CS203 Gene's Class - IPSec Overview

Lec6-IPsec.pdf
PDF

read the slide on ipsec

Here's a structured and comprehensive summary of the **IPSec lecture slides** you uploaded:

🔐 Why IPSec?

- **Problem:** IP (Internet Protocol) was not designed with security in mind.
- **Insecure by default:** Lacks origin authentication, integrity, confidentiality (both data & metadata).
- **Common attacks:** Source spoofing, eavesdropping, replay attacks, and packet reordering.

🌐 Where IPSec Sits in the Stack

- **Location:** Network layer
- **Transparent:** Works without application modification
- **Compared to SSL:** SSL works at transport/application layer; IPSec works lower.

🎯 Goals of IPSec

1. **Origin authentication**
2. **Freshness:** Prevent replay/reordering
3. **Integrity:** Detect packet tampering
4. **Confidentiality:** Hide content and metadata

🧱 IPSec Architecture

- **Key Components:**
  - **AH (Authentication Header):** Authenticates origin & integrity (no encryption)
  - **ESP (Encapsulating Security Payload):** Adds confidentiality (encryption) in addition to AH features
  - **IKE (Internet Key Exchange):** Handles key management and Security Association (SA) setup
- **Communication Modes:**

    - **Transport mode:** Protects just the payload
    - **Tunnel mode:** Encrypts the entire IP packet (adds a new IP header)
- **Use Cases:**
    1. Host-to-host
    2. Host-to-gateway
    3. Gateway-to-gateway

🛡️ Authentication Header (AH)

- **What it does:**
    - Origin authentication
    - Packet integrity
    - Replay protection (via sequence numbers)
- **What it does *not* do:**
    - No confidentiality (data is not encrypted)

🔐 Encapsulating Security Payload (ESP)

- **Everything AH provides**
- **Plus encryption:** Symmetric encryption to ensure confidentiality
- Also uses sequence numbers and HMAC for integrity

🔑 Internet Key Exchange (IKE)

- **Phase 1:**
    - Sets up a secure control channel
    - Uses **Diffie-Hellman key exchange**
    - Supports digital signatures, public keys, or pre-shared keys
- **Phase 2:**
    - Establishes actual **Security Associations (SAs)** for data transfer
    - Uses the secure Phase 1 channel
    - Only one mode: **Quick Mode**
- **Terminology:**
    - **SA (Security Association):** One-way secure channel (needs two per connection)
    - **SPI (Security Parameter Index):** Identifies SA in a packet
    - **SPD (Security Policy Database)** & **SAD (Security Association Database)**: Manage rules and states

📄 IPSec Policies

- **Phase 1:** Specifies protection suites (encryption, hash, auth method, DH group)
- **Phase 2:** Defines proposals (AH, ESP, IPComp with attributes like key length, lifetime)
- **Example Policy:**
  - Traffic to `128.104.120.0/24`:
    - Use ESP with 3DES encryption and HMAC-SHA integrity
    - Or use AH with HMAC-SHA only

🕸 Use Case: Virtual Private Networks (VPNs)

- **VPN = Virtual + Private Network**
- Uses **tunnels over public networks** with encryption
- IPSec VPNs commonly used in enterprise settings

⚠️ Limitations & Critiques

- **Authenticates machines**, not users
- **Doesn't prevent DoS**, and may make it easier
- **Complex configuration** (many acronyms, modes, and protocols)
- **Not always supported by routers or applications**

Would you like a diagram summarizing IPSec modes and headers (Transport vs Tunnel)?

lets say two hosts a and b, each has their gateway. ipsec connection can be between a and its gateway, between two gateways, between b and its gateway. another example can be direct ipsec between a and b directly

prof said

Yes, your professor is referring to the **three common deployment configurations** of IPSec based on where the IPSec processing happens. Let's break them down with your examples:

✅ 1. Host-to-Gateway (A → Gateway A)

- **Who sets up IPSec?** Host A and its own gateway.
- **Use case:** A remote worker (Host A) connects securely to their organization's network via a **VPN client**.

- **IPSec job:** Secure the connection from A to its gateway (e.g., traffic tunneled to company resources).

✅ 2. Gateway-to-Gateway (Gateway A ↔ Gateway B)

- **Who sets up IPSec?** The gateways at both ends.
- **Use case:** Two offices connected over the Internet; traffic between them is protected.
- **IPSec job:** Secure all traffic between the two networks behind the gateways. Hosts A and B don't even know IPSec is being used (transparent).

✅ 3. Host-to-Gateway (B ↔ Gateway B)

- **Same as case 1, but on the B-side.**

✅ 4. Host-to-Host (A ↔ B directly)

- **Who sets up IPSec?** The actual endpoints A and B.
- **Use case:** Two computers communicating securely without involving gateways. Often used in tightly controlled environments (e.g., secure servers).
- **IPSec job:** Full end-to-end security between applications.

📌 Summary Table

Setup Type	IPSec Endpoints	Tunnel or Transport Mode?	Common Use Case
Host-to-Host	A ↔ B	Transport or Tunnel	Secure direct app-to-app connection
Host-to-Gateway	A ↔ Gateway A (or B ↔ Gateway B)	Tunnel	Remote access VPN
Gateway-to-Gateway	Gateway A ↔ Gateway B	Tunnel	Secure site-to-site communication

---

Would you like a visual diagram of these four configurations?

explain two modes, transport and tunnel

Absolutely! Here's a clear and intuitive explanation of the two IPSec modes: **Transport Mode** and **Tunnel Mode**, as discussed in your lecture.

🚚 Transport Mode

✅ What It Does:

- Encrypts **only the payload (data)** of the IP packet.
- **Keeps the original IP header intact**.

🧠 Think of it as:

> "Just protect the contents, leave the envelope as is."

🔍 Used For:

- **End-to-end** communication between two hosts (e.g., A ↔ B).
- Often used in **host-to-host IPSec** scenarios.

🧱 Packet Structure:

| IP Header (Original) | IPSec Header (ESP/AH) | Payload (Encrypted) |

The IP header remains **visible** to the network (for routing), but the payload is **protected**.

🏠 Tunnel Mode

✅ What It Does:

- Encrypts **the entire original IP packet** (header + payload).
- Wraps it in a **new IP header** for routing.

🧠 Think of it as:

> "Put the entire original packet inside a secure envelope."

🔍 Used For:

- **Gateway-to-gateway** or **host-to-gateway** VPNs.
- Useful when traffic from one private network must be routed securely to another.

🧱 Packet Structure:

| New IP Header | IPSec Header (ESP/AH) | Original IP Header + Payload (Encrypted) |

So everything inside is **invisible** to outsiders — even the original IP addresses.

✨ Key Differences

Feature	Transport Mode	Tunnel Mode
What is encrypted?	Only the payload	Entire original packet (IP header + data)
IP header visibility	Visible (original header intact)	Hidden (new outer header is used)

Feature	Transport Mode	Tunnel Mode
Use case	Host-to-host	Gateway-to-gateway, VPNs
Overhead	Lower	Higher (due to encapsulation)

📦 Real-Life Analogy

Mode	Analogy
Transport	Sending a postcard with contents in a sealed envelope
Tunnel	Putting the postcard inside a box, then shipping the box

Let me know if you'd like a side-by-side diagram of both modes.

so in transport mode, if ipsec betn gateway of a and b is compromised, then the ip addr of a and b will be exposed

Exactly — that's a **key point**.

🔓 In Transport Mode: IP Addresses Are Exposed

If IPSec is used **in transport mode between the gateways of A and B**, and that connection is compromised:

- The **original IP header** is **not encrypted**.
- That means the IP addresses of **host A and host B** are **visible to anyone** who intercepts the traffic.
- So, an attacker can still **see who is talking to whom** even if they **can't read the payload**.

📌 Summary:

🔍 What is visible in transport mode?
✅ Source & destination IP addresses
✅ Protocol information (e.g., TCP/UDP)
❌ Encrypted payload (data)
❌ Integrity-protected payload

💡 If Metadata Privacy Is Important:

Use **Tunnel Mode** — because it **encrypts the entire original packet**, including:

- IP header (source & destination)

- Payload
- Protocol metadata

Only the **new outer IP header** (added by the gateway) is visible.

So yes, **transport mode leaks IP-level metadata**, making it unsuitable if you're worried about network-level surveillance (e.g., hiding who is communicating).

Would you like a table summarizing what's visible in each mode?

explain slide 15

Great — let's break down **Slide 15** of the IPSec lecture, which shows **IPSec Architecture: typical use-cases and modes**.

🖼️ What the Slide Shows

Two diagrams side-by-side:

- **Left:** Two routers connected — labeled **Tunnel Mode**
- **Right:** Two hosts connected — labeled **Transport Mode**

🔧 What It Means

🏠 Transport Mode (Right Diagram)

- **Who sets up IPSec?** Host A ↔ Host B
- **What is protected?** Only the **payload** of the IP packet.
- **What is not protected?** The **IP header** is left unchanged and visible to routers.
- **Use Case:** End-to-end security between two computers.
- **Benefits:** Low overhead, full protection of data contents.
- **Limitation:** Leaks addressing metadata (source/destination IPs are visible).

🛣️ Tunnel Mode (Left Diagram)

- **Who sets up IPSec? Routers or gateways** between Host A and Host B
- **What is protected?** The **entire original IP packet**, including the IP header.
- **How?** A new IP header is added for routing the encrypted packet between the gateways.
- **Use Case:** Site-to-site VPN or remote access VPN.
- **Benefits:** Full protection — even **source/destination IPs** are encrypted.
- **Limitation:** Slightly more overhead due to encapsulation.

🧠 Key Insight from Slide 15:

Feature	Transport Mode	Tunnel Mode
IPSec runs on	End hosts	Gateways/Routers
Data protected	Only payload	Entire IP packet
Use case	Host-to-host secure comms	VPNs (gateway-to-gateway/host)

So this slide visually distinguishes **where IPSec runs** and **what it protects**, depending on the **mode of operation**.

Would you like a simplified diagram of these two cases?

in AH tunnel mode, the original ip header is already exposed, although the ip header and data is encrypted for ipsec. then whats the point of tunnel mode to be honest?

Excellent question — and you're absolutely right to scrutinize this detail.

Let's clarify what's happening in **AH tunnel mode**, and why it might seem confusing at first.

✅ First, Recap What AH Does

AH (Authentication Header) provides:

- ✅ **Origin authentication**
- ✅ **Data integrity**
- ✅ **Replay protection**
- ❌ **No encryption**

So unlike ESP, **AH never provides confidentiality** — it does **not encrypt** the IP header or payload.

🧱 What Happens in AH Tunnel Mode

In AH Tunnel Mode:

1. The **entire original IP packet** (including its header and payload) is **wrapped** inside a new packet.
2. A **new IP header** is added outside.
3. The AH is applied over the **entire inner packet**, including the **original IP header and payload**.

But critically:

- The original packet is **not encrypted**.
- The outer IP header is **also not encrypted**.
- Only **authentication/integrity protection** is applied to the inner packet.

🤨 So What's the Point?

Even though the original IP addresses are exposed, **tunnel mode with AH is still useful when**:

1. 🔒 You want to detect tampering of the original IP header (which ESP cannot do).

- ESP **does not protect** the outer or inner IP headers in tunnel mode.
- AH **does** authenticate the entire original packet — including the original IP header.
- This is valuable when integrity of routing metadata is important.

2. 🧱 You're authenticating traffic across gateways, not encrypting.

- Sometimes you just want to ensure packets weren't modified in transit.
- Tunnel mode allows **gateway-to-gateway authentication** of complete original packets.

❌ When AH Tunnel Mode is *Not* Ideal

- If you're worried about **eavesdropping**, AH is useless — it doesn't hide anything.
- If **metadata confidentiality** is required, you should use **ESP in tunnel mode**.

✅ Summary: AH Tunnel Mode vs ESP Tunnel Mode

Feature	AH Tunnel Mode	ESP Tunnel Mode
Encrypts payload?	❌ No	✅ Yes
Hides original IP addresses?	❌ No (they're still visible)	✅ Yes (encrypted with payload)
Protects IP header integrity?	✅ Yes (authenticates inner IP hdr)	❌ No (ESP doesn't cover IP header)
Main benefit	Integrity/authentication of entire pkt	Full confidentiality and integrity of data

---

🧠 Final Thought

> **AH Tunnel Mode** is about **authenticating full packets** (including their routing info), not **hiding** them. It's not about secrecy — it's about **trusting that the packet wasn't tampered with**.

Let me know if you'd like to walk through an example packet layout or compare AH vs ESP graphically.

why ipsec does excryption first then authenticate, why not other way

Great question — this is a classic and **important cryptographic design decision** in IPSec. Let's unpack the reasoning clearly.

🔄 Why IPSec Does Encrypt-then-Authenticate

🔐 Step-by-step (in ESP):

1. **Encrypt the payload** using symmetric encryption (e.g., AES).
2. Compute a **MAC** (Message Authentication Code, like HMAC) **over the ciphertext**.
3. Send:
   → `Encrypted Payload + Authentication Tag`

🤔 Why Not Authenticate First, Then Encrypt?

If IPSec did **Authenticate-then-Encrypt**, it would:

1. Compute the MAC over the **plaintext**.
2. Then encrypt **both** the plaintext **and the MAC**.
3. Send the encrypted packet.

At first glance, that sounds fine — but here's why it's problematic.

⚠️ Problems with Authenticate-then-Encrypt (AtE)

1. ❌ Not secure under chosen ciphertext attacks (CCA)

- The receiver **cannot verify integrity** before decrypting.
- An attacker could send a malicious ciphertext that:
  - Causes predictable changes in decrypted output.
  - Triggers timing, padding, or error-based side-channels.
  - Leaks info about plaintext (as in the old **SSL/TLS padding oracle attacks**).

2. ❌ Error messages leak info

- If authentication is checked *after* decryption, and decryption fails visibly, an attacker can learn things about valid plaintexts.

✅ Benefits of Encrypt-then-Authenticate (EtA)

1. **Early rejection of invalid packets**
   - Receiver checks the MAC **before decryption**.
   - If the MAC is invalid → discard immediately.
   - Saves computation and prevents side-channel leaks.
2. **Strong security guarantees**

- EtA is provably secure under strong models (e.g., IND-CCA).
- That's why modern standards (e.g., IPSec, Signal, TLS 1.3) use EtA or integrated modes like **AEAD** (e.g., AES-GCM).

🧠 Summary Table

Order	Secure Against CCA?	Easy to Implement?	Used in IPSec?
Encrypt-then-Auth (EtA)	✅ Yes	✅ Yes	✅ Yes (ESP)
Auth-then-Encrypt (AtE)	❌ No	❌ No	❌ No
Encrypt-and-Auth	⚠️ Sometimes	✅ Yes	❌ Not in IPSec

🔍 Real Example

IPSec ESP (Encapsulating Security Payload):

- Uses EtA.
- So:
  - First: Encrypt the payload.
  - Then: MAC is computed over the encrypted content + some headers.
  - Receiver **authenticates first**, and **decrypts only if valid**.

So in short:

> **IPSec uses Encrypt-then-Authenticate because it's more secure — it prevents attacks that can arise when you try to authenticate unencrypted data.**

Would you like a diagram or code analogy of this order?

this is professor's transcribed lecture

that means we want to have both timeliness and freshness of our IP packets.
Of course, we're also equally interested in protecting integrity of data.
And make sure that the data in the payload, the data in the packet has not been modified in transit.
We want to sometimes protect confidentiality of the data, not always.
Just like in SSL, there's always, like remember there's an option not to use encryption,
you use null encryption, while in IPsec there are options also not to use encryption.
But authenticity, authenticity of origin and data integrity are mandatory.
Meaning you don't want to use IPsec without authenticity of origin and data integrity.
That's the minimum thing IPsec does.
In addition to that, you can protect what's called packet metadata.
That is to protect against eavesdropping.
So the typical example is that you have, let's say, two branches of the same company.

Let's say you have an East Coast branch and West Coast branch.
And inside each branch there's a private network, interconnected lands, all kinds of stuff.
All kinds of like departments have their own arrangement on the West Coast.
There's something like that, and on the East Coast, right?
And how do they communicate?
Well, normally, you know, IP posts on one on the East Coast, also IP posts on the West Coast.
And notice that if you eavesdrop on these packets in the middle of the Internet or outside these organizations,
you will know who is talking to whom, right?
So that's information that can be valuable, right?
Because you're also learning the internal structure of the private network, okay?
It exposes addresses within the private network, both source and destination.
So what IPsec allows you to do in a certain mode of operation is to encapsulate such packets
and hide the internal source of this connection so that anyone listening and eavesdropping on IP packets
anywhere in the middle of the Internet wilderness will only see that these packets go from right?
Company A East Coast branch and Company B West Coast branch, but will not know anything beyond that, okay?
This is very important.
This protects against what's called traffic analysis.
So the general model is like this, right?
The IPsec assumes that hosts and gateways, right, or border routers, are secure themselves.
It doesn't make any assumptions about software security, operating system security, etc.
But it assumes that hosts can be secured and gateways, but the communication lines themselves are insecure.
Right?
Whether they're wired, wireless, whatever medium you use, they are insecure.
Insecure means eavesdropping, insecure means active attacks, like deletion, insertion, manipulation,
delaying, reordering, etc.
Okay?
Now, we will not cover the entirety of the security model.
And the reason is because it's very complex and it's very abstruse.
And yes, if one deals with IPsec, one should know everything.
But I'm going to give you again just kind of an appetizer for IPsec.
And I'm going to talk mainly about IPsec formats and options.
But there is something called Ike, called Internet PHTH, which is a basically, think of that as a control, remember the control channel in SSLTLS?
The one that establishes security, you know, the handshakes and everything?
That's what Ike does.
Okay?
But unlike SSLTLS, where everything is put together in one protocol, right?
I cannot tell you about SSLTLS packets or record formats without describing the handshake, right?
Whereas an IPsec, I can't, and Ike is a separate component.
Okay?
So, what it does is that it is a protocol, or a set of protocols actually, that establish key material, cryptographic shared keys, between pairs of communicating IP addresses.
Okay?
So, we will not talk much about this.
Ike, in turn, works together with something called security policy, which is an organizationalist,

OpenAI

something that every organization that uses IPsec should ideally maintain.
It's not mandatory, but it needs to be maintained, generally.
And security policy is basically telling you what IPsec to use, what type of IPsec to use, where.
So, think about, like, you're trying to communicate within your organization.
You work for ABC Incorporated, and you are in your branch office, you're working in a particular office, you have your private network,
and you're trying to establish a connection, let's say, to a nearby post, right, inside.
And it might look at your packets and say, oh, this is inside, no IPsec required.
But as you are trying to maybe communicate, let's say, to a social network outside, it might say, for example, no.
Which means, not allowed, period.
IPsec, IPsec, not allowed.
Okay?
See, Facebook, there's no way.
Or it might see something like Gmail, right?
You're going to Gmail, right?
You're going to Gmail.
It might say, oh, if you're talking to Google, then you have to use IPsec in this mode, this specific mode.
And I'll show you examples of what that means.
So, the policy governs the security flavor that is applied to a specific host to host connection.
So, there are two formats in IPsec, and it can be confusing, so I'm going to try to step through it.
One is called S, or encapsulating security payload, and it's more complex.
And the simple one is sort of IPsec-like, it's called authentication header.
Okay?
So, you use one or the other, between the two hosts, you don't generally use more than one.
You pick one or the other.
And to make, well, more later, so you can use it in a following way.
You can use it in a host-to-host communication, just like an example I described.
You have host A in the West Coast branch, talks to host B in the East Coast branch.
Right?
So, these are host-to-host, ant-to-ant use of IPsec.
You can also use it host-to-gateway.
An example of that is, let's say you are an employee of that ABC incorporated.
You take your laptop with you on travel to Turkestan, and you are in a hotel room, or in an internet cafe, or something like Starbucks, and you want to dial back home.
You want to call home, check your mail, whatever, work on some internal documents.
So, in that sense, you are communicating your laptop, your host communicates to the gateway of your home office, of the private network in your home office.
So, the IPsec connection would be then between your interface on your laptop, and the IP interface, incoming IP interface on your gateway.
Okay?
Alternatively, it would be gateway to gateway.
That would be, for example, if the same as my first example, host A in West Coast branch office talks to host B in the East Coast branch office, but the connection is not IPsec.
End-to-end, it's between their respective gateways.
Does that make sense?
So, the gateways provide like a secure tunnel or secure pipe.
And then, within each branch office, there is no IPsec.
Meaning, between me, let's say I am host A, between me and my exit gateway, there is no

IPsec, and the IPsec starts there, goes to the gateway of the other branch office, over the wilderness of the internet, there it terminates, and from there on to the destination host B, there is no IPsec.
Okay?
That's, that's gateway to gateway.
Okay?
And, to make things even more interesting, all these things can be combined.
Okay?
It is, there was a marker here at some point, but I think it's gone.
What's going on now?
Is there?
Oh, wow.
Okay, so, you could have host A. This is the branch office, this is the branch office.
This here is bad, right?
Internet.
And, this host could actually have an IPsec connection here, to this gateway.
Then, there could be another IPsec connection between gateway B and a gateway A, and a third IPsec connection between gateway B and host B. That is if you, like, don't trust the internal networks, right, in either organization.
So, you could have that, that's totally fine.
What is also fine, one second, is you could have, everybody saw this, right?
You could have a connection that goes like this, okay?
Between A and B, that's called IPsec E to B, and this is IPsec G to G.
So, there is an encapsulation here.
Does that make sense?
So, you have an end-to-end connection from A to B, but from gateway to gateway, from gateway to gateway to gateway to B, you have another one on top of it, like a thicker pipe.
And so, the idea is that there may be many other hosts here, many other hosts there, but they will utilize the same pipe.
Question?
Yes.
Okay.
And, you can imagine other variations as well, right?
For example, my first example with three of them doesn't have to be like that.
It could be like A to gateway, there is no IPsec, then gateway to gateway, there is IPsec, and then again, gateway B to B is IPsec.
Totally fine.
Any combination is okay.
So, as I said, AH, authentication and ASP, and to make things, this is very important, to make things even more confusing and complicated, there are two modes of operation for each.
So, both authentication and encapsulating security payload can operate in two modes, transport and tunnel.
Okay?
So, what's the main difference?
You'll see the pictures, the pictures are on board worth a thousand words, but the main difference between tunnel and transport mode is in tunnel mode, you take an IP data here, IP packet, and you treat it as a block.
You slap a new header in front of it.
Does that make sense?
That's a tunnel.
It means you're hiding the entire IP packet, or protecting the entire IP packet as is.

That's tunneling.
Transport is, you don't do that, you just use the original IP header, and then in the middle, between IP header and the payload, right, the transport, etc., you stick an IPsec header.
So, it saves bandwidth, right?
And some process.
But, it's a trade-off.
Why?
Because, anybody can already see from my description where is the trade-off?
There's one trade-off.
Tunneling is more secure because non-tutneling, transport mode, exposes the end-to-end IP addresses.
Whereas, if you tunnel, right, for example, in this case, if we had here,
if this connection is transport mode, right, everything is cool except the A and B addresses will be exposed.
If the A, this gateway to gateway to gateway to B is in tunnel mode, then the A and B addresses are hidden.
Well, unless you use just authentication, which is stupid, but you don't want to do that.
But, they're hidden.
Which means, on the internet here in the wilderness, the only thing you would see as source destination IP would be gateway A, gateway B.
Okay?
So, now you sort of know the difference.
Right.
This is just a point if you're not familiar with gateway, but I'm using it in a sense of a border router.
Right?
So, if you know, you should know, that the internet is composed of what's called ASs or ADs sometimes.
AS or AD stands for Autonomous System or Autonomous Domain, and all of these terminology refers to inorganizational networks.
That is, could be a ginormous thing like AT&T that provides long range services as one AD or one AS, or it could be something smaller like UCI, which is also NAS.
Okay?
But, there's a difference in the internet between something called transit ASs and stub ASs.
And, just as the name suggests, transit is the one that is in the middle of the internet to provide services to endpoints, and stub is like the end.
UCI, for example, does not provide transit services to UCLA.
Right?
Or the city of Irvine.
We are a stub.
Okay?
Your home network is kind of a stub.
It's not really an AS, but let's say you are here, I don't know what you guys use.
You guys use UCI campus networking if you live on campus, but if you live off campus, maybe using Cox, right?
So, one of my experiences was like for many years I had Cox, right?
So, Cox is the internet, kind of default internet provider around here.
And they are an ISP and a stub domain.
Right?
Because they have what's called an AS, or domain number assigned to them, you know?
Anyway, so, typically, IPsec involves gateways, right?
And they sit on the border of a domain or autonomous system.

Okay?
So, this is just going to be a slightly better picture that shows you, right?
So, this would be the tunnel mode right here, the typical example, right?
It's not a mandatory example.
It's not the way to use it.
It's a way to use it.
But it's the one that makes sense, is that you would provide tunnel mode between routers.
This is like my example of branch office, east, west coast, east coast, right?
There would be tunnel mode between the routers, and then the transport mode between the hosts.
They do not interfere with each other.
They totally coexist and they don't care about each other.
Okay?
You might view some of this, of course, is a little bit redundant, right?
But, consider that hosts, why would hosts use transport mode?
Because maybe this, this is insecure, right?
Like, why use IPsec between here, you know, end to end?
Because they don't trust this network.
I mean, they don't trust this network.
They may also not trust the routers necessarily, right?
Different levels of security.
So, here may be everything, routers may say, okay, this is all secret, or level secret,
but for the host communication, they say, oh, no, no, we, this session is top secret.
Which means higher security, so they may want to have a separate session.
Right?
And this is yet another pictorial representation for those who are visual learners.
Right?
So, you see there, tunnel mode, right?
This is independent of whether you use AH or ESP.
The way that, at the top, you have a sort of original IP packet with a header and data,
and in a tunnel mode, the header and data are protected.
It says encrypted, it's strictly speaking, not always encrypted, but it's at the very least protected.
Okay?
Then it slaps an IPsec header in front, and before that prepends a new IP header.
Now, if you use this mode between two hosts, like in my previous example, previous picture,
if you use this kind of a tunnel mode, right, between two hosts, what happens is the same
source destination addresses are in the inner header, the protected header, as would be in
an outer header.
Do you see that?
Following me?
Because, right, it's the same two hosts, right?
Endpoints.
So, if you use the tunnel mode, the header that is protected, that is within, has the same
source destination as the header that is outside.
So, it might seem kind of weird, right?
In that case, we are not protecting traffic from traffic analysis.
What we might be protecting is, for example, what protocol is it?
So, the new IP header here will have the same source destination, right, if it's host to
host, right?
But it will not have, remember there's a protocol field in IP header that says, ah, I'm using
TCP, socket, there's like a support number, or UDP, or ICMP, or God knows what, right?
There's more than just TCP on top of IP in general, right?

So, that is potentially sensitive information.
So, two hosts may want to use tunnel mode to hide what kind of transport layer protocol they're using and what port numbers they're using.
Okay?
So, you still get something out of it.
And, um, transport mode, of course, is, as I said before, you take the original IP header, and you, you still use it, but you insert the IPsec header.
Okay?
The other thing that this, this actually, obviously I hijacked this figure from, from somewhere, going to like a textbook, but, um, what it doesn't show you is that, well, never mind,
I don't actually, never mind, I said, all good.
Um, that's another representation maybe easier to have.
You see, original datagram, this example has TCP, but it doesn't, remember it doesn't have to be TCP,
it could be UDP or some other protocol.
Transport mode, tunnel mode.
Okay?
So, this is the same information as you showed before.
Slightly different.
Any questions?
So, as I said before, it's, as I said, is precisely the set of format, packet formats.
Okay?
Not a protocol.
All right?
And the first two RFCs, remember I told you what RFCs are, they're kind of like internet standards written in a very special way.
Not as bad as patents or legal documents, but, but not exactly like technical papers.
And they're very detailed, they specify, they usually have state diagrams and all kinds of other transitions.
They encapsulate everything.
And the reason is, so you should be able to take an RFC, implement it in whatever gut-forsaken language you want,
on whatever gut-forsaken platform, hardware platform you want, whatever operating system you want,
and it will still interoperate with anything else that runs the program.
That's the whole idea.
Right?
Whether you use an 8-bit processor or a 64-bit processor, a supercomputer or an IoT device, they will interoperate.
If you faithfully implement the program.
So, the first two are IPsec.
And this is the item, you know, you can look at it, it's really complex.
And the reason it is so complex is because it was designed to be super open and flexible, meaning that the Internet Engineering Task Force, the organization that oversees Internet standards,
not just security, but all Internet standards, is international.
And it doesn't, it tries not to be, like, very centric about protocols or algorithms,
like encryption algorithms or authentication algorithms, so it tries to accommodate everything.
And so it tries to be super flexible and leave room for new, new, new techniques.
Right?
So that's why what AH and ESP tell you, only the formats.

Only the formats.
Okay, let's zoom into authentication here.
That's the lightest IPsec version.
It basically provides two things, origin authentication and data integrity.
It does not, even though you will see the word encrypt, it does not encrypt.
Okay?
It does not provide confidentiality.
It provides data integrity and origin authentication.
Okay?
Also protects against, like, replayed, or allows you to detect replayed and out-of-order messages, packets,
by using this sort of monotonically increasing sequence number.
Not the same thing as a sequence number inside that IP had.
Its own sequence number.
Okay?
So remember, no data confidentiality, no metadata confidentiality, in this version.
Okay?
So as I said, 32-bit sequence number, and then uses cryptographic hash algorithms, which are basically all kinds of versions of HMAC.
Remember HMAC?
Kind of quickly defined it, showed you how it works.
HMAC is a construct that can be used with any good hash function.
Where good means cryptographically strong hash function.
Only symmetric crypto.
Only symmetric.
Okay?
Here's the format of the beast.
First thing you see is next header, which is basically a pointer to what to do next.
Right?
Remember the packets are concentric, right?
Outer header, inner header, inner inner header, and then finally data, right?
So you should be able to, you know, your code, right, that runs and parses packets needs to go from processing one header to the next header to the next.
So it tells you the next header, where the next header is.
Then it says payload length, right?
Right?
That's the, that's the size.
Then it's reserved, God knows why.
60 bits, not used.
Then there's five, the security parameters index.
Don't ask me what, why they picked this terminology.
Well, actually what it is, is like a session ID.
Okay?
So this identifies, SPI identifies the session, this particular session between these two communicating IP posts or IP interfaces.
It is one way, just like SSL-TLS, remember?
It's one way.
Meaning that SPI used for host A to B is going to be different than the one used from B, or back at set, from B to A.
Just like SSL-TLS.
Which means like the keys associated, right?

With A to B direction will be different than those keys used in B to A direction.
Xenos number that I already told you about.
And then authentication data, which it has variable size depending on which cryptographic, like HMAC flavor you're using.
Because remember, HMAC outputs a value of a hash function, right?
Remember there's two concentric applications, right?
With a key, etc.
But the output varies.
If you're using SHA1, it might be 128, as little as 128 bits.
Using SHA2 will be at least 256 bits.
SHA3, there are a few other hash functions that have it.
So the size of the authentication token, this is the authentication data that authenticates the packet, will vary depending on the hash function used in the HMAC.
And how do you know which function is used?
Well, the SPI, right, is the identifier of the session that will point to a record somewhere, right?
They'll say, oh, this connection uses this hash function for HMAC.
Okay?
So there's no confusion, right?
You see, there's no confusion about how long this should be.
But it is a variable field.
Now, for a given host A to host B H session, you cannot change the hash function.
There's no change cipher spec like an SSLT has.
You are fixed for a given connection.
More pictures?
Okay, so this is, again, we're talking about authentication here, but remember, every version AH and ESP can be used in tunnel mode or transport mode.
So, up top is the IP diagram, right?
I think the colors make it kind of clear what we're talking about, right?
So, the top is untouched original IP packet.
It has an IPv4 header, and the protocol field says 6.
6 is reserved for TCP, right?
So, that's how it's parsed, right?
When you receive, and this is not really looked at so much or processed by the routers along the internet,
but the receiving host gets an IP packet.
It parses as the IP header, looks at the, oh, it's 6, so I pass it to TCP.
It's something else, I pass it to UDP or ICMP, whatever.
These numbers are reserved.
Okay?
They are part of the standard, IP standard.
Then you have TCP header, you see, right?
And then you have data.
And all of that TCP header plus the dark, dark green,
make IP data, right?
So, as far as IP, that's always IP data.
Okay, so what do we do in the transport mode?
In the transport mode, the IP has stays, right?
As I said, it stays almost the same, but the protocol field changes to 51.
51 is reserved for IP set.
So, what they're receiving those processes.

The header, it says, oh, not going to TCP, not going to GDP, not going to ICMP,
going to IP set, invoke IP set receipt.
Okay?
And then what follows this is the H header, authentication header that you saw earlier.
And inside it, there's all these fields you saw, but you see the next header, 6.
That means the next header is TCP, right?
Because that's how they are laid out.
IP, IPsec, TCP.
I mean, no mystery header, right?
Tunnel mode, by now, should be kind of obvious, right?
There was your original header that goes here.
Intact.
Notice nothing changes.
This whole thing, this whole block is exactly what you see at the top, right?
It's like verbatim, wholesale, taken and encapsulated.
Identification header, stuck here.
Okay?
New IP header in front.
So the packet begins with a new outer IP header.
The protocol is 51.
It means IPsec is next.
So as this header is parsed, IPsec is involved.
Right?
IPsec received.
IPsec looks at this and says, oh, let me authenticate.
First of all, it makes sure that it finds an entry.
It's like a table somewhere that says, I have a key for this connection, right?
It uses SPI to do that.
It looks up.
Now this is the key.
I authenticate.
If authentication fails, done.
Packet is discarded.
Okay?
But if the packet is okay, everything checks out, where do we go?
Next header.
Why four?
Four is the reserve for IP.
So it actually goes IP, IPsec, back to IP.
Because it now processes this as an IP packet.
And IP here says on the X protocol 6, TCP.
So it sounds a little weird, but it holds very robust.
It goes IP, IPsec, IP, TCP.
All within your kernel, right?
Or maybe outside the kernel, depending on which .
Questions?
Pretty straightforward, right?
Okay.
Now it gets a little more tricky, but just a little bit.
ESP.
Redundant, redundant, redundant.

Don't ask me why.
They designed it that way.
My guess is because, well, I kind of was around during that time.
I sort of remembered that the idea was to give people a choice between something very lightweight, like AH.
Which means that you could use upper layer encryption, like you could use something like SSLTLS, whatever, at a higher layer.
But at IP, you should just have basic protection, like integrity and origin authentication.
But everything else, leave it to the higher layer.
So that's one philosophy.
Right?
Another philosophy is protect everything.
And that's more like what ESP does.
So they do overlap.
Meaning that they actually, you could use one to, you know, you can configure one to be almost like the other.
But they do offer also different services.
So it provides everything that AH already offers, but also gives you data confidentiality, which means like actual encryption of data.
The details uses a counter also to detect, replace, and delay, and replace reward packets.
Uses similar integrity check coverage because it also needs integrity.
You cannot, you should not be just using encryption without integrity.
Okay?
But distinctive features, confidentiality, and metadata protection.
Right?
So metadata confidentiality as well, meaning that you can hide the actual endpoints from being used.
And here we go.
Here's the header.
The first thing you see in a header is SPI.
Now let's walk back to the application header.
What do we see?
You see SPI and a sequence number.
Here?
You see SPI, sequence number, but it's preceded by a few things.
So it's not exactly the same.
Yeah?
So you're making it sound like AH doesn't have the ability to protect the earth at the endpoints?
It cannot.
So what's the point of tunnel mode?
What is the point of tunnel mode?
Good question.
Good question.
Yeah?
If you use the tunnel mode, you get double protection maybe?
I mean, you could imagine that...
Let me see.
You could, you could imagine that, like to say you're using some higher layer, like session layer encryption,
and the tunnel mode maybe gives you some, a little extra because you could use maybe a tunnel mode between two gateways

and rely on, on application level security for hand to hand.
But I'm not sure if that, that actually holds water.
So that's one of the things that they, for compatibility, decided to provide both tunnel and transport for AH and ESP.
You had a comment?
Come on.
I just think that tunnel mode may be used for like, things like VPN.
Yes.
Tunnel mode is used by, for VPN.
But what his question is, since tunnel mode does not protect the source and destination addresses, the original source and destination, what is the point?
For AH.
For AH.
For AH, right?
So in AH, what does it actually give?
And I think there is something, there is something maybe that have to do with, ah, you remember, hey, so you have these mutable fields, right?
Remember mutable fields, the fields that change an IV header?
Well, if you tunnel, then you protect the fields of the inner header, everything.
Does that make sense?
Because, but, and it doesn't actually change, right?
Until the end of the tunnel.
But still there is something, there is, I didn't remember there was something.
So that's that.
That you get to protect these fields that normally change an IP.
So, let's just go back for a second.
Right, right.
So, let's look at the transport mode.
In the transport mode, the original IP header, right, is still there.
And it has these fields that change, remember?
Time to live, check some, fragmentation, right?
IHL, they can change.
In transit.
So you cannot really protect them.
It protects other fields, like source, destination, but not those.
Not the ones I have in red, remember?
But with the transport mode, sorry, with the tunnel mode, the former IP header is now here.
And it's protected fully.
Because these fields don't change in here, right?
As the packet moves through the internet.
They change in here.
Right?
Is it worth it?
Eh.
Eh.
It's a little bit of like a tiny delta you get out of it.
If you ask me, I wouldn't use it.
I think like, for example, in the second example, we can maybe defend the tray in the contact mode.
What?
Like the tray of that.
So that the receiver can know where actually this package comes from rather than in the transport

mode.
Well, no, no.
In the transport mode, you do.
You do.
Because the authentication header, I mean, I'm assuming here that the tunnel is end-to-end, not gate-to-end.
Okay?
So let's think of tunnel as end-to-end.
You do know where it comes from.
Because the authentication header in either mode, in either mode, protects the IP header, this IP header, or this IP header, but only the fields that don't change.
And source destination does not change.
Right?
But if you're using it in a gateway-to-gateway, right?
Gateway-to-gateway, like there, what would happen is, so there's no IPsec here, I suppose, no IPsec here, but there's IPsec here.
A will send a packet with IP header that has A, the source, B destination.
When it gets in a transport mode, it still has to be A source, B destination.
But when in tunnel mode, the outer header will say, gateway A source, gateway B destination.
See the difference?
Because it's tunnel, right?
So the outer IP header will have the end points of the IPsec connection, which is in this case, between these two.
And this type can still reach the packet.
Yeah, yeah, yeah, yeah, yeah, yeah, yeah, sure.
Sure.
As I said, AH provides zero confidentiality, only integrity and data and origin of integration.
So in that case, that I just described, gateway B will be able to verify that this packet really came from gateway A.
And that the data has not been modified.
But that's it.
Right.
Okay, so ESP.
So you see the area is a bit different.
The security parameter index followed by sequence number.
I mean, those are the same as in the previous, but the AH and other stuff before.
Then there's payload data.
This is different.
It's encapsulating.
Encapsulating means you can encapsulate, right?
Which means you don't just stick something in the beginning, you stick something in the end.
That's what encapsulation actually means in English, right?
Like you're surrounded.
Therefore, this is the actual payload that you're protecting.
See?
This is the variable length.
Okay?
Then there's padding because you need to align it on a 32-bit boundary.
If it already aligns, no padding.
It's optional, right?
Next header.

Next header.
Next header.
Next header.
Next header.
Same as before.
Followed by authentication data.
This is the variable length.
See?
This is the variable length.
Okay?
Then there's padding because you need to align it on a 32-bit boundary.
There's no padding.
There's no padding.
There's no padding.
It's optional, right?
Next header.
Next header.
Same as before.
Followed by authentication data.
And that's it.
So this is the end of the packet.
Right?
The word is trailer.
Header.
Trailer.
And in between is the packet.
Whereas AH was about header on the trailer.
Right?
That's it.
That's the main difference.
That encapsulation has both headers.
And the packet is stuck in the middle.
And AH just has a header.
So why do we put next header?
Header.
Header.
Trailer.
And in between is the packet.
Where as AH was about header on the trailer.
Right?
That's it.
That's the main difference.
That encapsulation has both header and trailer and the packet is stuck in the middle.
And AH just has a header.
So, why do we put next header here? Because, well, it tells you as you parse the packet, right? As you parse the packet, it tells you, oh, and check this authentication, right? It comes at the end of the packet.
So, it allows you essentially to, as you're receiving the packet, right, kind of keep computing in real time the authentication of that packet.
And then when you start, when you finally receive this, you compare it. So, it doesn't match? Good. Right?

And then it allows you also to right away jump to the next header after that.
But the next header isn't actually here, right? It's not below, right? This is the end.
The next header is actually there. Do you see how this is different?
So, before, the next header was like following, right? But there's nothing after this. This is the end.
The authentication data is the trailers. So, there's nothing after it. The next header is here.
Okay? So, this is the same picture. So, before, except this one is for PSP.
Same original data, original packet at the top.
So, then you have transport mode. Transport mode, the IPv4 header copy. Now, the protocol is 50.
Now, 51, 50 is reserved for ESP.
Okay? Then it's ESP header that you saw previously. Inside the whole TCP, the payload that you saw where they are. You see the, what's it, lilac followed by darker green.
And then, at the end is the ESP trailer, which says, next header, pointer. Okay?
Make sense? The meaning of next header is the same. It's which protocol to invoke next.
And then, the very last thing is the data, the authentication data.
Tunnel mode? Okay. Already, by now, it should be clear. Protocol, 50. New IP header. Brand new IP header.
ESP header. The original IP datagram, untouched, encapsulated, and encrypted. Okay?
And then, you have the ESP trailer, and then D. So, minor difference, right?
But, you protect the entire, you protect the meaning of next header.
Everything here, in this rectangle, is fully encrypted.
And should be, essentially, a black box.
Naturally, why have both, right?
So, I think you already kind of see one answer, right?
One is lightweight.
One of them says, you know what?
I don't deal with encryption.
I just give you the bare bones,
authenticity of origin, and data integrity.
That's it.
And the other one says,
I'm going to give you the whole enchilada.
I'm just going to give you everything you want.
Encrypt everything.
Why is it not using public key?
Well, public key is expensive.
You never want to use public key for bulk data, right?
That's, like, super obvious.
Right?
You always want to use symmetric key cryptography.
And just like SSL, you know, once you establish a connection,
only symmetric cryptography is used, say, here.
So, the way to think about IPsec is really more like
the record layer in SSL-TLS.
Remember that?
It had this record layer, and it had these protocols on top
that are doing the management, right?
Establish reconnection, alert, blah, blah, blah.
Well, IPsec is, like, the data, the record layer in SSL-TLS.
Algorithms, that's not for IPsec to decide, right?
IPsec just provides formats,
and it leaves the choice of algorithms both for computing and integrity check

and for encryption, it leaves those two specific endpoints policy, right?
Decisions.
So, I want you to remember, of course, this is more like, it's more than obvious, right?
That what you get is, at the very least, what I'm saying is authentication, right?
The data integrity, but who does it authenticate, or what does it actually authenticate?
It does not authenticate a human.
It does not authenticate a chair.
It does not authenticate a window on your screen, or a web browser, or a user who is logged in on a particular account.
What it actually authenticates is, like, this specific interface here.
And a given host, like this, as I said before, can have many interfaces, right?
You could have IP over Bluetooth.
You could have IP over wired Ethernet.
You could have IP over Wi-Fi.
You could have IP over ZigBee.
You could have IP over Pigeons, for all I can.
You could have IP over any medium, a data link Mac layer.
And so, it only authenticates the interface.
Okay?
Another important thing, denial of service.
I said nothing about denial of service.
Because IPsec sucks when it comes to denial of service.
And I don't blame it, because if you want security, denial of service is a security in and of itself is denial of service.
Why?
Because think about this.
You have a router, I'm sorry, a gateway, like one of those guys, or a host.
And they have an IP, one or more IP interfaces.
And let's say you're using IPsec, right?
And some adversary says, oh, you're using IPsec.
How nice.
I'm going to send you a barrage of IP packets that look like IPsec, right?
But the authentication data, right?
It's just frivolous.
It's just garbage.
Now, you cannot just send a random packet with random source to a given IP address.
You have to send it or an IP address for which the source already has a connection, right?
That association.
So what you do, you eavesdrop on the communication.
You say, oh, these two hosts, I don't care what they are, our gateways are, all right?
We'll just close.
Yeah, using IPsec.
You can tell this from the headers, right?
Remember, header says 50, 51, IPsec.
You can tell.
Nice.
Well, what you do, you say, okay, I see a packet that uses IPsec, and it has source A destination B.
I'm going to manufacture a giant quantity of packets in real time with source A, not my source, but I'm going to fake it and put source A destination B, IPsec, and the authentication of this garbage.
Right?

You with me?
And send high speed to the victim.
What is the poor receiver going to do?
Well, the receiver, it says, right?
Yes, sir.
IPsec packet received.
I have a connection with B, right?
Verify authentication token.
First thing.
Well, that's not easy.
That requires computing a HMAC.
You might think, okay, HMAC is not a RSA decryption like in TLS SSL, right?
But it's still inexpensive and frivolous operation because it prevents you from actually getting data by actual service.
That's why it's called denial of service.
Essentially, it's like, denial of service is like saying, you know, remember, I think I already mentioned this analogy.
You're trying to exit your house front door and somebody points a fire hose at you.
That's denial of service.
You can't exit your door because there's a fire hose with, like, high-pressure water.
Now, that's one type of denial of service.
If the adversary has a fire hose, what would be an equivalent?
A very high-speed interface that is able to generate a huge amount of traffic directed at the victim.
A version of that is a distributed denial of service.
Much, much craftier, much easier to mount that kind of an attack, which is the other system.
Then I don't have enough bandwidth or my interface is not fast enough to just, like, I have a trickle.
I can point, like, a guarded hose at your door.
But if I get a bunch of zombies or a botnet or out the internet to start sending you low bandwidth IPsec packets,
you follow what I'm saying?
It's like I get everybody in the room to take a guarded hose and point at the door.
20 guarding hoses equals fire hose.
You get the idea.
So, IPsec is actually a pain in the neck for denial of service because it forces the receiver to verify the authentication token.
That means it has to receive the entire packet.
It cannot say halfway.
It's like receiving the header, saying, ah, this doesn't look good.
No, no, no.
It has to receive the entire packet, right, because the A and B already have an IPsec connection.
It receives the entire packet, verifies the header, saying, ah, garbage.
By that time, CPU resources have been consumed.
Bandwidth resources have been consumed, right?
That's denial of service.
An IPsec makes it worse.
Oh, it makes DOS easier.
The other thing to remember, the order of operations.
First, you encrypt, then you authenticate.
So, if encryption is used, and this does not apply to authentication, this applies to ESP, right?
The first thing you do, whether you use transport mode or autonomous mode, it doesn't matter, is that you first encrypt this, right?

Or whether you're doing this or this, the first encrypt, and then you compute the authentication.
Why?
Good example.
Why don't do it the other way around?
Why not authenticate the data, put the authentication token inside here, and then encrypt the whole thing?
Sounds the same.
Yeah, I mean, you see that says the pink box, and at the end it says ESP authentication data.
Why not sleep that pink box inside here, right?
Like, compute the authentication first, and then encrypt also the authentication data.
Any bright ideas, or not so bright ideas, any ideas?
Yeah, it's not, so, kind of, yeah, yay.
And also, yeah.
There is a, today, encryption, in the past, encryption was always more expensive.
Like, remember, I talked about DES, right?
That's, like, a traditional encryption technique, more expensive, like, the order of magnitude.
Because they were, like, in the olden days, they were designed to work very fast on hardware, but work poorly in software.
But, today, modern encryption algorithms are pretty damn fast.
So, they are, like, pretty much the same, you can think of the same speed.
No, the answer is simple.
So, if you do it in the other way, which is, you authenticated and encrypt, you would have to perform decryption first, and then compute the authentication token.
That's double the work before you detect it.
It's cracked.
Right?
Whereas, if you authenticated the packet first, you see what I mean?
If you authenticated it first, and then decrypted, you don't need to worry about whether the decryption would succeed.
Because, you know, because it's authentic, right?
So, unless there was some broken implementation at the sender's end, you know that the decryption is going to work.
You don't have to wonder whether you will or not.
Okay?
So, that's a performance consideration.
Right?
Alright.
We're not going to talk about Ike.
I'm just going to tell you a few things.
So, Ike is, like, this giant standard for Internet Key Exchange.
And it allows, so it's what happens before you start IPsec, right?
Before you can send IPsec packets.
Right?
The keys need to be set out.
However, IPsec also works with pre-installed keys.
So, it can, in fact, like, if you have manually installed cryptographic keys,
have A and B, they don't really need Ike.
They can just, like, start using IPsec.
Okay?
And this is done sometimes with, these days, with Ike.
Well, it used to be done a lot, but people didn't know how to use Ike.

There was no Ike yet.
But it's done also today with, like, IoT devices sometimes.
Well, the manufacturer will pre-install a key or a set of keys,
and then an IoT device doesn't really talk to many hosts, right?
An IoT device might talk like that.
Your Echo, what does it talk to?
If you have an Echo voice assistant.
Who does it talk to?
Amazon.
It's mothership, right?
It doesn't talk to random other devices, really, right?
At least not the flow-end Echo.
So, for those types, they say, I'll just pre-install a key, you know?
So, that might be...
I'm not saying I don't know if Echo's use IPSec, but that's one way.
All right.
So, IPSec, what IPSec gives you is this something called security association.
And security association is where you get the SPI.
Remember the SPI, SPI, security parameters, and the session ID.
You get it based on a security association between post-Aid or IP address,
IP address, and it's always one way.
Remember, right?
One way.
A to B is different from B to A.
And then every host that uses IPSec is supposed to have a database,
really just a table of, oh, sad, security association database.
Okay, where it keeps a record for every security association where it will say,
I know who I am, I'm host A, but, like, there will be an entry.
Host B, what protocol to use, like AH, tunnel mode, what is the key, blah, blah, blah, etc.
Like, lifetime, something like that.
So, SPI, I already told you, right?
So, this is an entry that the SAP database, the security association database,
points you to the SPI.
So, when you receive a packet, right, you are on the receiving side of the IPSec packet,
you, remember, every header has a SPI, right?
Whether it's AH or ASP has a SPI field.
You take the SPI field and you look up that database immediately.
Okay, if you don't find it, toss the packet.
If you find it, that tells you, oh, this is the header I'm using.
So, how do you know which IPSec header, which mode?
Well, the database tells you.
Okay?
So, that's very important.
You know, it's mandatory.
Every IPSec application must maintain the database.
And then it has something called security policy database, which actually governs,
it's more like a, similar to what you will see firewalls do,
it governs, like, which connections require which parameters.
Policies, right?
It says, oh, if you're talking to an internal host, you don't need to use IPSec.
If you're talking to, I don't know, our branch office on the East Coast,

you must use authentication header, transport mode.
Okay?
Or if you're using, if you're talking to, I don't know, a host in a foreign country,
then you must use the host to gateway ESP transport mode, something like that.
It's a policy.
Something your security, every organization's security administrator configures.
All right, so it kind of fits together like this.
Then, of course, the blob is the internet.
And so that each host will have its own database of security associations
that will be governed by the security policy database.
Well, I think I'll just end on this, because I don't really want to go for it, Ike.
The idea is, the main thing you need to know about Ike is that it establishes a secure channel,
like, outside, on the side, like a control channel,
and it establishes a set of keys.
Okay?
And then, after that, we do like to say.
So Ike has its own message format.
It does not, this is where it differs from SSLTLS.
In SSLTLS, all the messages go for the record player, if I remember.
Well, in Ike, they don't.
Ike has its own message format.
And so that, this is like the example.
It might be helpful.
So in this case, we have host A, kind of like what I drew on the board earlier.
Host A, and then two gateways, right?
The blue arrow is the internet, right?
And then you have host B.
And so, in this case, A and B use a transport mode of either H or ESP, like end-to-end.
And in addition to that, the gateways between themselves use a tunnel mode to hide,
among other things, the IP addresses of the internal host.
So if we look at, like, inside the host A, its security policy database might say something like,
oh, if I have, if I'm talking to host B on any protocol, using any protocol, and using any port,
remember, any transport, when protocol means any transport, or about IP protocol,
and any port, I should use authentication header, IPsec, with HMAC MD5,
that is MD5, so HMAC instantiated with MD5 hash function to protect, to compute the
authentication data.
Okay, that's a policy database, it contains no keys, it does not refer to any active connection,
it is a general policy.
But the SAD, the security association database, will say, ah, currently, there is a connection
between A and B, and the protocol we use is AH, as policy dictates, right?
And the SPI is 12, so that's our session ID, and our, we're using HMAC MD5,
and probably what I don't show here is that the actual key will be stored there, too.
Okay?
Now, the gateways database, right, this is gateway C, will say something like,
oh, anybody from subnet, this refers not to A, but subnetwork, right?
That's what subnet is, right?
So anybody on A's subnet, talking to B's subnet, on using any protocol on any port,
must use ESP with triple tensing, or encryption.
And tunnel destination is D, that's our friend D over here, right?
That's the endpoint of the IPC tunnel.
So that, again, is policy, does not refer to any existing connection,

OpenAI

doesn't matter if any exists or not, this is the rule.
The actual, currently, existing connection is,
there is an A sub to B sub, anybody from A to B,
there is an existing connection using ESP,
SPI is 14, there's a number that I picked,
triple tensing, the key is there.
Okay, what does that mean?
If there was another host, A prime,
talking to another host, B prime,
they would use the same tunnel.
Nice, right?
They all fit in the same tunnel.
No reason that C and D should have multiple tunnels.
Everything is within the same tunnel.
So, an adversary who listens on that blue arrow on the internet,
it cannot distinguish between,
he does not know how many hosts are talking to,
how many other hosts at all.
The only thing they will see is the volume of traffic.
The only other thing they will see,
in addition to the volume of traffic,
is the timing of that traffic.
But what,
this is not part of IPC,
but what smart people do,
if they're really concerned about traffic analysis,
and hiding,
like, for example,
in the morning, the volume goes up,
in the afternoon, maybe at lunch, it goes down, right?
You might not think this is important,
but this is very important sometimes,
especially, like, in critical application,
defense, you know, et cetera.
What they do is they pat the traffic.
Meaning that C and D might send trash to each other,
just garbage,
to make it seem like real hosts are communicating.
Make sense?
No?
To obscure the patterns.
On that note,
we're done.
For today.
So remember,
I strongly urge you to show up on Tuesday
for the guest lecture.
Decision is always true.

Thanks for sharing this detailed lecture transcript — it confirms and deepens many of the core IPSec concepts you've already been exploring.

Here's a concise distillation of the **main takeaways** from your professor's talk, organized by theme and enriched with clarifications where needed:

🔐 1. Minimum Security Guarantees in IPSec

- **Mandatory:**
  - ✅ Origin authenticity
  - ✅ Data integrity
- **Optional:**
  - 🔒 Data confidentiality
  - 🔒 Metadata confidentiality (via tunneling)

IPSec can operate without encryption (like AH-only mode), but **never without authentication and integrity**.

🧱 2. Modes of Deployment

- **Host-to-Host (e.g., A ↔ B)**: End-to-end IPSec; typically uses transport mode.
- **Host-to-Gateway (e.g., Laptop ↔ Office Gateway)**: Remote access VPN.
- **Gateway-to-Gateway (e.g., East ↔ West Coast routers)**: Site-to-site VPN, typically tunnel mode.
- **Composite setups:** You can combine layers (e.g., A → GW_A → GW_B → B), using multiple nested IPSec sessions.

🚇 3. Tunnel vs Transport Mode

Feature	Transport Mode	Tunnel Mode
Protects IP header	❌ No (original IP header is visible)	✅ Yes (entire original IP packet is encapsulated)
Adds new IP hdr?	❌ No	✅ Yes
Typical use case	Host-to-Host	Gateway-to-Gateway / VPN
Benefit	Lower overhead	Hides internal network structure (traffic analysis protection)

Tunnel mode helps prevent traffic analysis by hiding internal host addresses — especially critical in **VPNs between enterprise sites**.

🛡️ 4. AH vs ESP

Feature	AH (Authentication Header)	ESP (Encapsulating Security Payload)
Data integrity	✅ Yes	✅ Yes

Feature	AH (Authentication Header)	ESP (Encapsulating Security Payload)
Origin authentication	✅ Yes	✅ Yes
Encryption (confidentiality)	❌ No	✅ Yes
Metadata confidentiality	❌ No (IP hdr exposed)	✅ Yes (with tunnel mode)
Typical usage	Lightweight auth-only scenarios	General use, especially in VPNs

- AH authenticates **header + payload** (excluding mutable fields), but offers **no secrecy**.
- ESP **encrypts + authenticates**, making it more complete and widely used.

🔄 5. Why Encrypt-then-Authenticate in ESP?

- Ensures authentication happens **before** expensive decryption.
- Prevents attacks that exploit decryption errors (like padding oracle).
- Allows early rejection of tampered packets.

Your professor confirms: this is done for **performance and security**.

📁 6. Security Associations (SA) and Policy

- **SA (Security Association):** One-way channel, defined by:
  - SPI (Security Parameter Index)
  - Keys, algorithms, mode (AH/ESP), lifetime
- **SAD (Security Association Database):** Holds active SAs.
- **SPD (Security Policy Database):** Defines rules like:
  - "All Gmail traffic must use ESP + 3DES"
  - "No IPsec for internal traffic"

These are analogous to firewall rules, but for security protocol enforcement.

💣 7. IPSec and Denial of Service (DoS)

- Vulnerable to DoS because it forces receivers to **verify authentication tags**.
- An attacker can:
  - Spoof legitimate-looking packets with invalid MACs.
  - Exhaust CPU/bandwidth on gateways.
- **IPSec cannot protect against DoS**, and may even **exacerbate it**.

🔍 8. AH Tunnel Mode — Is It Useless?

Your professor directly addresses this:

- No confidentiality (so IPs are still visible).
- But **inner IP header is authenticated fully**, including fields that **change in transit** (e.g., TTL).
- So **integrity of mutable fields is preserved**, which can matter in edge cases.
- Verdict: Minor benefit, rarely justifies use over ESP.

📦 9. Use of IKE (Internet Key Exchange)

- IKE is a **separate control protocol**.
- Handles:
  - Key exchange (via Diffie-Hellman)
  - SA negotiation (via SPD)
  - Multiple modes (Main, Aggressive, Quick)
- Your professor parallels it to SSL/TLS handshake phase.

IKE is **not mandatory** if keys are manually configured (common in IoT or embedded systems).

Would you like me to diagram the composite setup (e.g., A ↔ GW_A ↔ GW_B ↔ B) with tunnel vs transport layers labeled?

explain the diff between ip4 and ipv6 header

Great — let's break down the **differences between IPv4 and IPv6 headers**, focusing on both the structure and the security implications (especially for IPSec).

📦 1. Overall Design Philosophy

Feature	IPv4	IPv6
Designed in	Late 1970s–1980s	1990s–2000s
Address size	32 bits (4.3 billion addresses)	128 bits (vast address space)
Header size	Variable (20–60 bytes)	Fixed (40 bytes)
Header complexity	Many optional fields and fragmentation	Simplified; moved options to extensions

---

🧱 2. Structural Differences in the Header

✅ IPv4 Header Fields

Field	Size	Notes
Version	4 bits	Always **4** for IPv4

Field	Size	Notes
Header Length (IHL)	4 bits	Length of header (since it's variable)
Type of Service (ToS)	8 bits	Used for QoS (quality of service)
Total Length	16 bits	Size of entire packet (header + data)
Identification	16 bits	Used for fragmentation
Flags	3 bits	Fragment control
Fragment Offset	13 bits	Position of fragment
Time to Live (TTL)	8 bits	Packet lifetime
Protocol	8 bits	Indicates next layer (e.g., TCP=6, UDP=17, IPsec=50/51)
Header Checksum	16 bits	Verifies header only
Source IP Address	32 bits	Origin address
Destination IP Address	32 bits	Target address
Options + Padding	Variable	Rarely used, increases complexity

✅ IPv6 Header Fields

Field	Size	Notes
Version	4 bits	Always **6** for IPv6
Traffic Class	8 bits	Like IPv4's ToS
Flow Label	20 bits	Used for labeling packet flows for QoS
Payload Length	16 bits	Length of data after the header
Next Header	8 bits	Replaces Protocol field (can point to extension headers or TCP)
Hop Limit	8 bits	Replaces TTL
Source IP Address	128 bits	Larger address space
Destination IP Address	128 bits	Larger address space

✅ **No fragmentation fields, no checksum field.** These were removed for **simplicity and performance**. Fragmentation is handled only by the source host, not routers.

🔐 3. Implications for IPSec

Aspect	IPv4	IPv6
IPSec integration	Optional (not all IPv4 stacks support it)	**Built-in** and mandatory for IPv6 compliance (though often unused)
Header mutability	Many fields mutable (frag offset, TTL)	Fewer mutable fields (cleaner for AH usage)
Fragmentation	Happens in-network (more DoS vectors)	Handled only at source; simplifies auth
Checksum requirement	Must be recalculated per hop	Removed to save router CPU and simplify IPSec

IPv6 was **designed with IPSec in mind**, making it easier to secure and authenticate packets consistently.

🧠 Summary

Feature	IPv4	IPv6
Header size	Variable (20–60 bytes)	Fixed (40 bytes)
Fragmentation	Handled by routers	Only by source host
Checksum	Yes (header only)	❌ Removed
IPSec support	Optional	Mandatory in specification
Address size	32 bits	128 bits
Extension support	Limited, clunky	Cleanly modular via extension headers

Would you like a side-by-side diagram of the two header formats?

explain slide 15-17

Let's walk through **Slides 15–17** of your IPSec lecture PDF, using both the slide content and your professor's detailed explanations to unpack everything clearly.

🔹 Slide 15: IPsec Architecture — Typical Use-Cases: Modes

🔍 What's Shown:

- Two simple network diagrams:
  1. **Transport Mode**: Shows two end-hosts connected directly.
  2. **Tunnel Mode**: Shows two routers/gateways in between.

🔑 Meaning:

- **Transport Mode** is used in **host-to-host communication**:
  - IPSec runs on the **end devices** (e.g., A ↔ B).
  - Only the **payload** (e.g., TCP, UDP) is protected.
  - The **original IP header remains visible**.
- **Tunnel Mode** is used for **gateway-to-gateway or VPN-style connections**:
  - IPSec runs on **border routers/gateways**.
  - The **entire original IP packet (header + data)** is encapsulated and protected.
  - A **new outer IP header** is added for routing.
  - Hides internal source/destination IPs (helps against **traffic analysis**).

This slide introduces the **two IPSec modes**, and aligns them with **who runs IPSec** — hosts or gateways.

🔹 Slide 16: IPSec Modes — Summary

🔍 Visual Breakdown:

This slide shows two formats:

1. Transport Mode:

```css
[IP header][ESP/AH][Data]
```

- Original IP header is **kept**.
- ESP or AH is inserted **between IP header and payload**.
- Only the **data** is encrypted (with ESP) or authenticated (with AH).

2. Tunnel Mode:

```css
[New IP header][ESP/AH][Original IP header][Data]
```

- A **new outer IP header** is added.
- The **entire original IP packet** (header + data) is wrapped and protected.
- Only the new header is visible on the network.

🧠 Your Professor's Emphasis:

- **Tunnel Mode** is more secure:
  - It hides **source/destination IPs** from eavesdroppers.
  - Used in **VPNs**, enterprise-to-enterprise traffic, etc.
- **Transport Mode** is lighter-weight:
  - Less overhead.
  - But exposes IP metadata.

🔹 Slide 17: Diagram — IPSec Header Placement

This slide shows a more detailed layout of actual packet structures in **Transport** and **Tunnel** modes.

✅ Transport Mode Packet:

| IP Header | AH/ESP Header | TCP/UDP Header | Data |

- The IP header is **not encrypted**.
- The transport header and payload **can be encrypted** (ESP) or authenticated (AH).
- Used for **end-to-end security**.

✅ Tunnel Mode Packet:

| New IP Header | AH/ESP Header | Old IP Header | TCP/UDP Header | Data |

- The **entire original IP packet** is encapsulated.
- Useful when the **gateways apply security**, and internal IPs should remain hidden.
- **New IP header** contains gateway addresses.

🧠 Professor's Explanation Highlights:

- In **transport mode**, IP addresses are exposed, which is a privacy risk.
- In **tunnel mode**, observers **see only the gateways' IPs**, not internal hosts.
- Tunnel mode is **preferable when metadata confidentiality matters** (e.g., VPNs).

🧠 Summary: Slides 15–17

Feature	Transport Mode	Tunnel Mode
Who runs it?	End hosts	Gateways or routers
What is hidden?	Only payload (ESP)	Entire original packet (IP hdr + payload)
Use case	Host-to-host communication	Site-to-site VPN, hiding internal topology
Metadata leak?	Yes — IP addresses exposed	No — IP addresses are encapsulated
Packet format	**`[IP][IPsec][TCP][Data]`**	**`[New IP][IPsec][Orig IP][TCP][Data]`**

Let me know if you want these diagrams redrawn for clarity or turned into a visual chart.

answer the questions in slide 27

Absolutely — let's answer the questions listed on **Slide 27** of your IPSec lecture:

🔹 Slide 27: Questions

1. Why have both AH and ESP?

✅ Answer:

- They offer **different levels of protection**:
  - **AH (Authentication Header)**:
    - Provides **data integrity** and **origin authentication**.
    - Authenticates both **payload and some parts of the IP header**.
    - **Does not** provide confidentiality (no encryption).

- **ESP (Encapsulating Security Payload)**:
  - Provides **data confidentiality** (encryption).
  - Also provides **data integrity** and **origin authentication**.
  - **Does not protect the IP header** unless used in tunnel mode.

➡️ **Why both?**

- AH is **lighter-weight**, suitable when you just want integrity and authentication.
- ESP is **more complete**, but comes with added overhead.
- Sometimes, you **only want to authenticate**, not encrypt (e.g., for compliance or performance reasons).

---

2. Both AH and ESP use symmetric key–based algorithms. Why not public-key cryptography?

✅ **Answer:**

- **Public-key cryptography is too expensive** for bulk data:
  - Slow, computationally intensive.
  - Not suitable for high-speed, per-packet processing.
- Symmetric cryptography (e.g., AES + HMAC) is **much faster** and scalable for large volumes.

➡️ **Public key crypto is still used**, but only:

- **During key exchange** (e.g., IKE uses Diffie-Hellman + digital signatures).
- **Not** for encrypting the actual packet payload.

---

3. How are the keys being exchanged?

✅ **Answer:**

- Through the **IKE (Internet Key Exchange)** protocol:
  - Establishes shared symmetric keys securely.
  - Authenticates parties (via pre-shared key, digital signature, or public-key encryption).
  - Two phases:
    - Phase 1: Builds a secure control channel.
    - Phase 2: Negotiates the actual IPSec session keys.

➡️ Keys can also be **manually configured** (e.g., pre-shared keys in static environments like IoT).

---

4. What algorithms should we use?

✅ **Answer:**

- Depends on the **security policy** and performance trade-offs.
- Common algorithms:
  - **For encryption** (ESP): AES (recommended), 3DES (legacy)
  - **For integrity** (AH/ESP): HMAC with SHA-256 or SHA-1 (MD5 deprecated)
- Defined by **security policies** in the SPD (Security Policy Database).

➡️ Like TLS, IPSec is **algorithm-agile** — implementations can negotiate suites.

---

5. **Similar to deciding on the ciphersuite in SSL?**

✅ **Answer: Yes, exactly.**

- Just like SSL/TLS, IPSec endpoints negotiate:
  - Which encryption algorithm to use.
  - Which integrity check to apply.
  - Which Diffie-Hellman group (for key exchange).
- This negotiation is handled by **IKE Phase 1/2**, and depends on the organization's **policy**.

Let me know if you'd like examples of commonly used IPSec policy definitions or cipher configurations!

explain slide 28

Absolutely — let's explain **Slide 28** from your IPSec lecture PDF, integrating both the slide and your professor's detailed commentary.

🔹 Slide 28: Discussion

Slide Bullet Points:

1. **IPSec authenticates machines/hosts, not users**
2. **Does not stop denial-of-service attacks**
3. **In fact, makes DoS easier**
4. **Order of operations: encryption/authentication**

✅ 1. IPSec authenticates machines/hosts, not users

- **What it means:**
  IPSec authenticates the **IP interfaces** of communicating endpoints — **not** the user behind the device.

  - Example: IPSec will tell you “this packet came from IP address X,” but **not** who the logged-in user was.
- **Why it matters:**
  This limits IPSec in scenarios where **user-level granularity** is needed (e.g., per-user access control).
- **Your professor’s metaphor:**
  IPSec authenticates **interfaces**, like "WiFi IP" or "Ethernet IP," not actual human users, accounts, or software processes.

❌ 2. IPSec does not stop Denial-of-Service (DoS) attacks

- **Why:**
  DoS attacks aren’t about data being intercepted — they’re about **overloading the target**.
- **IPSec doesn't have built-in DoS defenses** like rate-limiting or client verification before processing.

⚠️ **3. IPSec can make DoS easier

- **How so:**
  - Attackers can flood a host/gateway with **fake IPSec packets** (e.g., forged ESP/AH packets).
  - The receiver must **verify each one**, which involves **expensive HMAC or decryption**.
  - That consumes **CPU and bandwidth**, **even if the packet is ultimately discarded**.
- **Example (from professor):**
  - Attacker sees that Host A ↔ Host B are using IPSec.
  - They spoof packets with A's IP as the source, but include **invalid authentication data**.
  - The target still **computes HMAC** to verify it → wasted effort → **DoS vector**.

🔁 4. Order of operations: encryption then authentication

- IPSec follows the **Encrypt-Then-Authenticate (EtA)** model:
  - First encrypt the payload.
  - Then compute the authentication tag over the **ciphertext**.
- **Why this order?**
  - If you **authenticate first and then encrypt**, the receiver must **decrypt before checking** integrity.
  - That’s inefficient and unsafe (e.g., padding oracle attacks in TLS 1.0/1.1).
  - EtA allows the receiver to **drop invalid packets before decrypting**, saving effort and improving security.

🧠 Summary of Slide 28

Key Point	Explanation
Host not user authentication	Authenticates network endpoints, not individuals
No built-in DoS protection	IPSec alone can't prevent bandwidth/CPU exhaustion attacks
May worsen DoS risk	Attackers can send spoofed packets with bad HMACs, forcing expensive checks
Encrypt-then-Auth ordering	Ensures authentication is verified before decryption for efficiency & safety

Let me know if you'd like a diagram of the DoS attack scenario or the EtA process!

explain slide 32-34

Let's explain **Slides 32 to 34** of your IPSec lecture, based on both the slide content and your professor's detailed lecture. These slides focus on how IPSec **Security Associations** work and how they are managed via two key databases: the **SPD (Security Policy Database)** and the **SAD (Security Association Database)**.

🔹 Slide 32: How They Fit Together

📦 What's in the Slide:

A visual diagram showing:

- The **SPD** (Security Policy Database)
- The **SAD** (Security Association Database)
- Multiple **Security Associations (SAs)** indexed by **SPI**

🧠 Key Concepts:

Term	Meaning
SA (Security Association)	A one-way IPSec connection with specific cryptographic parameters.
SPI (Security Parameter Index)	A unique identifier for an SA, included in the AH/ESP header.
SPD (Policy)	Says *what* to secure and *how* (e.g., use ESP with AES for Gmail traffic).
SAD (Active SAs)	Tracks *current* IPSec sessions and associated keys, modes, SPIs, etc.

> 🔑 The **SPD** decides **when and how** IPSec is applied, while the **SAD** tracks **active secure connections**.

🔹 Slide 33: IPSec and IKE in Practice

🖼️ What It Shows:

A network diagram with:

- **Certificate Authority** issuing **digital certificates**

- Gateways establishing **secure encrypted tunnels**
- Use of **ISAKMP** and **IKE** for session setup
- Internal networks behind each gateway

🔧 Breakdown:

1. **ISAKMP/IKE** (the control channel):
    - Establishes the secure tunnel (key exchange, SA negotiation).
    - Analogous to the handshake phase in SSL/TLS.
2. **SA**:
    - Defined by SPI, IP pairs, mode (tunnel/transport), algorithm, and keys.
3. **Certificate Authority (CA)**:
    - Optional, but used if **digital signatures** are needed to authenticate endpoints.

    🧠 **IKE + CA** = secure identity validation and dynamic key setup.
    After this setup, the IPSec **data channels** use **ESP or AH**.

🔹 Slide 34: SPD and SADB Example

📋 This slide shows actual database entries.

1. A's SPD (Policy Rules)

From	To	Protocol	Port	Policy
A	B	Any	Any	AH with HMAC-MD5

- Tells A: "If sending to B, use AH in transport or tunnel mode with HMAC-MD5."

2. A's SAD (Security Associations)

From	To	Protocol	SPI	SA Record
A	B	AH	12	HMAC-MD5 key info

- Says: "A currently has an active AH SA with B using SPI = 12."

3. C's SPD (another gateway)

From	To	Protocol	Port	Policy	Tunnel Destination
Any	Any	ESP	Any	3DES encryption	D

- Gateway C's policy: "Tunnel all traffic to D using ESP and 3DES."

4. C's SAD

Protocol	SPI	SA Record
ESP	14	3DES key and options

- Active SA: uses SPI 14 for ESP encryption to D with 3DES.

🧠 What This All Means:

- **SPD** defines *what should happen* when sending/receiving traffic.
- **SAD** keeps track of *how it is currently happening*, via SPIs.
- **SPIs** are how incoming IPSec packets are matched to the correct decryption and verification key.
- Multiple hosts can use **the same tunnel** (e.g., many A_sub to many B_sub connections through a single SA between gateways).

🔑 Summary of Slides 32–34

Component	Role
SPD	Security rules & policies — what to protect and how
SAD	Tracks current security associations with parameters and keys
SPI	Index used in packet headers to look up correct SA in SAD
IKE	Protocol that sets up the keys and SAs dynamically
Tunnel Reuse	One tunnel (SA) can carry many internal host connections securely

Would you like a diagram showing how a packet flows from host A to host B through these IPSec layers using SPD/SAD?

explain phase 1 of IKE in detail

Absolutely — let's walk through **Phase 1 of the Internet Key Exchange (IKE)** protocol in detail, based on the slides, your professor's lecture, and standard IKEv1 behavior.

🔐 Goal of IKE Phase 1

> Establish a secure, authenticated, and private channel between two IPSec endpoints (e.g., hosts or gateways) — this is called the **IKE Security Association (IKE SA)**.

This channel will then be used to **securely negotiate further SAs for actual IPSec data traffic** (in Phase 2).

🧱 What Phase 1 Provides

- **Mutual authentication**

- **Key exchange** via **Diffie-Hellman (DH)**
- **Anti-replay** via nonces
- **Protection against eavesdropping and tampering**

🔁 Two Modes in Phase 1

Mode	Messages	Features
Main Mode	6	More secure, hides identities
Aggressive Mode	3	Fewer messages, faster but leaks identity

Let's focus on **Main Mode**, since it's the standard/default and more secure.

🧭 Main Mode Message Exchange (6 Messages in 3 Rounds)

Round 1: SA Proposal Exchange

1. **Initiator → Responder:**
   `[Header, SA1]`
   - Proposes algorithms: encryption, hash, DH group, authentication method.
2. **Responder → Initiator:**
   `[Header, SA2]`
   - Accepts or modifies the proposal.

🔐 This establishes a **cryptographic vocabulary**: "Here's how we're going to talk."

Round 2: Key Exchange + Nonces

3. **Initiator → Responder:**
   `[Header, KEi, Ni, [Cert_Req]]`
   - Sends:
     - **KEi**: Initiator's DH public value
     - **Ni**: Initiator's random nonce
     - Optional: certificate request
4. **Responder → Initiator:**
   `[Header, KEr, Nr, [Cert_Req]]`
   - Sends:
     - **KEr**: Responder's DH public value
     - **Nr**: Responder's random nonce

🔐 This allows both sides to compute the **shared DH key** and ensures **freshness** (nonces prevent replay).

Round 3: Identity + Authentication

5. **Initiator → Responder:**
   `[Header, IDi, {CERT}, SIG_I]`
   - Sends:
     - **IDi**: Identity of the initiator (e.g., IP address, DNS name)
     - **{CERT}**: Optional certificate
     - **SIG_I**: Signature or MAC proving possession of key
6. **Responder → Initiator:**
   `[Header, IDr, {CERT}, SIG_R]`
   - Sends:
     - **IDr**: Responder's identity
     - **SIG_R**: Signature or MAC authenticating responder

🔐 These signatures authenticate the endpoints and bind identities to the DH key exchange (i.e., prevent MITM).

✅ At the End of Phase 1

- Both parties:
  - Have a **shared DH key**
  - Have authenticated each other
  - Have an **IKE SA** (a secure tunnel for Phase 2 negotiations)

This IKE SA is then used to:

- Protect messages in Phase 2
- Negotiate and install **IPSec SAs** for AH/ESP

🧠 Important Properties of Phase 1

Feature	Provided By
Confidentiality	Encryption of DH exchange & IDs
Integrity	Signatures or MACs
Replay protection	Nonces
Forward secrecy	Diffie-Hellman
Identity privacy	Only in **Main Mode** (Aggressive Mode leaks ID)

⚙️ Authentication Methods Supported

IKE Phase 1 supports four types of authentication:

1. **Pre-shared key (PSK)**
2. **Digital signature (RSA/DSA)**
3. **Public-key encryption (RSA)**
4. **Revised public-key encryption**

➡️ These are configured in the **security policy**, and define how SIG_I and SIG_R are computed and verified.

Would you like a visual message flow chart showing these six messages and their contents?

did the prof discuss ipsec policy from slide?

Yes — your professor did discuss **IPSec policy** as shown on **Slides 53–56**, and elaborated in quite a bit of detail during the lecture. Here's a structured summary combining both the **slide content** and the **professor's remarks**:

🧩 What Is an IPSec Policy?

An **IPSec policy** governs:

- **When** IPSec should be used,
- **What kind** of IPSec (AH or ESP),
- **Which cryptographic algorithms** to apply,
- And **between which endpoints** (e.g., IP addresses, ports, subnets).

Your professor called it a kind of **"organizational security rulebook"**, and emphasized that although not strictly mandatory, it is **essential in practice**.

🗃️ Where Is It Stored?

- In the **Security Policy Database (SPD)**
- Each host or gateway that uses IPSec maintains its **own SPD**.
- It's separate from the **SAD (Security Association Database)**, which stores live connection state.

🧱 Slide 53: Phase 1 Policy Format

- Each **Phase 1 policy** is defined as a **protection suite**, which includes:
  - **Encryption algorithm** (e.g., AES, 3DES)

- **Hash algorithm** (e.g., HMAC-SHA1)
- **Authentication method** (e.g., pre-shared key, certificate)
- **Diffie-Hellman group** (controls key strength)

✅ Optional fields:

- **Lifetime** of the SA
- Other crypto attributes

🧱 Slide 54: Phase 2 Policy Format

- Defined as **proposals**, which may contain:
  - **AH sub-proposals**
  - **ESP sub-proposals**
  - **IPComp sub-proposals** (for compression, not discussed)

Each proposal can specify:

- Key length
- Mode (transport/tunnel)
- Lifetime
- Other attributes

📑 Slide 55–56: Examples

Example 1 (Slide 55)

> All traffic to **`128.104.120.0/24`** must:

- Use **pre-hashed key** authentication
- Use **DH group MODP (1024-bit)**
- Use **HMAC-SHA** (128-bit key)
- Encrypt with **3DES**

This is encoded in policy as:

```csharp
[Auth = Pre-Hash; DH = MODP(1024); HASH = HMAC-SHA; ENC = 3DES]
```

Example 2 (Slide 56)

> All traffic to **`128.104.120.0/24`** must use one of:

- **AH** with HMAC-SHA
- **ESP** with:
  - 3DES + HMAC-MD5
  - or 3DES + HMAC-SHA

This is encoded in policy as:

```csharp
[AH: HMAC-SHA] or
[ESP: (3DES and HMAC-MD5) or (3DES and HMAC-SHA)]
```

🧠 Professor's Commentary Highlights:

- IPSec policies are like **firewall rules**, but instead of dropping packets, they **enforce security protocols**.
- **Internal traffic** may be exempt (policy says "no IPSec").
- **External traffic**, like to Gmail, may require **ESP in tunnel mode**.
- These policies **govern what kind of SA gets negotiated** by IKE.
- If traffic matches **no rule**, it's either blocked or bypassed — based on default policy.

Would you like a visual example showing how a packet matches against the SPD and selects a policy?

a sample question from prof
Suppose that, starting today, UCI is blocking all packets to/from your favorite streaming service using a
firewall that works by checking source and destination IP addresses and discarding packets if either the
source (of incoming) or destination (of outgoing) packets are in that streaming service.
To survive for the remainder of the quarter, you must find a way to bypass this restriction. Explain how IPSec
can help you to bypass the firewall and regain access to your favorite streaming services. Which IPSec mode
(Tunnel or Transport) is more appropriate considering this goal and why?

Great — let's walk through how to **answer this question** based on your understanding of IPSec and your professor's expectations.

📘 Question Recap (Simplified)

- UCI blocks traffic to/from your favorite streaming service by filtering **IP addresses** at the firewall.
- You need to **bypass** this restriction.

- You're allowed to use IPSec.
- You must:
  - Explain **how IPSec helps** bypass the firewall.
  - Decide **which mode** (Tunnel or Transport) is appropriate, and **why**.

✅ Model Answer

1. How IPSec Helps Bypass the Firewall

UCI's firewall blocks traffic based on the **source or destination IP address** of the packets.
This works because IP addresses in normal (unencrypted) packets are **visible** to the firewall.

However, if I set up an **IPSec VPN** connection to an external server (e.g., a VPN server in a different location), I can encapsulate my streaming traffic inside an **IPSec tunnel**. The packets would now:

- Appear to be going **to the VPN server**, not to the streaming service.
- Be **encrypted and encapsulated**, hiding the original destination IP address (i.e., the streaming server).

Thus, the firewall **only sees encrypted traffic** going between my device and the VPN server — it **cannot see** the actual destination IPs (the streaming service).
As a result, **the firewall cannot block** my traffic based on IP addresses — I have successfully bypassed the restriction.

---

2. Which IPSec Mode Is More Appropriate: Tunnel or Transport?

✅ **Tunnel Mode** is more appropriate for this scenario.

Why Tunnel Mode?

- **Tunnel Mode encrypts the entire original IP packet**, including:
  - The original IP header (which contains the streaming server's IP address)
  - The payload (e.g., video stream data)
- It then wraps the packet in a **new IP header**, addressed to the VPN server.
- The firewall sees only:

  ```pgsql
  [New IP header: me → VPN server][Encrypted IPSec tunnel]
  ```

  Not:

  ```csharp
  [IP header: me → streaming service]
  ```

If we used **Transport Mode**, the **original IP header remains visible**.
The firewall would still see you're talking to a streaming service and **block the packet**.

🧠 Summary

Point	Answer
What does IPSec do?	Hides destination IP via encryption
How does it help?	Prevents firewall from identifying/blocking streaming IPs
Which mode to use?	✅ Tunnel Mode
Why?	Hides the entire original IP header

Let me know if you'd like a visual diagram of how the firewall sees packets in tunnel vs transport mode.