PKI

https://linuxize.com/post/how-to-set-up-an-openvpn-server-on-centos-7/ Public Key Infraestrucure

previa

orientado a la instalación de un **openvpn**

creación de los siguientes elementos:

- CA[1)]
- SERVER certificate + key
- CLIENT certificate + key

usando la herramienta **easy-rsa**

CA

se recomienda crear todos estos ficheros en un servidor standalone / offline de la insfraestructura que se monte

1. editar fichero **vars** (vars.example) para adecuar los valores (y hacer `source vars`)
2. `./easyrsa init-pki`
3. `./easyrsa build-ca` (opcionalmente, **nopass**)
    - genera los ficheros **ca.crt** y **ca.key**
    - con estos ficheros podremos firmar los *requests* de certificados para servidor y cliente

Deffie-Hellman

usado para el intercambio de llaves y firma HMAC, añadiendo una capara se seguridad adicional

1. `./easyrsa gen-dh`
    - el archivo **dh.pem** generado se copia en **/etc/openvpn/**
2. generar HMAC: `openvpn –genkey –secret ta.key`
    - el archivo **ta.key** generado se copia en **/etc/openvpn/**

servidor

1. `./easyrsa gen-req server1 nopass`
    - **server1** es el nombre que le asignamos
    - **nopass** para que no nos solicite la contraseña al arrancar (openvpn server)
    - genera **server1.key** (copiar en **/etc/openvpn/**) y **server1.req**
2. importar (en caso que la PKI la tengamos en otra máquina) la *requests* del servidor: `./easyrsa import-req server1.req server1`
    - **server1.req** es el fichero creado
    - **server1** nombre del servidor
    - este comando copia el fichero **server1.req** en **pki/reqs**
3. firmar la *request*: `./easyrsa sign-req server server1`
    - el primer argumento puede ser **server** o **client**

- el segundo argumento es el *short name* de la entidad servidor
- genera un archivo **server1.crt**

configuración servidor

1. copiar el fichero de configuración de ejemplo desde **/usr/share/doc/openvpn-*/sample/sample-config-files/server.conf**
2. ajustar los valores de las llaves:
    1. ca
    2. cert
    3. key
    4. dh
    5. tls-auth
3. ajustar valores de redirección, si procede
    1. push «redirect-gateway»
    2. push «dhcp-option»
4. ajustar usuario y grupo (por nobody:nogroup o uno a medida)
    1. user nobody
    2. user nogroup
5. ajustar seguridad:
    1. auth SHA256
6. algo parecido a esto:

```
port 1194
proto udp
dev tun
ca ca.crt
cert server1.crt
key server1.key  # This file should be kept secret
dh dh.pem
server 10.8.0.0 255.255.255.0
ifconfig-pool-persist ipp.txt
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 208.67.222.222"
push "dhcp-option DNS 208.67.220.220"
keepalive 10 120
tls-auth ta.key 0 # This file is secret
cipher AES-256-CBC
user nobody
group nobody
persist-key
persist-tun
status openvpn-status.log
verb 3
explicit-exit-notify 1
auth SHA256
```

cliente

1. `./easyrsa gen-req client1 nopass`
    - **client1** es el nombre del fichero que generamos y el Common Name que nos sugiere
    - **nopass** para que no nos solicite la contraseña al accedr
    - genera **./pki/private/client1.key** y **./pki/reqs/server1.req**

2. importar (en caso que la PKI la tengamos en otra máquina) la *requests* del cliente: `./easyrsa import-req client1.req client1`
    - **client1.req** es el fichero creado
    - **client1** nombre del servidor
    - este comando copia el fichero **client1.req** en **pki/reqs**
3. firmar la *request*: `./easyrsa sign-req client client1`
    - el primer argumento puede ser **server** o **client**
    - el segundo argumento es el *short name* de la entidad cliente
    - genera un archivo **client1.crt**

AWS ACM

Para poder importar esta PKI y usarla en AWS (para crear un VPNEndPoint, por ejemplo), es necesario importar los siguientes archivos:

1. **./pki/issued/server1.crt** en cuerpo del certificado
2. **./pki/private/server1.key** en la clave privada del certificado
3. **./pki/ca.crt** en la cadena de certificados

[1)]
Certificate Authority