Lecture 24
Security - Technology

Peter Steenkiste

School of Computer Science

Carnegie Mellon University

15-441 Networking

Mutilated by Dave Eckhardt, F'04, S'06

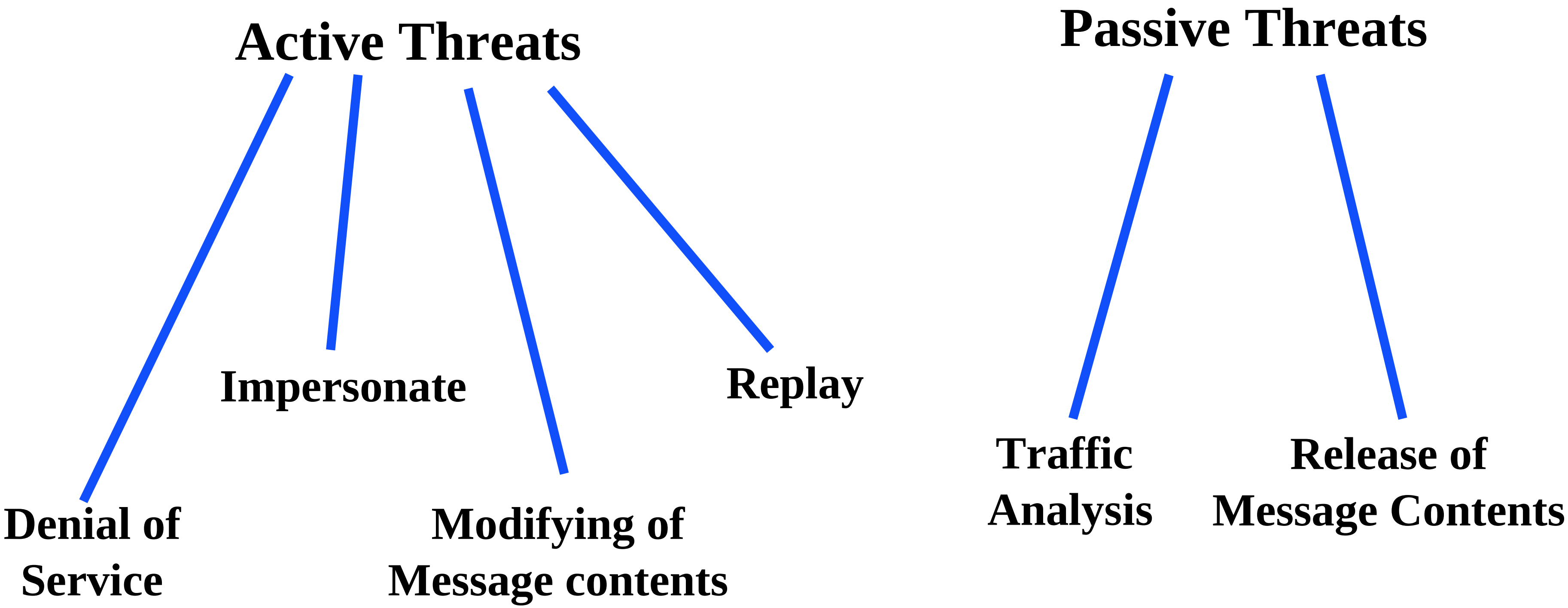
Outline

- **Textbook coverage**
  - » **Chapter 8**
  - » **Do not get bogged down in mathematics of DES, RSA**
  - » **Do understand how to use them to get jobs done**
- **Security threats and techniques.**
- **Encryption**
  - » **Private-key, public-key**
- **Hashing**
- **IP security (IPsec)**

Security Threats

- Impersonation.
  - » Pretend to be another user with the intent of getting access to information or services
- Secrecy.
  - » Get access to the contents of packets
- Message integrity.
  - » Change a message unbeknownst to the sender or receiver
- Repudiation
  - » Denying having sent a message
- Breaking into systems.
  - » To steal or destroy contents
- Denial of service.
  - » Flooding the system so users with legitimate needs cannot get service

Active Versus Passive Threats



Three Levels of Defense

- Using firewalls to limit access to the network.
  - Packets that cannot enter the network cannot cause harm
  - Packets that do not leave the network cannot leak secrets
- Securing the infrastructure at the network layer (IP).
  - Host to host or at a finer grain
  - Can be viewed as management tool: can be done without knowledge of applications
- Application level security.
  - Communicating peers execute protocols to secure their communication channel
  - Essential for critical applications: end-to-end security
  - Requires effort from both application developers and users

Encryption

Ciphertext = E(plaintext, K_E)

Plaintext = D(ciphertext, K_D)

Algorithm = E(), D()

- Algorithm should generally be public
  - » Otherwise when (!!) it is cracked you won't hear about it
  - » Easier to get known-good software implementations
  - » Encourages fast hardware implementations
- Keys are generally kept private
  - » Easier to change a key than an algorithm
- Given the ciphertext, it must be “very difficult” to calculate the plaintext without K_D
  - » Difficult = computationally very expensive
  - » Resistant to known attacks

Special Cases

Ciphertext = E(plaintext, K_E)

Plaintext = D(ciphertext, K_D)

Algorithm = E(), D()

- **Details**
  - » **E() and D() may be the same function**
  - » **K_E and K_D may be the same key**
  - » **This is called *symmetric* encryption**

Perfect Encryption: One-Time Pad

- "Pad" = large nonrepeating set of truly random key letters

plaintext	ONETIMEPAD
one-time pad	TBFRGFARFM..........................
ciphertext	IPKLPSFHGQ

- **Algorithm often simple**
  - » K_E == K_D, E() == D() == XOR()
- **Perfect <u>if and only if</u>**
  - » Key bits are <u>truly random</u>
  - » Key bits are <u>never re-used</u>

Simple Applications

- Maintain secrecy of message

A:	m = "secret msg"
	m' = E(m, K_E)
A⇒B:	m'
B:	m = D(m', K_D)

- Prove identity by knowing a key
  - » two parties must have a shared secret

A:	m = "I am A"
	m' = E(m, K_E)
A⇒B:	m, m'
B:	verify m = D(m', K_D)

Public Key versus Private Key Cryptography

- **Private key (symmetric, e.g., DES)**
  - » Two parties share (keep private) a key k
  - » Encrypt plaintext using k
  - » Also decrypt ciphertext using k -- symmetric
- **Public key (asymmetric, e.g., RSA)**
  - » Keys come in *pairs*, $K_{private}$ and K_{public}
  - » $K_{private}$ is kept private by its owner
  - » K_{public} is published
  - » Sender encrypts with recipient's public key $C=E(M, K_{public})$
  - » Recipient uses private key to decrypt $M=D(C, K_{private})$
  - » Must be "impossible" to derive private key from public

Authentication Revisited

Private key	
A:	m = "I am A"
	m' = $\{m\}k_{shared}$
A⇒B:	m, m'
B:	verify m' = $\{m\}k_{shared}$

Public key	
A:	m = "I am A"
	m' = $\{m\}k_{private}$
A⇒B:	m, m'
B:	verify m = $\{m'\}k_{public}$

- Parties must share a secret before they can communicate.
- Need a separate channel to establish the shared key.

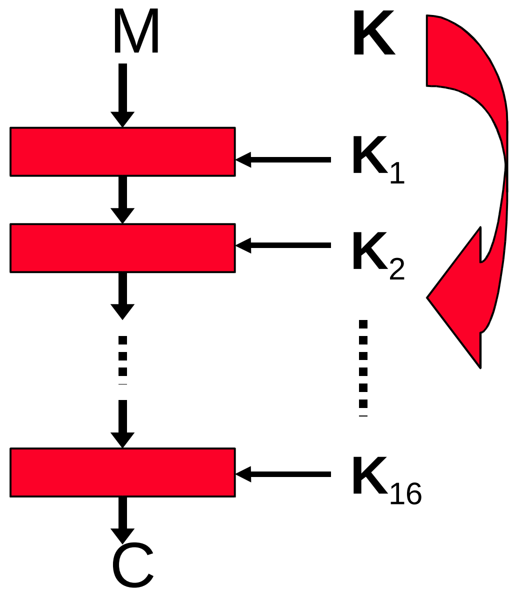
- Distribution of keys is easier: public directory of public keys
- Still need a way to reliably distribute public keys.

Data Encryption Standard DES

- Example of symmetric-key cryptography.
- Basically permutes the bits based on a 56-bit key.
  - » Substitution: reduce the relationship between plaintext and ciphertext
  - » Diffusion: move the bits around
- How secure is DES?
  - » It is becoming less secure as computers get faster
  - » DES has recently been "cracked" by teams of volunteers using both lots of idle workstations, and special-purpose hardware
- Security can be improved by running the algorithm several times, e.g. Triple-DES
  - » Odd fact: 2DES is <u>less safe</u> than DES!

DES Algorithm

- **Use a 64-bit key to encrypt data in 64-bit blocks**
  - » **Actually 56-bit key: every 8th bit is parity**
- **16 "rounds"**
  - » **The 56-bit key K is used to generate 16 48-bit keys $K_1 \ldots K_{16}$, one for each round**
- **In each round:**
  - » **Substitution (S-boxes)**
  - » **Permutation (P-boxes)**



RSA Algorithm

- Example of a public key system.
  - Name based on the names of its founders
- Key pair based on a pair of large prime numbers.
  - Different key sizes can be used
  - Larger key sizes are harder to crack but also result in more expensive encryption and decryption
- Encryption and decryption is based on exponentiation and remainder calculation.
- The security of RSA is based on the fact that there is no known algorithm for quickly factoring large numbers

Public vs. Private Key Systems

- **Scale of key management.**
  - » If N users want to communicate securely, private key systems require Nx(N-1)/2 keys while public key systems require only N key pairs
- **Computational cost.**
  - » Public key cryptography is much more expensive than private key cryptography
- **Compromise: use public key system to agree on temporary private keys**
- **Or: use an *authentication server* to reduce the key management complexity of private key systems.**
  - » Authentication server versus public key server

Cryptanalysis: Types of Attack

- Goal: recover plaintext or key.
- Basic assumptions
  - » Attacker has complete access to the communications (ciphertext)
  - » Cryptanalyst knows the cryptographic algorithms (and protocols)
- Ciphertext-only
  - » Given $C_1 = E_K(M_1)$, $C_2 = E_K(M_2)$, … , $C_N = E_K(M_N)$
  - » Deduce M_1, M_2, … , M_N, or K
- Known-plaintext
  - » Given M_1, $C_1 = E_K(M_1)$, M_2, $C_2 = E_K(M_2)$, … , M_N, $C_N = E_K(M_N)$
  - » Deduce K
- Chosen-plaintext
  - » Attacker chooses M_1, … , M_N and gets $C_1 = E_K(M_1)$, … , $C_N = E_K(M_N)$
  - » Deduce K

Hash Functions

- Usually operates on an arbitrary length message to generate a fixed length message digest.
- Properties of a good hash function:
  1. Pre-image Resistant: given f(x) "cannot" find x
  2. 2nd Pre-image Resistant: given x and f(x), it is difficult to find $x' \neq x$ such that that $f(x) = f(x')$
  3. Collision Resistant: it is difficult to find any x', x such that that $x' \neq x$ and $f(x) = f(x')$
- If 1,2 are satisfied, the function is said to be "one way"
- Example uses:
  - » Message Authentication
  - » Password Storage
  - » Key Generation

Hash Function Usage

- **Message Authentication**
  - » **A: “I have published the new OpenBSD CD-ROM image on lots of FTP servers.”**
  - » **B: “I have downloaded an image from ftp.asdfsdfa.org ... Is it the right one?”**
  - » **A: “Oh, the MD5 hash of the image I published is d41d8cd98f00b204e9800998ecf8427e.”**
- **Password Storage**
  - » **Storing passwords in a file makes the file <u>very</u> attractive to thieves...**
  - » **Solution: store MD5(password) instead. When user types in password, compute MD5(typed), compare to MD5(password).**

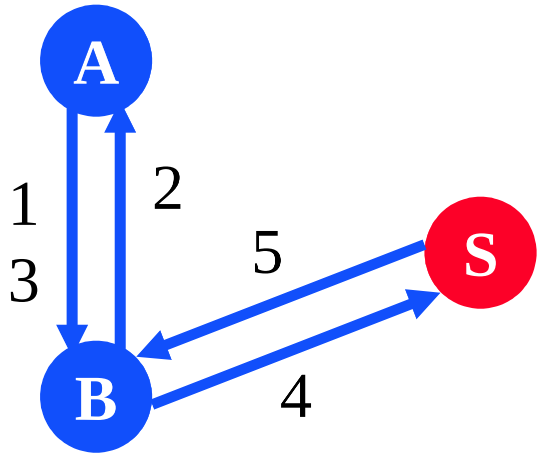
Thinking About Protocols

- **Goals**
  - » **Prove somebody has a certain identity, or certain authorization**
- **Tools**
  - » **“Reasonable” assumptions about keys**
    - – **A knows A's private key**
    - – **Nobody else knows A's private key**
  - » **Encryption really is secret**
    - – **If X can decrypt a message encrypted with A's private key, X must know A's private key (therefore X is A)**
  - » **Secure hashing really works**
    - – **If I send you a hash of a message, I must know the message contents (or I stole the hash from somebody who did know the message contents)**
    - – **If a hash and a message match, the message contents were not changed in transit**

Using an Authentication Server

- Avoid n^2 key problem: each principal shares a key with server.
  - » Server S helps in authenticating A to B

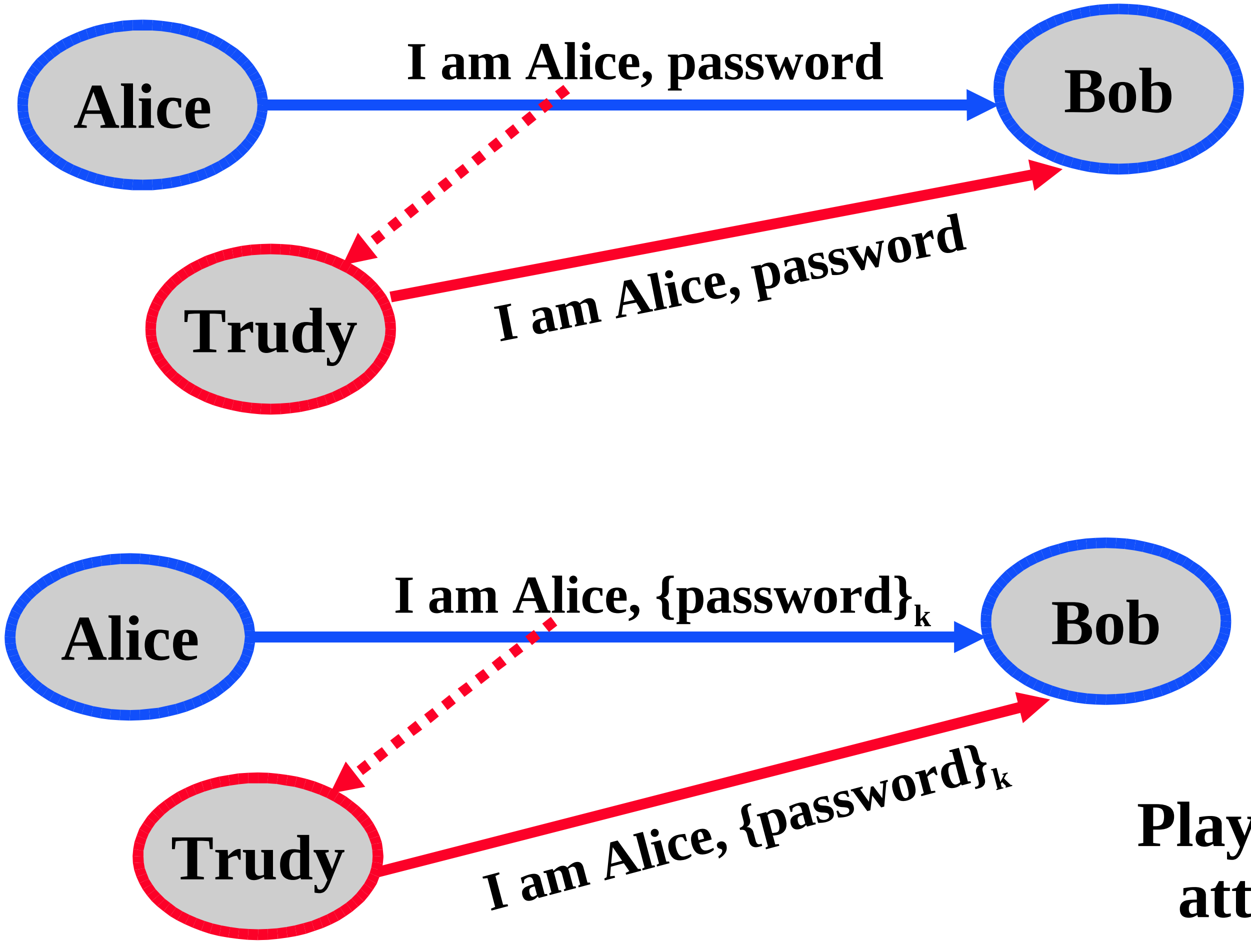
A⇒B:	"I am A"
B⇒A:	n
A:	n' = $\{n\}k_A$
A⇒B:	n'
B:	n'' = $\{A,n'\}k_B$
B⇒S:	n''
S:	recover A, n'
	extract n = $\{n'\}k_A$
	m = $\{n\}k_B$
S⇒B:	m
B:	verify m = $\{n\}k_B$



Authentication

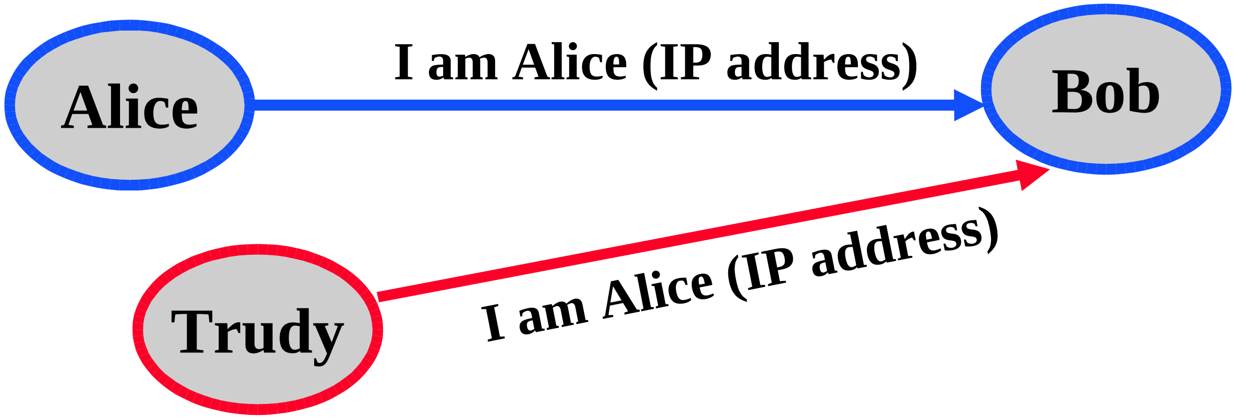
- Use authentication to illustrate some of the pittfalls of using cryptography to address security threats.
  - » Goal is for Alice to authenticate herself to Bob
- Passwords.
- Encrypted passwords.
- Use of a nonce.
- A challenge-based approach.

Plain or Encrypted Passwords



Playback attack

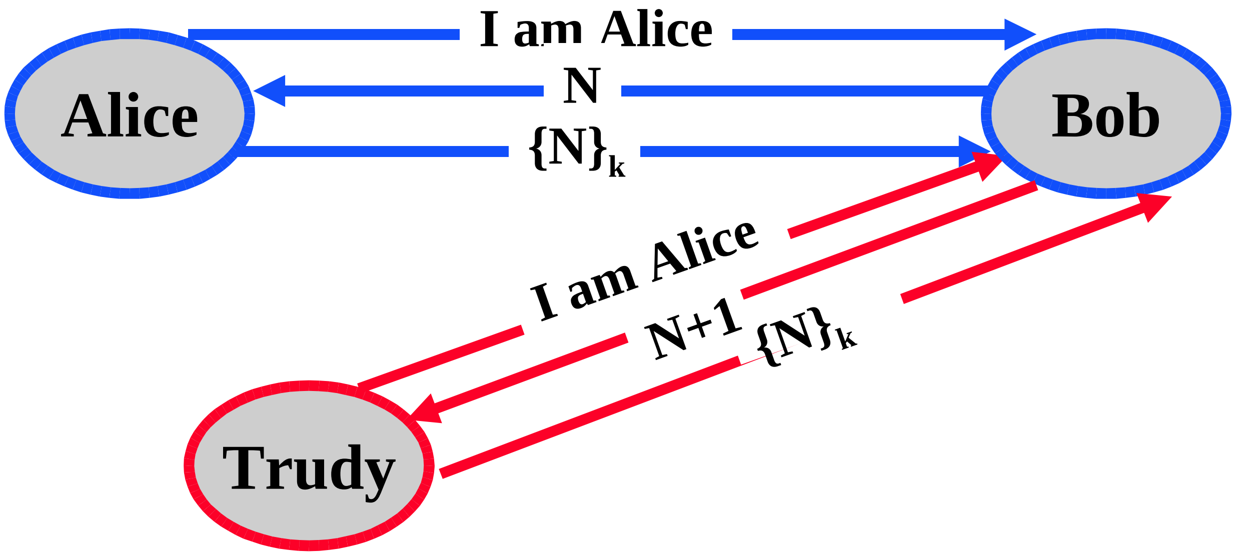
IP Spoofing



- **Fairly easy to generate packets with arbitrary IP source addresses.**
  - » **Certainly when you have access to the operating system**
- **Bob will send reply back to (the real) Alice.**
  - » **But Trudy could intercept these replies**

Preventing Replay Attack

- Include a *nonce*, a value that is used only once, in the message.
  - » Can be timestamp, random number, …
  - » Prevents a simple replay of requests or responses



Digital Signatures

- How can you prove somebody sent you a specific message?
  - » Prove identify of sender and exact message contents
- Digital signature: Bob sends Alice a plaintext message plus a cyphertext encrypted with his private key.
  - » Alice can verify that they are the same (everybody can)
  - » Alice has proof that only Bob could have sent this message
    - – since only Bob could have encrypted the message
  - » If either Bob or Alice modify the message, the other party can prove it
- Catch: what happens if Bob advertises his private key?

Message Digests

- Public key cryptography can be used to sign documents, but it is computationally expensive.
  - » Makes message nonforgeable, verifiable, nonrepudiable
- Message digests save on computation costs by computing a small digest of the message, which can then be signed.
  - » Uses a many-to-one hash function H, i.e., m = H(M)
  - » Given m, it is infeasible to find an N so m=H(N)
  - » It is infeasible to find an M and N so H(M)=H(N)
- Example: MD5.
  - » Computes a 128 bit digest
  - » Alternative: SHA-1, a US federal standard; creates a 160 bit digest

2004 Update

- Summry of 2004 was fun
  - ...in terms of cryptography...
  - ...where “fun” means “horror movie”...
- MD5 is probably blown
  - A Chinese group can come up with (m_1,m_2) pairs which hash to the same value...
  - ...fast.
- SHA-1 is “in trouble”
  - SHA-1's “little brother” SHA-0 is under pressure
  - Same technique might end up working for SHA-1
- So much for cryptographic hashing? Unknown!

IP Security Goals

- Provide a set of protocols that offer security at the network layer.
  - » Ideally every datagram sent over the Internet would be protected by IPsec
  - » Analogy: almost all letters travel in an envelope
- Security is supported from source host to destination host.
  - » Can cover all end-to-end information in the packet
    - – Layers 4 and up
    - – Raises some issues with regard to classification
      - flow id's, firewall policy
  - » IPsec may not be sufficient for some applications
    - – May want secure connection between two applications (instead of two hosts)
- Defined for both IPv4 and IPv6.

IP Security Components

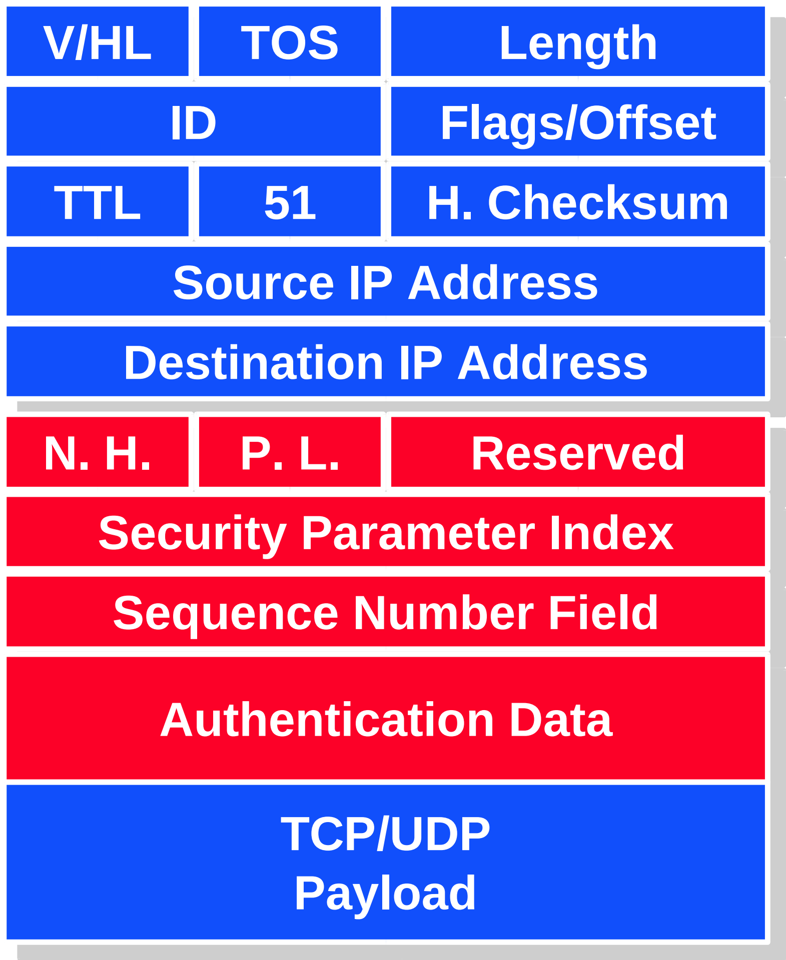
- IP "Authentication Header" protocol supports authentication and integrity.
  - » Based on cryptographic authentication function that is computed using a secret authentication key
- IP "Encapsulating Security Payload" protocol supports authentication, integrity, and confidentiality.
  - » Encrypt entire IP datagram or upper-layer protocol data
  - » New clear-text IP header is used to carry packet through the network
- Based on a "security association".
  - » Authentication/encryption state used to handle an incoming packet
  - » Selected by SPI (Security Parameter Index) field

Security Associations

- A security association manages IPsec state for one direction of a connection
- Calling host asks for a new SA; called host creates one and assigns SPI
- Calling host tags packets to that destination with the assigned SPI
- The Security Policy Database defines policies applicable to the node.
  - Specifies policy (discard, bypass IPsec, apply IPsec) for inbound and outbound traffic
  - Selectors identify flows: host-host or more fine grain
- The Security Association Database keeps track of the state of active connections.
  - Protocols selected, keys, sequence numbers, ..
  - Keys can be managed manually or using IKE

Authentication Header (AH) Protocol

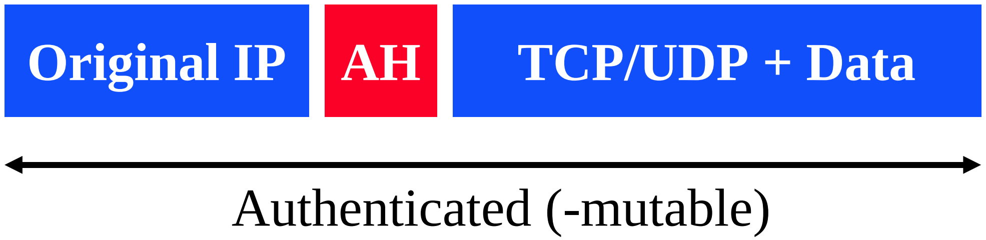
- AH sits between IP header and the payload.
  - » Protocol 51
- Next header: protocol number of payload packet (e.g., TCP)
- Payload length: length of AH in words (-2).
- Security Parameter Index identifies the session.
- Sequence number field can be used against replay attacks.
- Authentication data: Integrity Check Value.
  - » Signed digest, e.g. DES, keyed MD5, ...



Transport versus Tunnel Mode AH

Transport mode: host-to-host

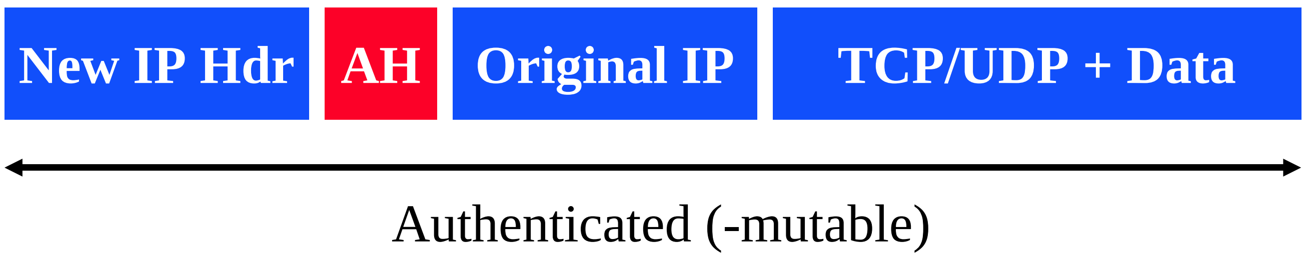
Origin host looks up SPI, adds AH header



Tunnel mode: router securely bridges two networks

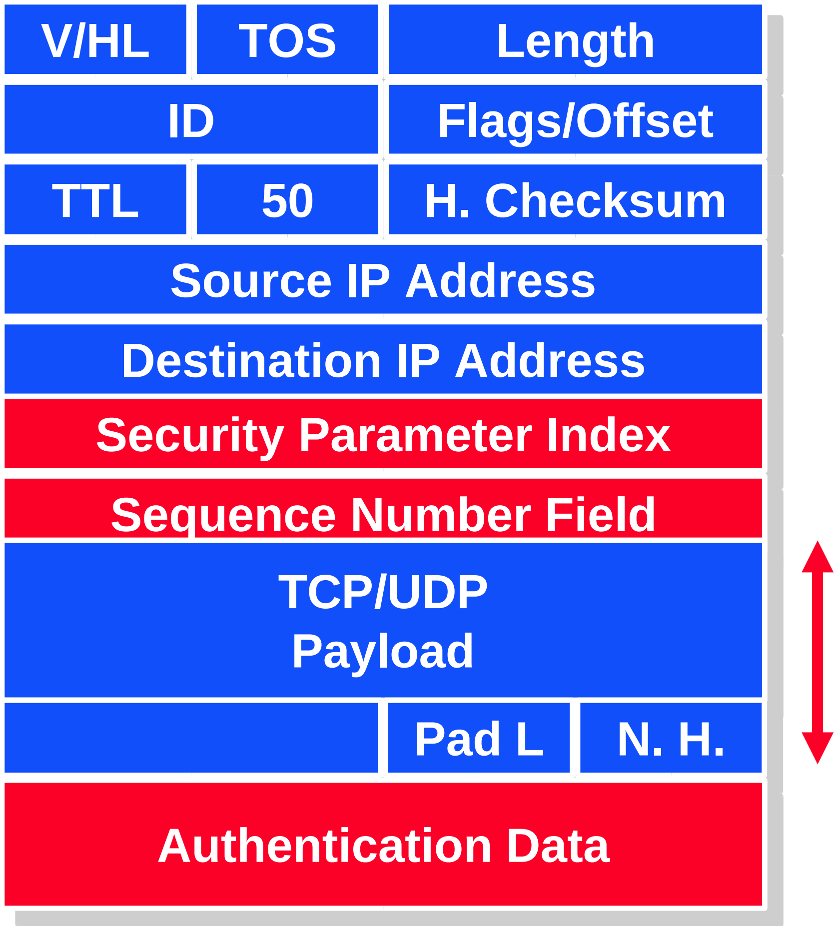
Preserves host-to-host IP header

Adds router-to-router header, uses router SA, SPI

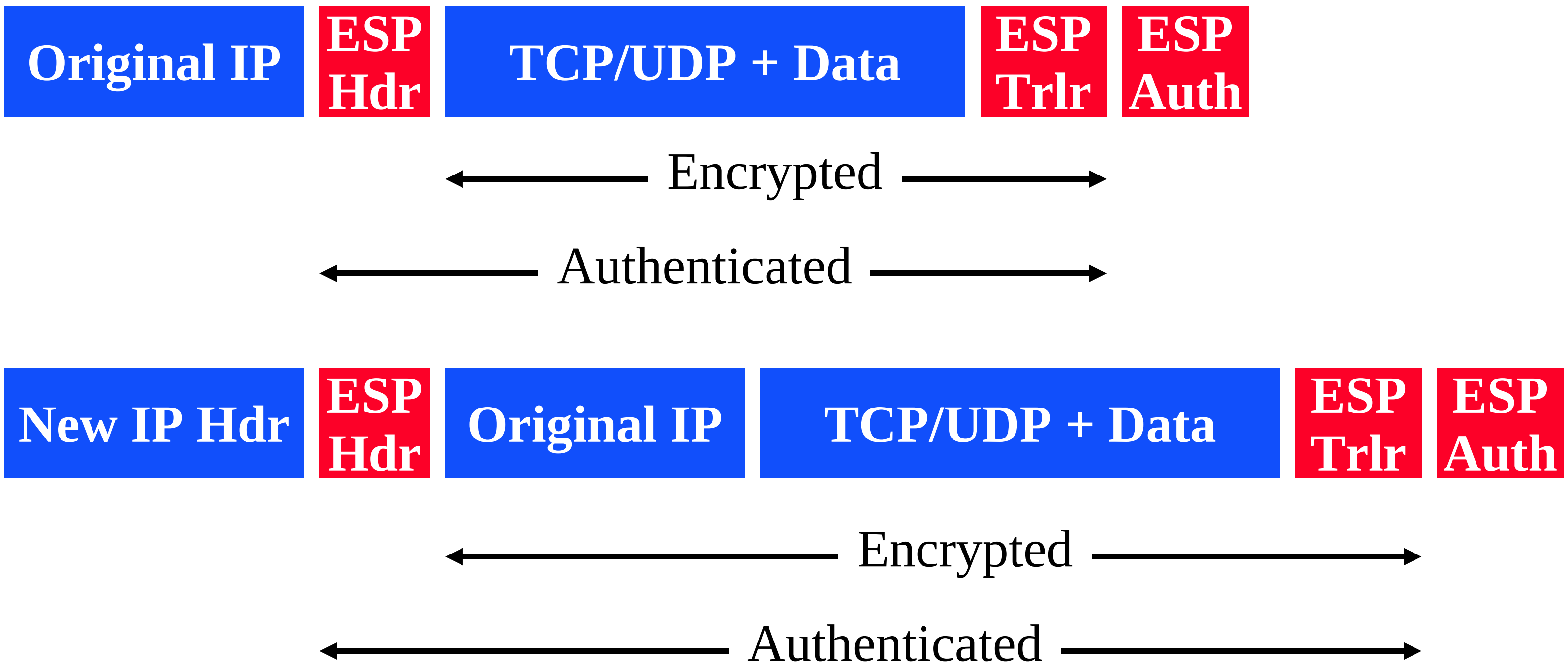


Encryption Support

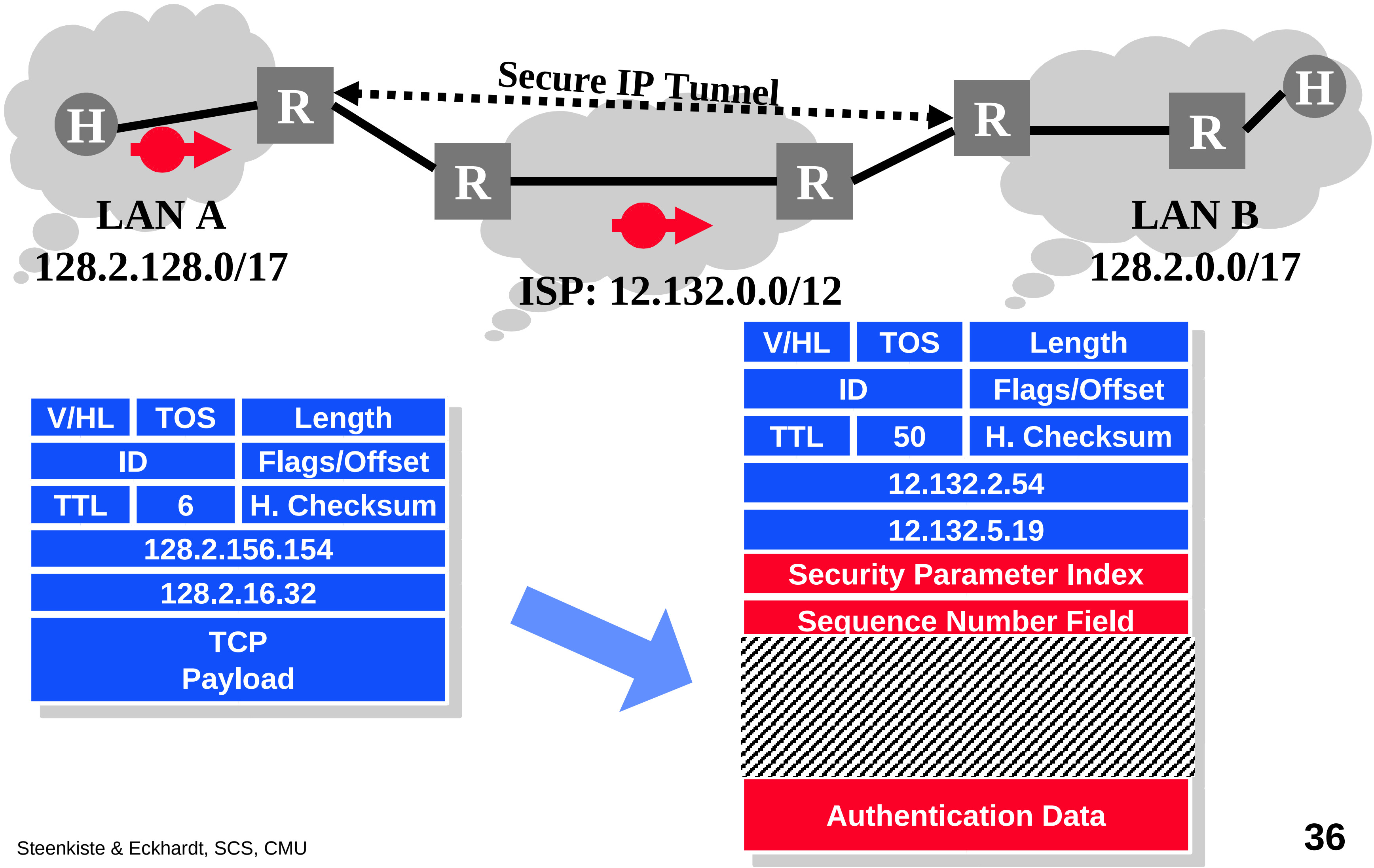
- ESP header follows IP header.
  - » Protocol Id = 50
- SPI and sequence number have same role as in AH.
- Padding is used to have make sure encrypted data is a multiple of 4 bytes, and is aligned on a 4 byte boundary.
- Authentication data: as in AH, but optional.



Transport versus Tunnel Mode ESP



Example: Virtual Private Networks



Summary

- Security threats and techniques
- Encryption
  - » Private-key, public-key
    - – Understand how to plug the parts together
    - – Who gets which keys?
    - – What do you encrypt and why?
- Hashing
- IP security (IPsec)