Virtual Private Networks (VPNs)

Overview, Properties, IPsec, WireGuard

13 October 2020

Network Security AS 2020

Markus Legner

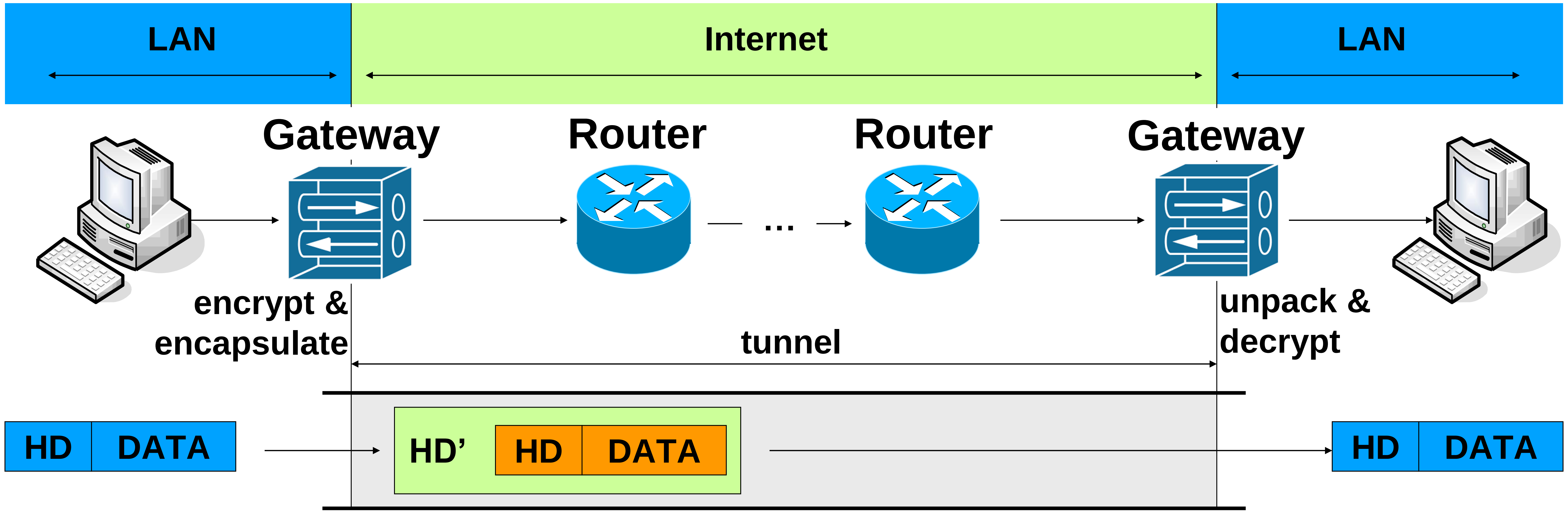
ETH zürich

Where are we in the course?

- Preliminaries and background
  - Networking basics
  - Cryptography
  - Public Key Infrastructures
- Securing the transport layer
  - TLS protocol and ecosystem
  - Attacks on TLS
  - TLS 1.3
- Today: securing the network layer
  - Concepts and use cases of VPNs
  - Example protocols: IPsec, WireGuard

What are VPNs and where are they useful?

A VPN creates a **secure channel** between two networks over an **untrusted network** (the Internet)



- Set-up phase: the tunnel endpoints **authenticate** each other and **set up keys** (similar to TLS handshake)
- Tunneling phase:
  - Packets are encapsulated at the first endpoint and decapsulated at the second
  - The original packet is (often) encrypted and authenticated with a MAC

Typical properties of VPN tunnels

- Similar security properties as the TLS record protocol:
  - Authentication of the source, integrity (MACs)
  - Confidentiality (symmetric encryption)
  - Replay suppression (sequence numbers)
- Some tunneling protocols do not provide encryption or authentication

Typical VPN setups

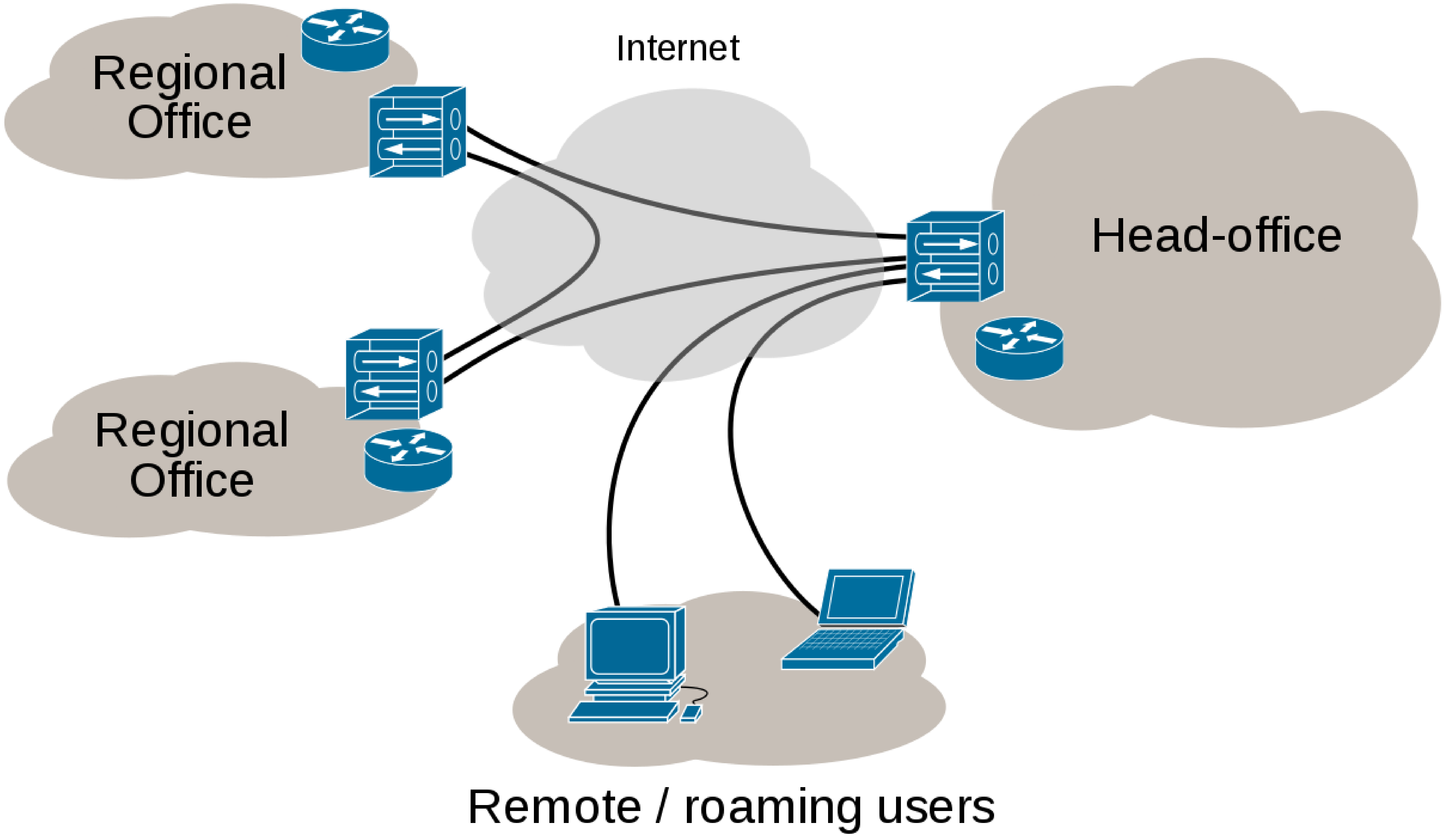
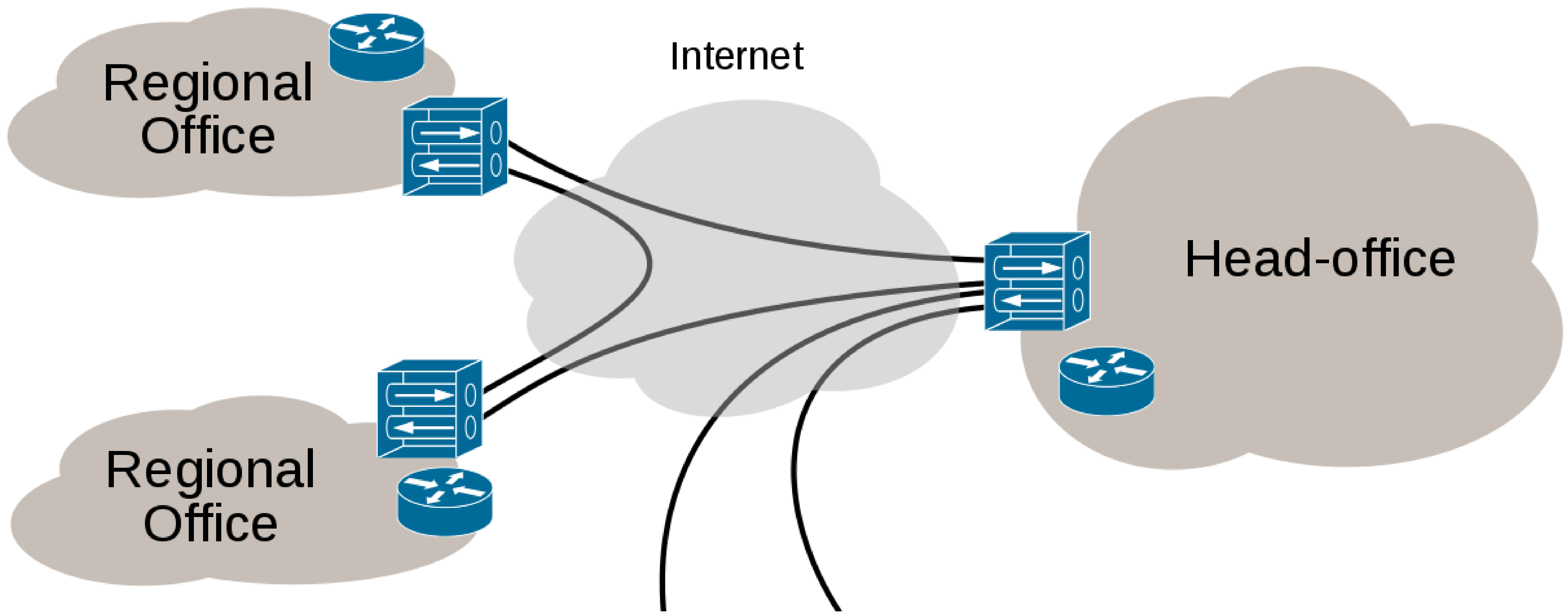


Image credits: Ludovic.ferre https://commons.wikimedia.org/wiki/File:Virtual_Private_Network_overview.svg (CC BY-SA 4.0)

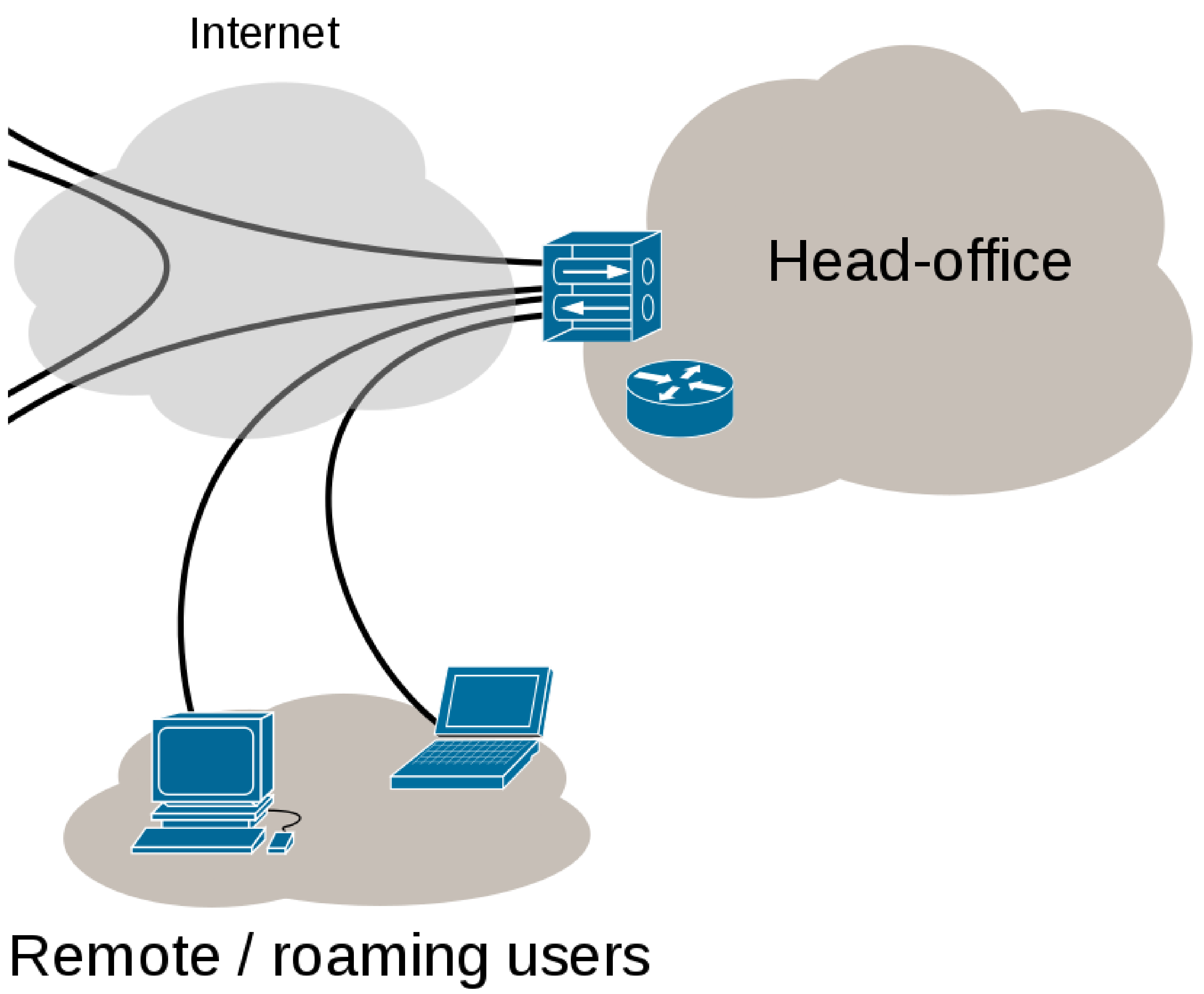
VPN setup 1: secure connection between two physically separated networks (site-to-site)

- Replaces private physical networks and leased lines
  - Even for leased lines, encryption may be desirable
- Used for example to connect regional offices with the head office of a company over the Internet



VPN setup 2: secure connection of a remote host to company/university network (host-to-site)

- Remote host can access resources in private network
  - Private IP addresses can be accessed without port forwarding, etc.
  - Services do not need to be exposed to the Internet
- All traffic between host and private network is secure



VPN setup 3: VPN as a “secure” proxy

- Circumvent censorship (e.g., access facebook in China)
- Avoid tracking by your ISP or in a public WiFi network
- Hide your IP address from websites
- Spoof your location for online shopping, video streaming, etc.
- Access restricted content (e.g., academic journals through ETH)
- Download torrents (only legal ones of course)

Important: VPN provider has access to metadata of all traffic.

VPN ≠ anonymity

- VPNs provide some limited anonymity properties:
  - Local network and ISP only see that you send traffic through some VPN
    - They do not see which websites you access
  - Web servers do not see your real IP address
    - Of course, if you use cookies or log in, anonymity is lost
- VPN server can monitor and record all traffic

See next lecture for details of anonymous communication

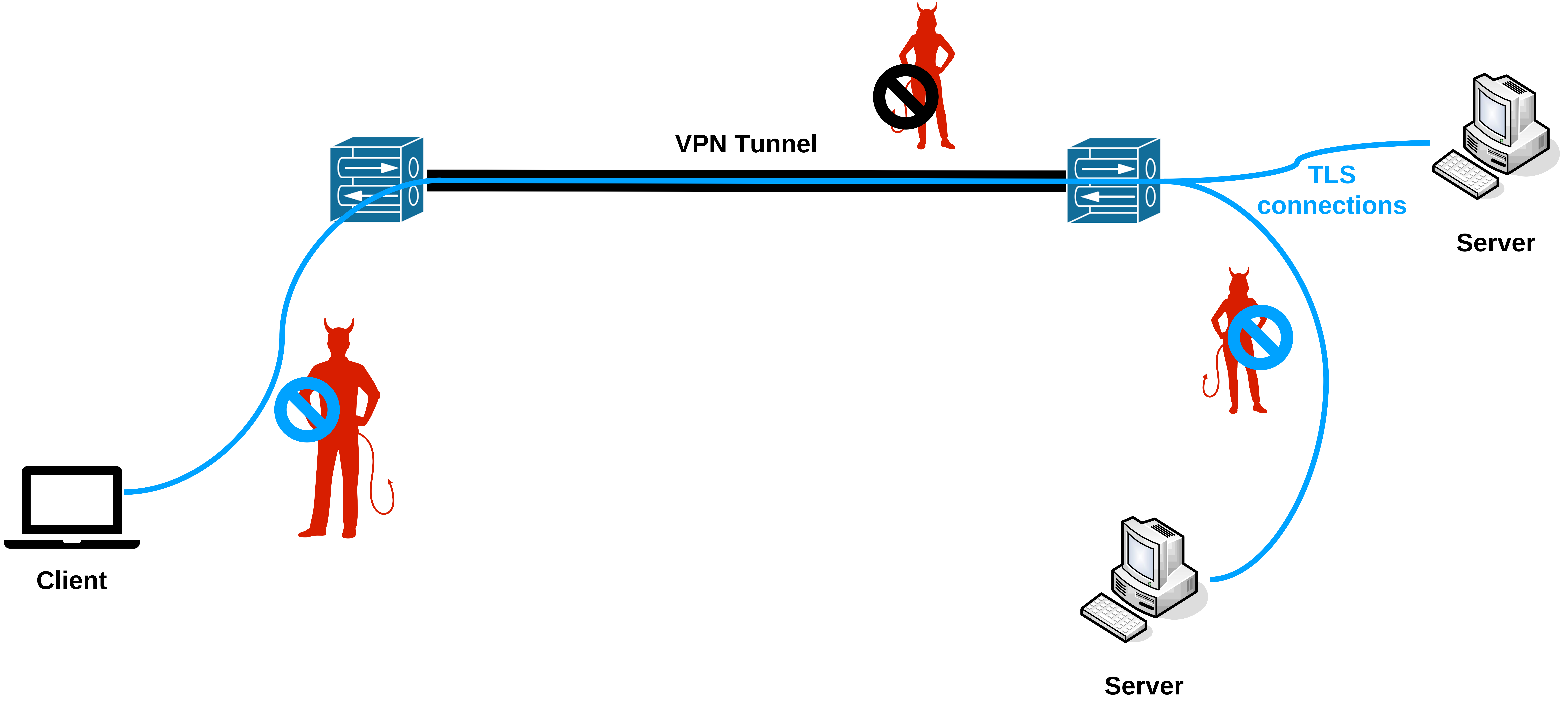
VPN vs. TLS

- Why do we need VPNs when we have TLS?
  - VPNs protect all traffic: “blanket” security
    - DNS requests
    - Access to webservers without TLS
  - VPNs can give access to services in private networks or behind firewalls
- Why do we need TLS when we have VPNs?
  - Data is only secure in the tunnel: no security outside of it
  - VPN server can see all unencrypted traffic → TLS still necessary
  - With a VPN it is not possible to authenticate the webserver, only the tunnel endpoint

Typical properties of VPNs and TLS

	TLS	VPN
Secured layer	**Transport layer** (L4)	**Link/network layer** (L2/3)
Protection	End-to-end	Tunnel
Client authentication	Not authenticated	Authenticated
Diversity	One / very few globally accepted standards	Many different protocols

Typical interaction of TLS and VPN



VPNs and availability/performance

- VPNs can negatively impact performance
  - Additional cryptographic operations
  - Potential detours
  - Limited bandwidth at VPN server
- Generally, VPNs do not provide higher availability
  - No built-in defense against (D)DoS attacks or routing attacks
- VPNs can defend against targeted packet filtering
  - Routers can recognize VPN packets but not content
  - Would need to drop all VPN packets

Countries that have (partially) banned VPNs

https://protonvpn.com/blog/are-vpns-illegal/

VPN vs. VLAN (virtual local area network)

- VPN: (securely) **connect/combine** two different networks
  - One virtual network over multiple physical networks
- VLAN: set up multiple **isolated virtual networks** on a single physical infrastructure
  - Multiple virtual networks over one physical network
  - Often used in cloud-computing environments for isolating communication between VMs
- VXLAN (virtual extensible LAN) combines both features

What types of VPNs are there?

Many different protocols and applications are used to implement VPN functionality

- Generic Routing Encapsulation (GRE) [1994]
- Internet Protocol Security (IPsec) [1995]
- Point-to-point tunneling protocol (PPTP) [1999]
- Layer-2 tunneling protocol (L2TP) [1999]
- OpenVPN [2001]
- Secure Socket Tunneling Protocol (SSTP) [2007]
- WireGuard [2016]
- …

Why are there so many VPN applications but only TLS?

Differences between systems and configuration parameters

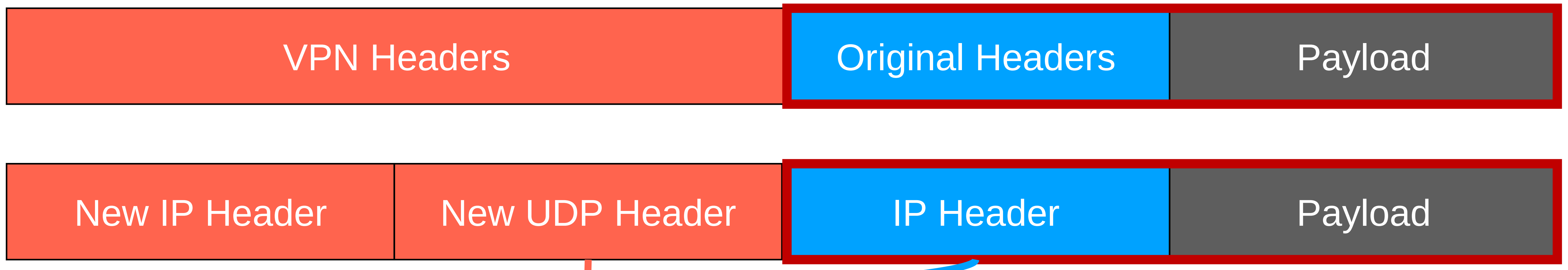
- Authentication mechanisms
  - Pre-shared key (PSK)
  - Public keys and certificates
  - Client: username/password

- Tunneling mechanism (tunnel protocol)
  - Custom protocols (IPsec)
  - Tunnel over TLS (SSTP)

- Layer of connected networks (inner protocol)
  - Layer 3 (Network layer)
  - Layer 2 (Link layer)

- Implementation
  - User space
  - Kernel module
  - Hardware

Tunnel vs. inner protocol



- Inner protocol
  - *Lowest* layer of encapsulated packet (outermost header)
  - Typically layer 2/3
- Tunnel (outer) protocol
  - *Highest* layer used in additional header (innermost header)
  - Typically network layer (IPsec) or transport layer (OpenVPN, WireGuard)

VPN endpoint at hosts

- VPN creates virtual network adapter (e.g., “tun0”, “eth2”, …)
- Can be used like any other network adapter
- VPN interface can be used for all traffic or only selectively (“split tunnel”)

```
markus@mlegner:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     256    0        0 eth0
129.132.99.164  192.168.1.1     255.255.255.255 U     0      0        0 eth0
0.0.0.0         10.6.208.1      0.0.0.0         U     0      0        0 tun0
10.6.208.0      0.0.0.0         255.255.240.0   U     0      0        0 tun0
```

New default route through VPN

Public IP of VPN server

VPN subnet without gateway

VPN processing at hosts


```
markus@mlegner:~$ route -n
Kernel IP routing table
Destination     Gateway       Iface
192.168.1.0     0.0.0.0       eth0
129.132.99.164  192.168.1.1   eth0
0.0.0.0         10.6.208.1    tun0
10.6.208.0      0.0.0.0       tun0
```

IPsec: securing the network layer since 1995

IPsec is a very large and complicated protocol

- First standardized in 1995 in RFCs 1825–1829
- Described in more than 50 different RFCs
- Multiple different modes
- A myriad of options for cipher suites, authentication mechanisms, etc.

We will focus on common cases and abstract from some details.

IPsec is a very large and complicated protocol

Standards track [edit]

- RFC 1829: The ESP DES-CBC Transform
- RFC 2403: The Use of HMAC-MD5-96 within ESP and AH
- RFC 2404: The Use of HMAC-SHA-1-96 within ESP and AH
- RFC 2405: The ESP DES-CBC Cipher Algorithm With Explicit IV
- RFC 2410: The NULL Encryption Algorithm and Its Use With IPsec
- RFC 2451: The ESP CBC-Mode Cipher Algorithms
- RFC 2857: The Use of HMAC-RIPEMD-160-96 within ESP and AH
- RFC 3526: More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)
- RFC 3602: The AES-CBC Cipher Algorithm and Its Use with IPsec
- RFC 3686: Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)
- RFC 3947: Negotiation of NAT-Traversal in the IKE
- RFC 3948: UDP Encapsulation of IPsec ESP Packets
- RFC 4106: The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)
- RFC 4301: Security Architecture for the Internet Protocol
- RFC 4302: IP Authentication Header
- RFC 4303: IP Encapsulating Security Payload
- RFC 4304: Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association and Key Management Protocol (ISAKMP)
- RFC 4307: Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)
- RFC 4308: Cryptographic Suites for IPsec
- RFC 4309: Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)
- RFC 4543: The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH
- RFC 4555: IKEv2 Mobility and Multihoming Protocol (MOBIKE)
- RFC 4806: Online Certificate Status Protocol (OCSP) Extensions to IKEv2
- RFC 4868: Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec
- RFC 4945: The Internet IP Security PKI Profile of IKEv1/ISAKMP, IKEv2, and PKIX
- RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile
- RFC 5282: Using Authenticated Encryption Algorithms with the Encrypted Payload of the Internet Key Exchange version 2 (IKEv2) Protocol
- RFC 5386: Better-Than-Nothing Security: An Unauthenticated Mode of IPsec
- RFC 5529: Modes of Operation for Camellia for Use with IPsec
- RFC 5685: Redirect Mechanism for the Internet Key Exchange Protocol Version 2 (IKEv2)
- RFC 5723: Internet Key Exchange Protocol Version 2 (IKEv2) Session Resumption
- RFC 5857: IKEv2 Extensions to Support Robust Header Compression over IPsec
- RFC 5858: IPsec Extensions to Support Robust Header Compression over IPsec
- RFC 7296: Internet Key Exchange Protocol Version 2 (IKEv2)
- RFC 7321: Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)
- RFC 7383: Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation
- RFC 7427: Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)
- RFC 7634: ChaCha20, Poly1305, and Their Use in the Internet Key Exchange Protocol (IKE) and IPsec

Experimental RFCs [edit]

- RFC 4478: Repeated Authentication in Internet Key Exchange (IKEv2) Protocol

Informational RFCs [edit]

- RFC 2367: PF_KEY Interface
- RFC 2412: The OAKLEY Key Determination Protocol
- RFC 3706: A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers
- RFC 3715: IPsec-Network Address Translation (NAT) Compatibility Requirements
- RFC 4621: Design of the IKEv2 Mobility and Multihoming (MOBIKE) Protocol
- RFC 4809: Requirements for an IPsec Certificate Management Profile
- RFC 5387: Problem and Applicability Statement for Better-Than-Nothing Security (BTNS)
- RFC 5856: Integration of Robust Header Compression over IPsec Security Associations
- RFC 5930: Using Advanced Encryption Standard Counter Mode (AES-CTR) with the Internet Key Exchange version 02 (IKEv2) Protocol
- RFC 6027: IPsec Cluster Problem Statement
- RFC 6071: IPsec and IKE Document Roadmap
- RFC 6379: Suite B Cryptographic Suites for IPsec
- RFC 6380: Suite B Profile for Internet Protocol Security (IPsec)
- RFC 6467: Secure Password Framework for Internet Key Exchange Version 2 (IKEv2)

Best current practice RFCs [edit]

- RFC 5406: Guidelines for Specifying the Use of IPsec Version 2

Obsolete/historic RFCs [edit]

- RFC 1825: Security Architecture for the Internet Protocol (obsoleted by RFC 2401)
- RFC 1826: IP Authentication Header (obsoleted by RFC 2402)
- RFC 1827: IP Encapsulating Security Payload (ESP) (obsoleted by RFC 2406)
- RFC 1828: IP Authentication using Keyed MD5 (historic)
- RFC 2401: Security Architecture for the Internet Protocol (IPsec overview) (obsoleted by RFC 4301)
- RFC 2406: IP Encapsulating Security Payload (ESP) (obsoleted by RFC 4303 and RFC 4305)
- RFC 2407: The Internet IP Security Domain of Interpretation for ISAKMP (obsoleted by RFC 4306)
- RFC 2409: The Internet Key Exchange (obsoleted by RFC 4306)
- RFC 4305: Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH) (obsoleted by RFC 4835)
- RFC 4306: Internet Key Exchange (IKEv2) Protocol (obsoleted by RFC 5996)
- RFC 4718: IKEv2 Clarifications and Implementation Guidelines (obsoleted by RFC 7296)
- RFC 4835: Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH) (obsoleted by RFC 7321)
- RFC 5996: Internet Key Exchange Protocol Version 2 (IKEv2) (obsoleted by RFC 7296)

https://en.wikipedia.org/wiki/IPsec

A typical IPsec session

Set up a security association (SA) via IKE

Encapsulate packets and tunnel them between SA endpoints

TLS Handshake

TLS Record Protocol

A typical IPsec session

Set up a security association (SA) via IKE

Encapsulate packets and tunnel them between SA endpoints

Internet key exchange (IKEv2)

Initiator (I)

Cryptographic algorithms, etc. for rest of IKE (similar to cipher suite in TLS)

Responder (R)

IKE_SA_INIT request: HDR(ConnID, Flags, …), proposal1, g^x, NonceI

IKE_SA_INIT response: HDR(ConnID, Flags, …), accept1, g^y, NonceR

Calculate SKEYSEED based on NonceI, NonceR, g^{xy}
Generate SA (keys SK, configuration, …) for rest of IKE exchange

IKE_AUTH: HDR, SK{ "I", [cert], auth{IKE_SA_INIT request, NonceR}, proposal2 }

IKE_AUTH: HDR, SK{ "R", [cert], auth{IKE_SA_INIT response, NonceI}, accept2 }

Identity of initiator

Signature (when using public keys) or MAC (when using pre-shared key)

Cryptographic algorithms, etc. for IPsec tunnel

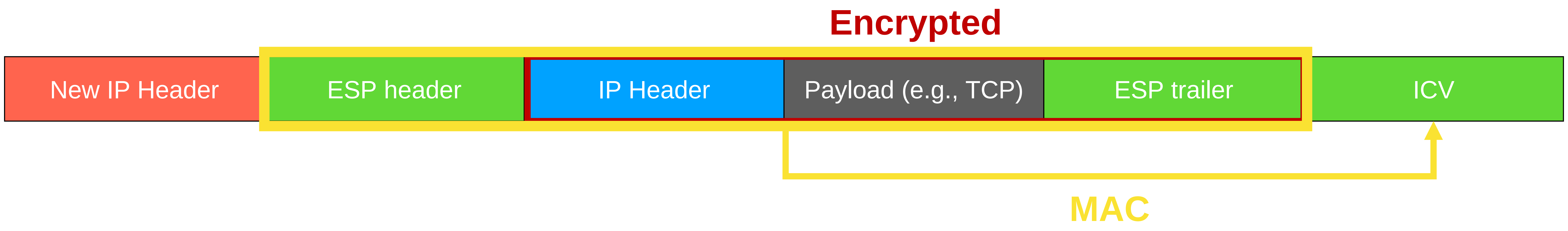
Generate SA (keys, configuration, …) for IPsec tunnel

A typical IPsec session

Set up a security association (SA) via IKE

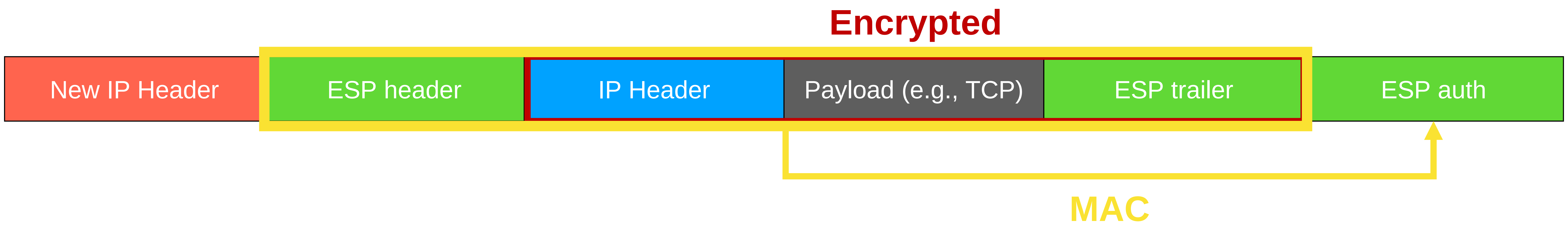
Encapsulate packets and tunnel them between SA endpoints

IPsec encapsulation security payload (ESP) in tunneling mode



- Add ESP trailer: Padding, type encapsulated (original) packet
- Encrypt packet and trailer
- Add ESP header: SA identification, sequence number
- Create Integrity Check Value (ICV): MAC over original packet, ESP header, ESP trailer
- Add new IP header

IPsec decapsulation and decryption



- Strip off outer IP header
- Look up keys and configuration using information in ESP header
- Check MAC
- Strip off authentication tag and ESP header
- Decrypt original packet
- Remove ESP trailer
- Forward original packet

IPsec and TLS: similarities and differences

	TLS	**IPsec**
Key exchange	TLS handshake	Additional protocol: Internet Key Exchange (IKE)
Authentication	Typically only server Typically using RSA certificate	Server and client Many different authentication mechanisms
Underlying transport	Reliable (runs on top of TCP)	Best-effort (runs on top of IP)

IKE and IPsec have many additional options

- Additional messages with IKEv2:
  - If Extensible Authentication Protocol (EAP) is used
    - For example, if username/password authentication required
  - If initiator chooses unsupported Diffie–Hellman group
  - Responder can use cookies for DoS defense
- Other IPsec modes:
  - Transport mode (instead of tunneling) for end-to-end connections
  - Authenticated Header (AH) protocol (instead of ESP) for only authentication but no encryption

Problems with IPsec

- Configuration is difficult and error-prone due to many options
- Some options do not provide any security
  - Possible to use NULL encryption and not use any message authentication
- Insecure ciphers are possible → similar problems as TLS
  - Example: SLOTH attack (Bhargavan & Leurent 2015) on weak hashes
  - Bleichenbacher attack on IKE (Felsch et. al 2018)

The Dangers of Key Reuse: Practical Attacks on IPsec IKE

Dennis Felsch, Martin Grothe, and Jörg Schwenk, *Ruhr-University Bochum;* Adam Czubak and Marcin Szymanek, *University of Opole*

https://www.usenix.org/conference/usenixsecurity18/presentation/felsch

WireGuard: a modern lightweight VPN

Motivation: popular VPN protocols have huge codebases and/or are slow

WireGuard overview

- Designed by Jason A. Donenfeld
- Initial implementation in 2016
- Part of the mainline Linux kernel since January 2020

WireGuard design concepts

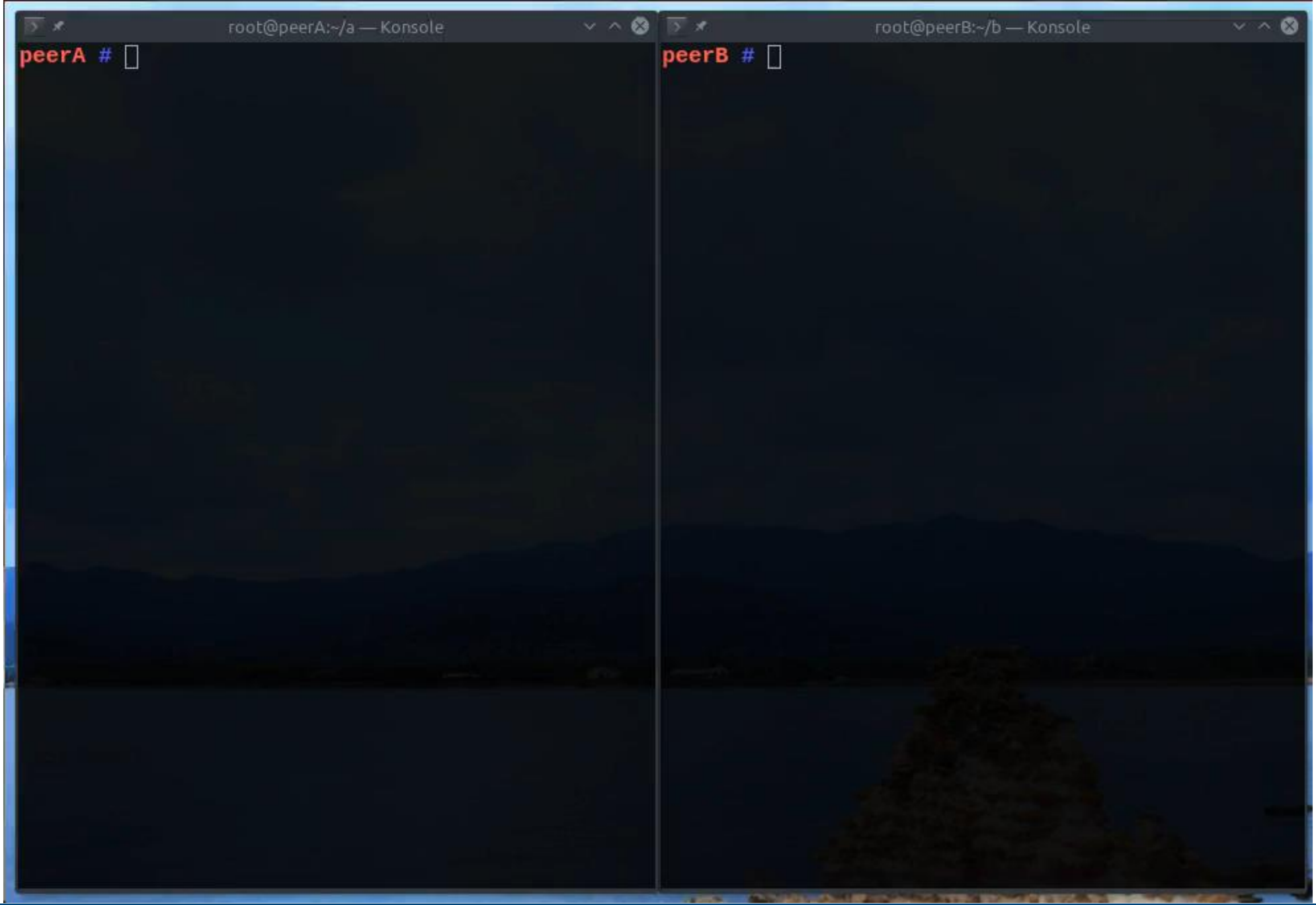
- No cryptographic agility (“cryptographically opinionated”)
  - Only use state-of-the-art primitives: Curve25519 (key exchange), ChaCha20 (encryption), Poly1305 (authentication), …
  - Simplify negotiation and remove insecure primitives
  - Question: What happens if cryptographic primitives are found to be vulnerable?
- Very simple configuration – similar to SSH authorized_keys
- Very small codebase → minimal attack surface, formally verifiable
  - Dowling&Paterson (2018), Kobeissi&Bhargavan (2018), …

Can I just once again state my love for [WireGuard] and hope it gets merged soon? Maybe the code isn't perfect, but I've skimmed it, and compared to the horrors that are OpenVPN and IPSec, it's a work of art.

[Linus Torvalds, https://lists.openwall.net/netdev/2018/08/02/124]

WireGuard authentication and keys

- Handshake follows the *Noise Protocol Framework* (noiseprotocol.org)
  - Built exclusively on (elliptic-curve) Diffie–Hellman exchanges
- Each peer has a static key pair
  - Initiator: S_I^{pub}, S_I^{priv}
  - Responder: S_R^{pub}, S_R^{priv}
- Peers specify in configuration which public keys are authorized
  - Similar to adding ssh keys to the “authorized_keys” file
- Each peer creates ephemeral key pair E_I^{pub}, E_I^{priv}; E_R^{pub}, E_r^{priv}
- Derive symmetric keys from four Diffie–Hellman combinations:
  $\{ DH(S_I, S_R), DH(S_I, E_R), DH(E_I, S_R), DH(E_I, E_R) \}$



https://www.wireguard.com/quickstart/

WireGuard 1-RTT handshake

Initiator (I)

Responder (R)

Static key-pair S_I

Static key-pair S_R

Generate ephemeral key-pair E_I

`S_I^pub, E_I^pub, timestamp, MAC calculated using key derived from DH(S_I,S_R), DH(E_I,S_R)` →

S_I^{pub}, E_I^{pub}, timestamp, MAC calculated using key derived from $DH(S_I,S_R)$, $DH(E_I,S_R)$ →

Check authorization of S_I
Check MACs
Generate ephemeral key-pair E_R

← E_R^{pub}, MAC calculated using key derived from $DH(S_I,S_R)$, $DH(E_I,S_R)$, $DH(E_I,E_R)$, $DH(S_I,E_R)$

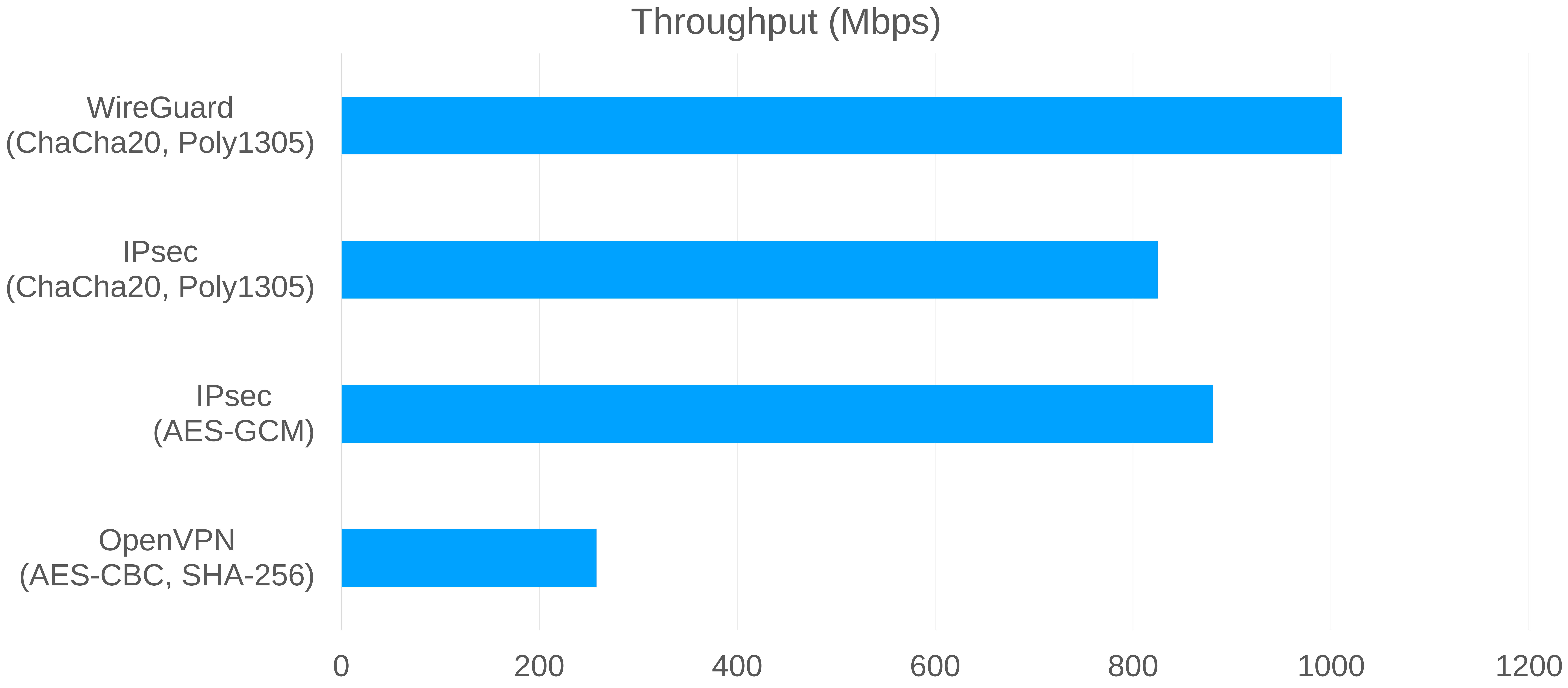
Check MACs
Derive keys for transport from $DH(S_I,S_R)$, $DH(S_I,E_R)$, $DH(E_I,S_R)$, $DH(E_I,E_R)$

Derive keys for transport from $DH(S_I,S_R)$, $DH(S_I,E_R)$, $DH(E_I,S_R)$, $DH(E_I,E_R)$

Encrypted and authenticated data →

← Encrypted and authenticated data

WireGuard is faster than IPsec or OpenVPN



WireGuard whitepaper: https://www.wireguard.com/papers/wireguard.pdf

Summary

What you should remember about VPNs

- VPNs create secure channels on network or link layer
- VPNs and end-to-end security (TLS) complement each other
- Many different VPN protocols and applications
  - IPsec has a long history and numerous configuration options
    - Very versatile but difficult to set up
  - WireGuard is a new VPN protocol with a focus on simplicity
    - Very few configuration parameters, no cryptographic agility
    - Simple to set up
    - Small codebase → small attack surface