OpenVPN gateway internet [CentOS 6.6]

Usare OpenVPN per accedere ad un'infrastruttura e uscire su internet direttamente dal server VPN.

Lo scenario è quello di avere dei consulenti in giro per clienti che si collegano ad internet per mezzo del proxy del cliente, questo blocca le connessioni di tutti i client, a partire da quello di posta (Outlook, Thunderbird, Mail, ecc…)

Iniziamo con la configurazione del server.

L'articolo tratta l'installazione del software su un sistema operativo Debian Squeeze, ma a pacchetti installati, le informazioni sono utilizzabili sulle più diffuse distribuzioni.

Diamo per scontato che la porta 443 TCP verso il vostro server sia raggiungibile.

Il primo step è naturalmente quello di installare openvpn:

```
# yum install openvpn.x86_64
```

Generazione dei certificati

Il pacchetto di OpenVPN fornisce una serie di script già pronti atti a tale scopo nel path */usr/share/doc/openvpn-2.2.2/easy-rsa/2.0/*:

```
# ls /usr/share/doc/openvpn-2.2.2/easy-rsa/2.0/
build-ca     build-key-pass    build-req-pass  Makefile          pkitool
vars
build-dh     build-key-pkcs12  clean-all       openssl-0.9.6.cnf  README
whichopensslcnf
build-inter  build-key-server  inherit-inter   openssl-0.9.8.cnf  revoke-
full
build-key    build-req         list-crl        openssl-1.0.0.cnf  sign-req
```

Per comodità spostiamo tutta la directory sotto */etc/openvpn/rsa/*.

```
# cp -r /usr/share/doc/openvpn-2.2.2/easy-rsa/2.0/ /etc/openvpn/rsa
# cd /etc/openvpn/rsa
```

Apriamo il file "*vars*" e editiamo i campi, questo velocizzerà la creazione dei certificato, è comodo per chi ha la necessità di creare molti

certificati.

I parametri da modificare sono i seguenti:

- KEY_SIZE
- KEY_COUNTRY
- KEY_PROVINCE
- KEY_CITY
- KEY_ORG
- KEY_EMAIL

Un esempio del file vars:

```
export KEY_SIZE=1024
...
export KEY_COUNTRY="IT"
export KEY_PROVINCE="IT"
export KEY_CITY="Roma"
export KEY_ORG="LBIT"
export KEY_EMAIL="vpn@lbit-solution.it"
```

A questo punto siamo pronti per generare la nostra **CA (certificate authority)**

```
# chmod 755 /etc/openvpn/rsa/whichopensslcnf
# chmod 755 clean-all
# source vars
NOTE: If you run ./clean-all, I will be doing a rm -rf on
/etc/openvpn/rsa/keys
# ./clean-all
```

È necessario richiamare anche lo script “clean-all” per iniziare con un ambiente pulito.

Ora possiamo generare la nostra **Certificate Authority**:

```
# chmod 755 build-ca
# chmod 755 /etc/openvpn/rsa/pkitool
# ./build-ca
Generating a 1024 bit RSA private key
.....................................................++++++
...++++++
writing new private key to 'ca.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [IT]:
State or Province Name (full name) [IT]:
Locality Name (eg, city) [Roma]:
Organization Name (eg, company) [LBIT]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) [LBIT CA]:
Email Address [vpn@lbit-solution.it]:
```

Avendo preconfigurato il file “vars” è sufficiente premere invio visto che il sistema ci propone come default i valori che avevamo inserito ad inizio procedura.
Ora possiamo creare il certificato per il server VPN:

```
# ./build-key-server GatewayVPN
```

GatewayVPN è il nome della macchina su cui sto installando il server VPN, per coerenza la coppia chiave/certificato avrà il nome dell’host su cui viene usato.

Per evitare che ad ogni riavvio di OpenVPN sia richiesta una password premere invio senza inserire nulla alla richiesta di password:

```
Generating a 1024 bit RSA private key
..................++++++
.++++++
writing new private key to 'GatewayVPN.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [IT]:
State or Province Name (full name) [IT]:
Locality Name (eg, city) [Roma]:
Organization Name (eg, company) [LBIT]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) [GatewayVPN]:
Email Address [vpn@lbit-solution.it]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:
Using configuration from /etc/openvpn/rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'IT'
stateOrProvinceName   :PRINTABLE:'IT'
localityName          :PRINTABLE:'Roma'
organizationName      :PRINTABLE:'LBIT'
commonName            :PRINTABLE:'GatewayVPN'
emailAddress          :IA5STRING:'vpn@lbit-solution.it'
Certificate is to be certified until Apr 25 13:50:00 2020 GMT (3650 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Generiamo ora il file Diffie-Hellman, necessario per l’avvio delle connessioni cifrate.

```
# chmod 755 build-dh
# ./build-dh
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
........+.............
```

Generiamo l’ultima chiave necessaria per l’instaurazione di una connessione sicura

```
# openvpn --genkey --secret keys/ta.key
```

Generazione dei certificati per i client

La procedura per generare i certificati dei client è identica a quella del server, nell’esempio li creiamo nominali per una semplice identificazione, in caso di grandi numeri è possible usare la matricola aziendale.

```
# chmod 755 build-key
#  ./build-key mcapasso
  Please edit the vars script to reflect your configuration,
  then source it with "source ./vars".
  Next, to start with a fresh PKI configuration and to delete any
  previous certificates and keys, run "./clean-all".
  Finally, you can run this tool (pkitool) to build certificates/keys.
root@webdav:/etc/openvpn/easy-rsa# source ./vars
NOTE: If you run ./clean-all, I will be doing a rm -rf on /etc/openvpn/easy-
rsa/keys
root@webdav:/etc/openvpn/easy-rsa#  ./build-key mcapasso
Generating a 1024 bit RSA private key
.........................................++++++
.................................................++++++
writing new private key to 'mcapasso.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [IT]:
State or Province Name (full name) [RM]:
Locality Name (eg, city) [Roma]:
Organization Name (eg, company) [LBIT]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) [mcapasso]:
Name []:Mirko Capasso
Email Address [supporto@lbit-solution.it]:mcapasso@lbit-solution.it

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /etc/openvpn/easy-rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'IT'
stateOrProvinceName   :PRINTABLE:'RM'
localityName          :PRINTABLE:'Roma'
organizationName      :PRINTABLE:'LBIT'
commonName            :PRINTABLE:'mcapasso'
name                  :PRINTABLE:'Mirko Capasso'
emailAddress          :IA5STRING:'mcapasso@lbit-solution.it'
Certificate is to be certified until Oct 19 14:29:37 2024 GMT (3650 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Configurazione del server

Ora andiamo a configurare il demone OpenVPN, anche in questo caso il pacchetto dovrebbe portare con se degli esempi.

```
# cp /usr/share/doc/openvpn-2.2.2/sample-config-files/server.conf
/etc/openvpn/
```

Di seguito un file di configurazione, dopo andiamo a spiegare le direttive:

```
# SERVER CONF
port 443
proto tcp
dev tun

ca rsa/keys/ca.crt
cert rsa/keys/GatewayVPN.crt
key rsa/keys/GatewayVPN.key
dh rsa/keys/dh1024.pem

client-config-dir ccd
server 10.1.1.0  255.255.255.0
route 10.1.1.0 255.255.255.0
ifconfig-pool-persist ipp.txt
cipher AES-256-CBC
comp-lzo
persist-key
persist-tun

status /var/log/openvpn-status.log 5
status-version 2
log-append /var/log/openvpn-status.log
verb 3  # verbose mode

# ROUTE THE CLIENT'S INTERNET ACCESS THROUGH THIS SERVER:
push "redirect-gateway def1"
push "remote-gateway 10.1.1.1"
push "dhcp-option DNS 8.8.8.8"
keepalive 10 60
```

La prima entry “*port*” è la porta sulla quale il servizio OpenVPN si metterà in ascolto, “*proto*” il protocollo, possiamo usare TCP o UDP, in questo scenario abbiamo scelto TCP per evitare che le connessioni UDP fossero droppate da firewall o proxy.

Non abbiamo usato la entry “*local*” poiché il nostro serve deve accettare connessioni su tutte le interfacce di rete, nel caso in cui ci fossero più interfacce ma solo una destinata al demone allora sarà necessario indicare

l’IP sul quale mettersi in ascolto, come l’esempio seguente:

```
local 10.10.256.25
```

Possiamo usare un tunnel al layer 3 del livello OSI, (**tap**) oppure un bridge di rete a livello 2 (**tun**), nel nostro file abbiamo inserito la seconda opzione.

A seguire la parte relativa ai certificati:

```
ca rsa/keys/ca.crt
cert rsa/keys/GatewayVPN.crt
key rsa/keys/GatewayVPN.key
dh rsa/keys/dh1024.pem
```

Le direttive da non dimenticare per consentire l’accesso ad internet tramite VPN sono le ultime, al posto di 10.1.1.1 va inserito l’IP della scheda tun0:

```
# ROUTE THE CLIENT'S INTERNET ACCESS THROUGH THIS SERVER:
push "redirect-gateway def1"
push "remote-gateway 10.1.1.1"
push "dhcp-option DNS 8.8.8.8"
```

Configurazione di IPTABLES

Per consentire ai client di uscire su internet tramite il gateway VPN andiamo ad abilitare il fowarding e il MASQUERADE tramite IPTABLES:

```
sysctl -w net.ipv4.ip_forward=1
iptables -t nat -A POSTROUTING -s 10.1.1.0/24 -o eth0 -j MASQUERADE
```

Se abbiamo IPTABLES configurato andiamo ad aggiungere anche le policy di ACCEPT:

```
iptables -A INPUT -i tun0 -j ACCEPT
iptables -A FORWARD -i tun0 -j ACCEPT
```

Per impostare in modo permanente le regole IPTABLES sopra descritte editiamo il file /etc/sysconfig/iptables:

```
# vi /etc/sysconfig/iptables

# Firewall configuration written by system-config-firewall
# Manual customization of this file is not recommended.
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A POSTROUTING -o eth0 -j MASQUERADE
COMMIT

*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -i tun0 -j ACCEPT
-A FORWARD -i tun0 -o eth0 -j ACCEPT
-A FORWARD -i eth0 -o tun0 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

Avviare il demone di OpenVPN e configurare i certificati dei client.

Configurazione dei client

Per prima cosa dobbiamo copiarci i certificati:

- La coppia certificato/chiave per il client (i due file .key e .crt)
- Il certificato della CA del server (il file ca.crt)
- La chiave di autenticazione TLS (il file ta.key)

Il file di configurazione di una macchina Windows non è complicato ma al primo errore smette di funzionare senza scrivere nei log:

```
client
dev tun
proto tcp
remote IP_SERVER_VPN 443
resolv-retry infinite
nobind
persist-key
persist-tun
# THE CSR FILE:
ca "C:\\Program Files\\OpenVPN\\config\\ca.crt"
cert "C:\\Program Files\\OpenVPN\\config\\dtricarico.crt"
key "C:\\Program Files\\OpenVPN\\config\\dtricarico.key"
ns-cert-type server
cipher AES-256-CBC
comp-lzo
redirect-gateway def1
verb 3
route-method exe
route-delay 2
```

OpenVPN gateway internet [Debian]

Usare OpenVPN per accedere ad un'infrastruttura e uscire su internet direttamente dal server VPN.

Lo scenario è quello di avere dei consulenti in giro per clienti che si collegano ad internet per mezzo del proxy del cliente, questo blocca le connessioni di tutti i client, a partire da quello di posta (Outlook, Thunderbird, Mail, ecc…)

Iniziamo con la configurazione del server.

L'articolo tratta l'installazione del software su un sistema operativo Debian Squeeze, ma a pacchetti installati, le informazioni sono utilizzabili sulle più diffuse distribuzioni.

Diamo per scontato che la porta 443 TCP verso il vostro server sia raggiungibile.

Il primo step è naturalmente quello di installare openvpn:

```
# apt-get install openvpn
```

Generazione dei certificati

Il pacchetto di OpenVPN fornisce una serie di script già pronti atti a tale scopo nel path */usr/share/doc/openvpn/examples/easy-rsa/2.0/*:

```
# ls /usr/share/doc/openvpn/examples/easy-rsa/2.0/
build-ca          build-key-server  Makefile              sign-req
build-dh          build-req         openssl-0.9.6.cnf.gz  vars
build-inter       build-req-pass    openssl.cnf           whichopensslcnf
build-key         clean-all         pkitool
build-key-pass    inherit-inter     README.gz
build-key-pkcs12  list-crl          revoke-full
```

Per comodità spostiamo tutta la directory sotto */etc/openvpn/rsa/*.

```
# cp -r /usr/share/doc/openvpn/examples/easy-rsa/2.0/ /etc/openvpn/rsa
# cd /etc/openvpn/rsa
```

Apriamo il file "*vars*" e editiamo i campi, questo velocizzerà la creazione

dei certificato, è comodo per chi ha la necessità di creare molti certificati.

I parametri da modificare sono i seguenti:

- KEY_SIZE
- KEY_COUNTRY
- KEY_PROVINCE
- KEY_CITY
- KEY_ORG
- KEY_EMAIL

Un esempio del file vars:

```
export KEY_SIZE=1024
...
export KEY_COUNTRY="IT"
export KEY_PROVINCE="IT"
export KEY_CITY="Roma"
export KEY_ORG="LBIT"
export KEY_EMAIL="vpn@lbit-solution.it"
```

A questo punto siamo pronti per generare la nostra **CA (certificate authority)**

```
# source vars
NOTE: If you run ./clean-all, I will be doing a rm -rf on
/etc/openvpn/rsa/keys
# ./clean-all
```

È necessario richiamare anche lo script “clean-all” per iniziare con un ambiente pulito.

Ora possiamo generare la nostra **Certificate Authority**:

```
# ./build-ca
Generating a 1024 bit RSA private key
.....................................................++++++
...++++++
writing new private key to 'ca.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [IT]:
State or Province Name (full name) [IT]:
Locality Name (eg, city) [Roma]:
Organization Name (eg, company) [LBIT]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) [LBIT CA]:
Email Address [vpn@lbit-solution.it]:
```

Avendo preconfigurato il file “vars” è sufficiente premere invio visto che il sistema ci propone come default i valori che avevamo inserito ad inizio procedura.
Ora possiamo creare il certificato per il server VPN:

```
# ./build-key-server GatewayVPN
```

GatewayVPN è il nome della macchina su cui sto installando il server VPN, per coerenza la coppia chiave/certificato avrà il nome dell’host su cui viene usato.

Per evitare che ad ogni riavvio di OpenVPN sia richiesta una password premere invio senza inserire nulla alla richiesta di password:

```
Generating a 1024 bit RSA private key
..................++++++
.++++++
writing new private key to 'GatewayVPN.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [IT]:
State or Province Name (full name) [IT]:
Locality Name (eg, city) [Roma]:
Organization Name (eg, company) [LBIT]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) [GatewayVPN]:
Email Address [vpn@lbit-solution.it]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:
Using configuration from /etc/openvpn/rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'IT'
stateOrProvinceName   :PRINTABLE:'IT'
localityName          :PRINTABLE:'Roma'
organizationName      :PRINTABLE:'LBIT'
commonName            :PRINTABLE:'GatewayVPN'
emailAddress          :IA5STRING:'vpn@lbit-solution.it'
Certificate is to be certified until Apr 25 13:50:00 2020 GMT (3650 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Generiamo ora il file Diffie-Hellman, necessario per l’avvio delle connessioni cifrate.

```
# ./build-dh
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
........+.............
```

Generiamo l’ultima chiave necessaria per l’instaurazione di una connessione sicura

```
# openvpn --genkey --secret keys/ta.key
```

Generazione dei certificati per i client

La procedura per generare i certificati dei client è identica a quella del server, nell’esempio li creiamo nominali per una semplice identificazione, in caso di grandi numeri è possible usare la matricola aziendale.

```
#  ./build-key mcapasso
  Please edit the vars script to reflect your configuration,
  then source it with "source ./vars".
  Next, to start with a fresh PKI configuration and to delete any
  previous certificates and keys, run "./clean-all".
  Finally, you can run this tool (pkitool) to build certificates/keys.
root@webdav:/etc/openvpn/easy-rsa# source ./vars
NOTE: If you run ./clean-all, I will be doing a rm -rf on /etc/openvpn/easy-
rsa/keys
root@webdav:/etc/openvpn/easy-rsa#  ./build-key mcapasso
Generating a 1024 bit RSA private key
.....................................++++++
.....................................++++++
writing new private key to 'mcapasso.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [IT]:
State or Province Name (full name) [RM]:
Locality Name (eg, city) [Roma]:
Organization Name (eg, company) [LBIT]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) [mcapasso]:
Name []:Mirko Capasso
Email Address [supporto@lbit-solution.it]:mcapasso@lbit-solution.it

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /etc/openvpn/easy-rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'IT'
stateOrProvinceName   :PRINTABLE:'RM'
localityName          :PRINTABLE:'Roma'
organizationName      :PRINTABLE:'LBIT'
commonName            :PRINTABLE:'mcapasso'
name                  :PRINTABLE:'Mirko Capasso'
emailAddress          :IA5STRING:'mcapasso@lbit-solution.it'
Certificate is to be certified until Oct 19 14:29:37 2024 GMT (3650 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Configurazione del server

Ora andiamo a configurare il demone OpenVPN, anche in questo caso il pacchetto dovrebbe portare con se degli esempi.

```
# cp /usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz
/etc/openvpn/
# cd /etc/openvpn
# gunzip server.conf.gz
```

Di seguito un file di configurazione, dopo andiamo a spiegare le direttive:

```
# SERVER CONF
port 443
proto tcp
dev tun

ca rsa/keys/ca.crt
cert rsa/keys/GatewayVPN.crt
key rsa/keys/GatewayVPN.key
dh rsa/keys/dh1024.pem

client-config-dir ccd
server 10.1.1.0  255.255.255.0
route 10.1.1.0 255.255.255.0
ifconfig-pool-persist ipp.txt
cipher AES-256-CBC
comp-lzo
persist-key
persist-tun

status /var/log/openvpn-status.log 5
status-version 2
log-append /var/log/openvpn-status.log
verb 3  # verbose mode

# ROUTE THE CLIENT'S INTERNET ACCESS THROUGH THIS SERVER:
push "redirect-gateway def1"
push "remote-gateway 10.1.1.1"
push "dhcp-option DNS 8.8.8.8"
keepalive 10 60
```

La prima entry “*port*” è la porta sulla quale il servizio OpenVPN si metterà in ascolto, “*proto*” il protocollo, possiamo usare TCP o UDP, in questo scenario abbiamo scelto TCP per evitare che le connessioni UDP fossero droppate da firewall o proxy.

Non abbiamo usato la entry “*local*” poiché il nostro serve deve accettare connessioni su tutte le interfacce di rete, nel caso in cui ci fossero più

interfacce ma solo una destinata al demone allora sarà necessario indicare l'IP sul quale mettersi in ascolto, come l'esempio seguente:

```
local 10.10.256.25
```

Possiamo usare un tunnel al layer 3 del livello OSI, (**tap**) oppure un bridge di rete a livello 2 (**tun**), nel nostro file abbiamo inserito la seconda opzione.

A seguire la parte relativa ai certificati:

```
ca rsa/keys/ca.crt
cert rsa/keys/GatewayVPN.crt
key rsa/keys/GatewayVPN.key
dh rsa/keys/dh1024.pem
```

Le direttive da non dimenticare per consentire l'accesso ad internet tramite VPN sono le ultime, al posto di 10.1.1.1 va inserito l'IP della scheda tun0:

```
# ROUTE THE CLIENT'S INTERNET ACCESS THROUGH THIS SERVER:
push "redirect-gateway def1"
push "remote-gateway 10.1.1.1"
push "dhcp-option DNS 8.8.8.8"
```

Configurazione di IPTABLES

Per consentire ai client di uscire su internet tramite il gateway VPN andiamo ad abilitare il fowarding e il MASQUERADE tramite IPTABLES:

```
sysctl -w net.ipv4.ip_forward=1
iptables -t nat -A POSTROUTING -s 10.1.1.0/24 -o eth0 -j MASQUERADE
```

Se abbiamo IPTABLES configurato andiamo ad aggiungere anche le policy di ACCEPT:

```
iptables -A INPUT -i tun0 -j ACCEPT
iptables -A FORWARD -i tun0 -j ACCEPT
```

Avviare il demone di OpenVPN e configurare i certificati dei client.

Configurazione dei client

Per prima cosa dobbiamo copiarci i certificati:

- La coppia certificato/chiave per il client (i due file .key e .crt)
- Il certificato della CA del server (il file ca.crt)

- La chiave di autenticazione TLS (il file ta.key)

Il file di configurazione di una macchina Windows non è complicato ma al primo errore smette di funzionare senza scrivere nei log:

```
client
dev tun
proto tcp
remote IP_SERVER_VPN 443
resolv-retry infinite
nobind
persist-key
persist-tun
# THE CSR FILE:
ca "C:\\Program Files\\OpenVPN\\config\\ca.crt"
cert "C:\\Program Files\\OpenVPN\\config\\dtricarico.crt"
key "C:\\Program Files\\OpenVPN\\config\\dtricarico.key"
ns-cert-type server
cipher AES-256-CBC
comp-lzo
redirect-gateway def1
verb 3
route-method exe
route-delay 2
```

DoS Apache - IDS e Firewall HTTP

DoS Apache – Prevenire attacchi Denial of Service e Distribuited Denial of Service con mod_evasive e mod_security

MOD EVASIVE

Proteggere il nostro webserver senza ricorrere a sistemi IDS particolarmente complessi o costosi è possibile, mod_evasive e mod_security sono i due moduli da installare e configurare per prevenire attacchi per Denial of Service (Dos) e Distribuited Denial of Service (DDoS), il primo lavora come un IDS, mentre il secondo usa delle regole similari ad un firewall.

Iniziamo impostando i valori di TimeOut e KeepAlive:

- La direttiva **RequestReadTimeout** consente di limitare il tempo di un client per effettuare una richiesta .
- Il valore della direttiva **TimeOut** dovrebbe essere abbassato su siti che

sono oggetto di attacchi DoS , è opportuno impostare questo a partire da un paio di secondi . Un valore troppo basso porterà problemi con l'esecuzione di script CGI che richiedo molto tempo per il loro completamento.

- Il parametro per la direttiva **KeepAliveTimeout** può essere abbassato anche su siti che sono oggetto di attacchi DoS . Disattivare il **KeepAlive** con impostazione Off, così come accade per alcuni siti, produce inconvenienti prestazionali, se impostata su On*, permette di usare, come da specifiche HTTP/1.1, la stessa connessione TCP per inviare più file, è pertanto consigliata questa configurazione, che evita l'apertura di una connessione TCP per ogni richiesta HTTP*.

Il mod_evasive intercetta e blocca un determinato indirizzo IP che svolge un determinato numero di richieste in un breve lasso di tempo.

Prima di procedere installiamo alcuni pacchetti fondamentali

```
# yum install make autoconf
# yum install gcc httpd-devel pcre-devel
# yum install libxml2 libxml2-devel curl curl-devel
```

Passiamo all'installazione,può essere fatta tramite yum:

```
# yum install -y mod_evasive
```

oppure scaricando il pacchetto e compilandolo:

```
# cd /usr/src
# wget
http://www.zdziarski.com/blog/wpcontent/uploads/2010/02/mod_evasive_1.10.1.ta
r.gz
# tar xzf mod_evasive_1.10.1.tar.gz
# cd mod_evasive
# apxs -cia mod_evasive20.c
```

Passiamo ora alla configurazione:

```
# vi /etc/httpd/conf/httpd.conf
```

Abilitiamo il modulo e inseriamo le direttive:

```
LoadModule evasive20_module   /usr/lib64/httpd/modules/mod_evasive20.so
```

Editiamo il file

```
# vim /etc/httpd/conf.d/mod_evasive.conf
```

Inseriamo le entry di base:

```
# mod_evasive configuration
LoadModule evasive20_module modules/mod_evasive20.so
<IfModule mod_evasive20.c>
```

```
    DOSHashTableSize    3097
    DOSPageCount        2
    DOSSiteCount        50
    DOSPageInterval     1
    DOSSiteInterval     1
    DOSBlockingPeriod   10
    DOSEmailNotify      dos@lbit-solution.it
    #DOSSystemCommand    "su - someuser -c '/sbin/... %s ...'"
    DOSLogDir           "/var/log/httpd/mod_evasive"
    DOSWhitelist  95.110.245.202
    #DOSWhitelist   192.168.0.*
</IfModule>
```

Ora vediamo nel dettaglio le direttive:

- **DOSHashTableSize**: dimensione della tabella di hash per la collezione dei dati di campionamento.
- **DOSPageCount**: identifica la soglia di richiesta di una stessa pagina da parte di un host in un certo intervallo di tempo.
- **DOSSiteCount**: identifica la soglia di richiesta di un qualsiasi oggetto da parte di un host in un certo intervallo di tempo.
- **DOSPageInterval**: intevallo di tempo per la soglia del parametro **DOSPageCount** in secondi.
- **DOSSiteInterval**: intevallo di tempo per la soglia del parametro **DOSSiteCount** in secondi.
- **DOSBlockingPeriod**: parametro che specifica l'intervallo di tempo utilizzato per mostare l'http error 403 ai client che stanno eseguendo un probabile attacco DoS.
- **DOSEmailNotify**: parametro che specifica l'indirizzo mail al quale inviare una mail di notifica, se un certo indirizzo IP sta eseguendo un probabile attacco Dos.
- **DOSWhitelist**: con questo parametro è possibile aggiungere una lista di IP che non devono essere bloccati dal modulo, nella configurazione di esempio abbiamo applicato la regola per l'indirizzo IP 95.110.245.202
- **DOSLogDir**: specifica un path alternativo alla temp directory per la collezione dei dati.
- **DOSSystemCommand**: lancia uno specifico comando quando viene superata la soglia da parte di un client. Per ricavare l'indirizzo IP che ha sfortato la soglia si deve usare la variabile "%s".

Per testare che tutto sia funzionante, e che le nostre richieste vengano bloccate possiamo usare uno script PERL:

```
#!/usr/bin/perl
# test.pl: small script to test mod_dosevasive's effectiveness
  use IO::Socket;
  use strict;
  for(0..100) {
      my($response);
      my($SOCKET) = new IO::Socket::INET( Proto   => "tcp",
```

```
                                        PeerAddr=> "127.0.0.1:80");
      if (! defined $SOCKET) { die $!; }
      print $SOCKET "GET /?$_ HTTP/1.0\n\n";
      $response = <$SOCKET>;
      print $response;
      close($SOCKET);
  }
```

Il risultato del test sarà il seguente:

```
HTTP/1.1 403 Forbidden
HTTP/1.1 403 Forbidden
HTTP/1.1 403 Forbidden
HTTP/1.1 403 Forbidden
HTTP/1.1 403 Forbidden
HTTP/1.1 403 Forbidden
HTTP/1.1 403 Forbidden
HTTP/1.1 403 Forbidden
```

MOD SECURITY

Anche per il mod_security vale la stessa regola del mod_evasive per l'installazione, possiamo scegliere se installarlo tramite repository oppure compilarlo.

Installazione tramite yum:

```
# yum install mod_security
```

Oppure scaricare il pacchetto ed installarlo:

```
# cd /usr/src
# wget http://www.modsecurity.org/download/modsecurity-apache_2.6.6.tar.gz
# tar xzf modsecurity-apache_2.6.6.tar.gz
# cd modsecurity-apache_2.6.6
# ./configure
# make install
# cp modsecurity.conf-recommended /etc/httpd/conf.d/modsecurity.conf
```

File di configurazione di mod_security

1. **/etc/httpd/conf.d/mod_security.conf** – file di configurazione principale del modulo mod_security di Apache
2. **/etc/httpd/modsecurity.d/** – tutti gli altri file di configurazione modulo Apache mod_security.
3. **/etc/httpd/modsecurity.d/modsecurity_crs_10_config.conf** – La

configurazione presente in questo file deve essere personalizzata in base alle vostre esigenze prima di essere messa in esercizio.
4. **/var/log/httpd/modsec_debug.log** – Usa i messaggi di debug per il debugging e altri problemi
5. **/var/log/httpd/modsec_audit.log** – Tutte le richieste che attivano ModSecuirty (come rilevato) o gli errori server ("RelevantOnly") vengono scritti nel file di log.

Editiamo il file /etc/httpd/modsecurity.d/modsecurity_crs_10_config.conf

```
# vi  /etc/httpd/modsecurity.d/modsecurity_crs_10_config.conf
```

E attiviamo la protezione del webserver

```
# SecRuleEngine On
```

Riavviamo il servizio httpd

```
# service httpd restart
```

Vediamo dal file di log se non si sono problemi:

```
# tail -f /var/log/httpd/error_log
```

Abbiamo terminato l'installazine dei due moduli che ridurranno gli attacchi, ora in base all'hardware e alle proprie esigenze andranno configurati tutti i servizi.

Scarica il PDF Proteggere Apache da attacchi DoS e DDoS.

[NETFILTER] Bloccare lista di IP con iptables

Aumentare la sicurezza del nostro firewall bloccando gli indirizzi IP noti per attacchi

Il sito internet BLOCKLIST.DE mette a disposizione file txt contenenti gli indirizzi IP che hanno attaccato i loro clienti nelle ultime 48 ore.

E' possible scaricarsi le liste suddivise per attacco, SSH, DDOS, FTP, MAIL, SPAM, ecc.. oppure una lista completa di tutti gli IP

all'indirizzo http://lists.blocklist.de/lists/.

Vediamo ora come interagire con il nostri iptables senza modificare la configurazione ottimale raggiunta con ore ed ore di test.

Uno script bash si occupa di scaricare la lista dal sito blocklist.de, ne legge il suo contenuto e, prima prova a togliere la regola per evitare di avere regole ridondanti:

```
/sbin/iptables -D INPUT -t filter -s $blkip -j DROP 2> /dev/null
```

e poi ne applica una

```
/sbin/iptables -A INPUT -t filter -s $blkip -j DROP
```

per ultima cosa scrive il comando per la rimozione della regola in un file, in modo da poterlo richiamare qualora si volesse pulire la catena di INPUT dagli IP sella black list senza buttare giù il firewall.

```
echo "/sbin/iptables -D INPUT -t filter -s $blkip -j DROP" >> $DROPRULE
```

Di seguito lo script, da lanciare con ./iptables-blk.sh start

```
#! /bin/sh
start(){
echo "start"
BLACKFILE="/usr/local/etc/blacklist.txt"
DROPRULE="/usr/local/etc/blacklistdrop.txt"
BLOCKLIST_URL="http://lists.blocklist.de/lists/all.txt"
> $DROPRULE
curl $BLOCKLIST_URL |grep -oE
'((1?[0-9][0-9]?|2[0-4][0-9]|25[0-5])\.){3}(1?[0-9][0-9]?|2[0-4][0-9]|25[0-5]
)' > $BLACKFILE
if [ $? -eq 0 ]
  then
  for blkip in `cat $BLACKFILE`; do
    echo "ACCESSO NEGATO A: $blkip"
    /sbin/iptables -D INPUT -t filter -s $blkip -j DROP 2>/dev/null
    /sbin/iptables -A INPUT -t filter -s $blkip -j DROP
    echo "/sbin/iptables -D INPUT -t filter -s $blkip -j DROP" >> $DROPRULE
  done
  else
  echo "$BLOCKLIST_URL ERROR"
fi
}

stop(){
```

```
DROPRULE="/usr/local/etc/blacklistdrop.txt"
echo "stop"
cat $DROPRULE|while read resetrule; do
echo "$resetrule"
$($resetrule)
done
}

restart(){
stop
sleep 5
start
}

case "$1" in
start)
start
;;
stop)
stop
;;

restart)
restart
;;
*)
echo "Usage: firewall {start|stop|restart}"
exit 1
esac

exit 0
```
