OpenVPN

Overview

The OpenVPN security model is based on SSL, the industry standard for secure communications via the internet. OpenVPN implements OSI layer 2 or 3 secure network extensions using the SSL/TLS protocol. Support IPv4, IPv6.

Introduction

OpenVPN has been ported to various platforms, including Linux and Windows, and its configuration is likewise on each of these systems, so it makes it easier to support and maintain. OpenVPN can run over User Datagram Protocol (UDP) or Transmission Control Protocol (TCP) transports, multiplexing created SSL tunnels on a single TCP/UDP port. OpenVPN is one of the few VPN protocols that can make use of a proxy, which might be handy sometimes.

Limitations

Currently, unsupported OpenVPN features:

- LZO compression
- authentication without username/password

OpenVPN username is limited to 27 characters and the password to 233 characters.

OVPN Client

Property	Description
add-default-route (*yes* | *no*; Default: **no**)	Whether to add OVPN remote address as a default route.
auth (*md5* | *sha1* | *null* | *sha256* | *sha512*; Default: **sha1**)	Allowed authentication methods.
certificate (*string* | *none*; Default: **none**)	Name of the client certificate
cipher (*null* | *aes128-cbc* | *aes128-gcm* | *aes192-cbc* | *aes192-gcm* | *aes256-cbc* | *aes256-gcm* | *blowfish128*; Default: **blowfish128**)	Allowed ciphers. In order to use GCM type ciphers, the "auth" parameter must be set to "null", because GCM cipher is also responsible for "auth", if used.
comment (*string*; Default:)	Descriptive name of an item
connect-to (*IP|IPv6*; Default:)	Remote address of the OVPN server.
disabled (*yes* | *no*; Default: **yes**)	Whether the interface is disabled or not. By default it is disabled.
mac-address (*MAC*; Default:)	Mac address of OVPN interface. Will be automatically generated if not specified.
max-mtu (*integer*; Default: **1500**)	Maximum Transmission Unit. Max packet size that the OVPN interface will be able to send without packet fragmentation.

mode (*ip* | *ethernet*; Default: **ip**)	Layer3 or layer2 tunnel mode (alternatively tun, tap)
name (*string*; Default:)	Descriptive name of the interface.
password (*string*; Default: **""**)	Password used for authentication.
port (*integer*; Default: **1194**)	Port to connect to.
profile (*name*; Default: **default**)	Specifies which PPP profile configuration will be used when establishing the tunnel.
protocol (*tcp* | *udp*; Default: **tcp**)	indicates the protocol to use when connecting with the remote endpoint.
verify-server-certificate (*yes* | *no*; Default: **no**)	Checks the certificates CN or SAN against the "connect-to" parameter. The IP or hostname must be present in the server's certificate.
tls-version (*any* | *only-1.2*; Default: **any**)	Specifies which TLS versions to allow
use-peer-dns (*yes* | *no*; Default: **no**)	Whether to add DNS servers provided by the OVPN server to IP/DNS configuration.
route-nopull (*yes* | *no*; Default: **no**)	Specifies whether to allow the OVPN server to add routes to the OVPN client instance routing table.
user (*string*; Default:)	User name used for authentication.

Also, it is possible to import the OVPN client configuration from a .ovpn configuration file. Such a file usually is provided from the OVPN server side and already includes configuration so you need to worry only about a few parameters.

```
/interface/ovpn-client/import-ovpn-configuration ovpn-password=securepassword \
key-passphrase=certificatekeypassphrase ovpn-user=myuserid skip-cert-import=no
```

OVPN client supports tls authentication. The configuration of tls-auth can be added only by importing .ovpn configuration file. Using tls-auth requires that you generate a shared-secret key, this key should be added to the client configuration file .ovpn.

```
key-direction 1
<tls-auth>
#
# 2048 bit OpenVPN static key
#
-----BEGIN OpenVPN Static key V1-----
-----END OpenVPN Static key V1-----
</tls-auth>
```

OVPN Server

An interface is created for each tunnel established to the given server. There are two types of interfaces in the OVPN server's configuration

- Static interfaces are added administratively if there is a need to reference the particular interface name (in firewall rules or elsewhere) created for the particular user.
- Dynamic interfaces are added to this list automatically whenever a user is connected and its username does not match any existing static entry (or in case the entry is active already, as there can not be two separate tunnel interfaces referenced by the same name).

Dynamic interfaces appear when a user connects and disappear once the user disconnects, so it is impossible to reference the tunnel created for that use in router configuration (for example, in the firewall), so if you need a persistent rule for that user, create a static entry for him/her. Otherwise, it is safe to use dynamic configuration.

In both cases PPP users must be configured properly - static entries do not replace PPP configuration.

Properties

Property	Description
auth (*md5* | *sha1* | *null* | *sha256* | *sha512*; Default: **sha1,md5,sha256,sha512**)	Authentication methods that the server will accept.
certificate (*name* | *none*; Default: **none**)	Name of the certificate that the OVPN server will use.
cipher (*null* | *aes128-cbc* | *aes128-gcm* | *aes192-cbc* | *aes192-gcm* | *aes256-cbc* | *aes256-gcm* | *blowfish128*; Default: **aes128-cbc,blowfish128**)	Allowed ciphers.
default-profile (*name*; Default: **default**)	Default profile to use.
enabled (*yes* | *no*; Default: **no**)	Defines whether the OVPN server is enabled or not.
protocol (*tcp* | *udp*; Default: tcp)	Indicates the protocol to use when connecting with the remote endpoint.
keepalive-timeout (*integer* | *disabled*; Default: **60**)	Defines the time period (in seconds) after which the router is starting to send keepalive packets every second. If no traffic and no keepalive responses have come for that period of time (i.e. 2 * keepalive-timeout), not responding client is proclaimed disconnected
mac-address (*MAC*; Default:)	Automatically generated MAC address of the server.
max-mtu (*integer*; Default: **1500**)	Maximum Transmission Unit. Max packet size that the OVPN interface will be able to send without packet fragmentation.
mode (*ip* | *ethernet*; Default: **ip**)	Layer3 or layer2 tunnel mode (alternatively tun, tap)
netmask (*integer*; Default: **24**)	Subnet mask to be applied to the client.
port (*integer*; Default: **1194**)	Port to run the server on.
require-client-certificate (*yes* | *no*; Default: **no**)	If set to yes, then the server checks whether the client's certificate belongs to the same certificate chain.
redirect-gateway (*def1* | *disabled* | *ipv6;* Default: **disabled**)	Specifies what kind of routes the OVPN client must add to the routing table. `def1` – Use this flag to override the default gateway by using 0.0.0.0/1 and 128.0.0.0/1 rather than 0.0.0.0/0. This has the benefit of overriding but not wiping out the original default gateway. `disabled` - Do not send redirect-gateway flags to the OVPN client. `ipv6` - Redirect IPv6 routing into the tunnel on the client side. This works similarly to the def1 flag, that is, more specific IPv6 routes are added (2000::/4 and 3000::/4), covering the whole IPv6 unicast space.
enable-tun-ipv6 (*yes* | *no;* Default: **no**)	Specifies if IPv6 IP tunneling mode should be possible with this OVPN server.
ipv6-prefix-len (*integer;* Default: **64**)	Length of IPv6 prefix for IPv6 address which will be used when generating OVPN interface on the server side.
reneg-sec (*integer;* Default: **3600)**	Key renegotiate seconds, the time the server periodically renegotiates the secret key for the data channel.
push-routes (*string*; Default:)	Push route support are added in 7.14, the maximum of possible input is limited to 1400 characters.
tls-version (any | *only-1.2 ;* Default: **any**)	TLS protocol setting.
tun-server-ipv6 (*IPv6 prefix;* Default: **::**)	IPv6 prefix address which will be used when generating the OVPN interface on the server side.

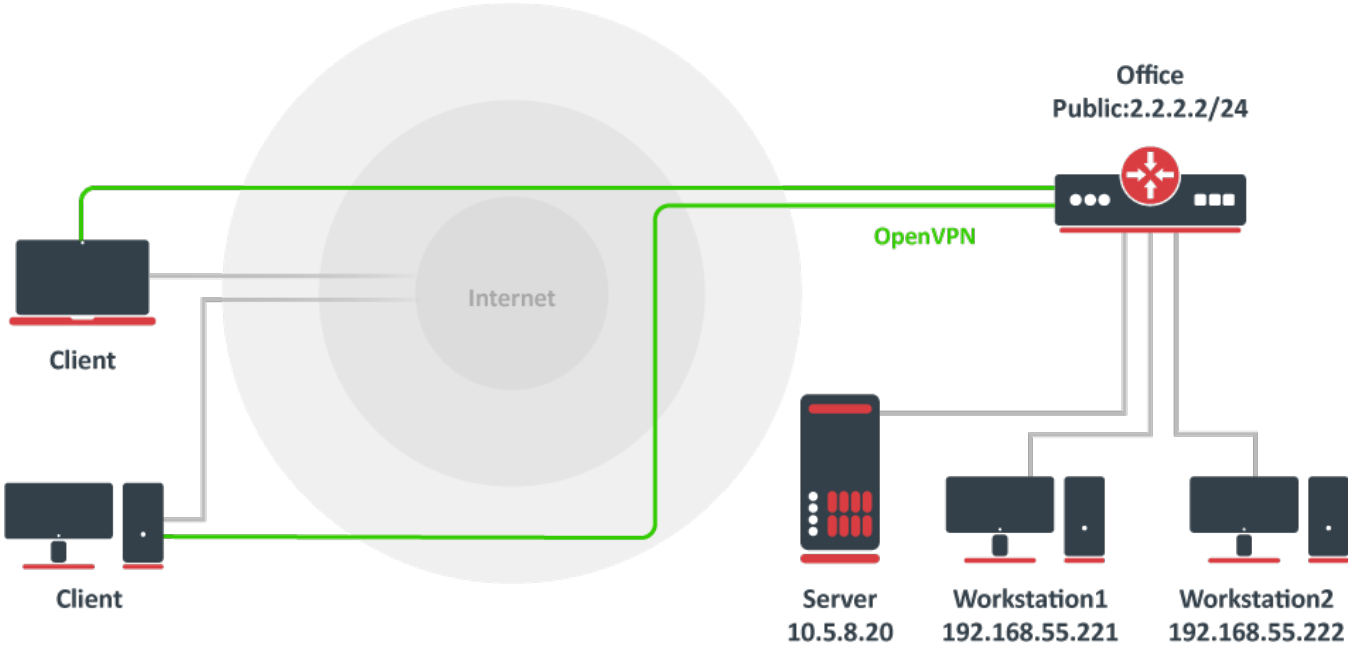
Also, it is possible to prepare a .ovpn file for the OVPN client which can be easily imported on the end device.

```
/interface/ovpn-server/server/export-client-configuration ca-certificate=myCa.crt \
client-certificate=client1.crt client-cert-key=client1.key server-address=192.168.88.1
```

It is very important that the date on the router is within the range of the installed certificate's date of expiration. To overcome any certificate verification problems, enable **NTP** date synchronization on both the server and the client.

Example

Setup Overview



Assume that Office public IP address is 2.2.2.2 and we want two remote OVPN clients to have access to 10.5.8.20 and 192.168.55.0/24 networks behind the office gateway.

Creating Certificates

All certificates can be created on the RouterOS server using the certificate manager. See example >>.

For the simplest setup, you need only an OVPN server certificate.

Server Config

The first step is to create an IP pool from which client addresses will be assigned and some users.

```
/ip pool add name=ovpn-pool range=192.168.77.2-192.168.77.254

/ppp profile add name=ovpn local-address=192.168.77.1 remote-address=ovpn-pool
/ppp secret
add name=client1 password=123 profile=ovpn
add name=client2 password=234 profile=ovpn
```

Assume that the server certificate is already created and named "server"

```
/interface ovpn-server server set enabled=yes certificate=server
```

Client Config

Push route support are added in 7.14, the maximum of possible input is limited to 1400 characters.
example: route network/IP [netmask] [gateway] [metric].

```
/interface ovpn-server server set push-routes="192.168.102.0 255.255.255.0 192.168.109.1 9"
```

To add manually which networks you want to access over the tunnel.

```
/interface ovpn-client
add name=ovpn-client1 connect-to=2.2.2.2 user=client1 password=123 disabled=no
/ip route
add dst-address=10.5.8.20 gateway=ovpn-client1
add dst-address=192.168.55.0/24 gateway=ovpn-client1
/ip firewall nat add chain=srcnat action=masquerade out-interface=ovpn-client1
```