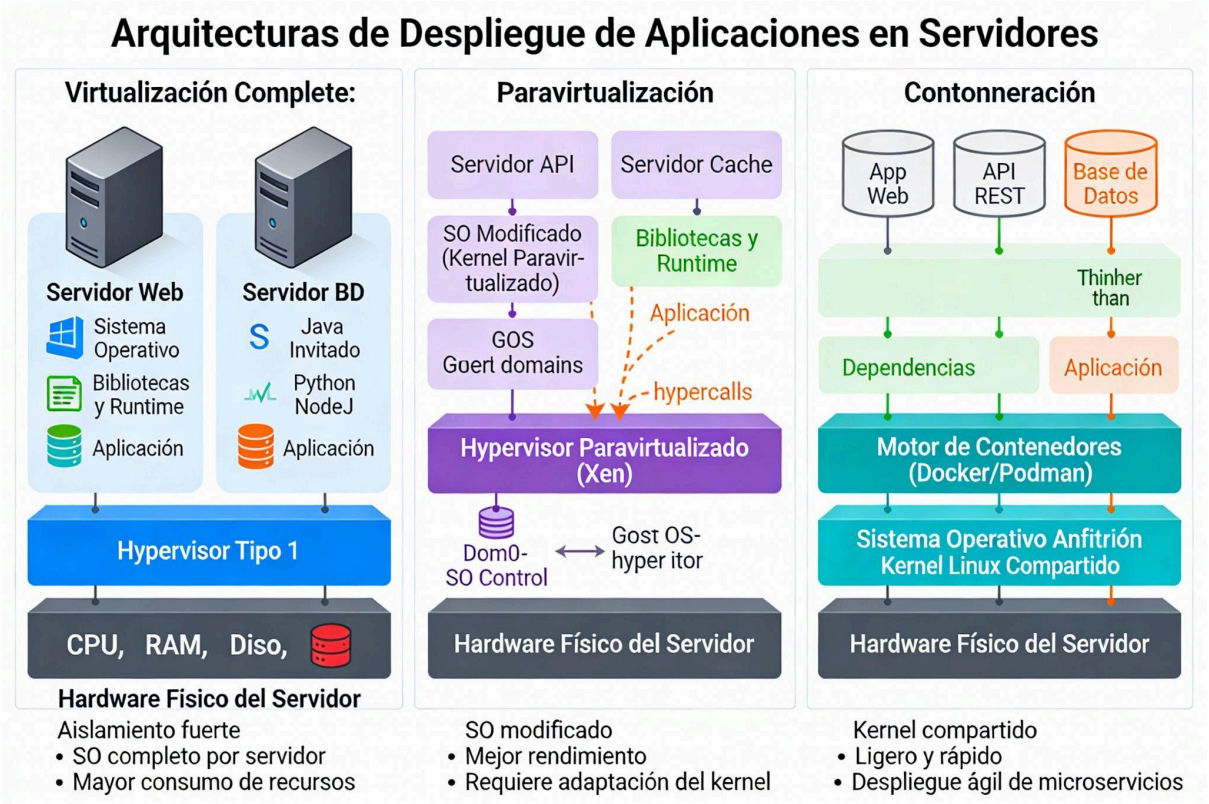
Este documento tiene el fin de brindar un pequeño manual a los estudiantes, con un poco de teoría, prácticas y ejemplos de creación de nubes privadas virtuales. He recopilado información de múltiples documentos reuniendo lo que he considerado las bases y los fundamentos tanto iniciales como básicos para realizar lo que se conoce como “nubes privadas virtuales”. Mucha de esta información es presentada en muchos documentos, sin embargo la idea es recopilar y mantener un manual de operaciones en creación y configuración de nubes privadas.



Dr. Apolinar González Potes

Arquitectura de Servidores

Ingeniería en Computación Inteligente

UNIVERSIDAD DE COLIMA

1. INTRODUCCIÓN

Se entiende por servidores y su arquitectura, el cómo ofrecer servicios a través de una red que generalmente es Internet. Estos servicios no los tienes alojados como cliente y lo recibes a través de lo que se conoce como la nube. Este cloud computing tiene una serie de características que son comunes a todos los servicios que se utilizan a través de estas redes. Una de las principales características es la agilidad, ¿por qué?, porque cuando se requieren estos servicios no sé tiene que montar ninguna infraestructura para poder tenerlos. Es decir se va a la empresa que quiere solicitar estos servicios y a través de un panel que tengan que los van a ofrecer es decir estos servicios que están alojados en otro sitio vienen de forma muy rápida con lo cual es muy ágil de contratar para las empresas. Si se tiene una empresa y se quiere montar una infraestructura se debe tener que hacer un estudio de la infraestructura, hay que tener que contratarla, instalarla y configurarla. Sin embargo en el caso del cloud computing es tan sencillo como solicitar a la empresa y tenerla disponible, así como el costo es mucho más barato.

2. DIFERENTES TIPOS DE SERVICIO

Hay diferentes tipos de servicios que se pueden adquirir a través del cloud computing. Actualmente, empiezan a ser cientos de ellos pero ha habido siempre tres servicios que son los más identificativos y los que más se siguen adquiriendo a día de hoy. Estos tres servicios son IaaS Infrastructure as a Service, PaaS, Platform as a Service y SaaS Software as a Service.

2.1 IaaS.

Significa que lo que se adquiere es una infraestructura que ya está montada para poder tenerla para su uso. Es decir lo que se adquiere como cliente son unos servicios de red unos servicios de servidores, servicios de almacenamiento pero abstraídos es decir al cliente como tal no te importa el hardware, lo que te importa es que ese hardware ofrezca el servicio que se necesita para poder montar un servidor tres servidores 20 servidores una red o diez redes. Lo importante es que ofrezcan el servicio, lo que hay detrás no importa.

2.2 PaaS.

En el caso del Platform as a Service lo que ofreces es una plataforma. Como bien dice el

nombre para el cliente. Esto se suele entender como una plataforma en la que el cliente directamente se olvida del sistema operativo que está corriendo detrás y lo que quiere es directamente trabajar sobre una máquina o sobre un contenedor. Platform as a Service se suele entender como contenedores por servicio. Normalmente se suele utilizar el Platform as a Service para equipos de desarrollo o para contenedores en los que suelen alojarse cosas tipo de desarrollo.

2.3 SaaS.

Y por último tenemos el software as a Service. Este software as a service es el más común y el que todo el mundo a día de hoy demanda. Software as a Service directamente lo que está haciendo es ejecutar una aplicación pero que los servicios están en la nube. Un ejemplo muy claro es por ejemplo Gmail, Hotmail. Lo que se tiene es una aplicación que conecta con los datos que están en la nube pero el cliente no tiene los mails alojados en su teléfono, los datos alojados en la aplicación, cuando se requiere el servicio, se va a la nube se rescata y los muestra.

3. Preparación de un entorno de laboratorio

Otra de las formas importantes de ejecutar ambientes basados en linux sin necesidad de hacer una instalación completa o una virtualización que ocupa muchos recursos, es usando los contenedores. Docker es una alternativa, sin embargo su uso se centra más en usarlo para una aplicación o con docker compose un conjunto de aplicaciones, pero para tener una alternativa más amplia y parecida a una máquina virtual, en mi opinión los contenedores linux LXC son la mejor alternativa.

3.1 Linux containers - LXC/LXD

LXD es una extensión de gestión de contenedores de código abierto para Linux Containers (LXC). LXD mejora las características existentes de LXC y proporciona nuevas características y funciones para crear y administrar contenedores de Linux.

LXD es una interfaz de programación de aplicaciones de transferencia de estado representacional (API REST) que se comunica con LXC a través de la biblioteca liblxc. LXD también proporciona un demonio del sistema que las aplicaciones pueden usar para acceder a LXC y tiene un sistema de distribución de plantillas para permitir una creación y operación de contenedores más rápida.

3.2 Inicio rápido

La forma más fácil de instalar LXD es usando snap:

sudo snap install lxd.

Una vez instalado, agregamos nuestro usuario común al grupo lxd. Esto nos permitirá manejar los contenedores sin sudo.

sudo usermod --append --groups lxd apogon (OJO, apogon es mi usuario)

Cerramos nuestra sesión y volvemos a logearnos a Ubuntu MATE con nuestro usuario.

lxd init, nos permite configurar LXD. Revise las opciones que pide, sin embargo casi a todo hay que dar enter.

Crear un contenedor:

lxc launch ubuntu:18.04 micontenedor

lxc list (lista los contendores)

lxc image list images: | less (listar imágenes de contenedores)

lxc exec micontenedor -- /bin/bash (se ingresa al contenedor y exit para salir del mismo).

3.3 Instalación de perfiles para la red

En general hay un problema con las redes basadas en wifi, los routers dan una ip a una sola mac address, así que los contenedores se crean usualmente con una red propia, bueno esto pasa igualmente en redes cableadas, es decir, el sistema por defecto instala los contenedores en una propia subred o segmento.. Los contenedores tienen su propia red y pueden salir del host, pero no están en la misma red. Hay dos formas básicas para tratar esto: 1. Crear rutas basadas en iptables y 2. Crear interfaces de puente. Los puentes son fáciles de usar si estamos en una red LAN cableada, pero por wifi no es posible usar el DHCP del router o Access Point, ya que este sólo conoce una MAC ADDRESS (la del host) y los contenedores no tienen su propia MAC ADDRESS, a diferencia de una virtualización completa que si emula todo el hardware. Existe la posibilidad de usar ipvlan o routed para asignar una ip de la LAN sin usar el DHCP y mantenernos en el mismo segmento de red.

lxc profile create ipvlan (crea un perfíl de red ipvlan)

lxc profile edit ipvlan (edita el profile ipvlan)

****

config:

user.network-config: |

#cloud-config

version: 2

ethernets:

eth0:

addresses:

- **192.168.68.200**/32

dhcp4: no

dhcp6: no

nameservers:

addresses: [8.8.8.8, 1.1.1.1]

routes:

- to: 0.0.0.0/0

via: 169.254.0.1

on-link: true

description: ipvlan LXD profile

devices:

eth0:

ipv4.address: **192.168.68.200**

nictype: ipvlan

parent: **wlo1**

type: nic

name: ipvlan

used_by:

- /1.0/instances/myipvlan

****

En negrita está lo que debe cambiarse, la ip y el parent. El parent se puede ver con la ip a, desde ahí se puede ver cual es la interfaz de red usada.

lxc profile copy ipvlan ipvlan_200 (hace una copia del profile)

lxc profile edit ipvlan_200 (edita el profile y cambie los valores que se anteriormente se mostraron en negrita)

Cada vez que se crea un nuevo contenedor hay que asignarle un perfil, por eso hay que copiar y cambiar los datos.

lxc launch ubuntu:20.04 myipvlan --profile default --profile ipvlan_200

El comando anterior, crea un contenedor myipvlan con el profile por defecto y adicionalmente con el profile ipvlan_200 el cuál tiene la ip de la red del segmento de red usado en el laboratorio.

lxc list y observe el estado del contenedor. Ingrese al contenedor y haga ping a otro host de la red o a otro contenedor construido con alguno de sus compañeros. Es importante organizar la asignación de ip para no tener conflictos. Los contendores pueden usar el segmento 192.168.68.0 desde la 200 a 250, para efectos de esta clase deben verificar con sus compañeros que no estarán usando la misma IP.

3.4 Creación de un contenedor de Linux con GUI

Inicialmente se deben crear los perfíles:

1. **OPCIONAL:** Perfíl ipvlan de red. Este perfíl ya saben crearlo, sin embargo hay una modificación cuando hay una distro basada en debian u otras distro's, debian y ubuntu son similares, pero hay pequeñas diferencias. Crear un ipvlan_debian y copiar lo proporcionado en el classroom. Como sólo crearemos un sólo debian linux, no hace falta hacer copias del perfíl de ipvlan_debian.

```
config:

user.network-config: |

#cloud-config

version: 2

ethernets:


eth0:

dhcp4: false

dhcp6: false

user.user-data: |

#cloud-config

bootcmd:

- echo 'nameserver 8.8.8.8' > /etc/resolv.conf

description: ipvlan profile for Debian container images

devices:
```

eth0:

ipv4.address: 192.168.68.201

name: eth0

nictype: ipvlan

parent: wlo1

type: nic
name: ipvlan_debian

2. El otro perfíl es para poder ejecutar aplicaciones con una salida GUI al host. Cree un perfil gui, editelo y copie el contenido de lxdguiprofile.txt a ese perfíl.

config:

environment.DISPLAY: :0

raw.idmap: both 1000 1000

user.user-data: |

#cloud-config

runcmd:

- 'sed -i "s/; enable-shm = yes/enable-shm = no/g" /etc/pulse/client.conf'

- 'echo export PULSE_SERVER=unix:/tmp/.pulse-native | tee --append /home/ubuntu/.profile'

packages:

- x11-apps

- mesa-utils

- pulseaudio

description: GUI LXD profile

devices:

PASocket:

path: /tmp/.pulse-native

source: /run/user/1000/pulse/native

type: disk

X0:

path: /tmp/.X11-unix/X0

source: /tmp/.X11-unix/X0

type: disk

mygpu:

type: gpu

name: gui
used_by:

Una vez creados los perfiles se puede crear la máquina virtual:

lxc launch –profile default –profile gui ubuntu:22.04

Verifique que el contenedor tenga red, no se puede hacer en muchos casos ping desde el host al contenedor (es una característica de ipvlan, sin embargo no lo he probado con un dongle usb). Una forma es que desde el otro contenedor creado inicialmente (ubuntu) hagan ping a la dirección del contenedor de linux-gui. El contenedor de linux-gui es muy básico y no tiene nada instalado, por eso tampoco podrá ejecutar un ping desde ese contenedor.

3.5 Preparación de linux-gui

Cómo mencioné anteriormente, el contendor linux-gui es muy básico y no tiene nada instalado, así que lo primero es verificar que tiene red y que el nameserver está asignado correctamente, en su defecto pueden revisar en:

vi /etc/ resolv.conf que este nameserver 8.8.8.8 para poder resolver los nombres, ese es el DNS de google (supongo que ya lo sabían). Salir del contenedor, y volver a reiniciarlo (esto en caso de que no tuviesen red).

lxc restart linux-gui

Ingresar nuevamente al contendor y ejecutar:

apt update

apt install iputils-ping (Para tener el ping)

apt install -y ubuntu-desktop-xfce

adduser usuario (crea un usuario, por favor cambie usuario por un nombre de usuario)

usermod -aG sudo usuario (se asignan privilegios de sudo al usuario usuario) sed

-i '1 i\TERM=xterm-256color' /home/usuario/.bashrc

echo "export DISPLAY=:0" >> /home/usuario/.bashrc

sh -c "echo 'Set disable_coredump false' > /etc/sudo.conf"

salga del contenedor con exit y desde el host ejecute:

lxc exec linux-gui -- sudo -u usuario xfce4-panel

Observe el resultado …. Otras opciones:

Start: lxc start linux-gui

Stop: lxc stop linux-gui

Remove: lxc destroy linux-gui

OPTIONAL: Para probar la exportación del display, un camino más rápido es simplemente instalar las librerías por ejemplo de firefox y no todo el entorno.

lxc profile create gui

cat gui.profile | lxc profile edit gui

lxc launch ubuntu:20.04 --profile default --profile gui micontenedor lxc

exec micontenedor -- sudo --user ubuntu --login

Una vez dentro del contenedor:

sudo apt-get install -y firefox

firefox

Observe el resultado.

3.6 Posibles problemas a resolver

Algunos linux funcionan basados en debian y cada distro tiene sus diferencias, después de revisar la asignación de red, es posible que tengamos que hacer los siguientes pasos adicionales una vez dentro del contenedor y por ejemplo con un usuario (ejemplos realizados con kali: linux)

sudo ifconfig eth0 192.168.68.211 netmask 255.255.255.0

sudo route add default gw 192.168.68.1 eth0

Aparte, verificar que en /etc/resolv.conf se encuentre nameserver 8.8.8.8, sino asignarlo.

3.7 Creación de un contenedor LXC con LXD para servicios web en puerto predeterminado

Ya hemos creado y ejecutado contenedores, así que ya se encuentran familiarizados con su manejo. Las instrucciones siguientes funcionan bien, sin embargo ya no las voy a detallar, pues ya deberían entender su uso y configuración de lo que se quiere realizar.

lxc launch ubuntu:20.04 test1 -c boot.autostart=true

En el contendor

lxc exec test1 bash

apt update && apt upgrade -y

apt install apache2

exit

En el host. Creamos un perfil para el puerto proxy 8080 en el host LXD hasta el puerto 80.

lxc profile create proxy-8080

lxc profile device add proxy-8080 hostport8080 proxy connect="tcp:127.0.0.1:80" listen="tcp:0.0.0.0:8080"

lxc profile show proxy-8080

Agregamos el nuevo perfil a nuestro contenedor test1:

lxc profile add test1 proxy-8080

lxc config show test1 -e

Luego puede ingresar a un navegador web en

http://LXD-Host-address:8080 y accederá a Apache en el contenedor test1.

Para eliminar el perfil de proxy del contenedor y eliminar el perfil. lxc

profile remove test1 proxy-8080

lxc profile delete proxy-8080

3.8 Instalar LXDWARE en LXD

Cómo ya está instalado lxd en cada host, los siguientes comandos permiten instalar LXDWARE.

lxc launch images:ubuntu/22.04 lxd-dashboard

lxc exec lxd-dashboard /bin/bash

En el contenedor

apt update && apt install wget nginx php-fpm php-curl sqlite3 php-sqlite3 -y

wget https://github.com/lxdware/lxd-dashboard/archive/v3.4.0.tar.gz tar -xzf

v3.4.0.tar.gz

cp -a lxd-dashboard-3.4.0/default /etc/nginx/sites-available/
cp -a lxd-dashboard-3.4.0/lxd-dashboard /var/www/html/

nano /etc/nginx/sites-enabled/default

apt-get install nano

nano /etc/nginx/sites-enabled/default

En ese archivo debe quedar así:

#fastcgi_pass unix:/run/php/php7.4-fpm.sock;

fastcgi_pass unix:/run/php/php8.1-fpm.sock;

sigue en el contenedor:

mkdir -p /var/lxdware/data/sqlite

mkdir -p /var/lxdware/data/lxd

mkdir -p /var/lxdware/backups

chown -R www-data:www-data /var/lxdware/

chown -R www-data:www-data /var/www/html

systemctl restart nginx

###

En el host revisar la ip del contendor y entrar a un navegado directamente a esa ip

OJO: dar un usuario y un password SIN REPETIR NADA, UNA SOLA VEZ reactivar la página (refrescar)

entrar al LXDWARE con ese usuario y password.

Ver el certificado y copiarlo TODO COMPLETO

ir al host y abrir una carpeta y escribir el certificado.

lxdware.crt llamar así el archivo

Donde esta el archivo y en una terminal ejecutar:
lxc config trust add lxdware.crt

lxc config set core.https_address [::]

Ahora volver a la web de LXDWARE y añadir un host (add host)

poner un nombre y sobretodo la ip del host validar

3.9 Acceso a los contenedores desde el exterior

Hacer una investigación de qué es y cómo funciona IPTABLES ??

Desde el host: Si queremos vincular un contenedor y un servicio en un puerto determinado, la siguiente instrucción da ese acceso

apogon@server:~$ `sudo iptables -t nat -I PREROUTING -p tcp --dport 6379 -j DNAT --to-destination 192.168.1.86:6379`

La IP es la del contenedor y el puerto de algún servicio, por ejemplo 192.168.1.86:6379 es el servicio del servidor de redis que están usando en la materia de análisis y visualización de datos.

Otra posibilidad es dar acceso a toda la interfaz del contenedor … con ip ver cual es la interfaz de los contendores (puede ser un lxdbr0 o algo así en lxd)

apogon@server:~$ sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE

apogon@server:~$apt-get install iptables-persistent

apogon@server:~$sudo netfilter-persistent save

4. Técnicas criptográficas y conexiones remotas

Algoritmos

• Cifrado simétrico.

• Cifrado asimétrico.
• Funciones de hash

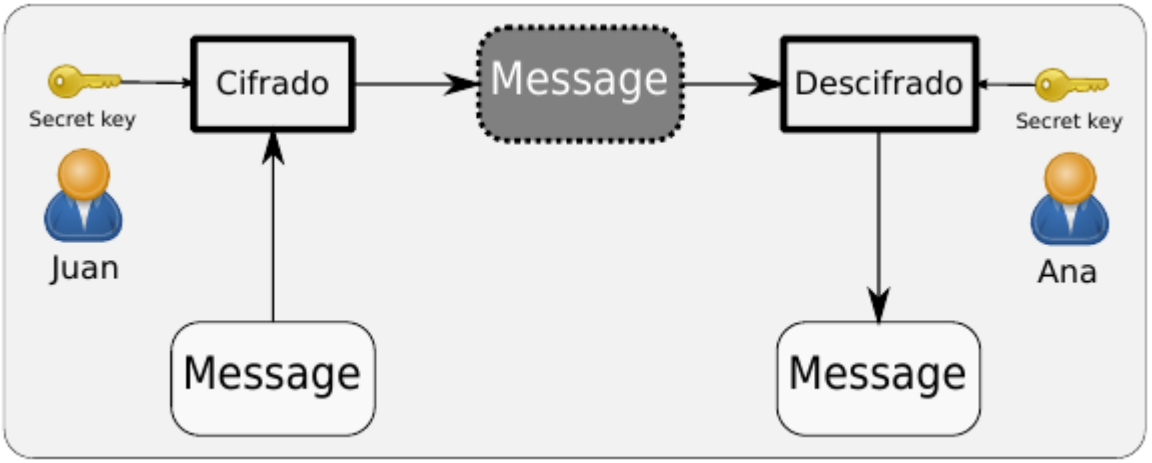


Figura 1.1: La misma clave se utiliza para cifrar y descifrar.
• Generación de números aleatorios.

Servicios
Autenticidad: Poder demostrar la autoría de un mensaje.
Confidencialidad: Sólo los autorizados pueden leer el mensaje.
Integridad: Asegurar la no modificación del mensaje.
No repudio: No poder negar la autoría de un mensaje.

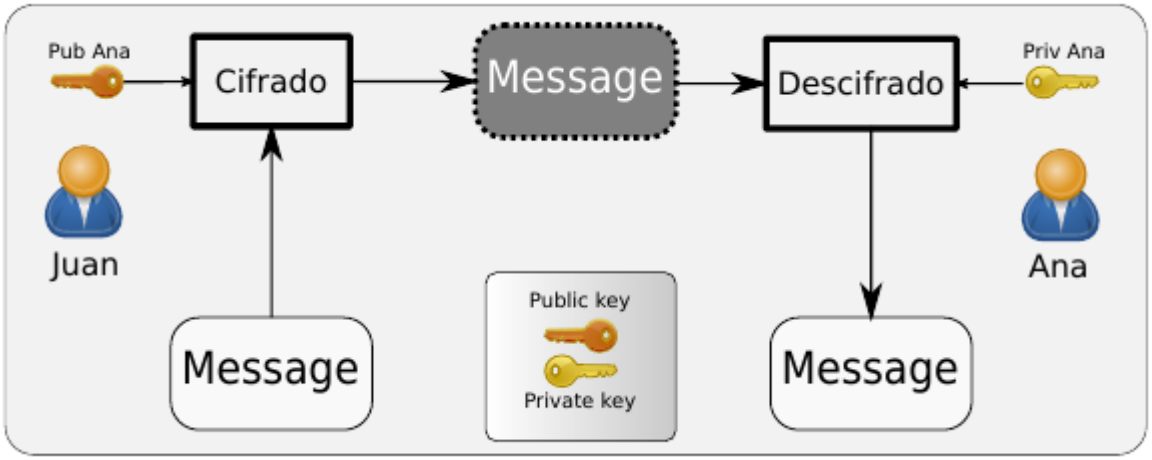
4.1 Criptografía de clave secreta

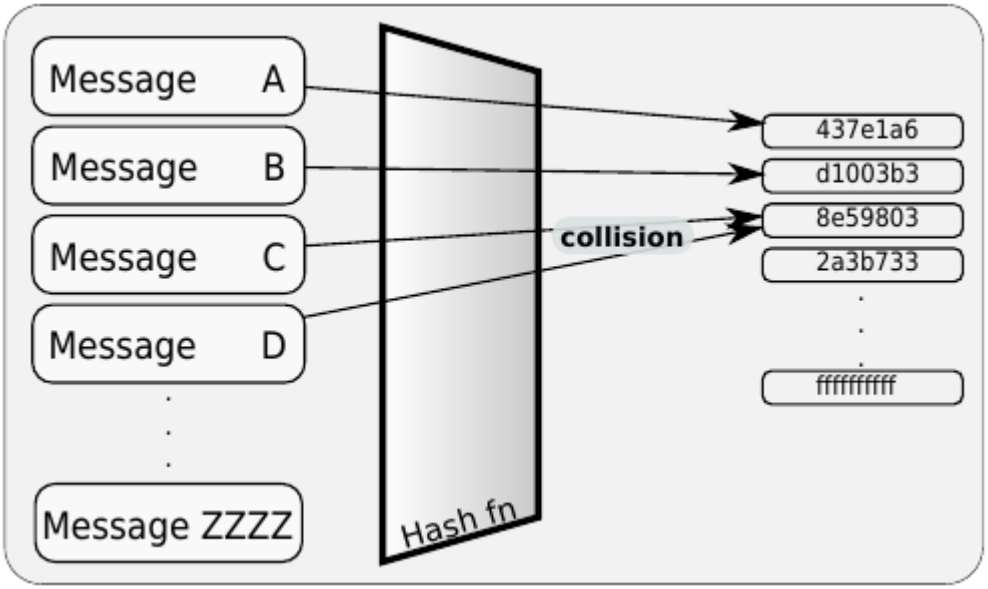
• Conocida como criptografía de clave secreta.
• Por regla general los algoritmos de cifrado y descifrado son públicos. La seguridad reside únicamente en el secreto de las claves.
• Los datos (plaintext) se cifran con una clave que ha de ser conocida sólo por el emisor y el/los receptores para garantizar la seguridad del sistema.
• El emisor y el receptor han de conocer la clave de cifrado.
• Es necesario que antes de enviar un mensaje ambos (emisor y receptor) conozcan la clave utilizando para ello un medio de comunicación seguro.
• Por regla general suelen ser más rápidos que los algoritmos de clave pública.
• Algoritmos de clave secreta: IDEA, ARCFOUR, BLOWFISH, AES

4.2 Criptografía de clave pública

• Conocida también como criptografía asimétrica. Inventada en 1975.
• Cada usuario tiene dos claves. Una de ellas es la clave privada que sólo ha de conocer el propietario de la clave, y la otra la clave pública que ha de ser conocida por el mayor número de personas posible.
• A diferencia de la clave secreta, la clave privada sólo tiene que ser conocida por una sola persona y no se tiene que compartir con nadie.

• Los datos cifrados con una clave sólo se pueden descifrar con la otra.



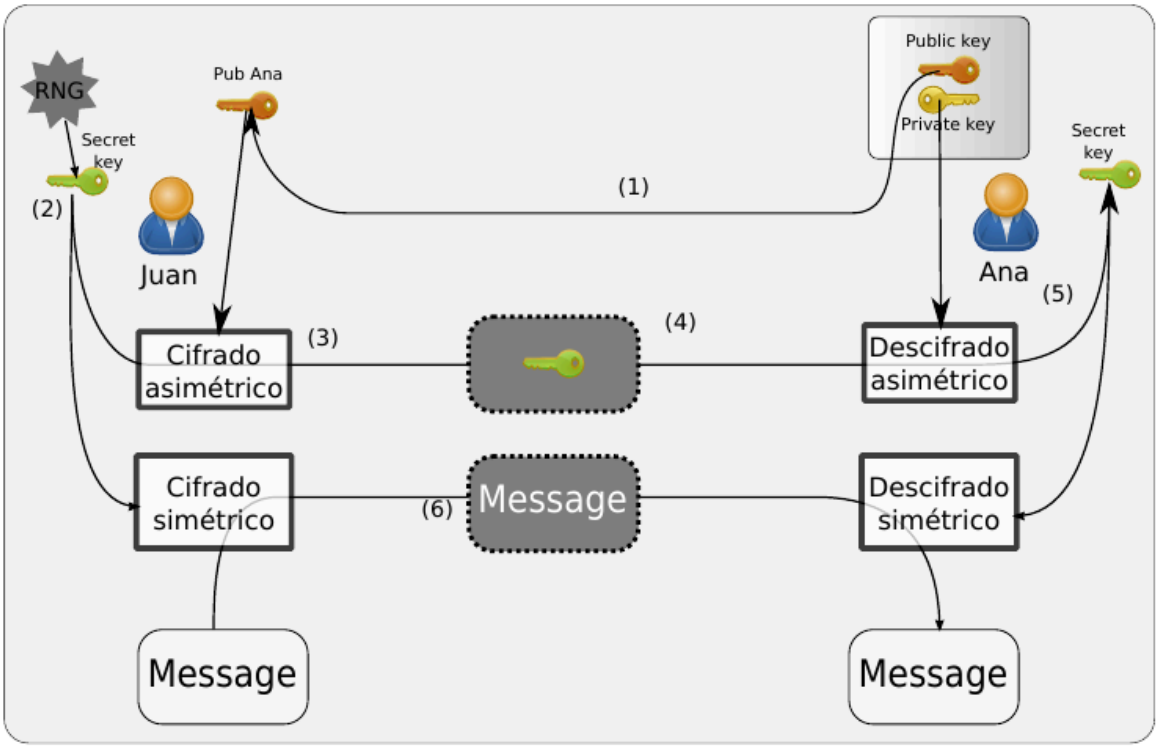


• Son algoritmos sencillos pero de alto coste computacional.
• Los principales algoritmos existentes son: DSA, RSA, ECC.
Cuantas más personas conozcan la clave pública, más seguro será el sistema

4.3 Funciones de hash

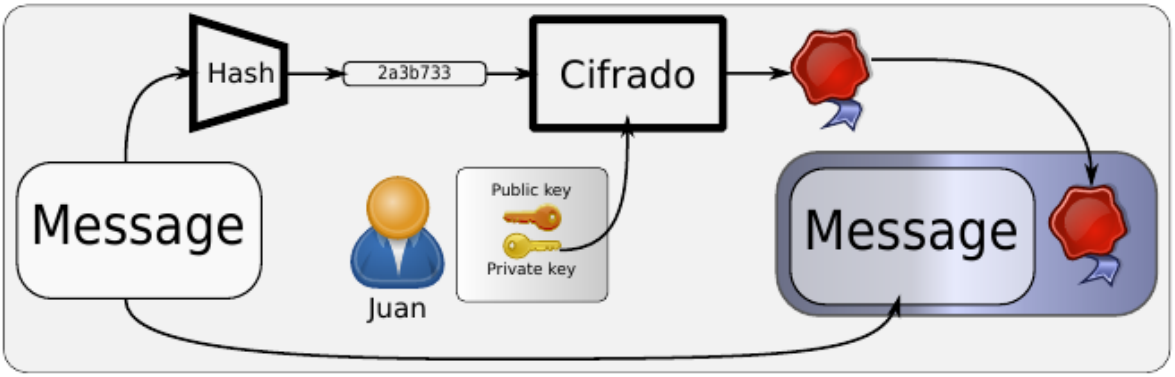
• En inglés se conocen como message digest (MD), o como message hash algorithm (SHA).
• Los algoritmos de hash, producen una salida que es mucho más reducida que el mensaje original y no se puede reconstruir el mensaje original a partir del resumen. Entre 128 y 512 bytes.
• Por contra, los algoritmos de cifrado toman como entrada una serie de datos y producen como salida otros datos a partir de los cuales se puede reconstruir el original. El tamaño del mensaje cifrado similar al original.
• Cada mensaje debería de producir un resumen distinto.
• Funciones de hash: SHA-224 SHA-512, MD5, ...
• Matemáticamente es imposible crear funciones de hash perfectas!

• Hay mayor número de mensajes que de resúmenes.



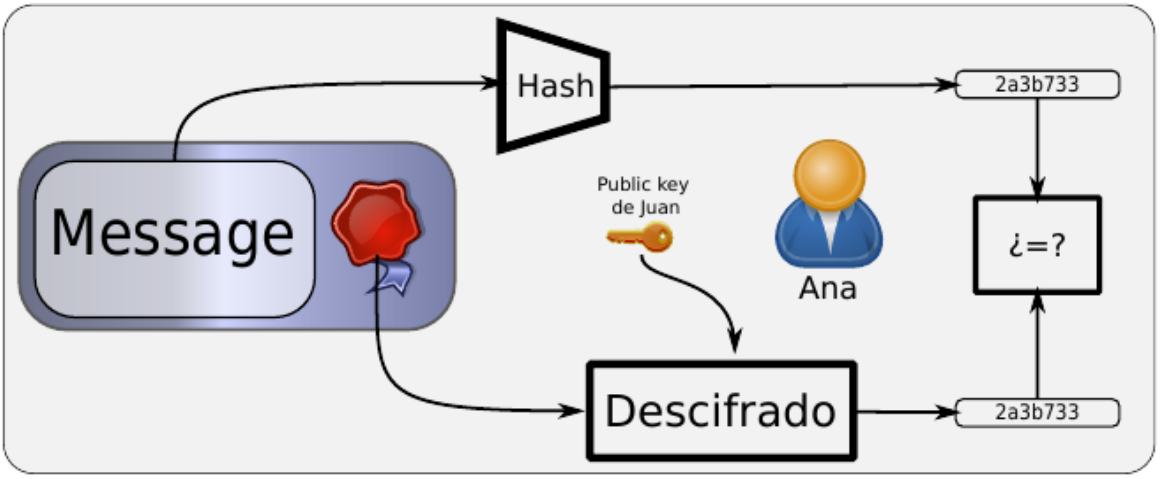
4.4 Firma digital

• Se dice firma digital a un esquema matemático que sirve para demostrar la autenticidad de un mensaje digital [WP].
• Una firma digital garantiza al destinatario que el mensaje fue creado por el remitente.
• Pasos para firmar un mensaje:
1. Calcular un resumen (MD5, SHA, etc.) del mensaje.
2. Cifrar el resumen resultante con la clave privada del remitente.
3. Adjuntar el resultado al mensaje original.



• Pasos para verificar una firma digital:
1. Extraer mensaje original.
2. Calcular un resumen (MD5, SHA, etc.) del mensaje original.
3. Extraer la firma digital.
4. Descifrar la firma digital con la clave pública de Juan.

5. Si coinciden los resultados entonces es correcta la firma.



4.5 La suite SSH (secure SHell)

4.5.1 Introducción

• ssh es un programa para realizar conexiones seguras entre máquinas conectadas mediante una red de comunicaciones insegura.
• Implementa varios algoritmos de encriptación y autentificación. Para el establecimiento de la conexión utiliza encriptación asimétrica. Para la transferencia de datos utiliza encriptación simétrica (más rápida).
• Diseñado para reemplazar a los antiguos programas de comunicaciones como: telnet, rlogin, rsh, rcp, rdist, etc., conocidas como “los programas que empiezan por R”, que tienen graves problemas de seguridad.
• Puede encriptar toda la información que el protocolo X0 transmite entre máquinas; permite realizar transferencias de archivos; para instalarlo no es necesario modificar ningún archivo del sistema.
• Algunos gobiernos creen que es demasiado bueno: Francia, Rusia, Iran, etc. Por lo que está prohibido su uso en estos países. Estados Unidos prohíbe exportar. Por tanto, no ningún ciudadano de USA puede trabajar en el código del ssh: Dug Song

4.5.2 Utilización básica de ssh

Suponiendo que todo el paquete de programas ssh está instalado tanto en nuestra propia máquina, como en la máquina remota. La forma de utilizar ssh para conectarse a una máquina remota es:

$ ssh máquina remota

• La primera vez se creará en la máquina local el directorio $HOME/.ssh y dentro de este directorio el archivo known hosts. Éste archivo contendrá la clave pública de la máquina remota a la que nos acabamos de conectar. Cada máquina que ofrezca el servicio ssh tendrá una clave pública y otra privada.

• Ssh nos pedirá el password y si lo conocemos entonces se ejecutará el shell de conexión normal, pero donde todo lo que tecleemos y lo que nos devuelva por pantalla se enviará encriptado.
• Si la conexión la realizamos desde un entorno gráfico en la máquina local (esto es, si la variable de entorno DISPLAY está definida) entonces ssh redireccionará automáticamente todas las aplicaciones gráficas que lancemos en la máquina remota para que se visualicen en la máquina que estamos conectados (local).
• Tanto el establecimiento de la conexión como todos los datos que se transmitan se hacen por canal seguro.

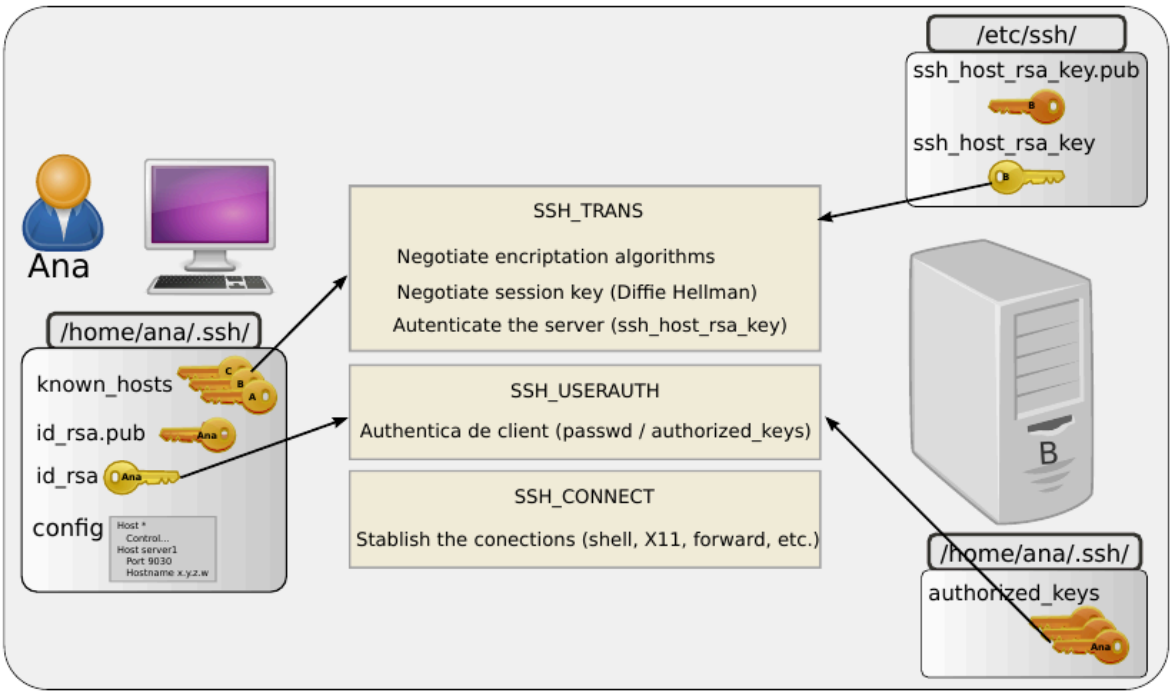
4.5.3 Claves de usuario

• Con el esquema básico de autenticación, la identidad de un usuario se decide utilizando el sistema de seguridad UNIX de la máquina remota, esto es, en función del archivo /etc/passwd, /etc/shadow. Nuestro password viaja hasta el servidor donde el propio servidor lo valida.
• Podemos utilizar el sistema de clave pública de ssh para crear una pareja de claves (una privada y otra pública) que sustituyan a la clave UNIX. Esto sólo se ha de hacer una vez. Es algo así como poner el password a la propia cuenta.
• Pasos a seguir para que el usuario “xxxxxx” de la máquina “local” pueda entrar en la cuenta del usuario “yyyyyyy” de la máquina “remota”.

1. Con la orden ssh-keygen generamos las dos claves (pública y privada). Cada una de ellas se guarda en un archivo distinto. Ejemplo de creación de claves del usuario kali:
$ ssh-keygen -t dsa
Es conveniente introducir una passphrase para proteger el archivo de clave privada. De esta forma, aunque un atacante obtenga nuestro archivo de clave privada no podrá utilizarlo.
¿Passphrase?
La clave ssh se llama passphrase en lugar de password para diferenciarla de la que aparece en el archivo /etc/shadow, y para “animarnos” a introducir una cadena larga.

2. Luego tenemos que copiar, sin preocuparnos que la capturen (aunque sí que la modifiquen!), nuestra clave pública al archivo $HOME/.ssh/authorized keys de la máquina remota:

$ rcp id_dsa.pub remota:.ssh/authorized_keys
claves públicas, todas ellas concatenadas.
3. Asegurarnos que los permisos del archivo authorized keys son correctos:
$ ssh remota
$ chmod 600 .ssh/authorized_keys
$ chmod 700 .ssh
• Cuando se utiliza este esquema de autenticación: La passphrase no circula por la red.



4.5.4 Copia segura con scp

• Al igual que rcp permite realizar copias de archivos entre distintas máquinas, scp hace lo mismo pero de forma segura.
• scp acepta los mismo parámetros que rcp, pero utiliza un canal seguro creado por ssh para realizar la transferencia de datos.
• scp sigue los mismos esquemas de comprobación que ssh. Por tanto la forma de funcionar dependerá de cómo cada usuario tenga configurado ssh (tenga clave propia o no y si la clave tiene passphrase o no).

4.5.5 Archivos de Configuración

• /etc/ssh/sshd config: Configuración del demonio sshd. Se puede establecer varios parámetros de seguridad (Ver sshd(8)).
/etc/ssh/ssh config $HOME/.ssh/configy : Configuración del cliente.
• /etc/ssh/ssh host key : Contiene la clave privada de la máquina. Sólo root ha de poder leerla.
• /etc/ssh/ssh host key.pub: Parte pública de la clave. Permiso de lectura universal.
• /etc/ssh/ssh known hosts y $HOME/.ssh/known hosts: Contienen la clave pública de máquinas remotas y se utiliza para comprobar que la máquina remota es realmente quien dice ser.

Tarea y reporte: Generar un reporte del uso de ssh cómo sistema de conexión segura entre máquinas:

Secure Shell es una aplicación que mediante el protocolo SSH en el puerto 22 realiza la conexión entre un equipo servidor y sus clientes para proporcionar el servicio de terminal segura, con el cual faculta al cliente de realizar tareas administrativas mediante un usuario previamente definido en nuestro servidor, al ser un servicio remoto estamos facultando el

control de acceso por tal motivo es importante definir el nivel de usuario deseado para cada nivel de acc. SSH es uno de los protocolos de comunicación más populares de Internet, ya que nos proporciona la capacidad de acceder de forma segura a sistemas locales y remotos utilizando un canal de comunicación encriptado. Además de ser Open-Source, es ampliamente usado por personal de TI (por ejemplo, Desarrolladores, Webmasters, Administradores de Sistemas, Pentesters, etc). Pasos para habilitar el soporte ssh en cliente y servidor:

Crear uno o varios usuarios para acceder a terminal en el SO, pueden utilizarse usuarios ya existentes, Abrir terminal y ejecutar el comando de instalación en servidor, como superusuario:

#apt-get install openssh-server openssh-client

#service ssh start
Una vez instalado, abrir el archivo del manual de open ssh:

 # man sshd_config
Con nmap revise los puertos abiertos e identifique el correspondiente al ssh ?
El archivo de configuración de ssh se encuentra en:
/etc/ssh/sshd_config
IMPORTANTE: Antes de cambiar el archivo de configuración, hacer una copia del archivo original y protegerlo contra escritura; para en caso de algún suceso, poder recuperarlo y restablecerlo.
Copiar el archivo /etc/ssh/sshd_config y protegerlo:

#sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.original

#sudo chmod a-w /etc/ssh/sshd_config.original

Una vez respaldado y protegido el archivo podremos editar el archivo inicial con nano en el entendido de que si se daña o desconfigura podremos reemplazarlo por el que acabamos de crear, una vez abierto vamos a analizar línea por línea e identificar la que contiene:

#what portss, IP's and protocols we listen for
port 22
Aquí ubicamos entonces la lista de direcciones IP, puertos y protocolos que serán escuchados (Listen) , asi como los archivos Key que contienen la identificación electrónica de los equipos que tienen acceso esto para en un futuro poder implementar VPN's

1. Cambiar el puerto de escucha en 2222, reiniciar y verificar luego con nmap ?
2. Configura el servidor SSH de forma adecuada para que acepte la redirección X11, de tal forma que se puedan ejecutar aplicaciones gráficas de forma remota. Haz pruebas y comprueba su funcionamiento. Mediante ssh existe la posibilidad de ejecutar aplicaciones gráficas en el equipo remoto (servidor) y manejarlas y visualizarlas en el local (cliente). El servidor ssh deberá tener activada la redirección del protocolo X, es decir, deberá tener el siguiente parámetro en el archivo de configuración /etc/ssh/sshd_config:

// Habilitar la redirección X en /etc/ssh/sshd_config
X11Forwarding yes
// Ejecutar aplicaciones gráficas

Vamos a simular que nuestra máquina no tiene acceso a internet, pero otra en el laboratorio si lo tiene y queremos navegar con firefox:

$ ssh -X usuarioPCremoto@IP_del_equiporemoto
Nota: -X para redirigir Xwindows.
$ firefox // Ejecutamos el Firefox

3. **Realizar una copia remota de archivos entre las dos máquinas (usando *scp*). Hacer capturas de pantalla**

REFERENCIAS BIBLIOGRÁFICAS

Libros y Documentación Técnica

M. Portnoy, *Virtualization Essentials*, 2nd ed. Indianapolis, IN: Sybex, 2016.

J. Turnbull, *The Docker Book: Containerization is the New Virtualization*. USA: James Turnbull, 2014.

K. Hess and A. Newman, *Practical Virtualization Solutions: Virtualization from the Trenches*. Upper Saddle River, NJ: Prentice Hall, 2010.

D. Merkel, "Docker: Lightweight Linux Containers for Consistent Development and Deployment," *Linux Journal*, vol. 2014, no. 239, 2014.

S. Soltesz et al., "Container-based Operating System Virtualization: A Scalable, High-performance Alternative to Hypervisors," in *Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems*, 2007, pp. 275–287.

Cloud Computing y Servicios

P. Mell and T. Grance, "The NIST Definition of Cloud Computing," National Institute of Standards and Technology, Special Publication 800-145, 2011. [Online]. Available:

https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, Jun. 2009.

L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A Break in the

Clouds: Towards a Cloud Definition," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 50–55, Jan. 2009.

Virtualización: Hypervisores y Tecnologías

P. Barham et al., "Xen and the Art of Virtualization," in *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, Bolton Landing, NY, 2003, pp. 164–177.

A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori, "KVM: the Linux Virtual Machine Monitor," in *Proceedings of the Linux Symposium*, vol. 1, Ottawa, Canada, 2007, pp. 225–230.

J. Sugerman, G. Venkitachalam, and B.-H. Lim, "Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor," in *Proceedings of the 2001 USENIX Annual Technical Conference*, Boston, MA, 2001, pp. 1–14.

G. J. Popek and R. P. Goldberg, "Formal Requirements for Virtualizable Third Generation Architectures," *Communications of the ACM*, vol. 17, no. 7, pp. 412–421, Jul. 1974.

Contenedores: LXC, LXD, Docker

"LXC - Linux Containers," Canonical Ltd. [Online]. Available:

https://linuxcontainers.org/lxc/introduction/

"LXD - System Container Manager," Canonical Ltd. [Online]. Available:

https://linuxcontainers.org/lxd/introduction/

Docker Inc., "Docker Documentation," 2024. [Online]. Available:

https://docs.docker.com/

W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An Updated Performance Comparison of Virtual Machines and Linux Containers," in *2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Philadelphia, PA, 2015, pp. 171–172.

R. Morabito, J. Kjällman, and M. Komu, "Hypervisors vs. Lightweight Virtualization: A Performance Comparison," in *2015 IEEE International Conference on Cloud Engineering*, Tempe, AZ, 2015, pp. 386–393.

Seguridad y SSH

T. Ylonen and C. Lonvick, "The Secure Shell (SSH) Protocol Architecture," RFC 4251, Internet Engineering Task Force (IETF), Jan. 2006. [Online]. Available:

https://www.rfc-editor.org/rfc/rfc4251

T. Ylonen and C. Lonvick, "The Secure Shell (SSH) Authentication Protocol," RFC 4252, IETF, Jan. 2006. [Online]. Available:

https://www.rfc-editor.org/rfc/rfc4252

M. Friedl, N. Provos, and W. Simpson, "Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol," RFC 4419, IETF, Mar. 2006.

D. J. Barrett, R. E. Silverman, and R. G. Byrnes, *SSH, The Secure Shell: The Definitive Guide*, 2nd ed. Sebastopol, CA: O'Reilly Media, 2005.

Redes y BGP

Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271, IETF, Jan. 2006. [Online]. Available:

https://www.rfc-editor.org/rfc/rfc4271

J. Stewart, *BGP4: Inter-Domain Routing in the Internet*. Reading, MA: Addison-Wesley, 1999.

G. Huston, "BGP Routing Table Analysis Reports," CIDR Report. [Online]. Available:

https://www.cidr-report.org/

Criptografía

W. Stallings, *Cryptography and Network Security: Principles and Practice*, 8th ed. Harlow, UK: Pearson Education, 2020.

B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed. New York: John Wiley & Sons, 1996.

National Institute of Standards and Technology (NIST), "Advanced Encryption Standard (AES)," FIPS PUB 197, Nov. 2001.

Recursos en Línea

"KVM - Kernel Virtual Machine," Red Hat Inc. [Online]. Available:

https://www.linux-kvm.org/

"Understanding Linux Containers," Red Hat Developer. [Online]. Available:

https://developers.redhat.com/blog/2018/02/22/container-terminology-practical-introduction

"LXDWARE - Web-based GUI for LXD," GitHub. [Online]. Available:

https://github.com/lxdware/lxd-dashboard