Security of Networks, Services, and Systems

Firewalls and VPNs

Ricardo Morla

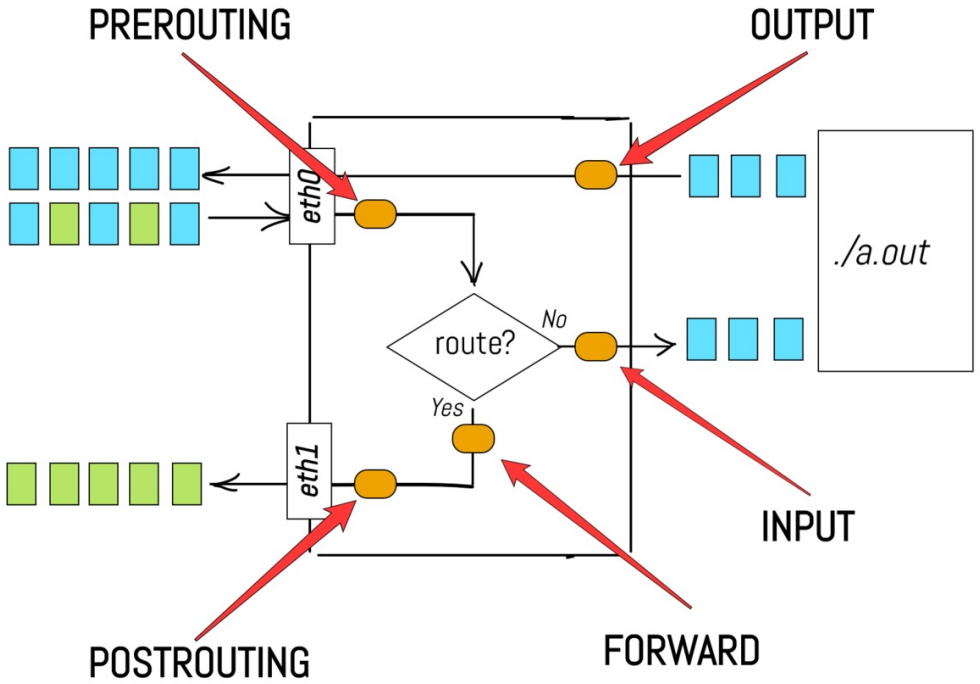
FEUP – SSR/M.EEC, SR/M.EIC

Firewall

- Network element
  - Controls ingress, egress, and local application traffic
  - Commercial devices, open source appliances, control host traffic
  - Also software in a host to control access to the host
- Applies actions to traffic that matches firewall rule
  - Accept, deny, reject, mangle, etc
  - IP address, port numbers, etc

Firewall on Linux – iptables

- Based on chain of rules applied at different points
  - INPUT/OUTPUT: packets destined to the host / generated by the host
  - FORWARDING: packets received by the host but not destined to the host
  - PREROUTING/POSTROUTING: packets received by the host / sent by the host
- Set of actions
  - ACCEPT, DROP, QUEUE (send to user space), RETURN (stop executing chain)

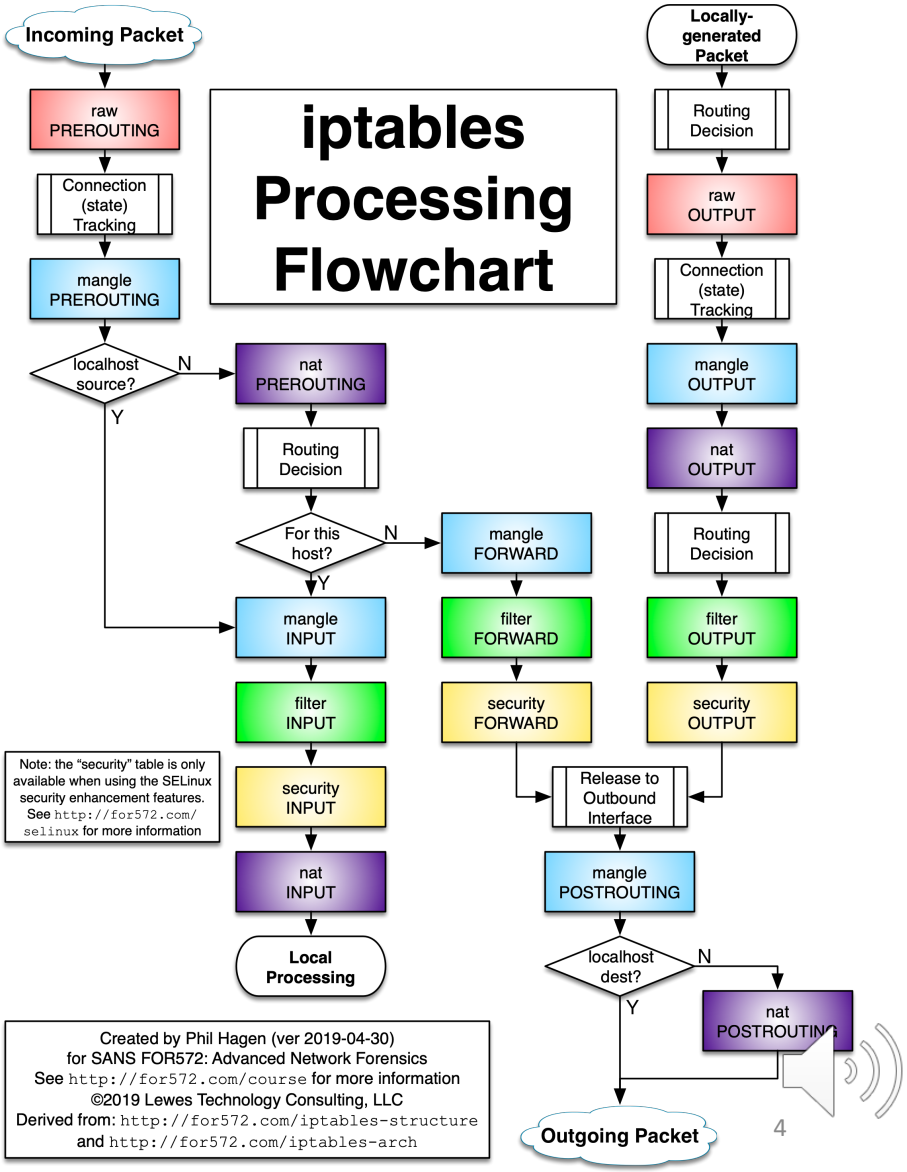


https://iximiuz.com/en/posts/laymans-iptables-101/

iptables flowchart

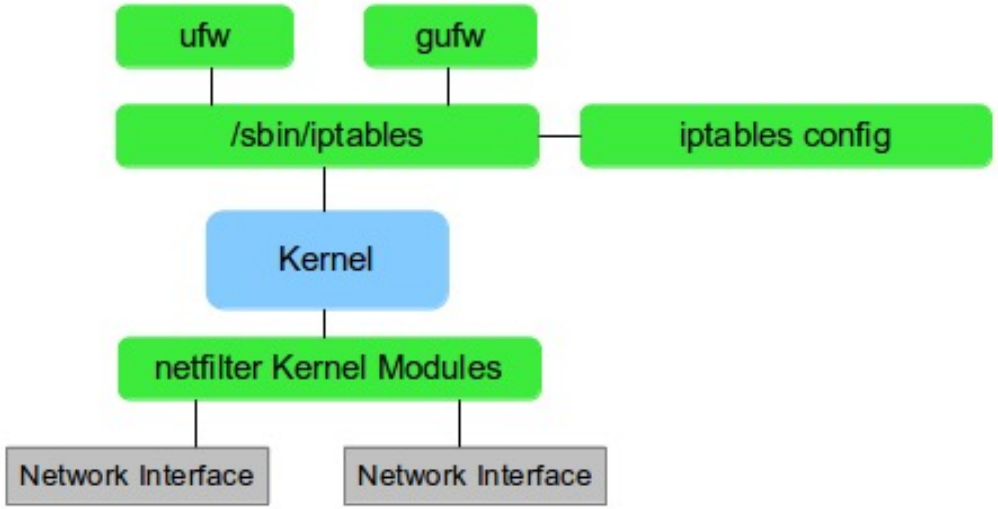
Tables↓/Chains→	PREROUTING	INPUT	FORWARD	OUTPUT	POSTROUTING
(routing decision)				✓	
raw	✓			✓	
(connection tracking enabled)	✓			✓	
mangle	✓	✓	✓	✓	✓
nat (DNAT)	✓			✓	
(routing decision)	✓			✓	
filter		✓	✓	✓	
security		✓	✓	✓	
nat (SNAT)		✓			✓

https://www.digitalocean.com/community/tutorials/a-deep-dive-into-iptables-and-netfilter-architecture



Firewall applications

- Direct rule configuration
- Applications that manage the iptables rules on behalf of the user



Example iptables rules

1. `iptables -t filter --list`
   - Lists the rules for the filter table; filter is the default table for iptables
2. `iptables -P INPUT DROP`
3. `iptables -P FORWARD DROP`
   - Default policy for chain: if the packet goes through the last rule it is dropped
4. `iptables -A INPUT -p tcp --dport 80 -j ACCEPT`
   - On web server: allows incoming TCP connections to the local web server
5. `iptables -A FORWARD -i eth0 -o eth1 -p tcp --dport 80 -j ACCEPT`
   - What does this do?

netfilter, ebtables, ebpf

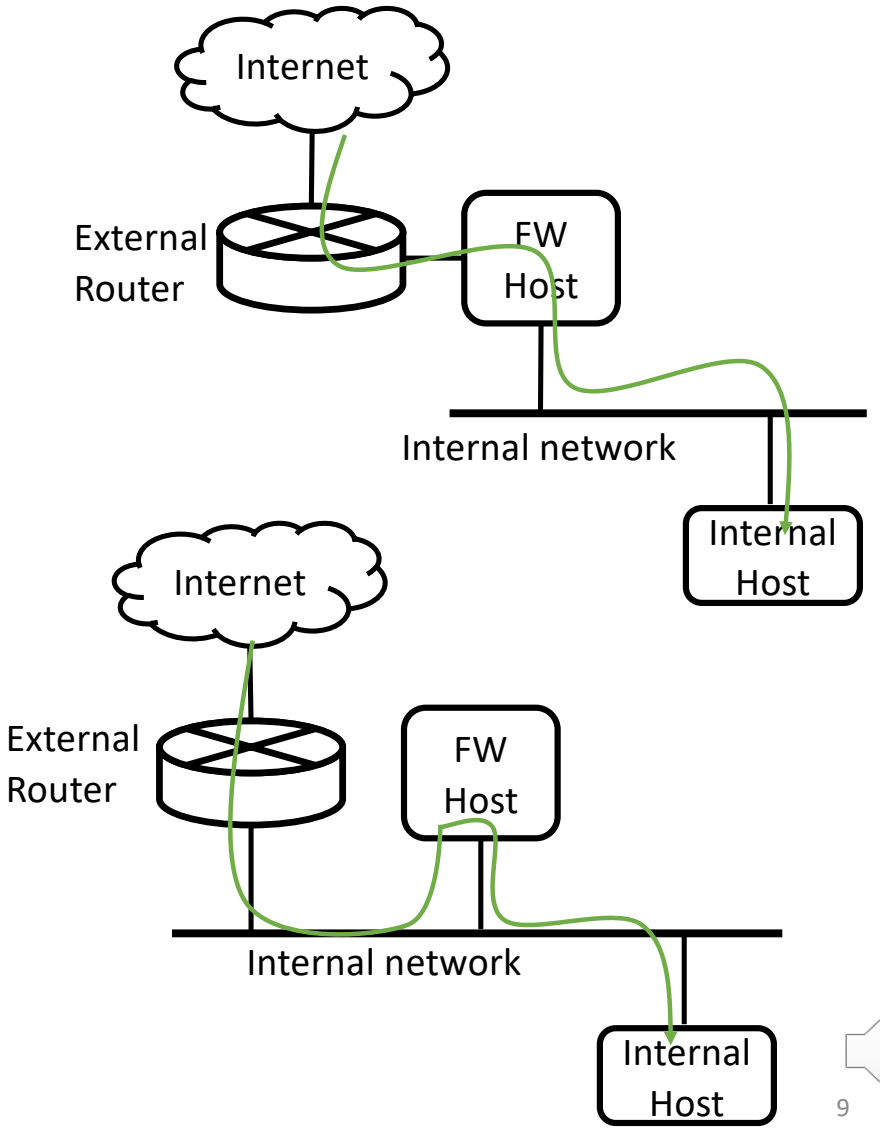
- Netfilter Hooks
  - Triggered when a packet goes through one of INPUT, OUTPUT, FWD, PRER., POSTR.
  - iptables uses netfilter hooks; sets up what to do when each hook function is called
  - https://www.netfilter.org
- Ebtables
  - Ethernet-level equivalent of Iptables for linux bridges
- eBPF
  - Allows sandboxed code to run in the kernel – extend kernel capabilities without recompiling kernel or loading kernel modules
  - Can run anywhere in the kernel (processes, calls, networking)
  - Networking: filtering but also load-balancing and others
  - https://ebpf.io

Stateless vs. stateful

- Stateless: apply rules to each packet individually
  - Filter on IP addresses (source, destination), port numbers, etc
  - This information is available on each packet
  - Can’t do both: accept DNS replies on UDP port 53 and prevent unsolicited responses on same port
- Stateful: remember previous related packets
  - Problem: outbound traffic typically has an inbound response on same TCP stream
  - Solution: stateful firewall accepts incoming packets if they belong to outbound TCP stream
  - Need to store information about which outbound TCP stream is active
  - `iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT`

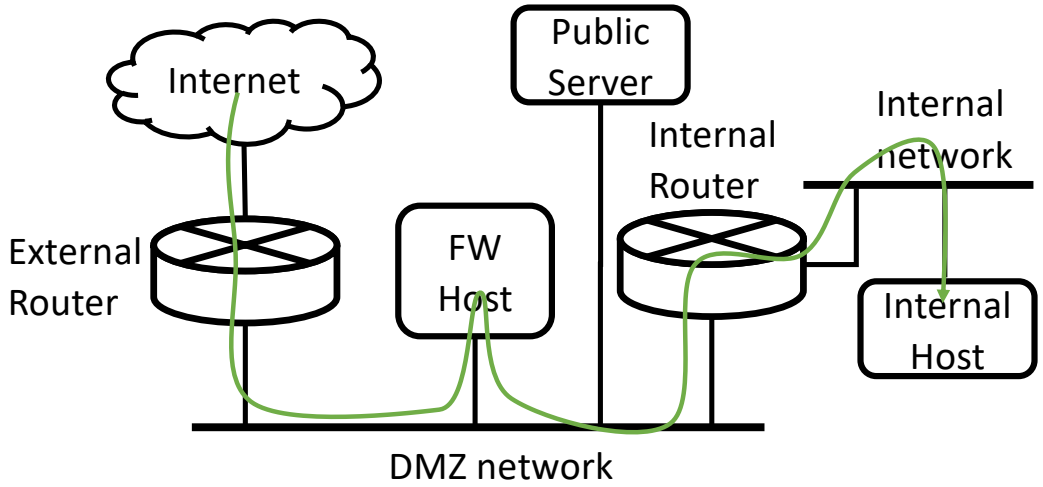
Firewall deployment architectures

- External router:
  - Forces incoming external traffic through the firewall
- Dual-homed gateway
  - external router sends traffic to firewall host with internal and external interfaces; internal interface connected to internal LAN
- Screening router, bastion host
  - external router routes all traffic to single interface firewall host



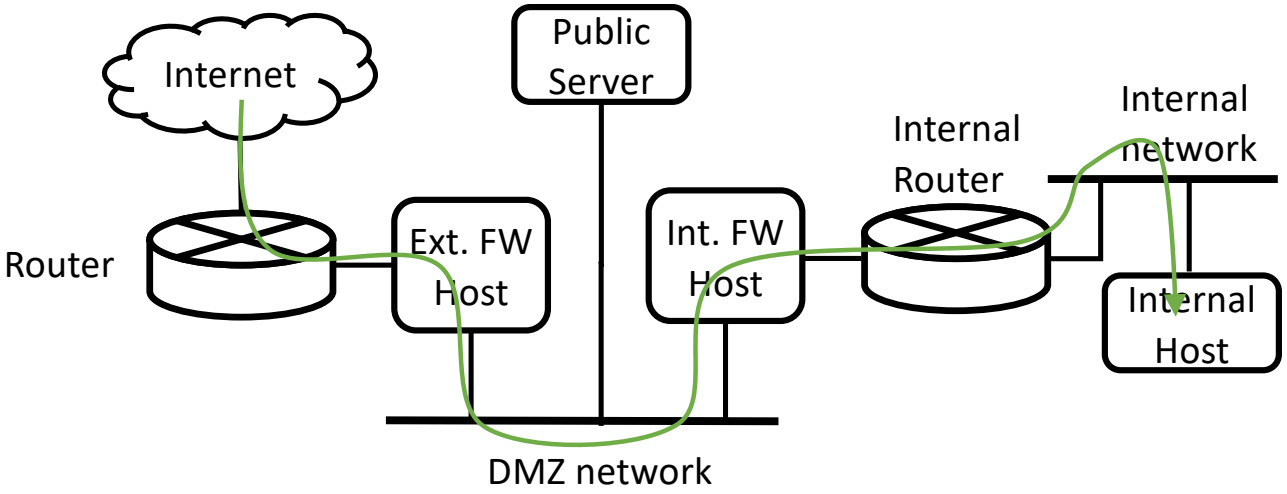
Firewall deployment architectures

- Internal router: forces outgoing internal traffic through the firewall
- Screened network: like dual-homed gateway except internal firewall interface connected to DMZ network including internal router



Firewall deployment architectures

- Dual firewall: like screened network with second firewall before internal router

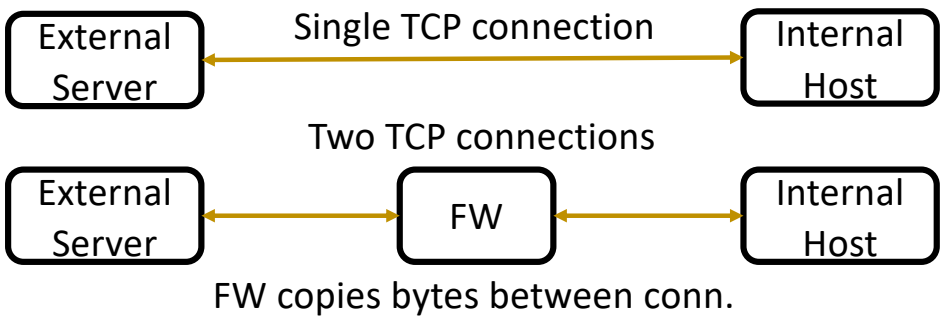


Other types of firewall

- Packet level firewall
- Transport level / circuit gateway firewall
- Application layer firewall

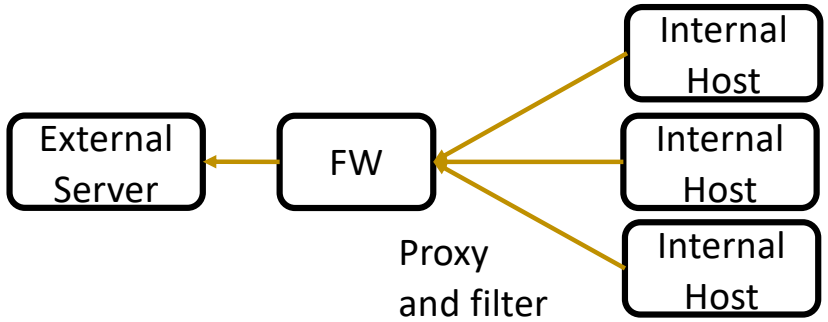
Transport / circuit gateway firewall

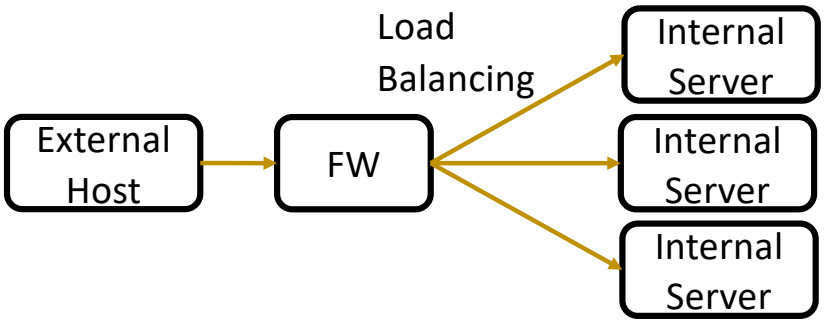
- Interrupts direct TCP connection to the external host at the firewall
- Firewall acts like server to internal host ; acts like client to external server



Transport / circuit gateway firewall

- TCP connection end-point at firewall
  - Egress: proxy and filtering of Internet access
  - Ingress: load balancing, redirects TCP requests to different internal servers





Transport / circuit gateway firewall

- Proxy, two modes:
  - Transparent: TCP wrapper, authorizes and redirects TCP connections
  - Non-transparent: SOCKS, negotiates access to destination server with firewall, sends destination IP address, port, etc to SOCKS server
- Possible mismatch between the firewall IP address seen at the server and information provided by the client, similar to NAT

Application layer firewall

- Deep packet inspection OR proxy
  - DPI: inspect data in TCP streams
  - Proxy: copies or modifies data in the transport stream
- Open-ended and application-specific
  - Squid proxy caches and filters specific HTTP content and URLs
  - Web application firewall filters SQL injection commands
  - Filters malicious downloaded files based on file hash blacklist
- Example operations:
  - Access control
  - Content filtering and modifying
  - Content logging and caching
- Not very different from Intrusion Detection System (more on this next)
- https://docs.nginx.com/nginx-waf/
- https://suricata.io

Control over non-essential traffic

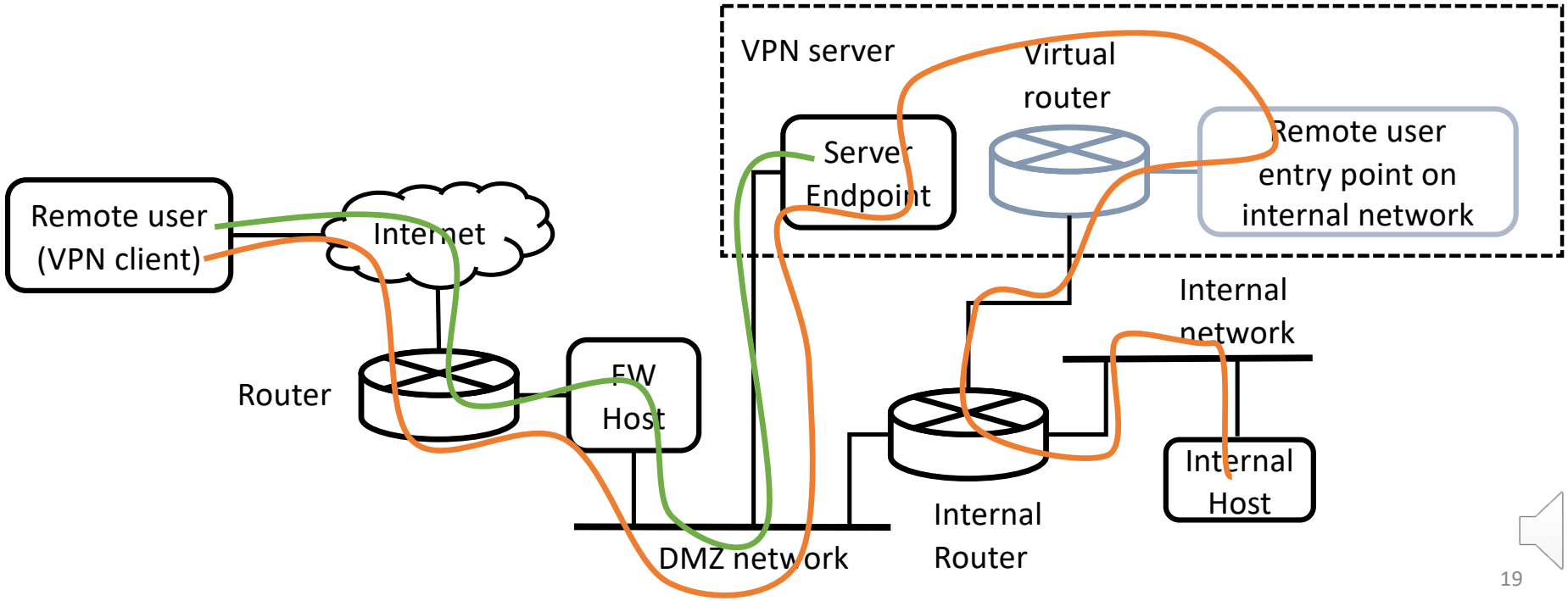
- Apply strict access control
  - Allow only essential traffic
  - To curb malicious traffic that may pose as legitimate yet unessential traffic
- DNS, VPN, some HTTP, ?
- Should you allow encrypted traffic through the firewall?
  - HTTPS, SSH, etc?
- Could malicious traffic also pose as legitimate traffic?

VPN

- Virtual networks built on top of other networks
  - IP-over-SDH, MPLS, IP-over-GRE, P2P networks, IPsec, TLS
  - Cryptographically protecting content (headers, payload)
  - Authentication using symmetric keys or asymmetric keys
- Entry-point into corporate network, for remote users
  - Host-to-net, secure connection from user device to corporate network
  - Potential point of entry that is physically outside the organization
- Service providers offering network access to clients
  - Personal VPN to hide the content of what you're browsing from your ISP
- For an attacker, secure connection to the victim's network

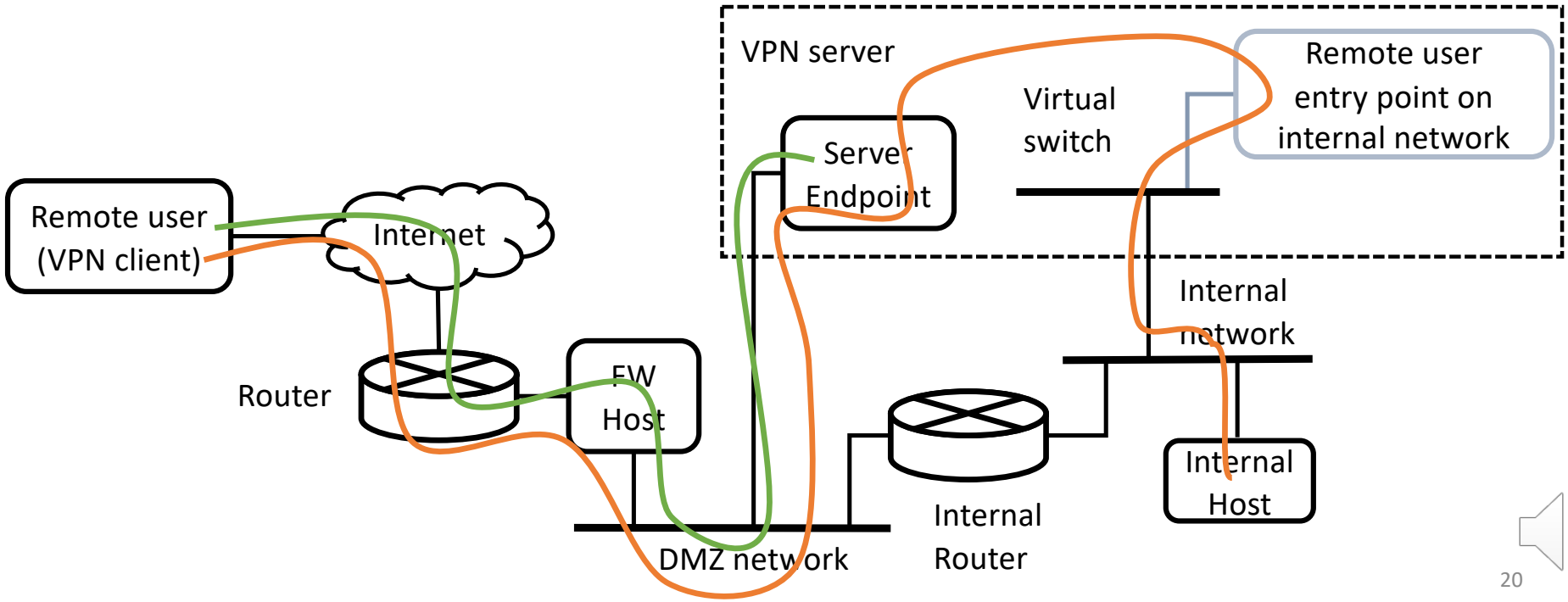
Types of host-to-net VPN

- Option 1 - Tunnel with IP route. VPN provides DHCP-like IP address, acts as IP gateway – no layer 2 traffic



Types of host-to-net VPN

- Option 2 - Ethernet bridge tunnel. VPN forwards Ethernet frames between remote host and corporate network

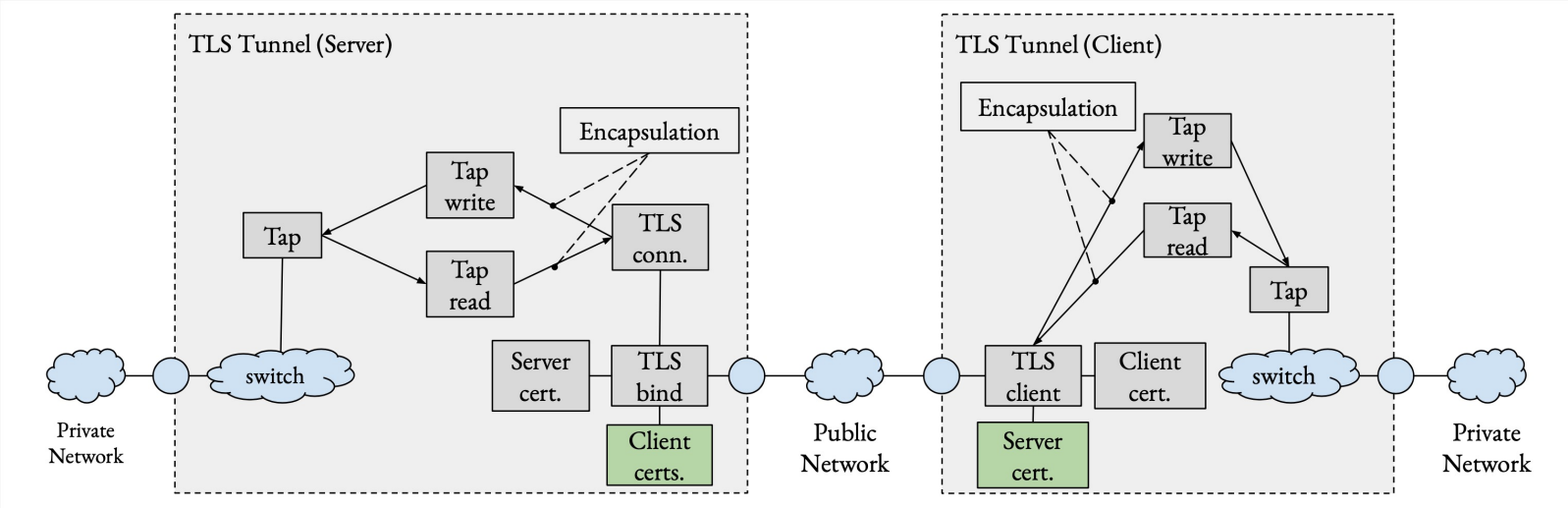


VPN Threat model

- Firewall configured to allow VPN traffic through
  - Allows external access to internal network
  - Threats?
- VPN authentication vulnerable?
  - Vulnerable protocol?
  - Vulnerable authentication?
    - username and password – weak passwords
    - user key pair – harder to compromise
- User keys compromised?
  - Attacker has access – can we still detect it? User profiling (IPs, other info)
- Apparently legitimate traffic used for accessing the external network?

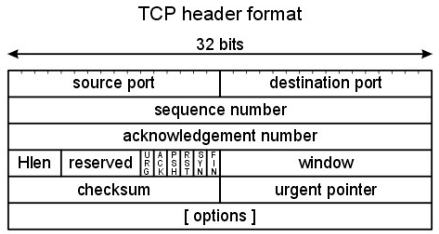
Non-conventional VPNs

- TAP/TUN based VPNs
  - Capture network packets at one tunnel endpoint
  - Encapsulate packets in tunnel packets, send over secure connection (e.g. TLS)
  - Receive, decapsulate, inject in the network interface of the other endpoint



Non-conventional VPNs

- Tunnels over non-VPN protocols like DNS, HTTP, SMTP, Facebook
  - Base64 encode the captured packet
  - Create HTTP GET request with base64 content in header
  - Issue GET request to server
  - Server decodes header, injects packet in network
  - Server to client? Polling



Encode packet in base64

```
Ckluc2NyZXZlLXRlIG5vIG5vc3NvIHNlcnZpZG9yISAKU2Vyw6EgbmVjZXNzw6FyaW8gY29uZmly
bWFyIHF1ZSDDqXMgZXN0dWRhbnRlIGRhIFVuaXZlcnNpZGFkZSBkbyBQb3J0byB1c2FuZG8gbyB0
ZXUgZW1haWwgaW5zdGl0dWNpb25hbC4KCkVzdGUgY29udml0ZSBzw7MgdGUgZMOhIGFjZXNzbyB0
ZW1wb3LDoXJpbyBhbyBzZXJ2aWRvci4gRW5xdWFudG8gbsOjbyB0ZSBhdXRlbnRpY2FyZXMsIG7D
o28gdGVyw6FzIGFjZXNzbyBwZXJtYW5lbnRlLgpTZSBmb3JlcyByZW1vdmlkbyBkYSBwbGF0YWZv
cm1hIGR1cmFudGUgbyBwcm9jZXNzbyBkZSBhdXRlbnRpY2HDp8OjbywgcG9kZXMgc2VtcHJlIHZv
bHRhciBhIGVudHJhciBhdHJhdsOpcyBkbyBsaW5rOgoKaHR0cHM6Ly9kaXNjb3JkLmdnL1hweVc4
RjRZcnYK
```

Send base64 encoded packet as header field in GET request

```
GET /auth/control.php HTTP/1.1
Host: php.testsparker.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
Cookie: PHPSESSID=bacda566f2c061dab81b61d2cd17eb60
Referer: http://php.testsparker.com/auth/login.php
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
X-Scanner: Netsparker
```

Security of Networks, Services, and Systems

Firewalls and VPNs

Ricardo Morla

FEUP – SSR/M.EEC, SR/M.EIC