Chapter 7: Network security

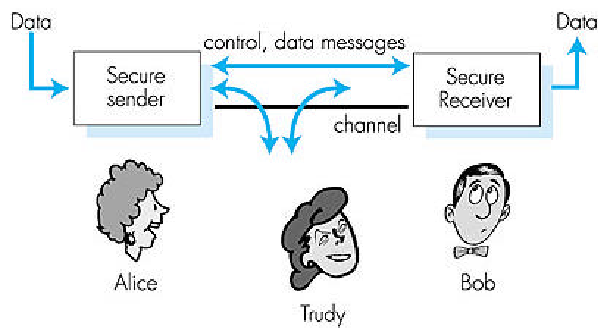
Foundations:

- what is security?
- cryptography
- authentication
- message integrity
- key distribution and certification

Security in practice:

- application layer: secure e-mail
- transport layer: Internet commerce, SSL, SET
- network layer: IP security

Friends and enemies: Alice, Bob, Trudy



- well-known in network security world
- Bob, Alice (lovers!) want to communicate "securely"
- Trudy, the "intruder" may intercept, delete, add messages

What is network security?

Secrecy: only sender, intended receiver should "understand" msg contents
- sender encrypts msg
- receiver decrypts msg

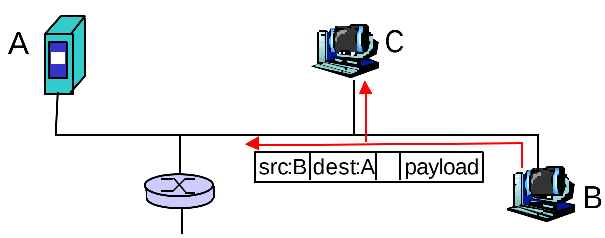
Authentication: sender, receiver want to confirm identity of each other

Message Integrity: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection

7: Network Security 3

Internet security threats

Packet sniffing:
- broadcast media
- promiscuous NIC reads all packets passing by
- can read all unencrypted data (e.g. passwords)
- e.g.: C sniffs B's packets

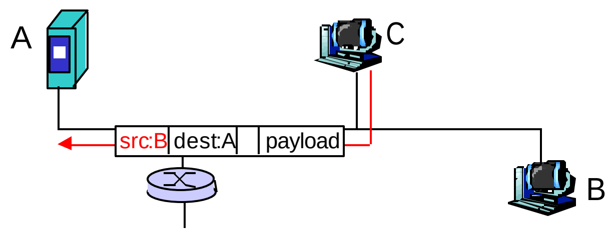


7: Network Security 4

Internet security threats

IP Spoofing:

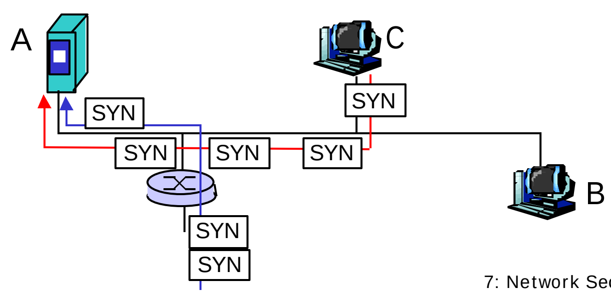
- can generate "raw" IP packets directly from application, putting any value into IP source address field
- receiver can't tell if source is spoofed
- e.g.: C pretends to be B



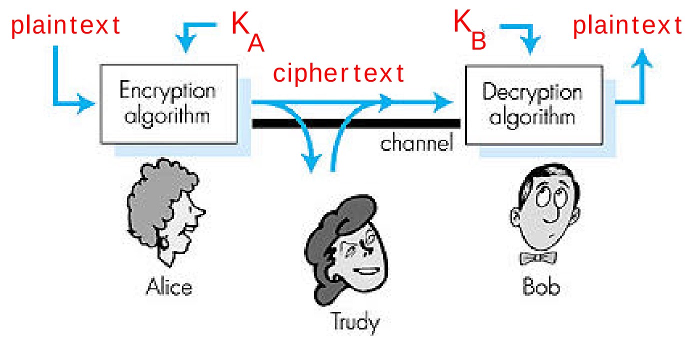
Internet security threats

Denial of service (DOS):

- flood of maliciously generated packets "swamp" receiver
- Distributed DOS (DDOS): multiple coordinated sources swamp receiver
- e.g., C and remote host SYN-attack A



The language of cryptography



symmetric key crypto: sender, receiver keys identical

public-key crypto: encrypt key *public*, decrypt key *secret*

Symmetric key cryptography

substitution cipher: substituting one thing for another

- monoalphabetic cipher: substitute one letter for another

```
plaintext:   abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

E.g.:
```
Plaintext:  bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc
```

Q: How hard to break this simple cipher?:

- brute force (how hard?)
- other?

Symmetric key crypto: DES

DES: Data Encryption Standard

- US encryption standard [NIST 1993]
- 56-bit symmetric key, 64 bit plaintext input
- How secure is DES?
  - DES Challenge: 56-bit-key-encrypted phrase ("Strong cryptography makes the world a safer place") decrypted (brute force) in 4 months
  - no known "backdoor" decryption approach
- making DES more secure
  - use three keys sequentially (3-DES) on each datum
  - use cipher-block chaining

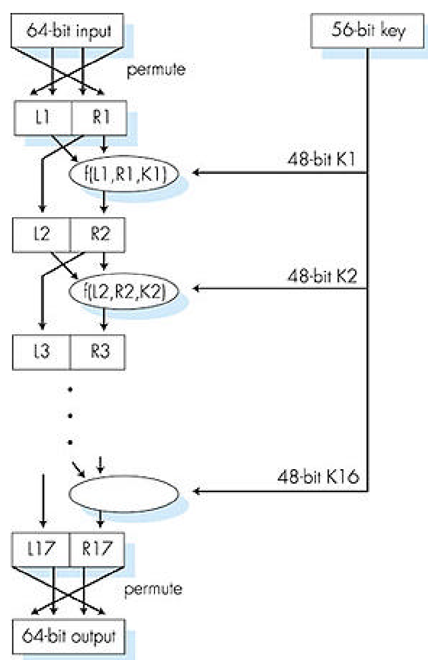
Symmetric key crypto: DES

DES operation

initial permutation

16 identical "rounds" of function application, each using different 48 bits of key

final permutation



Public Key Cryptography

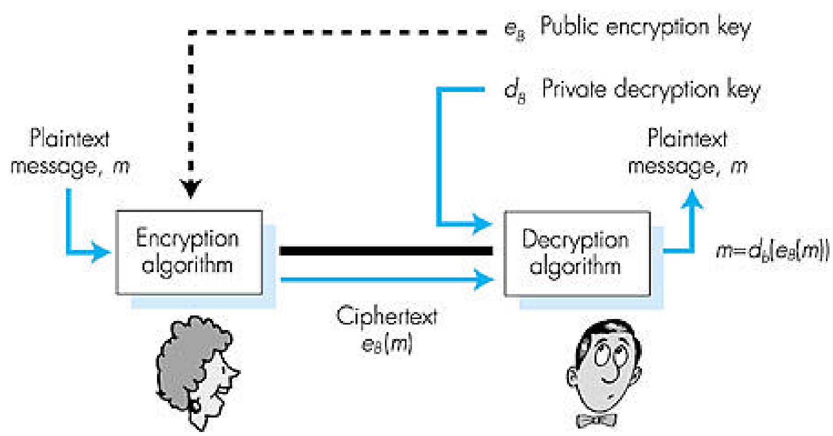
symmetric key crypto

- requires sender, receiver know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?

public key cryptography

- radically different approach [Diffie-Hellman76, RSA78]
- sender, receiver do *not* share secret key
- encryption key *public* (known to *all*)
- decryption key private (known only to receiver)

Public key cryptography



Public key encryption algorithms

Two inter-related requirements:

1. need $d_B(\cdot)$ and $e_B(\cdot)$ such that
   $d_B(e_B(m)) = m$

2. need public and private keys for $d_B(\cdot)$ and $e_B(\cdot)$

RSA: Rivest, Shamir, Adelson algorithm

RSA: Choosing keys

1. Choose two large prime numbers *p, q.* (e.g., 1024 bits each)
2. Compute $n = pq$, $z = (p-1)(q-1)$
3. Choose *e* (with $e<n$) that has no common factors with z. (*e, z* are "relatively prime").
4. Choose *d* such that $ed-1$ is exactly divisible by *z*. (in other words: $ed \bmod z = 1$).
5. *Public* key is *(n,e)*. *Private* key is *(n,d)*.

RSA: Encryption, decryption

0. Given (n,e) and (n,d) as computed above

1. To encrypt bit pattern, m, compute
   $c = m^e \bmod n$ (i.e., remainder when m^e is divided by n)

2. To decrypt received bit pattern, c, compute
   $m = c^d \bmod n$ (i.e., remainder when c^d is divided by n)

Magic happens! $m = (m^e \bmod n)^d \bmod n$

RSA example:

Bob chooses $p=5$, $q=7$. Then $n=35$, $z=24$.
$e=5$ (so e, z relatively prime).
$d=29$ (so $ed-1$ exactly divisible by z.

encrypt:

letter	m	m^e	$c = m^e \bmod n$
l	12	1524832	17

decrypt:

c	c^d	$m = c^d \bmod n$	letter
17	481968572106750915091411825223072000	12	l

RSA: Why: $m = (m^e \bmod n)^d \bmod n$

Number theory result: If p,q prime, $n = pq$, then

$$x^y \bmod n = x^{y \bmod (p-1)(q-1)} \bmod n$$

$$(m^e \bmod n)^d \bmod n = m^{ed} \bmod n$$

$$= m^{ed \bmod (p-1)(q-1)} \bmod n$$

(using number theory result above)

$$= m^1 \bmod n$$

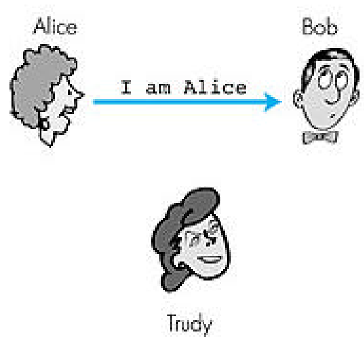
(since we chose ed to be divisible by $(p-1)(q-1)$ with remainder 1)

$$= m$$

Authentication

Goal: Bob wants Alice to "prove" her identity to him

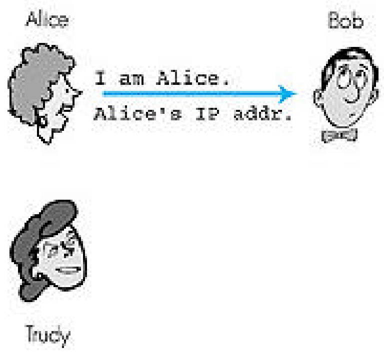
Protocol ap1.0: Alice says "I am Alice"



Failure scenario??

Authentication: another try

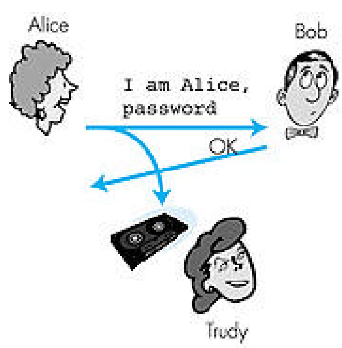
Protocol ap2.0: Alice says "I am Alice" and sends her IP address along to "prove" it.



Failure scenario??

Authentication: another try

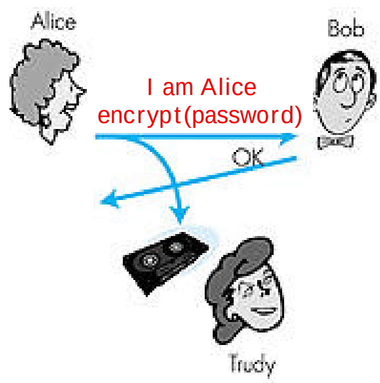
Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



Failure scenario?

Authentication: yet another try

Protocol ap3.1: Alice says "I am Alice" and sends her *encrypted* secret password to "prove" it.



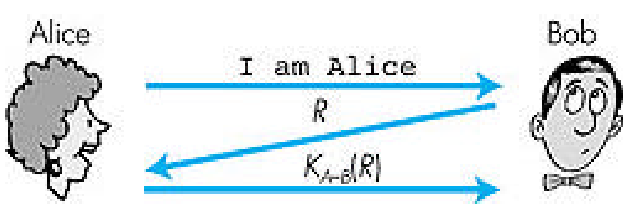
Failure scenario?

Authentication: yet another try

Goal: avoid playback attack

Nonce: number (R) used onlyonce in a lifetime

ap4.0: to prove Alice "live", Bob sends Alice nonce, R. Alice must return R, encrypted with shared secret key



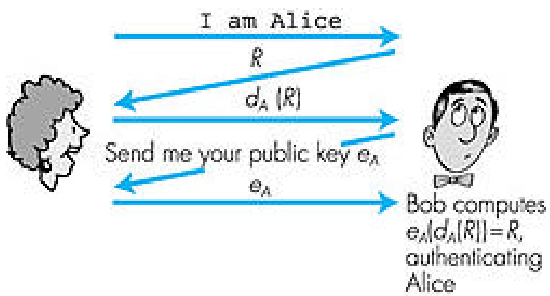
Failures, drawbacks?

Authentication: ap5.0

ap4.0 requires shared symmetric key

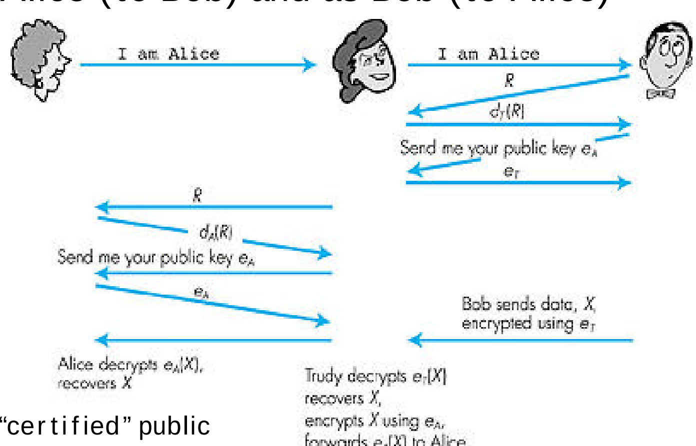
- problem: how do Bob, Alice agree on key
- can we authenticate using public key techniques?

ap5.0: use nonce, public key cryptography



ap5.0: security hole

Man (woman) in the middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)



Need "certified" public keys (more later …)

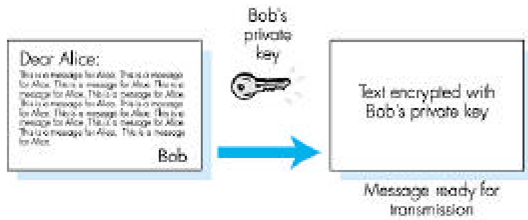
Digital Signatures

Cryptographic technique analogous to hand-written signatures.

- Sender (Bob) digitally signs document, establishing he is document owner/creator.
- Verifiable, nonforgeable: recipient (Alice) can verify that Bob, and no one else, signed document.

Simple digital signature for message m:

- Bob encrypts m with his public key d_B, creating signed message, $d_B(m)$.
- Bob sends m and $d_B(m)$ to Alice.



7: Network Security 25

Digital Signatures (more)

- Suppose Alice receives msg *m*, and digital signature $d_B(m)$
- Alice verifies *m* signed by Bob by applying Bob's public key e_B to $d_B(m)$ then checks $e_B(d_B(m)) = m$.
- If $e_B(d_B(m)) = m$, whoever signed *m* must have used Bob's private key.

Alice thus verifies that:

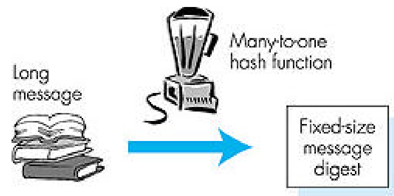
- Bob signed *m*.
- No one else signed *m*.
- Bob signed m and not *m'*.

Non-repudiation:

- Alice can take *m*, and signature $d_B(m)$ to court and prove that Bob signed *m*.

7: Network Security 26

Message Digests



Computationally expensive to public-key-encrypt long messages

Goal: fixed-length,easy to compute digital signature, "fingerprint"

- apply hash function H to *m*, get fixed size message digest, *H(m)*.

Hash function properties:

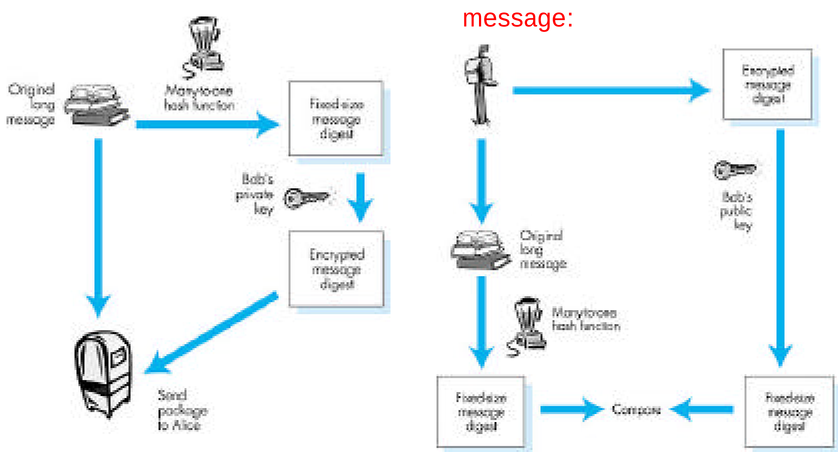
- Many-to-1
- Produces fixed-size msg digest (fingerprint)
- Given message digest x, computationally infeasible to find m such that x = H(m)
- computationally infeasible to find any two messages m and m' such that H(m) = H(m').

7: Network Security 27

Digital signature = Signed message digest

Bob sends digitally signed message:

Alice verifies signature and integrity of digitally signed message:



7: Network Security 28

Hash Function Algorithms

- Internet checksum would make a poor message digest.
  - Too easy to find two messages with same checksum.
- MD5 hash function widely used.
  - Computes 128-bit message digest in 4-step process.
  - arbitrary 128-bit string x, appears difficult to construct msg m whose MD5 hash is equal to x.
- SHA-1 is also used.
  - US standard
  - 160-bit message digest

7: Network Security 29

Trusted Intermediaries

Problem:
- How do two entities establish shared secret key over network?

Solution:
- trusted key distribution center (KDC) acting as intermediary between entities

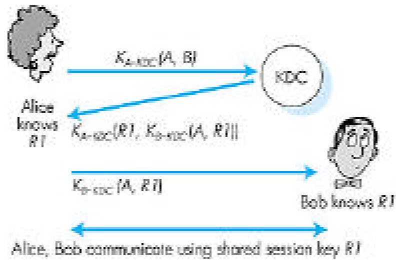
Problem:
- When Alice obtains Bob's public key (from web site, e-mail, diskette), how does she know it is Bob's public key, not Trudy's?

Solution:
- trusted certification authority (CA)

7: Network Security 30

Key Distribution Center (KDC)

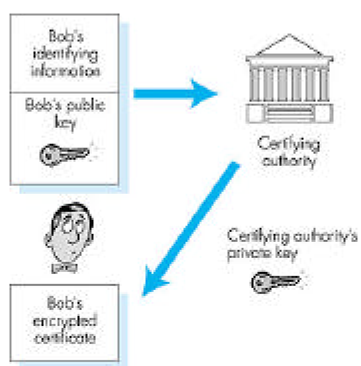
- Alice,Bob need shared symmetric key.
- KDC: server shares different secret key with each registered user.
- Alice, Bob know own symmetric keys, $K_{A\text{-}KDC}$ $K_{B\text{-}KDC}$, for communicating with KDC.



- Alice communicates with KDC, gets session key R1, and $K_{B\text{-}KDC}$(A,R1)
- Alice sends Bob $K_{B\text{-}KDC}$(A,R1), Bob extracts R1
- Alice, Bob now share the symmetric key R1.

Certification Authorities

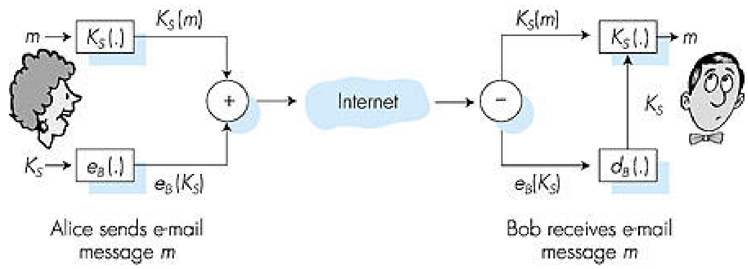
- Certification authority (CA) binds public key to particular entity.
- Entity (person, router, etc.) can register its public key with CA.
  - Entity provides "proof of identity" to CA.
  - CA creates certificate binding entity to public key.
  - Certificate digitally signed by CA.



- When Alice wants Bob's public key:
- gets Bob's certificate (Bob or elsewhere).
- Apply CA's public key to Bob's certificate, get Bob's public key

Secure e-mail

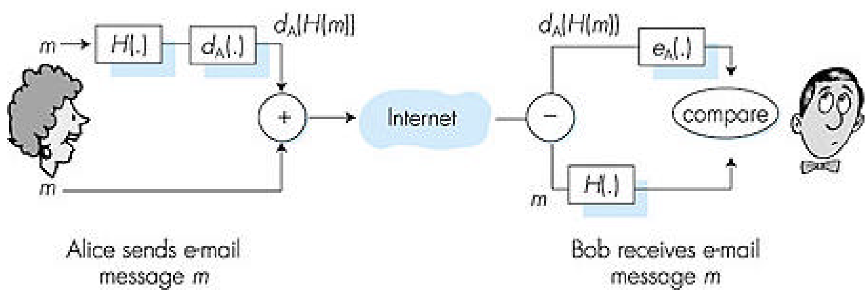
- Alice wants to send secret e-mail message, m, to Bob.



- generates random symmetric private key, K_S.
- encrypts message with K_S
- also encrypts K_S with Bob's public key.
- sends both $K_S(m)$ and $e_B(K_S)$ to Bob.

Secure e-mail (continued)

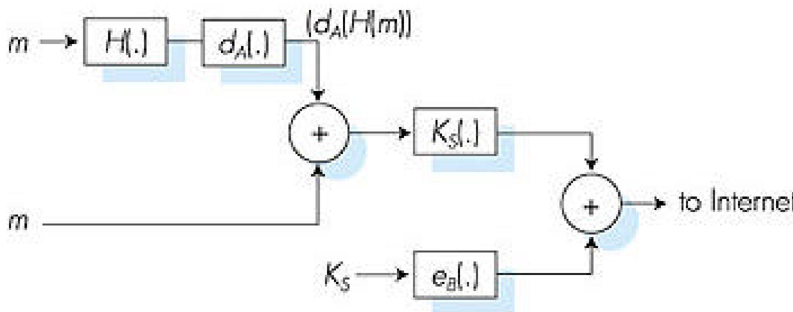
- Alice wants to provide sender authentication message integrity.



- Alice digitally signs message.
- sends both message (in the clear) and digital signature.

Secure e-mail (continued)

- Alice wants to provide secrecy, sender authentication, message integrity.



Note: Alice uses both her private key, Bob's public key.

Pretty good privacy (PGP)

- Internet e-mail encryption scheme, a de-facto standard.
- Uses symmetric key cryptography, public key cryptography, hash function, and digital signature as described.
- Provides secrecy, sender authentication, integrity.
- Inventor, Phil Zimmerman, was target of 3-year federal investigation.

A PGP signed message:

```
---BEGIN PGP SIGNED MESSAGE---
Hash: SHA1

Bob:My husband is out of town
   tonight.Passionately yours,
   Alice

---BEGIN PGP SIGNATURE---
Version: PGP 5.0
Charset: noconv
yhHJRHhGJGhgg/12EpJ+lo8gE4vB3mqJ
   hFEvZP9t6n7G6m5Gw2
---END PGP SIGNATURE---
```

Secure sockets layer (SSL)

- PGP provides security for a specific network app.
- SSL works at transport layer. Provides security to any TCP-based app using SSL services.
- SSL: used between WWW browsers, servers for I-commerce (shttp).
- SSL security services:
  - server authentication
  - data encryption
  - client authentication (optional)
- Server authentication:
  - SSL-enabled browser includes public keys for trusted CAs.
  - Browser requests server certificate, issued by trusted CA.
  - Browser uses CA's public key to extract server's public key from certificate.
- Visit your browser's security menu to see its trusted CAs.

7: Network Security 37

SSL (continued)

Encrypted SSL session:

- Browser generates symmetric session key, encrypts it with server's public key, sends encrypted key to server.
- Using its private key, server decrypts session key.
- Browser, server agree that future msgs will be encrypted.
- All data sent into TCP socket (by client or server) i encrypted with session key.
- SSL: basis of IETF Transport Layer Security (TLS).
- SSL can be used for non-Web applications, e.g., IMAP.
- Client authentication can be done with client certificates.

7: Network Security 38

Secure electronic transactions (SET)

- designed for payment-card transactions over Internet.
- provides security services among 3 players:
  - customer
  - merchant
  - merchant's bank

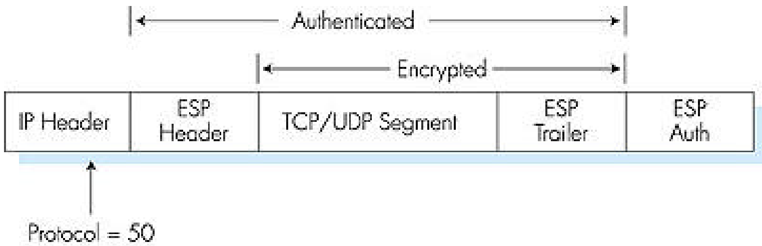
  All must have certificates.
- SET specifies legal meanings of certificates.
  - apportionment of liabilities for transactions
- Customer's card number passed to merchant's bank without merchant ever seeing number in plain text.
  - Prevents merchants from stealing, leaking payment card numbers.
- Three software components:
  - Browser wallet
  - Merchant server
  - Acquirer gateway
- See text for description of SET transaction.

Ipsec: Network Layer Security

- Network-layer secrecy:
  - sending host encrypts the data in IP datagram
  - TCP and UDP segments; ICMP and SNMP messages.
- Network-layer authentication
  - destination host can authenticate source IP address
- Two principle protocols:
  - authentication header (AH) protocol
  - encapsulation security payload (ESP) protocol
- For both AH and ESP, source, destination handshake:
  - create network-layer logical channel called a service agreement (SA)
- Each SA unidirectional.
- Uniquely determined by:
  - security protocol (AH or ESP)
  - source IP address
  - 32-bit connection ID

ESP Protocol

- Provides secrecy, host authentication, data integrity.
- Data, ESP trailer encrypted.
- Next header field is in ESP trailer.
- ESP authentication field is similar to AH authentication field.
- Protocol = 50.

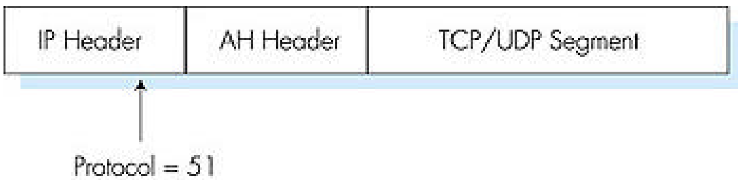


Authentication Header (AH) Protocol

- Provides source host authentication, data integrity, but not secrecy.
- AH header inserted between IP header and IP data field.
- Protocol field = 51.
- Intermediate routers process datagrams as usual.

AH header includes:

- connection identifier
- authentication data: signed message digest, calculated over original IP datagram, providing source authentication, data integrity.
- Next header field: specifies type of data (TCP, UDP, ICMP, etc.)



Network Security (summary)

Basic techniques…...

- cryptography (symmetric and public)
- authentication
- message integrity

…. used in many different security scenarios

- secure email
- secure transport (SSL)
- IP sec

See also: firewalls , in network management