IBS:
Bezpečnost a počítačové sítě

Site-to-Site VPNs

Brno University of Technology, Faculty of Information Technology
Božetěchova 1/2, 612 66 Brno - Královo Pole
Vladimír Veselý, veselyv@fit.vut.cz

10th March 2025

- Tři základní vlastnosti

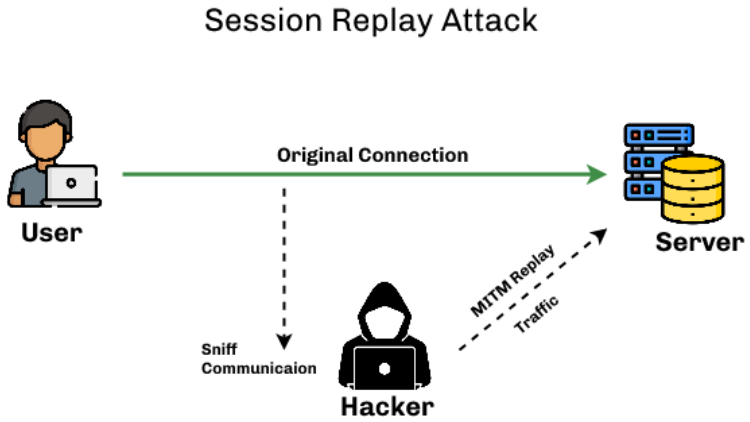
Autenticita

Integrita

Důvěrnost

Anti-replay Protection

- The primary purpose of **anti-replay protection** is to prevent an attacker from intercepting and retransmitting legitimate data packets to gain unauthorized access or disrupt network operations

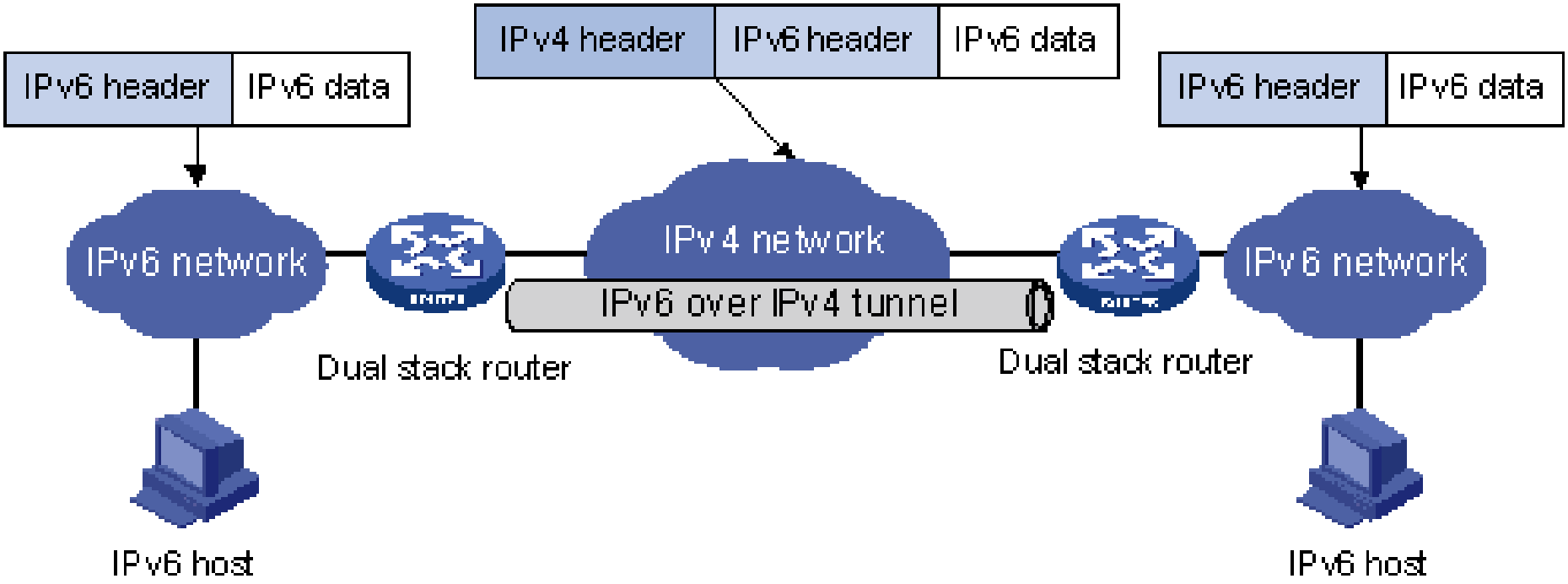


- **Sequence Numbers**
  - Each message or packet is assigned a unique sequence number.
  - The receiver keeps track of these numbers to ensure that any message received is fresh and has not been seen before.
- **Sliding Window**
  - Many protocols implement a "window" that represents a range of acceptable sequence numbers.
  - Only messages within this window are accepted.
- **Timestamping**
  - In some implementations, timestamps are used to verify the recency of messages. If a message is too old, it may be rejected even if its sequence number is within the expected range.

What Is Tunneling?

- Many times it's useful to create "illusion" of the new network above the existing one. Here are some motivations:
  - *Existing network doesn't recognize protocol which we would need to transfer across it or service we would like to use*
  - *We would like to use existing network as transport tool but we want it to be completely invisible from point of view of internal network*
  - *We need to interconnect multiple sites with potentially private IP address space*
  - *We don't trust existing network and we want to securely transfer data across it*
- **Tunneling** = technique where packet is reencapsulated into the new packet
  - Former packet becomes payload of new packet and therefore its content (L3 header) is not in attention of routers

Dual-Stack


IPv6 header
IPv6 data
IPv4 header
IPv6 header
IPv6 data
IPv6 header
IPv6 data
IPv6 network
IPv4 network
IPv6 network
IPv6 over IPv4 tunnel
Dual stack router
Dual stack router
IPv6 host
IPv6 host

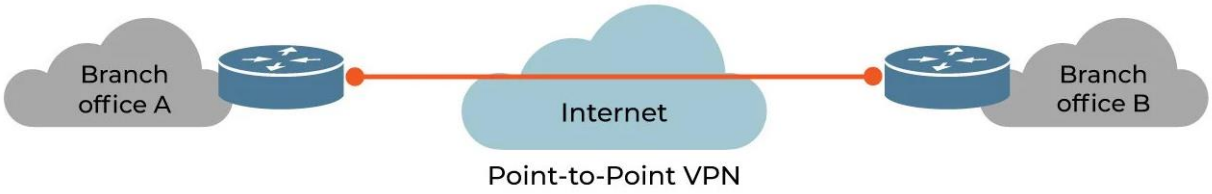
Protocols Used in Tunneling

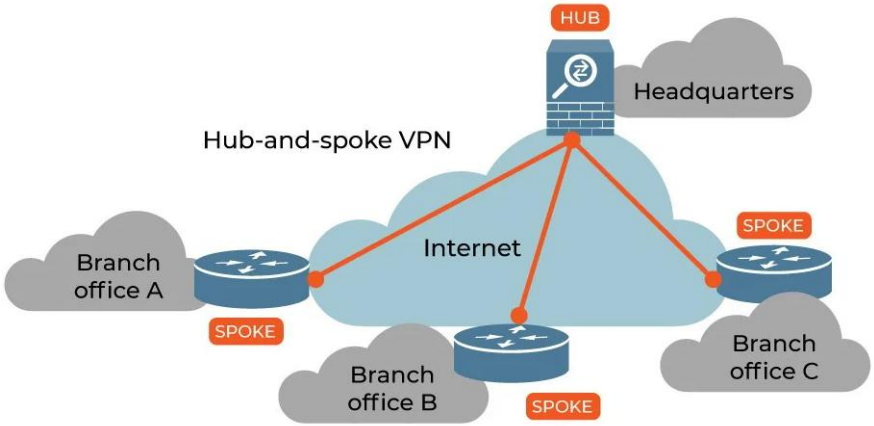
- **Passenger protocol**
  - *We would like to transfer datagrams of this protocol inside tunnel*
  - E.g. IPX, AppleTalk, IPv4, IPv6
- **Encapsulating/Tunneling protocol**
  - Header of this protocol is prepended before passenger protocol
  - It's used to identify passenger protocol and secure transmission with authentication, encryption, etc.
  - E.g. GRE, IPsec, L2F, PPTP, L2TP
- **Carrier protocol**
  - Existing network uses this protocol for transport and inside it encapsulating protocol is carried wrapped around passenger protocol
  - E.g. IP, Frame-relay, ATM, Ethernet

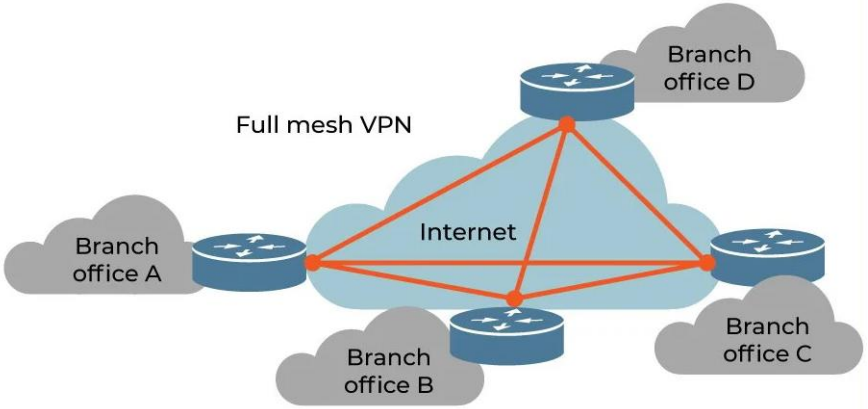
Encapsulating Protocols

- Tunneling could be achieved with or even without support of encapsulating protocol
- **Tunneling WITH encapsulating protocol**
  - Encapsulating protocol wraps around original data and then is inserted into new packet in carrier protocol
  - Authentication support, multiple tunnels between same devices, encryption
  - More features means potentially more overhead
  - E.g. GRE, L2TP, PPTP
- **Tunneling WITHOUT encapsulating protocol**
  - Original packet is directly inserted into the new one
  - Limited support of advanced tunneling features
  - Minimal overhead
  - E.g. IP-in-IP, IPv6-in-IPv4

Site-to-Site VPNs







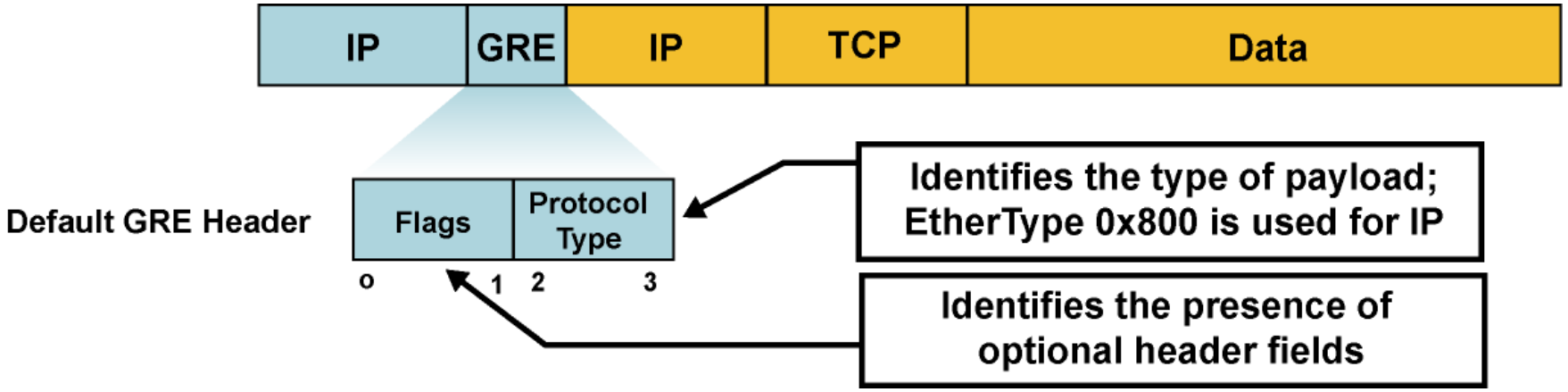
GRE

Generic Routing Encapsulation (GRE)

- **GRE** is encapsulating/tunneling protocol on L3
  - Supports multiple passenger protocols
  - Creates virtual point-to-point connection between pair of routers
  - Uses IP as carrier protocol
  - Allows transmission of multicast traffic (NBMA nature)
- GRE was originally invented by Cisco, but nowadays it's open standard specified in RFC2784

GRE Header (1)

- GRE is stateless without any signalization and traffic flow control
- GRE doesn't provide any security (no authentication, no encryption, no message integrity, no trustworthiness)
- Overhead of GRE tunnel is 24B (20B for new IPv4 header and 4B for GRE header)
  - implication for MTU and its mismatch



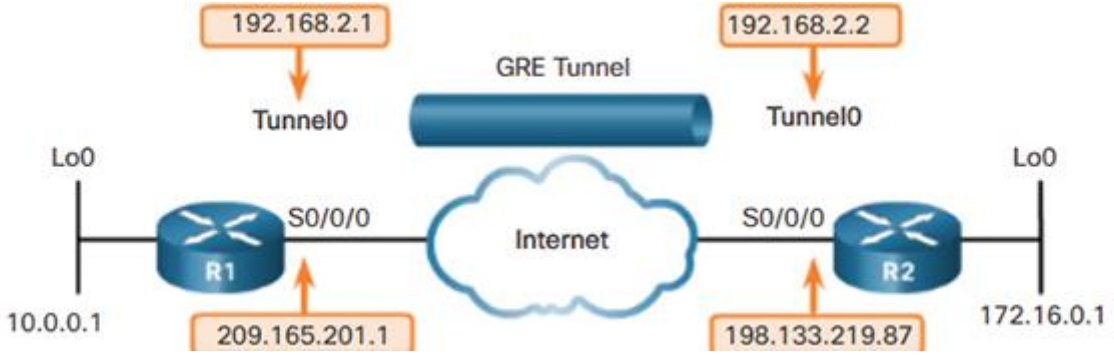
GRE Header (2)

- GRE **Flags** are stored in first 2B of header
  - Checksum Present (bit 0)
  - Key Present (bit 2)
  - Sequence Number Present (bit 3)
  - Version Number (bits 13–15)
    - GRE has version 0
    - PPTP has version 1

- **Protocol Type** specify type of passenger protocol, usually it has same value as in field EtherType L2 Ethernet frame

Tunnel Interface

- GRE tunnels are represented by virtual Tunnel interface
- Tunnel interface must have
  - own IP address (just like any other interface)
  - IP address of sender and receiver of (carrier protocol) packets
  - set proper tunneling mode
- Pair of Tunnel endpoint interfaces on opposite routers must met this criteria:
  - Tunnel endpoints own IP addresses must be in same network segment – *just like any other two directly interconnected interfaces*
  - IP addresses of sender and receiver must correspond on both endpoints – *IP address of receiver on one side must be IP address of sender on the opposite site and vice versa*

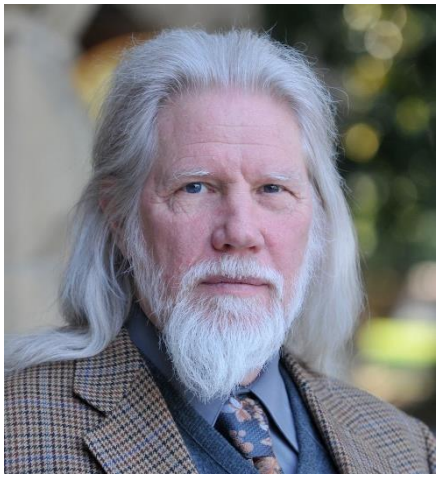


- **Site-to-Site VPNs**
  - GRE tunnels are frequently deployed in enterprise networks to connect remote offices by encapsulating traffic over the internet.
- **Dynamic Routing over Tunnels**
  - GRE supports dynamic routing protocols (e.g., OSPF, EIGRP) that rely on multicast traffic, enabling the creation of flexible and dynamic network topologies over tunnels.

DIFFIE-HELMANN

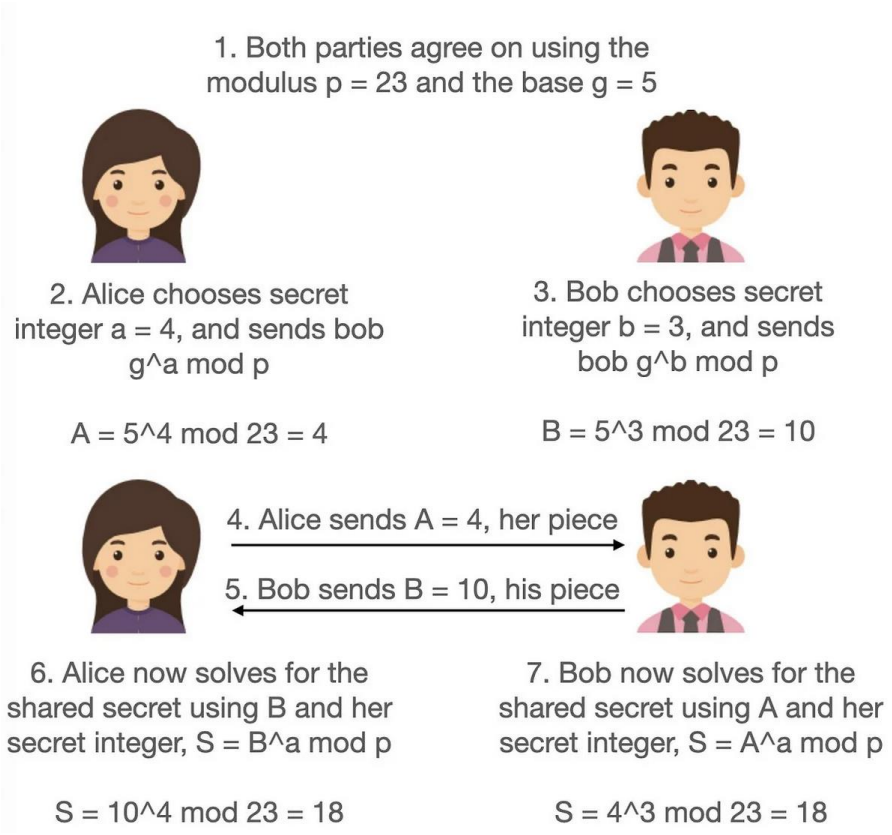
Motivation

- Imagine Alice and Bob wanting to communicate securely over an insecure channel.
  - They need a way to agree on a secret key without actually exchanging it.
- This is where the Diffie-Hellman key exchange comes in—a groundbreaking method developed by Whitfield Diffie and Martin Hellman in 1976.
- Diffie-Hellman allows two parties to establish a shared secret key without directly exchanging the key itself.

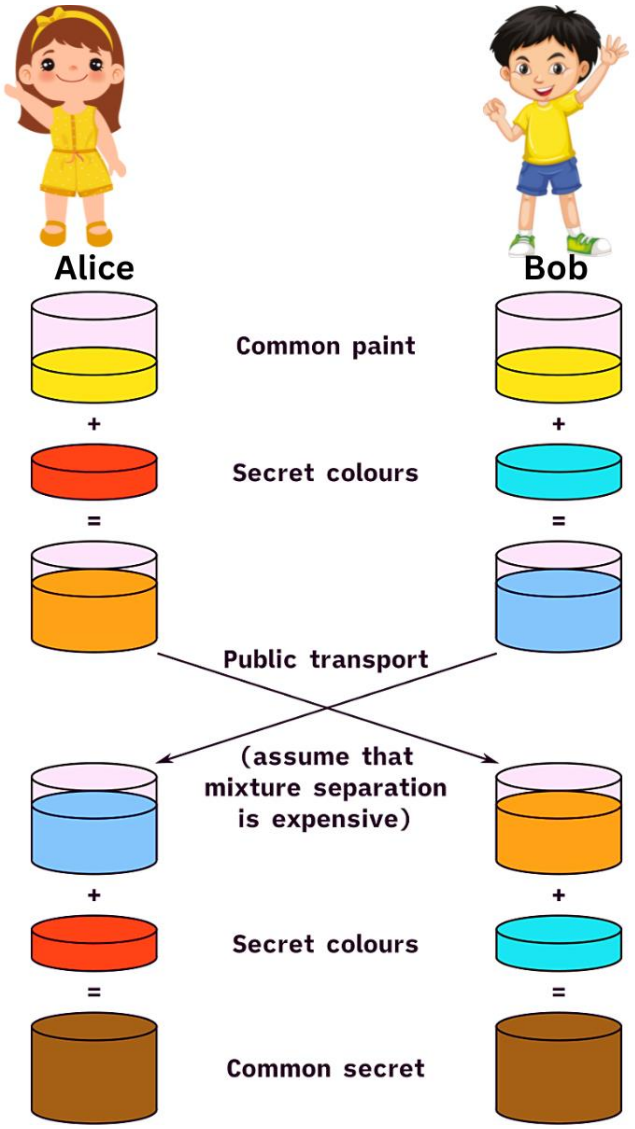


Diffie-Hellman Algorithm

1) Public Parameters (agreed publicly)
   - Large prime number p
   - Base (generator) g
2) Private/Public Key Generation
   - Alice chooses a private number a, computes public key: $A = g^a \bmod p$.
   - Bob chooses a private number b, computes public key $B = g^b \bmod p$.
3) Exchange Public Keys
   - Alice sends public key A to Bob.
   - Bob sends public key B to Alice.
4) Shared Secret Computation
   - Alice computes secret: $S = B^a \bmod p$
   - Bob computes secret: $S = A^b \bmod p$
5) Result
   - Both Alice and Bob end up with the same shared secret S, even though no private data was exchanged directly.



Color Analogy



Benefits

- The Diffie-Hellman key exchange provides strong security because an attacker would need to solve the discrete logarithm problem to deduce the shared secret key from the public values.
  - This problem is computationally infeasible with large numbers, ensuring the security of the key exchange.

- Enables secure communication over insecure channels.
- The actual secret key is never exchanged, reducing the risk of interception.
- Used in various cryptographic protocols, including SSL/TLS and IPsec.

DH Groups

- **DH groups** are predefined sets of parameters used in the Diffie-Hellman key exchange algorithm.

- Each DH group specifies a particular combination of:
  - Prime number (modulus) length (in bits)
  - Generator (base number) for exponentiation
  - Type of elliptic curve (for elliptic-curve groups)

- These parameters determine the strength and security of the resulting keys and how resistant the key exchange is against cryptographic attacks.

DH Groups

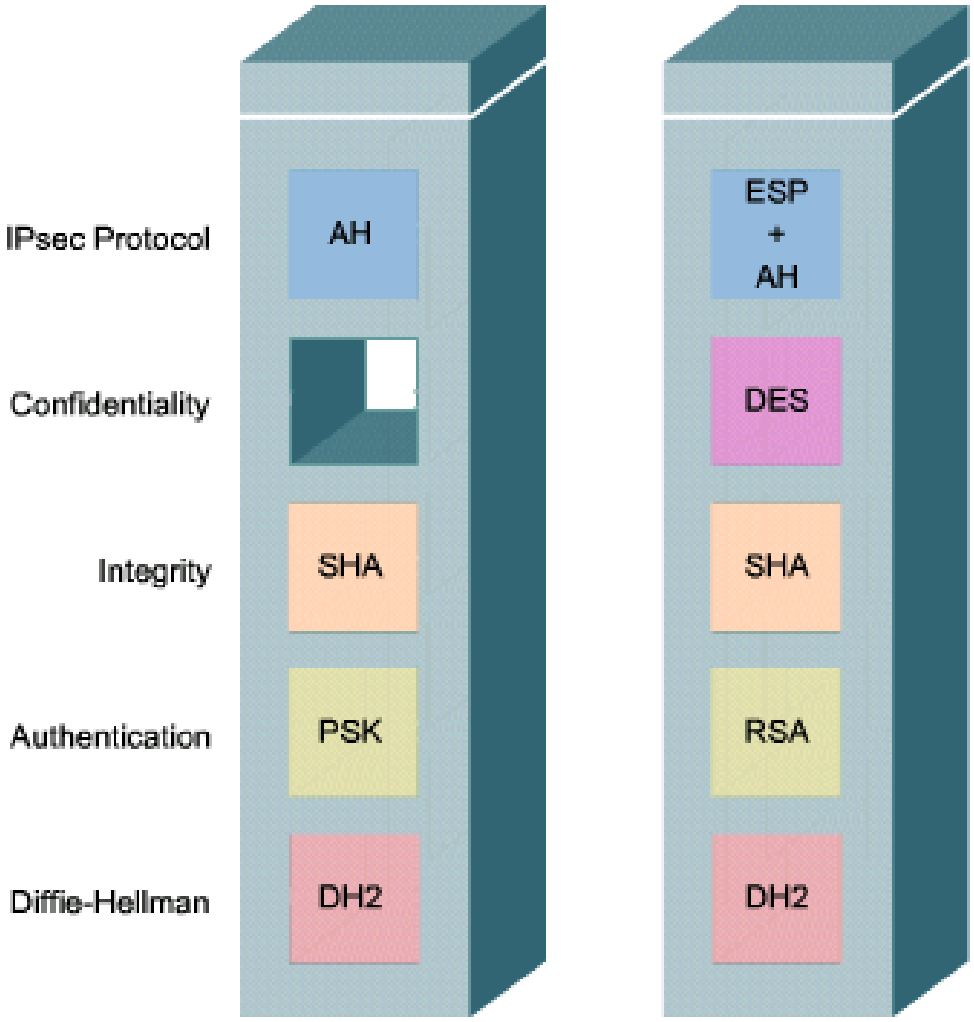
DH Group	Type	Key Length	Description & Security Level	Recommendation
Group 1	MODP	768-bit	Very insecure; easily compromised due to short key length.	Not recommended
Group 2	MODP	1024-bit	No longer secure due to computational advances.	Not recommended
Group 5	MODP	1536-bit	Weaker than current standards; avoid new deployments.	Not recommended
Group 14	MODP	2048-bit	Secure and widely used; safe for current implementations.	Recommended (minimum standard)
Group 15	MODP	3072-bit	Strong security, suitable for sensitive applications.	Recommended
Group 16	MODP	4096-bit	Very strong security, suitable for high-security scenarios.	Recommended for high-security use
Group 18	MODP	8192-bit	Highest security, computationally demanding.	Highly secure, specialized use
Group 19	ECC	256-bit	Modern elliptic-curve; highly secure and efficient.	Highly recommended
Group 20	ECC	384-bit	Strong security, suitable for higher-security needs.	Recommended
Group 21	ECC	521-bit	Very high security ECC, high computational cost.	Highly secure, specialized use

IP Security

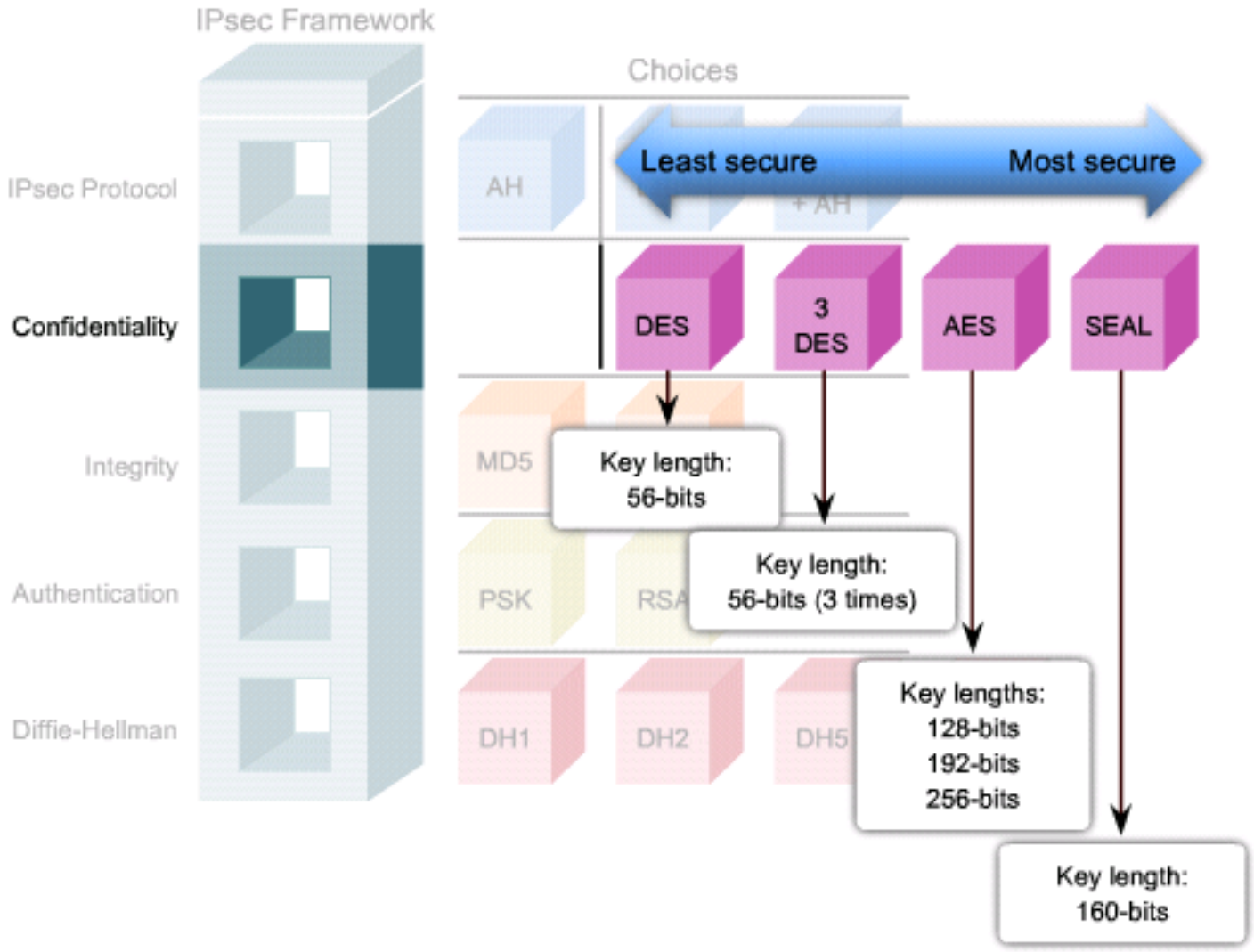
IPSEC

- Framework otevřených standardů v rámci IETF zajišťující vytvoření bezpečného tunelu
- RFC 2401
- IPsec není svázaný s jedním konkrétním šifrovacím algoritmem; naopak je modulární
- Používá dva páteřní protokoly
  - AH
  - ESP

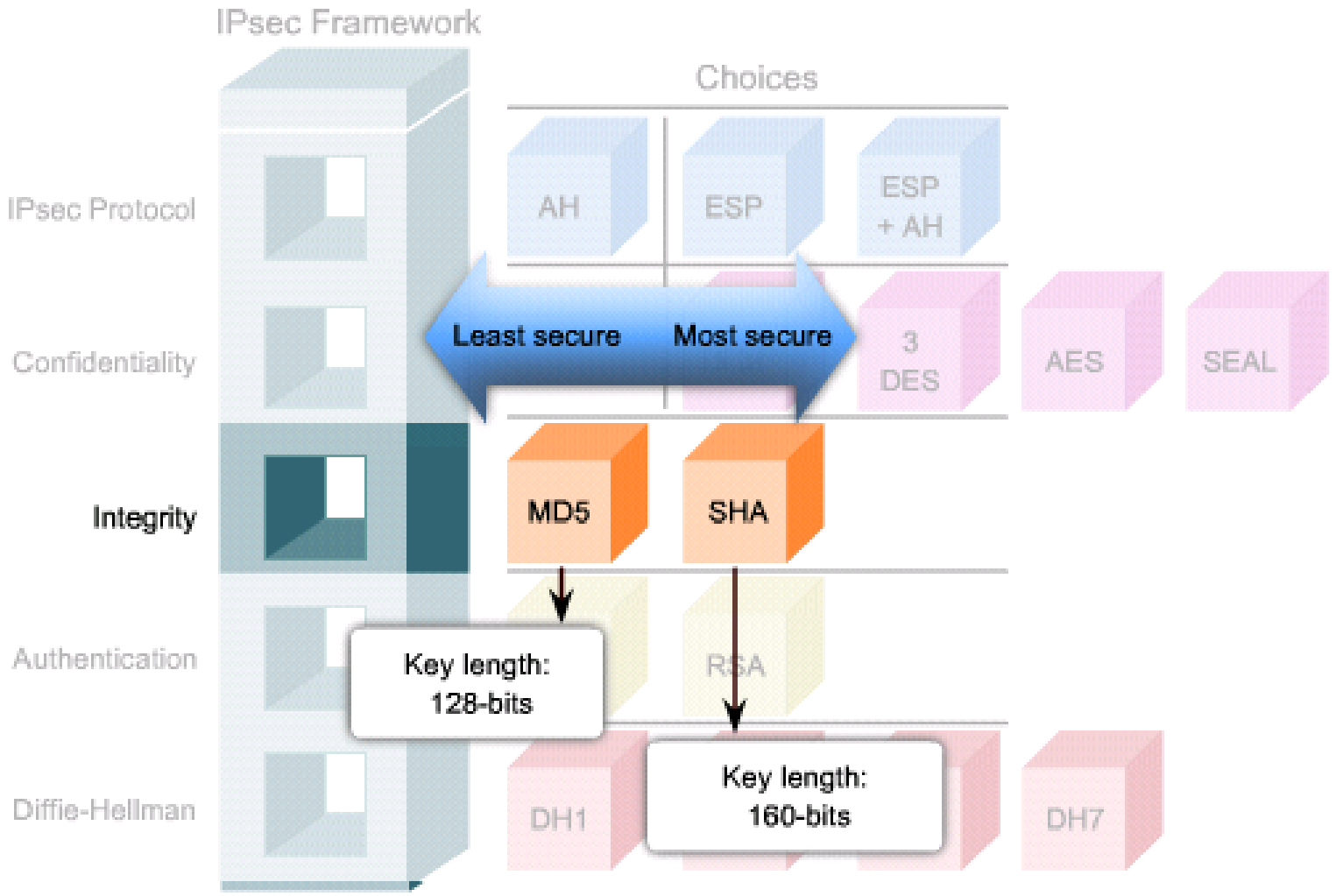
Modular Framework



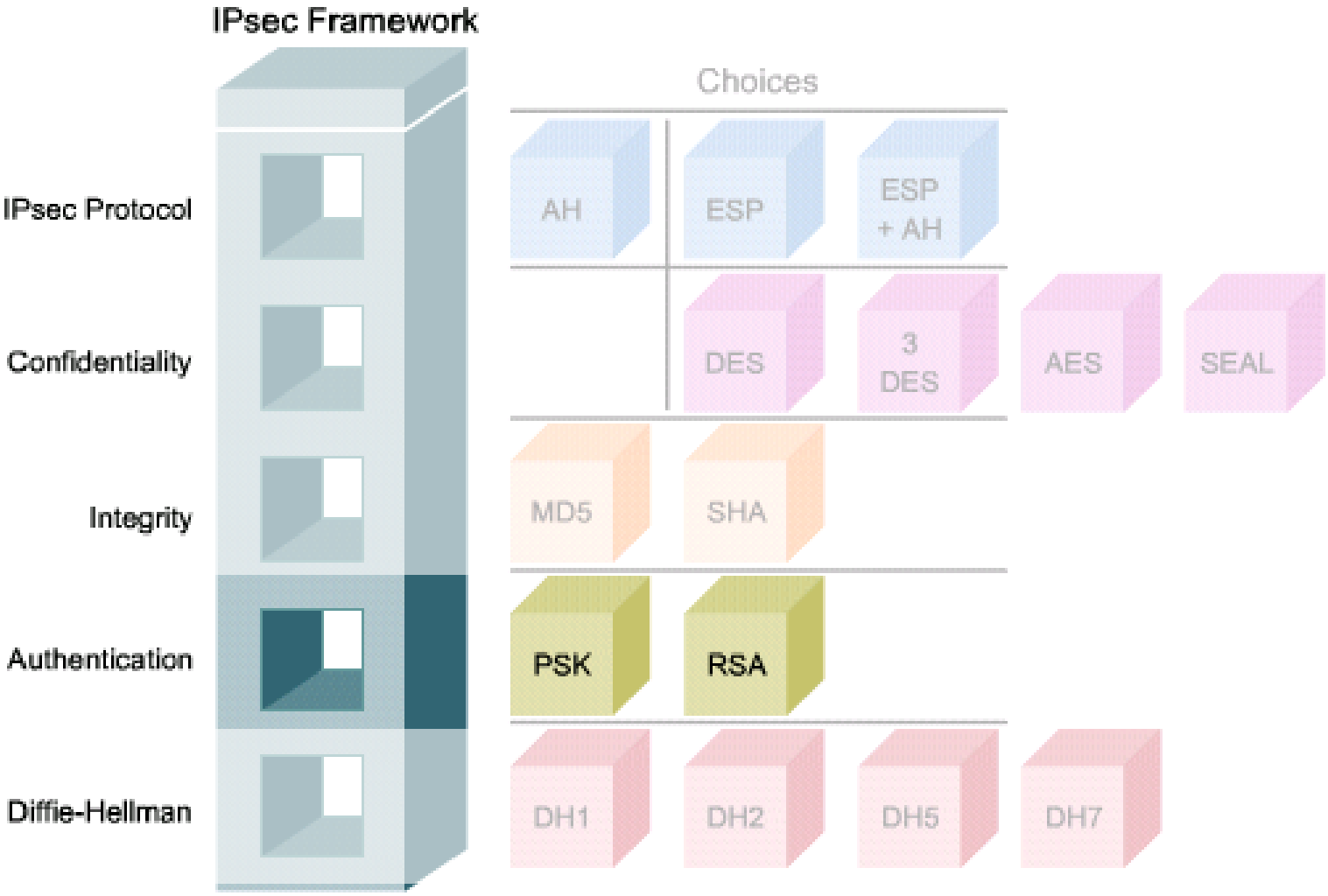
Confidentiality



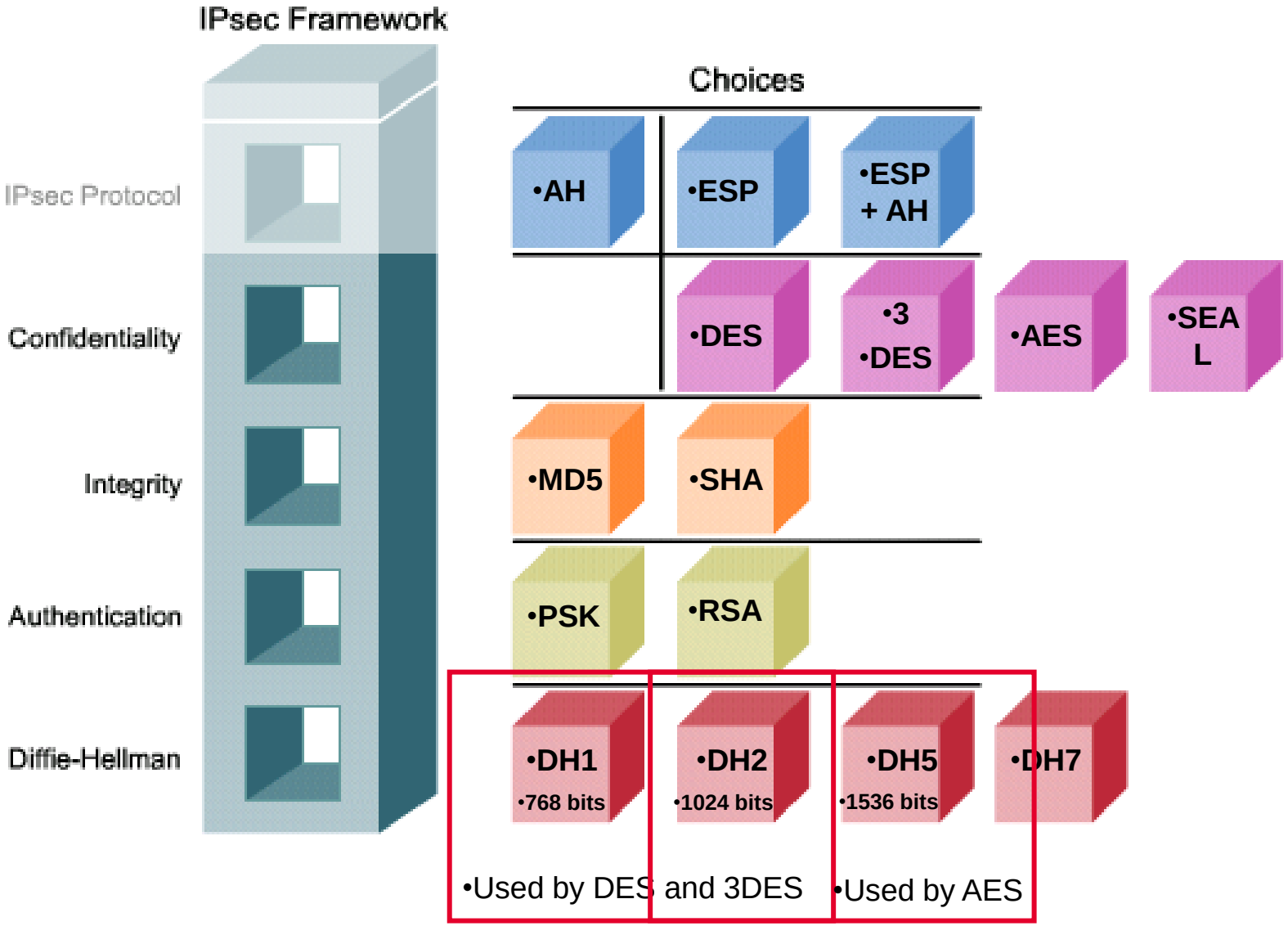
Integrity


IPsec Framework
Choices
IPsec Protocol
AH
ESP
ESP + AH
Confidentiality
Least secure
Most secure
3 DES
AES
SEAL
Integrity
MD5
SHA
Authentication
Key length: 128-bits
RSA
Diffie-Hellman
DH1
Key length: 160-bits
DH7

Authenticity

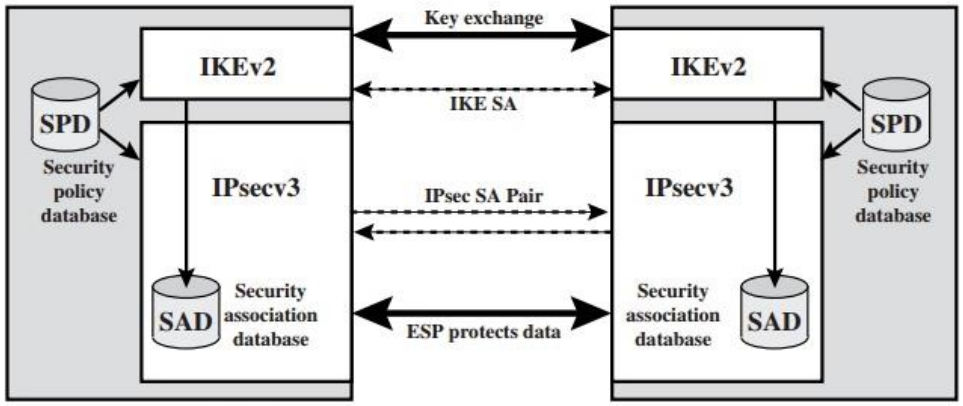


Secure Negotiation of Keys



- Works at the network layer (L3) to secure IP packets

- Components
  - Security Protocol
    - **Encapsulating Security Payload (ESP)**
    - Authentication Header (AH)
  - **Security Associations**
    - SA DB
    - Security Policy DB
  - **Internet Key Exchange Protocol**

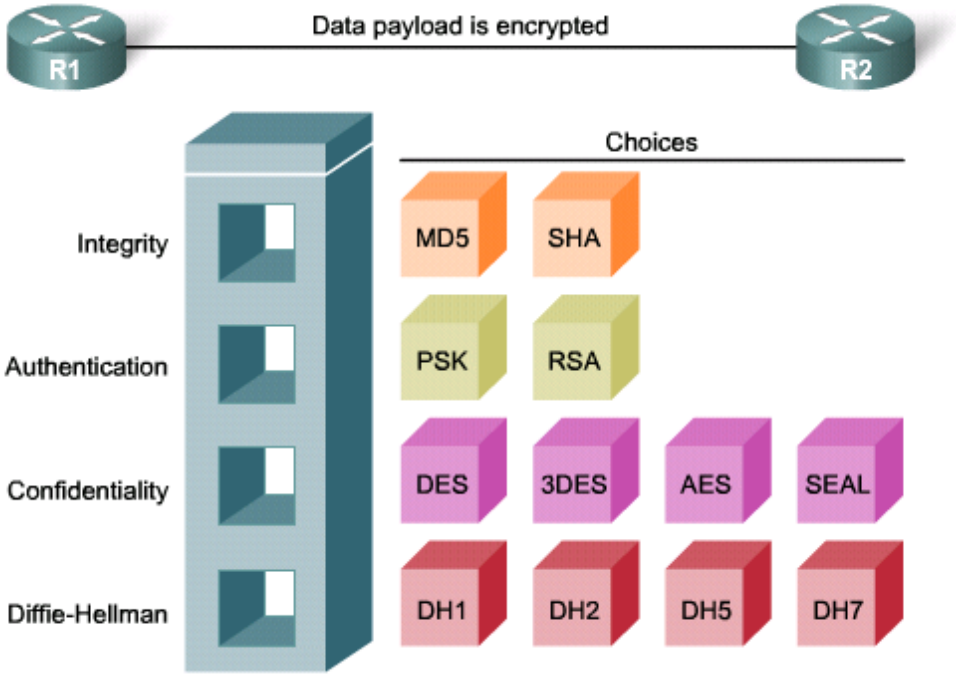


- **AH = Authentication Header**
- Poskytuje autenticitu a ochranu proti znovupoužití
- Neposkytuje důvěrnost
  - Data putují sítí v nezašifrované podobě

- Protokolové číslo 51
- RFC 2402

- **ESP = Encapsulating Security Payload**
- Umí to samé co AH + poskytuje důvěrnost
  - Nejdříve zašifruje obsah
  - Poté k cipher-textu přidá HMAC

- Protokolové číslo 50
- RFC 2406

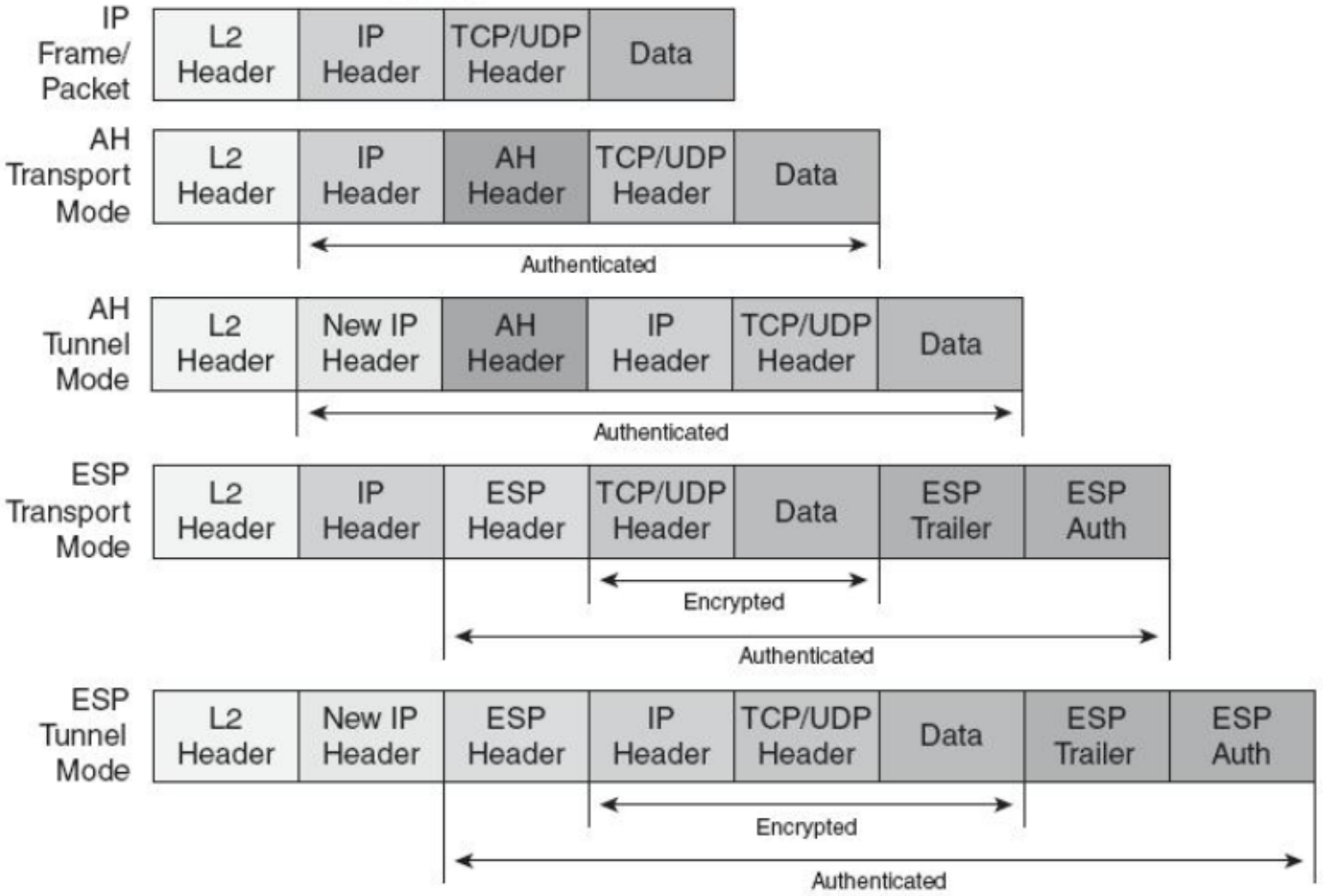


Transport vs. Tunel Mode (1)

- **Transport Mode**
  - Only the payload of the IP packet is encrypted and/or authenticated.
  - Commonly used for end-to-end communications between hosts.

- **Tunnel Mode**
  - The entire IP packet (header + payload) is encapsulated and protected
  - Commonly used in VPNs and gateway-to-gateway scenarios.

- ESP and AH may protect data in two ways



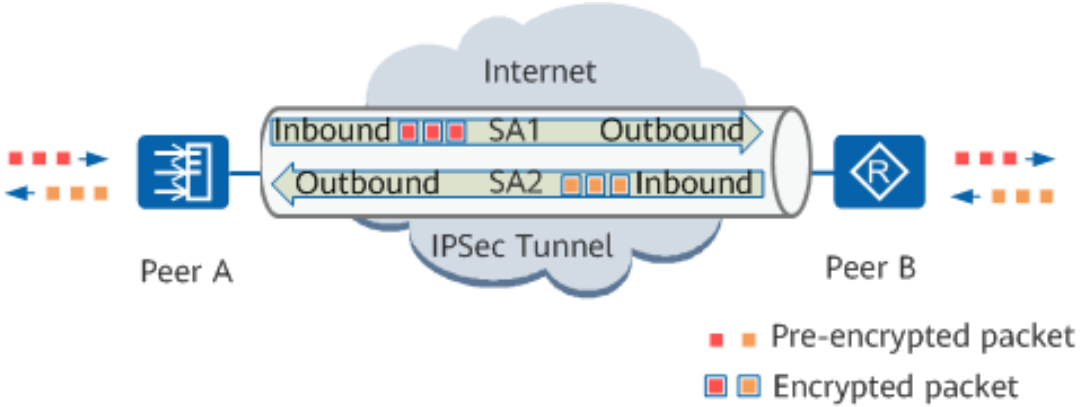
Security Policy DB

- **The Security Policy Database (SPD)** is a set of rules that defines the treatment of IP packets traversing an IPsec-enabled system. It specifies which packets require protection (via IPsec) and which should be processed normally.
- While the SPD defines the rules and policies, the SAD holds the actual operational SAs that enforce these policies.
  - The SPD tells the IPsec engine what to do with a packet, and if protection is required, the SAD provides the detailed instructions (keys, algorithms, etc.) to perform that protection.

Direction	IP Address	Protocol/Port	Action	Line of the policy
Out	fe80::11:7d00:21:1498	UDP/1000	Secure with SA-out	23
Out	fe80::11:7d00:21:1498	UDP/1001	Secure with SA-out	23
Out	fe80::11:7d00:21:1498	UDP/999	Drop	26
Out	fe80::11:7d00:21:1498	TCP/1000	Drop	26
Out	fe80::11:7d00:21:1498	UDP/3000	Secure with SA-out	23
Out	fe80::11:7d00:21:1498	UDP/3001	Bypass	29
Out	fe80::11:7d00:21:1500	UDP/3001	Bypass	29
Out	fe80::11:7d00:21:1500	UDP/3000	Bypass	29
Out	fe80::11:7d00:21:1500	UDP/2999	Drop	26
Out	fe80::11:7d00:21:1500	UDP/1	Drop	26
Out	1234::11:7d00:21:1500	UDP/1	Drop	26
Out	1234::11:7d00:21:1500	UDP/2999	Drop	26
Out	1234::11:7d00:21:1500	UDP/3000	Drop	No matching rule
In	fe80::11:7d00:21:1498	UDP/1001	Secure with SA-in	24
In	fe80::11:7d00:21:1498	UDP/999	Drop	27
In	fe80::11:7d00:21:1498	UDP/3001	Bypass	30

Security Associations

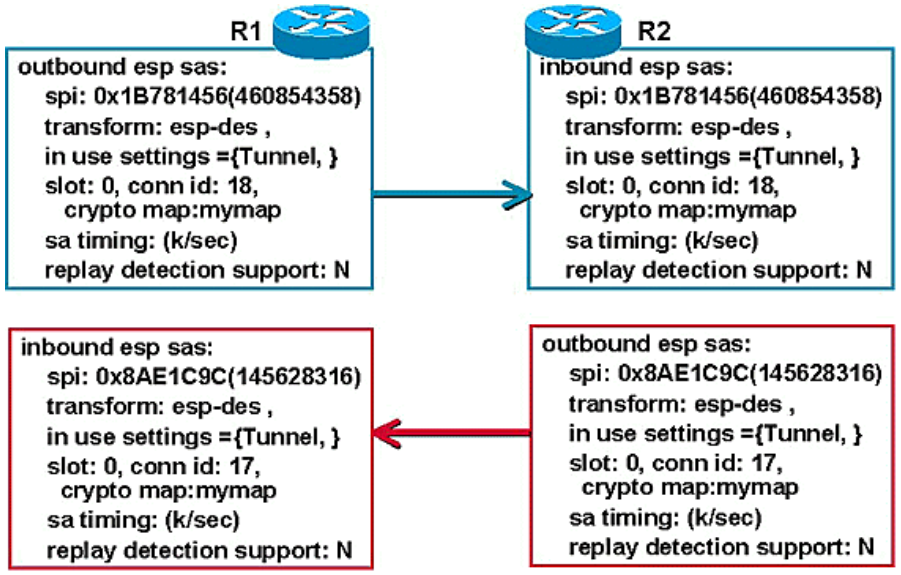
- **SA** = A unidirectional relationship that defines how two entities use IPsec protocols to communicate securely.
  - If bidirectional secure communication is required, separate SAs are established for each direction.



- SAs are being negotiated

Security Associations: Parameters

- SPI: Unique identifier for the SA.
- Destination and Source IP Addresses: To which the SA applies.
- Encryption and Authentication Algorithms: Used to secure the traffic.
- Cryptographic Keys: Actual keys used in encryption and integrity checks.
- Sequence Numbers: Used for anti-replay protection.
- Lifetime Information: Expiration and rekeying details.
- Mode: Indicates whether the SA operates in transport or tunnel mode.



SA Database

- The **Security Association Database (SAD)** stores active SAs and all related parameters needed to process IPsec-protected traffic.
- When a network device receives an IP packet, it examines the packet's header and checks the SAD for a matching SA.
  - If found, the device applies the appropriate security measures based on the parameters stored in the SAD

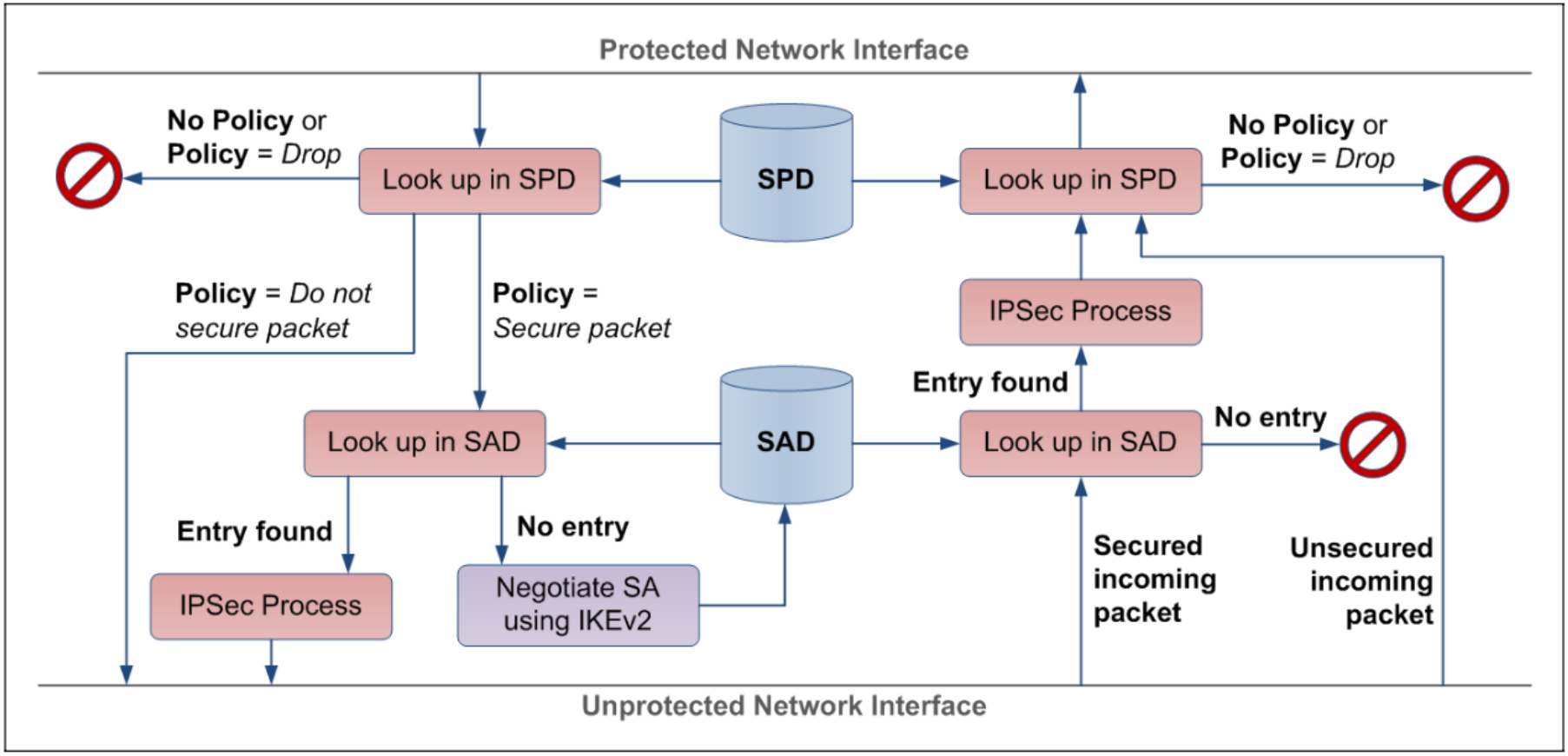
Index	SN	OF	ARW	AH/ESP	LT	Mode	MTU
< SPI, DA, P >							
< SPI, DA, P >							
< SPI, DA, P >							
< SPI, DA, P >							

Security Association Database

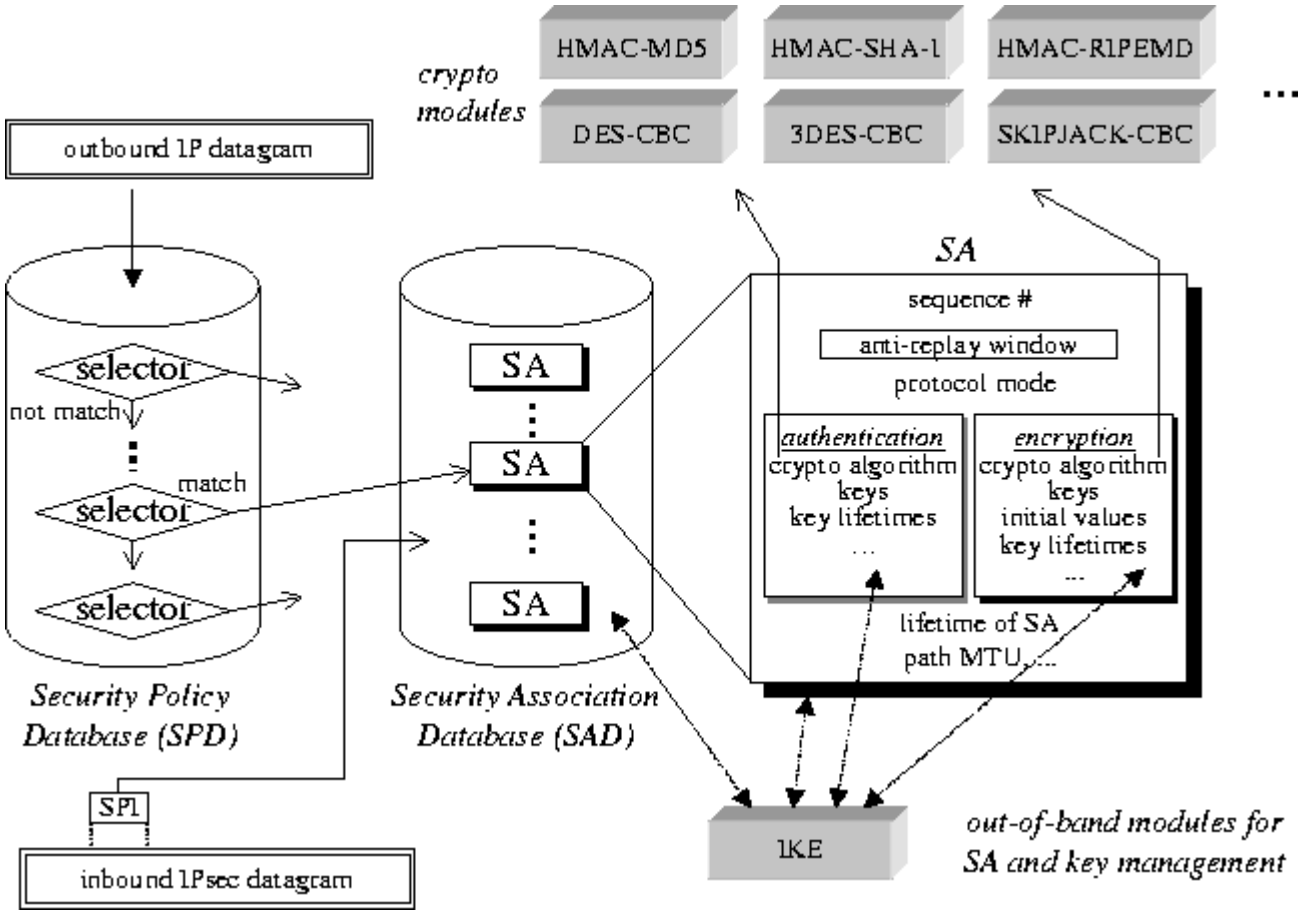
Legend:

SPI: Security Parameter Index	SN: Sequence Number
DA: Destination Address	OF: Overflow Flag
AH/ESP: Information for either one	ARW: Anti-Replay Window
P: Protocol	LT: Lifetime
Mode: IPSec Mode Flag	MTU: Path MTU

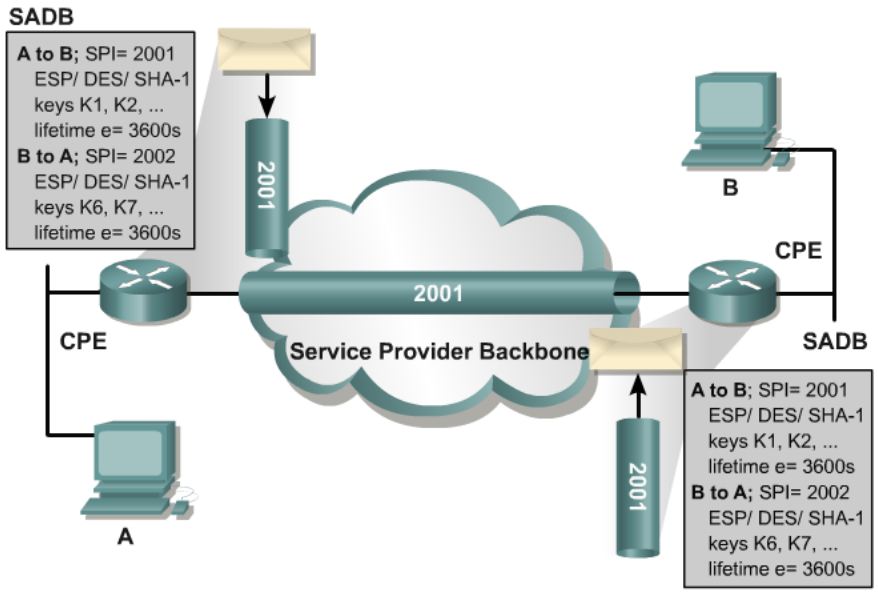
SPD and SAD


Protected Network Interface
No Policy or
Policy = Drop
Look up in SPD
SPD
Look up in SPD
No Policy or
Policy = Drop
Policy = Do not
secure packet
Policy =
Secure packet
IPSec Process
Entry found
Look up in SAD
SAD
Look up in SAD
No entry
Entry found
No entry
Secured
incoming
packet
Unsecured
incoming
packet
IPSec Process
Negotiate SA
using IKEv2
Unprotected Network Interface

SPD and SAD and SA Relations



- **ISAKMP** (RFC2408) is a protocol framework that defines procedures for establishing, negotiating, and managing SAs and cryptographic keys.
- It doesn't specify the key exchange mechanism itself but provides the structure for carrying out these tasks.
  - By separating these negotiation mechanics from the actual key exchange math, ISAKMP allows different key exchange protocols to plug into the framework
- Objectives
  - Security Association Management
  - Key Management
  - Flexibility: It supports multiple key exchange protocols (most notably IKE, KINE) and is designed to be extensible for various security protocols



ISAKMP: Messages and Payloads

- ISAKMP runs over UDP on port 500
  - or port 4500, when NAT traversal is used
- ISAKMP messages include headers that define the message type, version, and cookie values (random numbers used to identify sessions and protect against certain attacks).
- Following the header, a chain of **payloads** conveys the negotiation details.

- **Vendor ID Payload**: Allows peers to exchange vendor-specific information or extensions.
- **Security Association Payload**
- **Delete Payload**: Deletes or terminates SA (or multiple SAs)
- **Key Exchange Payload**: contains data necessary for key exchange (e.g., DH).
- **Nonce Payload**: random data to ensure the freshness
- **Identification Payload**
- **Certificate and Certificate Request Payloads**: used for mutual authentication
- **Hash Payload**
- **Notification Payload**: Sends informational notifications, such as errors, status messages, or alerts

ISAKMP: Features and Benefits

- Robust Key Exchange
  - ISAKMP, via protocols like IKE, enables secure Diffie-Hellman exchanges that ensure both parties derive the same cryptographic keys without exposing them to eavesdroppers.
- Mutual Authentication
  - By exchanging identification and certificate payloads, ISAKMP helps ensure that both endpoints are who they claim to be, protecting against impersonation attacks.
- Flexibility and Extensibility
  - The payload-based architecture allows ISAKMP to support a variety of cryptographic algorithms and to be extended with additional payload types as new security requirements emerge.
- Replay Protection
  - The use of cookies and nonces in ISAKMP messages helps protect against replay attacks, ensuring that each negotiation session is unique and fresh.

Internet Key Exchange

- **IKE** is the concrete implementation that uses the ISAKMP framework (along with other components) to perform the real work of key exchange and authentication.
  - ISAKMP provides the envelope and framework for these negotiations, while IKE is the specific protocol conducting the negotiation inside that framework
- IKE facilitates automatic, secure key exchange between two IPsec peers
  - Instead of manually configuring matching keys and parameters on each device, IKE negotiates these dynamically.
- IKE authenticates the peers (using credentials like pre-shared keys or certificates)
- IKE performs a Diffie–Hellman exchange to establish a shared secret from which encryption keys are derived

IKE Versions

- v1 RFC2409
  a) uses a two-phase process (Phase 1 and Phase 2) with multiple messages (6 in Main Mode, 3 in Aggressive Mode), making it slower and more resource-intensive
  b) offers basic security features but lacks support for advanced authentication protocols like EAP and does not support separate keys for each direction
  c) does not support MOBIKE, which means it cannot maintain connections during IP address changes. NAT traversal is optional and may require additional configuration
  d) requires symmetric authentication methods, meaning both sides must use the same authentication method
  e) was originally defined as a hybrid of three elements: ISAKMP (the framework), the Oakley protocol (which provides specific Diffie–Hellman key exchange methods), and SKEME (another key exchange technique)

- v2 RFC7296
  a) combines these phases into a single exchange, typically requiring only 4 messages, which reduces latency and improves efficiency
  b) supports EAP for enhanced authentication, uses separate keys for each direction, and supports more modern and secure encryption algorithms like AES and ChaCha20
  c) includes built-in NAT traversal and supports MOBIKE, allowing seamless network transitions without dropping the VPN connection
  d) supports asymmetric authentication, allowing one side to use RSA while the other uses a pre-shared key
  e) is not backward compatible with IKEv1

IKEv1: IPsec Tunnel Creation

1. Host A sends interesting traffic to Host B.

2. Routers A and B negotiate an IKE Phase 1 session.

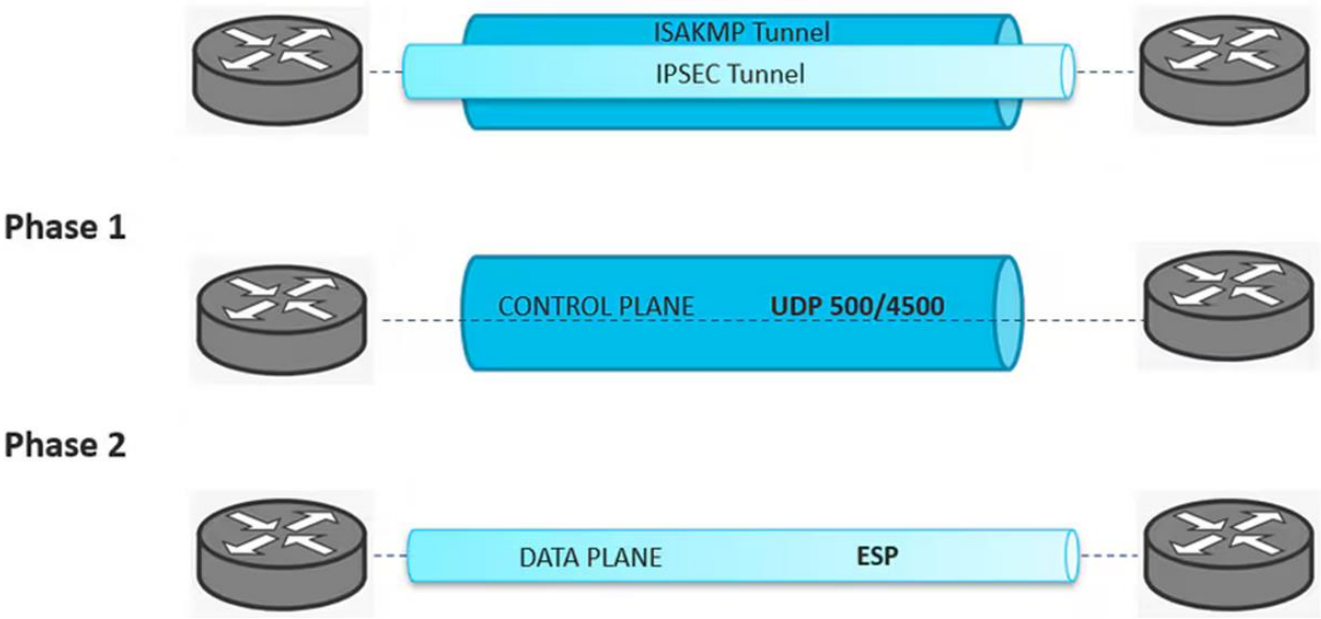
3. Routers A and B negotiate an IKE Phase 2 session.

4. Information is exchanged via the IPsec tunnel.

5. The IPsec tunnel is terminated.

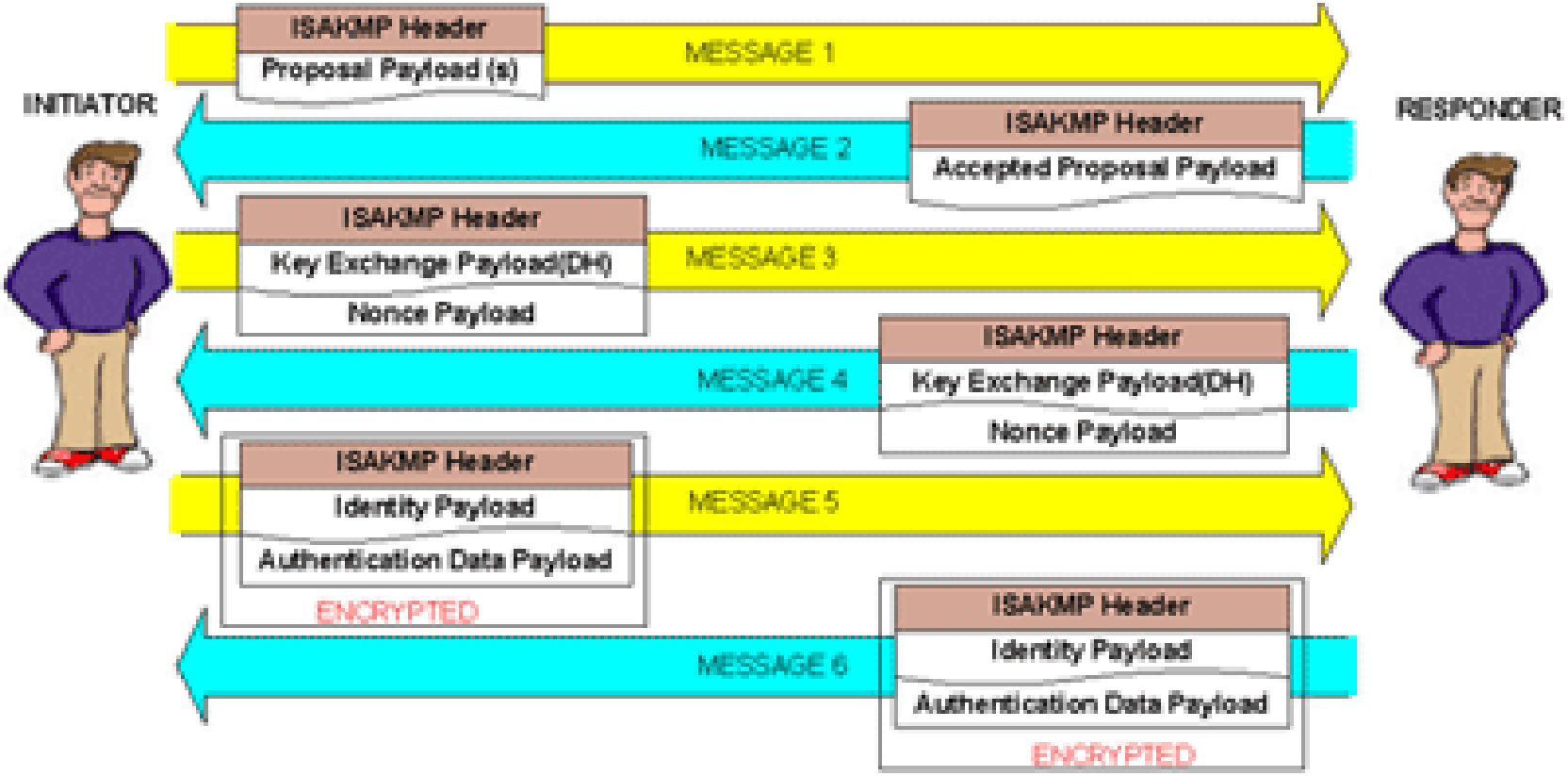
IKEv1: Phases

- The purpose of Phase 1 is to authenticate peers, agree on cryptographic parameters, and establish a secure, authenticated management channel.
- Phase 2 is used to establish the actual IPsec SA that protects user data packets. It relies on the secure channel established in Phase 1.



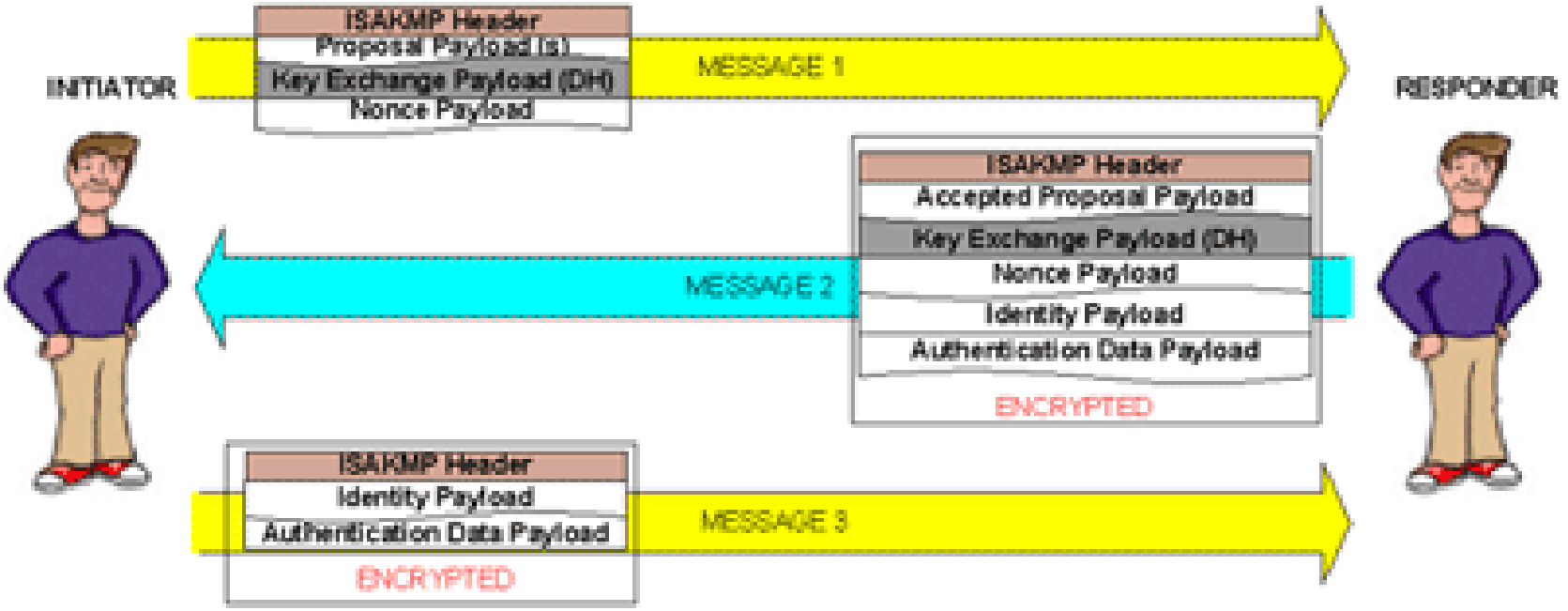
IKEv1: Phase 1 Main mode

- Provides identity protection by encrypting identity information.
- Exchange
  - Message 1 and 2 (Initiator→Responder): Negotiate security parameters (SA proposals).
  - Message 3–4: Diffie-Hellman key exchange (generate shared secret).
  - Message 5–6: Authentication (e.g., pre-shared key or certificate)

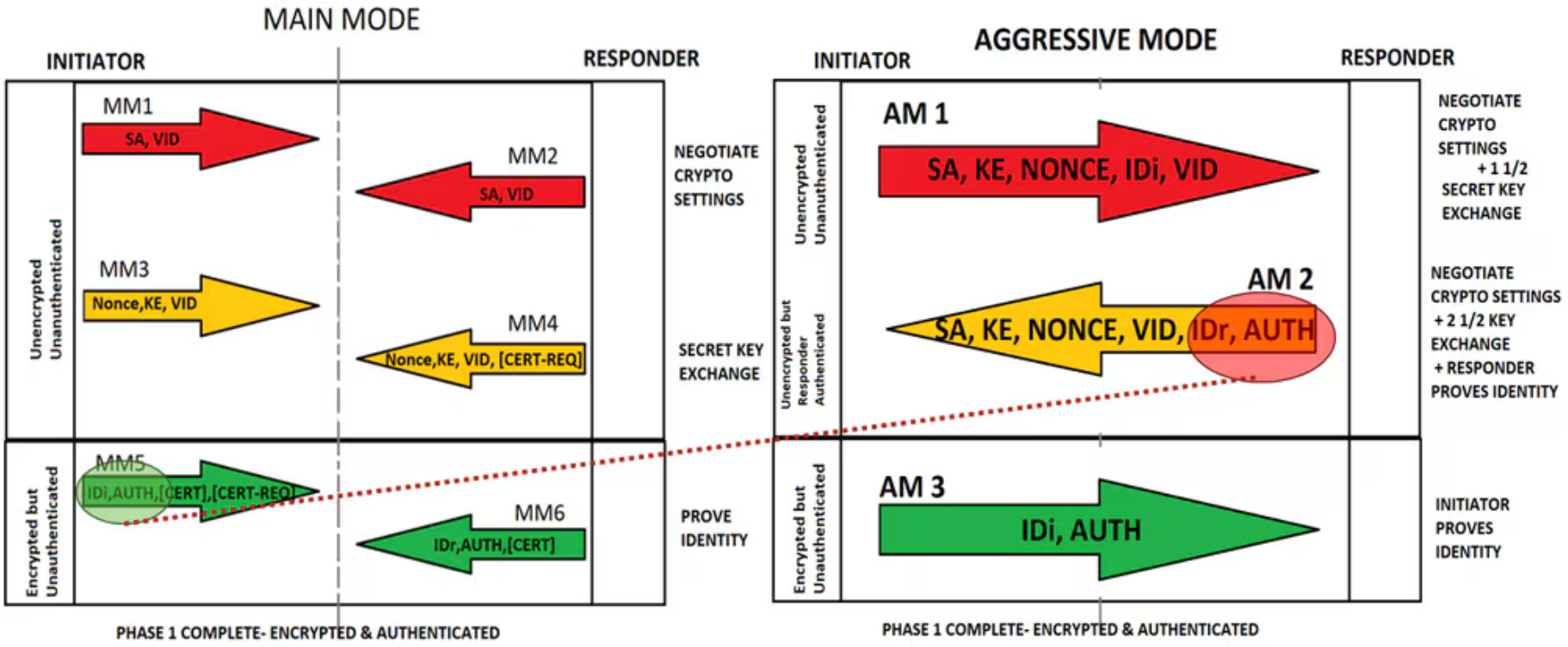


IKEv1: Phase 1 Aggressive mode

- Faster negotiation but exposes identities in clear text.
- Exchange
  - Message 1: Initiator sends proposals and identity openly
  - Message 2: Responder replies with selected parameters and its authentication data
  - Message 3: Initiator finalizes authentication

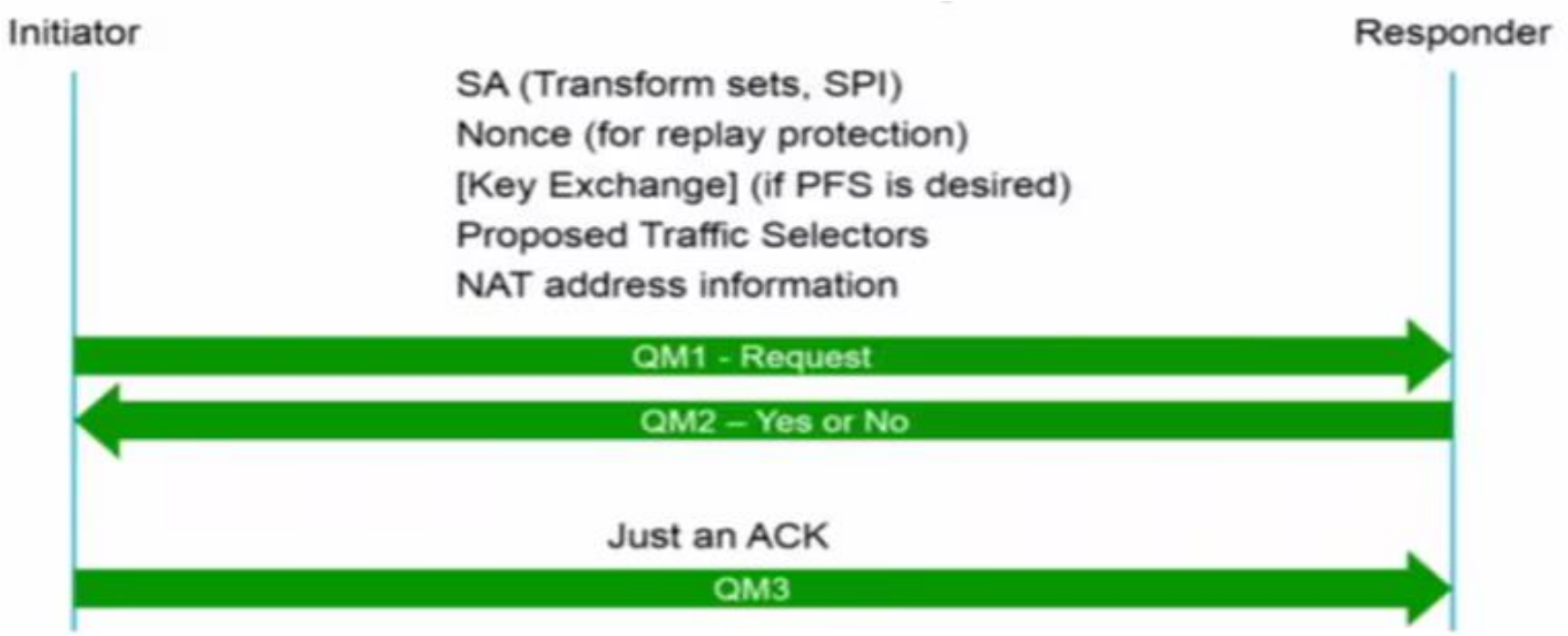


Main Mode vs Aggressive Mode



IKEv1: Phase 2

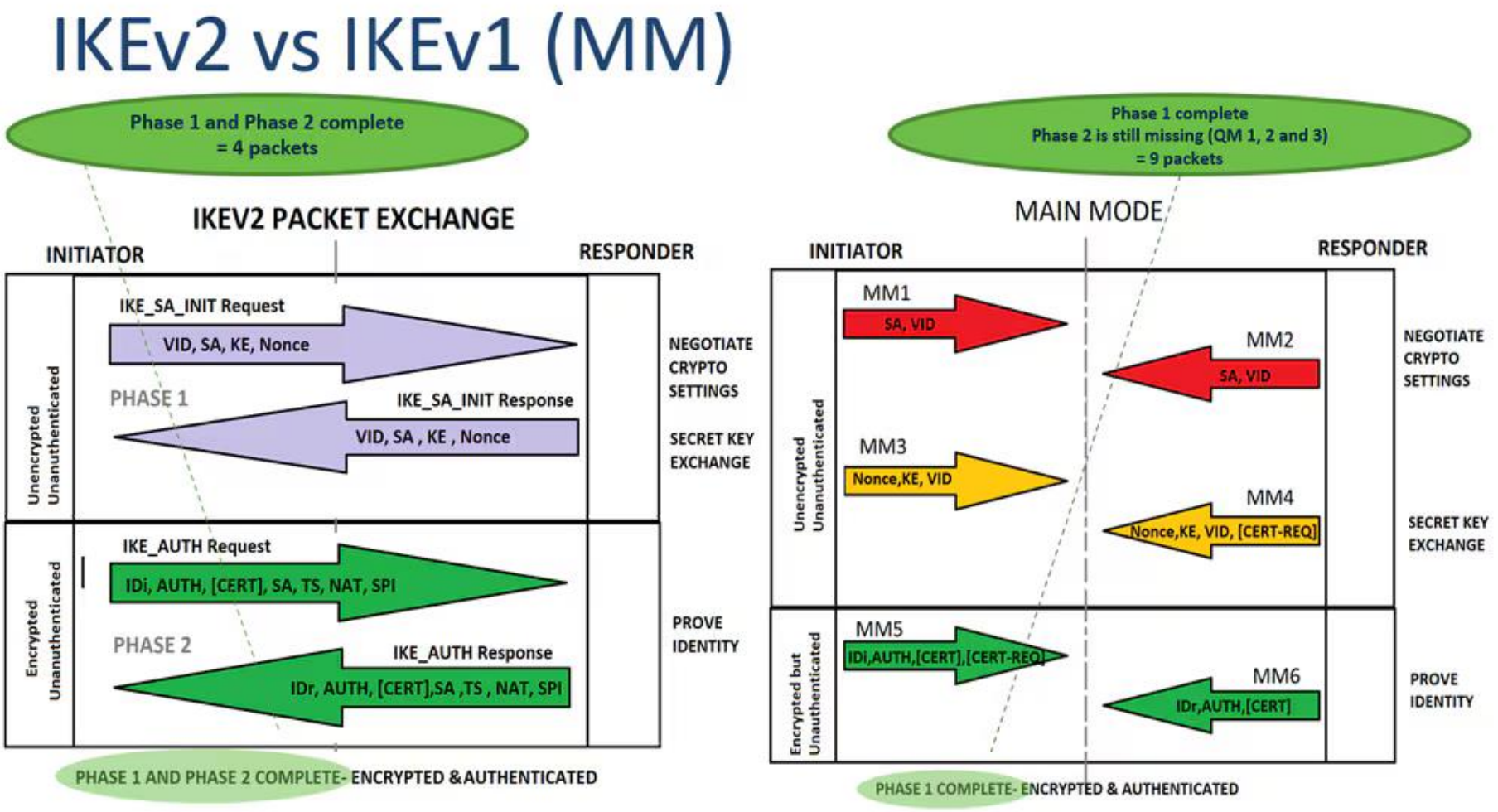
- Quick mode occurs after IKE has established the secure tunnel in phase 1.
- Quick Mode negotiates the shared IPSec policy, for the IPSec security algorithms and manages the key exchange for the IPSec SA establishment.
- The nonces are used to generate new shared secret key material and prevent replay attacks from bogus SAs generated.



IKEv2: Phase 1 + Phase 2 Message Exchange

- `IKE_SA_INIT` (Initial exchange)
  - establishes a secure channel between peers, negotiating cryptographic parameters and generating key material through a Diffie-Hellman (DH) key exchange
- `IKE_AUTH` (Authentication exchange)
  - authenticates both endpoints and completes the negotiation of the IPsec SAs that protect actual data traffic

- After the initial setup, additional exchanges may occur (`CREATE_CHILD_SA`, `INFORMATIONAL`) for maintaining, rekeying, or deleting SAs.

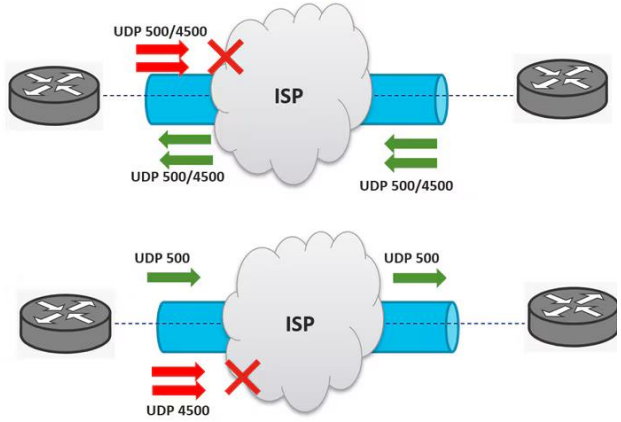

IKEv2 vs IKEv1 (MM)
Phase 1 and Phase 2 complete
= 4 packets
IKEV2 PACKET EXCHANGE
INITIATOR
RESPONDER
Unencrypted Unauthenticated
IKE_SA_INIT Request
VID, SA, KE, Nonce
PHASE 1
IKE_SA_INIT Response
VID, SA , KE , Nonce
NEGOTIATE CRYPTO SETTINGS
SECRET KEY EXCHANGE
Encrypted Unauthenticated
IKE_AUTH Request
IDi, AUTH, [CERT], SA, TS, NAT, SPI
PHASE 2
IKE_AUTH Response
IDr, AUTH, [CERT],SA ,TS , NAT, SPI
PROVE IDENTITY
PHASE 1 AND PHASE 2 COMPLETE- ENCRYPTED &AUTHENTICATED
Phase 1 complete
Phase 2 is still missing (QM 1, 2 and 3)
= 9 packets
MAIN MODE
INITIATOR
RESPONDER
Unencrypted Unauthenticated
MM1
SA, VID
MM2
SA, VID
NEGOTIATE CRYPTO SETTINGS
MM3
Nonce,KE, VID
MM4
Nonce,KE, VID, [CERT-REQ]
SECRET KEY EXCHANGE
Encrypted but Unauthenticated
MM5
IDi,AUTH,[CERT],[CERT-REQ]
MM6
IDr,AUTH,[CERT]
PROVE IDENTITY
PHASE 1 COMPLETE- ENCRYPTED & AUTHENTICATED

IPsec Usual Issues

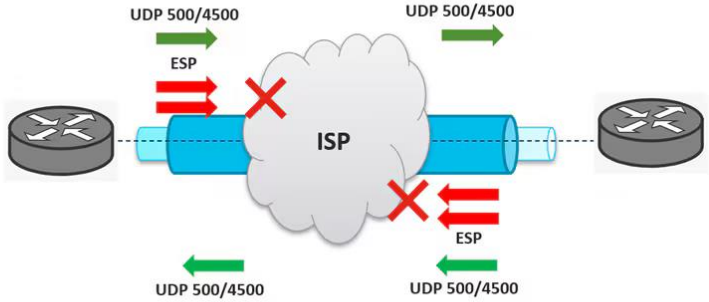
- It is a very common issue that the Internet Services Provider (ISP) blocks the UDP 500/4500 ports.
- For an IPsec tunnel establishment, two different ISPs can be engaged. One of them can block the ports, and the other allows them.

- ISP blocks the ESP traffic; however, it allows the UDP 500/4500 ports.
- For example, the UDP 500/4500 ports are allowed in bidirectional ways. Therefore, the tunnel is successfully established, but the ESP packets are blocked by the ISP or ISPs in both directions.

ISP Blocks UDP 500/4500



ISP Blocks ESP

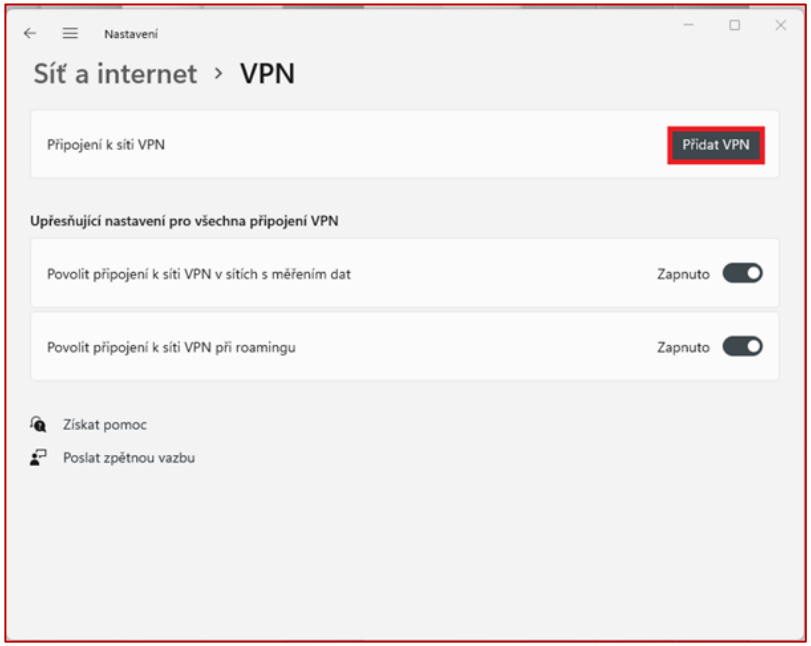


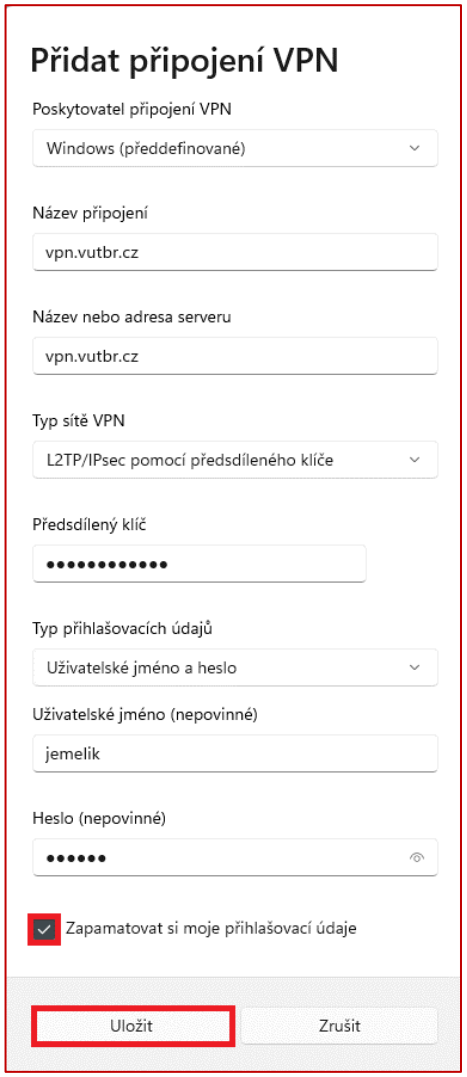
Implementations

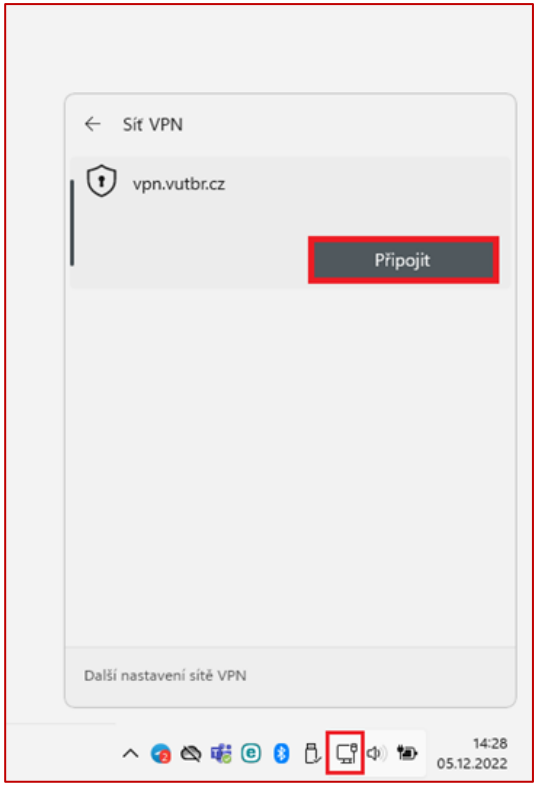
- **IKEv2/IPsec built-in clients**
  - Platform: Windows 10/11, macOS, iOS, Android (native VPN clients)
  - Notes: OS-integrated clients supporting IKEv2 natively, popular for remote access VPNs.
  - Example Use: Corporate remote access, personal VPN connections.
- **strongSwan**
  - Platform: Linux, Android, FreeBSD, macOS
  - Notes: Widely used, highly configurable, supports IKEv1 and IKEv2; preferred IPsec solution for modern Linux distributions.
  - Example Use: Enterprise VPNs, road-warrior setups, cloud-to-on-premises tunnels.
- **Libreswan**
  - Platform: Linux (RedHat, CentOS, Fedora)
  - Notes: Fork of Openswan, actively maintained, supports IKEv1/IKEv2; extensive documentation available.
  - Example Use: Site-to-site VPNs, Linux-based gateways.
- **Openswan**
  - Platform: Linux, embedded devices
  - Notes: Popular historically; original version now less maintained compared to strongSwan/Libreswan.
  - Example Use: Embedded Linux routers, older Linux deployments.
- **pfSense/OPNsense**
  - Platform: FreeBSD-based firewalls
  - Notes: Open-source firewalls with integrated IPsec implementation based on strongSwan (OPNsense) or Libreswan (pfSense).
  - Example Use: Small-to-medium businesses, branch office VPNs.

Exercise #1: VUT VPN

- https://www.vut.cz/intra/navody/vpn/windows11







ALTERNATIVES

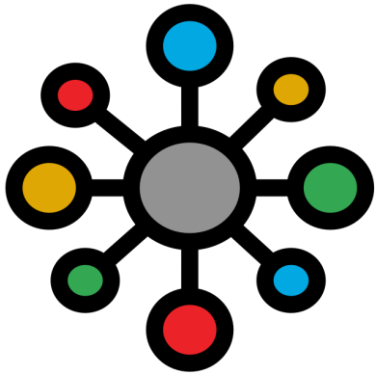
- L2TP

Q & A

Bibliography

- https://www.cisco.com/c/en/us/support/docs/security-vpn/ipsec-negotiation-ike-protocols/217432-understand-ipsec-ikev1-protocol.html#toc-hId--1518272186
- https://homecrew.dev/posts/dh.html
- https://fiona.onl/en/insecure_generators.html
- https://yurmagccie.wordpress.com/2019/01/02/ipsec-part-1-ikev1-main-mode-basics/

Vladimír Veselý veselyv@fit.vut.cz



- výzkumná skupina NES@FIT = specializace NNET
  - https://www.fit.vut.cz/research/groups/nes@fit/
  - https://nesfit.github.io
- repo s různými projekty
  - https://git.fit.vutbr.cz/NESFIT