RUT240 VPN (legacy WebUI)

The information in this page is updated in accordance with firmware version **RUT2XX_R_00.01.14.7**.

***Note**: this user manual page is for RUT240's old WebUI style available in earlier FW versions. **Click here** for information based on the latest FW version.*

□

Contents

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:

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

This chapter of the user manual provides an overview of the Firewall page for RUT240 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.

RUT240 routers run OpenVPN version **2.4.5**.

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 New' button. An OpenVPN client instance with the given name will appear in the "OpenVPN Configuration" list. A maximum of six OpenVPN client instances are allowed to be added.

To begin configuration, click the 'Edit' button 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 OpenVPN config from file	yes | no; default: **no**	Enables custom OpenVPN configuration from file.
Enable	yes | no; default: **no**	Turns the OpenVPN instance on or off.
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; default: **UDP**	Transfer protocol used for 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 router's firewall rules.
LZO	yes | no; default: **no**	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 | RC2-40-CBC 40 | CAST5-CBC 128 | RC2-64-CBC 64 | AES-128-CFB 128 | AES-128-CFB1 128 | AES-128-CFB8 128 | AES-128-OFB 128 | AES-128-CBC 128 | AES-128-GCM 128 | AES-192-CFB 192 | AES-192-CFB1 192 | AES-192-CFB8 192 | AES-192-OFB 192 | AES-192-CBC 192 | AES-192-GCM 192 | AES-256-CFB 256 | AES-256-CFB1 256 | AES-256-CFB8 256 | AES-256-OFB 256 | AES-256-CBC 256 | AES-256-GCM 256 | none ; default: **BF-CBC 128**	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 for 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: **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 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	ip; default: **none**	LAN IP address of the remote network (server).
Remote network IP netmask	netmask; default: **none**	LAN IP subnet mask of the remote network (server).
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.
Extra options	string; default: **none**	Extra OpenVPN options to be used by the OpenVPN instance.
Use PKCS #12 format	yes | no; default: **no**	Use PKCS #12 archive file format to bundle all the members of a chain of trust.
PKCS #12 passphrase	string; default: **none**	Passphrase to decrypt PKCS #12 certificates.
PKCS #12 certificate chain	string; default: **none**	Uploads PKCS #12 certificate chain file.
TLS/Password: HMAC authentication algorithm	none | SHA1 | SHA256 | SHA384 | SHA512; default: **SHA1**	HMAC authentication algorithm type.
TLS/Password: Additional HMAC authentication	none | Authentication only (tls-auth) | Authentication and encryption (tls-crypt); default: **none**	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 (CA) 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** 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 New' 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 'Edit' button next to the server instance. Refer to the figure and table below for information on the OpenVPN server's configuration fields:

Field	Value	Description
Enable OpenVPN config from file	yes | no; default: **no**	Enables custom OpenVPN configuration from file.
Enable	yes | no; default: **no**	Turns the OpenVPN instance on or off.
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; default: **UDP**	Transfer protocol used for the 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 the router's firewall rules.
LZO	yes | no; default: **no**	Turns LZO data compression on or off.
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.
Encryption	DES-CBC 64 | RC2-CBC 128 | DES-EDE-CBC 128 | DES-EDE3-CBC 192 | DESX-CBC 192 | RC2-40-CBC 40 | CAST5-CBC 128 | RC2-64-CBC 64 | AES-128-CFB 128 | AES-128-CFB1 128 | AES-128-CFB8 128 | AES-128-OFB 128 | AES-128-CBC 128 | AES-128-GCM 128 | AES-192-CFB 192 | AES-192-CFB1 192 | AES-192-CFB8 192 | AES-192-OFB 192 | AES-192-CBC 192 | AES-192-GCM 192 | AES-256-CFB 256 | AES-256-CFB1 256 | AES-256-CFB8 256 | AES-256-OFB 256 | AES-256-CBC 256 | AES-256-GCM 256 | none ; default: **BF-CBC 128**	Algorithm used for packet encryption.
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).
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 for this connection.
TLS/TLS/Password: Client to client	yes | no; default: **no**	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	ip; default: **none**	IP address of the OpenVPN network.
TLS/TLS/Password: Virtual network netmask	netmask; default: **none**	Subnet mask of the OpenVPN network.
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	yes | no; default: **no**	When enabled allows multiple clients to connect using the same certificates.
Use PKCS #12 format	yes | no; default: **no**	Use PKCS #12 archive file format to bundle all the members of a chain of trust.
PKCS #12 passphrase	string; default: **none**	Passphrase to decrypt PKCS #12 certificates.
PKCS #12 certificate chain	string; default: **none**	Uploads PKCS #12 certificate chain file.
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.
TLS/TLS/Password: Enable manual ccd upload	yes | no; default: **no**	Enable manual upload of client-config-dir files.

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** 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	ip; default: **none**	Client’s private network (LAN) IP netmask.

IPsec

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

To begin configuration, click the 'Edit' button located next to the instance.

IPsec configuration

The **IPsec configuration** 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 general settings section.

Field	Value	Description
Enable	yes | no; default: **no**	Turns the IPsec instance on or off
Enable IPv6	yes | no; default: **no**	Turns the IPv6 address of the left interface on or off
Left IPv6	IPv6 address; default: **none**	IPv6 address used as the source. If left empty, uses one of the available global addresses.
Authentication type	Pre-shared key | X.509; default: **Pre-shared key**	Authentication type accordingly to your IPsec configuration. IPsec
IKE version	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)
Mode	Main | Aggressive; default: **Main**	Internet Security and Key Management Protocol (ISAKMP) phase 1 exchange mode. • **Main** - performs three two-way exchanges between the initiator and the receiver (a total of 9 messages). • **Aggressive** - performs fewer exchanges 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
Ignore security	yes | no; default: **no**	If enabled responders are allowed to use IKEv1 Aggressive Mode with pre-shared keys. Discouraged to use due to security concerns.
Use additional xauth authentification	**yes** | no; default: **no**	Turns additional xauth authentification for this instance on or off.
Xauth password	string; default: **none**	Password for xauth.
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).

On startup	Ignore | Add | Route | Start; default: **Start**	Defines how the instance should act on router startup. • **Add** - loads a connection without starting it. • **Route** - starts the tunnel only if there is traffic. • **Start** - starts the tunnel on router startup.
My identifier	ip | string; default: **none**	Defines how the user (IPsec instance) will be identified during authentication.
Tunnel: Local IP address/Subnet mask	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.
Left firewall	off | on; default: **on**	Adds neccessary firewall rules to allow traffic of this IPsec instance on this router.
Force encapsulation	yes | no; default: **no**	Forces UDP encapsulation for ESP packets even if a "no NAT" situation is detected.
Dead Peer Detection	yes | no; default: **no**	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: Delay (sec)	integer; default: **none**	The frequency of checking whether a peer is still availaible or not.
Dead Peer Detection: Timeout (sec)	integer; default: **none**	Time limit after which the IPsec instance will stop checking the availability of a peer and determine it to be "dead" if no response is received.
Remote VPN endpoint	host | ip; default: **none**	IP address or hostname of the remote IPsec instance
Remote identifier	string | ip; default: **none**	FQDN or IP address of remote peer. Leave empty for any
Tunnel: Remote IP address/Subnet mask	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
Passthrough networks	None | LAN | Wired | WiFi | Mobile | custom; default: **none**	Select networks which should be passthrough and excluded from routing through tunnel
Right firewall	yes | no; default: **yes**	Adds neccessary firewall rules to allow traffic of from the opposite IPsec instance on this router
Allow WebUI access	yes | no; default: **no**	Allows WebUI access for hosts in the VPN network
Compatibility mode	yes | no; default: **no**	Enable this if multiple subnets do not work with a 3rd party IPsec peer.
Custom options	ipsec options; default: **none**	Provides the possibility to further customize the connection by adding extra IPsec options.

Additional notes:

- Some configuration fields become available only when certain other parameters are selected. Different color codes are used for different parameters:
  - Red for Type: Tunnel
  - Purple for Type: Transport

- Blue for Dead Peer Detection: Enabled
- After changing any of the parameters, don't forget to click the **Save** button located at the bottom-right side of the page.

Phase settings

IKE (Internet Key Exchange) is a protocol used to set up security associations (SAs) for the IPsec connection. This process is required before the 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 Aggressive mode (figure 2) • 3 packets exchanged • Identity information exchanged before a secure channel is established
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

Field	Value	Description
Encryption algorithm	DES | 3DES | AES128 | AES192 | AES256; default: **3DES**	Algorithm used for data encryption.
Authentication/Hash algorithm	MD5 | SHA1 | SHA256 | SHA384 | SHA512; default: **SHA1**	Algorithm used for exchanging authentication and hash information.
DH group/PFS group	MODP768 | MODP1024 | MODP1536 | MODP2048 | MODP3072 | MODP4096; 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.
Lifetime	integer; default: **8 hours**	Defines a time period after which the phase will re-initiate its exchange of information.

Pre-shared keys

A **pre-shared key** is a secret password used for authentication between IPsec peers before a secure tunnel is established. During authentication device will try to check if connection matches any **Secret's ID selector** and then the **pre-shared key** from the first match will be used.

To create a new key, click the 'Add' button.

The figure below is an example of the Pre-shared keys section and the table below provides

information on configuration fields contained in that section:

Field	Value	Description
Pre-shared key	string; default: **none**	A shared password used for authentication between IPsec peers before a secure channel is established.
Secret's ID selector	string; 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. **NOTE**: IKEv1 only supports IP address ID selector.

GRE Tunnel

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 Tunnel instance, go to the *Services → VPN → GRE Tunnel* 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 'Edit' button located next to the instance. Refer to the figure and table below for information on the fields located in the GRE Tunnel instance configuration section.

Field	Value	Description
Enabled	yes | no; default: **no**	Turns the GRE Tunnel 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.
Use Ipv6: Remote endpoint IPv6 address	ip; default: **none**	External IPv6 address of 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. A 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.
Don't fragment	yes | no; default: **yes**	When unchecked, sets the *nopmtudisc* option for tunnel. Can not be used together with the TTL option.
Keep alive	yes | no; default: **no**	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.
Use IPv6: Enabled Local GRE interface IPv6 address	ip; default: **none**	IPv6 address of the local GRE Tunnel network interface.

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 Use IPv6: Enabled
- After changing any of the parameters, don't forget to click the **Save** button located at the bottom-right side of the page.

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.

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 New' button. A PPTP client instance with the given name will appear in the "PPTP Configuration" list.

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

Field	Value	Description
Enable	yes | no; default: **no**	Turns the PPTP instance on or off.
Use as default gateway	yes | no; default: **no**	When turned on, this connection will become the router's 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. **NOTE**: this can only be used when WAN Failover is turned off.
Client to client	yes | no; default: **no**	Adds a route that makes other PPTP clients accessible within the PPTP network.
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.

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 New' button. A PPTP server instance with the given name will appear in the "PPTP Configuration" list. Only one PPTP 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 'Edit' button located next to the server instance. Refer to the figure and table below for information on the PPTP server's configuration fields:

Field	Value	Description
Enable	yes | no; default: **no**	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.
User name	string; default: **youruser**	Username used for authentication to this PPTP server.
Password	string; default: **yourpass**	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.

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

To begin configuration, click the 'Edit button located 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	yes | no; default: **no**	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 in authorization to the L2TP server.
Password	string; default: **none**	Password used in authorization to the L2TP server.
Authentication	string; default: **none**	Optional. Password used in L2TP tunnel CHAP authentication.
Keep alive	integer; default: **none**	Frequency (in seconds) at which LCP echo requests are sent to the remote instance in order to determine the health of the connection.

Default route	yes | no; default: **no**	When turned on, this connection will become the router's 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 WAN Failover is turned off.

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 New' 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 'Edit' button located 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	yes | no; default: **no**	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.
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.

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
Enabled	yes | no; default: **no**	Turns the SSTP instance on or off.
Use as default gateway	yes | no; default: **no**	When turned on, this connection will become the router's 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 WAN Failover is turned off.
Server IP address	ip | host; default: **none**	IP address or hostname of an SSTP server.
Username	string; default: **none**	Username used for authentication to the SSTP server.
Password	string; default: **none**	Password used for authentication to the SSTP server.
CA cert	.crt file; default: **none**	Uploads a Certificate authority (CA) file.

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.

For a more in-depth Stunnel configuration example visit this page: OpenVPN over Stunnel.

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
Use alternative config	yes | no; default: **no**	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 client/server

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	yes | no; default: **no**	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 OpenVPN 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. You can do this via the **Network → Firewall → Traffic Rulles → Open Ports On Router** section.
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.
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.
Protocol Authentication	**Connect**: Basic | NTLM; default: **Basic** **SMTP**: Plain | Login; default: **Plain**	Authentication type for the protocol negotiations.
Protocol Domain	string; default: **none**	Domain for the protocol negotiations.
Protocol Host	host:port; default: **none**	Specifies the final TLS server to be connected to by the proxy, and not the proxy server directly connected by Stunnel. The proxy server should be specified along with the *connect* option.
Protocol Username	string; Default: **none**	Username for authentication to the protocol negotiations.

Protocol Password	string; default: **none**	Password for authentication to 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.

ZeroTier

ZeroTier One is an open source software product which establishes Peer to Peer VPN (P2PVPN) connection between laptops, desktops, phones, embedded devices, cloud resources, and apps.

To make this section visible on the router, you must first install the **zerotier** package from the the **System → Packages** section.

Click here to see a usage example of ZeroTier One VPN.

ZeroTier General

The **General** section is used to enable the ZeroTier service.

Field	Value	Description
Enabled	yes | no; default: **no**	Turns the ZeroTier service on or off.
Address	string; default: **none**	Your ZeroTier address. This field is filled automatically after a successful connection.
Networks	string; default: **none**	ZeroTier network address. This value should be taken from your ZeroTier account.

ZeroTier VPN

The **VPN** section is used to turn ZeroTier VPN on or off and select its role.

Field	Value	Description
Enable VPN	yes | no; default: **no**	Turns ZeroTier VPN on or off.
Mode	Server | Client; default: **Server**	ZeroTier VPN operating mode.

See also

- Configuration examples for RUTxxx VPN services:
  - OpenVPN configuration examples
  - IPsec configuration examples

    - GRE Tunnel configuration examples
    - PPTP configuration examples
    - L2TP configuration examples
    - DMVPN configuration
- Configuration examples for third party VPN services
    - expressvpn.com
    - vpngate.net
    - vpnbook.com
    - hide.me
- Other related examples:
    - How to generate TLS certificates (Windows)?
    - L2TP over IPsec
    - OpenVPN traffic split
    - OpenVPN client on Windows