorno donde las aplicaciones se ponen a disposición «según sea necesario».
- Funciona de la siguiente manera:
  - **Paso 1.**El desarrollador crea una aplicación.
  - **Paso 2.**El desarrollador implementa la aplicación como un **contenedor**, para que pueda ejecutarse fácilmente en cualquier entorno apropiado.
  - **Paso 3.**El desarrollador implementa ese contenedor en un proveedor informático sin servidor. Esta implementación incluye una especificación de cuánto tiempo debe permanecer inactiva la función antes de que se desactive.
  - **Paso 4.**Cuando es necesario, la aplicación llama a la función.
  - **Paso 5.**El proveedor activa una instancia del contenedor, realiza la tarea necesaria y devuelve el resultado.

Modelos de Implementación (Cont.)

Solo incurre en costos cuando se ejecuta la aplicación. Si la aplicación sin servidor no es necesaria, no se está ejecutando y no se le cobrará por ella. Debido a que la capacidad aumenta y baja con la necesidad, generalmente se denomina "elástica" en lugar de "escalable".



Spin down=centrifugar
Spìng up = girar

Ejemplo aplicación sin servidor: AWS Lambda

Aws Lambda es de Amazon.
https://docs.aws.amazon.com/es_es/lambda/latest/dg/welcome.html

Lambda es un servicio informático que permite ejecutar código sin aprovisionar ni administrar servidores. Lambda ejecuta el código en una infraestructura informática de alta disponibilidad y realiza todas las tareas de administración de los recursos informáticos, incluido el mantenimiento del servidor y del sistema operativo, el aprovisionamiento de capacidad y el escalado automático, así como la monitorización del código y las funciones de registro.

Con Lambda, puede ejecutar código para prácticamente cualquier tipo de aplicación o servicio de backend. Lo único que tiene que hacer es suministrar el código en uno de los lenguajes que admite Lambda (node, python, java, c#, …). Se usa contenedores (Docker).

https://en.wikipedia.org/wiki/AWS_Lambda

https://towardsdatascience.com/aws-lambda-amazon-api-gateway-not-as-daunting-as-they-sound-part-1-d77b92f53626

Tipos de infraestructura

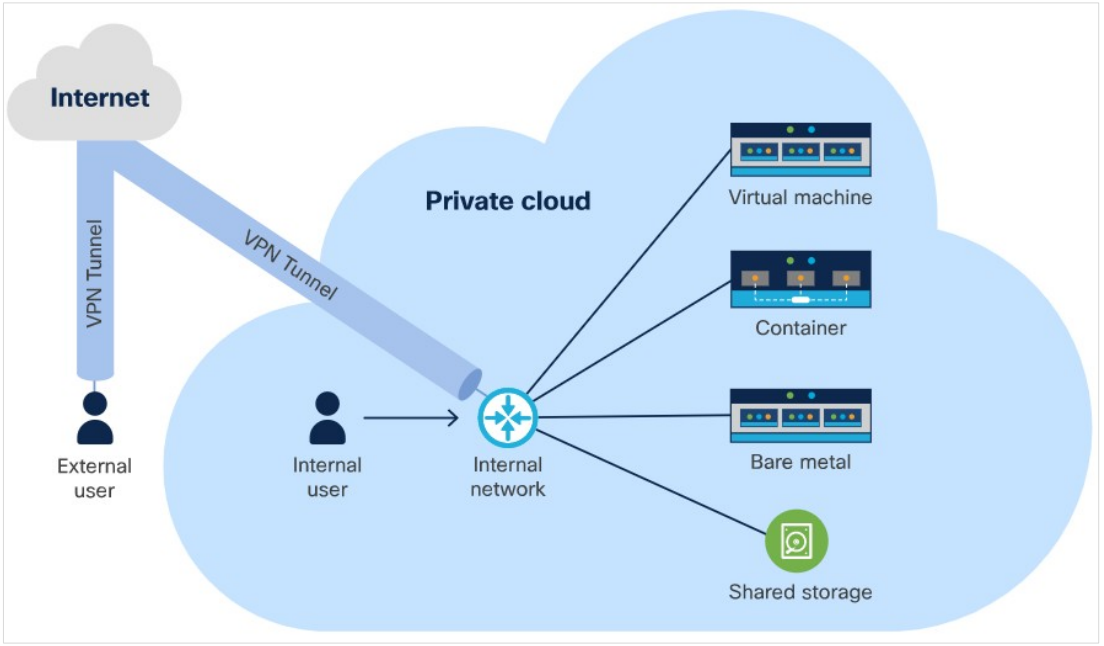
- En los primeros días de las computadoras, la infraestructura era bastante sencilla. El software se ejecutaba en un solo equipo y las redes podían vincular varios equipos.
- El modelo de **computación sin servidor** significa que tiene control cero sobre el equipo host, por lo que puede que no sea apropiado desde una perspectiva de seguridad.
- Ahora, la infraestructura se ha vuelto más complicada, con varias opciones disponibles para diseñar la infraestructura, como diferentes tipos de nubes, y lo que cada una hace y no funciona bien.

Local

- Local significa cualquier sistema que esté literalmente dentro de los límites del edificio.
- Los centros de datos locales son los centros de datos tradicionales que albergan máquinas individuales que se aprovisionan para aplicaciones, en lugar de nubes.
- Estos centros de datos tradicionales con servidores dedicados a aplicaciones individuales, o a máquinas virtuales, que permiten que un solo equipo actúe como varios equipos.
- El funcionamiento de un centro de datos local tradicional requiere que los servidores, los dispositivos de almacenamiento y los equipos de red se soliciten, reciban, ensamblen en racks, se muevan a una ubicación y se cableen para la alimentación y los datos. Toda esta configuración de infraestructura requiere tiempo y esfuerzo.
- Los problemas relacionados con las instalaciones locales se pueden resolver moviendo a una solución basada en la nube.

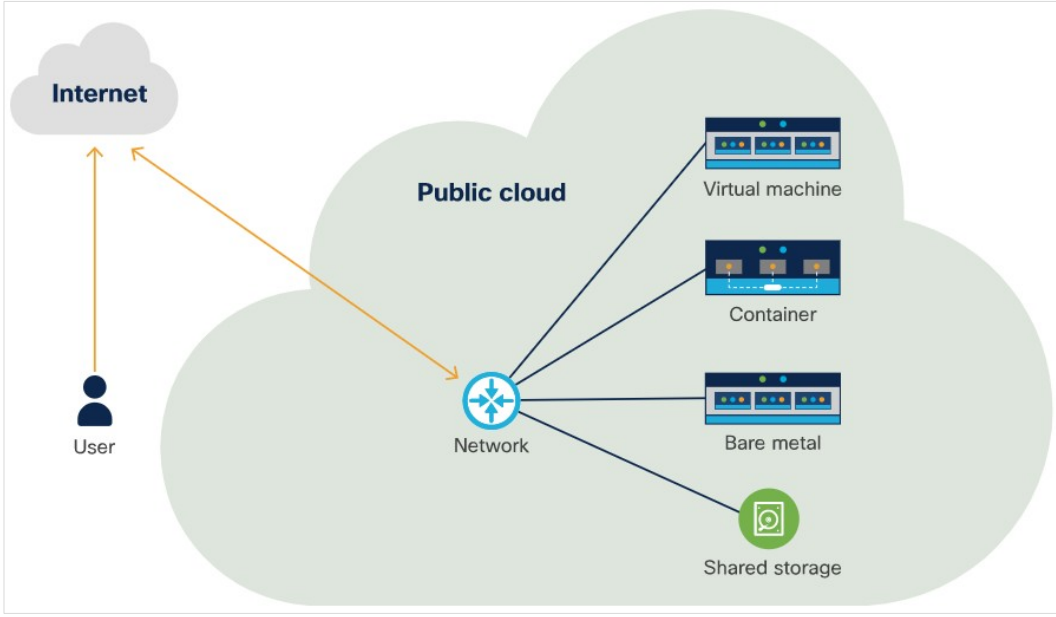
Nube Privada

- Una nube es un sistema que proporciona aprovisionamiento de autoservicio para recursos informáticos, redes y almacenamiento.
- En una infraestructura de nube privada, la organización controla todos los recursos.
- En la mayoría de los casos, una nube privada se encuentra en un centro de datos y todos los recursos que se ejecutan en el hardware pertenecen a la organización propietaria.
- La ventaja de una nube privada es que uno tiene un control completo sobre dónde se encuentra.
- Se requiere un equipo de operaciones para administrar la nube y mantenerla en funcionamiento.



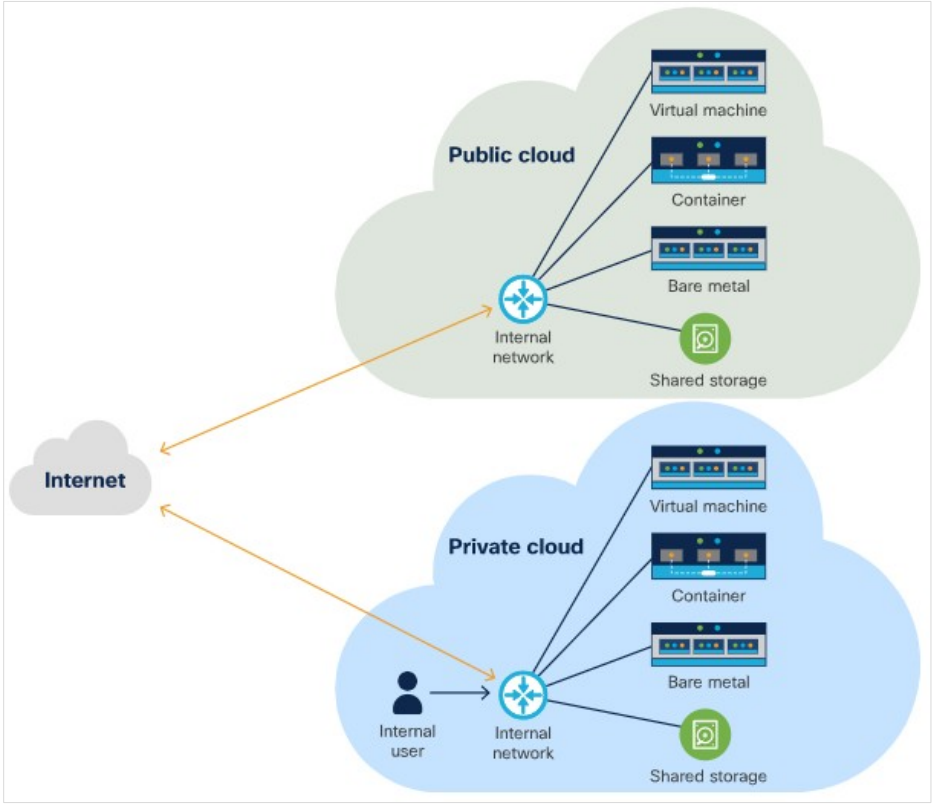
Nube Pública

- Una nube pública es lo mismo que una nube privada, pero es administrada por un proveedor de nube pública.
- Los clientes de nube pública pueden compartir recursos con otras organizaciones. Alternativamente, los proveedores de nube pública pueden proporcionar a los clientes una infraestructura dedicada.
- Con una nube pública, la organización no controla los recursos.
- Una nube pública es útil para escalar prácticamente todo el tiempo que la carga lo requiera y luego escalar hacia abajo cuando el tráfico es lento.
- Una desventaja de la nube pública se conoce como el problema del "vecino ruidoso (Noisy Neighbor)".



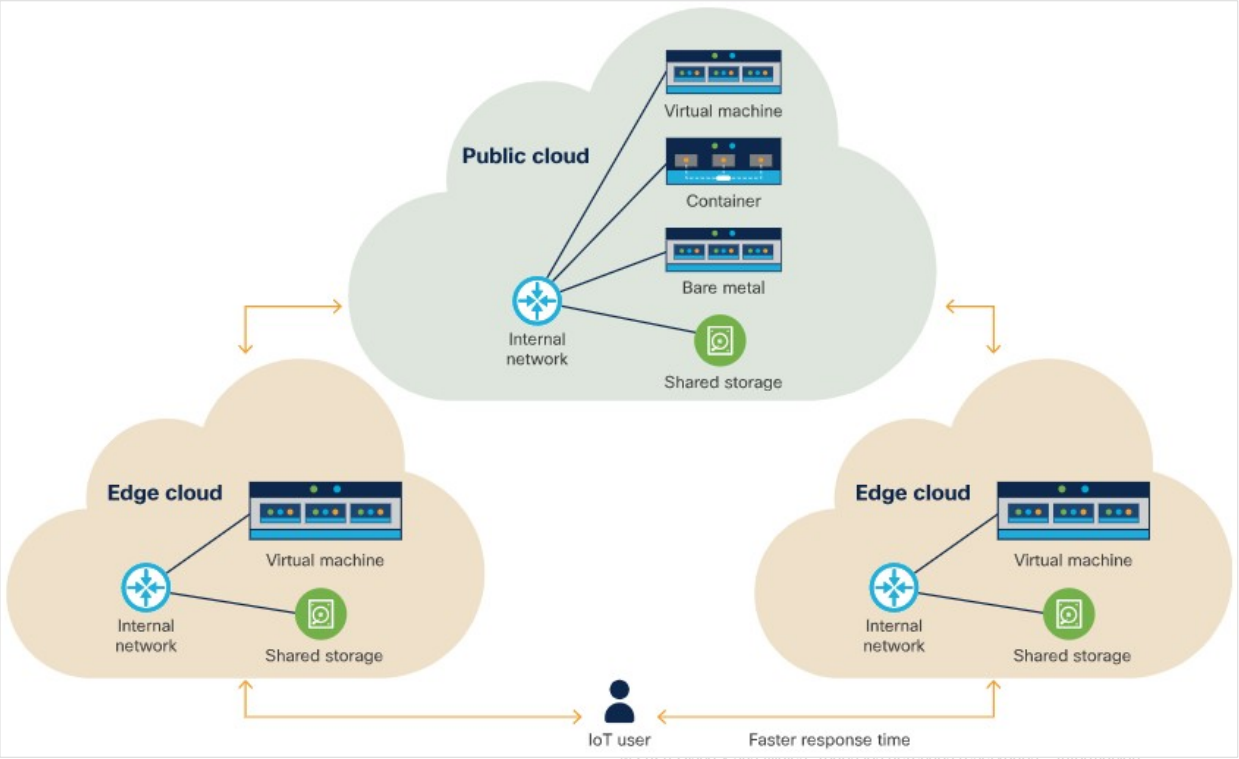
Nube Híbrida

- La nube híbrida es la combinación de dos tipos diferentes de nubes.
- La nube híbrida se utiliza para establecer un puente entre una nube privada y una nube pública dentro de una sola aplicación.
- La nube híbrida combina la nube pública y privada para proporcionar recursos adicionales y seguridad cuando sea necesario.
- La nube híbrida se distingue por el uso de más de una nube dentro de una sola aplicación.
- Los orquestadores de contenedores se han vuelto muy populares entre las empresas que utilizan implementaciones de nube híbrida.



Nube Perimetral (Edge)

- La nube perimetral (Edge) está ganando popularidad debido al crecimiento del Internet de las cosas (IoT).
- La nube perimetral permite que los recursos estén más cerca de donde se necesitan.
- La computación en nube perimetral comprende una o más nubes centrales que actúan como un centro para las propias nubes de borde.
- El hardware para las nubes de borde se encuentra lo más cerca posible del usuario.
- La nube perimetral se ejecuta en hardware mucho más pequeño, por lo que puede tener más recursos limitados.



6.2 Creación e implementación de una aplicación de muestra

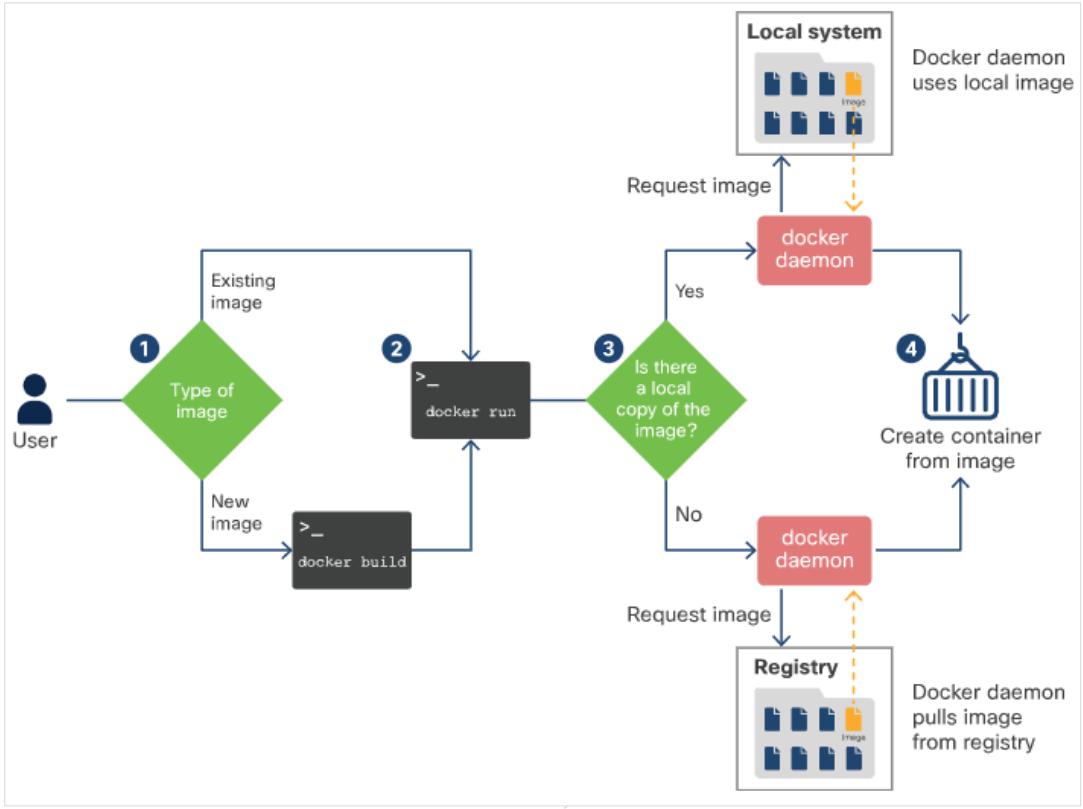
¿Qué es Docker?

- La forma más popular de **contenerizar** una aplicación es implementarla como un contenedor **Docker**.
- Un contenedor es una forma de encapsular todo lo que necesita para ejecutar su aplicación, de modo que pueda implementarse fácilmente en una variedad de entornos. Docker es una forma de crear y ejecutar ese contenedor.
- Docker es un formato que envuelve una serie de tecnologías diferentes para crear contenedores. Estas tecnologías son:
  - Espacios de nombres (Namespaces) : aíslan diferentes partes del contenedor en ejecución.
  - Grupos de control (Control Groups) – Estos grupos son un concepto estándar de Linux que permite al sistema limitar los recursos utilizados por una aplicación.
  - Sistemas de Unión de Archivos (Union File Systems) – Estos sistemas de unión de archivos (UnionFS) son sistemas de archivos que se crean capa por capa, combinando recursos.
- Una **imagen** Docker es un conjunto de archivos de sólo lectura que no tienen estado y contiene código fuente, bibliotecas y otras dependencias necesarias para ejecutar una aplicación.
- Un **contenedor** Docker es la instancia en tiempo de ejecución de una imagen Docker.
- La creación de un contenedor implica extraer una imagen o una plantilla de un repositorio y, a continuación, utilizarla para crear un contenedor.

¿Qué es Docker? (Cont.)

El flujo de trabajo para crear un contenedor es el siguiente:

- Paso 1:Crear una nueva imagen usando **docker build** o extraiga una copia de una imagen existente de un registro usando docker pull.
- Paso 2:Ejecutar un contenedor basado en la imagen usando **docker run** o docker container create.
- Paso 3:El docker daemon comprueba si tiene una copia local de la imagen. Si no la tiene, extrae la imagen del registro.
- Paso 4:El Docker daemon crea un contenedor basado en la imagen y, si se utilizó docker run , inicia sesión en él y ejecuta el comando solicitado.



¿Qué es un Dockerfile?

- **Dockerfile** es un archivo de texto simple que se requiere para compilar el código.
- Definir los pasos que toma el comando de **docker build** para crear una imagen que se puede utilizar para crear el contenedor de destino.

Pasos para generar un Dockerfile que crea un contenedor Ubuntu:

- Crear un archivo denominado Dockerfile y guardarlo en el directorio actual.
- Ejecutar el comando de docker build para construir la imagen usando un Dockerfile en el directorio actual (.) y darle el nombre de myubuntu.

```
devasc@labvm:~$ docker build -t myubuntu:latest .
Sending build context to Docker daemon  983.3MB
Step 1/1 : FROM ubuntu:latest
latest: Pulling from library/ubuntu
692c352adcf2: Pull complete
97058a342707: Pull complete
2821b8e766f4: Pull complete
4e643cc37772: Pull complete
Digest: sha256:55cd38b70425947db71112eb5dddfa3aa3e3ce307754a3df2269069d2278ce47
Status: Downloaded newer image for ubuntu:latest
 ---> adafef2e596e
Successfully built adafef2e596e
Successfully tagged myubuntu:latest
devasc@labvm:~$
```

¿Qué es un Dockerfile? (Cont.)

- Introduzca el comando **docker images** para ver su imagen en la lista de imágenes en la máquina virtual de DEVASC.

```
devasc@labvm:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
myubuntu            latest              adafef2e596e        3 days ago          73.9MB
ubuntu              latest              adafef2e596e        3 days ago          73.9MB
devasc@labvm:~$
```

- Cambie al directorio principal y escriba Is para ver que está vacío y listo para usar.
- Introducir exit para salir del contenedor Docker y volver al sistema operativo principal de DEVASC VM.

```
devasc@labvm:~$ docker run -it myubuntu:latest /bin/sh# lsbin  boot  dev  etc  home  lib  lib32  lib64
libx32  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var# cd home# ls##
exitdevasc@labvm:~$
```

Anatomía de un Dockerfile

- Considere el siguiente Dockerfile que contiene una aplicación Python y la explicación de los comandos son los siguientes:
  - El comando FROM instala Python en la imagen Docker.
  - El comando WORKDIR indica a Docker que use /home/ubuntu como directorio de trabajo.
  - El comando COPY le indica a Docker que copie el archivo del directorio actual de Dockerfile en /home/ubuntu.
  - El comando RUN permite ejecutar comandos directamente en el contenedor.
  - El comando CMD iniciará el servidor cuando el usuario ejecute el contenedor real.
  - El comando EXPOSE le dice a Docker que el usuario desea exponer el puerto 8080.

```
FROM python
WORKDIR /home/ubuntu
COPY ./sample-app.py /home/ubuntu/.
RUN pip install flask
CMD python /home/ubuntu/sample-app.py
EXPOSE 8080
```

Anatomía de un archivo Dockerfile (Cont.)

- Docker aprovecha lo que se almacena en la memoria caché para acelerar el proceso.
- El comando docker build se usa para compilar la imagen. En la salida dada, la imagen se construyó previamente.
- El Docker pasa por cada paso en el archivo Dockerfile, comenzando con la imagen base, Python. Si esta imagen no existe en el sistema, Docker la extrae del Registro. El registro predeterminado es Docker Hub.
- Entre pasos como ejecutar un comando, Docker crea un nuevo contenedor y compila una imagen intermedia, una nueva capa, guardando ese contenedor.

```
$ docker build -t sample-app-image .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM python
 ---> 0a3a95c81a2b
Step 2/6 : WORKDIR /home/ubuntu
 ---> Using cache
 ---> 17befcf89bab
Step 3/6 : COPY ./sample-app.py /home/ubuntu/.
 ---> Using cache
 ---> c0b3a4f9c568
Step 4/6 : RUN pip install flask
 ---> Using cache
 ---> 8cf8226c9f31
Step 5/6 : CMD python /home/ubuntu/sample-app.py
 ---> Running in 267c5d569356
Removing intermediate container 267c5d569356
 ---> 75cd4bf1d02a
Step 6/6 : EXPOSE 8080
 ---> Running in cc82eaca2028
Removing intermediate container cc82eaca2028
 ---> 9616439582f8
Successfully built 9616439582f8
Successfully tagged sample-app-image:latest
$
```

Iniciar un contenedor Docker localmente

- Después de crear la **imagen** usando dockerfile, creamos un nuevo **contenedor** y realizamos un trabajo ingresando el comando docker run.
- El parámetro -d es la abreviatura de -detach e indica que la imagen debe ejecutarse en segundo plano.
- El parámetro -P indica a Docker que lo publique en el puerto que se expuso.

```
$ docker run -d -P sample-app-image
1688a2c34c9e7725c38e3d9262117f1124f54685841e97c3c5225af88e30bfc5
$
```

Observe los procesos de publicación del contenedor:

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                   NAMES
90edd03a9511        sample-app-image    "/bin/sh -c 'python …"   5 seconds ago       Up 3 seconds
0.0.0.0:32774->8080/tcp   jovial_sammet
$
```

Iniciar un contenedor Docker localmente (Cont.)

- Observe que Docker ha asignado al contenedor un nombre como pythontest. La nomenclatura también se realiza con la opción -name.

```
docker run -d -P --name pythontest sample-app-image
```

- Aunque el contenedor está escuchando en el puerto 80, es solo un puerto interno. Docker ha especificado un puerto externo como 32774, que se reenviará al puerto interno.
- Esto le permite ejecutar varios contenedores que escuchan en el mismo puerto sin tener conflictos.
- Para extraer el sitio web de la aplicación de muestra, utilice la dirección IP pública del servidor host y ese puerto se utiliza.

```
$ curl localhost:32774
You are calling me from 172.17.0.1
$
```

Iniciar un contenedor Docker localmente (Cont.)

- Docker también permite especificar un puerto particular para reenviar, de modo que se pueda crear un sistema más predecible.

```
$ docker run -d -p 8080:8080 --name pythontest sample-app-image
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                    NAMES
a51da037bf35        sample-app-image    "/bin/sh -c 'python …"   28 seconds ago      Up 27 seconds
0.0.0.0:8080->8080/tcp    pythontest
90edd03a9511        sample-app-image    "/bin/sh -c 'python …"   24 minutes ago      Up 24 minutes
0.0.0.0:32774->8080/tcp   jovial_sammet
$
```

- Cuando el contenedor se está ejecutando, iniciar sesión en él se puede ejecutar usando el comando exec

```
$ docker exec -it pythontest /bin/sh
# whoami
root
# pwd
/var/www/html
# exit
$
```

Iniciar un contenedor Docker localmente (Cont.)

- Para detener y eliminar un contenedor en ejecución, llamarlo por su nombre:

```
$ docker stop pythontest
pythontest
$ docker rm pythontest
pythontest
$
```

- Observar de nuevo los procesos en ejecución y el contenedor en ejecución se ha eliminado.

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                    NAMES
90edd03a9511        sample-app-image    "/bin/sh -c 'python …"   25 minutes ago      Up 25 minutes
0.0.0.0:32774->8080/tcp   jovial_sammet
$
```

Guardar una imagen de Docker en un Registro

- Para que la imagen esté disponible para los usuarios, guárdela en un registro de imágenes.
- De forma predeterminada, Docker utiliza el registro Docker Hub, pero los usuarios también pueden crear su propio registro. Para iniciar el proceso:
- Iniciar sesión en el registro.

```
$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head
over to https://hub.docker.com to create one.
Username: devnetstudent # This would be your username
Password:               # This would be your password
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
$
```

Guardar una imagen Docker en un Registro (Cont.)

- Confirmar el contenedor en ejecución con el comando docker commit

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                    NAMES
54c44606344c        sample-app-image    "/bin/sh -c 'python …"   4 seconds ago       Up 2 seconds
0.0.0.0:8080->8080/tcp    pythontest
$ docker commit pythontest sample-app
Sha256:bddc326383032598a1c1c2916ce5a944849d90e4db0a34b139eb315af266e68b
$
```

  - Utiliazar el comando Docker tag para dar a la imagen la etiqueta que se ha confirmado.

```
<repository>/<imagename>:<tag>
```

- La primera parte, el repositorio, suele ser el nombre de usuario de la cuenta que almacena la imagen. El siguiente es el nombre de la imagen y, finalmente, la etiqueta opcional.

```
$ docker tag sample-app devnetstudent/sample-app:v1
$
```

Guardar una imagen Docker en un Registro (Cont.)

- Ahora la imagen está lista para ser enviada al repositorio.

```
$ docker push devnetstudent/sample-app:v1
The push refers to repository [docker.io/nickchase/sample-app]
e842dba90a43: Pushed
868914f88a69: Pushed
c7d71f6230b3: Pushed
1ed9b15dd229: Pushed
00947a3aa859: Mounted from library/python
7290ddeeb6e8: Mounted from library/python
d3bfe2faf397: Mounted from library/python
cecea5b3282e: Mounted from library/python
9437609235f0: Mounted from library/python
bee1c15bf7e8: Mounted from library/python
423d63eb4a27: Mounted from library/python
7f9bf938b053: Mounted from library/python
f2b4f0674ba3: Mounted from library/python
v1: digest: sha256:28e119f43e9c8e5e44f167d9baf113cc91d4f8b461714cd6bb578ebb0654f243 size: 3052
$
```

- Observar que la nueva imagen se almacena localmente.

```
$ docker images
REPOSITORY              TAG                 IMAGE ID            CREATED              SIZE
sample-app              latest              bddc32638303        About a minute ago   410MB
devnetstudent/sample-app   v1                  bddc32638303        About a minute ago   410MB
$
```

Crear un entorno de desarrollo

- El entorno de desarrollo está destinado a ser conveniente para el desarrollador. Solo necesita coincidir con el entorno de producción donde sea relevante.
- Un entorno de desarrollo puede consistir en cualquier cantidad de herramientas, desde Entornos de Desarrollo Integrados (Integrated Development Environments, IDE) hasta bases de datos y almacenamiento de objetos, como Eclipse, hasta bases de datos y almacenamiento de objetos. La parte importante aquí es que tiene que ser cómodo para el desarrollador.

6.2.7.Laboratorio - Compilar una Aplicación Web de Muestra en un Contenedor Docker

- En esta práctica de laboratorio se cumplirán los siguientes objetivos:
  - **Parte 1**: Iniciar DEVASC VM
  - **Parte 2**: Crear un Bash Script Simple
  - **Parte 3**: Crear una Aplicación Web de Muestra
  - **Parte 4**:Configurar la Aplicación Web para Usar Archivos Web
  - **Parte 5**:Crear un Bash Script para Compilar y Ejecutar un Contenedor Docker
  - **Parte 6**: Compilar, Ejecutar y Verificar el Contenedor Docker

6.3 Integración continua/Implementación continua (CI/CD)

Introducción a CI/CD

- Integración continua/implementación continua (CI/CD) es una filosofía para la implementación de software que ocupa un lugar destacado en el campo de DevOps.
- DevOps se trata de la comunicación y asegurarse de que todos los miembros del equipo están trabajando juntos para garantizar un funcionamiento sin problemas.

Integración continua

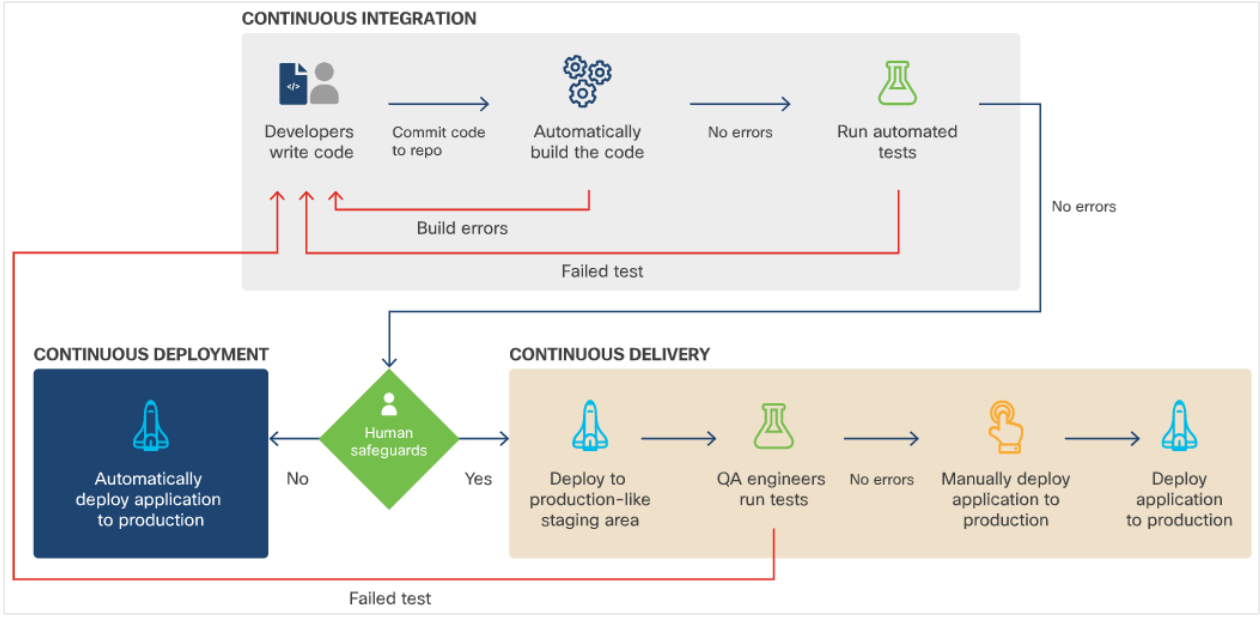
- La integración continua permite a los desarrolladores del proyecto combinar continuamente los cambios con la rama principal de la aplicación existente.
- El proceso de integración continua proporciona una serie de ventajas adicionales:
  - Compilación de código
  - Ejecutar las pruebas unitarias
  - Análisis de código estático
  - Pruebas de integración
  - Embalaje y control de versiones
  - Publicar el paquete de versión en Docker Hub u otros repositorios de paquetes

Integración continua (Cont.)

Entrega continua

Es el proceso de desarrollo en lapsos cortos para que el código esté siempre en un estado implementable. Involucra los siguientes pasos:

- **Paso 1:** Comenzar con el artefacto de versión creado como parte del proceso de CI.
- **Paso 2:** Implementar automáticamente la versión candidata en el ensayo.
- **Paso 3:** Ejecutar pruebas de gating identificadas por el equipo u organización.
- **Paso 4:** Si todas las pruebas de ajuste pasan, etiquetar esta compilación como adecuada para la producción.



Integración continua (Cont.)

Implementación continua

- La implementación continua es la expresión definitiva de CI/CD.
- Es un tipo especial de Entrega Continua en el que, cada versión de software que está marcada como lista para la producción se implementa.

Evitar el impacto en los usuarios

Para evitar afectar a los usuarios o limitar el impacto, se pueden utilizar estas estrategias de implementación:

- **Actualización sucesiva:** Los cambios se implementan periódicamente de tal manera que no afectan a los usuarios actuales, y nadie debería tener que volver a instalar el software.
- **Canalización Canaria (Canary Pipeline):** La nueva versión se presenta a un subconjunto de usuarios. Si estos usuarios experimentan problemas, los cambios se pueden revertir fácilmente. Si estos usuarios no experimentan problemas, los cambios se implementan en el resto de la producción.
- **Implementación azul-verde:** se crea un entorno completamente nuevo (azul) con el nuevo código en él, pero el entorno antiguo (verde) se mantiene en reserva.

Beneficios de CI/CD

Los beneficios de usar CI/CD para el desarrollo incluyen:

- Integración con metodologías ágiles
- Tiempo medio más corto para la resolución (Mean Time To Resolution, MTTR)
- Implementación automática
- Liberaciones de funciones menos disruptivas
- Calidad mejorada
- Mejor tiempo de salida al mercado

Ejemplo de trabajo de compilación para Jenkins

- Las canalizaciones de implementación se crean normalmente con una herramienta de compilación como Jenkins. Estas canalizaciones pueden manejar tareas como recopilar y compilar código fuente, probar y compilar artefactos como archivos tar u otros paquetes.

Ejemplo de trabajo de compilación para Jenkins

- La unidad fundamental de Jenkins es el proyecto, también conocido como el trabajo. Los trabajos se crean para hacer todo tipo de cosas, desde recuperar código de un repositorio de administración de código fuente hasta crear una aplicación usando un script o herramienta de compilación, empaquetarlo y ejecutarlo en un servidor.

Ejemplo de trabajo de compilación para Jenkins (Cont.)

- Para crear un trabajo simple que recupere una versión de la aplicación de ejemplo de GitHub y ejecute el script de compilación, realicemos los pasos que se enumeran a continuación:
  - **Paso 1**: Crear un nuevo elemento en la interfaz de Jenkins haciendo clic en el enlace "crear nuevos trabajos" en la página de bienvenida.
  - **Paso 2**: Introducir un nombre, elegiremos Proyecto Freestyle (para que tenga la mayor flexibilidad) y haga clic en **Aceptar**.
  - **Paso 3**: Desplazarse hacia abajo hasta administrador de código fuente (Source Code Management) y seleccionar Git, luego ingresar una URL del repositorio de GitHub para la URL del repositorio.
  - **Paso 4**: Desplazarse hacia abajo hasta Compilar (Build) y hacer clic en **Agregar paso de compilación (Add Build Step)**. Elegir **Ejecutar** shell.
  - **Paso 5**: En el cuadro de comandos, agregar el comando: buildscript.sh
  - [redacted]
  - [redacted]

Ejemplo de trabajo de compilación para Jenkins (Cont.)

Para crear un segundo trabajo que pruebe la compilación para asegurarse de que funciona correctamente, realizaremos los siguientes pasos:

- **Paso 1**: Hacer clic en el enlace Jenkins y Nuevo elemento para iniciar un nuevo trabajo, luego crear otro trabajo de estilo libre llamado TestAppJob.
- **Paso 2**: Esta vez, deje la Administración de código fuente como Ninguno (None).r Pero hay una opción para establecer un desencadenador de compilación para que este trabajo se ejecute justo después del trabajo anterior, BuildAppJob.
- **Paso 3**: Desplazarse hacia abajo y una vez más agregue un paso de compilación de script de shell Execute.
- **Paso 4**: Agregar la siguiente secuencia de comandos (scripts) como el comando, utilizando la dirección IP de un servidor Jenkins de muestra y compruebe si una condición se devuelve como verdadera.

```
if [ "$(curl localhost:8000/test)" = "You are calling me from 172.17.0.1" ]; then
   exit 0
else
   exit 1
fi
```

Laboratorio de integración continua/implementación continua (CI/CD): cree una canalización de CI/CD usando Jenkins

- En esta práctica de laboratorio se cumplirán los siguientes objetivos:
  - **Parte 1**: Iniciar DEVASC VM
  - **Parte 2**: Confirmar la aplicación de prueba a Git
  - **Parte 3**: Modificar la aplicación de prueba y enviar los cambios a Git
  - **Parte 4**: Descargar y ejecutar la imagen de Jenkins Docker
  - **Parte 5**: Configurar Jenkins
  - **Parte 6**: Usar Jenkins para ejecutar una compilación de la aplicación
  - **Parte 7**: Usar Jenkins para probar una compilación
  - **Parte 8**: Crear una canalización en Jenkins

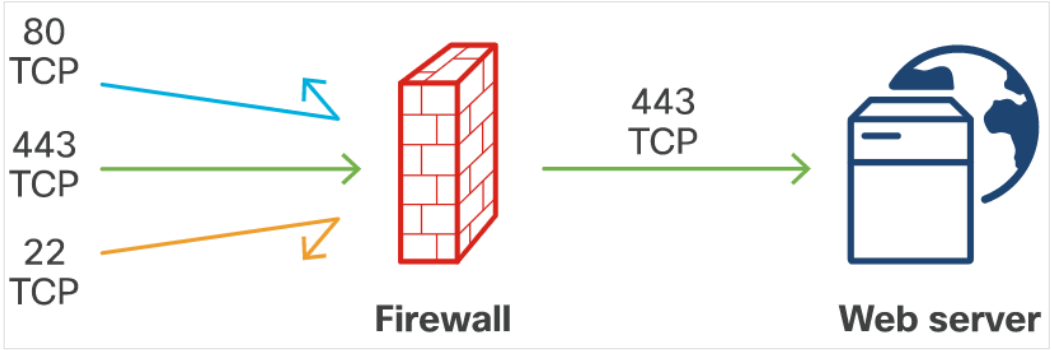
6.4 Redes para el desarrollo de aplicaciones y seguridad

Introducción

- Cuentas de red para todos los casos de uso, excepto los más simples, como las implementaciones en la nube y contenedores.
- A continuación se presentan algunas de las aplicaciones que deben tenerse en cuenta para la implementación en la nube:
  - Firewalls
  - Equilibradores de carga
  - Sistema de Nombres de Dominio (Domain Name Service, DNS)
  - Proxies Inversos

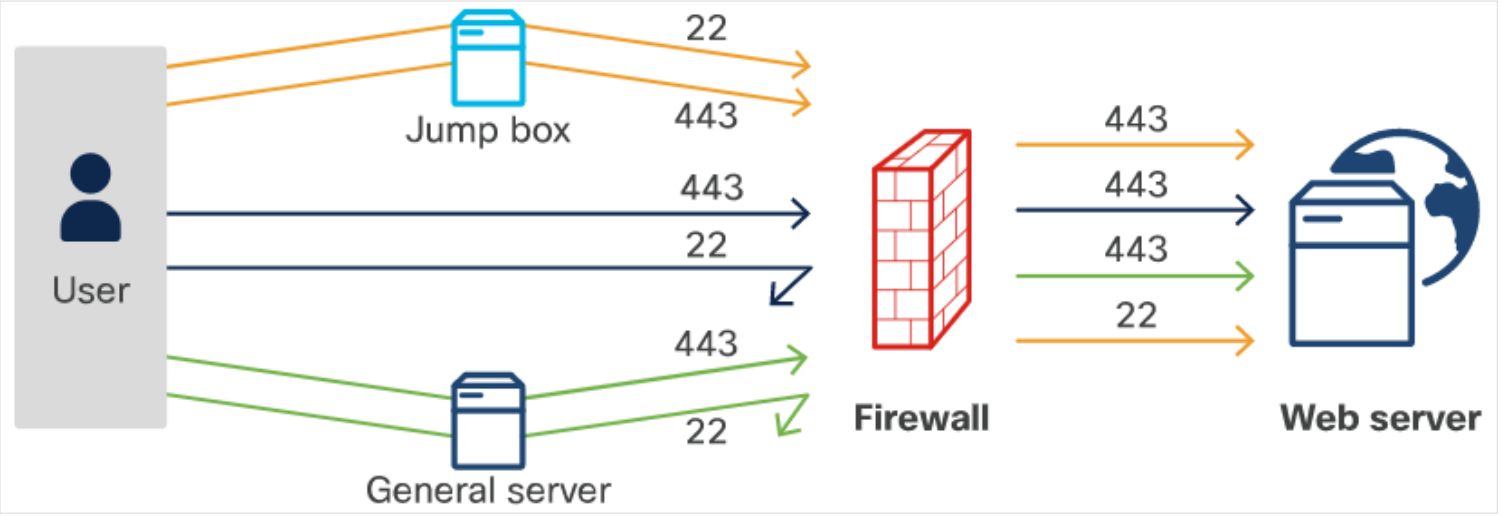
Cortafuegos (Firewall)

- Los firewalls son la defensa más básica de un equipo contra el acceso no autorizado de personas o aplicaciones. Pueden adoptar cualquier número de formas. Desde un dispositivo de hardware dedicado hasta una configuración dentro del sistema operativo de un equipo individual.
- En su nivel más básico, un firewall acepta o rechaza paquetes basados en las direcciones IP y los puertos a los que se dirigen.
- Los firewalls se pueden configurar con "reglas" específicas, que están en capas una encima de la otra.
- Un firewall puede permitir algunas conexiones y rechazar otras.



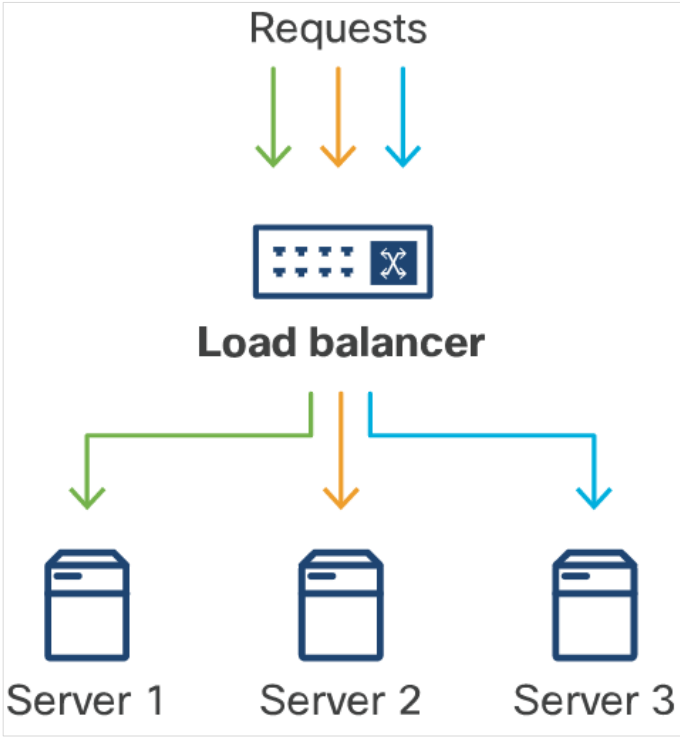
Cortafuegos (Firewall) (Cont.)

- En algunos casos, puede configurar los sistemas para que los inicios de sesión en sistemas confidenciales solo puedan provenir de un único equipo. Esto se llama una "caja de salto (Jumpbox)".
- Una caja de salto se puede utilizar para proporcionar acceso adicional mientras proporciona una capa adicional de seguridad. Establece los sistemas para que los inicios de sesión solo puedan venir de una sola máquina y todos deben iniciar sesión en ese servidor primero y, a continuación, iniciar sesión en el equipo de destino desde allí.



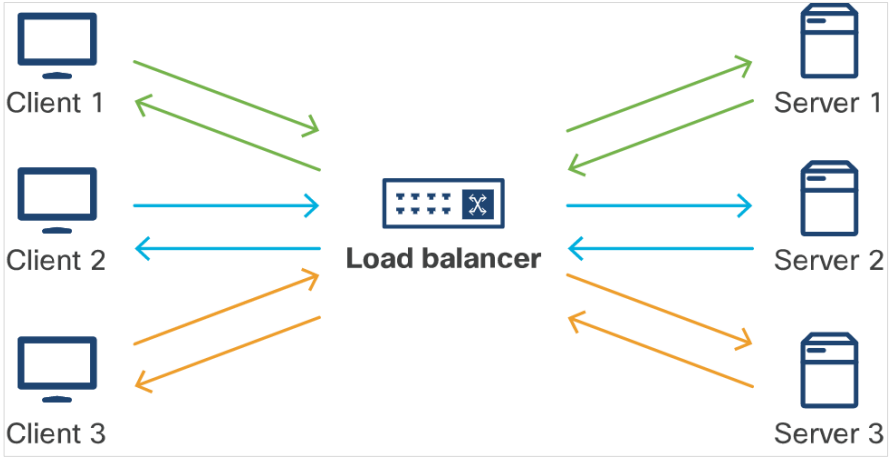
Equilibrador de carga

- Un equilibrador de carga toma solicitudes y las equilibra distribuyéndolas entre varios servidores.
- Un equilibrador de carga envía solicitudes a diferentes servidores.
- Los equilibradores de carga toman sus decisiones sobre qué servidores deben obtener una solicitud particular de varias maneras diferentes.

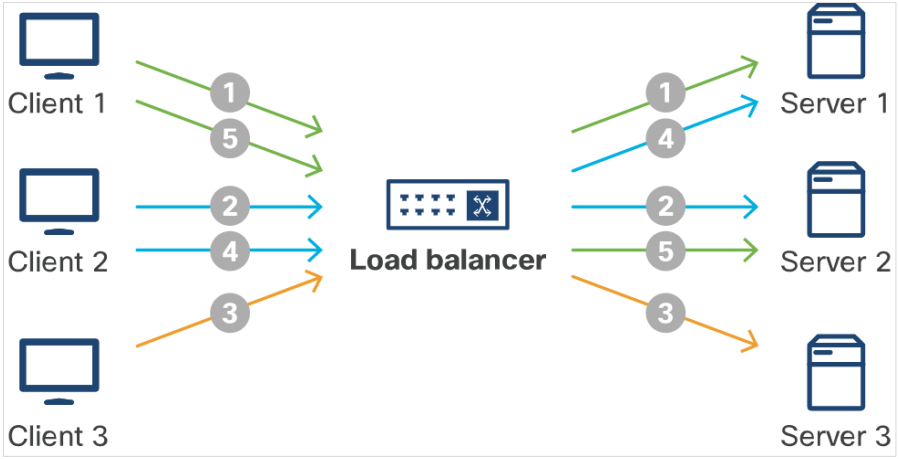


Equilibrador de carga (Cont.)

Sesiones persistentes: si una aplicación requiere una sesión persistente, un usuario debe iniciar sesión y el equilibrador de carga enviará solicitudes al servidor que gestiona la sesión.

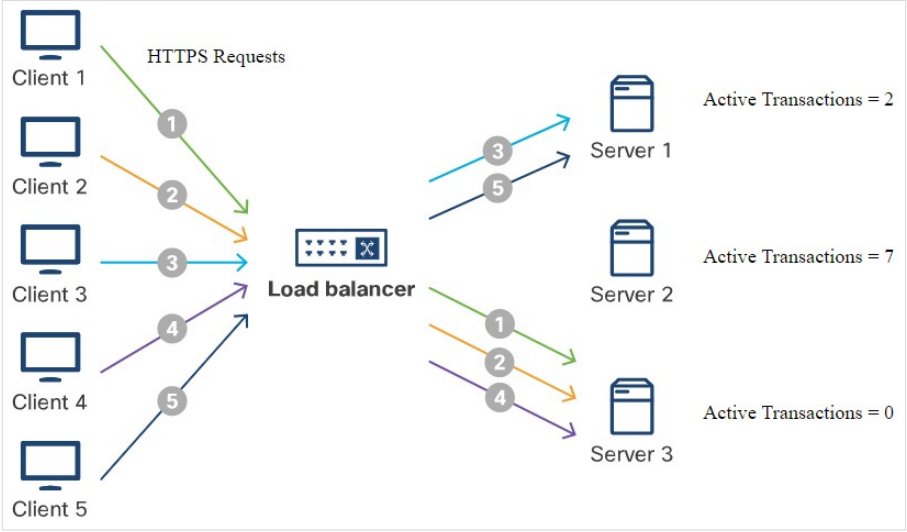


Round robin - Con el equilibrio de carga de round robin, el servidor envía cada solicitud al siguiente servidor de la lista.

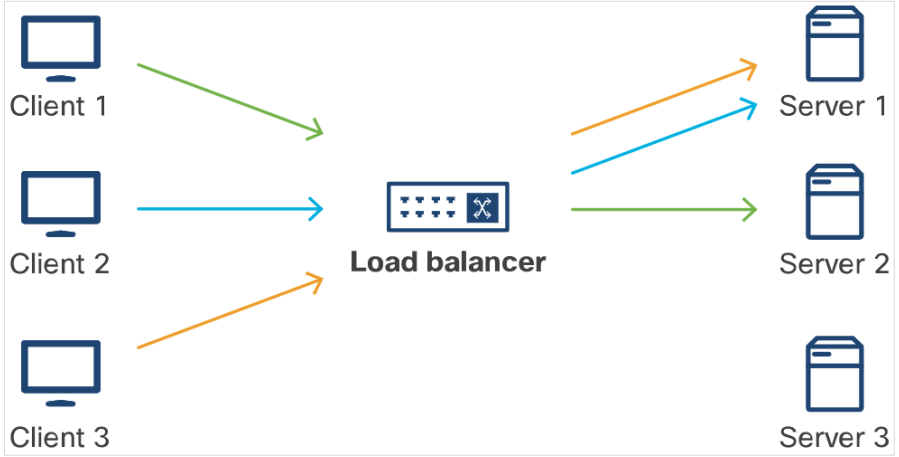


Equilibrador de carga (Cont.)

Menos conexiones: el equilibrador de carga envía la solicitud al servidor que está menos ocupado, el menor número de conexiones activas.

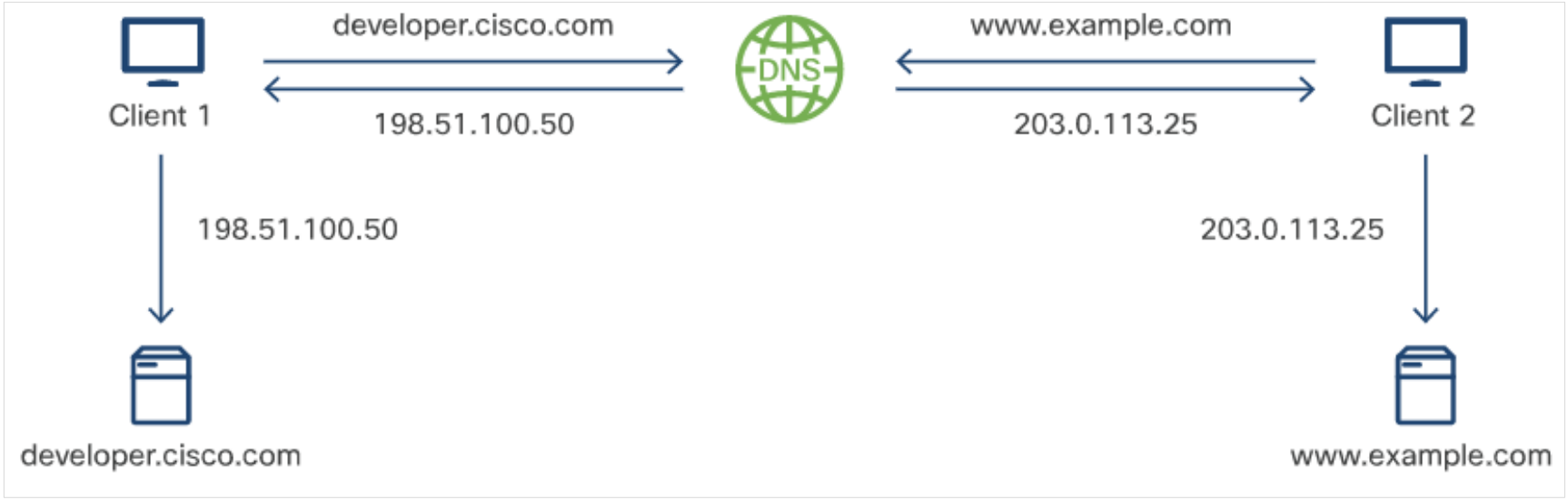


Hash IP - Con este algoritmo, el equilibrador de carga toma una decisión basada en un hash (un valor codificado basado en la dirección IP de la solicitud).



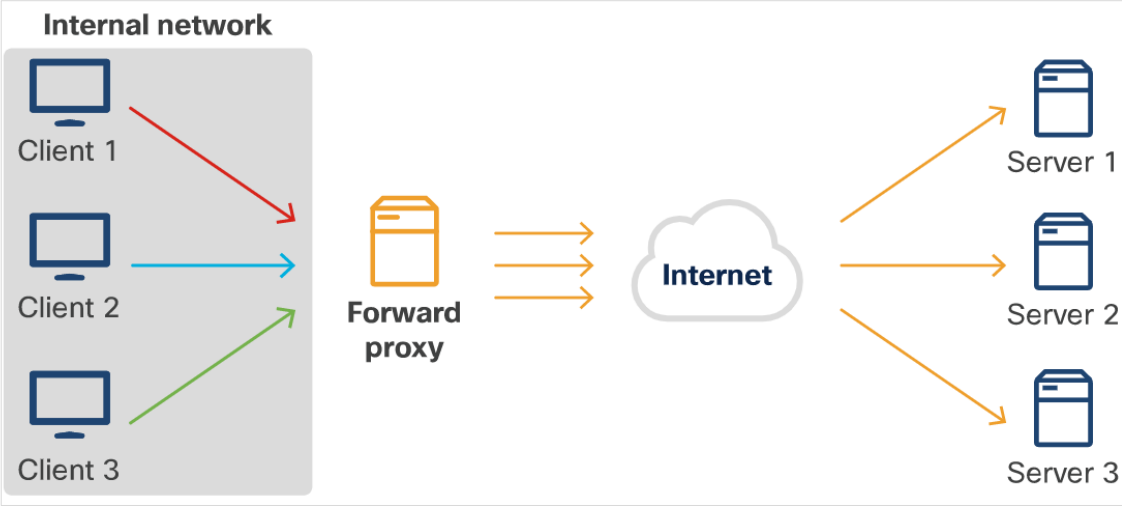
DNS

- El Sistema de nombres de dominio (DNS) proporciona una forma para que los servidores de Internet traduzcan nombres legibles por humanos en direcciones IP enrutables por máquina. Estas direcciones IP son necesarias para navegar realmente por Internet.
- DNS traduce nombres de host en direcciones IP (inventadas).



Proxy Inverso

- Un proxy inverso es similar a un proxy normal, sin embargo, mientras que un proxy regular trabaja para hacer que las solicitudes de varios equipos parezcan que todas provienen del mismo cliente, un proxy inverso trabaja para asegurarse de que las respuestas parezcan que todas provienen del mismo servidor.
- Un proxy inverso puede evaluar el tráfico y actuar en consecuencia. De esta manera, es similar a, y se puede utilizar como, un firewall o un equilibrador de carga.



6.5 Protección de Aplicaciones

Protección de los datos

Prácticas recomendadas para almacenar datos cifrados

Las violaciones de datos se producen cuando los datos se almacenan pero no se protegen. Cuando se trata de proteger los datos en reposo, hay algunas cosas a considerar.

Encriptar datos

El cifrado de datos garantiza que cuando se obtiene un acceso no autorizado al sistema, los datos no son visibles en su forma real. Existen dos métodos para cifrar datos:

Cifrado unidireccional (One-way encryption)	Cifrado bidireccional (Two-way encryption)
El cifrado unidireccional es más simple, ya que puede crear fácilmente un valor cifrado sin necesidad de usar una clave específica, pero no puede descifrarlo. Se usaría para la información que no necesita recuperar, solo necesita comparar, como contraseñas.	El cifrado bidireccional, encripta los datos usando una clave y, a continuación, puede usar esa clave (o una variación en ella) para descifrar los datos y recuperarlos en texto sin formato. Se usaría para información a la que necesitaría acceder en su forma original, como registros médicos o números de seguridad social.

Protección de los datos (Cont.)

Vulnerabilidades de software

- La mayoría de los desarrolladores no son expertos en seguridad y pueden codificar accidentalmente vulnerabilidades de seguridad en la aplicación. Asegúrese de que alguien de la organización sea responsable de mantenerse al día con las vulnerabilidades más recientes y parches según corresponda.

Almacenar demasiados datos

- A menos que los datos sean necesarios para una función esencial, no debemos almacenarlos.

Almacenamiento de datos en la nube

- Recordemos que al almacenar datos en la nube, se almacenan en la computadora de otra persona. Asegurémonos de que los datos de la nube estén cifrados o protegidos de otro modo.

Dispositivos Itinerantes (Roaming Devices)

- Las aplicaciones están cada vez más en dispositivos que incluso más portátiles que las laptops, como tabletas y especialmente teléfonos móviles. Simplemente son más fáciles de perder. Asegúrese de que no está dejando sus datos vulnerables encriptándolos siempre que sea posible.

Protección de los datos (Cont.)

Prácticas recomendadas para el transporte de datos

Los datos también son vulnerables cuando se transmiten. Se puede utilizar lo siguiente para evitar problemas de vulnerabilidad de datos:

- **SSH** - SSH proporciona autenticación y cifrado de mensajes entre las máquinas de origen y destino, lo que hace difícil o imposible husmear (snooping) las acciones de los usuarios.
- **TLS** : TLS proporciona autenticación de mensajes y cifrados más fuertes que SSL.
- **VPN** : una VPN mantiene todo el tráfico relacionado con las aplicaciones dentro de la red, que actúa como proxy y cifra todo el tráfico hacia y desde el usuario.

¿Qué es la inyección SQL?

- La inyección SQL es una técnica de inyección de código que se utiliza para atacar aplicaciones controladas por datos, en la que las instrucciones SQL malintencionadas se insertan en un campo de entrada para su ejecución.
- La inyección SQL aprovecha una vulnerabilidad de seguridad en el software de una aplicación.Este ataque permite a los atacantes suplantar identidad, alterar los datos existentes, permitir la revelación completa de todos los datos del sistema, destruir los datos o hacer que no estén disponibles de otro modo, y convertirse en administradores del servidor de base de datos.

SQL en páginas Web

- La inyección SQL es una de las técnicas de piratería (hacking) web más comunes. Es la colocación de código malicioso en sentencias SQL, a través de la entrada de página web.
- Ocurre cuando se le pide entrada a un usuario, como nombre de usuario/ID de usuario, y en su lugar el usuario da una instrucción SQL que se ejecuta sin saberlo en la base de datos.

¿Qué es la inyección SQL? (Cont.)

- En este ejemplo se crea una instrucción SELECT agregando una variable uid a una cadena de selección. La variable se obtiene de la entrada del usuario usando request.args ("uid").

```
uid = request.args("uid");
str_sql = "SELECT * FROM Users WHERE UserId = " + uid;
```

- La inyección SQL basada en 1=1 siempre es verdadera. Crear una instrucción SQL para seleccionar el perfil de usuario por UID, con un UID de UserProfile dado.
- Si no hay validador de entrada para evitar que un usuario ingrese una entrada «incorrecta», el usuario puede introducir alguna entrada como UID: 2019 OR 1=1
- La sentencia SQL de salida será:

```
SELECT * FROM UserProfiles WHERE UID = 2019 OR 1=1;
```

¿Qué es la inyección SQL? (Cont.)

- La instrucción SQL anterior es válida, pero devolverá todas las filas de la tabla UserProfiles, porque OR 1=1 siempre es TRUE.
- Si la tabla UserProfiles contiene nombres, correos electrónicos, direcciones y contraseñas, la instrucción SQL será:

```
SELECT UID, Address, Email, Name, Password FROM UserProfiles WHERE UID = 2019 or 1=1;
```

- Un creador de malware o un hacker puede obtener acceso a todos los perfiles de usuario de la base de datos, simplemente escribiendo 2019 OR 1=1 en el campo de entrada.

¿Qué es la inyección SQL? (Cont.)

Inyección SQL basada en sentencias SQL por lotes

- La mayoría de las bases de datos admiten instrucciones SQL por lotes. Un lote de sentencias SQL es un grupo de dos o más sentencias SQL, separadas por punto y coma.
- La instrucción SQL siguiente devolverá todas las filas de la tabla UserProfiles y, a continuación, eliminará la tabla UserImages.

```
SELECT * FROM UserProfiles; DROP TABLE UserImages
```

¿Cómo detectar y prevenir la inyección SQL?

La vulnerabilidad de inyección SQL existe porque algunos desarrolladores no se preocupan por la validación de datos y la seguridad. Hay herramientas que pueden ayudar a detectar defectos y analizar código.

- **Herramientas de código abierto**: para detectar fácilmente un ataque de inyección SQL, los desarrolladores han creado buenos motores de detección como SQLMap o SQLNinja.
- **Herramientas de análisis de** código fuente: Las herramientas de análisis de código fuente, también conocidas como herramientas de prueba de seguridad de aplicaciones estáticas (Static Aplication Security Testing, SAST), están diseñadas para analizar código fuente y/o versiones compiladas de código para ayudar a encontrar defectos de seguridad como desbordamientos de búfer, defectos de inyección SQL y otros.
- **Trabajar con un firewall de base de**datos: los firewalls de base de datos detectan inyecciones SQL en función del número de consultas no válidas de un host, mientras que hay bloques OR y UNION dentro de la solicitud, u otros.

¿Cómo detectar y prevenir la inyección SQL? (Cont.)

Usar sentencias preparadas

- Las instrucciones preparadas con enlace de variables, también conocidas como consultas parametrizadas, son utilizadas por los desarrolladores para escribir consultas de base de datos. Las consultas parametrizadas obligan al desarrollador a definir primero todo el código SQL y, a continuación, pasar el parámetro a la consulta.
- Las instrucciones preparadas garantizan que un atacante no puede cambiar la intención de una consulta, incluso si los comandos SQL son insertados por un atacante.

```
// Get customer's name from parameter
String              custname = request.getParameter("custname");
// Perform input validation to detect attacks
String              query = "SELECT account_balance FROM user_data WHERE user_name = ? ";
PreparedStatement   pStatement = connection.prepareStatement( query );
pStatement.setString( 1, custname);
ResultSet           results = pStatement.executeQuery();
```

¿Cómo detectar y prevenir la inyección SQL? (Cont.)

Usar procedimientos almacenados

- Los procedimientos almacenados no siempre están a salvo de la inyección SQL.
- La diferencia entre las instrucciones preparadas y los procedimientos almacenados es que el código SQL de un procedimiento almacenado se define y almacena en la propia base de datos y, a continuación, se llama desde la aplicación.
- Ambas técnicas tienen la misma eficacia en la prevención de la inyección SQL.

Validación de entrada de la lista blanca

- Varias partes de consultas SQL no son ubicaciones legales para el uso de variables de enlace.
- En tales situaciones, la validación de entrada o el rediseño de consultas es la defensa más adecuada.
- Si los valores de parámetros de usuario se utilizan para focalizar diferentes nombres de tabla y columnas, los valores de parámetro deben asignarse a los nombres de tabla o columna legal/esperados para garantizar que la entrada de usuario no validada no termine en la consulta.

¿Cómo detectar y prevenir la inyección SQL? (Cont.)

Escapar toda la entrada proporcionada por el usuario

- Esta técnica sólo debe utilizarse como último recurso cuando ninguna de las técnicas es factible e implica escapar la entrada del usuario antes de ponerla en una consulta.
- El Escaping funciona de tal manera que cada DBMS admite uno o más esquemas de escape de caracteres específicos para ciertos tipos de consultas.
- Hay bibliotecas y herramientas utilizadas para Escape de entrada.
- Las bibliotecas ESAPI facilitan a los programadores la adaptación de la seguridad a las aplicaciones y sirven de base sólida.

¿Cómo detectar y prevenir la inyección SQL? (Cont.)

Defensas adicionales

Para proporcionar defensa en profundidad, se pueden adoptar estas defensas adicionales:

- **Privilegio mínimo**: los privilegios asignados a cada cuenta de base de datos deben minimizarse para reducir el daño potencial de un ataque de inyección SQL exitoso. Minimizar los privilegios reducirá los intentos de acceso no autorizados, incluso cuando un atacante no intente usar la inyección SQL como parte de su explotación (exploit).
- **Varios usuarios de bases de datos**: **los diseñadores de aplicaciones**
- **Vistas SQL**: las vistas SQL se utilizan para aumentar aún más el detalle de acceso limitando el acceso de lectura a campos específicos de una tabla o combinaciones de tablas.

Proteger la aplicación

¿Qué es OWASP?

Open Web Application Security Project (OWASP) se centra en proporcionar educación, herramientas y otros recursos para ayudar a los desarrolladores a evitar problemas de seguridad en aplicaciones basadas en web. Los recursos proporcionados por OWASP incluyen:

Herramientas	Proyectos de código	Proyectos de Documentación
• OWASP Zed Attack Proxy (ZAP) • Comprobación de dependencia • OWASP DefectDojo	• OWASP ModSecurity Core Rule Set (CRS) • OWASP CSRFGuard	• OWASP Appplication Security Verification Standard • OWASP Top Ten • OWASP Cheat Sheet Series

Scripting Entre Sitios (Cross-Site Scripting, XSS)

- Los ataques de scripting entre sitios ocurren cuando el contenido enviado por el usuario que no se ha "desinfectado" se muestra a otros usuarios.
- La versión más obvia de este exploit es donde un usuario envía un comentario que incluye un script que realiza una acción maliciosa y cualquiera que vea la página de comentarios tiene ese script ejecutado en su equipo.
- Hoy en día, el mayor problema es que los usuarios están tratando con más que los datos que se almacenan en la base de datos, o "Ataques XSS almacenados". Por ejemplo, considere esta página, que muestra el contenido de un parámetro de solicitud:

```
...
<h1>Search results for {{ request.args['search_term'] }}</h1>
{ for item in cursor }
…
```

- Un hacker podría engañar a alguien para que visite la página con un enlace en un correo electrónico que proporcione código malicioso en un parámetro:

```
http://www.example.com?search_term=%3Cscript%3Ealert%28%27Gotcha%21%27%29%3C%2Fscript%3E
```

Scripting Entre Sitios (Cross-Site Scripting, XSS) (Cont.)

- Este enlace, que incluye una versión codificada por URL del script, daría como resultado que un usuario desprevenido viera una página de:

```
...
<h1>Search results for <script>alert('Gotcha!')</script></h1>
...
```

- Esto se llama un **Ataque XSS Reflejado**. Para evitar un ataque XSS reflejado**,** la estrategia principal es desinfectar el contenido cuando sea posible y, si no se puede desinfectar, no lo muestre.
- OWASP recomienda nunca mostrar contenido que no sea de confianza en las siguientes ubicaciones:
  - Etiquetas de script internas
  - Comentarios internos
  - Como parte de los nombres de atributo
  - Como parte de los nombres de etiquetas
  - En CSS (dentro de etiquetas de estilo)

Scripting Entre Sitios (Cross-Site Scripting, XSS) (Cont.)

- El contenido se puede mostrar en algunas ubicaciones, si se desinfecta primero. Estas ubicaciones incluyen:
  - Contenido de una etiqueta HTML
  - Valor de un atributo
  - Variable dentro de Javascript
- Desinfectar contenido puede ser un proceso complicado de hacer correctamente, ya que hay una amplia variedad de opciones que tiene un atacante.

Falsificación de Solicitudes Entre Sitios (Cross-Site Request Forgery, CSRF)

- Otro tipo de ataque que comparte algunos aspectos de los ataques XSS es la falsificación de solicitudes entre sitios (CSRF), a veces llamado "Sea Surf".
- En ambos casos, el atacante pretende que el usuario ejecute el código del atacante, generalmente sin siquiera saberlo.
- La principal diferencia es que los ataques CSRF generalmente no están dirigidos al sitio de destino, sino más bien a un sitio **diferente**, uno en el que el usuario ya se ha autenticado.
- Un aspecto interesante de CSRF es que el atacante nunca obtiene los resultados del ataque. Sólo pueden juzgar los resultados después del hecho, y tienen que ser capaces de predecir cuáles serán los efectos para aprovechar un ataque exitoso.

Falsificación de Solicitudes Entre Sitios (Cross-Site Request Forgery, CSRF) (Cont.)

- Un método para evitar ataques CSRF es incluir un token oculto que debe acompañar cualquier solicitud del usuario.

```
...
<form action="https://greatbank.example.com" method="POST">
Username: <input type="text" name="username" style="width: 200px" />
Password: <input type="text" name="password" style="width: 200px" />
<br >
<input type="hidden" name="CSRFToken" value="d063937d-c117-46e6-8354-6f5d8faff095" />
<input type="submit" value="Log in">
</form>
…
```

- Ese Token CSRF tiene que acompañar cada solicitud del usuario para que se considere legítimo, ya que es imposible para el atacante predecir ese token.

El Top Ten de OWASP

La lista de ataques de OWASP incluye:

- **Inyección**: incluye todo tipo de ataques de inyección que se pueden prevenir mediante el uso de API parametrizadas, el escape de la entrada del usuario y el uso**de**cláusulas LIMIT.
- **Autenticación rota**: se relaciona con varios problemas con las credenciales de usuario. Estos ataques se pueden prevenir evitando contraseñas predeterminadas, utilizando la autenticación multifactor y utilizando técnicas como la prolongación de los períodos de espera después de inicios de sesión fallidos.
- **Exposición de datos sensibles**: se refiere a escenarios en los que los atacantes roban información confidencial. Estos escenarios pueden evitarse almacenando la menor cantidad posible de información personal y utilizando el cifrado.
- **Entidades externas XML (XXE)**: Se trata de ataques que han hecho posible una característica XML que permite incorporar información externa mediante entidades, y se pueden evitar deshabilitando el procesamiento de Entidades XML y DTD, o utilizando el formato JSON.
- **Control de acceso roto**: Esto se refiere a la necesidad de garantizar que una aplicación que permita a los usuarios eludir los requisitos de autenticación existentes no se debe crear y se puede evitar protegiendo todos los recursos y funciones del lado del servidor.

El Top Ten de OWASP (Cont.)

- **Configuración incorrecta de seguridad:** Esto se refiere a la necesidad de asegurarse de que el sistema en sí está configurado correctamente. La prevención de este tipo de problemas requiere un endurecimiento cuidadoso y coherente de los sistemas y las aplicaciones.
- **Scripting entre sitios (XSS):** Se refiere a la capacidad de un atacante de utilizar las funciones dinámicas de un sitio para inyectar contenido malintencionado en la página. Estos ataques se pueden prevenir considerando cuidadosamente dónde incluir el contenido que no es de confianza, así como desinfectando cualquier contenido que no sea de confianza.
- **Deserialización insegura:** Describe los problemas que pueden producirse si los atacantes pueden acceder a versiones serializadas de datos y objetos y, potencialmente, cambiarlas. Para evitar estos problemas, no acepte objetos serializados de orígenes que no sean de confianza.
- **Uso de componentes con vulnerabilidades conocidas:** La mayoría de las funciones principales probablemente se hayan escrito e incluido en un paquete de software existente, y probablemente sea de código abierto. Muchos de los paquetes que están disponibles también incluyen vulnerabilidades disponibles públicamente. Para solucionar esto, asegúrese de que estén usando solo las características necesarias y paquetes seguros.
- **Registro y monitoreo insuficientes:** Es importante asegurarse de que los registros estén en un formato común para que puedan ser consumidos fácilmente por las herramientas de informes y que sean auditables para detectar manipulaciones.

Evolución de los Sistemas de Contraseñas

Contraseñas simples de texto sin formato

- Las primeras contraseñas eran simples contraseñas de texto sin formato que permitían que varios usuarios que utilizaban el mismo procesador principal tuvieran una configuración de privacidad única.
- El texto plano es una forma insegura de almacenar contraseñas. Si la base de datos fue pirateada, las contraseñas del usuario se expondrían directamente a los hackers.

Hashing de contraseñas

- Almacenar contraseñas es arriesgado y complejo al mismo tiempo.
- Un enfoque simple para almacenar contraseñas es crear una tabla en la base de datos que asigne un nombre de usuario con una contraseña.
- La fortaleza de seguridad y la resistencia de este modelo dependen del formato de almacenamiento de contraseñas, que es texto claro.
- Almacenar contraseñas en texto claro equivale a escribirlas en un pedazo de papel digital. Si un atacante entra en la base de datos y roba la tabla de contraseñas, el atacante podría acceder a cada cuenta de usuario.

Evolución de los Sistemas de Contraseñas (Cont.)

Hashing

- Hash es una forma más segura de almacenar una contraseña en la que se transforma en datos que no se pueden convertir de nuevo a la contraseña original.
- Como indica OWASP, las funciones hash utilizadas en la criptografía tienen las siguientes propiedades clave:
  - Es fácil y práctico calcular el hash, pero difícil o imposible volver a generar la entrada original si solo se conoce el valor hash.
  - Es difícil crear una entrada inicial que coincida con una salida deseada específica.

Contraseña Salt

- Para garantizar la singularidad de las contraseñas, aumente su complejidad, se agrega un Salt, que es simplemente datos aleatorios, a la entrada de una función hash.

Evolución de los Sistemas de Contraseñas (Cont.)

Uso de hash criptográfico para un almacenamiento de contraseñas más seguro

- Una propiedad crítica que hace que las funciones hash sean adecuadas para el almacenamiento de contraseñas es que son deterministas.
- Una función determinista es una función que, dada la misma entrada, siempre produce la misma salida. Esto es vital para la autenticación porque uno necesita tener la garantía de que una contraseña dada siempre producirá el mismo hash. De lo contrario, sería imposible verificar consistentemente las credenciales de usuario con esta técnica.

Agregar el Salt al hash de contraseñas

- Se agrega un Salt al proceso de hash para forzar la unicidad de hash, lo que aumenta la complejidad sin aumentar los requisitos del usuario y mitigar los ataques de contraseña como las tablas del arcoíris.
- El hash único producido al agregar el Salt puede proteger contra diferentes vectores de ataque, mientras ralentiza los ataques de diccionario y de fuerza bruta.

Evolución de los Sistemas de Contraseñas (Cont.)

Mitigar los ataques de contraseña con un Salt

- Para mitigar el daño que podría causar una tabla arcoíris o un ataque de diccionario, agregamos un Salt a las contraseñas.
- De acuerdo con las directrices de OWASP, una Salt es un valor aleatorio criptográficamente fuerte de longitud fija que se agrega a la entrada de funciones hash para crear hashes únicos para cada entrada, independientemente de si la entrada es única.
  - Digamos que tiene la contraseña devnet_password1 y la Salt salt706173776f726473616c74a
  - Puede Salt esa contraseña agregando o anteponiendo la Salt a ella.

Evolución de los Sistemas de Contraseñas (Cont.)

Factores adicionales para la autenticación

La incorporación de otros factores de autenticación confunde a los hackers que pueden haber descifrado la contraseña. Algunos de estos factores son los siguientes:

Autenticación de un solo factor (SFA)

- La autenticación de un solo factor es la forma más simple de métodos de autenticación, utilizando los cuales, una persona coincide con una credencial para verificarse a sí misma en línea. El ejemplo más popular de esto sería una contraseña (credencial) de un nombre de usuario.
- SFA tiene sus riesgos, ya que los sitios en línea pueden tener contraseñas de los usuarios filtradas por un hacker. Un usuario malintencionado puede adivinar la contraseña ya que conoce al usuario personalmente, o pudo averiguar ciertas cosas sobre el usuario.
- Un usuario malintencionado también puede descifrar la contraseña mediante el uso de un bot para generar la combinación correcta de letras y números para que coincida con el método de identificación simple y secreta de los usuarios.

Evolución de los Sistemas de Contraseñas (Cont.)

Autenticación de dos factores (2FA)

- La autenticación de dos factores (Two-factor authentication) utiliza la misma combinación de contraseña/nombre de usuario, pero con la adición de que se le pide que verifique la identidad de las personas mediante el uso de algo de su propiedad, como un dispositivo móvil.

Autenticación multi factor (MFA)

- La autenticación multifactor (Multi-Factor Authentication, MFA) es un método de control de acceso al equipo en el que solo se concede acceso al usuario después de presentar con éxito varias pruebas separadas a un mecanismo de autenticación.
- Se requieren al menos dos de las categorías mencionadas para el AMF: conocimiento, posesión e herencia.
- 2FA es sólo un tipo de MFA donde sólo se necesitan dos pruebas, dos "factores".

Descifrado de Contraseñas

Las técnicas para encontrar una contraseña que permita la entrada se conocen como descifrar la seguridad prevista por la contraseña. Las siguientes son algunas de las técnicas:

Adivinación de contraseña

- La adivinación de contraseñas es una técnica en línea que implica intentar autenticar a un usuario en particular en el sistema.
- Se puede detectar supervisando los registros del sistema de inicio de sesión fallidos.
- Los bloqueos de cuenta se utilizan para evitar que un atacante pueda simplemente adivinar la contraseña correcta al intentar un gran número de contraseñas potenciales.

Ataque de diccionario

- Un ataque de diccionario se basa en probar todas las cadenas de un listado preorganizado, derivado de una lista de palabras como en un diccionario.
- Éstos tienen éxito porque muchas personas tienen una tendencia a elegir contraseñas cortas que son palabras ordinarias o contraseñas comunes.

Descifrado de Contraseñas (Cont.)

Ataque de diccionario pre-calculado o ataque de tabla arcoíris

- Es posible lograr una compensación de tiempo/espacio pre-calculando una lista de hashes de palabras del diccionario, y almacenándolos en una base de datos usando el hash como llave.
- Los ataques de diccionario pre-calculados son efectivos cuando se va a descifrar un gran número de contraseñas.
- Los ataques de diccionario pre-calculados pueden ser frustrados por el uso de sal, una técnica que obliga al diccionario hash a ser recalculado para cada contraseña buscada, haciendo inviable el precálculo, siempre que el número de posibles valores de sal sea lo suficientemente grande.

Ingeniería social

- La ingeniería social para el descifrado de contraseñas implica que una persona convenza o engañe a otra persona para proporcionar acceso al atacante.

Descifrado de Contraseñas (Cont.)

Seis principios clave de la influencia humana

- **Reciprocidad**— Nuestras normas sociales implican que tendemos a devolver un favor cuando se nos pide.
- **Compromiso y consistencia**: cuando las personas se comprometen, ya sea en persona, por escrito o en un sitio web, son más propensas a cumplir ese compromiso con el fin de preservar su propia imagen.
- **Prueba social**— Cuando la gente ve a otra persona haciendo algo, como mirar hacia arriba, otros se detendrán a hacer lo mismo.
- **Autoridad**: este principio de autoridad significa que los atacantes que parecen estar autorizados o representar a una figura de autoridad tienen más probabilidades de obtener acceso.
- **Simpatía**— La gente simpática es capaz de persuadir a los demás de manera más efectiva. Las personas son fácilmente persuadidas por personas familiares que les agradan.
- **Escasez**— Cuando la gente cree que algo está limitado en cantidad, la gente actuará de manera positiva y rápida para recoger el elemento deseado.

Descifrado de Contraseñas (Cont.)

Hay cuatro vectores de ingeniería social, o líneas de ataque, que pueden aprovechar estos principios de influencia.

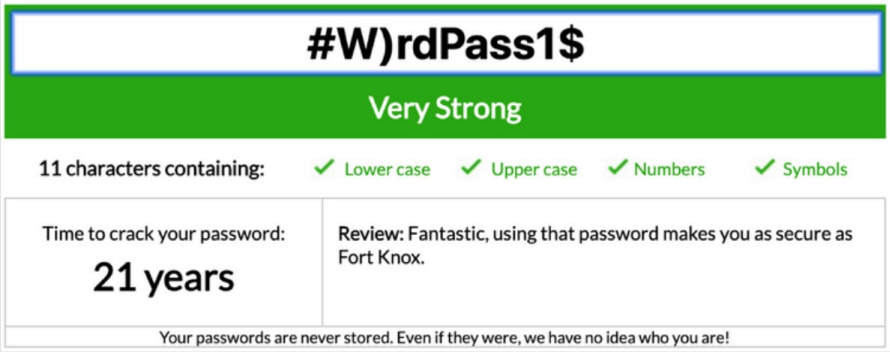
- **Phishing** significa que la persona está obteniendo información fraudulenta, especialmente a través de solicitudes de información financiera. A menudo, los intentos parecen un sitio web real o correo electrónico, pero enlazan a un sitio falso en su lugar.
- **Vishing** significa suplantación de identidad por voz(phishing por voz), por lo que se asocia con llamadas telefónicas de voz para recopilar información personal privada con fines de lucro.
- **Smishing** implica usar mensajes de texto SMS tanto para urgencia como para pedir un curso de acción específico, como hacer clic en un enlace falso o enviar información de la cuenta.
- **La suplantación** implica escenarios presenciales, como usar un uniforme de proveedor de servicios para obtener acceso interno a un edificio o sistema.

Descifrado de Contraseñas (Cont.)

Fortaleza de la contraseña: la fortaleza de la contraseña es la medida de la eficiencia de una contraseña para resistir ataques de desciframiento de contraseñas. La fortaleza de una contraseña está determinada por:

- **Longitud:** Este es el número de caracteres que contiene la contraseña.
- **Complejidad:** Esto significa que utiliza una combinación de letras, números y símbolos.
- **Imprevisibilidad:** Algo que un atacante puede adivinar fácilmente.

Aquí, la contraseña **#W)rdPass1** tiene fuerza y tardaría aproximadamente 21 años en descifrarla.



Descifrado de Contraseñas (Cont.)

Comprobadores de fuerza de contraseña y herramientas de validación

- La herramienta de validación de la fuerza de contraseña está integrada con el sistema de contraseñas para asegurarse de que la contraseña del usuario es compatible con las últimas directrices de administración de identidades.
- Los administradores de contraseñas (password managers) son la herramienta para garantizar la solidez de la contraseña.

Procedimientos recomendados

- Existen algunas prácticas recomendadas para proteger los intentos de inicio de sesión del usuario. Incluye notificar a los usuarios de comportamientos sospechosos, limitando el número de intentos de inicio de sesión de contraseña y nombre de usuario.

Descifrado de Contraseñas (Cont.)

Pautas de identidad digital de NIST

Aquí hay un breve resumen de las Pautas de Identidad Digital NIST 800-63B:

- Un mínimo de 8 caracteres cuando un humano lo establece y un mínimo de 6 caracteres cuando lo establece el sistema/servicio.
- Admite al menos 64 caracteres de longitud máxima y todos los caracteres ASCII.
- El truncamiento de la contraseña no se realizará cuando se procese.
- Comprobar la contraseña elegida con los diccionarios de contraseñas conocidos.
- Permitir al menos 10 intentos de contraseña antes del bloqueo.
- Sin requisitos de complejidad, período de caducidad de la contraseña, sugerencias de contraseña.
- No usar SMS para la autenticación de dos factores, autenticación basada en conocimientos.

Laboratorio: Explorar la Evolución de los Métodos de Contraseña

En este laboratorio, se completarán los siguientes objetivos:

- **Parte 1**: Iniciar DEVASC VM
- **Parte 2**: Explorar el código Python almacenando contraseñas en texto sin formato
- **Parte 3**: Explorar el código Python almacenando contraseñas usando un hash

6.6 Resumen: Implementación de Aplicaciones y Seguridad

¿Qué aprendí en este módulo?

Entender las opciones de implementación con diferentes modelos

- Las grandes organizaciones utilizan una estructura de cuatro niveles: desarrollo, pruebas, ensayo y producción.
- Las opciones para implementar el software son: hardware específico (bare metal), máquinas virtuales, contenedores y computación sin servidor.
- Local-significa cualquier sistema que esté dentro de los límites de su edificio.
- Las nubes proporcionan acceso de autoservicio a recursos informáticos, como máquinas virtuales, contenedores e incluso hardware específico (bare metal).
- La ventaja de una nube privada es que el usuario tiene un control completo sobre dónde se encuentra.
- Una nube pública es esencialmente lo mismo que una nube privada, pero es administrada por un proveedor de nube pública.
- La nube híbrida se utiliza para establecer un puente entre una nube privada y una nube pública dentro de una sola aplicación.
- Una nube perimetral acerca la informática a donde se necesita.

¿Qué aprendí en este módulo? (Cont.)

Crear e Implementar una Aplicación de Muestra

- Un contenedor es una forma de encapsular todo lo necesario para ejecutar la aplicación, de modo que se pueda implementar fácilmente en una variedad de entornos.
- Docker es una forma de crear y ejecutar ese contenedor.
- El entorno de desarrollo está diseñado para ser conveniente para el desarrollador; solo necesita coincidir con el entorno de producción
- Un entorno de desarrollo puede consistir en cualquier número de herramientas, desde IDE hasta bases de datos y almacenamiento de objetos.

¿Qué aprendí en este módulo? (Cont.)

Integración continua/Implementación continua (CI/CD)

- CI/CD es una filosofía para la implementación de software que ocupa un lugar destacado en el campo de DevOps.
- Integración continua de todos los desarrolladores en el proyecto, fusionar continuamente sus cambios con la rama principal de la aplicación existente.
- Una canalización de implementación, se puede crear con una herramienta de compilación como Jenkins.

Redes para Desarrollo de Aplicaciones y Seguridad

- Las aplicaciones que debe tener en cuenta cuando se trata de la implementación en la nube incluyen: firewalls, equilibradores de carga, DNS y proxies inversos.
- En su nivel más básico, un firewall acepta o rechaza paquetes basados en las direcciones IP y los puertos a los que se dirigen.

¿Qué aprendí en este módulo? (Cont.)

Protección de Aplicaciones

- Proteger los datos en dos métodos mediante el cifrado de datos: cifrado unidireccional y cifrado bidireccional.
- La inyección SQL debe aprovechar una vulnerabilidad de seguridad en el software de una aplicación.
- Una forma más segura de almacenar una contraseña es transformarla en datos que no se pueden convertir de nuevo a la contraseña original, conocida como hash.
- Por criptografía, las contraseña se hacen para ser seguras.


CISCO