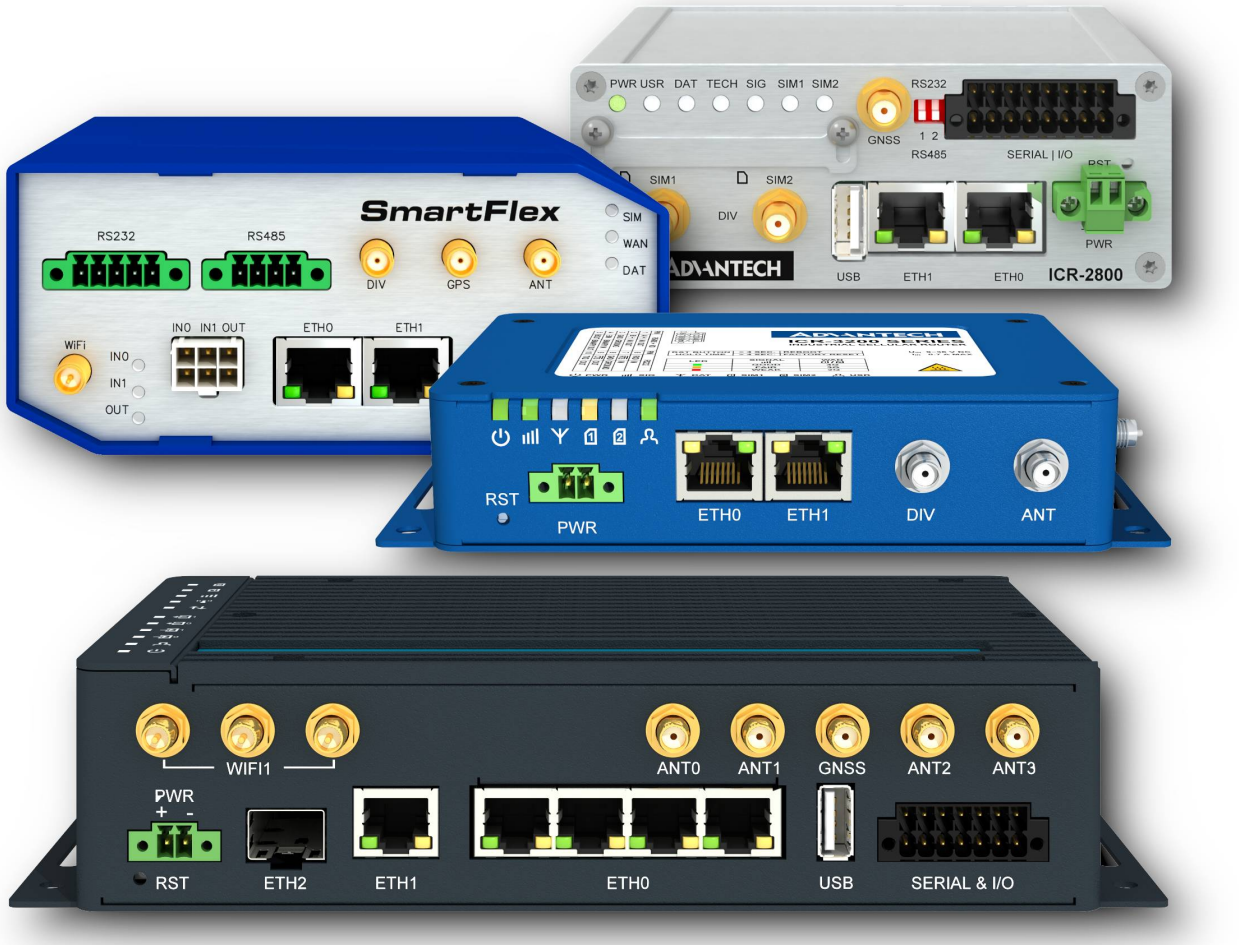
# IPsec Tunnel

## APPLICATION NOTE



# Used symbols

*Danger* – Information regarding user safety or potential damage to the router.

*Attention* – Problems that can arise in specific situations.

*Information, notice* – Useful tips or information of special interest.

*Example* – Example of function, command or script.

# GPL license

Source codes under GPL license are available free of charge by sending an email to:

techSupport@advantech-bb.com

Advantech Czech s.r.o., Sokolska 71, 562 04 Usti nad Orlici, Czech Republic.
Document No. APP-0006-EN, revision from September 1, 2022. Released in the Czech Republic.

# Contents

# List of Tables

# 1. IPsec and its Protocols

IPsec (Internet Protocol Security) is a security extension of IP protocol based on authentication and encryption of every IP datagram. Within the OSI architecture, it is security at the network layer, which means that IPsec provides security for any transfer (any network application).

IPsec pay attention to these major security issues:

- **Authenticating** – Allows to verify the origin of the data, so if a packet is received, it is possible to verify that the transmitted packet corresponds to the sender or whether the sender exists at all (Phase I, IKE phase, Main mode). At PSK ends with key exchange.
- **Encrypting** – Both of sides agree on the form of packet encryption in advance. Thereafter the entire packet apart from the IP header will be encrypted, alternatively the entire packet will be encrypted and a new IP header will be added (Phase II, IPsec phase, Quick mode). Ends with establishing of a tunnel.

IPsec consists of two basic protocols – *Authentication Header (AH)* and *Encapsulating Security Payload (ESP)*. AH protocol is not supported in router's Web interface configuration. Both protocols together are often unsupported by some gateways on the way in the Internet.

Part of IPsec is also *IKE (Internet Key Exchange)* protocol (key management). IKE creates logical channels which are called *Security Associations (SA)*. These channels are always unidirectional therefore it is necessary to use two separate channels (SA) for duplex. IKE also supports automatic generation and recovery of encryption keys.

## 1.1 Encapsulating Security Payload (ESP)

Encapsulating Security Payload (ESP) protocol ensures the confidentiality of transmitted data (encrypts packets) and optionally the original authentication, data integrity and protection against reverse queries. As with the Authentication Header (AH) protocol, additional header is attached to an IP packet. This header contains the security parameters which are followed by encrypted data. However, the outer header is not protected and its integrity is not guaranteed.

In case of requirement for encryption and authentication, system which responds first authenticates packet and if the first step is successful, continues with encryption. This type of configuration reduces both overhead of processing and vulnerability in case of attack when denial of service.

### 1.1.1 Usage of Encapsulating Security Payload protocol

ESP protocol can be used in two ways – in *transport mode* or in *tunnel mode*. Transport mode inserts ESP header behind the IP header of the original IP datagram. ESP trailer and optional authentication data follow data of the original datagram. Transport mode requires less overhead when processing than the tunnel mode, but does not provide such security of data protection.

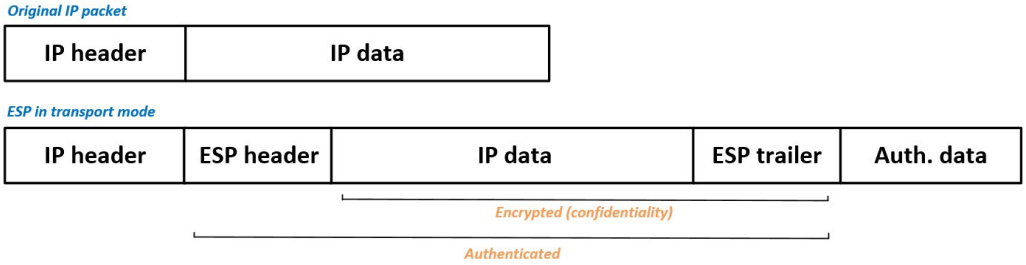



Figure 1: ESP – transport mode

Tunnel mode (sometimes tunneling mode) creates a new IP header which is followed by header of Encapsulating Security Payload protocol. This is followed by the entire original datagram packaged as new data datagram. This allows to completely protect original datagram (in case that encryption and authentication are used). ESP trailer and optional authentication data follow data of the original datagram.

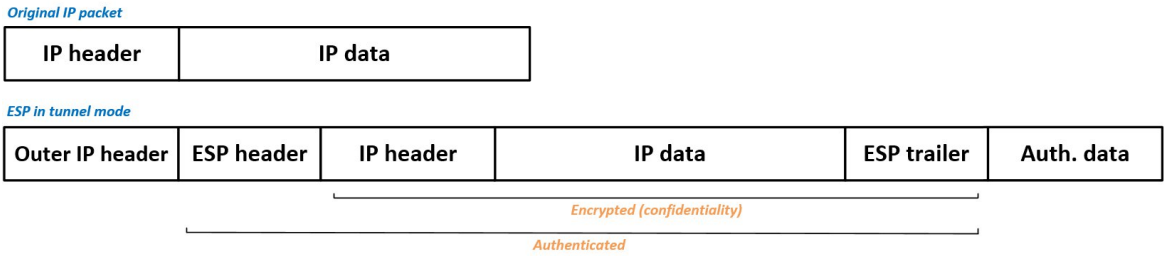



Figure 2: ESP – tunnel mode

# 2. Configuration of IPsec Tunnel

The IPsec tunnel function allows you to create a secured connection between two separate LAN networks. Advantech routers allows you to create **up to four IPsec tunnels**.

To open the IPsec tunnel configuration page, click *IPsec* in the *Configuration* section of the main menu. The menu item will expand and you will see four separate configuration pages: *1st Tunnel*, *2nd Tunnel*, *3rd Tunnel* and *4th Tunnel*. Supported are both, **policy-based** and **route-based** VPN approaches, see the different configuration scenarios in Chapter 2.1.

IPv4 and IPv6 tunnels are supported (**dual stack**), you can transport IPv6 traffic through IPv4 tunnel and vice versa. For different IPsec authentication scenarios, see Chapter 2.2.

The dual stack IPsec tunnels are not supported by routers of v2 product line.

FRRouting (FRR) router app is an Internet routing protocol suite for Advantech routers. This UM includes protocol daemons for BGP, IS-IS, LDP, OSPF, PIM, and RIP.

To encrypt data between the local and remote subnets, specify the appropriate values in the subnet fields on both routers. To encrypt the data stream between the routers only, leave the local and remote subnets fields blank.

If you specify the protocol and port information in the *Local Protocol/Port* field, then the router encapsulates only the packets matching the settings.

For optimal an secure setup, we recommend to follow instructions on the Security Recommendations *strongSwan* web page.

## 2.1 Route-based Configuration Scenarios

There are more different route-based configuration options which can be configured and used in Advantech routers. Below are listed the most common cases which can be used (for more details see Route-based VPNs *strongSwan* web page):

1. **Enabled Installing Routes**
   - Remote (local) subnets are used as traffic selectors (routes).
   - It results to the same outcome as a policy-based VPN.
   - One benefit of this approach is the possibility to verify non-encrypted traffic passed through an IPsec tunnel number X by tcdump tool: `tcpdump -i ipsecX`.
   - Set up the *Install Routes* to *yes* option.
2. **Static Routes**
   - Routes are installed statically by an application as soon as the IPsec tunnel is up.
   - As an application for static routes installation can be used for example FRR/STATICD application.
   - Set up the *Install Routes* to *no* option.

3. **Dynamic Routing**
   • Routes are installed dynamically while running by an application using a dynamic protocol.
   • As an application for dynamic routes installation can be used for example FRR/BGP or FRR/OSPF application. This application gains the routes dynamically from an (BGP, OSPF) server.
   • Set up the *Install Routes* to *no* option.

4. **Multiple Clients**
   • Allows to create VPN network with multiple clients. One Advantech router acts as the server and assigns IP address to all the clients on the network.
   • The server has *Remote Virtual Network* and *Remote Virtual Mask* items configured and the client has *Local Virtual Address* item configured.
   • Set up the *Install Routes* to *yes* option.

## 2.2 IPsec Authentication Scenarios

There are four basic authentication options which can be configured and used in Advantech routers:

1. **Pre-shared Key**
   • Set *Authenticate Mode* to *pre-shared key* option.
   • Enter the shared key to the *Pre-shared key* field.

2. **Public Key**
   • Set *Authenticate Mode* to *X.509 certificate* option.
   • Enter the public key to the *Local Certificate / PubKey* field.
   • CA certificate is not required.

3. **Peer Certificate**
   • Set *Authenticate Mode* to *X.509 certificate* option.
   • Enter the remote key to the *Remote Certificate / PubKey* field. Users with this certificate will be allowed.
   • CA certificate is not required.

4. **CA Certificate**
   • Set *Authenticate Mode* to *X.509 certificate* option.
   • Enter the CA certificate or a list of CA certificates to the *CA Certificate* field. Any certificate signed by the CA will be accepted.
   • Remote certificate is not required.

**Notes:**

- The Peer and CA Certificate (options 3 and 4) can be configured and used simultaneously – authentication can be done by one of this method.
- The Local ID is significant. When using certificate authentication, the IKE identity must be contained in the certificate, either as subject or as `subjectAltName`.

## 2.3 Configuration Items Description

The configuration GUI for IPsec is shown in Figure 3 and the description of all items, which can be configured for an IPsec tunnel, are described in Table 1.

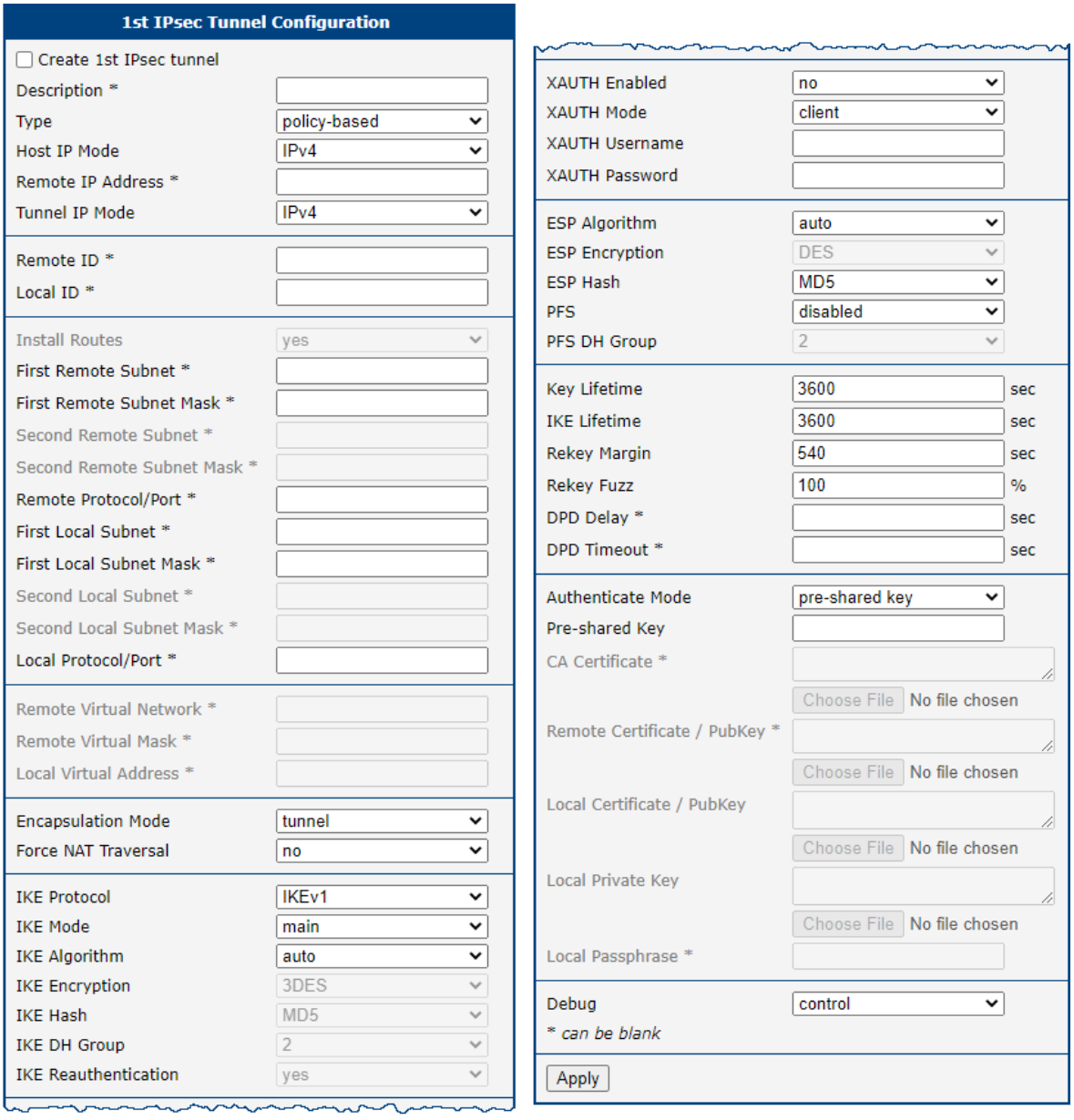



Figure 3: IPsec Tunnels Configuration

| Item | Description |
|---|---|
| Description | Name or description of the tunnel. |
| Type | • **policy-based** – Choose for the policy-based VPN approach.<br>• **route-based** – Choose for the route-based VPN approach.<br>Note: Data throughput via route-based VPN is slightly lower in comparison with policy-based VPN. |
| Host IP Mode | • **IPv4** – The router communicates via IPv4 with the opposite side of the tunnel.<br>• **IPv6** – The router communicates via IPv6 with the opposite side of the tunnel. |
| Remote IP Address | IPv4, IPv6 address or domain name of the remote side of the tunnel, based on selected *Host IP Mode* above. |
| Tunnel IP Mode | • **IPv4** – The IPv4 communication runs inside the tunnel.<br>• **IPv6** – The IPv6 communication runs inside the tunnel. |
| Remote ID | Identifier (ID) of remote side of the tunnel. It consists of two parts: a *hostname* and a *domain-name*. |
| Local ID | Identifier (ID) of local side of the tunnel. It consists of two parts: a *hostname* and a *domain-name*. |
| Install Routers | For route-based type only. Choose **yes** to use traffic selectors as route(s). |
| First Remote Subnet | IPv4 or IPv6 address of a network behind remote side of the tunnel, based on *Tunnel IP Mode* above. |
| First Remote Subnet Mask/Prefix | IPv4 subnet mask of a network behind remote side of the tunnel, or IPv6 prefix (single number 0 to 128). |
| Second Remote Subnet | IPv4 or IPv6 address of the second network behind remote side of the tunnel, based on *Tunnel IP Mode* above. For *IKE Protocol* = IKEv2 only. |
| Second Remote Subnet Mask/Prefix | IPv4 subnet mask of the second network behind remote side of the tunnel, or IPv6 prefix (single number 0 to 128). For *IKE Protocol* = IKEv2 only. |
| Remote Protocol/Port | Specifies Protocol/Port of remote side of the tunnel. The general form is *protocol*/*port*, for example 17/1701 for UDP (protocol 17) and port 1701. It is also possible to enter only the number of protocol, however, the above mentioned format is preferred. |
| First Local Subnet | IPv4 or IPv6 address of a local network, based on *Tunnel IP Mode* above. |
| First Local Subnet Mask/Prefix | IPv4 subnet mask of a local network, or IPv6 prefix (single number 0 to 128). |

Continued on next page

Continued from previous page

| Item | Description |
|---|---|
| Second Local Subnet | IPv4 or IPv6 address of the second local network, based on *Tunnel IP Mode* above. For *IKE Protocol* = IKEv2 only. |
| Second Local Subnet Mask/Prefix | IPv4 subnet mask of the second local network, or IPv6 prefix (single number 0 to 128). For *IKE Protocol* = IKEv2 only. |
| Local Protocol/Port | Specifies Protocol/Port of a local network. The general form is *protocol*/*port*, for example 17/1701 for UDP (protocol 17) and port 1701. It is also possible to enter only the number of protocol, however, the above mentioned format is preferred. |
| Remote Virtual Network | Specifies virtual remote network for server (responder). |
| Remote Virtual Mask | Specifies virtual remote network mask for server (responder). |
| Local Virtual Address | Specifies virtual local network address for client. To get address from server set up the address to 0.0.0.0. |
| Encapsulation Mode | Specifies the IPsec mode, according to the method of encapsulation.<br>• **tunnel** – entire IP datagram is encapsulated.<br>• **transport** – only IP header is encapsulated. Not supported by route-based VPN.<br>• **beet** – the ESP packet is formatted as a transport mode packet, but the semantics of the connection are the same as for tunnel mode. |
| Force NAT Traversal | Enable NAT traversal enforcement (UDP encapsulation of ESP packets). |
| IKE Protocol | Specifies the version of IKE (**IKEv1**/**IKEv2**, **IKEv1** or **IKEv2**). |
| IKE Mode | Specifies the mode for establishing a connection (*main* or *aggressive*). If you select the aggressive mode, then the router establishes the IPsec tunnel faster, but the encryption is permanently set to 3DES-MD5. **We recommend that you not use the *aggressive* mode due to lower security!** |
| IKE Algorithm | Specifies the means by which the router selects the algorithm:<br>• **auto** – The encryption and hash algorithm are selected automatically.<br>• **manual** – The encryption and hash algorithm are defined by the user. |
| IKE Encryption | Encryption algorithm – **3DES**, **AES128**, **AES192**, **AES256**, **AES128GCM128**, **AES192GCM128**, **AES256GCM128**. |
| IKE Hash | Hash algorithm – **MD5**, **SHA1**, **SHA256**, **SHA384** or **SHA512**. |

Continued on next page

Continued from previous page

| Item | Description |
|---|---|
| IKE DH Group | Specifies the Diffie-Hellman groups which determine the strength of the key used in the key exchange process. Higher group numbers are more secure, but require more time to compute the key. |
| IKE Reauthentication | Enable or disable IKE reauthentication (for IKEv2 only). |
| XAUTH Enabled | Enable extended authentication (for IKEv1 only). |
| XAUTH Mode | Select XAUTH mode (client or server). |
| XAUTH Username | XAUTH username. |
| XAUTH Password | XAUTH password. |
| ESP Algorithm | Specifies the means by which the router selects the algorithm:<br>• **auto** – The encryption and hash algorithm are selected automatically.<br>• **manual** – The encryption and hash algorithm are defined by the user. |
| ESP Encryption | Encryption algorithm – **3DES**, **AES128**, **AES192**, **AES256**, **AES128GCM128**, **AES192GCM128**, **AES256GCM128**. |
| ESP Hash | Hash algorithm – **MD5**, **SHA1**, **SHA256**, **SHA384** or **SHA512**. |
| PFS | Enables/disables the *Perfect Forward Secrecy* function. The function ensures that derived session keys are not compromised if one of the private keys is compromised in the future. |
| PFS DH Group | Specifies the Diffie-Hellman group number (see *IKE DH Group*). |
| Key Lifetime | Lifetime key data part of tunnel. The minimum value of this parameter is 60 s. The maximum value is 86400 s. |
| IKE Lifetime | Lifetime key service part of tunnel. The minimum value of this parameter is 60 s. The maximum value is 86400 s. |
| Rekey Margin | Specifies how long before a connection expires that the router attempts to negotiate a replacement. Specify a maximum value that is less than half of IKE and Key Lifetime parameters. |
| Rekey Fuzz | Percentage of time for the Rekey Margin extension. |
| DPD Delay | Time after which the IPsec tunnel functionality is tested. |
| DPD Timeout | The period during which device waits for a response. |
| Authenticate Mode | Specifies the means by which the router authenticates:<br>• **Pre-shared key** – Sets the shared key for both sides of the tunnel.<br>• **X.509 Certificate** – Allows X.509 authentication in multiclient mode. |

Continued on next page

Continued from previous page

| Item | Description |
|---|---|
| Pre-shared Key | Specifies the shared key for both sides of the tunnel. The prerequisite for entering a key is that you select pre-shared key as the authentication mode. |
| CA Certificate | Certificate for X.509 authentication. |
| Remote Certificate \ PubKey | Certificate for X.509 authentication or PubKey for public key signature authentication. |
| Local Certificate \ PubKey | Certificate for X.509 authentication or PubKey for public key signature authentication. |
| Local Private Key | Private key for X.509 authentication. |
| Local Passphrase | Passphrase used during private key generation. |
| Debug | Choose the level of logging verbosity from: **silent**, **audit**, **control** (default), **control-more**, **raw**, **private** (most verbose including the private keys). See Logger Configuration in *strongSwan* web page for more details. |

Table 1: IPsec Tunnel Configuration

We recommend that you keep up the default settings. When you set key exchange times higher, the tunnel produces lower operating costs, but the setting also provides less security. Conversely, when you reducing the time, the tunnel produces higher operating costs, but provides for higher security. The changes in settings will apply after clicking the *Apply* button.

**Do not miss:**

- If local and remote subnets are not configured then only packets between local and remote IP address are encapsulated, so only communication between two routers is encrypted.
- If protocol/port fields are configured then only packets matching these settings are encapsulated.

## 2.4 Certificate Generation

The following procedure describes how to generate certificates and keys without a password phrase:

```
******************** certification authority ************************
openssl rand -out private/.rand 1024
openssl genrsa -des3 -out private/ca.key 2048
openssl req -new -key private/ca.key -out tmp/myrootca.req
openssl x509 -req -days 7305 -sha1 -extensions v3_ca -signkey
private/ca.key -in tmp/myrootca.req -out ca.crt

******************** server cert ************************************
openssl genrsa -out private/server.key 2048
openssl req -new -key private/server.key -out tmp/server.req
openssl x509 -req -days 7305 -sha1 -extensions v3_req -CA ca.crt -CAkey
private/ca.key -in tmp/server.req -CAserial ca.srl -CAcreateserial
-out server.crt

******************** client cert ************************************
openssl genrsa -out private/client.key 2048
openssl req -new -key private/client.key -out tmp/client.req
openssl x509 -req -days 7305 -sha1 -extensions v3_req -CA ca.crt -CAkey
private/ca.key -in tmp/client.req -CAserial ca.srl -CAcreateserial
-out client.crt
```

Listed below are the certificates with password phrase "router" (certification authority remains unchanged):

```
******************** server cert ************************************
openssl genrsa -des3 -passout pass:router -out private/server.pem 2048
openssl req -new -key private/server.pem -out tmp/server.req
openssl x509 -req -days 7305 -sha1 -extensions v3_req -CA ca.crt -CAkey
private/ca.key -in tmp/server.req -CAserial ca.srl -CAcreateserial
-out server.crt

******************** client cert ************************************
openssl genrsa -des3 -passout pass:router -out private/client.pem 2048
openssl req -new -key private/client.pem -out tmp/client.req
openssl x509 -req -days 7305 -sha1 -extensions v3_req -CA ca.crt -CAkey
private/ca.key -in tmp/client.req -CAserial ca.srl -CAcreateserial
-out client.crt
```

The IPsec function supports the following types of identifiers (ID) for both sides of the

tunnel, *Remote ID* and *Local ID* parameters:

- IP address (for example, 192.168.1.1)
- DN (for example, C=CZ,O=CompanyName,OU=TP,CN=A)
- FQDN (for example, @director.companyname.cz) – **the @ symbol proceeds the FQDN.**
- User FQDN (for example, director@companyname.cz)

The certificates and private keys have to be in the PEM format. Use only certificates containing start and stop tags.

The random time, after which the router re-exchanges new keys is defined as follows:

*Lifetime - (Rekey margin + random value in range (from 0 to Rekey margin * Rekey Fuzz/100))*

The default exchange of keys is in the following time range:

- Minimal time: 1h - (9m + 9m) = 42m
- Maximal time: 1h - (9m + 0m) = 51m

## 2.5 IPsec Status – Tunnel Established

Selecting the *IPsec* option in the *Status* menu of the web page will bring up the information for any IPsec Tunnels that have been established. If the tunnel has been built correctly, the screen will display **ESTABLISHED** and the number of running IPsec connections **1 up** (orange highlighted in the figure below.) If there is no such text in log (e.g. "0 up"), the tunnel was not created!

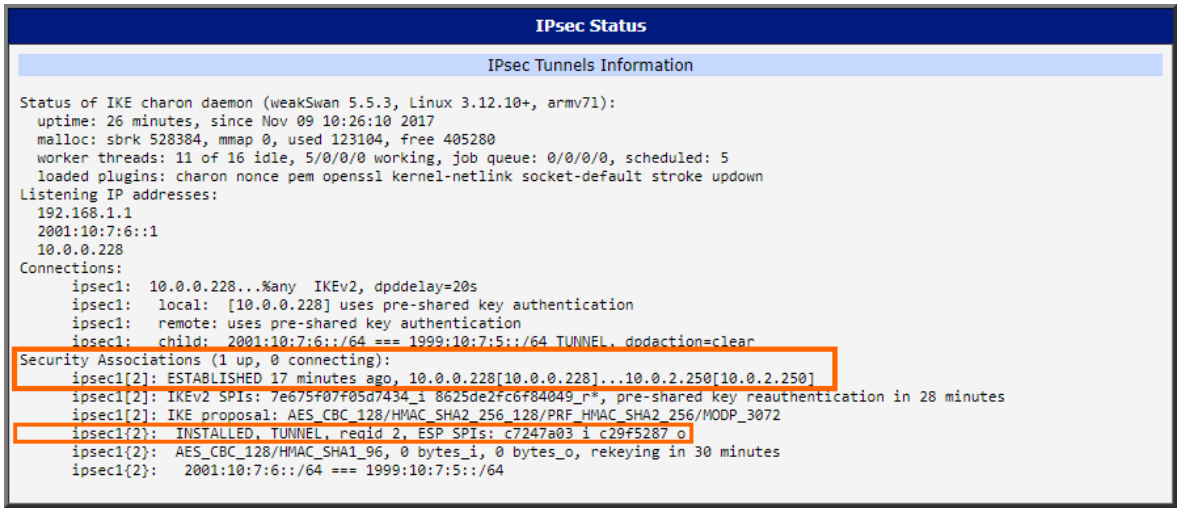



Figure 4: IPsec Status

# 3. Examples of Use

## 3.1 IPv6 IPsec Tunnel over IPv4 Internet

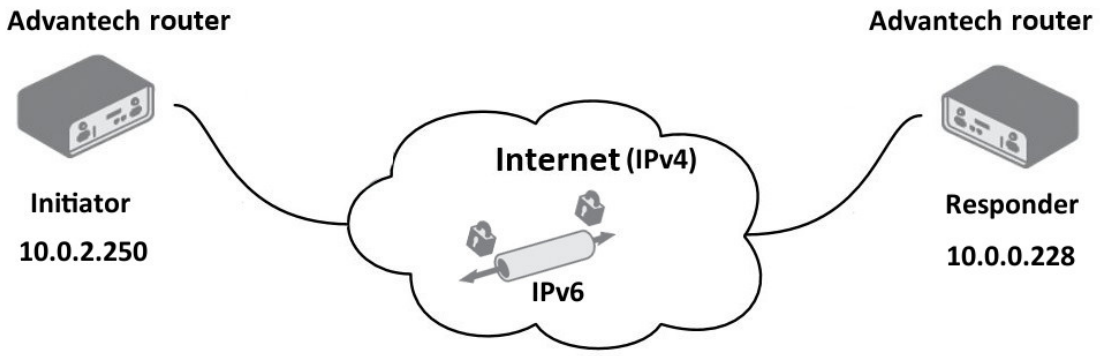



Figure 5: IPv6 IPsec tunnel over IPv4 Internet – two Advantech routers

This is an example of IPsec tunnel establishment for IPv6 network. Two Advantech v3 Routers are used (IPv6 is supported on v3 routers). One Advantech router as IPsec initiator and one Advantech router as IPsec responder. The routers are connected to the Internet via IPv4 but the communication inside established IPsec tunnel is IPv6, so the IPv6 networks on both sides can be connected to each other. See the configuration forms and IPsec Status pages on the following figures.

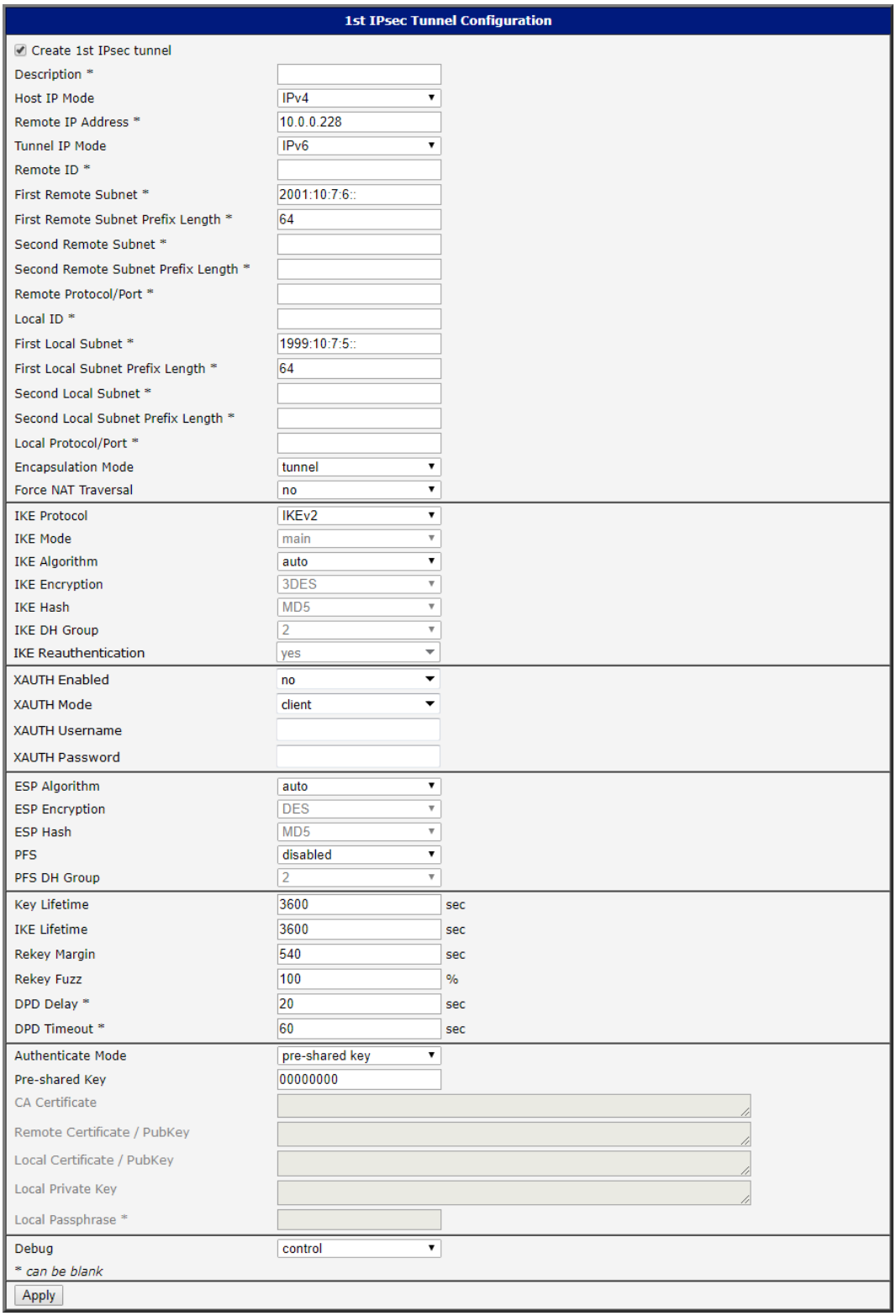



Figure 6: Initiator configuration of the IPv6 over IPv4 IPsec tunnel

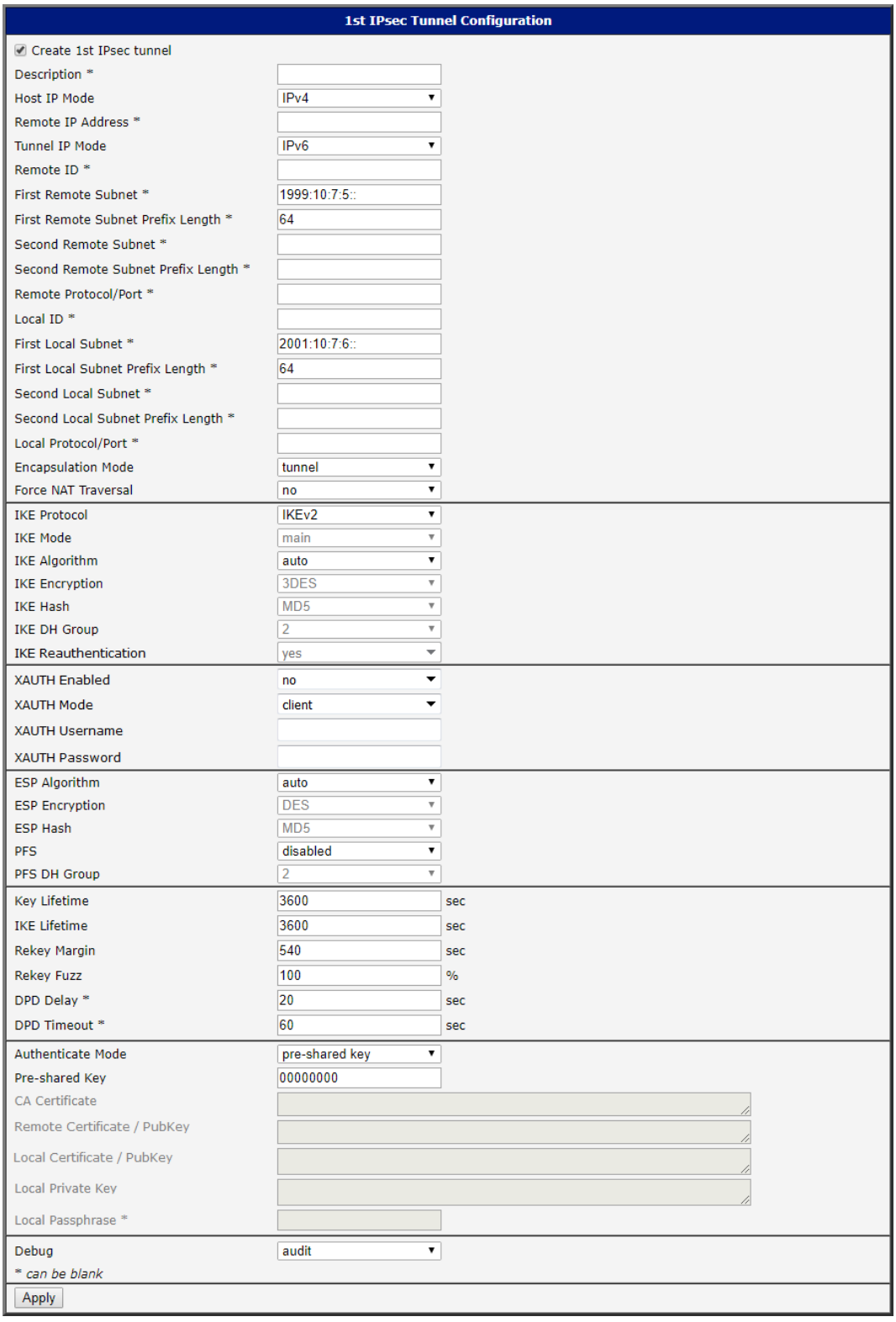
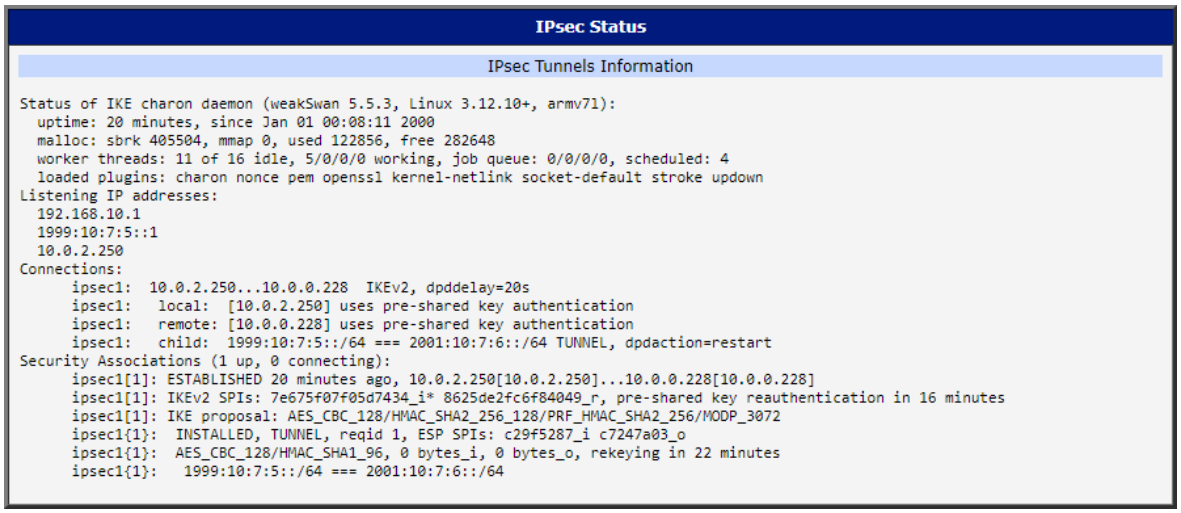



Figure 7: Responder configuration of the IPv6 over IPv4 IPsec tunnel

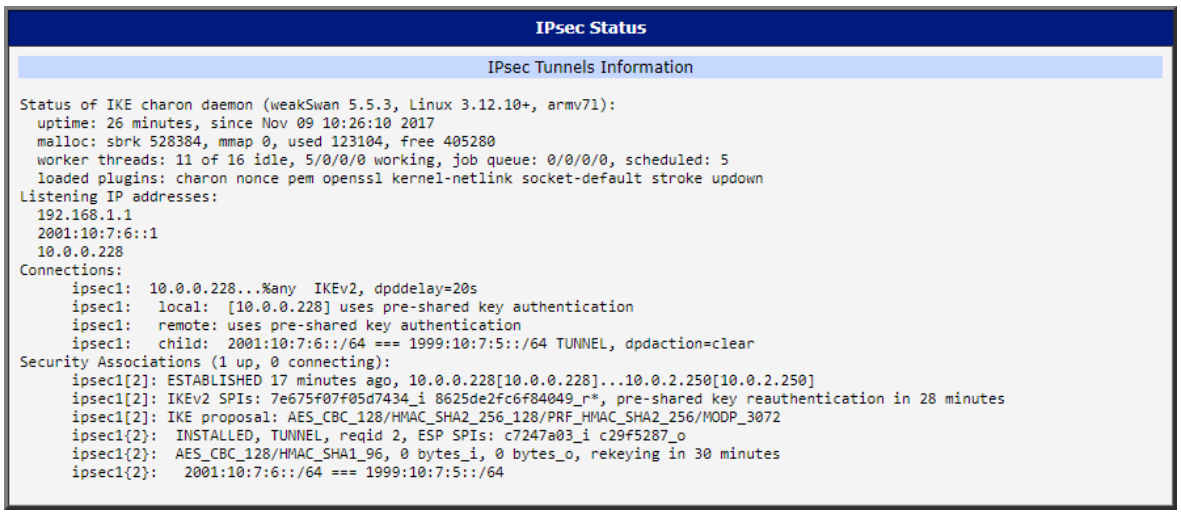


**IPsec Status**

IPsec Tunnels Information

```
Status of IKE charon daemon (weakSwan 5.5.3, Linux 3.12.10+, armv7l):
  uptime: 20 minutes, since Jan 01 00:08:11 2000
  malloc: sbrk 405504, mmap 0, used 122856, free 282648
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 4
  loaded plugins: charon nonce pem openssl kernel-netlink socket-default stroke updown
Listening IP addresses:
  192.168.10.1
  1999:10:7:5::1
  10.0.2.250
Connections:
      ipsec1:  10.0.2.250...10.0.0.228  IKEv2, dpddelay=20s
      ipsec1:   local:  [10.0.2.250] uses pre-shared key authentication
      ipsec1:   remote: [10.0.0.228] uses pre-shared key authentication
      ipsec1:   child:  1999:10:7:5::/64 === 2001:10:7:6::/64 TUNNEL, dpdaction=restart
Security Associations (1 up, 0 connecting):
      ipsec1[1]: ESTABLISHED 20 minutes ago, 10.0.2.250[10.0.2.250]...10.0.0.228[10.0.0.228]
      ipsec1[1]: IKEv2 SPIs: 7e675f07f05d7434_i* 8625de2fc6f84049_r, pre-shared key reauthentication in 16 minutes
      ipsec1[1]: IKE proposal: AES_CBC_128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
      ipsec1{1}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: c29f5287_i c7247a03_o
      ipsec1{1}:  AES_CBC_128/HMAC_SHA1_96, 0 bytes_i, 0 bytes_o, rekeying in 22 minutes
      ipsec1{1}:   1999:10:7:5::/64 === 2001:10:7:6::/64
```

Figure 8: IPsec Status of the Initiator



**IPsec Status**

IPsec Tunnels Information

```
Status of IKE charon daemon (weakSwan 5.5.3, Linux 3.12.10+, armv7l):
  uptime: 26 minutes, since Nov 09 10:26:10 2017
  malloc: sbrk 528384, mmap 0, used 123104, free 405280
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 5
  loaded plugins: charon nonce pem openssl kernel-netlink socket-default stroke updown
Listening IP addresses:
  192.168.1.1
  2001:10:7:6::1
  10.0.0.228
Connections:
      ipsec1:  10.0.0.228...%any  IKEv2, dpddelay=20s
      ipsec1:   local:  [10.0.0.228] uses pre-shared key authentication
      ipsec1:   remote: uses pre-shared key authentication
      ipsec1:   child:  2001:10:7:6::/64 === 1999:10:7:5::/64 TUNNEL, dpdaction=clear
Security Associations (1 up, 0 connecting):
      ipsec1[2]: ESTABLISHED 17 minutes ago, 10.0.0.228[10.0.0.228]...10.0.2.250[10.0.2.250]
      ipsec1[2]: IKEv2 SPIs: 7e675f07f05d7434_i 8625de2fc6f84049_r*, pre-shared key reauthentication in 28 minutes
      ipsec1[2]: IKE proposal: AES_CBC_128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
      ipsec1{2}:  INSTALLED, TUNNEL, reqid 2, ESP SPIs: c7247a03_i c29f5287_o
      ipsec1{2}:  AES_CBC_128/HMAC_SHA1_96, 0 bytes_i, 0 bytes_o, rekeying in 30 minutes
      ipsec1{2}:   2001:10:7:6::/64 === 1999:10:7:5::/64
```

Figure 9: IPsec Status of the Responder

## 3.2 Advantech Router and Cisco Basic IPsec Tunnel Configurations

There is a bug in Cisco ASA 5500-X Series Firewalls. IKEv2 between ASA and strongswan (IKEv2 aes256/sha256) does not work. More info at https://quickview.cloudapps.cisco.com/quickview/bug/CSCvb21927.

### 3.2.1 IKEv1 Pre-shared key Tunnel

IP address of the SIM card inserted into Advantech router can be either static or dynamic, because IPsec tunnel is established by initiator on the router. In this case, Linux server (Cisco router) offers services for IPsec tunnel therefore it has to be always available on a static IP address or on a domain name.

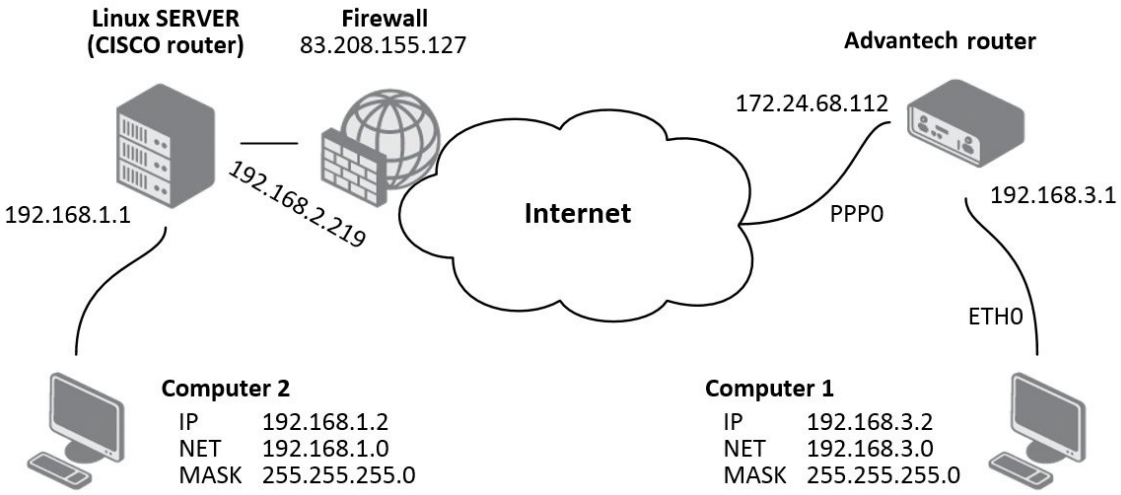



Figure 10: IPsec tunnel – initiator on the router

If addresses of tunnel ends are visible to one another, all you have to do is specify these items: *Description*, *Remote IP address*, *First Remote Subnet*, *First Remote Subnet Mask*, *First Local Subnet* and *First Local Subnet Mask*. If not (one end of the tunnel is in a private network), it is necessary to set *Force NAT Traversal* to *yes*.

If *NAT Traversal* is active, it is also necessary to set *Remote ID*. The FQDN (Fully Qualified Domain Name) has to be provided as the Remote ID, which is the designation for a fully specified domain name of the computer. It is also possible to set authentication using certificates, but then there is no need to enter *Remote ID*.

The following table provides an example of IPsec tunnel settings which correspond to the Figure from the beginning of this chapter:

| Item | Value |
|---|---|
| Remote IP Address | 83.208.155.127 |
| Remote ID | ciscoasa@default.domain |
| First Remote Subnet | 192.168.1.0 |
| First Remote Subnet Mask | 255.255.255.0 |
| First Local Subnet | 192.168.3.0 |
| First Local Subnet Mask | 255.255.255.0 |
| Force NAT Traversal | yes |
| Pre-shared Key | test |

Table 2: IPsec tunnel settings (initiator)

Other parameters can be left in default settings. If the *Remote IP Address* parameter is empty on one side of the IPsec tunnel, then this side will wait for a connection and will not attempt to establish a connection.

All items that are not mentioned in the sample settings and are marked with an asterisk (*) may not be filled in. They are used to accurate identification of the tunnel.

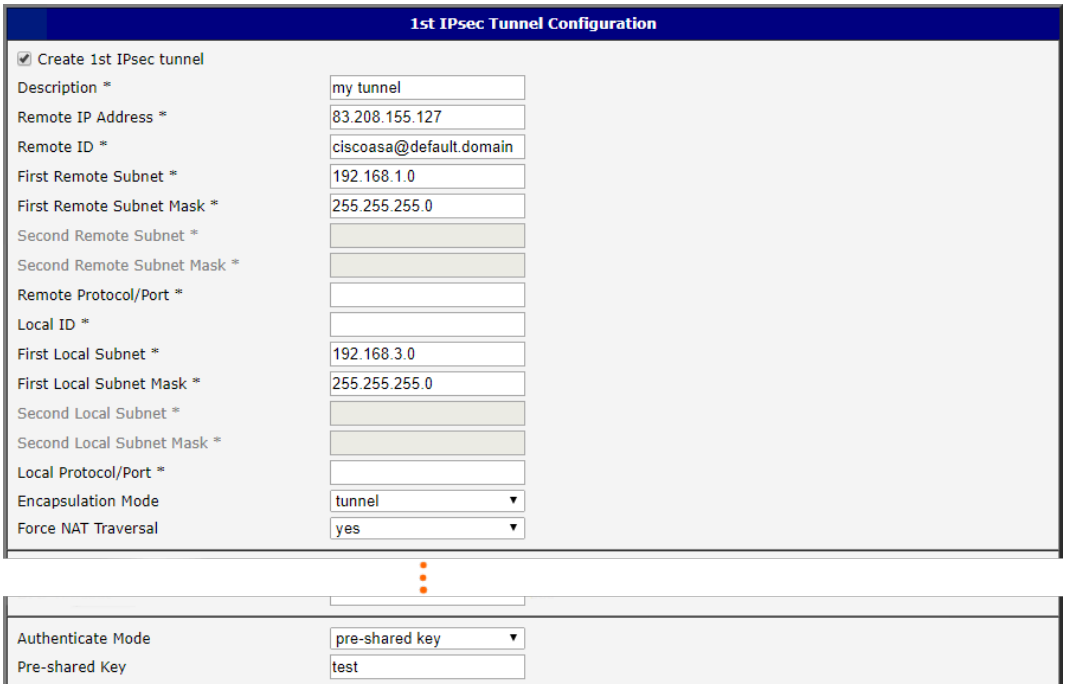



Figure 11: IPsec tunnel – example configuration of initiator on the router

Information about the active IPsec tunnel can be found in the *Status* section on the *IPsec* page of the router web interface.

### Advantech Router as IPsec Responder

Advantech router must have an available static IP address or dynamic IP address of the SIM card in case of using translation of dynamically assigned IP addresses to DynDNS domain name. In this case, Linux server (Cisco router) is initiator and establishes IPsec tunnel.

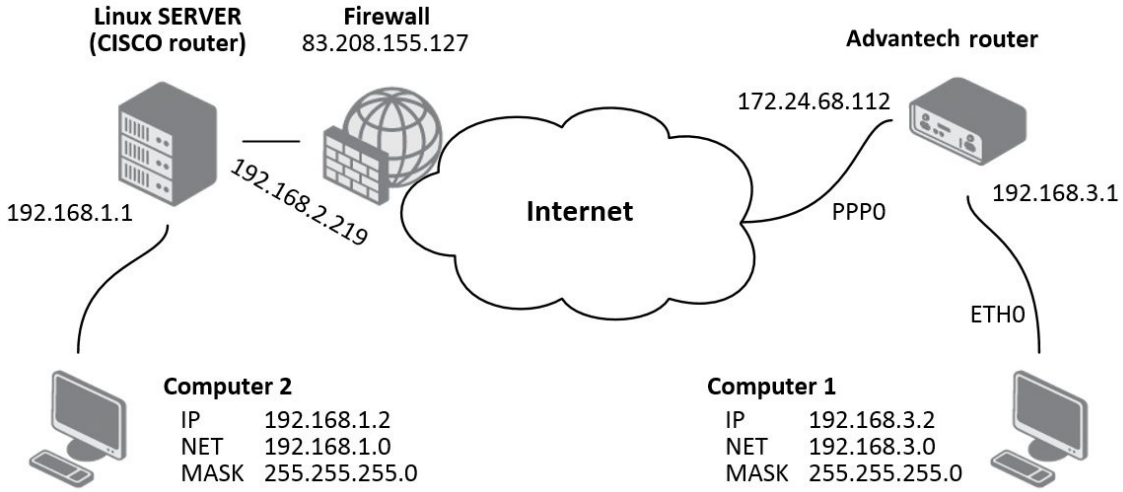



Figure 12: IPsec tunnel – responder on the router

If addresses of tunnel ends are visible to one another, all you have to do is specify these items: *Description*, *First Remote Subnet* and *First Remote Subnet Mask*. If not (one end of the tunnel is in a private network), it is necessary to set *Force NAT Traversal* to *yes*.

If *Force NAT Traversal* is active, it is also necessary to set *Remote ID*. As the ID has to be filled FQDN (Fully Qualified Domain Name), which is the designation for a fully specified domain name of the computer. It is also possible to set authentication using certificates, but then there is no need to enter *Remote ID*.

The following table provides an example of IPsec tunnel settings which correspond to the figure from the beginning of this page:

| Item | Value |
|---|---|
| Remote ID | ciscoasa@default.domain |
| First Remote Subnet | 192.168.2.219 |
| First Remote Subnet Mask | 255.255.255.255 |
| Force NAT Traversal | yes |
| Pre-shared Key | test |

Table 3: IPsec tunnel settings (responder)

Other parameters can be left in default settings. If the *Remote IP Address* parameter is empty on one side of IPsec tunnel, then this side will wait for a connection and will not attempt to establish a connection.

All items that are not mentioned in the sample settings and are marked with an asterisk (*) may not be filled in. They are used to accurate identification of the tunnel.

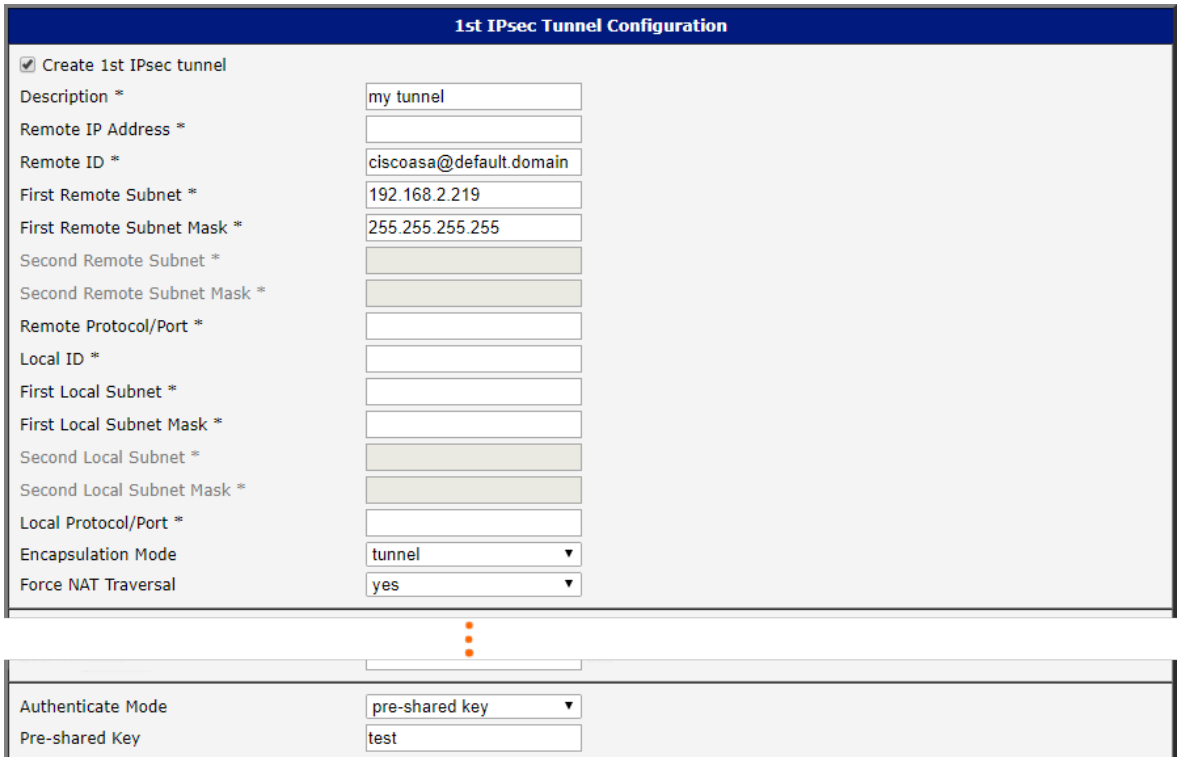



Figure 13: IPsec tunnel – example configuration of responder on the router

Information about the active IPsec tunnel can be found in the *Status* section on the *IPsec* page of the router web interface.

### Linux Server IPsec Configuration

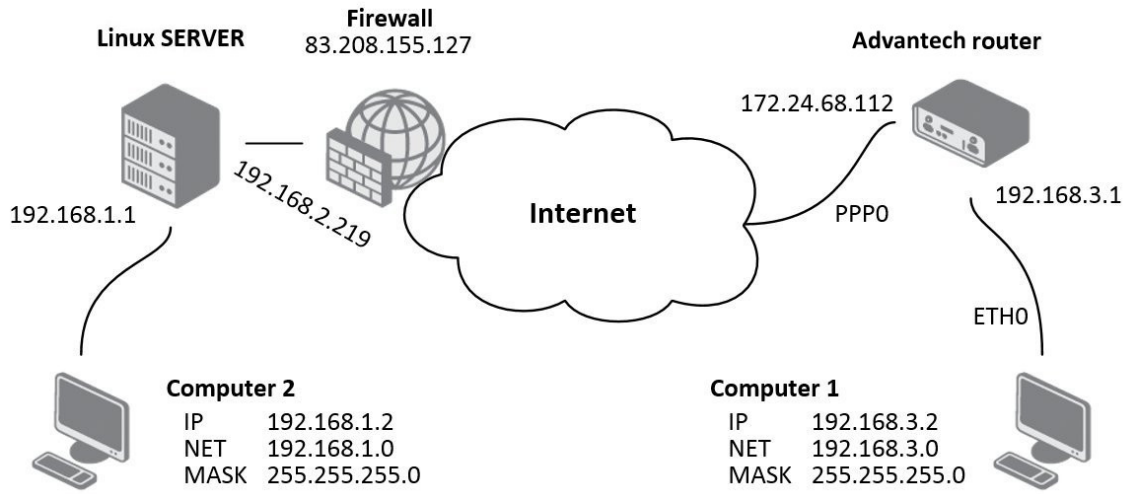



Figure 14: IPsec tunnel – Linux server

On the Linux server is needed to configure *ipsec.conf* and *ipsec.secrets* files. Configuration of *ipsec.conf* file can be performed for example like this:

```
conn advantechrouter
   authby=secret
   type=tunnel
   left=83.208.155.127
   leftsubnet=192.168.1.0/24
   right=172.24.68.112
   rightsubnet=192.168.3.0/24
   ikelifetime=3600s
   keylife=3600s
   pfs=no
   auto=add
```

*ipsec.secrets* file shall be configured as follows:

```
83.208.155.127 172.24.68.112: PSK "test"
```

### Cisco Router as Initiator – IPsec Configuration

Please note that Cisco routers support IPsec protocol since IOS version no. 7.1.

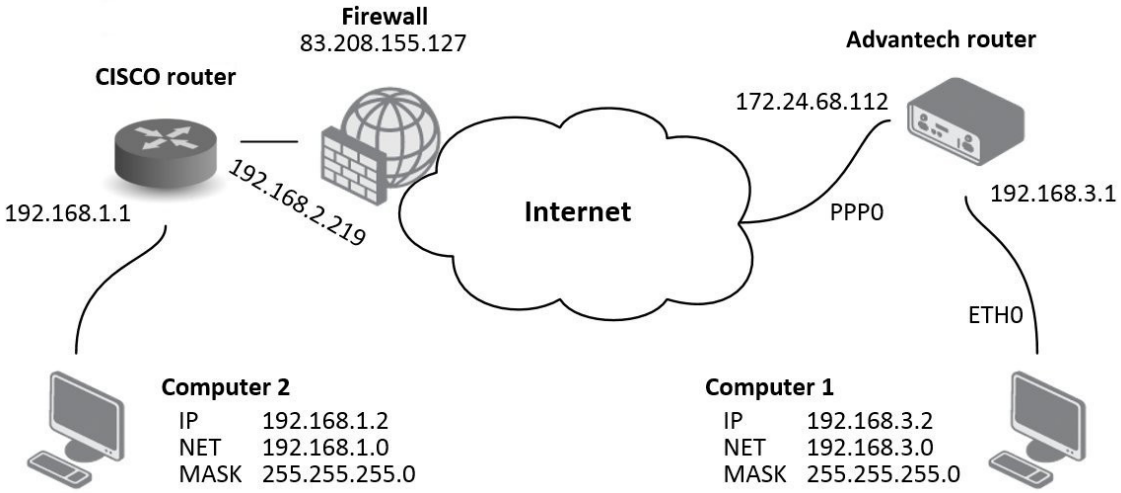



Figure 15: IPsec tunnel – Cisco router as initiator

```
access-list outside_2_cryptomap extended permit ip 192.168.1.0
255.255.255.0 192.168.3.0 255.255.255.0
crypto ipsec transform-set ESP-DES-MD5 esp-des esp-md5-hmac
crypto ipsec transform-set ESP-DES-SHA esp-des esp-sha-hmac
crypto ipsec transform-set ESP-3DES-SHA esp-3des esp-sha-hmac
crypto ipsec transform-set ESP-AES-128-SHA esp-aes esp-sha-hmac
crypto ipsec transform-set ESP-AES-256-MD5 esp-aes-256 esp-md5-hmac
crypto ipsec transform-set ESP-AES-256-SHA esp-aes-256 esp-sha-hmac
crypto ipsec transform-set ESP-AES-128-MD5 esp-aes esp-md5-hmac
crypto ipsec transform-set ESP-AES-192-MD5 esp-aes-192 esp-md5-hmac
crypto ipsec transform-set ESP-AES-192-SHA esp-aes-192 esp-sha-hmac
crypto ipsec transform-set UR1 esp-3des esp-none
crypto ipsec transform-set UR2 esp-des esp-none
crypto ipsec transform-set ESP-3DES-MD5 esp-3des esp-md5-hmac
crypto map outside_map 1 match address outside_2_cryptomap
crypto map outside_map 1 set connection-type answer-only
crypto map outside_map 1 set peer 172.24.68.112
crypto map outside_map 1 set transform-set ESP-3DES-MD5
crypto map outside_map interface outside
```

```
crypto isakmp identity hostname
crypto isakmp enable outside
crypto isakmp policy 10
authentication pre-share
encryption 3des
hash md5
group 2
lifetime 3600
crypto isakmp nat-traversal 20

group-policy DfltGrpPolicy attributes
banner none
wins-server none
dns-server none
dhcp-network-scope none
vpn-access-hours none
vpn-simultaneous-logins 3
vpn-idle-timeout none
vpn-session-timeout none
vpn-filter none
vpn-tunnel-protocol IPSec l2tp-ipsec webvpn
password-storage disable
ip-comp disable
re-xauth disable
group-lock none
pfs disable
ipsec-udp enable
ipsec-udp-port 10000
split-tunnel-policy tunnelall
split-tunnel-network-list none
default-domain none

tunnel-group DefaultL2LGroup ipsec-attributes
pre-shared-key *
isakmp keepalive threshold 20 retry 10
tunnel-group 172.24.68.112 type ipsec-l2l
tunnel-group 172.24.68.112 ipsec-attributes
pre-shared-key *
tunnel-group-map enable rules
tunnel-group-map default-group DefaultL2LGroup
prompt hostname context
```

### Cisco Router as Responder – IPsec Configuration

Please note that Cisco routers support IPsec protocol since IOS version no. 7.1.

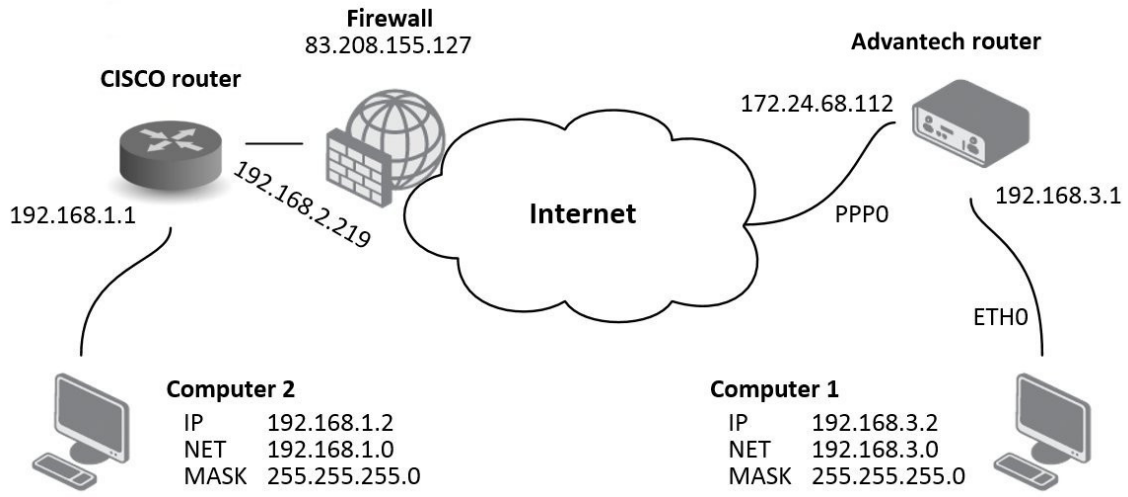



Figure 16: IPsec tunnel – Cisco router as responder

```
access-list outside_2_cryptomap extended permit ip 192.168.1.0
      255.255.255.0 192.168.3.0 255.255.255.0
crypto ipsec transform-set ESP-DES-MD5 esp-des esp-md5-hmac
crypto ipsec transform-set ESP-DES-SHA esp-des esp-sha-hmac
crypto ipsec transform-set ESP-3DES-SHA esp-3des esp-sha-hmac
crypto ipsec transform-set ESP-AES-128-SHA esp-aes esp-sha-hmac
crypto ipsec transform-set ESP-AES-256-MD5 esp-aes-256 esp-md5-hmac
crypto ipsec transform-set ESP-AES-256-SHA esp-aes-256 esp-sha-hmac
crypto ipsec transform-set ESP-AES-128-MD5 esp-aes esp-md5-hmac
crypto ipsec transform-set ESP-AES-192-MD5 esp-aes-192 esp-md5-hmac
crypto ipsec transform-set ESP-AES-192-SHA esp-aes-192 esp-sha-hmac
crypto ipsec transform-set UR1 esp-3des esp-none
crypto ipsec transform-set UR2 esp-des esp-none
crypto ipsec transform-set ESP-3DES-MD5 esp-3des esp-md5-hmac
crypto map outside_map 1 match address outside_2_cryptomap
crypto map outside_map 1 set connection-type originate-only
crypto map outside_map 1 set peer 172.24.68.112
crypto map outside_map 1 set transform-set ESP-3DES-MD5
crypto map outside_map interface outside
```

```
crypto isakmp identity hostname
crypto isakmp enable outside
crypto isakmp policy 10
  authentication pre-share
  encryption 3des
  hash md5
  group 2
  lifetime 3600
crypto isakmp nat-traversal 20

group-policy DfltGrpPolicy attributes
  banner none
  wins-server none
  dns-server none
  dhcp-network-scope none
  vpn-access-hours none
  vpn-simultaneous-logins 3
  vpn-idle-timeout none
  vpn-session-timeout none
  vpn-filter none
  vpn-tunnel-protocol IPSec l2tp-ipsec webvpn
  password-storage disable
  ip-comp disable
  re-xauth disable
  group-lock none
  pfs disable
  ipsec-udp enable
  ipsec-udp-port 10000
  split-tunnel-policy tunnelall
  split-tunnel-network-list none
  default-domain none

tunnel-group DefaultL2LGroup ipsec-attributes
  pre-shared-key *
isakmp keepalive threshold 20 retry 10
tunnel-group 172.24.68.112 type ipsec-l2l
tunnel-group 172.24.68.112 ipsec-attributes
  pre-shared-key *
tunnel-group-map enable rules
tunnel-group-map default-group DefaultL2LGroup
prompt hostname context
```

### 3.2.2 Certificate Generation

This chapter is an example showing how to generate the certificates on a Linux or Windows based machine.

1. CA - ca.key, ca.csr, ca.crt

   - `mkdir certs; cd certs; touch index.txt`
   - `import openssl.conf`
   - private key:
     `openssl genrsa -des3 -out ca.key 2048`
   - certificate signing request:
     `openssl req -verbose -new -key ca.key -out ca.csr -sha256`
   - self-sign CA certificate (see Appendix [A] for example of *openssl.conf*):
     `openssl ca -create_serial -extensions v3_ca -config ./openssl.conf -out ca.crt -keyfile ca.key -verbose -selfsign -md sha256 -enddate 301231235959Z -infiles ca.csr`
   - check CA certificate:
     `openssl x509 -noout -text -in ca.crt`

2. Server cisco cert - server_cisco.key, server_cisco.csr, server_cisco.crt

   - private key:
     `openssl genrsa -des3 -out server_cisco.key 2048`
   - certificate signing request: (see Appendix [B] for example of *server_req.conf*)
     `openssl req -verbose -new -key server_cisco.key -out server_cisco.csr -config server_req.conf`
   - self-sign server_cisco certificate:
     `openssl ca -config ./server_req.conf -extensions v3_req -enddate 301231235959Z -out server_cisco.crt -keyfile ca.key -infiles server_cisco.csr`
   - check server_cisco certificate:
     `openssl x509 -noout -text -in server_cisco.crt`

3. Client router cert - client_router.key, client_router.csr, client_router.crt

   - private key:
     `openssl genrsa -des3 -out  client_router.key 2048`
   - certificate signing request: (see Appendix [C] for example of *client_req.conf*)
     `openssl req -verbose -new -key  client_router.key -out client_router.csr -config client_req.conf`

- self-sign server_cisco certificate:
  `openssl ca -config ./client_req.conf -extensions v3_req -enddate 3012312359592 -out client_router.crt -keyfile ca.key -infiles client_router.csr`
- check client_router certificate:
  `openssl x509 -noout -text -in client_router.crt`

4. Verify if certs/keys are really corectly generated - they have to match.
   - `openssl x509 -noout -modulus -in [client_router/server_cisco].crt | openssl md5`
   - `openssl rsa -noout -modulus -in [client_router/server_cisco].key | openssl md5`
   - hashes have to be equal

### 3.2.3 How to Import Certificates to Cisco

This chapter is an example showing how to import ca, server key and server certificates to a Cisco device.

1. `configure terminal`
2. `crypto pki trustpoint server.ciso`

```
no revocation-check
enrollment terminal pem
exit
```

3. `crypto pki import server.cisco pem terminal password <password>`

```
paste ca certificate in PEM format
paste encrypted private server key in PEM format
paste server certificate in PEM format
exit
```

4. `crypto pki certificate map ike_v2_certmap 10`

```
subject-name co client
```

5. `show crypto pki trustpoint server.cisco status`

```
Trustpoint server.cisco:
Issuing CA certificate configured:
  Subject Name:
   e=advantech@advantech.com,cn=www.advantech.com,ou=Advantech CZ,o=Advantech,
                                                          st=Czechia,c=CZ
  Fingerprint MD5: 20514117 B5B696F5 00375153 A9DC864C
  Fingerprint SHA1: 532AA251 EB16DAEC 89BB97C4 DDE0D3E3 F7A07270
```

```
Router General Purpose certificate configured:
  Subject Name:
   cn=server@cisco,ou=Advantech CZ,o=Advantech,st=Czechia,c=CZ
  Fingerprint MD5: 1712292C A41F36FE 56F12682 1A503577
  Fingerprint SHA1: 01C99D4C 4064AFF6 123421A1 5A9F23BB 8DEA2D60
State:
  Keys generated ..... Yes (General Purpose, non-exportable)
  Issuing CA authenticated ..... Yes
  Certificate request(s) ..... Yes
```

**Note:** If cisco is configured by copy/paste raw config via terminal then private keys are not imported (only ca and cert is imported). In this case you can use these cmd to import private key:

1. `crypto key import rsa <name> terminal <password>`
2. `crypto pki trustpoint <name>`
   `rsakeypair <name>`

### 3.2.4 IKEv1 Certificate-based Tunnel

This chapter describes how to set up an IKEv1 certificate-based tunnel between the Cisco device and the Advantech router. The Cisco device acts as the server and the Advantech router as the client. See chaper 3.2.2 for demonstration of cerfificate generation and chapter 3.2.3 to see how to import it.

#### Setup of Cisco

1. `configure terminal`
2. `crypto pki certificate map ikev1_map 10`

```
subject-name co client
```

3. `crypto isakmp policy 10`

```
encr aes 256
hash sha256
group 14
```

4. `crypto isakmp identity dn` (identity is DN of server.cisco certificate)
5. `crypto isakmp profile ikev1`

```
ca trust-point server.cisco
match certificate ikev1_map
local-address <IP address>
```

6. `crypto map ike_v1_map 10 ipsec-isakmp`

   `set peer <IP address>`
   `set transform-set aeset` (esp algs and mode is the same as for ikev2)
   `set isakmp-profile ikev1`
   `match address ike_v2_acl` (traffic selector is the same as for ikev2)

7. `interface GigabitEthernet0`

   ```
   ip address <IP address> <mask>
     duplex auto
     speed auto
     no keepalive
     crypto map ike_v1_map
   ```

8. `exit`
9. `show crypto isakmp sa detail`

```
Codes: C - IKE configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal
T - cTCP encapsulation, X - IKE Extended Authentication
psk - Preshared key, rsig - RSA signature
renc - RSA encryption
IPv4 Crypto ISAKMP SA

C-id  Local           Remote     I-VRF  Status Encr Hash  Auth DH Lifetime Cap.

2966  <IP address>    <IP address>        ACTIVE aes  sha256 rsig 14 00:53:05
Engine-id:Conn-id =  SW:966
```

### Setup of Advantech Router

```
IPSEC_ENABLED=1
IPSEC_DESCRIPTION=
IPSEC_HOST_IPMODE=4
IPSEC_REMOTE_IPADDR=<IP address>
IPSEC_TUNNEL_IPMODE=4
IPSEC_REMOTE_ID=C=CZ,ST=Czechia,O=Advantech,OU=AdvantechCZ,CN=server@cisco
IPSEC_REMOTE_NETWORK=<IP address>
IPSEC_REMOTE_NETMASK=<mask>
IPSEC_REMOTE_NETWORK2=
IPSEC_REMOTE_NETMASK2=
IPSEC_REMOTE_PROTOPORT=
IPSEC_LOCAL_ID=<IP address>
IPSEC_LOCAL_NETWORK=<IP address>
```

```
IPSEC_LOCAL_NETMASK=<mask>
IPSEC_LOCAL_NETWORK2=
IPSEC_LOCAL_NETMASK2=
IPSEC_LOCAL_PROTOPORT=
IPSEC_IKE_PROTOCOL=ikev1
IPSEC_IKE_ALG=manual
IPSEC_IKE_ENC=aes256
IPSEC_IKE_HASH=sha2_256
IPSEC_IKE_DH=modp2048
IPSEC_IKE_REAUTH=0
IPSEC_XAUTH_ENABLED=0
IPSEC_XAUTH_MODE=client
IPSEC_XAUTH_USER=
IPSEC_XAUTH_PASS=
IPSEC_ESP_ALG=manual
IPSEC_ESP_ENC=aes256
IPSEC_ESP_HASH=sha2_256
IPSEC_PFS=0
IPSEC_PFS_DH=
IPSEC_KEY_LIFE=3600
IPSEC_IKE_LIFE=3600
IPSEC_REKEY_MARGIN=540
IPSEC_REKEY_FUZZ=100
IPSEC_DPD_DELAY=20
IPSEC_DPD_TIMEOUT=60
IPSEC_ENCAP=tunnel
IPSEC_FORCE_ENCAPS=0
IPSEC_AGGRESSIVE=0
IPSEC_AUTHBY=rsa
IPSEC_PSK=
IPSEC_CA_CERT=LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1.....
IPSEC_REMOTE_CERT=LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0.....
IPSEC_LOCAL_CERT=LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t.....
IPSEC_LOCAL_KEY=LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktL.....
IPSEC_LOCAL_PASS=conel000
IPSEC_DEBUG=1
```

### 3.2.5 IKEv2 Certificate-based Tunnel

This chapter describes how to set up an IKEv2 certificate-based tunnel between the Cisco device and the Advantech router. The Cisco device acts as the server and the Advantech router as the client. See chaper 3.2.2 for demonstration of cerfificate generation and chapter 3.2.3 to see how to import it.

#### Setup of Cisco

1. `configure terminal`
2. `crypto ikev2 authorization policy ike_v2_policy`

```
crypto ikev2 proposal ike_v2_proposal
encryption aes-cbc-256
integrity sha256
group 14
```

3. `crypto ikev2 policy ike_v2_policy`

```
proposal ike_v2_proposal
crypto ikev2 profile ike_v2_profile
match certificate ike_v2_certmap
identity local [ fqdn server.cisco | email server@cisco | address XX.XX.XX.XX ]
authentication remote rsa-sig
authentication local rsa-sig
pki trustpoint server.cisco
```

4. `crypto ipsec transform-set aeset esp-aes 256 esp-sha256-hmac`

```
mode tunnel
```

5. `crypto map ike_v2_map 10 ipsec-isakmp`

```
set peer <IP address>
set transform-set aeset
set ikev2-profile ike_v2_profile
match address ike_v2_acl
```

6. `ip access-list extended ike_v2_acl`

```
permit ip <local subnet> 0.0.0.255 <remote subnet> 0.0.0.255
```

7. interface GigabitEthernet0

```
ip address <IP address> <mask>
  duplex auto
  speed auto
  no keepalive
  crypto map ike_v2_map
```

8. exit
9. show crypto ikev2 session

```
IPv4 Crypto IKEv2 Session

Session-id:28, Status:UP-ACTIVE, IKE count:1, CHILD count:1

Tunnel-id Local              Remote             fvrf/ivrf          Status
1         <IP address>/4500     <IP address>/4500    none/none     READY
      Encr: AES-CBC, keysize: 256, Hash: SHA256, DH Grp:14, Auth sign: RSA,
                                                          Auth verify: RSA
      Life/Active Time: 86400/1149 sec
Child sa: local selector  192.168.6.0/0 - 192.168.6.255/65535
          remote selector 192.168.1.0/0 - 192.168.1.255/65535
          ESP spi in/out: 0xE5E902B1/0xC8A42CE4
```

10. show crypto ipsec sa

```
interface: GigabitEthernet0
    Crypto map tag: ike_v2_map, local addr <IP address>

   protected vrf: (none)
   local  ident (addr/mask/prot/port): (192.168.6.0/255.255.255.0/0/0)
   remote ident (addr/mask/prot/port): (192.168.1.0/255.255.255.0/0/0)
   current_peer <IP address> port 4500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 0, #pkts encrypt: 0, #pkts digest: 0
    #pkts decaps: 0, #pkts decrypt: 0, #pkts verify: 0
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0
    #send errors 0, #recv errors 0

     local crypto endpt.: <IP address>, remote crypto endpt.: <IP address>
     path mtu 1500, ip mtu 1500, ip mtu idb GigabitEthernet0
     current outbound spi: 0xC8A42CE4(3366202596)
     PFS (Y/N): N, DH group: none
```

```
        inbound esp sas:
         spi: 0xE5E902B1(3857253041)
           transform: esp-256-aes esp-sha256-hmac ,
           in use settings ={Tunnel, }
           conn id: 55, flow_id: Onboard VPN:55, sibling_flags 80000040,
                                                    crypto map: ike_v2_map
           sa timing: remaining key lifetime (k/sec): (4608000/2356)
           IV size: 16 bytes
           replay detection support: Y
           Status: ACTIVE(ACTIVE)

        inbound ah sas:
        inbound pcp sas:

        outbound esp sas:
         spi: 0xC8A42CE4(3366202596)
           transform: esp-256-aes esp-sha256-hmac ,
           in use settings ={Tunnel, }
           conn id: 56, flow_id: Onboard VPN:56, sibling_flags 80000040,
                                                    crypto map: ike_v2_map
           sa timing: remaining key lifetime (k/sec): (4608000/2356)
           IV size: 16 bytes
           replay detection support: Y
           Status: ACTIVE(ACTIVE)
```

11. show runnig-config

```
crypto pki trustpoint server.cisco

 revocation-check none
 rsakeypair server.cisco
!
!
!
crypto pki certificate map ike_v2_certmap 10
 subject-name co client
!
crypto pki certificate chain server.cisco
 certificate 29BEF8C0BE9377F585E4C9E7E569B4B1FEA8544A
  3082035A 30820242 A0030201 02020900 89CE1443 6667652F 300D0609 2A864886
  .....
  7A8B2AE7 2EF6FBB7 F9BE79B3 6DBD32C1 3F63EA9F 28460A23 122785C2 0504
  quit
certificate ca 29BEF8C0BE9377F585E4C9E7E569B4B1FEA8543C
  3082035D 30820245 A0030201 02020900 C32DDAD5 EF9ADEDE 300D0609 2A864886
```

```
  .....
  9AD70CB3 05431A4F DDA40424 657A29FF 5F1174FD 21171128 A541B781 CEAB845A C6
  quit
ip cef
!
!
!
!
!
crypto ikev2 authorization policy ike_v2_policy
!
crypto ikev2 proposal ike_v2_proposal
 encryption aes-cbc-256
 integrity sha256
 group 14
!
crypto ikev2 policy ike_v2_policy
 proposal ike_v2_proposal
!
!
crypto ikev2 profile ike_v2_profile
 match certificate ike_v2_certmap
 identity local fqdn server.cisco
 authentication remote rsa-sig
 authentication local rsa-sig
 pki trustpoint server.cisco
!
!
!
crypto ipsec transform-set aeset esp-aes 256 esp-sha256-hmac
 mode tunnel
!
!
crypto ipsec transform-set aeset esp-aes 256 esp-sha256-hmac
 mode tunnel
!
crypto map ike_v2_map 10 ipsec-isakmp
 set peer <IP address>
 set transform-set aeset
 set ikev2-profile ike_v2_profile
 match address ike_v2_acl
!
!
!
```

```
!
interface GigabitEthernet0
 ip address <IP address> <mask>
 ip access-group 101 in
 duplex auto
 speed auto
 no keepalive
 crypto map ike_v2_map
!
interface Vlan1
 ip address <cisco subnet> <mask>
!
!
!
ip access-list extended ike_v2_acl
 permit ip <cisco's subnet> <mask> <router's subnet> <mask>
!
access-list 101 permit ip any any
access-list 101 permit icmp any any
```

### Setup of Advantech Router

```
IPSEC_ENABLED=1
IPSEC_DESCRIPTION=
IPSEC_HOST_IPMODE=4
IPSEC_REMOTE_IPADDR=<IP address>
IPSEC_TUNNEL_IPMODE=4
IPSEC_REMOTE_ID=server.cisco
IPSEC_REMOTE_NETWORK=<IP address>
IPSEC_REMOTE_NETMASK=<mask>
IPSEC_REMOTE_NETWORK2=
IPSEC_REMOTE_NETMASK2=
IPSEC_REMOTE_PROTOPORT=
IPSEC_LOCAL_ID=client.router
IPSEC_LOCAL_NETWORK=<IP address>
IPSEC_LOCAL_NETMASK=<mask>
IPSEC_LOCAL_NETWORK2=
IPSEC_LOCAL_NETMASK2=
IPSEC_LOCAL_PROTOPORT=
IPSEC_IKE_PROTOCOL=ikev2
IPSEC_IKE_ALG=manual
IPSEC_IKE_ENC=aes256
IPSEC_IKE_HASH=sha2_256
IPSEC_IKE_DH=modp2048
IPSEC_IKE_REAUTH=1
IPSEC_XAUTH_ENABLED=0
IPSEC_XAUTH_MODE=
IPSEC_XAUTH_USER=
IPSEC_XAUTH_PASS=
IPSEC_ESP_ALG=manual
IPSEC_ESP_ENC=aes256
IPSEC_ESP_HASH=sha2_256
IPSEC_PFS=0
IPSEC_PFS_DH=
IPSEC_KEY_LIFE=3600
IPSEC_IKE_LIFE=3600
IPSEC_REKEY_MARGIN=540
IPSEC_REKEY_FUZZ=100
IPSEC_DPD_DELAY=20
IPSEC_DPD_TIMEOUT=60
IPSEC_ENCAP=tunnel
IPSEC_FORCE_ENCAPS=0
IPSEC_AGGRESSIVE=0
IPSEC_AUTHBY=rsa
```

```
IPSEC_PSK=
IPSEC_CA_CERT=LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1.....
IPSEC_REMOTE_CERT=LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0.....
IPSEC_LOCAL_CERT=LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t.....
IPSEC_LOCAL_KEY=LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktL.....
IPSEC_LOCAL_PASS=conel000
IPSEC_DEBUG=1
```

### 3.2.6 IKEv2 with Asymmetric Pre-shared Key

This chapter describes how to set up an IKEv2 with asymetric pre-shared key tunnel between the Cisco device and the Advantech router.

#### Setup of Cisco

```
!
aaa new-model
!
aaa authorization network FLEXVPN-AAA-AUTHORIZATION local
!
crypto ikev2 authorization policy ike_v2_policy
!
crypto ikev2 authorization policy IKE-AUTH-POLICY
 pool VPN-SPLIT-TUNNEL-ADDRESSES
 route set interface
!
crypto ikev2 proposal ike_v2_proposal
 encryption aes-gcm-256
 prf sha256
 group 21
!
crypto ikev2 policy ike_v2_policy
 proposal ike_v2_proposal
!
!
crypto ikev2 profile ike_v2_profile
 match identity remote any
 identity local fqdn server.cisco
 authentication remote pre-share key router
 authentication local pre-share key cisco
 aaa authorization group psk list FLEXVPN-AAA-AUTHORIZATION IKE-AUTH-POLICY
 virtual-template 20
!
crypto ipsec transform-set aes-gcm esp-gcm 256
 mode transport
!
crypto ipsec profile FlexVPN
 set security-policy limit 100
 set transform-set aes-gcm
 set pfs group21
 set ikev2-profile ike_v2_profile
 responder-only
```

```
!
interface Loopback2
 ip address 172.16.100.1 255.255.255.255
!
interface GigabitEthernet0/0/0
 ip address 10.40.29.128 255.255.252.0
 ip nat outside
 ip access-group 101 in
 negotiation auto
 spanning-tree portfast disable
!
interface GigabitEthernet0/0/1.202
 encapsulation dot1Q 202
 ip address 192.168.202.254 255.255.255.0
!
interface GigabitEthernet0
 vrf forwarding Mgmt-intf
 no ip address
 negotiation auto
!
interface Virtual-Template20 type tunnel
 ip unnumbered Loopback2
 no ip redirects
 tunnel source 10.40.29.128
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile FlexVPN
!
ip local pool VPN-SPLIT-TUNNEL-ADDRESSES 172.16.100.2 172.16.100.200
ip route 0.0.0.0 0.0.0.0 10.40.30.1
ip route 0.0.0.0 0.0.0.0 GigabitEthernet0/0/0
ip route 172.16.100.0 255.255.255.0 Null0
!
ip access-list extended FlexVPN_ACL
 permit ip 192.168.202.0 0.0.0.255 192.168.133.0 0.0.0.255
ip access-list extended NAT-ACL
 deny   ip any 192.168.1.0 0.0.0.255
access-list 20 permit 192.168.202.0 0.0.0.255
access-list 101 permit ip any any
access-list 101 permit esp any any
access-list 101 permit gre any any
access-list 101 permit icmp any any
!
!
```

### Setup of Advantech Router

```
IPSEC_ENABLED=1
IPSEC_DESCRIPTION="FlexVPN with asym. PSK"
IPSEC_TYPE=route
IPSEC_HOST_IPMODE=4
IPSEC_REMOTE_IPADDR=10.40.29.128
IPSEC_REMOTE_IPADDR2=
IPSEC_TUNNEL_IPMODE=4
IPSEC_REMOTE_ID=server.cisco
IPSEC_LOCAL_ID=client@router
IPSEC_INSTALL_ROUTES=0
IPSEC_REMOTE_NETWORK=0.0.0.0
IPSEC_REMOTE_NETMASK=0.0.0.0
IPSEC_REMOTE_NETWORK2=
IPSEC_REMOTE_NETMASK2=
IPSEC_REMOTE_PROTOPORT=
IPSEC_LOCAL_NETWORK=0.0.0.0
IPSEC_LOCAL_NETMASK=0.0.0.0
IPSEC_LOCAL_NETWORK2=
IPSEC_LOCAL_NETMASK2=
IPSEC_LOCAL_PROTOPORT=
IPSEC_MTU=1426
IPSEC_REMOTE_VIRTUAL_NETWORK=
IPSEC_REMOTE_VIRTUAL_MASK=
IPSEC_LOCAL_VIRTUAL_IP=0.0.0.0
IPSEC_CISCO_FLEXVPN=1
IPSEC_IKE_PROTOCOL=ikev2
IPSEC_IKE_ALG=manual
IPSEC_IKE_ENC=aes256gcm128
IPSEC_IKE_HASH=sha2_256
IPSEC_IKE_DH=ecp521
IPSEC_IKE_REAUTH=1
IPSEC_XAUTH_ENABLED=0
IPSEC_XAUTH_MODE=
IPSEC_XAUTH_USER=
IPSEC_XAUTH_PASS=
IPSEC_ESP_ALG=manual
IPSEC_ESP_ENC=aes256gcm128
IPSEC_ESP_HASH=
IPSEC_PFS=1
IPSEC_PFS_DH=ecp521
IPSEC_KEY_LIFE=3600
IPSEC_IKE_LIFE=3600
```

```
IPSEC_REKEY_MARGIN=540
IPSEC_REKEY_FUZZ=100
IPSEC_DPD_DELAY=10
IPSEC_DPD_TIMEOUT=20
IPSEC_ENCAP=tunnel
IPSEC_FORCE_ENCAPS=0
IPSEC_AGGRESSIVE=0
IPSEC_AUTHBY=secret
IPSEC_PSK=router
IPSEC_REMOTE_PSK=cisco
IPSEC_CA_CERT=
IPSEC_REMOTE_CERT=
IPSEC_LOCAL_CERT=
IPSEC_LOCAL_KEY=
IPSEC_LOCAL_PASS=
IPSEC_REVOCATION=
IPSEC_DEBUG=1
```

## 3.3 Windows Computer IPsec Tunnel with Advantech Router

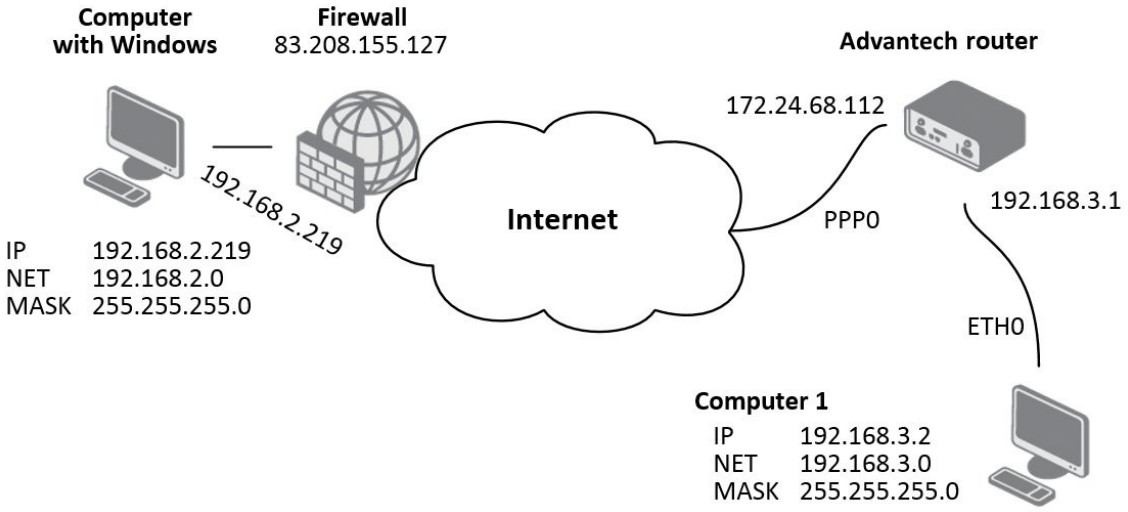



Figure 17: IPsec tunnel – Windows

Recommended program for Windows operating system is *NCP Secure Entry Client* on which the following description is based on.

### 3.3.1 Windows IPsec Configuration – NCP Secure Entry Client

The figure below shows the environment of the NCP Secure Entry Client (version 9.32, build 218).

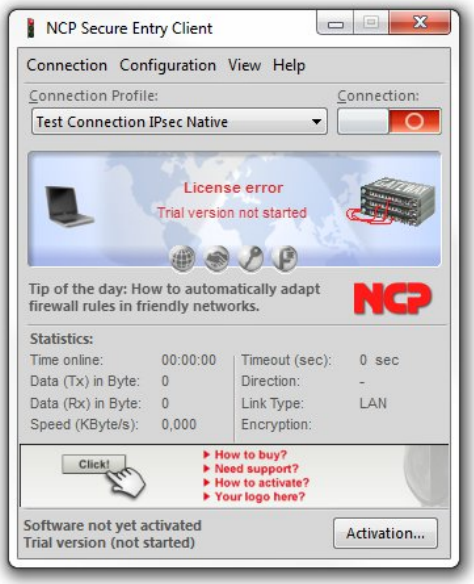



Figure 18: NCP Secure Entry Client

First it is necessary to create a profile for establishing IPsec tunnel. Select *Configuration* tab in the menu (of NCP Secure Entry Client program) and then select *Profiles* item. The following window will be open:

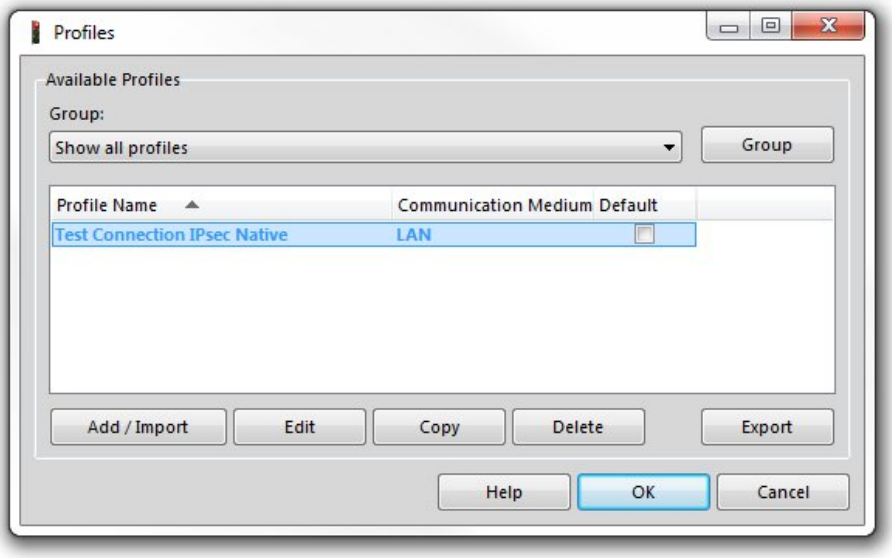



Figure 19: NCP Secure Entry Client – Profiles

Add a new profile using the *Add/Import* button. On the second screen, you must enter the profile name. In other cases (on the other screens) it is possible only to confirm using the *Next* button (on the last screen using the *Finish* button) and make the necessary settings later.

Configuration of the IPsec tunnel is done by marking the profile and pressing *Edit* button.

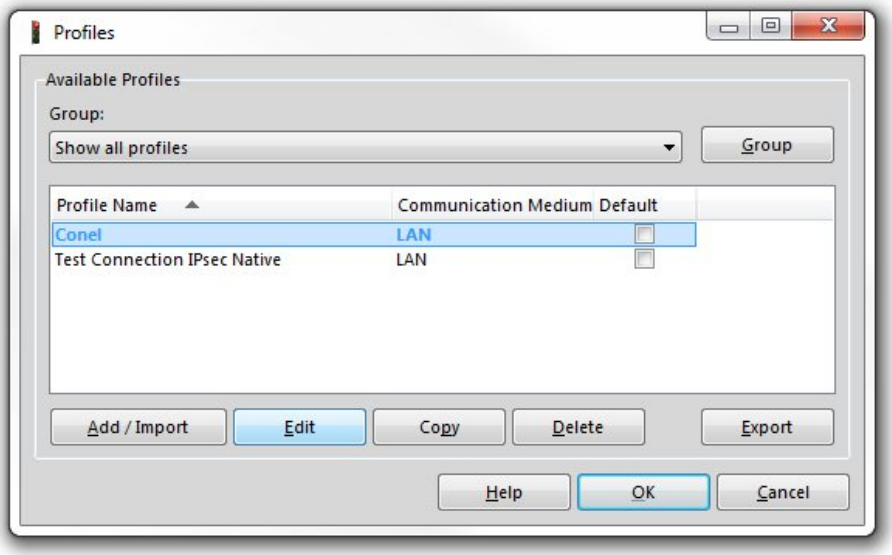



Figure 20: NCP Secure Entry Client – Edit

Select *IPsec General Settings* item in the menu on the left side. Then press *Police Editor. . .* button on the right side.

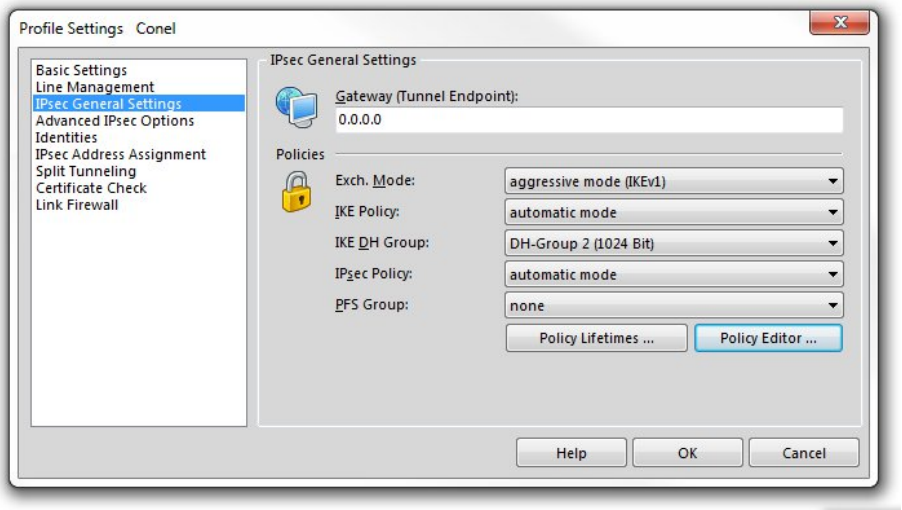



Figure 21: NCP Secure Entry Client – IPsec General Settings

In the new window highlight the *Pre-shared Key* item (in *IKE Policy* section) and then press *Edit* button.

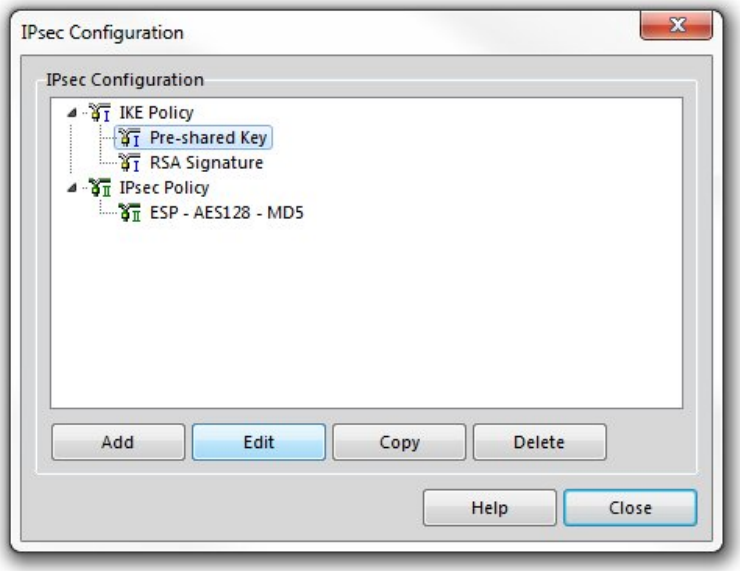



Figure 22: NCP Secure Entry Client – Policy Editor

This opens a window in which select encryption and hash algorithm (for example *Triple DES* and *MD5*) and then confirm by pressing the *OK* button.

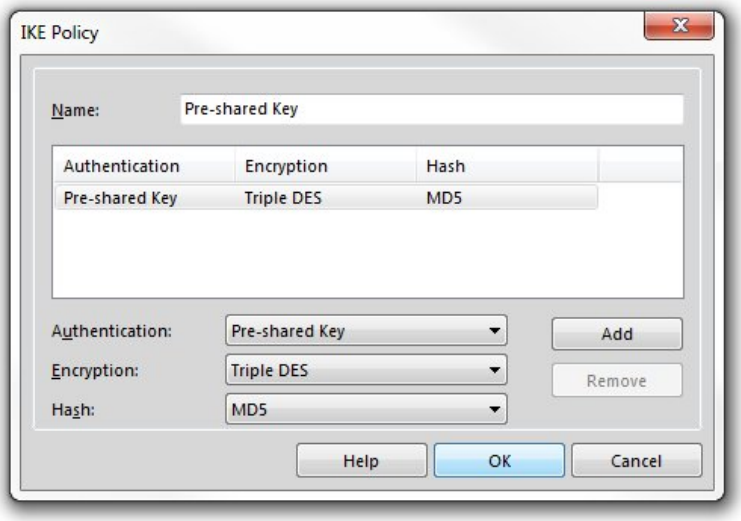



Figure 23: NCP Secure Entry Client – Pre-shared Key

Now, select the only available item in *IPsec Policy* section of configuration window. The item has a name *ESP - AES128 - MD5*. Then press *Edit* button.

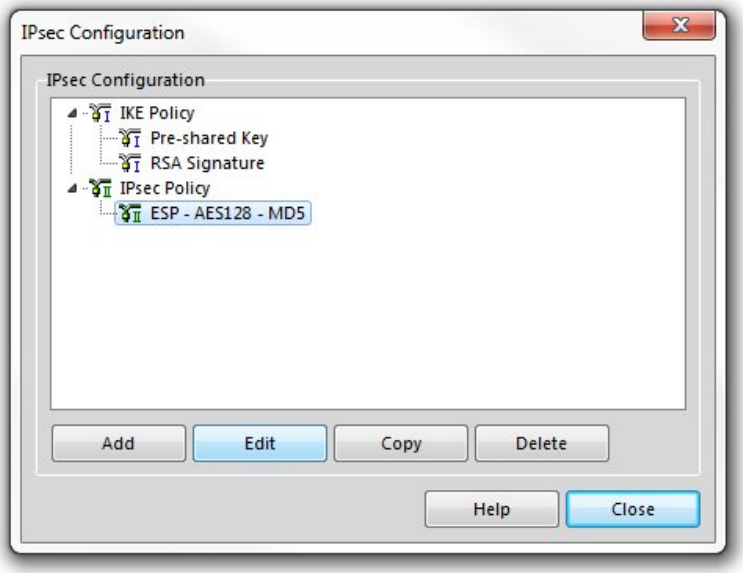



Figure 24: NCP Secure Entry Client – Policy Editor

Enter the desired name (for example *IPsec*) in the new window and select encryption and hash algorithm (for example *Triple DES* and *MD5*). Then confirm it by pressing the *OK* button.

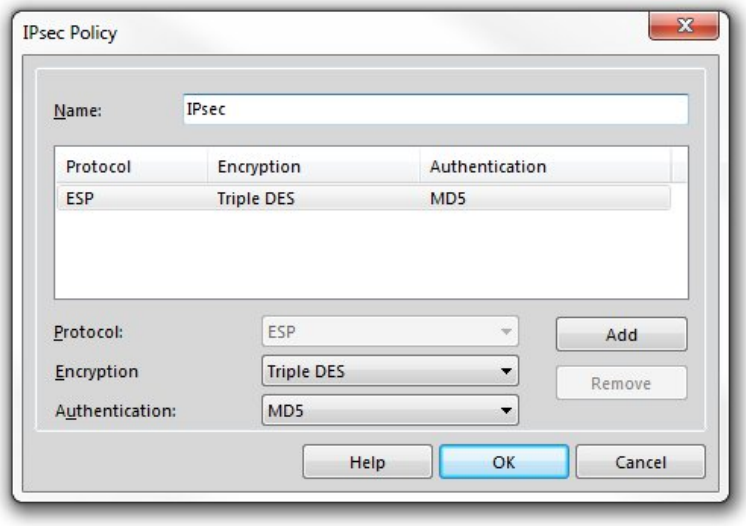



Figure 25: NCP Secure Entry Client – IPsec Policy

Go back to the main window of *IPsec General Settings* item and set *IKE Policy* and *IPsec Policy* items based on the previous configuration (see the figure below). *IKE DH Group* item will have a value of *DH-Group 2 (2014 bit)*.

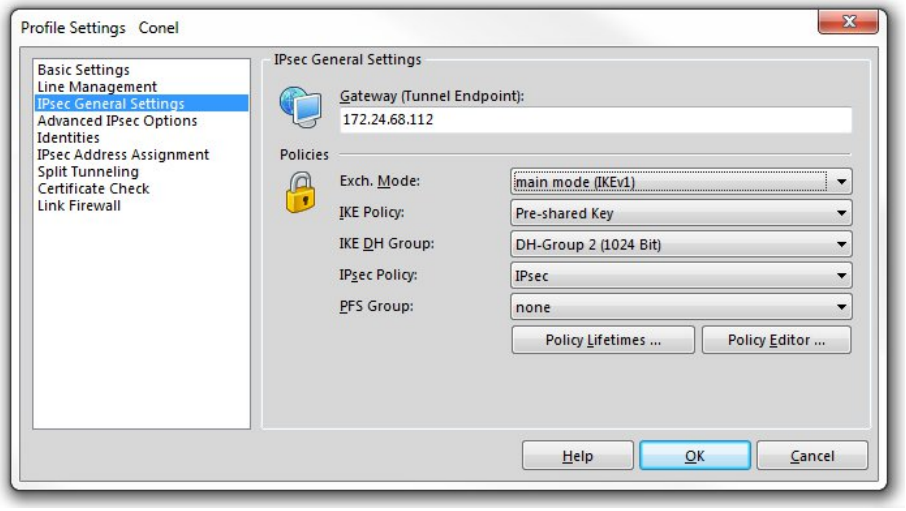



Figure 26: NCP Secure Entry Client – IPsec General Settings

Now, select *Identities* item in the menu on the left side and fill in the configuration form as shown below. Note that the IP address corresponds to the exemplary situation from the beginning of this section.

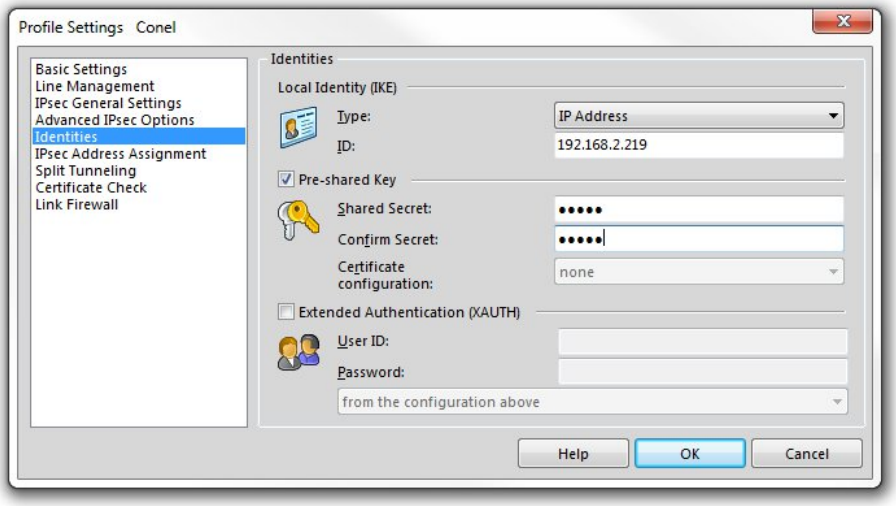



Figure 27: NCP Secure Entry Client – Identities

The same IP address (192.168.2.219 acccording to the exemplary situation) is also required on the *IPsec Address Assignment* page.

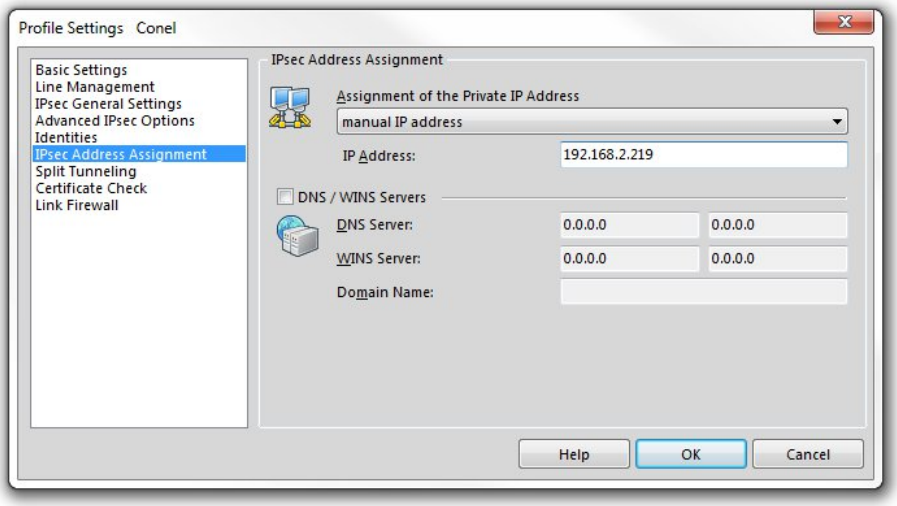



Figure 28: NCP Secure Entry Client – IPsec Address Assignment

Press *Add* button on the *Split Tunneling* page and enter the IP address of the subnet behind the router Advantech (192.168.3.0 in the exemplary situation) and relevant subnet mask (255.255.255.0) to the newly opened window. Confirm it by pressing the *OK* button.

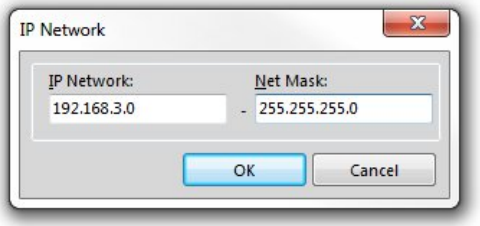



Figure 29: NCP Secure Entry Client – Add IP network

Specified data are displayed in the original window of the *Split Tunneling* page.

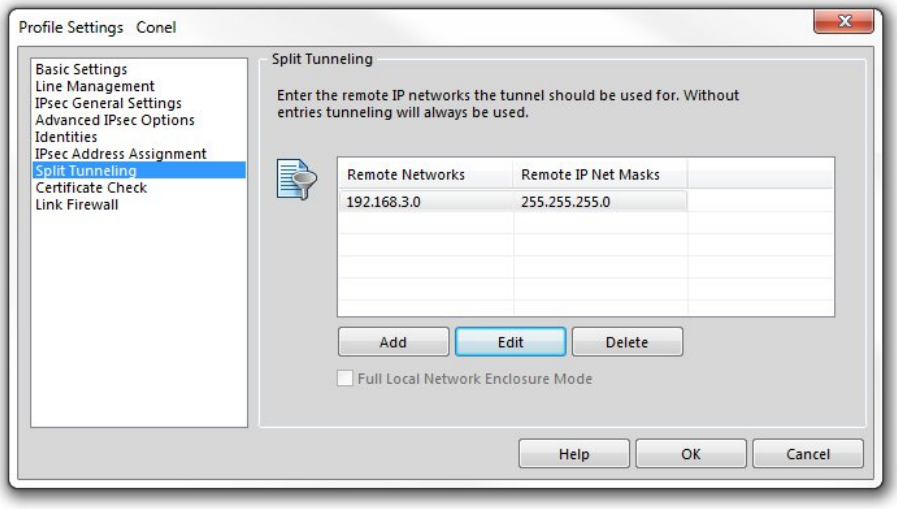



Figure 30: NCP Secure Entry Client – Split Tunneling

### 3.3.2 IPsec Configuration of Advantech Router

On the following page is displayed configuration form with IPsec tunnel settings. Entered values correspond to the exemplary situation from the beginning of this section.

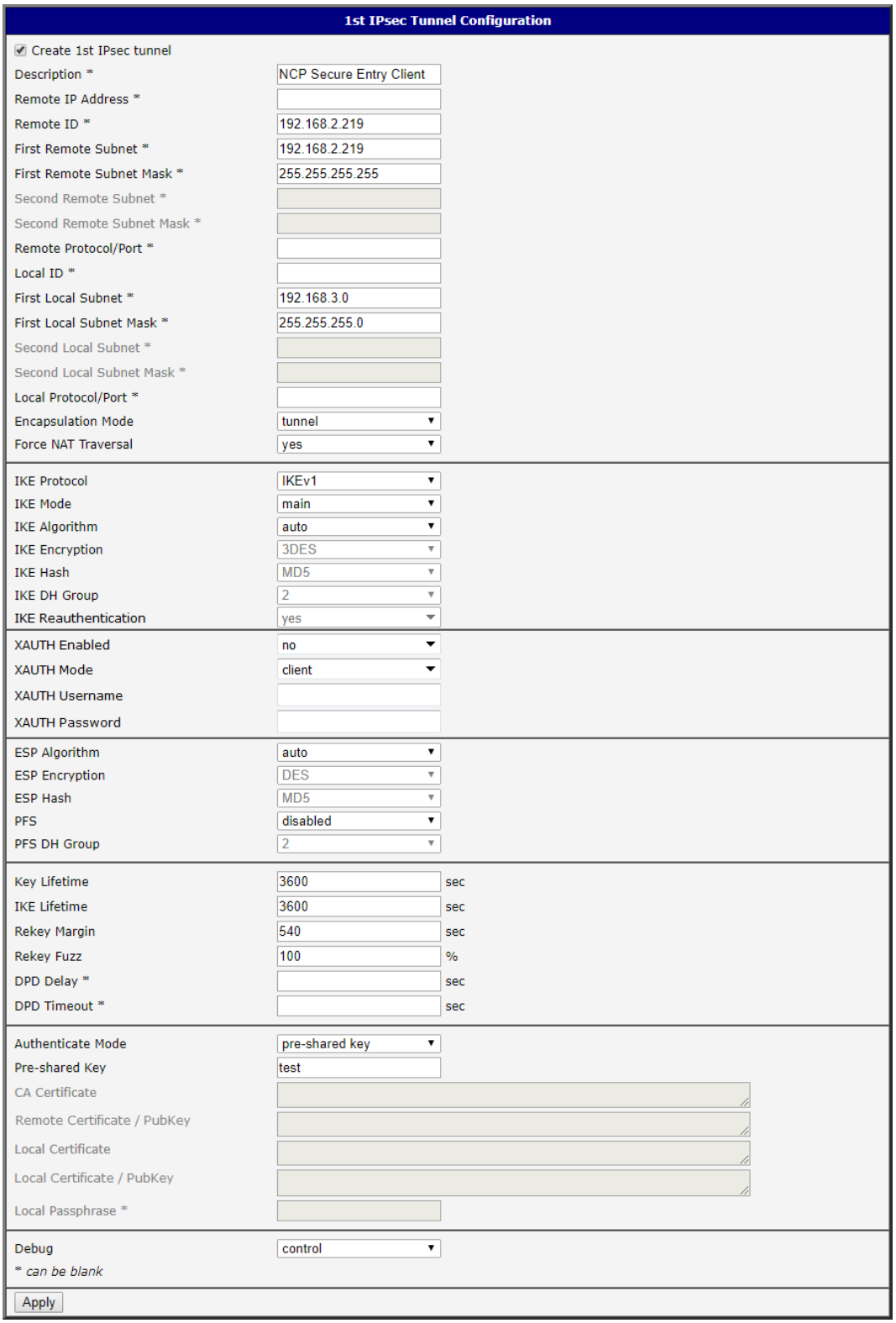



Figure 31: Configuration of Advantech router

## 3.4 Route-based IPsec

For more information about route-based IPsec configuration see the Route-based VPNs strongSwan webpage.

### 3.4.1 Multiple Clients

This example demonstrates the configuration of multiple IPsec clients, where one Advantech router (IP 10.65.0.64) acts as the server and assigns IP addressed to all the clients (IP 10.64.0.65) on the network. For more information see the Virtual IP strongSwan webpage.

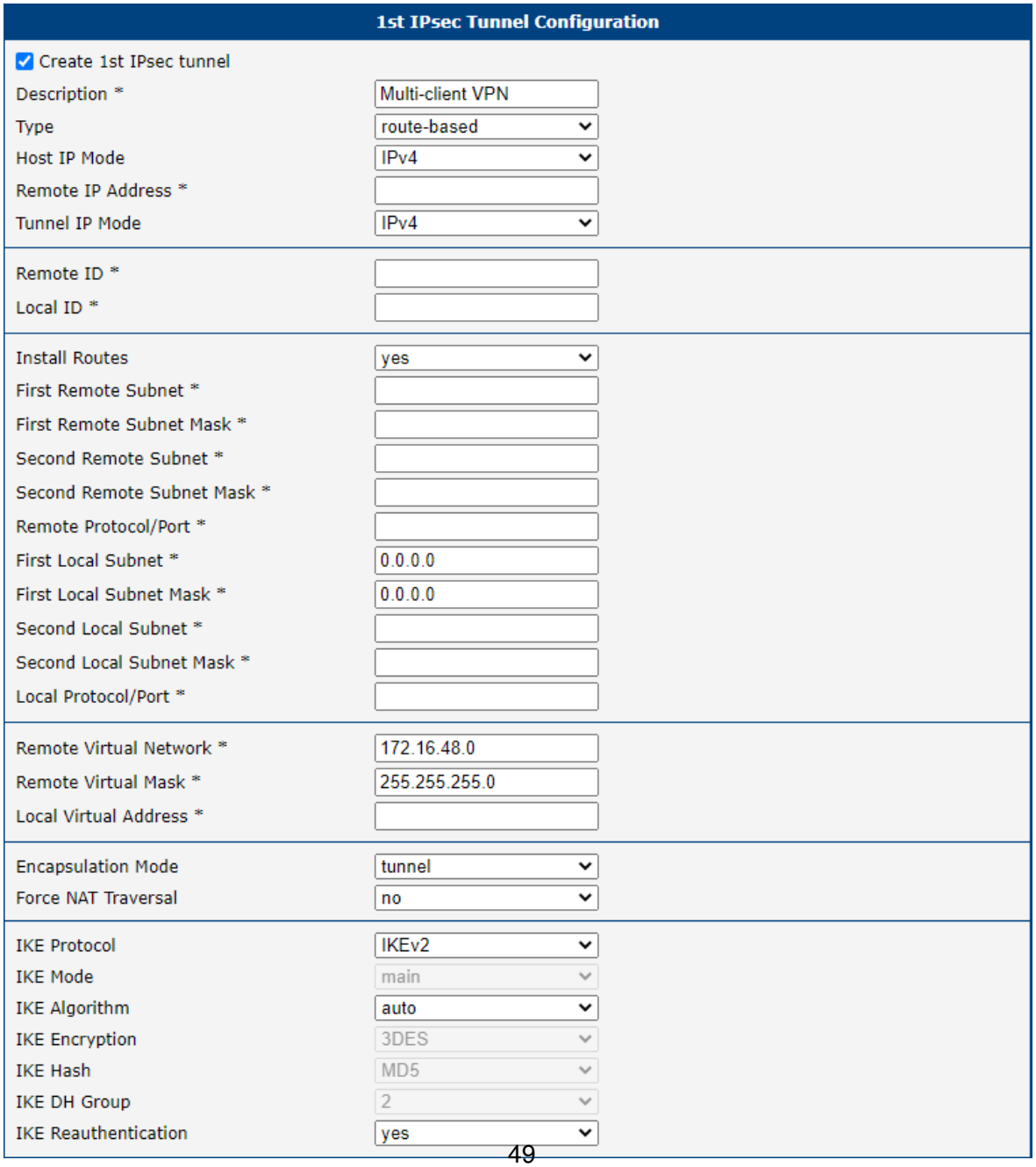



Figure 32: Server Configutaion

**1st IPsec Tunnel Configuration**

| | |
|---|---|
| ☑ Create 1st IPsec tunnel | |
| Description * | Multi-client VPN |
| Type | route-based |
| Host IP Mode | IPv4 |
| Remote IP Address * | 10.65.0.64 |
| Tunnel IP Mode | IPv4 |
| Remote ID * | |
| Local ID * | |
| Install Routes | yes |
| First Remote Subnet * | 0.0.0.0 |
| First Remote Subnet Mask * | 0.0.0.0 |
| Second Remote Subnet * | |
| Second Remote Subnet Mask * | |
| Remote Protocol/Port * | |
| First Local Subnet * | |
| First Local Subnet Mask * | |
| Second Local Subnet * | |
| Second Local Subnet Mask * | |
| Local Protocol/Port * | |
| Remote Virtual Network * | |
| Remote Virtual Mask * | |
| Local Virtual Address * | 0.0.0.0 |
| Encapsulation Mode | tunnel |
| Force NAT Traversal | no |
| IKE Protocol | IKEv2 |
| IKE Mode | main |
| IKE Algorithm | auto |
| IKE Encryption | 3DES |
| IKE Hash | MD5 |
| IKE DH Group | 2 |
| IKE Reauthentication | yes |

Figure 33: Client Configuration

**IPsec Status**

IPsec Tunnels Information

```
Daemon Information:

strongSwan swanctl 5.9.2
uptime: 35 minutes, since May 10 08:35:02 2021
worker threads: 16 total, 11 idle, working: 4/0/1/0
job queues: 0/0/0/0
jobs scheduled: 6
IKE_SAs: 2 total, 0 half-open
mallinfo: sbrk 671744, mmap 0, used 465992, free 205752
loaded plugins: charon nonce revocation pubkey pem openssl curl kernel-netlink socket-default vici updown xauth-generic

Connections:

ipsec1: IKEv2, reauthentication every 3060s, no rekeying
  local:  0.0.0.0
  remote: 0.0.0.0
  local pre-shared key authentication:
  remote pre-shared key authentication:
  ipsec1: TUNNEL, rekeying every 3060s
    local:  0.0.0.0/0
    remote: dynamic

Security Associations:

ipsec1: #3, ESTABLISHED, IKEv2, 99b579c0cd7af3a0_i 689b4c428785f7e5_r*
  local  '10.65.0.64' @ 10.65.0.64[4500]
  remote '10.65.0.65' @ 10.65.0.65[4500] [172.16.48.1]
  AES_CBC-128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
  established 100s ago, reauth in 2837s
  ipsec1: #3, reqid 1, INSTALLED, TUNNEL, ESP:AES_CBC-128/HMAC_SHA1_96
    installed 100s ago, rekeying in 2556s, expires in 3500s
    in  c3c1434b (-|0x00000001),      0 bytes,     0 packets
    out c10a9e02 (-|0x00000001),      0 bytes,     0 packets
    local  0.0.0.0/0
    remote 172.16.48.1/32
ipsec1: #2, ESTABLISHED, IKEv2, 8b9e0a6637a3c663_i 3bb02d38d4c3a93c_r*
  local  '10.65.0.64' @ 10.65.0.64[4500]
  remote '10.65.0.66' @ 10.65.0.66[4500] [172.16.48.2]
  AES_CBC-128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
  established 2059s ago, reauth in 934s
  ipsec1: #2, reqid 2, INSTALLED, TUNNEL, ESP:AES_CBC-128/HMAC_SHA1_96
    installed 2059s ago, rekeying in 840s, expires in 1541s
    in  c5197826 (-|0x00000001),   1176 bytes,    14 packets
    out cf5f7a8e (-|0x00000001),   1176 bytes,    14 packets,  1958s ago
    local  0.0.0.0/0
    remote 172.16.48.2/32


pool-ipsec1          172.16.48.0                       2 / 0 / 254
```

Figure 34: Server IPsec Status

**IPsec Status**

IPsec Tunnels Information

```
Daemon Information:

strongSwan swanctl 5.9.2
uptime: 40 minutes, since May 10 08:34:50 2021
worker threads: 16 total, 11 idle, working: 4/0/1/0
job queues: 0/0/0/0
jobs scheduled: 7
IKE_SAs: 1 total, 0 half-open
mallinfo: sbrk 708608, mmap 0, used 577056, free 131552
loaded plugins: charon nonce revocation pubkey pem openssl curl kernel-netlink socket-default vici updown xauth-generic

Connections:

ipsec1: IKEv2, reauthentication every 3060s, no rekeying
  local:  0.0.0.0
  remote: 10.65.0.64
  local pre-shared key authentication:
  remote pre-shared key authentication:
  ipsec1: TUNNEL, rekeying every 3060s
    local:  dynamic
    remote: 0.0.0.0/0

Security Associations:

ipsec1: #3, ESTABLISHED, IKEv2, 99b579c0cd7af3a0_i* 689b4c428785f7e5_r
  local  '10.65.0.65' @ 10.65.0.65[4500] [172.16.48.1]
  remote '10.65.0.64' @ 10.65.0.64[4500]
  AES_CBC-128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
  established 387s ago, reauth in 2009s
  ipsec1: #3, reqid 1, INSTALLED, TUNNEL, ESP:AES_CBC-128/HMAC_SHA1_96
    installed 388s ago, rekeying in 2150s, expires in 3213s
    in  c10a9e02 (-|0x00000001),      0 bytes,     0 packets
    out c3c1434b (-|0x00000001),      0 bytes,     0 packets
    local  172.16.48.1/32
    remote 0.0.0.0/0
```

Figure 35: Client IPsec Status

Route Table

| Destination | Gateway | Genmask | Flags | Metric | Ref | Use | Iface |
|---|---|---|---|---|---|---|---|
| 0.0.0.0 | 192.168.253.254 | 0.0.0.0 | UG | 0 | 0 | 0 | usb0 |
| 10.64.0.0 | 0.0.0.0 | 255.255.252.0 | U | 0 | 0 | 0 | eth0 |
| 10.65.0.0 | 0.0.0.0 | 255.255.252.0 | U | 0 | 0 | 0 | eth1 |
| 172.16.48.1 | 0.0.0.0 | 255.255.255.255 | UH | 0 | 0 | 0 | ipsec0 |
| 172.16.48.2 | 0.0.0.0 | 255.255.255.255 | UH | 0 | 0 | 0 | ipsec0 |
| 192.168.253.254 | 0.0.0.0 | 255.255.255.255 | UH | 0 | 0 | 0 | usb0 |

Figure 36: Server Route Table

Route Table

| Destination | Gateway | Genmask | Flags | Metric | Ref | Use | Iface |
|---|---|---|---|---|---|---|---|
| 0.0.0.0 | 0.0.0.0 | 128.0.0.0 | U | 0 | 0 | 0 | ipsec0 |
| 0.0.0.0 | 192.168.253.254 | 0.0.0.0 | UG | 0 | 0 | 0 | usb0 |
| 10.64.0.0 | 0.0.0.0 | 255.255.252.0 | U | 0 | 0 | 0 | eth0 |
| 10.65.0.0 | 0.0.0.0 | 255.255.252.0 | U | 0 | 0 | 0 | eth1 |
| 128.0.0.0 | 0.0.0.0 | 128.0.0.0 | U | 0 | 0 | 0 | ipsec0 |
| 192.168.253.254 | 0.0.0.0 | 255.255.255.255 | UH | 0 | 0 | 0 | usb0 |

Figure 37: Client Route Table

### 3.4.2 Static Routes

This example demonstrates the configuration of IPsec server (IP 10.64.0.64) and client (IP 10.64.0.65), where the routes are installed statically by *FRR/zebra* and *FRR/staticd* applications configured in the FRR Router App, which has to be installed and configured on both routers. For more information about the FRR, free software IP routing suite, see FRRouting User Guide.

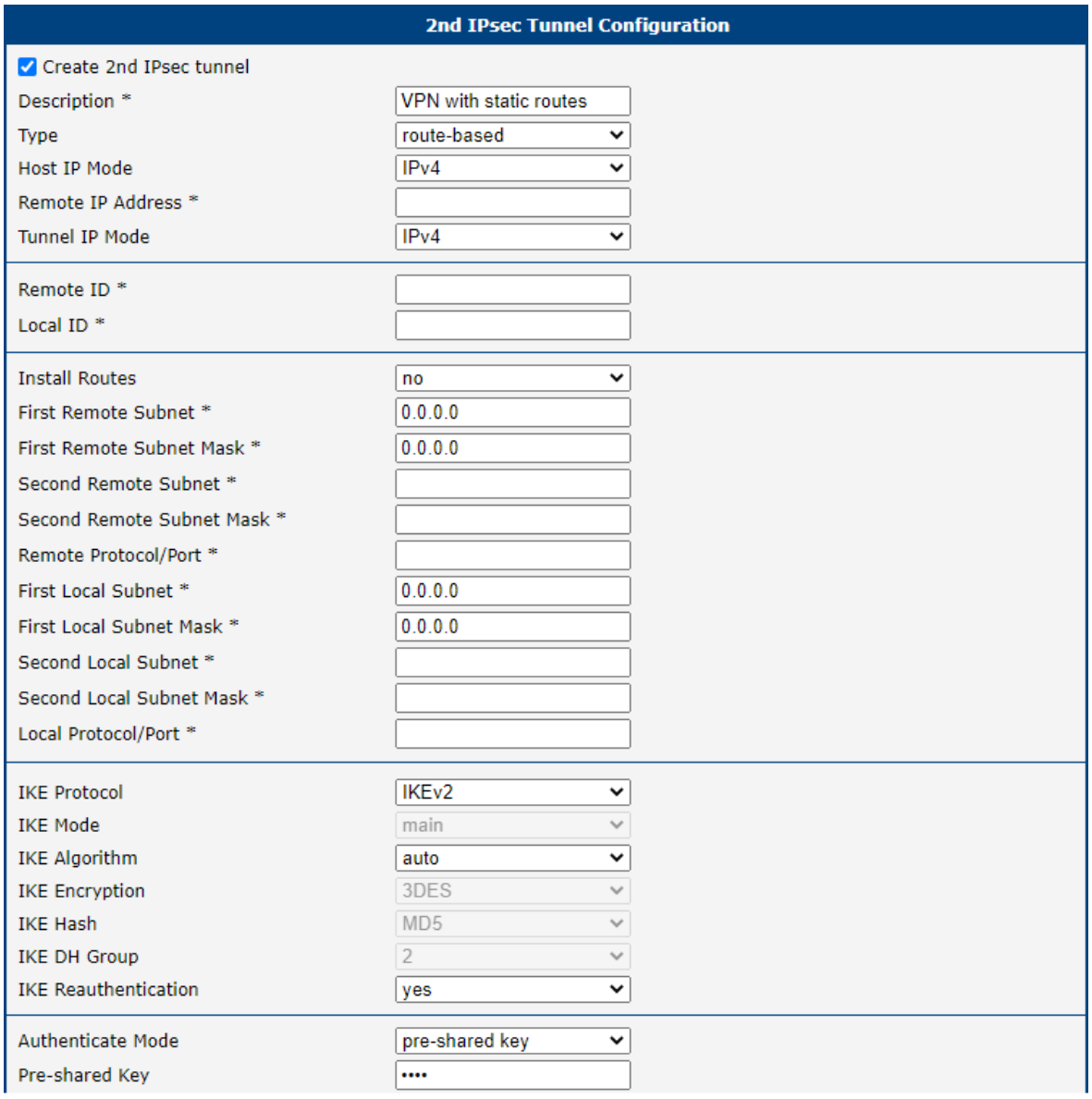



Figure 38: Server Configutaion

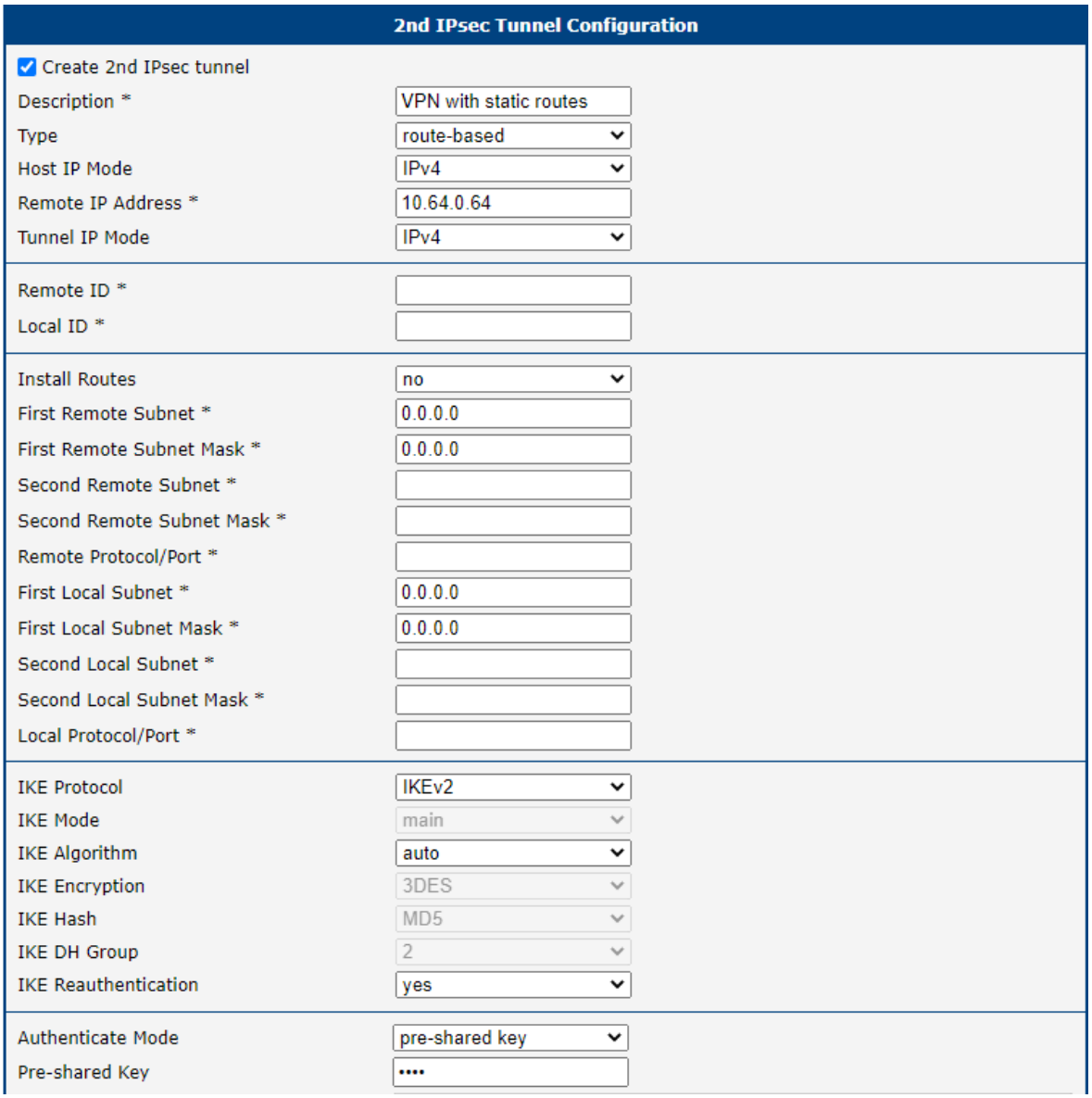



Figure 39: Client Configuration

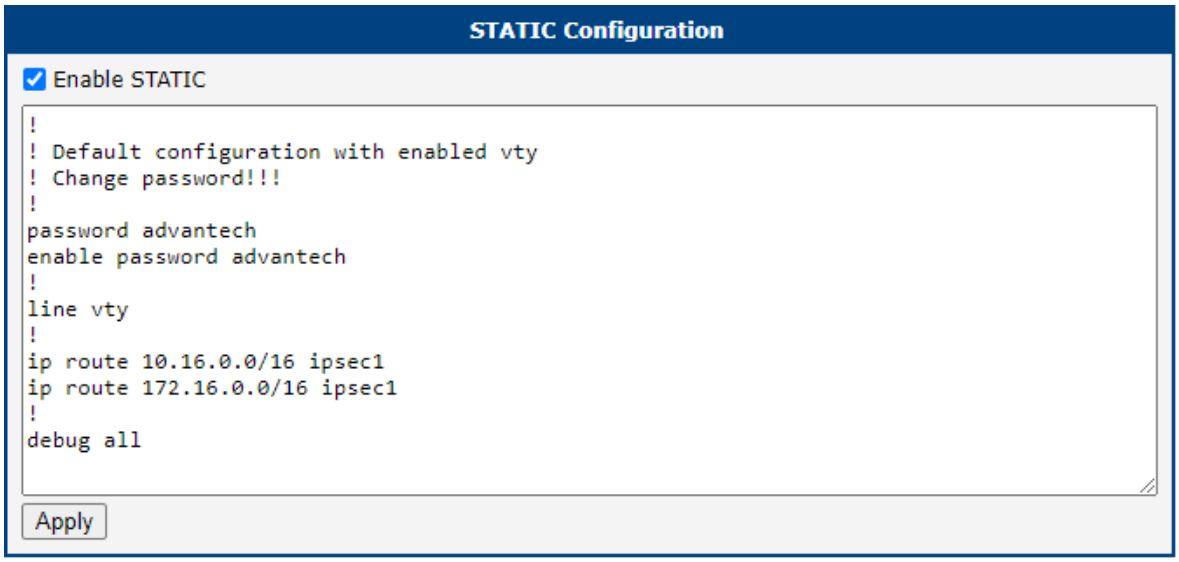



Figure 40: Server FRR Static Configuration

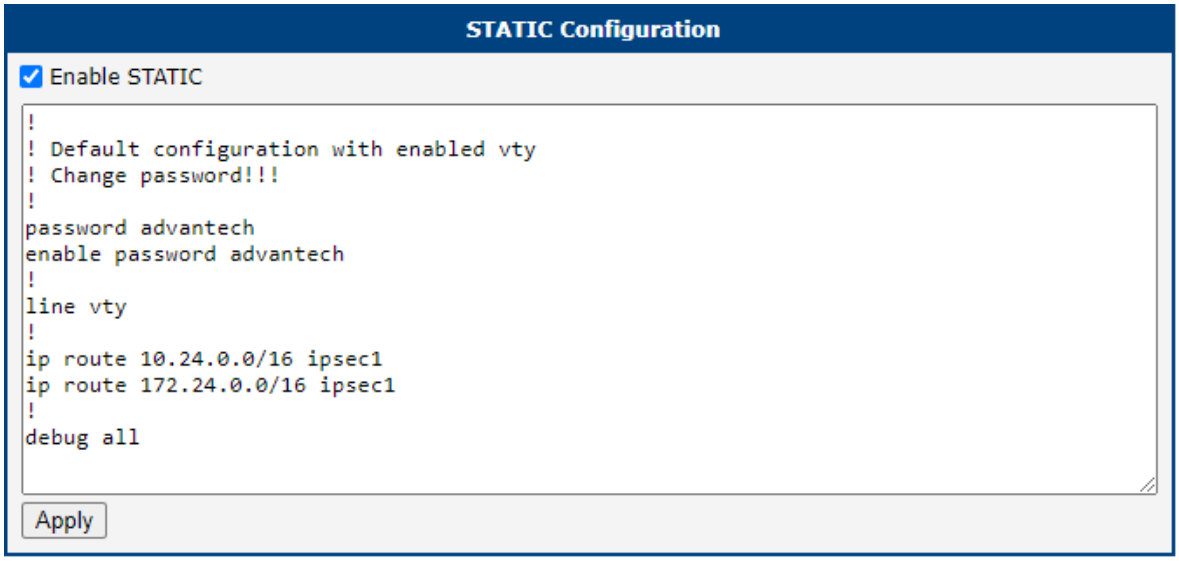



Figure 41: Client FRR Static Configuration

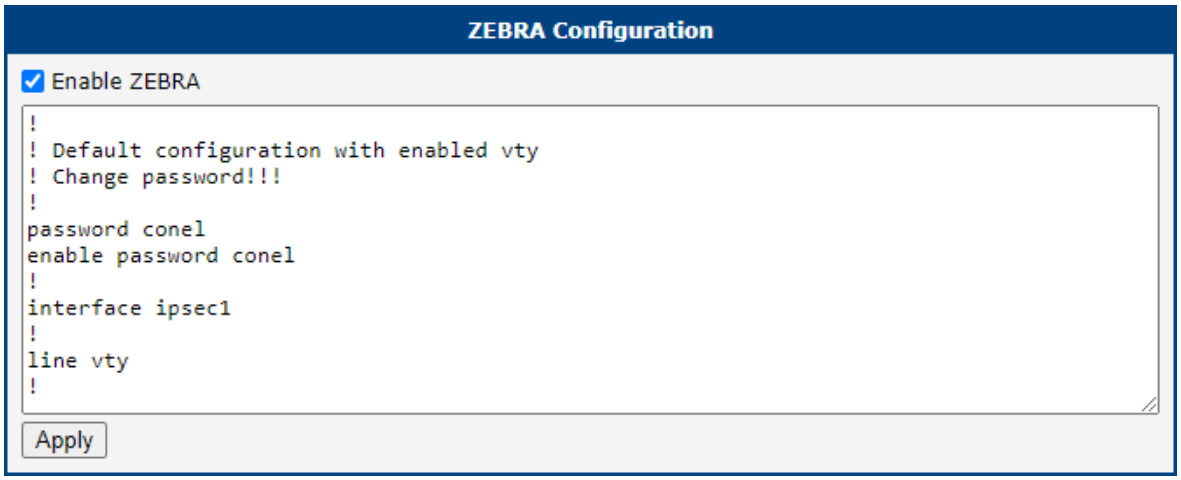



Figure 42: Client and Server FRR Zebra Configuration

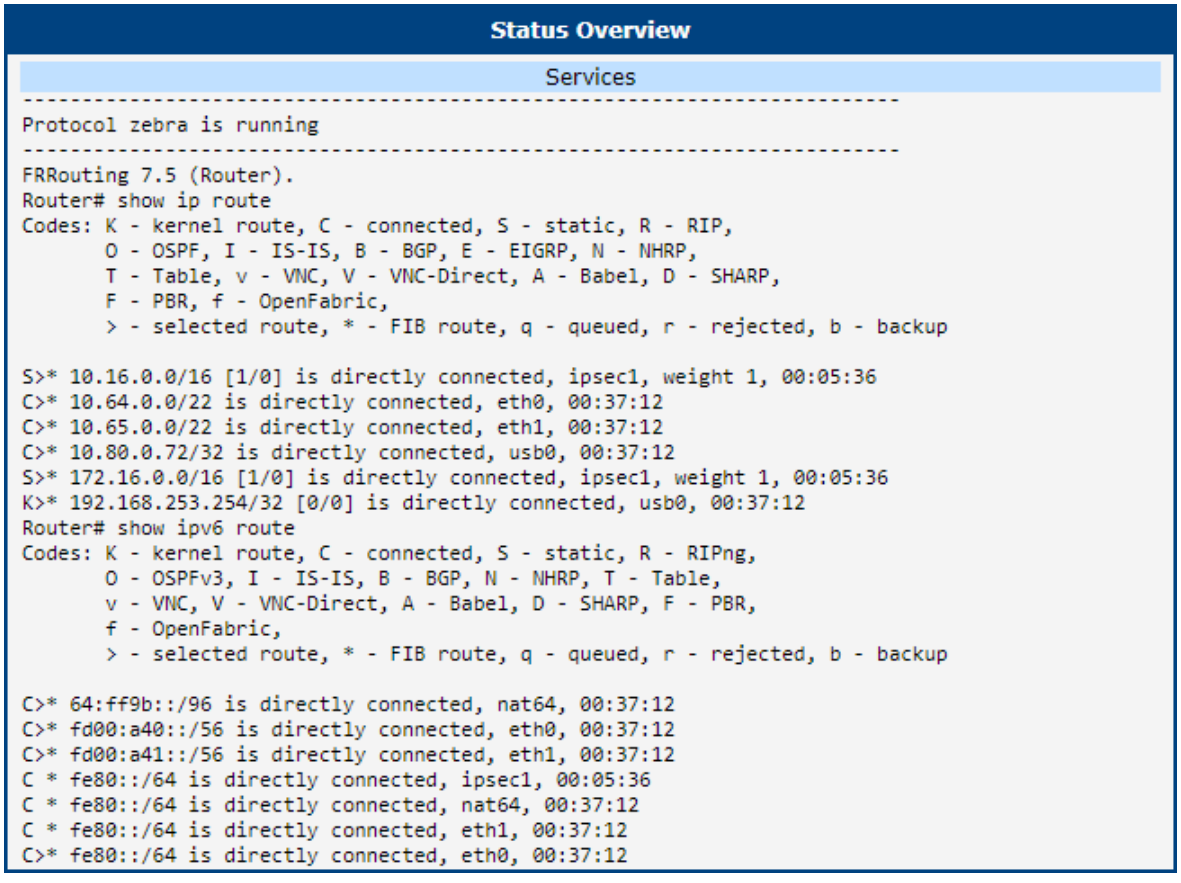



Figure 43: Server FRR Status Overview

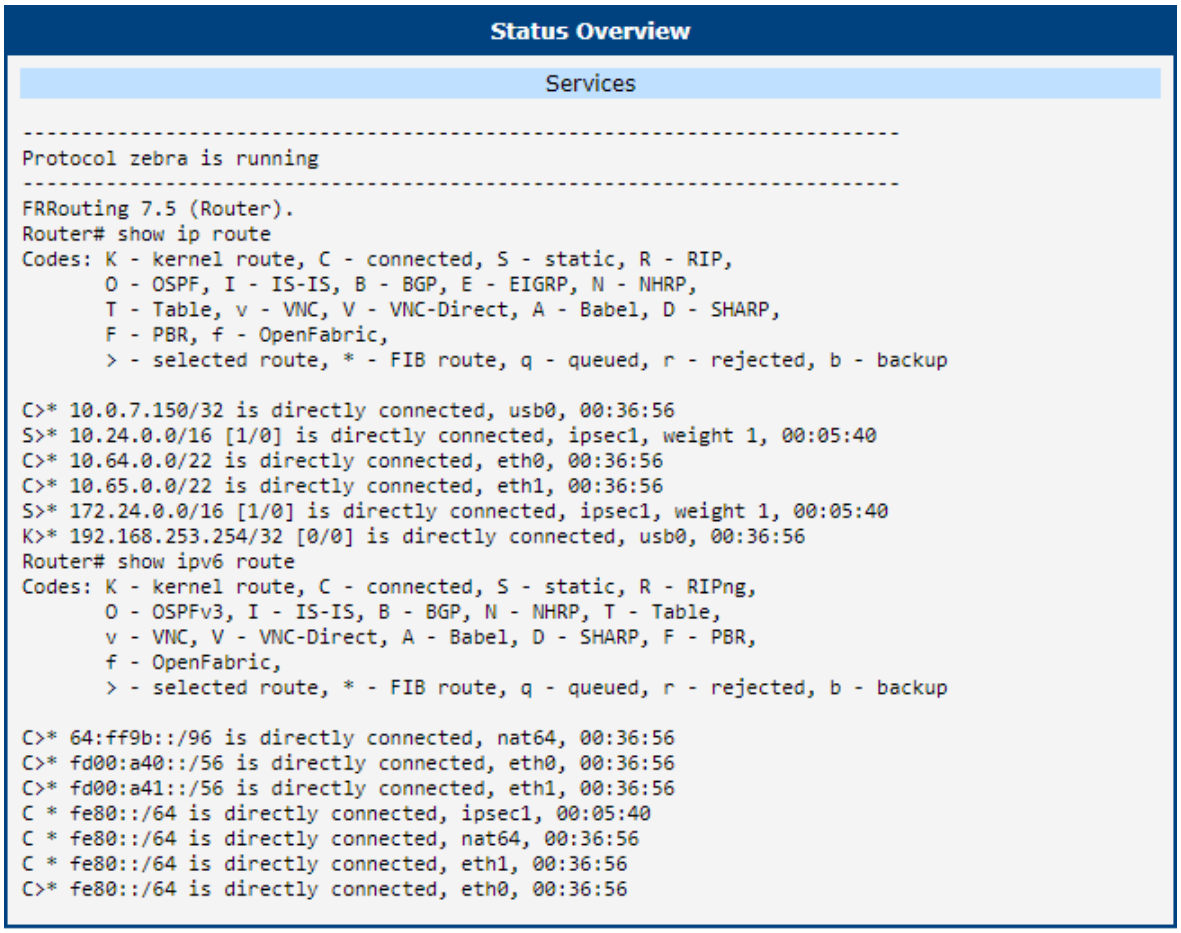



Figure 44: Client FRR Status Overview

**IPsec Status**

IPsec Tunnels Information

```
Daemon Information:

strongSwan swanctl 5.9.2
uptime: 111 minutes, since May 10 08:35:03 2021
worker threads: 16 total, 11 idle, working: 4/0/1/0
job queues: 0/0/0/0
jobs scheduled: 8
IKE_SAs: 1 total, 0 half-open
mallinfo: sbrk 733184, mmap 0, used 652480, free 80704
loaded plugins: charon nonce revocation pubkey pem openssl curl kernel-netlink socket-default vici updown xauth-generic

Connections:

ipsec2: IKEv2, reauthentication every 3060s, no rekeying
  local:  0.0.0.0
  remote: 0.0.0.0
  local pre-shared key authentication:
  remote pre-shared key authentication:
  ipsec2: TUNNEL, rekeying every 3060s
    local:  0.0.0.0/0
    remote: 0.0.0.0/0

Security Associations:

ipsec2: #8, ESTABLISHED, IKEv2, fff77f54b0bfeda5_i 84ca9e337120c74b_r*
  local  '10.64.0.64' @ 10.64.0.64[4500]
  remote '10.64.0.65' @ 10.64.0.65[4500]
  AES_CBC-128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
  established 1646s ago, reauth in 1078s
  ipsec2: #6, reqid 1, INSTALLED, TUNNEL, ESP:AES_CBC-128/HMAC_SHA1_96
    installed 1646s ago, rekeying in 1129s, expires in 1954s
    in  cf56c495 (-|0x00000002),      0 bytes,     0 packets
    out c1daa6f7 (-|0x00000002),      0 bytes,     0 packets
    local  0.0.0.0/0
    remote 0.0.0.0/0
```

Figure 45: Server IPsec Status

Route Table

| Destination | Gateway | Genmask | Flags | Metric | Ref | Use | Iface |
|---|---|---|---|---|---|---|---|
| 0.0.0.0 | 192.168.253.254 | 0.0.0.0 | UG | 0 | 0 | 0 | usb0 |
| 10.16.0.0 | 0.0.0.0 | 255.255.0.0 | U | 20 | 0 | 0 | ipsec1 |
| 10.64.0.0 | 0.0.0.0 | 255.255.252.0 | U | 0 | 0 | 0 | eth0 |
| 10.65.0.0 | 0.0.0.0 | 255.255.252.0 | U | 0 | 0 | 0 | eth1 |
| 172.16.0.0 | 0.0.0.0 | 255.255.0.0 | U | 20 | 0 | 0 | ipsec1 |
| 192.168.253.254 | 0.0.0.0 | 255.255.255.255 | UH | 0 | 0 | 0 | usb0 |

Figure 46: Server Route Table

**IPsec Status**

IPsec Tunnels Information

```
Daemon Information:

strongSwan swanctl 5.9.2
uptime: 70 minutes, since May 10 09:58:36 2021
worker threads: 16 total, 11 idle, working: 4/0/1/0
job queues: 0/0/0/0
jobs scheduled: 3
IKE_SAs: 1 total, 0 half-open
mallinfo: sbrk 745472, mmap 0, used 626296, free 119176
loaded plugins: charon nonce revocation pubkey pem openssl curl kernel-netlink socket-default vici updown xauth-generic

Connections:

ipsec2: IKEv2, reauthentication every 3060s, no rekeying
  local:  0.0.0.0
  remote: 10.64.0.64
  local pre-shared key authentication:
  remote pre-shared key authentication:
  ipsec2: TUNNEL, rekeying every 3060s
    local:  0.0.0.0/0
    remote: 0.0.0.0/0

Security Associations:

ipsec2: #2, ESTABLISHED, IKEv2, 97722e1fac5db468_i* 2ec31fcec00ae96a_r
  local  '10.64.0.65' @ 10.64.0.65[4500]
  remote '10.64.0.64' @ 10.64.0.64[4500]
  AES_CBC-128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
  established 2065s ago, reauth in 172s
  ipsec2: #2, reqid 1, INSTALLED, TUNNEL, ESP:AES_CBC-128/HMAC_SHA1_96
    installed 2066s ago, rekeying in 720s, expires in 1535s
    in  c960339f (-|0x00000002),      0 bytes,     0 packets
    out ca9dee4e (-|0x00000002),      0 bytes,     0 packets
    local  0.0.0.0/0
    remote 0.0.0.0/0
```

Figure 47: Client IPsec Status

Route Table

```
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.253.254 0.0.0.0         UG    0      0        0 usb0
10.24.0.0       0.0.0.0         255.255.0.0     U     20     0        0 ipsec1
10.64.0.0       0.0.0.0         255.255.252.0   U     0      0        0 eth0
10.65.0.0       0.0.0.0         255.255.252.0   U     0      0        0 eth1
172.24.0.0      0.0.0.0         255.255.0.0     U     20     0        0 ipsec1
192.168.253.254 0.0.0.0         255.255.255.255 UH    0      0        0 usb0
```

Figure 48: Client Route Table

### 3.4.3 Dynamic Routing

This example demonstrates the configuration of two routers, where the routes are installed dynamically by *FRR/zebra* and *FRR/BGP* applications configured in the FRR Router App, which has to be installed and configured on both routers. For more information about the FRR, free software IP routing suite, see FRRouting User Guide.

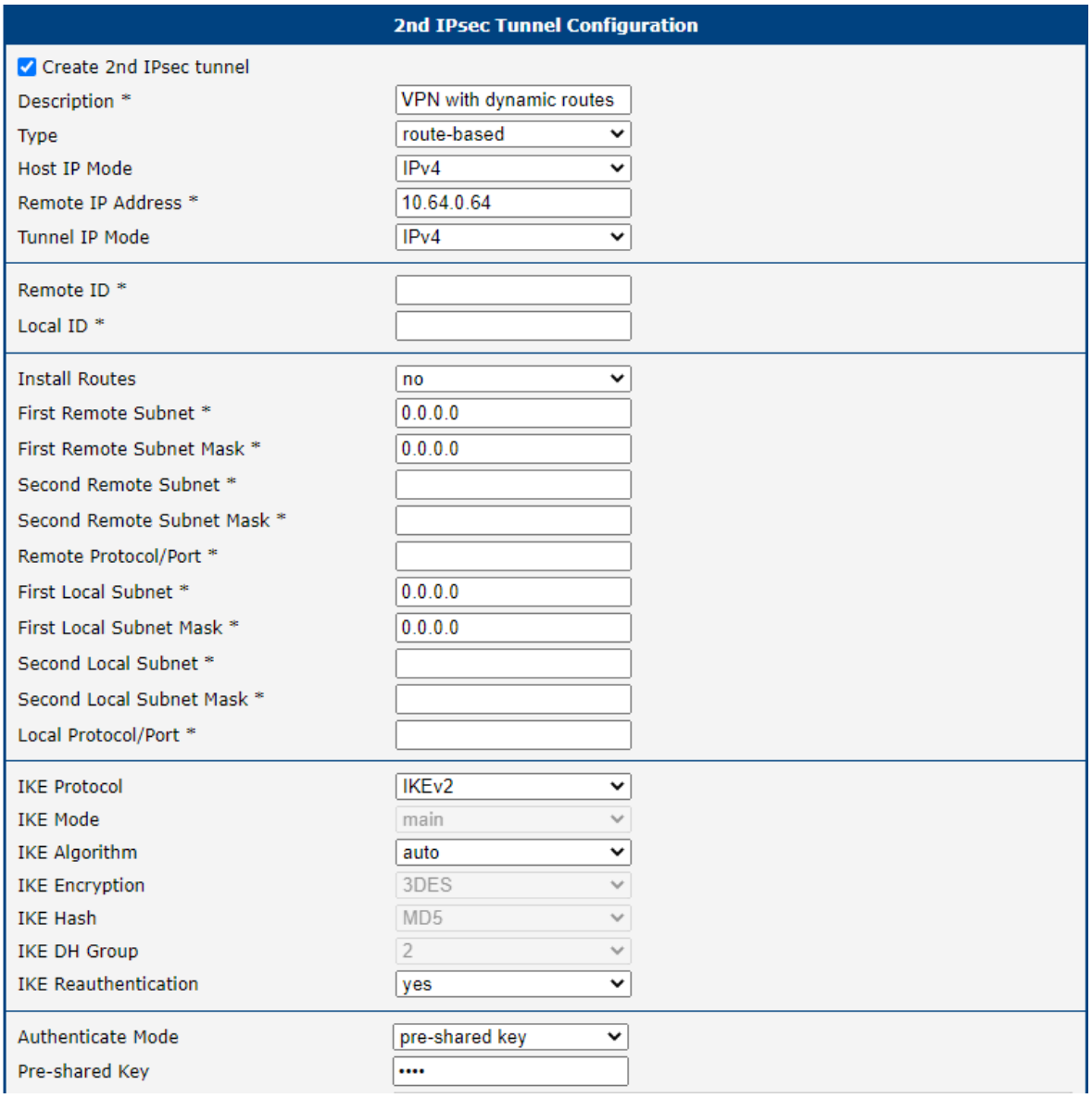



Figure 49: Client 1 Configutaion

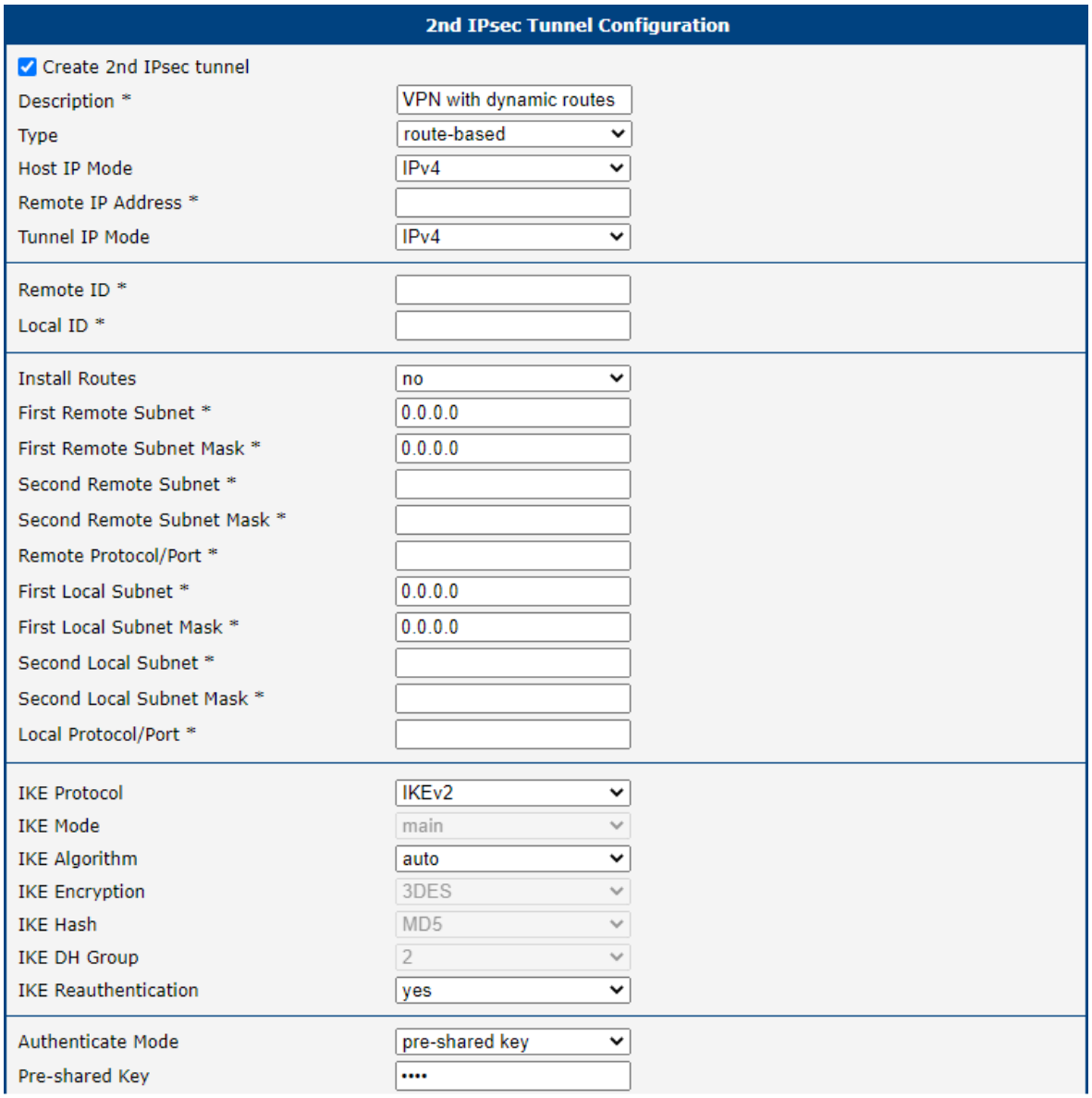



Figure 50: Client 2 Configuration

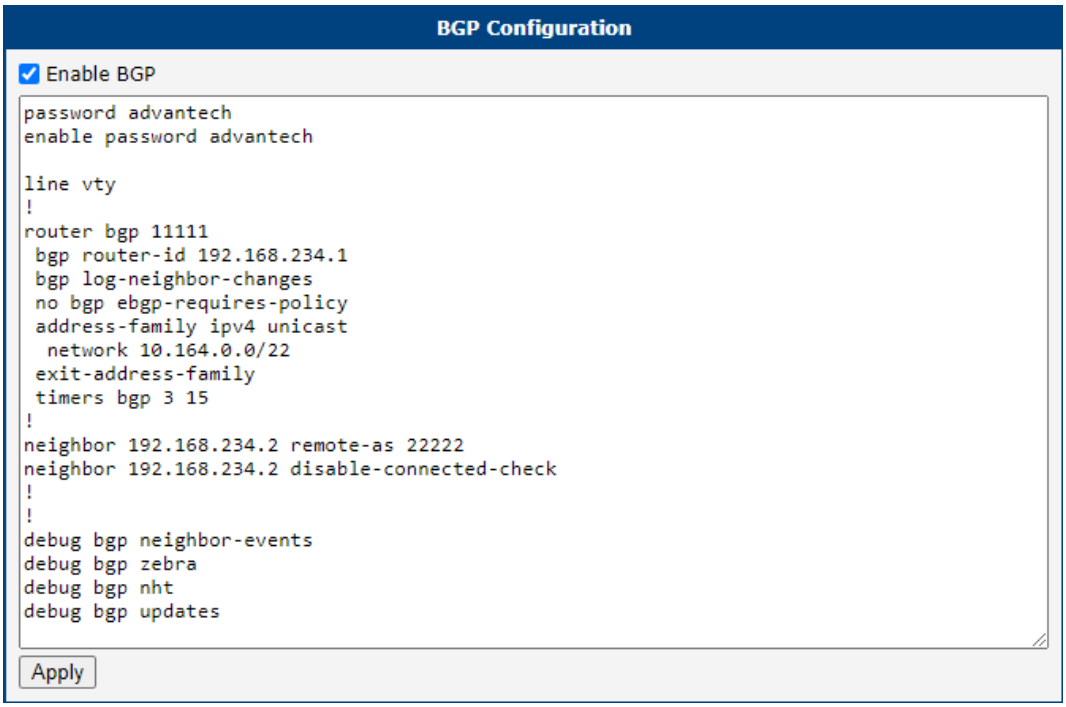



Figure 51: Client 1 FRR BGP Configuration

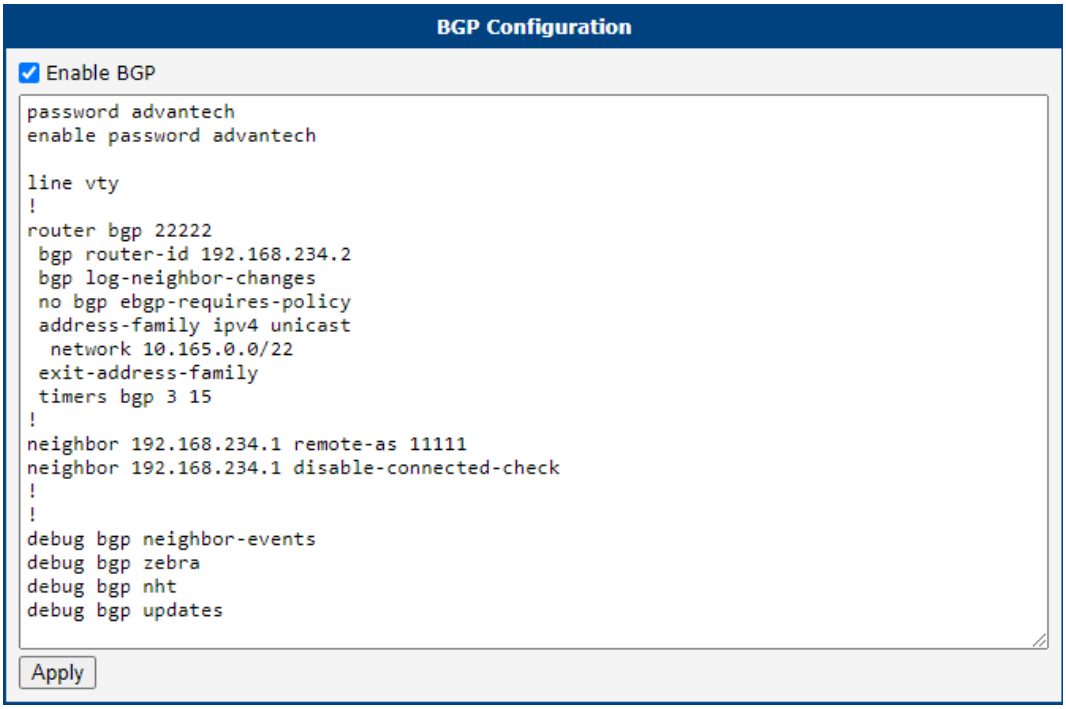



Figure 52: Client 2 FRR BGP Configuration

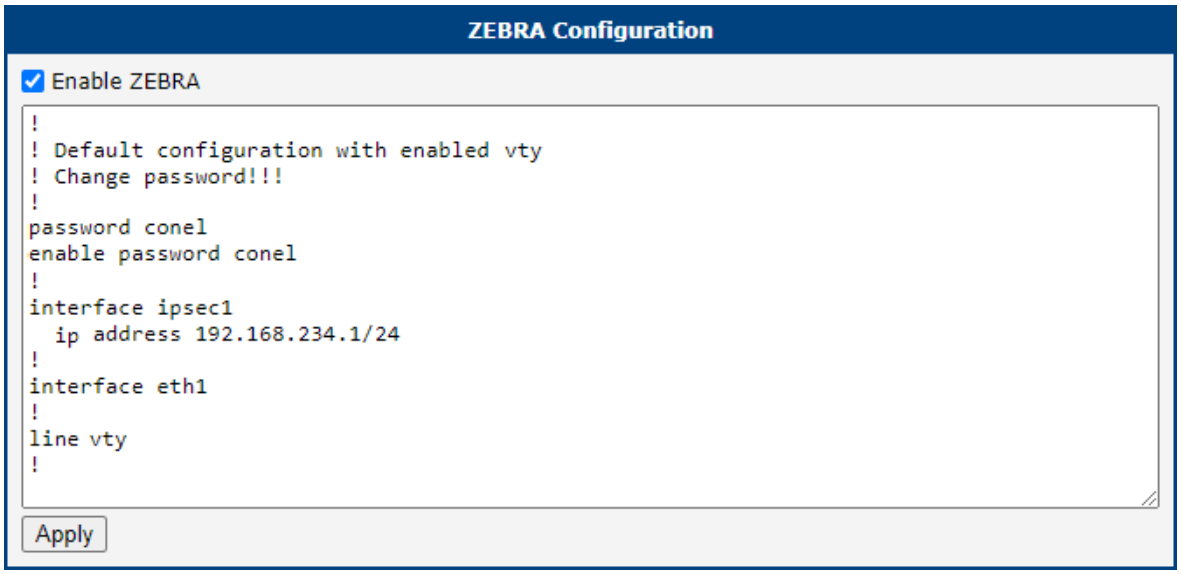



Figure 53: Client 1 FRR Zebra Configuration

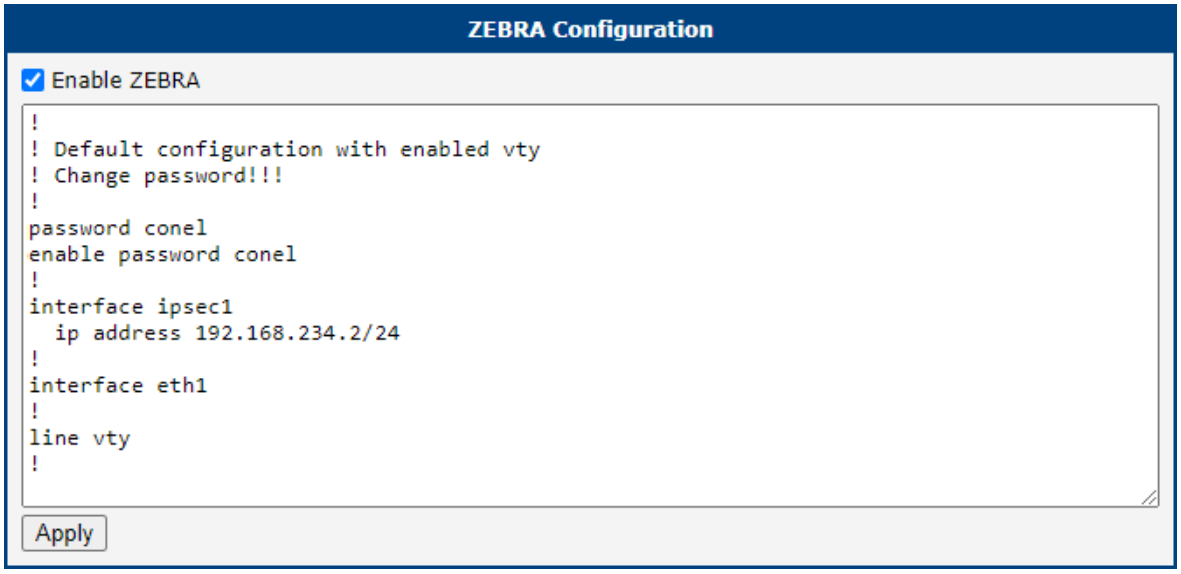



Figure 54: Client 2 FRR Zebra Configuration

**Status Overview**

Services

```
------------------------------------------------------------------------
Protocol zebra is running
------------------------------------------------------------------------
FRRouting 7.5 (Router).
Router# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

K>* 0.0.0.0/0 [0/0] via 192.168.253.254, usb0, 00:28:29
C>* 10.64.0.0/22 is directly connected, eth0, 00:28:29
C>* 10.80.0.72/32 is directly connected, usb0, 00:28:29
C>* 10.164.0.0/22 is directly connected, eth1, 00:28:29
B>* 10.165.0.0/22 [20/0] via 192.168.234.2, ipsec1, weight 1, 00:27:19
C>* 192.168.234.0/24 is directly connected, ipsec1, 00:28:29
K>* 192.168.253.254/32 [0/0] is directly connected, usb0, 00:28:29
Router# show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

C>* 64:ff9b::/96 is directly connected, nat64, 00:28:29
C>* fd00:a40::/56 is directly connected, eth0, 00:28:29
C>* fd00:a41::/56 is directly connected, eth1, 00:28:29
C * fe80::/64 is directly connected, ipsec1, 00:28:29
C * fe80::/64 is directly connected, nat64, 00:28:29
C * fe80::/64 is directly connected, eth1, 00:28:29
C>* fe80::/64 is directly connected, eth0, 00:28:29


------------------------------------------------------------------------
Protocol nhrp is stopped
------------------------------------------------------------------------
------------------------------------------------------------------------
Protocol bgp is running
------------------------------------------------------------------------
Router# show ip bgp
BGP table version is 4, local router ID is 192.168.234.1, vrf id 0
Default local pref 100, local AS 11111
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.164.0.0/22    0.0.0.0                  0         32768 i
*> 10.165.0.0/22    192.168.234.2            0             0 22222 i

Displayed  2 routes and 2 total paths
```

Figure 55: Client 1 FRR Status Overview

**Status Overview**

Services

```
------------------------------------------------------------------------
Protocol zebra is running
------------------------------------------------------------------------
FRRouting 7.5 (Router).
Router# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

K>* 0.0.0.0/0 [0/0] via 192.168.253.254, usb0, 00:28:35
C>* 10.0.7.150/32 is directly connected, usb0, 00:28:35
C>* 10.64.0.0/22 is directly connected, eth0, 00:28:35
B>* 10.164.0.0/22 [20/0] via 192.168.234.1, ipsec1, weight 1, 00:27:15
C>* 10.165.0.0/22 is directly connected, eth1, 00:28:35
C>* 192.168.234.0/24 is directly connected, ipsec1, 00:28:35
K>* 192.168.253.254/32 [0/0] is directly connected, usb0, 00:28:35
Router# show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

C>* 64:ff9b::/96 is directly connected, nat64, 00:28:35
C>* fd00:a40::/56 is directly connected, eth0, 00:28:35
C>* fd00:a41::/56 is directly connected, eth1, 00:28:35
C * fe80::/64 is directly connected, ipsec1, 00:28:35
C * fe80::/64 is directly connected, nat64, 00:28:35
C * fe80::/64 is directly connected, eth1, 00:28:35
C>* fe80::/64 is directly connected, eth0, 00:28:35


------------------------------------------------------------------------
Protocol nhrp is stopped
------------------------------------------------------------------------
------------------------------------------------------------------------
Protocol bgp is running
------------------------------------------------------------------------
Router# show ip bgp
BGP table version is 2, local router ID is 192.168.234.2, vrf id 0
Default local pref 100, local AS 22222
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.164.0.0/22    192.168.234.1            0             0 11111 i
*> 10.165.0.0/22    0.0.0.0                  0         32768 i

Displayed  2 routes and 2 total paths
```

Figure 56: Client 2 FRR Status Overview

**IPsec Status**

IPsec Tunnels Information

```
Daemon Information:

strongSwan swanctl 5.9.2
uptime: 37 minutes, since May 10 13:45:47 2021
worker threads: 16 total, 11 idle, working: 4/0/1/0
job queues: 0/0/0/0
jobs scheduled: 6
IKE_SAs: 1 total, 0 half-open
mallinfo: sbrk 688128, mmap 0, used 511256, free 176872
loaded plugins: charon nonce revocation pubkey pem openssl curl kernel-netlink socket-default vici updown xauth-generic

Connections:

ipsec2: IKEv2, reauthentication every 3060s, no rekeying
  local:  0.0.0.0
  remote: 0.0.0.0
  local pre-shared key authentication:
  remote pre-shared key authentication:
  ipsec2: TUNNEL, rekeying every 3060s
    local:  0.0.0.0/0
    remote: 0.0.0.0/0

Security Associations:

ipsec2: #3, ESTABLISHED, IKEv2, b0acaf0bd7172747_i bb23ac60586d8534_r*
  local  '10.64.0.64' @ 10.64.0.64[4500]
  remote '10.64.0.65' @ 10.64.0.65[4500]
  AES_CBC-128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
  established 110s ago, reauth in 2502s
  ipsec2: #3, reqid 1, INSTALLED, TUNNEL, ESP:AES_CBC-128/HMAC_SHA1_96
    installed 109s ago, rekeying in 2532s, expires in 3491s
    in  ccc1a077 (-|0x00000002),   5288 bytes,    84 packets
    out c76ae1f3 (-|0x00000002),   3476 bytes,    49 packets,     0s ago
    local  0.0.0.0/0
    remote 0.0.0.0/0
```

Figure 57: Client 1 IPsec Status

Route Table

| Destination | Gateway | Genmask | Flags | Metric | Ref | Use | Iface |
|---|---|---|---|---|---|---|---|
| 0.0.0.0 | 192.168.253.254 | 0.0.0.0 | UG | 0 | 0 | 0 | usb0 |
| 10.64.0.0 | 0.0.0.0 | 255.255.252.0 | U | 0 | 0 | 0 | eth0 |
| 10.164.0.0 | 0.0.0.0 | 255.255.252.0 | U | 0 | 0 | 0 | eth1 |
| 10.165.0.0 | 192.168.234.2 | 255.255.252.0 | UG | 20 | 0 | 0 | ipsec1 |
| 192.168.234.0 | 0.0.0.0 | 255.255.255.0 | U | 0 | 0 | 0 | ipsec1 |
| 192.168.253.254 | 0.0.0.0 | 255.255.255.255 | UH | 0 | 0 | 0 | usb0 |

Figure 58: Client 1 Route Table

**IPsec Status**

IPsec Tunnels Information

```
Daemon Information:

strongSwan swanctl 5.9.2
uptime: 35 minutes, since May 10 13:47:27 2021
worker threads: 16 total, 11 idle, working: 4/0/1/0
job queues: 0/0/0/0
jobs scheduled: 5
IKE_SAs: 1 total, 0 half-open
mallinfo: sbrk 540672, mmap 0, used 444800, free 95872
loaded plugins: charon nonce revocation pubkey pem openssl curl kernel-netlink socket-default vici updown xauth-generic

Connections:

ipsec2: IKEv2, reauthentication every 3060s, no rekeying
  local:  0.0.0.0
  remote: 10.64.0.64
  local pre-shared key authentication:
  remote pre-shared key authentication:
  ipsec2: TUNNEL, rekeying every 3060s
    local:  0.0.0.0/0
    remote: 0.0.0.0/0

Security Associations:

ipsec2: #2, ESTABLISHED, IKEv2, b0acaf0bd7172747_i* bb23ac60586d8534_r
  local  '10.64.0.65' @ 10.64.0.65[4500]
  remote '10.64.0.64' @ 10.64.0.64[4500]
  AES_CBC-128/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_3072
  established 96s ago, reauth in 1976s
  ipsec2: #2, reqid 1, INSTALLED, TUNNEL, ESP:AES_CBC-128/HMAC_SHA1_96
    installed 97s ago, rekeying in 2564s, expires in 3504s
    in  c76ae1f3 (-|0x00000002),   3061 bytes,    43 packets,    95s ago
    out ccc1a077 (-|0x00000002),   4673 bytes,    74 packets,     2s ago
    local  0.0.0.0/0
    remote 0.0.0.0/0
```

Figure 59: Client 2 IPsec Status

Route Table

```
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.253.254 0.0.0.0         UG    0      0        0 usb0
10.64.0.0       0.0.0.0         255.255.252.0   U     0      0        0 eth0
10.164.0.0      192.168.234.1   255.255.252.0   UG    20     0        0 ipsec1
10.165.0.0      0.0.0.0         255.255.252.0   U     0      0        0 eth1
192.168.234.0   0.0.0.0         255.255.255.0   U     0      0        0 ipsec1
192.168.253.254 0.0.0.0         255.255.255.255 UH    0      0        0 usb0
```

Figure 60: Client 2 Route Table

## 3.5 Known Issues

### 3.5.1 Several Subnets in one CHILD_SA

If you use IKEv2, you can if the peers support it, some do not (e.g. devices by Checkpoint, Cisco and Fortinet, see interoperability[1] for details).

If you are using strongSwan with different IPsec solution, please consult https://wiki.strongswan.org/projects/strongswan/wiki/Interoperability in case of any problems before contacting our technical support.

[1] https://wiki.strongswan.org/projects/strongswan/wiki/Interoperability

# 4. Related Literature


**[1]** Advantech Czech: **v2 Routers Configuration Manual** (MAN-0021-EN)
**[2]** Advantech Czech: **SmartFlex Configuration Manual** (MAN-0023-EN)
**[3]** Advantech Czech: **SmartMotion Configuration Manual** (MAN-0024-EN)
**[4]** Advantech Czech: **SmartStart Configuration Manual** (MAN-0022-EN)
**[5]** Advantech Czech: **ICR-3200 Configuration Manual** (MAN-0042-EN)


Product-related documents can be obtained on *Engineering Portal* at icr.advantech.cz address.

# Appendix A: opensssl.conf

```
#
# OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#

# Note that you can include other files from the main configuration
# file using the .include directive.
#.include filename

# This definition stops the following lines choking if HOME isn't
# defined.
HOME = .
RANDFILE = $ENV::HOME/.rnd

# Extra OBJECT IDENTIFIER info:
#oid_file = $ENV::HOME/.oid
oid_section = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ new_oids ]

# We can add new OIDs in here for use by 'ca', 'req' and 'ts'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6

# Policies used by the TSA examples.
tsa_policy1 = 1.2.3.4.1
tsa_policy2 = 1.2.3.4.5.6
tsa_policy3 = 1.2.3.4.5.7

####################################################################
[ ca ]
```

```
default_ca = CA_default # The default ca section

####################################################################
[ CA_default ]

dir = ./ # Where everything is kept
certs = $dir # Where the issued certs are kept
crl_dir = $dir # Where the issued crl are kept
database = $dir/index.txt # database index file.
#unique_subject = no # Set to 'no' to allow creation of
# several certs with same subject.
new_certs_dir = $dir # default place for new certs.

certificate = $dir/ca.crt  # The CA certificate
serial = $dir/serial  # The current serial number
crlnumber = $dir/crlnumber # the current crl number
# must be commented out to leave a V1 CRL
crl = $dir/crl.pem  # The current CRL
private_key = $dir/private/ca.key# The private key
RANDFILE = $dir/private/.rand # private random number file

x509_extensions = usr_cert # The extensions to add to the cert

# Comment out the following two lines for the "traditional"
# (and highly broken) format.
name_opt  = ca_default # Subject Name options
cert_opt  = ca_default # Certificate field options

# Extension copying option: use with caution.
# copy_extensions = copy

# Extensions to add to a CRL. Note: Netscape communicator chokes on V2 CRLs
# so this is commented out by default to leave a V1 CRL.
# crlnumber must also be commented out to leave a V1 CRL.
# crl_extensions = crl_ext

default_days = 365 # how long to certify for
default_crl_days= 30 # how long before next CRL
default_md = default # use public key default MD
preserve = no # keep passed DN ordering

# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
```

```
policy = policy_match

# For the CA policy
[ policy_match ]
countryName = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

####################################################################
[ req ]
default_bits = 2048
default_keyfile  = privkey.pem
distinguished_name = req_distinguished_name
attributes = req_attributes
x509_extensions = v3_ca # The extensions to add to the self signed cert

# Passwords for private keys if not present they will be prompted for
# input_password = secret
# output_password = secret

# This sets a mask for permitted string types. There are several options.
# default: PrintableString, T61String, BMPString.
# pkix  : PrintableString, BMPString (PKIX recommendation before 2004)
# utf8only: only UTF8Strings (PKIX recommendation after 2004).
# nombstr : PrintableString, T61String (no BMPStrings or UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: ancient versions of Netscape crash on BMPStrings or UTF8Strings.
string_mask = utf8only
```

```
# req_extensions = v3_req # The extensions to add to a certificate request

[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = AU
countryName_min = 2
countryName_max = 2

stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = Some-State

localityName = Locality Name (eg, city)

0.organizationName = Organization Name (eg, company)
0.organizationName_default = Internet Widgits Pty Ltd

# we can do this but it is not needed normally :-)
#1.organizationName = Second Organization Name (eg, company)
#1.organizationName_default = World Wide Web Pty Ltd

organizationalUnitName = Organizational Unit Name (eg, section)
#organizationalUnitName_default =

commonName = Common Name (e.g. server FQDN or YOUR name)
commonName_max = 64

emailAddress = Email Address
emailAddress_max = 64

# SET-ex3 = SET extension number 3

[ req_attributes ]
challengePassword = A challenge password
challengePassword_min = 4
challengePassword_max = 20

unstructuredName = An optional company name

[ usr_cert ]

# These extensions are added when 'ca' signs a request.

# This goes against PKIX guidelines but some CAs do it and some software
# requires this to avoid interpreting an end user certificate as a CA.
```

```
basicConstraints=CA:FALSE

# Here are some examples of the usage of nsCertType. If it is omitted
# the certificate can be used for anything *except* object signing.

# This is OK for an SSL server.
# nsCertType = server

# For an object signing certificate this would be used.
# nsCertType = objsign

# For normal client use this is typical
# nsCertType = client, email

# and for everything including object signing:
# nsCertType = client, email, objsign

# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment

# This will be displayed in Netscape's comment listbox.
nsComment = "OpenSSL Generated Certificate"

# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer

# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
# subjectAltName=email:copy
# An alternative to produce certificates that aren't
# deprecated according to PKIX.
# subjectAltName=email:move

# Copy subject details
# issuerAltName=issuer:copy

#nsCaRevocationUrl = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName
```

```
# This is required for TSA certificates.
# extendedKeyUsage = critical,timeStamping

[ v3_req ]

# Extensions to add to a certificate request

basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
subjectAltName = @alt_names

[alt_names]
IP =  <IP address>

[ v3_ca ]


# Extensions for a typical CA


# PKIX recommendation.

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid:always,issuer

basicConstraints = critical,CA:true

# Key usage: this is typical for a CA certificate. However since it will
# prevent it being used as an test self-signed certificate it is best
# left out by default.
# keyUsage = cRLSign, keyCertSign

# Some might want this also
# nsCertType = sslCA, emailCA

# Include email address in subject alt name: another PKIX recommendation
# subjectAltName=email:copy
# Copy issuer details
# issuerAltName=issuer:copy

# DER hex encoding of an extension: beware experts only!
# obj=DER:02:03
```

```
# Where 'obj' is a standard or added object
# You can even override a supported extension:
# basicConstraints= critical, DER:30:03:01:01:FF

[ crl_ext ]

# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always

[ proxy_cert_ext ]
# These extensions should be added when creating a proxy certificate

# This goes against PKIX guidelines but some CAs do it and some software
# requires this to avoid interpreting an end user certificate as a CA.

basicConstraints=CA:FALSE

# Here are some examples of the usage of nsCertType. If it is omitted
# the certificate can be used for anything *except* object signing.

# This is OK for an SSL server.
# nsCertType = server

# For an object signing certificate this would be used.
# nsCertType = objsign

# For normal client use this is typical
# nsCertType = client, email

# and for everything including object signing:
# nsCertType = client, email, objsign

# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment

# This will be displayed in Netscape's comment listbox.
nsComment = "OpenSSL Generated Certificate"

# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer
```

```
# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
# subjectAltName=email:copy
# An alternative to produce certificates that aren't
# deprecated according to PKIX.
# subjectAltName=email:move

# Copy subject details
# issuerAltName=issuer:copy

#nsCaRevocationUrl = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName

# This really needs to be in place for it to be a proxy certificate.
proxyCertInfo=critical,language:id-ppl-anyLanguage,pathlen:3,policy:foo

####################################################################
[ tsa ]

default_tsa = tsa_config1 # the default TSA section

[ tsa_config1 ]

# These are used by the TSA reply generation only.
dir = ./demoCA # TSA root directory
serial = $dir/tsaserial # The current serial number (mandatory)
crypto_device = builtin # OpenSSL engine to use for signing
signer_cert = $dir/tsacert.pem  # The TSA signing certificate
# (optional)
certs = $dir/cacert.pem # Certificate chain to include in reply
# (optional)
signer_key = $dir/private/tsakey.pem # The TSA private key (optional)
signer_digest  = sha256 # Signing digest to use. (Optional)
default_policy = tsa_policy1 # Policy if request did not specify it
# (optional)
other_policies = tsa_policy2, tsa_policy3 # acceptable policies (optional)
digests     = sha1, sha256, sha384, sha512  # Acceptable message digests (mandatory)
accuracy = secs:1, millisecs:500, microsecs:100 # (optional)
clock_precision_digits  = 0 # number of digits after dot. (optional)
```

```
ordering = yes # Is ordering defined for timestamps?
# (optional, default: no)
tsa_name = yes # Must the TSA name be included in the reply?
# (optional, default: no)
ess_cert_id_chain = no # Must the ESS cert id chain be included?
# (optional, default: no)
ess_cert_id_alg = sha1 # algorithm to compute certificate
# identifier (optional, default: sha1)
```

# Appendix B: server_req.conf

```
####################################################################
[ ca ]
default_ca = CA_default # The default ca section

####################################################################
[ CA_default ]

dir = ./ # Where everything is kept
certs = $dir # Where the issued certs are kept
crl_dir = $dir # Where the issued crl are kept
database = $dir/index.txt # database index file.
new_certs_dir = $dir # default place for new certs.

certificate = $dir/ca.crt  # The CA certificate
serial = $dir/serial  # The current serial number
crlnumber = $dir/crlnumber # the current crl number
# must be commented out to leave a V1 CRL
crl = $dir/crl.pem  # The current CRL
private_key = $dir/private/ca.key# The private key
RANDFILE = $dir/private/.rand # private random number file
name_opt  = ca_default # Subject Name options
cert_opt  = ca_default # Certificate field options
default_days = 365 # how long to certify for
default_crl_days= 30 # how long before next CRL
default_md = default # use public key default MD
preserve = no # keep passed DN ordering
policy = policy_match
# For the CA policy
[ policy_match ]
countryName = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

[req]
distinguished_name = server
req_extensions = v3_req
prompt = no
```

```
[server]
C = CZ
ST = Czechia
L = Usti
O = Advantech
OU = Advantech CZ
CN = server@cisco

[v3_req]
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
IP = 85.207.4.118
DNS = server.cisco
email = server@cisco
```

# Appendix C: client_req.conf

```
####################################################################
[ ca ]
default_ca = CA_default # The default ca section

####################################################################
[ CA_default ]

dir = ./ # Where everything is kept
certs = $dir # Where the issued certs are kept
crl_dir = $dir # Where the issued crl are kept
database = $dir/index.txt # database index file.
new_certs_dir = $dir # default place for new certs.
certificate = $dir/ca.crt  # The CA certificate
serial = $dir/serial  # The current serial number
crlnumber = $dir/crlnumber # the current crl number
# must be commented out to leave a V1 CRL
crl = $dir/crl.pem  # The current CRL
private_key = $dir/private/ca.key# The private key
RANDFILE = $dir/private/.rand # private random number file
name_opt  = ca_default # Subject Name options
cert_opt  = ca_default # Certificate field options
default_days = 365 # how long to certify for
default_crl_days= 30 # how long before next CRL
default_md = default # use public key default MD
preserve = no # keep passed DN ordering
policy = policy_match
# For the CA policy
[ policy_match ]
countryName = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

[req]
distinguished_name = client
req_extensions = v3_req
prompt = no
[client]
```

```
C = CZ
ST = Czechia
L = Usti
O = Advantech
OU = Advantech CZ
CN = client@router

[v3_req]
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
IP = 62.141.23.118
DNS = client.router
email = client@router
```