

HolePunch-UDPTunnel

An UDP tunnel system using NAT hole-punching to provide point-to-point tunnels between two clients across networks, through the use of an intermediary information server for punching, then creating a VPN tunnel interface and forwarding all traffic through the hole-punched UDP ports.

Project Structure

The project consists of two linked goolang projects, HolePunchUDPTunnel is the main project and it calls the executable produced by UDPTunnel:

- HolePunchUDPTunnel
 - Server mode: runs the hole-punch info exchange server.
 - Client mode: presents text user interface to select a client to connect to, performs hole-punch and launches the UDPTunnel.
- UDPTunnel
 - Runs a tunnel client/server with the given IP/ports, creates a vpn between clients when launched automatically by HolePunchUDPTunnel.

Directory Tree

Below is the directory tree of the project:

```
.
├── bin
│   └── HolePunch-UDPTunnel
├── holepunchudptunnel
│   ├── go.mod
│   ├── go.sum
│   ├── go.work
│   ├── main.go
│   ├── natholepunch
│   │   ├── clientdata.go
│   │   ├── client.go
│   │   ├── go.mod
│   │   └── server.go
│   ├── tui
│   │   ├── go.mod
│   │   ├── logging.go
│   │   └── ui.go
│   └── tunnelman
│       ├── go.mod
│       ├── tunneldata.go
│       ├── tunnelexec.go
│       ├── tunnelman.go
│       └── udptunnel
├── LICENSE
├── Makefile
└── README.md
```

```
└─ udptunnel
   ├── filter.go
   ├── go.mod
   ├── go.sum
   ├── LICENSE.md
   ├── logger.go
   ├── main.go
   └─ tunnel.go
```

7 directories, 27 files

Module Function Description

- **HolePunchUDPTunnel**

- Main program source directory, relies on compiled executable of support program UDPTunnel
- **main.go**
 - Starts hole-punch server or hole-punch client plus text user interface (TUI) in client mode
- **natholepunch**
 - Sub-module for hole-punch server and client code
 - **server.go**
 - Code for running the hole-punch information exchange server
 - **client.go**
 - Code for running the hole-punch client and performing UDP hole-punching
 - **clientdata.go**
 - Defines structs that hold client data for both client and server
- **tui**
 - Sub-module for TUI code
 - **ui.go**
 - Code for displaying, rendering and updating the TUI with the information from the other sub-modules
 - **logging.go**
 - Code for redirecting StdOut output from other modules into a text view inside the TUI
- **tunnelman**
 - Sub-module for UDPTunnel manager code
 - **tunnelman.go**
 - Code for managing UDPTunnel configuration and creating UDPTunnel configuration based on hole-punch operation
 - **tunnelexec.go**
 - Code for executing the UDPTunnel program
 - The binary executable compiled from the UDPTunnel directory is embedded into our HolePunchUDPTunnel executable from this file.
 - **tunneldata.go**
 - Defines structs that hold UDPTunnel configuration data

- **UDPTunnel**

- Support program source directory, embedded into and called by HolePunchUDPTunnel
- **main.go**

- Reads configuration file and launches a tunnel server/client
- **tunnel.go**
 - Creates a new tunnel interface (such as tun0) and listens to trafic
 - Forwards raw packets to tunnel interface (local client -> UDPTunnel -> remote client)
 - Accepts raw packets from tunnel interface (local client <- UDPTunnel <- remote client)
- **filter.go**
 - Security filter to only allow specified ports through the UDP tunnel
- **logger.go**
 - Prints log and UDP tunnel traffic statistics

Count Lines of Code

Below is the result of counting the lines of code (generated by gocloc):

Language	files	blank	comment
code			

Go	13	378	643
1789			
Markdown	3	36	0
176			
Makefile	1	11	8
34			

TOTAL	17	425	651
1999			

Compilation

1. Begin by going to the project directory.
2. Compile the code by running `make all` (if compiling outside of China, change `prepare-cn` to `prepare` in `Makefile` line 12).
3. Go to the `bin` directory by running `cd bin`.
4. The compiled project executable binary is found as `HolePunch-UDPTunnel` here.

Usage

Make sure to launch the executable from the same folder as the executable (i.e. `cd bin && ./HolePunch-UDPTunnel`) To use this project, at least three devices are required:

- Hole-punch info exchange server
 - Must be run on a server with a public IP
 - Run `./HolePunch-UDPTunnel --server`

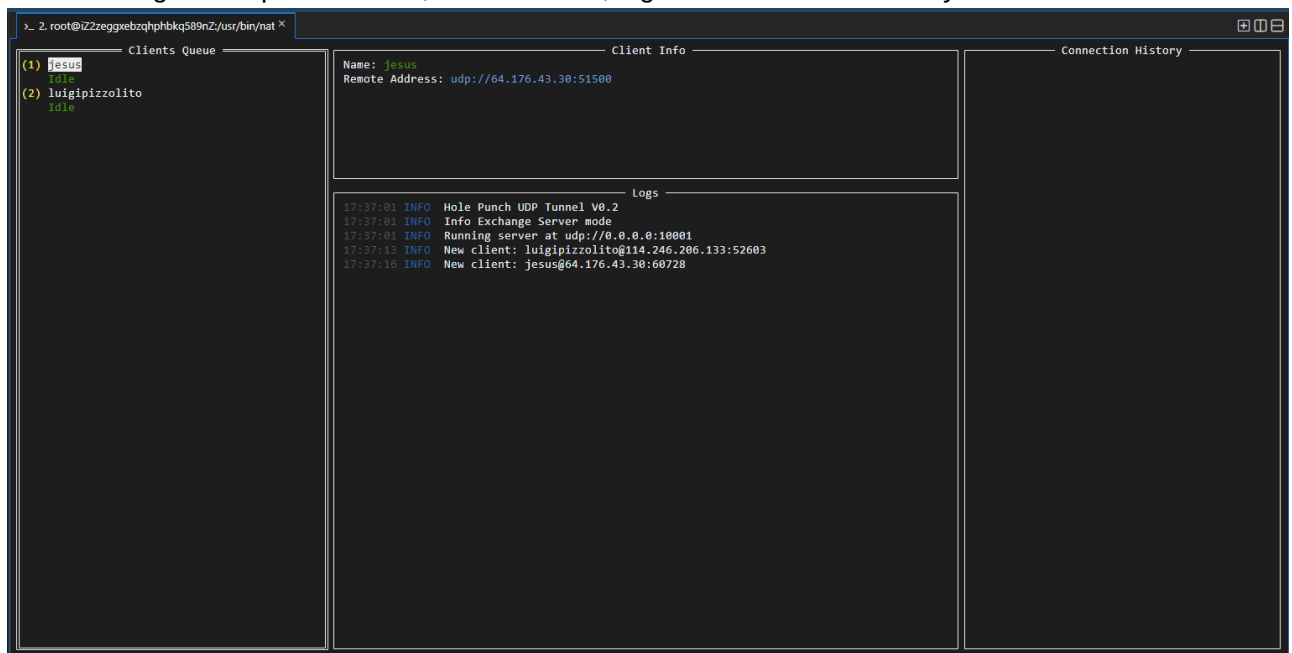
- Hole-punch client (2 or more)
 - Run on clients behind NATs
 - Shows text user interface (TUI) to user
 - Run `./HolePunch-UDPTunnel -a <Hole-Punch Server Public IP>`
 - replace `<Hole-Punch Server Public IP>` to the actual public IP of the server running the hole-punch info exchange.
 - Default username is taken to be the OS username
 - Other options available through `./HolePunch-UDPTunnel --help`:

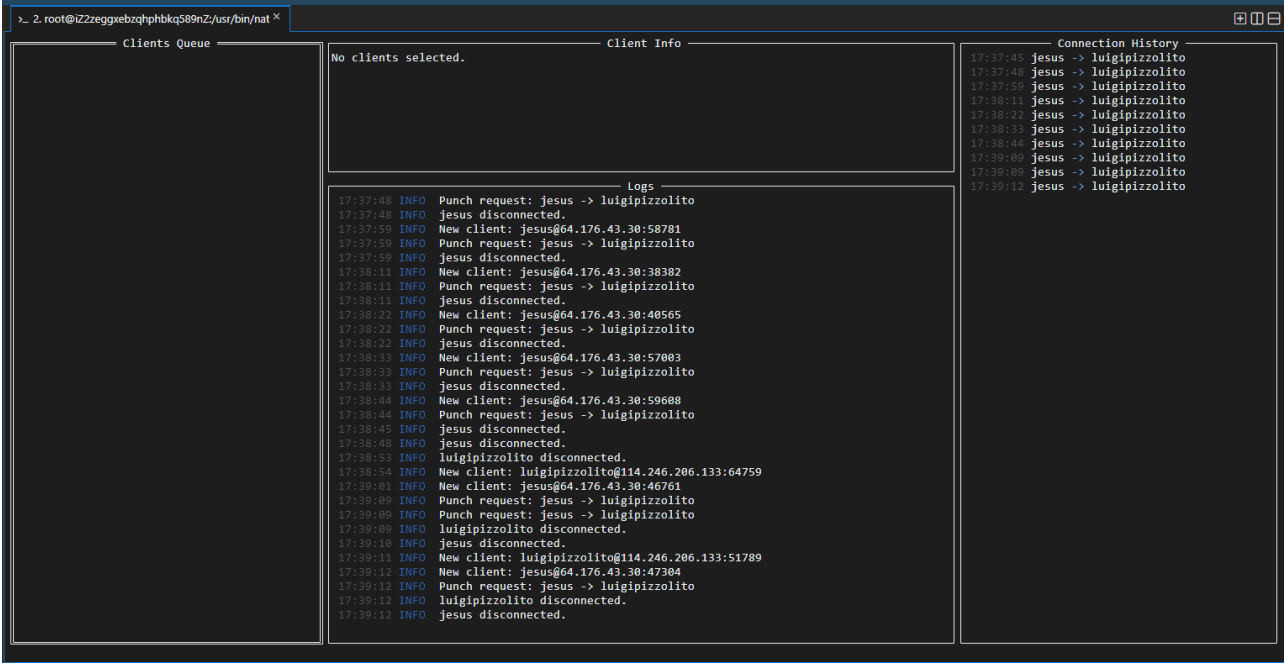
Usage of bin/HolePunch-UDPTunnel:

```
-a string
    Info exchange server IP address to connect to (default "127.0.0.1")
-l string
    Specify Local side ID (default "luigipizzolito")
-p string
    Info exchange server port to listen on or connect to (default
"10001")
-r string
    (optional) Specify remote side ID to tunnel to
-server
    Run in info exchange server mode (run this on a publicly accesible
IP)
-t duration
    Specify reconnection timeout (default 2s)
```

Server TUI

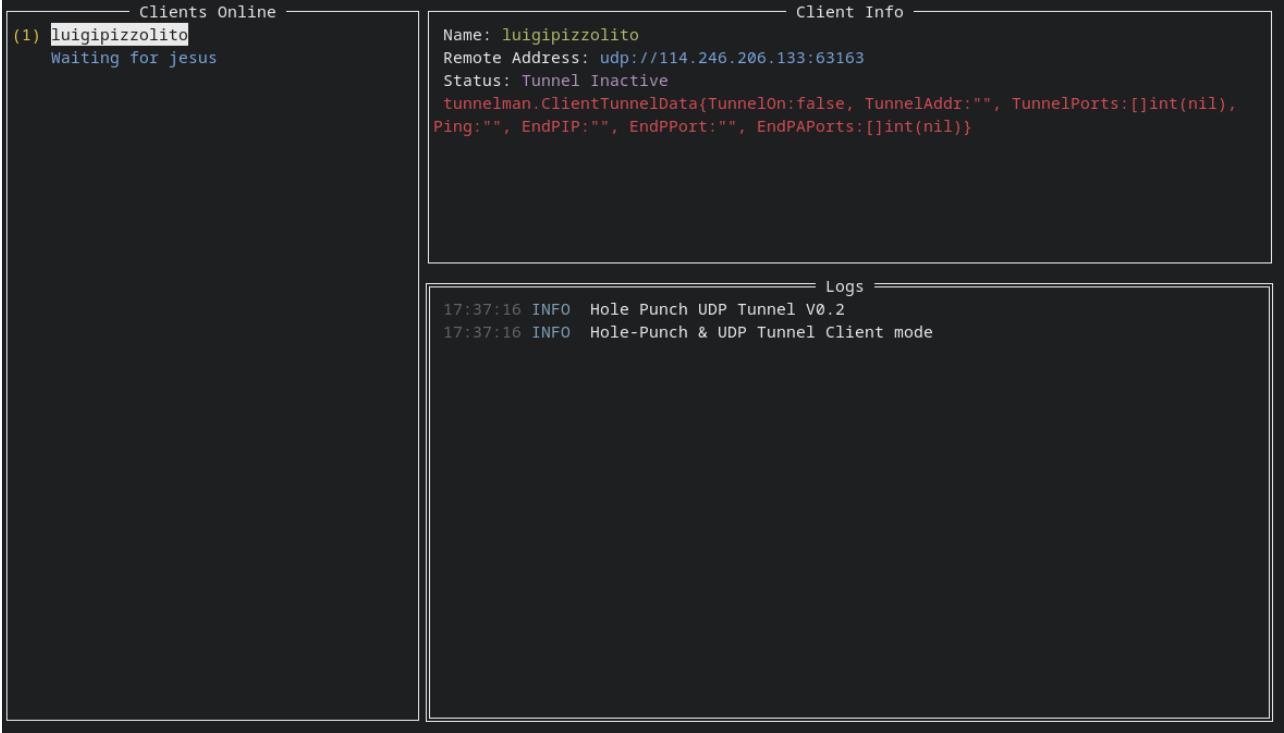
- After running a hole-punch server, the client info, logs and connection history can be seen:





Client TUI

- After running a hole-punch client, the TUI can be seen:
- Click, press enter, or press a number key to select a client to connect to.



- The hole-punch operation will begin, and pings will be sent in between clients:

```

Client Info
Name: luigipizzolito
Remote Address: udp://114.246.206.133:51789
Status: Tunnel Inactive
tunnelman.ClientTunnelData{TunnelOn:false, TunnelAddr:"", TunnelPorts:[]int(nil), Ping:"", EndPIP:"", EndPPort:"", EndPAPorts:[]int(nil)}

Logs
17:39:05 INFO Request for hole punch to luigipizzolito
17:39:05 WARN Remote client does not want to connect
17:39:05 INFO Remote client is Idle
17:39:05 INFO Please ask remote client to connect to you too
17:39:05 INFO Waiting in loop for luigipizzolito
17:39:09 INFO luigipizzolito accepted our connection, performing punch now
17:39:09 INFO Hole punching to luigipizzolito
17:39:12 INFO Got hole-punch addr from exchange server: luigipizzolito@114.246.206.133:51789
17:39:13 INFO Ping sent to luigipizzolito@114.246.206.133:51789
17:39:13 INFO received ping from jesu
17:39:13 INFO Ping: 56.733178ms
17:39:14 INFO Ping sent to luigipizzolito@114.246.206.133:51789
17:39:15 INFO received ping from jesu
17:39:15 INFO Ping: 331.457479ms
17:39:16 INFO Ping sent to luigipizzolito@114.246.206.133:51789
17:39:16 INFO received ping from jesu
17:39:16 INFO Ping: 23.662097ms

```

- After successful hole-punch, `sudo` password is prompted, enter sudo password to launch the UDP tunnel:

```

Client Info
Name: luigipizzolito
Remote Address: udp://114.246.206.133:51789
Status: Tunnel Inactive
Ping: 23.745256ms
tunnelman.ClientTunnelData{TunnelOn:false, TunnelAddr:"<ip_here>", TunnelPorts:[]int{}, Ping:"23.745256ms", EndPIP:"114.246.206.133", EndPPort:"51789", EndPAPorts:[]int{}}

Logs
17:39:18 INFO Completed 5 pings
17:39:18 INFO Ready to open tunnel
17:39:18 INFO Determined to be tunnel server
17:39:18 WARN Elevating privileges to open tunnel
17:39:18 WARN Enter Sudo Password:
17:39:21 INFO Tunnel server connected to :47304
17:39:21 INFO Wrote configuration file for UDP tunnel
17:39:21 INFO Starting Tunnel Daemon now
17:39:21 INFO Extracted udptunnel executable to /tmp/embedded-executable-1883575105
17:39:21 INFO File copied successfully.
17:39:21 INFO Copied config file to /tmp
17:39:21 INFO 2024/01/11 17:39:21 main.go:178: loaded config:
{
  17:39:21 INFO 2024/01/11 17:39:21 main.go:238: embedded-executable-1883575105 starting in server mode
  17:39:21 INFO 2024/01/11 17:39:21 tunnel.go:77: created tun device: tun0
}

```

- After launching UDP tunnel, one client is assigned IP `10.0.0.1` and the other client is assigned IP `10.0.0.2`
- You may now access the other client with any application or protocol (including TCP) directly by using their IP.

- For example, SSHing into the other client:

```
[jesus@fedora ~]$ ssh luigipizzolito@10.0.0.2
luigipizzolito@10.0.0.2's password:
```

```
[jesus@fedora ~]$ ssh luigipizzolito@10.0.0.2
luigipizzolito@10.0.0.2's password:
Last login: Thu Jan 11 18:07:27 2024 from 10.0.0.1
> neofetch

      .-
     .o+
    `ooo/
   `+oooo:
  `+oooooo:
 -+ooooooo+:
`/:-:++oooo+:
`/++++/+++++++:
`/+++++++/+++++++:
`/+++o0000000000000/`
./oooo00000++o000000+`
.o0000000-````/o000000+`
-o000000.      :000000.
:0000000/       o0000+++.
/o0000000/      +000000/-
`/o000000+/-:-  -:/+00000+-
`+000+:.-      `.-/+000:
++:.          `.-/+/
`/           `./

      luigipizzolito@LPArchLinuxWorkstation
-----
OS: Arch Linux x86_64
Host: 82RG Legion R9000P ARH7H
Kernel: 6.6.10-arch1-1
Uptime: 4 hours, 13 mins
Packages: 1825 (pacman), 23 (flatpak)
Shell: zsh 5.9
Resolution: 1920x1080
Terminal: /dev/pts/9
CPU: AMD Ryzen 7 6800H with Radeon Graphics (16) @ 1.800GHz
GPU: AMD ATI Radeon 680M
GPU: NVIDIA GeForce RTX 3060 Mobile / Max-Q
Memory: 17745MiB / 31279MiB
Disk (/): 195G / 200G (100%)
Local IP: 192.168.1.105

  system  luigipizzolito@LPArchLinuxWorkstation  18:09:40  100%
```