RaspTunnel GRE Tunneling (Deprecated)

This page is deprecated and no longer in use.

Tornevall Networks RaspTunnel is an experimental project where we're adding micro servers into a DMZ-controlled environment. Main purpose is ipv6 GRE-tunneling with the help from your primary router, which should be configured to route traffic to a DMZ-address, where this micro server, known as Raspberry pi, are the tunnel gateway.

The tunnel project has a primary goal: To configure ipv6 tunnels, where they are normally not reachable (ISP's that for example have no idea of what they are doing), or where the ip range is weakly configured without proper reverse PTR's.

How does it work?

An installer injects required data into a raspberry pi, which is configured with a static ip address matching the network that needs it. It should be installed at the same ip range as your DHCP server and your incoming gateway shoud be configured with a DMZ-pointer to this machine.

With Tornevall Networks API, the pi will download data required for a controller to connect to the service, since the gateway should (unless the user have full control by themselves) be able to administered from an external point (Tornevall Networks IP-ranges). The API will also be used to configure dynamic tunnels.

As this is based on Linux (Raspian for pi 1, and Ubuntu 16 for pi 2 and pi 3), not only ipv6 tunnels may be considered. However, if there is ipv4 included here, we should make sure that the primary gateway are really routing networks properly. A recommended path for ipv4 tunnels should be somewhere at 10.0.0.0/8

This is a infrastructure issue, so the primary location of the ongoing project issue is located here:

> ⚠ INFRA-15 - Jira project doesn't exist or you don't have permission to view it.

Dead zones

In the current solution, there will be dead routes at two points: The entry points XX:1/128 will look dead for the customer endpoint and XX:2/128 will look dead for our endpoint. Everything from XX:3/112 and above will answer if they are set up (without firewalls)

Usage

Basically, everything is going to be handled through the pi. There are however an interface that automatically reconfigures the local tunnel ip address in case that is changing. The link looks like this:

https://api.tornevall.net/2.0/vpn/update/auth/tunnelUserName/key/tunnelUserKey/ip/auto/

One issue with the above address is that it is resolving to both ipv4 networks and ipv6 networks. If there are connection problems, you may want to try the pure ipv4-address instead, like this:

TorneAPI ipv4 only: https://api-4.tornevall.net/2.0/vpn/update/auth/tunnelUserName/key/tunnelUserKey/ip/auto/

To identify the current ip address http://identifier.tornevall.net/ip.php

Configuring the tunnel may be done manually with regular commands:

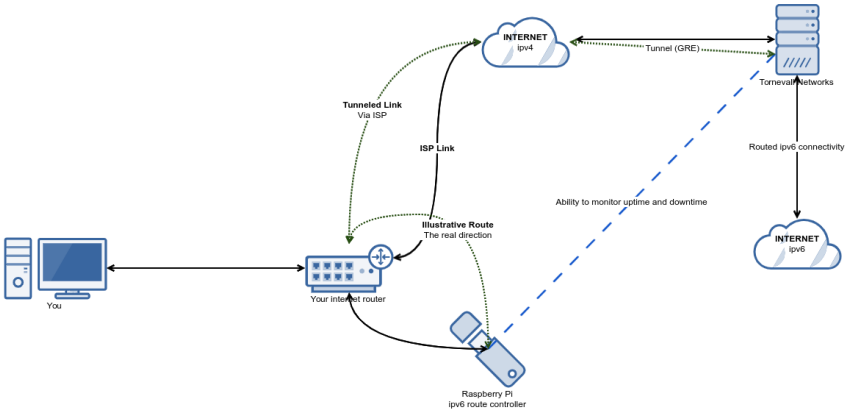
ip tunnel manually

On the preinstalled raspberry pi there is however scripts that is making thing for you in /usr/local/sbin

Script	What it does
/usr/local/sbin/pi-tunnel.run	Contains the current remote address required on connection and the current prefix. It actually runs pi-tunnel.sh in the same path

/usr/local/sbin/pi-tunnel.sh	Automatically configures a tunnel based on the data entered at pi-tunnel.run, so this goes with two parameters: <remote-connection-address> <your-prefix>

Visual view



Firewalls

Yes. They will be there in a new future, since we are not focused on this part. The important thing is that everything is blocked when the pi fires up in production mode, and somehow, some base services only, will be allowed to connect to the DMZ-address.

The project itself

The JIRA links is removed from this page.

Tornevall Networks RaspTunnel is an experimental project where we're adding micro servers into a DMZ-controlled environment. Main purpose is ipv6 GRE-tunneling with the help from your primary router, which should be configured to route traffic to a DMZ-address, where this micro server, known as Raspberry pi, are the tunnel gateway.

The tunnel project has a primary goal: To configure ipv6 tunnels, where they are normally not reachable (ISP's that for example have no idea of what they are doing), or where the ip range is weakly configured without proper reverse PTR's.

How does it work?

An installer injects required data into a raspberry pi, which is configured with a static ip address matching the network that needs it. It should be installed at the same ip range as your DHCP server and your incoming gateway shoud be configured with a DMZ-pointer to this machine.

With Tornevall Networks API, the pi will download data required for a controller to connect to the service, since the gateway should (unless the user have full control by themselves) be able to administered from an external point (Tornevall Networks IP-ranges). The API will also be used to configure dynamic tunnels.

As this is based on Linux (Raspian for pi 1, and Ubuntu 16 for pi 2 and pi 3), not only ipv6 tunnels may be considered. However, if there is ipv4 included here, we should make sure that the primary gateway are really routing networks properly. A recommended path for ipv4 tunnels should be somewhere at 10.0.0.0/8

This is a infrastructure issue, so the primary location of the ongoing project issue is located here:

> ⚠ INFRA-15 - Jira project doesn't exist or you don't have permission to view it.

Dead zones

In the current solution, there will be dead routes at two points: The entry points XX:1/128 will look dead for the customer endpoint and XX:2/128 will look dead for our endpoint. Everything from XX:3/112 and above will answer if they are set up (without firewalls)

Usage

Basically, everything is going to be handled through the pi. There are however an interface that automatically reconfigures the local tunnel ip address in case that is changing. The link looks like this:

https://api.tornevall.net/2.0/vpn/update/auth/tunnelUserName/key/tunnelUserKey/ip/auto/

One issue with the above address is that it is resolving to both ipv4 networks and ipv6 networks. If there are connection problems, you may want to try the pure ipv4-address instead, like this:

TorneAPI ipv4 only: https://api-4.tornevall.net/2.0/vpn/update/auth/tunnelUserName/key/tunnelUserKey/ip/auto/

To identify the current ip address http://identifier.tornevall.net/ip.php

Configuring the tunnel may be done manually with regular commands:

ip tunnel manually

On the preinstalled raspberry pi there is however scripts that is making thing for you in /usr/local/sbin

Script	What it does
/usr/local/sbin/pi-tunnel.run	Contains the current remote address required on connection and the current prefix. It actually runs pi-tunnel.sh in the same path
/usr/local/sbin/pi-tunnel.sh	Automatically configures a tunnel based on the data entered at pi-tunnel.run, so this goes with two parameters: <remote-connection-address> <your-prefix>