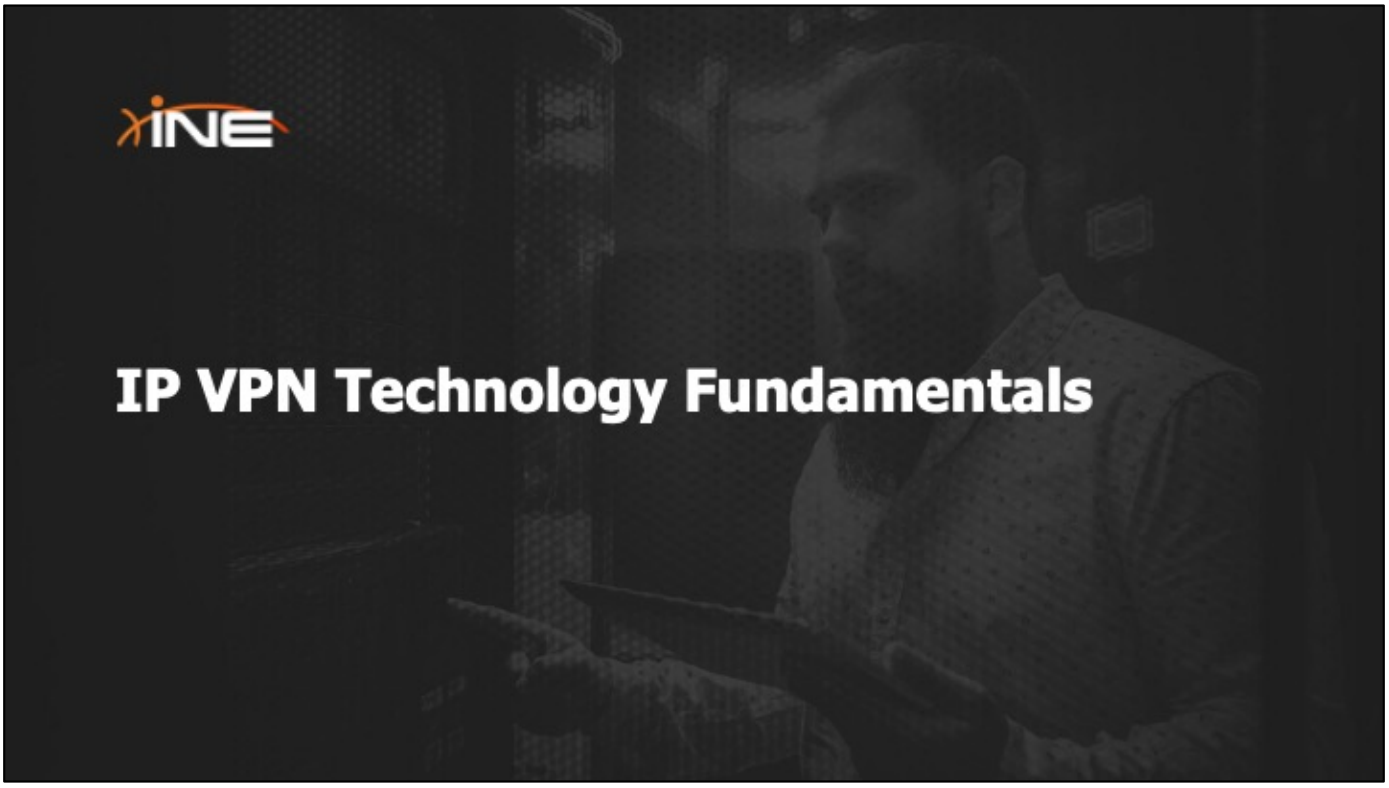


Course Objectives

- Explain the purpose and use-cases for VPNs
- Provide an overview of the primary technologies used to create IP Virtual Private Networks
- Explain the operations of some protocols used with IPsec
- Understand the differences beween IPsec and SSL/TLS VPNs
- Explore alternative VPN technologies such as DMVPN & GETVPN

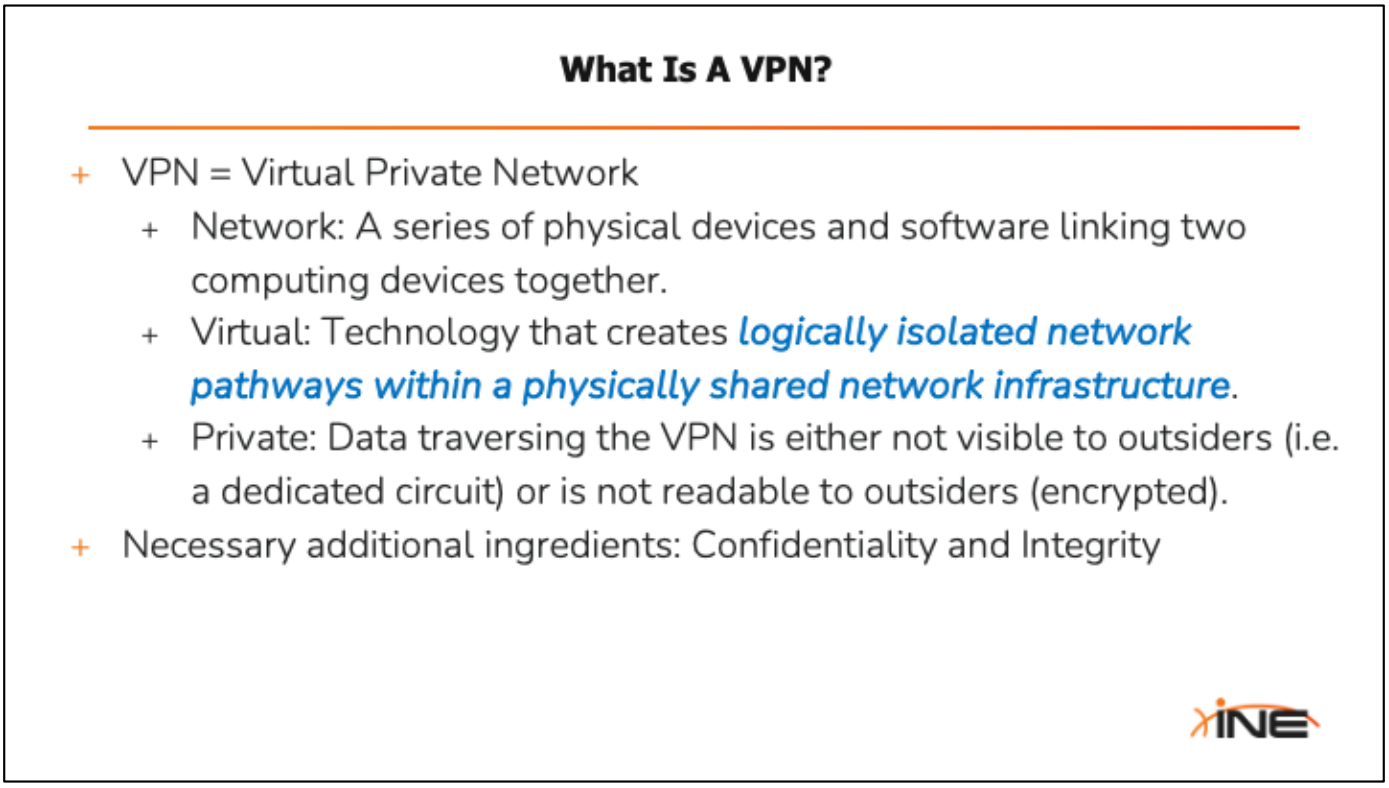
Course Prerequisites

- Understanding of IP packet networking including IP packet structure and encapsulation
- Familiarity with the concepts of routers and firewalls
- For Labs: Basic Cisco IOS CLI familiarity

Let's Get Started!

Topic Overview

+ What Is A VPN?
+ Why Use VPNs?
+ A Brief Introduction to Popular VPN Technologies



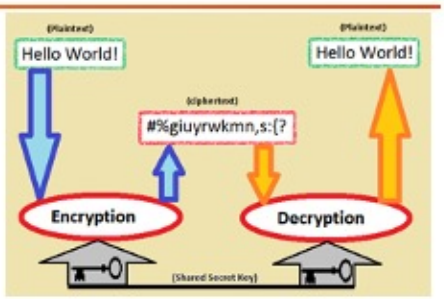
- Logical isolation means that data packets are handled in a way that keeps them separate from those of other users, despite traversing the same physical network. This is achieved using various technologies.
- MPLS VPNs are an example of a VPN technology that is only “private” in the sense that data entering the VPN cannot accidentally end up misdirected to an untrusted network. However, there is no confidentiality unless IPSec (or other tech) is also implemented to encrypt that data.

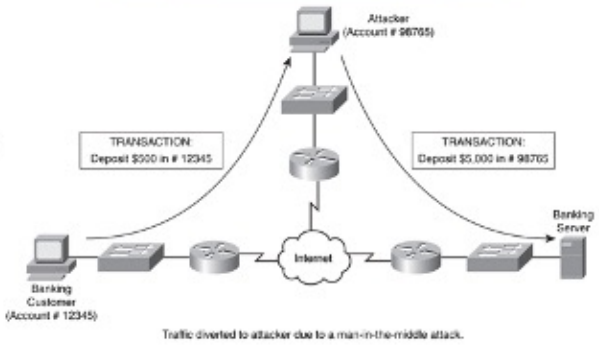
Why Use A VPN?

- Alternatives to VPN technology = Dedicated P2P Links
- Benefits to using VPNs across a public infrastructure;
  - Cost (Point-to-Point links are $$$$$)
  - Scalability (each Point-to-Point link requires a unique router interface. But hundreds of VPNs could terminate on a single interface)

Characteristics of IP VPNs

- Confidentiality
  - Publicly known algorithms
  - Secret Keys
  - Symmetric or Asymmetric
- Data Integrity
  - Provided by hash algorithms

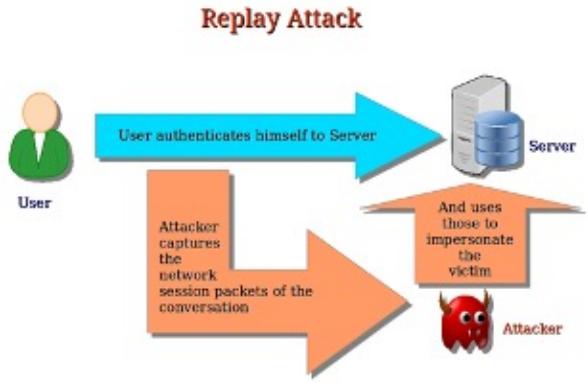




- Confidentiality – Of course, some companies also have proprietary algorithms that only work with their equipment.
- ----Symmetric: Same key used to encrypt and decrypt

Characteristics of IP VPNs

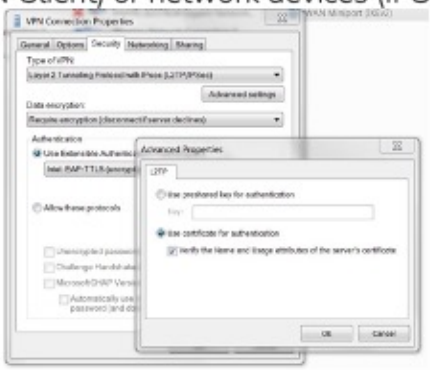
- Authentication
  - Pre-shared keys
  - Public/Private key pairs
  - User Authentication
- Antireplay Protection



- Antireplay prevents someone from replaying packets back to a VPN peer in an attempt to spoof you and gain illegitimate access to VPN.

Primary VPN Technologies: IPsec

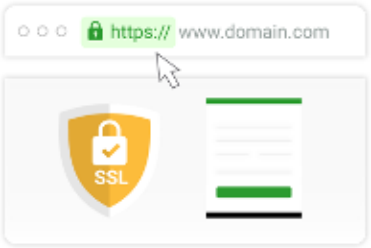
- IPSec (IP Security)
  - Generic term that encompasses several protocols to provide confidentiality, integrity and authentication.
  - Provides security for IP packets at Layer-3 of OSI Model
  - Typically implemented using special applications on clients (i.e. Cisco Anyconnect VPN Client) or network devices (IPSec features in Cisco IOS)



- Provides the network admin a selection of protocols from which to choose.
- -
- Once an IPSec tunnel is built, any application (that utilizes IP) can send packets across that tunnel.

Primary VPN Technologies: SSL

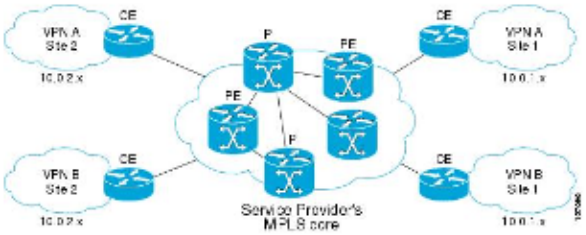
- SSL (Secure Sockets Layer)
  - Implemented at the Transport Layer (Layer-4) of the OSI model.
  - Implements security of TCP sessions over encrypted SSL tunnels.
  - Commonly used to secure applications that rely on secure HTTP as the transport (HTTPS).



- Technologies similar to SSL do exist if one wants to encrypt UDP traffic (such as DTLS…Datagram Transport Layer Security).

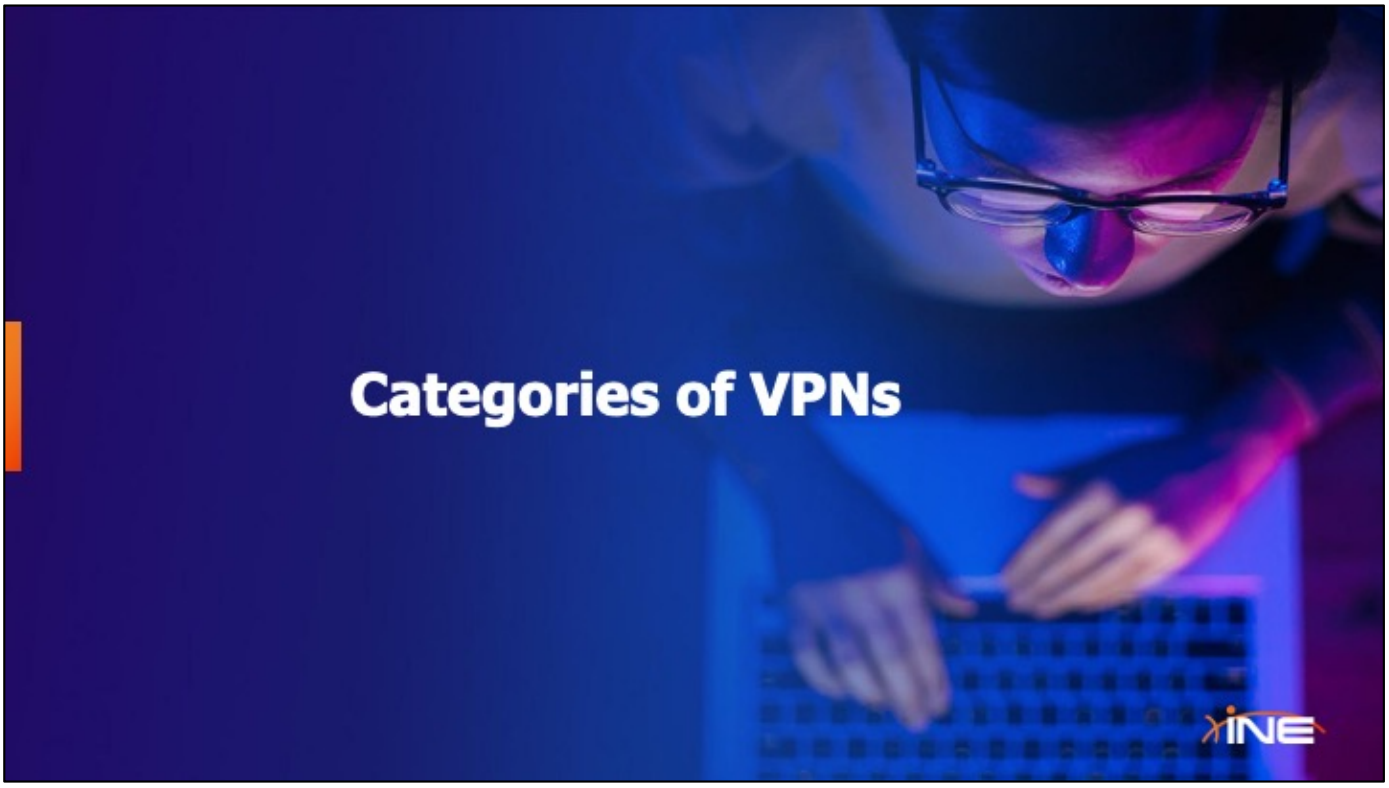
Primary VPN Technologies: MPLS

- MPLS (Multi Protocol Label Switching)
  - Secures traffic by allowing it only across pre-defined paths in Service Provider's network.
  - Pathways are selected based on Labels applied between L2 and L3 headers.
  - Does not provide any inherent confidentiality, integrity, or authentication mechanisms.



- If you wish to learn more about MPLS VPNs I recommend you view my course: Introduction to MPLS VPNS
- https://my.ine.com/Networking/courses/eaab1bdc/introduction-to-mpls-vpns

Thank you for watching!

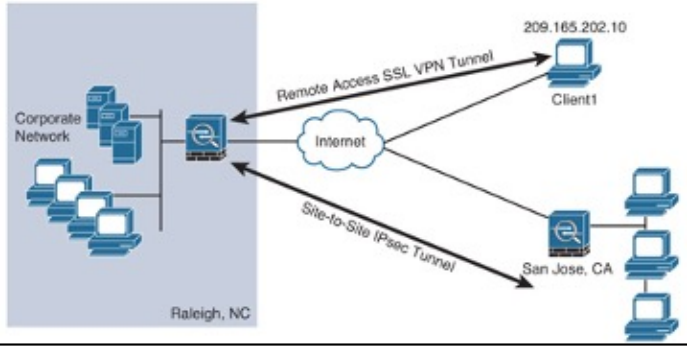


Topic Overview

+ Introducing Site to Site VPNs
+ Introducing Remote Access VPNs
+ Clientless & Client-Based VPNs
+ Route-Based VPNs
+ Policy-Based VPNs

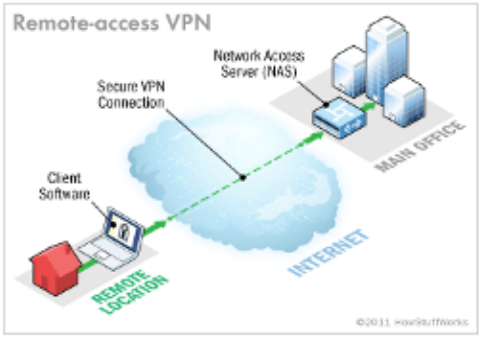
Categories Of VPNs: Site to Site

- Site-to-Site VPN
  - VPN endpoints are dedicated equipment (Routers, ASAs, etc)
  - Designed for scalability
    - Traffic from dozens/hundreds of hosts at one site can pass through a single VPN tunnel to the Corporate site.
    - No special VPN software required on end-user devices.
    - Traditionally leverages IPSec technologies.



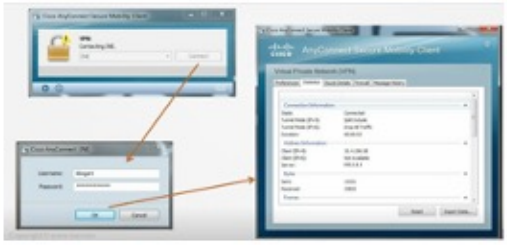
Categories Of VPNs: Remote Access

- Remote Access VPN
  - VPN initiated/built from your personal device (i.e laptop, tablet, computer, etc).
  - Can use IPsec or SSL, however SSL VPNs are less complex.



Clientless & Client-Based VPNs

- These terms relate to Remote Access VPNs
- *Clientless* VPNs
  - Require no special software on the endpoint
  - An example would be browser-based SSL/TLS VPN technology
- *Client-based* VPNs:
  - One end of the VPN tunnel is initiated from ***VPN client software*** running on an endpoint (laptop, PC, tablet, etc)
  - This is typically required when accessing IPsec-based VPNs
  - Many VPN clients are free to download



Route & Policy-Based VPNs

- IPSec VPNs can be categorized as Route-Based or Policy-Based
- The categorization is determined by the method the router (or Firewall) uses to determine which traffic should be sent over the VPN tunnel
- Route-based VPNs rely on the IPv4 (or IPv6) routing table directing traffic to a VPN "Tunnel" interface
  - Any packet directed to the logical Tunnel interface will automatically be sent over the VPN
- Policy-based VPNs rely on a configured policy (such as Access-Control Lists) to dictate which traffic should be sent across the VPN

Policy-Based VPNs

+ Policy-based VPNs typically rely on Cisco IOS “crypto-maps” paired with ACLs
  - Crypto maps

Traffic Selectors

10.10.0.0/16
190.168.0.0/24

```
ip access-list extended TS
 permit ip 10.10.0.0 0.0.255.255 10.20.20.0 0.0.0.255
 permit ip 10.10.0.0 0.0.255.255 10.20.30.0 0.0.0.255
 permit ip 192.168.0.0 0.0.0.255 10.20.20.0 0.0.0.255
 permit ip 192.168.0.0 0.0.0.255 10.20.30.0 0.0.0.255
 exit
```

Traffic Selectors

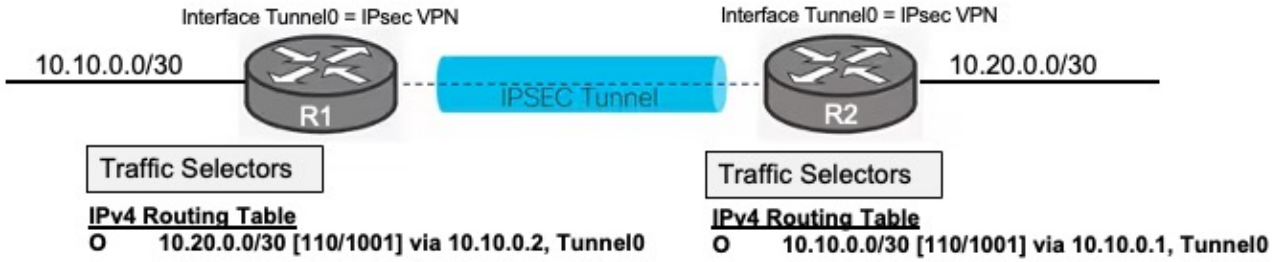
10.20.20.0/24
10.20.30.0/24

```
ip access-list extended TS
 permit ip 10.20.20.0 0.0.0.255 10.10.0.0 0.0.255.255
 permit ip 10.20.30.0 0.0.0.255 10.10.0.0 0.0.255.255
 permit ip 10.20.20.0 0.0.0.255 192.168.0.0 0.0.0.255
 permit ip 10.20.30.0 0.0.0.255 192.168.0.0 0.0.0.255
 exit
```

Image courtesy of:
https://www.cisco.com/c/en/us/support/docs/security-vpn/ipsec-negotiation-ike-protocols/217432-understand-ipsec-ikev1-protocol.html

Route-Based VPNs

+ Route-based VPNs support dynamic routing protocols across the VPN tunnel



+ Packets are routed to the Tunnel interface based on destination address

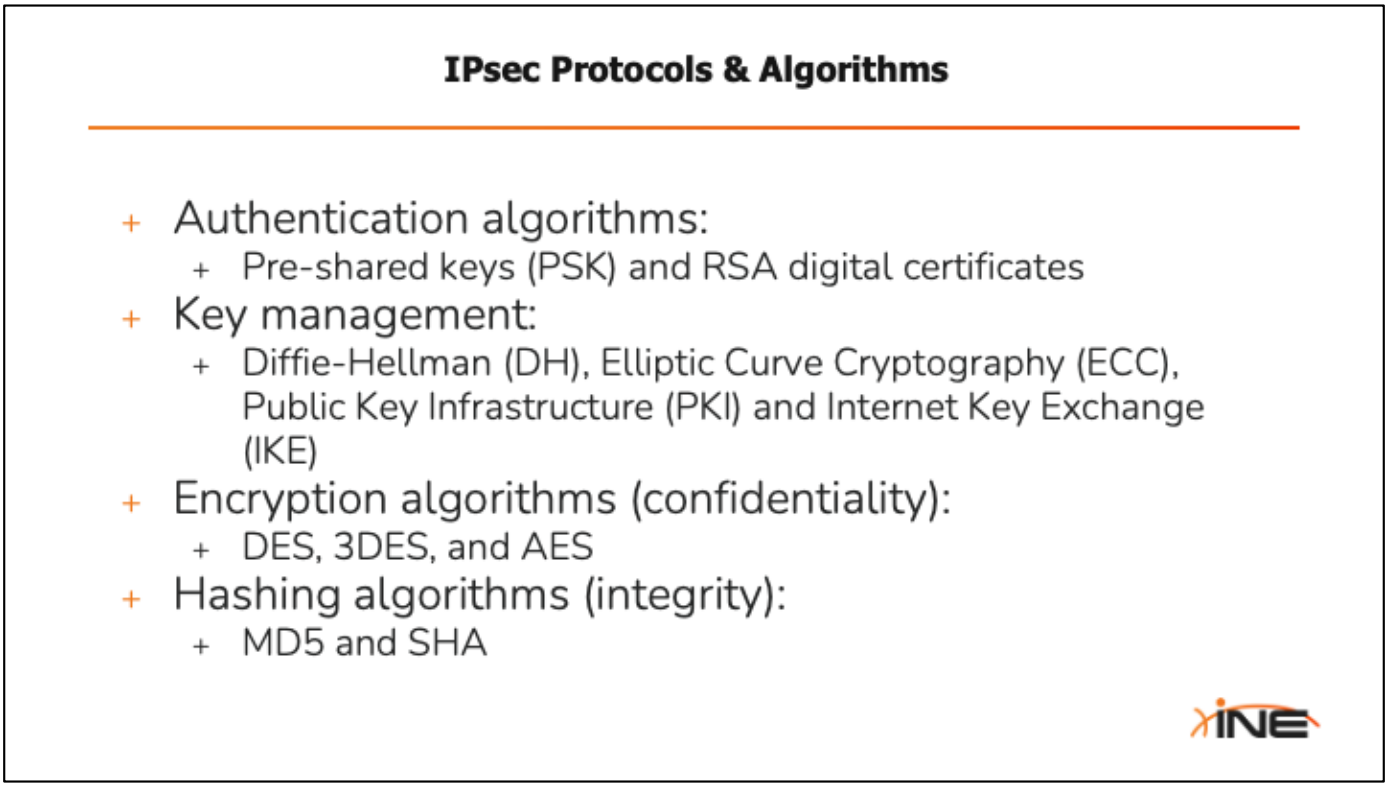
Thank you for watching!

Topic Overview

+ IPsec Overview
+ Introduction to IPsec Algorithms & Protocols
+ Functional Differences between AH & ESP
+ Transport vs Tunnel Modes

Introduction to IPsec

- Provides an architectural framework for IP data security.
- IPsec provides access to a suite of protocols and algorithms that can provide;
  - Confidentiality via encryption
  - Integrity via hashing
  - Authentication via keys or digital certificates
  - Anti-Replay protection
- IP Security (IPsec) originally defined in a group of RFCs
  - RFC 1825 through RFC 1829



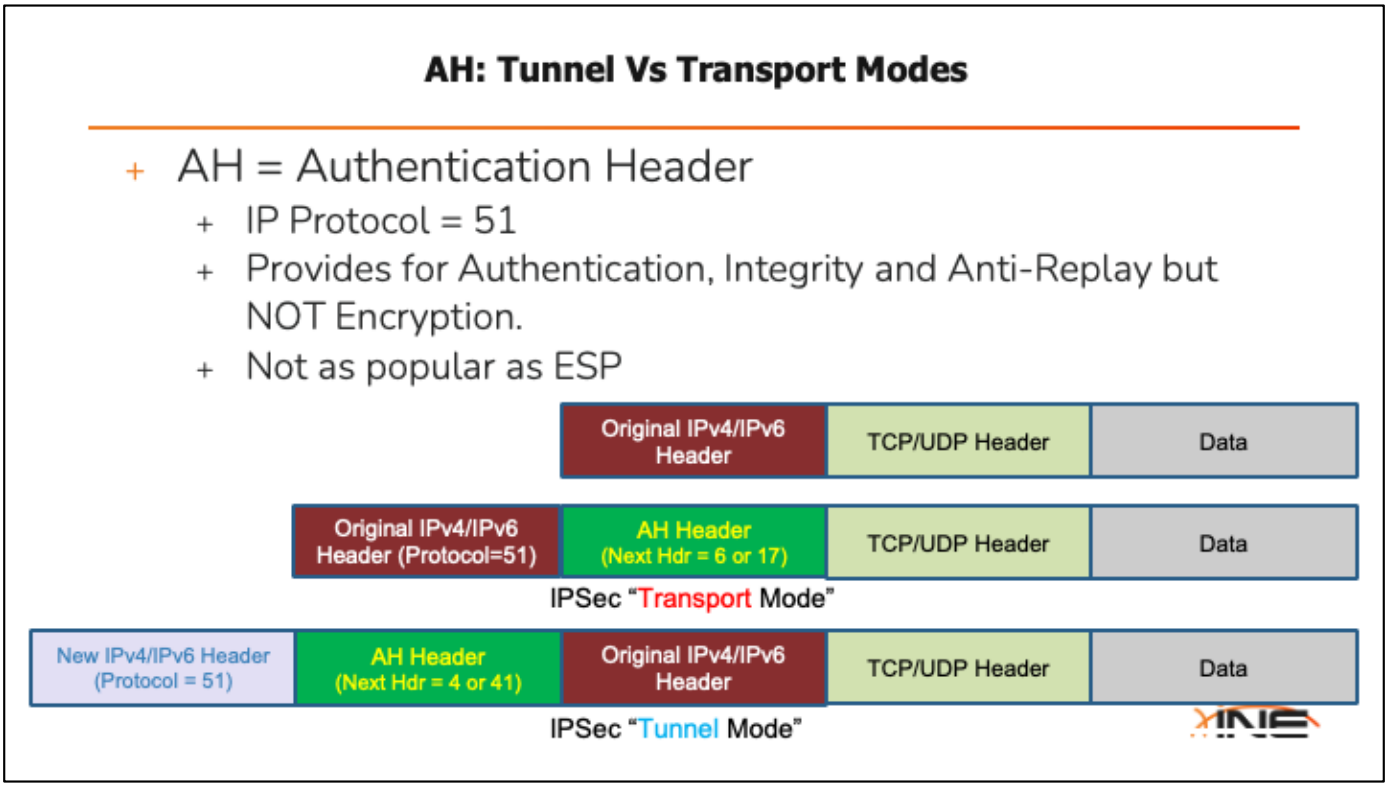
- DH can be used to dynamically generate symmetric keys
- PKI supports the functionality of digital certificates
- IKE performs much of the negotiation and management of keys between vpn peers.
- Just as ISAKMP is a framework (describes processes, packet formats, etc but not the actual mathematics and cryptographic algorithms themselves) IKE is also considered a “Framework” which can further use PKI or Diffie-Hellman as specific examples of protocols to fit the needs of its framework.

IPsec AH & ESP Protocols

- IPsec provides two implementations of packet security
  - Authentication Header (AH)
  - Encapsulating Security Payload (ESP)
- Authentication Header
  - Provides connectionless (i.e. UDP) integrity, data origin authentication, and an optional anti-replay service for IP packets.
  - AH does not provide encryption and confidentiality.
- Encapsulating Security Payload
  - Provides confidentiality (encryption), data origin authentication, integrity, and an optional anti-replay service.
- The first step of any IPsec design is to decide between AH or ESP

IPsec Modes

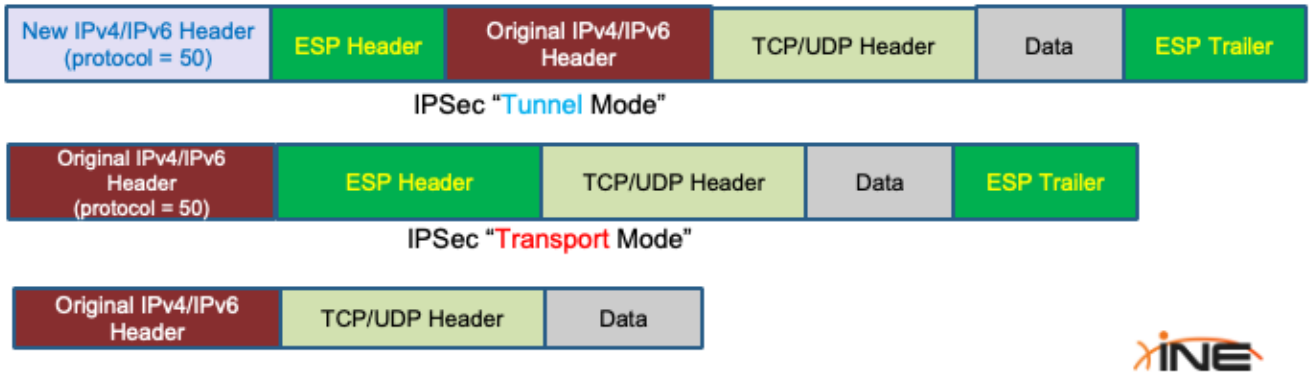
- IPsec can transport an IP packet in one of two modes
  - Transport Mode
  - Tunnel Mode
- Tunnel mode is more popular
  - Encapsulates entire IP packet (headers and data) within a new IP header
  - When encryption is used, this effectively hides the entire original packet
  - Good for when one must keep original source and destination of packet secret
- Transport mode
  - IPsec retains original IP packet header
  - IPsec AH and/or ESP headers are placed after original packet header
  - If encryption is used, only affects Layer-4 headers and data



- In Tunnel Mode, the Authentication Header, containing a "next-header" value of "41" indicates that an IPv6 header follows.
- U.S. federal laws prevent the export of certain encryption algorithms to some countries and localities. For this reason, if one lives in one of those places and you wish to have access to the authentication and integrity features of IPsec the Authentication Header is available.

ESP: Tunnel Vs Transport Modes

- ESP = Encapsulating Security Payload
  - IP Protocol = 50
  - Provides all features of IPsec, most critically, encryption.
  - Hides/encrypts all packet data (and possibly packet header) via encapsulation into a new IP header.



Thank you for watching!

Topic Overview

- What is an IPsec Security Association (SA) & Why is it Important?
- How SA's are Created
- Viewing IPsec SA Information
- What is an IPsec SPI?

Security Associations

- The goal of an IPsec implementation is to create a bidirectional, secure channel (i.e. "tunnel") between a pair of IPsec speakers.
- A Security Association (SA) defines the relationship between two IPsec peers and how the entities will communicate securely
- An SA is identified by a combination of three values:
  - SPI (Security Parameter Index)
  - Destination peer IP address
  - Method of security (AH or ESP) that has been implemented

- RFC 1825, section 1.4 identifies all of the elements commonly associated with an IPsec Security Association.

Security Associations

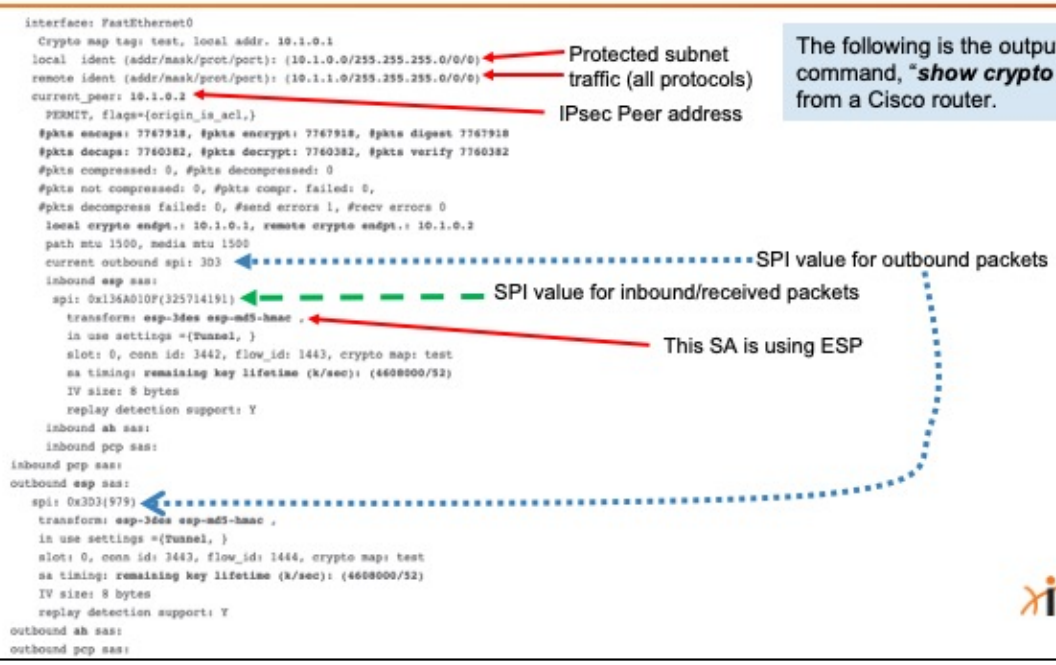
- IPsec peers create a Security Association by negotiating Authentication, Data Integrity, Confidentiality...or all three
- A security association is one-way.
  - An authenticated communications session between two hosts will normally have two Security Parameter Indexes (SPIs) in use (one in each direction).
  - The combination of a particular Security Parameter Index and a particular Destination Address uniquely identifies the Security Association.
- IPsec originally utilized *ISAKMP* for negotiating Security Associations
  - *I*nternet *S*ecurity *A*ssociation *K*ey *M*anagement *P*rotocol
  - Serves as a framework for authentication and key exchange, primarily designed to be agnostic of the specific mechanisms used for these purposes.
  - ISAKMP is practically implemented using IKE (Internet Key Exchange)

Guidelines for SA Configuration

- Security Associations must be defined for each IPsec peer identifying things such as:
  - Hashing algorithms to use for authentication
  - Encryption algorithms to use for confidentiality
  - Shared Secret Passwords
  - Much more...
- Security associations can be:
  - Manually configured between peers
  - Dynamically negotiated between peers
- Dynamic negotiation involves the IKE protocol

- IPsec can be configured without IKE, but IKE enhances IPsec by providing additional features, flexibility, and ease of configuration for the IPsec standard.

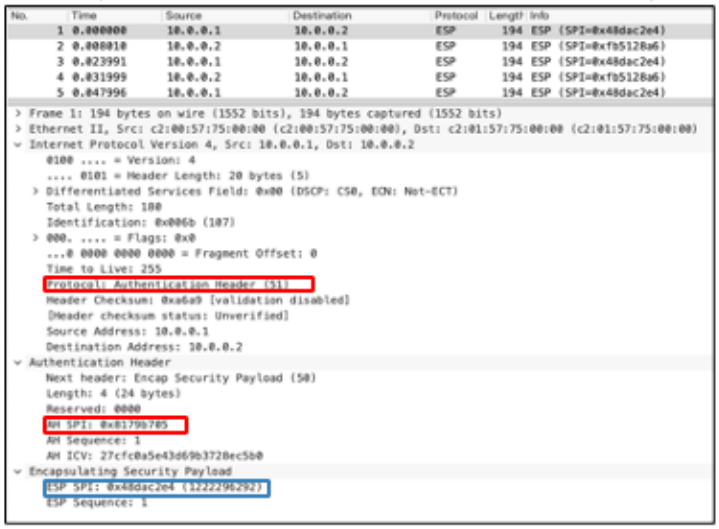
Example IPsec SA



The following is the output of the command, "***show crypto ipsec sa***" from a Cisco router.

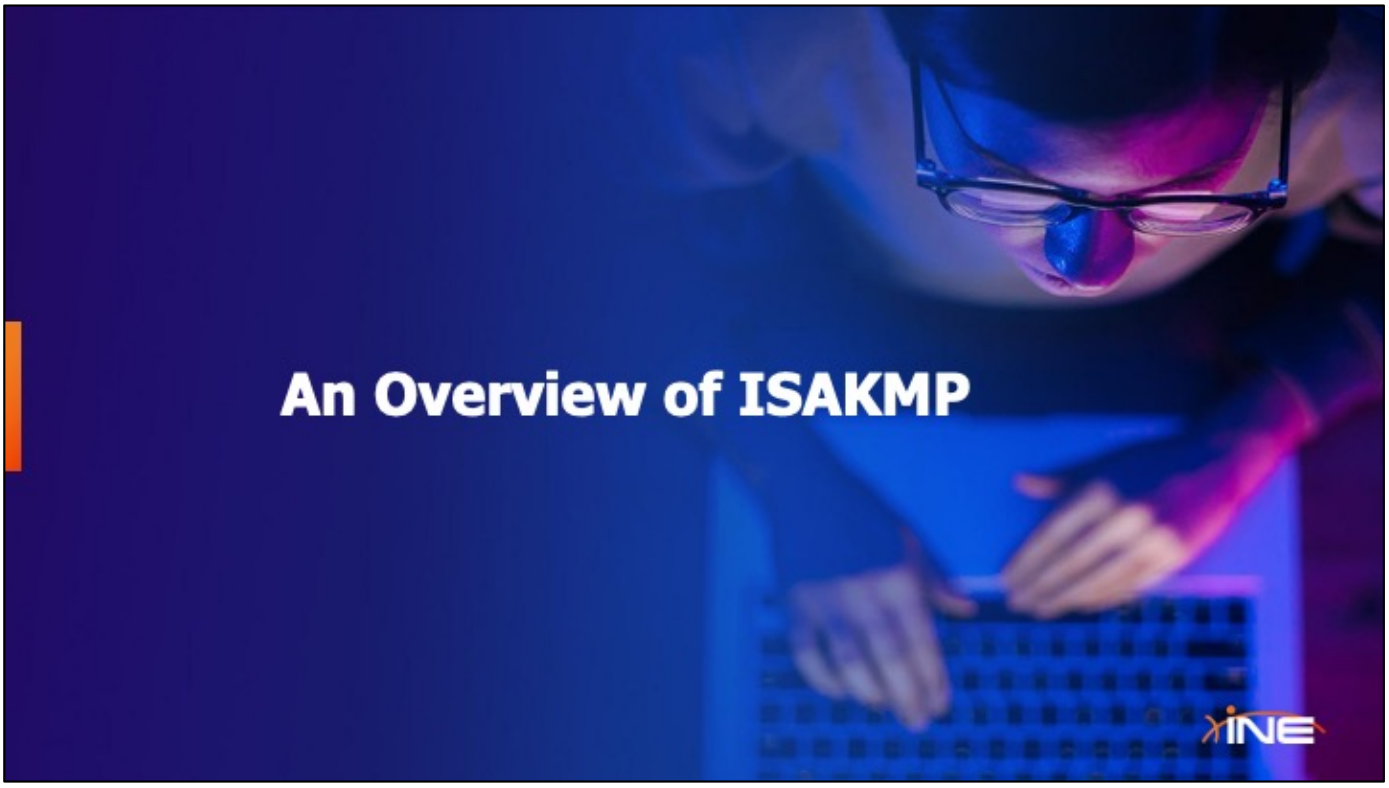
Conveying the IPsec SPI

+ Every IPsec packet transmitted contains an SPI value which the receiver uses to associate that packet with the correct IPsec Security Association (SA)



- In the context of IPsec, combining the Authentication Header (AH) with the Encapsulating Security Payload (ESP) can enhance security. Although ESP can provide authentication, integrity, and encryption, AH is used to offer additional protection against certain attacks, including replay attacks. It authenticates the entire packet, including the IP header, which ESP does not do when in transport mode.

Thank you for watching!



Topic Overview

- The Objectives of ISAKMP
- Overview of Algorithms Used by ISAKMP
- ISAKMP Header Structure
- ISAKMP Phases

Cryptographic Key Usage

- IPsec's goal: to create security associations between IPsec peers
- Security associations require cryptographic keys;
  - *Authentication*: keys are needed to create a digital signature or message authentication code (MAC) that validates the sender's identity.
  - *Integrity*: keys are required to implement keyed hash functions.
  - *Confidentiality*: keys are needed to encrypt and decrypt data
- Both AH and ESP require IPsec to derive, maintain, rotate and delete cryptographic keys

ISAKMP: What Problem Was Solved?

- With regards to cryptographic keys IPsec required a protocol that answered questions such as;
  - What types of messages need to be exchanged between peers to create an SA?
  - How can message types be identified?
  - How will cryptographic keys be exchanged, rotated and deleted?
  - What other procedures and packet formats need to be defined to negotiate, establish, modify and delete Security Associations?
  - How can all of this be done while maintaining independence about specific mechanisms and algorithms to provide versatility and scalability?
- ISAKMP (Internet Security Association Key Management Protocol) was designed to answer these questions

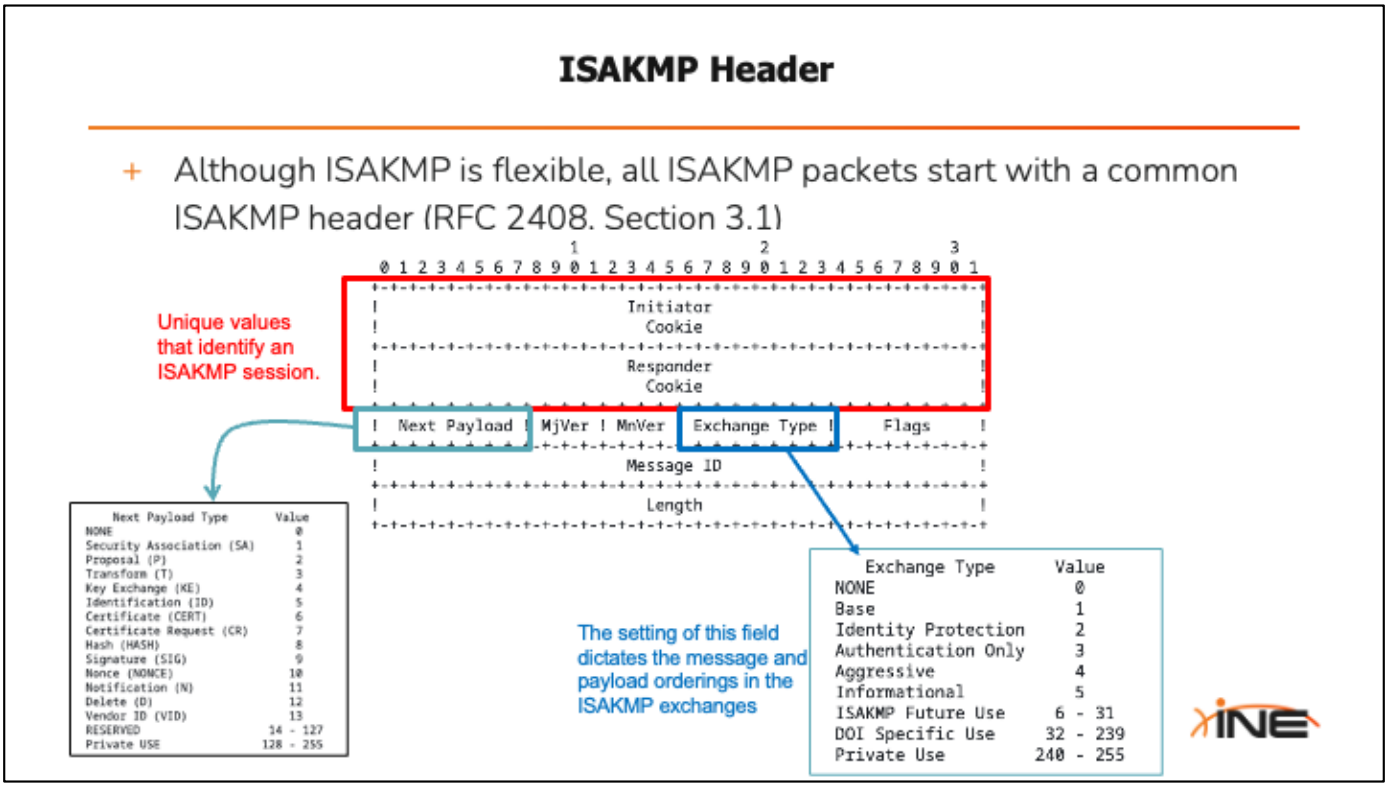
ISAKMP Architectural Goals

- Structured around a plug-and-play model that allows it to support various key exchange protocols, encryption algorithms, and authentication methods.
- ISAKMP can operate over any transport protocol *but typically uses UDP port 500*
- ISAKMP can accommodate new and evolving encryption technologies and strategies.
- The protocol's architecture separates the negotiation of SAs from the methodologies used for key exchange, authentication, and encryption
- Provides a flexible framework that can be adapted to meet the needs of different security environments.

Examples of ISAKMP's Flexibility

- RFC 2408 which specified ISAKMP was published in 1998
- At that time, the encryption algorithms in use (and supported by ISAKMP) included:
  - DES (Data Encryption Standard)
  - 3DES (called, "Triple DES")
  - RC4 and RC5 (Rivest Cipher, also called, "Ron's Code" after it's inventor Ron Rivest)
  - IDEA (International Data Encryption Algorithm)
- Since 1998, newer and stronger encryption algorithms have been developed which are also supported by ISAKMP including;
  - AES (Advanced Encryption Standard)
  - Blowfish
  - CAST-128/CAST5

- The encryption algorithms highlighted in red are those you should probably memorize if you plan on taking any certification exam.



- Most certification exams won't require you to know the structure or fields in the ISAKMP header. This information is given purely so that you can have a better understanding of ISAKMP and see how its structure lends itself to adaptability.
- RFC 2408 doesn't provide specifics of how the Cookies are to be generated. The main goal is that if a device has multiple, concurrent ISAKMP sessions each cookie needs to be a unique value.
- Next Payload: The ISAKMP packet can hold multiple kinds of "payloads" and more than one payload type can be contained in a single packet. Each payload type (defined in RFC 2408 beginning Section 3.4) has its own header with those headers ALSO containing a "next payload" field.
- RFC 2408 says "Major Version" should be 1 and "Minor Version" could be zero or 1.
- The "Exchange Type" field defines the overall nature of the exchange between the two entities. It determines the kind of ISAKMP interaction or negotiation that is to be conducted. Examples include:
    - Base Exchange: A basic form of exchange that might be used for initial setup.
    - Identity Protection (Main Mode): A more secure form of exchange providing identity protection.
    - Aggressive Mode: A faster but less secure exchange typically used where speed is preferred over security, like in scenarios requiring

fewer round-trips.

- Quick Mode: Used within an already established ISAKMP SA to negotiate a Security Association for use by IPSec.

- Each of these exchange types specifies a different sequence of messages and different requirements for the payloads within those messages. Essentially, the "Exchange Type" determines the procedure or script that the ISAKMP transaction follows, including how many messages will be exchanged, in what order, and what contents those messages should carry.

ISAKMP Phases: What Problem Was Solved?

- Although ISAKMP is primarily associated with IPsec, it was designed to be protocol agnostic and capable of supporting any security protocol that would require:
  - Creation of Security Associations
  - Exchange, maintenance and deletion of cryptographic keys
- ISAKMP had to answer a fundamental question; How can cryptographic keys for a security protocol (such as IPsec) be securely transferred between entities that have never spoken to each other?
- ISAKMP implements two “Phases” to answer this question.

ISAKMP Phases

- ISAKMP Phase-1
  - First set of ISAKMP messages exchanged between ISAKMP peers
  - Creates an ISAKMP Security Association (SA)
  - Creates a secure tunnel to further protect negotiations for the Security Protocol (i.e. IPsec) utilizing ISAKMP
- ISAKMP Phase-2
  - Establishes the SA for IPsec (or any future security protocols that may utilize ISAKMP)
  - This is the phase in which IPsec cryptographic keys used for IPsec AH and/or ESP are exchanged as well as other SA-related information.
  - Messages exchanged during this phase are protected as a result of the Phase-1 SA.

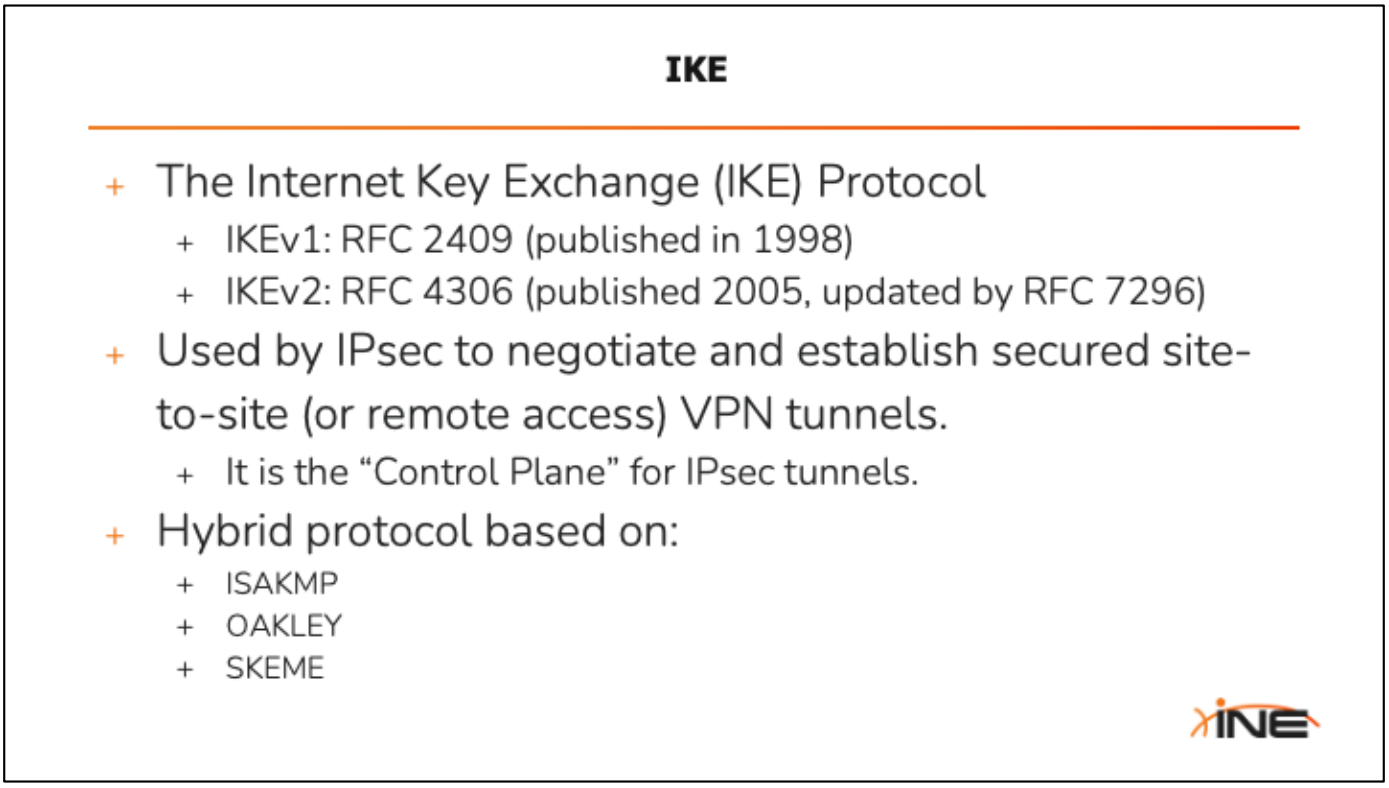
Practical Implementations of ISAKMP

- Created as a framework for secure key exchange and the creation of Security Associations, several protocols have been created that are (or were) practical implementations of ISAKMP
  - IKE (Internet Key Exchange): The de facto standard protocol used by IPsec VPNs
  - SKEME: Developed by Hugo Krawczyk and an early precursor of IKE
  - Oakley Key Determination Protocol: Designed to establish and authenticate key agreement over an insecure connection. Also factored into the development of IKE
  - KINK (Kerberized Internet Negotiation of Keys): This protocol is designed to establish IPsec security associations for Kerberos-authenticated clients without requiring the extensive negotiations typically involved in IKE.

Thank you for watching!

Topic Overview

+ IKE Objectives
+ IKE History
+ IKE Phases
+ Introduction to Diffie-Hellman
+ IKE Main Mode Steps
+ IKE Aggressive Mode Overview



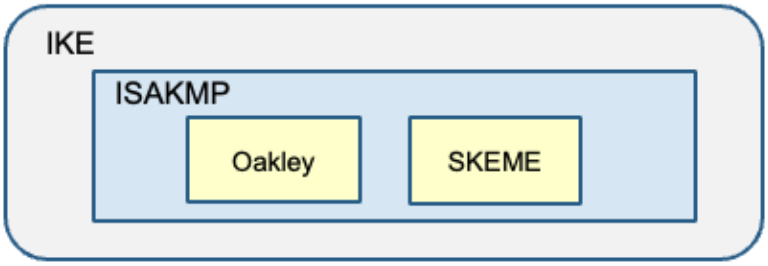
- RFC 2409 defining IKEv1 was published the same year as the publication date for RFC 2408 (ISAKMP). This is because IKEv1 was the first practical application/protocol to make use of the framework defined by ISAKMP.
- One of the first sentences in the IKEv1 RFC (2409) describes IKE as a "Hybrid Protocol". From the RFC (2409 Section-2);

  This does not implement the entire Oakley protocol, but only a subset necessary to satisfy its goals. It does not claim conformance or compliance with the entire Oakley protocol nor is it dependant in any way on the Oakley protocol.

  Likewise, this does not implement the entire SKEME protocol, but only the method of public key encryption for authentication and its concept of fast re-keying using an exchange of nonces. This protocol is not dependent in any way on the SKEME protocol.

Components Of IKE

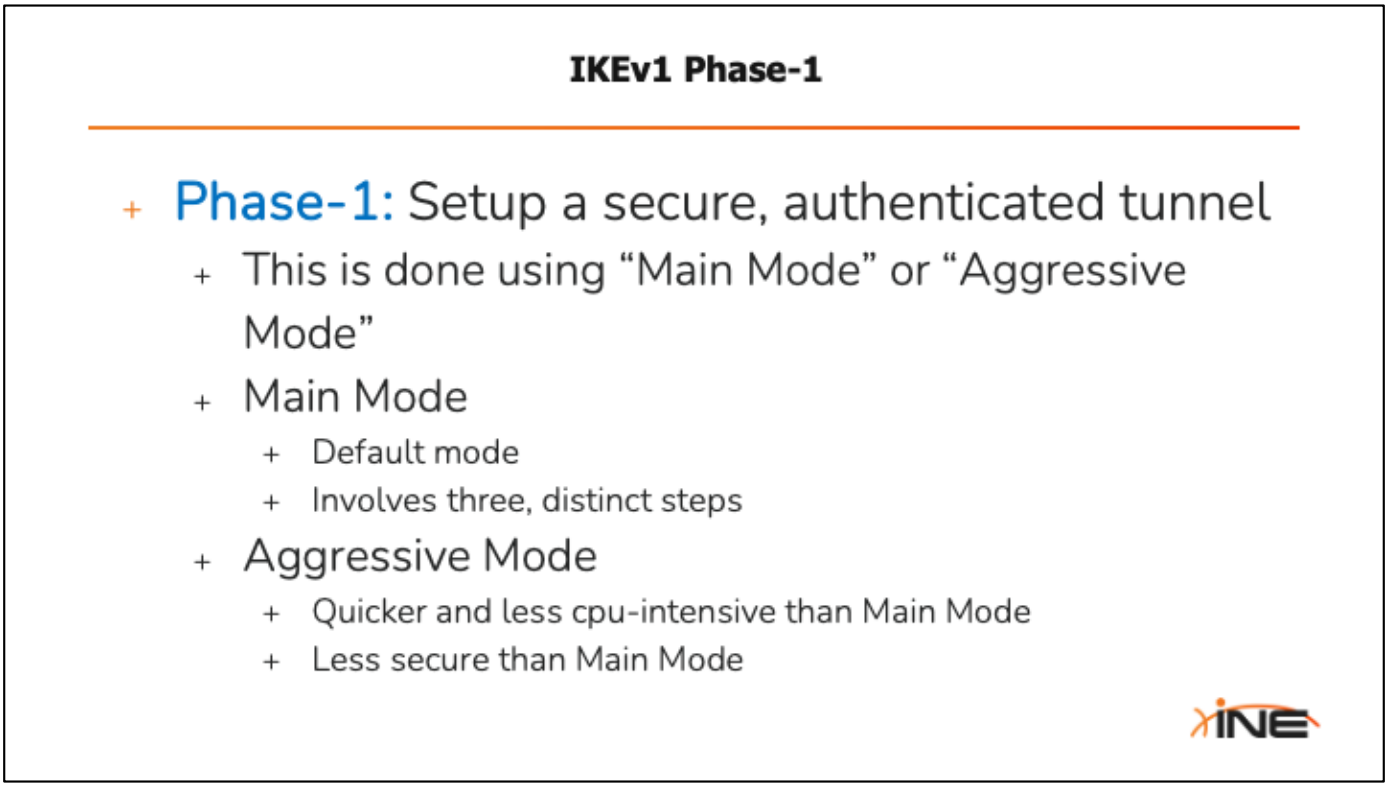
- ISAKMP: Internet Security Association and Key Management Protocol
  - A framework defining the rules, packet formats, and mechanics of implementing a key exchange protocol, and the negotiation of a security association.
- Oakley: describes a series of key exchanges (Modes) and services
- SKEME: key exchange technique that provides anonymity, nonrepudiation, and key refreshment



- ISAKMP is more conceptual/theoretical in nature. IKE is a practical realization of one way, of implementing ISAKMP.
- -
- Nonrepudiation means, “you can’t challenge me”… is the assurance that someone cannot deny something

Building The IPsec Tunnel With IKEv1

- IKEv1 creates two, secured tunnels
- This process takes place over two (2) phases
  - Phase-1: Setup a secure, authenticated tunnel
  - Phase-2: Negotiates Security Associations (SA's) for the data plane protocol
- IKE is carried by *UDP, Port-500*



- Main mode: Uses a total of six packets (three “pairs” of packets) to accomplish the job. Provides “Peer Identity Protection”
- Aggressive mode: Reduces this to a simple 3-way handshake (3-packets). Does NOT provide Peer Identity Protection.

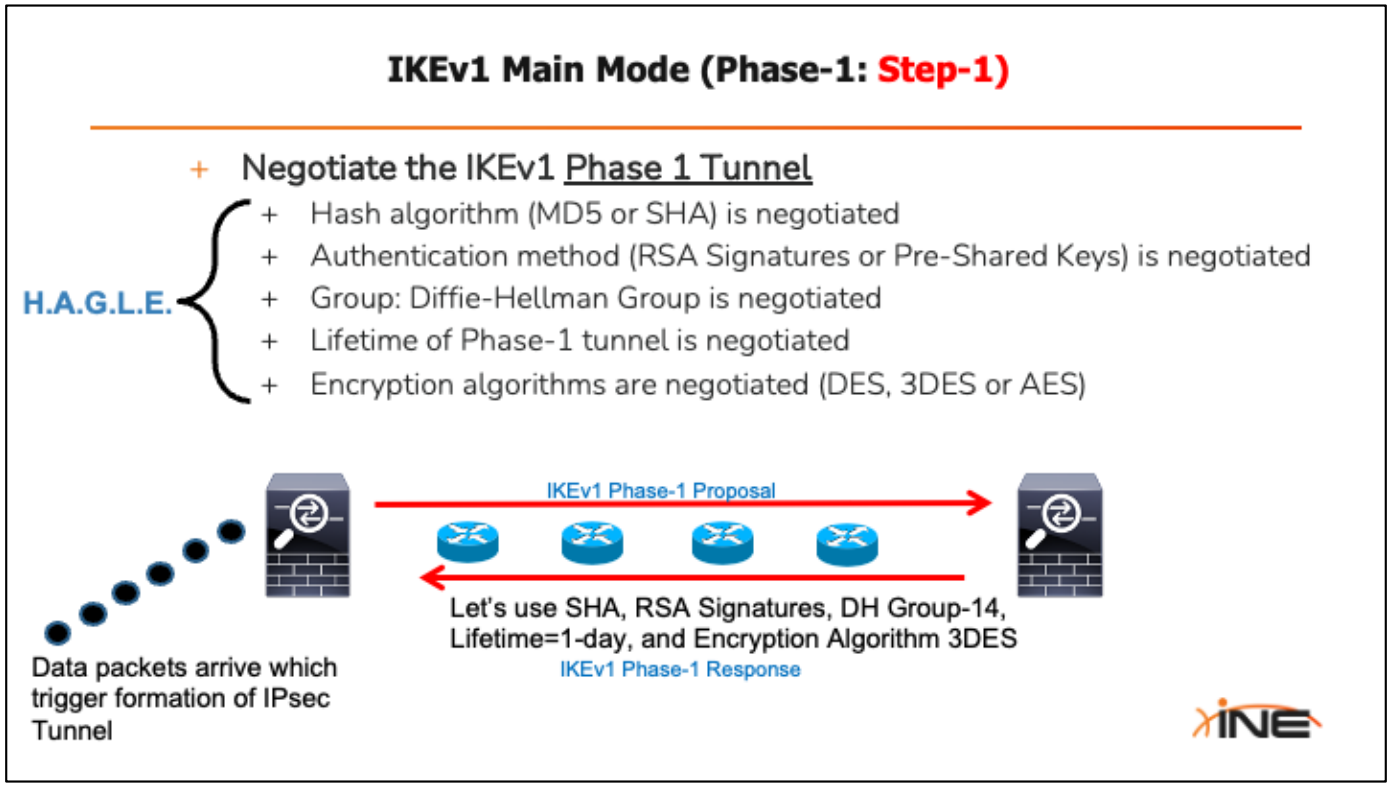
A Problem IKE Had To Solve

- The main challenge presented to IKE; securely generate a shared secret (key) between two parties over an unsecured communication channel.
  - If someone is watching your back-and-forth communications, how can you securely create and exchange cryptographic keys?
- What is this "shared secret" used for?
  - Protect (via encryption) the authentication credentials passed between IPsec peers during IKEv1 Phase-1
  - Protect (via encryption) the negotiations performed during Phase-2
- The Diffie-Hellman key-exchange algorithm solves this problem

An Overview of Diffie-Hellman

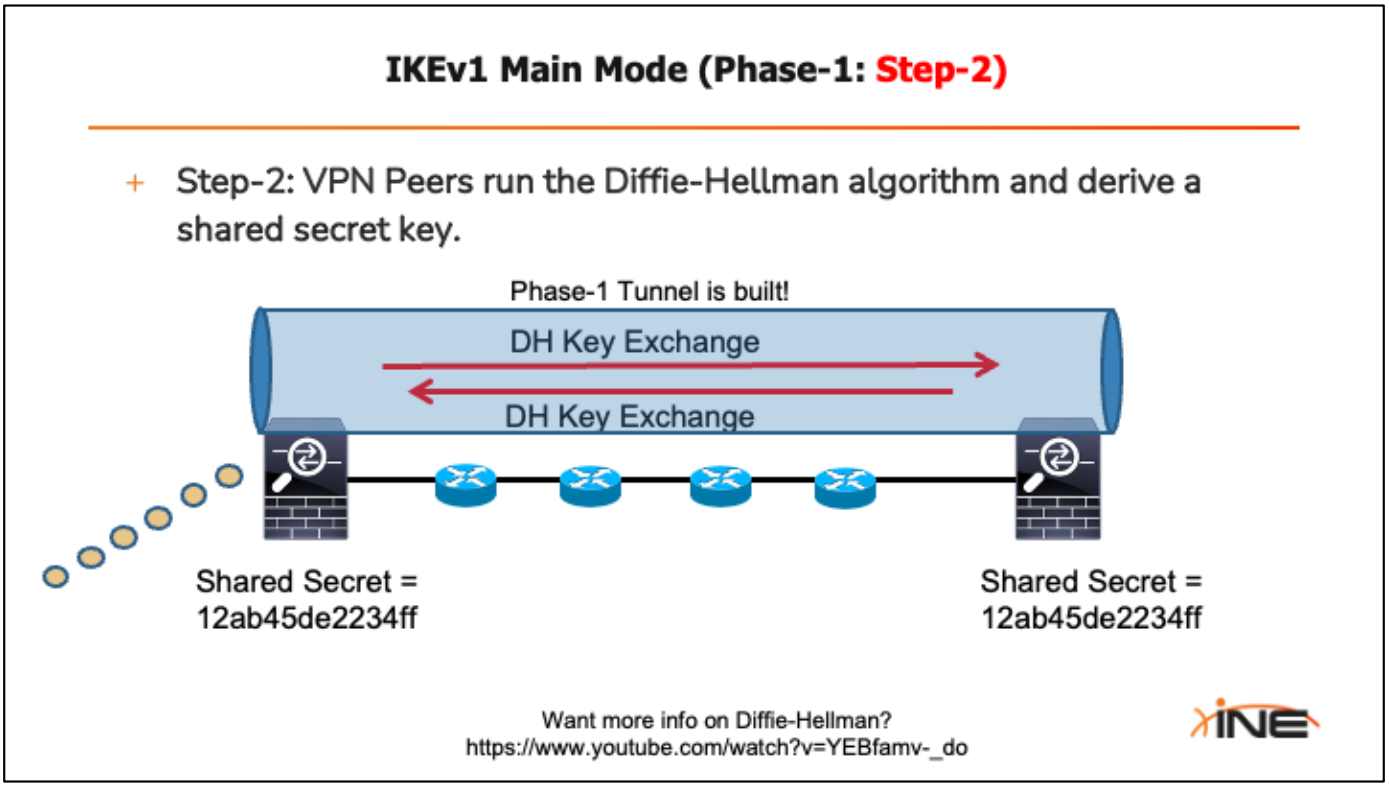
- The Diffie-Hellman algorithm is extremely mathematically complex.
- Involves the generation and exchange of prime numbers, a "base" number and other values so that both DH peers can derive a shared secret key without ever sending that key across the network.
- Diffie-Hellman provides different "groups" which provide predefined sets of parameters used in the Diffie-Hellman key exchange algorithm

* **Group 1**: Offers the least security with a 768-bit prime. It's considered deprecated due to its susceptibility to modern cryptographic attacks.
* **Group 2**: Uses a 1024-bit prime, providing moderate security. However, advancements in computational power have also rendered this group potentially vulnerable to well-funded attackers.
* **Group 5**: Features a 1536-bit prime, offering stronger security and is suitable for many contemporary applications.
* **Group 14**: Steps up to a 2048-bit prime, aligning with current recommendations for sufficient security against current attack methods.
* **Group 15 and 16**: These groups use 3072-bit and 4096-bit primes, respectively, providing even higher security levels suitable for environments with very high-security requirements.
* **Groups 19, 20, and 21**: Introduce Elliptic Curve Diffie-Hellman (ECDH) groups, using elliptic curve cryptography for key exchange, providing similar levels of security to larger prime-based groups but with smaller key sizes, leading to performance improvements.

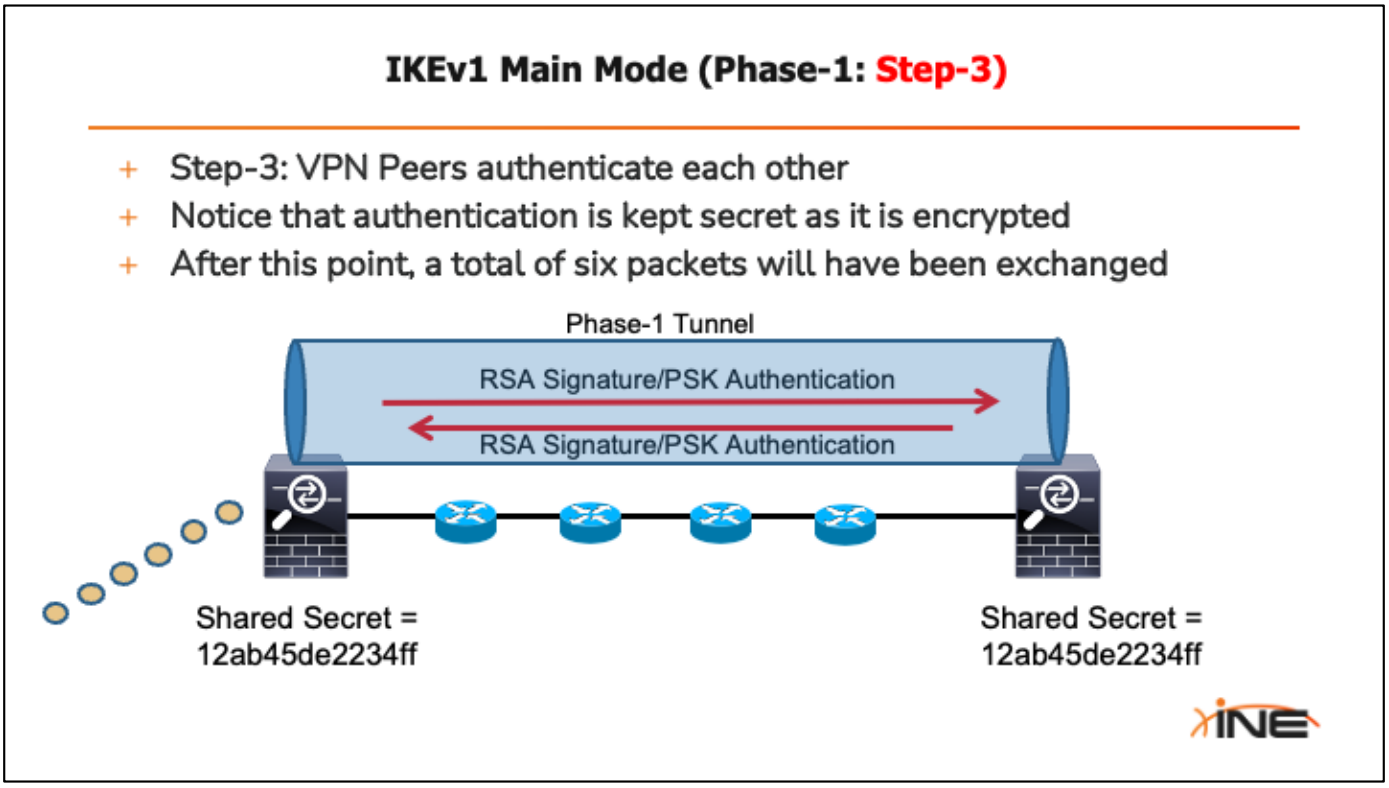


- Typically, packets have to arrive first that match an ACL to trigger the building of the IPsec tunnel.
- ---
- IKE Phases utilize UDP packets (src/dest port 500)

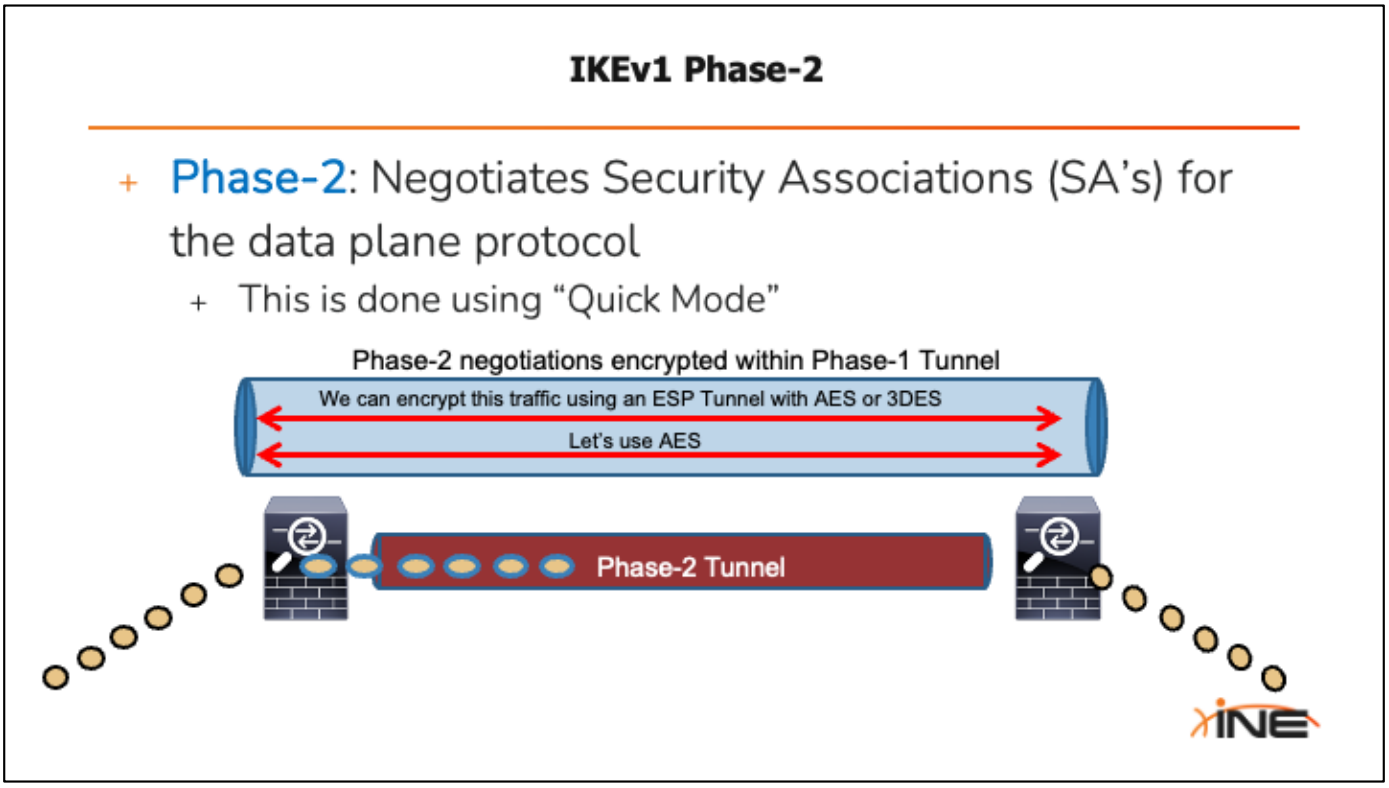
- What is being demonstrated on the next few slides is IKEv1 using Main-Mode.
- -
- Diffie-Hellman: A complex, mathematical function in which two VPN peers can exchange a shared number and (by mathematically combining that with their own secret values) come up with a shared secret such that any eavesdropper watching this exchange would not be able to derive the same shared secret.



- By the end of Step-2, enough information has been exchanged for the Phase-1 tunnel to be built, so all other exchanges after this can be encrypted…but there is still one more Phase-1 step to go…
- -
- When DH is done, both devices have their own unique Public/Private Keys (that have been dynamically derived) as WELL as
- A “shared secret key” that was derived at the end of the DH process. This shared secret key is used to encrypt everything after this point but it is NOT used to authenticate each other. Authentication (which will take place in Step-3 of Phase-1 (next slide) uses either a pre-shared key (that you’ve manually configured) or RSA Signatures.



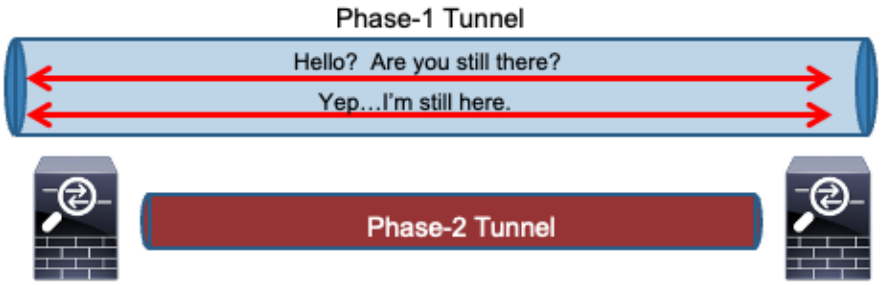
- Remember that with the first bidirectional exchange of IKE Phase-1 packets both peers agreed to the encryption algorithm to be used and how they would authenticate each other. Once the secret cryptographic keys were derived by Diffie-Hellman in the subsequent exchange of packets (packets-3 and 4) everything was ready to encrypt the final exchange of packets (shown on this slide) so that the peers could securely authenticate each other.



- IKEv1 Phase-2 utilizes an exchange of three packets.
- -
- The Tunnel created after Phase-1 is completed is never used to actually encrypt and transport user-data. Instead, it is used to send Ipsec control traffic.

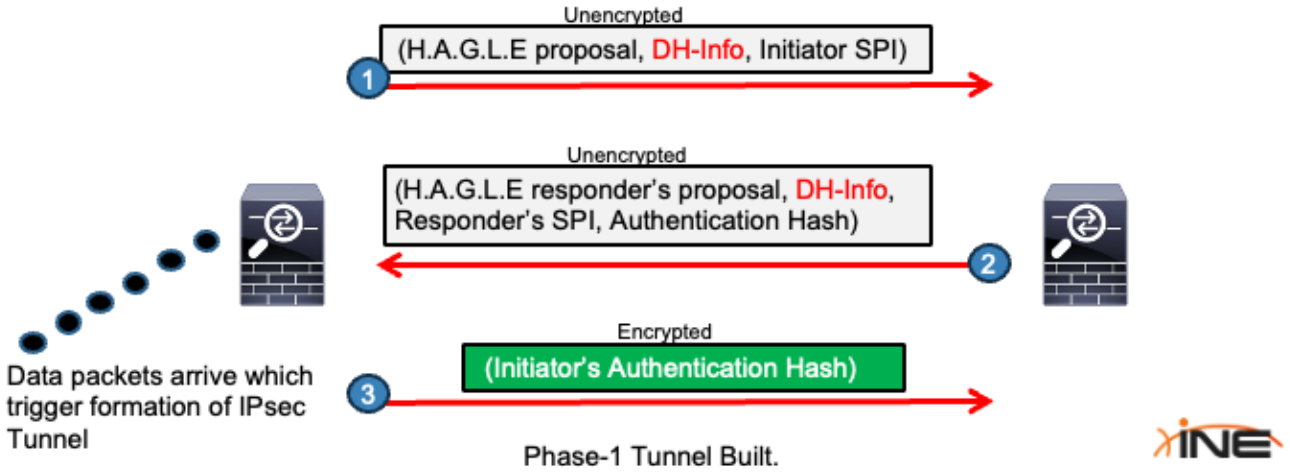
IPsec Tunnel Maintenance

- Even after there are no more packets to encrypt, Phase-1 tunnel remains up for tunnel maintenance
- Phase-1 tunnel = Control Plane for IPsec



IKEv1 Aggressive Mode

+ Although not generally recommended for use, Aggressive mode reduces the Phase-1 packet exchange from six packets down to three (3) but sacrifices some security in the process.



- Recall that with IKE, if a Pre-Shared key is selected as the HAGLE authentication method the key itself is never sent across the wire but rather a hash digest of that key. Because the second packet in the Aggressive-Mode exchange is NOT encrypted yet, if the pre-shared key was something easy to guess (such as “admin” or “Cisco”) then the unencrypted hash of it sent from the IKE Responder could easily be reverse-engineered using a dictionary attack.
- Similarly, if Digital Certificates were selected as the HAGLE authentication method, the Responder’s Digital Certificate would be sent, unencrypted in that second packet.
- This underscores the potential vulnerabilities of using Aggressive mode and the general recommendation is NOT to use it.

Thank you for watching!

Topic Overview

+ High-Level Overview of IKEv1 Configuration Process
+ IKEv1 Cisco IOS Commands

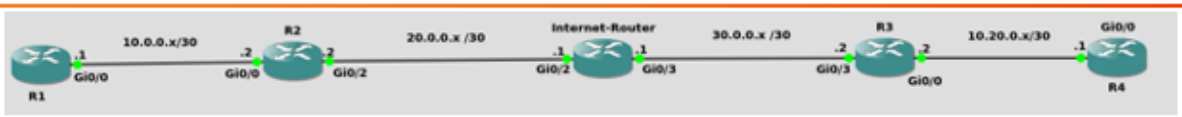
The Five Steps

- Configuring a site-to-site IPsec (IKEv1) VPN on a Cisco router involves five steps:

  ***Po**lite **Ke**ith **Tra**umatizes **Prof**ound **Prof**essors*

  1. Configure a Crypto *ISAKMP Policy* (defines Phase-1 attributes)
  2. Configure a Crypto *ISAKMP Key* (when using Pre-Shared key authentication)
  3. Configure a Crypto IPsec Transform-Set (defines Phase-2 attributes)
  4. Configure a Crypto *IPsec Profile*
  5. Apply Crypto *IPsec Profile* to Tunnel interface
- Additionally, something must exist which routes interesting traffic to the tunnel interface such as;
  - Static routes
  - A routing protocol

IKEv1 Phase-1 Configuration

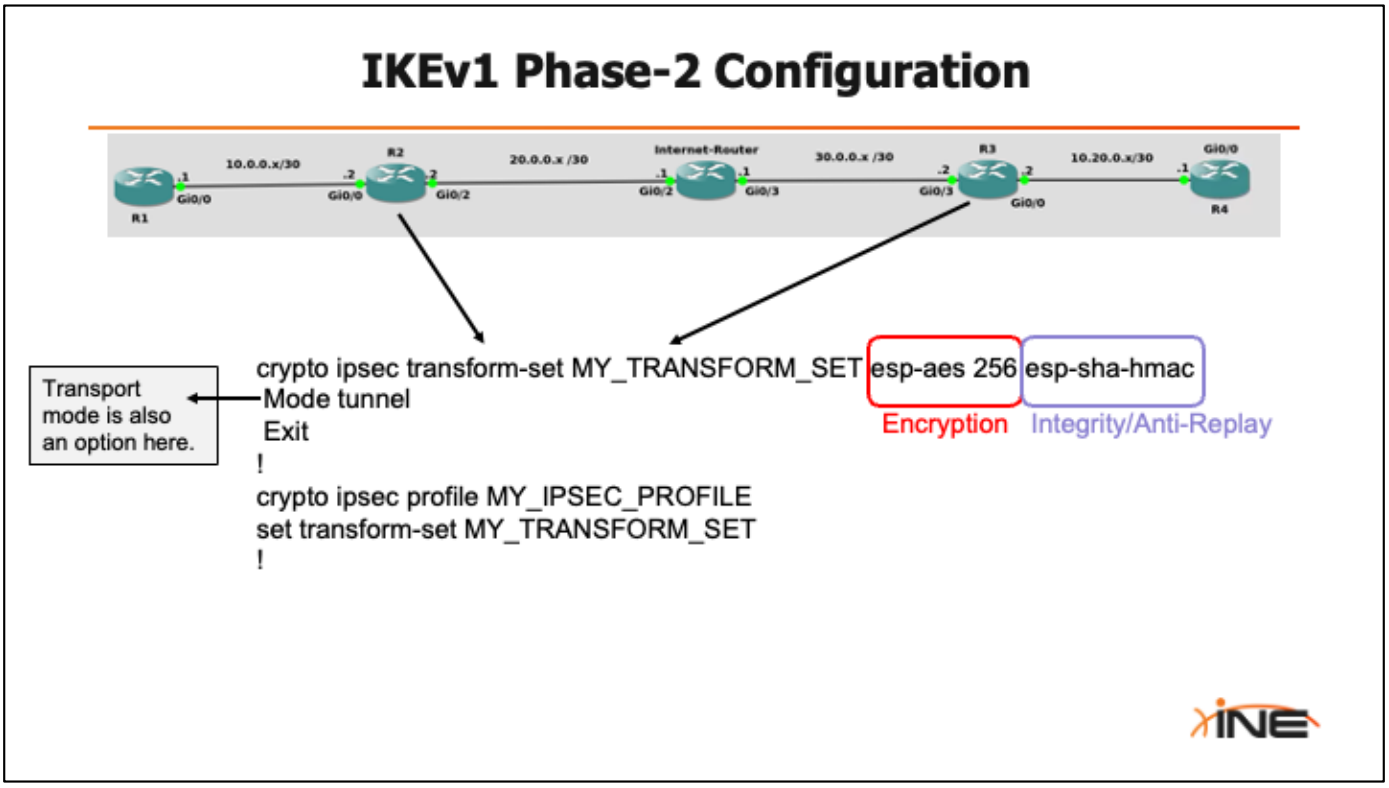



```
crypto isakmp policy 10
 hash sha256
 authentication pre-share
 group 15
 lifetime 43200
 encryption aes 256
!
crypto isakmp key MY_KEY address 30.0.0.2
!
```

IKEv1
Phase-1
H.A.G.L.E.

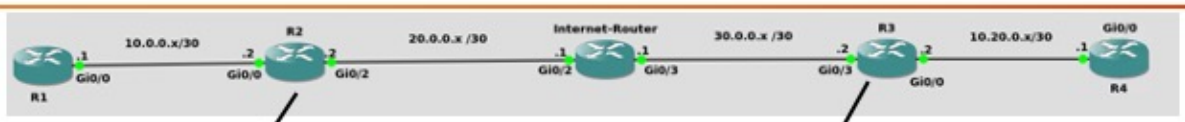
```
crypto isakmp policy 10
 hash sha256
 authentication pre-share
 group 15
 lifetime 43200
 encryption aes 256
!
crypto isakmp key MY_KEY address 20.0.0.2
!
```

Shared secret authentication key
(DON'T use this one in real life!)



- Remember that the Authentication features of IPsec already occur during the IKEv1 Phase-1 process. So in the transform-set, the second variable (shown as “esp-sha-hmac” in this case) does not relate to authentication as that process has already completed.
- Remember that when selecting “Transport Mode” the original source and destination IPv4/IPv6 packet headers will be visible and only the payload of the packet will be encrypted making that a suitable option to run on IPsec peers located within an Enterprise but not across the public Internet.

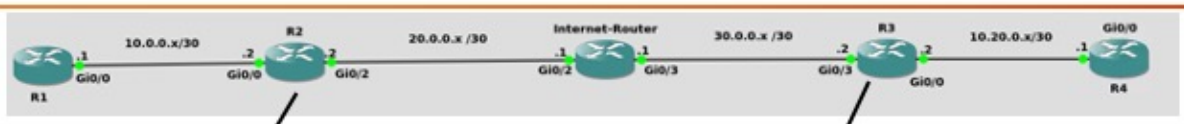
Associating The Config to an Interface




```
crypto ipsec profile MY_IPSEC_PROFILE
set transform-set MY_TRANSFORM_SET
!
interface Tunnel0
ip address 10.10.0.1 255.255.255.252
tunnel source Gig0/2
tunnel destination 30.0.0.2
tunnel mode ipsec ipv4
tunnel protection ipsec profile MY_IPSEC_PROFILE
ip ospf 1 area 0
```

```
crypto ipsec profile MY_IPSEC_PROFILE
set transform-set MY_TRANSFORM_SET
!
interface Tunnel0
ip address 10.10.0.2 255.255.255.252
tunnel source Gig0/3
tunnel destination 20.0.0.2
tunnel mode ipsec ipv4
tunnel protection ipsec profile MY_IPSEC_PROFILE
ip ospf 1 area 0
```

Static Route Alternative




```
crypto ipsec profile MY_IPSEC_PROFILE
set transform-set MY_TRANSFORM_SET
!
interface Tunnel0
ip address 10.10.0.1 255.255.255.252
tunnel source Gig0/2
tunnel destination 30.0.0.2
tunnel mode ipsec ipv4
tunnel protection ipsec profile MY_IPSEC_PROFILE
!
ip route 10.20.0.0 255.255.255.252 tunnel0
```

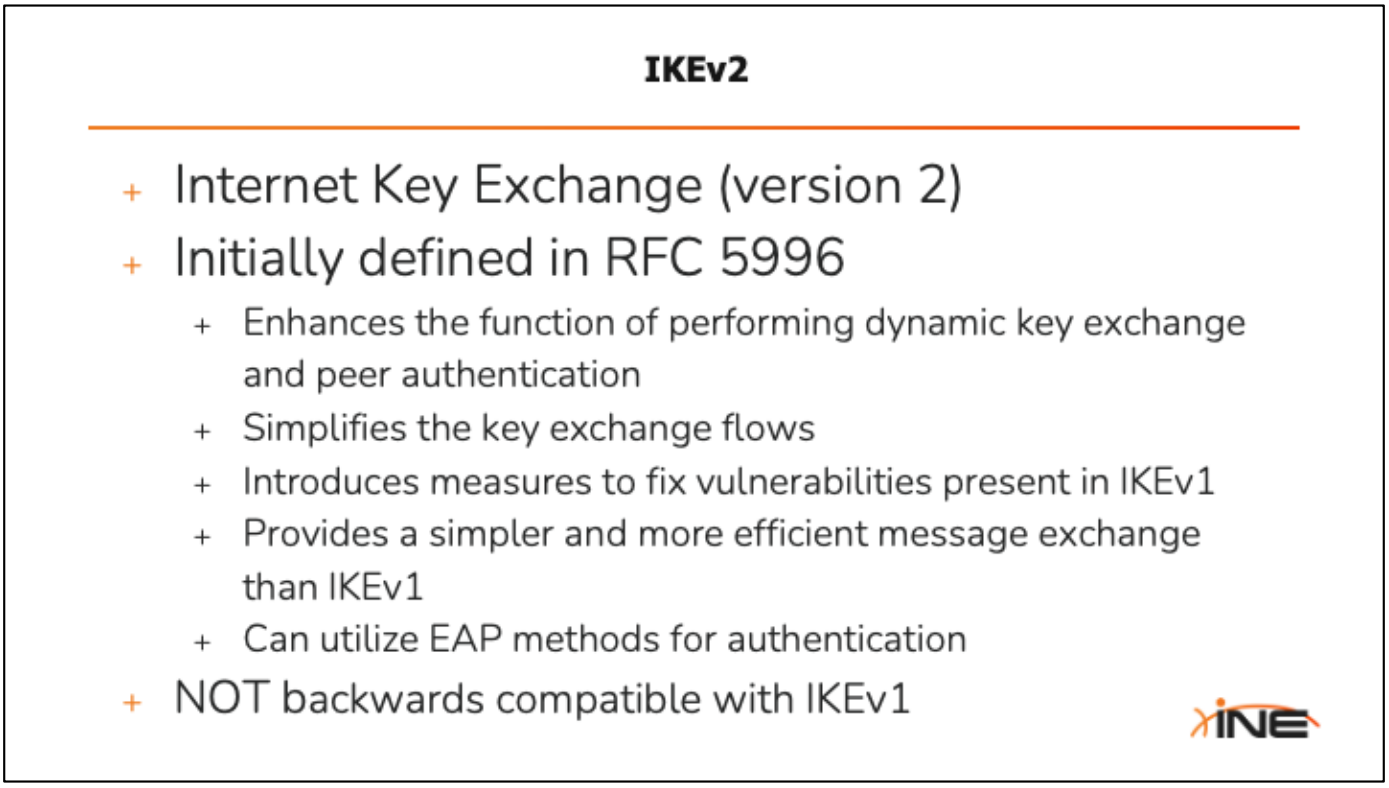
```
crypto ipsec profile MY_IPSEC_PROFILE
set transform-set MY_TRANSFORM_SET
!
interface Tunnel0
ip address 10.10.0.2 255.255.255.252
tunnel source Gig0/3
tunnel destination 20.0.0.2
tunnel mode ipsec ipv4
tunnel protection ipsec profile MY_IPSEC_PROFILE
!
ip route 10.0.0.0 255.255.255.252 tunnel0
```

Use the command, "***show crypto ipsec sa***" to verify Security Associations

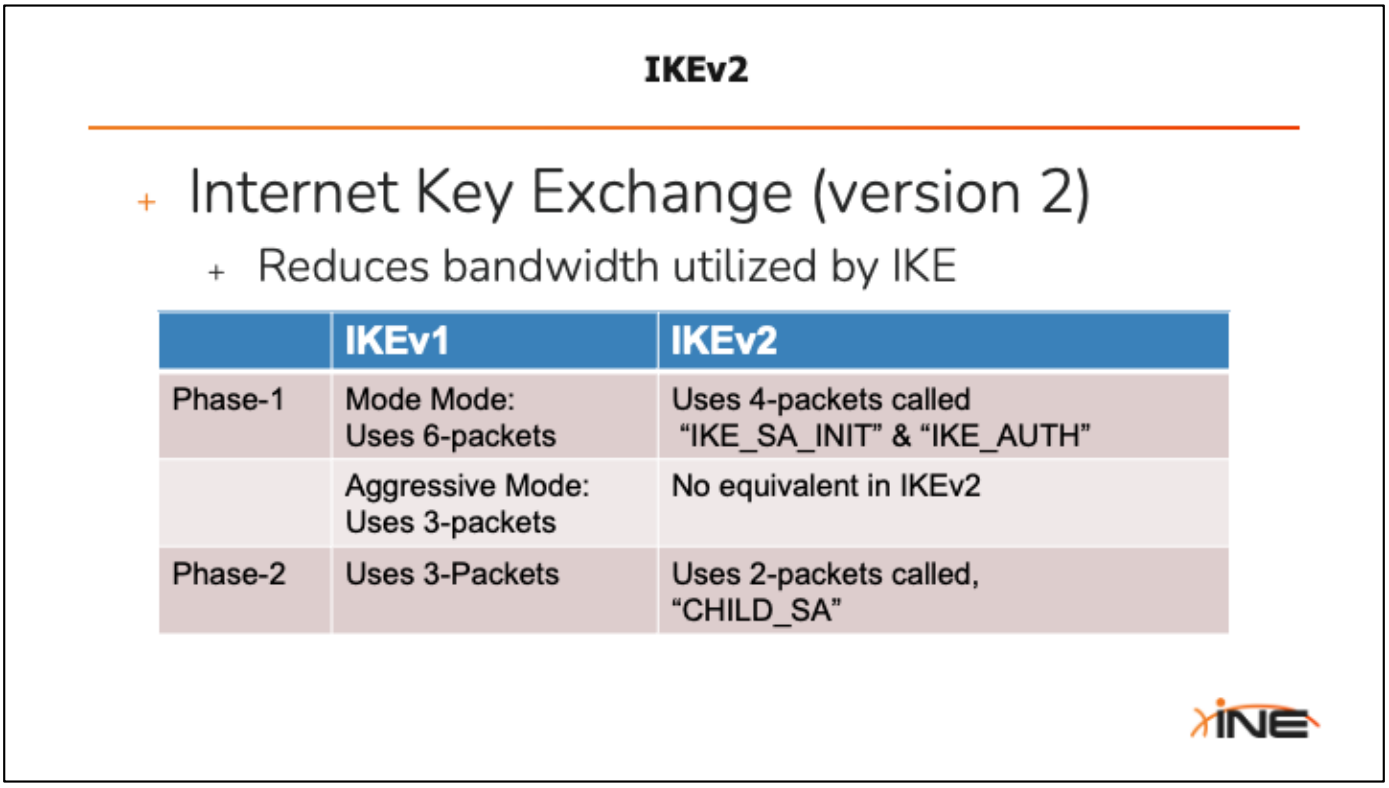
Thank you for watching!

Topic Overview

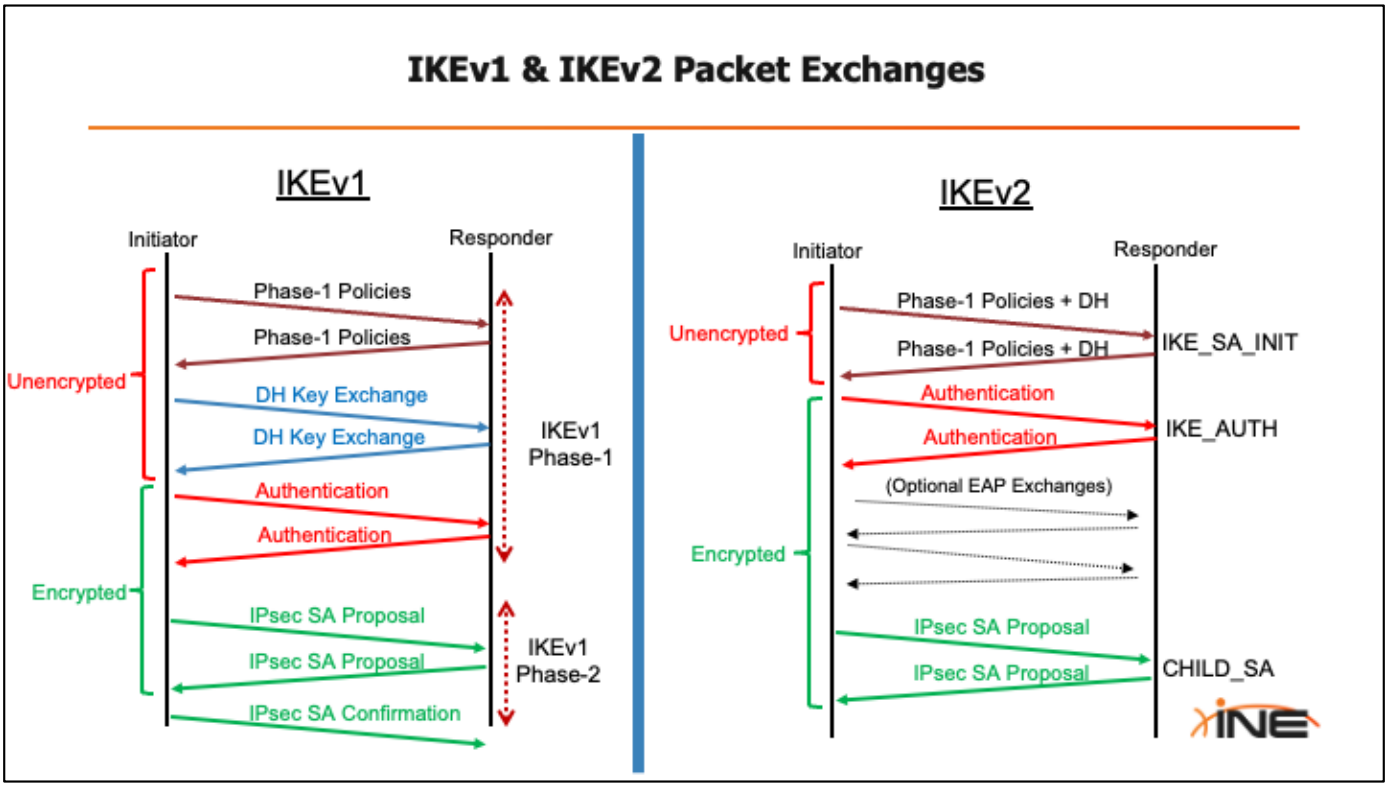
- Introduction to IKEv2
- Some Differences between IKE versions
- Contrasting IKEv1 & IKEv2 packet exchanges
- Contrasting IKEv1 & IKEv2 headers



- The latest RFC for IKEv2 is 7296.



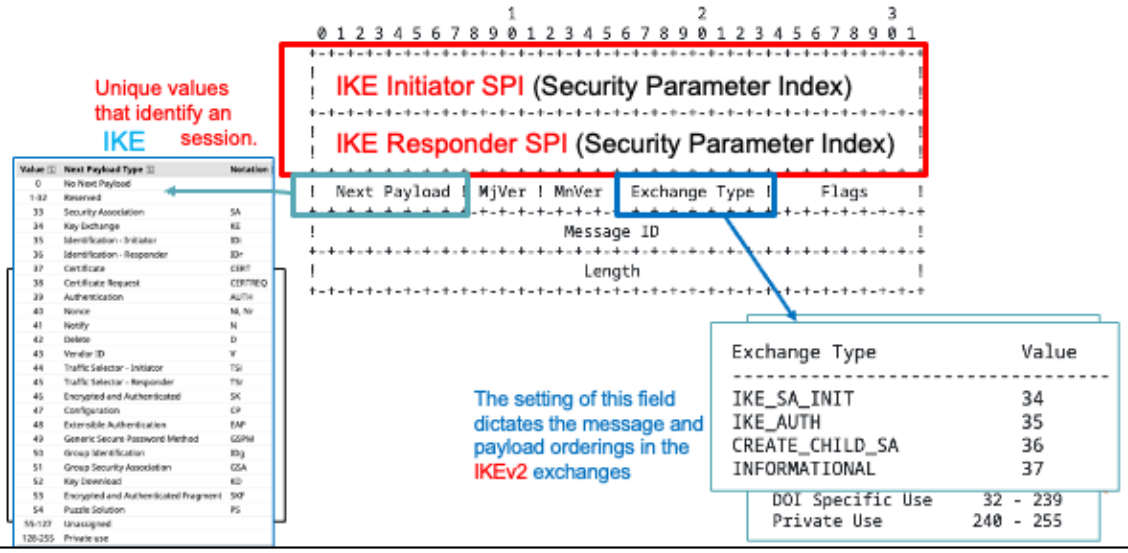
- IKE-SA = IKE Security Association



- IKEv2 doesn't support the concepts of "Main Mode" or "Aggressive Mode".

Contrasting the ISAKMP & IKEv2 Headers

+ Because IKEv1/v2 utilize ISAKMP as its foundation, they borrow the same header structure from ISAKMP



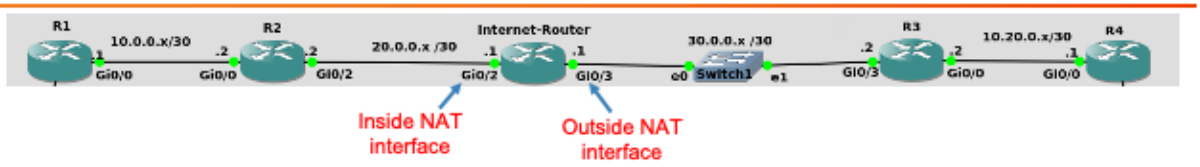
- Most certification exams won't require you to know the structure or fields in the IKE header. This information is given purely so that you can have a better understanding of IKE and see how its structure lends itself to adaptability.
- Next Payload: The IKE packet can hold multiple kinds of "payloads" and more than one payload type can be contained in a single packet. Each payload type (defined in RFC 7296) has its own header with those headers ALSO containing a "next payload" field.
- You'll notice that IKE retained the original ISAKMP "Next Payload" type values of 0 through 32, defining its own IKE-related Payloads beginning with value 33.
- Although (most of) the names within the "Next Payload" field are the same as those used with IKEv1, to make a clean differentiation between IKEv1 and IKEv2 and prevent confusion, the values for each of these fields was changed for IKEv2.

Thank you for watching!

Topic Overview

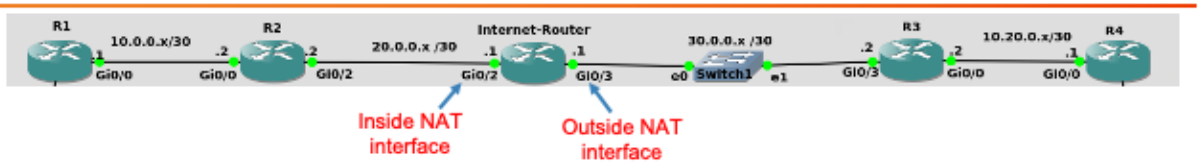
- Quick Review of NAT
- Interoperability Challenges between NAT & IPsec
- Brief Introduction to IPsec Pass-Through
- NAT Traversal
- NAT's Affect on IPsec Data Integrity

Quick Review of NAT



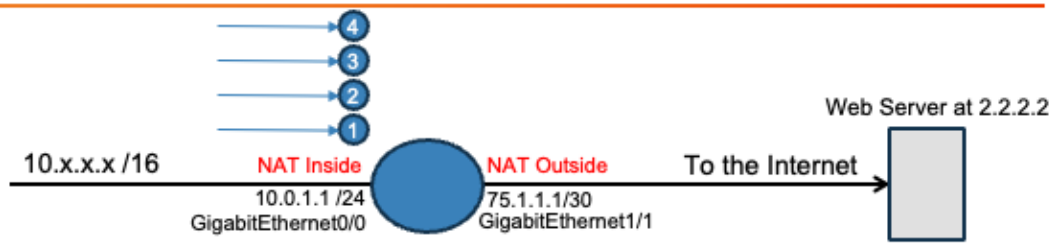
- NAT = Network Address Translation
- A protocol typically used to allow networks with private (RFC 1918) IPv4 addresses to be translated to public, globally-routable addresses
- Translates source IP address (and possibly source TCP/UDP port number) of packets traversing NAT router
- Requires three elements:
  - Definition of one-or-more NAT inside interfaces
  - Definition of one-or-more NAT outside interfaces
  - A NAT translation policy defining which packets to translate and how to translate them.

NAT Implementation



- NAT is typically implemented as PAT (Port Address Translation) in which;
  - All packets matching NAT policy are translated to a single, "Outside Global" address
  - Every packet flow leaving the NAT/PAT router must have unique source TCP/UDP port numbers
  - The combination of <***source IP***>, <**destination IP**>, <***source TCP or UDP port***> number helps PAT router maintain distinction between translated flows.

A Sample NAT Table

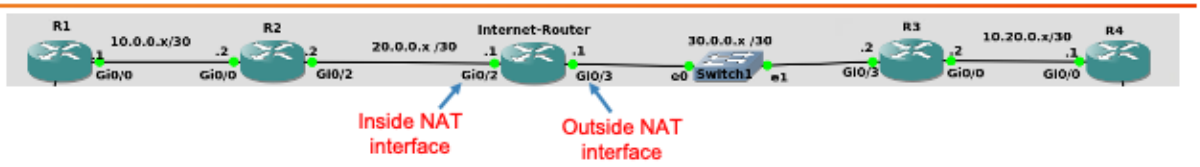


Inside Local	Outside Global	Original Source Port	Inside Global	New Source Port
10.0.0.1	2.2.2.2	34001	75.1.1.1	34001
10.0.0.2	2.2.2.2	34556	75.1.1.1	34556
10.0.0.3	2.2.2.2	34001	75.1.1.1	34002
10.0.0.4	2.2.2.2	37996	75.1.1.1	37996

What Problems Must Be Solved?

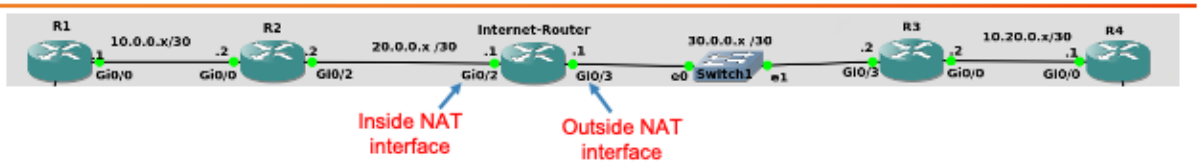
- Two challenges exist when an IPsec VPN tunnel must pass through a NAT router:
  - Creation of NAT translation entries
  - Preserving data integrity checks
- Both ESP and AH are Layer-3 protocols
  - Each has its own IP protocol value
  - Both the ESP and AH headers are placed immediately after the IPv4/IPv6 header
  - Neither ESP nor AH uses TCP or UDP, so no Layer-4 port numbers exist
  - How can a PAT router create a translation if no Layer-4 port number exists?
- Some (not all) IPsec protocols include the IPv4/IPv6 header in the SHA/MD5 data integrity checks
  - How is data integrity preserved if the IPv4/IPv6 header is modified by NAT?

NAT Translation Solutions for IPsec



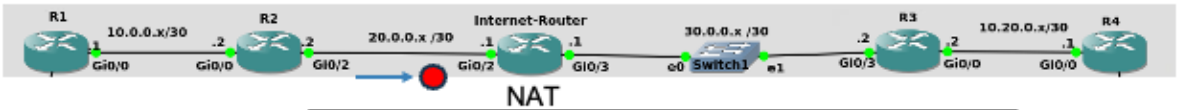
- To allow IPsec packets to successfully pass through a NAT/PAT router two solutions exist:
  - IPsec Pass-Through
  - NAT Traversal (NAT-T)
- IPsec Pass-Through
  - PAT router builds NAT entry using SPI (Security Parameter Index) value found in AH or ESP IPsec header.
  - Typically, only works on ESP-encapsulated packets
  - Not as popular as NAT-T and rarely implemented

NAT-Traversal



- NAT-T is a feature implemented by VPN endpoints
- Used to detect (and adjust to) the presence of NAT devices in the path between IPsec VPN endpoints
- IPsec endpoints include NAT-T payloads within ISAKMP exchanges which contain a hash derived from IPv4 packet header
- If received HASH value doesn't compute;
  - Subsequent ISAKMP/IKE packets change their UDP port from 500 to 4500
  - A UDP header (source and destination ports 4500) is prepended to IPsec AH or ESP packets.

NAT-T Detection



NAT

```
▸ Internet Protocol Version 4, Src: 20.0.0.2, Dst: 30.0.0.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
    Total Length: 192
    Identification: 0x015c (348)
  ▸ Flags: 0x0000
    Fragment offset: 0
    Time to live: 255
    Protocol: UDP (17)
    Header checksum: 0x870d [validation disabled]
    [Header checksum status: Unverified]
    Source: 20.0.0.2
    Destination: 30.0.0.2
▸ User Datagram Protocol, Src Port: 500, Dst Port: 500
▾ Internet Security Association and Key Management Protocol
    Initiator SPI: 41edb4acf082c2a4
    Responder SPI: 0000000000000000
    Next payload: Security Association (1)
  ▸ Version: 1.0
    Exchange type: Identity Protection (Main Mode) (2)
  ▸ Flags: 0x00
    Message ID: 0x00000000
    Length: 164
  ▸ Payload: Security Association (1)
  ▾ Payload: Vendor ID (13) : RFC 3947 Negotiation of NAT-Traversal in the IKE
      Next payload: Vendor ID (13)
      Reserved: 00
      Payload length: 20
      Vendor ID: 4a131c81070358455c5728f20e95452f
      Vendor ID: RFC 3947 Negotiation of NAT-Traversal in the IKE
  ▸ Payload: Vendor ID (13) : draft-ietf-ipsec-nat-t-ike-07
  ▸ Payload: Vendor ID (13) : draft-ietf-ipsec-nat-t-ike-03
  ▸ Payload: Vendor ID (13) : draft-ietf-ipsec-nat-t-ike-02\n
```

NAT-T Hash Value derived from IPv4 Header

NAT-T Detection

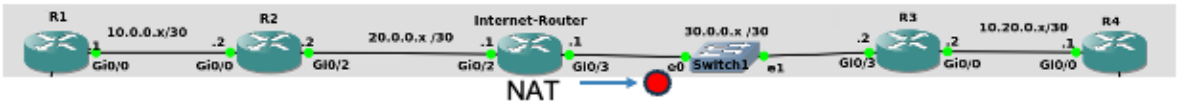
NAT

```
▾ Internet Protocol Version 4, Src: 30.0.0.1, Dst: 30.0.0.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
    Total Length: 192
    Identification: 0x011a (282)
  ▸ Flags: 0x0000
    Fragment offset: 0
    Time to live: 254
    Protocol: UDP (17)
    Header checksum: 0x7e50 [validation disabled]
    [Header checksum status: Unverified]
    Source: 30.0.0.1
    Destination: 30.0.0.2
▸ User Datagram Protocol, Src Port: 500, Dst Port: 500
▾ Internet Security Association and Key Management Protocol
    Initiator SPI: 41edb4ac902e18e1
    Responder SPI: 0000000000000000
    Next payload: Security Association (1)
  ▸ Version: 1.0
    Exchange type: Identity Protection (Main Mode) (2)
  ▸ Flags: 0x00
    Message ID: 0x00000000
    Length: 164
  ▸ Payload: Security Association (1)
  ▾ Payload: Vendor ID (13) : RFC 3947 Negotiation of NAT-Traversal in the IKE
      Next payload: Vendor ID (13)
      Reserved: 00
      Payload length: 20
      Vendor ID: 4a131c81070358455c5728f20e95452f
      Vendor ID: RFC 3947 Negotiation of NAT-Traversal in the IKE
  ▸ Payload: Vendor ID (13) : draft-ietf-ipsec-nat-t-ike-07
  ▸ Payload: Vendor ID (13) : draft-ietf-ipsec-nat-t-ike-03
```

NAT has changed source IP address which renders computed hash below invalid = Presence of NAT router

NAT-T Hash Value derived from IPv4 Header

NAT-T Effect on IPsec

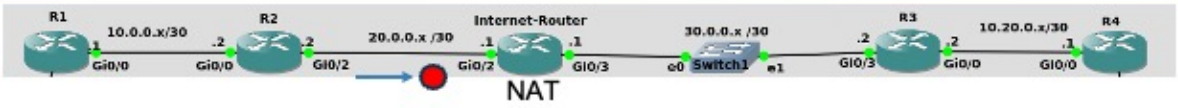


UDP header applied by NAT-T

```
Internet Protocol Version 4, Src: 30.0.0.1, Dst: 30.0.0.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
    Total Length: 160
    Identification: 0x0236 (566)
  ▸ Flags: 0x0000
    Fragment offset: 0
    Time to live: 254
    Protocol: UDP (17)
    Header checksum: 0x7d54 [validation disabled]
    [Header checksum status: Unverified]
    Source: 30.0.0.1
    Destination: 30.0.0.2
  ▸ User Datagram Protocol, Src Port: 4500, Dst Port: 4500
    UDP Encapsulation of IPsec Packets
  ▾ Encapsulating Security Payload
    ESP SPI: 0x6f1661d8 (1863737816)
    ESP Sequence: 161
```

- Notice how the UDP ports numbers have been changed (from 500 for ISAKMP) to 4500.

NAT Translation Entries with NAT-T



- Prior to NAT-T only ISAKMP/IKE packets could be translated because ISAKMP/IKE natively utilizes UDP

```
Internet-Router#sho ip nat trans
Pro Inside global      Inside local       Outside local      Outside global
udp 30.0.0.1:500       20.0.0.2:500       30.0.0.2:500       30.0.0.2:500
Internet-Router#
```

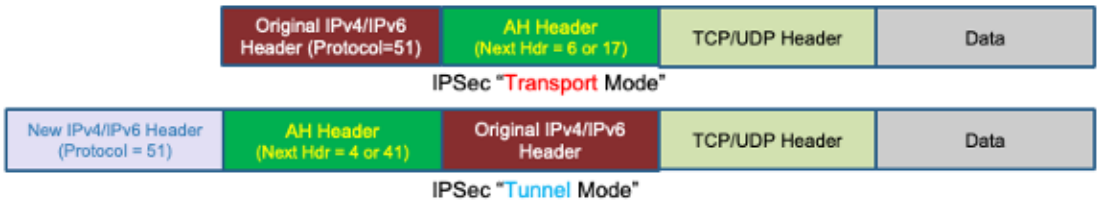
- After NAT-T applies UDP header and changes port to 4500, NAT translation entry can double for both IKE and IPsec traffic

This entry will timeout. →

```
Internet-Router#sho ip nat trans
Pro Inside global      Inside local       Outside local      Outside global
udp 30.0.0.1:500       20.0.0.2:500       30.0.0.2:500       30.0.0.2:500
udp 30.0.0.1:4500      20.0.0.2:4500      30.0.0.2:4500      30.0.0.2:4500
Internet-Router#
```

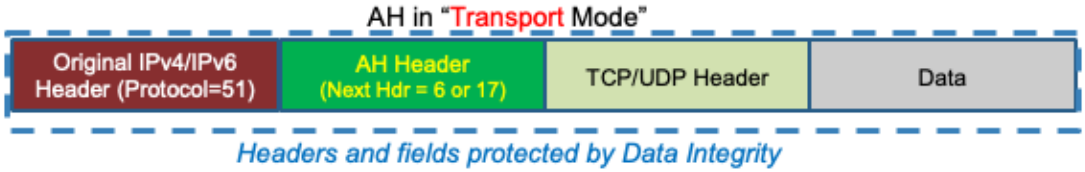
NAT & IPsec Data Integrity

- Both Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols of IPsec ensure data integrity by generating hash values on certain fields of the IP packet.
- If the source IP header is changed by NAT, how does that affect IPsec integrity checks?



IPsec & NAT Restrictions

- IPsec protocols that include the *outer IP header* into integrity checks *cannot be used with NAT*.



  - IPsec Authentication Header *in Transport Mode*, validates various fields from all (Layer-3 and above) headers as well as data with integrity checks.
  - This mode of IPsec is the ONLY mode that is incompatible with NAT
- The outer IP header is NOT included in IPsec integrity checks when used with:
  - AH in Tunnel Mode
  - ESP in both Transport and Tunnel Modes

- In practice, NAT-T is typically only supported when using ESP.

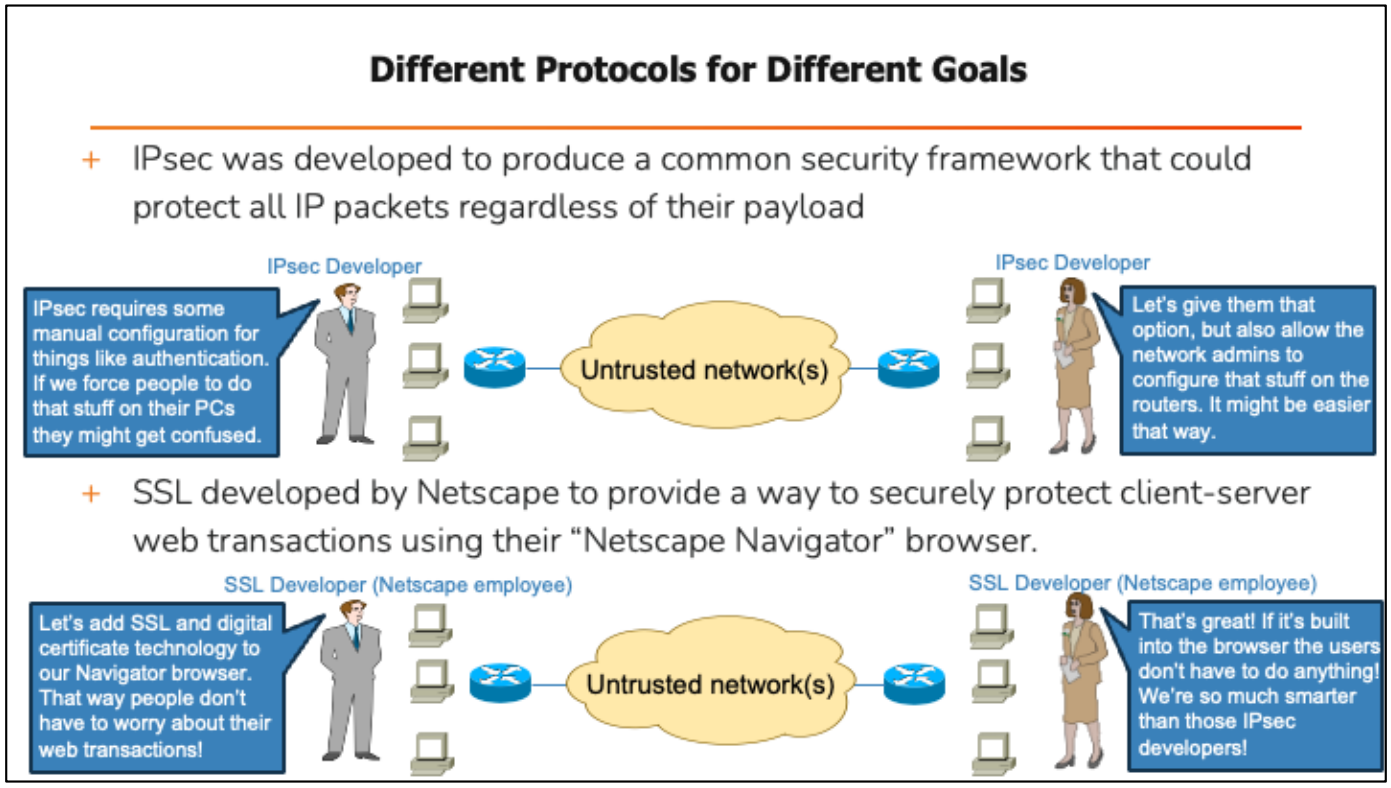
Thank you for watching!

Topic Overview

+ SSL Overview & Historical Timeline
+ Objective Differences Between IPsec and SSL
+ Introduction to SSL/TSL Tunnel Formation

Concurrent Security Timelines

- Both SSL and IPsec were developed around the same time, but by different organizations and for different reasons
  - *IPsec* developed by *IETF* in mid 1990s
    - First document (RFC 1825) released in 1995
  - *SSL* developed by *Netscape* independent of the RFC process
    - SSL v1.0 produced in 1994 but inherent security flaws prompted an update (SSL v2.0) in 1995



- While the initial specifications for IPsec could apply to both remote access VPNs (initiated from PCs and Laptops) as well as site-to-site VPNs (between two network infrastructure devices like firewalls or routers) early adopters found remote-access VPNs difficult to implement with IPsec. So initially, IPsec was primarily used for site-to-site VPNs.

What are SSL & TLS?

- Stands for Secure Socket Layer
  - Operates at the OSI Transport Layer, where the socket is about to be opened
  - Utilizes TCP (Transmission Control Protocol)
- SSL was created by Netscape Communications in 1994 to protect/secure web transactions
  - Last update to SSL was version 3.0 in 1996
  - Officially deprecated in June 2015 by the Internet Engineering Task Force (IETF) in RFC 7568 due to several inherent vulnerabilities
- IETF enhanced and fixed flaws in 1999, named it TLS (Transport Layer Security)
  - TLS v1.0 defined in RFC 2246 and TLS v1.2 defined in RFC 5246
  - TLS v1.3 (*current version*) was finalized by IETF in 2018

- Some might question why TLS wasn't simply called SSL v4.0 if was simply another upgrade to SSL. However, the name change from SSL to TLS reflects two important differences:
  - SSL was proprietary to Netscape. The IETF (a consortium of people from different companies, including some individuals from Netscape) created TLS.
  - TLS used SSL as a foundation but provided a major overhaul in several areas of the protocol such as MAC (Message Authentication Code) construction, improved flexibility and security in the negotiation of cipher suites, expanded the list of alert codes, and much more.

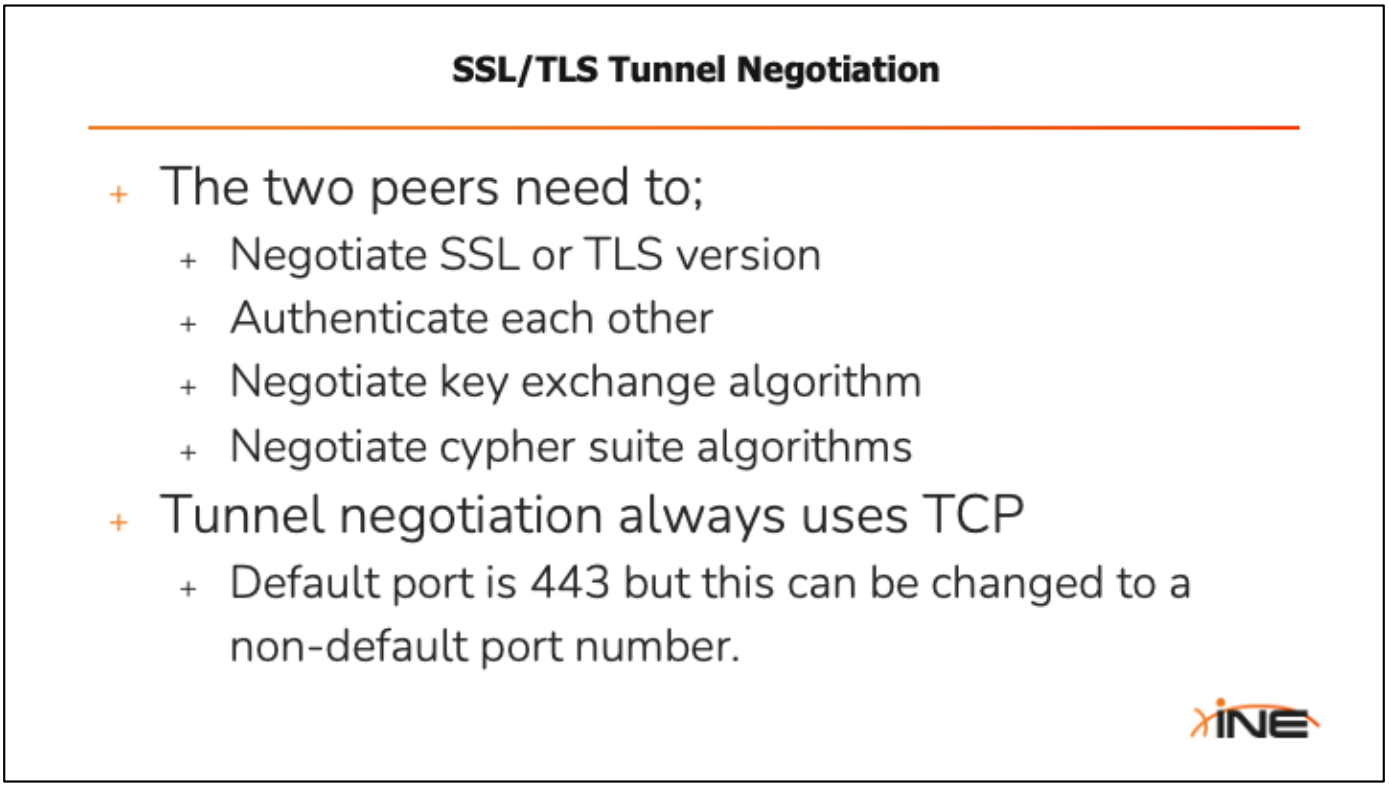
Why Use SSL Instead of IPsec?

- When used for Remote-Access VPNs, IPsec requires specialized software (IPsec "client") running on end-user devices...not every device has this.
- SSL/TLS provides many of the same benefits of IPsec
  - Authentication
  - Confidentiality
  - Integrity
  - Anti-reply
- SSL/TLS built-into most web browsers
- SSL/TLS browsing sessions identified by "http**S**"

- SSL/TLS can be used to secure other Application Layer protocols, not only HTTP
  - SMTPS – TCP port 465
  - IMAPS – TCP port 993
  - POP3S – TCP port 995
  - FTPS – TCP port 990
  - LDAPS – TCP port 636

How does SSL/TLS work ?

- SSL/TLS session works in two phases, like IPsec
  - Negotiate the SSL/TLS tunnel (control-plane)
  - Transmit data over the SSL/TLS tunnel (data-plane)
- Both phases make use of the SSL/TLS *Record* Layer which encapsulates multiple SSL/TLS protocol transactions.
- SSL and TLS are two distinct protocols
  - There are differences in the tunnel negotiation process
  - TLS supports stronger security



- When utilizing Cisco's clientless SSL VPN service your browser would connect to the Cisco firewall on TCP port 4443.

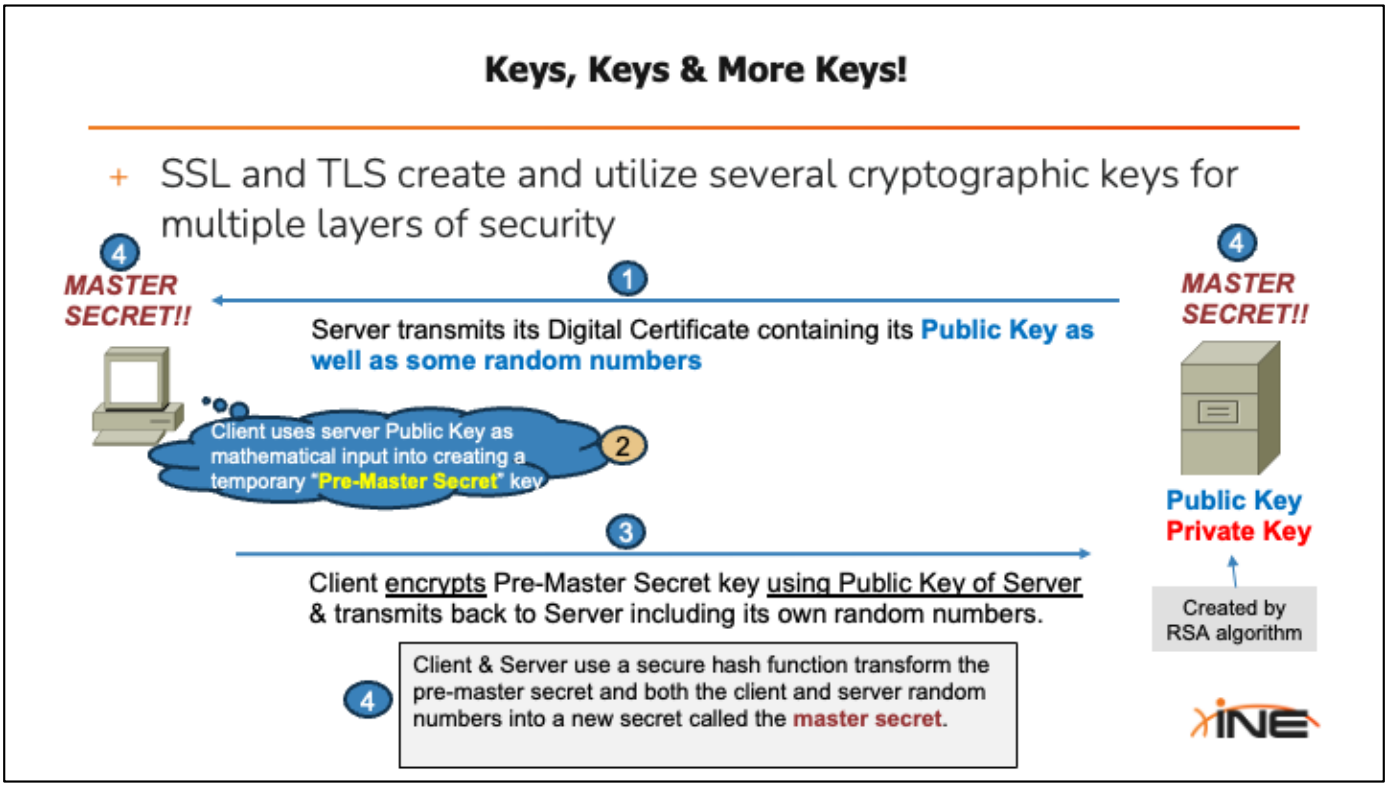
Thank you for watching!

Topic Overview

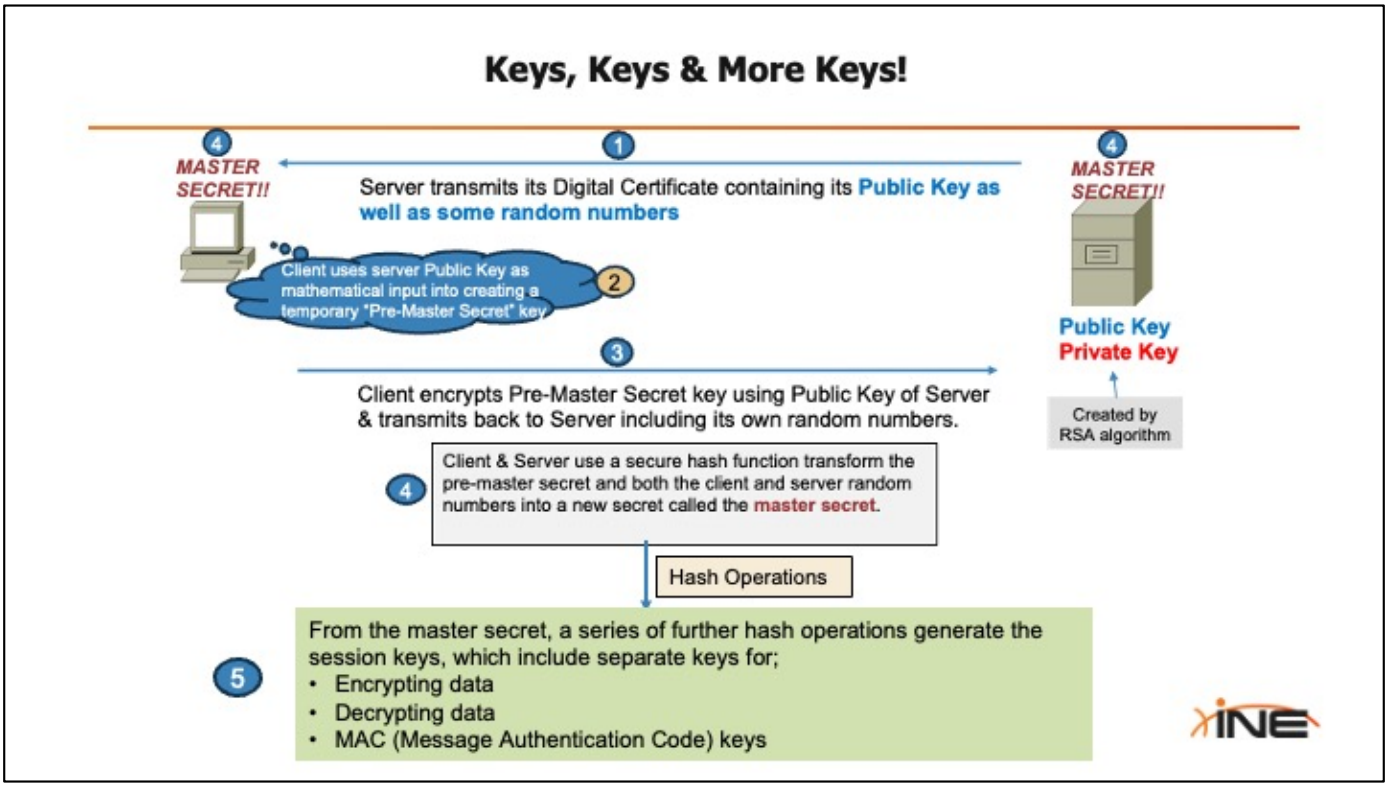
+ SSL/TLS Key Types & Generation

SSL/TLS Cryptography

- Authentication primarily utilizes the exchange of Digital Certificates
  - For usage on IoT devices and other situations in which certificates are not practical, versions of SSL/TLS exist (such as wolfSSL) that implement PSK
- Cryptographic Keying material (for data encryption) is usually derived through asymmetric algorithms
  - RSA handshake uses RSA public/private key for key establishment and authentication verification
  - DH handshake uses DH for key establishment and RSA for authentication



- As an alternative, SSL and TLS can make use of Diffie-Hellman to generate the Pre-Master Secret Key". In this case, right after the server transmits its certificate it would also send Diffie-Hellman information so that both it and the Client can jointly create a pre-master secret key.
- If you're familiar with the steps involved with IKE(IPsec) Phase-1, Step-3 in this diagram is similar to the point in IKE Phase-1 in which a secure, encrypted tunnel is built so that further negotiations can (securely) take place to derive other keys and encryption algorithms for finally encrypting user data.



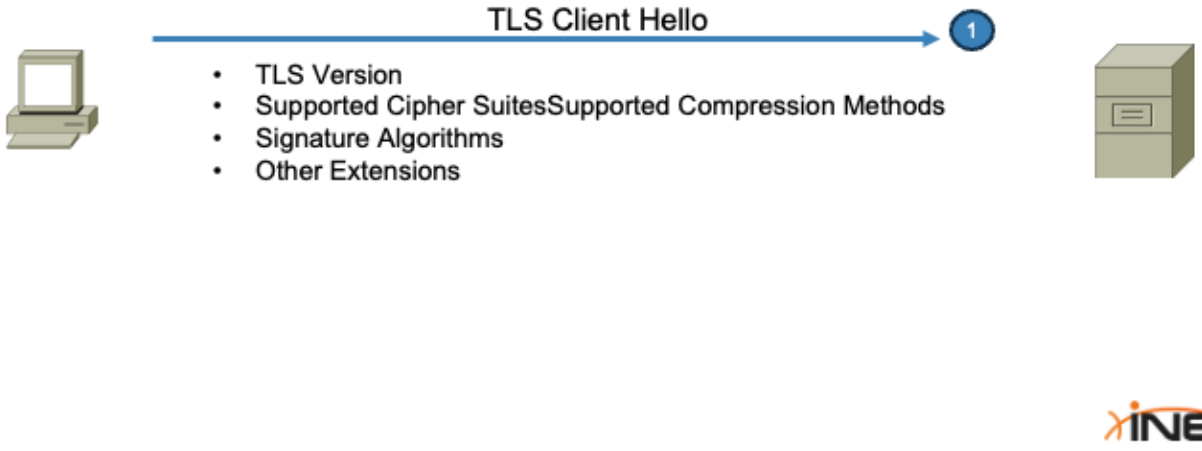
- Both the Server and Client perform steps-4 and 5.
- With the exception of the original Public-Private keypair on the Server, all of these other keys that are generated are only temporary and used only for the duration of the HTTPS session. Once the session is closed these keys are deleted and new keys will need to be derived for future HTTPS sessions.
- This is what makes SSL/TLS so secure.

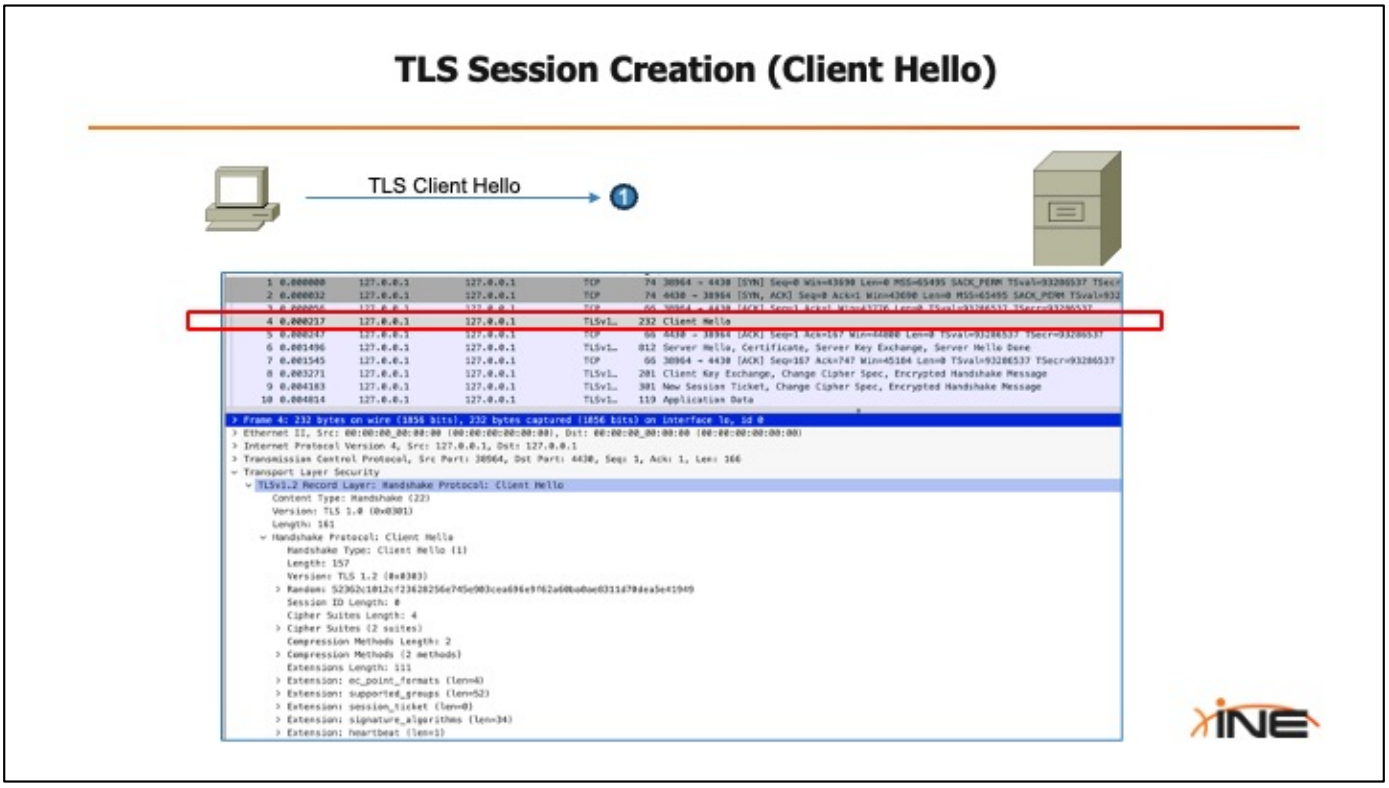
Thank you for watching!

Topic Overview

+ TLS Message Exchanges & Packet Captures

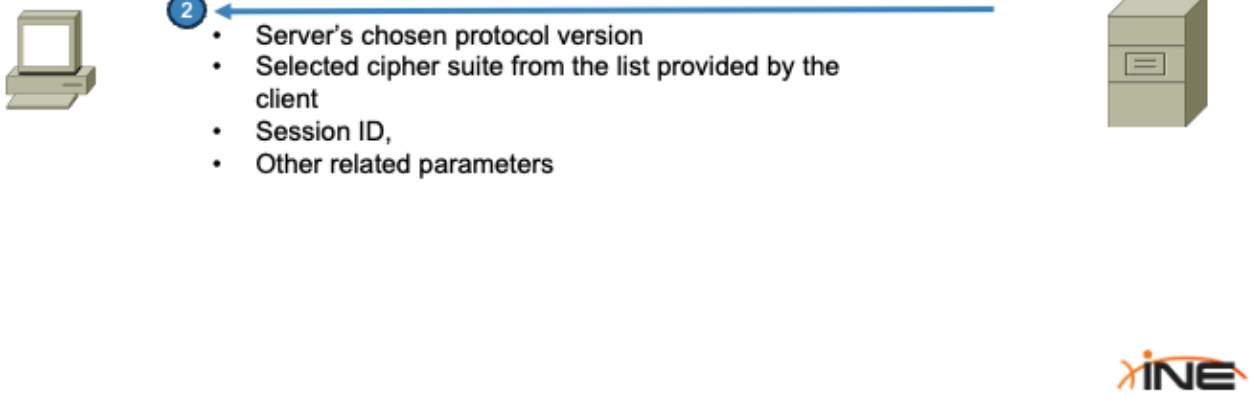
TLS Message Exchanges & Names





- SSL/TLS typically used a reserved port number of TCP 443. In these captures you'll notice that the TCP port is NOT the standard, reserved port. This is because applications using SSL/TLS have the option of being configured to listen to a non-standard port number if the developer of that application wishes to do so.
- Please note; Several websites exist that provide instructions on how to log your SSL/TLS session keys to a file, and then use the keys in that file to decrypt your traffic for viewing in Wireshark. There are a few major gotchas to that however;
    - The processes described in those documents will only work for websites that utilize SSL or TLS v1.0
    - The developers of TLS saw the process of storing session keys (for Wireshark captures) as a vulnerability and intentionally made changes to TLS 1.2 and 1.3 rendering this process impossible
    - Wireshark can only decrypt RSA ciphers, if it's one of the ECDHE ciphers, there's no way to decrypt it.

TLS Message Exchanges & Names



TLS Message Exchanges & Names

3 TLS Server Certificate

- Digital Certificate signed by Trusted Certificate Authority
- Allows TLS client to authenticate TLS server

4 TLS Server Key Exchange

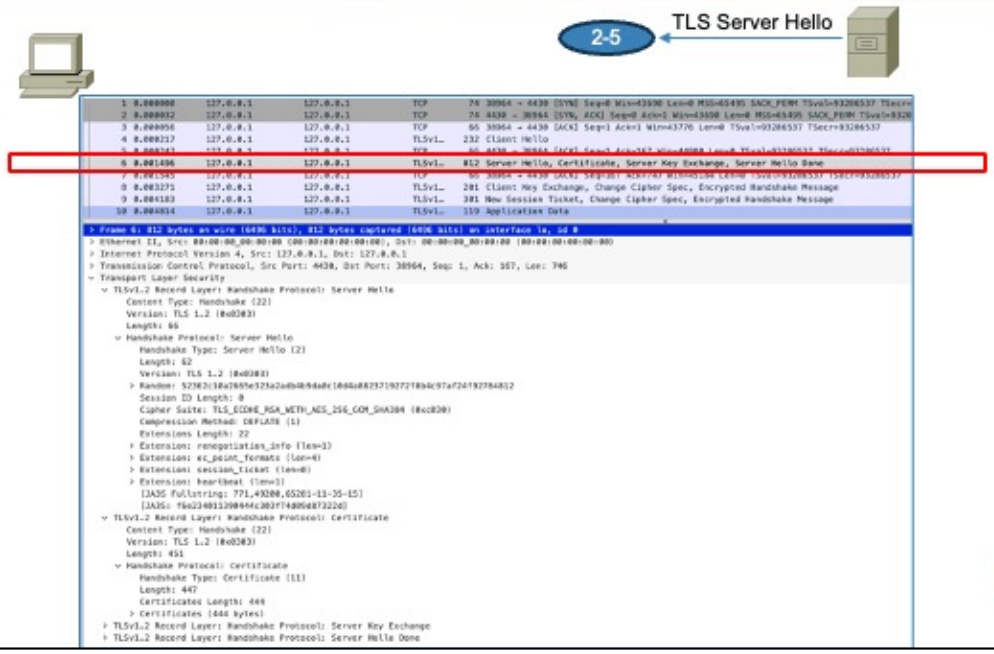
- (Optional)
- Sent if agreed-upon key exchange method requires additional data (e.g., Diffie-Hellman parameters)

5 TLS Server Hello Done

- Indicates that the server has finished sending its part of the negotiation parameters and is now awaiting the client's response.

TLS allows all these messages (as well as the preceding "Server Hello") to be bundled within a single TCP segment.

TLS Session Creation (Server Hello)



TLS Message Exchanges & Names

TLS Client Key Exchange 6

- Contains the pre-master secret encrypted with the public key of the server (if RSA is used)
- May contain other key exchange data depending on the selected cipher suite

TLS Change Cipher Spec 7

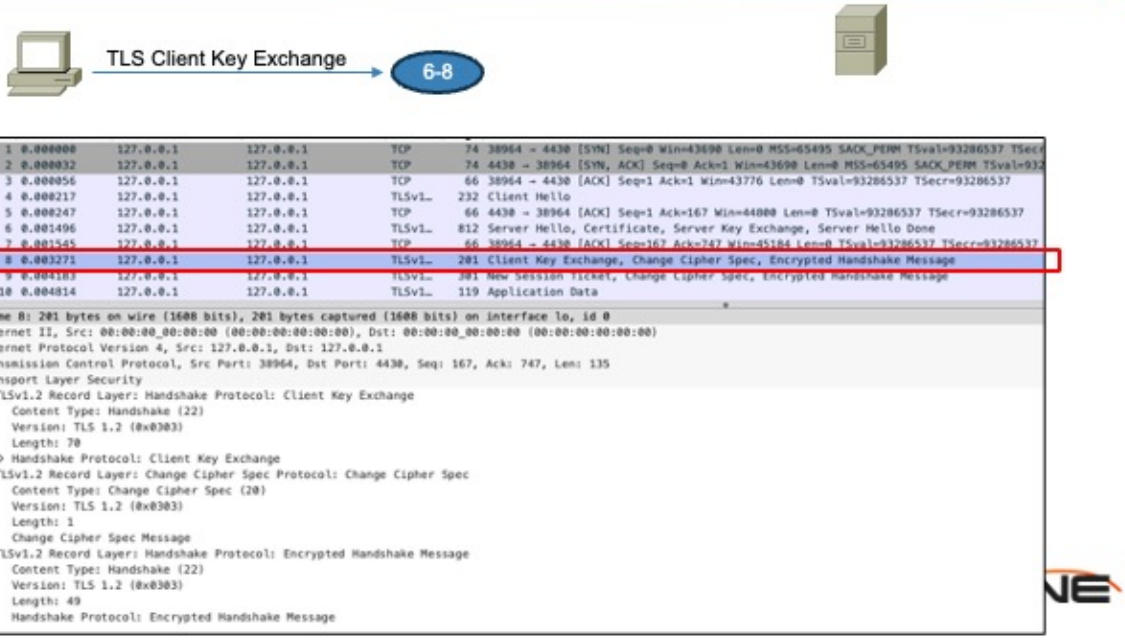
- Signals that subsequent records will be protected under the newly negotiated Cipher Spec and keys

TLS Client Finished 8

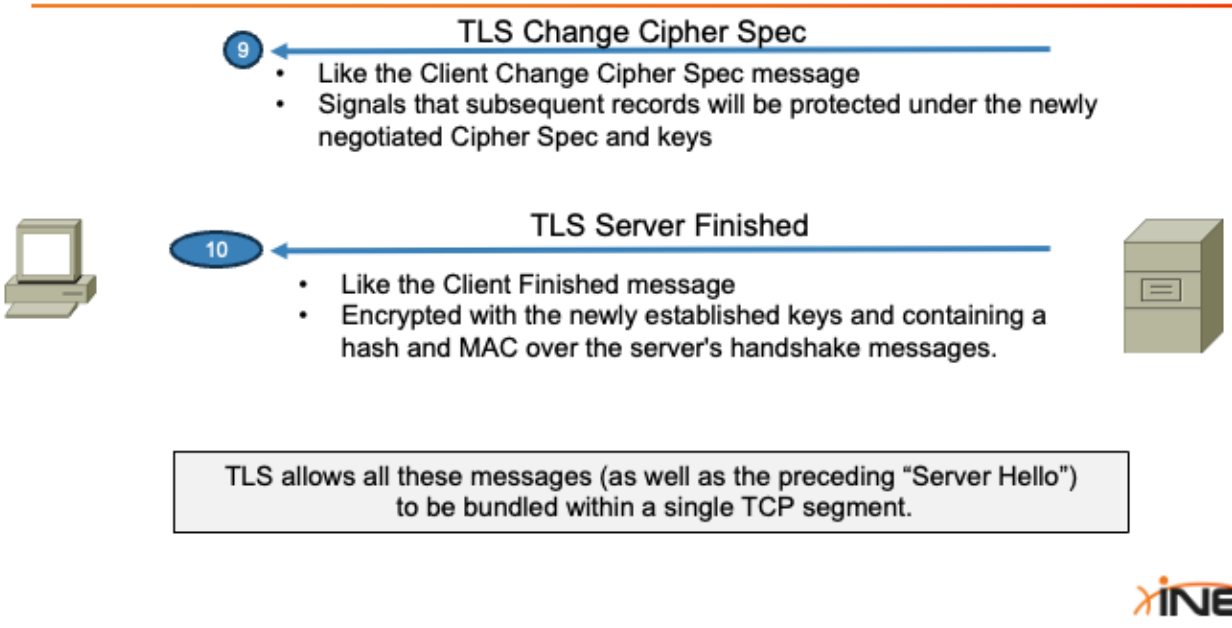
- Encrypted message
- Contains a hash and MAC over the previous handshake messages.
- Used to verify that the key exchange and authentication processes were successful.

TLS allows all these messages to be bundled within a single TCP segment.

TLS Session Creation (Client Key Exchange)



TLS Message Exchanges & Names



TLS allows all these messages (as well as the preceding "Server Hello") to be bundled within a single TCP segment.

TLS Session Creation (Server Session Ticket)

9-10 TLS Change Cipher Spec

```
    1 0.000000    127.0.0.1     127.0.0.1     TCP      74 38964 → 4430 [SYN] Seq=0 Win=43690 Len=0 MSS=65495 SACK_PERM TSval=93286537 TSecr=0
    2 0.000032    127.0.0.1     127.0.0.1     TCP      74 4430 → 38964 [SYN, ACK] Seq=0 Ack=1 Win=43690 Len=0 MSS=65495 SACK_PERM TSval=93286
    3 0.000056    127.0.0.1     127.0.0.1     TCP      66 38964 → 4430 [ACK] Seq=1 Ack=1 Win=43776 Len=0 TSval=93286537 TSecr=93286537
    4 0.000217    127.0.0.1     127.0.0.1     TLSv1…  232 Client Hello
    5 0.000247    127.0.0.1     127.0.0.1     TCP      66 4430 → 38964 [ACK] Seq=1 Ack=167 Win=44800 Len=0 TSval=93286537 TSecr=93286537
    6 0.001496    127.0.0.1     127.0.0.1     TLSv1…  812 Server Hello, Certificate, Server Key Exchange, Server Hello Done
    7 0.001545    127.0.0.1     127.0.0.1     TCP      66 38964 → 4430 [ACK] Seq=167 Ack=747 Win=45184 Len=0 TSval=93286537 TSecr=93286537
    8 0.003271    127.0.0.1     127.0.0.1     TLSv1…  201 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
    9 0.004183    127.0.0.1     127.0.0.1     TLSv1…  301 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
   10 0.004814    127.0.0.1     127.0.0.1     TLSv1…  119 Application Data

> Frame 9: 301 bytes on wire (2408 bits), 301 bytes captured (2408 bits) on interface lo, id 0
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 4430, Dst Port: 38964, Seq: 747, Ack: 302, Len: 235
v Transport Layer Security
  v TLSv1.2 Record Layer: Handshake Protocol: New Session Ticket
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 170
    > Handshake Protocol: New Session Ticket
  v TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
      Content Type: Change Cipher Spec (20)
      Version: TLS 1.2 (0x0303)
      Length: 1
      Change Cipher Spec Message
  v TLSv1.2 Record Layer: Handshake Protocol: Encrypted Handshake Message
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 49
      Handshake Protocol: Encrypted Handshake Message
```

SSL/TLS Data-Plane

- Once the control-plane tunnel has been successfully created all data will be protected as negotiated over the data-plane tunnel
  - Encryption algorithms (RC4,DES,3DES,AES)
  - Hashing algorithms (MD5,SHA-1,SHA-2)
- Data-plane can use TCP or UDP
  - UDP encapsulation uses DTLS
  - DTLS is only supported by TLS
  - Latest version of DTLS is 1.3, defined in RFC 9147

- Pretty much all current websites these days utilize TLS v1.3
- Web browsers often allow you to change their settings such that your browser could disallow TLS 1.3 and only allow earlier versions, but when you do this virtually all websites will then fail to display.

Thank you for watching!

Topic Overview

+ Types of SSL/TLS VPNs

Why Do We Need SSL/TLS VPNs?

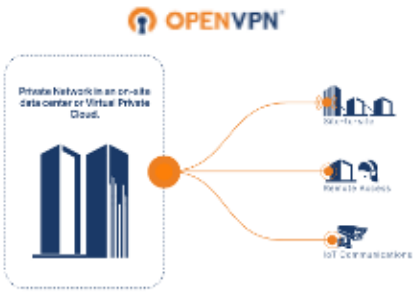
- A VPN that utilizes SSL/TLS technology to enforce CIA (Confidentiality, Integrity and Authentication).
- Why was it invented ?
  - In some countries, Internet access is restricted to HTTP/HTTPS
    - IKE is filtered, thus IPsec VPN will not work
  - Remote-access VPNs may be needed but without the need to have a client VPN application installed
    - Like when working from an Internet Café or Airport
- Has two variations
  - Client-based (Cisco AnyConnect)
  - Clientless (browser based)

- In some countries, certain VPN technologies, particularly those using IPsec (which relies heavily on IKE for establishing secure tunnels), are restricted or regulated. These restrictions are often due to government concerns about encryption technologies that might prevent law enforcement from monitoring communications for security reasons. Governments may restrict or control the use of strong encryption technologies to ensure they have the capability to intercept and decrypt data if legally warranted.

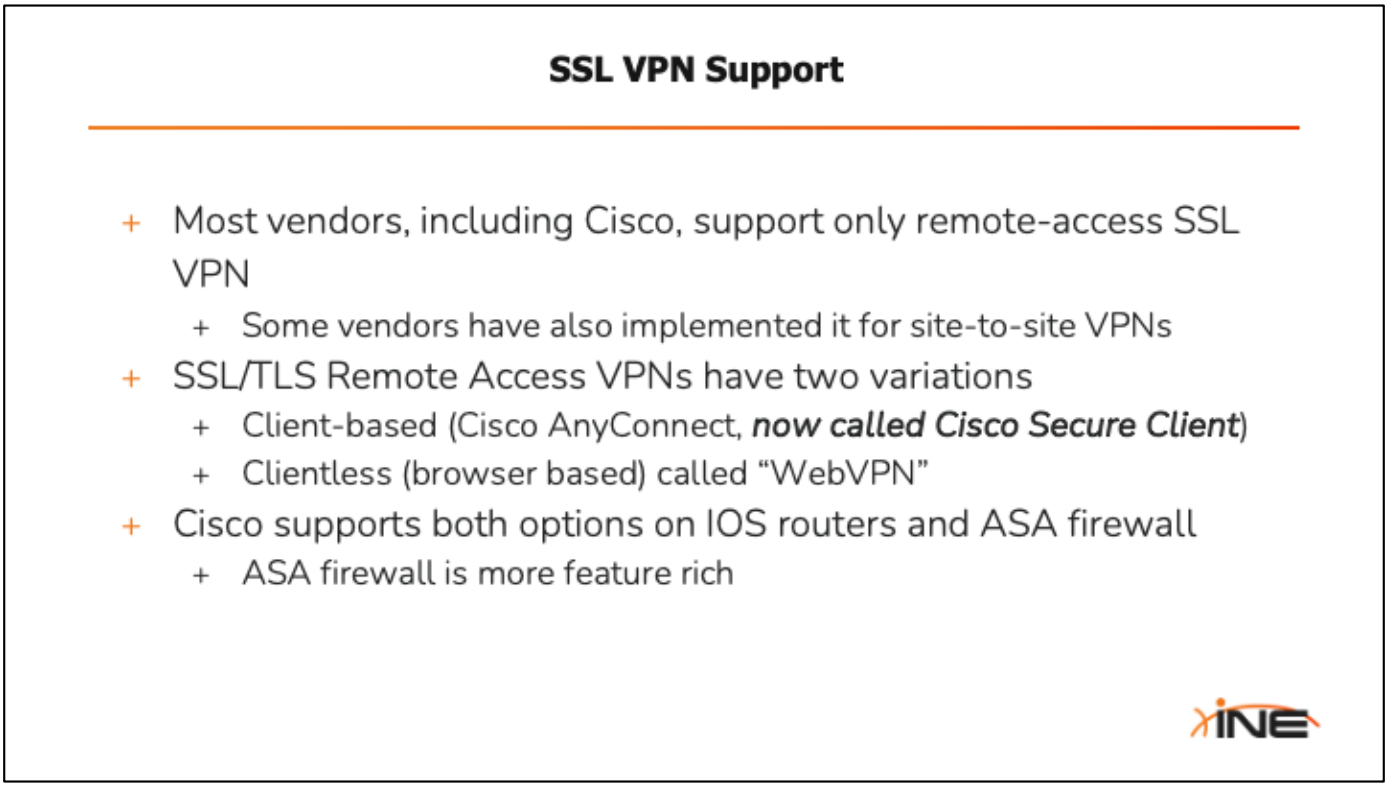
- China is a prominent example of a country with such restrictions. The Chinese government regulates the use of cryptographic services and technologies, including VPNs. They require that all VPN services be authorized and licensed by the government, and unauthorized VPN services are often blocked, particularly if they use strong encryption that does not allow for government oversight.

What is SSL VPN ?

- Control-plane negotiation happens through SSL/TLS
  - Which runs over TCP 443 (can be changed)
- Data-plane traffic being sent through the secure tunnel
  - TCP adds too much overhead and inconvenience for real-time applications
  - Thus data-plane will make use of DTLS, runs over UDP 443
  - Latest version of DTLS is 1.3, defined in RFC 9147



OpenVPN Access Server: An example of an open-source SSL/TLS/DTLS VPN solution.



- Although the commands to configure the WebVPN feature on Cisco IOS routers still exist, Cisco stopped supporting this feature several years ago. So if you need to implement Clientless SSL VPN (ie. WebVPN) it's best to do so on a Cisco ASA or Firepower firewall device.
- When utilizing the Cisco Secure Client (aka Cisco AnyConnect client) a remote-access VPN established between the VPN Client and VPN Server (Cisco ASA or Firepower firewall) will actually create two VPN tunnels, one using TLS and another using DTLS
    - TLS is used for secure web browsing (and any other TCP-based applications) over TCP, providing a reliable and secure transport.
    - DTLS is used for VoIP over UDP (and any other UDP-based applications), allowing for secure communication without significantly impacting latency and call quality.

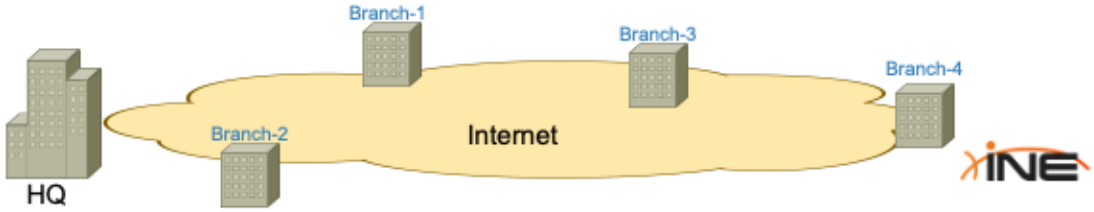
Thank you for watching!

Topic Overview

- The Problem To Be Solved
- Overview of DMVPN
- Protocols used in DMVPN
  - GRE
  - NHRP
- Operation of DMVPN
- Benefits of DMVPN

The Problem to be Solved

+ A company consists of a headquarters location and several branch offices.
+ Branch offices need to directly communicate to exchange data in a secure manner.
+ The company is growing with new branch offices expected at an unpredictable rate.
+ Configuring a full-mesh of individual, point-to-point IPsec tunnels is not scalable and is prone to configuration mistakes.



- DMVPN can indeed be an excellent choice for various types of real-time interactions between branch offices that require efficient, secure, and direct communication paths. There are several scenarios where DMVPN shines:
    - Real-Time Data Replication: For businesses that require real-time synchronization of databases across different geographical locations, DMVPN can facilitate the necessary speedy data exchanges. This is particularly crucial for industries like finance or retail, where up-to-date information on transactions, stock levels, or customer data needs to be consistent across multiple sites.
    - Disaster Recovery and Business Continuity: In scenarios where businesses must ensure high availability and resilience, DMVPN allows for real-time backup and disaster recovery operations between sites. If one site goes down, others can take over without significant downtime, ensuring continuous business operations.
    - Collaborative Computing: For projects involving real-time collaboration on digital products or services, such as software development or digital media creation, DMVPN can enable seamless inter-office connectivity. This helps in ensuring that file transfers, live editing, and updates are conducted without noticeable delays, fostering better teamwork and productivity.

- Streaming of Operational Data: In industries like manufacturing or utilities, real-time data streaming from various sensors or operational devices is crucial for monitoring and controlling remote equipment. DMVPN can help establish direct and secure connections to facilitate this continuous flow of information, enabling timely decisions and responses to operational data.

- Emergency Services and Response Coordination: For organizations involved in emergency response and public safety, DMVPN can be vital for coordinating efforts across multiple locations in real-time, ensuring that communication remains uninterrupted and secure during critical operations.

Dynamic Multipoint VPN (DMVPN)

- Point-to-multipoint Layer 3 overlay VPN
  - Public Internet connections typically used as the underlay
- Builds a logical hub and spoke topology between sites
  - Direct spoke to spoke traffic is supported
- DMVPN uses a combination of...
  - Multipoint GRE Tunnels (mGRE)
  - Next Hop Resolution Protocol (NHRP)
  - IPsec Crypto Profiles
  - Routing

- Why is it called, “Dynamic”?
- In older, legacy VPN methods, one would need to statically configure each end of the VPN tunnel so that tunnel endpoints (i.e. customer routers or firewalls) could reach each other and protect traffic.
- This meant that if a single router was going to be a VPN endpoint for dozens (or hundreds) of tunnels a LOT of pre-configuration work would be necessary.
- With DMVPN being a hub-and-spoke solution, each spoke only needs pre-configuration for a single VPN tunnel (to the Hub). After that, spokes will dynamically learn of each other and can form dynamic tunnels to each other.

Why Use DMVPN

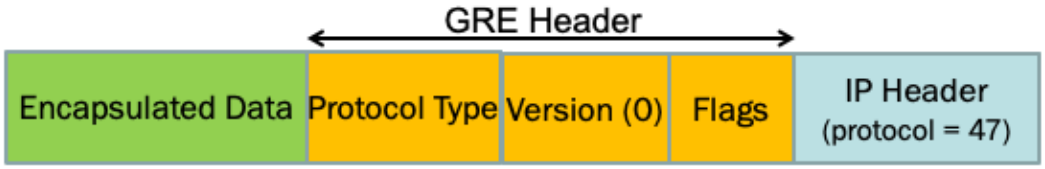
- Independent of SP access method
  - Only requirement is IP connectivity between edge routers at each site
- Routing policy is not dictated by SP
  - E.g. MPLS L3VPN restrictions
- Highly scalable
  - If properly designed

DMVPN Components

- Hub Router
  - Reachable via static, public IP address
- Spoke Routers
  - Reachable via static, or dynamic, public IP addresses
- Multipoint GRE tunnels
- IPsec (optional, but typically used)

What Is GRE?

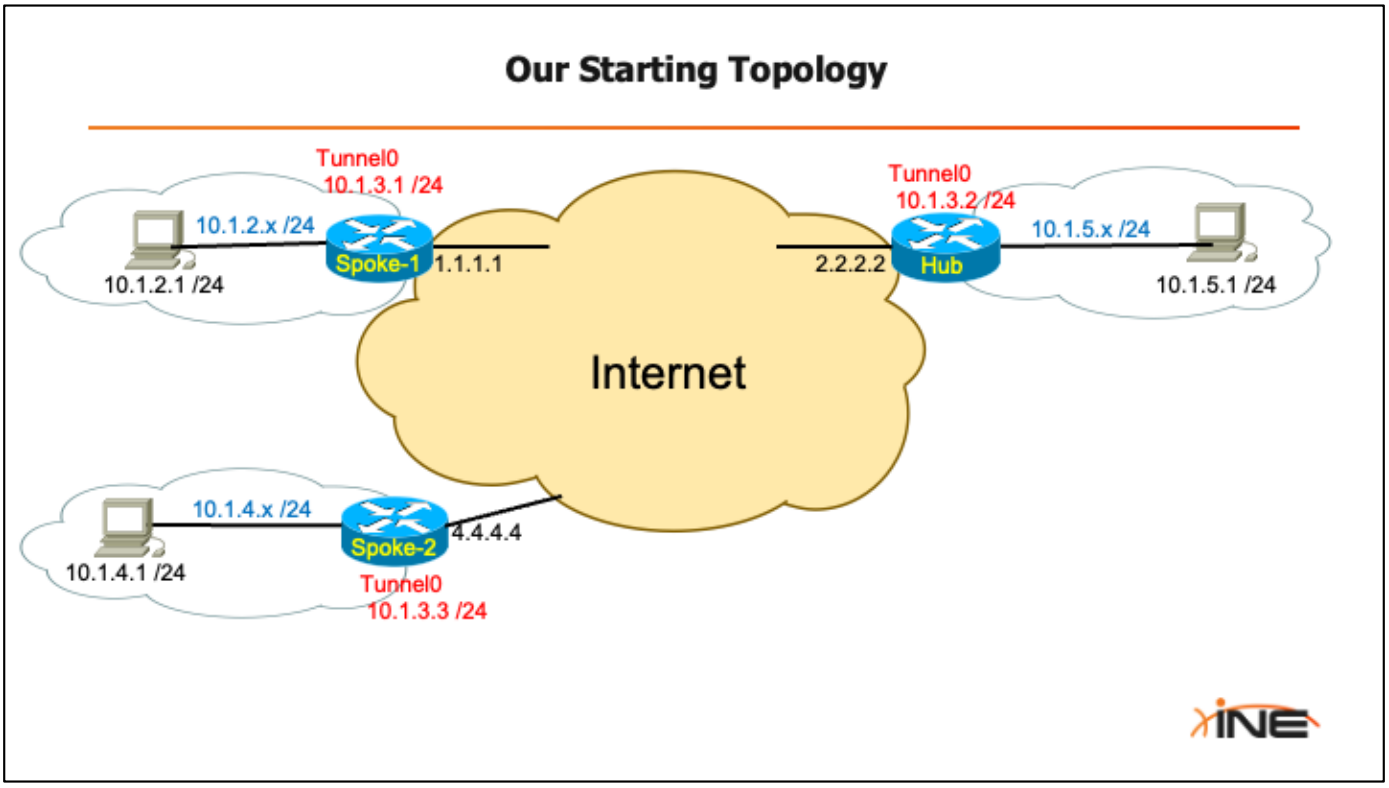
- GRE = Generic Route Encapsulation
- Defined in RFC 2784
- Tunneling mechanism and a critical component to DMVPN
- GRE can tunnel many different protocols inside a GRE Header.
- Default encapsulation for Cisco IOS Tunnel Interfaces



- GRE is a static, Point-to-Point tunneling method. It can be used to encapsulate many things, including IGP Routing Protocol traffic between two non-adjacent peers.
- Typically, we think of GRE having an outer IPv4 header…but technically GRE can use other protocols as its “delivery protocol”.
- GRE uses RFC 1700 “EtherType” values within its “Protocol Type” field.

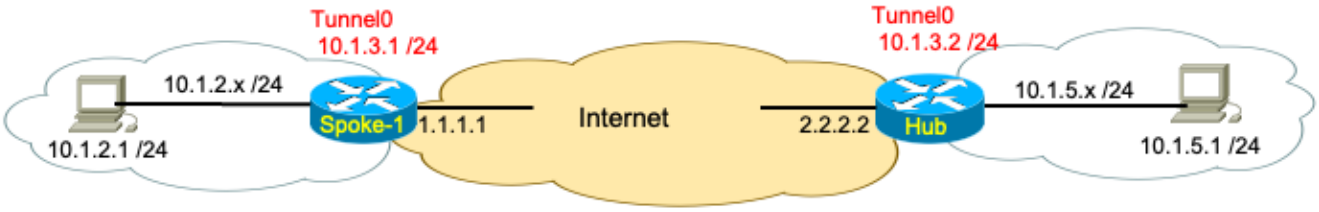
What Is NHRP?

- Next Hop Resolution Protocol
- Defined in RFC 2332
- Layer-2 Resolution Protocol and Cache
- Used in DMVPN to map a peer's tunnel IP address to that peer's public IP address.
- NHRP can populate the NHRP cache via static or dynamic entries (similar to ARP)



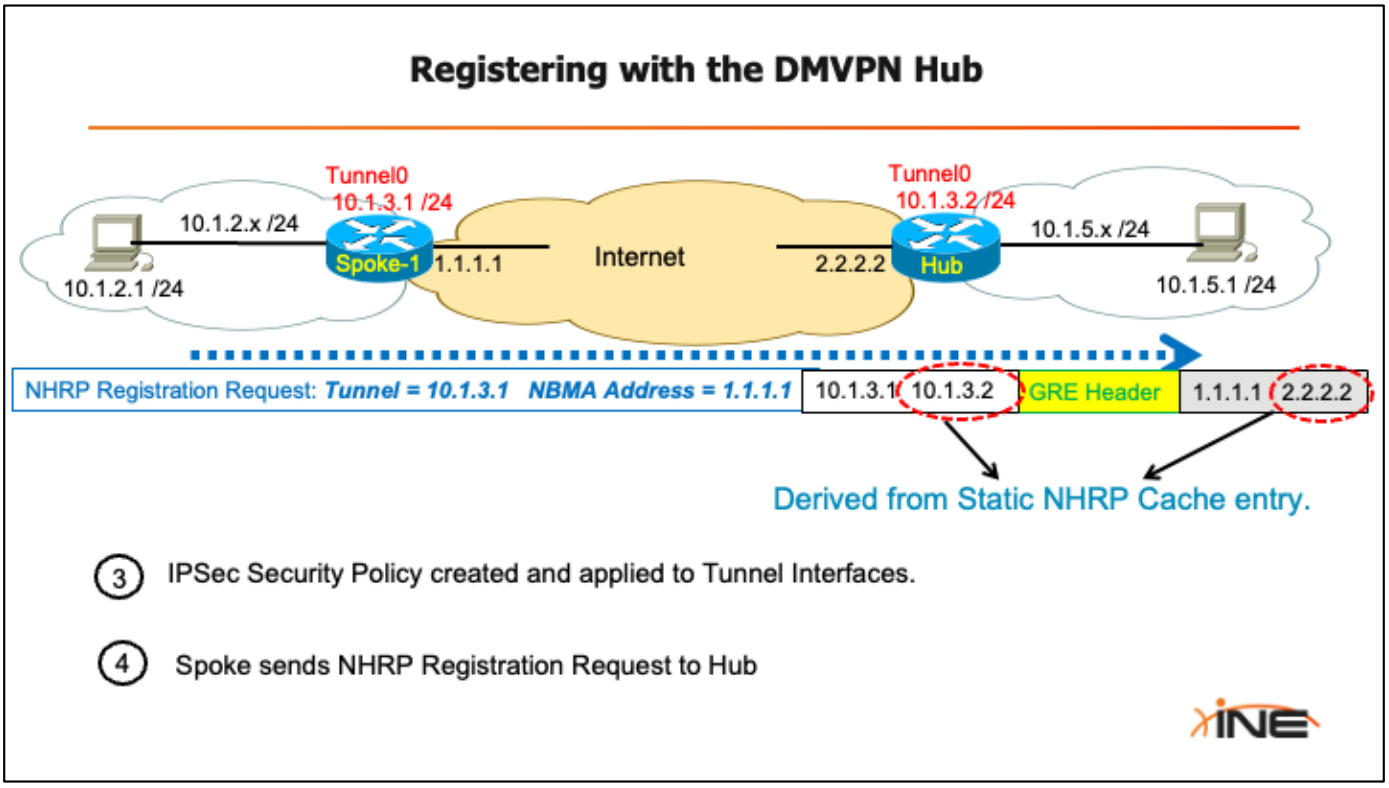
- In this topology, if we were to configure static IPsec tunnels to create a full-mesh we'd need N-1 tunnels configured on each device (with "N" being the total count of locations or sites).
- Considering that this small topology only has three sites (HQ and two "Spokes" or Branch Offices) that would require each site to have 3-1 = 2 IPsec tunnels created on every router.
- Imagine if we had a topology of 100-sites!!
- We're going to see that with DMVPN, we only need a SINGLE IPsec tunnel configuration applied to each site, which can be dynamically replicated as new sites come online.

DMVPN – Initial NHRP Setup



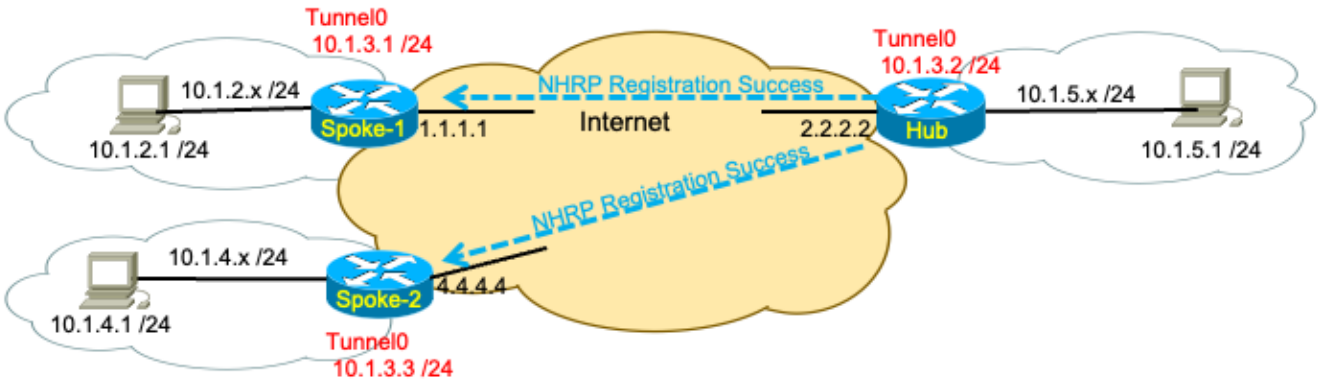
1. Tunnel interfaces created on each router including:
   - Tunnel Source (public IP address)
   - Tunnel Mode GRE Multipoint

2. Next Hop Server (Hub) statically defined in Spoke and static NHRP Cache entry created.
   - NHRP NHS = 10.1.3.2
   - NHRP Cache: 10.1.3.2 → 2.2.2.2 (static)

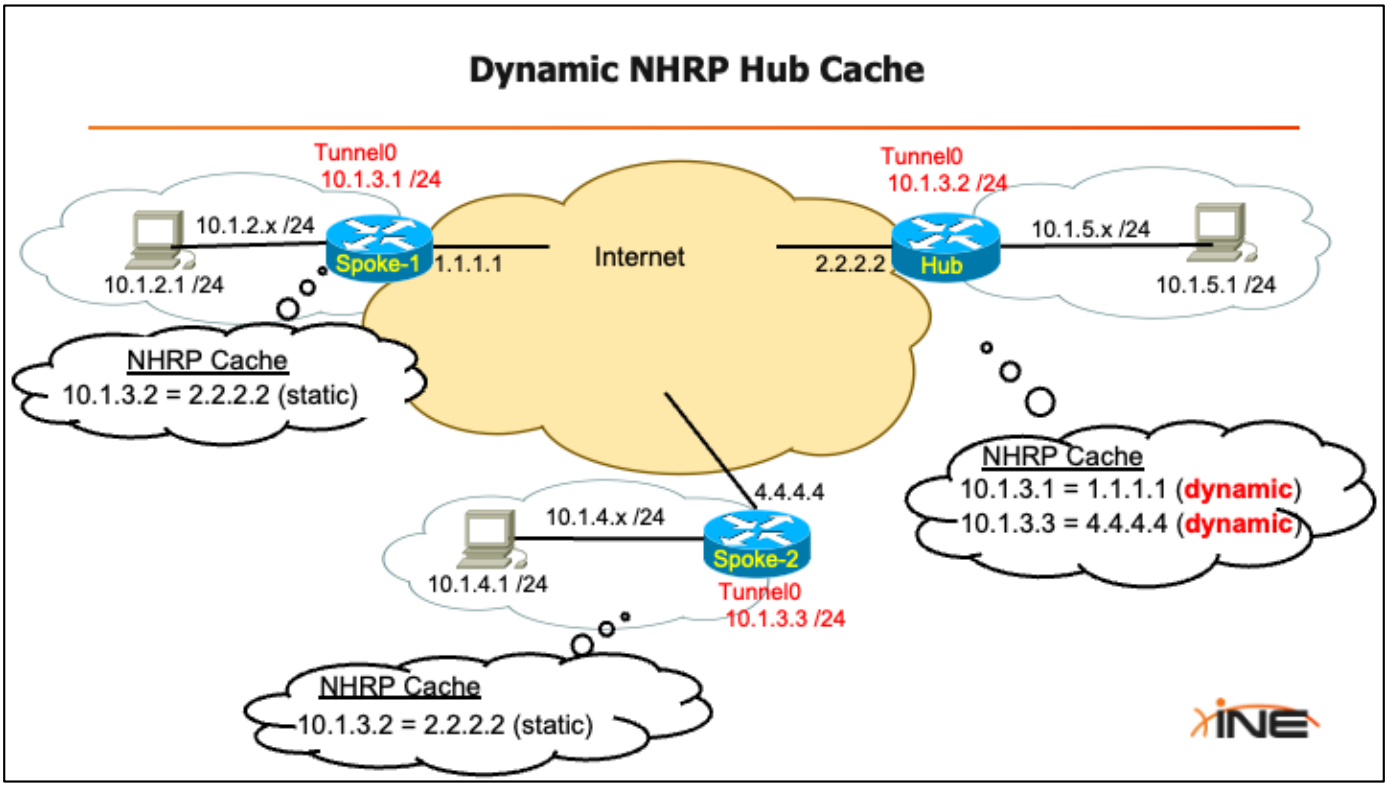


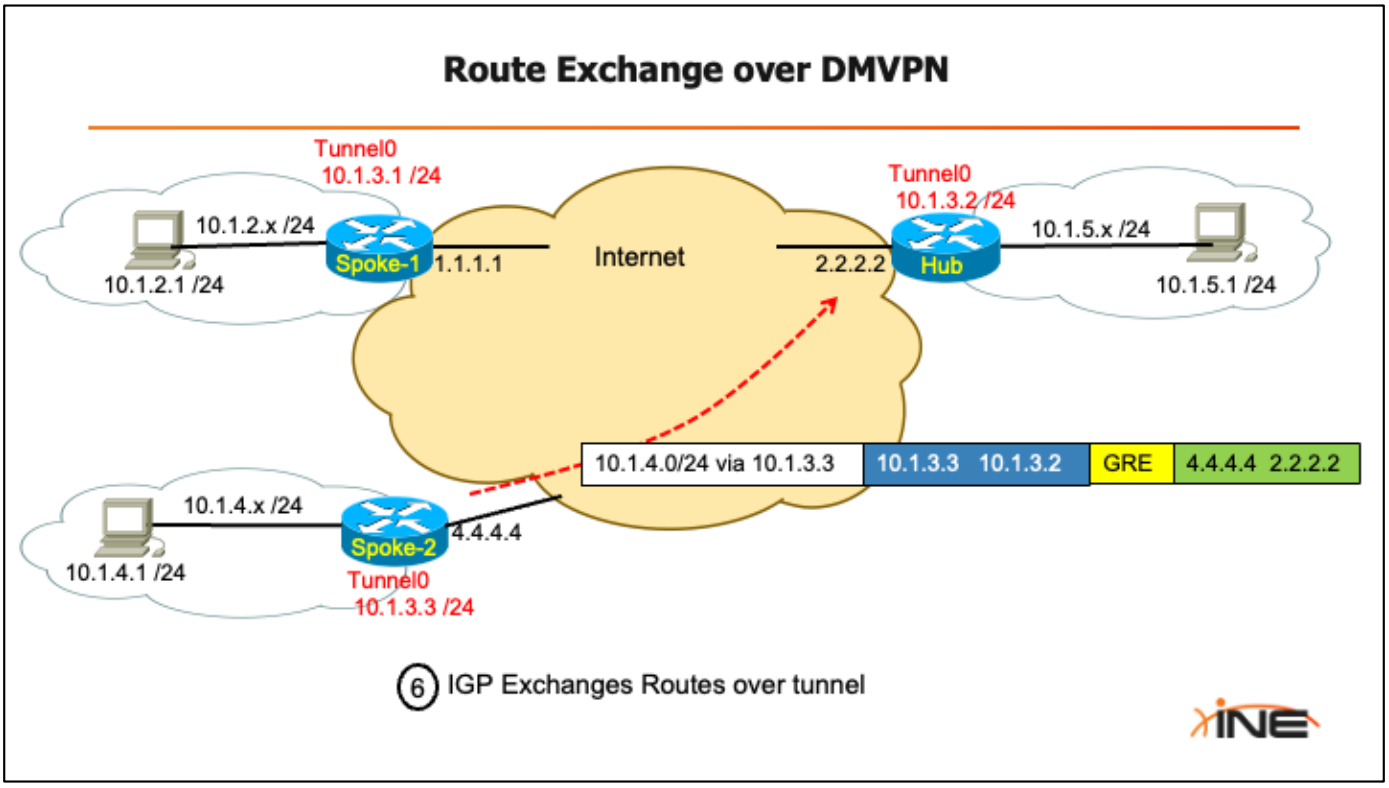
- At this point, all traffic that is sent from tunnel interface is encapsulated by GRE and secured by IPSec…even the NHRP protocol transactions.

Registration Completion

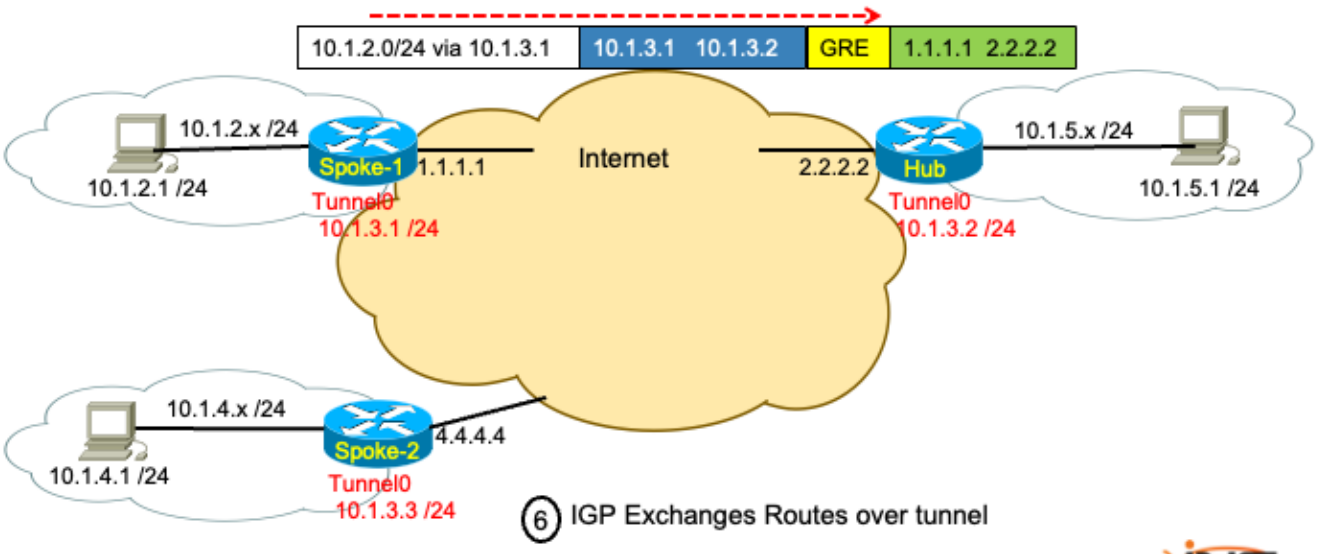


(5) Hub replies with NHRP Registration Reply

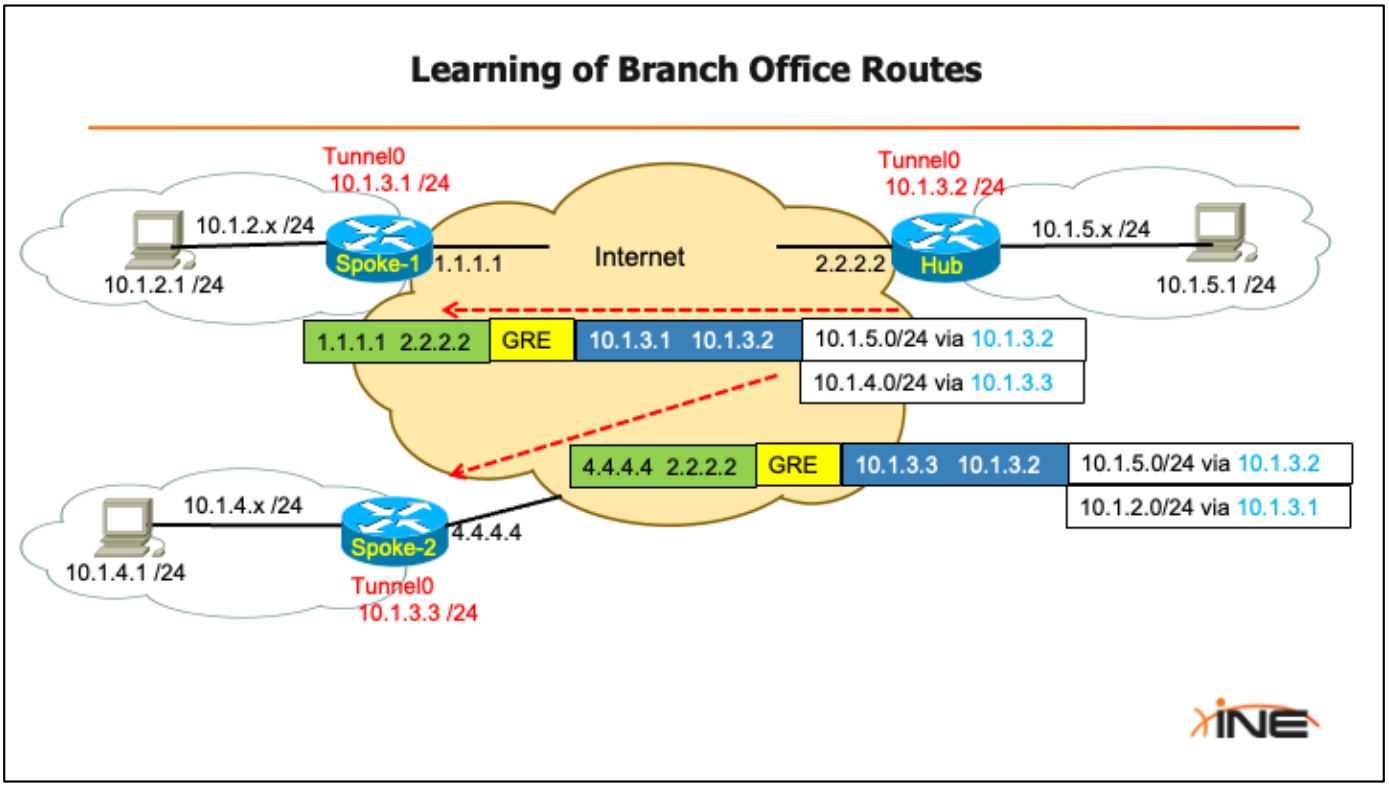




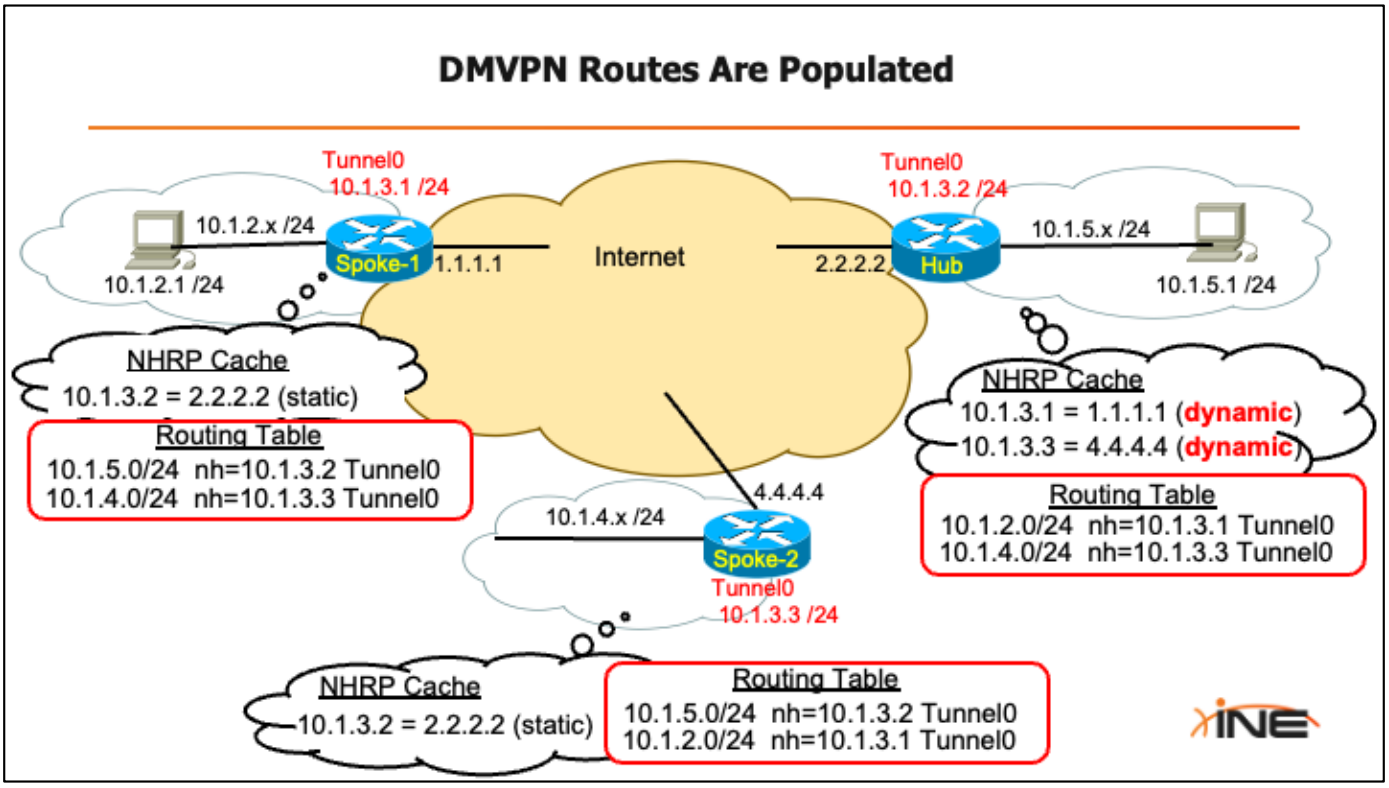
Route Exchange over DMVPN

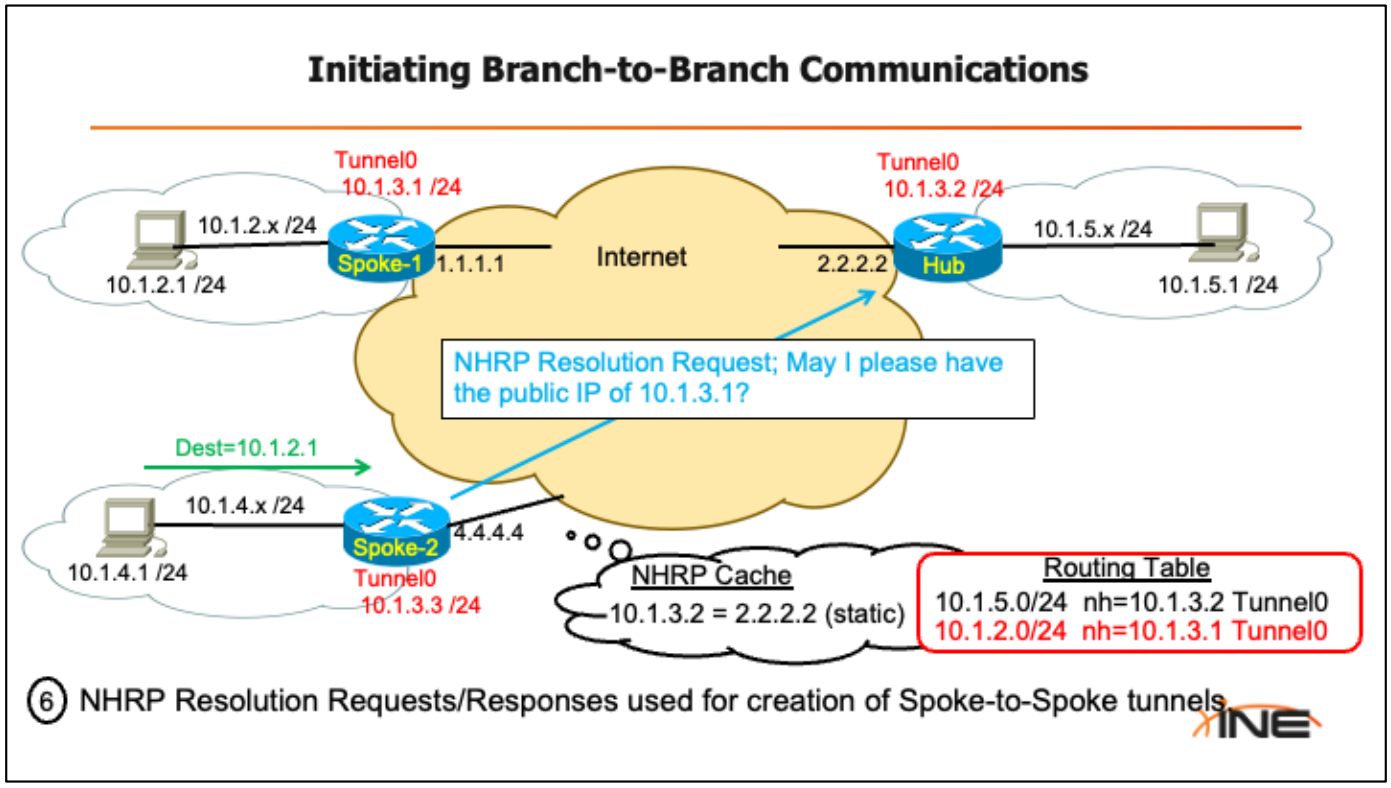


6 IGP Exchanges Routes over tunnel

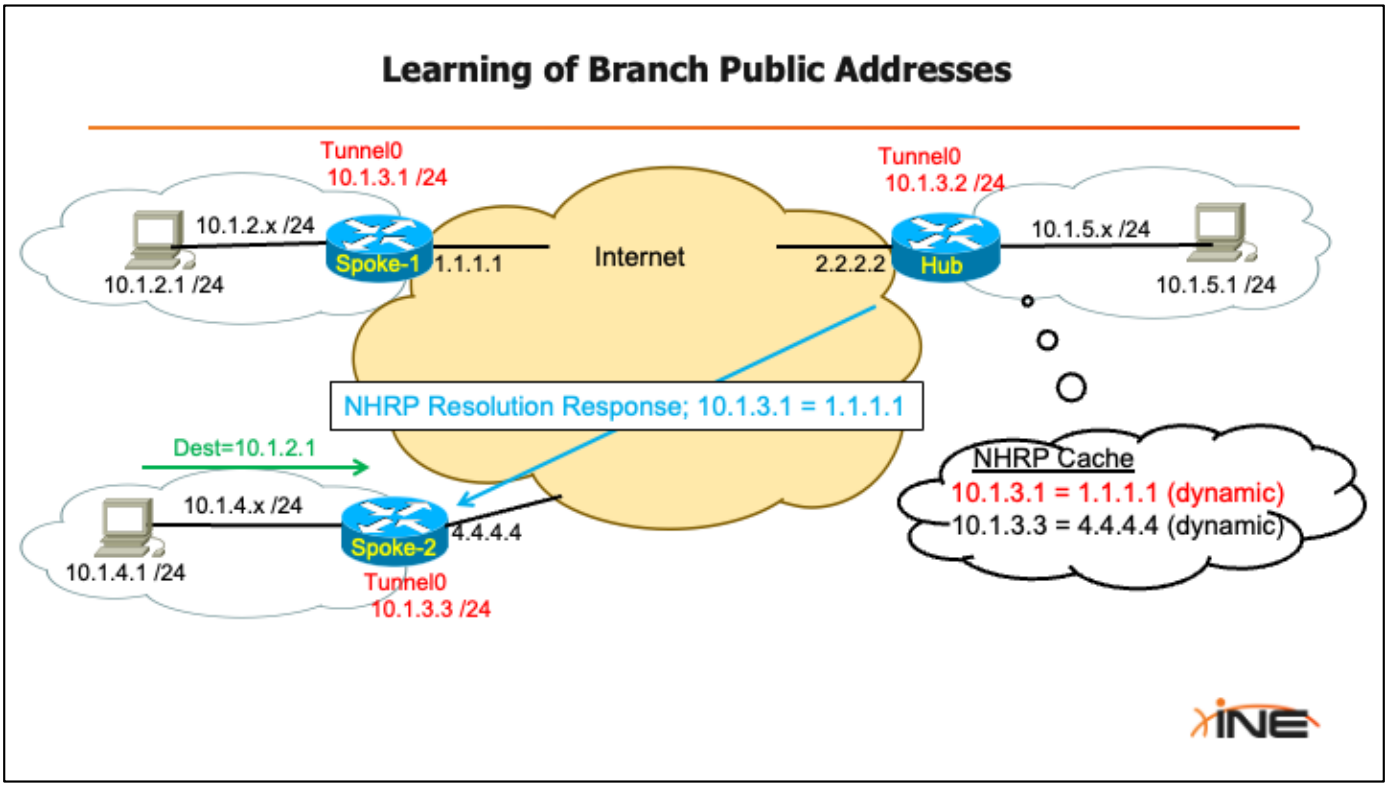


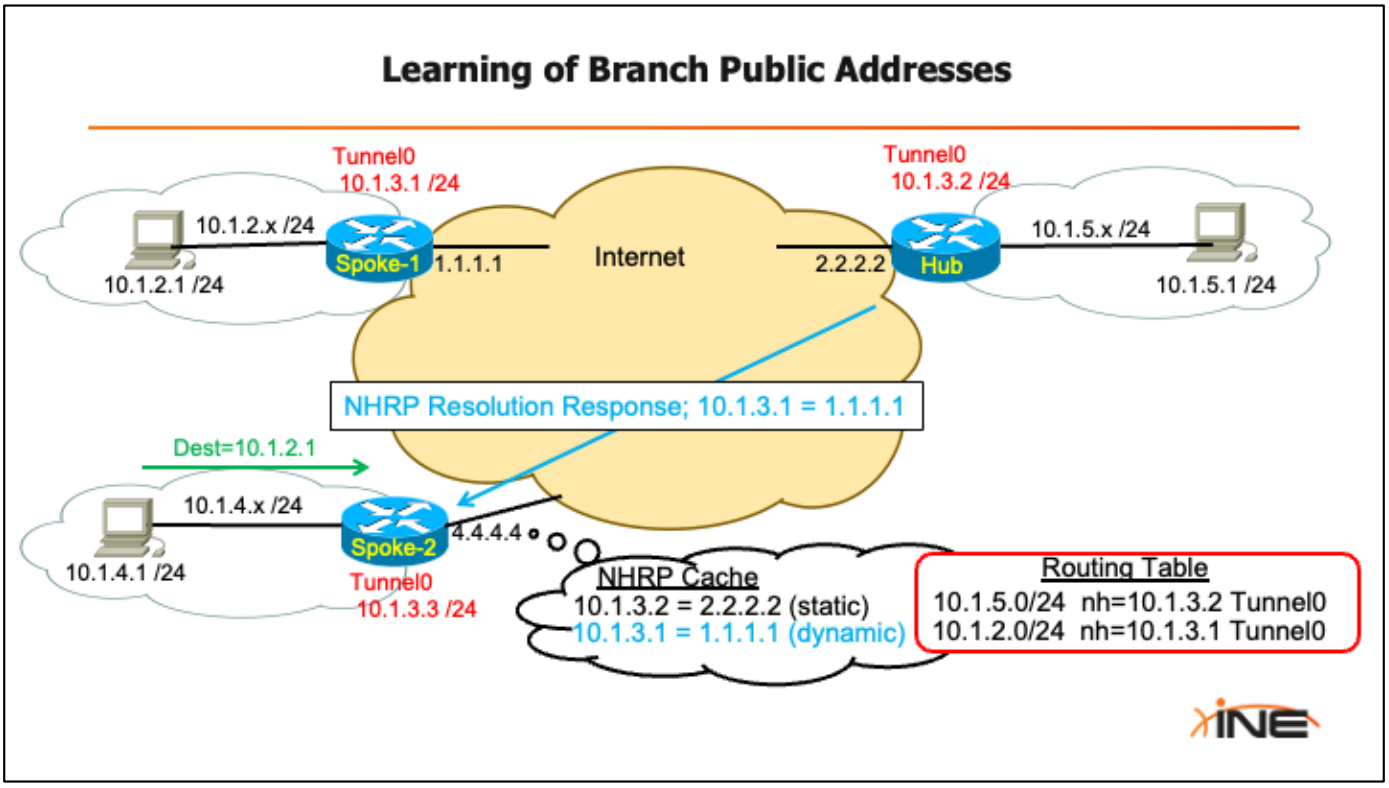
- Notice the next-hop advertised by the Hub.
- -
- Default behavior is that Hub would advertise itself as next-hop…so Spoke-to-Spoke traffic would never occur.
- -
- If spoke-to-spoke is desired, Hub must be configured to preserve next-hop in advertisements.

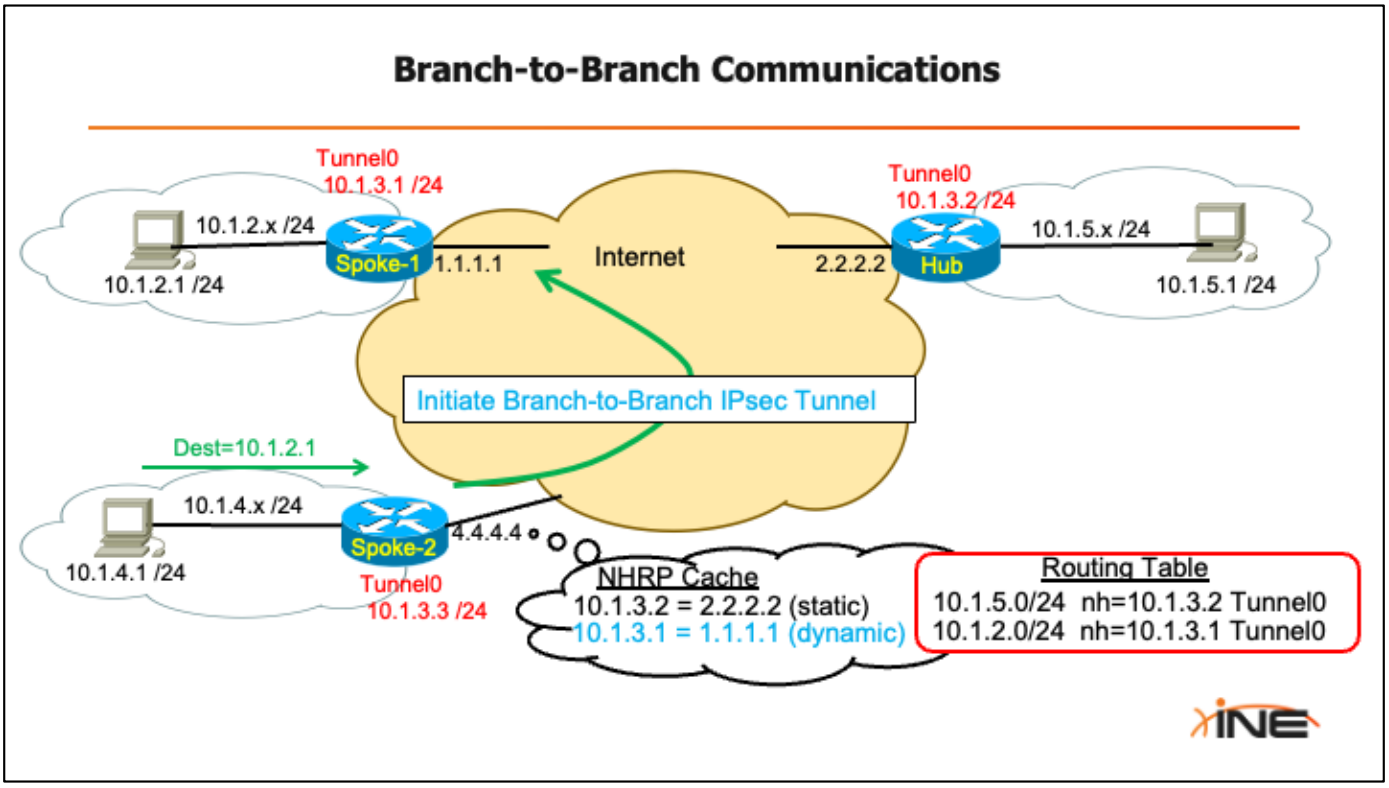




Traffic is still forwarded to the Hub while the spoke-to-spoke tunnel is being built.





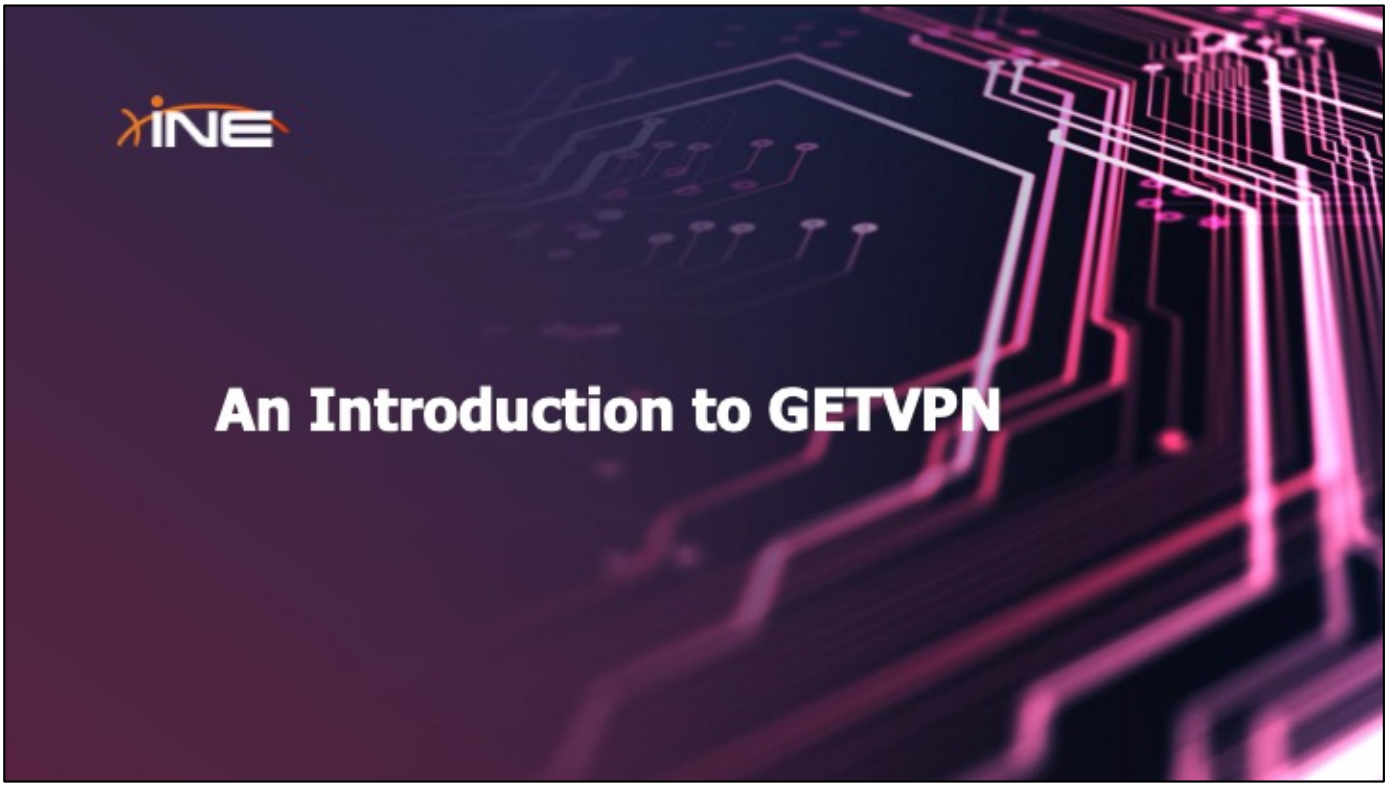


- Spoke to Spoke tunnels stay up as long as data is using them.
- -
- Each data packet using the tunnel refreshes a “Used” flag
- -
- If data stops flowing, eventually (around 2-minutes) the Spoke-to-Spoke tunnel goes down.

Benefits Of DMVPN

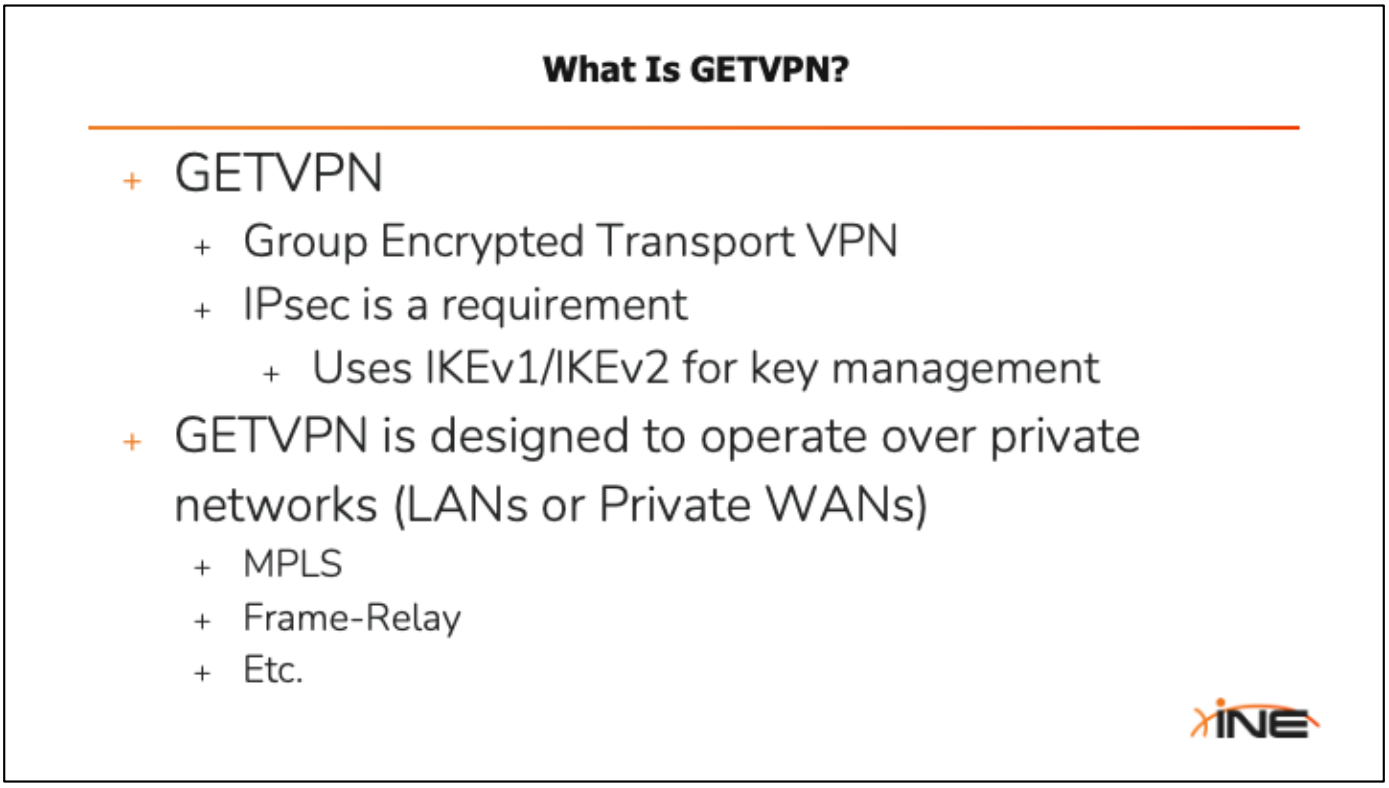
- Hub routers only need a single, pre-configured mGRE tunnel interface and a single IPsec profile
- Automatic IPsec initiation and auto-creation of GRE tunnels without any IPsec peering configuration
- Supports IP multicast, IP unicast and dynamic routing protocols between CE devices
- Remote spoke routers can have static, or dynamic, public IP addresses
- Supports all of the benefits of using GRE tunnels such as:
  - QoS
  - Deterministic routing
  - Redundancy scenarios

Thank you for watching!



Topic Overview

- GETVPN Overview
- What Problem is Solved?
- GETVPN Key Types
- GETVPN Device Roles
- Comparison of GETVPN to Other VPN Technologies



- This is a Cisco Proprietary technology although it is based on GDOI defined in RFC 6407.

What Problem Is Solved?

- Traditional methods to secure site-to-site traffic involve *creating a pair of IPsec Security Associations (SAs) for each VPN tunnel between sites.*
  - Each pair of VPN endpoints must maintain an IPsec control plane connection between themselves
  - Hundreds of tunnels/SAs per device is not scalable
- GETVPN allows each router to utilize only a single IPsec SA pair no matter how many remote sites it is speaking with.
  - One Security Association for inbound
  - One Security Association for outbound

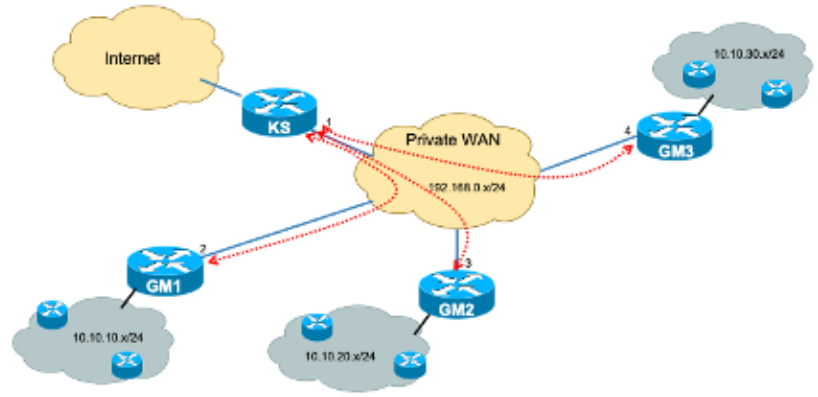
- Even with DMVPN, while your router might only have a single, configured Tunnel interface it will still need to dynamically build-and-maintain an IPsec Security Association for each encrypted endpoint.

GETVPN Roles

- GETVPN members can have one of the two roles
  - Key Server (KS)
  - Group Member (GM)
- GETVPN offers complete separation between control-plane and data-plane
  - A router cannot be KS and GM at the same time
  - KS is responsible for control-plane (key management)
  - GM is responsible for data-plane (traffic encryption)

GETVPN Device Roles

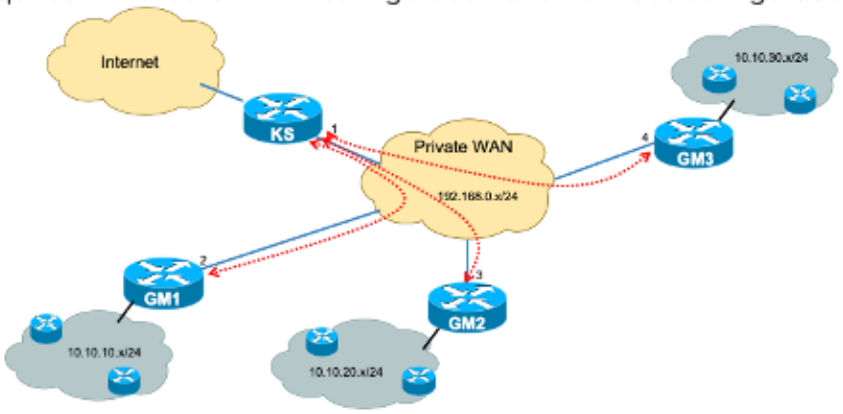
- Key Server
  - Maintains and distributes the GDOI to registered Group Members
    - GDOI = Group Domain Of Interpretation (group security policy)
    - IPsec encryption and authentication policy
  - Does not perform encryption or decryption of data
  - Maintains an IPsec SA with each GM for GDOI distribution



- The IPsec Security Association (SA) between the Key Server and each Group Member is strictly used to security transmit GDOI information and security policies…not to encrypt and decrypt actual user data.
- GDOI (like IKE) is a particular protocol used to implement what is defined in the ISAKMP framework. GDOI is defined in RFC 6407. However, whereas IKE is an implementation of ISAKMP for creating Security Associations between a pair of IPsec devices, GDOI is an implementation of ISAKMP designed to create a single Security Association that be can shared among multiple devices.

GETVPN Device Roles

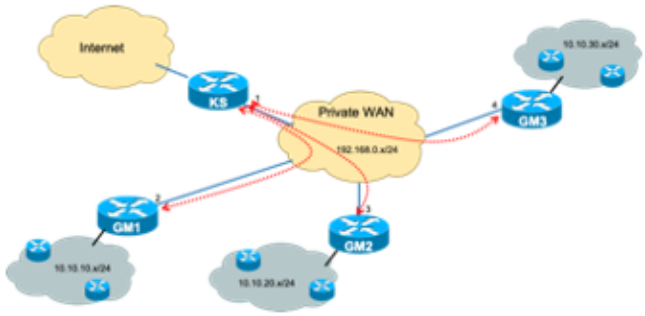
- Group Member
  - Registers with KS to receive GDOI
  - Performs encryption and decryption of data with other GMs
  - Maintains a single IPsec SA with KS only
  - Requires minimal ISAKMP configuration and no IPsec configuration



- Group Members must first authenticate with the Key Server before they are given the GDOI information.

GETVPN Key Types

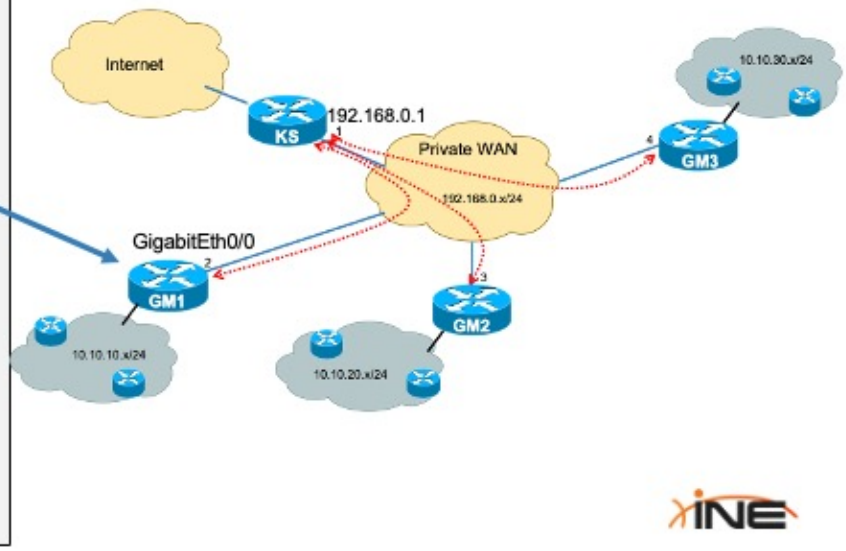
- After authenticating with the Key Server, Group Members receive two keys:
  - KEK (*Key encryption Key*) – Used for encrypting rekey messages from the KS. Is also used by GMs to encrypt/decrypt relay messages to/from Key Servers
  - TEK (*Traffic encryption key*) – The IPsec SA which is used by all GMs to encrypt traffic between each other



- GETVPN supports topologies in which Group Members (GMs) might not have direct accessibility to the Key Server (KS). In these scenarios, a GM can send a request to another GM to renew its keys or receive updated policy. This message can then be relayed by the intermediary GM to the KS. These are called “relay messages”.

GETVPN Device Roles

```
!
crypto isakmp policy 10
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0
!
crypto gdoi group GET
 identity number 1
 server address ipv4 192.168.0.1
!
crypto map MAP1 10 gdoi
 set group GET
!
Interface GigabitEthernet0/0
  ip address x.x.x.x
  crypto map MAP1
```



- Notice what is lacking in GM1s configuration:
  - No ACLs or other indicators about what destination or source subnets should be encrypted (or not encrypted)
  - No mention about what IPsec algorithms/protocols should be used to encrypt or authenticate data
  - No mention about how long IPsec encryption/authentication algorithms should be used for (eg. Lifetime)
- All of that information (and more) that is normally pre-configured on VPN endpoints is only configured on the Key Server and dynamically provided to the GMs after they register with the KS.
- The ONLY information the GMs need is:
  - What is the IP address of the Key Server?
  - What shared key (or Certificate if you wish to use that) should be used when initially registering with the Key Server?
  - Which egress interface will be used to encrypt/decrypt packets to other GMs?

Breaking Down A Name

- Named “Group” because
  - All routers protecting traffic use a single shared encryption key (same SPI value)
- Named “Transport”, as it does not tunnel traffic in the traditional sense
  - Uses IPsec ESP in tunnel mode with address preservation. No differences between outer and inner IP headers.
    - GETVPN just protects the layer 3 payload
    - This is why it was designed to operate on private networks
- Considered tunnel-less because
  - Routers performing encryption do not build direct IPsec tunnels
  - No control-plane traffic between these routers

- GETVPN DOES tunnel the original packet, but unlike a normal IPsec Tunnel, in GETVPN both the inner packet header as well as the (visible) outer packet header are exactly the same.
- SPI = Security parameter index. It is an IPsec term that defines all the protocols and values used to create and maintain a secure connection between two endpoints.

Thank you for watching!