Prelude

〈elevator Music〉

WHOAMI?

In brief. Im Tree, a big nerd

Establish Credibility

- Started breaking computers when I was about 12
- Somehow ended up with a job writing a software model checker at Victoria University
- Started working in cybersecurity (for Aura Information Security)
- And have been ever since for the last 5 years
- Moved from pentesting to infrastructure
- Built a lot of random crap during that time

Order of business

RSSH -> SSH -> WAG -> Wireguard & eBPF

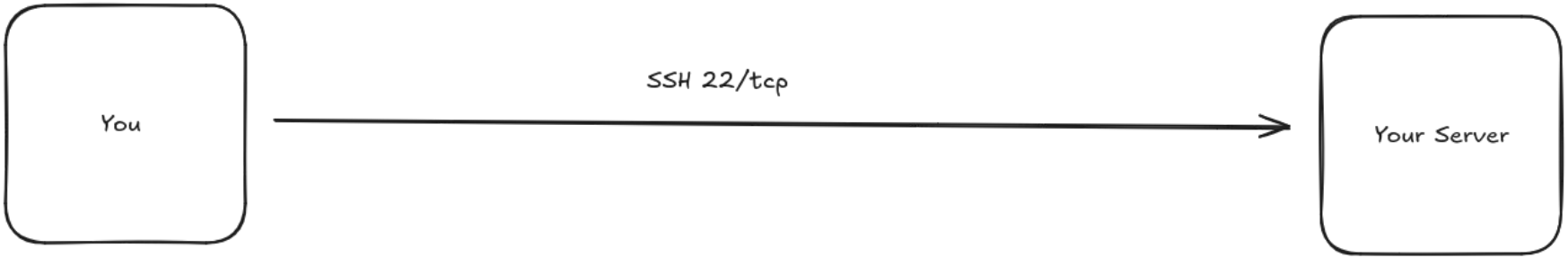
(many tangents)

Reverse SSH (RSSH)

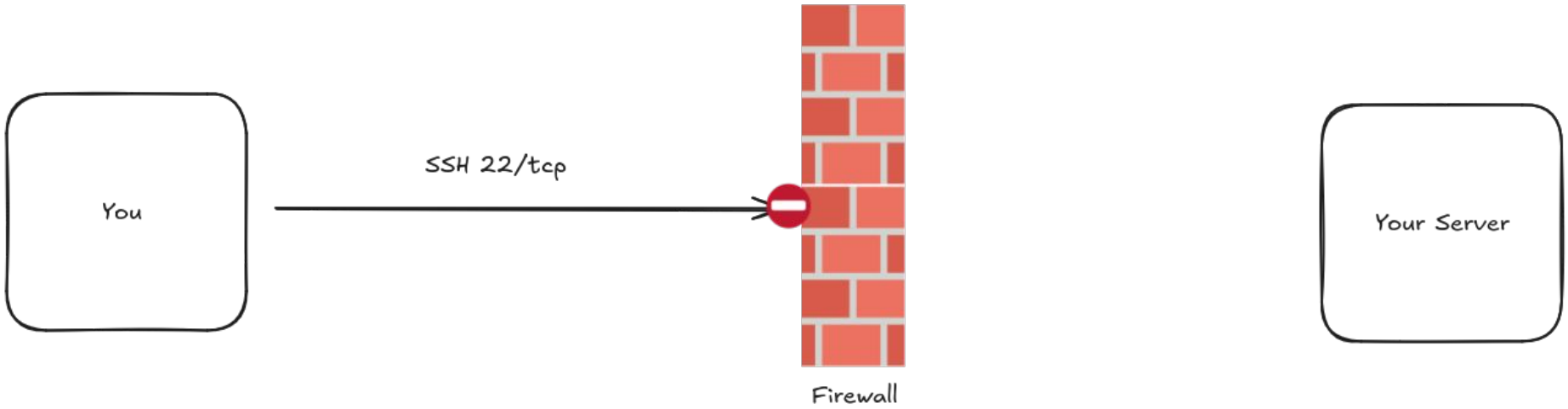
RSSH- Reason for existing

I wanted to do SSH but in reverse 😎

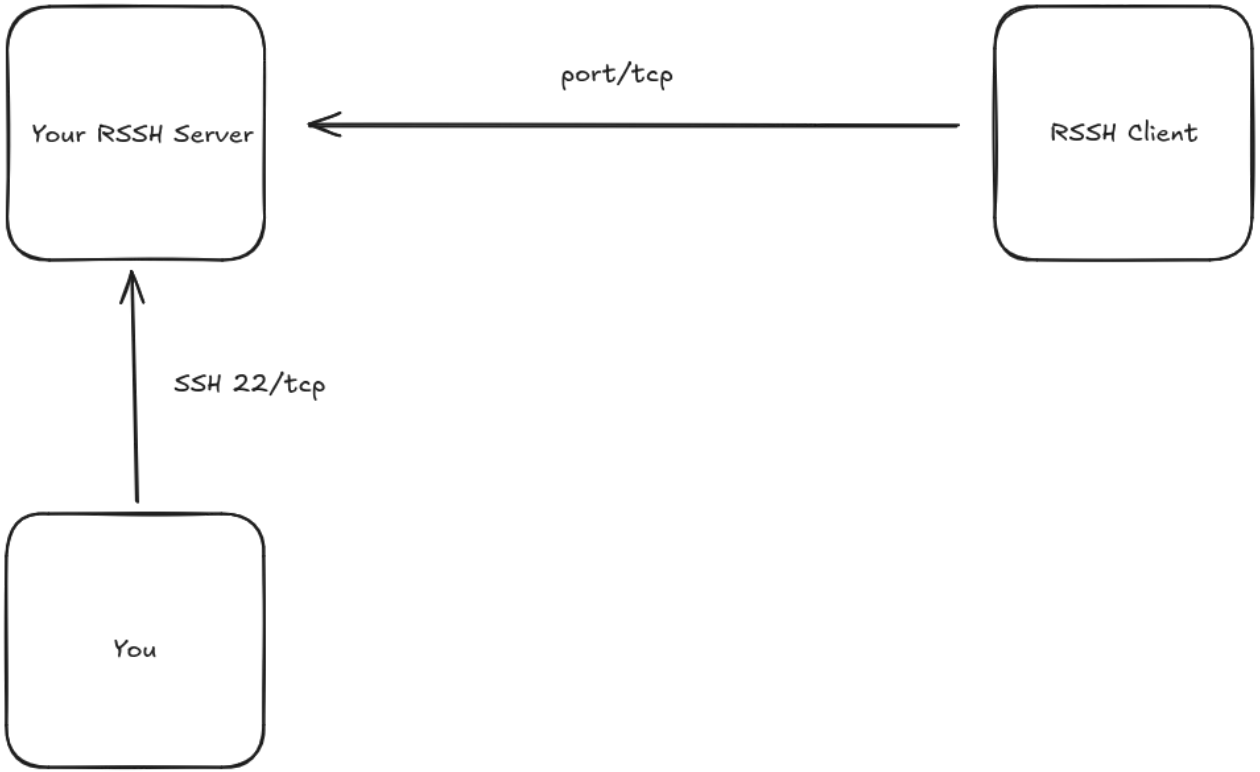
RSSH vs SSH



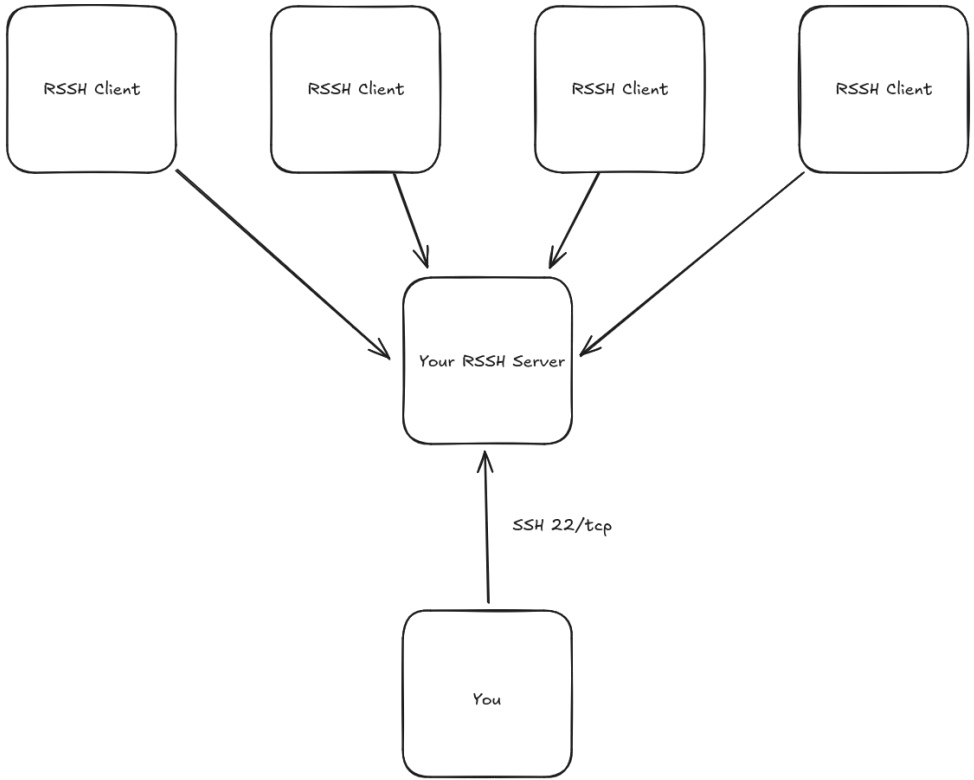
RSSH vs SSH



RSSH vs SSH



RSSH vs SSH


RSSH Client
RSSH Client
RSSH Client
RSSH Client
Your RSSH Server
SSH 22/tcp
You

RSSH- Features, Functions and FUN

- Single Golang Binary
- Implements the entire feature set of the OpenSSH toolset
  - TCP Forwarding
  - Tun (VPN)
  - File Copying
- No need to rely on SSH being installed on the target system

https://github.com/NHAS/reverse_ssh

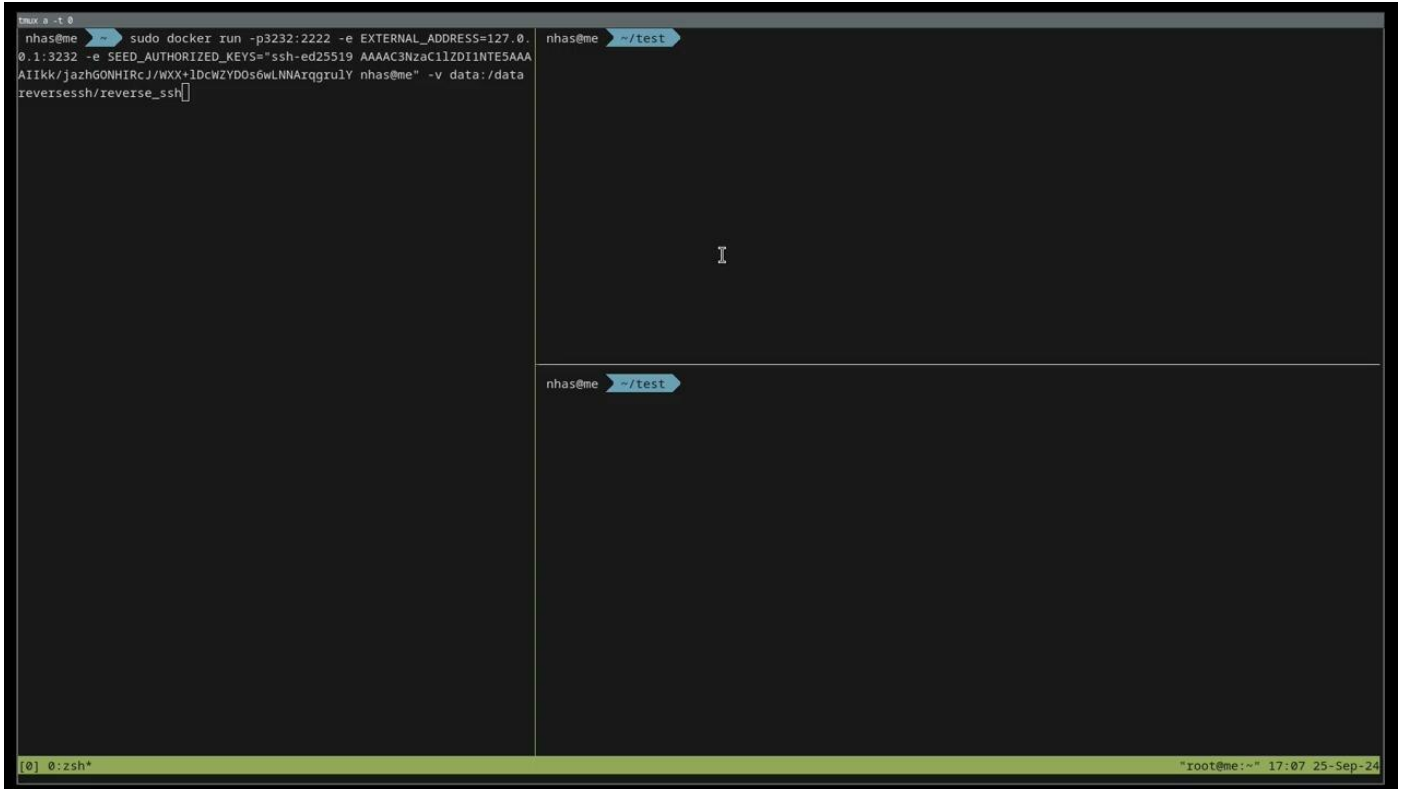
RSSH- Features, Functions and FUN

RSSH also:

- Supports multiple network transports
  - HTTP
  - WebSockets
  - TLS
- Windows? We do that too
- Multi-arch
- VT100 console emulation for fun
- Usefulness? Yes…

https://github.com/NHAS/reverse_ssh

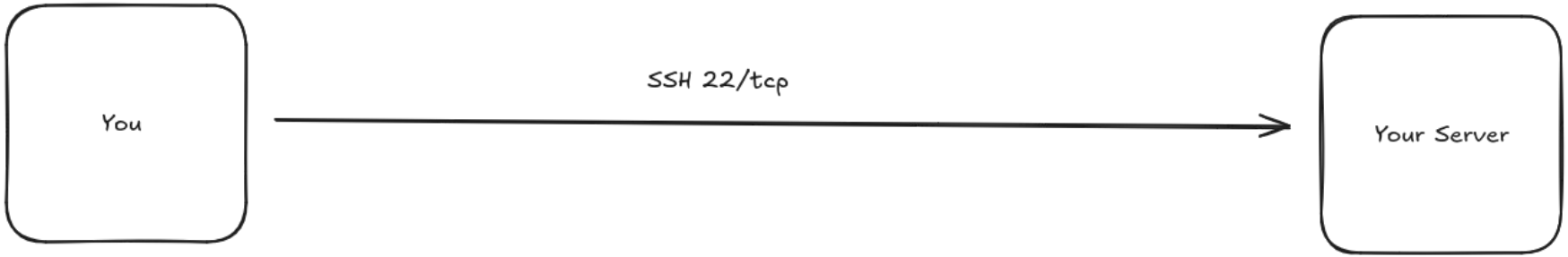
RSSH- Features, Functions and FUN



Learning Time!

SSH Crash course - The Protocol

- Most people think of SSH just like this:
- Single connection, then magic happens

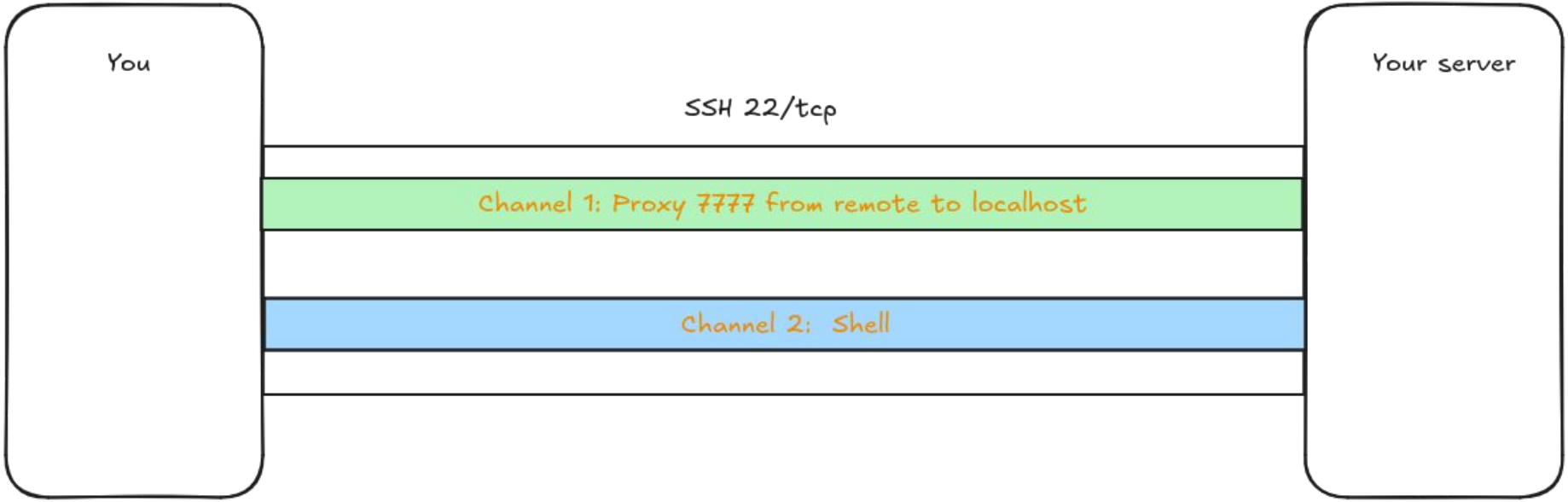


SSH Crash course - The Protocol

- But!
- SSH is actually quite interesting, it’s multiplexed
- Meaning that while there is *one* connection, over that connection it has multiple logical connections called channels
- So if we do something like and zoom in:

```
ssh -L 7777:127.0.0.1:7777 your.cool.ssh.server
```

SSH Crash course - The Protocol



SSH Crash course - interoperability

- Very simple (as long as doing the crypto and underlying protocol is done for you)
- https://datatracker.ietf.org/doc/html/rfc4254
  - Defines the Channels that the SSH tool uses
  - "session"
  - "direct-tcpip"
  - “forwarded-tcpip”
  - "tcpip-forward"

From there it's all implementation details

SSH Crash course - A Bit On golang

- SSH library

```
// Before use, a handshake must be performed on the incoming net.Conn.
sshConn, chans, reqs, err := ssh.NewServerConn(tcpConn, sshConfig)
if err != nil {
        log.Printf("failed to handshake (%s)", err)
        continue
}

// Check remote address
log.Printf("new ssh connection from %s (%s)", sshConn.RemoteAddr(), sshConn.ClientVer

// Print incoming out-of-band Requests
go handleRequests(reqs)
// Accept all channels
go handleChannels(chans)
```

SSH Crash course - A Bit On golang

```go
func handleChannels(chans <-chan ssh.NewChannel) {
        // Service the incoming Channel channel.
        for newChannel := range chans {
                // Channels have a type, depending on the application level
                // protocol intended. In the case of a shell, the type is
                // "session" and ServerShell may be used to present a simple
                // terminal interface.
                if t := newChannel.ChannelType(); t != "session" {
                        newChannel.Reject(ssh.UnknownChannelType, fmt.Sprintf("unknown channel type: %s", t))
                        continue
                }
```

SSH - TL:DL

- Multiplexed
- Extremely extensible
- Fun to write your own clients
- Secure

WAG

A VPN! Kind of...

Theft

- The best way to have fun is to commit crime
- And by crime I mean adding to a well known open source VPN technology and making it better

That’s right, it’s time to talk about WireGuard™

Wireguard - The initial theft

Wireguard:

- An extremely simple VPN
- Uses the Noise Protocol 😀
- Layer 3 only
- UDP only (https://openvpn.net/faq/what-is-tcp-meltdown/)
- Hard to setup wrong

WireGuard - The initial theft

```
[Interface]
PrivateKey=+DkFshdf4EnFtMyhx9FUR/n7/iZR1+a788TxlKfa71E=
Address=172.24.25.3
DNS=192.168.234.95

[Peer]
# z20-demo
PublicKey=t1GmyuY/lKn7lOlLNNraEShEDPkJJ4KC1mVo2cMMdmo=
Endpoint=10.111.56.95:51820
AllowedIPs=172.24.25.1,192.168.234.0/24
```

Wireguard- The Downsides

- No MFA :(
- No Enrolment
- No DHCP
- No restrictions per service/port
- Difficult user management
- Why Jason why

Why jason why

- Jason Donenfeld, creator of Wireguard™
- Tailscale vectorized I/O (2022 -> 2023, 188)
    - Left on read for 2 months
    - 3Gb/s -> 5Gb/s
    - 52 lines of code changed
- Wg-quick, abandoned wg-dynamic
    - “WireGuard currently uses static addresses everywhere. This is because that is mostly a better way to design your network. But in some cases, insane people want dynamic IP addresses or other dynamic configuration.”
- No kernel netlink

Hi Jordan,

> Howdy,
> Sorry if this is terribly rude (Im not entirely sure of the etiquette
> around mailing kernel contributors).
I'm also not entirely sure about the etiquette, but I did not find it
rude at all :-).
>
> I saw your work on adding netfilter notifications on wireguard peer
> changes back in 13 Mar 2021
> (https://lore.kernel.org/lkml/c97061f5-2d28-0323-c16a-aacacbdc734f@lotz.l=
i/)
>
>
> Just wanted to say I absolutely love this idea and your work. Did that
> ever move forward at all? I cant find any other reference to it.
That is nice to hear!

I did update this patch twice:

v2 https://www.spinics.net/lists/netdev/msg714366.html - fixed a
possible uninitialized use.

v3 https://www.spinics.net/lists/kernel/msg4312979.html - disables the
notifications per default and allows enabling the monitoring per device
via netlink.

Unfortunately I did not get any answer to the last patch and I need to
ask Jason what would be needed to get it added.

> Again, sorry if this is a pain. Just curious (and want it to be added
> to the kernel haha)

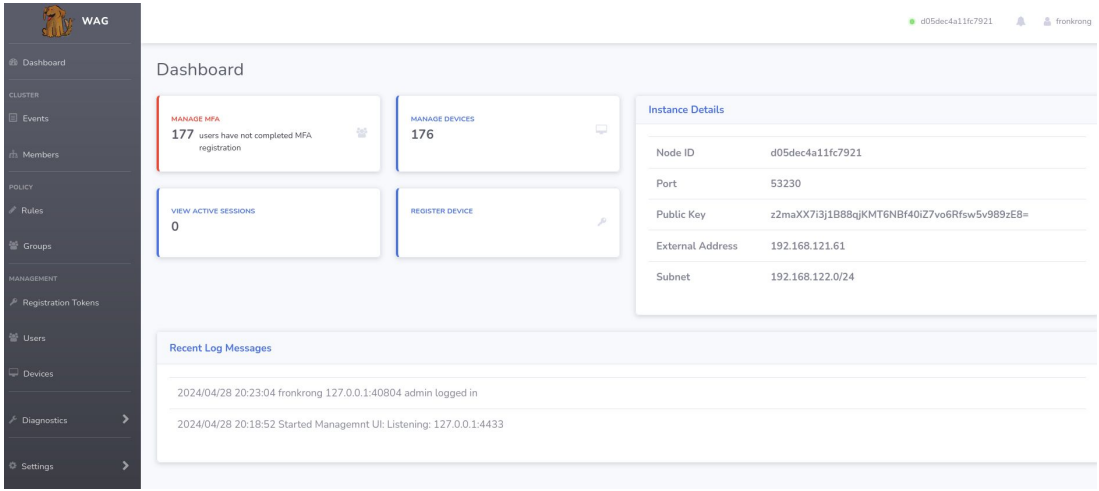
No worries :-)

Cheers,

Linus

WAG

- MFA 😀
- Enrolment 😀
- Magic eBPF firewall 😀
- User & Device management 😀
- Distributed VPN 😀

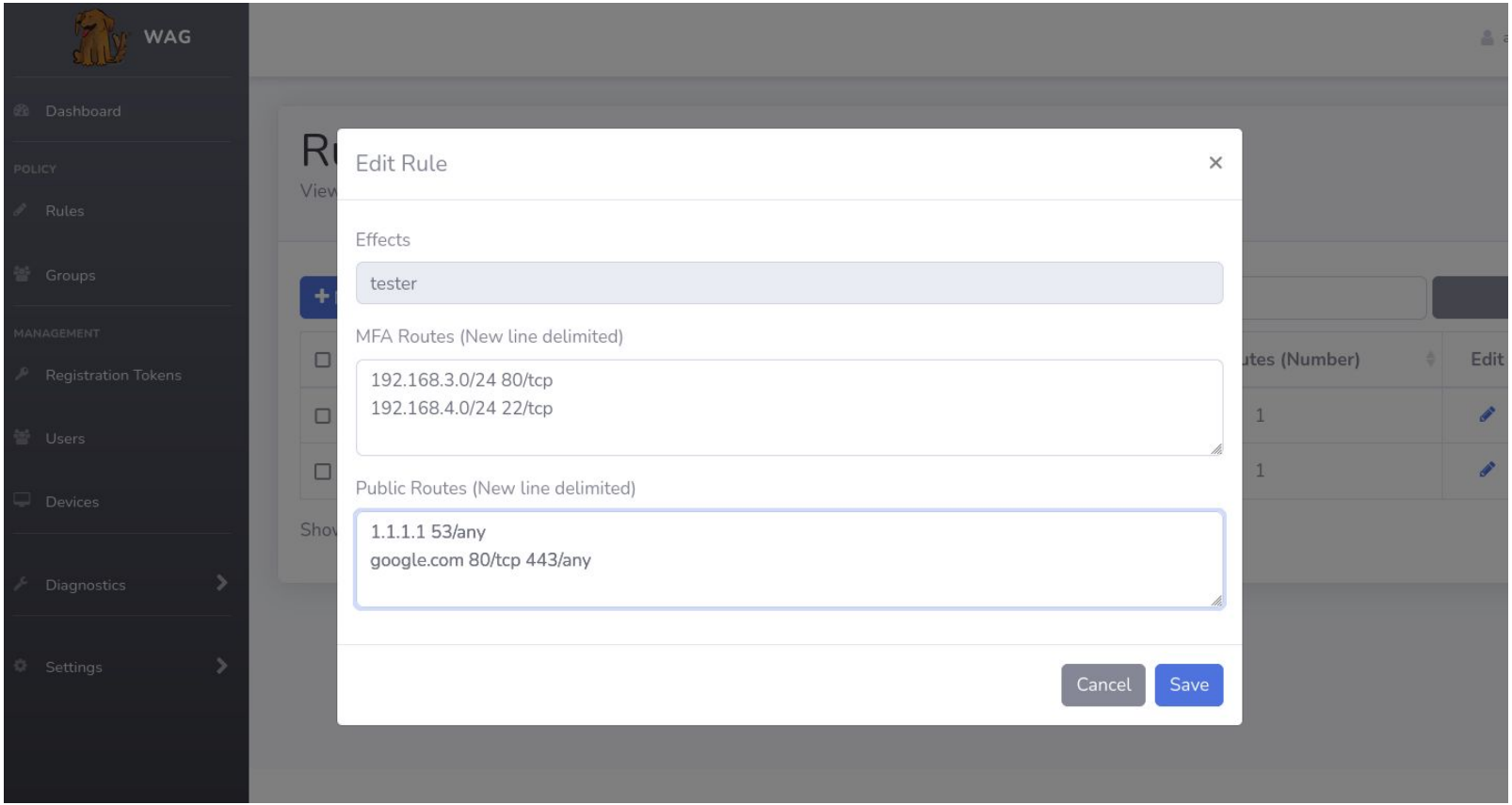


WAG - MFA

MFA: It’s a bit silly

- Retrieve list of peers
- Store current real ip
- Check if it’s changed
- Profit?

WAG - FIREWALL



Ebpf - Crash Course

> eBPF is a revolutionary technology with origins in the Linux kernel that can run sandboxed programs in a privileged context such as the operating system kernel. It is used to safely and efficiently extend the capabilities of the kernel without requiring to change kernel source code or load kernel modules.

- https://ebpf.io/what-is-ebpf/

eBPF - Crash Course

- XDP
  - Specialised eBPF for fast networking
  - Used everywhere to filter packets

https://engineering.fb.com/2018/05/22/open-source/open-sourcing-katran-a-scalable-network-load-balancer/

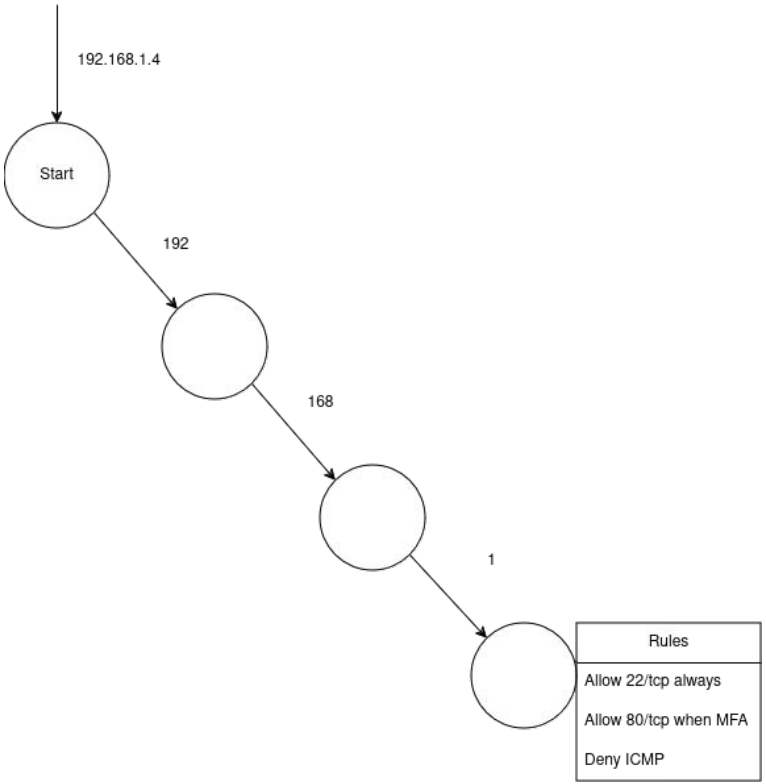
https://github.com/gamemann/XDP-Firewall

Ebpf - Crash Course

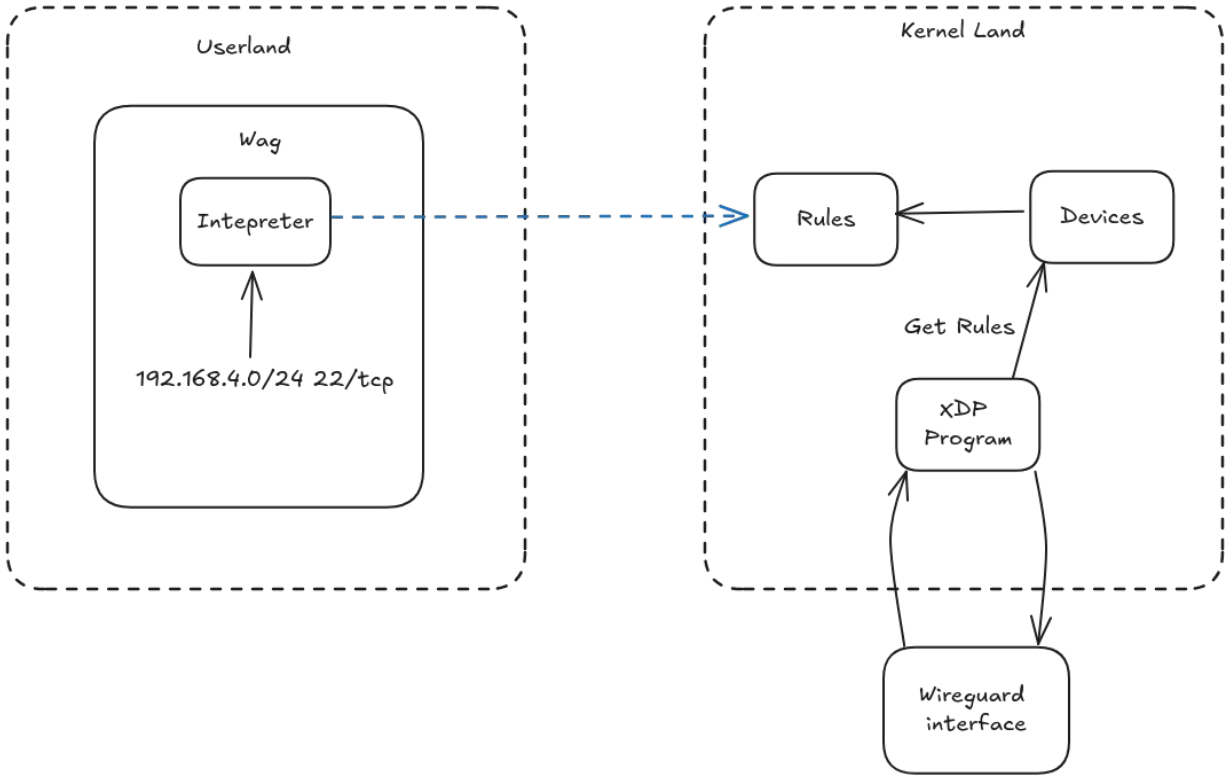
Userland -> Kernel land:

- HashMaps
- Longest prefix matching trie
- And more…

WAG - EBPF PUTTING IT TOGETHER

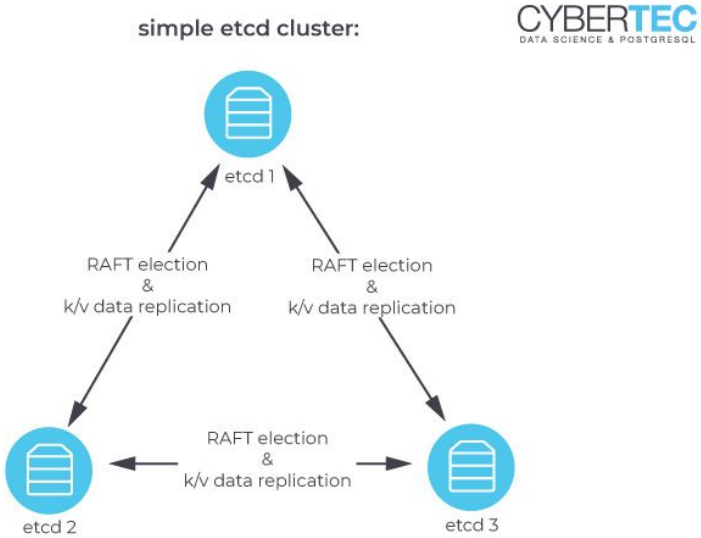


WAG - EBPF PUTTING IT TOGETHER



WAG - A brief Word on ETCD

- Embeddable in golang
- Distributed Key Value store (Raft)
- Does events (through watching values)



The End

38 slides and some change.

Thanks for listening!