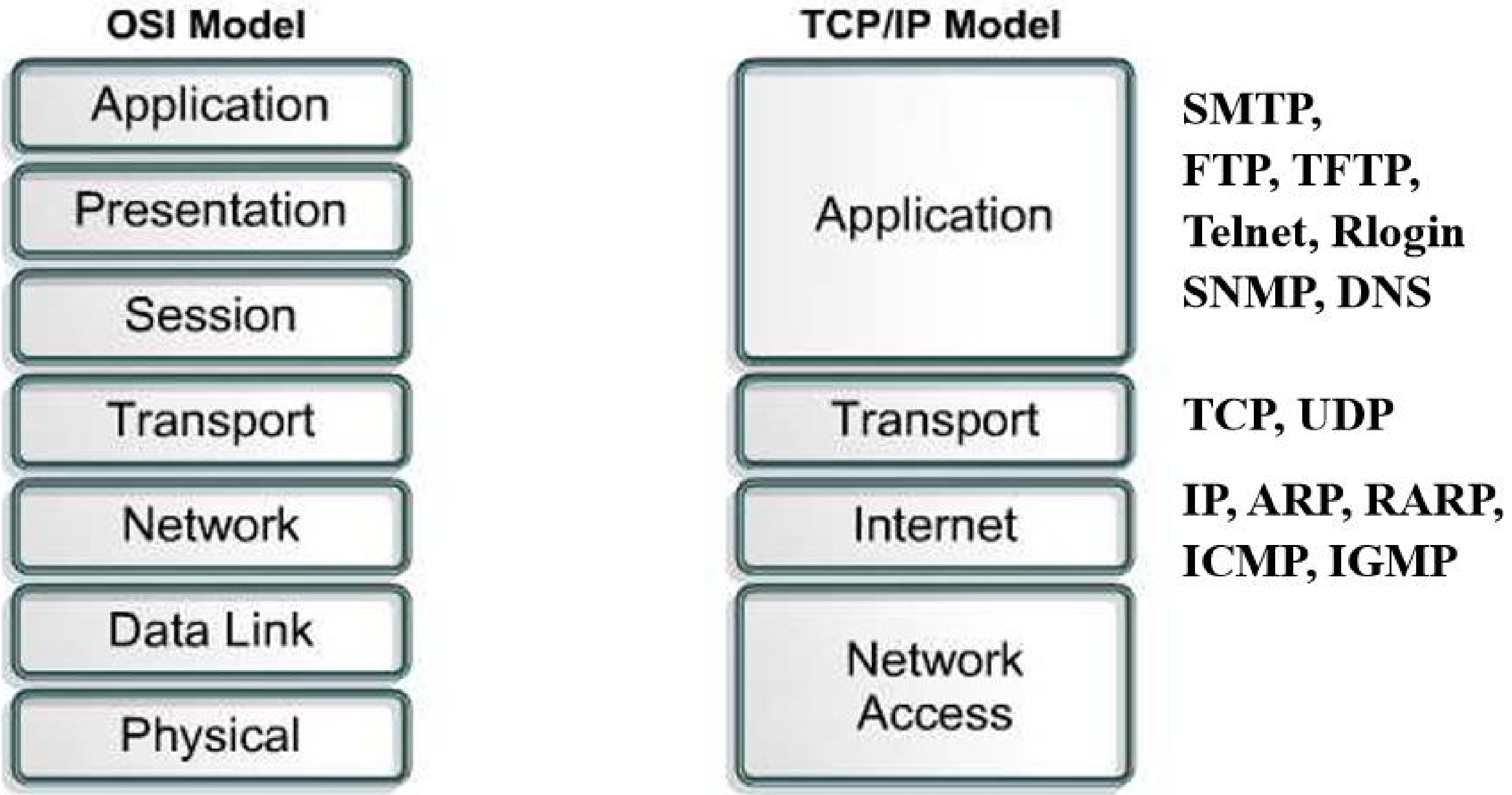
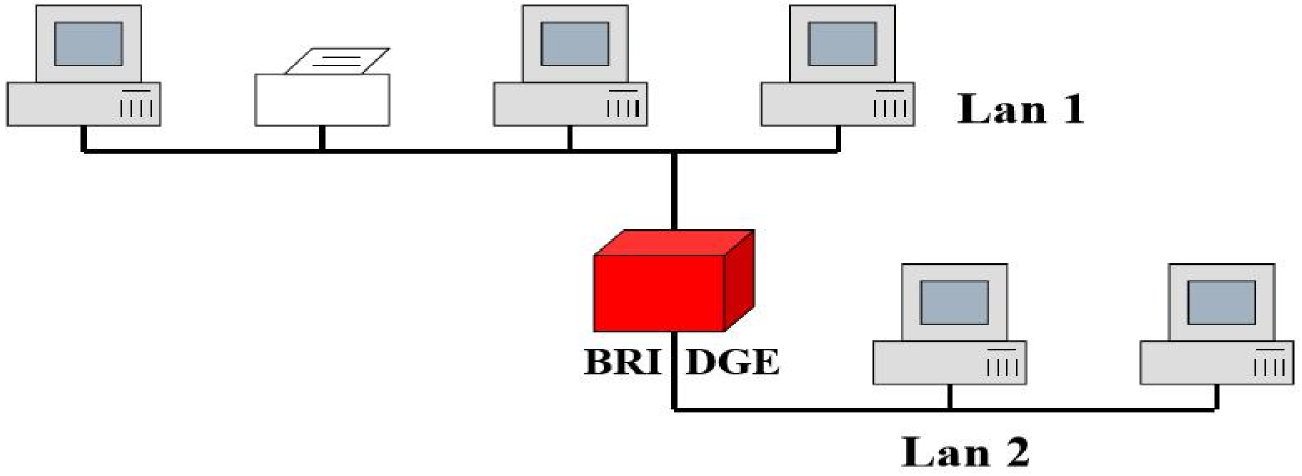
Network Security

Internet

- Internet = Inter-net
  - Interconnection of heterogeneous networks
  - Unreliable communication
  - Layered architecture

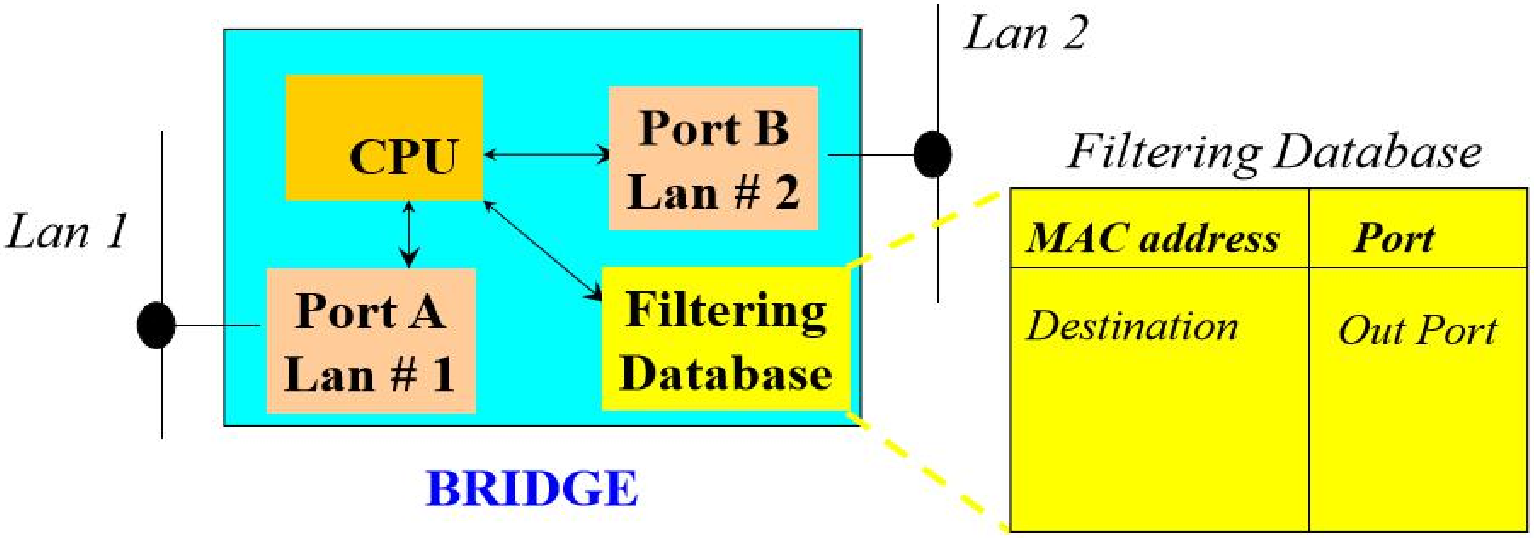


Network access (data link) layer



- Bridge, swtich
  - Filtering: if a frame generated within LAN 1 is destined to LAN 1, it remains confined within LAN 1
  - Relaying: if a frame originated within LAN 1 is destined to LAN 2, it is relayed by the bridge

Forwarding Data Base



- Filtering and Relaying are performed according to a local Forwarding Data Base
- A bridge is self-learning
  - Database: initially empty
  - Whenever a frame is received, associate the interface with the source MAC address in the frame
  - Delete switch table entries if they have not been used for some time

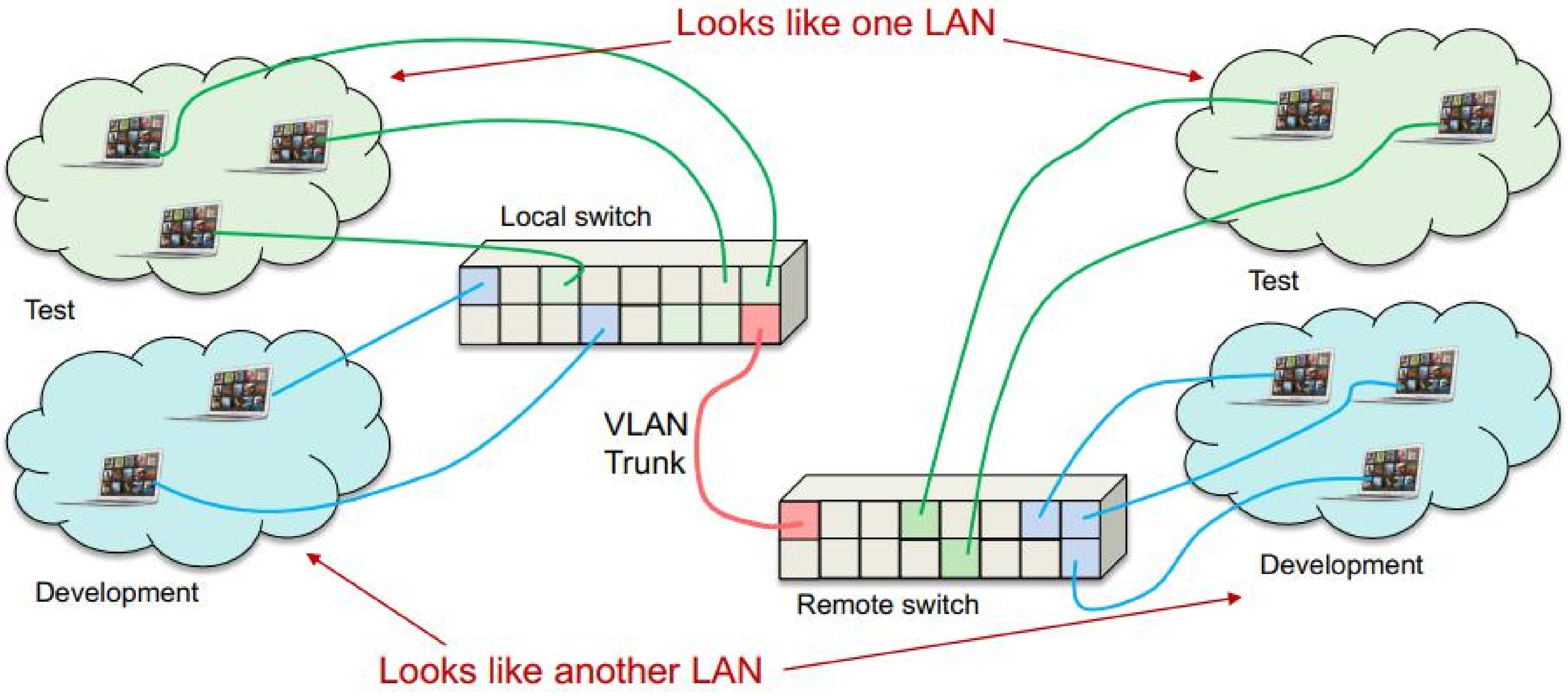
Forwarding Data Base attack

- Exploit size limit of Forwarding Data Base
  - Send bogus frames with random source MAC addresses
  - When table is full, legitimate traffic will be broadcast to all links
  - A host on any port can now see all traffic
  - Attack turns a bridge into a hub

- Countermeasure
  - Limit # addresses per port

Virtual Local Area Network (VLAN)

- Create multiple VLANs over single switch infrastructure
  - VLAN Trunking



VLAN Hopping Attack

- Switch spoofing
  - Attacker spoofs as a switch with a trunk connection and become a member of all VLANs
  - Can see all traffic of all VLANs

- Countermeasures
  - Disable unused ports
  - Disable auto-trunking
  - Explicitly configure trunking

ARP table Spoofing

- Address Resolution Protocol
  - Given IP address, find MAC address
- Based on ARP table
  - Filled by ARP queries
- ARP table poisoning
  - Attacker fakes responses to ARP queries by its own MAC address

- Countermeasures
  - Manually configure ARP entries
  - Ignore replies that are not associated with queries

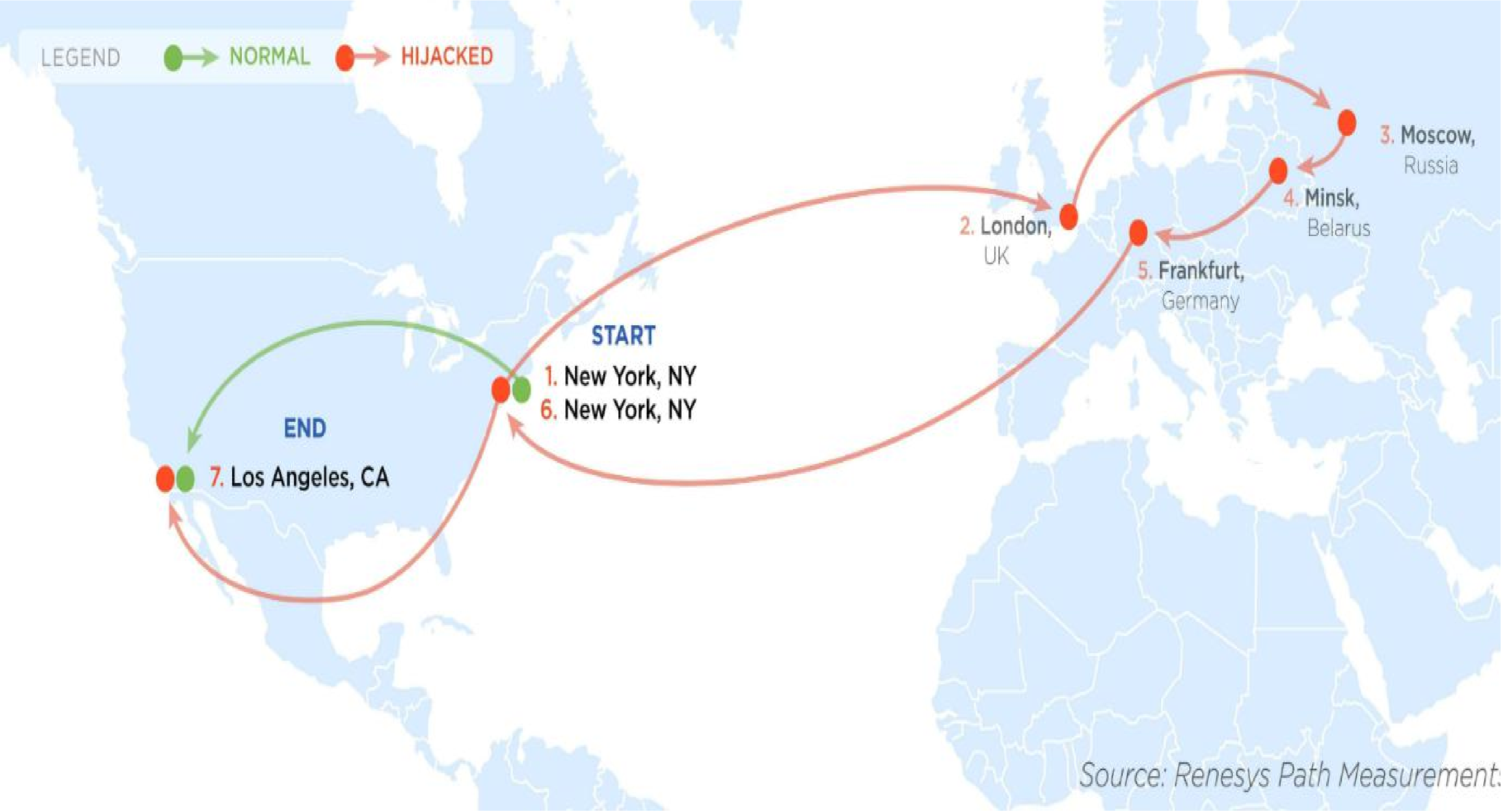
DHCP Server Spoofing

- When joining a network: needs to be configured
  - Dynamic Host Configuration Protocol
  - Broadcasts a DHCP Discover message
- DHCP server sends back a response
  - IP address, subnet mask, gateway, DNS servers, lease time
- Attack: spoof responses by a valid DHCP server
- Countermeasures
  - Switch ports can be configured as trusted or untrusted
  - Only specific machines are allowed to send DHCP responses
  - Use DHCP data to track client behavior

Network Layer

- Machine to machine delivery of packets
  - Routing
  - Fragmentation
- No address authentication
  - Attacker can spoof source addresses
- Attack on routers
  - Denial of Service (DOS): flood the router
  - Routing table poisoning: send false routing update to create fake routes

BGP hijacking



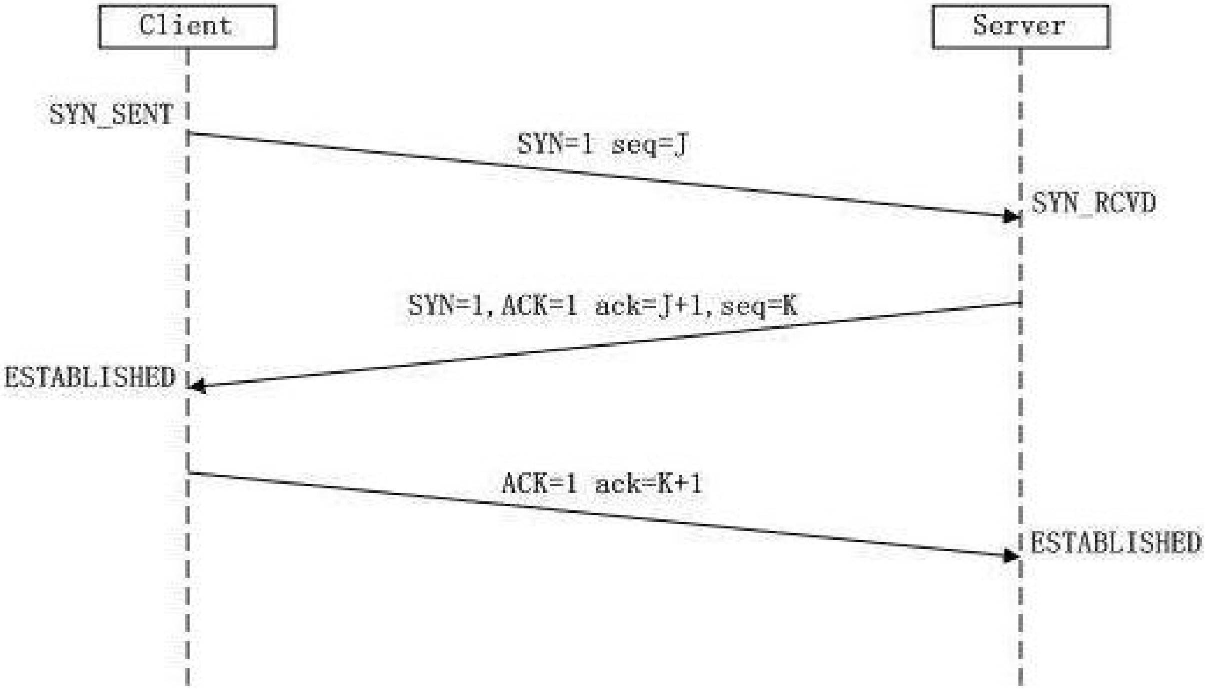
Source: Renesys Path Measurement

Transport layer

- TCP: Transmission Control Protocol
  - Stateful, connection-oriented & reliable
  - Every packet contains a sequence number (byte offset)
  - Receiver assembles packets into correct order
  - Sends acknowledgements
  - Missing packets are retransmitted
  - Congestion control

- UDP: User Datagram Protocol
  - Stateless, connectionless & unreliable
  - Anyone can send forged UDP messages

TCP

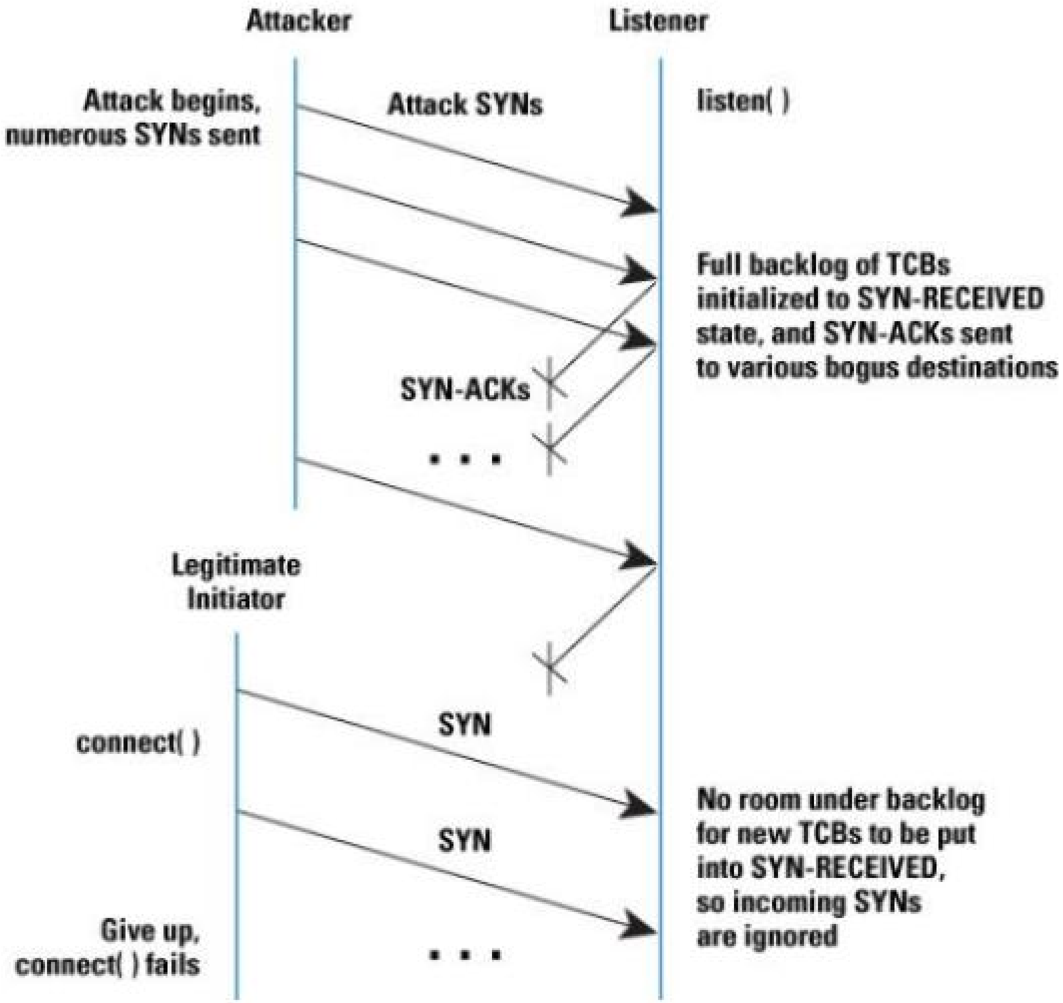


- Random sequence number

TCP SYN flooding

- Send lots of SYN
- Never complete handshake
- Cannot accept connections

- Do not allocate buffers and state when a SYN segment is received



Attacker
Listener
Attack begins, numerous SYNs sent
Attack SYNs
listen( )
Full backlog of TCBs initialized to SYN-RECEIVED state, and SYN-ACKs sent to various bogus destinations
SYN-ACKs
. . .
Legitimate Initiator
connect( )
SYN
SYN
No room under backlog for new TCBs to be put into SYN-RECEIVED, so incoming SYNs are ignored
Give up, connect( ) fails
. . .

Application layer: DNS vulnerabilities

- Domain Name System
  - Maps domain names to IP addresses
  - Via DNS servers
- How to find the right DNS server?
  - Start at the root
  - 13 root servers, 10 in US
- Attacks
  - DNS spoofing
  - DNS rebinding

SSL/TLS

What is SSL / TLS?

- Secure Sockets Layer and Transport Layer Security protocols
  - Same protocol design, different crypto algorithms
- End-to-end secure communications in presence of attacker
  - Attacker completely owns the network: controls Wi-Fi, DNS, routers,
  - Can listen to, modify, inject any packet
- De facto standard for Internet security
- Deployed in every Web browser; also VoIP, payment systems, distributed systems, etc.
- Scenario:
  - You are reading your email from an Internet café connected via a rooted Wi-Fi access point to a dodgy ISP in a hostile authoritarian country

What is SSL / TLS?

- Goal: provide a transport layer security protocol
- After setup, applications feel like they are using TCP sockets
  - SSL: Secure Socket Layer

- Created with HTTP in mind
  - Web sessions should be secure
    - Encrypted, tamper-proof, resilient to man-in-the-middle attacks
  - Mutual authentication is usually not needed
    - Client needs to identify server, but server not expected to know clients
- Rely on password authentication after secure channel is set up

SSL history

- SSL 1.0 – internal Netscape design, early 1994?
  - Lost in the mists of time
- SSL 2.0 – Netscape, Nov 1994
  - Several weaknesses
- SSL 3.0 – Netscape and Paul Kocher, Nov 1996

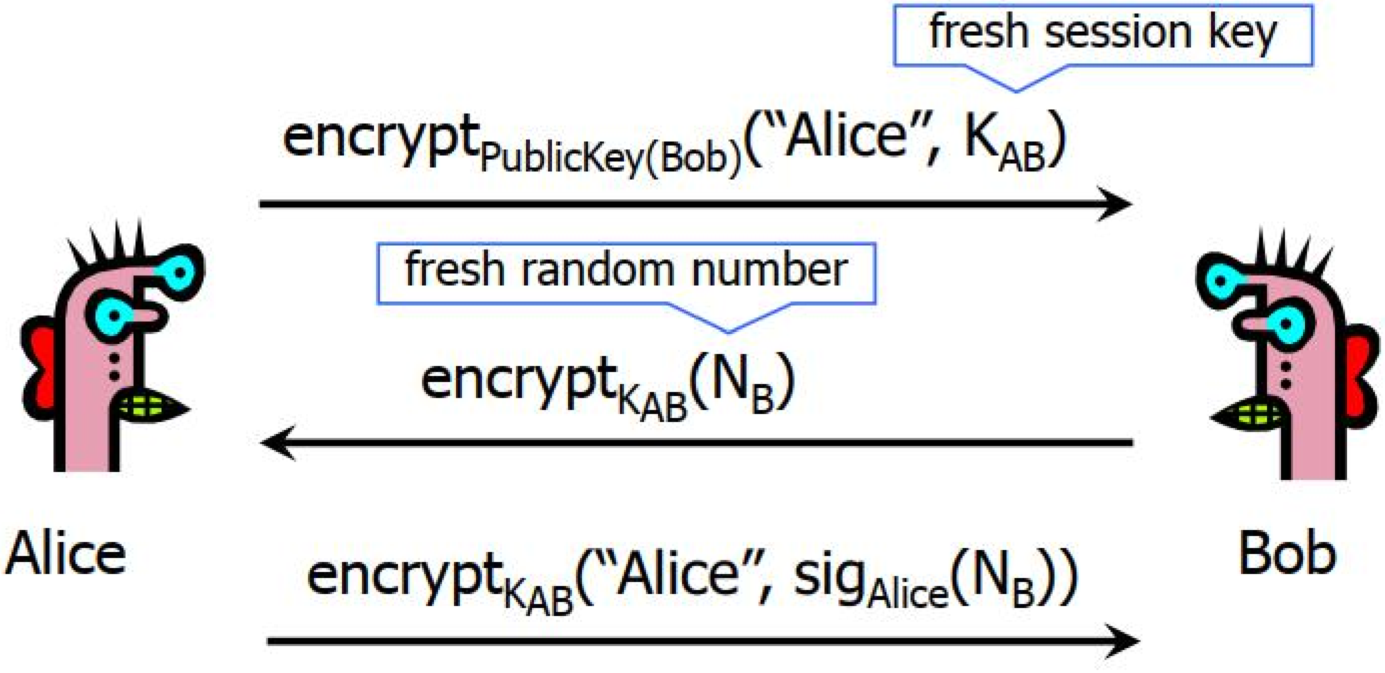
- SSL evolved to TLS

- TLS 1.0 – Internet standard, Jan 1999
  - Based on SSL 3.1, but not interoperable (different cryptographic algorithms)
- TLS 1.1 – 2006
- TLS 1.2 – 2008
- TLS 1.3 – 2018

TLS Protocol

- Goal
  - Provide authentication (usually one-way), privacy, & data integrity between two applications
- Data encryption
  - Use symmetric cryptography to encrypt data
  - Key exchange: keys generated at the start of each session
- Data integrity
  - Include a MAC with transmitted data to ensure message integrity
- Authentication
  - Use public key cryptography & X.509 certificates for authentication
  - Optional: can authenticate 0, 1, or both parties
- Interoperability & evolution
  - Support different key exchange, encryption, authentication protocols
    - negotiate what to use at the start of a session

SSL early version



fresh session key

$\text{encrypt}_{\text{PublicKey(Bob)}}(\text{“Alice”}, K_{AB})$

fresh random number

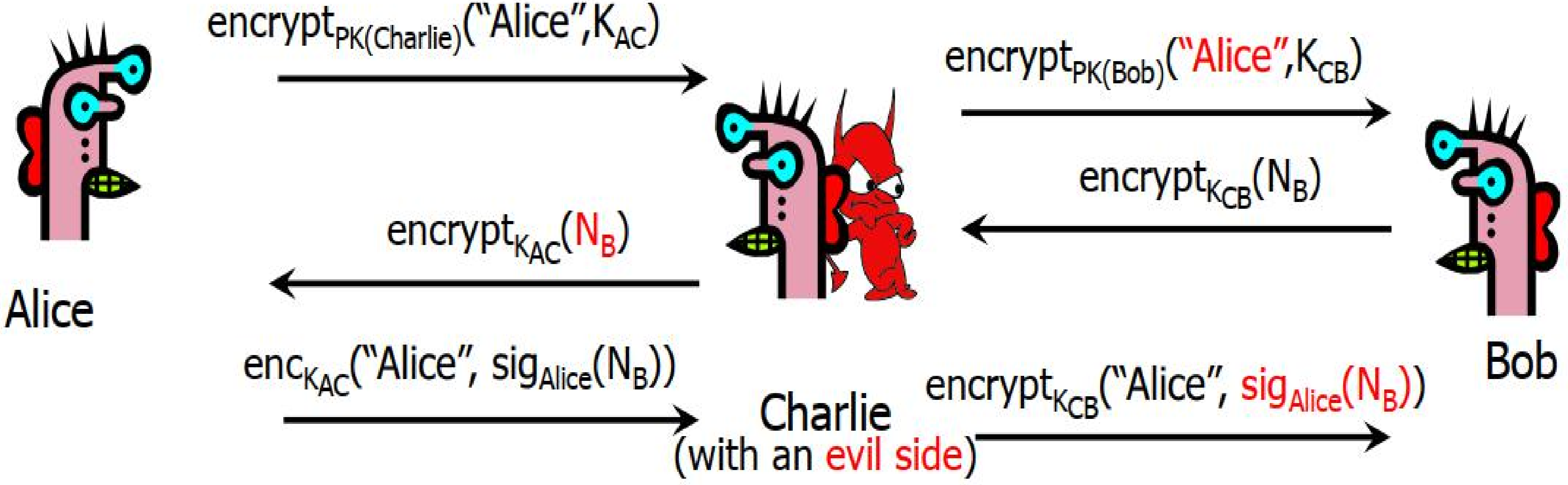
$\text{encrypt}_{K_{AB}}(N_B)$

Alice

Bob

$\text{encrypt}_{K_{AB}}(\text{“Alice”}, \text{sig}_{\text{Alice}}(N_B))$

Breaking SSL early version



encrypt$_{PK(Charlie)}$(“Alice”,K_{AC})

encrypt$_{PK(Bob)}$(“Alice”,K_{CB})

encrypt$_{K_{CB}}$(N_B)

encrypt$_{K_{AC}}$(N_B)

Alice

enc$_{K_{AC}}$(“Alice”, sig$_{Alice}$(N_B))

Charlie
(with an evil side)

encrypt$_{K_{CB}}$(“Alice”, sig$_{Alice}$(N_B))

Bob

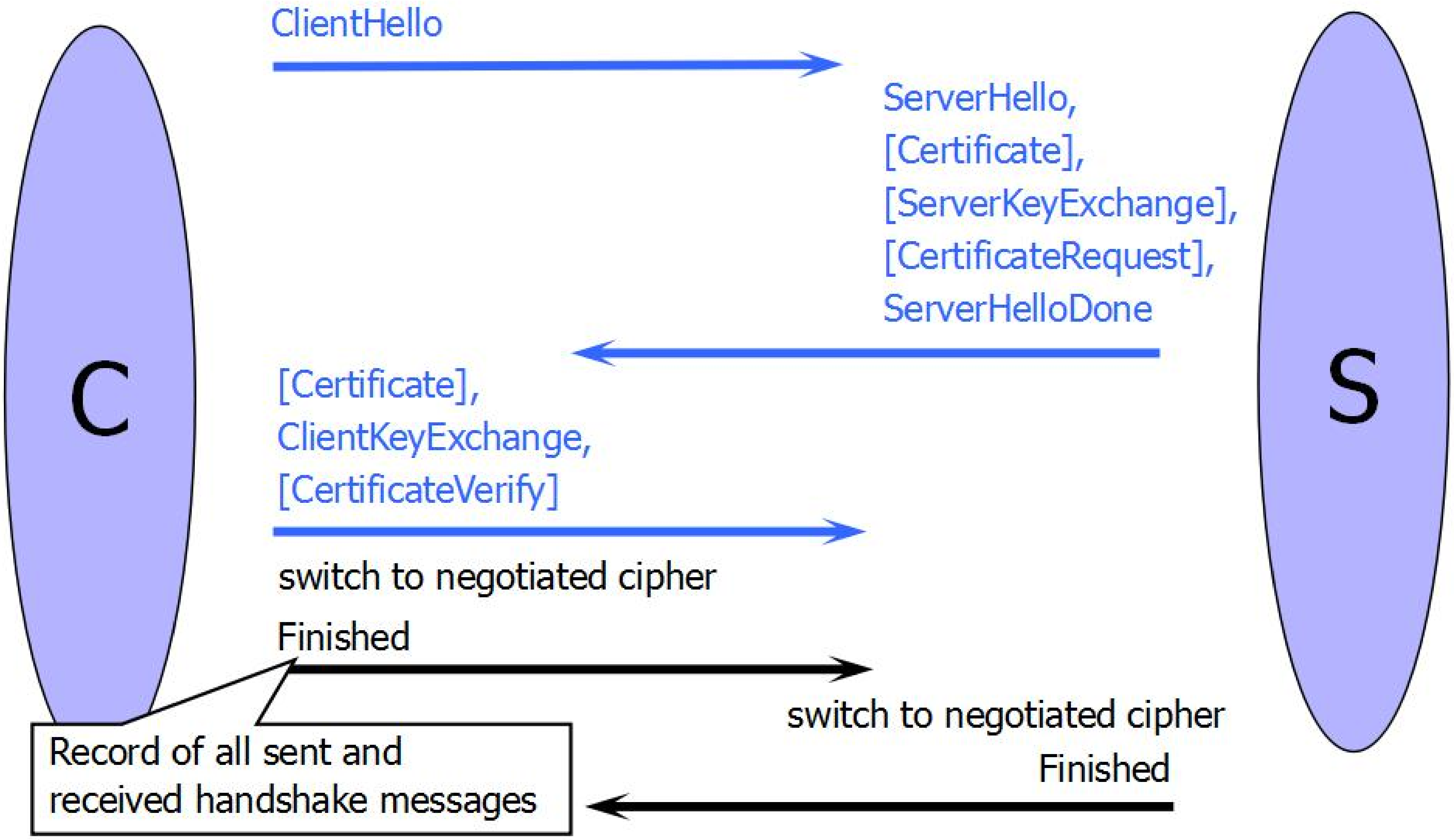
SSL basics

- SSL consists of two protocols
- Handshake protocol
  - Uses public-key cryptography to establish secret keys between client and server
- Record protocol
  - Uses the secret keys for data exchange

Handshake Protocol

- Runs between client and server
    - For example, client = Web browser, server = website
- Negotiate version of the protocol and the set of cryptographic algorithms to be used
    - Interoperability between different implementations
- Authenticate server and client (optional)
    - Use digital certificates to learn public keys and verify identity
    - Often only the server is authenticated
- Use public keys to establish a shared secret

Handshake Protocol



ClientHello

ServerHello,
[Certificate],
[ServerKeyExchange],
[CertificateRequest],
ServerHelloDone

[Certificate],
ClientKeyExchange,
[CertificateVerify]

switch to negotiated cipher

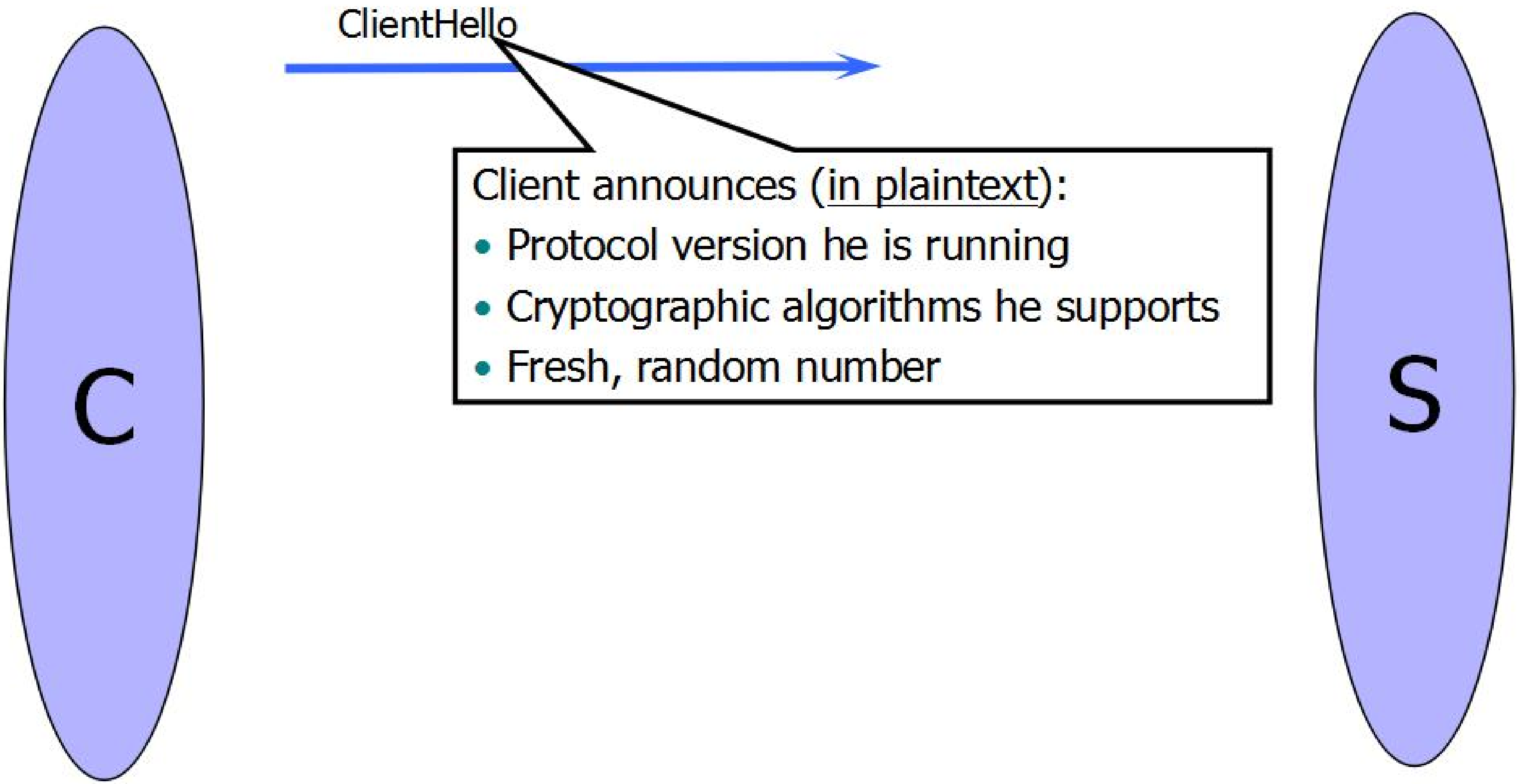
Finished

switch to negotiated cipher

Finished

Record of all sent and received handshake messages

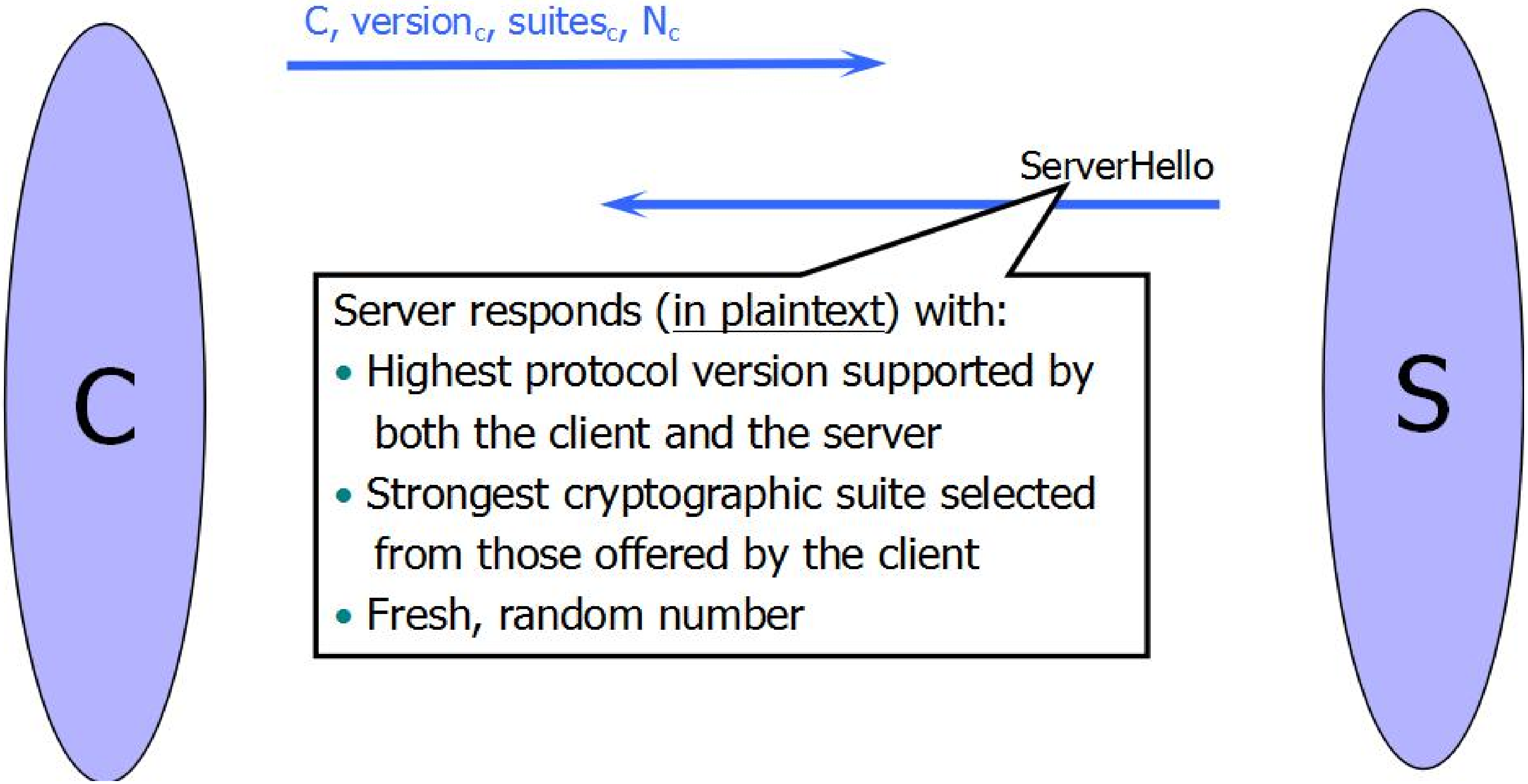
ClientHello



Client announces (in plaintext):

- Protocol version he is running
- Cryptographic algorithms he supports
- Fresh, random number

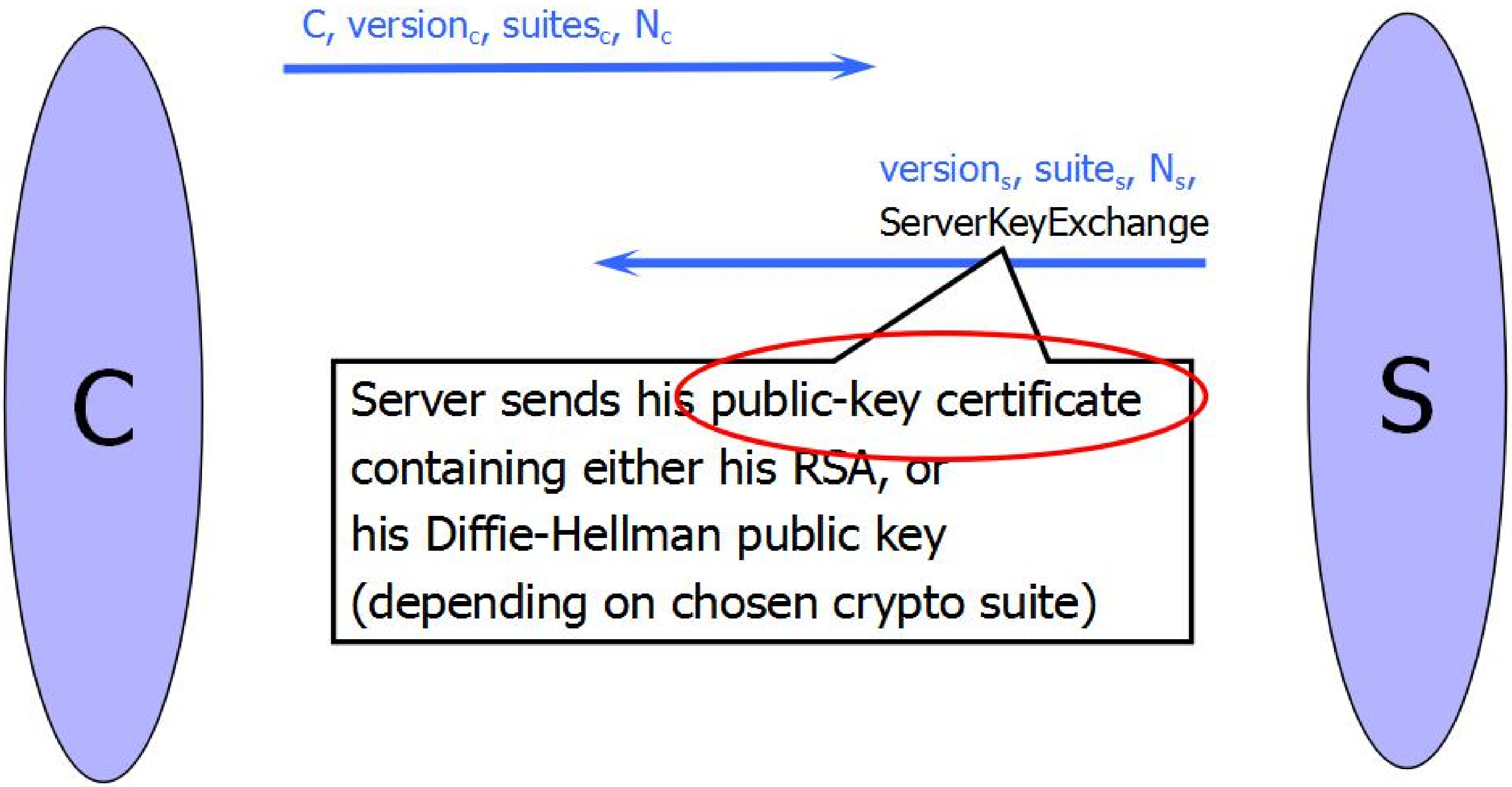
ServerHello



Server responds (in plaintext) with:

- Highest protocol version supported by both the client and the server
- Strongest cryptographic suite selected from those offered by the client
- Fresh, random number

ServerKeyExchange



C

S

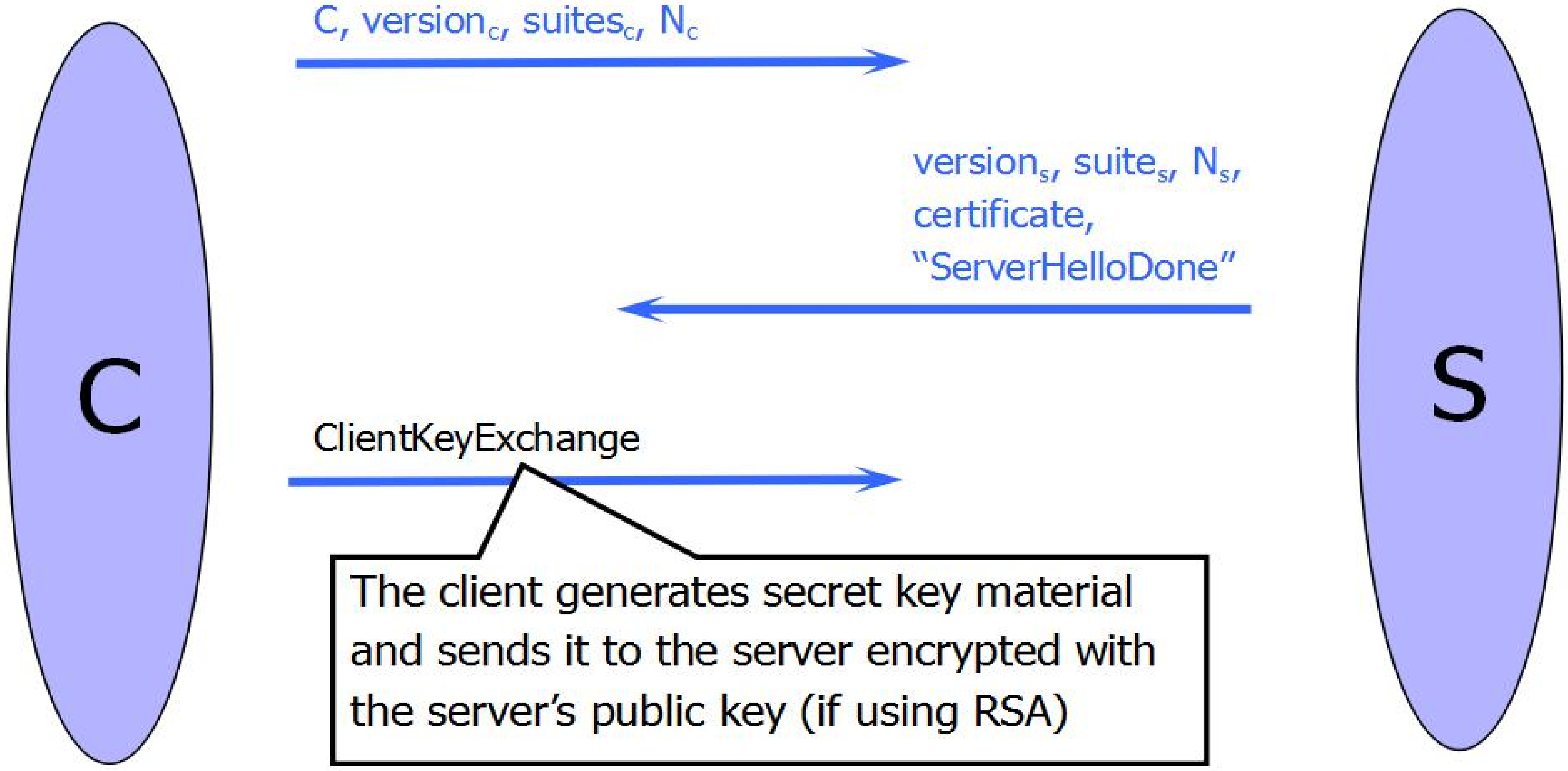
C, version_c, suites_c, N_c

version_s, suite_s, N_s,

ServerKeyExchange

Server sends his public-key certificate containing either his RSA, or his Diffie-Hellman public key (depending on chosen crypto suite)

ClientKeyExchange



C

S

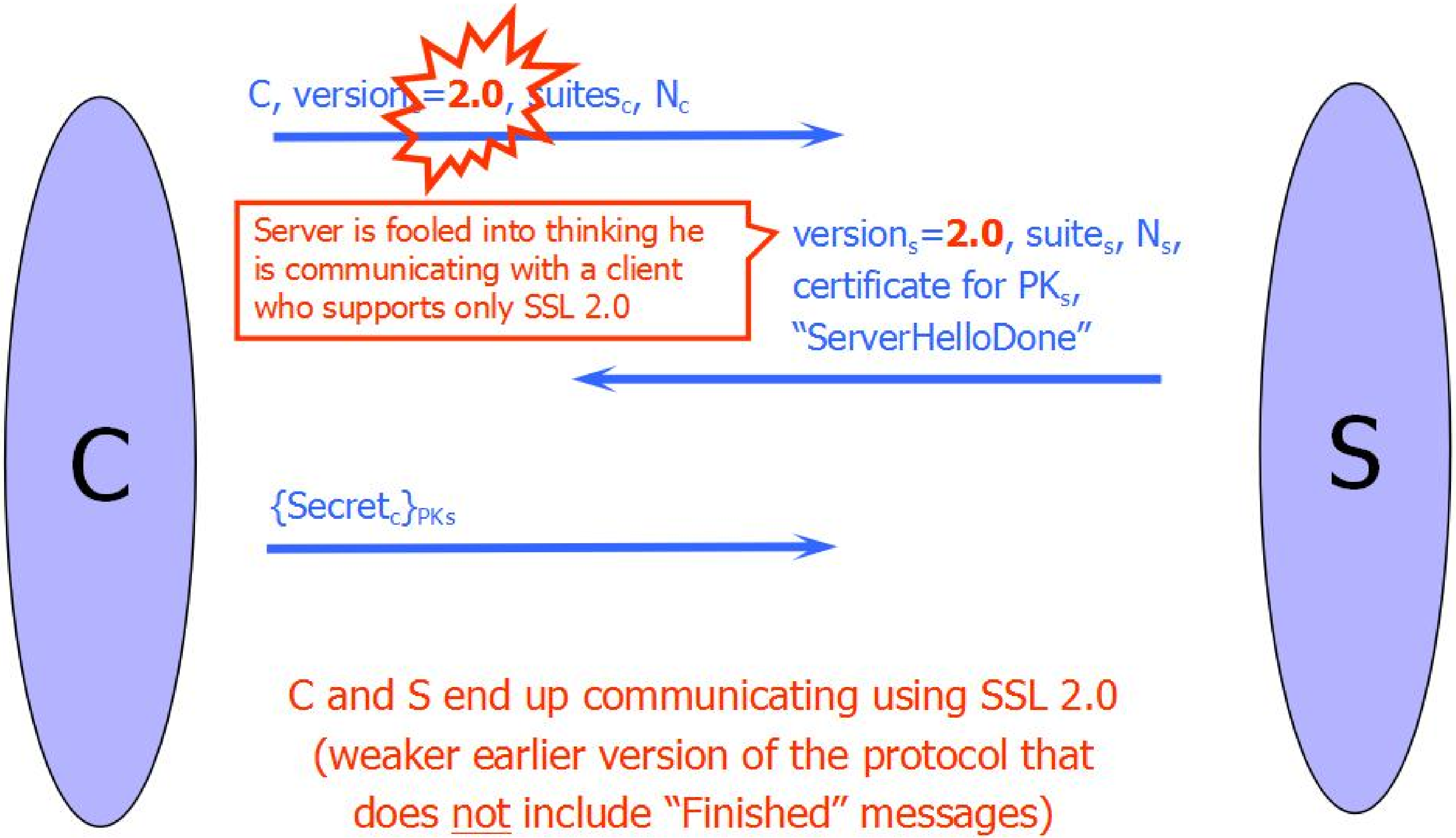
C, version$_c$, suites$_c$, N$_c$

version$_s$, suite$_s$, N$_s$,
certificate,
“ServerHelloDone”

ClientKeyExchange

The client generates secret key material and sends it to the server encrypted with the server’s public key (if using RSA)

Version Rollback Attack



C, version$_c$=**2.0**, suites$_c$, N$_c$

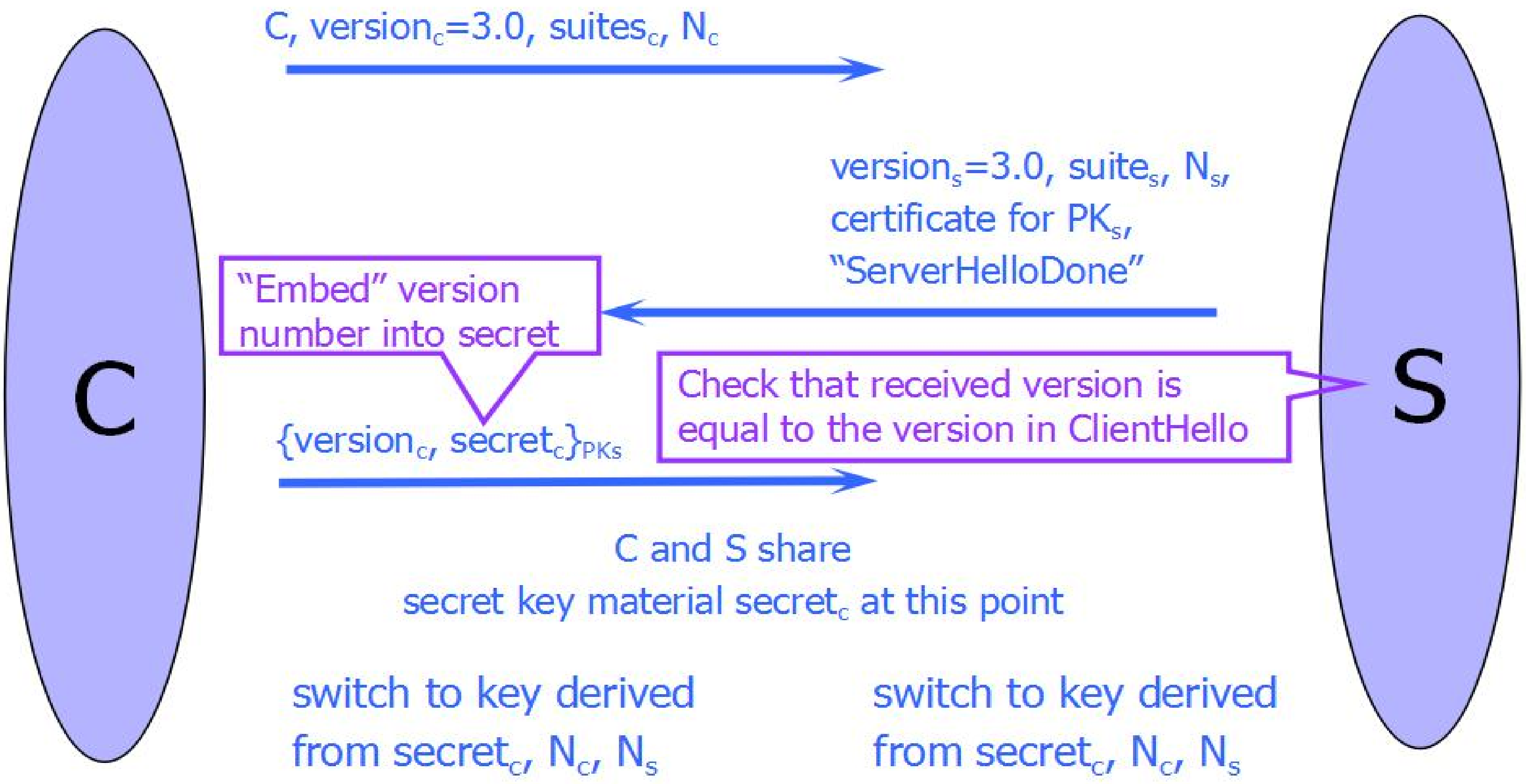
Server is fooled into thinking he is communicating with a client who supports only SSL 2.0

version$_s$=**2.0**, suite$_s$, N$_s$,
certificate for PK$_s$,
"ServerHelloDone"

{Secret$_c$}$_{PKs}$

C and S end up communicating using SSL 2.0
(weaker earlier version of the protocol that
does not include "Finished" messages)

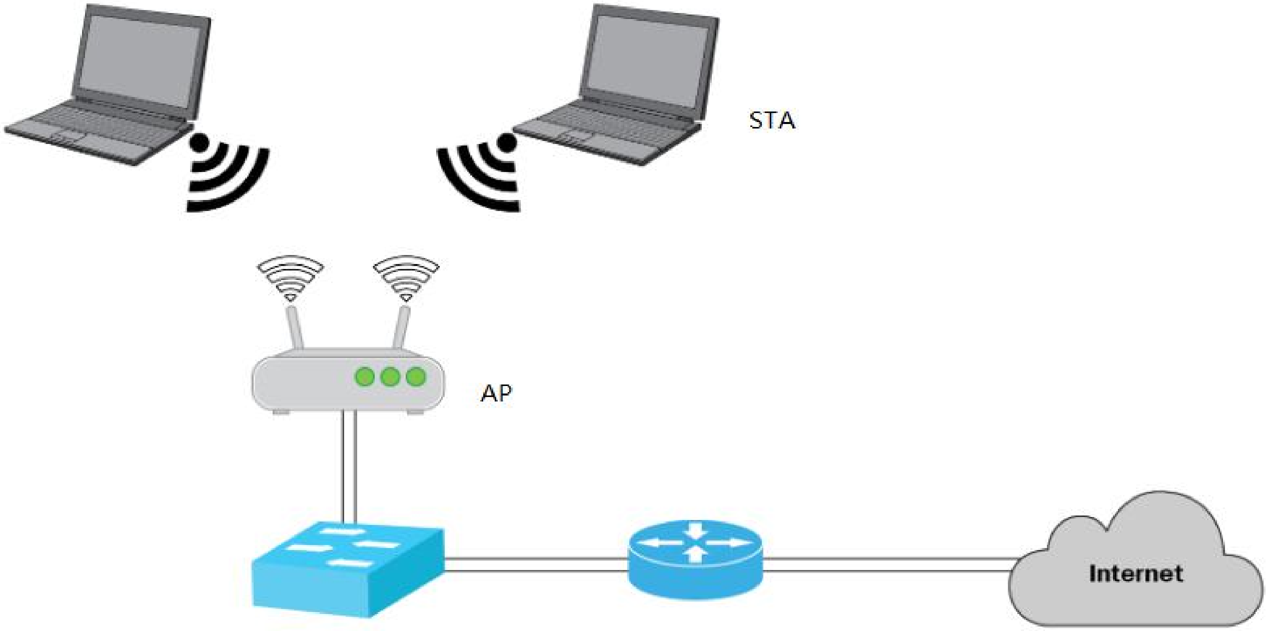
Version Check in SSL 3.0



C
S
C, version_c=3.0, suites_c, N_c
version_s=3.0, suite_s, N_s,
certificate for PK_s,
"ServerHelloDone"
"Embed" version number into secret
Check that received version is equal to the version in ClientHello
$\{\text{version}_c, \text{secret}_c\}_{PKs}$
C and S share
secret key material secret_c at this point
switch to key derived from secret_c, N_c, N_s
switch to key derived from secret_c, N_c, N_s

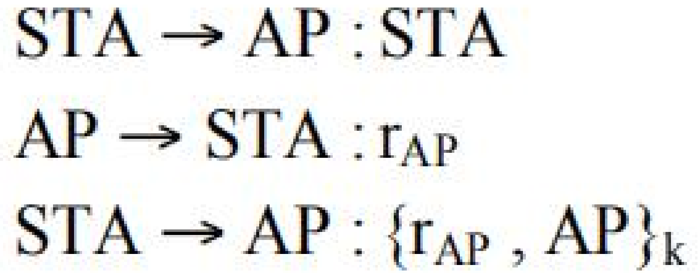
WEP security

WiFi



- Wireless medium
  - Broadcast

WiFi authentication



$$\mathrm{STA} \rightarrow \mathrm{AP} : \mathrm{STA}$$

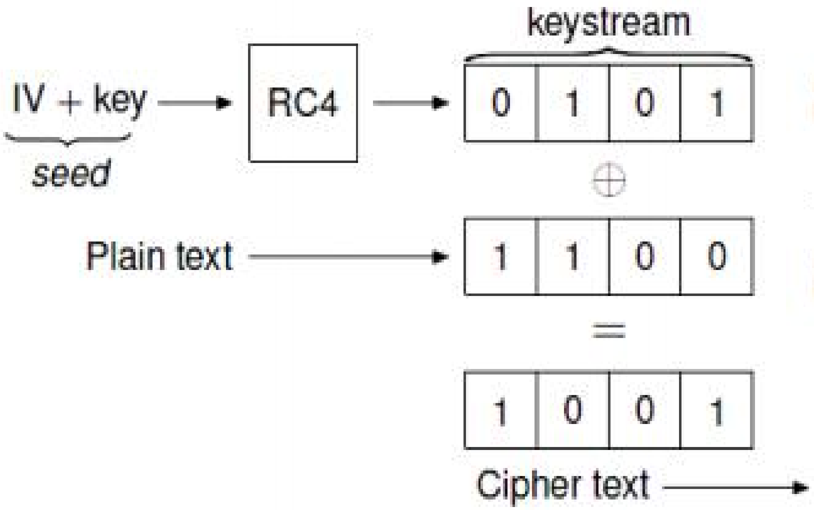
$$\mathrm{AP} \rightarrow \mathrm{STA} : r_{\mathrm{AP}}$$

$$\mathrm{STA} \rightarrow \mathrm{AP} : \{r_{\mathrm{AP}}\ ,\ \mathrm{AP}\}_k$$

- k: secret key shared between all STAs and AP

- Problems?

WiFi encryption: confidentiality



$$STA \rightarrow AP: \quad STA$$

$$AP \rightarrow STA: \quad r_{AP}$$

$$STA \rightarrow AP: \quad IV,\ (r_{AP}, AP) \oplus RC4(k \| IV)$$

- Stream cipher RC4
  - IV: 24 bits, increments by 1 per pkt
  - Key: 40 bits

- Problems?

WiFi checksum: integrity

- Add Cyclic Redundancy Check (CRC) to pkt
  - M||CRC(M)
  - Ciphertext M||CRC(M) XOR S

$$CRC(M \oplus M') = CRC(M) \oplus CRC(M')$$

- Problems?