RUTX08 VPN

The information in this page is updated in accordance with firmware version **RUTX_R_00.07.06.8**.

□

Contents

- 1 Summary
- 2 OpenVPN
  - 2.1 OpenVPN Client
  - 2.2 OpenVPN Server
    - 2.2.1 TLS Clients
    - 2.2.2 PKCS #12
    - 2.2.3 OpenVPN Server Brute-force Prevention
- 3 GRE
  - 3.1 GRE: Main & Tunnel Settings
  - 3.2 GRE: Routing Settings
- 4 IPsec
  - 4.1 IPsec Instance
    - 4.1.1 Global Secrets Settings
  - 4.2 IPsec Instance: Connection Settings
    - 4.2.1 General Settings
    - 4.2.2 Advanced settings
  - 4.3 IPsec instance: proposal settings
    - 4.3.1 Phase 1
    - 4.3.2 Phase 2
- 5 PPTP
  - 5.1 PPTP client
  - 5.2 PPTP server
- 6 SSTP
  - 6.1 SSTP configuration
- 7 Stunnel
  - 7.1 Stunnel globals
  - 7.2 Stunnel instance
- 8 DMVPN
  - 8.1 DMVPN configuration
- 9 L2TP
  - 9.1 L2TP client
  - 9.2 L2TP server
- 10 L2TPv3
  - 10.1 Local settings
  - 10.2 Peer settings

    - 10.3 Instance Settings
- 11 ZeroTier
- 12 WireGuard
    - 12.1 General Instance Settings
        - 12.1.1 Advanced Settings
    - 12.2 Peers
        - 12.2.1 General Peer Settings
        - 12.2.2 Advanced Peer Settings
- 13 Tinc
    - 13.1 General Interface Settings
        - 13.1.1 Advanced Settings
    - 13.2 Hosts
        - 13.2.1 General Tinc Host Settings

Summary

Virtual Private Network (**VPN**) is a method of connecting multiple private networks across the Internet. VPNs can serve to achieve many different goals, but some of its main purposes are:

- access between remote private networks;
- data encryption;
- anonymity when browsing the Internet.

This page is an overview of the different types of VPNs supported by RUTX08 devices.

OpenVPN

OpenVPN is an open-source software application that implements virtual private network (VPN) techniques for creating secure point-to-point or site-to-site connections in routed or bridged configurations and remote access facilities. It is often regarded as being the most universal VPN protocol because of its flexibility, support of SSL/TLS security, multiple encryption methods, many networking features and compatibility with most OS platforms.

RUTX08 devices run OpenVPN version **2.5.3**.

OpenVPN Client

An **OpenVPN client** is an entity that initiates a connection to an OpenVPN server. To create a new client instance, go to the *Services → VPN → OpenVPN* section, select *Role: Client*, enter a custom name and click the 'Add' button. An OpenVPN client instance with the given name will appear in the "OpenVPN Configuration" list.

To begin configuration, click the button that looks like a pencil next to the client instance. Refer to the figure and table below for information on the OpenVPN client's configuration fields:

Field	Value	Description
Enable	off | on; default: **off**	Turns the OpenVPN instance on or off.

Enable external Services	off |on; default: **off**	Turns the OpenVPN external Services on or off.
VPN providers	Express VPN | Nord VPN; default: **Nord VPN**	Represents a list of available VPN providers.
VPN servers	United Kingdom | USA | Australia | South Africa | Custom; default: **United Kingdom**	Represents a list of available VPN servers.
Username	string; default: **none**	Username used for authentication to the VPN server.
Password	string; default: **none**	Password used for authentication to the VPN server.
Enable OpenVPN config from file	off | on; default: **off**	Enables or disables custom OpenVPN config from file.
OpenVPN configuration file	-(interactive button)	Upload OpenVPN configuration. Warning! This will overwrite your current configuration.
Upload OpenVPN authentications files	off | on; default: **off**	Upload OpenVPN authentication files, which will be automatically included in configuration.
TUN/TAP	TUN (tunnel) | TAP (bridged); default: **TUN (tunnel)**	Virtual network device type. • **TUN** - a virtual point-to-point IP link which operates at the network layer (OSI layer 3), used when routing is required. • **TAP** - a virtual Ethernet adapter (switch), operates at the data link layer (OSI layer 2), used when bridging is required.
Protocol	UDP | TCP | **UDP6** | **TCP6**; default: **UDP**	Transfer protocol used by the OpenVPN connection. • **Transmission Control Protocol** (**TCP**) - most commonly used protocol in the Internet Protocol (IP) suite. It ensures the recipient will receive packets in the order they were sent by numbering, analysing response messages, checking for errors and resending them if an issue occurs. It should be used when reliability is crucial (for example, in file transfer). • **User Datagram Protocol** (**UDP**) - packets are sent to the recipient without error-checking or back-and-forth quality control, meaning that when packets are lost, they are gone forever. This makes it less reliable but faster than TCP; therefore, it should be used when transfer speed is crucial (for example, in video streaming, live calls).

Port	integer [0..65535]; default: **1194**	TCP/UDP port number used for the connection. Make sure it matches the port number specified on the server side. **NOTE**: traffic on the selected port will be automatically allowed in the device firewall rules.
LZO	Yes | No | None; default: **None**	Turns LZO data compression on or off.
Authentication	TLS | Static Key | Password | TLS/Password; default: **TLS**	Authentication mode, used to secure data sessions. • **Static key** is a secret key used for server–client authentication. • **TLS** authentication mode uses X.509 type certificates: - Certificate Authority (CA) - Client certificate - Client key All mentioned certificates can be generated using OpenVPN or Open SSL utilities on any type of host machine. One of the most popular utilities used for this purpose is called Easy-RSA. • **Password** is a simple username/password based authentication where the owner of the OpenVPN server provides the login data. • **TLS/Password** uses both TLS and username/password authentication.
Encryption	DES-CBC 64 | RC2-CBC 128 | DES-EDE-CBC 128 | DES-EDE3-CBC 192 | DESX-CBC 192 | BF-CBC 128 | RC2-40-CBC 40 | CAST5-CBC 128 | RC2-64CBC 64 | AES-128-CBC 128 | AES-128-CFB 128 | AES-128-CFB1 128 | AES-128-CFB8 128 | AES-128-OFB 128 | AES-128-GCM 128 | AES-192-CBC 192 | AES-192-CFB 192 | AES-192-CFB1 192 | AES-192-CFB8 192 | AES-192-OFB 192 | AES-192-GCM 192 | AES-256-CBC 256 | AES-256-CFB 256 | AES-256-CFB1 256 | AES-256-CFB8 256 | AES-256-OFB 256 | AES-256-GCM 256 | none; default: **AES-256-CBC 256**	Algorithm used for packet encryption.

TLS: TLS cipher	All | DHE+RSA | Custom; default: **All**	Packet encryption algorithm cipher.
TLS: Allowed TLS ciphers	All | DHE+RSA | Custom; default: **All**	A list of TLS ciphers accepted by this connection.
Remote host/IP address	ip; default: **none**	IP address or hostname of an OpenVPN server.
Resolve retry	integer | infinite; default: **infinite**	In case server hostname resolve fails, this field indicates the amount of time (in seconds) to retry the resolve. Specify *infinite* to retry indefinitely.
Keep alive	two integers separated by a space; default: **10 120**	Defines two time intervals: the first is used to periodically send ICMP requests to the OpenVPN server, the second one defines a time window, which is used to restart the OpenVPN service if no ICMP response is received during the specified time slice. When this value is specfiied on the OpenVPN server, it overrides the 'keep alive' values set on client instances. **Example**: *10 120*
Static key: Local tunnel endpoint IP	ip; default: **none**	IP address of the local OpenVPN network interface.
Static key: Remote tunnel endpoint IP	ip; default: **none**	IP address of the remote OpenVPN network (server) interface.
Remote network IP address	ip4; default: **none**	LAN IP address of the remote network (server).
Remote network netmask	netmask; default: **none**	LAN IP subnet mask of the remote network (server).
Remote network IPv6 address	ip6; default: **none**	IPv6 address of the remote network (server). This field is becomes visible when protocol is set to UDP6 or TCP6
Password: User name	string; default: **none**	Username used for authentication to the OpenVPN server.
Password: Password	string; default: **none**	Password used for authentication to the OpenVPN server.
TLS/Password:Use PKCS #12 format	off | on; default: **off**	Turn PKCS #12 format on or off.
Extra options	string; default: **none**	Extra OpenVPN options to be used by the OpenVPN instance.
Certificate files from device	off | on; default: **off**	Turn on this option if you want to select generated certificate files from device.
TLS/Password: HMAC authentication algorithm	none | SHA1 | SHA256 | SHA384 | SHA512; default: **SHA1**	HMAC authentication algorithm type.
TLS/Password:Additional HMAC authentication	off | on; default: **off**	An additional layer of HMAC authentication on top of the TLS control channel to protect against DoS attacks.
TLS/Password: HMAC authentication key	.key file; default: **none**	Uploads an HMAC authentication key file.

TLS/Password: HMAC key direction	0 | 1 | none; default: **1**	The value of the key direction parameter should be complementary on either side (client and server) of the connection. If one side uses *0*, the other side should use *1*, or both sides should omit the parameter altogether.
TLS/Password: Certificate authority	.ca file; default: **none**	Certificate authority is an entity that issues digital certificates. A digital certificate certifies the ownership of a public key by the named subject of the certificate.
TLS: Client certificate	.crt file; default: **none**	Client certificate is a type of digital certificate that is used by client systems to make authenticated requests to a remote server. Client certificates play a key role in many mutual authentication designs, providing strong assurances of a requester's identity.
TLS: Client key	.key file; default: **none**	Authenticates the client to the server and establishes precisely who they are.
TLS: Private key decryption password (optional)	string; default: **none**	A password used to decrypt the server's private key. Use only if server's .key file is encrypted with a password.
Static key: Static pre-shared key	.key file; default: **none**	Uploads a secret key file used for server–client authentication.

Additional notes:

- Some configuration fields become available only when certain other parameters are selected. The names of the parameters are followed by a prefix that specifies the authentication type under which they become visible. Different color codes are used for different prefixes:
  - Red for Authentication: TLS
  - Purple for Authentication: Static key
  - Blue for Authentication: Password
- After changing any of the parameters, don't forget to click the **Save & Apply** button located at the bottom-right side of the page.

OpenVPN Server

An **OpenVPN server** is an entity that waits for incoming connections from OpenVPN clients. To create a new server instance, go to the *Services → VPN → OpenVPN* section, select *Role: Server*, enter a custom name and click the 'Add' button. An OpenVPN server instance with the given name will appear in the "OpenVPN Configuration" list. Only one OpenVPN server instance is allowed to be added.

A server needs to have a public IP address in order to be available from the public network (the Internet).

To begin configuration, click the button that looks like a pencil next to the server instance. Refer to the figure and table below for information on the OpenVPN server's configuration fields:

Which SERVER LAN networks should be reachable from this client

Field	Value	Description
Enable	off | on; default: **off**	Turns the OpenVPN instance on or off.
Enable OpenVPN config from file	off | on; default: **off**	Enables or disables custom OpenVPN config from file.
TUN/TAP	TUN (tunnel) | TAP (bridged); default: **TUN (tunnel)**	Virtual network device type. • **TUN** - a virtual point-to-point IP link which operates at the network layer (OSI layer 3), used when routing is required. • **TAP** - a virtual Ethernet adapter (switch), operates at the data link layer (OSI layer 2), used when bridging is required.
Protocol	UDP | TCP | **UDP6** | **TCP6**; default: **UDP**	Transfer protocol used by the OpenVPN connection. • **Transmission Control Protocol** (**TCP**) - most commonly used protocol in the Internet Protocol (IP) suite. It ensures the recipient will receive packets in the order they were sent by numbering, analysing response messages, checking for errors and resending them if an issue occurs. It should be used when reliability is crucial (for example, file transfer). • **User Datagram Protocol** (**UDP**) - packets are sent to the recipient without error-checking or back-and-forth quality control, meaning that when packets are lost, they are gone forever. This makes it less reliable but faster than TCP; therefore, it should be used when transfer speed is crucial (for example, video streaming, live calls).
Port	integer [0..65535]; default: **1194**	TCP/UDP port number used for the connection. Make sure it matches the port number specified on the server side. **NOTE**: traffic on the selected port will be automatically allowed in device firewall rules.
LZO	Yes | No | None; default: **None**	Turns LZO data compression on or off.

Encryption	DES-CBC 64 | RC2-CBC 128 | DES-EDE-CBC 128 | DES-EDE3-CBC 192 | DESX-CBC 192 | BF-CBC 128 | RC2-40-CBC 40 | CAST5-CBC 128 | RC2-64CBC 64 | AES-128-CBC 128 | AES-128-CFB 128 | AES-128-CFB1 128 | AES-128-CFB8 128 | AES-128-OFB 128 | AES-128-GCM 128 | AES-192-CBC 192 | AES-192-CFB 192 | AES-192-CFB1 192 | AES-192-CFB8 192 | AES-192-OFB 192 | AES-192-GCM 192 | AES-256-CBC 256 | AES-256-CFB 256 | AES-256-CFB1 256 | AES-256-CFB8 256 | AES-256-OFB 256 | AES-256-GCM 256 | none; default: **AES-256-CBC 256**	Algorithm used for packet encryption.
Authentication	TLS | Static Key | TLS/Password; default: **TLS**	Authentication mode, used to secure data sessions. • **Static key** is a secret key used for server–client authentication. • **TLS** authentication mode uses X.509 type certificates: - Certificate Authority (CA) - Client certificate - Client key All mentioned certificates can be generated using OpenVPN or Open SSL utilities on any type of host machine. One of the most popular utilities used for this purpose is called Easy-RSA. • **TLS/Password** uses both TLS and username/password authentication.
Static key: Local tunnel endpoint IP	ip; default: **none**	IP address of the local OpenVPN network interface.
Static key: Remote tunnel endpoint IP	ip; default: **none**	IP address of the remote OpenVPN network (client) interface.
Static key: Remote network IP address	ip; default: **none**	LAN IP address of the remote network (client).
Static key: Remote network IP netmask	netmask; default: **none**	LAN IP subnet mask of the remote network (client).
Static key: Authentication algorithm	None | MD5 | SHA1 | SHA256 | SHA384 | SHA512; default: **SHA1**	Algorithm used for exchanging authentication and hash information.

TLS/TLS/Password: TLS cipher	All | DHE+RSA | Custom; default: **All**	Packet encryption algorithm cipher.
TLS/Password: Allowed TLS ciphers	All | DHE+RSA | Custom; default: **All**	A list of TLS ciphers accepted by this connection.
TLS/TLS/Password: Client to client	off | on; default: **off**	Allows OpenVPN clients to communicate with each other on the VPN network.
TLS/TLS/Password: Keep alive	two integers separated by a space; default: **none**	Defines two time intervals: the first is used to periodically send ICMP requests to the OpenVPN server, the second one defines a time window, which is used to restart the OpenVPN service if no ICMP response is received during the specified time slice. When this value is specifiied on the OpenVPN server, it overrides the 'keep alive' values set on client instances. **Example**: *10 120*
TLS/TLS/Password: Virtual network IP address	ip4; default: **none**	IPv4 address of the OpenVPN network.
TLS/TLS/Password: Virtual network netmask	netmask; default: **none**	Subnet mask of the OpenVPN network.
TLS/TLS/Password: **Virtual network IPv6 address**	ip6; default: **none**	IPv6 address of the OpenVPN network. This field becomes visible when protocol is set to UDP6 or TCP6
TLS/TLS/Password: Push option	OpenVPN options; default: **none**	Push options are a way to "push" routes and other additional OpenVPN options to connecting clients.
TLS/TLS/Password: Allow duplicate certificates	off | on; default: **off**	When enabled allows multiple clients to connect using the same certificates.
TLS/Password: User name	string; default: **none**	Username used for authentication to this OpenVPN server.
TLS/Password: Password	string; default: **none**	Password used for authentication to this OpenVPN server.
Static key: Static pre-shared key	.key file; default: **none**	Uploads a secret key file used for server–client authentication.
TLS/TLS/Password: Certificate authority	.ca file; default: **none**	Certificate authority is an entity that issues digital certificates. A digital certificate certifies the ownership of a public key by the named subject of the certificate.
TLS/TLS/Password: Server certificate	.crt file; default: **none**	A type of digital certificate that is used to identify the OpenVPN server.
TLS/TLS/Password: Server key	.key file; default: **none**	Authenticates clients to the server.
TLS/TLS/Password: Diffie Hellman parameters	.pem file; default: **none**	DH parameters define how OpenSSL performs the Diffie-Hellman (DH) key-exchange.
TLS/TLS/Password: CRL file (optional)	.pem file | .crl file; Default: **none**	A certificate revocation list (CRL) file is a list of certificates that have been revoked by the certificate authority (CA). It indicates which certificates are no longer acccepted by the CA and therefore cannot be authenticated to the server.

Additional notes:

- Some configuration fields become available only when certain other parameters are selected. The names of the parameters are followed by a prefix that specifies the authentication type under which they become visible. Different color codes are used for different prefixes:
  - Red for Authentication: TLS
  - Purple for Authentication: Static key
  - Blue for Authentication: TLS/Password
- After changing any of the parameters, don't forget to click the **Save & Apply** button located at the bottom-right side of the page.

TLS Clients

TLS Clients is a way to differentiate clients by their Common Names (CN), which are found in the client certificate file. It can be used to assign specific VPN addresses to corresponding clients and bind them to their LAN addresses, making the server aware of which client has which LAN IP address.

The TLS Clients section can be found in the OpenVPN Server configuration window, provided that the OpenVPN server uses TLS or TLS/Password authentication methods. To create a new TLS client, type in the new client‘s name in the text field found bellow the TLS Clients tab and click the 'Add' button. Refer to the figure and table below for information on the TLS Clients' configuration fields:

Field	Value	Description
Endpoint name	string; default: **none**	A custom name for the client.
Common name (CN)	string; default: **none**	Client’s Common Name (CN) found in the client certificate file.
Virtual local endpoint	ip; default: **none**	Client’s local address in the virtual network.
Virtual remote endpoint	ip; default: **none**	Client’s remote address in the virtual network.
Private network	ip; default: **none**	Client’s private network (LAN) IP address.
Private netmask	netmask; default: **none**	Client’s private network (LAN) IP netmask.
Covered Network	network(); default: **none**	Selects which networks should be made accessible to this client.

PKCS #12

Enable **PKCS #12 format** if you wish to use a PKCS #12 archive file format to bundle all the members of a chain of trust instead of uploading certificates separately.

PKCS #12 configuration settings become visible when the **Use PKCS #12 format** slider is turned on.

Field	Value	Description
Use PKCS #12 format	off | on; default: **off**	Turn PKCS #12 format on or off.
PKCS #12 passphrase	string; default: **none**	Passphrase used to decrypt PKCS #12 certificates.
PKCS #12 certificate chain	-(interactive button)	Use to upload certificate chain file.

OpenVPN Server Brute-force Prevention

OpenVPN Servers with **Authentication** set to **TLS/Password** or **Password**, **Protocol** set to **UDP** and running on **Port 1194** have a feature where after a client attempts to connect to the server 10 times with incorrect credentials (password and/or username) they are then blocked from the server.

To check which addresses are blocked one first needs to connect to their device's CLI.

After connecting to your device's CLI use the command **ipset list** and find the section named **ipb_port**. There under **Members** you should see all IP addresses that are blocked.

```
Name: ipb_port
Type: hash:ip,port
Revision: 5
Header: family inet hashsize 1024 maxelem 65536
Size in memory: 164
References: 2
Number of entries: 1
Members:
188.XXX.XXX.XXX,udp:1194
```

Another way to check blocked IP addresses is to use the command **ubus call ip_block show**. This will show all ip addresses that failed to connect to your device. If the **counter** atribute of the IP address entry is larger or equal then **max_attempt_count** then that IP address is blocked.

```
{
        "globals": {
                "max_attempt_count": 10
        },
        "ip_blockd 188.XXX.XXX.XXX": {
                "ip": "188.XXX.XXX.XXX",
                "port": "udp:1194",
                "counter": "1"
        },
        "ip_blockd 188.XXX.XXX.XXX": {
                "ip": "188.XXX.XXX.XXX",
                "port": "udp:1194",
                "counter": "10"
        }
}
```

To unblock a blocked client's IP address use the command **ubus call ip_block unblock**

'{"ip":"<blocked_ip_address>","port":"udp:1194"} (replace <blocked_ip_address> inside the quotes with your blocked IP address). If the IP address was unblocked succesfully you should see a similar response:

```
{
        "unblocked": {
                "ip": "188.XXX.XXX.XXX",
                "port": "udp:1194"
        }
}
```

GRE

Generic Routing Encapsulation (**GRE**) is a tunneling protocol used to establish point-to-point connections between remote private networks. GRE tunnels encapsulate data packets in order to route other protocols over IP networks.

GRE: Main & Tunnel Settings

To create a new GRE instance, go to the *Services → VPN → GRE* section, enter a custom name and click the 'Add' button. A GRE instance with the given name will appear in the "GRE Configuration" list.

To begin configuration, click the button that looks like a pencil located next to the instance. Refer to the figure and table below for information on the fields located in the GRE instance configuration section.

Field	Value	Description
Enabled	off | on; default: **off**	Turns the GRE instance on or off.
Tunnel source	network interface; default: **none**	Network interface used to establish the GRE Tunnel.
Remote endpoint IP address	ip; default: **none**	External IP address of another GRE instance used to establish the initial connection between peers.
MTU	integer; default: **1476**	Sets the maximum transmission unit (MTU) size. It is the largest size of a protocol data unit (PDU) that can be transmitted in a single network layer transaction.
TTL	integer [0..255]; default: **255**	Sets a custom TTL (Time to Live) value for encapsulated packets. TTL is a field in the IP packet header which is initially set by the sender and decreased by 1 on each hop. When it reaches 0 it is dropped and the last host to receive the packet sends an ICMP "Time Exceeded" message back to the source.
Outbound key	integer [0..65535]; default: **none**	A key used to identify outgoing packets. This value should match the "Inbound key" value set on the opposite GRE instance or both key values should be omitted on both sides.

Inbound key	integer [0..65535]; default: **none**	A key used to identify incoming packets. This value should match the "Outbound key" value set on the opposite GRE instance or both key values should be omitted on both sides.
Path MTU Discovery	off | on; default: **on**	When unchecked, sets the *nopmtudisc* option for tunnel. Can not be used together with the TTL option.
Keep alive	off | on; default: **off**	Turns "keep alive" on or off. The "keep alive" feature sends packets to the remote instance in order to determine the health of the connection. If no response is received, the device will attempt to re-establish the tunnel.
Keep alive interval	integer [0..255]; default: **none**	Frequency (in seconds) at which "keep alive" packets are sent to the remote instance.
Local GRE interface IP address	ip; default: **none**	IP address of the local GRE Tunnel network interface.
Local GRE interface netmask	netmask; default: **none**	Subnet mask of the local GRE Tunnel network interface.

GRE: Routing Settings

Routing settings are used to configure routes to networks that are behind the device that hosts the opposite GRE instance. To add a new route, simply click the 'Add' button. For information on configuring the route refer to the figure and table below.

Field	Value	Description
Remote subnet IP address	ip; default: **none**	IP address of the network behind the device that hosts the remote GRE instance.
Remote subnet netmask	netmask; default: **none**	Subnet mask of the network behind the device that hosts the remote GRE instance.

IPsec

To create a new IPsec instance, go to the *Services → VPN → IPsec* section, enter a custom name and click the 'Add' button. An IPsec instance with the given name will appear in the "IPsec Configuration" list.

IPsec Instance

The **general settings** section is used to configure the main IPsec parameters. Refer to the figure and table below for information on the configuration fields located in the general settings section.

Field	Value	Description

Enable	off | on; default: **off**	Turns the IPsec instance on or off.
Remote endpoint	host | ip; default: **none**	IP address or hostname of the remote IPsec instance.
Authentication method	Pre-shared key | X.509 | EAP | PKCS#12; default: **Pre-shared key**	Specify authentication method. Choose between Pre-shared key and X.509 certificates.
PKCS#12: PKCS12 container	string; default: **none**	
PKCS#12: PKCS12 decryption passphrase	string; default: **none**	
Pre-shared key: Pre shared key	string; default: **none**	A shared password used for authentication between IPsec peers before a secure channel is established.
X.509: | EAP: Key	A private key file; default: **none**	A private key file.
X.509: | EAP: Key decryption passphrase	A password for private key files; default: **none**	If the private key file is encrypted, the passphrase must be defined.
X.509: | EAP: Local Certificate	.der file; default: **none**	A local certificate file.
X.509: | EAP: CA Certificate	.der file; default: **none**	A certificate authority file.
Local identifier	ip | string; default: **none**	Defines how the user (left participant) will be identified during authentication. • **IP** - Internet Protocol address. • **FQDN** - identity defined by fully qualified domain name. It is the complete domain name for a host (for example, *something.somedomain.com*). Only supported with IKEv2.
Remote identifier	ip | string; default: **none**	Defines how the right participant will be identified during authentication. • **IP** - Internet Protocol address. • **FQDN** - identity defined by fully qualified domain name. It is the complete domain name for a host (for example, *something.somedomain.com*). Only supported with IKEv2.
Multiple secrets	off | on; default: **off**	Enable to show **Global Secret Settings** section for configuring multiple secrets.

Additional notes:

- Some configuration fields become available only when certain other parameters are selected. The names of the parameters are followed by a prefix that specifies the authentication type under which they become visible. Different color codes are used for different prefixes:
  - Chocolate for Authentication method: Pre-shared key
  - Dark red for Authentication method: X.509/EAP
  - Blue for Authentication method: PKCS#12

Global Secrets Settings

This section is displayed when **Multiple secrets** is enabled in General settings. You can add new instances by pressing **Add**.

Field	Value	Description
ID Selector	%any, IP or FQDN; default: **none**	Each secret can be preceded by a list of optional ID selectors. A selector is an IP address, a Fully Qualified Domain Name, user@FQDN or %any. When using IKEv1 use IP address. **NOTE:** IKEv1 only supports IP address ID selector.
Type	PSK | XAUTH | EAP | RSA | PKCS#12; default: **PSK**	IPSec secret type. **NOTE:** XAUTH secrets are IKEv1 only.
Secret	string; default: **none**	A shared password to authenticate between the peers. Minimum length is 5 symbols. All characters are allowed except `.
RSA | PKCS#12: Secret	Private key file; default: **none**	A private key file.
RSA | PKCS#12: Key decryption passphrase	A password for private key files; default: **none**	If the private key file is encrypted, the passphrase must be defined.

IPsec Instance: Connection Settings

The **connection settings** section is used to configure the main parameters of an IPsec connection. Refer to the figure and table below for information on the configuration fields located in the connection settings section.

General Settings

Field	Value	Description
Mode	Start | Add | Route; default: **Start**	Specifies which operation will be done automatically at IPSec startup.
Type	Tunnel | Transport; default: **Tunnel**	Type of connection. • **Tunnel** - protects internal routing information by encapsulating the entire IP packet (IP header and payload); commonly used in site-to-site VPN connections; supports NAT traversal. • **Transport** - only encapsulates IP payload data; used in client-to-site VPN connections; does not support NAT traversal; usually implemented with other tunneling protocols (for example, L2TP).
Tunnel: Default route	off | on; default: **off**	Turn on to route all traffic through IPSec tunnel.

Tunnel: Local subnet	ip/netmask | default: **none**	Local IP address and subnet mask used to determine which part of the network can be accessed in the VPN network. Netmask range [0..32]. If left empty, IP address will be selected automatically.
Tunnel: Remote subnet	ip/netmask; default: **none**	Remote network IP address and subnet mask used to determine which part of the network can be accessed in the VPN network. Netmask range [0..32]. This value must differ from the device's LAN IP.
Transport: Bind to	GRE interface; L2TP interface; default: **none**	Bind to GRE or L2TP interface to create GRE/L2TP over IPsec.
Key exchange	IKEv1 | IKEv2; default: **IKEv1**	Internet Key Exchange (IKE) version used for key exchange. • **IKEv1** - more commonly used but contains known issues, for example, dealing with NAT. • **IKEv2** - updated version with increased and improved capabilities, such as integrated NAT support, supported multihosting, deprecated exchange modes (does not use main or aggressive mode; only 4 messages required to establish a connection).
Enable XAuth	on | off; default: **off**	Enables Extended Authentication.

Additional notes:

- Some configuration fields become available only when certain other parameters are selected. The names of the parameters are followed by a prefix that specifies the authentication type under which they become visible. Different color codes are used for different prefixes:
  - Red for Type: Tunnel
  - Purple for Type: Transport

Advanced settings

Field	Value	Description
Aggressive	off | on; default: **off**	Turn aggressive mode on or off for outgoing connections. Aggressive mode performs fewer exchanges (a total of 4 messages) than Main mode (a total of 6 messages) by storing most data into the first exchange. In aggressive mode, the information is exchanged before there is a secure channel, making it less secure but faster than main mode. Aggressive mode is available only with IKEv1; if IKEv2 is selected this field becomes hidden.
Force encapsulation	off | on; default: **off**	Forces UDP encapsulation for ESP packets even if a "no NAT" situation is detected.
Local firewall	off | on; default: **on**	Adds necessary firewall rules to allow traffic of this IPsec instance on this device.
Remote firewall	off | on; default: **on**	Adds necessary firewall rules to allow traffic of from the opposite IPsec instance on this device.

Compatibility mode	off | on; default: **off**	Turns on compatibility mode to help deal with a 3rd party remote peer with multiple subnets.
Inactivity	integer; default: **none**	Defines timeout interval, after which a CHILD_SA is closed if it did not send or receive any traffic.
Dead Peer Detection	off | on; default: **off**	A function used during Internet Key Exchange (IKE) to detect a "dead" peer. It used to reduce traffic by minimizing the number of messages when the opposite peer in unavailable and as failover mechanism.
Dead Peer Detection: DPD action	Restart | Hold | Clear | None; default: **Restart**	Controls the use of the Dead Peer Detection protocol where notification messages are periodically sent in order to check the liveliness of the IPsec peer.
Dead Peer Detection: DPD Delay	integer; default: **none**	The frequency of sending R_U_THERE messages or INFORMATIONAL exchanges to peer.
Dead Peer Detection: DPD Timeout	integer; default: **none**	Defines the timeout interval, after which all connections to a peer are deleted in case of inactivity.
XAuth identity	string; default: **none**	The identity/username the client uses to reply to an XAuth request. If not defined, the IKEv1 identity will be used as XAuth identity.
Tunnel: Remote source IP	ip; default: **none**	The internal source IP to use in a tunnel for the remote (right) peer.
Tunnel: Local source IP	ip; default: **none**	The internal (left) source IP to use in a tunnel, also known as virtual IP.
Tunnel: Remote DNS	ip; default: **none**	List of DNS server addresses to exchange as configuration attributes. On the responder, only fixed IPv4/IPv6 addresses are allowed and define DNS servers assigned to the client.
Locally allowed protocols	string; default: **none**	Allowed protocols and ports over connection, also called Port Selectors. Defines in form of 'protocol/port' eg:'17/1701' or '17/%any' or 'udp/l2f'.
Remotely allowed protocols	string; default: **none**	Allowed protocols and ports over connection, also called Port Selectors. Defines in form of 'protocol/port' eg:'17/1701' or '17/%any' or 'udp/l2f'.
Custom option	string; default: **none**	Add custom connection parameters.
Local source IP	ip; default: **none**	The internal (left) source IP to use in a tunnel, also known as virtual IP.
Tunnel: Passthrough interfaces	network interfaces; default: **none**	Network interfaces to include in IPsec Passthrough.
Tunnel: Passthrough subnets	ip/netmask; default: **none**	Networks to include in IPsec Passthrough.

Additional notes:

- Some configuration fields become available only when certain other parameters are selected. The names of the parameters are followed by a prefix that specifies the authentication type under which they become visible. Different color codes are used for different prefixes:
  - Red for Type: Tunnel
  - Blue for Dead Peer Detection: Enabled

IPsec instance: proposal settings

The **proposal settings** section is used to configure IKE (Internet Key Exchange) phase 1 & 2 settings. IKE is a protocol used to set up security associations (SAs) for the IPsec connection. This process is required before any IPsec tunnel can be established. It is done in two phases:

Phase	Mode
Phase 1 • Establishes a secure channel between peers • Authenticates peers • Negotiates SA policy • Shares secret keys • Establishes secure tunnel for phase 2	Main mode (figure 1) • 6 packets exchanged • Identity protected during exchange
Phase 2 • Sets up matching IPsec SAs • Periodically renegotiates IPsec SAs	Quick mode • 3 packets exchanged • IPsec SA parameters (ESP/AH, SHA/MD5) established • SA lifetime set

Figure 1	Figure 2
	Aggressive mode (figure 2) • 3 packets exchanged • Identity information exchanged before a secure channel is established

Phase 1

Field	Value	Description
Encryption algorithm	3DES | AES 128 | AES 192 | AES 256 | AES128 GCM8 | AES192 GCM8 | AES256 GCM8 | AES128 GCM12 | AES192 GCM12 | AES256 GCM12 | AES128 GCM16 | AES192 GCM16 | AES256 GCM16; default: **AES 128**	Algorithm used for data encryption.
Authentication	MD5 | SHA1 | SHA256 | SHA384 | SHA512; default: **SHA1**	Algorithm used for exchanging authentication and hash information.
DH group	MODP768 | MODP1024 | MODP1536 | MODP2048 | MODP3072 | MODP4096 | ECP192 | ECP224 | ECP256 | ECP384 | ECP521 | No PFS; default: **MODP1536**	Diffie-Hellman (DH) group used in the key exchange process. Higher group numbers provide more security, but take longer and use more resources to compute the key. Must match with another incoming connection to establish IPSec.
Force crypto proposal	off | on; default: **off**	Only chosen proposals will be used when turned on.
IKE Lifetime	string; default: **none**	How long the keying channel of a connection (ISAKMP or IKE SA) should last before being renegotiated. The time is specified in seconds. The s, m, h and d suffixes explicitly define the units for seconds, minutes, hours and days, respectively.

Phase 2

Field	Value	Description
Encryption algorithm	3DES | AES 128 | AES 192 | AES 256 | AES128 GCM8 | AES192 GCM8 | AES256 GCM8 | AES128 GCM12 | AES192 GCM12 | AES256 GCM12 | AES128 GCM16 | AES192 GCM16 | AES256 GCM16; default: **3DES**	Algorithm used for data encryption.
Hash algorithm	MD5 | SHA1 | SHA256 | SHA384 | SHA512; default: **MD5**	Algorithm used for exchanging authentication and hash information.
PFS group	MODP768 | MODP1024 | MODP1536 | MODP2048 | MODP3072 | MODP4096 | ECP192 | ECP224 | ECP256 | ECP384 | ECP521 | No PFS; default: **MODP768**	The PFS (Perfect Forward Secrecy). Must match with another incoming connection to establish IPSec.
Force crypto proposal	off | on; default: **off**	Only chosen proposals will be used when turned on.
Lifetime	string; default: **none**	How long the keying channel of a connection (ISAKMP or IKE SA) should last before being renegotiated. The time is specified in seconds. The s, m, h and d suffixes explicitly define the units for seconds, minutes, hours and days, respectively.

PPTP

Point-to-Point Tunneling Protocol (**PPTP**) is a type of VPN protocol that uses a TCP control channel and a Generic Routing Encapsulation tunnel to encapsulate PPP packets.

PPTP client

A **PPTP client** is an entity that initiates a connection to a PPTP server. To create a new client instance, go to the *Services → VPN → PPTP* section, select *Role: Client*, enter a custom name and click the 'Add' button:

You should be redirected to the configuration page for the newly added PPTP Client which should look similar to this:

Field	Value	Description
Enable	off | on; default: **off**	Turns the PPTP instance on or off.

Server	ip | host; default: **none**	IP address or hostname of a PPTP server.
Username	string; default: **none**	Username used for authentication to the PPTP server.
Password	string; default: **none**	Password used for authentication to the PPTP server.
Client to client	off | on; default: **off**	Adds route to make other PPTP clients reachable.
Default route	off | on; default: **off**	When turned on, this connection will become the device default route. This means that all traffic directed to the Internet will go through the PPTP server and the server's IP address will be seen as this device's source IP to other hosts on the Internet.

PPTP server

A **PPTP server** is an entity that waits for incoming connections from PPTP clients. To create a new server instance, go to the *Services → VPN → PPTP* section, select *Role: Server*, enter a custom name and click the 'Add' button:

You should be redirected to the configuration page for the newly added PPTP Server which should look similar to this:

Field	Value	Description
Enable	off | on; default: **off**	Turns the PPTP instance on or off.
Local IP	ip; default: **192.168.0.1**	IP address of this PPTP network interface.
Remote IP range start	ip; default: **192.168.0.20**	PPTP IP address leases will begin from the address specified in this field.
Remote IP range end	ip; default: **192.168.0.30**	PPTP IP address leases will end with the address specified in this field.
Timeout	integer; default: **none**	Time in seconds. If no packets pass through the tunnel between this server and a client, the server will terminate the connection to that client after the amount of time specified in this field passes.
Primary DNS	ip; default: **none**	IP address of the primary DNS server.
Secondary DNS	ip; default: **none**	IP address of the secondary DNS server.

The **User List** section is used to keep authentication settings required to successfully connect to this server. The list is empty by default. Click the 'Add' button to create a new PPTP User:

Field	Value	Description

Username	string; default: **none**	Username used for authentication to this PPTP server.
Password	string; default: **none**	Password used for authentication to this PPTP server.
PPTP Client's IP	ip; default: **none**	Assigns an IP address to the client that uses the adjacent authentication info. This field is optional and if left empty the client will simply receive an IP address from the IP pool defined above.

Note: there can only be one PPTP Server configuration on the device.

SSTP

Secure Socket Tunneling Protocol (SSTP) is a VPN protocol designed to transport PPP traffic via a secure SSL/TLS channel.

SSTP configuration

To create a new SSTP instance, go to the Services → VPN → SSTP section, enter a custom name and click the 'Add' button. An SSTP instance with the given name will appear in the "SSTP Configuration" list.

To begin configuration, click the 'Edit' button located next to the instance. Refer to the figure and table below for information on the SSTP instance's configuration fields:

Field	Value	Description
Enable	off | on; default: **off**	Turns the SSTP instance on or off.
Server IP address	ip | host; default: **none**	IP address or hostname of a SSTP server.
User name	string; default: **none**	Username used for authentication to the SSTP server.
Password	string; default: **none**	Password used for authentication to the SSTP server.
Certificate files from device	off | on; default: **off**	Turn on this option if you want to select generated certificate files from device.
CA cert	.crt file; default: **none**	Upload a Certificate authority (CA) file.
Default route	off | on; default: **off**	When turned on, this connection will become the device default route. This means that all traffic directed to the Internet will go through the SSTP server and the server's IP address will be seen as this device's source IP to other hosts on the Internet. **NOTE**: this can only be used when Failover is turned off.

Stunnel

Stunnel is an open-source a proxy service that adds TLS encryption to clients and servers already existing on a VPN network. TLS encryption provided by Stunnel can be used as an additional layer of encryption for data sent by VPN. This procedure increases the security of the established connection and provides higher chances of passing a Deep packet inspection (DPI) check.

Stunnel globals

The Stunnel Globals section is used to manage the Stunnel service as a whole. Refer to the figure and table below for information on the fields contained in the Stunnel Globals section.

Field	Value	Description
Enabled	off | on; default: **off**	Turns the Stunnel service on or off. If this is unchecked, Stunnel instances will not start (even if they are enabled individually); therefore, it is necessary to check this field in order to make Stunnel active on the router.
Debug Level	integer [0..7]; default: **5**	Debugging to log output level. • **0 (emergency)** - a panic condition, i.e., system is no longer usable. • **1 (alert)** - a condition that must be corrected immediately. • **2 (critical)** - critical conditions, device errors. • **3 (error)** - errors that are fatal to the operation, but not the service or application (can't open a required file, missing data, etc.) Solving these types of errors will usually require user intervention. • **4 (warning)** - anything that can potentially cause application oddities, but for which the system is automatically recovering from (e.g., retrying an operation, missing secondary data, etc.) • **5 (notice)** - conditions that are not error conditions, but that may require special handling. • **6 (info)** - general useful information (e.g., configuration changes, starts and stops of services, etc.) • **7 (debug)** - contains basic information that is diagnostically helpful to most people (i.e., not just engineers).
Use alternative config	off | on; default: **off**	Turns the possibility to upload an external Stunnel configuration file on or off.if you turn this on, other Stunnel configurations present in the router will become inactive.
Upload alternative config	file; default: **none**	Uploads an Stunnel configuration file.

Stunnel instance

To create a new Stunnel instance, go to the *Services → VPN → Stunnel* section, enter a custom name and click the 'Add' button. An Stunnel instance with the given name will appear in the "Stunnel Configuration" list.

To begin configuration, click the 'Edit' button located next to the instance. Refer to the figure and table below for information on the Stunnel instance's configuration fields:

Field	Value	Description
Enable	off | on; default: **off**	Turns the Stunnel instance on or off.
Operating Mode	Server | Client; default: **Server**	Selects the Stunnel instance's role. • **Server** - listens for connecting Stunnel clients. • **Client** - listens for connecting clients and connects to an Stunnel server.
Listen IP	ip; default: **none**	Makes the instance "listen" for incoming connections on the specified IP address. When left empty, the value of this field defaults to *localhost* (*127.0.0.1*).
Listen Port	integer [0..65535]; default: **none**	Makes the instance "listen" for incoming connections on the specified TCP port. Make sure you chose a port that is not being used by another service. You will also have to allow traffic on the specified port.
Connect IP's	ip:port; default: **none**	IP:Port to listen for VPN connections. When left empty the value of this field is interpreted as *localhost*. Must contain at least one item. If multiple options are specified, remote address is chosen using a round-robin algorithm.
TLS Cipher	None | Secure | Custom; default: **None**	Packet encryption algorithm cipher.
Custom: Allowed TLS Ciphers	string; default: **none**	A list of TLS ciphers accepted for this connection.
Application Protocol	Connect | SMTP | Not specified; default: **Not specified**	This option enables initial, protocol-specific negotiation of the TLS encryption. The protocol option should not be used with TLS encryption on a separate port.
Connect: Protocol Authentication	Basic | NTLM; default: **Basic**	Authentication type for the protocol negotiations.
SMTP: Protocol Authentication	Plain | Login; default: **Plain**	Authentication type for the protocol negotiations.
Certificate File	.crt file; default: **none**	TLS client or server certificate file.
Private Key	.key file; default: **none**	TLS client or server key file.

Additional notes:

- Some configuration fields become available only when certain other parameters are selected. The names of the parameters are followed by a prefix that specifies the authentication type under which they become visible. Different color codes are used for different prefixes:
  - Blue for TLS Cipher: Custom
  - Dark red for Application Protocol: Connect
  - Chocolate for Application Protocol: SMTP

DMVPN

Dynamic Multipoint VPN (**DMVPN**) is a method of building scalable IPsec VPNs. DMVPN is configured as a hub-and-spoke network, where tunnels between spokes are built dynamically; therefore, no change in configuration is required on the hub in order to connect new spokes.

DMVPN configuration

To create a new DMVPN instance, go to the *Services → VPN → DMVPN* section, enter a custom name and click the 'Add' button. A DMVPN instance with the given name will appear in the "DMVPN Configuration" list.

To begin configuration, click the button that looks like a pencil located next to the instance. Refer to the figures and tables below for information on the DMVPN instance configuration:

Field	Value	Description
Enabled	off | on; default: **off**	Turns the DMVPN instance on or off.
Working mode	Spoke | Hub; default: **Spoke**	Selects the role of this instance • **Hub** - the central instance of DMVPN that connects other peers (spokes) into single network. There is no need to reconfigure the hub when connecting new spokes to it. • **Spoke** - an instance that connects to the hub.
Hub address	ip | host; default: **none**	IP address or hostname of a DMVPN hub.

Field	Value	Description
Tunnel source	network interface; default: **Any**	Network interface used to establish the GRE Tunnel.
Local GRE interface IP address	ip; default: **none**	IP address of the local GRE Tunnel instance.
Spoke: Remote GRE interface IP address	ip; default: **none**	IP address of the remote GRE Tunnel instance.
Hub: Local GRE interface netmask	netmask; default: **none**	Subnet mask of the local GRE Tunnel instance.
GRE MTU	integer; default: **1476**	Sets the maximum transmission unit (MTU) size. It is the largest size of a protocol data unit (PDU) that can be transmitted in a single network layer transaction.

Outbound key	integer [0..65535]; default: **none**	A key used to identify outgoing GRE packets. This value should match the **Inbound key** value set on the opposite GRE tunnel instance or both key values should be omitted on both sides.
Inbound key	integer [0..65535]; default: **none**	A key used to identify incoming GRE packets. This value should match the **Outbound key** value set on the opposite GRE instance or both key values should be omitted on both sides.

Field	Value	Description
Local identifier	ip | string; default: **none**	Defines how the user (IPsec instance) will be identified during authentication. • **IP** - Internet Protocol address. • **FQDN** - identity defined by fully qualified domain name. It is the complete domain name for a host (for example, *something.somedomain.com*). Only supported with IKEv2.
Remote identifier	ip | string; default: **none**	Defines how the right participant will be identified during authentication. • **IP** - Internet Protocol address. • **FQDN** - identity defined by fully qualified domain name. It is the complete domain name for a host (for example, *something.somedomain.com*). Only supported with IKEv2.
Pre-shared key	string; default: **none**	A shared password used for authentication between IPsec peers.
Encryption algorithm	3DES | AES 128 | AES 192 | AES 256 | AES128 GCM8 | AES192 GCM8 | AES256 GCM8 | AES128 GCM12 | AES192 GCM12 | AES256 GCM12 | AES128 GCM16 | AES192 GCM16 | AES256 GCM16; default: **AES 128**	Algorithm used for data encryption.
Authentication/Hash algorithm	MD5 | SHA1 | SHA256 | SHA384 | SHA512; default: **SHA1**	Algorithm used for exchanging authentication and hash information.

DH/PFS group	MODP768 | MODP1024 | MODP1536 | MODP2048 | MODP3072 | MODP4096 | ECP192 | ECP224 | ECP256 | ECP384 | ECP521; default: **MODP1536**	Diffie-Hellman (DH) group used in the key exchange process. Higher group numbers provide more security, but take longer and use more resources to compute the key. Must match with another incoming connection to establish IPSec.
Force crypto proposal	off | on; default: **off**	Only chosen proposals will be used when turned on.
IKE Lifetime	string; default: **none**	How long the keying channel of a connection (ISAKMP or IKE SA) should last before being renegotiated. The time is specified in seconds. The s, m, h and d suffixes explicitly define the units for seconds, minutes, hours and days, respectively.

Field	Value	Description
NHRP network ID	integer; default: **1**	An identifier used to define the NHRP domain. This is a local parameter and its value does not need to match the values specified on other domains. However, the NHRP ID is added to packets which arrive on the GRE interface; therefore, it may be helpful to use the same ID for troubleshooting purposes.
NHRP authentication key	string; default: **none**	A key used in NHRP authentication.
NHRP hold time	integer; default: **7200**	Specifies the holding time for NHRP Registration Requests and Resolution Replies sent from this interface or shortcut-target. The hold time is specified in seconds and defaults to two hours.
Redirect	off | on; default: **off**	Enables or disables DMVPN traffic redirection.
Hub: NFLOG group	integer [1..65535]; default: **none**	Specify NFLOG group to be used by NHRP instance

L2TP

In computer networking, **Layer 2 Tunneling Protocol** (**L2TP**) is a tunneling protocol used to support virtual private networks (VPNs). It is more secure than PPTP but, because it encapsulates the transferred data twice, but it is slower and uses more CPU power.

L2TP client

An **L2TP client** is an entity that initiates a connection to an L2TP server. To create a new client instance, go to the *Services → VPN → L2TP* section, select *Role: Client*, enter a custom name and

click the 'Add' button. An L2TP client instance with the given name will appear in the "L2TP Configuration" list.

To begin configuration, click the button that looks like a pencil next to the client instance. Refer to the figure and table below for information on the L2TP client's configuration fields:

Field	Value	Description
Enable	off | on; default: **off**	Turns the L2TP instance on or off.
Server	ip | host; default: **none**	IP address or hostname of an L2TP server.
Username	string; default: **none**	Username used for authentication to the L2TP server.
Password	string; default: **none**	Password used for authentication to the L2TP server.
CHAP secret	string; default: **none**	A secret used for L2TP Tunnel Authentication.
MTU	[68..9200]; default: **none**	Sets the maximum transmission unit (MTU) size. It is the largest size of a protocol data unit (PDU) that can be transmitted in a single network layer transaction.
Default route	off | on; default: **off**	When turned on, this connection will become device default route. This means that all traffic directed to the Internet will go through the L2TP server and the server's IP address will be seen as this device's source IP to other hosts on the Internet. **NOTE**: this can only be used when Failover is turned off.
Allow CHAP	on | off; default: **on**	Allows CHAP authentication method to be used.
Allow PAP	on | off; default: **off**	Allows PAP authentication method to be used.
Allow MSCHAP-v2	on | off; default: **on**	Allow MSCHAP-v2 authentication method to be used.

Active L2TP client instance example:

L2TP server

An **L2TP server** is an entity that waits for incoming connections from L2TP clients. To create a new server instance, go to the *Services → VPN → L2TP* section, select *Role: Server*, enter a custom name and click the 'Add' button. An L2TP server instance with the given name will appear in the "L2TP Configuration" list. Only one L2TP server instance is allowed to be added.

A server needs to have a public IP address in order to be available from the public network (the Internet).

To begin configuration, click the button that looks like a pencil next to the server instance. Refer to the figure and table below for information on the L2TP server's configuration fields:

Field	Value	Description
Enable	off | on; default: **off**	Turns the L2TP instance on or off.
Local IP	ip; default: **192.168.0.1**	IP address of this L2TP network interface.
Remote IP range begin	ip; default: **192.168.0.20**	L2TP IP address leases will begin from the address specified in this field.
Remote IP range end	ip; default: **192.168.0.30**	L2TP IP address leases will end with the address specified in this field.
Enable CHAP	on | off; default: **off**	Challenge-Handshake Authentication Protocol for L2TP.
CHAP authentication	on | off; default: **on**	When turned on it requires CHAP authentication method to be used, when turned off it refuses CHAP authentication method.
PAP authentication	on | off; default: **off**	When turned on it requires PAP authentication method to be used, when turned off it refuses PAP authentication method.
MSCHAP-v2 authentication	on | off; default: **on**	When turned on it requires MSCHAP-v2 authentication method to be used, when turned off it refuses MSCHAP-v2 authentication method.
User name	string; default: **user**	Username used for authentication to this L2TP server.
Password	string; default: **pass**	Password used for authentication to this L2TP server.
L2TP Client's IP	ip; default: **none**	Assigns an IP address to the client that uses the adjacent authentication info. This field is optional and if left empty the client will simply receive an IP address from the IP pool defined above.

Active L2TP server instance example:

L2TPv3

L2TPv3 (Layer Two Tunneling Protocol Version 3) is a point-to-point layer two over IP tunnel. This means you can tunnel L2 protocols like Ethernet, Frame-relay, ATM, HDLC, PPP, etc. over an IP network.

Local settings

Field	Value	Description
Enable	off | on; default: **off**	Turns the L2TPv3 instance on or off.
Local address	ip; default: **none**	IP address of device's WAN. Note: If left empty, it will take IP address automatically from main WAN interface.

Tunnel ID	integer [1..4294967295]; default: **none**	Uniquely identifies the tunnel. The value used must match the peer tunnel ID value being used at the peer.
Session ID	integer [1..4294967295]; default: **none**	The value used must match the tunnel ID value being used at the peer.
Cookie	string; default: **none**	Must be specified in hexidecimal form and be length of 8 or 16. eg.: 89ABCDEF.

Peer settings

Field	Value	Description
Peer address	ip; default: **none**	Peer Endpoint IP address.
Peer Tunnel ID	integer [1..4294967295]; default: **none**	ID of peer's tunnel is specified from 1 to 4294967295. It must match other end tunnel ID.
Peer Session ID	integer [1..4294967295]; default: **none**	UID of peer's session is specified from 1 to 4294967295. It must match other end session ID.
Peer Cookie	string; default: **none**	Must be specified in hexidecimal form and be length of 8 or 16. eg.: 89ABCDEF. It must match other end Cookie.

Instance Settings

Field	Value	Description
Bridge to	None | LAN; default: **None**	Peer Endpoint IP address.
IP Address	ip; default: **none**	IP address of standalone L2TPv3 interface.
Netmask	netmask; default: **none**	Netmask of standalone L2TPv3 interface.
MTU	integer [64..9000]; default: **none**	Sets the maximum transmission unit (MTU) size. It is the largest size of a protocol data unit (PDU) that can be transmitted in a single network layer transaction.
Encapsulation	IP | UDP; default: **IP**	Specify technology to use when connecting to other end.
Layer 2 specific header type	None | Linux Default; default: **Linux Default**	It might be neccessary to set this option to none when using other hardware.

ZeroTier

ZeroTier One is an open source software which can establish Peer to Peer VPN (P2PVPN) connection between various devices running various operating systems. It also provides network management possibilities such as routing and creating firewall rules.

Note: ZeroTier is additional software that can be installed from the **System → Package Manager**

page.

In order to create a new ZeroTier Instance, look to the Add New ZeroTier Configuration section; enter a custom name and click the 'Add' button:

You should be redirected to the configuration page for the newly added ZeroTier Instance which should look similar to this:

Field	Value	Description
Enabled	off | on; default: **off**	Turns the ZeroTier Instance on or off.

ZeroTier network configuration instance should look similar to this:

Field	Value	Description
Enabled	off | on; default: **off**	Turns the ZeroTier Instance on or off.
Port	integer [0..65535]; default: **9993**	ZeroTier Network port.
Network ID	hex string; default: **none**	ZeroTier Network ID. Log in to your ZeroTier account in order to locate the ZeroTier Network ID, which should be a string of hexadecimal characters.
Bridge to	None | LAN; default: **None**	Specify to which interface this ZeroTier instance should be bridged to.
Allow default route	off | on; default: **off**	Allows ZeroTier to override system default route
Allow global IP	off | on; default: **off**	Allows ZeroTier managed IPs and routes to overlap public IP space
Allow managed IP	off | on; default: **on**	Assigns ZeroTier managed IPs and routes
Allow DNS	off | on; default: **off**	Applies DNS servers that are set at the network controller

WireGuard

WireGuard is simple, fast, lean, and modern VPN that utilizes secure and trusted cryptography. It intends to be more performant than OpenVPN. WireGuard is designed as general purpose VPN, fit for many different circumstances and while it is currently under heavy development, it already might be regarded as the most secure, easiest to use, and simplest VPN solution.

WireGuard works by adding an interface which acts as a tunnel. To create one enter its name and click the **Add** button. This should add a new Wireguard instance and open a configuration window.

General Instance Settings

This section contains General settings of created WireGuard Instance. Here you can find its Public and Private keys and generate them, specify Port and IP addresses for communication.

Field	Value	Description
Enable	off | on; default: **off**	Turns WireGuard Instance on or off.
Private Key	string; default: **none**	Private Key used in authentication.
Public Key	string; default: **-**	Public Key used in authentication.
Generate	-(interactive button)	Click to generate Public Key and Private Key.
Listen Port	integer [0..65535]; default: **none**	Specify port to listen for incomming connections. It will be set to a random integer if left empty.
IP Addresses	ip; default: **none**	A single IP address or a list of them for this instance associated with public keys.

Advanced Settings

Advanced Settings section contains Metric and MTU configuration for this WireGuard interface.

Field	Value	Description
Metric	positive integer; default: **none**	Specify metric for this tunnel interface. Lower number means higher priority.
MTU	integer [1280..1420]; default: **none**	Maximum Transmission Unit for this tunnel interface.
DNS servers	ip | ips; default: **none**	DNS server(s) for this Wireguard interface.

Peers

The Peers section is used to create and configure all the peers for this interface. To create one enter its name and click the **Add** button. To configure it click the **Edit** button.

General Peer Settings

In the General section of Peer instance you can configure basic information about the endpoint to allow communications.

Field	Value	Description
Public Key	string; default: **none**	Endpoint's Public Key.
Allowed IPs	ip; default: **none**	A single IP address or a list of them which are allowed to communicate with this peer.
Description	string; default: **none**	Description of the peer.
Route Allowed IPs	off | on; default: **off**	Enable to create routes for **Allowed IPs** for this peer.

Advanced Peer Settings

In the Advanced section of Peer instance you are able to configure additional settings such as its Description, Endpoint Host and Port, Preshared Key and other. See more information below.

Field	Value	Description
Description	string; default: **none**	Description of this peer.
Pre-Shared Key	string; default: **none**	Base64-encoded preshared key. Adds in an additional layer of symmetric-key cryptography for post-quantum resistance.
Route Allowed IPs	off | on; default: **off**	Enable to create routes for **Allowed IPs** for this peer.
Endpoint Host	ip | url; default: **none**	IP or URL of Remote Endpoint.
Endpoint Port	integer [0..65535]; default: **none**	Specify port to connect to Remote Endpoint. It will be set to **51820** if left empty.
Persistent Keep Alive	integer [0..65535]; default: **none**	Specify time amount in seconds between Keep Alive messages. By default this option is **0** which means it is disabled. Recommended value for a device behind NAT is 25.

Tinc

Tinc is a Virtual Private Network (VPN) daemon that uses tunnelling and encryption to create a secure private network between hosts on the Internet. Tinc is Free Software and licensed under the General Public License version 2 or later. Because the VPN appears to the IP level network code as a normal network device, there is no need to adapt any existing software. This allows VPN sites to share information with each other over the Internet without exposing any information to others.

Tinc works by adding an interface which acts as Router/Switch/Hub mode. To create one enter its name and click the **Add** button. This should add a new Tinc interface and open a configuration window

General Interface Settings

This section contains General settings of created Tinc Interface. Here you can find its Public and

Private keys, specify Subnet and Host to Connect to for communication.

Field	Value	Description
Enable	off | on; default: **off**	Turns Tinc Interface on or off.
Subnet	ip; default: **none**	The subnet which this tinc daemon will serve. Multiple subnet lines can be specified for each Daemon.
Private Key	.key file; default: **none**	Generated RSA private key.
Public Key	.key file; default: **none**	Generated RSA public key.
Host to Connect to	Not specified | Host name; default: **Not specified**	Specifies which other tinc daemon to connect to on startup. Multiple variables may be specified, in which case outgoing connections to each specified tinc daemon are made.

Advanced Settings

Advanced Settings section.

Field	Value	Description
Address Family	Any | IPv4 | IPPv6; default: **Any**	This option affects the address family of listening and outgoing sockets.
Bind To Address	ip; default: **none**	Addresses for Tinc VPN to use for listening sockets.
Bind To Interface	Any | LAN | WAN | WAN6 | Mobile; default: **Any**	Interface to listen for incoming connections.
Key Expire	integer; default: **3600**	This option controls the time the encryption keys used to encrypt the data are Valid.
Mode	Router | Switch | Hub; default: **Router**	This option selects the way packets are routed to other daemons.
Ping Interval	integer; default: **60**	The number of seconds of inactivity that tinc will wait before sending a probe to the other end.
Ping Timeout	integer; default: **5**	The number of seconds to wait for a response to pings or to allow meta connections to block. If the other end doesn't respond within this time, the connection is terminated, and the others will be notified of this.

Hosts

The Hosts section is used to add your VPN hosts. To create one enter its name and click the **Add** button. To configure it click the **Edit** button.

General Tinc Host Settings

In the General section of Hosts instance you can configure basic information about the host.

Field	Value	Description
Enable	off | on; default: **off**	Turn this tinc host on/off.
Description	string; default: **none**	Optional. Description of host.
Address	ip; default: **none**	It must resolve to the external IP address where the host can be reached, not the one that is internal to the VPN.
Subnet	ip; default: **none**	The subnet which this tinc daemon will serve. Multiple subnet lines can be specified for each Daemon.
Public Key	.key file; default: **none**	Generated RSA public key.