PennState

CSE 543: Computer Security
Module: Network Security

Prof. Syed Rafiul Hussain
Department of Computer Science and Engineering
The Pennsylvania State University

Communication Security

- Want to establish a secure channel to remote hosts over an untrusted network
  - Users - when logging in to a remote host
  - Applications - when communicating across network
  - Hosts - when logically part of the same isolated network
- The communication service must …
  - Authenticate the end-points (each other)
  - Negotiate what security is necessary (and how achieved)
  - Establish a secure channel (e.g., key distribution/agreement)
  - Process the traffic between the end points

- Also known as *communications security*.

Users' Communications Security

- Login to a host over an untrusted network
  - Using unauthenticated login - telnet, rsh - up to this point
- Problems
  - How does user authenticate host?
  - How does host authenticate user?

SSH (Secure Shell)

- Secure communication protocol…
  - Between user's client and remote machine (server)
  - Used to implement remote login
  - Runs on any transport layer (TCP/IP)
- Setup
  - Authentication agent on client
    - To produce and process messages on behalf of user
  - SSH Server
    - To handle user logins to that host
    - Forward X and TCP communications
- Remote machine use approximates local machine

SSH Protocol

- How to authenticate server-user and user-server?
  - Users lack public keys
  - But, servers may hold login passwords of users

SSH Protocol

- How to authenticate server-user and user-server?
  - Users lack public keys
  - But, servers may hold login passwords of users
- How to establish a secure channel?
  - Between the client and server
  - For remote processing of commands

SSH Protocol

- (1) Client opens connection to server
- (2) Server responds with its host key and server key
  - Public keys identifying server and enabling communication
- (3) Client generates random number and encrypts with host and server keys
- (4) Server extracts random number (key) and can use
  - Server is authenticated
- (5) Server authenticates user
  - Password and RSA authentication
- (6) Preparatory phase
  - To setup TCP/IP, X11 forwarding, etc.
- (7) Interactive session phase

SSH Protocol

- How to authenticate server-user and user-server?
  - Users lack public keys
  - But, servers may hold login passwords of users

- **Answer**:

SSH Protocol

- How to authenticate server-user and user-server?
  - Users lack public keys
  - But, servers may hold login passwords of users

- **Answer**: Server public keys (host and server) and user passwords

- How are we sure that these are the legitimate public keys for the server?

SSH Protocol

- How to authenticate server-user and user-server?
  - Users lack public keys
  - But, servers may hold login passwords of users
- How to establish a secure channel?
  - Between the client and server
  - For remote processing of commands

- **Answer**:

SSH Protocol

- How to authenticate server-user and user-server?
  - Users lack public keys
  - But, servers may hold login passwords of users
- How to establish a secure channel?
  - Between the client and server
  - For remote processing of commands

- **Answer**: Client chooses key

- How does client know what kind of key to pick?

SSHv2 Protocol

PennState

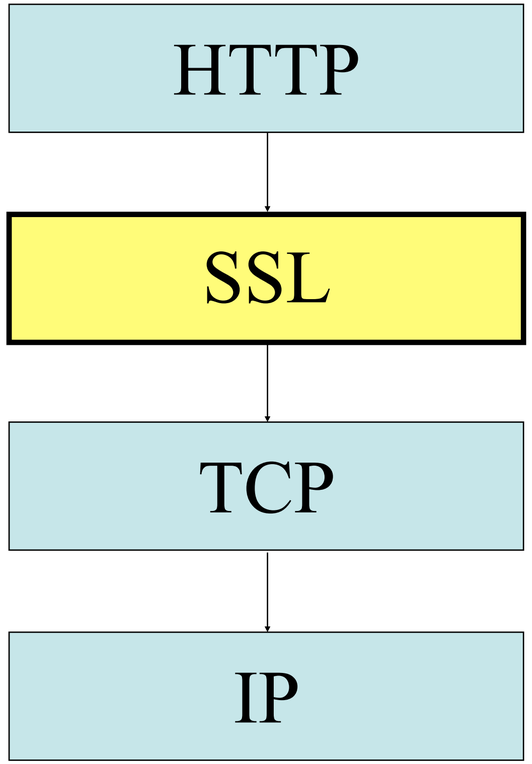
- A number of improvements were made to the SSHv2 protocol (see Section 5)
  - Stronger use of crypto - better algorithms
  - Performance - 1.5 round trips on average
  - Prevent eavesdropping - encrypt all SSH traffic
  - Prevent IP spoofing - always validates server identity
  - Prevent hijacking - integrity checking using HMAC
- Not backwards compatible with SSHv1

Application Comm Security

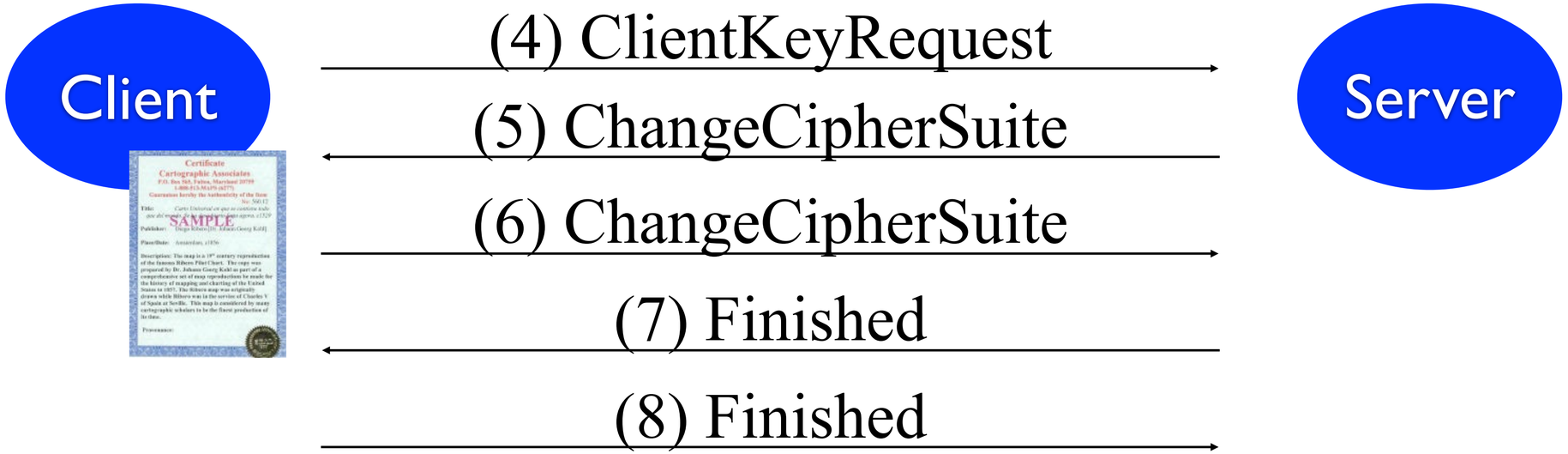
- Applications may want to construct secure communication channels transparently to users
  - How can they do that?

Application (Web) Security: SSL

- Secure socket Layer (SSL/TLS)
- Used to authenticate servers
  - Uses certificates, "root" CAs
- **Can** authenticate clients
- Inclusive security protocol
- Security at the socket layer
  - Transport Layer Security (TLS)
  - Provides
    - authentication
    - confidentiality
    - integrity



SSL Handshake



Simplified Protocol Detail

Participants: Alice/A (client) and Bob/B (server)
Crypto Elements : Random R, Certificate C, k_i^+ Public Key (of i)
Crypto Functions : Hash function $H(x)$, Encryption $E(k, d)$, Decryption $D(k, d)$, Keyed MAC $HMAC(k, d)$

1.	Alice $\rightarrow$ Bob	R_A
2.	Bob $\rightarrow$ Alice	R_B, C_B
	Alice	pick pre-master secret S
	Alice	calculate master secret $K = H(S, R_A, R_B)$
3.	Alice $\rightarrow$ Bob	$E(k_B^+, S), HMAC(K,' CLNT' + [\#1, \#2])$
	Bob	recover pre-master secret $S = D(k_B^-, E(k_B^+, S))$
	Bob	calculate master secret $K = H(S, R_A, R_B)$
4.	Bob $\rightarrow$ Alice	$HMAC(K,' SRVR' + [\#1, \#2])$

Note: Alice and Bob : IV Keys, Encryption Keys, and Integrity Keys 6 keys,where each key $k_i = g_i(K, R_A, R_B)$, and g_i is key generator function.

CLNT' = Alice, SRVR' = Bob, K'= K

SSL Tradeoffs

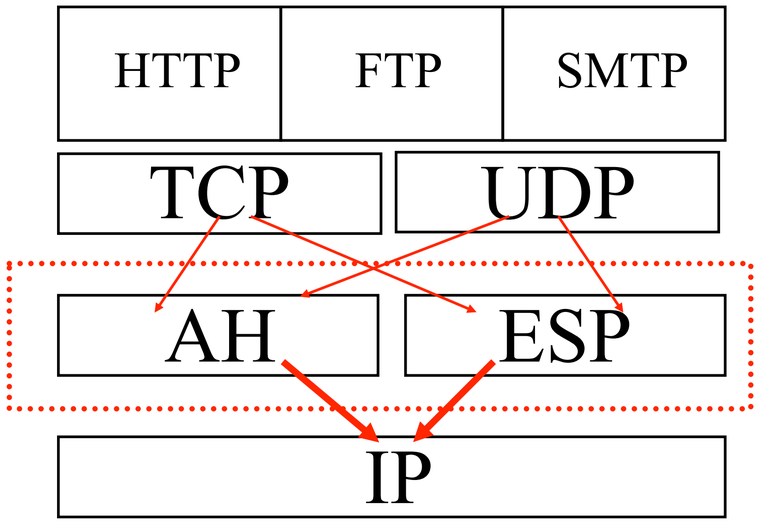
- Pros
  - Server authentication*
  - GUI clues for users
  - Built into every browser
  - Easy to configure on the server
  - Protocol has been analyzed like crazy
- Cons
  - Users don't check certificates
  - Too easy to obtain certificates
  - Too many roots in the browsers
  - Some settings are terrible

IPsec (not IPSec!)

- Host-level protection service
  - IP-layer security (below TCP/UDP)
  - De-facto standard for host level security
  - Developed by the IETF (over many years)
  - Available in most operating systems/devices
    - E.g., XP, Vista, OS X, Linux, BSD*, …
  - Implements a wide range of protocols and cryptographic algorithms
- *Selectively* provides ….
  - Confidentiality, integrity, authenticity, replay protection, DOS protection

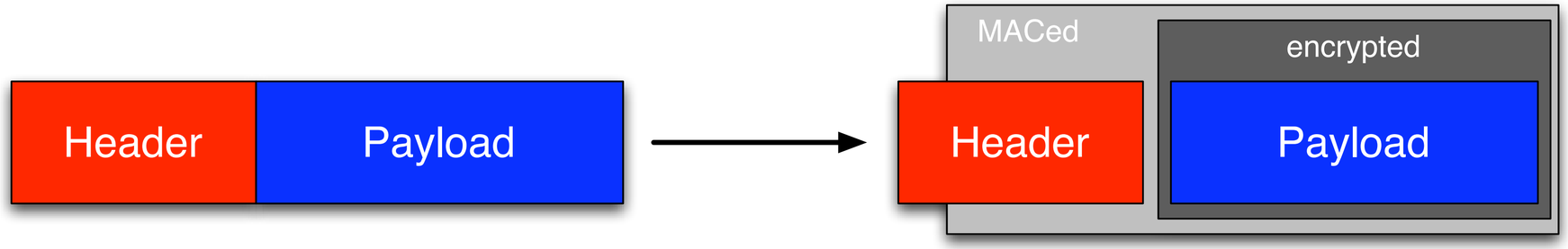
IPsec and the IP protocol stack

- IPsec puts the two main protocols in between IP and the other protocols
  - AH - authentication header
  - ESP - encapsulating security payload
- Other functions provided by external protocols and architectures

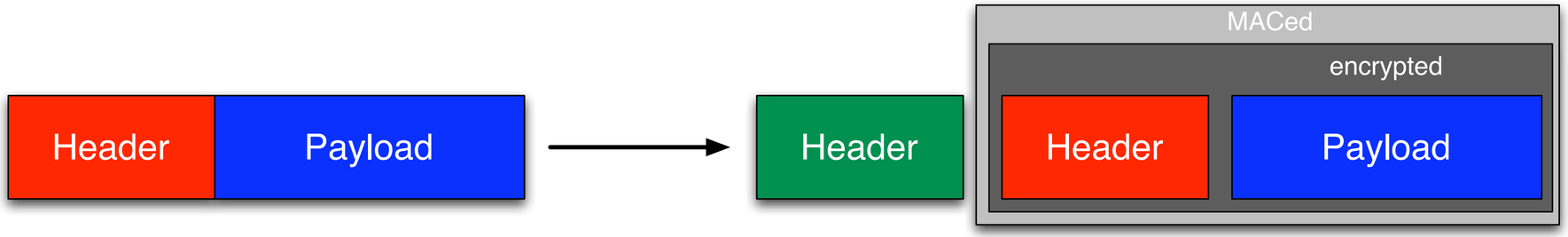


Modes of operation

- Transport : the payload is encrypted and the non-mutable fields are integrity verified (via MAC)

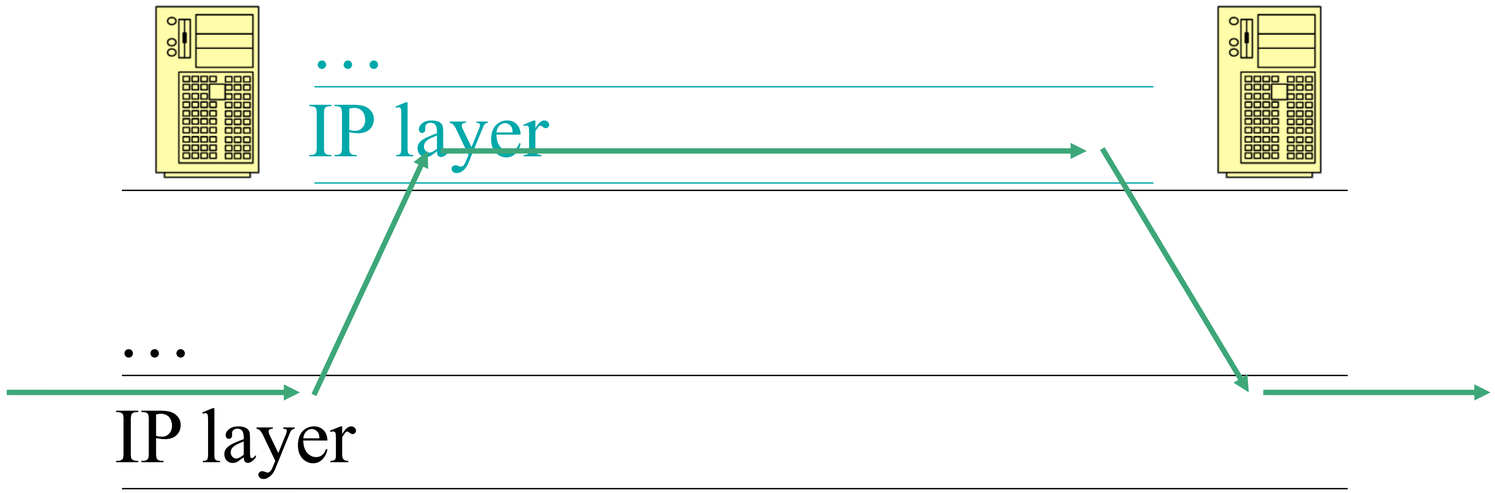


- Tunnel : each packet is completely encapsulated (encrypted) in an outer IP packet
  - Hides not only data, but some routing information



Tunneling

- "IP over IP"
  - Network-level packets are encapsulated
  - Allows traffic to evade firewalls

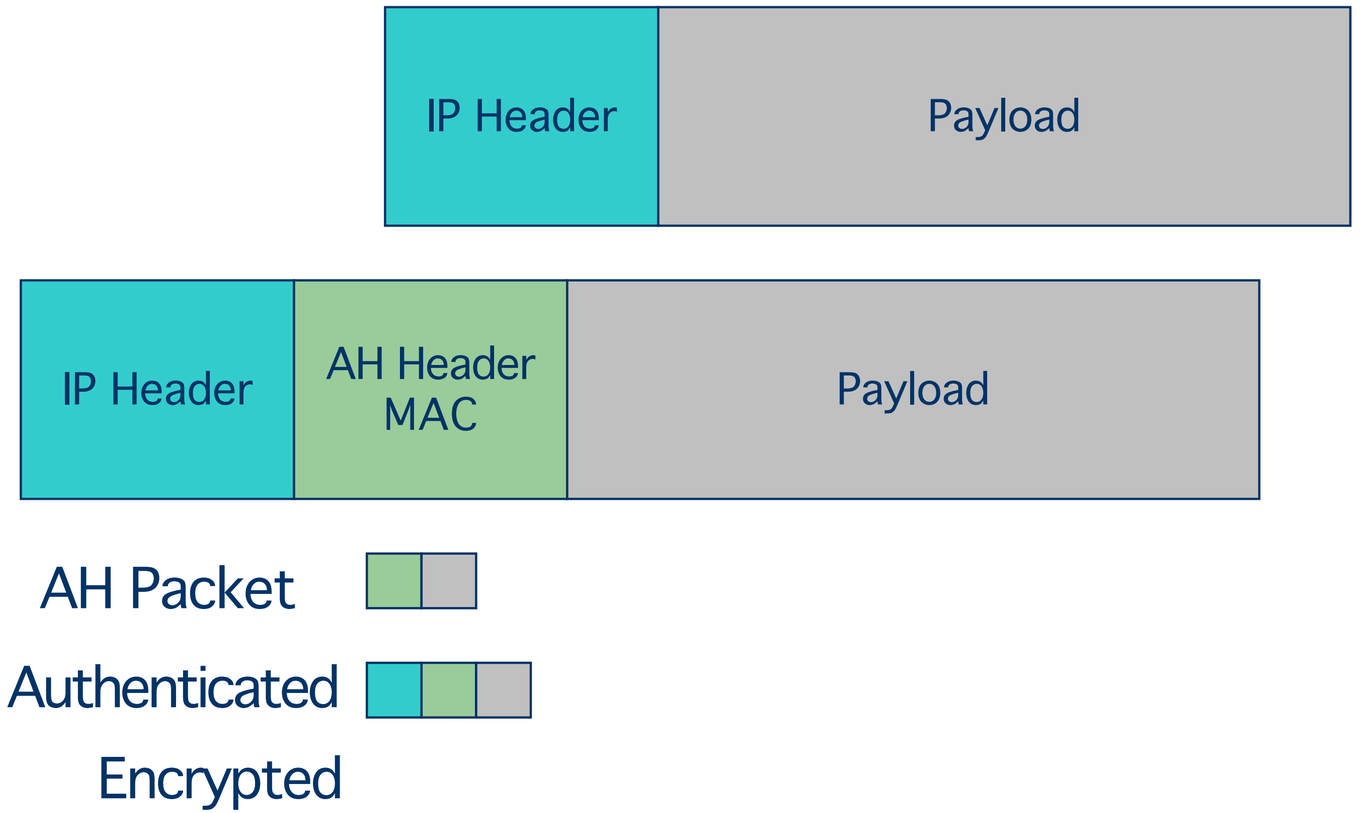


Authentication Header (AH)

- Authenticity and integrity
  - via HMAC
  - over IP headers and data
- Advantage: the authenticity of data and IP header information is protected
  - it gets a little complicated with *mutable* fields, which are supposed to be altered by network as packet traverses the network
  - some fields are *immutable*, and are protected
- Confidentiality of data is *not* preserved
- Replay protection via AH sequence numbers
  - note that this replicates some features of TCP (good?)

Authentication Header (AH)

- Modifications to the packet format

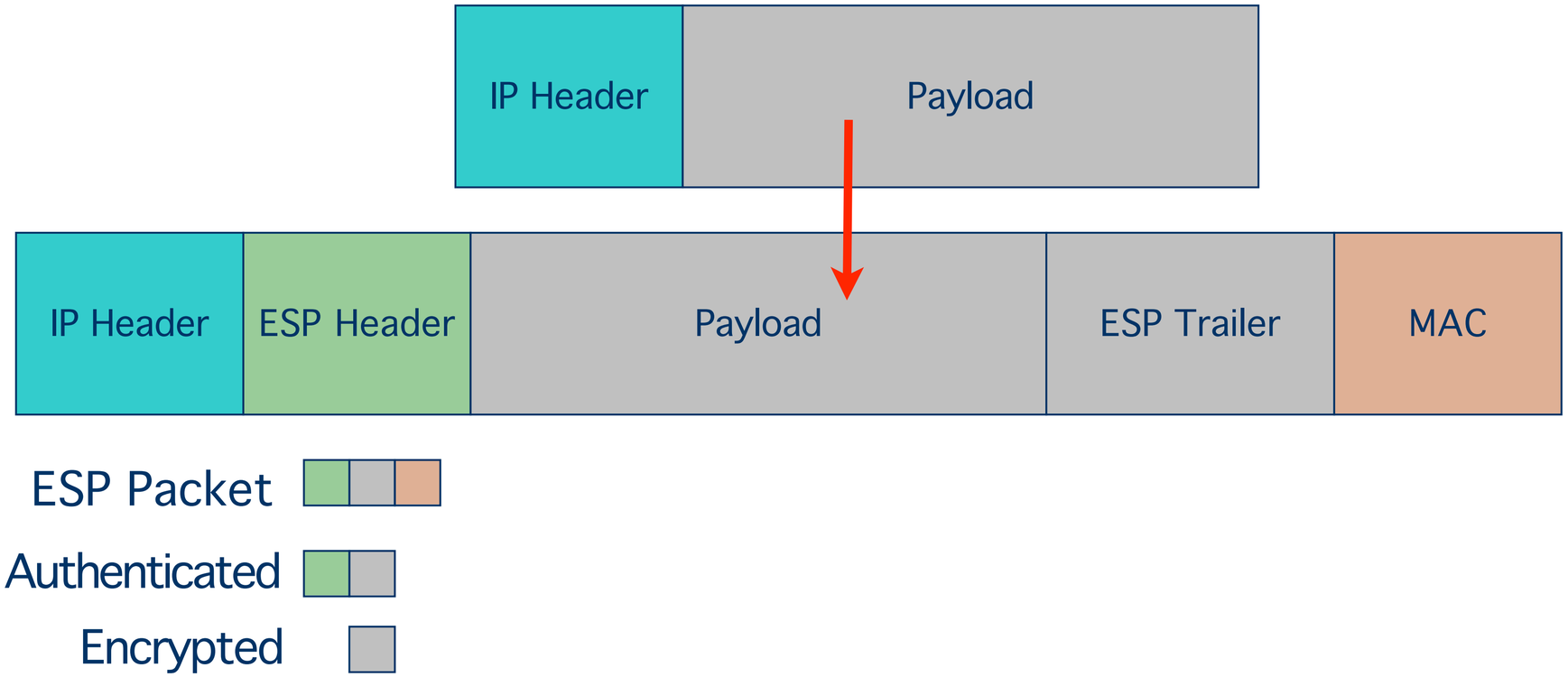


Encapsulating Security Payload (ESP)

- Confidentiality, authenticity and integrity
  - via encryption and HMAC
  - over IP *payload* (data)
- Advantage: the security manipulations are done solely on user data
  - TCP packet is fully secured
  - simplifies processing
- Use "null" encryption to get authenticity/integrity only
- Note that the TCP ports are hidden when encrypted
  - good: better security, less is known about traffic
  - bad: impossible for FW to filter/traffic based on port
- Cost: can require many more resources than AH

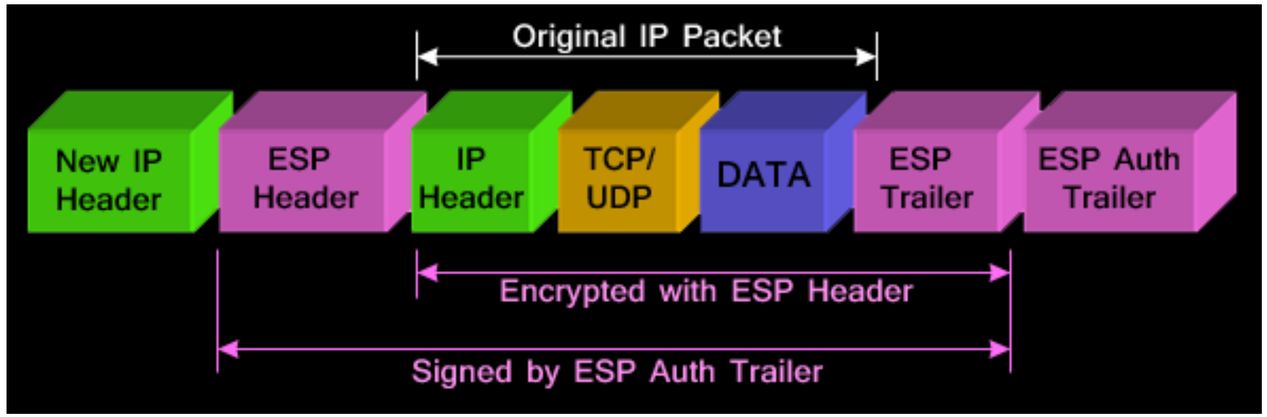
Encapsulating Security Payload (ESP)

- Modifications to packet format

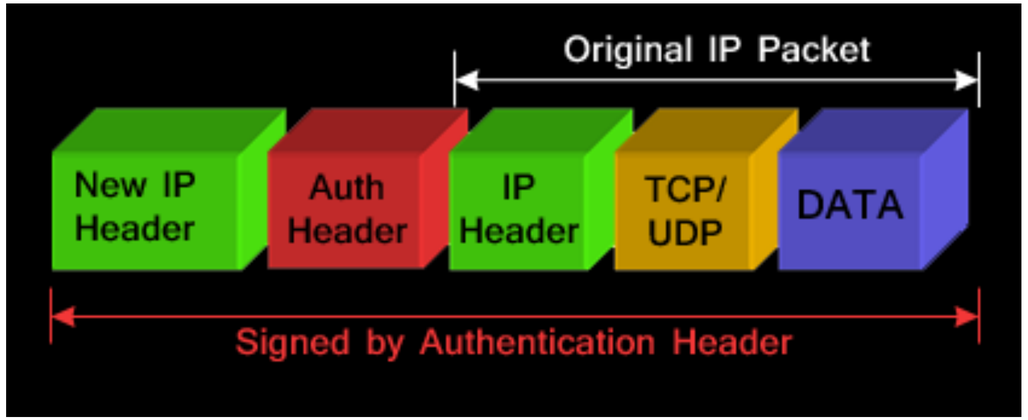


Tunnel mode with ESP and AH

IPSec Tunnel mode with ESP header:

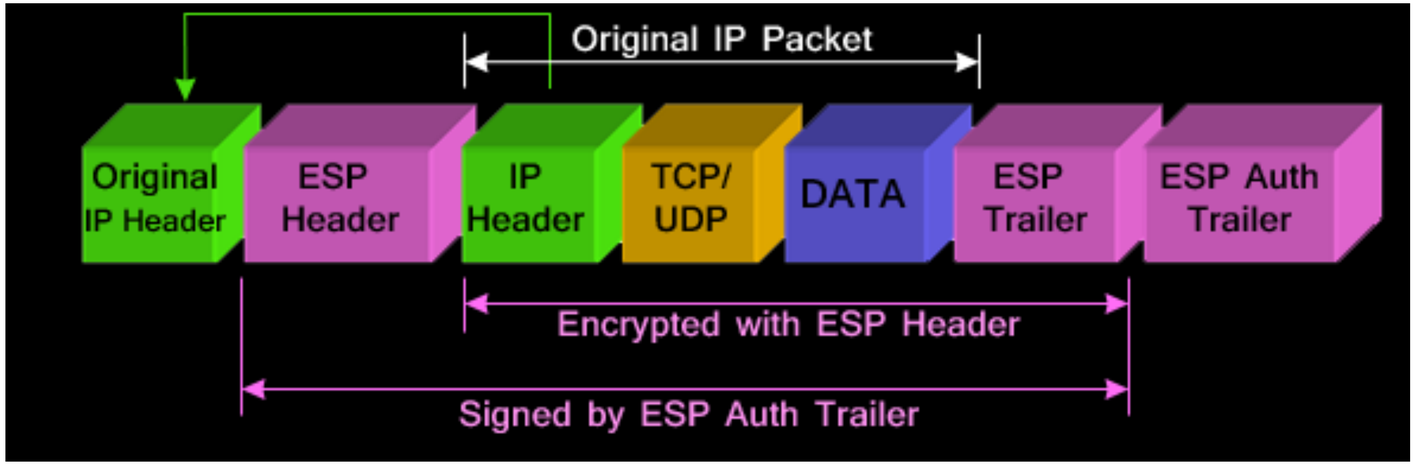


IPSec Tunnel mode with AH header:

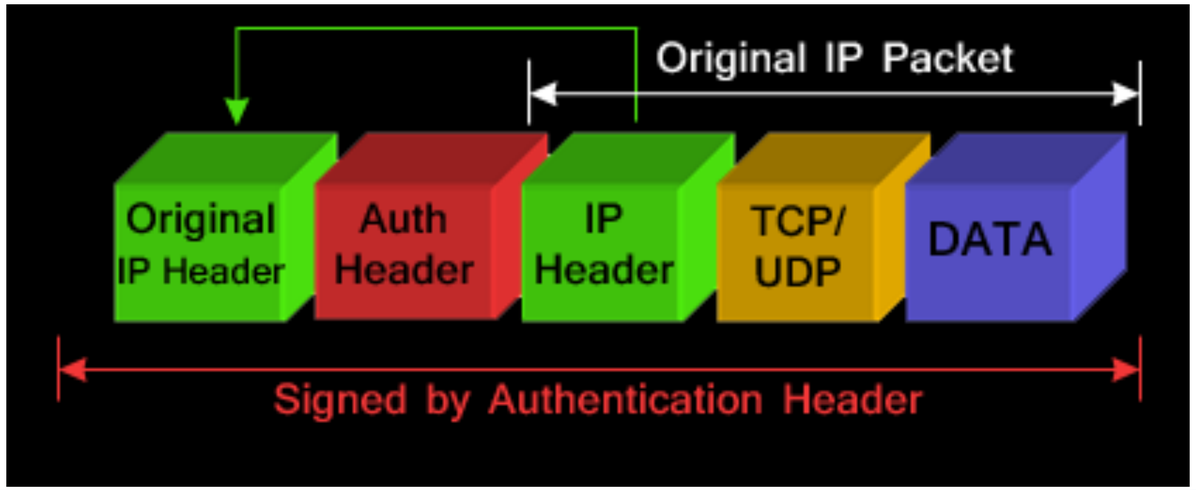


Transport mode with ESP and AH

IPSec Transport mode with ESP header:



IPSec Tunnel mode with AH header:



Tunnel and Transport Mode

	Transport Mode SA	Tunnel Mode SA
AH	Authenticates IP payload and selected portions of IP header and IPv6 extension headers.	Authenticates entire inner IP packet (inner header plus IP payload) plus selected portions of outer IP header and outer IPv6 extension headers.
ESP	Encrypts IP payload and any IPv6 extension headers following the ESP header.	Encrypts entire inner IP packet.
ESP with Authentication	Encrypts IP payload and any IPv6 extension headers following the ESP header. Authenticates IP payload but not IP header.	Encrypts entire inner IP packet. Authenticates inner IP packet.

Practical Issues and Limitations

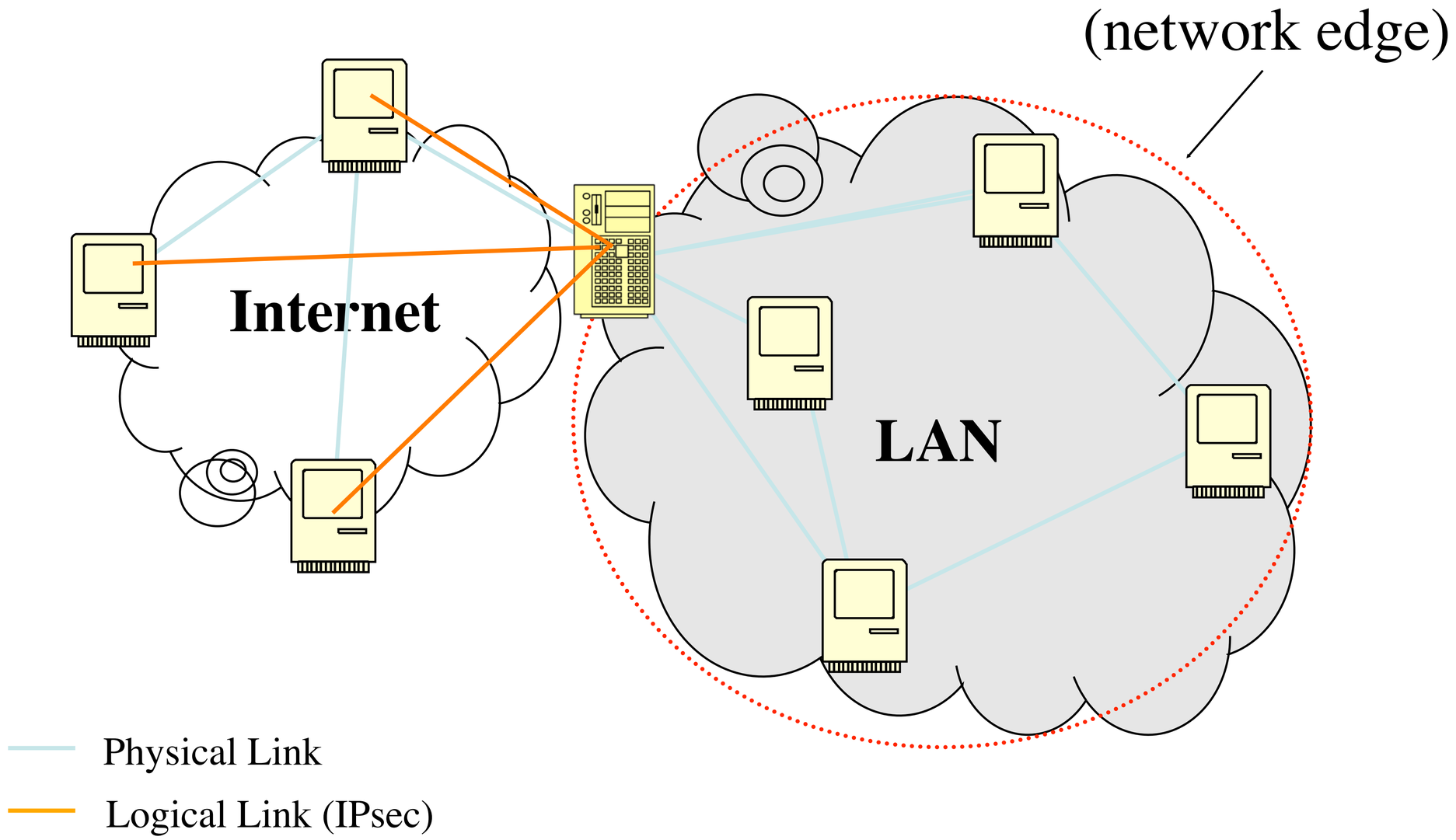
- IPsec implementations
  - Large footprint
    - resource poor devices are in trouble
    - New standards to simplify (e.g, JFK, IKE2)
  - Slow to adopt new technologies
  - Configuration is really complicated/obscure

- Issues
  - IPsec tries to be "everything for everybody at all times"
    - Massive, complicated, and unwieldy
  - Policy infrastructure has not emerged
  - Large-scale management tools are limited (e.g., CISCO)
  - Often not used securely (common pre-shared keys)

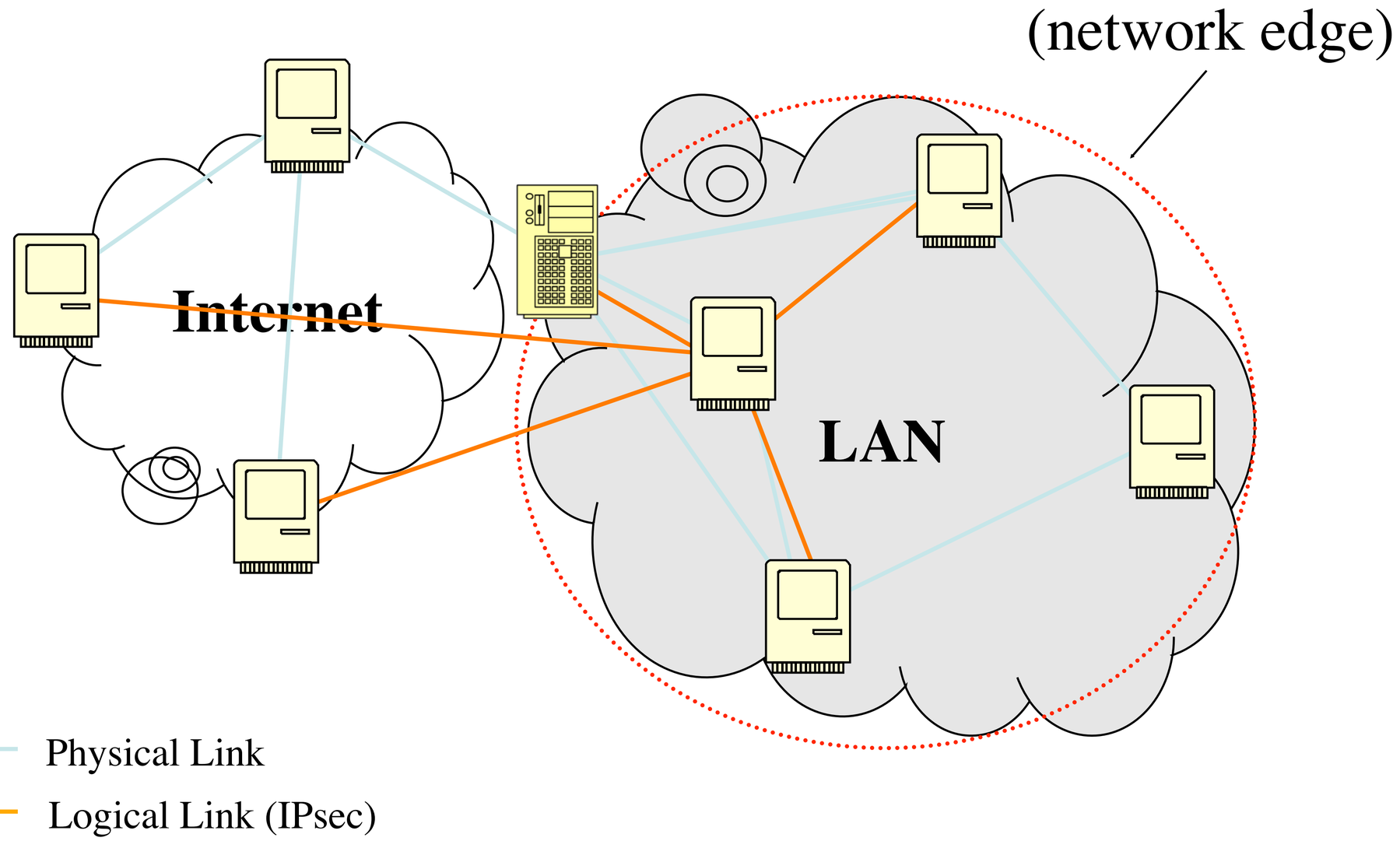
Network Isolation: VPNs

- Idea: I want to create a collection of hosts that operate in a coordinated way
  - E.g., a virtual security perimeter over physical network
  - Hosts work as if they are isolated from malicious hosts
- Solution: Virtual Private Networks
  - Create virtual network topology over physical network
  - Use communications security protocol suites to secure virtual links "tunneling"
  - Manage networks as if they are physically separate
  - Hosts can route traffic to regular networks (*split-tunneling*)

VPN Example: RW/Telecommuter



VPN Example: Hub and Spoke



VPN Example: Mesh