An Introduction to IPsec

Daniel Migault

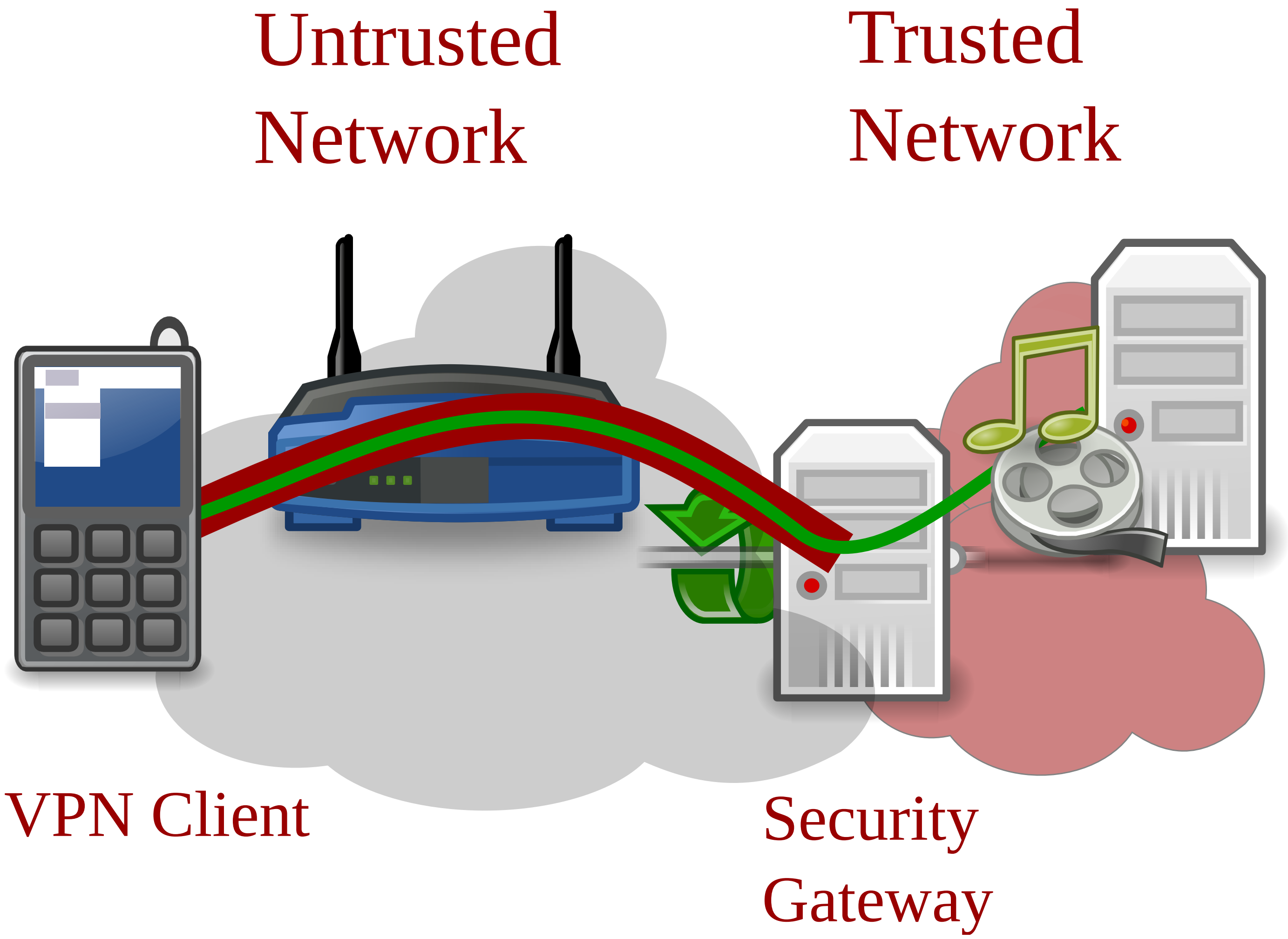
IPsec Overview

IPsec designates an architecture [RFC4301] as well as a suite of protocols AH [RFC4302], ESP [RFC4303], IKEv2 [RFC7296]

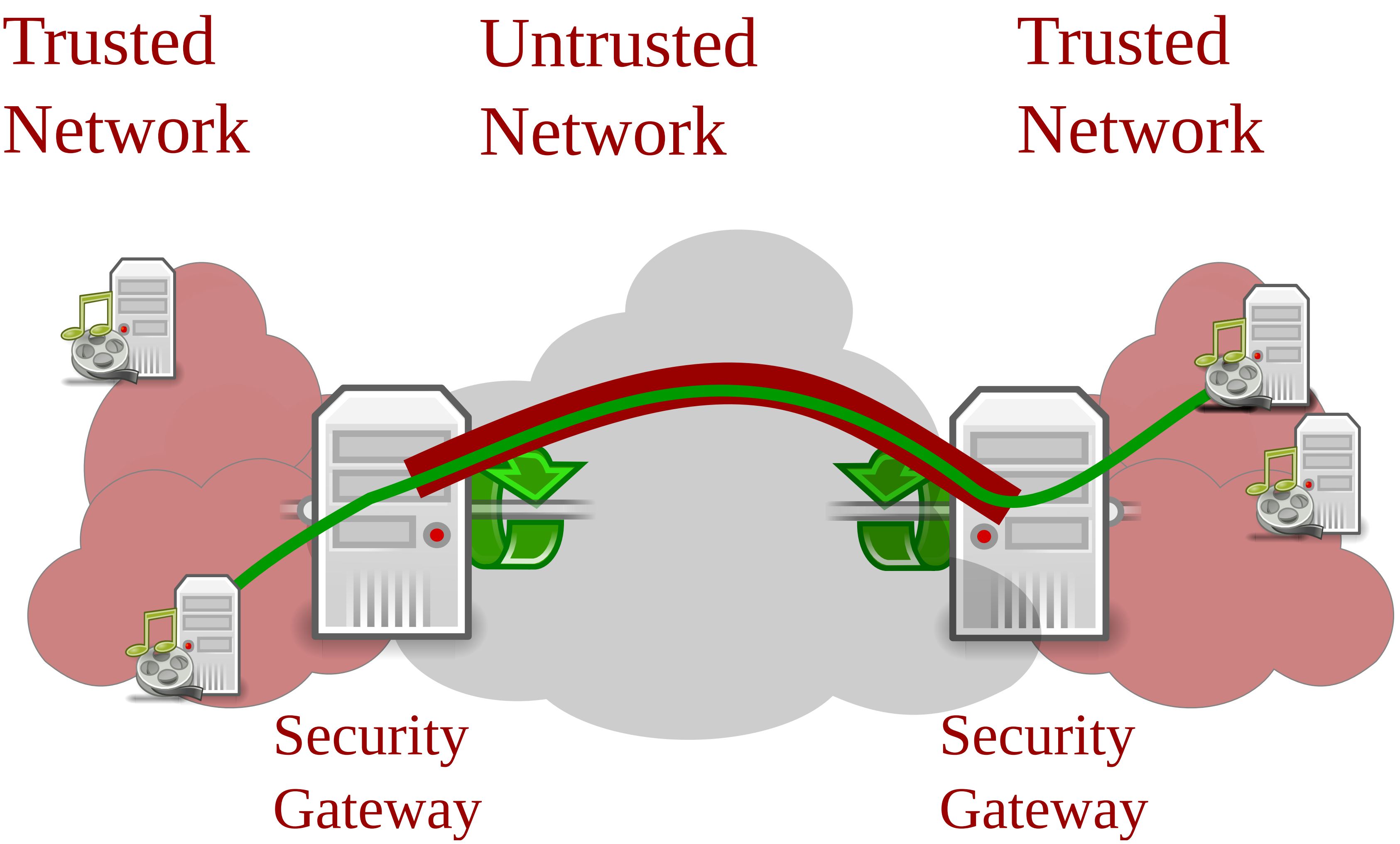
IPsec secures IP traffic for:

- Virtual Private Network (VPN):
  - Extend a trusted domain over an untrusted domain
- Gateway-to-Gateway:
  - Interconnect two trusted domain over an untrusted network
- End-to-End secure communication
  - Terminating nodes secure communications using IPsec similarly to TLS

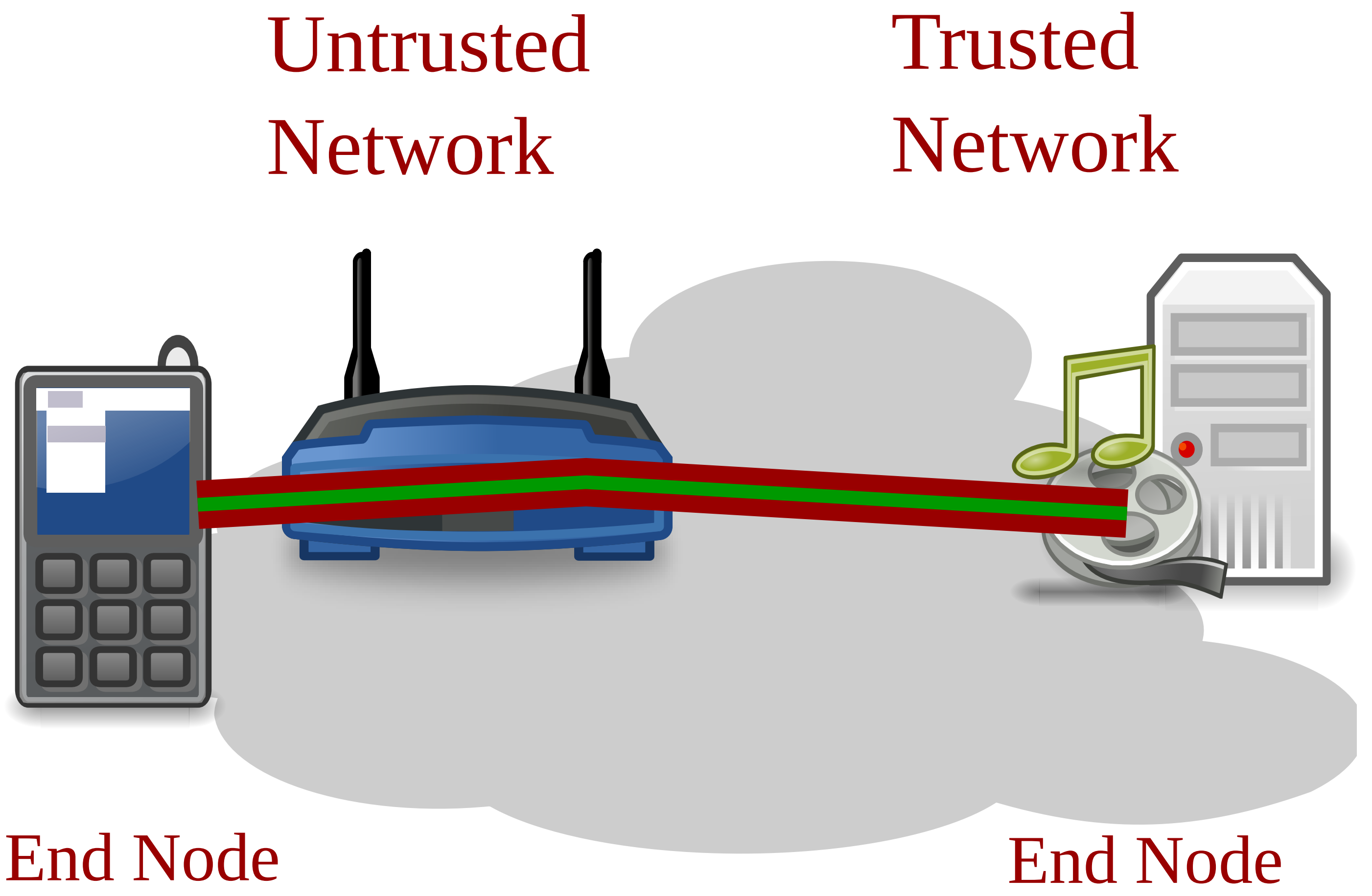
IPsec Overview - Use Case - VPN



IPsec Overview - Use Case - GW-to-GW

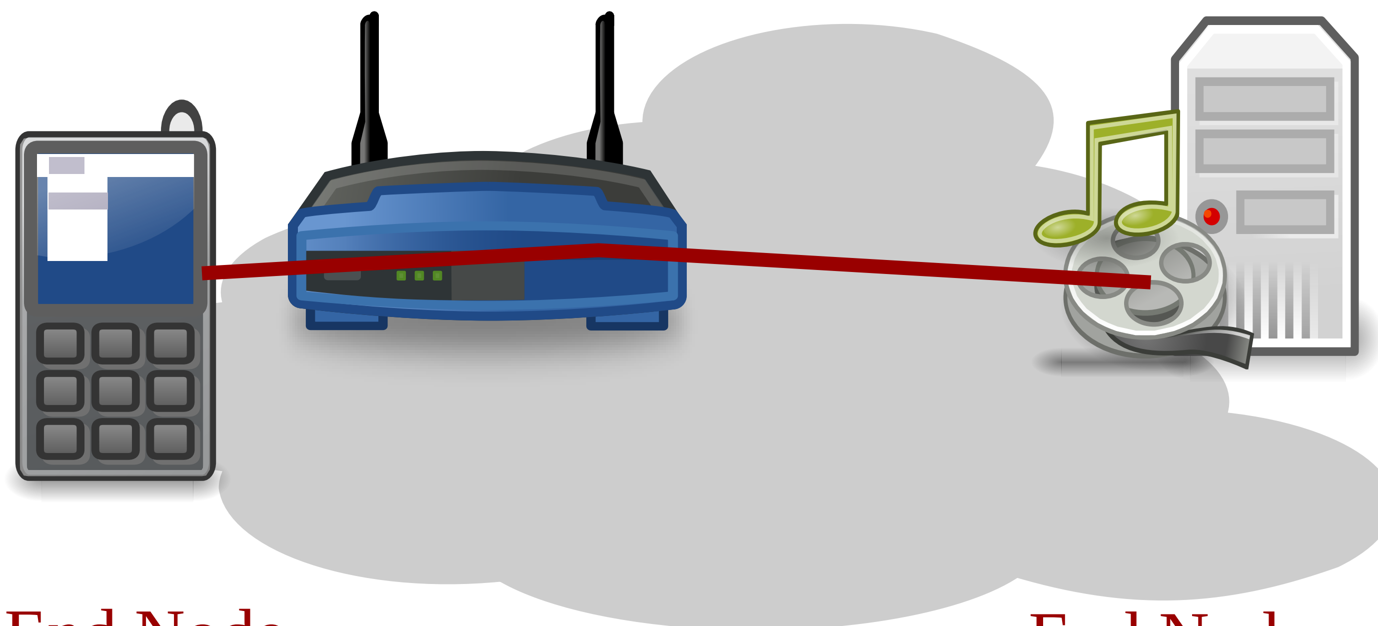


IPsec Overview - Use Case - E2E Security



IPsec Overview - Use Case - E2E Security

Untrusted
Network



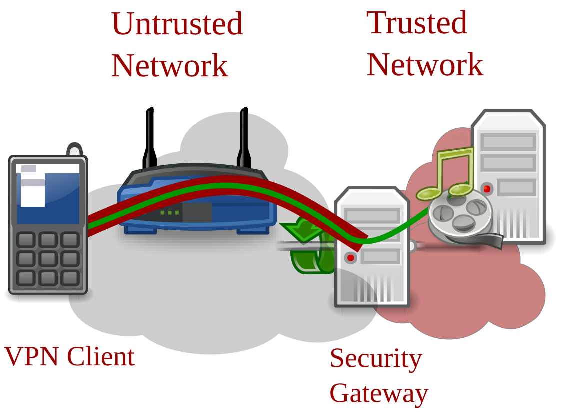
End Node

End Node

IPsec Overview

IPsec secures traffic over untrusted network.

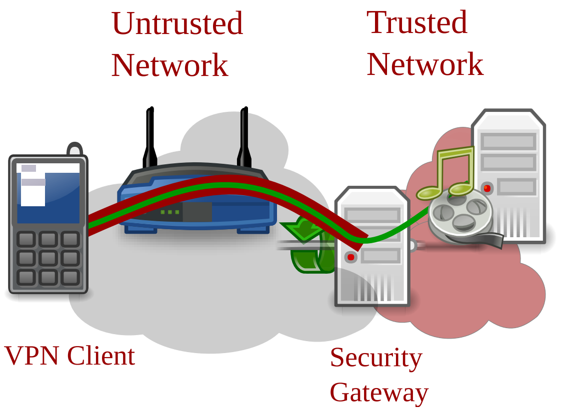
- Endpoints are trusted and authenticated
- Endpoint are configured to secure traffic with its peer.
    - Inbound and Outbound traffic



IPsec Overview

Securely exchanging traffic requires an agreement on:

- Keying material and cryptographic algorithms
- Packet format (protocols, modes, ...)



Manual configuration is possible, but using key exchange agreement is recommended.

- Internet Key Exchange Protocol (IKEv2)

IPsec Overview

Using IKEv2 enhances:

- Cryptographic material management:
  - Negotiation is delegated to the peers
  - Automated life cycle
  - ...
- Security:
  - Avoid misconfigurations (reuse of keys, IV, ...)
  - Avoids centralized key distribution and key sharing outside of the peers
  - ...

IPsec Overview - Trust Model

IPsec trust model can be recaped as follows:

- Peers trusted each other within a security domain
- Peers are configured to secure there respective inbound /outbound traffic
- Peers agree on the keying material to be used
- Peers exchange securly traffic over an untrusted network.
    - encapsulated (VPN): tunnel mode
    - protected via IP options: transport mode

IPsec Overview - Protocol Suite

IPsec defines an architecture and a suite of protocols:

- Architecture that defines the IPsec engine:
  - Security Policies (SP),
  - Security Associations (SA)
- IPsec packet processing (Data Plan):
  - ESP: Encapsulation Security *Payload*
  - AH: Authentication *Header*
- IKEv2: Internet Key Exchange Protocol (Control Plan)

IPsec Overview - Position to TLS

IPsec secures infrastructure:

- Both peers belong to the same administrative domain
  - Configuration on both peers
  - Peers authenticate each other
- IPsec secures IP traffic (IP packet, IP payload)

TLS secures Web traffic:

- TLS protects the application payloads (TCP payload)
- TLS traffic as TCP port 443 (Implicit single Security Policy)
- Only the TLS Web browser authenticates the Web Server
  - Web application authenticates the user (login, passwords)

IPsec Overview - Position to TLS

IPsec and TLS share a similar construction:

- Key Exchange protocol to configure / manage the data plan
- Exchange traffic through the data plan

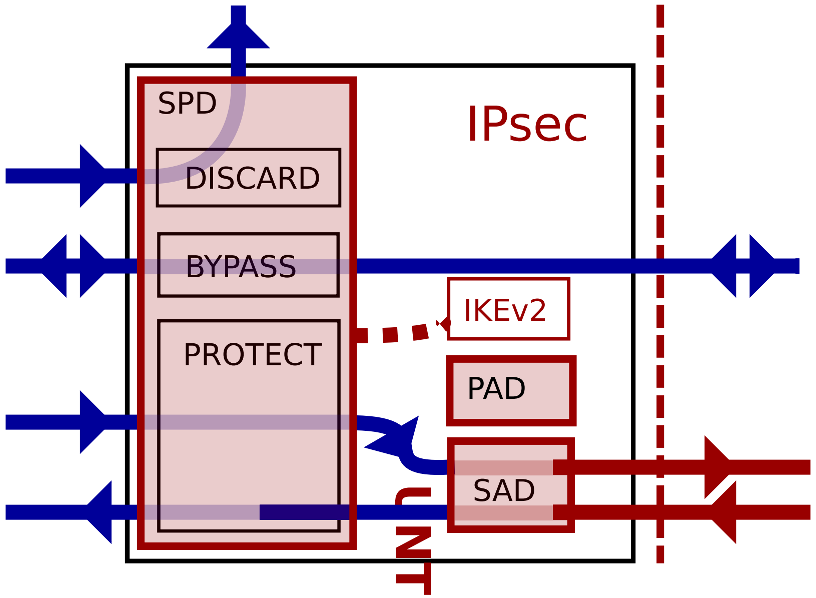
While they have different usages both protocols can likely be interchanged:

- TLS can perform mutual authentication
- IPsec can also leave one peer unauthenticated

IPsec Architecture

IPsec architecture involves:

- SPD: Security Policy Databases
- SAD: Security Association Database
- PAD: Peer Authentication Database



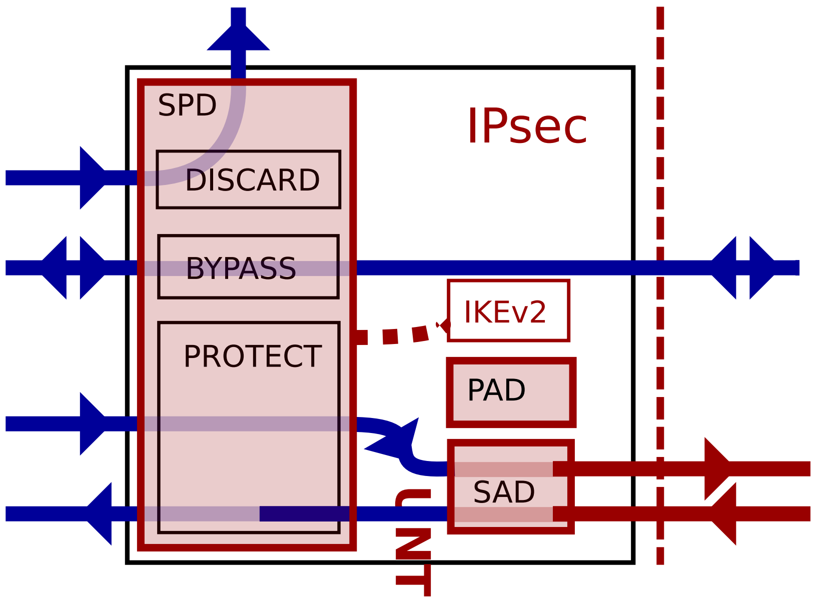
SPD provides packet policy DISCARD, BYPASS or PROTECT.
SAD provides information to encrypt and decrypt IPsec packets.
PAD provides information for IKEv2 authentication.

IPsec Architecture:

IPsec processes traffic according to these databases.

IPsec processes Inbound / Outbound differently

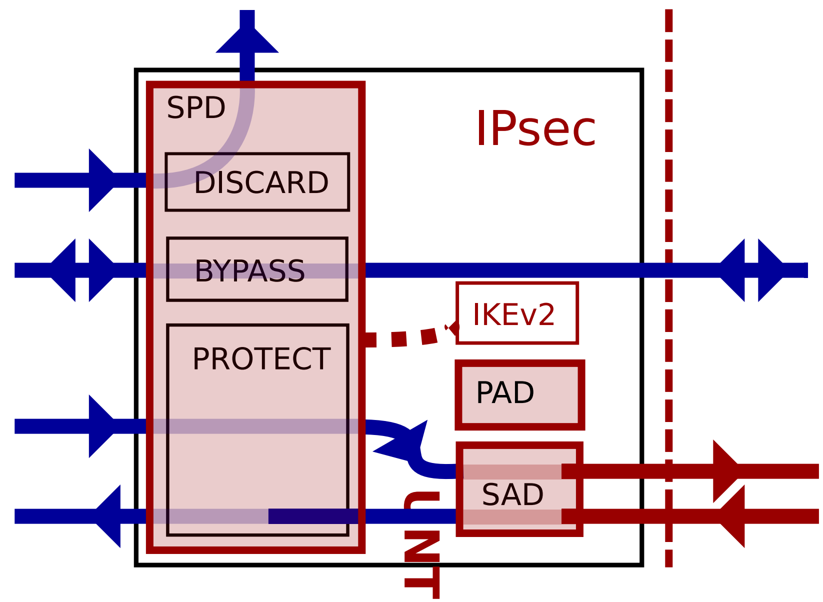
- Interactions SAD, SPD, PAD are different



IPsec Architecture - Processing Outbound Packets

Outgoing packet is *matched* against the SPD:

- Traffic Selectors (TS)
- BYPASS, DISCARD are processed immediately
- PROTECT requires a SA



If no SA has been created:

- IKEv2 lookup the PAD to authenticate the peer
- IKEv2 proceeds to SA agreement

The packet is encrypted as indicated by the SA

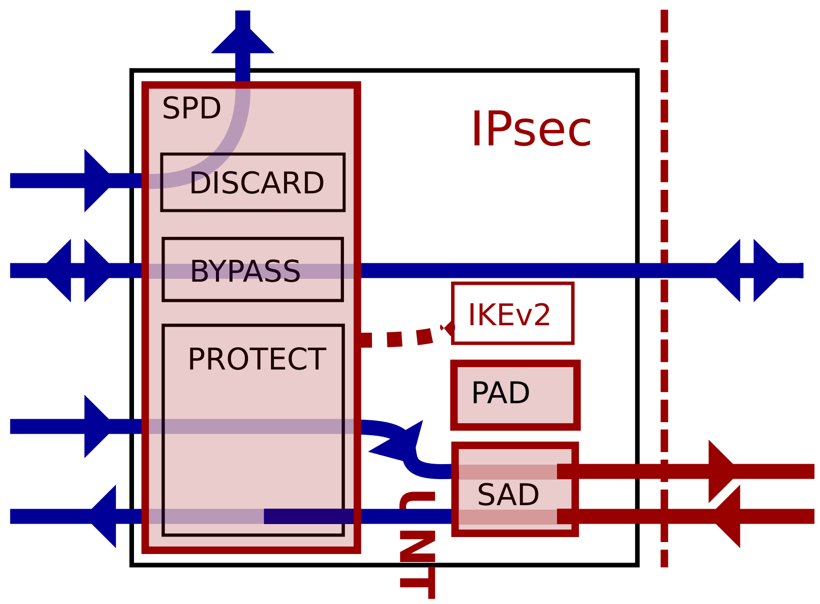
IPsec Architecture - Processing Inbound Packets

Incoming IPsec packet is *matched* against the SAD

- Security Policy Index (SPI)
- Non matching packets are discarded

SA provides information to decrypt the packet

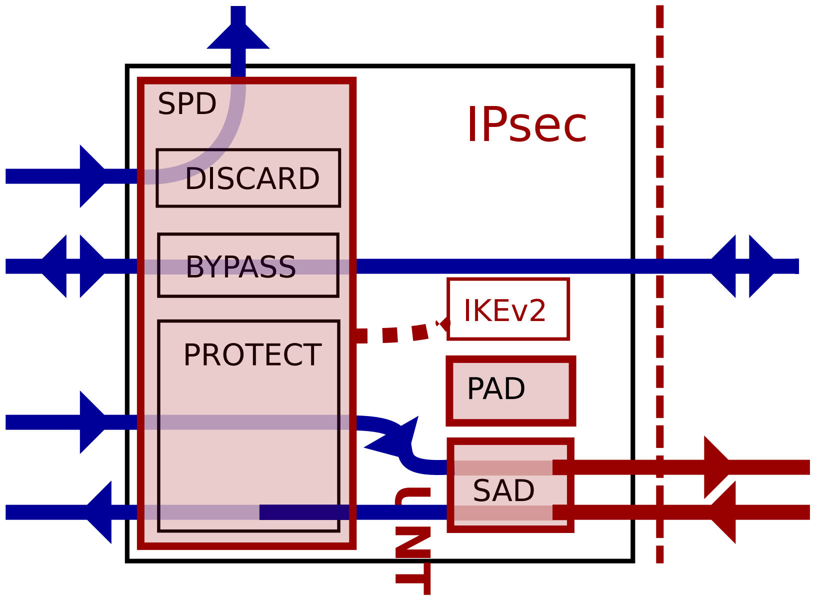
Clear text packet is matched against the SPD



IPsec Architecture - Processing Inbound Packets

Incoming non IPsec packet is *matched* against the SPD

- Traffic Selectors (TS)



IPsec Architecture - Security Policy Database

SPD contains a high level description of the Security Policies

- Matching rules are defined via Traffic Selectors (TS) that can be read from the packet (i.e IP addresses, ports...)
- Matching packet are associated a DISCARD, BYPASS or PROTECT policy

IPsec Architecture - Security Policy Database

The SPD's complexity results from opposite goals:

- Performance: matching every inbound / outbound packets
- Manageability: expressing human readable / abstract policies.

SPD is usually subdivided in multiple sub databases distributed over the system.

- SPD-I for BYPASSed inbound traffic
- SPD-O for BYPASSed outbound traffic
- SPD-S for traffic that needs to be PROTECTed
- SPD cache reflects in the kernel a packet level expression
    - derived from more abstracted expression of SP.

IPsec Architecture - Security Policy Database

SP indicates the policy associated to the traffic:

- Traffic Selectors (TS):
  - Source / Destination IP Address
  - Next Layer Protocol,
  - Source / Destination Port ...
  - Name
  - PFP Flag: Populated From Packet
- Processing Info:
  - Policy: BYPASS, DISCARD, PROTECT
  - IPsec mode, Tunnel IP header
  - IPsec protocol: ESP, AH
  - Cryptographic Algo

IPsec Architecture - Security Policy Database

SP remains abstract as TS do not necessarily match packet:

- TS uses ANY, OPAQUE words, Names, value ranges

Name designates:

- The peer ID (*company.com*) setting a SA with me via IKEv2
  - VPN Security Gateway (IKEv2 Responder) not the VPN Client
- Generated dynamically with system informations (UID)
  - Used by the Sender not the Destination

Special words:

- ANY: any value
- OPAQUE: field should not be looked at

IPsec Architecture - Security Policy Database

Generating SPD Cache:

- PFP defines whether:
  - Any TS match is associated to a different SA
  - All TS matches are associated to a single SA

SPD is an ordered database

SPD Cache lookup is generally faster, as such for outbound traffic SPD Cache lookup is usually performed first, follwed by a SPD lookup.

IPsec Architecture: SPD - Security Gateway Example

A Security Gateway:

- PROTECTes traffic of all the users@company.com
  - Each user is provided an IP address
  - Each user has a specific IPsec session (a different key)
- DISCARDs any other traffic

The ordered SPD will look like:

- FQDN *company.com* PROTECT
  - for each IP generates a specific SA
- ANY DISCARD

VPN Example: Initial SPD - SG

- BYPASSing IKEv2 channel

Traffic Selectors	PFP	SPD Cache	Processing
IP_dst: ANY	0	IP_{MN}^{outer}	Policy: BYPASS
IP_src: IP_{SG}	0	IP_{SG}	
Port_dst: 500	0	500	
Port_src: 500	0	500	
IP_dst: IP_{SG}	0	IP_{SG}	Policy: BYPASS
IP_src: ANY	0	IP_{MN}^{outer}	
Port_dst: 500	0	500	
Port_src: 500	0	500	

VPN Example: Initial SPD - SG

- Protecting the VPN Client traffic

Traffic Selectors	PFP	SPD Cache	Processing
Name: company.com	0	-	Policy: PROTECT
IP_dst: ANY	0	ANY	Mode: Tunnel
IP_src: ANY	1	IP_{MN}^{innner}	Tunnel Header: IP_{MN}^{outer} , IP_{SG}
Port_dst: ANY	0	ANY	Protocol: ESP
Port_src: ANY	0	ANY	Algorithm: AES-GCM

VPN Example: Initial SPD - SG

- Protecting the VPN Client traffic

Traffic Selectors	PFP	SPD Cache	Processing
Name: company.com	0	-	Policy: PROTECT
IP_dst: ANY	1	IP_{MN}^{innner}	Tunnel Header: IP_{SG}, IP_{MN}^{outer}
IP_src: ANY	0	ANY	Mode: Tunnel
Port_dst: ANY	0	ANY	Protocol: ESP
Port_src: ANY	0	ANY	Algorithm: AES-GCM

VPN Example: Initial SPD - SG

- Preventing any other traffic

Traffic Selectors	PFP	SPD Cache	Processing
IP_dst: ANY	0	ANY	Policy: DISCARD
IP_src: ANY	0	ANY	
Port_dst: ANY	0	ANY	
Port_src: ANY	0	ANY	

IPsec Architecture: SAD

SA implements the policy associated to the traffic:

- Processing info to implement the PROTECT Policy:
  - keying meterial and algorithm
  - IP processing information (ESP, AH, mode)
- Traffic Selectors (TS):
  - Source / Destination IP Address
  - Next Layer Protocol,
  - Source / Destination Port ...
- Security Protocol Index (SPI)

Note:

- SA TS are used to perform the SPD lookup
- SPI identifies the SA associated to encrypted packet

IPsec Architecture: SAD

Outbound packets use SA to implement a PROTECT SP:

- The appropriated SA is provided *by* the SP matching the clear text packet, using internal structure of the system (pointer)
- The SA enables encryption of the packet

Inbound packet uses the SA to implement a PROTECT SP:

- The appropriated SA is provided *by* the SPI
- The SA enables encryption of the packet

IPsec Architecture: SAD - SA Structure

The SA contains the following main information:

- Cryptographic material
  - keys and cryptographic algorithms
  - IPsec mode: Tunnel or Transport
  - Security Protocols: AH / ESP
  - SA lifetime
- IPsec signaling:
  - SPI
  - Sequence Number
  - Sequence Counter overflow
  - Anti Reaply Window
- Traffic Selector
  - inner packets

IPsec Architecture: SAD - SA Structure

The SA contains the following additional information:

- Additional checking information:
  - Stateful fragment checking
  - Don't Fragment (DF) bit
  - Differentiated Services Code Point (DSCP)
  - MTU

IPsec Architecture: PAD

Peer Authorization Database:

- Contains information on who can communicate with the IPsec layer and how IKEv2 must proceed to the authentication.
- Should be seen as a meta database that provides inputs for IKEv2 to configure the Security Policies.

IPsec ESP / AH

IPsec protectes traffic with a combination of two protocols:

- ESP Encapsulated Security Payload securing the IP payload
    - ESP leaves unprotected the IP header
    - Widely deployed
- AH Authentication Header authenticating the IP packet
    - NAT change the IP headers
    - Hardly deployed

IPsec Encapsulated Security Payload

```
0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ----
|               Security Parameters Index (SPI)                 | ^Int.
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|                      Sequence Number                          | |ered
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | ----
|                    Payload Data* (variable)                   | |   ^
~                                                               ~ |   |
|                                                               | |Conf.
+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|               |     Padding (0-255 bytes)                     | |ered*
+-+-+-+-+-+-+-+-+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |   |
|                               |  Pad Length   | Next Header   | v   v
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ------
|         Integrity Check Value-ICV   (variable)                |
~                                                               ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPsec Encapsulated Security Payload

- SPI: identifies the SA for inbound packets
- Sequence Number: to provide anti-replay protection
- Payload Data: The clear text information carried by the IPsec packet
- Padding: provides 32 bit alignment
- ICV: authenticate the packet

IPsec ESP Transport mode

```
                 BEFORE APPLYING ESP
            ---------------------------------------
      IPv6  |             | ext hdrs |      |      |
            | orig IP hdr |if present| TCP  | Data |
            ---------------------------------------

                 AFTER APPLYING ESP
            ---------------------------------------------------------
      IPv6  | orig |hop-by-hop,dest*,|   |dest|   |    | ESP   | ESP|
            |IP hdr|routing,fragment.|ESP|opt*|TCP|Data|Trailer| ICV|
            ---------------------------------------------------------
                                         |<--- encryption ---->|
                                     |<------ integrity ------>|

                * = if present, could be before ESP, after ESP, or both
```

IPsec ESP Tunnel mode

```
                 BEFORE APPLYING ESP
            ---------------------------------------
      IPv6  |                | ext hdrs |      |      |
            | orig IP hdr |if present| TCP | Data |
            ---------------------------------------

                AFTER APPLYING ESP

            ------------------------------------------------------------
      IPv6  | new* |new ext |   | orig*|orig ext |   |    | ESP   | ESP|
            |IP hdr| hdrs*  |ESP|IP hdr| hdrs *  |TCP|Data|Trailer| ICV|
            ------------------------------------------------------------
                                |<--------- encryption ---------->|
                            |<------------ integrity ------------>|

            * = if present, construction of outer IP hdr/extensions and
                modification of inner IP hdr/extensions is discussed in
                the Security Architecture document.
```

IPsec Authentication Header

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  Payload Len  |          RESERVED             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Security Parameters Index (SPI)               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Sequence Number Field                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                Integrity Check Value-ICV (variable)           |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPsec AH Transport mode

```
                  BEFORE APPLYING AH
            ----------------------------------------
IPv6       |                 | ext hdrs |     |      |
           | orig IP hdr |if present| TCP | Data |
            ----------------------------------------

                  AFTER APPLYING AH
            ------------------------------------------------------------
IPv6       |             |hop-by-hop, dest*, |    | dest |     |      |
           |orig IP hdr  |routing, fragment. | AH | opt* | TCP | Data |
            ------------------------------------------------------------
           |<--- mutable field processing -->|<-- immutable fields -->|
           |<---- authenticated except for mutable fields ----------->|

                 * = if present, could be before AH, after AH, or both
```

IPsec AH Tunnel mode

```
                 BEFORE APPLYING AH
            ----------------------------------------
      IPv6  |              | ext hdrs |      |      |
            | orig IP hdr  |if present| TCP  | Data |
            ----------------------------------------

                 AFTER APPLYING AH
           --------------------------------------------------------------
      IPv6 |            | ext hdrs*|    |             | ext hdrs*|   |    |
           |new IP hdr* |if present| AH |orig IP hdr*|if present|TCP|Data|
           --------------------------------------------------------------
           |<--- mutable field -->|<---------- immutable fields -------->|
           |       processing     |
           |<-- authenticated except for mutable fields in new IP hdr ->|

             * = if present, construction of outer IP hdr/extensions and
                 modification of inner IP hdr/extensions is discussed in
                 the Security Architecture document.
```

IKEv2

IKEv2 is the Internet Key Exchange Protocol:

- IKEv2 authenticates the peers
- Establishes a control channel between the peers
    - IKE_SA
- Negotiates the SA between the peers
- Manage the SA:
    - SA life cycle (rekeying)
    - Update SA in mobility / multihoming scenarios

IKEv2

The IKEv2 negotiation performs the following exchange type:

- IKE_INIT
- IKE_AUTH
- CREATE_CHILD_SA

```
Initiator                                  Responder
-------------------------------------------------------------------
HDR, SAi1, KEi, Ni  -->
                             <--  HDR, SAr1, KEr, Nr, [CERTREQ]

HDR, SK {IDi, [CERT,] [CERTREQ,]
        [IDr,] AUTH, SAi2,
        TSi, TSr}  -->
                            <--  HDR, SK {IDr, [CERT,] AUTH,
                                          SAr2, TSi, TSr}
```

INFORMATIONAL exchanges are performed latter

IKEv2 IKE_INIT

IKE_INIT results in the agreement of:

- SKEYSEED from which all IKE_SA keys are derives
- cryptographic algorithm use for the IKE channel (IKE_SA)

```
Initiator                                   Responder
-------------------------------------------------------------------
HDR, SAi1, KEi, Ni  -->
                                          <--  HDR, SAr1, KEr, Nr, [CERTREQ]
```

- HDR: IKE Header - SPI, version, Exchange Type, Message ID
- SAi1, SAr1: proposal and selection of IKE_SA crypto algorithm
- KEi, KEr: public Diffie Hellman Value to generate SKEYSEED
- Ni, Nr: nonces to generate SKEYSEED

At this stage the peers have exchanged a shared secret, but...

- Peers are still unauthenticated
- The IKEv2 channel is still unencrypted
- Negotiation of the SA has not yet even started.

Focus on Diffie Hellman

Diffie Hellman enables Alice and Bob to agree on a shared secret without revealing that secret.

Alice and Bob agree on a finite cyclic group G:

- of order n -- a very large number (2048 bit long)
- and a generator g in G.

```
Alice                              Bob

a                g^a mod n
                --------->
                 g^b mod n      b
                <---------
(g^b)^a  mod n                  (g^a)^b mod n
```

Discrete logarithm says that it is hard to infer a from g^a

Focus on Diffie Hellman

Diffie Hellman is sensible to the man-in-the-middle attack

```
Alice                    Eve                   Bob

a       g^a mod n      c    g^c mod n
       ---------->         ---------->
        g^b mod n  b        g^d mod n   d
       <----------         <----------
(g^b)^a mod n    (g^c)^c mod n       (g^c)^d mod n
```

Focus on Key derivation

Protected communications usually involve multiple keys:

- Each key is usually dedicated to a specific task:
  - encryption, authentication, ...
  - initiator, responder, ...

To avoid one negotiation per key, we usually have:

1. Negotiation of a shared secret
2. Derivation of the keys from the share secret

Focus on Key derivation

The agreed shared secret is used to derive multiple secrets to:

- Protect the IKE_SA IKEv2 Channel:
    - Authentication SK_a
    - Encryption SK_e
- Derive the keying material associated to the SA
    - SK_d
- Authenticate the peers
    - SK_p

```
SKEYSEED = prf(Ni | Nr, g^ir)
{SK_d | SK_ai | SK_ar | SK_ei | SK_er | SK_pi | SK_pr}
         = prf+ (SKEYSEED, Ni | Nr | SPIi | SPIr)
```

IKEv2 IKE_AUTH

```
Initiator                                  Responder
-------------------------------------------------------------------
HDR, SK {IDi, [CERT,] [CERTREQ,]
    [IDr,] AUTH }  -->
                            <--  HDR, SK {IDr, [CERT,] AUTH}
```

- SK: encryption key
- IDi, IDr: Identities
- CERT: public key asociated to ID (optional)
- CERTREQ: Trusted Anchors
- AUTH: authentication payload

Focus on Digital Signature

The purpose of digital signature is to cryptographicaly prove:

- It is really *me* sending the message
- The message you receive is the one I have sent, e.g. has not been altered

A cryptographic signature is:

1. Generated by the owner of the private key
2. Validated with the associated public key, the signature and message

```
Alice                                           Bob

k_prv = private_key,
k_pbl = public_key,
m = message

s = sign(k_prv, m)

               m, s, k_pbl
               ------------------------->
                                   verify(m, s, k_pbl)
```

Focus on AUTH

AUTH is not limited to a single message m, but the IKE_SA, that is:

- IKE_INIT (where the shared secret has been agreed)
- IKE_AUTH with IDs, CERT

AUTH can be seen as the signature where the signed data is:

- A known structure with sent and received information
- Built by each peer for signing / verifying

What has been sent corresponds to what has been received

Focus on AUTH

Building AUTH:

- build data
- sign data

Verifying AUTH:

- build peer's data
- verifying the signature

```
AUTHi = sign( IKE_INIT_request', Nr, prf(SK_pi, IDi') )
AUTHr = sign( IKE_INIT_response', Ni, prf(SK_pr, IDr') )
```

Focus on IDs

AUTH enables to binds the IKE_SA to the owner of the private key

This onwer is designated by its ID, but:

- How to ensure ID owns the private key ?
- How to make sure it is an authorized peer ?

CERT binds the ID and the public Key

- This may involve a Trusted Anchor (CA)

PAD indicates the peer is authorized

IKEv2 CREATE_CHILD_SA

CREATE_CHILD_SA negotiates the SA:

- Traffic Selectors (TS)
- Cryptographic Keys and algorithms
- Protocols (ESP, AH)
- IPsec Mode (default Tunnel)
- ...

IKEv2 CREATE_CHILD_SA

```
Initiator                                         Responder
-------------------------------------------------------------------
HDR, SK {SAi2, TSi, TSr}  -->
                             <--  HDR, SK {SAr2, TSi, TSr}
```

- SAi2, SAr2: proposal and selection of IKE_SA crypto algorithm
- TSi, TSr: Traffic Selectors

IKEv2 INFORMATIONAL exchanges

INFORMATIONAL exchange are based on query / response.

- used to Delete SA among other

```
Initiator                                  Responder
-------------------------------------------------------------------
HDR, SK {[N,] [D,]
[CP,] ...}  -->
                                   <--  HDR, SK {[N,] [D,]
                                             [CP,] ...}
```

- D Delete Payload

Other exchange mais involve other payloads:

- N Notify Payload
- CP : Configuration Payload (VPN)
- ...

VPN Example: Initial SPD - Security Gateway

SPD of the VPN Security Gateway

- BYPASSing IKEv2 channel

Traffic Selectors	PFP	SPD Cache	Processing
IP_dst: ANY	0	IP_{MN}^{outer}	Policy: BYPASS
IP_src: IP_{SG}	0	IP_{SG}	
Port_dst: 500	0	500	
Port_src: 500	0	500	

Traffic Selectors	PFP	SPD Cache	Processing
IP_dst: IP_{SG}	0	IP_{SG}	Policy: BYPASS
IP_src: ANY	0	IP_{MN}^{outer}	
Port_dst: 500	0	500	
Port_src: 500	0	500	

VPN Example: Initial SPD - Security Gateway

- Protecting the VPN Client traffic

Traffic Selectors	PFP	SPD Cache	Processing
Name: company.com	0	-	Policy: PROTECT
IP_dst: ANY	0	ANY	Mode: Tunnel
IP_src: ANY	1	IP_{MN}^{innner}	Tunnel Header: IP_{MN}^{outer} , IP_{SG}
Port_dst: ANY	0	ANY	Protocol: ESP
Port_src: ANY	0	ANY	Algorithm: AES-GCM

Traffic Selectors	PFP	SPD Cache	Processing
Name: company.com	0	-	Policy: PROTECT
IP_dst: ANY	1	IP_{MN}^{innner}	Tunnel Header: IP_{SG}, IP_{MN}^{outer}
IP_src: ANY	0	ANY	Mode: Tunnel
Port_dst: ANY	0	ANY	Protocol: ESP
Port_src: ANY	0	ANY	Algorithm: AES-GCM

VPN Example: Initial SPD, SAD - Security Gateway

- Preventing any other traffic

Traffic Selectors	PFP	SPD Cache	Processing
IP_dst: ANY	0	ANY	Policy: DISCARD
IP_src: ANY	0	ANY	
Port_dst: ANY	0	ANY	
Port_src: ANY	0	ANY	

- SPD Cache has not been populated
- SAD has not been populated

VPN Example: Initial SPD, SAD - VPN Client

SPD of the VPN Client

Traffic Selectors	PFP	SPD Cache	Processing
Name: IKEv2 UID	0	-	Policy: PROTECT
IP_dst: ANY	0	ANY	Mode: Tunnel
IP_src: ANY	1	IP_{MN}^{innner}	Tunnel Header: IP_{MN}^{outer} , IP_{SG}
Port_dst: ANY	0	ANY	Protocol: ESP
Port_src: ANY	0	ANY	Algorithm: AES-GCM

Traffic Selectors	PFP	SPD Cache	Processing
Name: IKEv2 UID	0	-	Policy: PROTECT
IP_dst: ANY	1	IP_{MN}^{innner}	Tunnel Header: IP_{SG}, IP_{MN}^{outer}
IP_src: ANY	0	ANY	Mode: Tunnel
Port_dst: ANY	0	ANY	Protocol: ESP
Port_src: ANY	0	ANY	Algorithm: AES-GCM

VPN Example: Initial SPD, SAD - VPN Client

- VPN Client BYPASS and DISCARD Policies are as those of Security Gateway
- SP has not been populated in the SPD Cache
- SA has not been populated in the SAD

VPN Example: IKEv2 Exchange

VPN Client initiates an Session with Security Gateway

- Echange happen between IP_{MN}^{outer}:500 and IP_{SG}:500

```
Initiator                          Responder
-----------                        -----------
HDR, SAi1, KEi, Ni  -->
                      <--  HDR, SAr1, KEr, Nr

HDR, SK { IDi, CERT, AUTH,
          CP(CFG_REQUEST),
          SAi2, TSi, TSr }  -->
                      <--  (IP_R1:500 -> IP_I1:500)
                           HDR, SK { IDr, CERT, AUTH,
                                     CP(CFG_REPLY),
                                     SAr2, TSi, TSr }
```

- IPsec mode is tunnel
    - Tunnel IPs (IP_{MN}^{outer}, IP_{SG}) are read from IP header
- IP_{MN}^{inner} is provided by the Configuration Payload CFG_REPLY

VPN Example: SPD Cache

SPD of the VPN Security Gateway

- BYPASSing IKEv2 channel

Traffic Selectors	Processing
IP_dst: IP_{MN}^{outer}	Policy: BYPASS
IP_src: IP_{SG}	
Port_dst: 500	
Port_src: 500	

Traffic Selectors	Processing
IP_dst: IP_{SG}	Policy: BYPASS
IP_src: IP_{MN}^{outer}	
Port_dst: 500	
Port_src: 500	

VPN Example: SPD Cache

- Protecting the VPN Client traffic

Traffic Selectors	Processing
IP_dst: ANY	$SA_{Clt->SG}$
IP_src: IP_{MN}^{innner}	
Port_dst: ANY	
Port_src: ANY	

Traffic Selectors	Processing
IP_dst: IP_{MN}^{innner}	$SA_{SG->Clt}$
IP_src: ANY	
Port_dst: ANY	
Port_src: ANY	

VPN Example: SPD Cache

- Preventing any other traffic

Traffic Selectors	SPD Cache	Processing
IP_dst: ANY	ANY	Policy: DISCARD
IP_src: ANY	ANY	
Port_dst: ANY	ANY	
Port_src: ANY	ANY	

VPN Example: SAD

SAD of the VPN Client, Security Gateway

Action	Traffic Selectors	IPsec Signaling	Crypto material
$SPI_{Clt->SG}$	IP_dst: ANY	IP_{src}^{Tunnel}: IP_{MN}^{outer}	Crypto: K_e, K_a
	IP_src: IP_{MN}^{inner}	IP_{dst}^{Tunnel}: IP_{SG}	Counters: C_{ESP}, C_w
	Port_dst: ANY	Proto: ESP	
	Port_src: ANY	Algo: AES-GCM	
$SPI_{SG->Clt}$	IP_dst: IP_{MN}^{inner}	IP_{src}^{Tunnel}: IP_{SG}	Crypto: K_e, K_a
	IP_src: ANY	IP_{dst}^{Tunnel}: IP_{MN}^{outer}	Counters: C_{ESP}, C_w
	Port_dst: ANY	Proto: ESP	
	Port_src: ANY	Algo: AES-GCM	

VPN Example: Encrypted Traffic

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
I|version| traffic class |               flow label              |
P+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
v|         payload length        |  next header  |   hop limit   |
6+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ~                      inner source IP                          ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ~                    inner destination IP                       ~
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+---
E|               Security Parameters Index (SPI)                 |  ^
S+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
P|                      Sequence Number (SN)                     |  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
 |                                                               |  |
 |                             IV                                |  |
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- |
I|version| traffic class |               flow label              |^ |
P+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+| |
v|         payload length        |  next header  |   hop limit   || |
6+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+| a
 ~                      inner source IP                          ~| u
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+e t
 ~                    inner destination IP                       ~n h
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+c e
T|          source port          |           dest port           |r n
C+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+y t
P|                      Sequence Number (SN)                     |p i
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+t c
 |                     ACK Sequence Number                       |e a
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+d t
 |Off. | Rserv |      Flags      |         Window Size           || e
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+| d
 |             Checksum          |      Urgent Pointer           || |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+| |
 |                                                               || |
 ~                        APPLICATION DATA                       ~| |
 |                                                               || |
-|                                               +-+-+-+-+-+-+-+-+| |
E|                                               |    Padding    || |
S+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+| |
P|     Padding (continue)        |  Pad Length   | Next Header   |V V
-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+---
 |         Integrity Check Value-ICV   (variable)                |
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

MOBIKE

IP adresses of a packet are important Traffic Selectors likely used in:

- SPD, SPD Cache
- SAD

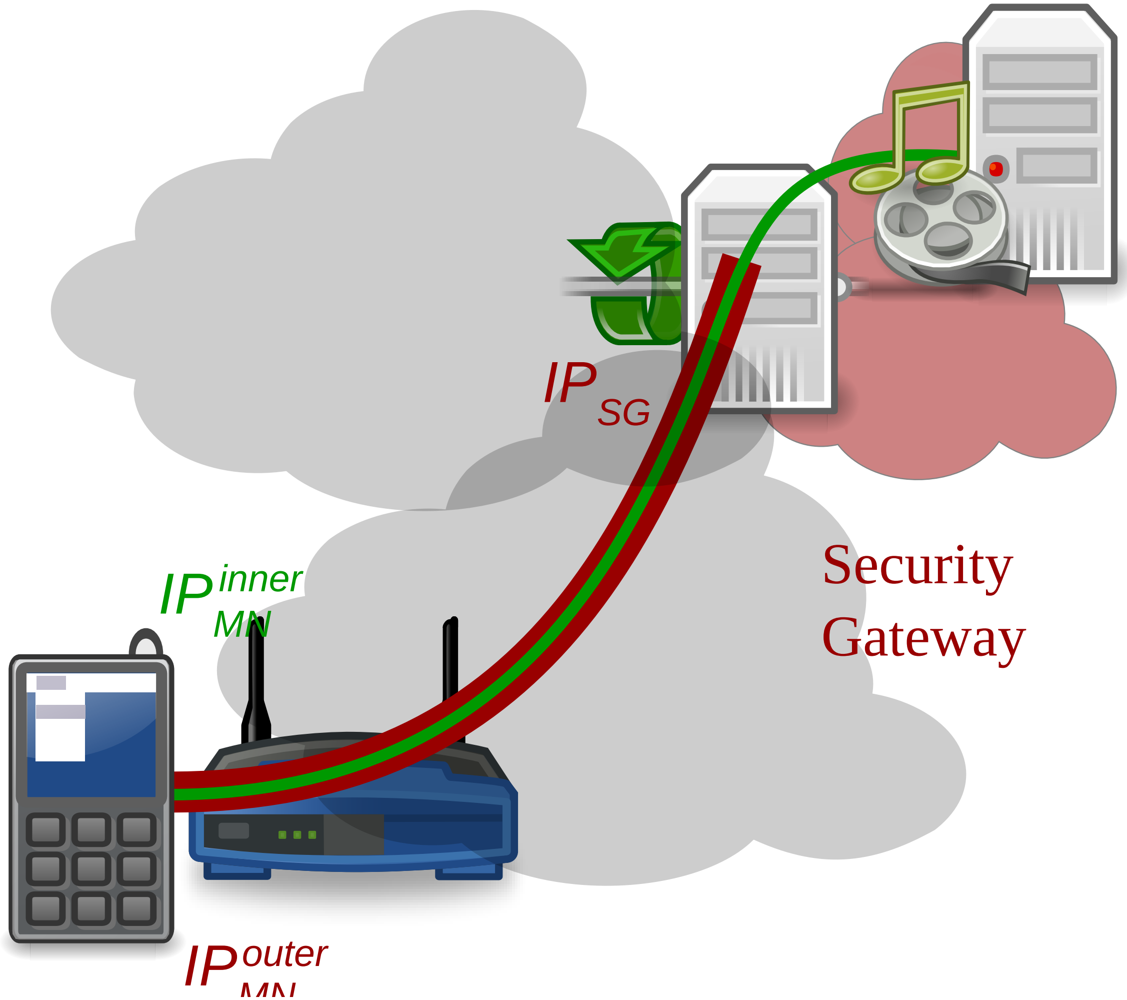
A Mobile Node (MN) IP changes need to be reflected in IPsec SAD/SPD:

- MOBIKE is the IKEv2 extension to enable updating IPsec configuration in Mobility and Multihoming environment

MOBIKE - Mobility

Initial Phase:

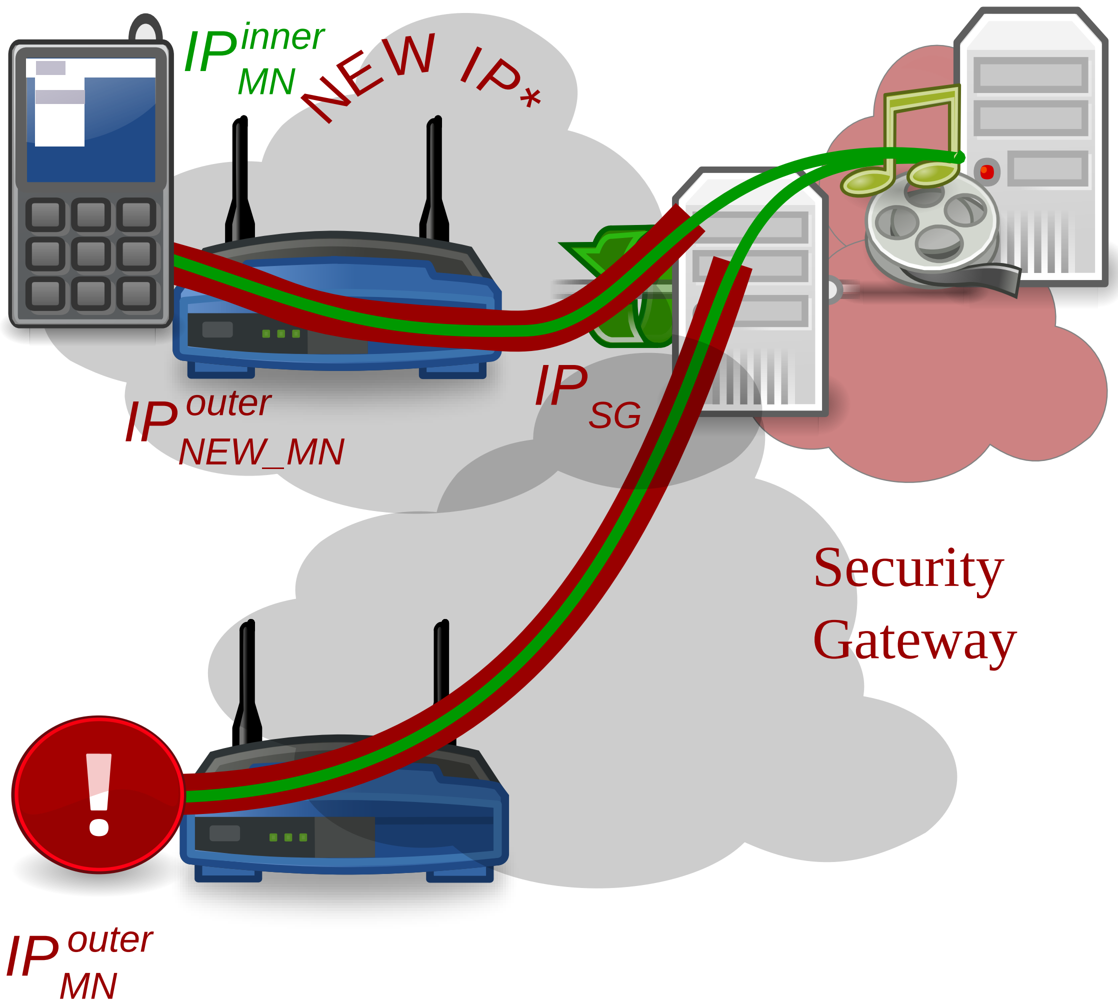
- MN connects a server with a private session: IP^{inner}_{MN} , IP^{inner}_{MEDIA}
- MN tunnels the private session to the Security Gateway: IP^{outer}_{MN} and IP_{SG}



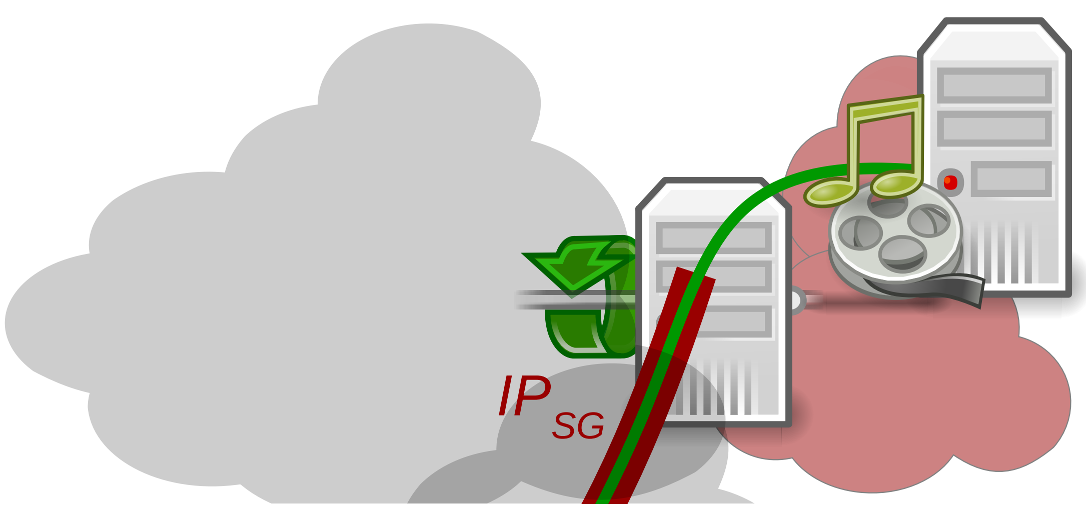
MOBIKE - Mobility

Mobility Phase:

- MN changes ISP and the tunnel endpoint becomes $IP^{outer}_{NEW\_MN}$



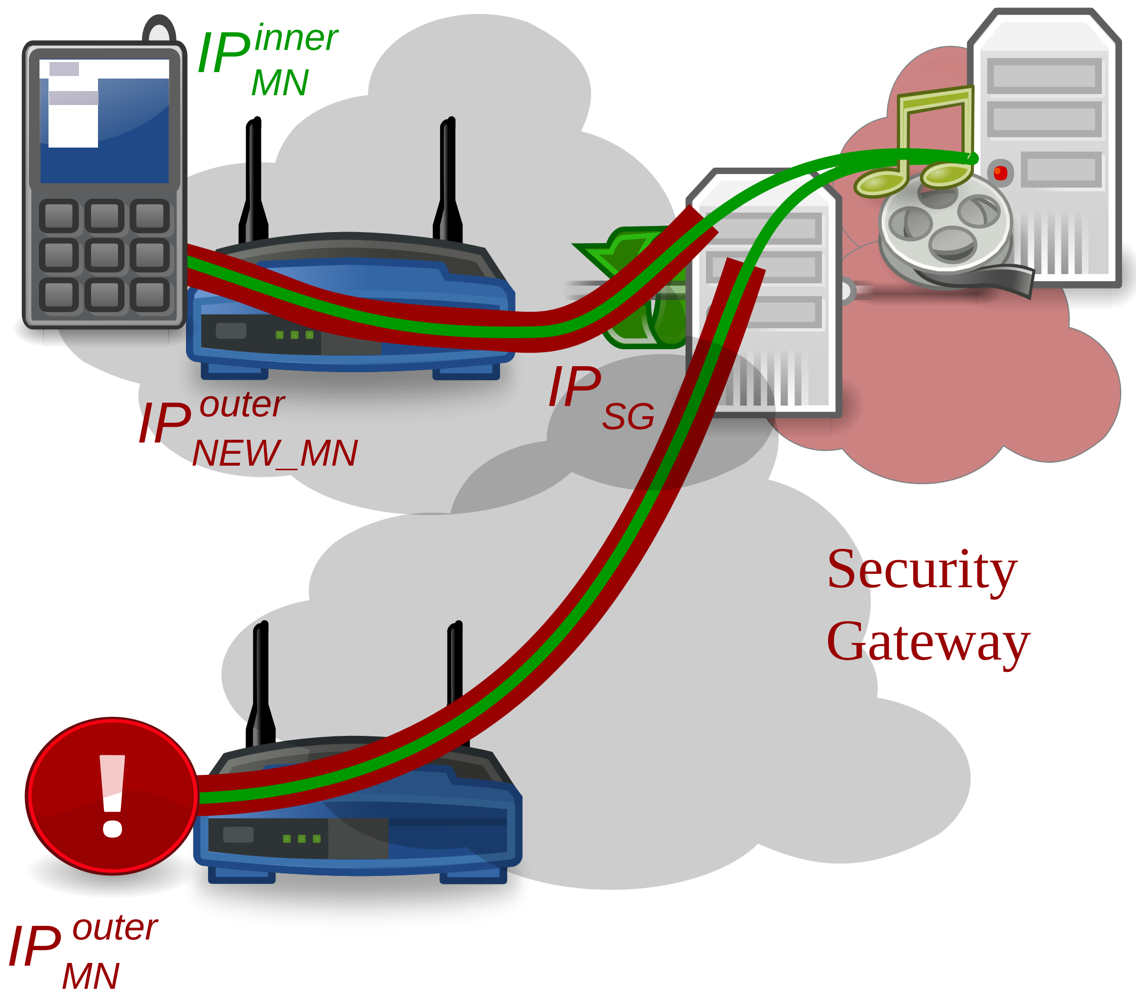
MN advertises being also reachable on IP_{NEW_MN} if IP_{MN} fails



MOBIKE - Multihoming

Mobility Phase:

- MN tunnel endpoint IP_{MN}^{outer} is not reachable
- Media server switches traffic to $IP_{NEW\_MN}^{outer}$



MOBIKE - SPD - Cache

SPD Cache is not impacted by the change of IP address

- SPD concerns the inner traffic

Traffic Selectors	Processing
IP_dst: ANY	$SA_{Clt->SG}$
IP_src: IP_{MN}^{innner}	
Port_dst: ANY	
Port_src: ANY	

Traffic Selectors	Processing
IP_dst: IP_{MN}^{innner}	$SA_{SG->Clt}$
IP_src: ANY	
Port_dst: ANY	
Port_src: ANY	

MOBIKE - SAD Initial State

SAD (Outbound Traffic)

SAD of the VPN Client, Security Gateway

Action	Traffic Selectors	IPsec Signaling	Crypto material
$SPI_{Clt->SG}$	IP_dst: ANY	$IP_{src}^{Tunnel}: IP_{MN}^{outer}$	Crypto: K_e, K_a
	IP_src: IP_{MN}^{inner}	$IP_{dst}^{Tunnel}: IP_{SG}$	Counters: C_{ESP}, C_w
	Port_dst: ANY	Proto: ESP	
	Port_src: ANY	Algo: AES-GCM	
$SPI_{SG->Clt}$	IP_dst: IP_{MN}^{inner}	$IP_{src}^{Tunnel}: IP_{SG}$	Crypto: K_e, K_a
	IP_src: ANY	$IP_{dst}^{Tunnel}: IP_{MN}^{outer}$	Counters: C_{ESP}, C_w
	Port_dst: ANY	Proto: ESP	
	Port_src: ANY	Algo: AES-GCM	

SAD (Outbound Traffic)

SAD of the VPN Client, Security Gateway

Action	Traffic Selectors	IPsec Signaling	Crypto material
$SPI_{Clt->SG}$	IP_dst: ANY	IP_{src}^{Tunnel}: $IP_{NEW\_MN}^{outer}$	Crypto: K_e, K_a
	IP_src: IP_{MN}^{inner}	IP_{dst}^{Tunnel}: IP_{SG}	Counters: C_{ESP}, C_w
	Port_dst: ANY	Proto: ESP	
	Port_src: ANY	Algo: AES-GCM	
$SPI_{SG->Clt}$	IP_dst: IP_{MN}^{inner}	IP_{src}^{Tunnel}: IP_{SG}	Crypto: K_e, K_a
	IP_src: ANY	IP_{dst}^{Tunnel}: $IP_{NEW\_MN}^{outer}$	Counters: C_{ESP}, C_w
	Port_dst: ANY	Proto: ESP	

MOBIKE - IKEv2 Exchanges

In order to provide IPsec in mobility / multihoming environement

- SA needs to be updated in sync
- Mobility:
  - the MN needs to be able to update its SA
  - the MN needs to inform the SG of the new IP address
  - the SG needs to be able to update the SA
- Multihoming:
  - the MN needs to provide alternate IP addresses
  - the SG neeeds to update its SA
  - the SG needs to check reachability
  - the MN needs to update its SA

MOBIKE - Mobility - IKEv2 Initial Exchanges

```
       Initiator                           Responder
      -----------                         -----------
    1) (IP_I1:500 -> IP_R1:500)
       HDR, SAi1, KEi, Ni,
            N(NAT_DETECTION_SOURCE_IP),
            N(NAT_DETECTION_DESTINATION_IP)  -->

                                 <--  (IP_R1:500 -> IP_I1:500)
                                      HDR, SAr1, KEr, Nr,
                                           N(NAT_DETECTION_SOURCE_IP),
                                           N(NAT_DETECTION_DESTINATION_IP)

    2) (IP_I1:4500 -> IP_R1:4500)
       HDR, SK { IDi, CERT, AUTH,
                 CP(CFG_REQUEST),
                 SAi2, TSi, TSr,
                 N(MOBIKE_SUPPORTED) }  -->

                                 <--  (IP_R1:4500 -> IP_I1:4500)
                                      HDR, SK { IDr, CERT, AUTH,
                                                CP(CFG_REPLY),
                                                SAr2, TSi, TSr,
                                                N(MOBIKE_SUPPORTED) }
```

MOBIKE - Mobilty - Update

Initiator gets information from lower layers that its attachment point and address have changed.

```
    3) (IP_I2:4500 -> IP_R1:4500)
       HDR, SK { N(UPDATE_SA_ADDRESSES),
                 N(NAT_DETECTION_SOURCE_IP),
                 N(NAT_DETECTION_DESTINATION_IP) }  -->

                             <-- (IP_R1:4500 -> IP_I2:4500)
                                 HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                                      N(NAT_DETECTION_DESTINATION_IP) ]
```

MOBIKE - Mobility - Routability Check

Responder verifies that the initiator has given it a correct IP address.

```
  4)                         <-- (IP_R1:4500 -> IP_I2:4500)
                                 HDR, SK { N(COOKIE2) }

     (IP_I2:4500 -> IP_R1:4500)
     HDR, SK { N(COOKIE2) }  -->
```

MOBIKE - Multihoming - IKEv2 Initial Exchanges

```
   Initiator                           Responder
  -----------                         -----------
1) (IP_I1:500 -> IP_R1:500)
   HDR, SAi1, KEi, Ni,
        N(NAT_DETECTION_SOURCE_IP),
        N(NAT_DETECTION_DESTINATION_IP)  -->

                              <--  (IP_R1:500 -> IP_I1:500)
                                   HDR, SAr1, KEr, Nr,
                                        N(NAT_DETECTION_SOURCE_IP),
                                        N(NAT_DETECTION_DESTINATION_IP)

2) (IP_I1:4500 -> IP_R1:4500)
   HDR, SK { IDi, CERT, AUTH,
             CP(CFG_REQUEST),
             SAi2, TSi, TSr,
             N(MOBIKE_SUPPORTED) }  -->

                              <--  (IP_R1:4500 -> IP_I1:4500)
                                   HDR, SK { IDr, CERT, AUTH,
                                             CP(CFG_REPLY),
                                             SAr2, TSi, TSr,
                                             N(MOBIKE_SUPPORTED),
                                             N(ADDITIONAL_IP4_ADDRESS) }
```

MOBIKE - Detection

The initiator suspects a problem in the currently used address pair and probes its liveness.

```
  3) (IP_I1:4500 -> IP_R1:4500)
      HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                N(NAT_DETECTION_DESTINATION_IP) }  -->

      (IP_I1:4500 -> IP_R1:4500)
      HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                N(NAT_DETECTION_DESTINATION_IP) }  -->
```

MOBIKE - Detection

Eventually, the initiator gives up on the current address pair and tests the other available address pair.

```
    4) (IP_I1:4500 -> IP_R2:4500)
       HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                 N(NAT_DETECTION_DESTINATION_IP) }

                            <--  (IP_R2:4500 -> IP_I1:4500)
                                 HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                                           N(NAT_DETECTION_DESTINATION_IP)
```

MOBIKE - Update

This worked, and the initiator requests the peer to switch to new addresses.

```
5) (IP_I1:4500 -> IP_R2:4500)
   HDR, SK { N(UPDATE_SA_ADDRESSES),
             N(NAT_DETECTION_SOURCE_IP),
             N(NAT_DETECTION_DESTINATION_IP),
             N(COOKIE2) }  -->

                         <--  (IP_R2:4500 -> IP_I1:4500)
                              HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                                        N(NAT_DETECTION_DESTINATION_IP),
                                        N(COOKIE2) }
```

Reference

- Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005
- Kent, S. "IP Authentication Header", RFC 4302
- Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005
- Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 7296, October 2014
- P. Eronen, "IKEv2 Mobility and Multihoming Protocol (MOBIKE)" RFC 4555, June 2006

Acknowledgment

- Open Security Architecture