CPS 590.5 Computer Security Lecture 9: Introduction to Network Security

Xiaowei Yang
xwy@cs.duke.edu

Previous lectures

- Worm
- Fast worm design

Today

- Network security
  - Cryptography building blocks
  - Existing secure protocols/applications
- Looking forward
  - What's missing

Overview

- Why studying network security?
- Basic cryptography building blocks
- Security protocols
- Non-cryptography based security: firewalls

The Internet is insecure

- Attackers may eavesdrop, modify, or drop your packets!

Network security

- Confidentiality:
  - Do you want to send your credit card #, login password over the Internet in plaintext?
- Integrity
  - Data integrity: Imagine an Amazon transaction. Do you want your payment to be modified from $10.0 to $100?
  - Replay attack: You do not want the same transaction confirmation to be sent multiple times!
  - Timeliness: delay a stock purchase
- Authenticity
  - Entity authentication: who are you talking to? Phishing attack
  - Message authentication: who sent this message?
- Availability
  - Denial of service attacks
- Non-repudiation
  - You've clicked the confirmation button!

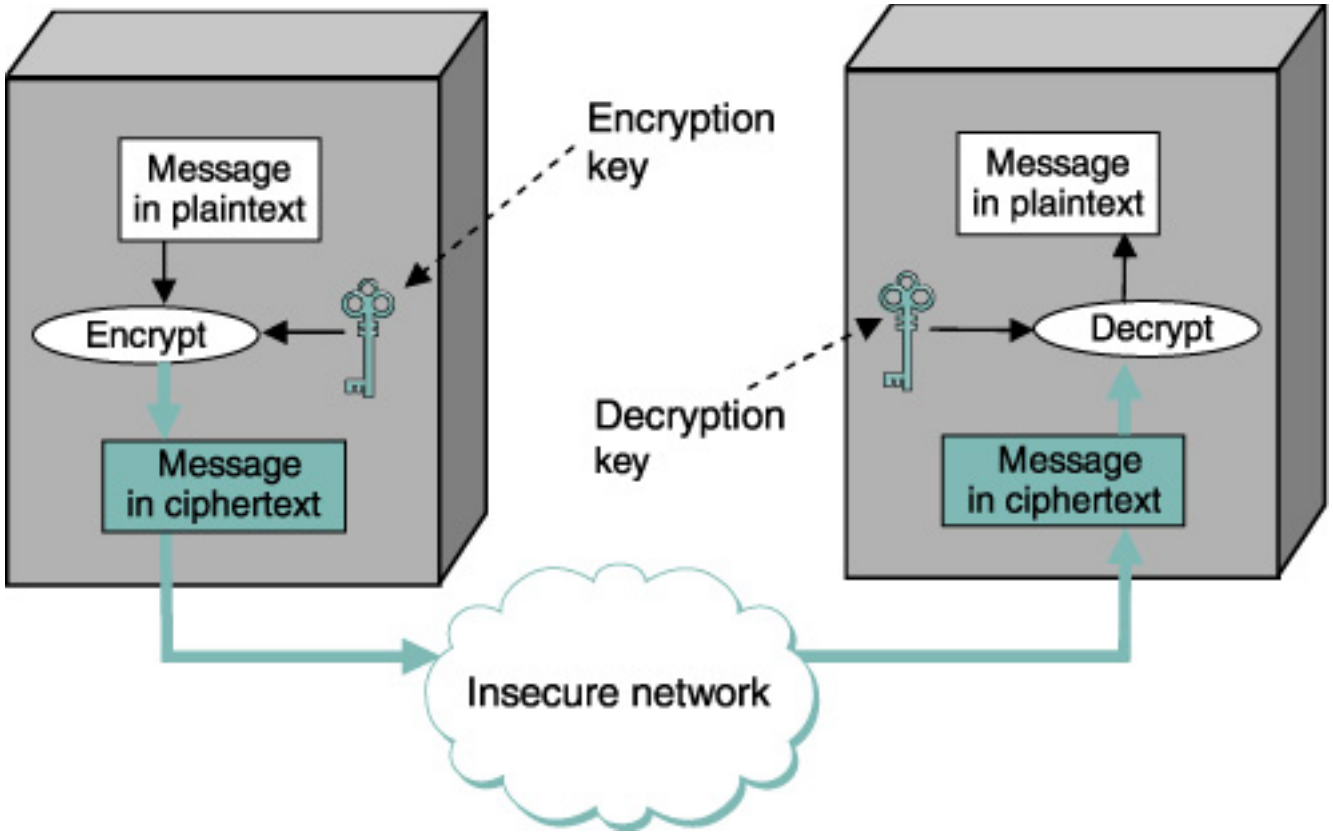
How to address those problems

- Cryptography building blocks
  - Confidentiality
    - Encryption
  - Authenticity
    - Public key signatures
    - Authentication protocols (later)
- Non-cryptographic approach
  - Firewalls

Cryptographic tools

- Cryptographic algorithms
  - Ciphers and Cryptographic hashes
  - Not a solution in themselves, but building blocks from which a solution can be built
- Key distribution
- Protocols built on cryptographic algorithms
  - System builders need to get familiar with the tools

Principles of Ciphers

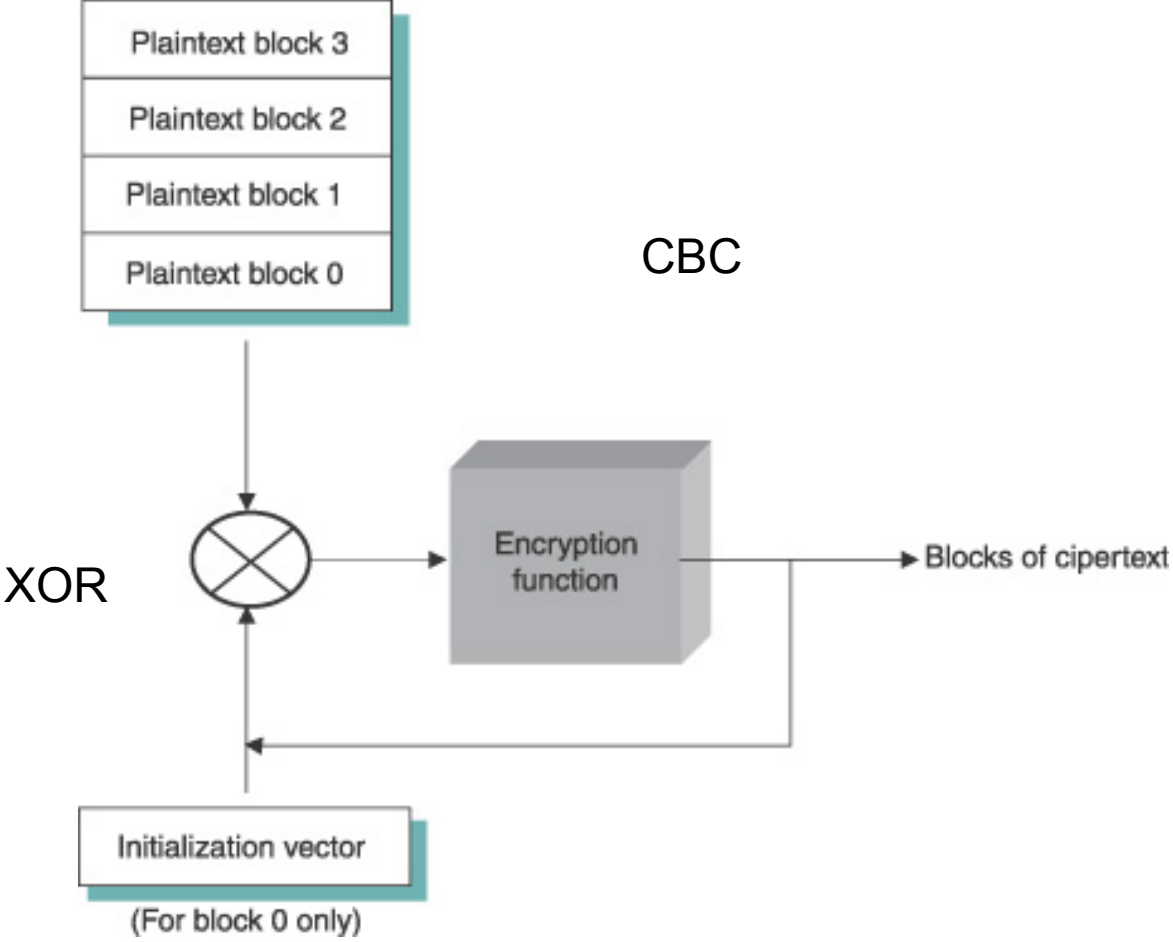


- $Encrypt_{key}(plaintext) \rightarrow ciphertext$
- Ciphertext is unintelligible
- $Decrypt_{key}(ciphertext) \rightarrow plaintext$
- The transformation is called a cipher

Security of a Cipher

- Encrypt() and Decrypt() are public knowledge
- Only key is secret
- Designing a cipher is like a black art
- No news is good news

- Cryptanalysis (code breaking)
  - Known plaintext
    - Know the plaintext and its encrypted version and make use of them to guess other part of secrete information such as secrete keys
  - Chosen plaintext analysis
    - An attacker can get arbitrary plaintext encrypted
    - Some plaintext has known vulnerabilitites



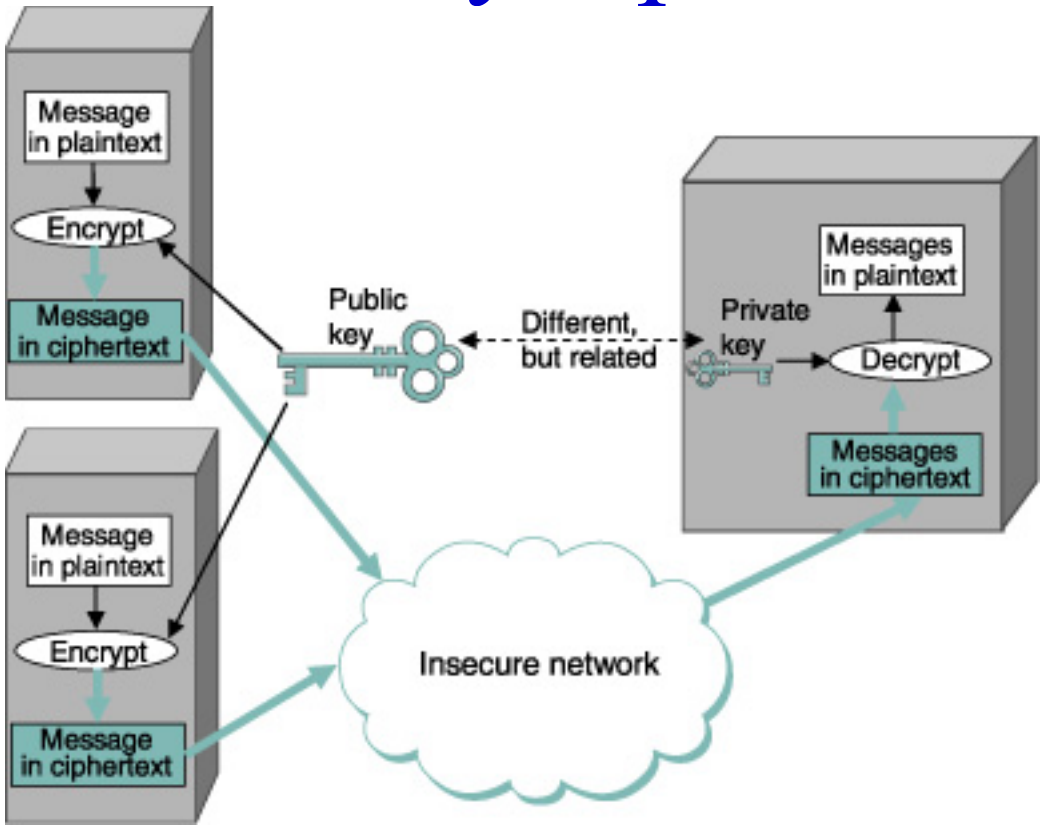
- Input is a fixed size block of text, eg, 64-128 bits
- Modes of operation
  - Electronic codebook (ECB) mode: each block is encrypted independently
    - The same block value will always result in the same cipher text block
  - Cipher block chaining
    - Each plaintext block is XORed with the previous block's ciphertext before being encrypted
    - Context dependent

Standard symmetric-key ciphers

- National Institute of Standards and Technology (NIST) issued ciphers

- Data encryption standard (DES)
  - 56-bit key
  - 64-bit block size
  - Insecure against brute-force attacks
- Triple DES (3DES)
  - First encrypt using DES-key1, decrypt using DES-key2, and encrypt using DES-key3
  - Backward compatible: can be decrypted by DES
- Advanced encryption standard (AES)
  - Originally named Rijndael
  - 128, 192, 256-bits

Public-key ciphers

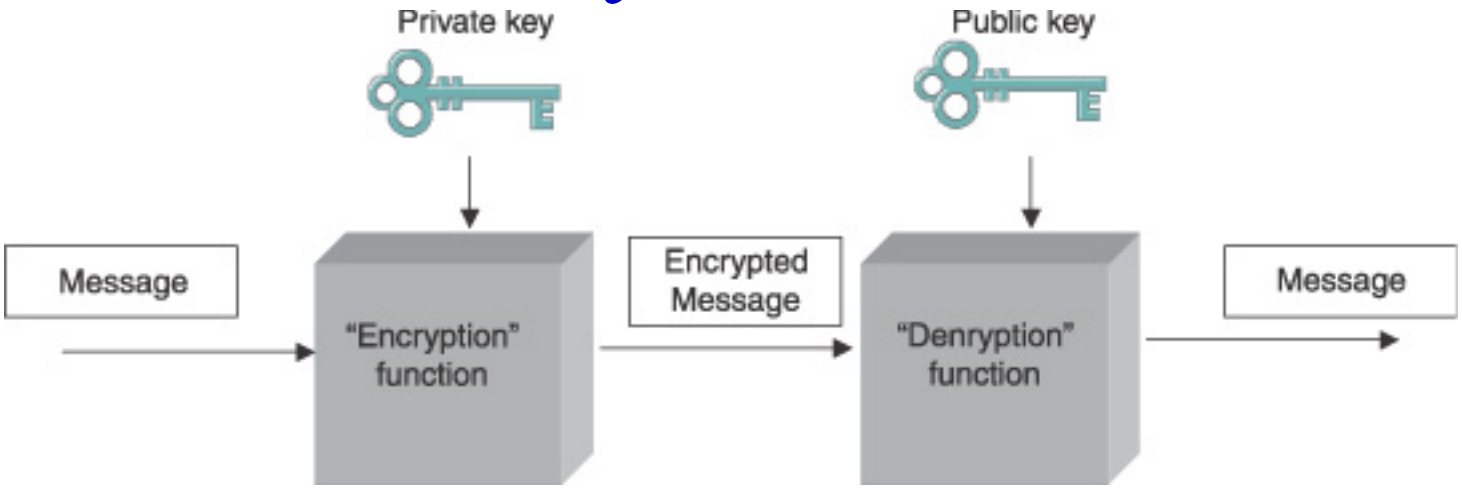


- RSA
  - Difficult to factor large numbers
  - Key length >= 1024 bits
- ElGamal
  - Discrete logarithm is hard
  - Key length >= 1024 bits
- Public-key ciphers are orders of magnitude slower than symmetric cipher

Cryptography building blocks

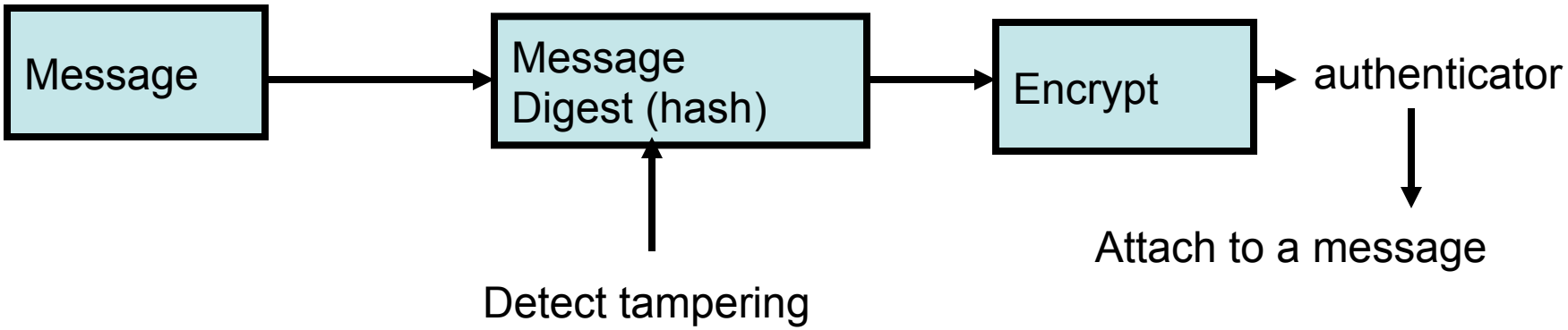
- Confidentiality
  - Encryption
- Authenticity
  - Public key signatures
  - Authentication protocols (later)

Public key authentication



- Everyone can validate who sends the message
- Not good enough
  - No data integrity: modifying a ciphertext message may produce a readable output
    - "I owe you $10" → "I owe you $100000"

Authenticators



- Problem: Encryption alone does not provide data integrity
  - Modifying a cipher may still allow decrypting to a valid plaintext

- An authenticator is a value, to be included in a transmitted message that can be used to verify simultaneously the authenticity and the data integrity of a message
  - Why are these two properties combined?

- 1. Message digest + encryption
  - Modifying the message cannot produce the correct authenticator

Authenticator methods

- Asymmetric cryptography
  - Digital signatures
- Symmetric cryptography
  - Message authentication code (MAC)
    - Another MAC!

Hash functions

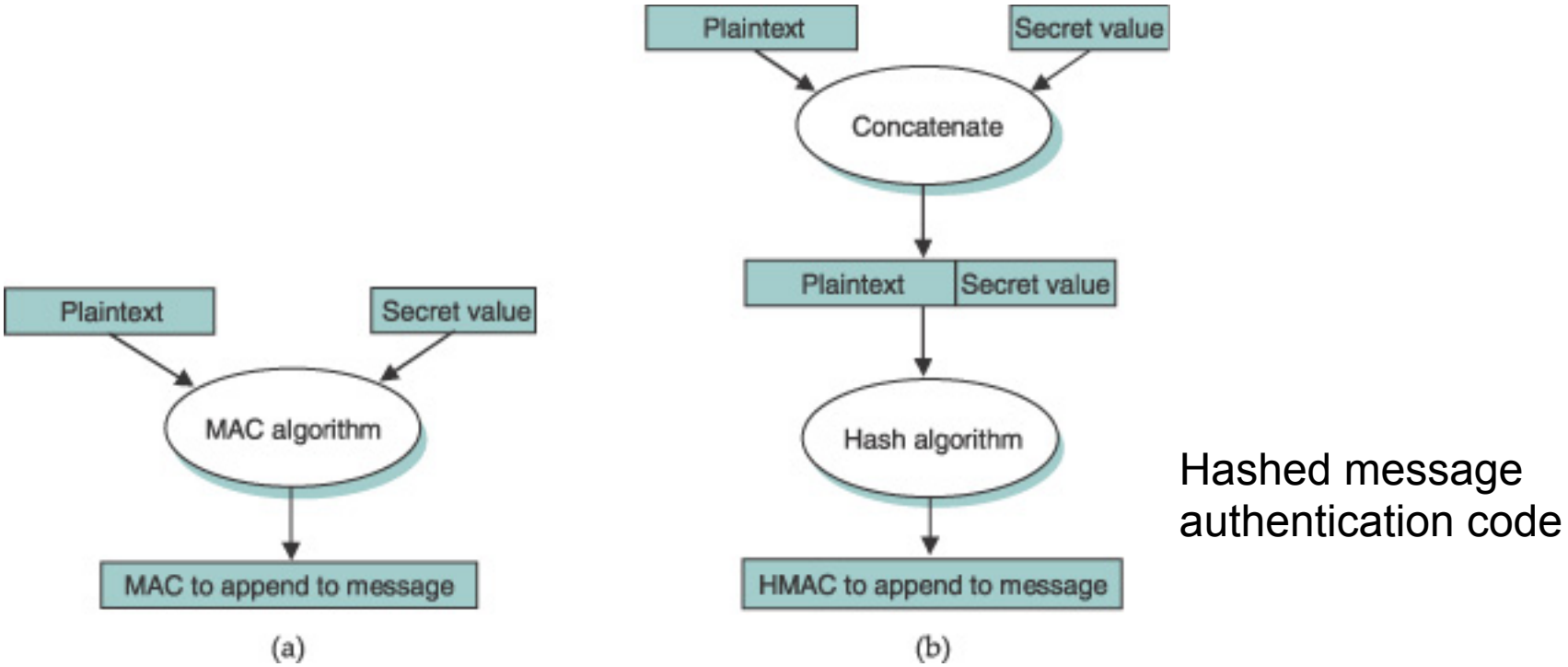
- A secure one-way function f(x)
  - Knowing f(x) gives little knowledge about x

- Collision attacks
  - Attacks finding any collision
- Preimage attacks
  - A 2nd message that collides with a given first message
- Common ones: MD5, SHA-1
  - SHA-1 expires by 2010
  - SHA-2

Digital signatures

- A digest encrypted using the private key of a public-key algorithm
- Common digital signatures
  - Digital signature standard (DSS)
    - May use any one of three public-key ciphers
    - RSA, ElGamal, Elliptic Curve Digital Signature Algorithm

Authenticators – Message Authentication Code



Hashed message authentication code

- Instead of encrypting a hash, it uses a hash-like function that takes a secret value (known only to the sender and the receiver) as a parameter.
- How does two ends obtain the key?
- Security of HMAC: what if hash's not one-way?

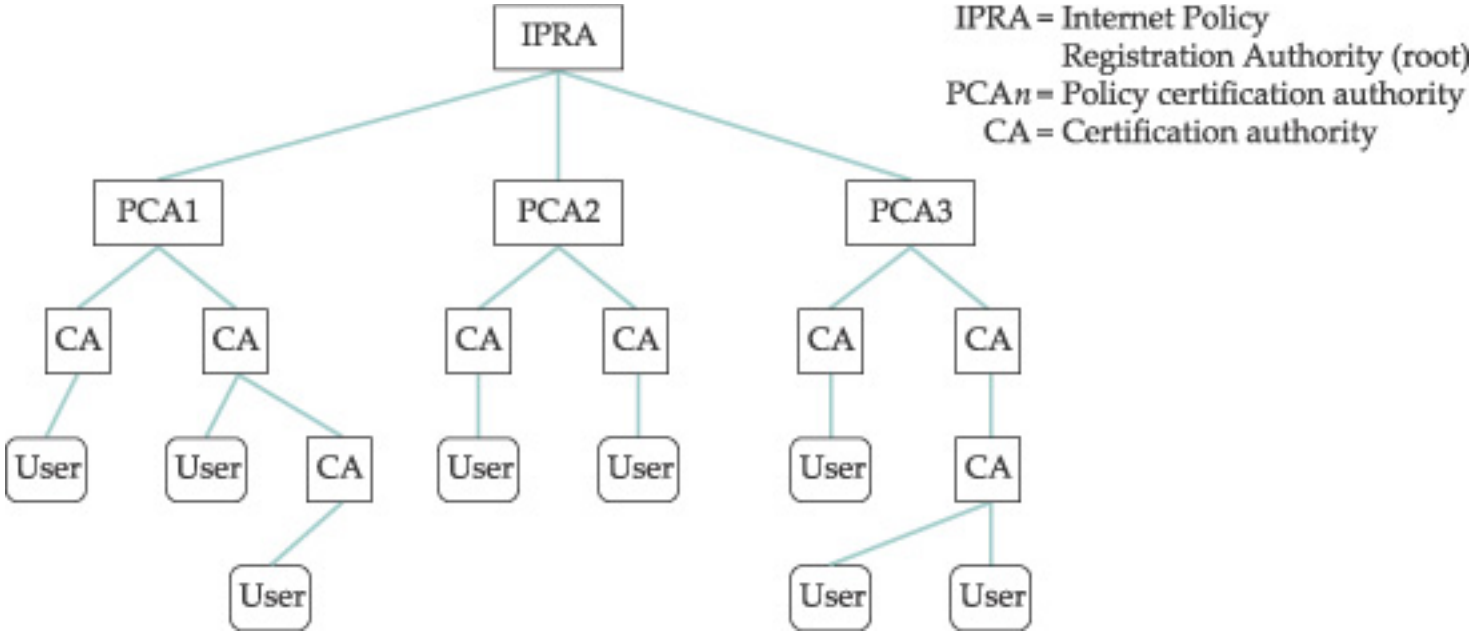
Key distribution

- Two problems:
  - How do participants know which entity has which public key?
    - A complete scheme for certifying bindings between public keys and identities – what keys belongs to who – is called a public key infrastructure (PKI)
      - Comments: not easy to scale
      - People don't use it that much
  - How do each end know the symmetric shared key?

Distributing public keys

- A public-key certificate is a digitally signed statement that binds the identity of the entity to a public key
- If A trusts B, and knows B's public key, then A can learn C's public key if B issues a public key certification of C
- X.509 certificate
  - The ID of the entity
  - The public key of the entity
  - The identity of the signer
  - The digital signature
  - A digital signature algorithm
  - Optional: expiration time

Certification authorities



- A CA is an entity claimed to be trustworthy to verify identities and issuing public key certificates
  - Verisign
- CAs can be organized into a tree
- Trust is binary: yes or no
  - Everyone trusts the root

Multiple CAs

- In the real world, there is no single rooted trust
- Multiple CAs whose public keys are trusted by different people
- Self-certifying certificates
  - Signer is self
  - Accepted by TLS

Web of Trust

- Pretty Good Privacy:
  - No single hierarchical
  - Establishing trust is a personal matter and gives users the raw material to make their own decisions

- IETF's PGP signing session:
  - Collect public keys from others whose identity one knows
  - Provide his public key to others
  - Get his public key signed by others
  - Sign the public key of others
  - Collect the certificate from other individuals whom he trusts enough to sign keys

- Trust is a matter of degree
  - A public-key certificate includes a confidence level
  - Trust dependent on the number of certificates of a key, and the confidence level of each certificate

Certificate Revocation

- Certificate revocation list
  - Periodically updated and publicly available
  - Digitally signed
  - Lists may be large
- Online certificate status protocol
  - Query the status of a certificate

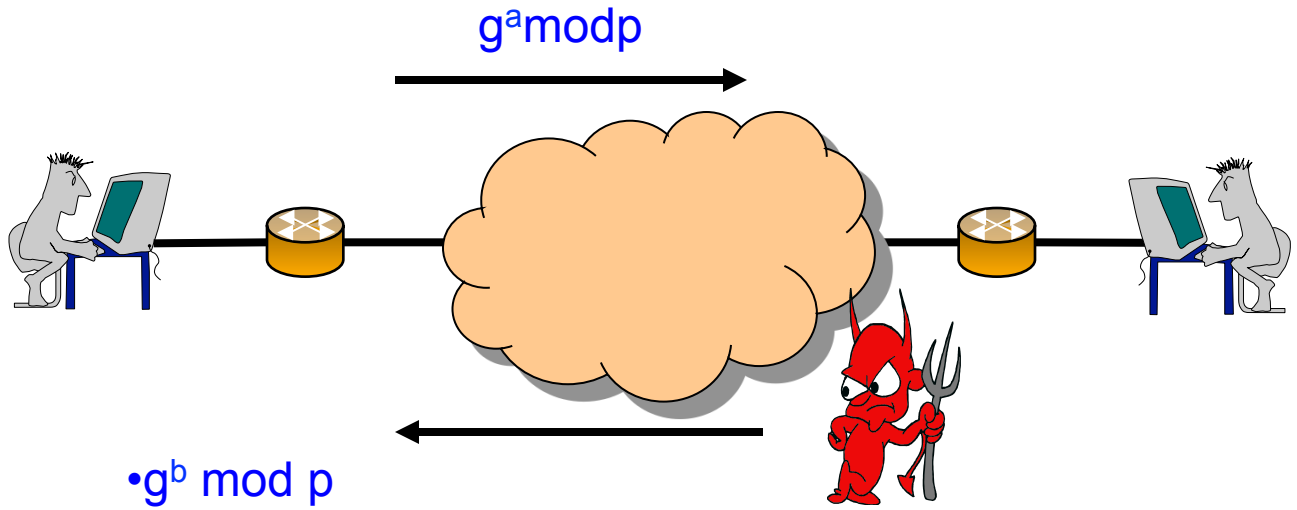
Key distribution

- Two problems:
  - How do participants know which entity has which public key?
    - A complete scheme for certifying bindings between public keys and identities – what keys belongs to who – is called a public key infrastructure (PKI)
      - Comments: not easy to scale
      - People don't use it that much
  - How do each end know the symmetric shared key?

Symmetric key distributions

- If there are N entities, N(N-1)/2 keys
- Key distribution center (KDC)
  - A trusted entity
  - Each user maintains a key with the KDC
  - KDC generates a session key when a user wants to communicate with another destination
    - Kerberos is a widely used key-distribution system

Diffie-Hellman key agreement

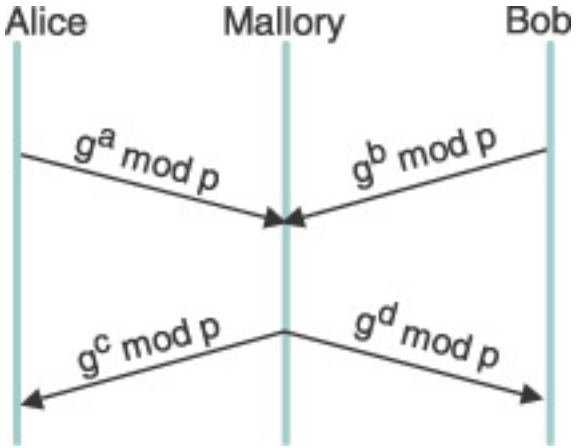


- Long considered as the invention of public key cryptography
- Establishes a session key without using any predistributed keys
- Discrete log is hard

Diffie-Hellman Key Agreement

- Two parameters: g, and p
  - p: a prime; g: a primitive root of p s.t. for every number of n from 1 through p-1 there must be some value k such that $n=g^k \bmod p$
  - $1=2^0 \bmod 5$, $2 = 2^1 \bmod 5$, $3=2^3 \bmod 5$, $4=2^2 \bmod 5$
- Alice picks a private value a, and sends $g^a \text{mod} p$
- Bob picks b, ands sends $g^b \bmod p$
- $g^{ab} \bmod p = g^{ba} \bmod p$
- Discrete log is hard
  - Attackers cannot guess a, or b, even when they see $g^a \bmod p$ or $g^b \bmod p$

Man in the middle attack



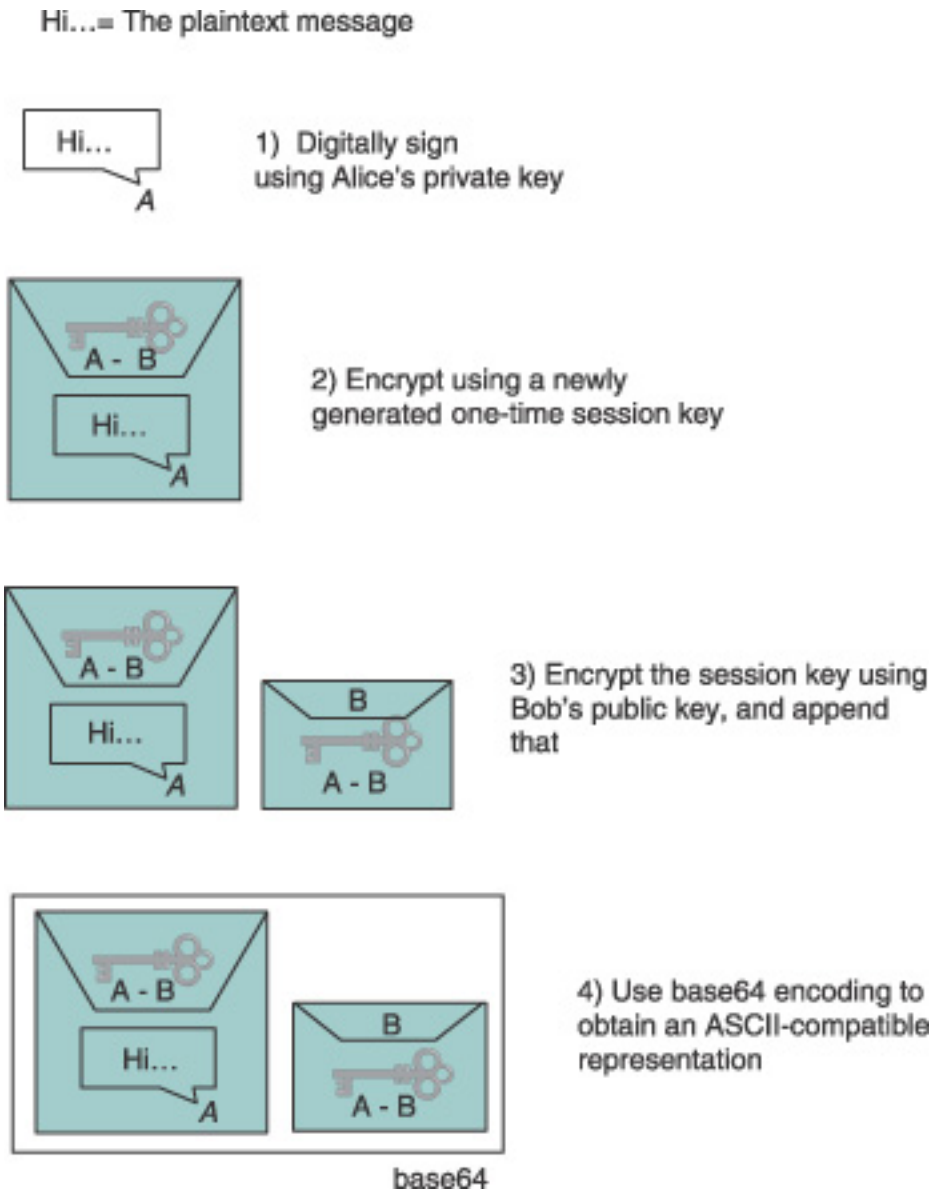
- Fixed DH: Alice and Bob has fixed a, and b values
- $g^a \bmod p$ is certified

Overview

- Why studying network security?
  - The topic itself is worth another class
- Basic cryptography building blocks
- Security protocols
  - Case studies of using cryptography to build secure systems
- Non-cryptography based security: firewalls

Secure systems

- PGP for email security
  - Works well for email
  - Could be replayed, but a user can detect
  - No need of prior message exchange
  - Confidentiality
  - Does not prove Alice is talking to Bob



Secure Shell (ssh)

- Both the SSH protocol and applications that use it
- Widely used
- Replacing telnet, rsh, rcp
  - No need to send plaintext passwords to authenticate
  - Prior to it, passwords are sent in plaintext!

Components of SSH2

- SSH 2 consists of three protocols
  - SSH-TRANS: a transport layer protocol
    - Provides an encrypted channel between the client and server machines
    - A secure channel is established before a client authenticates
    - A client authenticates the server to establish the channel
  - SSH-AUTH, an authentication protocol
    - Client authenticates to the server
    - You may type your password! SSH-TRANS takes care of encryption
  - SSH-CONN, a connection protocol
    - Used for port forwarding

SSH's server key distribution

- A server tells the client its public key at connection time
  - Attackers are not always present
  - Vulnerability window is small
- The first time a key is sent, ssh asks the user
- If accepts, ssh remembers the key and compares the stored key with an offered key in a subsequent connection
  - Prompts the user if changed
  - Otherwise accept

Establishing a secure channel

- After a client authenticates the server
- Establish a session key
- All subsequent messages are encrypted using the session key

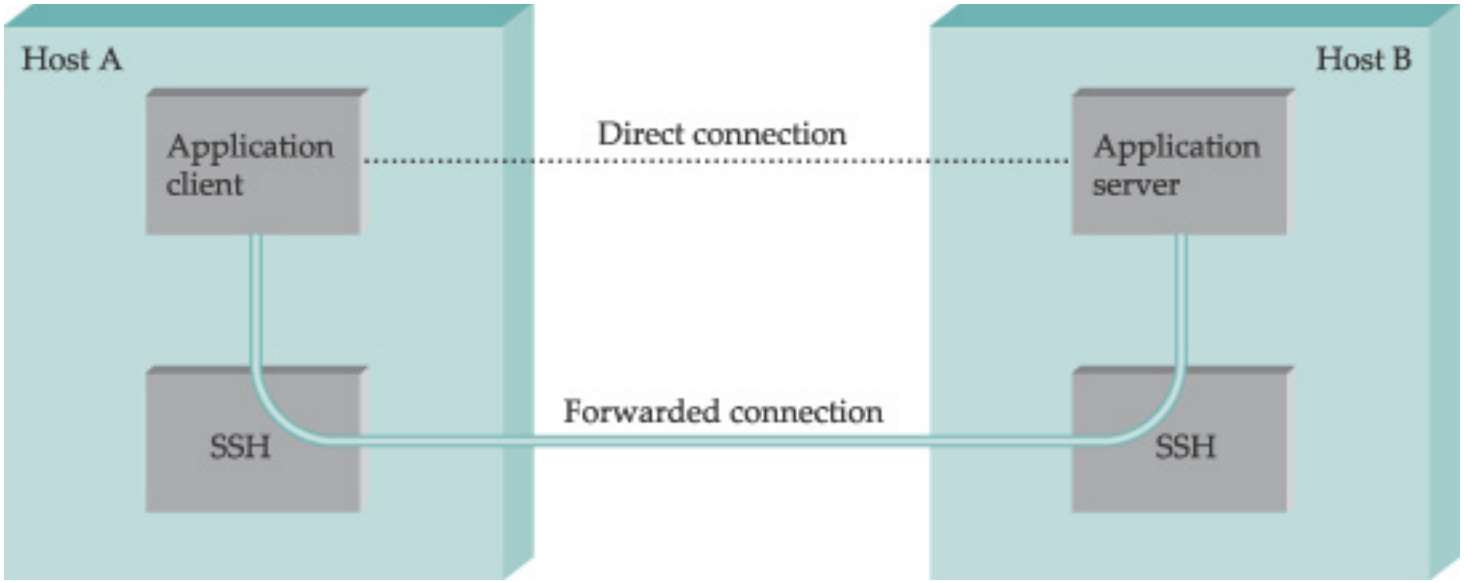
SSH's client authentication

- Password
  - A secure channel is already established!
- Public key encryption
  - Places your public key in ~/.ssh/authorized_keys
- Host authentication
  - A user claiming to be so-and-so from a certain set of trusted hosts is automatically believed to be the same user on the server
  - The client host authenticates itself to the server
    - SSH-TRANS only authenticates server by default
    - User claims to be so-and-so from a set of trusted hosts is believed to be the same user on the server

SSH login w/o typing in your passwords

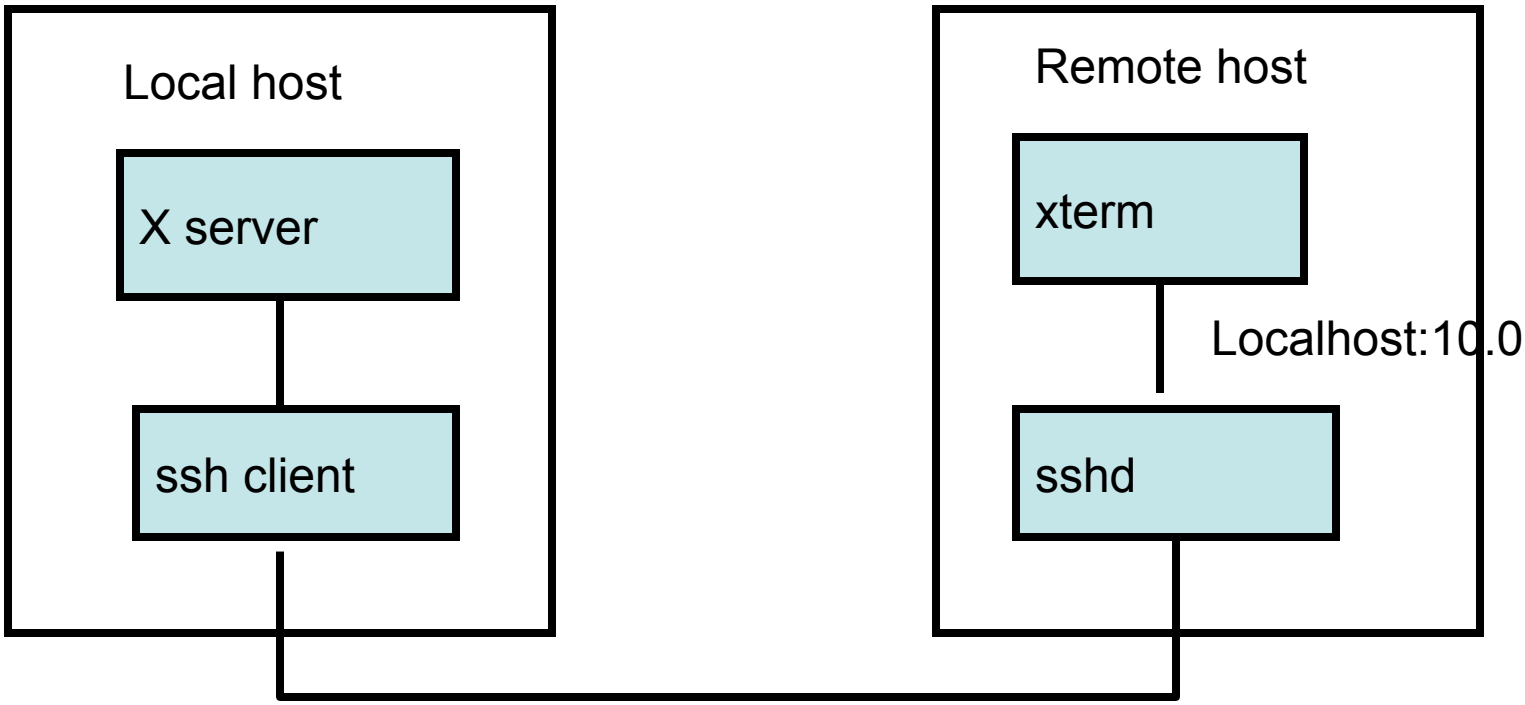
- Use ssh-keygen to generate a public/private key pair
  - ssh-keygen –t dsa
- Append .id_dsa.pub to .ssh/authorized_keys on the server
  - scp ~/.ssh/id_dsa.pub linux1.cs.duke.edu:~/
  - ssh linux1.cs.duke.edu “cat ~/id_dsa.pub >> ~/.ssh/authorized_keys”

SSH port forwarding



- SSH can be used to establish a secure channel between two hosts using the SSH-CONN protocol
- Secure legacy applications

Example: X11 forwarding



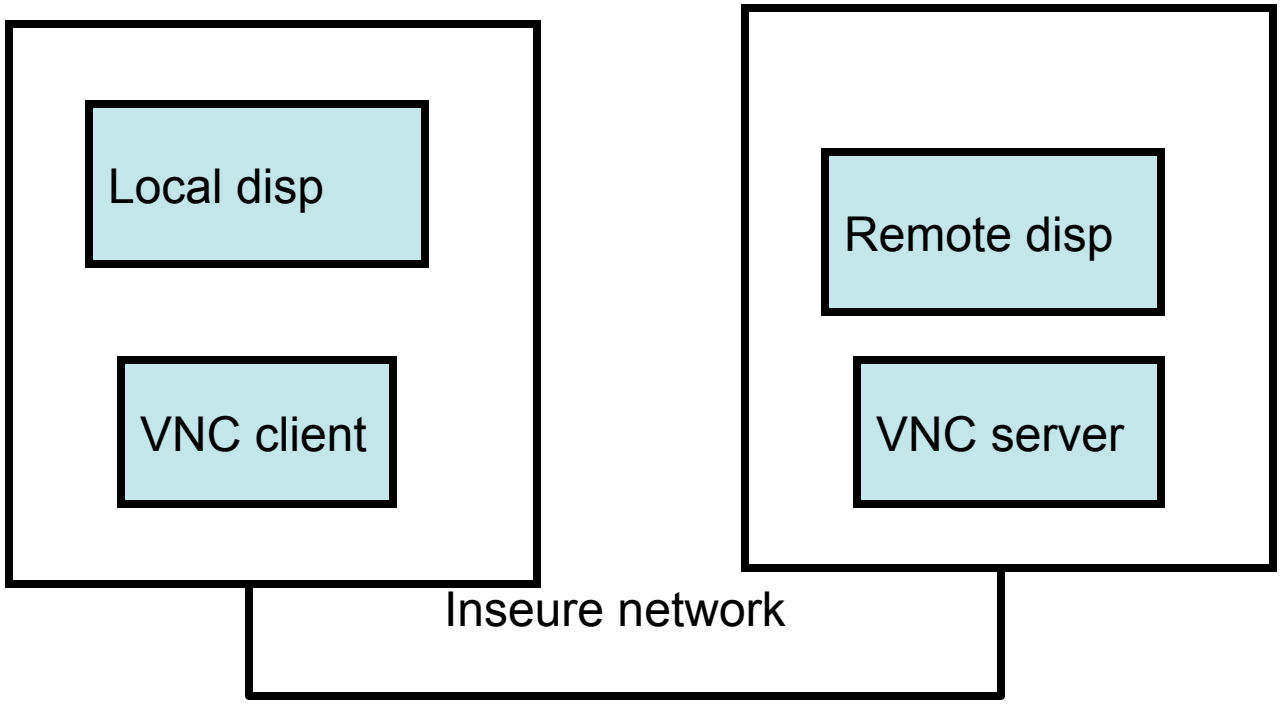
- ssh –X remote-host
- sshd at the remote host creates a TCP listening socket (6010), and X authentication information, and set your display variable to localhost:10.0
- When you type xterm, it speaks the X11 protocol with the sshd faked X server port (10.0), and sshd forwards it back to the ssh client at local host.
- The ssh client forwards back to the X server running on your local host
- You see the xterm displays on your local host, and all commands you type in the xterm is encrypted!

SSH port forwarding

- Some legacy applications do not have security mechanism built-in: pop3
- How can you read your email without sending your password in cleartext?
  - ssh –L 9999:localhost:110 mail.cs.duke.edu
  - Run your pop3 mail client, and make it use localhost: 9999
  - All commands will be sent via an encrypted connection

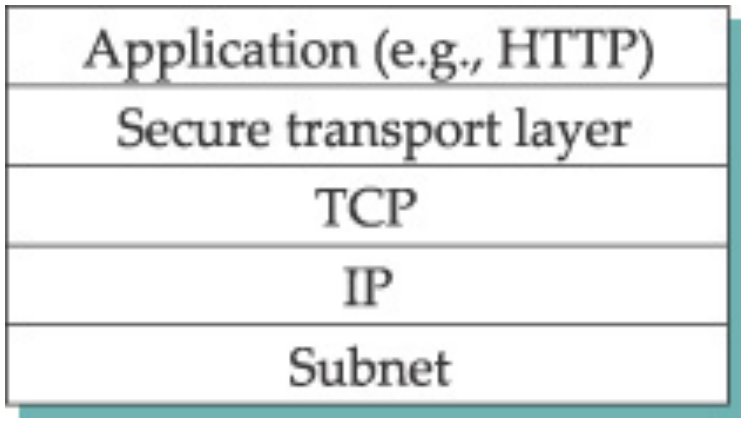
- Pop3 ←→ localhost:9999 ←→ ssh client ←→ mail.cs.duke.edu:sshd ←→ mail.cs.duke.edu:110

The VNC example



- A real world application
  - The free VNC servers do not have encryption
    - Figure out how to do it yourself!
    - Vncviewer sends your password in plaintext a vncserver
      - Unless you purchase the non-free version
      - How can we establish a secure tunnel between the vncclient and server?

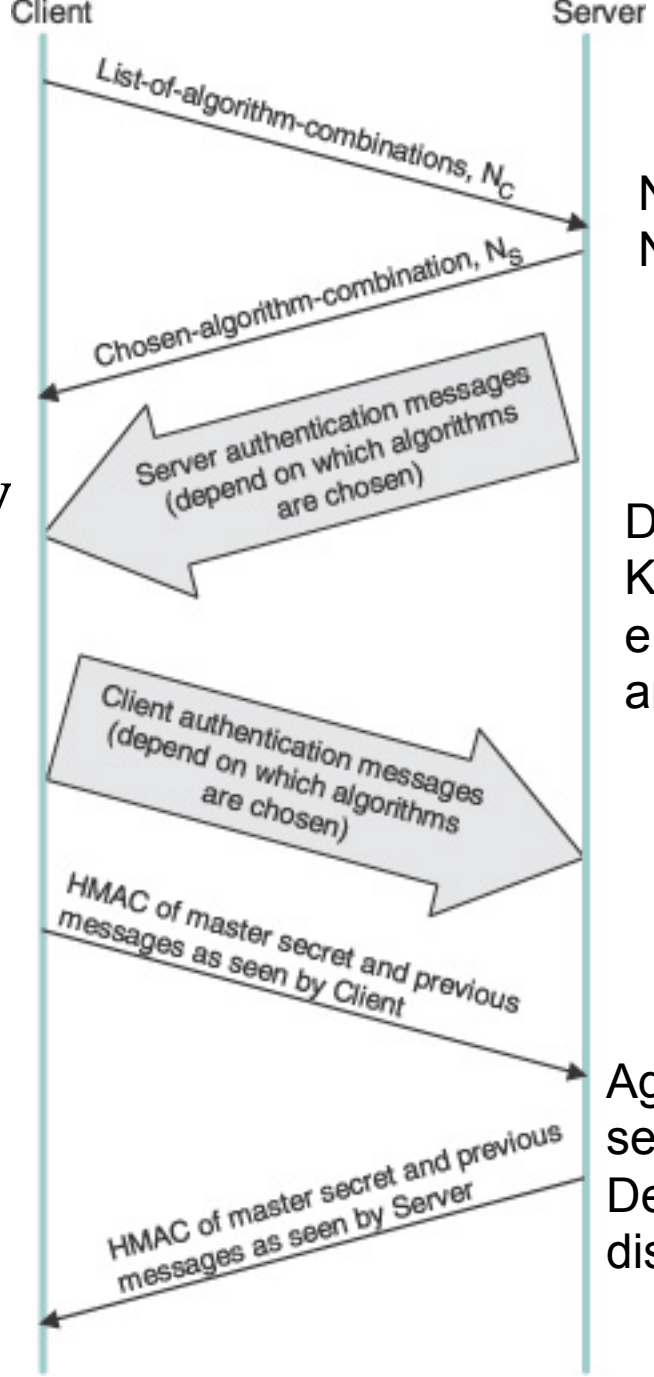
SSL/TLS



- Transport layer security (TLS) is based on Secure Socket Layer (SSL)
- https: port 443
- A handshake protocol for negotiating parameters, and a secret session key
  - Each direction has a key
- A record protocol to transmit messages

The handshake protocol

- Negotiate encryption hash, symmetric key cipher, and session key establishment protocol
  - Mutual authentication
  - Or authenticating one participant only
  - Or no authentication



Nc: client nonce
Ns: server nonce

Diffie-Hellman Key exchange e.g. a certificate and DH parameters

Agree on the master secret
Detect any discrepancy

The record protocol

- Fragmented or coalesced into blocks of a convenient size
- Optionally compressed
- Integrity-prote
  each record
- Encrypted usi
- Passed to the transport layer
- Each record message has its own sequence number to prevent replay attacks.
  - TCP's sequence number is not sufficient!
  - An active adversary can modify TCP sequence numbers in packets!

Comments: quite heavy
https://gmail.com was not the default for a long time

IPSec

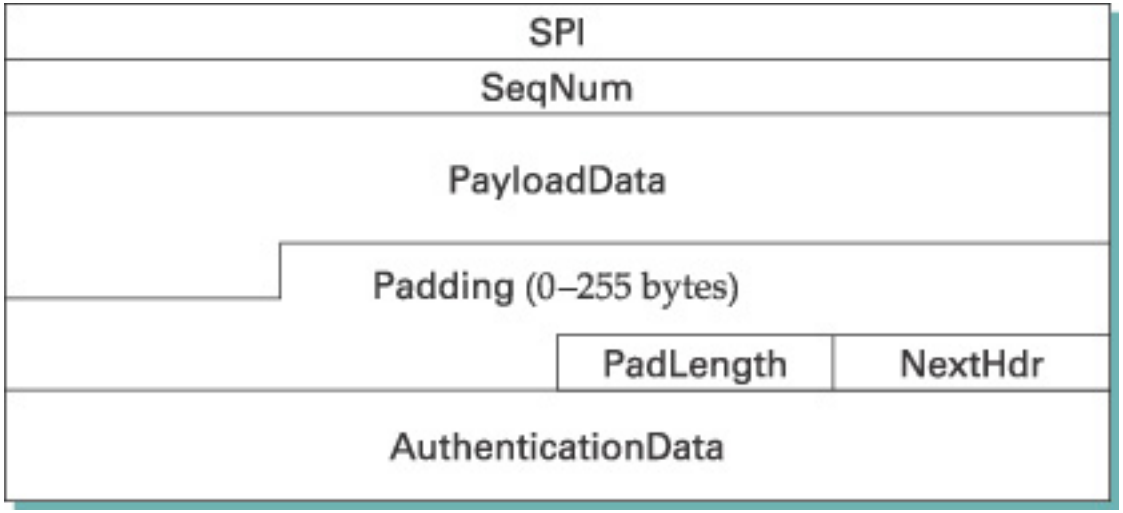
- A framework specifies how to secure all IP traffic between two machines

- Two parts
  - Security services: Authentication header (AH): rarely used, Encapsulating Security Payload (ESP)
  - Key management: Internet security association and key management protocol (ISAKMP).
    - Defines message format, not the detailed key generation algos.

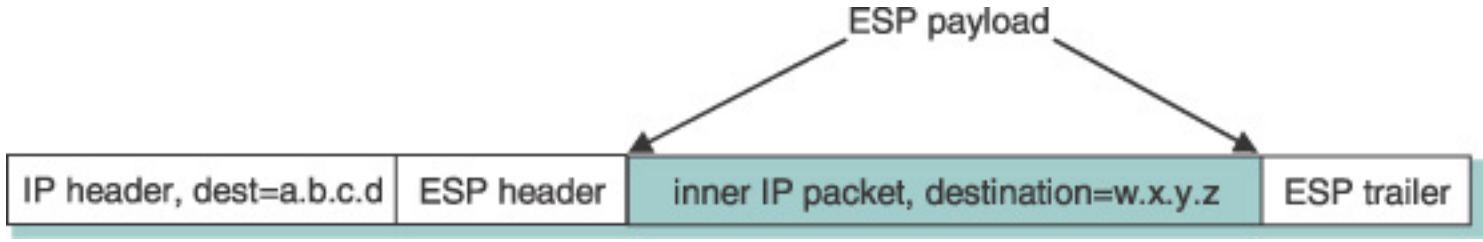
Security Association

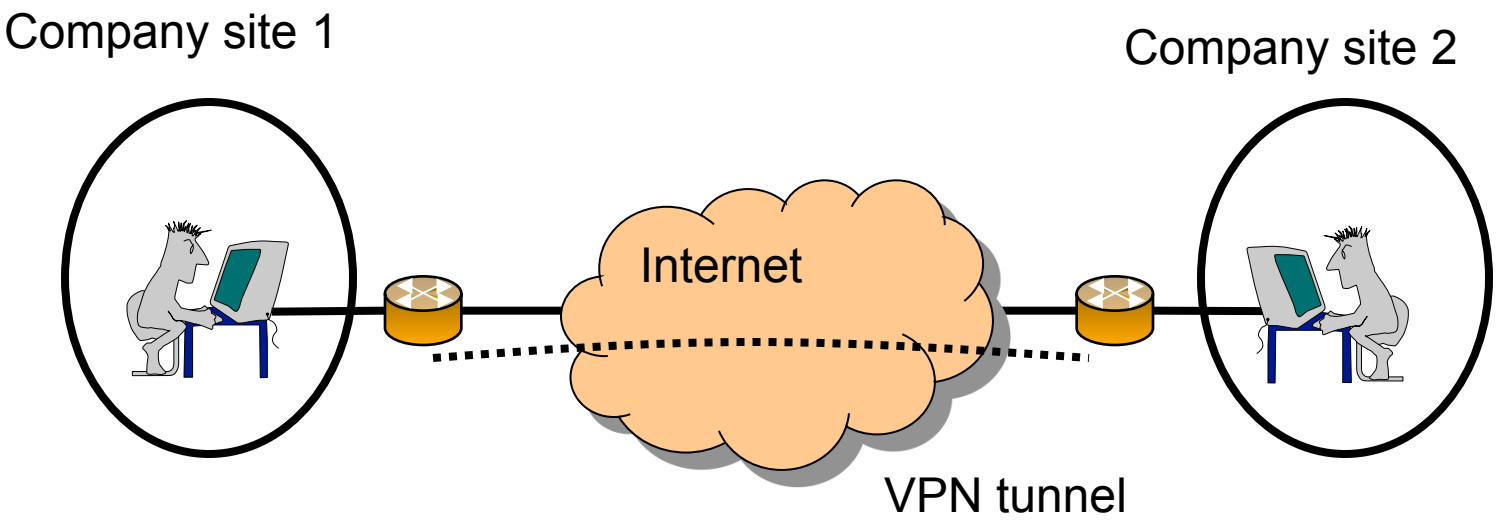
- A security association (SA) is created for each direction
  - IP is connectionless, but IPSec is not
  - An SA includes connection state such as keys, and sequence numbers
  - An SA is identified by a security parameter SPI (a multiplexing key)
  - SPI and destination address identifies an SA
  - SAs are established, negotiated, modified and deleted using ISAKMP
  - Internet Key Exchange (IKE) is one key exchange protocol

The ESP header



- Padding is necessary due to cipher requirement
- Payload's encrypted
- Two modes: tunnel, or transport





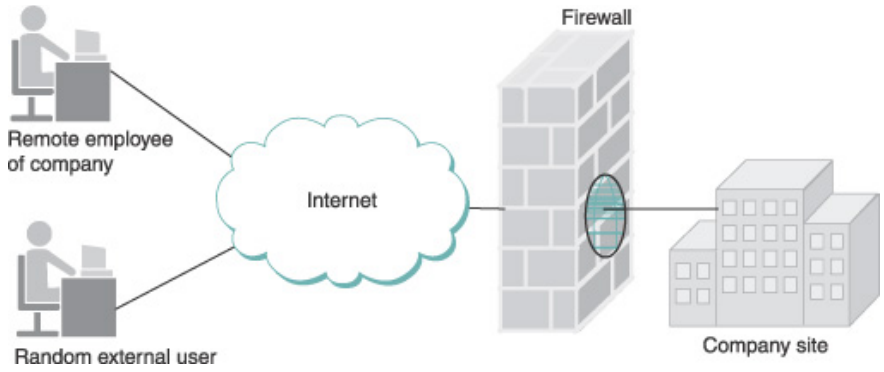
- Tunnel mode: bump-in-the-wire
  - Useful in creating VPNs
  - Payload is an IP packet

- Transport model
  - Upper layer payload is encrypted
  - UDP, TCP

Discussion

- Differences between IPsec and TLS
  - Pros and cons

Firewalls



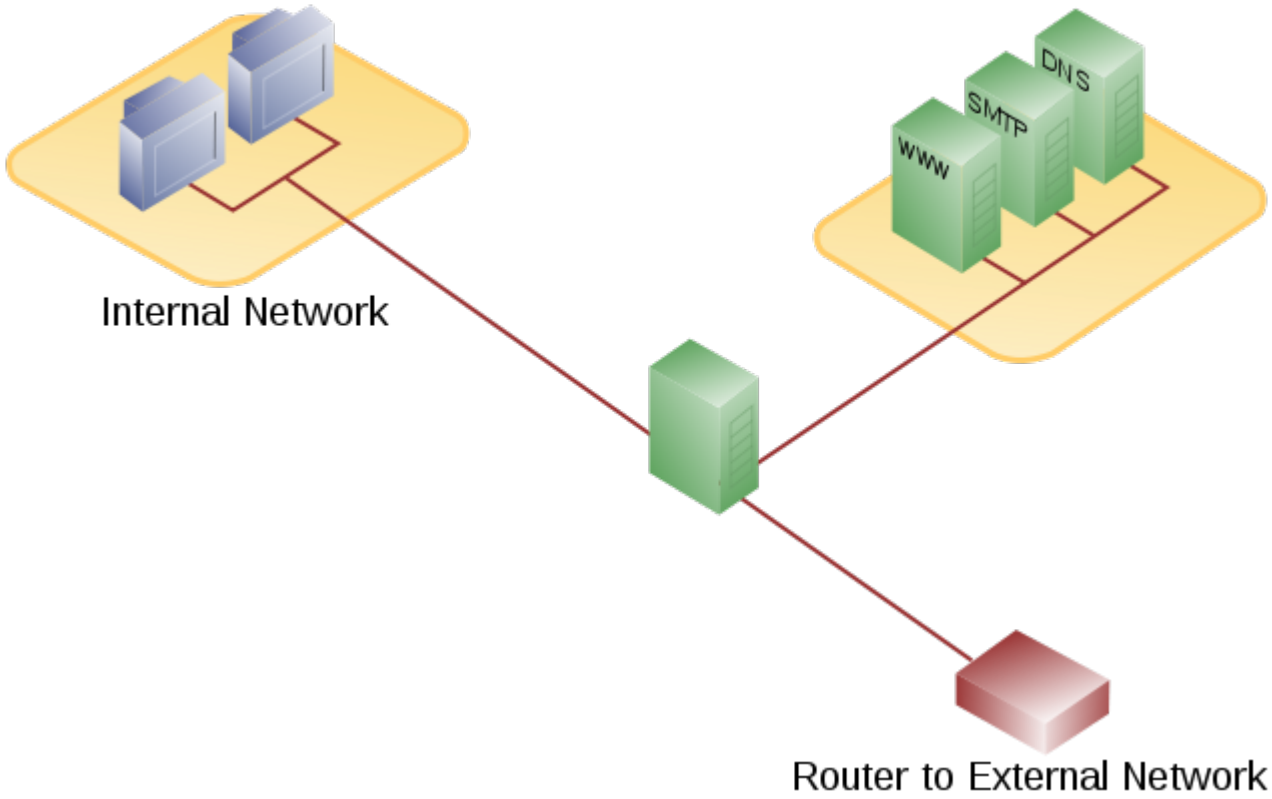
- Firewalls create zones of trust
  - The internal network
  - Demilitarized zone (DMZ)
    - DNS, email servers
    - Hosts in DMZ accessible by anyone
    - Cannot access internal hosts
    - DMZ can be periodically restored
  - The rest of the Internet
- Widely used in practice
  - Unilaterally deployed

Firewall configurations

- Access lists: similar to tcpdump's filter lists
- Allows outside connection to Duke CS's main mail server one.cs.duke.edu:
  - (*,*,152.3.140.161, 25, allow)

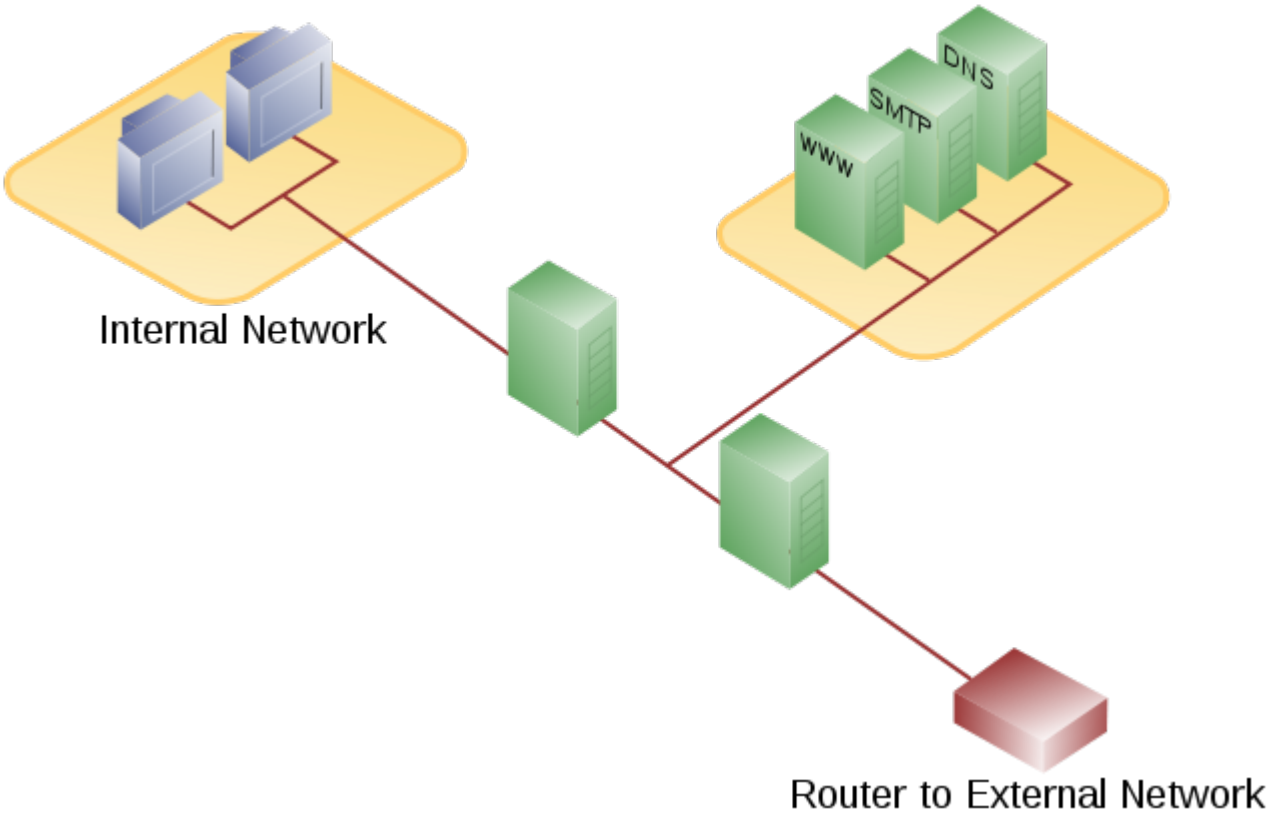
- Disallow to internal mail server
  - (152.3/16, *, 152.3.140.1, 25, allow)
  - (*,*,152.3.140.1, 25, deny)

Using Firewalls to Create DMZ



- A firewall with three interfaces

Using Firewalls to Create DMZ



- Dual firewalls

Conclusion

- Network security in practice
  - Building blocks
    - Symmetric key cryptography
    - Public key cryptography
  - Protocols and applications
    - SSH
    - TLS
    - IPSec
  - Firewalls