Quaderno di progetto

Installazione e configurazione della VPN di Istituto

Autori: C. Gaibisso, B. Martino
Ultima revisione: 04 marzo '19
Versione: 1.1

Sommario

1 Contenuto

Questo documento illustra nel dettaglio le modalità di istallazione e di configurazione dell'infrastruttura software che garantisce un accesso sicuro, anche da reti diverse dalle 150.146.5.x (reti interne all'edificio di Via dei Taurini), ai servizi telematici di Istituto ritenuti critici per la sicurezza dei dati gestiti e la tutela della privacy (richieste di ferie, di rimborsi, di acquisti, gestione delle presenze, accesso alle risorse di calcolo, ecc....),.

2 Caratteristiche della soluzione software

L'infrastruttura implementata ricorre alle funzionalità offerte da:

- *OpenVPN (vers. 2.4.4):*

  per la realizzazione di una *VPN* (*Virtual Private Network*) basata su tecnologia *SSL* (*Secure Socket Layer*). In quanto segue, le componenti client e server di *OpenVPN*, saranno riferite come *Server VPN* e *Client VPN*, rispettivamente.

- *Easy-RSA (vers. 3.0.6):*

  per l'implementazione di una soluzione di *PKI* (*Public Key Infrastructure*), anch'essa open-source, in grado di:
  - dotare i server che erogano i servizi, e gli utenti che li utilizzano, delle credenziali digitali (chiave privata e certificato digitale) necessarie al reciproco riconoscimento e a criptare le comunicazioni;
  - svolgere le funzionalità tipiche di una *CA (Certification Authorithy)*. Tale soluzione è riferita come *Server CA*, in quanto segue.

Per motivi di sicurezza, le componenti *Server VPN* e *Server CA* sono ospitate su differenti *host* in ambiente *Ubuntu Server (vers. 18.04.1)*: il *Server VPN* da un *container Proxmox (vers. 5.2.1.)*, raggiunginile all'indirizzo 150.146.5.253; il *Server CA*, da una VM *Oracle VM Virtual Box (vers. 6.0)*, raggiungibile all'indirizzo è 150.146.5.224.

3 Operazioni preliminari

Sul *Server VPN (root)*

a) Aggiorna i pacchetti installati

```
$ apt update
$ apt upgrade
$apt dist-upgrade
```

b) Verifica l'esistenza del device */dev/net/tun*

```
$ ls -l /dev/net/tun
```

c) Se il device non esiste, inserisci nel file */etc/rc.local* le seguenti linee di comando

```
#!/bin/bash
cd /dev
mkdir net
mknod net/tun c 10 200
chmod 0666 net/tun
```

d) Rendi il file */etc/rc.local* eseguibile

```
$ chmod ugo+x /etc/rc.local
```

e) Esegui il *reboot* del container

```
$ reboot
```

f) Verifica nuovamente l'esistenza del device */dev/net/tun*

```
$ ls -l /dev/net/tun
```

Sul *Server CA (system)*

a) Aggiorna i pacchetti installati

```
$ sudo apt update
$ sudo apt upgrade
$ sudo apt dist-upgrade
```

4 Installazione *OpenVPN* e *EasyRSA*

Sul *Server VPN (root)*

a) Installa *OpenVPN*

```
$ apt update
$ apt install openvpn
```

b) Esegui

```
$ wget -P ~/ <downloadLink>
```

dove *<downloadLink>* è il link all'ultima versione di *EasyRSA,* così come riportato dalla pagina *Release* del sito ufficiale del progetto *EasyRSA GitHub*.

All'atto del rilascio di questo documento, tale link è

https://github.com/OpenVPN/easy-rsa/releases/download/v3.0.6/EasyRSA-unix-v3.0.6.tgz

c) Estrai il tarball scaricato

```
$ cd ~
$ tar xvf EasyRSA-unix-v3.0.6.tgz
```

Sul *Server CA (system)*

a) Esegui

```
$ wget -P ~/ <downloadLink>
```

dove *<downloadLink>* è il link all'ultima versione di *EasyRSA,* così come riportato dalla pagina *Release* del sito ufficiale del progetto *EasyRSA GitHub*.

All'atto del rilascio di questo documento, tale link è

https://github.com/OpenVPN/easy-rsa/releases/download/v3.0.6/EasyRSA-unix-v3.0.6.tgz

b) Estrai il tarball scaricato

```
$ cd ~
$ tar xvf EasyRSA-unix-v3.0.6.tgz
```

5 Generazione delle credenziali elettroniche dei server e dei file funzionali al processo di cifratura

Sul *Server CA (system)*

a) Crea, all'interno della home, le directory per tutti i certificati, le chiavi e i file di configurazione dei *Client VPN*

```
$ mkdir -p ~/client-configs/keys
```

b) Proteggi opportunamente la directory *~/client-configs*

```
chmod -R 700 ~/client-configs
```

c) Posizionati nella directory *EasyRSA-v3.0.6*

```
$ cd ~/EasyRSA-v3.0.6/
```

d) Copia in *vars* il file *vars.example* fornito come esempio di configurazione di *Easy-RSA*

```
$ cp vars.example vars
```

e) Modifica così il contenuto del file *vars*

```
. . .
set_var EASYRSA_REQ_COUNTRY     "IT"
set_var EASYRSA_REQ_PROVINCE    "Italy"
set_var EASYRSA_REQ_CITY        "Rome"
set_var EASYRSA_REQ_ORG         "C.N.R."
set_var EASYRSA_REQ_EMAIL       "helpdesk@iasi.cnr.it"
set_var EASYRSA_REQ_OU          "I.A.S.I."

. . .
```

f) Inizializza la *PKI*

```
$ ./easyrsa init-pki
```

Output

```
. . .
init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /home/system/EasyRSA-v3.0.6/pki
```

g) Genera la *CA*, il suo certificato pubblico (*ca.crt*) e la sua chiave private (*ca.key)*

```
$ ./easyrsa build-ca nopass
```

Quando richiesto

```
Common Name (eg: your user, host, or server name) [Easy-RSA CA]
```

non confermare la scelta di default, specificando in sua vece *IASI CA*

Output

```
. . .
CA creation complete and you may now import and sign cert requests.
Your new CA certificate file for publishing is at:
/home/system/EasyRSA-v3.0.6/pki/ca.crt
```

h) Copia il certificato pubblico della *CA (ca.crt)* nella directory */client-configs/keys*

```
$ cp ~/EasyRSA-v3.0.6/pki/ca.crt ~/client-configs/keys/
```

Sul *Server VPN (root)*

a) Posizionati nella cartella ~/EasyRSA-v3.0.6

```
$ cd ~/EasyRSA-v3.0.6
```

b) Genera una firma *HMAC* per rafforzare le capacità di verifica di integrità *TLS* del server

$ openvpn --genkey --secret ta.key

c) Copia il file *~/EasyRSA-v3.0.6/ta.key* nella cartella ~/client-configs/keys del *Server CA*,

$ scp ~/EasyRSA-v3.0.6/ta.key system@150.146.5.224:~/client-configs/keys

d) Copia, dal *Server CA*, il file *~/EasyRSA-v3.0.6/pki/ca.crt* nella directory /etc/openvpn

$ scp system@150.146.5.224:~/EasyRSA-v3.0.6/pki/ca.crt /etc/openvpn

e) Copia il file *~/EasyRSA-v3.0.6/ta.key* nella directory */etc/openvpn*

$ cp ~/EasyRSA-v3.0.6/ta.key /etc/openvpn

f) Inizializza la *PKI*

$ ./easyrsa init-pki

Output

. . .

init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /home/system/EasyRSA-v3.0.6/pki

g) Genera la chiave privata del *Server VPN* (*serverVPN.key*) e la sua richiesta di certificato (*serverVPN.req*) da inoltrare alla CA

$ ./easyrsa gen-req serverVPN nopass

Quando richiesto

Common Name (eg: your user, host, or server name) [serverVPN]:

conferma la scelta di default con un *return*

Output

. . .

Keypair and certificate request completed. Your files are:
req: /home/system/EasyRSA-v3.0.6/pki/reqs/serverVPN.req
key: /home/system/EasyRSA-v3.0.6/pki/private/serverVPN.key

h) Copia la chiave privata del *Server VPN* nella directory */etc/openvpn/*

$ cp ~/EasyRSA-v3.0.6/pki/private/serverVPN.key /etc/openvpn/

i) Trasferisci, con un metodo sicuro, la richiesta di certificato del *Server VPN* (*serverVPN.req*) nella directory /tmp del *Server CA*

$ scp ~/EasyRSA-v3.0.6/pki/reqs/serverVPN.req system@150.146.5.224:/tmp

Sul Server CA (system)

a) Posizionati nella directory *EasyRSA-v3.0.6*

$ cd ~/EasyRSA-v3.0.6/

b) Importa la richiesta di certificato del *Server VPN*

$ ./easyrsa import-req /tmp/serverVPN.req serverVPN

Output

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.1.0g 2 Nov 2017

The request has been successfully imported with a short name of: serverVPN

You may now use this name to perform signing operations on this request.

c) Genera, a partire dalla richiesta, il certificato pubblico del *ServerVPN* (*serverVPN.crt*)

$ ./easyrsa sign-req server serverVPN

Output

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.1.0g 2 Nov 2017

You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request has not been cryptographically verified. Please be sure it came from a trusted source or that you have verified the request checksum with the sender.

Request subject, to be signed as a server certificate for 1080 days:

subject=
commonName = serverVPN

Type the word 'yes' to continue, or any other input to abort.
Confirm request details: yes
Using configuration from /home/system/EasyRSA-v3.0.6/pki/safessl-easyrsa.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName :ASN.1 12:'serverVPN'
Certificate is to be certified until Jan 26 14:04:54 2022 GMT (1080 days)

Write out database with 1 new entries

Data Base Updated

Certificate created at: /home/system/EasyRSA-v3.0.6/pki/issued/serverVPN.crt

Sul Server VPN (root)

a) Posizionati nella directory *EasyRSA-v3.0.6*

$ cd ~/EasyRSA-v3.0.6/

b) Crea una chiave *forte* di Diffie-Hellman da utilizzarsi nello scambio delle chiavi

$ ./easyrsa gen-dh

Output

. . .

DH parameters of size 2048 created at /home/system/EasyRSA-v3.0.6/pki/dh.pem

c) Copia il file *~/EasyRSA-v3.0.6/pki/dh.pem* nella directory */etc/openvpn/*

$ cp ~/EasyRSA-v3.0.6/pki/dh.pem /etc/openvpn/

d) Copia il certificato del *serverVPN* (*serverVPN.crt*), nella directory */etc/openvpn/*

```
$ scp system@150.146.5.224:~/EasyRSA-v3.0.6/pki/issued/serverVPN.crt
/etc/openvpn
```

6 Configurazione del *Server VPN*

Sul Server VPN (root)

a) Posizionati sulla *home* di *root*

```
$ cd ~
```

b) Crea il file di configurazione */etc/openvpn/server.conf* con il seguente contenuto

```
#################################################
# Sample OpenVPN 2.0 config file for            #
# multi-client server.                          #
#                                               #
# This file is for the server side              #
# of a many-clients <-> one-server              #
# OpenVPN configuration.                        #
#                                               #
# OpenVPN also supports                         #
# single-machine <-> single-machine             #
# configurations (See the Examples page         #
# on the web site for more info).               #
#                                               #
# This config should work on Windows            #
# or Linux/BSD systems.  Remember on            #
# Windows to quote pathnames and use            #
# double backslashes, e.g.:                     #
# "C:\\Program Files\\OpenVPN\\config\\foo.key" #
#                                               #
# Comments are preceded with '#' or ';'         #
#################################################

# Which local IP address should OpenVPN
# listen on? (optional)
;local a.b.c.d

# Which TCP/UDP port should OpenVPN listen on?
# If you want to run multiple OpenVPN instances
# on the same machine, use a different port
# number for each one.  You will need to
# open up this port on your firewall.
;port 1194
port 443

# TCP or UDP server?
proto tcp
;proto udp

# "dev tun" will create a routed IP tunnel,
# "dev tap" will create an ethernet tunnel.
# Use "dev tap0" if you are ethernet bridging
# and have precreated a tap0 virtual interface
# and bridged it with your ethernet interface.
# If you want to control access policies
# over the VPN, you must create firewall
# rules for the the TUN/TAP interface.
# On non-Windows systems, you can give
# an explicit unit number, such as tun0.
# On Windows, use "dev-node" for this.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel if you
# have more than one.  On XP SP2 or higher,
# you may need to selectively disable the
# Windows firewall for the TAP adapter.
# Non-Windows systems usually don't need this.
;dev-node MyTap

# SSL/TLS root certificate (ca), certificate
# (cert), and private key (key).  Each client
# and the server must have their own cert and
# key file.  The server and all clients will
# use the same ca file.
#
# See the "easy-rsa" directory for a series
# of scripts for generating RSA certificates
# and private keys.  Remember to use
# a unique Common Name for the server
# and each of the client certificates.
#
# Any X509 key management system can be used.
```

```
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca ca.crt
cert serverVPN.crt
key serverVPN.key  # This file should be kept secret

# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh2048.pem 2048
dh dh.pem

# Network topology
# Should be subnet (addressing via IP)
# unless Windows clients v2.0.9 and lower have to
# be supported (then net30, i.e. a /30 per client)
# Defaults to net30 (not recommended)
;topology subnet

# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.8.0.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server

# on 10.8.0.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 10.8.0.0 255.255.255.0

# Maintain a record of client <-> virtual IP address
# associations in this file.  If OpenVPN goes down or
# is restarted, reconnecting clients can be assigned
# the same virtual IP address from the pool that was
# previously assigned.
ifconfig-pool-persist /var/log/openvpn/ipp.txt

# Configure server mode for ethernet bridging.
# You must first use your OS's bridging capability
# to bridge the TAP interface with the ethernet
# NIC interface.  Then you must manually set the
# IP/netmask on the bridge interface, here we
# assume 10.8.0.4/255.255.255.0.  Finally we
# must set aside an IP range in this subnet
# (start=10.8.0.50 end=10.8.0.100) to allocate
# to connecting clients.  Leave this line commented
# out unless you are ethernet bridging.
;server-bridge 10.8.0.4 255.255.255.0 10.8.0.50 10.8.0.100

# Configure server mode for ethernet bridging
# using a DHCP-proxy, where clients talk
# to the OpenVPN server-side DHCP server
# to receive their IP address allocation
# and DNS server addresses.  You must first use
# your OS's bridging capability to bridge the TAP
# interface with the ethernet NIC interface.
# Note: this mode only works on clients (such as
# Windows), where the client-side TAP adapter is
# bound to a DHCP client.
;server-bridge

# Push routes to the client to allow it
# to reach other private subnets behind
# the server.  Remember that these
# private subnets will also need
# to know to route the OpenVPN client
# address pool (10.8.0.0/255.255.255.0)
# back to the OpenVPN server.
;push "route 192.168.10.0 255.255.255.0"
;push "route 192.168.20.0 255.255.255.0"

# To assign specific IP addresses to specific
# clients or if a connecting client has a private
# subnet behind it that should also have VPN access,
# use the subdirectory "ccd" for client-specific
# configuration files (see man page for more info).

# EXAMPLE: Suppose the client
# having the certificate common name "Thelonious"
# also has a small subnet behind his connecting
# machine, such as 192.168.40.128/255.255.255.248.
# First, uncomment out these lines:
;client-config-dir ccd
;route 192.168.40.128 255.255.255.248
# Then create a file ccd/Thelonious with this line:
#   iroute 192.168.40.128 255.255.255.248
# This will allow Thelonious' private subnet to
# access the VPN.  This example will only work
# if you are routing, not bridging, i.e. you are
# using "dev tun" and "server" directives.

# EXAMPLE: Suppose you want to give
# Thelonious a fixed VPN IP address of 10.9.0.1.
# First uncomment out these lines:
;client-config-dir ccd
;route 10.9.0.0 255.255.255.252
# Then add this line to ccd/Thelonious:
#   ifconfig-push 10.9.0.1 10.9.0.2

# Suppose that you want to enable different
# firewall access policies for different groups
# of clients.  There are two methods:
```

```
# (1) Run multiple OpenVPN daemons, one for each
#     group, and firewall the TUN/TAP interface
#     for each group/daemon appropriately.
# (2) (Advanced) Create a script to dynamically
#     modify the firewall in response to access
#     from different clients.  See man
#     page for more info on learn-address script.
;learn-address ./script

# If enabled, this directive will configure
# all clients to redirect their default
# network gateway through the VPN, causing
# all IP traffic such as web browsing and
# and DNS lookups to go through the VPN
# (The OpenVPN server machine may need to NAT
# or bridge the TUN/TAP interface to the internet
# in order for this to work properly).
;push "redirect-gateway def1 bypass-dhcp"
;push "redirect-gateway def1 bypass-dhcp"
push "redirect-gateway autolocal def1 bypass-dhcp"
;push "redirect-gateway autolocal"

# Certain Windows-specific network settings
# can be pushed to clients, such as DNS
# or WINS server addresses.  CAVEAT:
# http://openvpn.net/faq.html#dhcpcaveats
# The addresses below refer to the public
# DNS servers provided by opendns.com.
;NEXT push "dhcp-option DNS 208.67.222.222"
;NEXT push "dhcp-option DNS 208.67.220.220"

# Uncomment this directive to allow different
# clients to be able to "see" each other.
# By default, clients will only see the server.
# To force clients to only see the server, you
# will also need to appropriately firewall the
# server's TUN/TAP interface.
;client-to-client

# Uncomment this directive if multiple clients
# might connect with the same certificate/key
# files or common names.  This is recommended
# only for testing purposes.  For production use,
# each client should have its own certificate/key
# pair.
#
# IF YOU HAVE NOT GENERATED INDIVIDUAL
# CERTIFICATE/KEY PAIRS FOR EACH CLIENT,
# EACH HAVING ITS OWN UNIQUE "COMMON NAME",
# UNCOMMENT THIS LINE OUT.
;duplicate-cn

# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
# the other side has gone down.
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120

# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
#
# Generate with:
#   openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
tls-auth ta.key 0 # This file is secret
key-direction 0

# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC
auth SHA256

# Enable compression on the VPN link and push the
# option to the client (v2.4+ only, for earlier
# versions see below)
;compress lz4-v2
;push "compress lz4-v2"

# For compression compatible with older clients use comp-lzo
# If you enable it here, you must also
# enable it in the client config file.
;comp-lzo

# The maximum number of concurrently connected
# clients we want to allow.
;max-clients 100

# It's a good idea to reduce the OpenVPN
```

```
# daemon's privileges after initialization.
#
# You can uncomment this out on
# non-Windows systems.
user nobody
group nogroup

# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
persist-key
persist-tun

# Output a short status file showing
# current connections, truncated
# and rewritten every minute.
status /var/log/openvpn/openvpn-status.log

# By default, log messages will go to the syslog (or
# on Windows, if running as a service, they will go to
# the "\Program Files\OpenVPN\log" directory).
# Use log or log-append to override this default.
# "log" will truncate the log file on OpenVPN startup,
# while "log-append" will append to it.  Use one
# or the other (but not both).
log         /var/log/openvpn/openvpn.log
log-append  /var/log/openvpn/openvpn.log

# Set the appropriate level of log
# file verbosity.
#
# 0 is silent, except for fatal errors
# 4 is reasonable for general usage
# 5 and 6 can help to debug connection problems
# 9 is extremely verbose
verb 3

# Silence repeating messages.  At most 20
# sequential messages of the same message
# category will be output to the log.
;mute 20

# Notify the client that when the server restarts so it
# can automatically reconnect.
explicit-exit-notify 0
```

c) Sostituisci così il contenuto del file di configurazione */etc/sysctl.conf*

```
#
# /etc/sysctl.conf - Configuration file for setting system variables
# See /etc/sysctl.d/ for additional system variables.
# See sysctl.conf (5) for information.
#

#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3

##############################################################3
# Functions previously found in netbase
#

# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1


###################################################################
# Additional settings - these settings can improve the network
# security of the host and prevent against some network attacks
# including spoofing attacks and man in the middle attacks through
# redirection. Some network environments, however, require that these
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
#net.ipv4.conf.all.accept_redirects = 0
#net.ipv6.conf.all.accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net.ipv4.conf.all.secure_redirects = 1
#
# Do not send ICMP redirects (we are not a router)
```

```
#net.ipv4.conf.all.send_redirects = 0
#
# Do not accept IP source route packets (we are not a router)
#net.ipv4.conf.all.accept_source_route = 0
#net.ipv6.conf.all.accept_source_route = 0
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1
#

###################################################################
# Magic system request Key
# 0=disable, 1=enable all
# Debian kernels have this set to 0 (disable the key)
# See https://www.kernel.org/doc/Documentation/sysrq.txt
# for what other values do
#kernel.sysrq=1

###################################################################
# Protected links
#
# Protects against creating or following links under certain conditions
# Debian kernels have both set to 1 (restricted)
# See https://www.kernel.org/doc/Documentation/sysctl/fs.txt
#fs.protected_hardlinks=0
#fs.protected_symlinks=0
```

d) Rendi effettive le modifiche apportate al file */etc/sysctl.conf*

```
$ sysctl -p
```

e) Sostituisci il contenuto del file di configurazione */etc/ufw/before.rules*

```
#
# rules.before
#
# Rules that should be run before the ufw command line added rules. Custom
# rules should be added to one of these chains
#   ufw-before-input
#   ufw-before-output
#   ufw-before-forward
#

# START OPENVPN RULES
# NAT table rules
*nat
:POSTROUTING ACCEPT [0:0]
# Allow traffic from OpenVPN client to eth0 (change to the interface you discovered!)
-A POSTROUTING -s 10.8.0.0/8 -o eth0 -j MASQUERADE
COMMIT
# END OPENVPN RULES

# Don't delete these required lines, otherwise there will be errors
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines

# allow all on loopback
-A ufw-before-input -i lo -j ACCEPT
-A ufw-before-output -o lo -j ACCEPT

# quickly process packets for which we already have a connection
-A ufw-before-input -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-output -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-forward -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

# drop INVALID packets (logs these in loglevel medium and higher)
-A ufw-before-input -m conntrack --ctstate INVALID -j ufw-logging-deny
-A ufw-before-input -m conntrack --ctstate INVALID -j DROP

# ok icmp codes for INPUT
-A ufw-before-input -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-input -p icmp --icmp-type source-quench -j ACCEPT
-A ufw-before-input -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-input -p icmp --icmp-type parameter-problem -j ACCEPT
-A ufw-before-input -p icmp --icmp-type echo-request -j ACCEPT

# ok icmp code for FORWARD
-A ufw-before-forward -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type source-quench -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type parameter-problem -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type echo-request -j ACCEPT

# allow dhcp client to work
-A ufw-before-input -p udp --sport 67 --dport 68 -j ACCEPT

#
# ufw-not-local
#
-A ufw-before-input -j ufw-not-local

# if LOCAL, RETURN
-A ufw-not-local -m addrtype --dst-type LOCAL -j RETURN
```

```
# if MULTICAST, RETURN
-A ufw-not-local -m addrtype --dst-type MULTICAST -j RETURN

# if BROADCAST, RETURN
-A ufw-not-local -m addrtype --dst-type BROADCAST -j RETURN

# all other non-local packets are dropped
-A ufw-not-local -m limit --limit 3/min --limit-burst 10 -j ufw-logging-deny
-A ufw-not-local -j DROP

# allow MULTICAST mDNS for service discovery (be sure the MULTICAST line above
# is uncommented)
-A ufw-before-input -p udp -d 224.0.0.251 --dport 5353 -j ACCEPT

# allow MULTICAST UPnP for service discovery (be sure the MULTICAST line above
# is uncommented)
-A ufw-before-input -p udp -d 239.255.255.250 --dport 1900 -j ACCEPT

# don't delete the 'COMMIT' line or these rules won't be processed
COMMIT
```

f) Sostituisci così il contenuto del file di configurazione */etc/default/ufw*

```
# /etc/default/ufw
#

# Set to yes to apply rules to support IPv6 (no means only IPv6 on loopback
# accepted). You will need to 'disable' and then 'enable' the firewall for
# the changes to take affect.
IPV6=yes

# Set the default input policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_INPUT_POLICY="DROP"

# Set the default output policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_OUTPUT_POLICY="ACCEPT"

# Set the default forward policy to ACCEPT, DROP or REJECT.  Please note that
# if you change this you will most likely want to adjust your rules
DEFAULT_FORWARD_POLICY="ACCEPT"

# Set the default application policy to ACCEPT, DROP, REJECT or SKIP. Please
# note that setting this to ACCEPT may be a security risk. See 'man ufw' for
# details
DEFAULT_APPLICATION_POLICY="SKIP"

# By default, ufw only touches its own chains. Set this to 'yes' to have ufw
# manage the built-in chains too. Warning: setting this to 'yes' will break
# non-ufw managed firewall rules
MANAGE_BUILTINS=no

#
# IPT backend
#
# only enable if using iptables backend
IPT_SYSCTL=/etc/ufw/sysctl.conf

# Extra connection tracking modules to load. Complete list can be found in
# net/netfilter/Kconfig of your kernel source. Some common modules:
# nf_conntrack_irc, nf_nat_irc: DCC (Direct Client to Client) support
# nf_conntrack_netbios_ns: NetBIOS (samba) client support
# nf_conntrack_pptp, nf_nat_pptp: PPTP over stateful firewall/NAT
# nf_conntrack_ftp, nf_nat_ftp: active FTP supporot
# nf_conntrack_tftp, nf_nat_tftp: TFTP support (server side)
IPT_MODULES="nf_conntrack_ftp nf_nat_ftp nf_conntrack_netbios_ns"
```

g) Abilita le connessioni tcp sulla porta 443

```
ufw allow 443/tcp
```

h) Abilita le connessioni ssh

```
ufw allow OpenSSH
```

i) Segni e riaccendi il firewall

```
ufw disable
ufw enable
```

7 Attivazione del *Server VPN*

a) Rimuovi la cartella *~/EasyRSA-v3.0.6*

```
$ rm -rf ~/EasyRSA-v3.0.6/
```

a) Verifica che nella directory */etc/openvpn* sia presente un solo file con estensione *.conf*

```
$ ls -l /etc/openvpn
```

b) Avvia il *Server VPN*

```
$ systemctl start openvpn@server
```

c) Controlla se il servizio è attivo

```
$ systemctl status openvpn@server
```

Output

```
● openvpn.service - OpenVPN service
   Loaded: loaded (/lib/systemd/system/openvpn.service; enabled; vendor preset: enabled)
   Active: active (exited) since Tue 2019-02-12 09:37:25 CET; 21min ago
   Process: 7641 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
   Main PID: 7641 (code=exited, status=0/SUCCESS)

Feb 12 09:37:25 vpnServer systemd[1]: Starting OpenVPN service...
Feb 12 09:37:25 vpnServer systemd[1]: Started OpenVPN service.

root@vpnServer:~# systemctl status openvpn@server
● openvpn@server.service - OpenVPN connection to server
   Loaded: loaded (/lib/systemd/system/openvpn@.service; indirect; vendor preset
   Active: active (running) since Wed 2019-02-13 12:20:20 CET; 5h 0min ago
     Docs: man:openvpn(8)
           https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
           https://community.openvpn.net/openvpn/wiki/HOWTO
 Main PID: 5944 (openvpn)
   Status: "Initialization Sequence Completed"
    Tasks: 1 (limit: 2319)
   CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
           └─5944 /usr/sbin/openvpn --daemon ovpn-server --status /run/openvpn/s

Feb 13 12:20:20 vpnServer systemd[1]: Starting OpenVPN connection to server...
Feb 13 12:20:20 vpnServer systemd[1]: Started OpenVPN connection to server.
```

8 Predisposizione del *Server CA* alla generazione delle credenziali dei *Client VPN*

Sul *Server CA (system)*

a) Crea una nuova directory dove memorizzare le credenziali dei *Client VPN*

```
$ mkdir -p ~/client-configs/files
```

b) Crea il file *~/client-configs/baseWin.conf* con il seguente contenuto

```
##############################################
# Sample client-side OpenVPN 2.0 config file #
# for connecting to multi-client server.     #
#                                            #
# This configuration can be used by multiple #
# clients, however each client should have   #
# its own cert and key files.                #
#                                            #
# On Windows, you might want to rename this  #
# file so it has a .ovpn extension           #
##############################################

# add somewhere
key-direction 1
auth-nocache

# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
client

# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun
```

```
# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel
# if you have more than one.  On XP SP2,
# you may need to disable the firewall
# for the TAP adapter.
;dev-node MyTap

# Are we connecting to a TCP or
# UDP server?  Use the same setting as
# on the server.
proto tcp
;proto udp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 150.146.5.253 443
;remote my-server-2 1194

# Choose a random host from the remote
# list for load-balancing.  Otherwise
# try hosts in the order specified.
;remote-random

# Keep trying indefinitely to resolve the
# host name of the OpenVPN server.  Very useful
# on machines which are not permanently connected
# to the internet such as laptops.
resolv-retry infinite

# Most clients don't need to bind to
# a specific  port number.
nobind

# Downgrade privileges after initialization (non-Windows only)
;user nobody
;group nogroup

# Try to preserve some state across restarts.
persist-key
persist-tun

# If you are connecting through an
# HTTP proxy to reach the actual OpenVPN
# server, put the proxy server/IP and
# port number here.  See the man page
# if your proxy server requires
# authentication.
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]

# Wireless networks often produce a lot
# of duplicate packets.  Set this flag
# to silence duplicate packet warnings.
;mute-replay-warnings

# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
# for each client.  A single ca
# file can be used for all clients.
;ca ca.crt
;cert client.crt
;key client.key

# Verify server certificate by checking that the
# certicate has the correct key usage set.
# This is an important precaution to protect against
# a potential attack discussed here:
#  http://openvpn.net/howto.html#mitm
#
# To use this feature, you will need to generate
# your server certificates with the keyUsage set to
#   digitalSignature, keyEncipherment
# and the extendedKeyUsage to
#   serverAuth
# EasyRSA can do this for you.
remote-cert-tls server

# If a tls-auth key is used on the server
# then every client must also have the key.
;tls-auth ta.key 1

# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC
auth SHA256

# Enable compression on the VPN link.
# Don't enable this unless it is also
# enabled in the server config file.
#comp-lzo

# Set log file verbosity.
verb 3
```

```
# Silence repeating messages
;mute 20

# If your client is running Linux and has an /etc/openvpn/update-resolv-conf file,
# uncomment these lines from the client's configuration file after it has been generated.
# script-security 2
# up /etc/openvpn/update-resolv-conf
# down /etc/openvpn/update-resolv-conf
```

c) Crea il file *~/client-configs/baseUbuntu.conf* con il seguente contenuto

```
##############################################
# Sample client-side OpenVPN 2.0 config file #
# for connecting to multi-client server.     #
#                                            #
# This configuration can be used by multiple #
# clients, however each client should have   #
# its own cert and key files.                #
#                                            #
# On Windows, you might want to rename this  #
# file so it has a .ovpn extension           #
##############################################

# add somewhere
key-direction 1
auth-nocache

# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
client

# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel
# if you have more than one.  On XP SP2,
# you may need to disable the firewall
# for the TAP adapter.
;dev-node MyTap

# Are we connecting to a TCP or
# UDP server?  Use the same setting as
# on the server.
proto tcp
;proto udp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 150.146.5.253 443
;remote my-server-2 1194

# Choose a random host from the remote
# list for load-balancing.  Otherwise
# try hosts in the order specified.
;remote-random

# Keep trying indefinitely to resolve the
# host name of the OpenVPN server.  Very useful
# on machines which are not permanently connected
# to the internet such as laptops.
resolv-retry infinite

# Most clients don't need to bind to
# a specific  port number.
nobind

# Downgrade privileges after initialization (non-Windows only)
user nobody
group nogroup

# Try to preserve some state across restarts.
persist-key
persist-tun

# If you are connecting through an
# HTTP proxy to reach the actual OpenVPN
# server, put the proxy server/IP and
# port number here.  See the man page
# if your proxy server requires
# authentication.
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]

# Wireless networks often produce a lot
# of duplicate packets.  Set this flag
# to silence duplicate packet warnings.
;mute-replay-warnings

# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
```

```
# for each client.  A single ca
# file can be used for all clients.
;ca ca.crt
;cert client.crt
;key client.key

# Verify server certificate by checking that the
# certicate has the correct key usage set.
# This is an important precaution to protect against
# a potential attack discussed here:
#  http://openvpn.net/howto.html#mitm
#
# To use this feature, you will need to generate
# your server certificates with the keyUsage set to
#   digitalSignature, keyEncipherment
# and the extendedKeyUsage to
#   serverAuth
# EasyRSA can do this for you.
remote-cert-tls server

# If a tls-auth key is used on the server
# then every client must also have the key.
;tls-auth ta.key 1

# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC
auth SHA256

# Enable compression on the VPN link.
# Don't enable this unless it is also
# enabled in the server config file.
#comp-lzo

# Set log file verbosity.
verb 3

# Silence repeating messages
;mute 20

# If your client is running Linux and has an /etc/openvpn/update-resolv-conf file,
# uncomment these lines from the client's configuration file after it has been generated.
# script-security 2
# up /etc/openvpn/update-resolv-conf
# down /etc/openvpn/update-resolv-conf
```

d) Crea il file *~/client-configs/baseMacOS.conf* con il seguente contenuto

```
################################################
# Sample client-side OpenVPN 2.0 config file #
# for connecting to multi-client server.     #
#                                            #
# This configuration can be used by multiple #
# clients, however each client should have   #
# its own cert and key files.                #
#                                            #
# On Windows, you might want to rename this  #
# file so it has a .ovpn extension           #
################################################

# add somewhere
key-direction 1
auth-nocache

# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
client

# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel
# if you have more than one.  On XP SP2,
# you may need to disable the firewall
# for the TAP adapter.
;dev-node MyTap

# Are we connecting to a TCP or
# UDP server?  Use the same setting as
# on the server.
proto tcp
;proto udp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 150.146.5.253 443
;remote my-server-2 1194
```

```
# Choose a random host from the remote
# list for load-balancing.  Otherwise
# try hosts in the order specified.
;remote-random

# Keep trying indefinitely to resolve the
# host name of the OpenVPN server.  Very useful
# on machines which are not permanently connected
# to the internet such as laptops.
resolv-retry infinite

# Most clients don't need to bind to
# a specific  port number.
nobind

# Downgrade privileges after initialization (non-Windows only)
user nobody
group nogroup

# Try to preserve some state across restarts.
persist-key
persist-tun

# If you are connecting through an
# HTTP proxy to reach the actual OpenVPN
# server, put the proxy server/IP and
# port number here.  See the man page
# if your proxy server requires
# authentication.
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]

# Wireless networks often produce a lot
# of duplicate packets.  Set this flag
# to silence duplicate packet warnings.
;mute-replay-warnings

# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
# for each client.  A single ca
# file can be used for all clients.
;ca ca.crt
;cert client.crt
;key client.key

# Verify server certificate by checking that the
# certicate has the correct key usage set.
# This is an important precaution to protect against
# a potential attack discussed here:
#  http://openvpn.net/howto.html#mitm
#
# To use this feature, you will need to generate
# your server certificates with the keyUsage set to
#   digitalSignature, keyEncipherment
# and the extendedKeyUsage to
#   serverAuth
# EasyRSA can do this for you.
remote-cert-tls server

# If a tls-auth key is used on the server
# then every client must also have the key.
;tls-auth ta.key 1

# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC
auth SHA256

# Enable compression on the VPN link.
# Don't enable this unless it is also
# enabled in the server config file.
#comp-lzo

# Set log file verbosity.
verb 3

# Silence repeating messages
;mute 20

# If your client is running Linux and has an /etc/openvpn/update-resolv-conf file,
# uncomment these lines from the client's configuration file after it has been generated.
# script-security 2
# up /etc/openvpn/update-resolv-conf
# down /etc/openvpn/update-resolv-conf
```

e) Crea il file *~/client-configs/makeConfigs.sh* con il seguente contenuto

```
#!/bin/bash

# First argument: Client identifier

KEY_DIR=~/client-configs/keys
OUTPUT_DIR=~/client-configs/files
BASE_CONFIG=~/client-configs/baseWin.conf
```

```
cat ${BASE_CONFIG} |
   <(echo -e '<ca>') |
   ${KEY_DIR}/ca.crt |
   <(echo -e '</ca>|n<cert>') |
   ${KEY_DIR}/${1}.crt |
   <(echo -e '</cert>|n<key>') |
   ${KEY_DIR}/${1}.key |
   <(echo -e '</key>|n<tls-auth>') |
   ${KEY_DIR}/ta.key |
   <(echo -e '</tls-auth>') |
   > ${OUTPUT_DIR}/${1}.ovpn
```

f) Rendi lo script eseguibile

```
$ chmod 700 ~/client-configs/makeWinConfig.sh
```

9 Revision history

Data	Versione	Descrizione	Autori
22 febbraio 2019	1.0	• Rilascio	Carlo Gaibisso Bruno Martino
04 marzo 2019	1.1	• Apportate modifiche trascurabili	Carlo Gaibisso Bruno MArtino

10 Task

Task	Data Inserimento	Data Inizio	Data Termine	Note
Prevedere l'utilizzo di un file system criptato	04/03/2019	dd/gg/aaaa	dd/gg/aaaa	
Introdurre una sezione sullke soluzioni di sivurezza adottate	04/03/2019	dd/gg/aaaa	dd/gg/aaaa	