Vsebina

[**Prepare VirtualBox images** 2](#_Toc503253583)

[**Onemogočanje IPv6** 2](#_Toc503253584)

[**Konfiguracija APACHE strežnika** 2](#_Toc503253585)

[**Spreminjanje host nastavitev** 2](#_Toc503253586)

[**IPTABLES** 3](#_Toc503253587)

[STATELESS 3](#_Toc503253588)

[STATEFULL 3](#_Toc503253589)

[FORWARD 3](#_Toc503253590)

[**SSH strežnik** 4](#_Toc503253591)

[Napredne nastavitve 4](#_Toc503253592)

[Generiranje ključev za ssh server 4](#_Toc503253593)

[Preverjanje public ključev: 4](#_Toc503253594)

[Odstranjevanje neveljavnih ssh public ključev: 4](#_Toc503253595)

[Generiranje ključev za klienta 4](#_Toc503253596)

[Prijava na ssh server s public ključem 4](#_Toc503253597)

[Reverse SSH Tunneling 5](#_Toc503253598)

[**Kloniranje slike** 5](#_Toc503253599)

[**Nastaviteve za router virtualko** 6](#_Toc503253600)

[ROUTER 6](#_Toc503253601)

[Posodobimo interface omrežja 7](#_Toc503253602)

[CLIENT 7](#_Toc503253603)

[**IPTABLES potek dela** 8](#_Toc503253604)

[**VPN** 8](#_Toc503253605)

[Checkpoint 8](#_Toc503253606)

[VPN IPsec tunnel 9](#_Toc503253607)

[hq\_router 9](#_Toc503253608)

[branch\_router 9](#_Toc503253609)

[FreeRADIUS 10](#_Toc503253610)

[radius1 10](#_Toc503253611)

[Debuging 11](#_Toc503253612)

[HTTP Basic authentication with Apache and FreeRADIUS 11](#_Toc503253613)

[Roaming 12](#_Toc503253614)

# **Prepare VirtualBox images**

Start VirtualBox and change the MAC address of the machine network interface: Settings > Network > Adapter 1 > Advanced > MAC Address > Generates a new random MAC address.

Next, change the Network adapter:

* If you are on a home network or on university ethernet network: Settings > Network > Adapter 1 > Attached to: Bridged
* If you are on Eduroam, you cannot use bridged networking. In this case either connect your laptop to a university ethernet network and set the networking to bridged as described above, or create a new NAT network (File > Preferences > Networks > NAT Networks > Add new NAT network, leave all settings to defaults) and set the Adapter 1 to use the NAT network that you have created previously.

# **Onemogočanje IPv6**

Dodaj v:

sudo nano /etc/sysctl.conf

net.ipv6.conf.all.disable\_ipv6 = 1

net.ipv6.conf.default.disable\_ipv6 = 1

net.ipv6.conf.lo.disable\_ipv6 = 1

Potrdi spremembe z ukazom:

sudo sysctl -p

**You should run this command each time you start up the image; IPv6 turns on by default at start.**

# **Konfiguracija APACHE strežnika**

Install packages that will be used for testing firewall rules

* sudo apt-get install apache2 curl
* Generate default digital certificates for Apache2: sudo make-ssl-cert generate-default-snakeoil --force-overwrite
* Enable Apache2 SSL Site: sudo a2ensite default-ssl.conf
* Enable Apache2 TLS/SSL module sudo a2enmod ssl
* Restart Apache server sudo service apache2 restart
* Check if Apache2 works by running the web browser and opening both http://localhost and https://localhost. Alternatively, test with curl.

# **Spreminjanje host nastavitev**

sudo nano /etc/hosts

Dodamo: 127.0.1.1 ssh-server

Uveljavimo sprmemebe: sudo hostnamectl set-hostname ssh-server

# **IPTABLES**

(<https://github.com/lem-course/isp-iptables>) (<https://www.digitalocean.com/community/tutorials/how-to-list-and-delete-iptables-firewall-rules>) Change downloaded file's execution permissions: **chmod +x iptables1.sh**

Izpis aktivnih pravil: sudo iptables -S

Za določeno verigo: sudo iptables -S TCP

## STATELESS

(<https://ucilnica.fri.uni-lj.si/mod/page/view.php?id=8650>)

# Disable INPUT

iptables --policy INPUT DROP

### Allow all trafic on localhost

iptables -A INPUT -i lo -j ACCEPT

iptables -A OUTPUT -o lo -j ACCEPT

### Allow DNS lookups as a client ### (1) Allow access to a particular DNS server. ### The IP address of the DNS server is given in variable NAMESERVER

iptables -A OUTPUT -o $INET\_IFACE -p udp -s $IPADDR --sport $UNPRIVPORTS -d $NAMESERVER --dport 53 -j ACCEPT

iptables -A INPUT -i $INET\_IFACE -p udp -s $NAMESERVER --sport 53 -d $IPADDR --dport $UNPRIVPORTS -j ACCEPT

### SSH

### (2) Allow outgoing SSH connections

iptables -A OUTPUT -p tcp --dport 22 -j ACCEPT

iptables -A INPUT -p tcp ! --syn --sport 22 -j ACCEPT

### (3) Allow incoming SSH connections

iptables -A INPUT -p tcp --dport 22 -j ACCEPT

iptables -A OUTPUT -p tcp --sport 22 -j ACCEPT

## STATEFULL

(https://ucilnica.fri.uni-lj.si/mod/page/view.php?id=8751)

# (1) Allow all incoming packets that belong to ESTABLISHED or RELATED connections.

iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# (3) Allow outgoing DNS requests to the DNS server in variable NAMESERVER

iptables -A OUTPUT -p udp -d $NAMESERVER --dport 53 -m state --state NEW -j ACCEPT

# (4) TODO: Allow outgoing SSH connections to remote SSH servers

iptables -A OUTPUT -p tcp --dport 22 -m state --state NEW -j ACCEPT

# (5) TODO: Allow incomming connections to local SSH server

iptables -A INPUT -p tcp --dport 22 -m state --state NEW -j ACCEPT

## FORWARD

# Do NAT for internet-bound traffic

iptables -t nat -A POSTROUTING -o $INET\_IFACE -j MASQUERADE

# (14) Forward pings

iptables -A FORWARD -p icmp --icmp-type echo-request -m state --state NEW -j ACCEPT

# (15) Forward DNS requests from subnets to Internet and permit in corresponding responses

iptables -A FORWARD -o $INET\_IFACE -p udp -m multiport --ports 53 -m state --state NEW -j ACCEPT

# **SSH strežnik**

(https://ucilnica.fri.uni-lj.si/mod/page/view.php?id=8957)

Install packages that will be used for testing firewall rules

* sudo apt-get install openssh-server
* Check if SSH server works by running ssh localhost, answer with yes and provide password isp. Press ctrl+d to exit.

## Napredne nastavitve

### Generiranje ključev za ssh server

sudo ssh-keygen -t ecdsa -f /etc/ssh/ssh\_host\_ecdsa\_key

sudo ssh-keygen -t rsa -f /etc/ssh/ssh\_host\_rsa\_key

sudo ssh-keygen -t dsa -f /etc/ssh/ssh\_host\_dsa\_key

sudo ssh-keygen -t ed25519 -f /etc/ssh/ssh\_host\_ed25519\_key

Izbira ključa:

 Name the keys according to HostKey directive in sudo nano /etc/ssh/sshd\_config file.

### Preverjanje public ključev:

* For ECDSA key: ssh-keygen -lf /etc/ssh/ssh\_host\_ecdsa\_key.pub
* For RSA key: ssh-keygen -lf /etc/ssh/ssh\_host\_rsa\_key.pub
* For DSA key: ssh-keygen -lf /etc/ssh/ssh\_host\_dsa\_key.pub

### Odstranjevanje neveljavnih ssh public ključev:

Iz: sudo nano ~/.ssh/known\_hosts

### Generiranje ključev za klienta

Ključi za uporabnika se shranijo v mapo: ~/.ssh

* ssh-keygen -t rsa
* ssh-keygen -t dsa
* ssh-keygen -t ecdsa

### Prijava na ssh server s public ključem

ssh -i ~/.ssh/id\_rsa isp@$SERVER

* To enable public key authentication, you have to (1) copy your public key to the remote computer and then (2) enable and link it to specific account. Both actions can be done with ssh-copy-id which copies public key to the chosen account and adds public key to authorized keys list. Simply run: ssh-copy-id isp@$SERVER.
* Once the key has been copied and added to the authorized\_keys list, try connecting and authenticating using only public keys: ssh $SERVER. You should now login to server without providing password. (We can even omit the username, since the username on the server and on the client are the same.)
* Finally, let's disable password-based login attempts and always require client authentication with public keys. On the ssh-server, open file /etc/ssh/sshd\_config and add command PasswordAuthentication no. Save the file and restart the SSH server with sudo service ssh restart.

Because we have already copied our public key to the server, our client will by default try to authenticate itself with the public key. So we have to explicitly state that we want to authenticate with the username/password pair, if we want to test the most recent change. Run the following on ssh-client: ssh -o PreferredAuthentications=password -o PubkeyAuthentication=no $SERVER. If you configured the sever correctly, the connection attempt should be rejected.

## Reverse SSH Tunneling

A reverse SSH tunnel is similar to a normal SSH tunnel, the difference is in the agent that initiates the tunnel. In a reverse SSH tunnel, the machine that provides the service is also the machine that sets up the tunnel. (Contrary to the local port-forwarding where the tunnel was set up by the machine that consumed the provided service.)

[Onemogoči IPv6.](#_Onemogočanje_IPv6)

Next, you may reuse the iptables script from the previous week's lab session. Modify the script to contain the following entries:

iptables -A INPUT -i lo -j ACCEPT

iptables -A OUTPUT -o lo -j ACCEPT

iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A OUTPUT -p tcp --dport 22 -m state --state NEW -j ACCEPT

Additionally, remove the Apache access control that we added in the previous assignment by commenting out (or deleting) the following lines in /etc/apache2/sites-available/000-default.conf:

<Directory /var/www/html>

Require ip 127.0.0.1/8

</Directory>

Remember to reload the configuration once you have changed the file: sudo service apache2 reload.

At this point, you should be able to curl localhost on the ssh-server machine, while a curl $SERVER run on the ssh-client should fail. Moreover, you should also be unable to ssh into server from ssh-client; the firewall should block both HTTP and SSH access from the outside.

Now, the ssh-server machine is allowed to connect onto ssh-client and establish a reverse tunnel that will allow ssh-client to access the Apache pages on ssh-server. On the ssh-server, run the following:

ssh -R 127.0.0.1:8080:127.0.0.1:80 -N isp@$CLIENT

With the reverse tunnel set up, you should be able to curl localhost:8080 on the ssh-client and access the contents of the Apache server pages on the ssh-server machine.

# SCP

Ukaz za kopiranje prek scp-ja

scp file isp@ip:/home/isp

cp file /location/

mv file /location/

# Izdelava lastne certifikatne agencije

First, [**generate**](https://wiki.strongswan.org/projects/strongswan/wiki/IpsecPKIGen) a private key, the default generates a 2048 bit RSA key (if this command blocks, refer to [**this note about hosts with low entropy**](https://wiki.strongswan.org/projects/strongswan/wiki/IpsecPKIGen#Problems-on-Hosts-with-Low-Entropy)):

ipsec pki --gen > caKey.der

For a real-world setup, make sure to keep this key absolutely private.

Now [**self-sign**](https://wiki.strongswan.org/projects/strongswan/wiki/IpsecPKISelf) a CA certificate using the generated key:

ipsec pki --self --in caKey.der --dn "C=CH, O=strongSwan, CN=strongSwan CA" --ca > caCert.der

For **each** peer, i.e. for all VPN clients and VPN gateways in your network, generate an individual private key and [**issue**](https://wiki.strongswan.org/projects/strongswan/wiki/IpsecPKIIssue) a matching certificate using your new CA:

ipsec pki --gen > branchKey.der

For instance, when creating the certificate for the branch router (whose identity is @branch), you can use the following command: ipsec pki --pub --in branchKey.der | ipsec pki --issue --cacert caCert.der --cakey caKey.der --dn "C=SL, O=FRI-UL, CN=branch" --san @branch > branchCert.der. (This command assumes that you have previously created the private key in file branchKey.der and that the CA's certificate and the corresponding private key are in files caCert.der and caKey.der.)

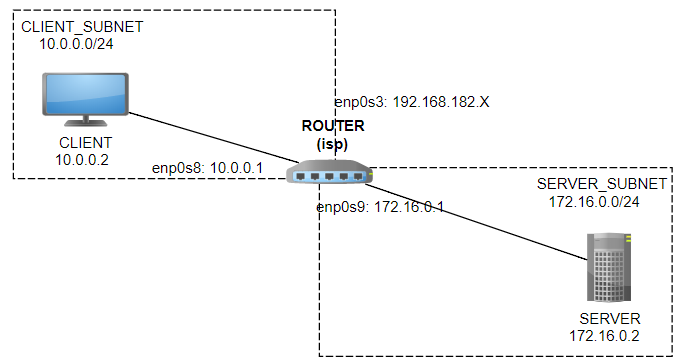
Next, copy the client's certificate and private key to the appropriate machines. Additionally, you will also have to copy the CA's certificate to both machines. Place files in the appropriate subfolders within /etc/ipsec.d/.

# **Kloniranje slike**

* Right click on the (power off) image in VirtualBox and select Clone (Ctrl+O);
* Choose Expert mode:
  + Give a name to the cloned image, for instance isp-2;
  + Select Linked clone;
  + Select the option to Reinitialize the MAC address of all network cards;
* Confirm by clicking Clone.

# **Nastaviteve za router virtualko**

(https://ucilnica.fri.uni-lj.si/mod/page/view.php?id=8751)



Create two additional virtual images by cloning the existing image. Name the first clone client and the second server. (We'll assume that the image you have been using so far is named isp.) You may create linked clones. Do not forget to generate new MAC addresses for the newly created images.

Configure the isp machine to use two additional network interface cards (NICs): Machine > Settings > Network > Adapter 2. Tick Enable Network Adapter, select Internal Network and put client\_subnet in Name field. Then switch to tab Adapter 3 and repeat the process, but this time name the Internal Network card server\_subnet.

The first netwok adapter on isp can be set to NAT, Bridged Adapter or NAT Network. It does not really matter which, as long as it provides Internet connectivity. Confirm the changes by clicking OK.

The client and the server machine will have only a single NIC. First, configure the NIC on the client by setting its network interface to Internal Network and selecting client\_subnet as name.

Finally, configure the NIC on the server by setting its network interface to Internal Network and selecting server\_subnet as name.

## ROUTER

Start the isp machine. Notice that the machine has three NIC cards: run ifconfig and observe enp0s3, enp0s8 and enp0s9. You'll see that only enp0s3 managed to obtain an IP address, while enp0s{8,9} did not. The reason is that the subnets which enp0s8 and enp0s9 connect to, do not have DCHP servers. This means that we'll have to set up IPs manually.

Let's assign IPs to isp machine for enp0s8 and enp0s9. Since the client\_subnet uses addresses from 10.0.0.0/24 and server\_subnet addresses from 172.16.0.0/24, we'll use the first available address that come to mind: 10.0.0.1 for enp0s8 and 172.16.0.1 for enp0s9.

### Posodobimo interface omrežja

sudo nano /etc/network/interfaces

auto enp0s8

iface enp0s8 inet static

address 10.0.0.1

netmask 255.255.255.0

auto enp0s9

iface enp0s9 inet static

address 172.16.0.1

netmask 255.255.255.0

To apply these changes:

* Restart the network manager with sudo service network-manager restart, and
* bring up those two interfaces: sudo ifup enp0s8 and sudo ifup enp0s9.
* Confirm that the addresses have been successfully set by running ifconfig.
* **Next, enable routing for IPv4** so that the isp will actually behave as a proper router: echo 1 | sudo tee /proc/sys/net/ipv4/ip\_forward.

## CLIENT

Configuring the client and the server is simpler. We have to do three things: assign them IP addresses (10.0.0.2), DNS servers (8.8.8.8) and instruct them to send packets through the isp (10.0.0.1) machine.

sudo nano /etc/network/interfaces

auto enp0s3

iface enp0s3 inet static

# assign the IP address

address 10.0.0.2

# set the netmask /24

netmask 255.255.255.0

# set the default route through isp

gateway 10.0.0.1

# use Google's DNS

dns-nameservers 8.8.8.8

sudo service network-manager restart

sudo ifup enp0s3

# **IPTABLES potek dela**

A typical cycle goes as follows:

* Solve a task.
* Start the firewall rules script sudo ./iptables1.sh start or sudo ./iptables1.sh restart.
* Inspect which rules have been activated: sudo iptables --list -nv.
* Test rules by running the appropriate program. In some cases, you'll need the other machine for testing (for instance, to test requests to the local services (HTTP and alike)):
  + ICMP with ping;
  + DNS with dig, e.g. dig www.fri.uni-lj.si;
  + HTTP with curl, e.g. curl google.com;
  + SSH client: ssh isp@ip-of-the-machine-your-are-connecting-to.
* Clear all rules by running sudo ./iptables1.sh reset.

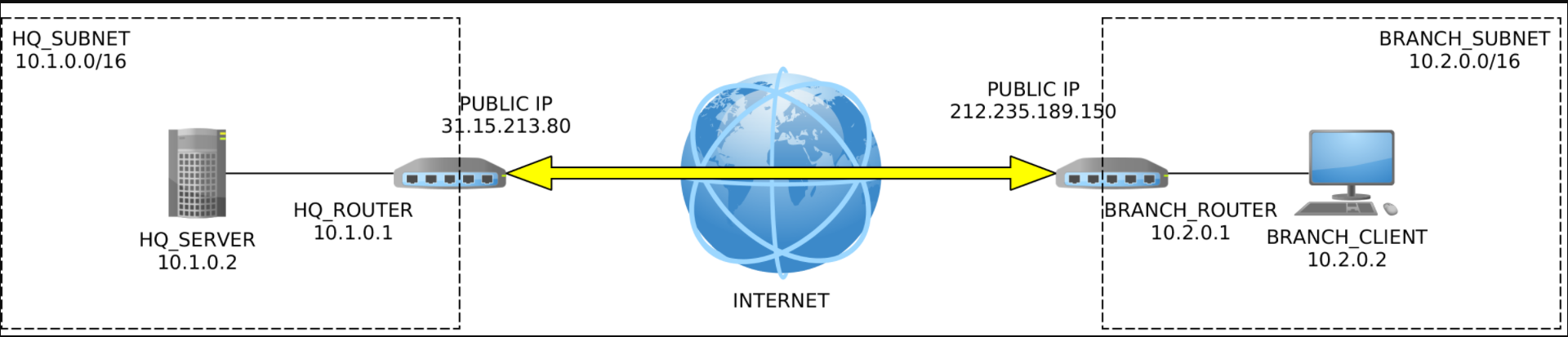
# **VPN**

(https://ucilnica.fri.uni-lj.si/mod/page/view.php?id=9139)

sudo apt update

sudo apt install strongswan ipsec-tools apache2 wireshark

Should non-superusers be able to capture packets?, select yes.



[Naredimo in konfiguriramo hq\_subnet in branch subnet.](#_Nastaviteve_za_router)

## Checkpoint

Let's make a sanity check before continuing. Assure that you can do the following:

* Send (and receive) pings between hq\_router and hq\_server (network 10.1.0.0/16);
* Send (and receive) pings between branch\_router and branch\_client (network 10.2.0.0/16);
* Send (and receive) pings between hq\_router and branch\_router. In this case, you should ping the public addresses of hq\_router and branch\_router. By public, I refer to the IPs assigned to routers on the enp0s3 interfaces. At university, these are the IP addresses from the 192.168.182.0/24. (These are in fact **private** IP addresses, but if we were setting up a real network, they'd be public. So for pedagogical purposes, we'll pretend they are public.) From here on, I'll refer the the public IPs of the routers with $HQ\_IPand $BRANCH\_IP for the IPs of the hq\_router and the branch\_router respectively.

## VPN IPsec tunnel

### hq\_router

At the hq\_router open the sudo nano /etc/ipsec.conf and fill it with the following content.

config setup

conn %default

ikelifetime=60m

keylife=20m

rekeymargin=3m

keyingtries=1

keyexchange=ikev2

authby=secret

conn net-net

leftsubnet=10.1.0.0/16

leftfirewall=yes

leftid=@hq

right=$BRANCH\_IP

rightsubnet=10.2.0.0/16

rightid=@branch

auto=add

Next, open file sudo nano /etc/ipsec.secrets and add the following line.

@hq @branch : PSK "secret"

Finally, restart the IPsec sudo ipsec restart so that the changes get loaded.

### branch\_router

sudo nano /etc/ipsec.conf

config setup

conn %default

ikelifetime=60m

keylife=20m

rekeymargin=3m

keyingtries=1

keyexchange=ikev2

authby=secret

conn net-net

leftsubnet=10.2.0.0/16

leftid=@branch

leftfirewall=yes

right=$HQ\_IP

rightsubnet=10.1.0.0/16

rightid=@hq

auto=add

 you can set multiple CIDR values, if you separate them with a comma – for instance:  leftsubnet=10.1.0.0/16,10.2.0.0/16.

sudo nano /etc/ipsec.secrets

@hq @branch : PSK "secret"

sudo ipsec restart

Which cipher suites are being used? Run sudo ipsec statusall to find out. Now change the configuration files /etc/ipsec.conf on both routers so that the the ESP and the IKE traffic will be secured with the following cipher suite: AES\_GCM\_16\_256. You may find this StrongSwan example useful.

<https://www.strongswan.org/testresults.html>

v seznamu klikneš na ikev2, nato klikneš na alg-aes-gcm (nasplošno pa tist algoritem, ki je zahtevan v navodilih)

nato klikneš na ipsec.conf (od moon - če gledaš za router), da vidiš vsebino

config setup

conn %default

ikelifetime=60m

keylife=20m

rekeymargin=3m

keyingtries=1

keyexchange=ikev2

authby=secret

ike=aes256gcm16

esp=aes256gcm16

conn net-net

leftsubnet=10.1.0.0/16

leftfirewall=yes

leftid=@hq

right=10.0.2.13

rightsubnet=10.2.0.0/16

rightid=@branch

auto=add

For login with certificate: (<https://www.strongswan.org/testing/testresults/ikev2/net2net-cert/>)

moon.ipsec.conf

config setup

conn %default

ikelifetime=60m

keylife=20m

rekeymargin=3m

keyingtries=1

keyexchange=ikev2

mobike=no

conn net-net

left=192.168.0.1

leftcert=moonCert.pem

leftid=@moon.strongswan.org

leftsubnet=10.1.0.0/16

leftfirewall=yes

right=192.168.0.2

rightid=@sun.strongswan.org

rightsubnet=10.2.0.0/16

auto=add

moon.ipsec.secrets

# /etc/ipsec.secrets - strongSwan IPsec secrets file

: RSA moonKey.pem

Namesto pem datotek imamo lahko der.

### Odprava težave z delovanjem spletne povezave

če ti internet ne dela na branch\_client / hq\_server

moraš na routerjih pognat: sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE

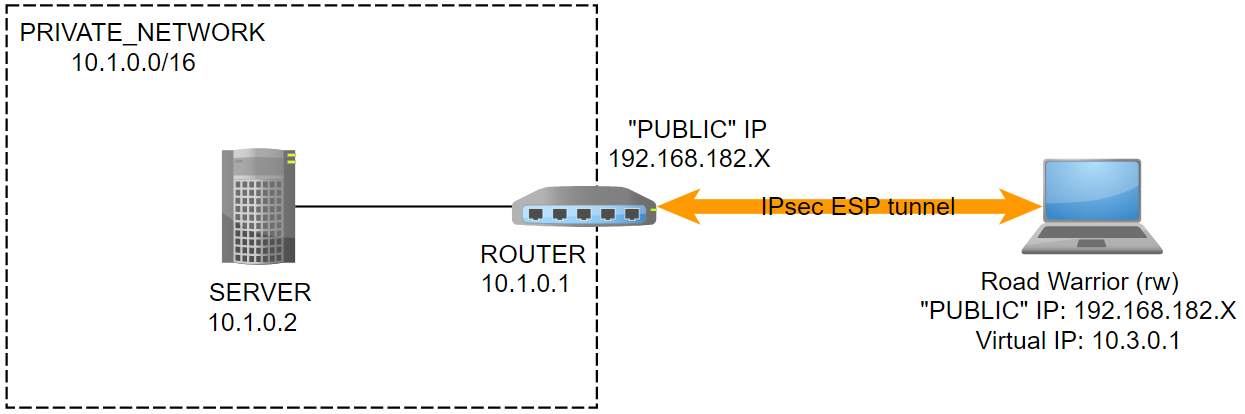
odgovor na tole z vaj za VPN

# FreeRADIUS

sudo apt update

sudo apt install freeradius freeradius-utils apache2 libapache2-mod-auth-radius

Power-off isp machine. Configure it to have a single NIC: go to Machine > Settings > Network and disable all Adapters except Adapter 1. Set it to either Bridged or NAT network (do not use NAT).



## radius1

First, let's register a new client (network access server, NAS) to the Radius server. Open sudo nano /etc/freeradius/clients.conf and make sure it contains the following entry. (The entry should already be there, this is just to be sure.)

client localhost {

ipaddr = 127.0.0.1

secret = testing123

require\_message\_authenticator = no

nastype = other

}

Next, let's add a new supplicant (end-user) to the database. We'll manage the database in a file. Open sudo nano /etc/freeradius/users and add the lines below. Make sure that you indent the second line with a tab.

"alice" Cleartext-Password := "password"

Reply-Message = "Hello, %{User-Name}"

echo "User-Name=alice, User-Password=password" | radclient 127.0.0.1 auth testing123 -x

## Debuging

First, stop the service with sudo service freeradius stop.

It may be best that you start FreeRADIUS in a new terminal window, or alternatively, in a new tab of the existing terminal widow by pressing ctrl+shift+t. This will allow you to monitor the output of the server, while also give you the ability to run additional commands. Start the server in the foreground with all logging and debugging turned on: sudo freeradius -X -d /etc/freeradius.

## HTTP Basic authentication with Apache and FreeRADIUS

First, enable auth\_radius module for apache and restart the apache server.

sudo a2enmod auth\_radius

sudo service apache2 restart

Next, configure Apache Radius settings in sudo nano /etc/apache2/ports.conf. Add the following lines.

# FreeRADIUS runs on localhost:1812 (standard RADIUS port).

# Apache will authenticate itself to the AAA server with PSK 'testing123'.

# The request shall time-out after 5 seconds, and retry at most 3 times.

AddRadiusAuth localhost:1812 testing123 5:3

# Next line configures the time (in minutes) in which the authentication cookie

# set by the Apache server expires

AddRadiusCookieValid 1

Next, tell Apache which pages require authentication. Open sudo nano /etc/apache2/sites-available/000-default.conf and add the following lines inside <VirtualHost \*:80> block. (Since, folder /var/www/html represents Apache's HTTP root folder, this in effect covers all pages.)

<Directory /var/www/html>

Options Indexes FollowSymLinks MultiViews

AllowOverride None

# ADD LINE 1

# Use basic password authentication

# AuthType Digest won't work with RADIUS

AuthType Basic

# ADD LINE 2

# Tell the user the realm to which they are authenticating.

AuthName "RADIUS Authentication for my site"

# ADD LINE 3

# Set RADIUS to be provider for this basic authentication

AuthBasicProvider radius

# ADD LINE 4

# Require that mod\_auth\_radius returns a valid user,

# otherwise access is denied.

Require valid-user

</Directory>

Reload Apache's configuration file with sudo service apache2 reload.

Finally, start the FreeRADIUS server in the foreground with sudo freeradius -X -d /etc/freeradius.

## Roaming

Start radius2. Assert the IP addresses of both machines. Let $RADIUS1 and $RADIUS2 denote the IP addresses of radius1 and radius2, respectively.

On radius1, create a new domain (or realm) called finland. Open sudo nano /etc/freeradius/proxy.conf and add the following.

home\_server hs\_finland {

type = auth+acct

ipaddr = $RADIUS2

port = 1812

secret = testing123

}

home\_server\_pool pool\_finland {

type = fail-over

home\_server = hs\_finland

}

realm finland {

pool = pool\_finland

nostrip

}

On radius2, create a new (local) domain called finland. Open sudo nano /etc/freeradius/proxy.conf and add the following two lines.

realm finland {

}

On radius2, define a new AAA client (AAA proxy) and define its credentials. Open sudo nano /etc/freeradius/clients.conf and add the following lines.

client $RADIUS1 {

secret = testing123

}

On radius2, create a new supplicant (end-user). Open sudo nano /etc/freeradius/users and define his or hers credentials. An instance is given below. Make sure the second line is tab-indented.

"pekka" Cleartext-Password := "password"

Reply-Message = "Hello, %{User-Name}"

Everything should now be set up. Make sure the FreeRADIUS server is running on both machines and in both cases in the foreground with sudo freeradius -X -d /etc/freeradius.

If you get an error stating that the port is already taken, stop the running instance of the server. If it is running in the background, you can stop it with sudo service freeradius stop. If it is running in the foreground, navigate to the terminal that shows the server output console and press ctrl+c.

Use the first machine to test whether the scenario works. Open a web browser and navigate to http://localhost. The browser should require you log-in. This time, log-in with pekka@finland and the appropriate password.

**Hint.** If you are using a normal browser, make use of private browsing for fast log-outs -- to log-out simply close the window. Alternatively, you can test using the terminalcurl --user pekka@finland:password http://localhost -v