Advanced Computer Networking (ACN)

IN2097 – WiSe 2024–2025

Prof. Dr.-Ing. Georg Carle, Sebastian Gallenmüller

Christian Dietze, Max Helm, Benedikt Jaeger,
Marcel Kempf, Jihye Kim, Patrick Sattler

Chair of Network Architectures and Services
School of Computation, Information, and Technology
Technical University of Munich

Tunnel Protocols

Tunnel Protocols

Introduction
Tunneling

Definition

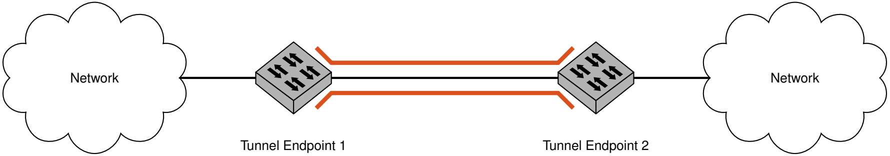
- Tunneling encapsulates one datagram within another datagram.
- The outer packet and its headers are regarded for switching / routing purposes of the underlay network.
- The inner packet is opaque to the underlay network.
- The overlay network handles the inner packet, including switching and routing.
- May be used at any layer of the ISO OSI model.

Possible benefits

- Build overlay structure
- Deal with heterogeneous protocols
- Protect traffic
- Isolate customers (data center)

But ...

- More overhead
- Configuration effort
- MUCH room for misconfiguration



Possible Tunneling Use Cases

What can be achieved with a tunnel?

- Force packet to reach specific node in the network (different path than from regular routing), e.g. using IP-in-IP tunnel - RFC 2003
- Traverse incompatible nodes, e.g. IPv6 tunnel over IPv4 only nodes
- Provide secure connection between different nodes, e.g. using IPsec

Which considerations when using tunneling?

- Performance
  - Processing overhead
  - Packet length overhead: reduced MTU, possible fragmentation, limited visibility to end systems
- Security
  - Correct configuration and tunnel setup not trivial
  - Inner and outer headers need to be verified
  - Tunnels may circumvent security policies (e.g. bypassing filters / firewalls)

Representative Tunneling Technologies

- Traffic management and isolation
  - VLAN
  - MPLS
  - VXLAN
- Secure tunnels
  - IPsec
  - TLS, DTLS
  - Wireguard
  - ssh
  - TOR - Onion Routing Overlay
- Protocol innovation; incremental protocol deployment
  - IP multicast overlays, e.g. “Mbone” (“multicast backbone”)
  - various IPv6 transition technologies
  - Peer-to-Peer overlays

Introduction
Virtual Private Network (VPN)

What is a VPN?

- In general, just another tunneling protocol
- VPNs are usually encrypted
- Provide secured connections between different nodes

Use cases:

- Securely connect different offices to HQ
- Build secure connection from a laptop to a company network
- Anonymization

Introduction
Virtual Private Network (VPN)

What does TUM/LRZ use?

- LRZ offers eduVPN
  - part of the GÉANT project (co-funded by the EU)
  - very easy to configure
  - supports OpenVPN and WireGuard
  - supports split and full tunnel
- Other VPNs LRZ used to offer in the past:
  - Cisco AnyConnect
    - TLS-based signalling
    - DTLS transport of tunneled VPN traffic
    - fallback to TLS-based transport where UDP is blocked
  - Cisco IPsec-based VPN
    - with IKEv1 signalling protocol

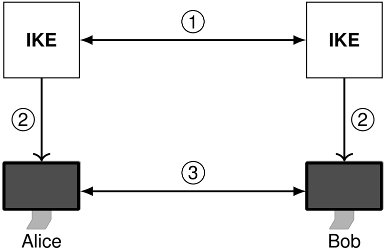
Tunnel Protocols

IPsec
Overview

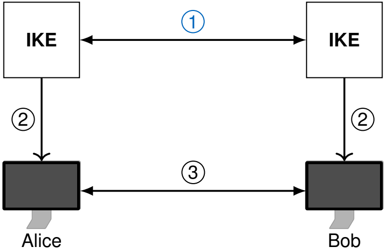
- Standardized by a number of RFCs (most important RFC 4301 [1])
- 2 modes of operation
  - Tunnel Mode: (a) Subnet to Subnet, Endpoints are called Security Gateways, or (b) Host to Security Gateway
  - Transport Mode: Host to Host
- 2 phases of operation
  - Handshake: Establish one or more Security Associations (SA), IPsec signalling protocols that establish SAs: IKEv1 (old), IKEv2
  - Data transfer: Use SAs to send encrypted and/or integrity protected traffic, Protocols used: Encapsulated Security Payload (ESP), Authentication Header (AH)
- Implementations
  - Commercial implementations by major hardware vendors (Cisco, Juniper, Arista, ...)
  - Open Source implementations (IKEv1 / IKEv2 / ESP / AH)
    - IKEv1 (deprecated - don't use it) - implementations include: vpnc
    - IKEv2 (State-of-the-art) - implementations include: strongSwan, libreswan
    - ESP / AH: Linux / FreeBSD kernel
- Usage scenarios
  - Connections between different sites (e.g. branch office to HQ)
  - Connection of client into enterprise network (road warrior scenario)

IPsec

How does IPsec work?



How does IPsec work?

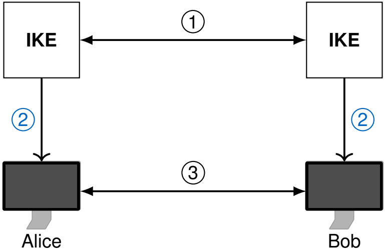


① Authentication, key establishment and negotiation of crypto algorithms

- Possible protocols: ISAKMP, Internet Key Exchange (IKE), IKEv2

IPsec

How does IPsec work?



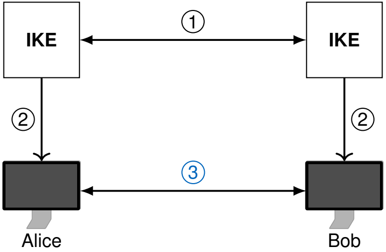
① Authentication, key establishment and negotiation of crypto algorithms

- Possible protocols: ISAKMP, Internet Key Exchange (IKE), IKEv2

② Set keys and cryptographic algorithms

IPsec

How does IPsec work?

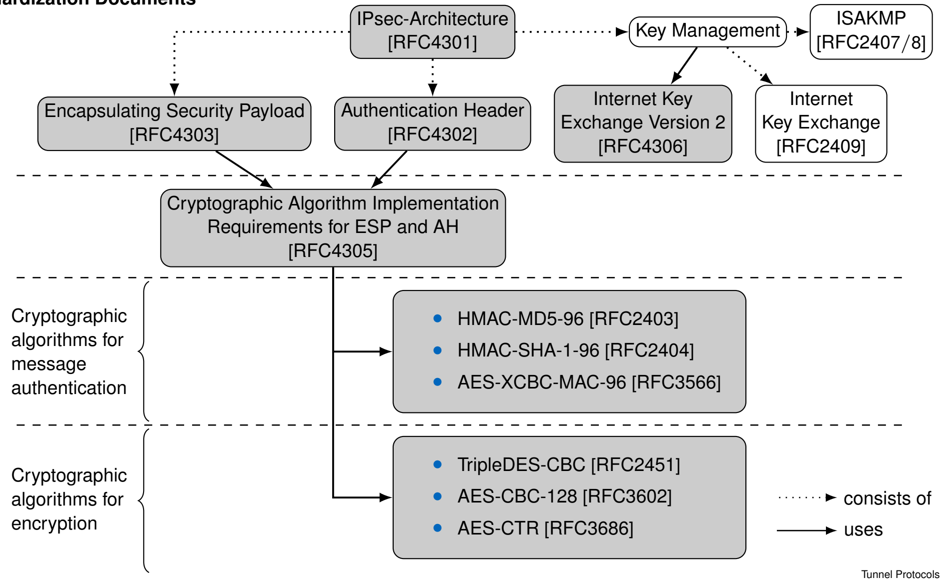


1. Authentication, key establishment and negotiation of crypto algorithms
   - Possible protocols: ISAKMP, Internet Key Exchange (IKE), IKEv2
2. Set keys and cryptographic algorithms
3. Secure channel which provides
   - Data Integrity via Authentication Header (AH) or Encapsulating Security Payload (ESP)
   - Confidentiality using ESP

Note: ESP can provide both data integrity and encryption while AH only provides data integrity

IPsec

IPsec Standardization Documents



IPsec

RFC 4301 defines the basic architecture of IPsec:

- Concepts
  - Security Association (SA) and Security Association Database (SAD)
  - Security Policy (SP) and Security Policy Database (SPD)
- Fundamental Protocols
  - Authentication Header (AH)
  - Encapsulation Security Payload (ESP)
- Protocol Modes
  - Transport Mode
  - Tunnel Mode
- Key Management Protocols
  - ISAKMP, IKE, IKEv2

IPsec

RFC 4301 defines the basic architecture of IPsec:

- Concepts
  - Security Association (SA) and Security Association Database (SAD)
  - Security Policy (SP) and Security Policy Database (SPD)
- Fundamental Protocols
  - Authentication Header (AH)
  - Encapsulation Security Payload (ESP)
- Protocol Modes
  - Transport Mode
  - Tunnel Mode
- Key Management Protocols
  - ISAKMP, IKE, IKEv2

List of IPsec related RFCs: https://datatracker.ietf.org/wg/IPsec/documents/

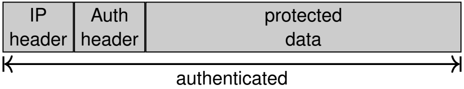
- Most RFCs updated in 2005 after several years of revision
- Support for integration of new crypto primitives for encryption and integrity
- Reduced complexity and better protocol design

IPsec

Available Protocols

Authentication Header (AH):

- Data origin authentication and replay protection
- Inserted between the IP header and the data to be protected

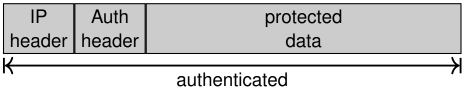


IPsec

Available Protocols

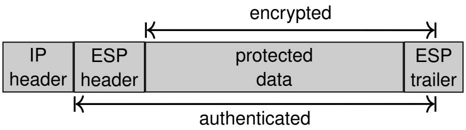
Authentication Header (AH):

- Data origin authentication and replay protection
- Inserted between the IP header and the data to be protected



Encapsulating Security Payload (ESP):

- Data origin authentication, confidentiality and replay protection
- A header and a trailer encapsulating the data to be protected



Key management and setup of Security Associations (SA)

Internet Security Association Key Management Protocol (ISAKMP)

- Defines generic framework for authentication, key exchange and SA parameters [RFC2408]
- Does not define a specific authentication protocol but defines
  - Packet formats
  - Retransmission formats
  - Message construction requirements
- Use of ISAKMP for IPsec is further described in [RFC2407]

Key management and setup of Security Associations (SA)

Internet Security Association Key Management Protocol (ISAKMP)

- Defines generic framework for authentication, key exchange and SA parameters [RFC2408]
- Does not define a specific authentication protocol but defines
  - Packet formats
  - Retransmission formats
  - Message construction requirements
- Use of ISAKMP for IPsec is further described in [RFC2407]

Internet Key Exchange Version 2 [RFC4306]

- Defines an authentication and key exchange protocol
- Reduced complexity by better protocol design and by omitting unnecessary features

IPsec Replay Protection

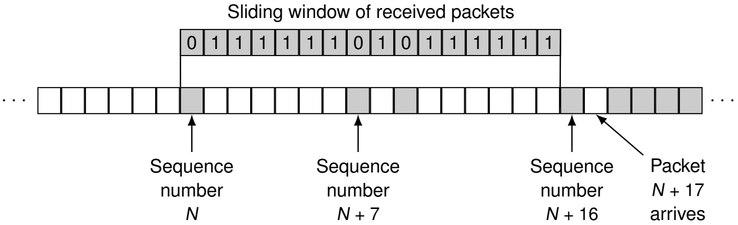
AH- and ESP-protected packets carry a sequence number

- On setup of a Security Association (SA) this sequence number is initialized to zero
- The sequence number is increased with every IP packet sent
- The sequence number is 32-bit long and a new session key is needed before a wrap-around occurs
- The receiver checks if the sequence number is contained in a window of acceptable numbers

IPsec Replay Protection

AH- and ESP-protected packets carry a sequence number

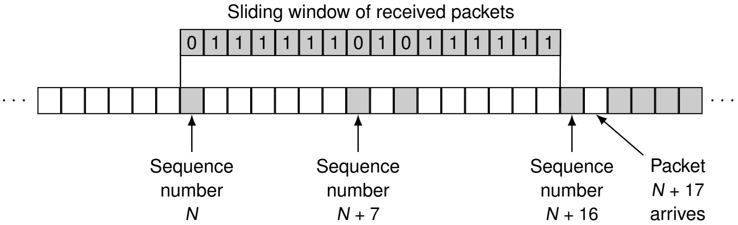
- On setup of a Security Association (SA) this sequence number is initialized to zero
- The sequence number is increased with every IP packet sent
- The sequence number is 32-bit long and a new session key is needed before a wrap-around occurs
- The receiver checks if the sequence number is contained in a window of acceptable numbers



IPsec Replay Protection

AH- and ESP-protected packets carry a sequence number

- On setup of a Security Association (SA) this sequence number is initialized to zero
- The sequence number is increased with every IP packet sent
- The sequence number is 32-bit long and a new session key is needed before a wrap-around occurs
- The receiver checks if the sequence number is contained in a window of acceptable numbers



- Packet with sequence number N can still be accepted
- Window size has to be at least 32 in practice

IPsec Replay Protection

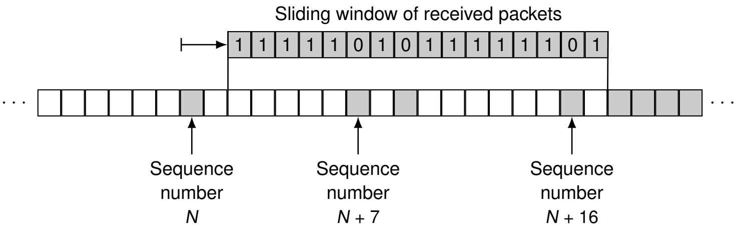
If a received packet has a sequence number which

- is left of the current window $\Rightarrow$ the receiver rejects the packet
- is inside the current window $\Rightarrow$ the receiver accepts the packet and advances the window
- is right of the current window $\Rightarrow$ the receiver accepts the packet and advances the window

IPsec Replay Protection

If a received packet has a sequence number which

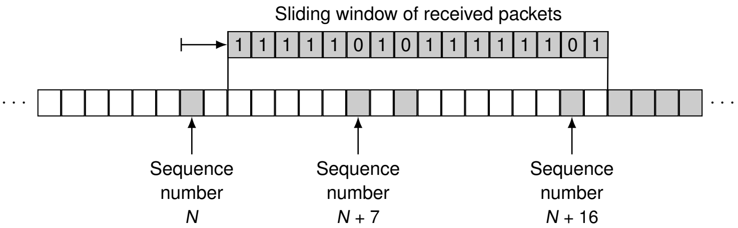
- is left of the current window ⇒ the receiver rejects the packet
- is inside the current window ⇒ the receiver accepts the packet and advances the window
- is right of the current window ⇒ the receiver accepts the packet and advances the window



IPsec Replay Protection

If a received packet has a sequence number which

- is left of the current window ⇒ the receiver rejects the packet
- is inside the current window ⇒ the receiver accepts the packet and advances the window
- is right of the current window ⇒ the receiver accepts the packet and advances the window



- Packet with sequence number *N* can no longer be accepted
- IP packets are only accepted after successful authentication verification
- The window is never advance before this verification

IPsec Security Protocol Modes

Transport Mode

- Only usable between communication endpoints
  - Host ↔ Host
  - Host ↔ Gateway (e.g. Gateway is end-point for network management)
- Adds a security specific header (+ trailer if ESP is employed)

IP header	IPsec header	protected data

IPsec Security Protocol Modes

Transport Mode

- Only usable between communication endpoints
  - Host ↔ Host
  - Host ↔ Gateway (e.g. Gateway is end-point for network management)
- Adds a security specific header (+ trailer if ESP is employed)

IP header	IPsec header	protected data

Tunnel Mode

- Usable with arbitrary peers
- Encapsulates IP packets

IP header	IPsec header	IP header	protected data

- Allows for e.g. a gateway, protecting traffic on behalf of hosts in subnetwork

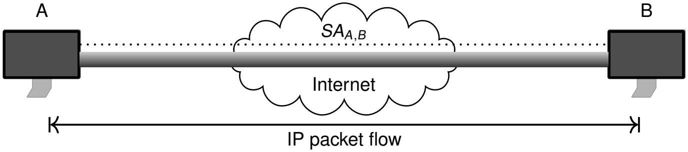
IPsec Security Protocol Modes

Transport Mode

- Used when the *cryptographic endpoints* are also the *communication endpoints* of the secured packets
  - Cryptographic Endpoints: Entities that process IPsec headers
  - Communication Endpoints: Source and destination of an IP packet

Transport Mode

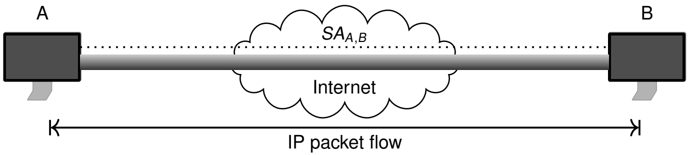
- Used when the *cryptographic endpoints* are also the *communication endpoints* of the secured packets
  - Cryptographic Endpoints: Entities that process IPsec headers
  - Communication Endpoints: Source and destination of an IP packet



IPsec Security Protocol Modes

Transport Mode

- Used when the *cryptographic endpoints* are also the *communication endpoints* of the secured packets
  - Cryptographic Endpoints: Entities that process IPsec headers
  - Communication Endpoints: Source and destination of an IP packet



- In most cases, *communication endpoints* are hosts
- But not always the case (e.g. Gateway being managed by SNMP)

IPsec Security Protocol Modes

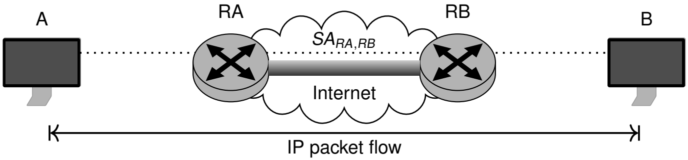
Tunnel Mode

- Used when at least one *cryptographic endpoint* is not a *communication endpoint*.
- This allows for gateways securing IP traffic on behalf of other entities

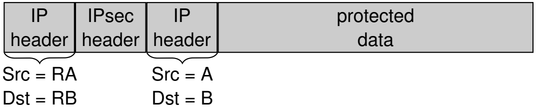
IPsec Security Protocol Modes

Tunnel Mode

- Used when at least one *cryptographic endpoint* is not a *communication endpoint*.
- This allows for gateways securing IP traffic on behalf of other entities



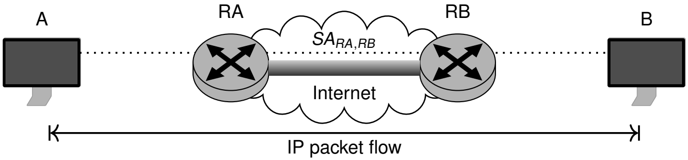
Packet Structure

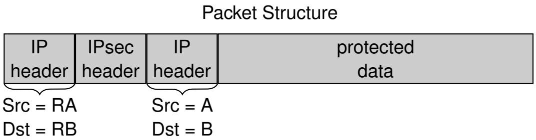


IPsec Security Protocol Modes

Tunnel Mode

- Used when at least one *cryptographic endpoint* is not a *communication endpoint*.
- This allows for gateways securing IP traffic on behalf of other entities





What if only one cryptographic endpoint is a communication endpoint?

IPsec Security Protocol Modes

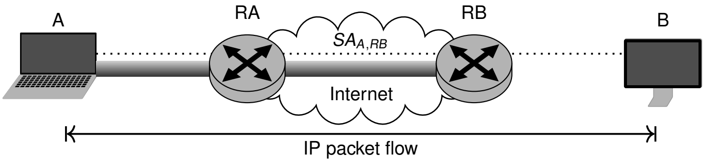
Tunnel Mode

- Used when at least one *cryptographic endpoint* is not a *communication endpoint*.
- Example: Security gateway, ensuring authentication and/or confidentiality between subnetwork and host

IPsec Security Protocol Modes

Tunnel Mode

- Used when at least one *cryptographic endpoint* is not a *communication endpoint*.
- Example: Security gateway, ensuring authentication and/or confidentiality between subnetwork and host



Packet Structure

IP header	IPsec header	IP header	protected data
Src = A Dst = RB		Src = A Dst = B	

IPsec
Security Policies

Traffic Selectors

A Traffic Selector (TS) is a set of properties used to characterize IP packets. Each TS may contain:

- IP source address
  - Specific host, network prefix, address range or wildcard
- IP destination address
  - Specific host, network prefix, address range or wildcard
  - In case of incoming tunneled packets the inner header is evaluated
- Name
  - DNS name, X.500 name or other name types
- Protocol
  - Protocol identifier of the transport protocol for this packet (e.g. TCP/UDP)
  - This may not be accessible when a packet is secured with ESP

Traffic Selectors

A Traffic Selector (TS) is a set of properties used to characterize IP packets. Each TS may contain:

- IP source address
  - Specific host, network prefix, address range or wildcard
- IP destination address
  - Specific host, network prefix, address range or wildcard
  - In case of incoming tunneled packets the inner header is evaluated
- Name
  - DNS name, X.500 name or other name types
- Protocol
  - Protocol identifier of the transport protocol for this packet (e.g. TCP/UDP)
  - This may not be accessible when a packet is secured with ESP

Traffic Selectors are used to define Security Policies!

IPsec
Security Policies

Definition

A Security Policy (SP) specifies which and how security services should be provided to IP packets.

This includes

- Selectors that identify specific IP flows
- Required security attributes for each flow
    - Security protocol (AH / ESP)
    - Protocol Mode (Transport / Tunnel)
    - Other parameters (e.g. policy lifetime, port number, ...)
- Actions (e.g. Discard, Secure, Bypass)

IPsec
Security Policies

Definition

A Security Policy (SP) specifies which and how security services should be provided to IP packets.

This includes

- Selectors that identify specific IP flows
- Required security attributes for each flow
  - Security protocol (AH / ESP)
  - Protocol Mode (Transport / Tunnel)
  - Other parameters (e.g. policy lifetime, port number, ...) ⇐ IPsec protection can be specified for specific applications!
- Actions (e.g. Discard, Secure, Bypass)

Security Policies

Definition

A Security Policy (SP) specifies which and how security services should be provided to IP packets.

This includes

- Selectors that identify specific IP flows
- Required security attributes for each flow
  - Security protocol (AH / ESP)
  - Protocol Mode (Transport / Tunnel)
  - Other parameters (e.g. policy lifetime, port number, ...) $\Leftarrow$ IPsec protection can be specified for specific applications!
- Actions (e.g. Discard, Secure, Bypass)

Security Policies are stored in the Security Policy Database (SPD)

IPsec
Security Associations

Definition

A Security Association (SA) is a simplex channel that describes the way how packets need to be processed

As such

- An SA defines employed encryption / authentication algorithms and keys
- An SA is associated with either AH or ESP but not both
- Bidirectional communication requires two security associations
- SAs can be setup as
  - Host $\leftrightarrow$ Host
  - Host $\leftrightarrow$ Gateway
  - Gateway $\leftrightarrow$ Gateway

Security Associations are stored in the Security Association Database (SAD)

Security Associations

In the Security Association Database (SAD)

- an entry (SA) is uniquely identified by a Security Parameter Index (SPI)
- the SPI value is specified by the receiving side during SA negotiation
- the SPI value for construction of AH/ESP headers is looked up for outbound SAs
- the SPI value is used to map the traffic to the appropriate SA for inbound traffic

IPsec
Security Associations

In the Security Association Database (SAD)

- an entry (SA) is uniquely identified by a Security Parameter Index (SPI)
- the SPI value is specified by the receiving side during SA negotiation
- the SPI value for construction of AH/ESP headers is looked up for outbound SAs
- the SPI value is used to map the traffic to the appropriate SA for inbound traffic

An SA entry in the SAD includes

- Security Parameter Index (SPI)
- IP source / destination address
- A security protocol identified (AH / ESP)
- Current sequence number counter (replay protection)
- Protocol algorithms, modes, IVs and keys for authentication and encryption
- Security Association Lifetime
- IPsec protocol mode (tunnel / transport)
- Additional information (see RFC4301, Section 4.4.2.1)

IPsec
Security Associations

In the Security Association Database (SAD)

- an entry (SA) is uniquely identified by a Security Parameter Index (SPI)
- the SPI value is specified by the receiving side during SA negotiation
- the SPI value for construction of AH/ESP headers is looked up for outbound SAs
- the SPI value is used to map the traffic to the appropriate SA for inbound traffic

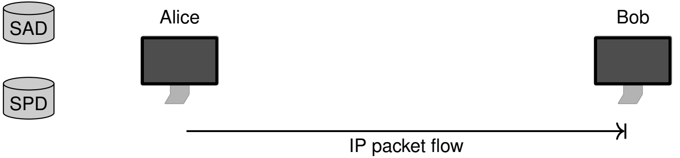
An SA entry in the SAD includes

- Security Parameter Index (SPI)
- IP source / destination address
- A security protocol identified (AH / ESP)
- Current sequence number counter (replay protection)
- Protocol algorithms, modes, IVs and keys for authentication and encryption
- Security Association Lifetime
- IPsec protocol mode (tunnel / transport)
- Additional information (see RFC4301, Section 4.4.2.1)

Now, how does this work in practice?

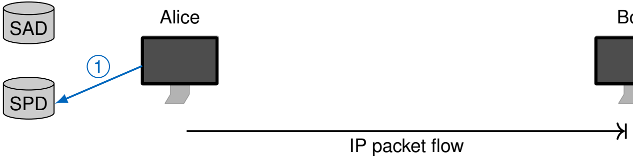
IPsec
Processing of IPsec Traffic

Outgoing Traffic



Alice wants to send data to Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

Outgoing Traffic

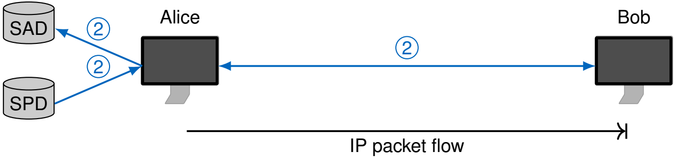


Alice wants to send data to Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

(1) Determine if and how the outgoing packet needs to be secured
- Perform a lookup in the SPD based on traffic selectors
- If the policy specifies *discard* then drop the packet ⇒ Done
- If the policy does not need to be secured, send it ⇒ Done

Processing of IPsec Traffic

Outgoing Traffic

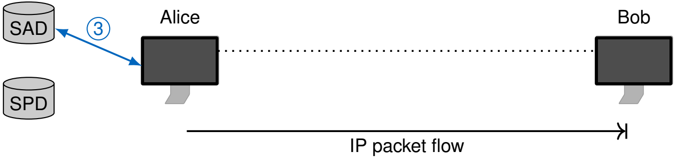


Alice wants to send data to Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

1. Determine if and how the outgoing packet needs to be secured
   - Perform a lookup in the SPD based on traffic selectors
   - If the policy specifies *discard* then drop the packet ⇒ Done
   - If the policy does not need to be secured, send it ⇒ Done
2. Determine which SA should be applied to the packet
   - There may be more than one SA matching the packet (e.g. one for AH, one for ESP)
   - If no SA is established perform IKE

IPsec
Processing of IPsec Traffic

Outgoing Traffic

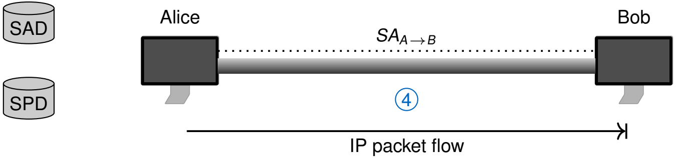


Alice wants to send data to Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

1. Determine if and how the outgoing packet needs to be secured
   - Perform a lookup in the SPD based on traffic selectors
   - If the policy specifies *discard* then drop the packet ⇒ Done
   - If the policy does not need to be secured, send it ⇒ Done
2. Determine which SA should be applied to the packet
   - There may be more than one SA matching the packet (e.g. one for AH, one for ESP)
   - If no SA is established perform IKE
3. Look up the determined or freshly created SA in the SAD

Processing of IPsec Traffic

Outgoing Traffic

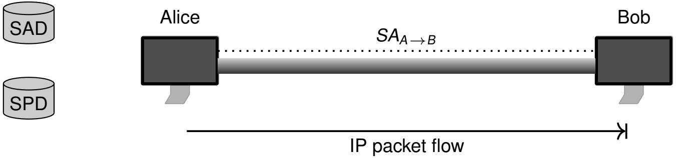


Alice wants to send data to Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

1. Determine if and how the outgoing packet needs to be secured
   - Perform a lookup in the SPD based on traffic selectors
   - If the policy specifies *discard* then drop the packet ⇒ Done
   - If the policy does not need to be secured, send it ⇒ Done
2. Determine which SA should be applied to the packet
   - There may be more than one SA matching the packet (e.g. one for AH, one for ESP)
   - If no SA is established perform IKE
3. Look up the determined or freshly created SA in the SAD
4. Perform the *security transforms*, specified in the SA
   - This results in the construction of an AH or ESP header
   - Possibly a new (outer) IP header will be created (tunnel mode)

Processing of IPsec Traffic

Outgoing Traffic

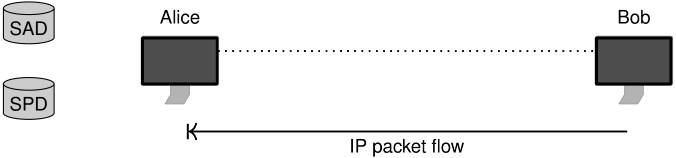


Alice wants to send data to Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

1. Determine if and how the outgoing packet needs to be secured
   - Perform a lookup in the SPD based on traffic selectors
   - If the policy specifies *discard* then drop the packet ⇒ Done
   - If the policy does not need to be secured, send it ⇒ Done
2. Determine which SA should be applied to the packet
   - There may be more than one SA matching the packet (e.g. one for AH, one for ESP)
   - If no SA is established perform IKE
3. Look up the determined or freshly created SA in the SAD
4. Perform the *security transforms*, specified in the SA
   - This results in the construction of an AH or ESP header
   - Possibly a new (outer) IP header will be created (tunnel mode)
5. Send the resulting packet ⇒ Done

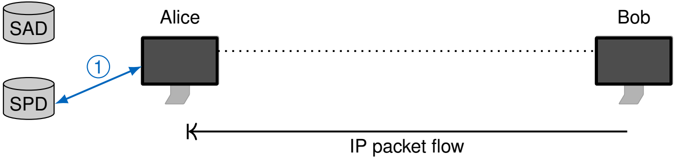
IPsec
Processing of IPsec Traffic

Incoming Traffic



Alice receives data from Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

Incoming Traffic

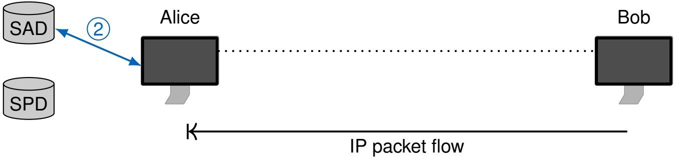


Alice receives data from Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

1. If packet contains an IPsec header
   - Perform a lookup in the SPD, if Alice is supposed to process the packet
   - Retrieve the respective policy

Processing of IPsec Traffic

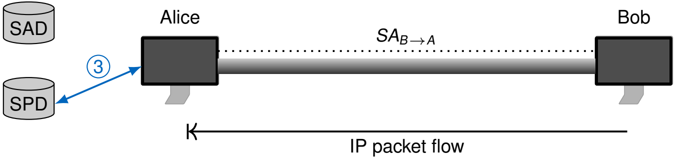
Incoming Traffic



Alice receives data from Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

1. If packet contains an IPsec header
   - Perform a lookup in the SPD, if Alice is supposed to process the packet
   - Retrieve the respective policy
2. If Alice is supposed to process the packet
   - Extract the SPI from the IPsec header, look up the SA in the SAD and perform the appropriate processing
   - If there's no SA referenced by the SPI ⇒ Drop the packet

Incoming Traffic

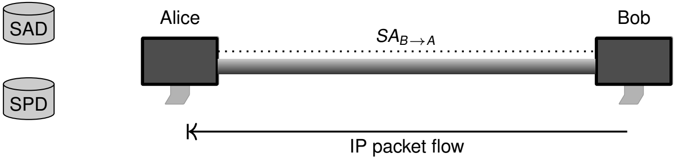


Alice receives data from Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

1. If packet contains an IPsec header
   - Perform a lookup in the SPD, if Alice is supposed to process the packet
   - Retrieve the respective policy
2. If Alice is supposed to process the packet
   - Extract the SPI from the IPsec header, look up the SA in the SAD and perform the appropriate processing
   - If there's no SA referenced by the SPI ⇒ Drop the packet
3. Determine if and how the packet should have been protected
   - Perform a lookup in the SPD, evaluating the inner IP header in case of tunneled packets
   - If the respective policy specifies *discard* ⇒Drop the packet
   - If the protection of the packet did not match the policy ⇒Drop the packet

Processing of IPsec Traffic

Incoming Traffic

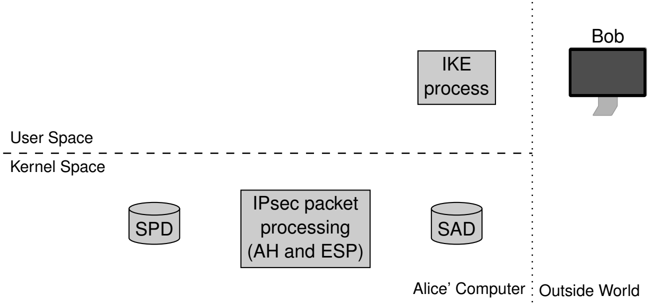


Alice receives data from Bob. To support IPsec, the IP layer of Alice has to perform the following steps:

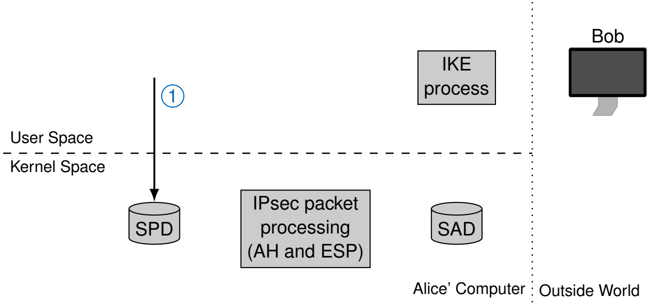
1. If packet contains an IPsec header
   - Perform a lookup in the SPD, if Alice is supposed to process the packet
   - Retrieve the respective policy
2. If Alice is supposed to process the packet
   - Extract the SPI from the IPsec header, look up the SA in the SAD and perform the appropriate processing
   - If there's no SA referenced by the SPI ⇒ Drop the packet
3. Determine if and how the packet should have been protected
   - Perform a lookup in the SPD, evaluating the inner IP header in case of tunneled packets
   - If the respective policy specifies *discard* ⇒Drop the packet
   - If the protection of the packet did not match the policy ⇒Drop the packet
4. Deliver to the appropriate protocol entity (e.g. network / transport layer)

Processing of IPsec Traffic

Architecture View



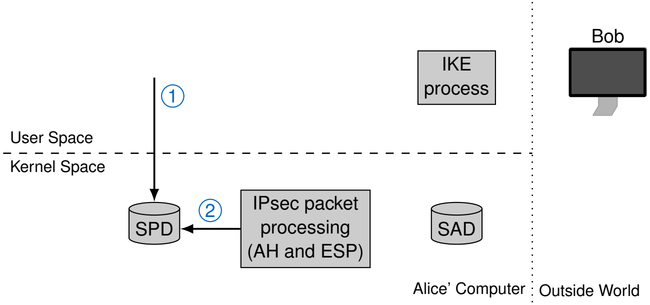
Architecture View



1 The administrator sets a policy in SPD

IPsec
Processing of IPsec Traffic

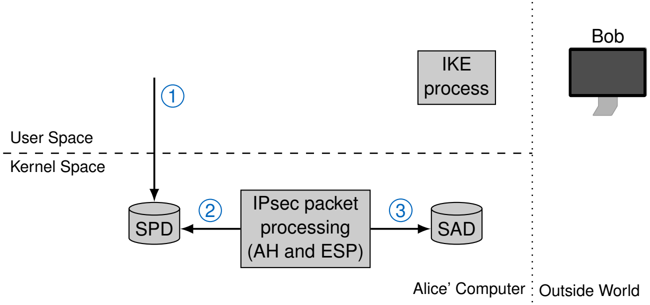
Architecture View



1. The administrator sets a policy in SPD
2. The IPsec processing module refers to the SPD in order to make a decision on applying IPsec on packet

IPsec
Processing of IPsec Traffic

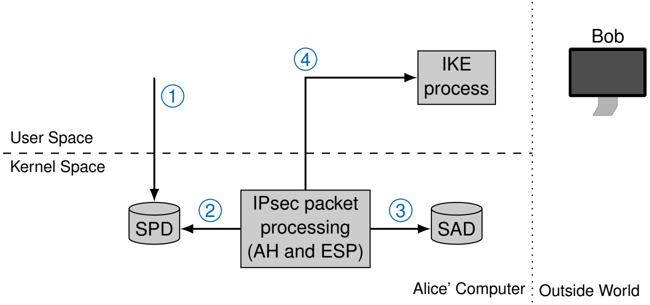
Architecture View



1. The administrator sets a policy in SPD
2. The IPsec processing module refers to the SPD in order to make a decision on applying IPsec on packet
3. If IPsec is required, then the IPsec module looks for the IPsec SA in the SAD

IPsec
Processing of IPsec Traffic

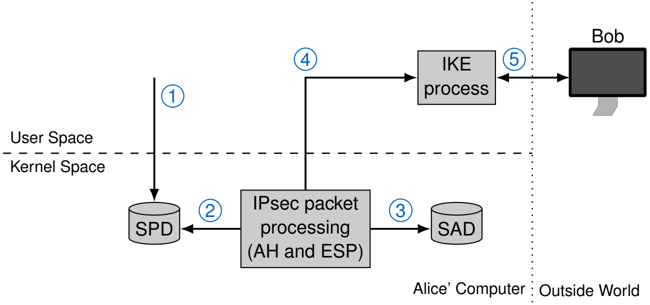
Architecture View



1. The administrator sets a policy in SPD
2. The IPsec processing module refers to the SPD in order to make a decision on applying IPsec on packet
3. If IPsec is required, then the IPsec module looks for the IPsec SA in the SAD
4. If there is no SA yet, the IPsec module sends a request to the IKE process to create an SA

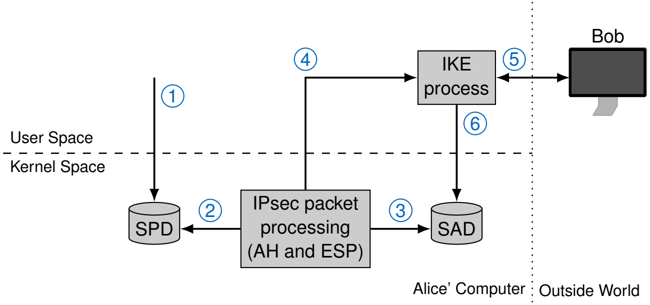
IPsec
Processing of IPsec Traffic

Architecture View



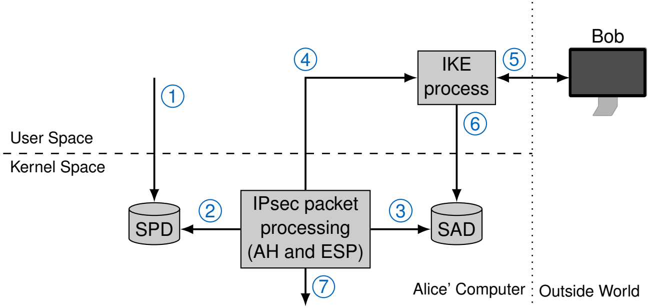
1. The administrator sets a policy in SPD
2. The IPsec processing module refers to the SPD in order to make a decision on applying IPsec on packet
3. If IPsec is required, then the IPsec module looks for the IPsec SA in the SAD
4. If there is no SA yet, the IPsec module sends a request to the IKE process to create an SA
5. The IKE process negotiates keys and crypto algorithms with the peer host using the IKE/IKEv2 protocol

Architecture View



1. The administrator sets a policy in SPD
2. The IPsec processing module refers to the SPD in order to make a decision on applying IPsec on packet
3. If IPsec is required, then the IPsec module looks for the IPsec SA in the SAD
4. If there is no SA yet, the IPsec module sends a request to the IKE process to create an SA
5. The IKE process negotiates keys and crypto algorithms with the peer host using the IKE/IKEv2 protocol
6. The IKE process writes the key and all required parameters into the SAD

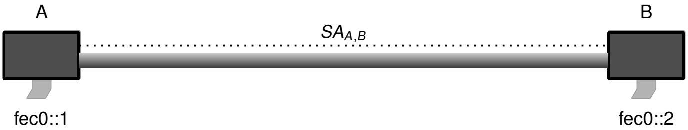
Architecture View



① The administrator sets a policy in SPD
② The IPsec processing module refers to the SPD in order to make a decision on applying IPsec on packet
③ If IPsec is required, then the IPsec module looks for the IPsec SA in the SAD
④ If there is no SA yet, the IPsec module sends a request to the IKE process to create an SA
⑤ The IKE process negotiates keys and crypto algorithms with the peer host using the IKE/IKEv2 protocol
⑥ The IKE process writes the key and all required parameters into the SAD
⑦ The IPsec module can now send a packet with applied IPsec

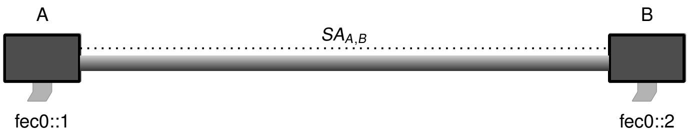
Setup of IPsec Security Policies

Example:
IPv6 connection with *ESP* and *Transport Mode*



IPsec
Setup of IPsec Security Policies

Example:
IPv6 connection with *ESP* and *Transport Mode*



Configuration at Host A

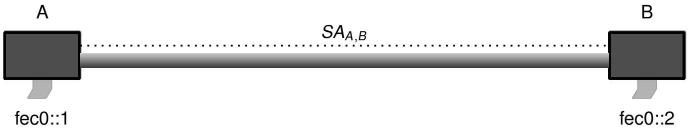
- `spdadd fec0::1 fec0::2 any -P out IPsec esp/transport//require;`
- `spdadd fec0::2 fec0::1 any -P in IPsec esp/transport//require;`

where

- First IP corresponds to source in the IP header
- Second IP corresponds to destination in the IP header
- `any` defines the upper-layer protocol (e.g. ip4, ...)
- `out` defines the policy to hold for outgoing packets
- `in` defines the policy to hold for incoming packets

IPsec
Setup of IPsec Security Policies

Example:
IPv6 connection with *ESP* and *Transport Mode*



Configuration at Host A

- `spdadd fec0::1 fec0::2 any -P out IPsec esp/transport//require;`
- `spdadd fec0::2 fec0::1 any -P in IPsec esp/transport//require;`

⇓

Configuration at Host B

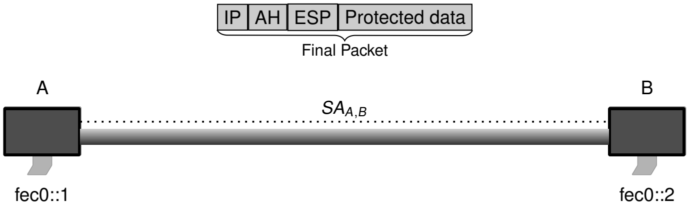
- `spdadd fec0::2 fec0::1 any -P out IPsec esp/transport//require;`
- `spdadd fec0::1 fec0::2 any -P in IPsec esp/transport//require;`

where

- First IP corresponds to source in the IP header
- Second IP corresponds to destination in the IP header
- `any` defines the upper-layer protocol (e.g. ip4, ...)
- `out` defines the policy to hold for outgoing packets
- `in` defines the policy to hold for incoming packets

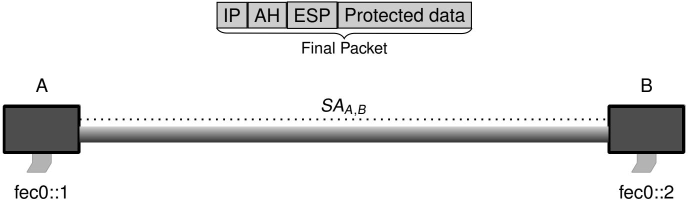
IPsec
Setup of IPsec Security Policies

Example:
IPv6 connection with *ESP/Transport* applied first and *AH/Transport* applied next:



IPsec
Setup of IPsec Security Policies

Example:
IPv6 connection with *ESP/Transport* applied first and *AH/Transport* applied next:

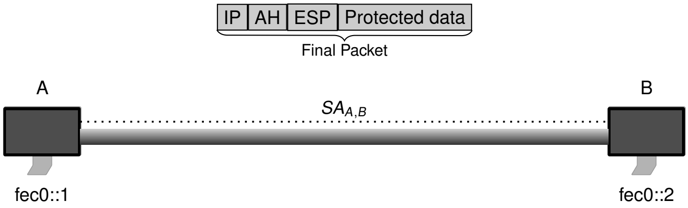


Configuration at Host A

- `spdadd fec0::1 fec0::2 any -P out IPsec esp/transport//require ah/transport//require;`
- `spdadd fec0::2 fec0::1 any -P in IPsec esp/transport//require ah/transport//require;`

Setup of IPsec Security Policies

Example:
IPv6 connection with *ESP/Transport* applied first and *AH/Transport* applied next:



Configuration at Host A

- `spdadd fec0::1 fec0::2 any -P out IPsec esp/transport//require ah/transport//require;`
- `spdadd fec0::2 fec0::1 any -P in IPsec esp/transport//require ah/transport//require;`

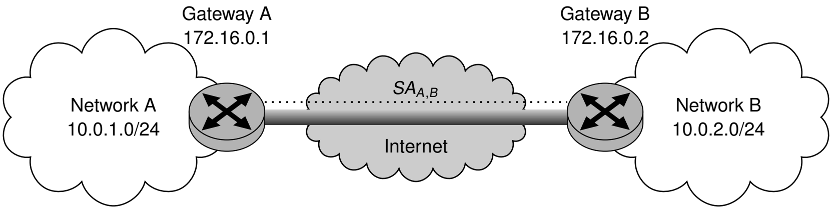
⇓

Configuration at Host B:

- `spdadd fec0::2 fec0::1 any -P out IPsec esp/transport//require ah/transport//require;`
- `spdadd fec0::1 fec0::2 any -P in IPsec esp/transport//require ah/transport//require;`

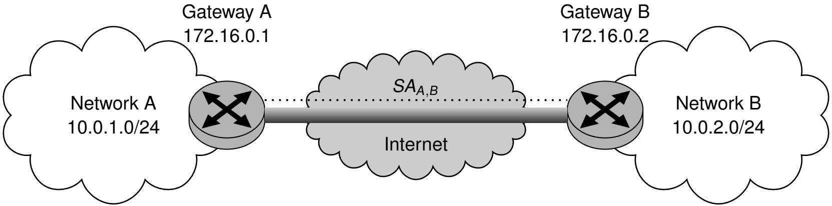
IPsec
Setup of IPsec Security Policies

Example:
ESP Tunnel for VPN



IPsec
Setup of IPsec Security Policies

Example:
ESP Tunnel for VPN

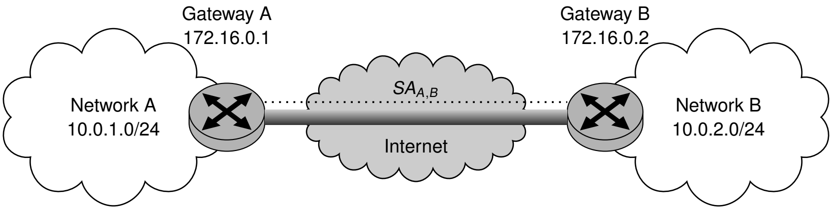


Configuration at Gateway A

- `spdadd 10.0.1.0/24 10.0.2.0/24 any -P out IPsec esp/tunnel/172.16.0.1-172.16.0.2/require;`
- `spdadd 10.0.2.0/24 10.0.1.0/24 any -P in IPsec esp/tunnel/172.16.0.2-172.16.0.1/require;`

IPsec
Setup of IPsec Security Policies

Example:
ESP Tunnel for VPN



Configuration at Gateway A

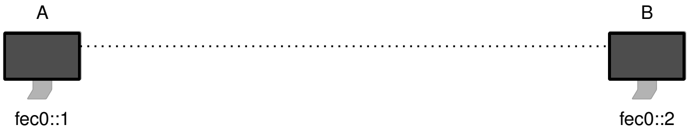
- `spdadd 10.0.1.0/24 10.0.2.0/24 any -P out IPsec esp/tunnel/172.16.0.1-172.16.0.2/require;`
- `spdadd 10.0.2.0/24 10.0.1.0/24 any -P in IPsec esp/tunnel/172.16.0.2-172.16.0.1/require;`

⇓

Configuration at Gateway B:

- `spdadd 10.0.2.0/24 10.0.1.0/24 any -P out IPsec esp/tunnel/172.16.0.2-172.16.0.1/require;`
- `spdadd 10.0.1.0/24 10.0.2.0/24 any -P in IPsec esp/tunnel/172.16.0.1-172.16.0.2/require;`

IPsec
Setup of IPsec Security Associations



Example:
Manually setting up an *AH SA*:

```
# add src dst proto spi -A authalgo key;
add fec0::1 fec0::2 ah 700 -A hmac-md5 0xbf9a081e7ebdd4fa824c822ed94f5226;
add fec0::2 fec0::1 ah 800 -A hmac-md5 0xbf9a081e7ebdd4fa824c822ed94f5226;
```

Manually setting up an *ESP SA*:

```
# add src dst proto spi -E encalgo key;
add fec0::1 fec0::2 esp 701 -E 3des-cbc 0xdafb418410b2ca6a2ba144561fab354640080e5b7a;
add fec0::2 fec0::1 esp 801 -E 3des-cbc 0xdafb418410b2ca6a2ba144561fab354640080e5b7a;
```

WARNING: Setting up an SA manually is error prone!

- The administrator might choose insecure keys
- The set of SAs might be inconsistent
- It is better to rely on an IKE daemon for setting up SAs

Tunnel Protocols

- Layer 3 secure network tunnel for IPv4 and IPv6
- Implemented in the Linux kernel since version 5.6
- UDP based, easy firewall holepunching
- Modern cryptographic algorithms
- Emphasis on simplicity and auditability
- Authentication model similar to `ssh authorized_keys`
- Replacement for OpenVPN and IPsec

The following slides contain content from Jason A. Donenfeld (wireguard.com)

WireGuard

Easily auditable

- openvpn: **116,730** LoC + OpenSSL!
- Linux IPSec: XFRM (119,363 LoC) + StrongSwan: **405,894** LoC!
- SoftEther: **329,853** LoC
- WireGuard: **4,561** LoC

Easy to set up

- Just a network interface
- `ip link add wg0 type wireguard`
- Endpoint roaming like in `mosh`/`tinc`
- Identities are static public keys like in `ssh`

Packet flow

- Userspace: `send(data)`
- Standard Kernel: Standard routing decision for `wg0`
- WireGuard: Destination IP selects peer
- WireGuard: `encrypt(data)`
- Standard Kernel: `send(encrypted)`
- ... Internet ...
- Standard Kernel: `receive(encrypted)`
- WireGuard: `decrypt(packet)` & determine peer
- WireGuard: Check source IP against allowed peer IPs
- Standard Kernel: further packet processing

Further design ideas

- Fixed width header fields, **no parsing needed**.
- **No State allocation** during work, only during config
- **No Memory allocation** when handling received packets
  - requires the crypto to work with finite amount of preallocated memory
- No state is modified when handling unauthenticated packets
- WireGuard grew out of kernel rootkit project
  - Does not respond to any unauthenticated packets
  - No keepalives

Modern Crypto Primitives

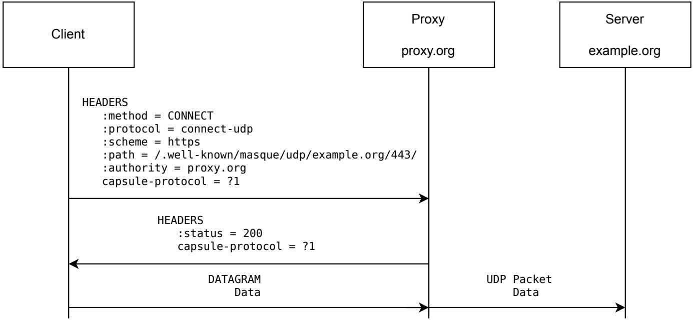
- ChaCha20 for symmetric encryption
- authenticated with Poly1305
- using RFC7539's AEAD construction
- Curve25519 for ECDH
- BLAKE2s for hashing and keyed hashing described in RFC7693
- SipHash24 for hashtable keys
- HKDF for key derivation, as described in RFC5869
- Noise_IKpsk2 key exchange protocol

MASQUE

- A set of protocols for proxying over HTTP
- Under active development
- Standardized by the IETF:
  - proxying of IP traffic (`CONNECT-IP`, RFC 9484)
  - proxying of UDP traffic (`CONNECT-UDP`, RFC 9298)
- Under development by the IETF:
  - proxying of L2 traffic (`CONNECT-Ethernet`)
  - QUIC-aware proxying
- Benefits from using mainly QUIC as transport
- MASQUE traffic is hard to distinguish from HTTP
  - therefore hard to block

MASQUE

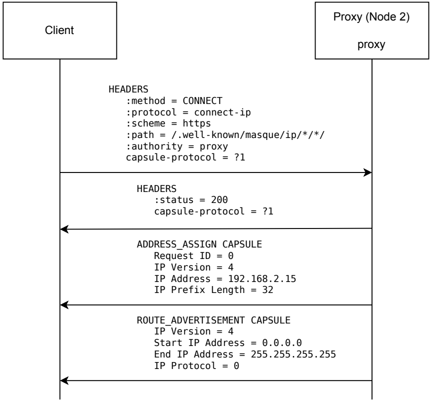
CONNECT-UDP



- Client signals destination and data type via HTTP headers
- Proxy acknowledges connection setup with HTTP status code 200
- Client sends application data in DATAGRAM frames, proxy forwards data in UDP packets

MASQUE

CONNECT-IP



- Client opens tunnel via HTTP headers
- Proxy acknowledges connection setup with HTTP status code 200
- Client sends complete IP packets in DATAGRAM frames, proxy sends them out

MASQUE
iCloud Private Relay

- A privacy service from Apple using MASQUE
- Available for iCloud+ subscribers
- Enabled by default on. . .
  - . . . iOS 15 and later
  - . . . macOS Monterey (12) and later
- Uses Privacy Partitioning principle

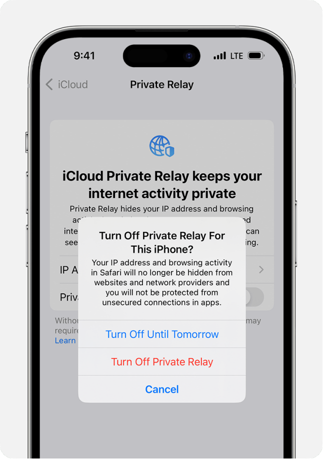


Image taken from https://support.apple.com/en-us/102022

iCloud Private Relay - Privacy Partitioning

- Two nested MASQUE connections
- Data is sent to the ingress proxy
  - operated by Apple
  - Knows client’s IP address
  - Does not know content or destination IP address
- Data is forwarded to the egress proxy
  - operated by "third-party content provider"
    - a CDN like Akamai or Cloudflare
  - Knows destination IP address
  - Does not know the client’s IP address . . .
  - . . . and the content, depending on the protocol used

Can access	Client	Ingress	Egress	Server
Data	✓	✗	○	✓
Client IP	✓	✓	✗	✗
Destination	✓	✗	✓	✓

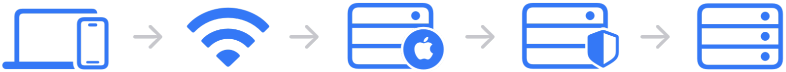


Figure 1: Data flow from client to server with Apple iCloud Private Relay. Illustration taken from https://support.apple.com/en-us/102602.

Tunnel Protocols

TLS/SSL-based VPNs
OpenVPN [2]

Overview

- Key exchange is based on TLS/SSL
- Can be used on top of UDP or TCP (Why is TCP a bad idea?)
- Traffic encryption uses custom scheme
- Good NAT traversal properties
- Easy to use
- Not an industry standard
- Not very “professional”, but hacker community likes it
- Open Source

Use case:

- Road warriors (laptops connecting to the office)
- Students etc. building a cheap VPN

TLS/SSL-based VPNs
Cisco AnyConnect [3][4]

Overview

- Proprietary Cisco software
- Supports several protocols:
  - (Mostly) SSL/TLS based
  - Can use Datagram TLS (DTLS), DTLS uses UDP instead of TCP
  - Can run on port 443 (HTTPS) $\rightarrow$ usually no problem with firewalls
- No problems with NAT

Use cases:

- Big corporations supporting mobile endpoints (laptops)
- Corporations with existing Cisco infrastructure
- Academic compute centers (e.g. LRZ) deployed Cisco AnyConnect

Tunnel Protocols

Other protocols
Point-to-Point Tunneling Protocol (PPTP) [5]

- Standardized in 1999
- Mostly deployed in Microsoft Windows environments
- Weak authentication and encryption schemes
- Should not be used anymore
- Uses a modified version GRE for tunneling

Other protocols

Other well-known tunneling protocols

- Generic Routing Encapsulation (GRE)
- Layer 2 Tunneling Protocol (L2TP, RFC 3355)
- SSH tunnel (port forwarding)
- IP-over-IP (RFC 2003)
- HTTP tunnel
- ICMP tunnel
- DNS tunnel
- ...

Tunnel Protocols

Summary

Different protocols for different use-cases

- Simplifying L2 networks administration and separation: VLAN, VXLAN
- Connect remote workers to company resources over the Internet: IPsec, SSL-based VPNs
- Evade some firewalls: MASQUE, IP-over-(HTTP/DNS/ICMP), . . .

Different protocols for different features

- Encryption and authentication
- Easier addressing
- Performance (e.g. TCP-over-TCP)

Different protocols for different software support

- Some tunneling protocols are directly supported by operating systems

Tunnel Protocols

[1] S. Kent and K. Seo, Security Architecture for the Internet Protocol, https://tools.ietf.org/html/rfc4301, 2005.

[2] OpenVPN Developers, OpenVPN security overview, https://openvpn.net/index.php/open-source/documentation/security-overview.html, 2017.

[3] Cisco, Cisco IOS Secure Sockets Layer (SSL) VPN Technology Overview, https://www.cisco.com/c/dam/en/us/products/collateral/security/ios-sslvpn/IOS_SSL_VPN_TDM_v8-jz-an.pdf, 2008.

[4] Cisco, Cisco AnyConnect Secure Mobility Client Data Sheet, https://www.cisco.com/c/en/us/products/collateral/security/anyconnect-secure-mobility-client/datasheet-c78-733184.html, 2017.

[5] K. Hamzeh, G. Pall, W. Verthein, J. T. W. Little, and G. Zorn, Point-to-Point Tunneling Protocol, https://tools.ietf.org/html/rfc2637, 1999.