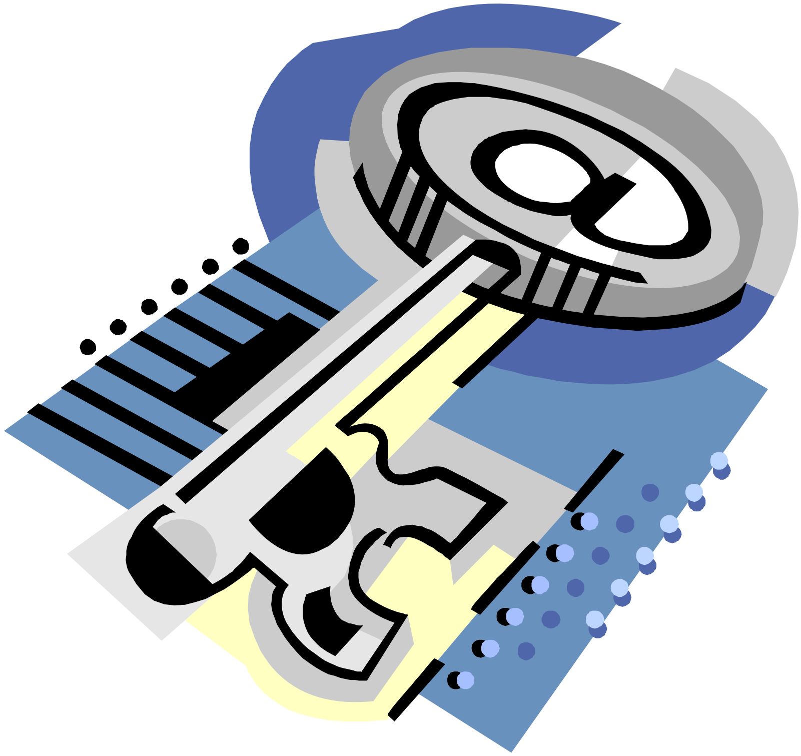
Paper signatures v/s Digital Signatures

V/s

Parameter	Paper	Electronic
Authenticity	May be forged	Hard to be copied
Integrity	Signature independent of the document	Signature depends on the contents of the document
Non-repudiation	a. Handwriting expert needed b. Error prone	a. Any computer user b. Error free



- **Key Generation**
  - Random Numbers
  - RSA Key Pair [Private/Public Key]
- **Digital Signature**
  - Generate Message Digest [SHA2]
  - Encrypting Digest using Private Key [Signatures]
  - Attaching the Signatures to the message.
- **Verification of Signatures**
  - Run the test for Authentication, Integrity and Non repudiation.
- **Digital Signature Certificate**
  - ITU X.509 v3

Private key protection

- The Private key generated is to be protected and kept secret. The responsibility of the secrecy of the key lies with the owner.
- The key is secured using
  - PIN Protected soft token
  - Smart Cards
  - Hardware Tokens

PIN protected soft tokens

- The Private key is encrypted and kept on the Hard Disk in a file, this file is password protected.
- This forms the lowest level of security in protecting the key, as
  - The key is highly reachable.
  - PIN can be easily known or cracked.
- Soft tokens are also not preferred because
  - The key becomes static and machine dependent.
  - The key is in known file format.

Smart Cards

- The Private key is generated in the crypto module residing in the smart card.
- The key is kept in the memory of the smart card.
- The key is highly secured as it doesn't leave the card, the message digest is sent inside the card for signing, and the signatures leave the card.
- The card gives mobility to the key and signing can be done on any system. (Having smart card reader)

Hardware Tokens

- They are similar to smart cards in functionality as
  - Key is generated inside the token.
  - Key is highly secured as it doesn’t leave the token.
  - Highly portable.
  - Machine Independent.
- iKEY is one of the most commonly used token as it doesn’t need a special reader and can be connected to the system using USB port.

Hardware Tokens

iKey

Smart Card

Biometrics – adds another level of security to these tokens

Public Key Infrastructure (PKI)

- Some Trusted Agency is required which certifies the association of an individual with the key pair.

  Certifying Authority (CA)

- This association is done by issuing a certificate to the user by the CA

  Public key certificate (PKC)

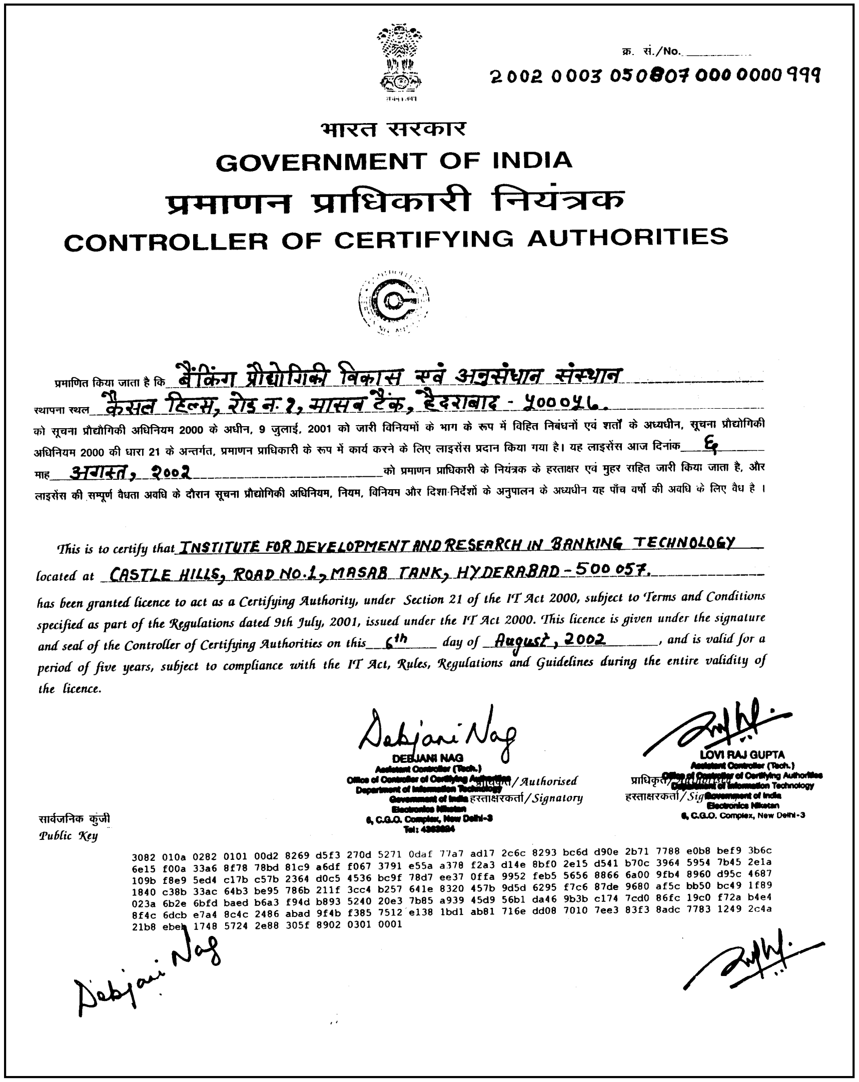
- All public key certificates are digitally signed by the CA

Certifying Authority

- Must be widely known and trusted
- Must have well defined Identification process before issuing the certificate
- Provides online access to all the certificates issued
- Provides online access to the list of certificates revoked
- Displays online the license issued by the Controller
- Displays online approved Certification Practice Statement (CPS)
- Must adhere to IT Act/Rules/Regulations and Guidelines

IDRBT Certificate

Paper



क्र. सं./No. 2002 0003 050807 000 0000 111

भारत सरकार

GOVERNMENT OF INDIA

प्रमाणन प्राधिकारी नियंत्रक

CONTROLLER OF CERTIFYING AUTHORITIES

प्रमाणित किया जाता है कि बैंकिंग प्रौद्योगिकी विकास एवं अनुसंधान संस्थान
स्थापना स्थल कैसल हिल्स, रोड नं. १, मासब टैंक, हैदराबाद - ५०००५७.
को सूचना प्रौद्योगिकी अधिनियम 2000 के अधीन, 9 जुलाई, 2001 को जारी विनियमों के भाग के रूप में विहित निबंधनों एवं शर्तों के अध्यधीन, सूचना प्रौद्योगिकी अधिनियम 2000 की धारा 21 के अन्तर्गत, प्रमाणन प्राधिकारी के रूप में कार्य करने के लिए लाइसेंस प्रदान किया गया है। यह लाइसेंस आज दिनांक ६ माह अगस्त, २००२ को प्रमाणन प्राधिकारी के नियंत्रक के हस्ताक्षर एवं मुहर सहित जारी किया जाता है, और लाइसेंस की सम्पूर्ण वैधता अवधि के दौरान सूचना प्रौद्योगिकी अधिनियम, नियम, विनियम और दिशा-निर्देशों के अनुपालन के अध्यधीन यह पाँच वर्षों की अवधि के लिए वैध है।

This is to certify that INSTITUTE FOR DEVELOPMENT AND RESEARCH IN BANKING TECHNOLOGY *located at* CASTLE HILLS, ROAD NO.1, MASAB TANK, HYDERABAD - 500 057. *has been granted licence to act as a Certifying Authority, under Section 21 of the IT Act 2000, subject to Terms and Conditions specified as part of the Regulations dated 9th July, 2001, issued under the IT Act 2000. This licence is given under the signature and seal of the Controller of Certifying Authorities on this* 6th *day of* August, 2002, *and is valid for a period of five years, subject to compliance with the IT Act, Rules, Regulations and Guidelines during the entire validity of the licence.*

Debjani Nag
DEBJANI NAG
Assistant Controller (Tech.)
Office of Controller of Certifying Authorities
Department of Information Technology
Government of India
Electronics Niketan
6, C.G.O. Complex, New Delhi-3
Tel: 4363694
प्राधिकृत/*Authorised*
हस्ताक्षरकर्ता/*Signatory*

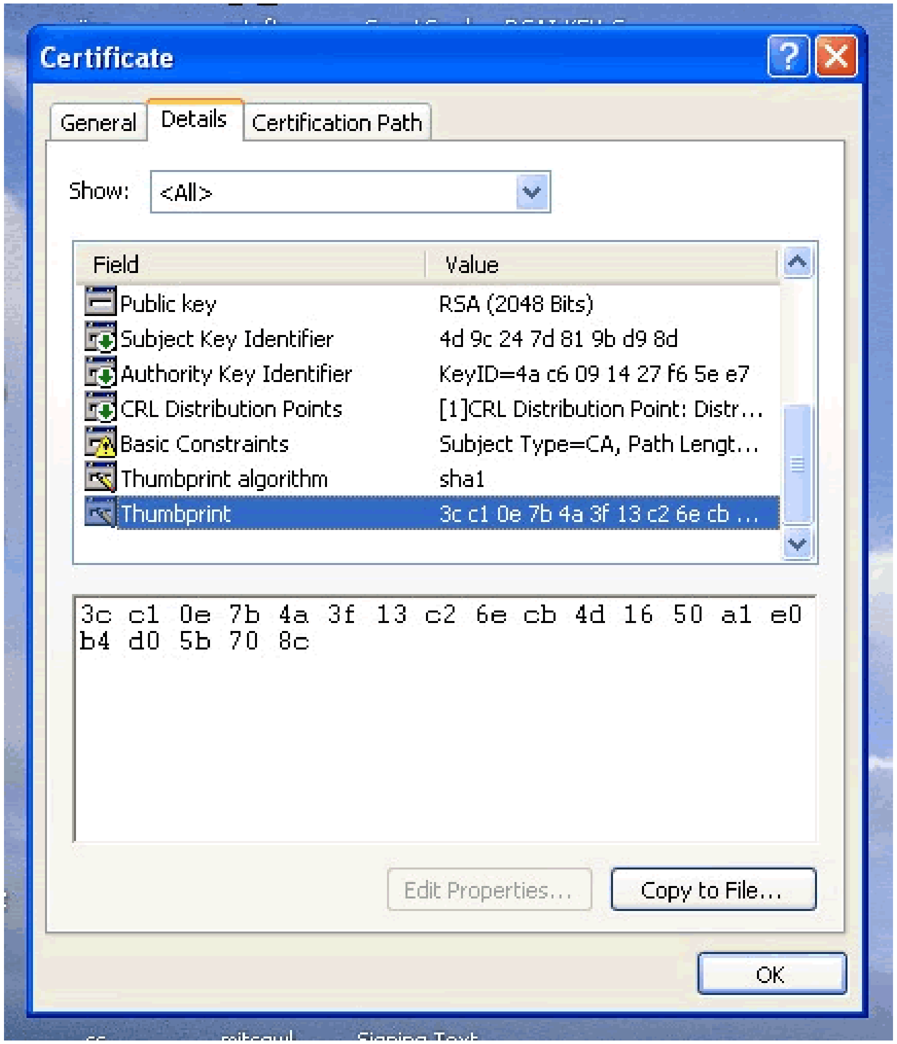
LOVI RAJ GUPTA
Assistant Controller (Tech.)
Office of Controller of Certifying Authorities
Department of Information Technology
Government of India
Electronics Niketan
6, C.G.O. Complex, New Delhi-3
प्राधिकृत/*Authorised*
हस्ताक्षरकर्ता/*Signatory*

सार्वजनिक कुंजी
Public Key

```
3082 010a 0282 0101 00d2 8269 d5f3 270d 5271 0daf 77a7 ad17 2c6c 8293 bc6d d90e 2b71 7788 e0b8 bef9 3b6c
6e15 f00a 33a6 8f78 78bd 81c9 a6df f067 3791 e55a a378 f2a3 d14e 8bf0 2e15 d541 b70c 3964 5954 7b45 2e1a
109b f8e9 5ed4 c17b c57b 2364 d0c5 4536 bc9f 78d7 ee37 0ffa 9952 feb5 5656 8866 6a00 9fb4 8960 d95c 4687
1840 c38b 33ac 64b3 be95 786b 211f 3cc4 b257 641e 8320 457b 9d5d 6295 f7c6 87de 9680 af5c bb50 bc49 1f89
023a 6b2e 6bfd baed b6a3 f94d b893 5240 20e3 7b85 a939 45d9 56b1 da46 9b3b c174 7cd0 86fc 19c0 f72a b4e4
8f4c 6dcb e7a4 8c4c 2486 abad 9f4b f385 7512 e138 1bd1 ab81 716e dd08 7010 7ee3 83f3 8adc 7783 1249 2c4a
21b8 ebeb 1748 5724 2e88 305f 8902 0301 0001
```

Debjani Nag

Electronic



Show: <All>

Field	Value
Public key	RSA (2048 Bits)
Subject Key Identifier	4d 9c 24 7d 81 9b d9 8d
Authority Key Identifier	KeyID=4a c6 09 14 27 f6 5e e7
CRL Distribution Points	[1]CRL Distribution Point: Distr...
Basic Constraints	Subject Type=CA, Path Lengt...
Thumbprint algorithm	sha1
Thumbprint	3c c1 0e 7b 4a 3f 13 c2 6e cb ...

```
3c c1 0e 7b 4a 3f 13 c2 6e cb 4d 16 50 a1 e0
b4 d0 5b 70 8c
```

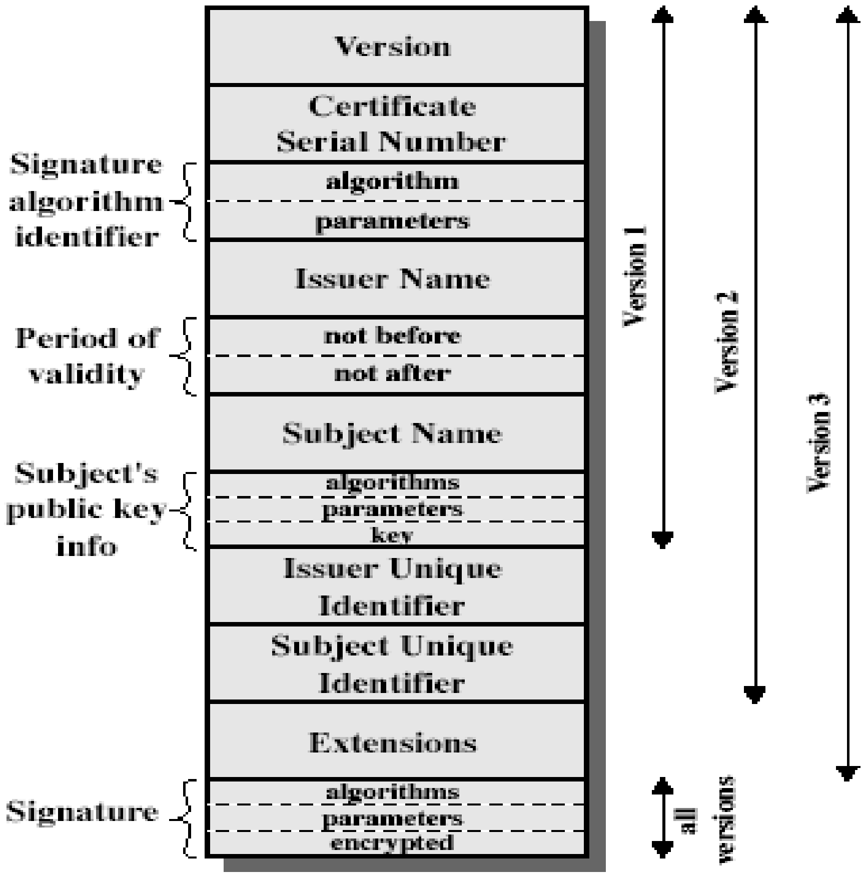
X.509 Authentication Service

- part of CCITT X.500 directory service standards
  - distributed servers maintaining some info database
- defines framework for authentication services
  - directory may store public-key certificates
  - with public key of user
  - signed by certification authority
- also defines authentication protocols
- uses public-key crypto & digital signatures
  - algorithms not standardised, but RSA recommended

X.509 Certificates

- issued by a Certification Authority (CA), containing:
  - version (1, 2, or 3)
  - serial number (unique within CA) identifying certificate
  - signature algorithm identifier
  - issuer X.500 name (CA)
  - period of validity (from - to dates)
  - subject X.500 name (name of owner)
  - subject public-key info (algorithm, parameters, key)
  - issuer unique identifier (v2+)
  - subject unique identifier (v2+)
  - extension fields (v3)
  - signature (of hash of all fields in certificate)
- notation CA<<A>> denotes certificate for A signed by CA

X.509 Certificates



Version

Certificate Serial Number

Signature algorithm identifier: algorithm, parameters

Issuer Name

Period of validity: not before, not after

Subject Name

Subject's public key info: algorithms, parameters, key

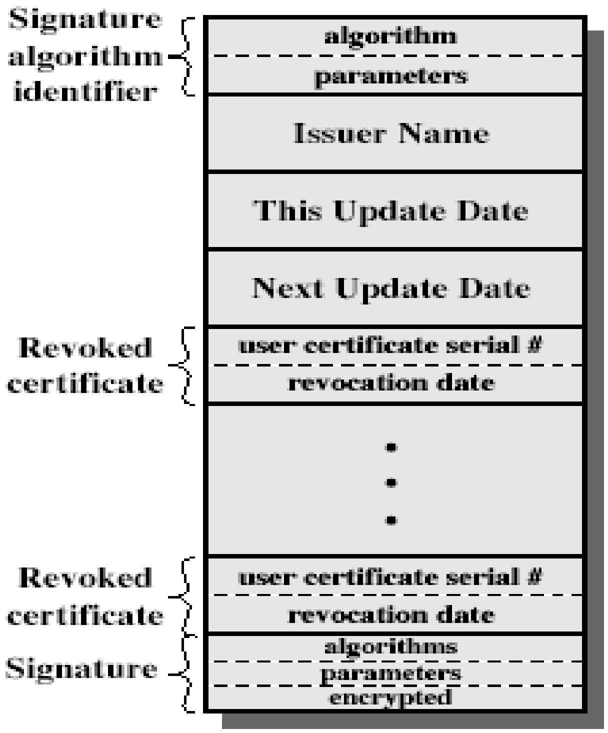
Issuer Unique Identifier

Subject Unique Identifier

Extensions

Signature: algorithms, parameters, encrypted

(a) X.509 Certificate



Signature algorithm identifier: algorithm, parameters

Issuer Name

This Update Date

Next Update Date

Revoked certificate: user certificate serial #, revocation date

•
•
•

Revoked certificate: user certificate serial #, revocation date

Signature: algorithms, parameters, encrypted

(b) Certificate Revocation List

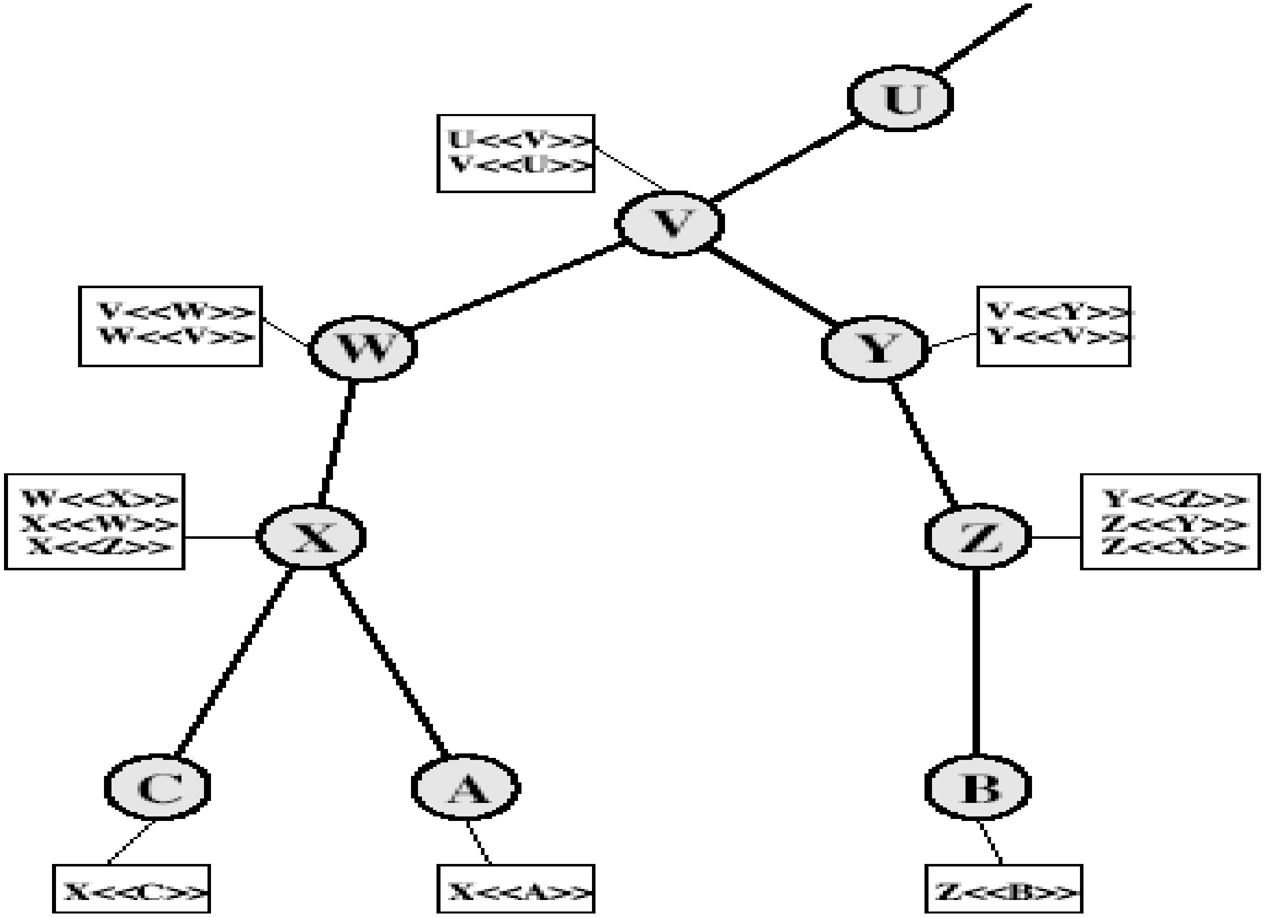
Obtaining a Certificate

- any user with access to CA can get any certificate from it
- only the CA can modify a certificate
- because cannot be forged, certificates can be placed in a public directory

CA Hierarchy

- if both users share a common CA then they are assumed to know its public key
- otherwise CA's must form a hierarchy
- use certificates linking members of hierarchy to validate other CA's
  - each CA has certificates for clients (forward) and parent (backward)
- each client trusts parents certificates
- enable verification of any certificate from one CA by users of all other CAs in hierarchy

CA Hierarchy Use



Certificate Revocation

- certificates have a period of validity
- may need to revoke before expiry, eg:
  1. user's private key is compromised
  2. user is no longer certified by this CA
  3. CA's certificate is compromised
- CA’s maintain list of revoked certificates
  – the Certificate Revocation List (CRL)
- users should check certs with CA’s CRL

Authentication Procedures

- X.509 includes three alternative authentication procedures:
- One-Way Authentication
- Two-Way Authentication
- Three-Way Authentication
- all use public-key signatures

One-Way Authentication

- 1 message (A->B) used to establish
  - the identity of A and that message is from A
  - message was intended for B
  - integrity & originality of message
- message must include timestamp, nonce, B's identity and is signed by A

Two-Way Authentication

- 2 messages (A->B, B->A) which also establishes in addition:
  - the identity of B and that reply is from B
  - that reply is intended for A
  - integrity & originality of reply
- reply includes original nonce from A, also timestamp and nonce from B

Three-Way Authentication

- 3 messages (A->B, B->A, A->B) which enables above authentication without synchronized clocks
- has reply from A back to B containing signed copy of nonce from B
- means that timestamps need not be checked or relied upon

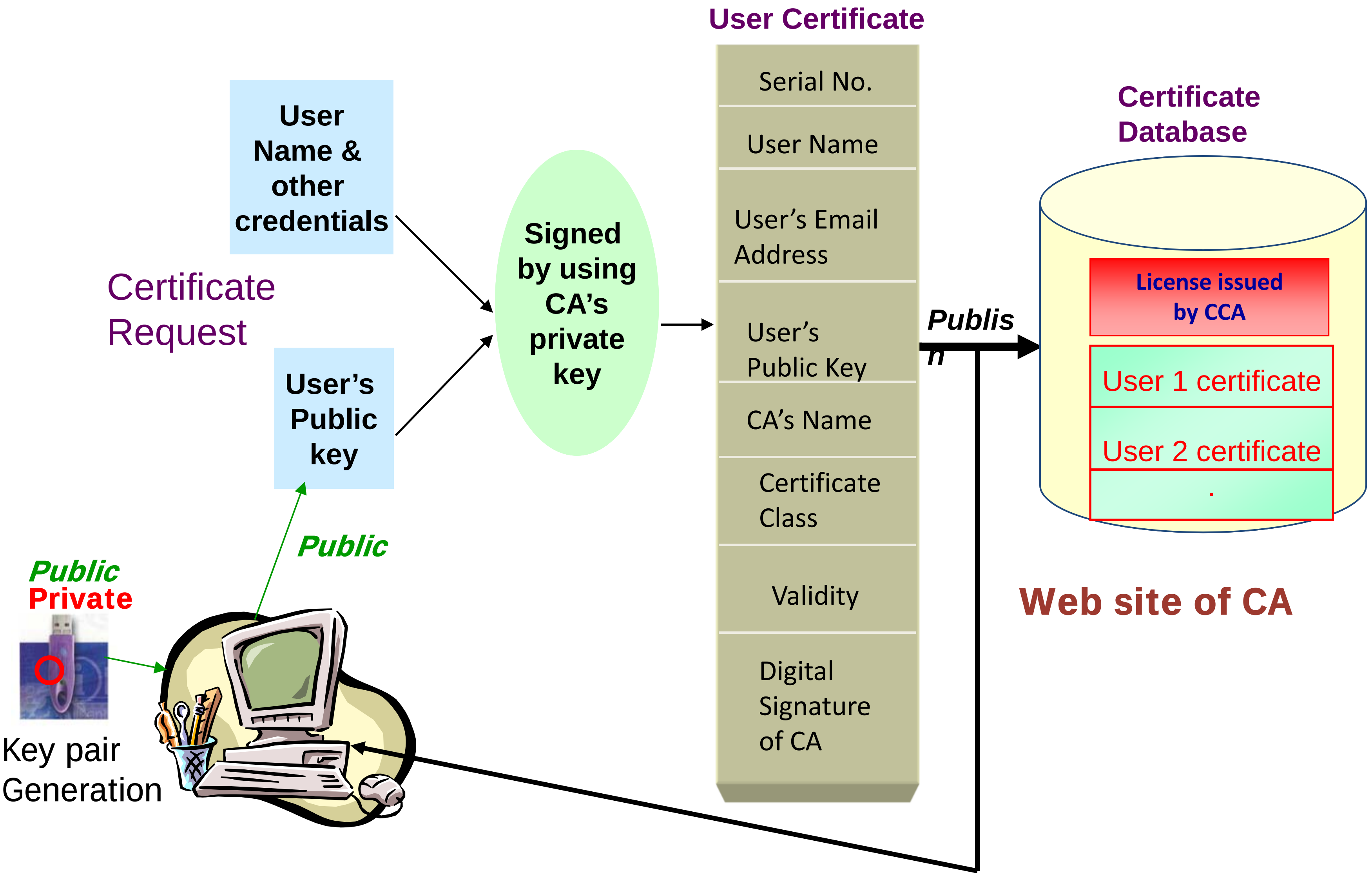
X.509 Version 3

- has been recognised that additional information is needed in a certificate
  - email/URL, policy details, usage constraints
- rather than explicitly naming new fields defined a general extension method
- extensions consist of:
  - extension identifier
  - criticality indicator
  - extension value

Certificate Extensions

- key and policy information
  - convey info about subject & issuer keys, plus indicators of certificate policy
- certificate subject and issuer attributes
  - support alternative names, in alternative formats for certificate subject and/or issuer
- certificate path constraints
  - allow constraints on use of certificates by other CA's

Public-Key Certification



User Certificate

- Serial No.
- User Name
- User's Email Address
- User's Public Key
- CA's Name
- Certificate Class
- Validity
- Digital Signature of CA

Certificate Request

User Name & other credentials

User's Public key

Signed by using CA's private key

Publish

Certificate Database

License issued by CCA

User 1 certificate

User 2 certificate

.

Web site of CA

Public

Public
Private

Key pair Generation

Private key of CA or CCA require highest level of security

Hardware Security Module (HSM) is used for storing the Private Key

More than one person are required for signing

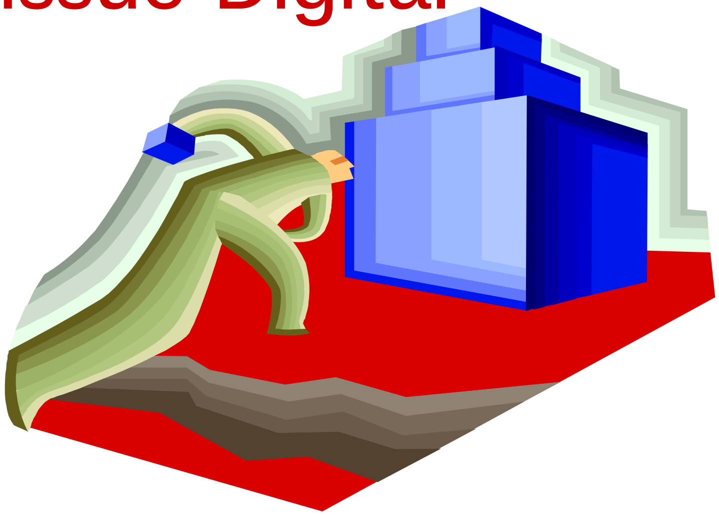
HSM is housed in a strong room with video surveillance on 24x7 basis.

Trust Path

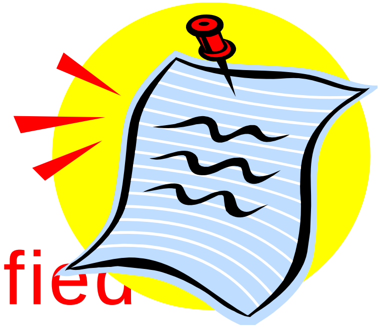
- Controller is the Root certifying authority responsible for regulating Certifying Authorities (CAs)
- Controller certifies the association of CA with his public key
- Certifying Authority (CA) is the trusted authority responsible for creating or certifying identities.
- CA certifies the association of an individual with his public key

Role of controller

Controller of Certifying Authorities as the “Root” Authority certifies the technologies,infrastructure and practices of all the Certifying Authorities licensed to issue Digital Signature Certificates



Summary

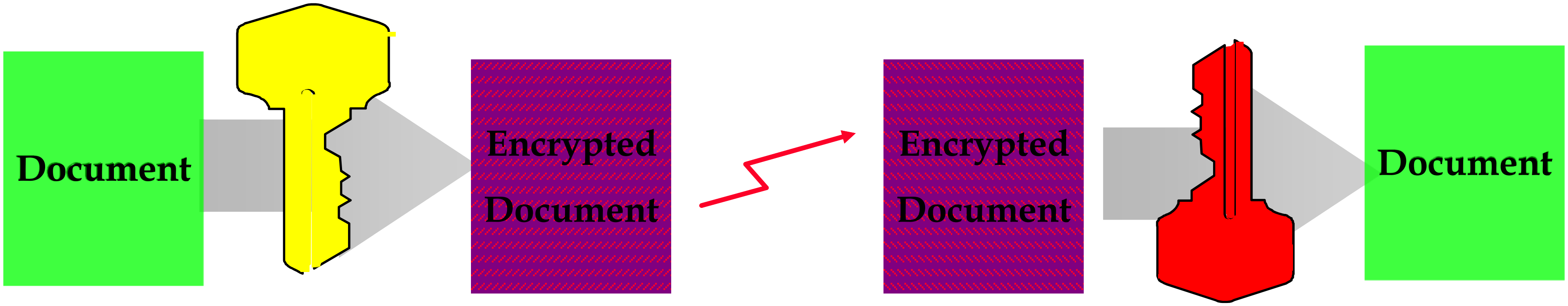


- Each individual has a pair of keys
- Public key of each individual is certified by a CA (Certifying Authority)
- Public keys of CAs are certified by the Controller
- Public key of the Controller is self certified
- Public keys of everyone are known to all concerned and are also available on the web
- Certification Practice Statement is displayed on the web site

Public Key Cryptography

Encryption Technologies

Confidentiality



Document

Encrypted Document

Encrypted Document

Document

Public Key of B

Private Key of B

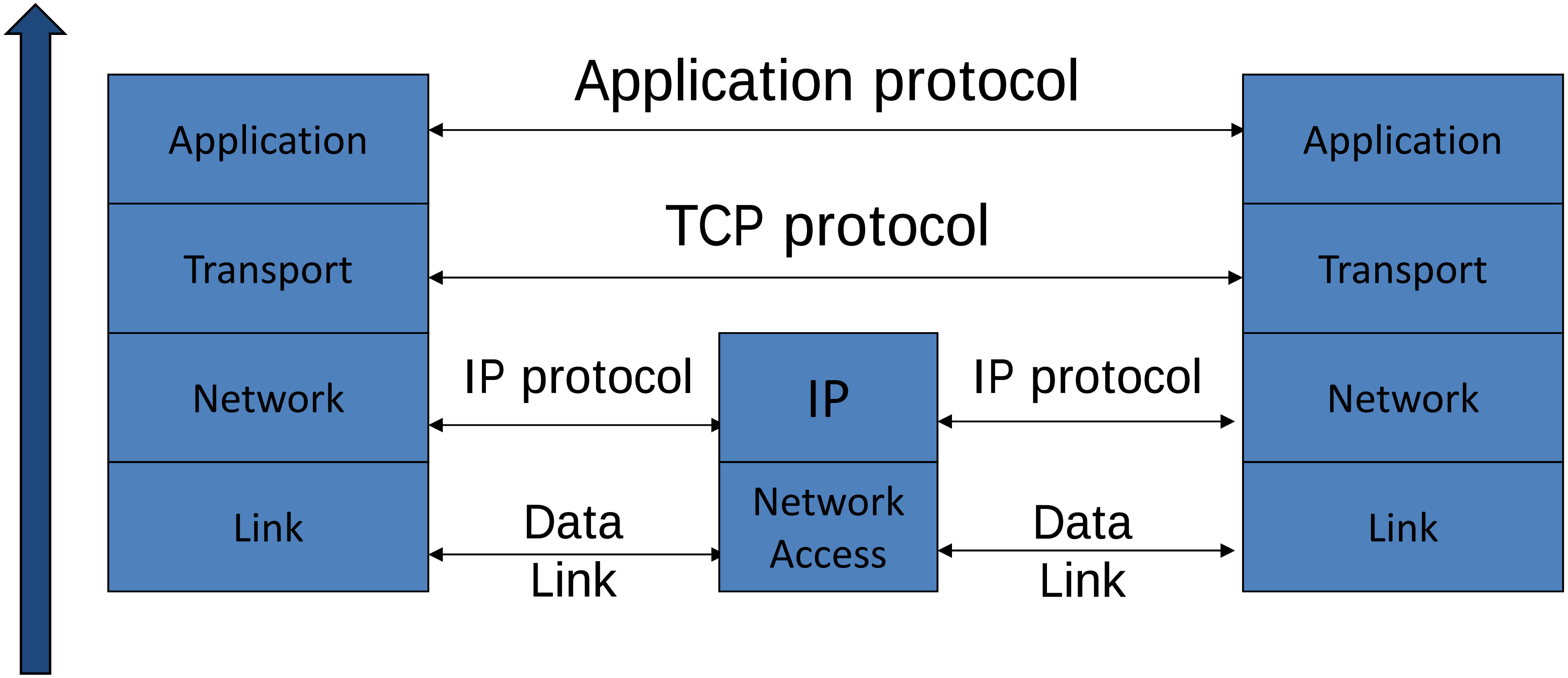
Rest of the Networking Security

- Network protocol security
  - Wireless access– 802.11i/WPA2
  - IPSEC
  - BGP instability and S-BGP
  - DNS rebinding and DNSSEC
- Standard network defenses
  - Firewall
    - Packet filter (stateless, stateful), Application layer proxies
  - Intrusion detection
    - Anomaly and misuse detection

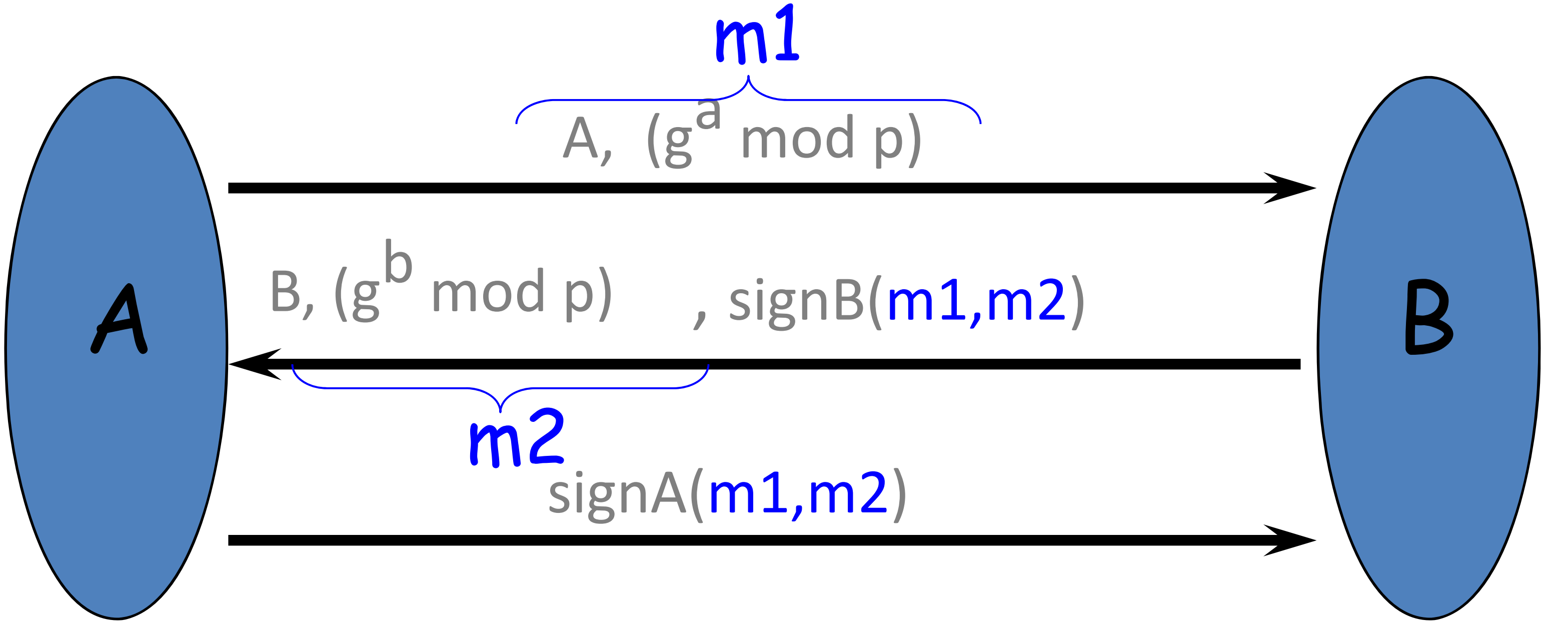
Last Module

- Basic network protocols
  - IP, TCP, UDP, BGP, DNS
- Problems with them
  - TCP/IP
    - No SRC authentication: can't tell where packet is from
    - Packet sniffing
    - Connection spoofing, sequence numbers
  - BGP: advertise bad routes or close good ones
  - DNS: cache poisoning, rebinding
    - Web security mechanisms rely on DNS

Network Protocol Stack



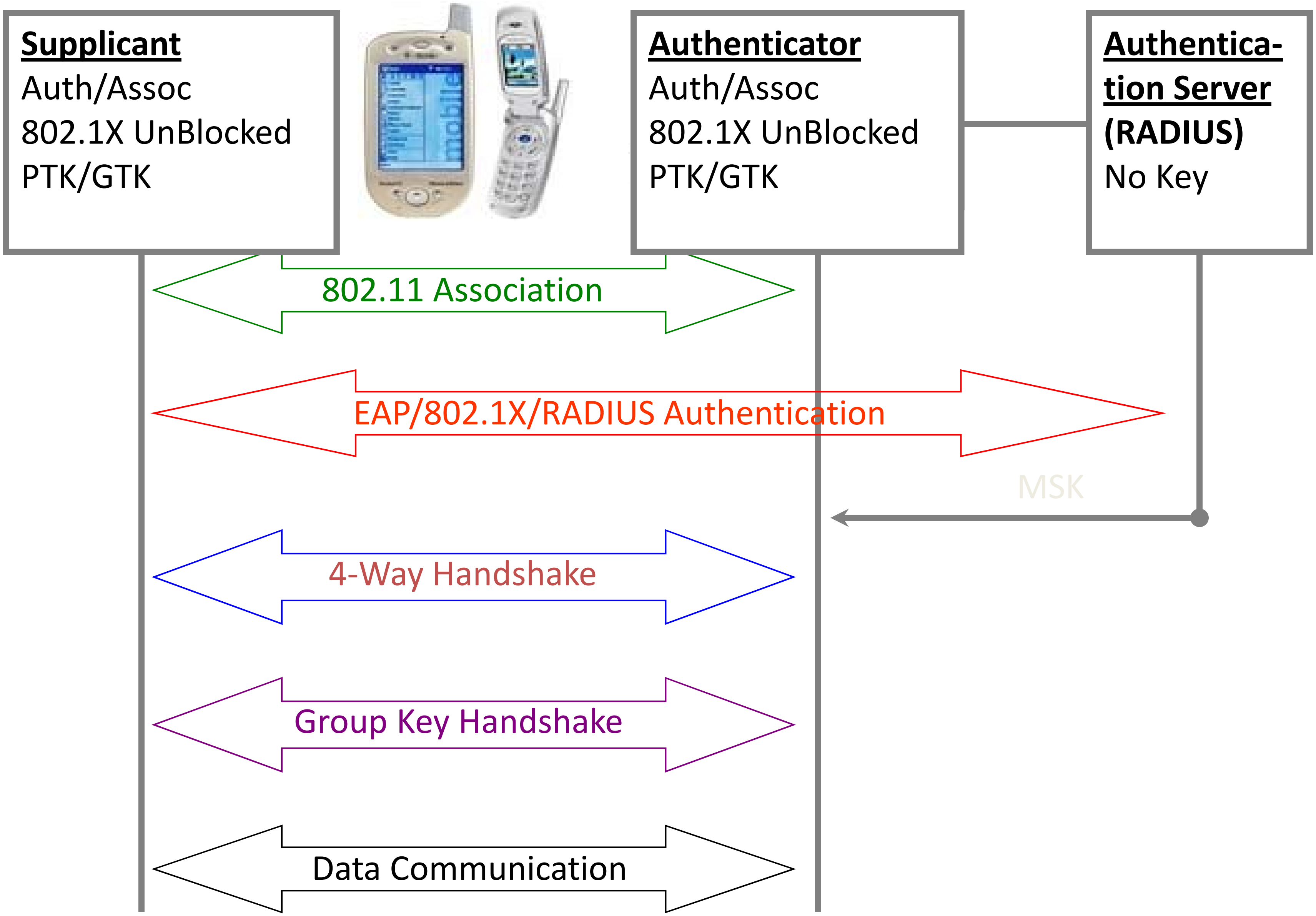
IKE subprotocol from IPSEC



Result: A and B share secret $g^{ab} \bmod p$

Link-layer connectivity

802.11i Protocol



Supplicant
Auth/Assoc
802.1X UnBlocked
PTK/GTK

Authenticator
Auth/Assoc
802.1X UnBlocked
PTK/GTK

Authentication Server (RADIUS)
No Key

802.11 Association

EAP/802.1X/RADIUS Authentication

MSK

4-Way Handshake

Group Key Handshake

Data Communication

TCP/IP connectivity

Transport layer security (from last lecture)

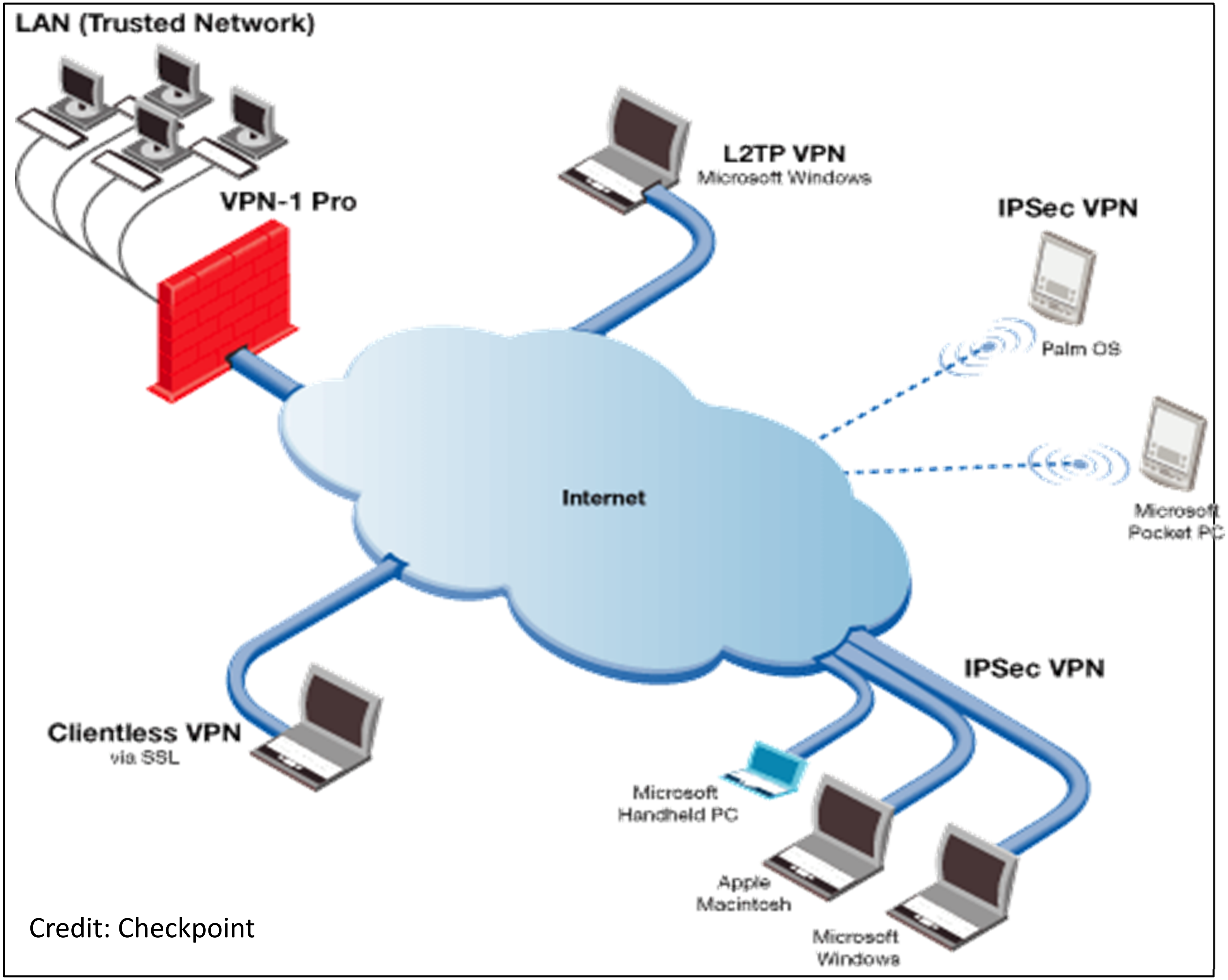
Basic Layer 2-3 Security Problems

- Network packets pass by untrusted hosts
  - Eavesdropping, packet sniffing
  - Especially easy when attacker controls a machine close to victim
- TCP state can be easy to guess
  - Enables spoofing and session hijacking

Virtual Private Network (VPN)

- Three different modes of use:
  - – Remote access client connections
  - – LAN-to-LAN internetworking
  - – Controlled access within an intranet
- Several different protocols
  - – PPTP – Point-to-point tunneling protocol
  - – L2TP – Layer-2 tunneling protocol

    Data layer
  - – IPsec (Layer-3: network layer)

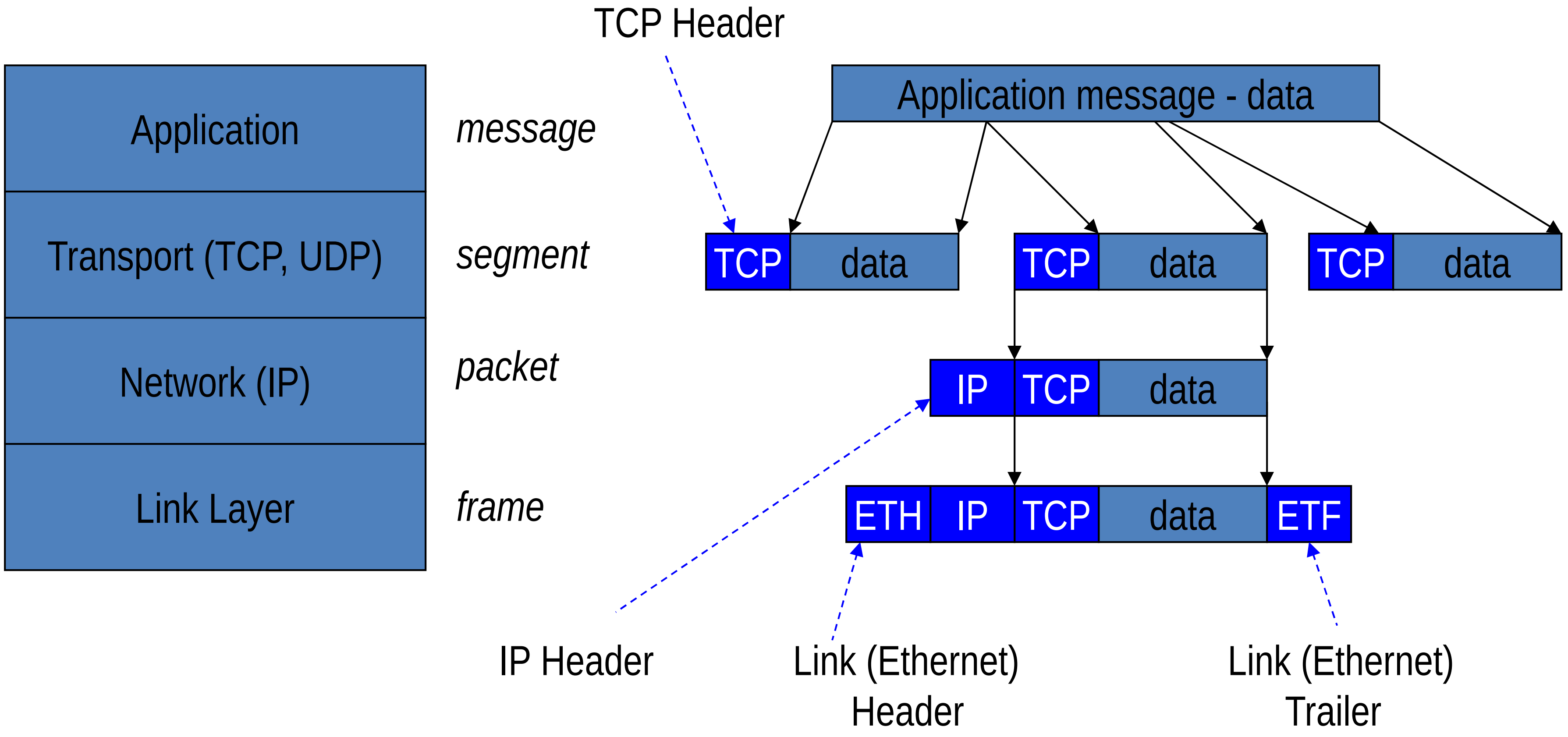


Credit: Checkpoint

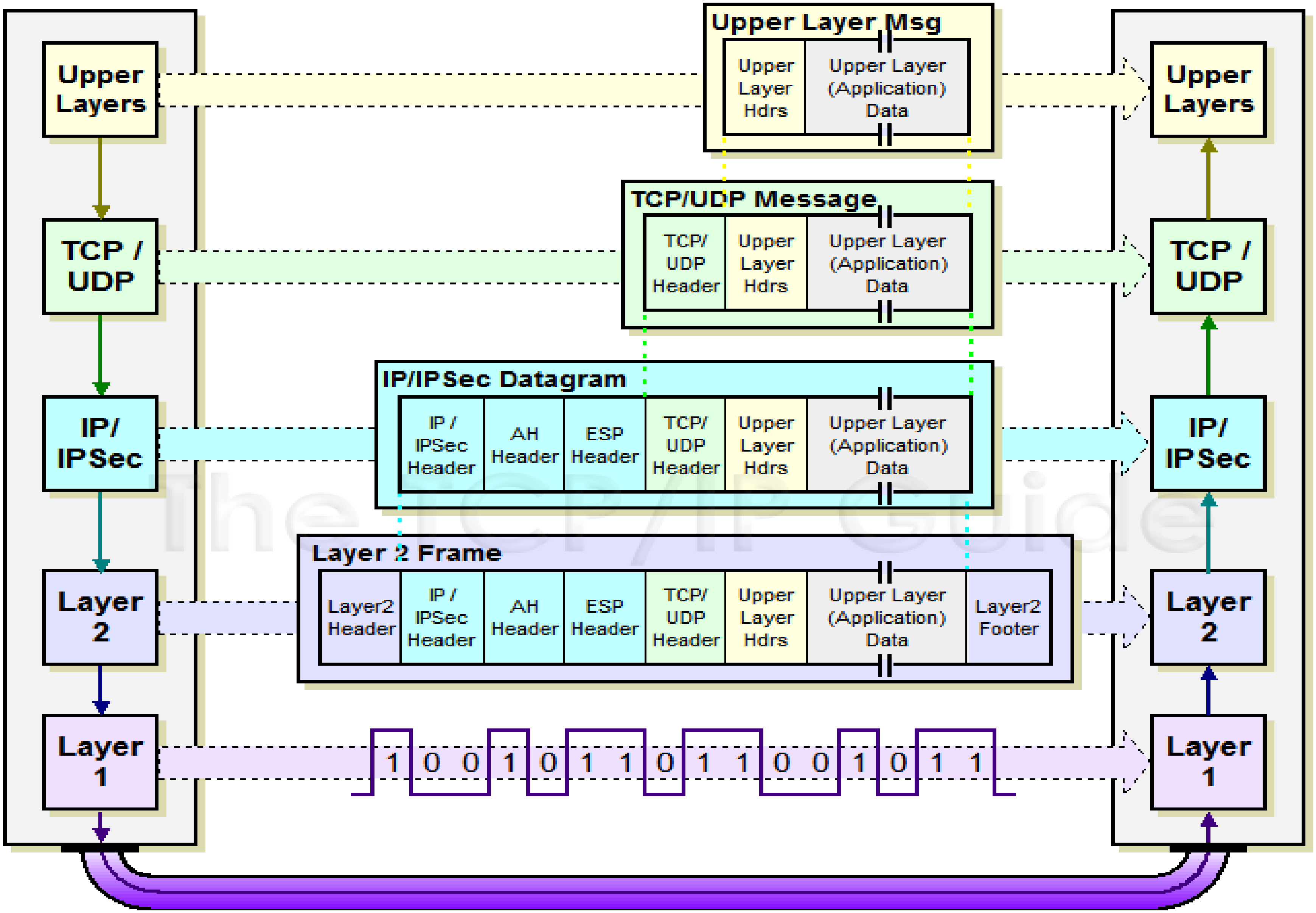
IPSEC

- Security extensions for IPv4 and IPv6
- IP Authentication Header (AH)
  - Authentication and integrity of payload and header
- IP Encapsulating Security Protocol (ESP)
  - Confidentiality of payload
- ESP with optional ICV (integrity check value)
  - Confidentiality, authentication and integrity of payload

Recall packet formats and layers

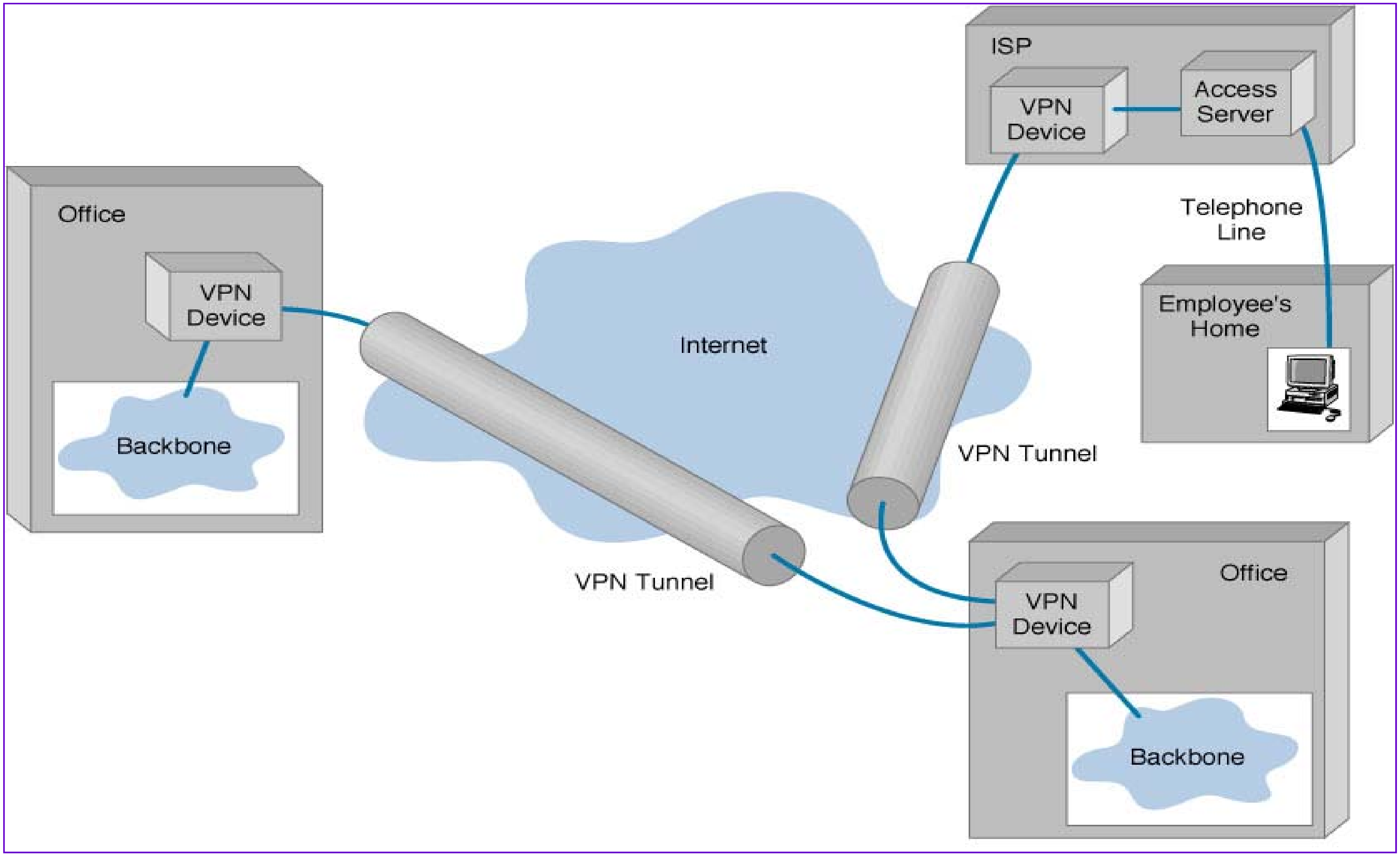


IPSec Transport Mode: IPSEC instead of IP header

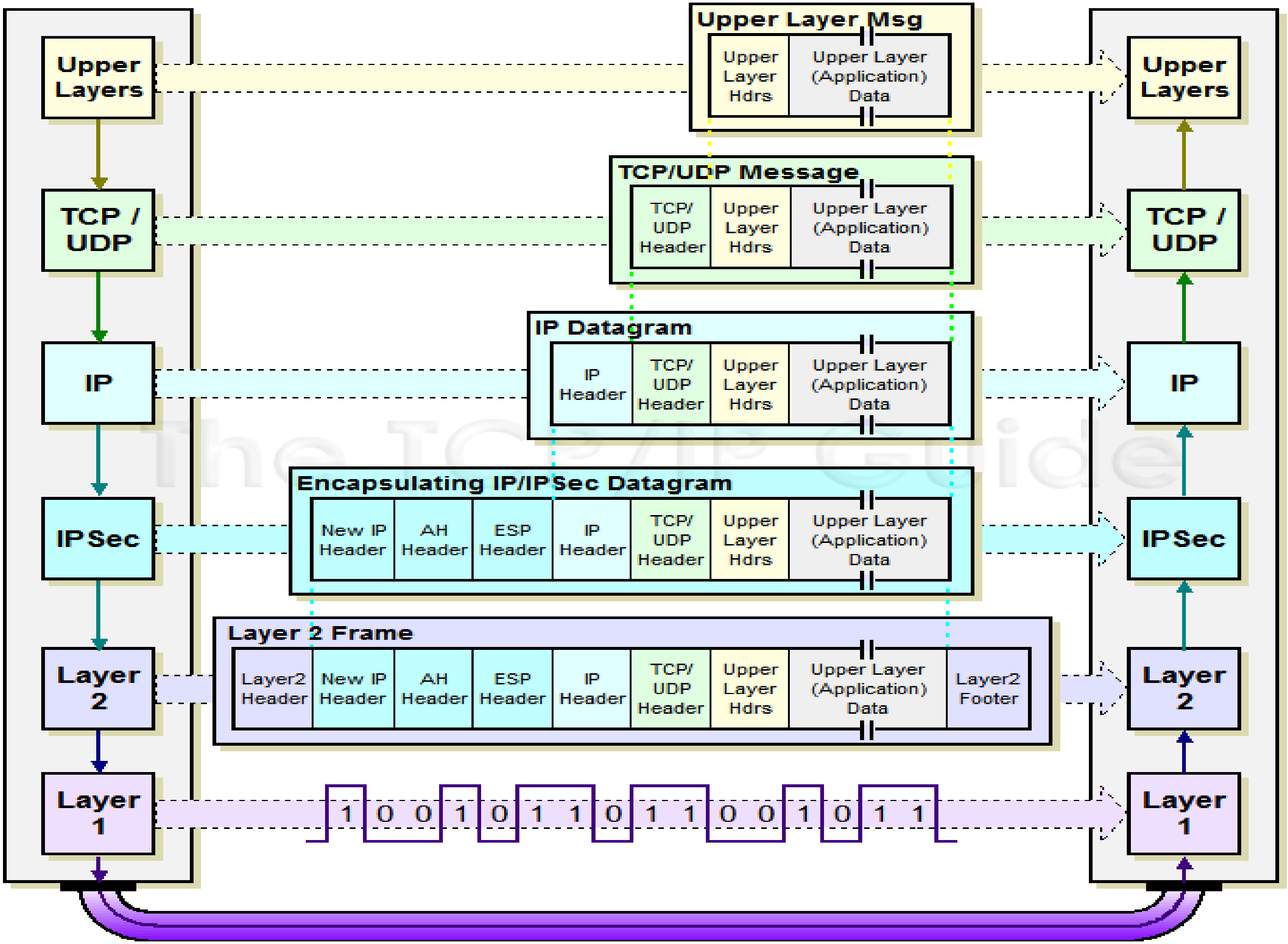


http://www.tcpipguide.com/free/t_IPSecModesTransportandTunnel.htm

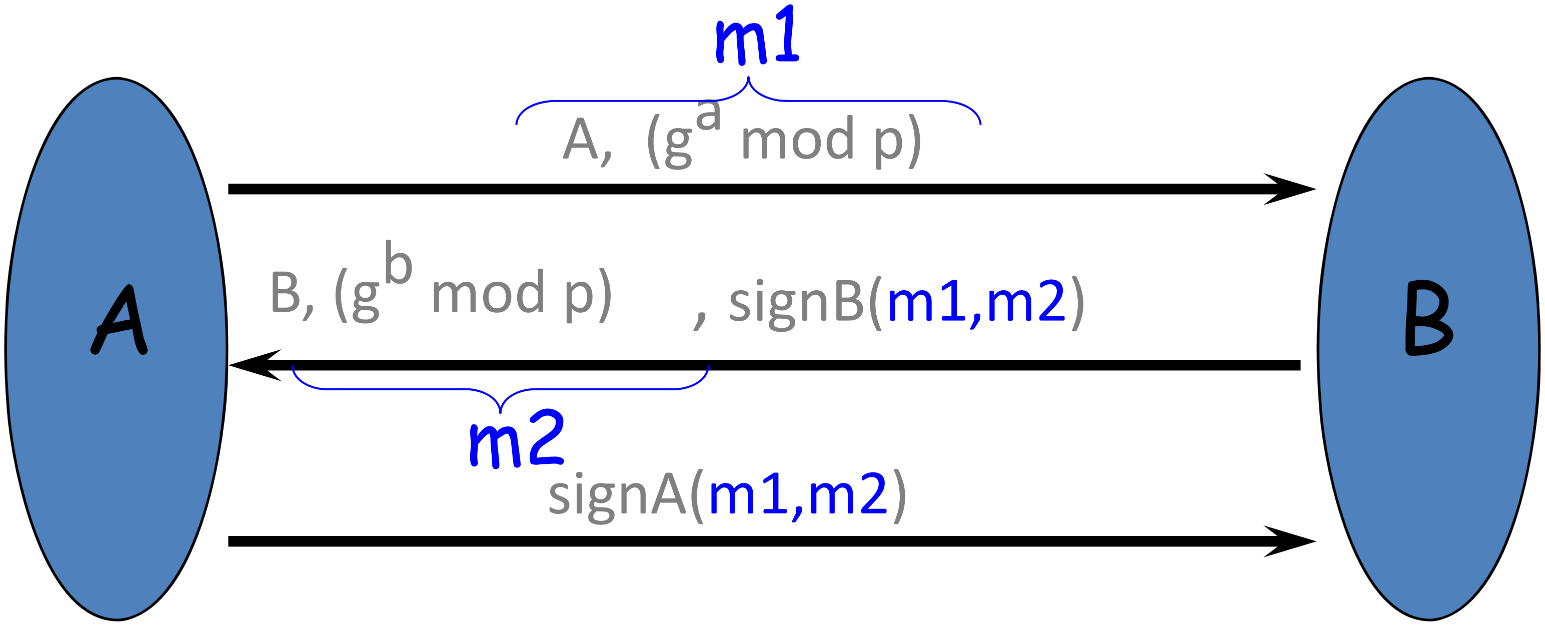
IPSEC Tunnel Mode



IPSec Tunnel Mode: IPSEC header + IP header

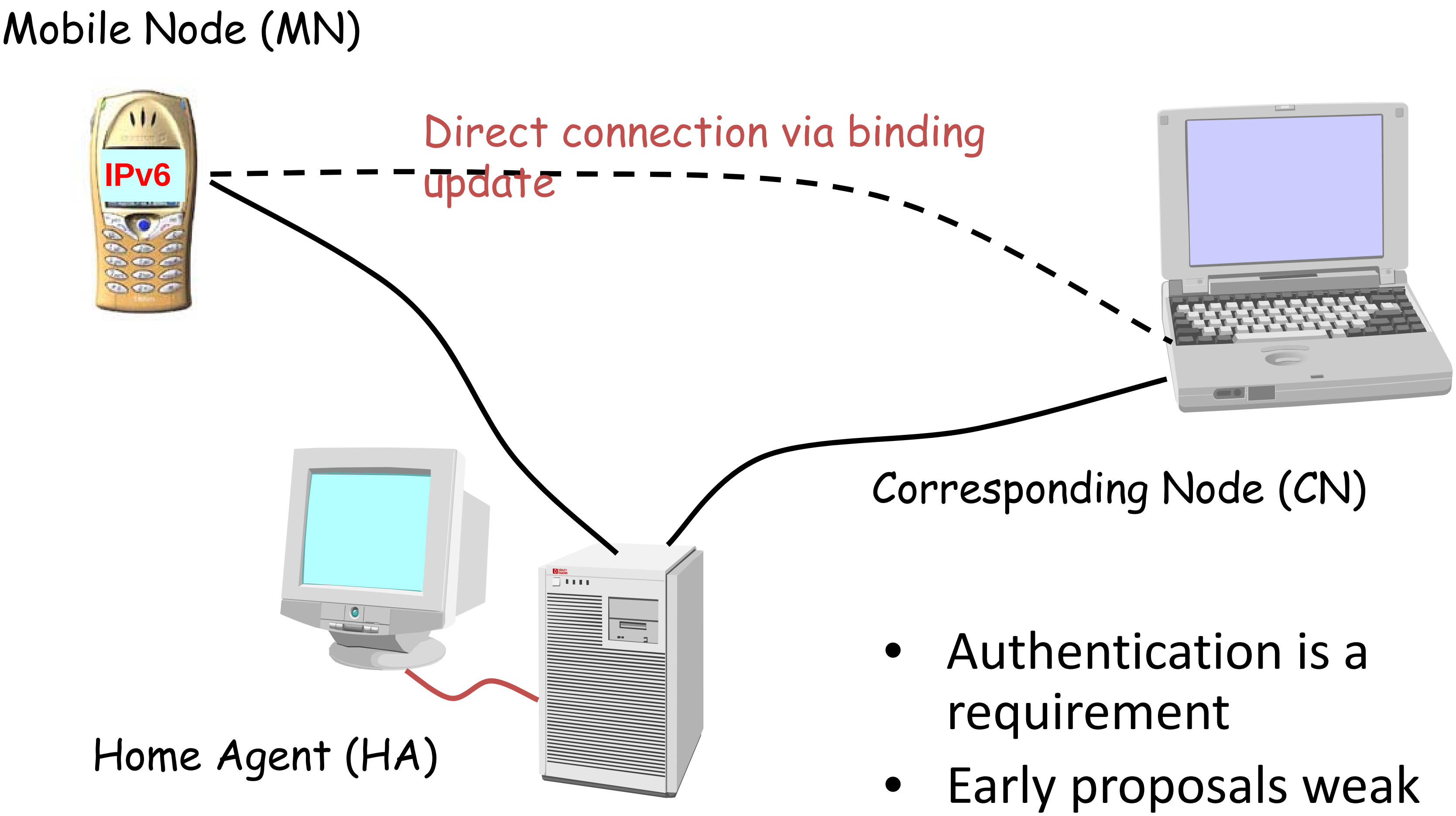


IKE subprotocol from IPSEC



Result: A and B share secret $g^{ab} \bmod p$

Mobile IPv6 Architecture



Mobile Node (MN)

Direct connection via binding update

Corresponding Node (CN)

Home Agent (HA)

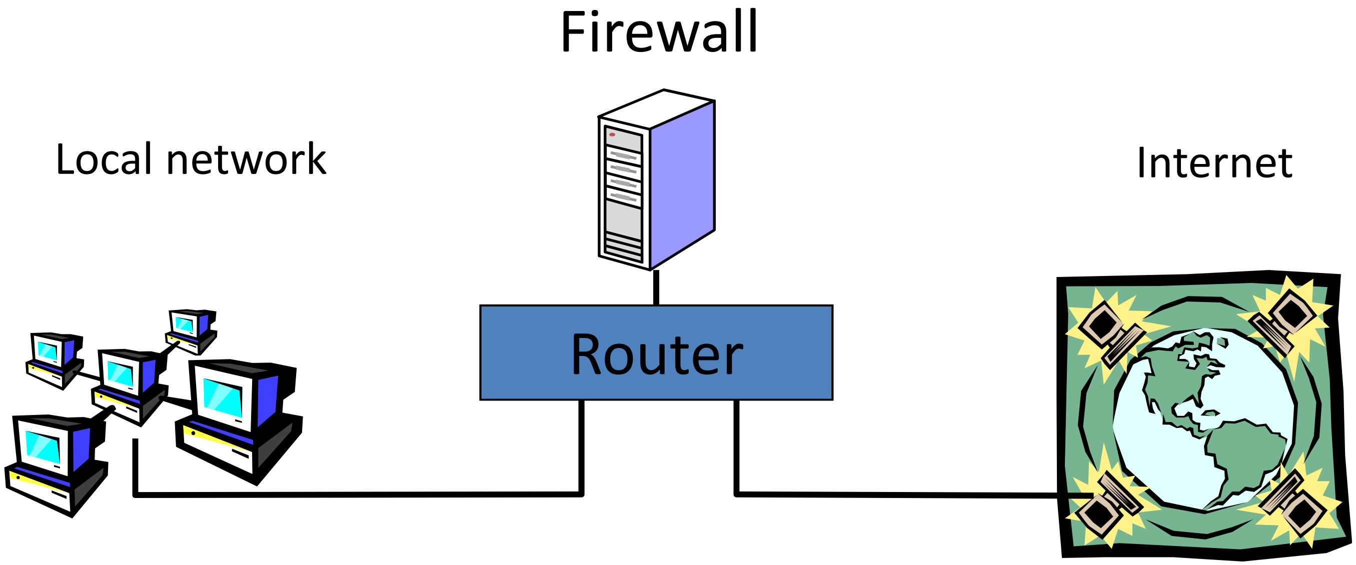
- Authentication is a requirement
- Early proposals weak

Filtering network traffic

(starting at IP, transport layer …)

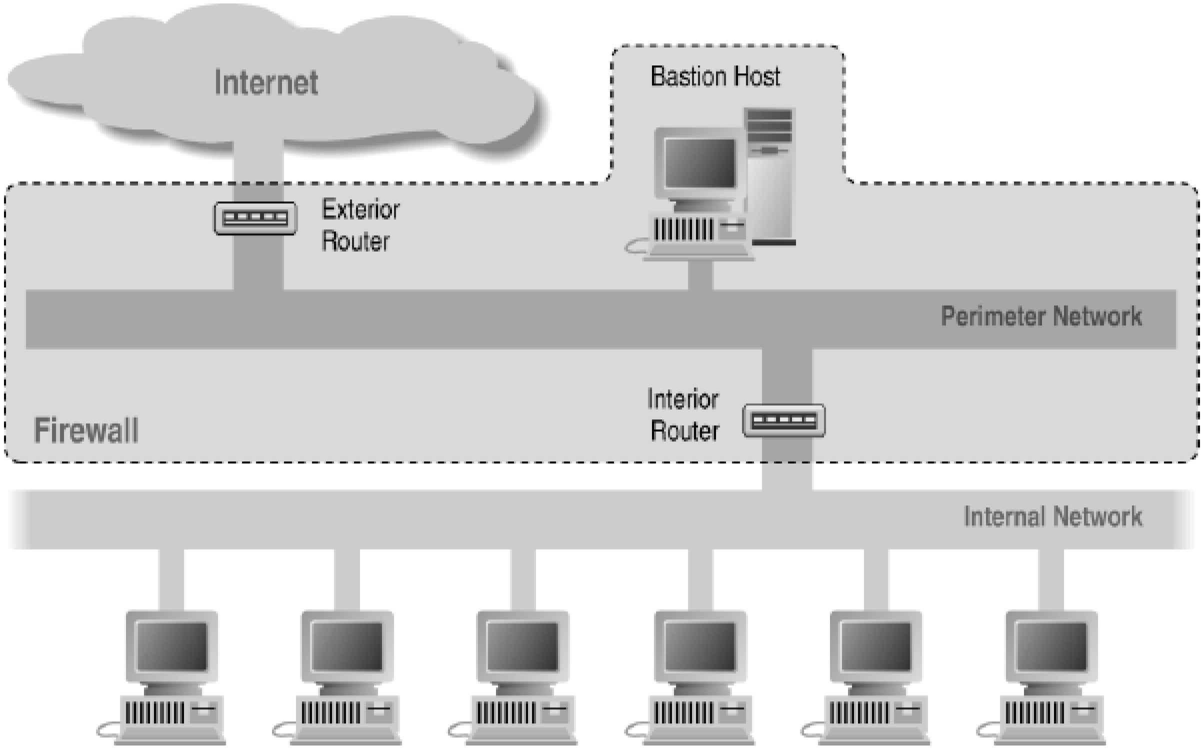
Basic Firewall Concept

- Separate local area net from internet

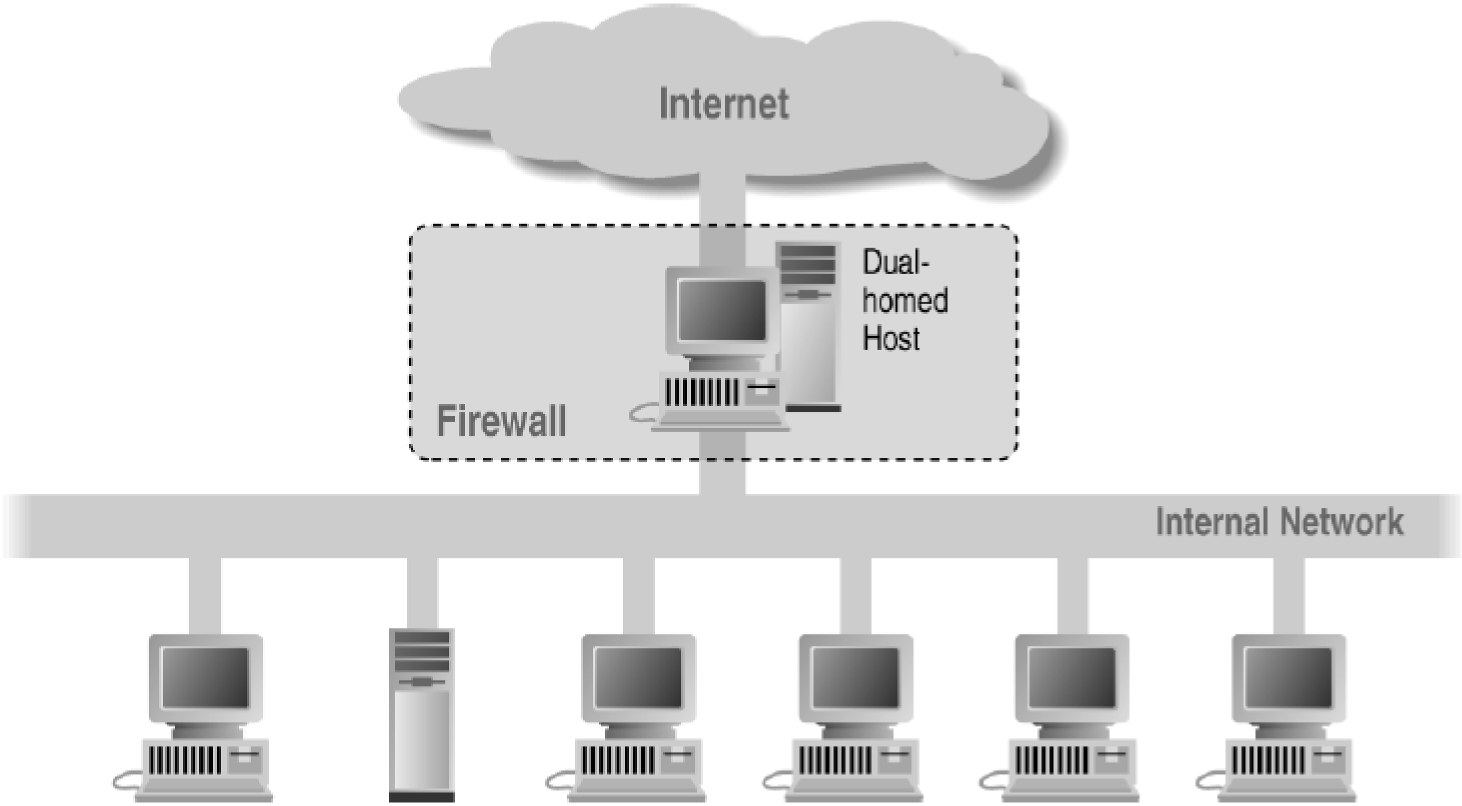


All packets between LAN and internet routed through firewall

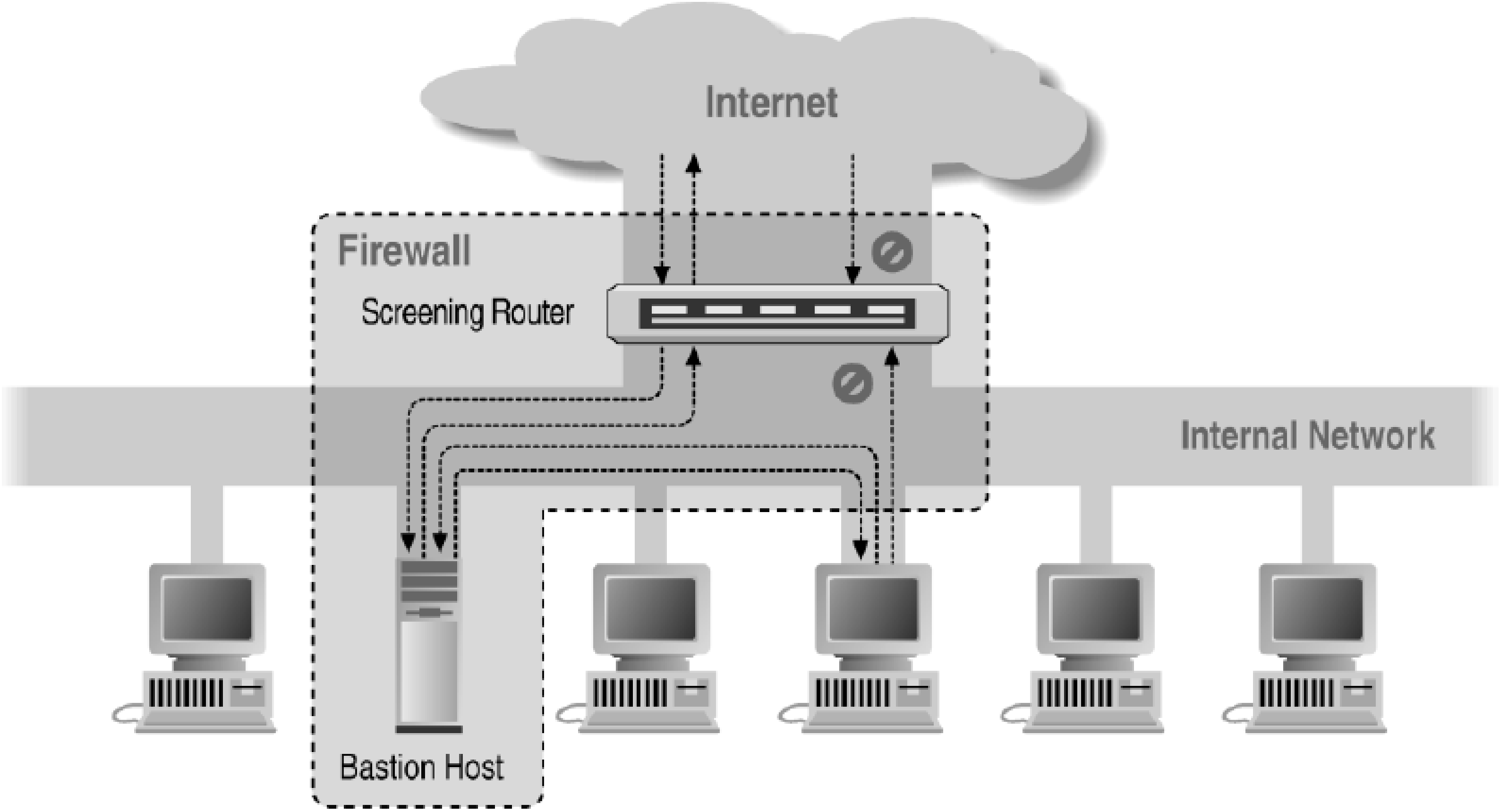
Screened Subnet Using Two Routers



Alternate 1: Dual-Homed Host



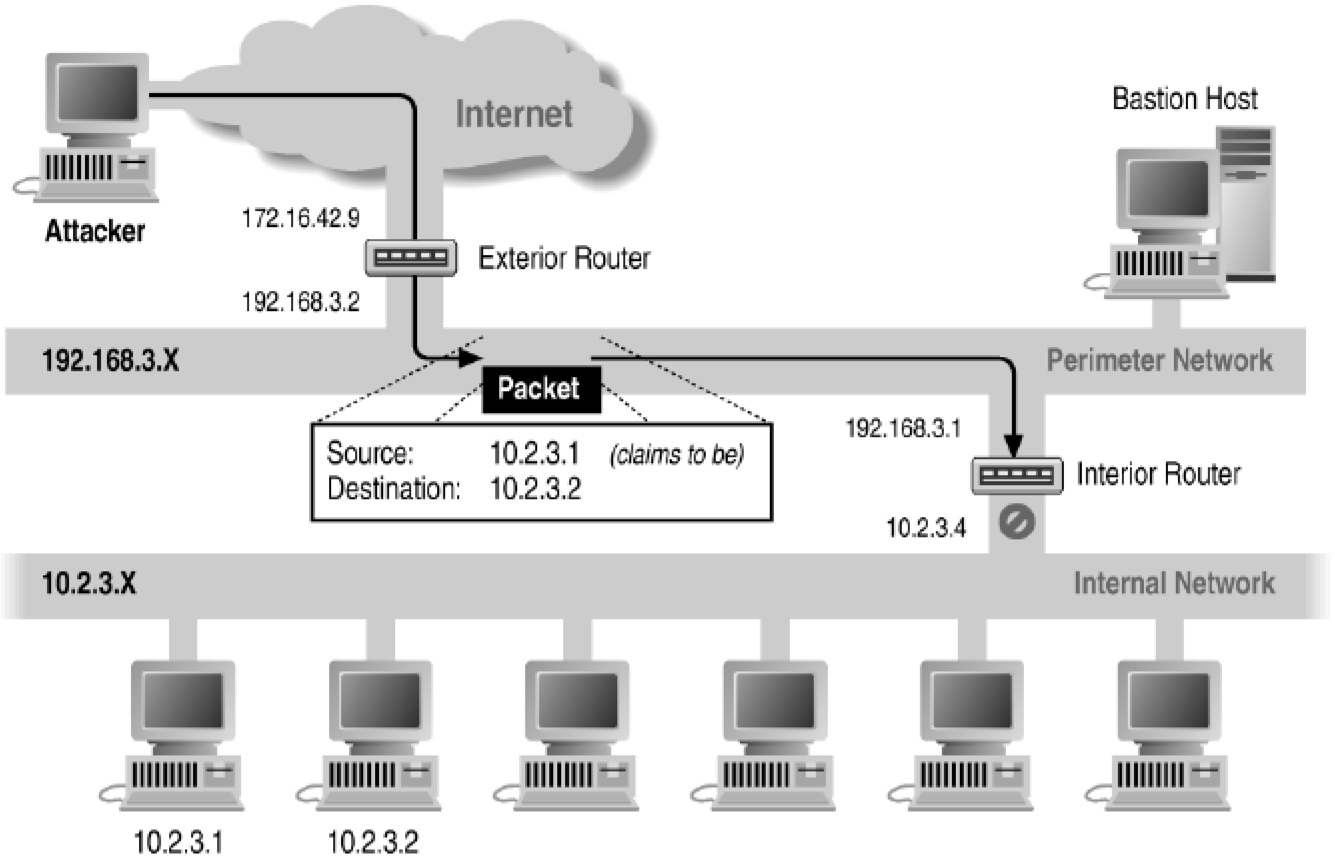
Alternate 2: Screened Host



Basic Packet Filtering

- Uses transport-layer information only
  - IP Source Address, Destination Address
  - Protocol (TCP, UDP, ICMP, etc)
  - TCP or UDP source & destination ports
  - TCP Flags (SYN, ACK, FIN, RST, PSH, etc)
  - ICMP message type
- Examples
  - DNS uses port 53
    - Block incoming port 53 packets except known trusted servers
- Issues
  - Stateful filtering
  - Encapsulation: address translation, other complications
  - Fragmentation

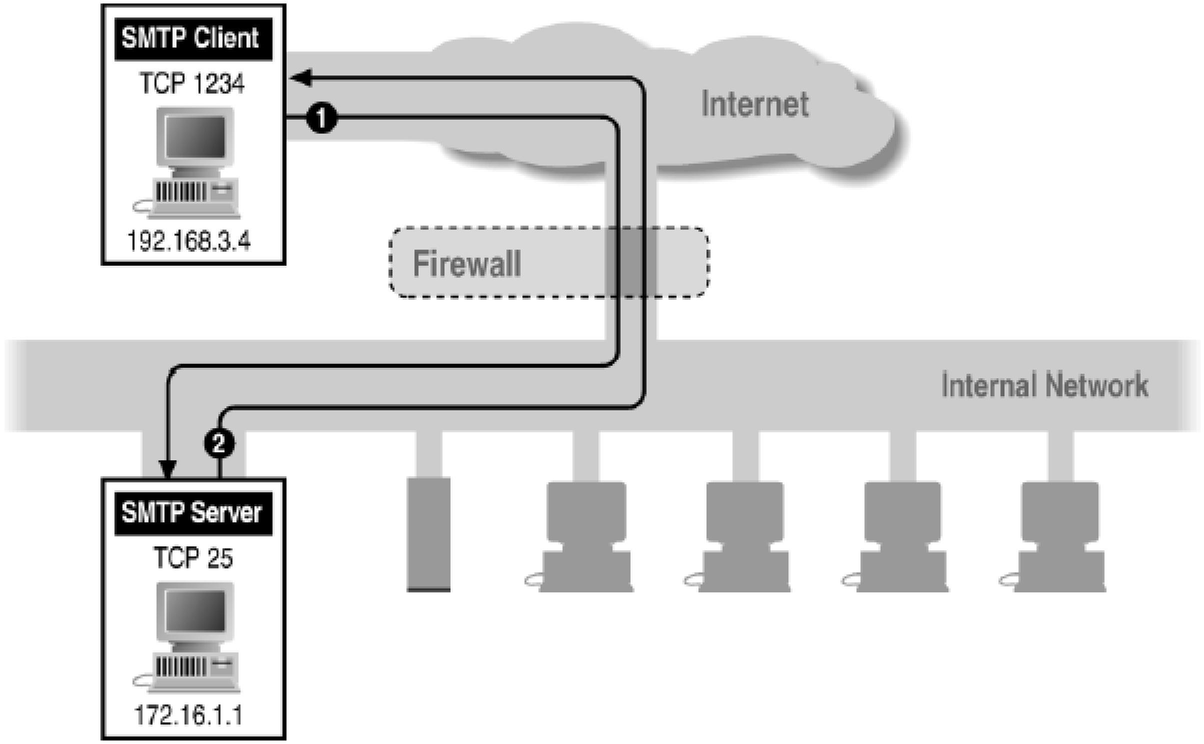
Source/Destination Address Forgery



More about networking: port numbering

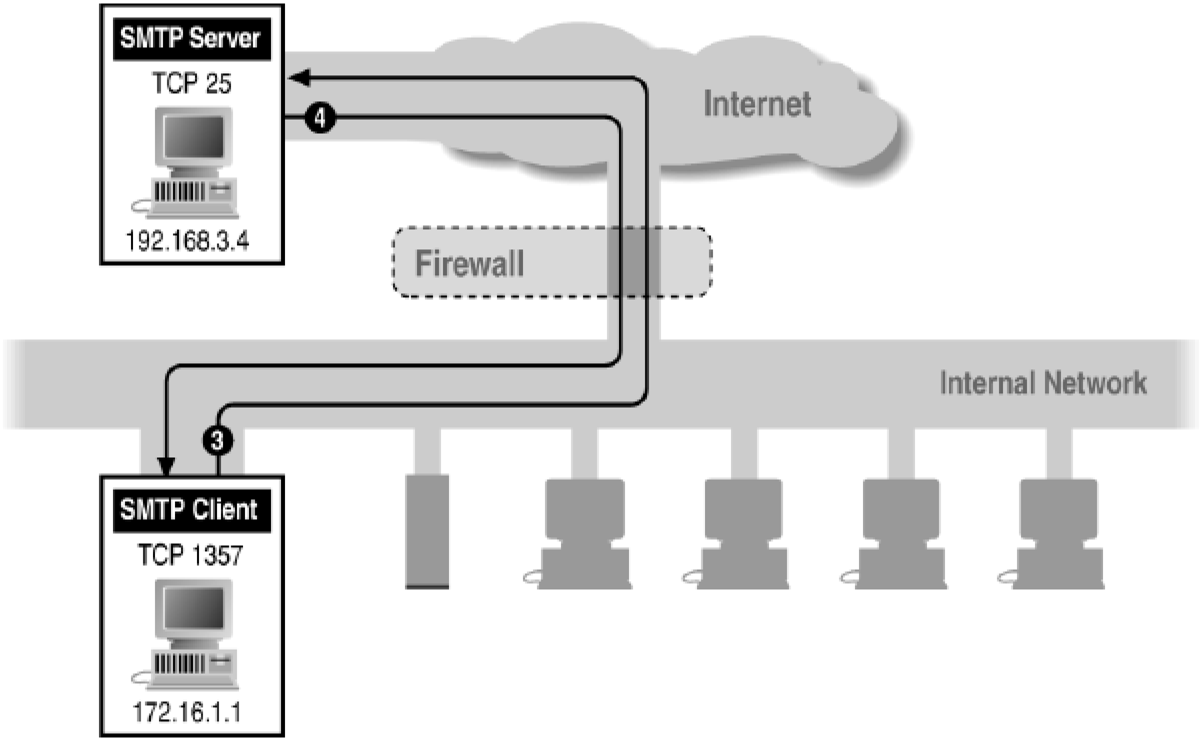
- TCP connection
  - Server port uses number less than 1024
  - Client port uses number between 1024 and 16383
- Permanent assignment
  - Ports <1024 assigned permanently
    - 20,21 for FTP 23 for Telnet
    - 25 for server SMTP 80 for HTTP
- Variable use
  - Ports >1024 must be available for client to make connection
  - Limitation for stateless packet filtering
    - If client wants port 2048, firewall must allow incoming traffic
  - Better: stateful filtering knows outgoing requests
    - Only allow incoming traffic on high port to a machine that has initiated an outgoing request on low port

Filtering Example: Inbound SMTP



Can block external request to internal server based on port number

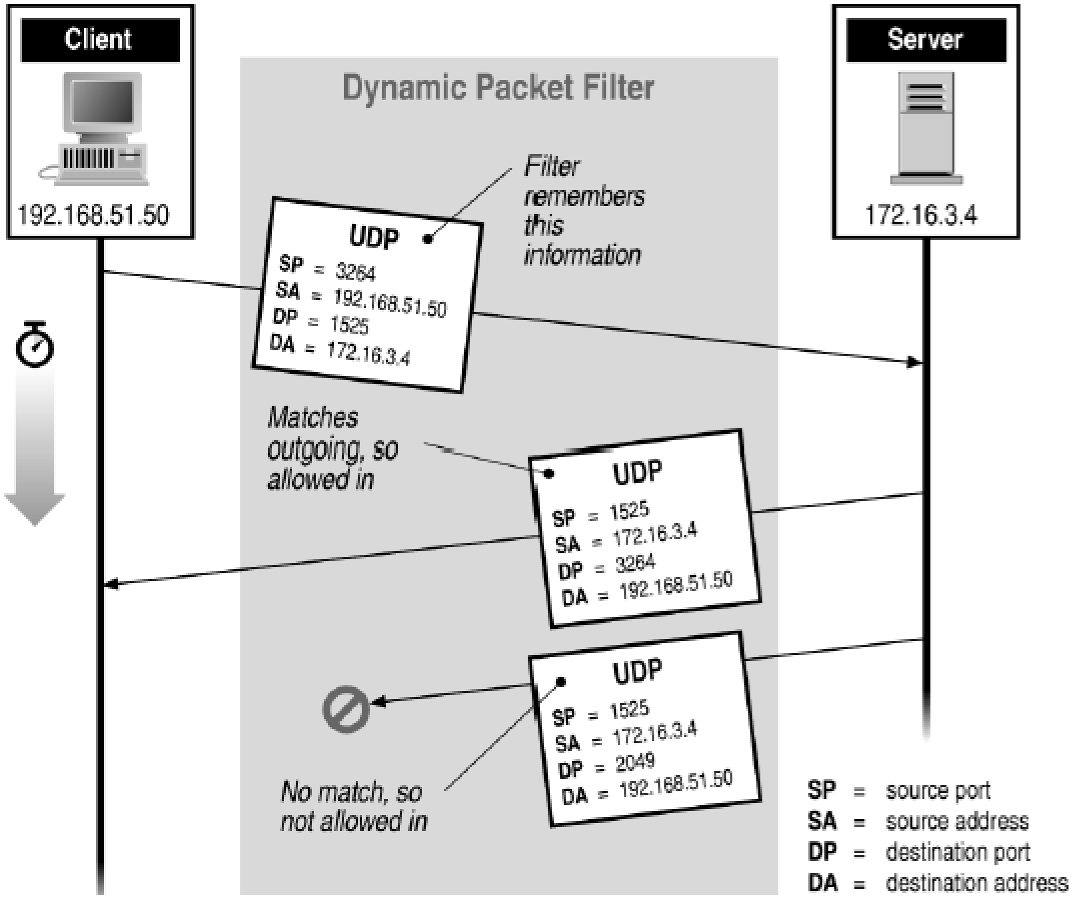
Filtering Example: Outbound SMTP



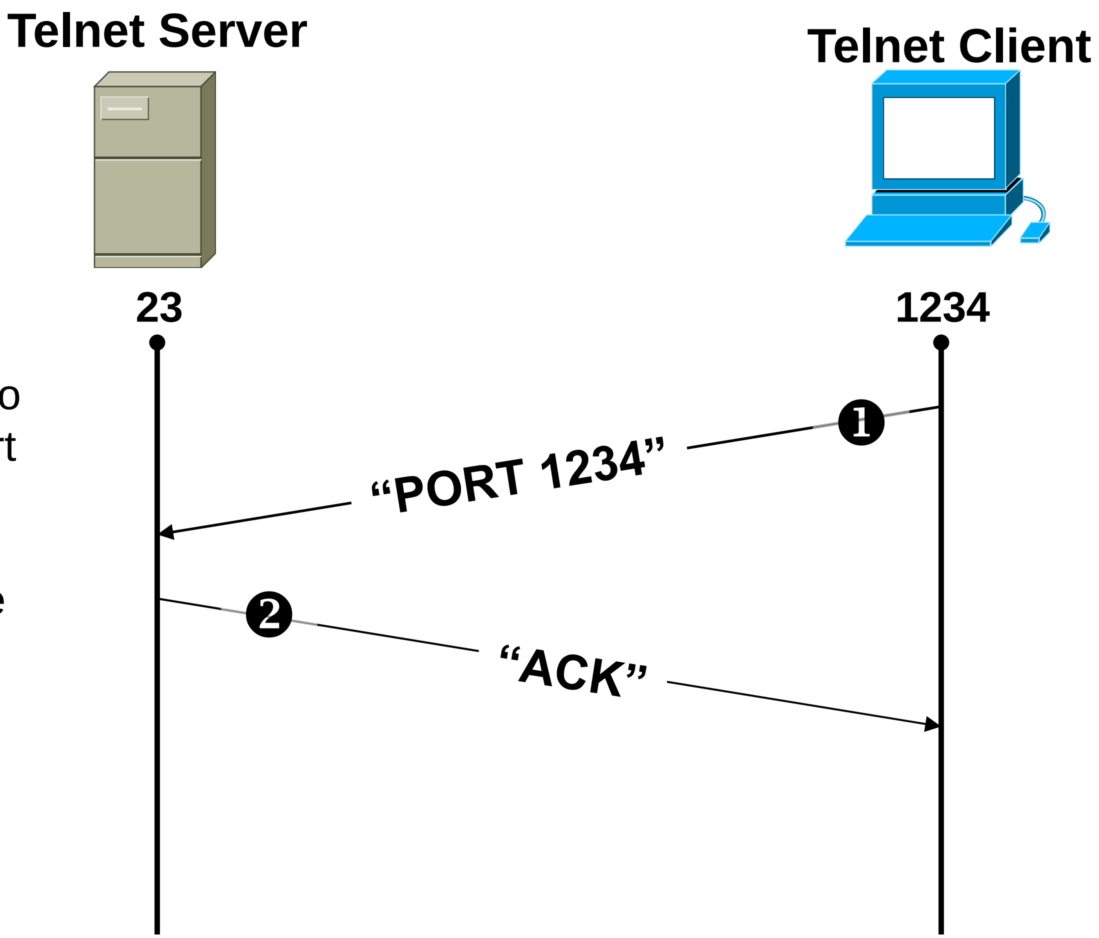
Known low port out, arbitrary high port in

If firewall blocks incoming port 1357 traffic then connection fails

Stateful or Dynamic Packet Filtering



Telnet



❶ Client opens channel to server; tells server its port number. The ACK bit is not set while establishing the connection but will be set on the remaining packets

❷ Server acknowledges

Stateful filtering can use this pattern to identify legitimate sessions

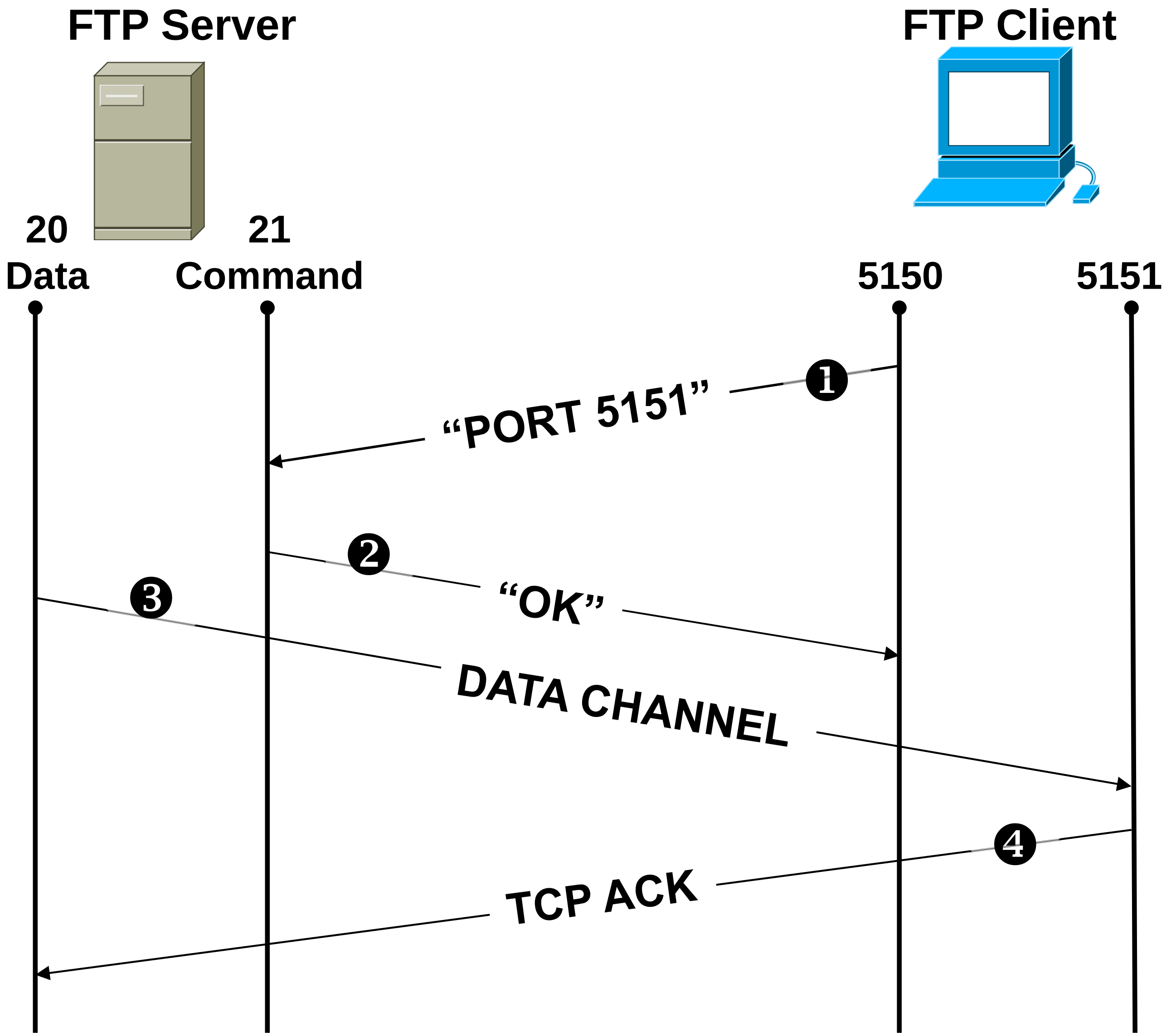
FTP

❶ Client opens command channel to server; tells server second port number

❷ Server acknowledges

❸ Server opens data channel to client’s second port

❹ Client acknowledges



FTP Server

FTP Client

20
Data

21
Command

5150

5151

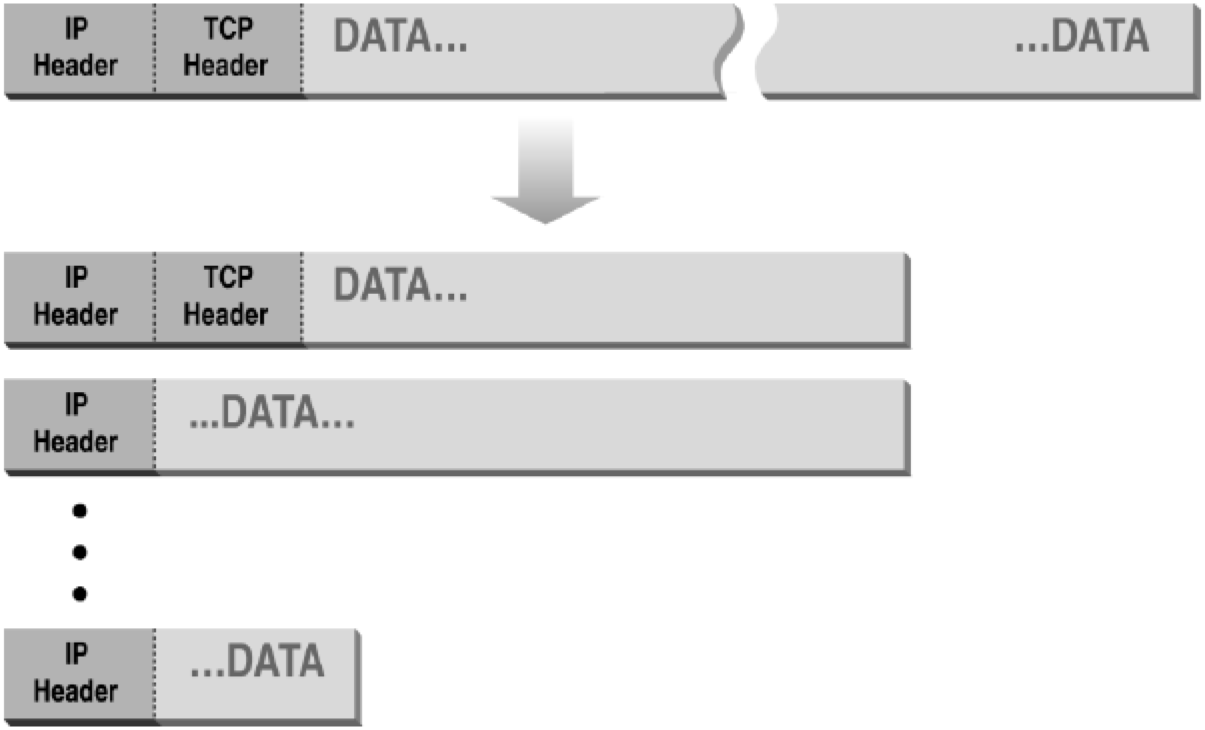
❶ “PORT 5151”

❷ “OK”

❸ DATA CHANNEL

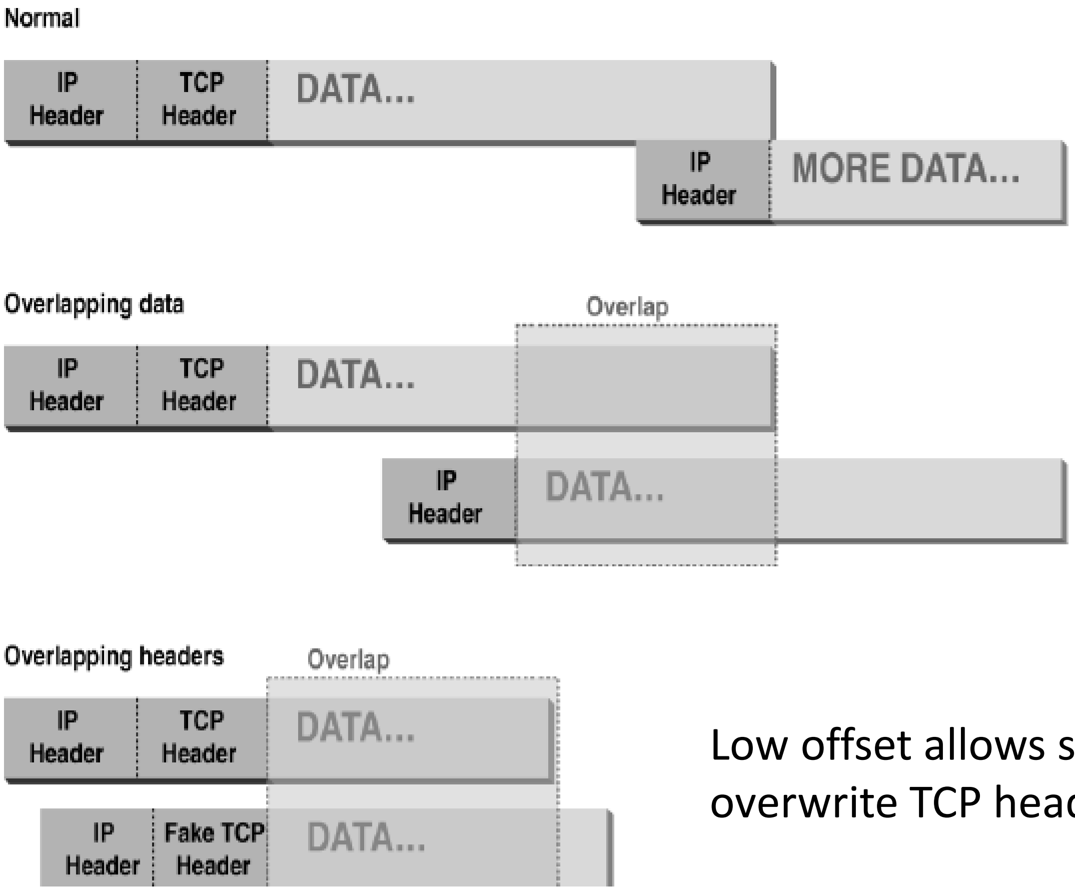
❹ TCP ACK

Normal IP Fragmentation



Flags and offset inside IP header indicate packet fragmentation

Abnormal Fragmentation

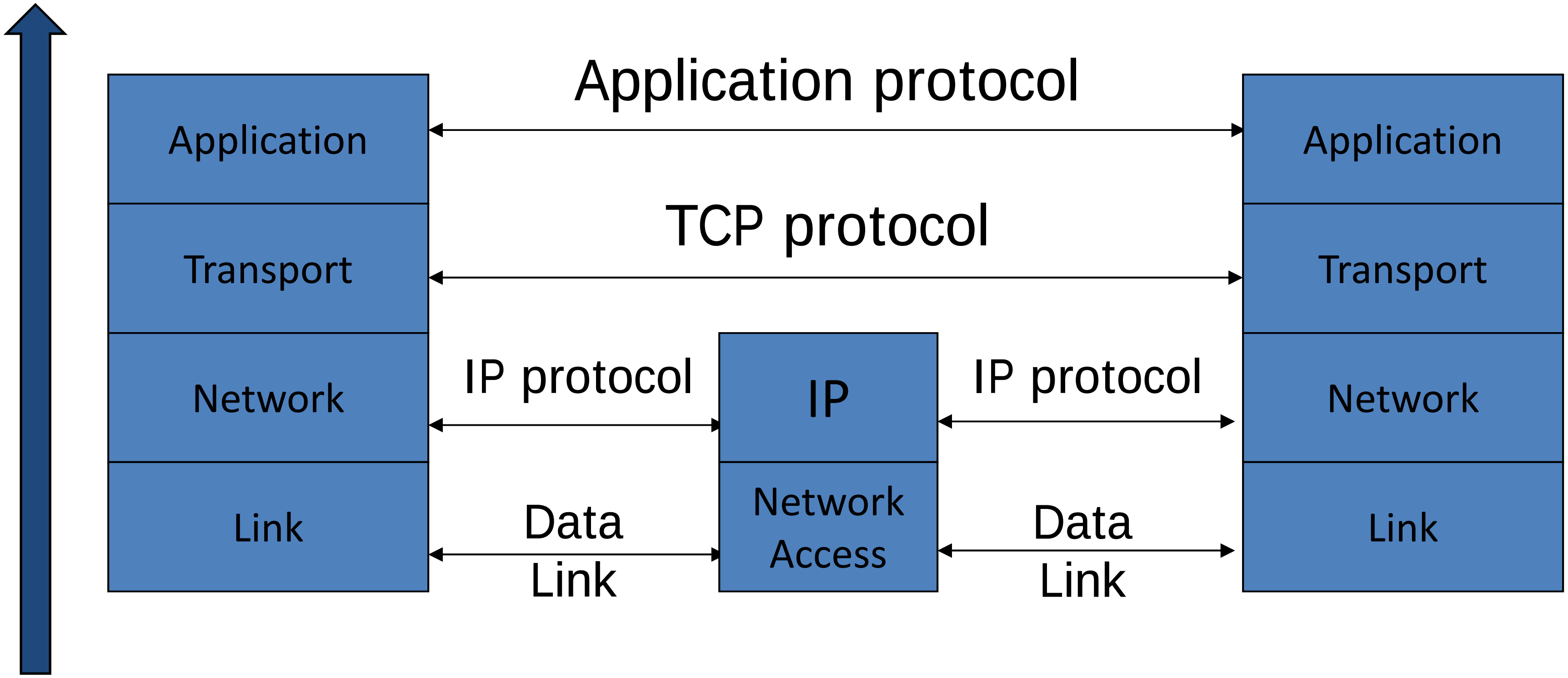


Low offset allows second packet to overwrite TCP header at receiving host

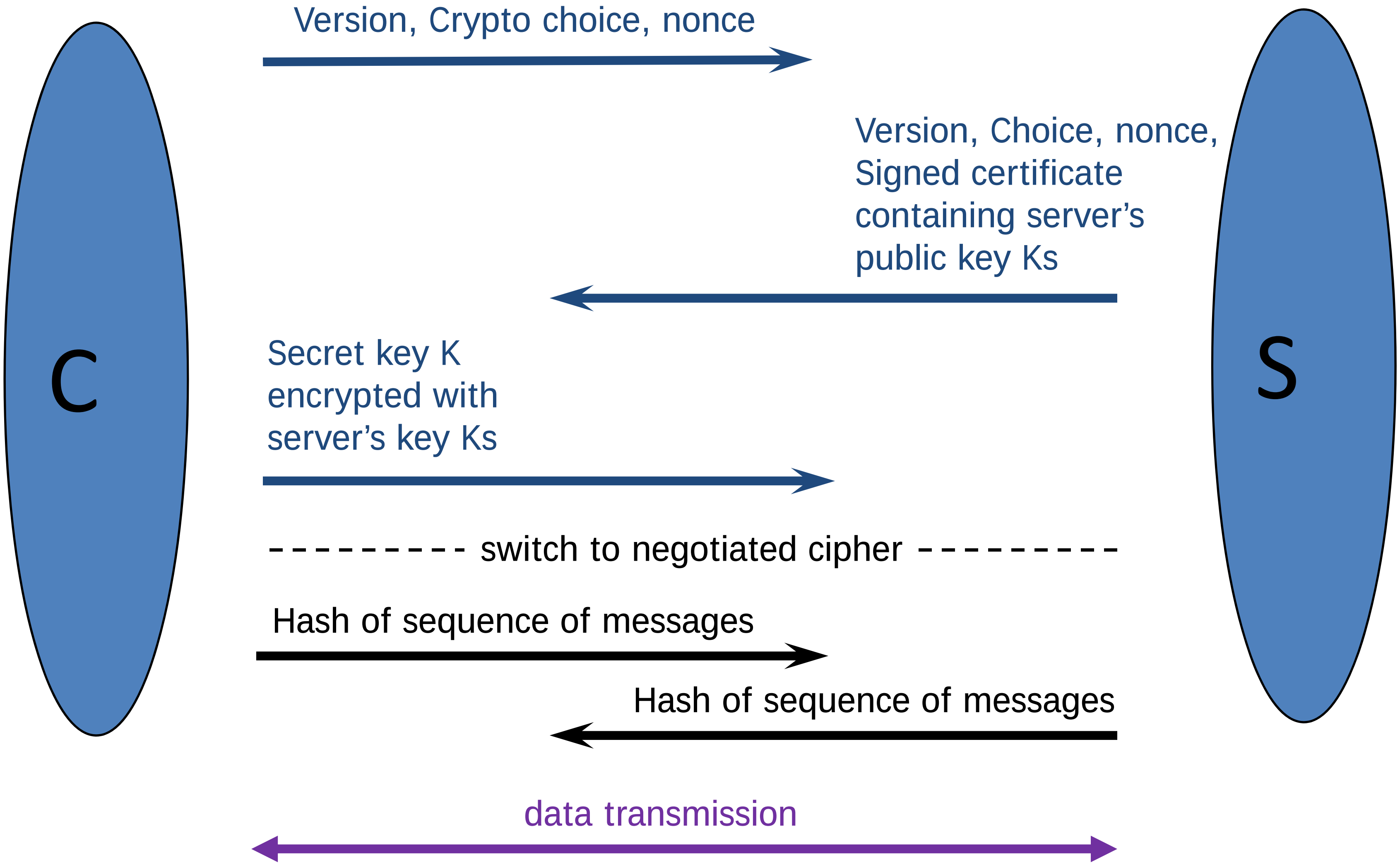
Packet Fragmentation Attack

- Firewall configuration
  - TCP port 23 is blocked but SMTP port 25 is allowed
- First packet
  - Fragmentation Offset = 0.
  - DF bit = 0 : "May Fragment"
  - MF bit = 1 : "More Fragments"
  - Destination Port = 25. TCP port 25 is allowed, so firewall allows packet
- Second packet
  - Fragmentation Offset = 1: second packet overwrites all but first 8 bits of the first packet
  - DF bit = 0 : "May Fragment"
  - MF bit = 0 : "Last Fragment."
  - Destination Port = 23. Normally be blocked, but sneaks by!
- What happens
  - Firewall ignores second packet “TCP header” because it is fragment of first
  - At host, packet reassembled and received at port 23

TCP Protocol Stack



Remember SSL/TLS



Version, Crypto choice, nonce

Version, Choice, nonce, Signed certificate containing server's public key Ks

Secret key K encrypted with server's key Ks

---------- switch to negotiated cipher ----------

Hash of sequence of messages

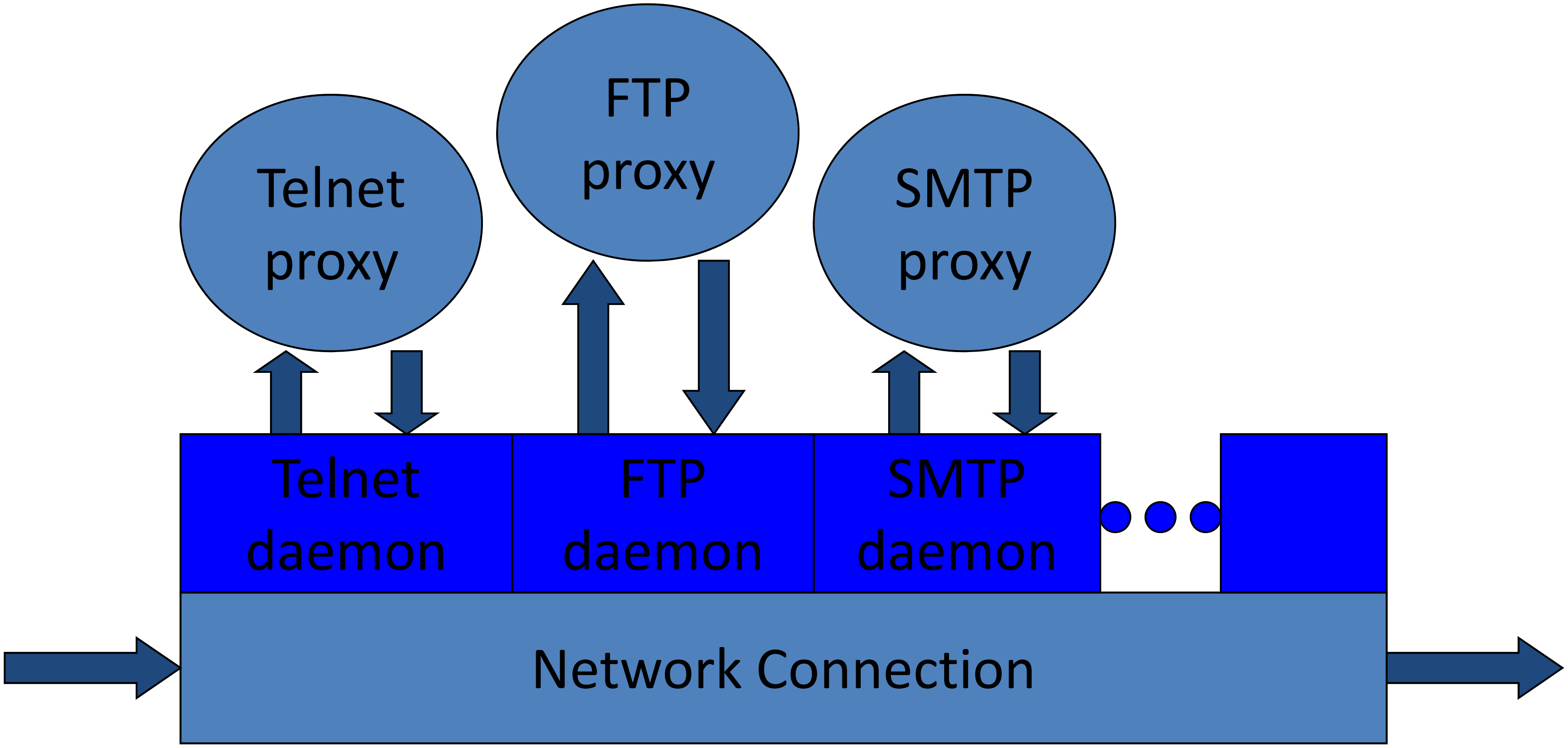
Hash of sequence of messages

data transmission

Proxying Firewall

- Application-level proxies
  - Tailored to http, ftp, smtp, etc.
  - Some protocols easier to proxy than others
- Policy embedded in proxy programs
  - Proxies filter incoming, outgoing packets
  - Reconstruct application-layer messages
  - Can filter specific application-layer commands, etc.
    - Example: only allow specific ftp commands
    - Other examples: ?
- Several network locations – see next slides

Firewall with application proxies



Daemon spawns proxy when communication detected …

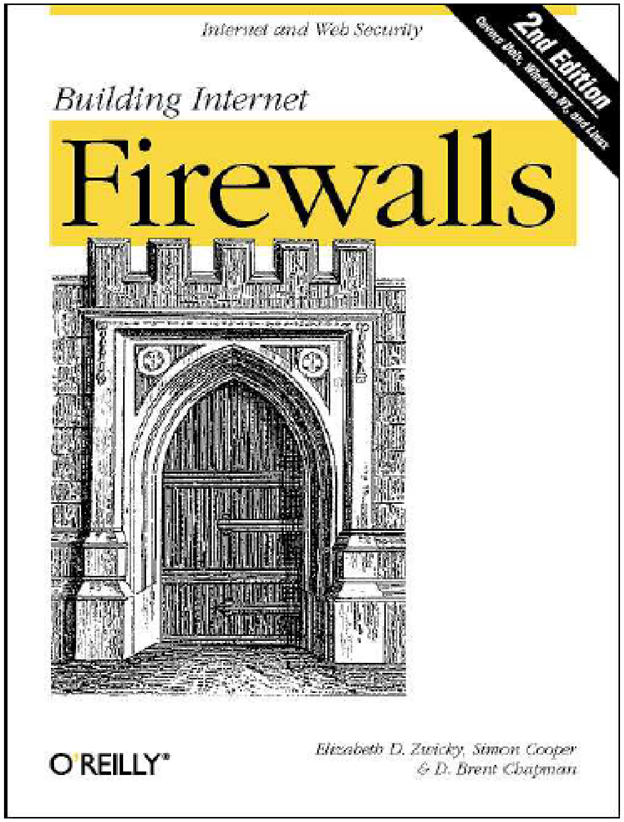
Application-level proxies

- Enforce policy for specific protocols
  - E.g., Virus scanning for SMTP
    - Need to understand MIME, encoding, Zip archives
  - Flexible approach, but may introduce network delays
- “Batch” protocols are natural to proxy
  - SMTP (E-Mail) NNTP (Net news)
  - DNS (Domain Name System) NTP (Network Time Protocol
- Must protect host running protocol stack
  - Disable all non-required services; keep it simple
  - Install/modify services you want
  - Run security audit to establish baseline
  - Be prepared for the system to be compromised

Web traffic scanning

- Intercept and proxy web traffic
  - Can be host-based
  - Usually at enterprise gateway
- Block known bad sites
- Block pages with known attacks
- Scan attachments
  - Usually traditional virus scanning methods

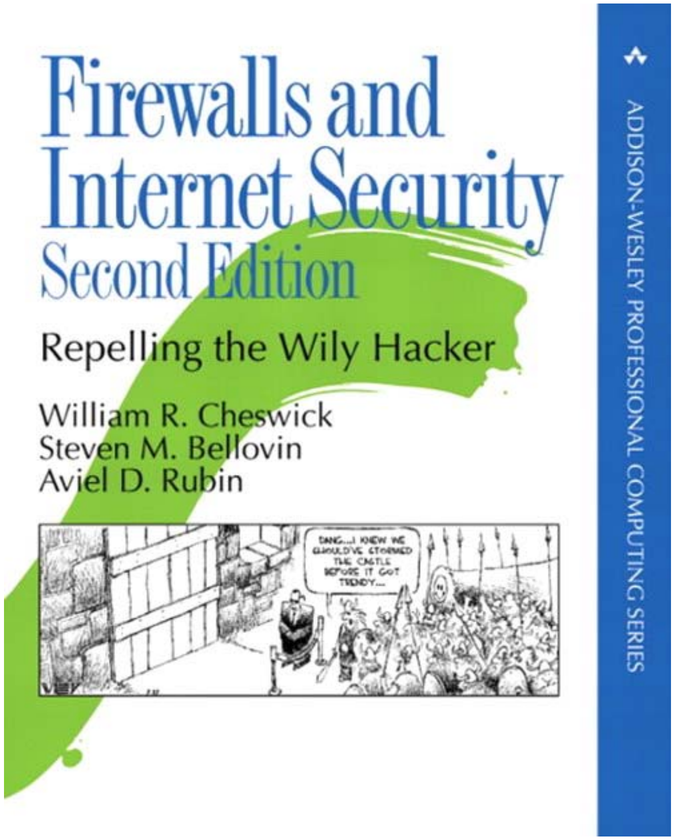
Firewall references


Internet and Web Security
2nd Edition
Covers Unix, Windows NT, and Linux
Building Internet
Firewalls
O'REILLY
Elizabeth D. Zwicky, Simon Cooper
& D. Brent Chapman

Elizabeth D. Zwicky

Simon Cooper

D. Brent Chapman

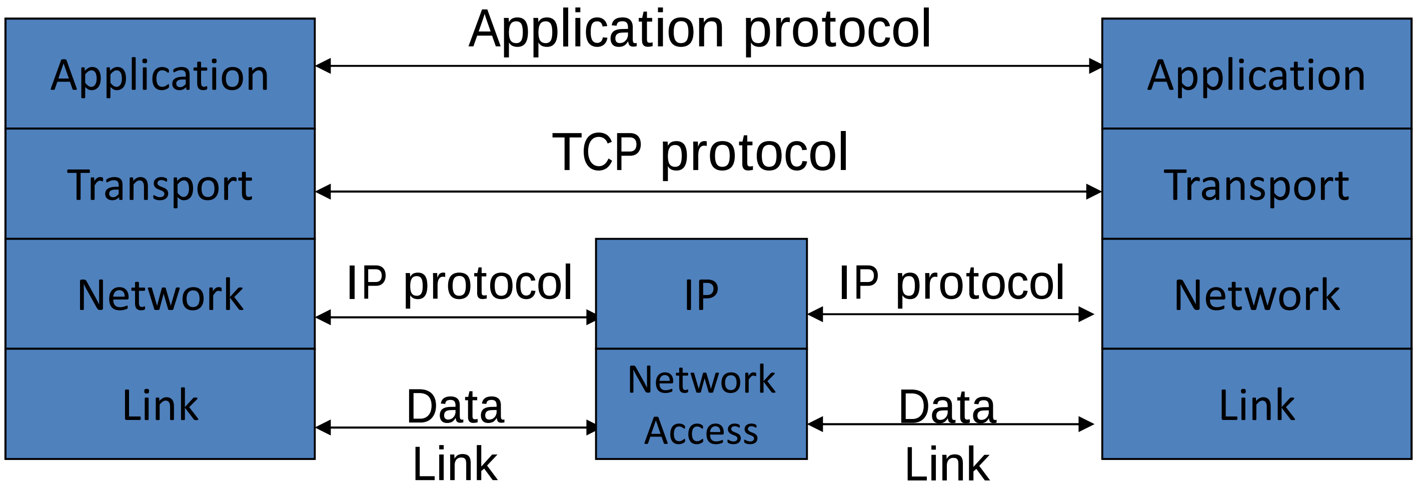

Firewalls and
Internet Security
Second Edition
Repelling the Wily Hacker
William R. Cheswick
Steven M. Bellovin
Aviel D. Rubin
DANG... I KNEW WE SHOULD'VE STORMED THE CASTLE BEFORE IT GOT TRENDY...
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

William R Cheswick

Steven M Bellovin

Aviel D Rubin

TCP Protocol Stack



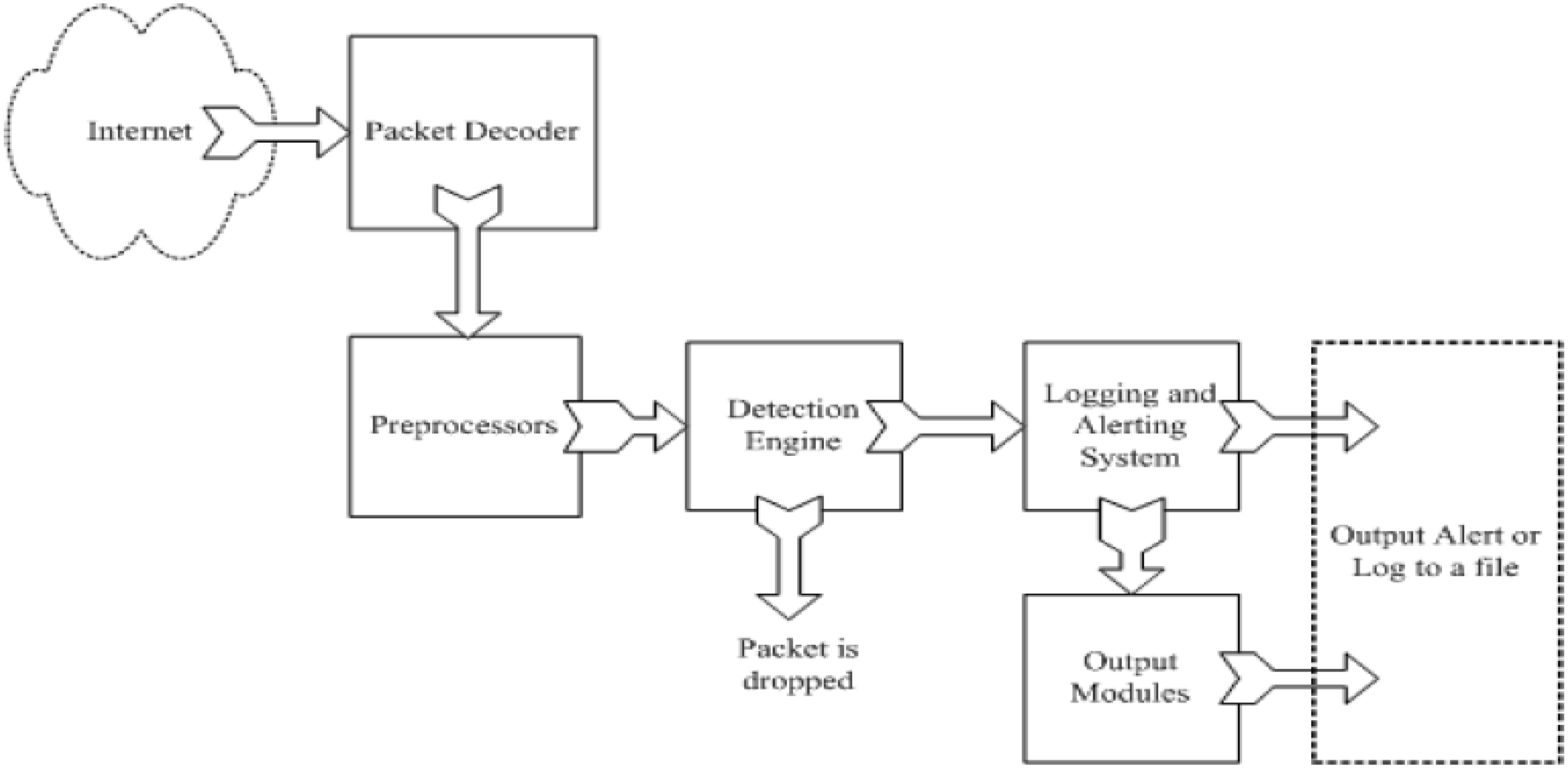
- Intrusion detection
- Infrastructure protocols
  - BGP
  - DNS

Intrusion detection

- Many intrusion detection systems
  - Close to 100 systems with current web pages
  - Network-based, host-based, or combination
- Two basic models
  - Misuse detection model
    - Maintain data on known attacks
    - Look for activity with corresponding signatures
  - Anomaly detection model
    - Try to figure out what is “normal”
    - Report anomalous behavior
- Fundamental problem: too many false alarms

Example: Snort

http://www.snort.org/

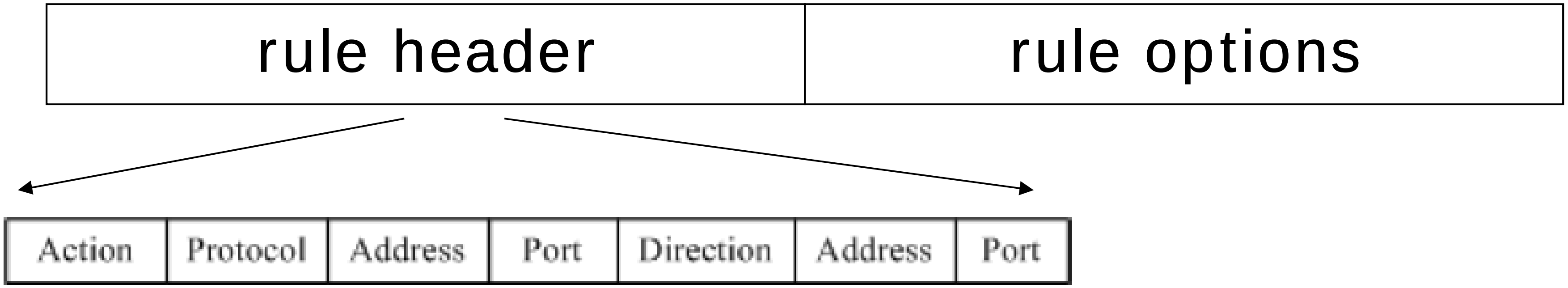


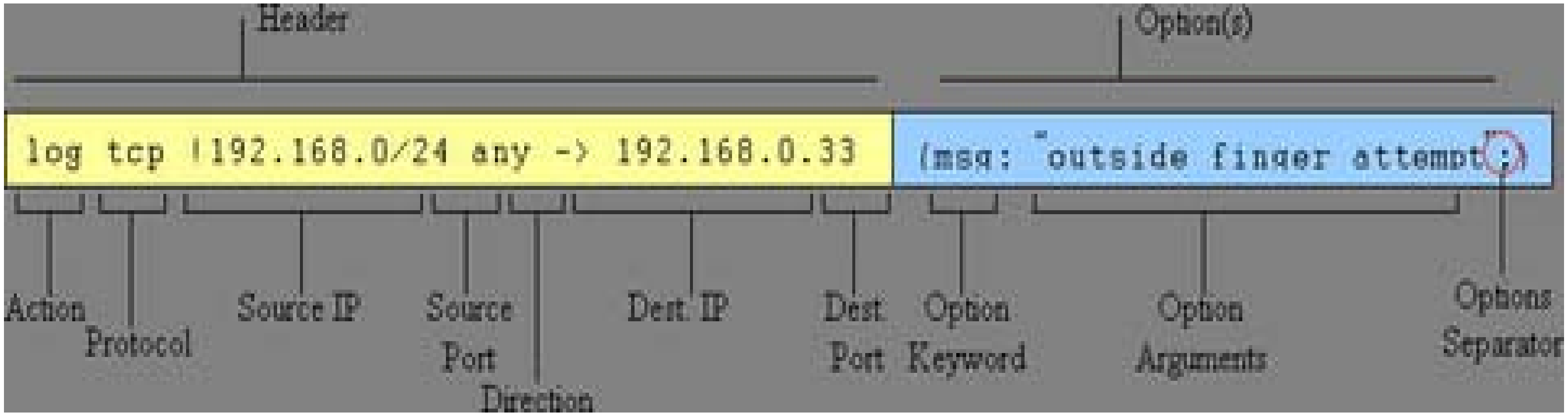
From: Rafeeq Ur Rehman, *Intrusion Detection Systems with Snort: Advanced IDS Techniques with Snort, Apache, MySQL, PHP, and ACID.*

Snort components

- Packet Decoder
  - input from Ethernet, SLIP, PPP…
- Preprocessor:
  - detect anomalies in packet headers
  - packet defragmentation
  - decode HTTP URI
  - reassemble TCP streams
- Detection Engine: applies rules to packets
- Logging and Alerting System
- Output Modules: alerts, log, other output

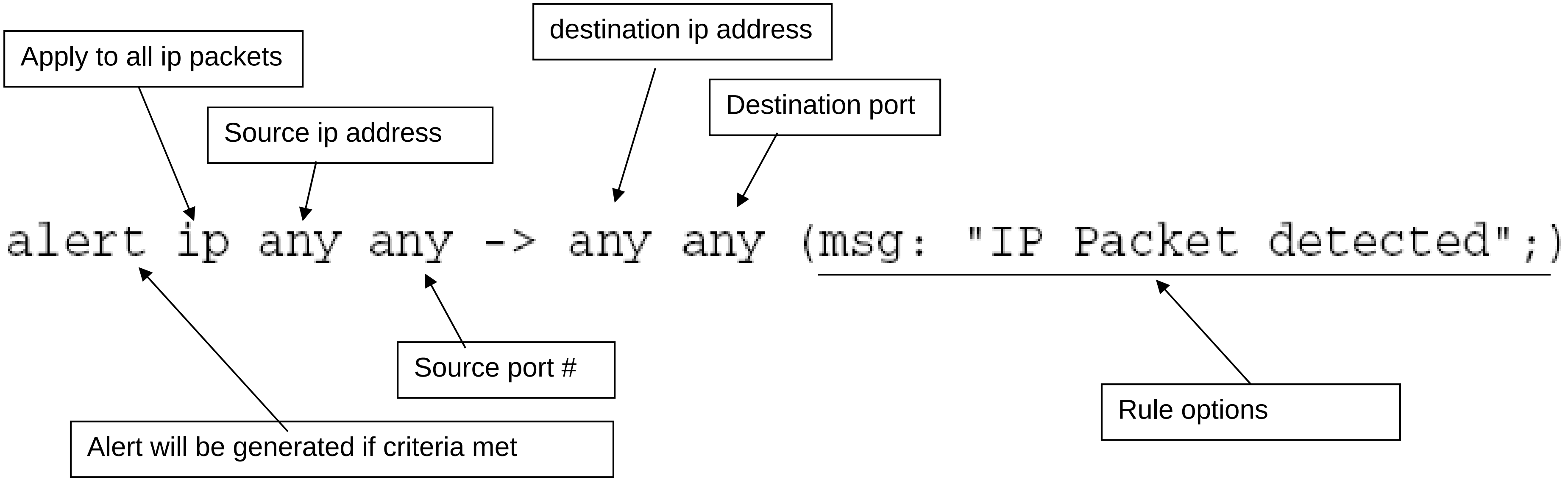
Snort detection rules





```
log tcp !192.168.0/24 any -> 192.168.0.33 (msg: "outside finger attempt";)
```

Additional examples



Apply to all ip packets

Source ip address

destination ip address

Destination port

```
alert ip any any -> any any (msg: "IP Packet detected";)
```

Alert will be generated if criteria met

Source port #

Rule options

```
alert tcp $TELNET_SERVERS 23 -> $EXTERNAL_NET any (msg:"TELNET
  Attempted SU from wrong group"; flow:
from_server,established; content:"to su root"; nocase;
  classtype:attempted-admin; sid:715; rev:6;)
```

Snort challenges

- Misuse detection – avoid known intrusions
  - Database size continues to grow
    - Snort version 2.3.2 had 2,600 rules
  - Snort spends 80% of time doing string match

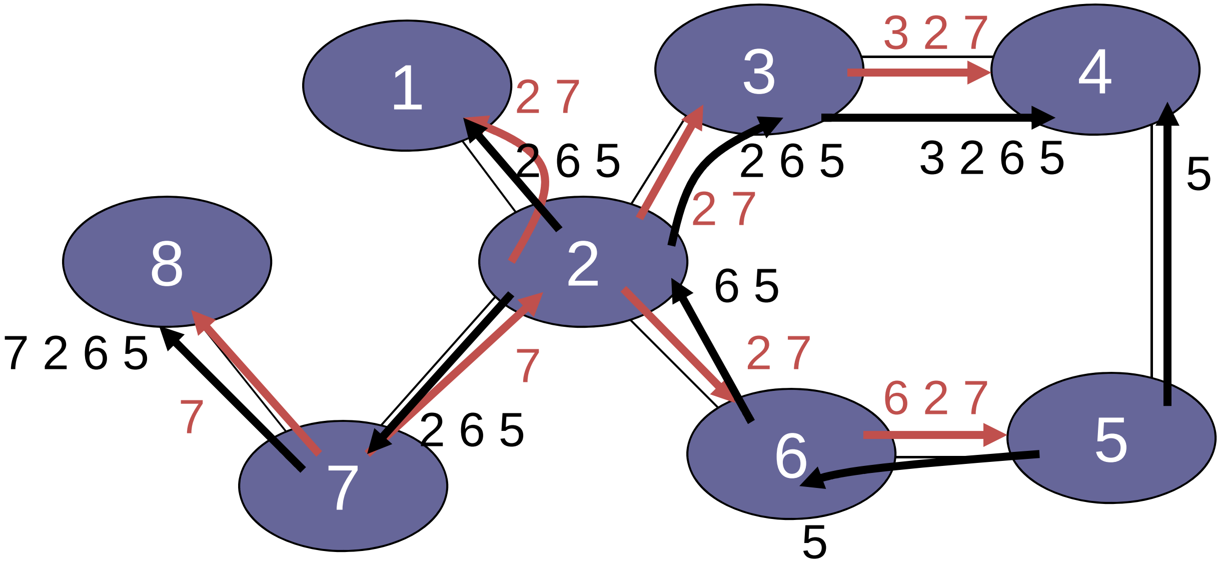
- Anomaly detection – identify new attacks
  - Probability of detection is low

Difficulties in anomaly detection

- Lack of training data
  - Lots of “normal” network, system call data
  - Little data containing realistic attacks, anomalies
- Data drift
  - Statistical methods detect changes in behavior
  - Attacker can attack gradually and incrementally
- Main characteristics not well understood
  - By many measures, attack may be within bounds of “normal” range of activities
- False identifications are very costly
  - Sys Admin spend many hours examining evidence

INFRASTRUCTURE PROTOCOLS: BGP, DNS

BGP example



- Transit: 2 provides transit for 7
- Algorithm seems to work OK in practice
  - BGP is does not respond well to frequent node outages

Figure: D. Wetherall

BGP Security Issues

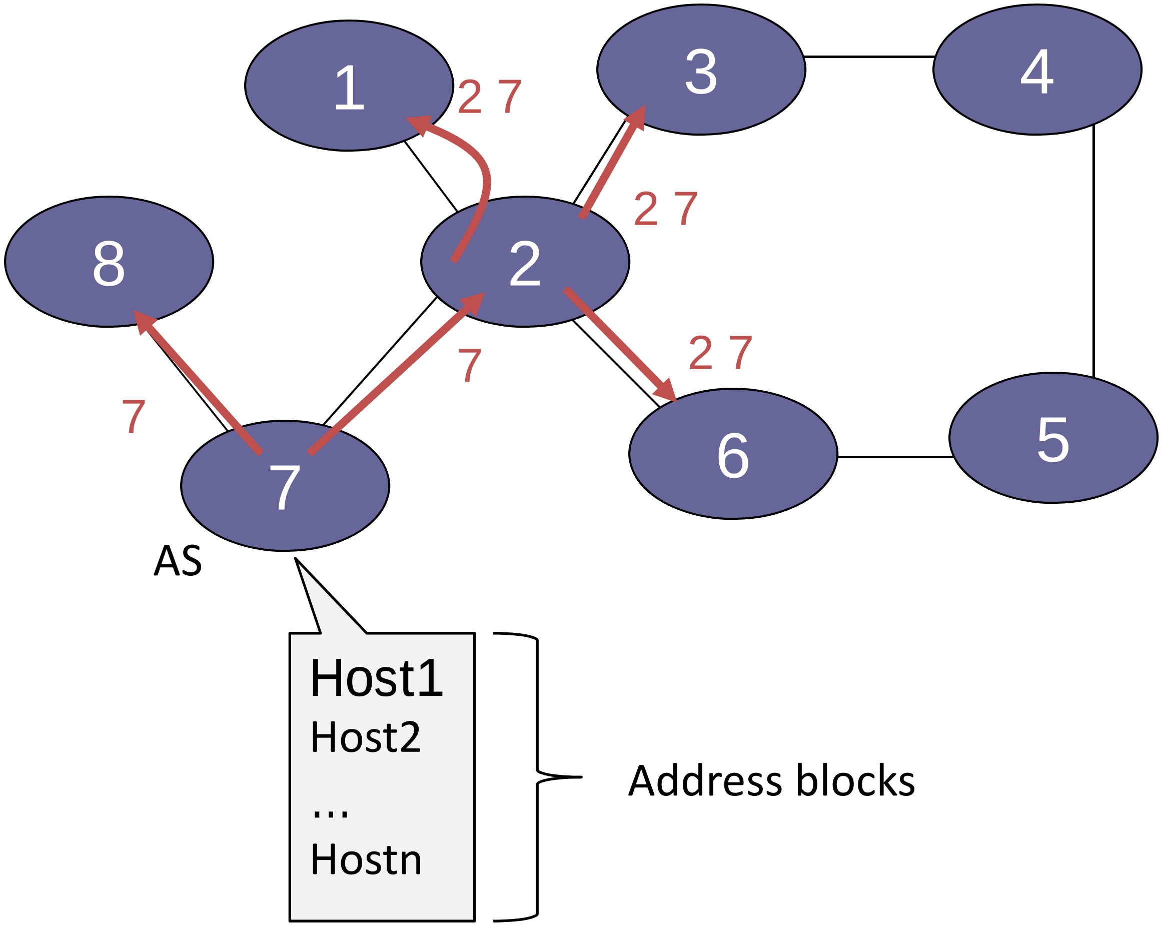
- BGP is used for all inter-ISP routing
- Benign configuration errors affect about 1% of all routing table entries at any time
- Highly vulnerable to human errors, malicious attacks
  - Actual routing policies can be very complicated
- MD5 MAC is rarely used, perhaps due to lack of automated key management, addresses only one class of attacks

S-BGP Design Overview

- IPsec: secure point-to-point router communication
- Public Key Infrastructure: authorization for all S-BGP entities
- Attestations: digitally-signed authorizations
  - Address: authorization to advertise specified address blocks
  - Route: Validation of UPDATEs based on a new path attribute, using PKI certificates and attestations
- Repositories for distribution of certificates, CRLs, and address attestations
- Tools for ISPs to manage address attestations, process certificates & CRLs, etc.

Slide: Steve Kent

BGP example



Address Attestation

- Indicates that the final AS listed in the UPDATE is authorized by the owner of those address blocks to advertise the address blocks in the UPDATE
- Includes identification of:
  - owner's certificate
  - AS to be advertising the address blocks
  - address blocks
  - expiration date
- Digitally signed by owner of the address blocks
- Used to protect BGP from erroneous UPDATEs (authenticated but misbehaving or misconfigured BGP speakers)

Route Attestation

- Indicates that the speaker or its AS authorizes the listener's AS to use the route in the UPDATE
- Includes identification of:
  - AS's or BGP speaker's certificate issued by owner of the AS
  - the address blocks and the list of ASes in the UPDATE
  - the neighbor
  - expiration date
- Digitally signed by owner of the AS (or BGP speaker) distributing the UPDATE, traceable to the IANA ...
- Used to protect BGP from erroneous UPDATEs (authenticated but misbehaving or misconfigured BGP speakers)

Validating a Route

- To validate a route from AS_n, AS_{n+1} needs:
  - address attestation from each organization owning an address block(s) in the NLRI
  - address allocation certificate from each organization owning address blocks in the NLRI
  - route attestation from every AS along the path (AS_1 to AS_n), where the route attestation for AS_k specifies the NLRI and the path up to that point (AS_1 through AS_{k+1})
  - certificate for each AS or router along path (AS_1 to AS_n) to check signatures on the route attestations
  - and, of course, all the relevant CRLs must have been checked

Slide: Kent et al.

INFRASTRUCTURE PROTOCOLS: BGP, DNS

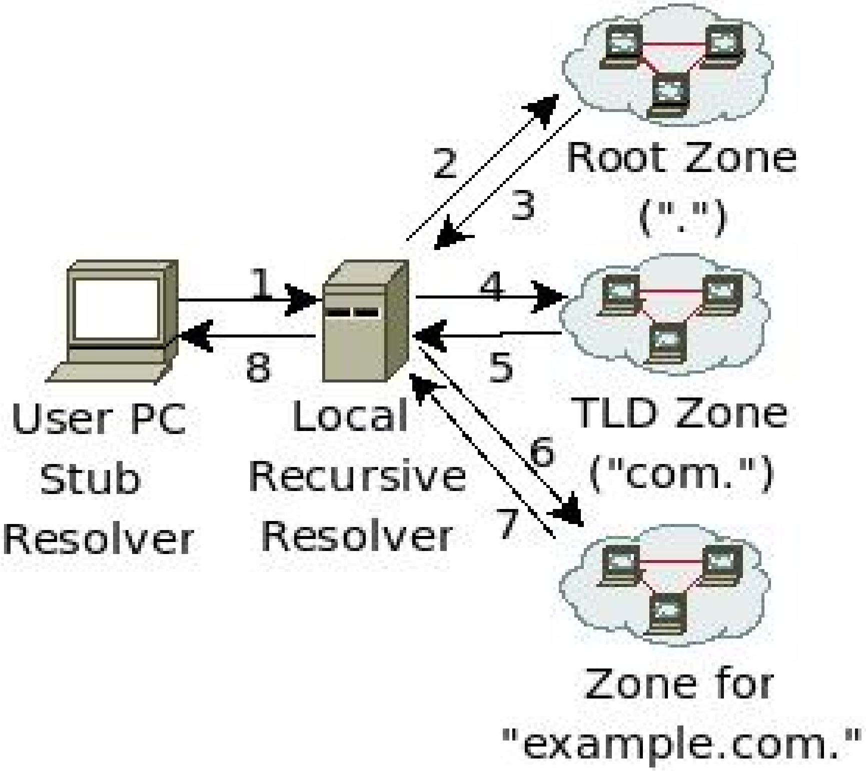
⬆

Recall: DNS Lookup

Query: "www.example.com A?"

Reply	Resource Records in Reply
3	"com. NS a.gtld.net" "a.gtld.net A 192.5.6.30"
5	"example.com. NS a.iana.net" "a.iana.net A 192.0.34.43"
7	"www.example.com A 1.2.3.4"
8	"www.example.com A 1.2.3.4"

se



DNS is Insecure

- Packets sent over UDP, < 512 bytes
- 16-bit TXID, UDP Src port are only “security”
- Resolver accepts packet if above match
- Packet from whom? Was it manipulated?

- Cache poisoning
  - Attacker forges record at resolver
  - Forged record cached, attacks future lookups
  - Kaminsky (BH USA08)
    - Attacks delegations with “birthday problem”

DNSSEC Goal

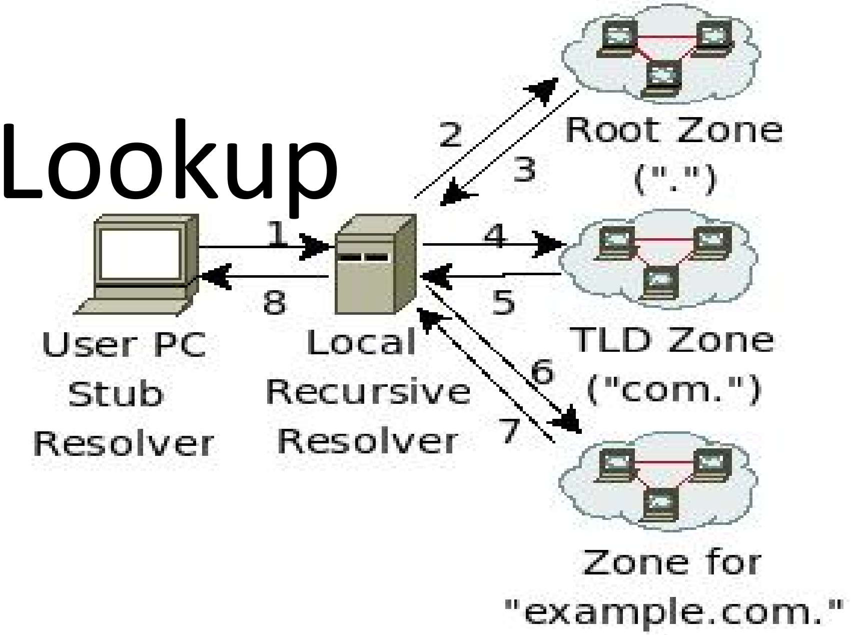
"The Domain Name System (DNS) security extensions provide origin authentication and integrity assurance services for DNS data, including mechanisms for authenticated denial of existence of DNS data."

-RFC 4033

DNSSEC

- Basically no change to packet format
  - Goal is security of DNS data, not channel security
- New Resource Records (RRs)
  - RRSIG : signature of RR by private zone key
  - DNSKEY : public zone key
  - DS : crypto digest of child zone key
  - NSEC / NSEC3 authenticated denial of existence
- Lookup referral chain (unsigned)
- Origin attestation chain (PKI) (signed)
  - Start at pre-configured trust anchors
    - DS/DNSKEY of zone (should include root)
  - DS → DNSKEY → DS forms a link

DNSSEC Lookup



Query: "www.example.com A?"

Reply	RRs in DNS Reply	Added by DNSSEC
3	"com. NS a.gtld.net" "a.gtld.net A 192.5.6.30"	"com. DS" "RRSIG(DS) by ."
5	"example.com. NS a.iana.net" "a.iana.net A 192.0.34.43"	"com. DNSKEY" "RRSIG(DNSKEY) by com." "example.com. DS" "RRSIG(DS) by com."
7	"www.example.com A 1.2.3.4"	"example.com DNSKEY" "RRSIG(DNSKEY) by example.com." "RRSIG(A) by example.com."
8	"www.example.com A 1.2.3.4"	Last Hop?

Authenticated Denial-of-Existence

- Most DNS lookups result in denial-of-existence
- NSEC (Next SECure)
  - Lists all extant RRs associated with an owner name
  - Points to next owner name with extant RR
  - Easy zone enumeration
- NSEC3
  - Hashes owner names
  - Opt-out bit for TLDs to support incremental adoption
    - For TLD type zones to support incremental adoption
    - Non-DNSSEC children not in NSEC3 chain

Insecure Sub-Namespace

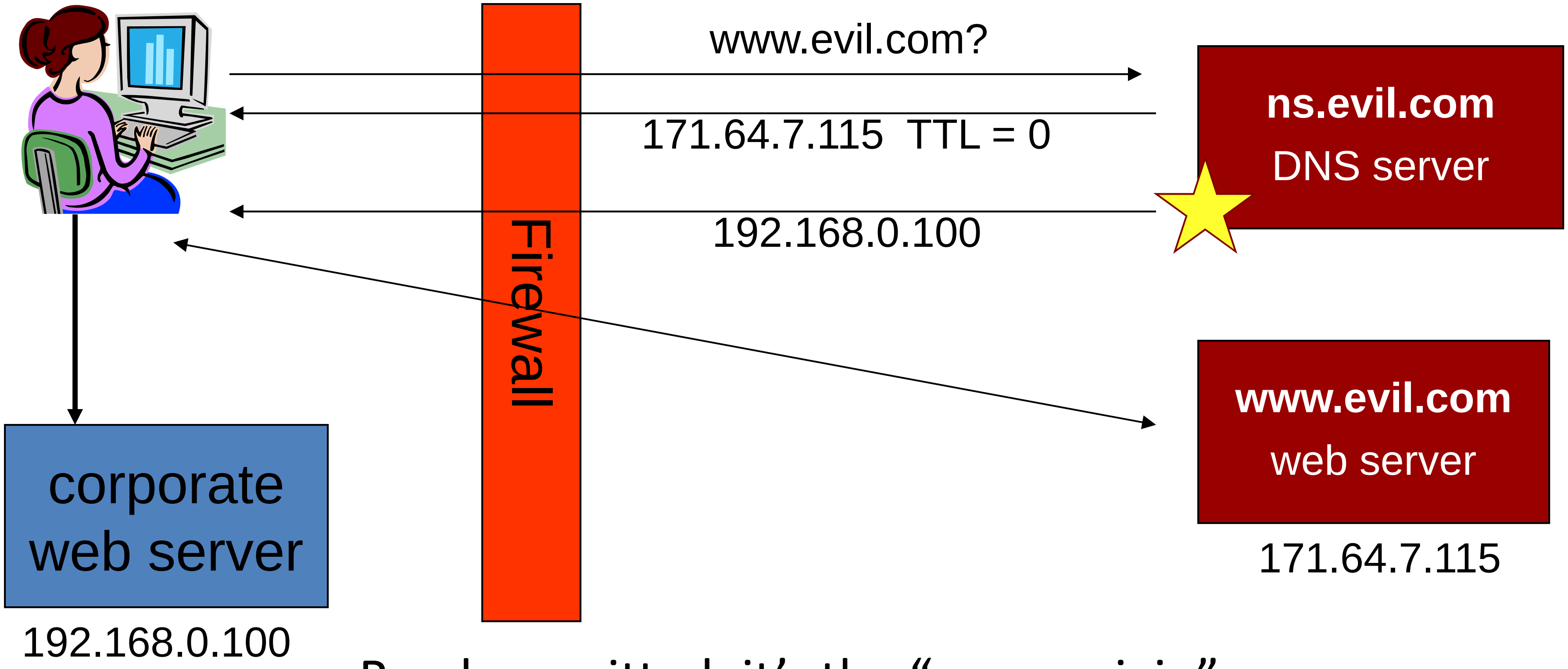
- NSEC3 Opt-out
  - "Does not assert the existence or non-existence of the insecure delegations that it may cover" (RFC 5155)
  - Only thing asserting this is insecure glue records
- Property: Possible to insert bogus pre-pended name into otherwise secure zone. (RFC 5155)
- Insecure delegation from secure zone
  - Spoofs possible for resultant lookup results
- Acceptable for TLD, bad for enterprises

[DWF’96, R’01]

DNS Rebinding Attack

<iframe src="**http://www.evil.com**">

DNSSEC cannot
stop this attack



Read permitted: it’s the “same origin”

DNS Rebinding Defenses

- Browser mitigation: DNS Pinning
  - Refuse to switch to a new IP
  - Interacts poorly with proxies, VPN, dynamic DNS, …
  - Not consistently implemented in any browser
- Server-side defenses
  - Check Host header for unrecognized domains
  - Authenticate users with something other than IP
- Firewall defenses
  - External names can’t resolve to internal addresses
  - Protects browsers inside the organization

Summary

- Network protocol security
  - Wireless security – 802.11i/WPA2
  - IPSEC
  - BGP instability and S-BGP
  - DNSSEC, DNS rebinding
- Standard network perimeter defenses
  - Firewall
    - Packet filter (stateless, stateful), Application layer proxies
  - Traffic shaping
  - Intrusion detection
    - Anomaly and misuse detection

Module 10: HTTPS

Goals for this lecture

Brief overview of HTTPS:

- How the SSL/TLS protocol works (very briefly)
- How to use HTTPS

Integrating HTTPS into the browser

- Lots of user interface problems to watch for

Threat Model: Network Attacker

Network Attacker:

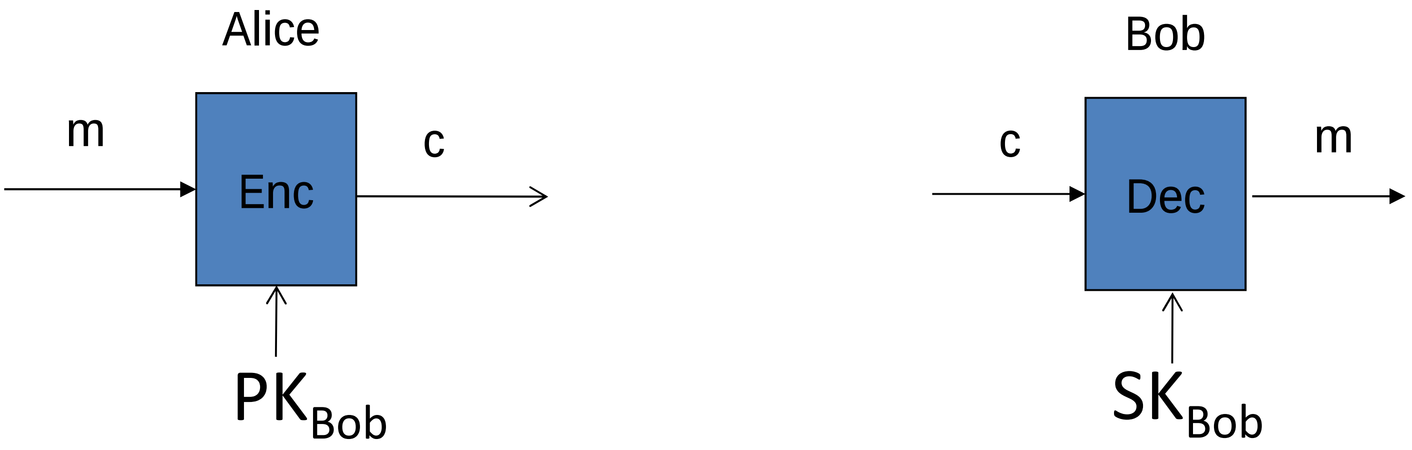
- Controls network infrastructure: Routers, DNS
- Eavesdrops, injects, blocks, and modifies packets

Examples:

- Wireless network at Internet Café
- Internet access at hotels (untrusted ISP)

SSL/TLS overview

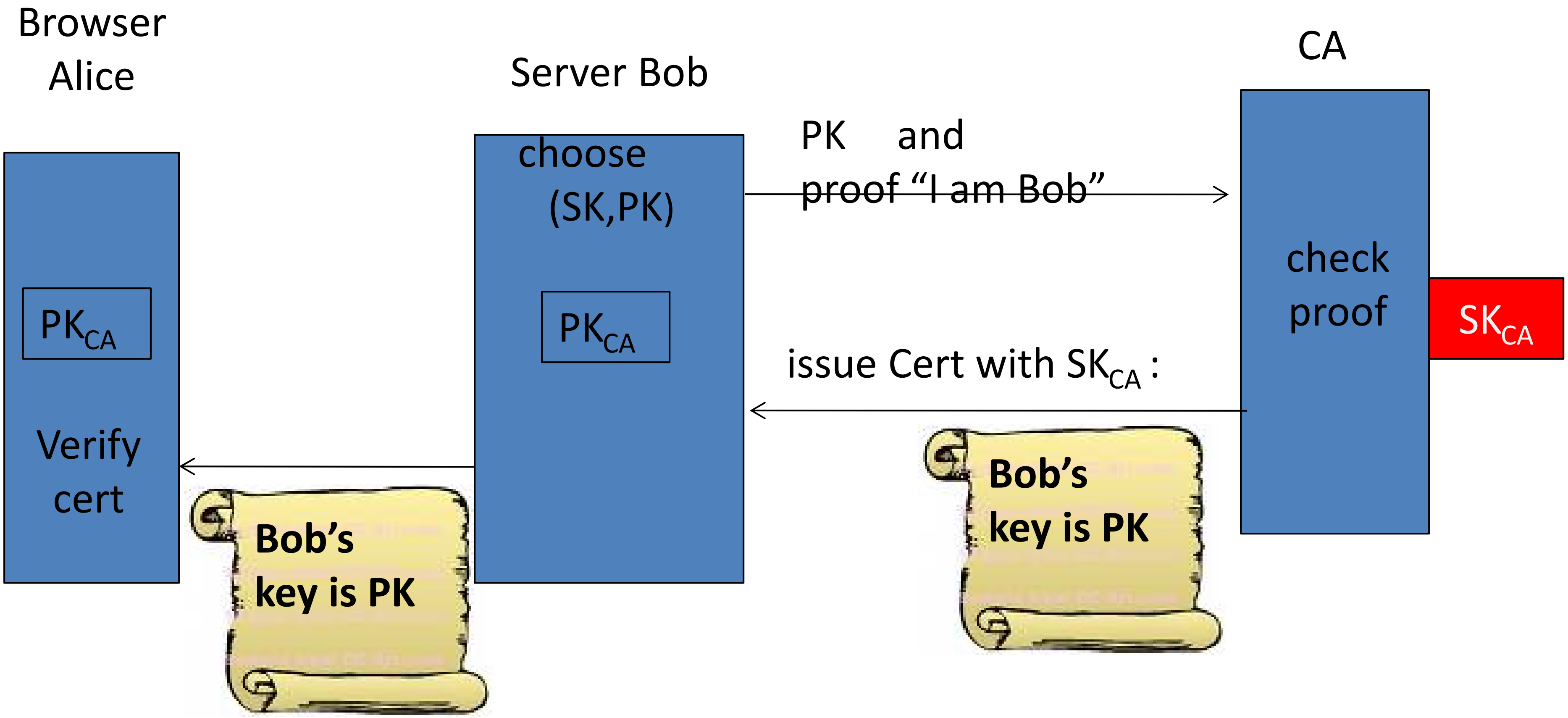
Public-key encryption:



- Bob generates (SK_{Bob} , PK_{Bob})
- Alice: using PK_{Bob} encrypts messages and only Bob can decrypt

Certificates

How does Alice (browser) obtain PK_{Bob} ?



Browser Alice

Server Bob

CA

choose (SK,PK)

PK_{CA}

PK_{CA}

PK and proof "I am Bob"

check proof

SK_{CA}

issue Cert with SK_{CA} :

Bob's key is PK

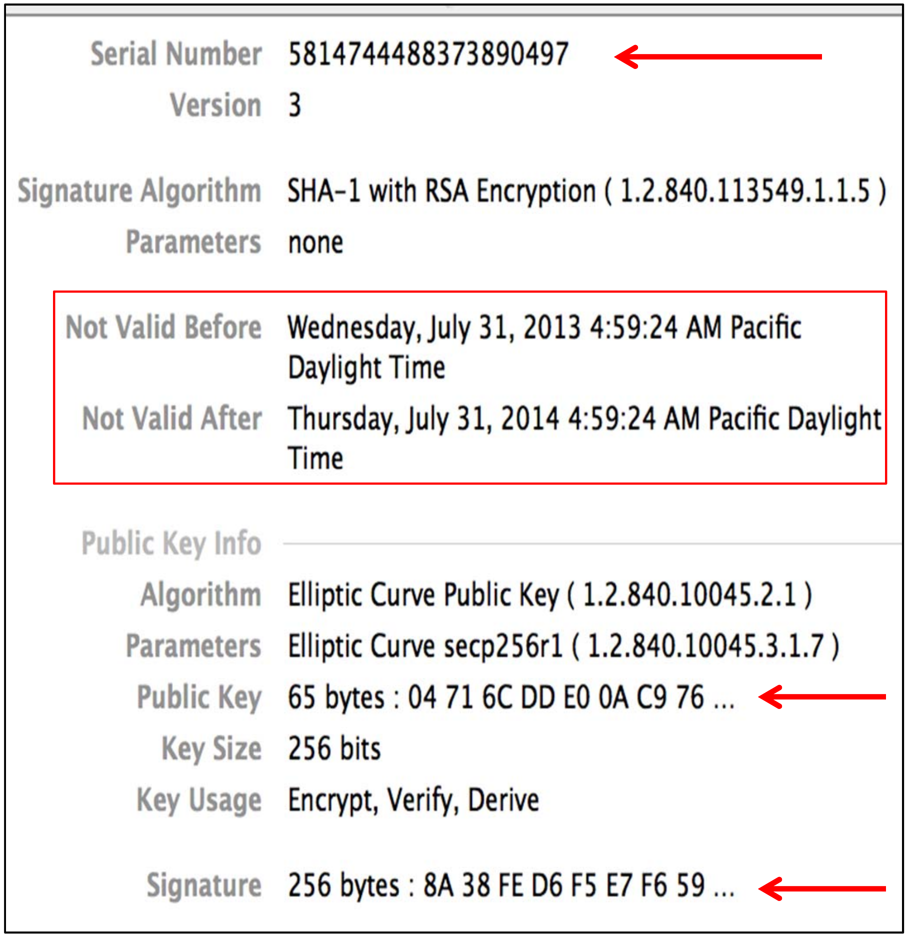
Verify cert

Bob's key is PK

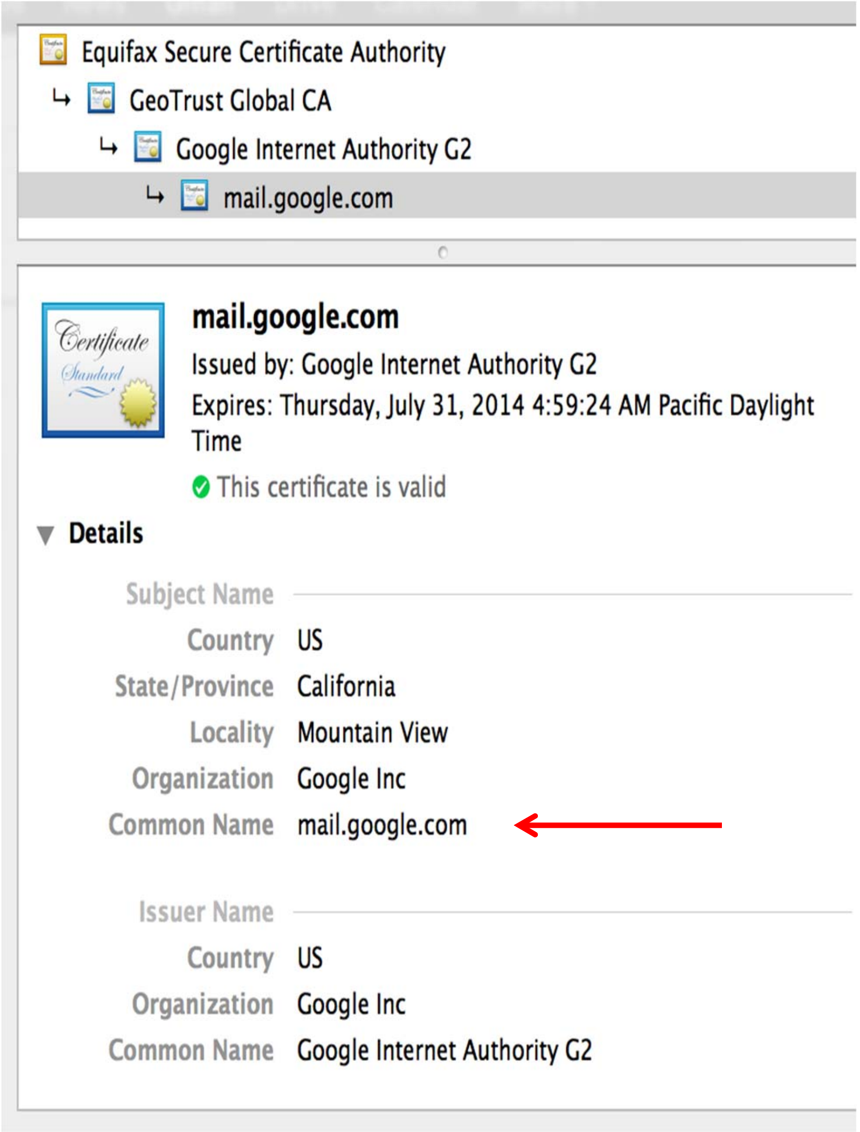
Bob uses Cert for an extended period (e.g. one year)

Certificates: example

Important fields:



Serial Number	5814744488373890497
Version	3
Signature Algorithm	SHA-1 with RSA Encryption (1.2.840.113549.1.1.5)
Parameters	none
Not Valid Before	Wednesday, July 31, 2013 4:59:24 AM Pacific Daylight Time
Not Valid After	Thursday, July 31, 2014 4:59:24 AM Pacific Daylight Time
Public Key Info	
Algorithm	Elliptic Curve Public Key (1.2.840.10045.2.1)
Parameters	Elliptic Curve secp256r1 (1.2.840.10045.3.1.7)
Public Key	65 bytes : 04 71 6C DD E0 0A C9 76 ...
Key Size	256 bits
Key Usage	Encrypt, Verify, Derive
Signature	256 bytes : 8A 38 FE D6 F5 E7 F6 59 ...



Equifax Secure Certificate Authority
↳ GeoTrust Global CA
↳ Google Internet Authority G2
↳ mail.google.com

mail.google.com

Issued by: Google Internet Authority G2

Expires: Thursday, July 31, 2014 4:59:24 AM Pacific Daylight Time

✓ This certificate is valid

▼ **Details**

Subject Name	
Country	US
State/Province	California
Locality	Mountain View
Organization	Google Inc
Common Name	mail.google.com
Issuer Name	
Country	US
Organization	Google Inc
Common Name	Google Internet Authority G2

Certificates on the web

Subject’s CommonName can be:

- An explicit name, e.g. cs.stanford.edu , or
- A wildcard cert, e.g. *.stanford.edu or cs*.stanford.edu

matching rules:

“*” must occur in leftmost component, does not match “.”

example: *.a.com matches x.a.com but not y.x.a.com

(as in RFC 2818: “HTTPS over TLS”)

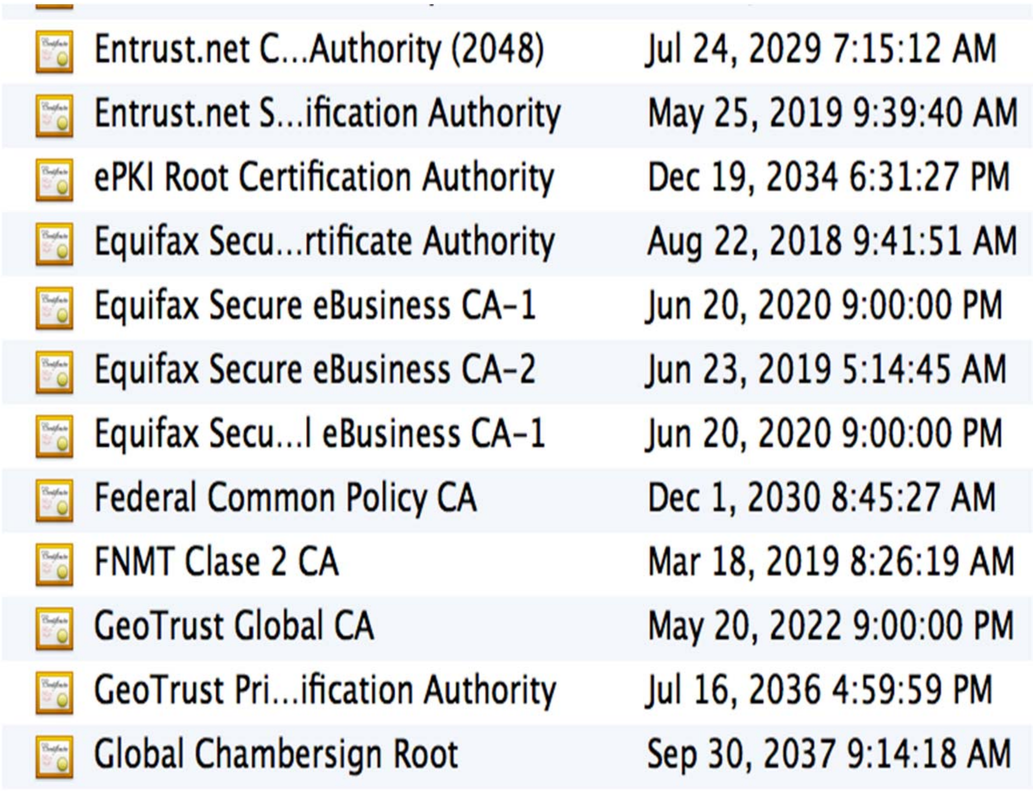
Certificate Authorities

Browsers accept certificates from a large number of CAs

Top level CAs ≈ 60

Intermediate CAs ≈ 1200

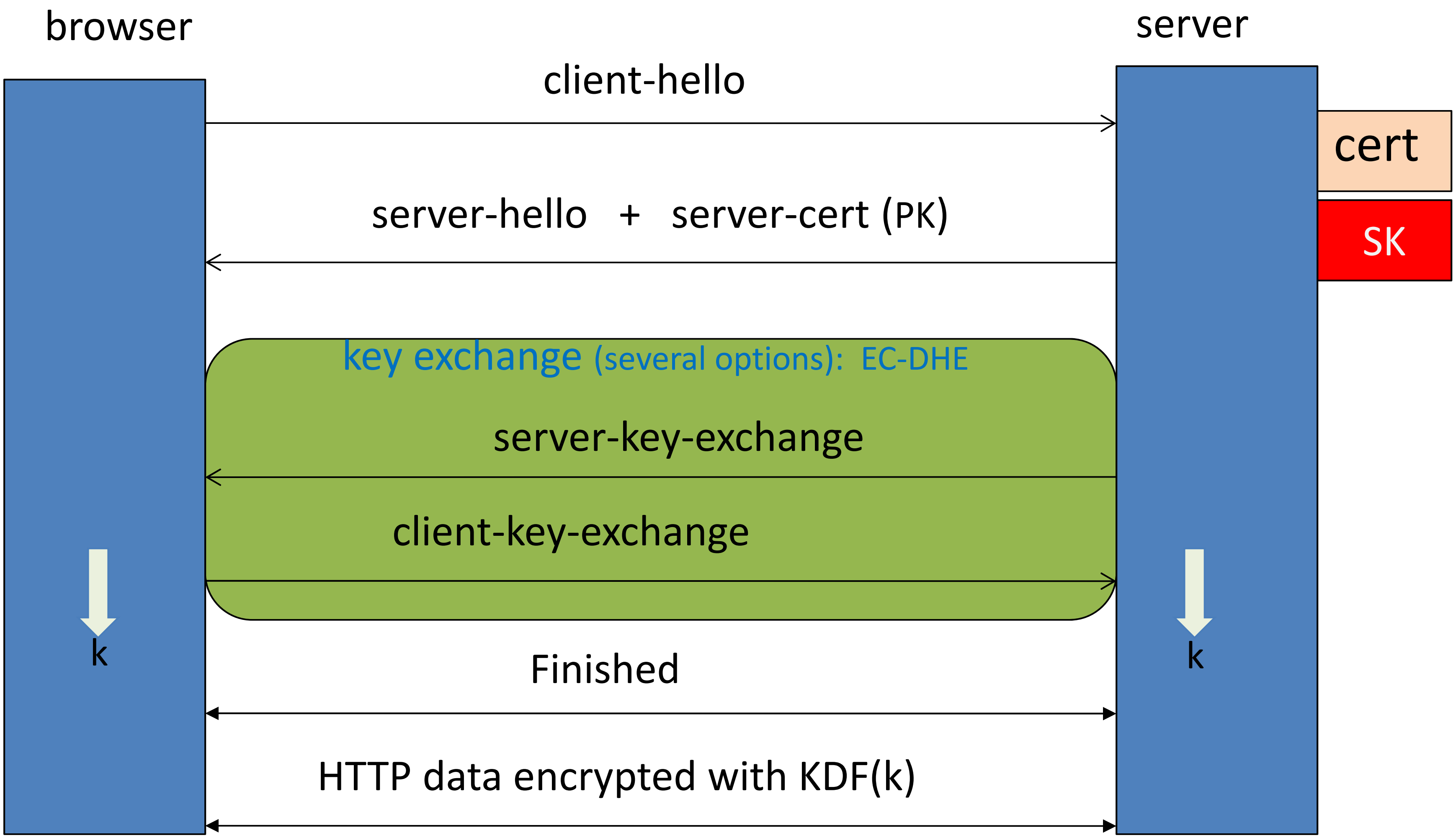
⋮



Entrust.net C...Authority (2048)	Jul 24, 2029 7:15:12 AM
Entrust.net S...ification Authority	May 25, 2019 9:39:40 AM
ePKI Root Certification Authority	Dec 19, 2034 6:31:27 PM
Equifax Secu...rtificate Authority	Aug 22, 2018 9:41:51 AM
Equifax Secure eBusiness CA-1	Jun 20, 2020 9:00:00 PM
Equifax Secure eBusiness CA-2	Jun 23, 2019 5:14:45 AM
Equifax Secu...l eBusiness CA-1	Jun 20, 2020 9:00:00 PM
Federal Common Policy CA	Dec 1, 2030 8:45:27 AM
FNMT Clase 2 CA	Mar 18, 2019 8:26:19 AM
GeoTrust Global CA	May 20, 2022 9:00:00 PM
GeoTrust Pri...ification Authority	Jul 16, 2036 4:59:59 PM
Global Chambersign Root	Sep 30, 2037 9:14:18 AM

⋮

Brief overview of SSL/TLS


browser
server
client-hello
cert
SK
server-hello + server-cert (PK)
key exchange (several options): EC-DHE
server-key-exchange
client-key-exchange
k
k
Finished
HTTP data encrypted with KDF(k)

Most common: server authentication only

Integrating SSL/TLS with HTTP: HTTPS

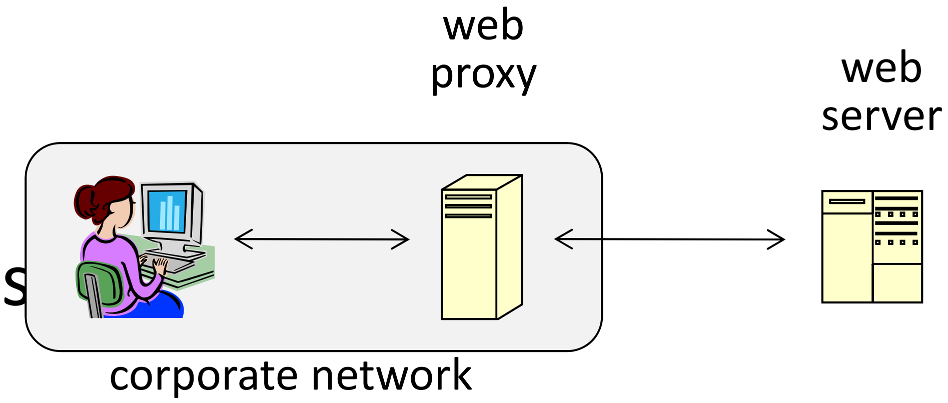
Two complications

Web proxies

solution: browser sends

CONNECT domain-name

before client-hello



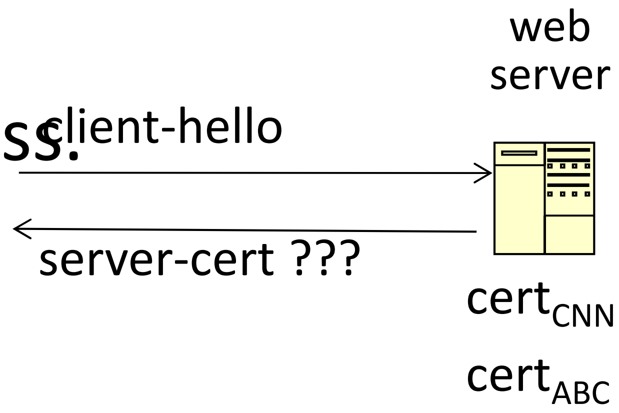
Virtual hosting:

two sites hosted at same IP address.

solution in TLS 1.1: SNI (June 2003)

client_hello_extension: server_name=cnn.com

implemented since FF2 and IE7 (vista)



Why is HTTPS not used for all web traffic?

- Crypto slows down web servers (but not by much if done right)

- Some ad-networks do not support HTTPS (2015 stats: 20%)
  - Reduced revenue for publishers

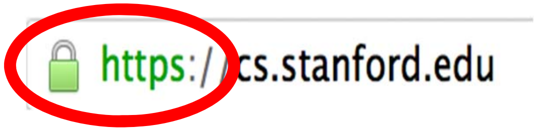
- Incompatible with virtual hosting (older browsers)

  March 2015: IE6 ≈ 1% (ie6countdown.com)

Aug 2014: Google boosts ranking of sites supporting HTTPS

HTTPS in the Browser

The lock icon: SSL indicator



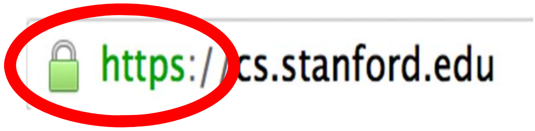
https://cs.stanford.edu

<u>Intended goal</u>:

- Provide user with identity of page origin
- Indicate to user that page contents were not viewed or modified by a **network attacker**

<u>In reality</u>: many problems (next few slides)

When is the (basic) lock icon displayed



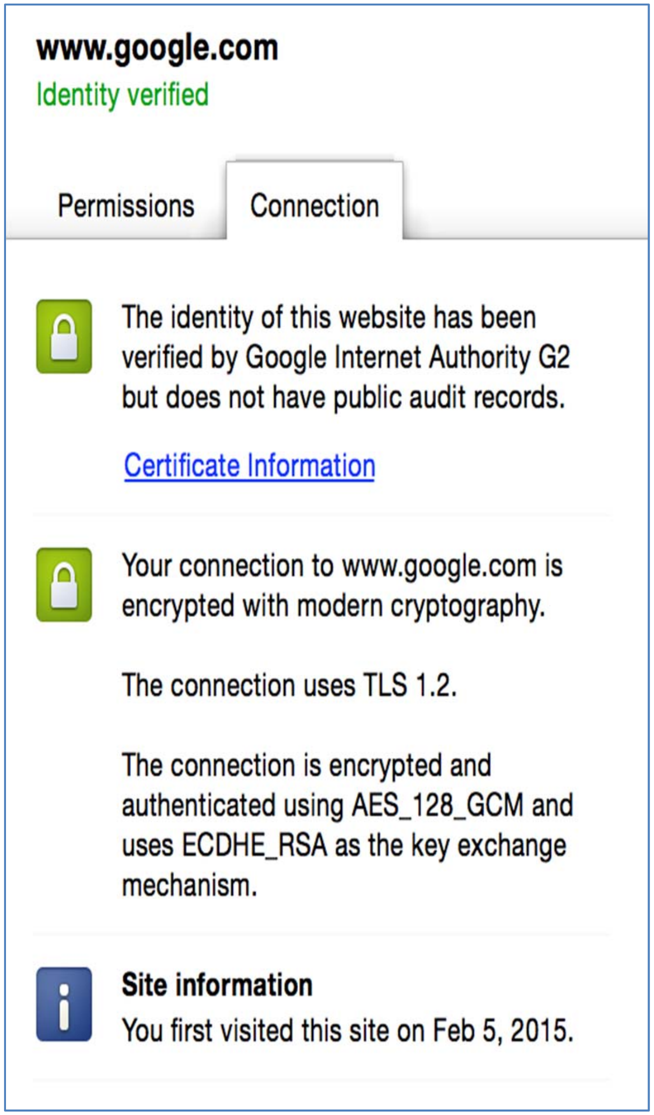
https://cs.stanford.edu

All elements on the page fetched using HTTPS

For all elements:

- HTTPS cert issued by a CA trusted by browser
- HTTPS cert is valid (e.g. not expired)
- CommonName in cert matches domain in URL

The lock UI: help users authenticate site



www.google.com

Identity verified

The identity of this website has been verified by Google Internet Authority G2 but does not have public audit records.

Certificate Information

Your connection to www.google.com is encrypted with modern cryptography.

The connection uses TLS 1.2.

The connection is encrypted and authenticated using AES_128_GCM and uses ECDHE_RSA as the key exchange mechanism.

Site information

You first visited this site on Feb 5, 2015.

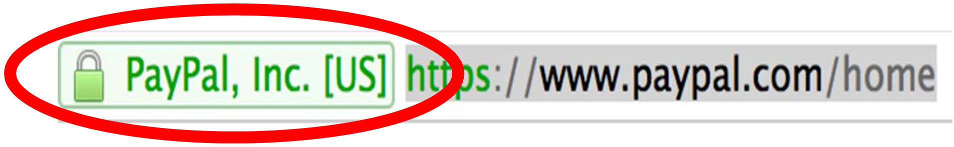
uninformative

The lock UI: Extended Validation Certs

Harder to obtain than regular certs

- requires human at CA to approve cert request
- no wildcard certs (e.g. *.stanford.edu)

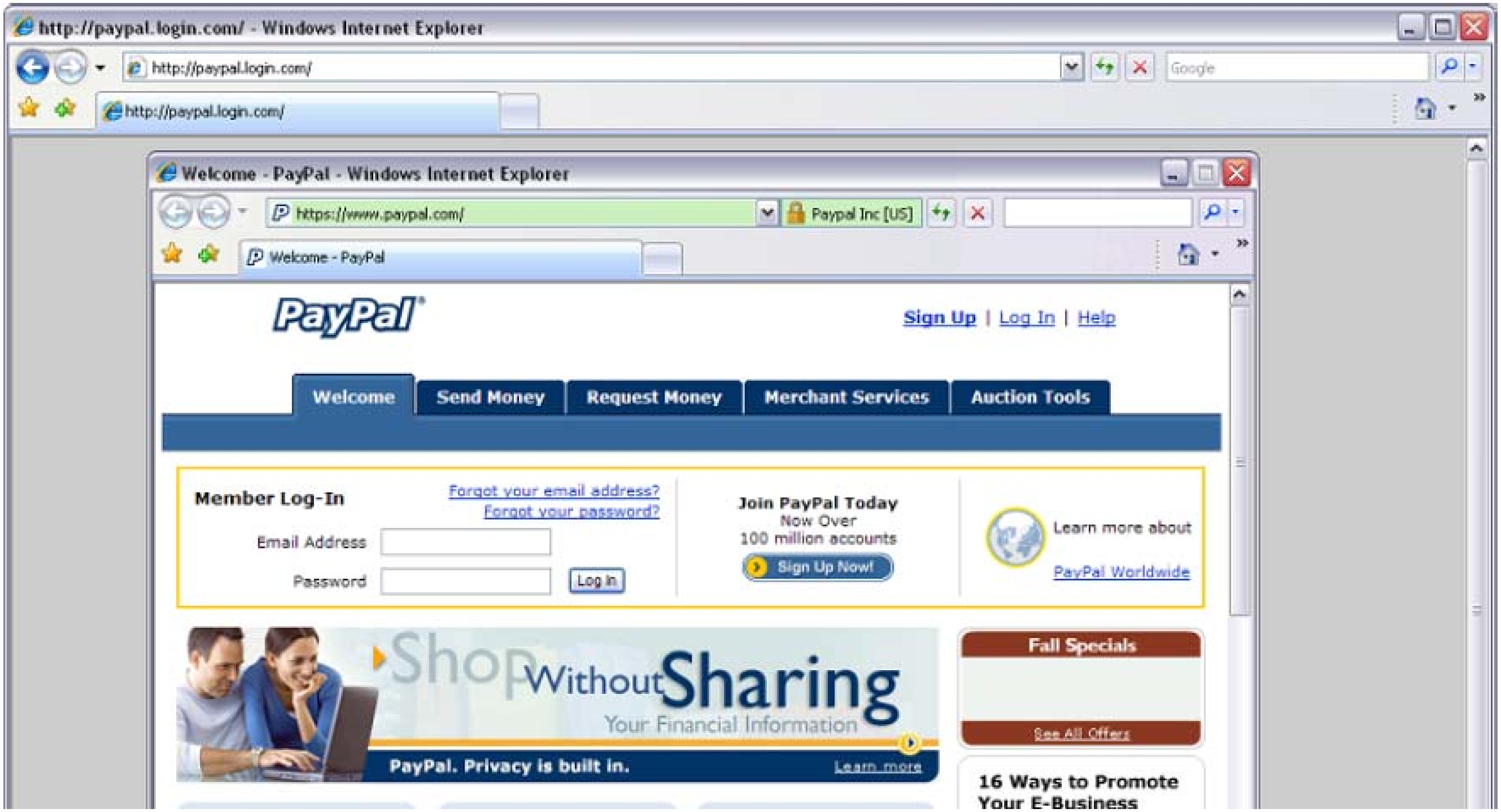
Helps block “semantic attacks”: www.bankofthevvest.com



PayPal, Inc. [US] https://www.paypal.com/home

note: HTTPS-EV and HTTPS are in the same origin

A general UI attack: picture-in-picture



Member Log-In

Forgot your email address?
Forgot your password?

Email Address

Password

Join PayPal Today
Now Over
100 million accounts

Learn more about
PayPal Worldwide

16 Ways to Promote
Your E-Business

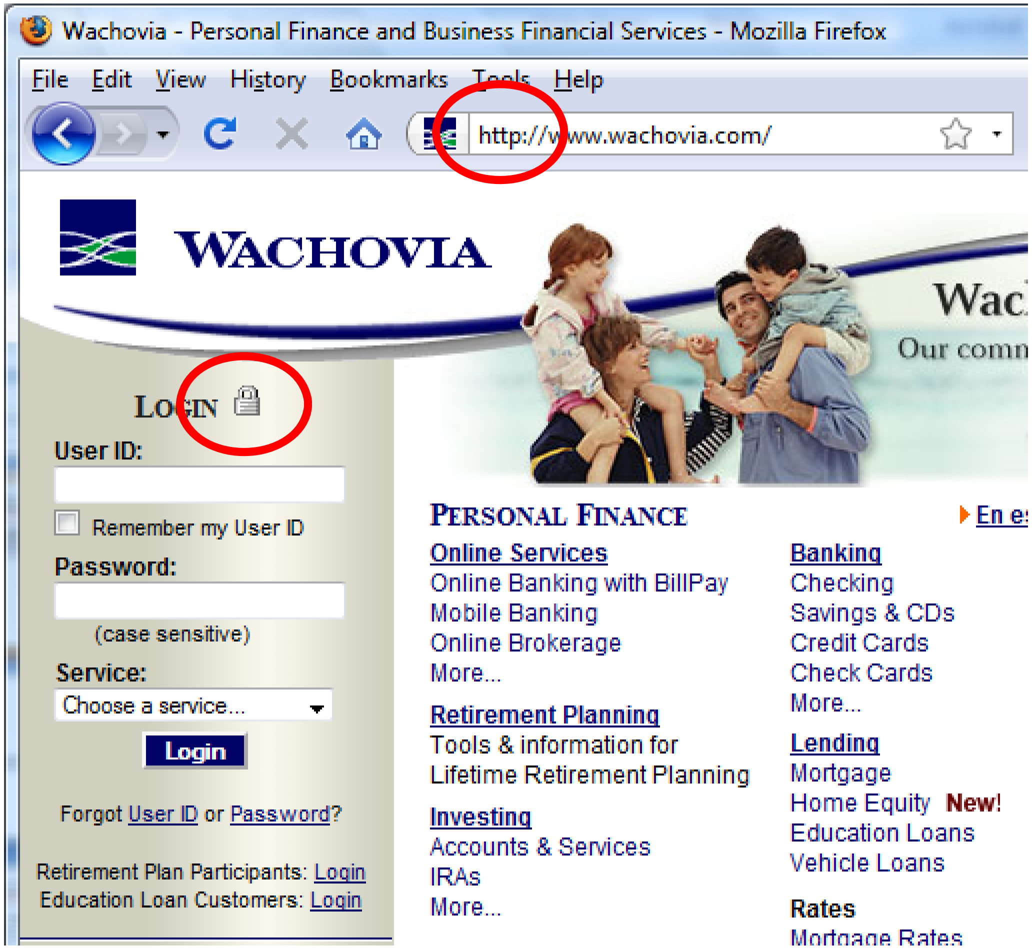
Trained users are more likely to fall victim to this [JSTB'07]

HTTPS and login pages: incorrect usage

Users often land on login page over HTTP:

- Type HTTP URL into address bar

- Google links to HTTP page


Wachovia - Personal Finance and Business Financial Services - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.wachovia.com/
WACHOVIA
Wac
Our comn
Login
Choose a service...

LOGIN

User ID:

☐ Remember my User ID

Password:

(case sensitive)

Service:

Forgot User ID or Password?

Retirement Plan Participants: Login
Education Loan Customers: Login

PERSONAL FINANCE

En es

Online Services
Online Banking with BillPay
Mobile Banking
Online Brokerage
More...

Retirement Planning
Tools & information for Lifetime Retirement Planning

Investing
Accounts & Services
IRAs
More...

Banking
Checking
Savings & CDs
Credit Cards
Check Cards
More...

Lending
Mortgage
Home Equity New!
Education Loans
Vehicle Loans

Rates
Mortgage Rates

(old site)

View source:

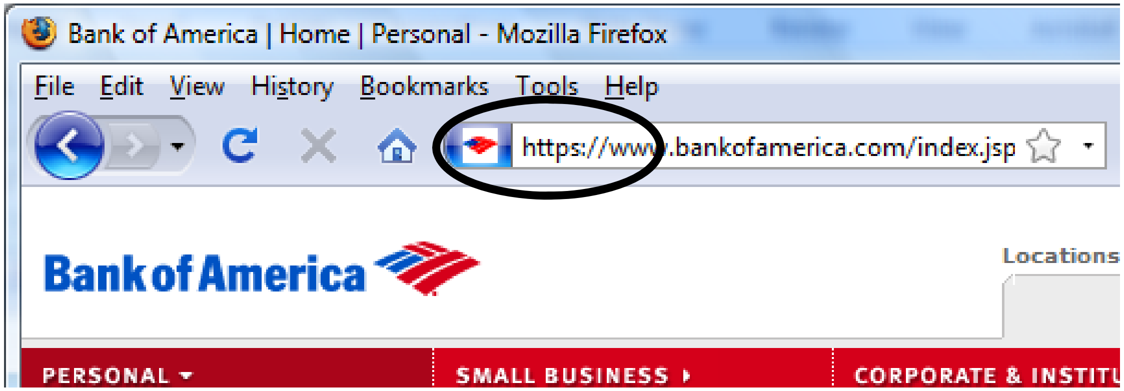
```
<form method="post"
      action="https://onlineservices.wachovia.com/..."
```

HTTPS and login pages: guidelines

General guideline:

Response to http://login.site.com

should be Redirect: https://login.site.com



https://www.bankofamerica.com/index.jsp

Bank of America | Home | Personal - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Bank of America
Locations
PERSONAL
SMALL BUSINESS
CORPORATE & INSTITU

Problems with HTTPS and the Lock Icon

Problems with HTTPS and the Lock Icon

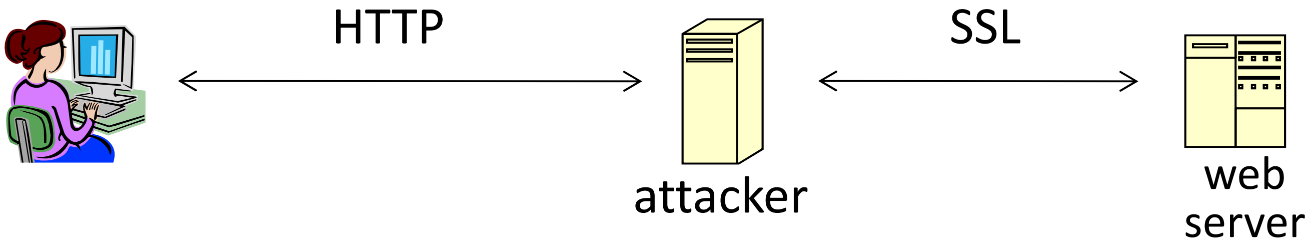
1. Upgrade from HTTP to HTTPS
2. Forged certs
3. Mixed content: HTTP and HTTPS on the same page
4. Does HTTPS hide web traffic?
   - Problems: traffic analysis, compression attacks

1. HTTP → HTTPS upgrade

Common use pattern:

- browse site over HTTP; move to HTTPS for checkout
- connect to bank over HTTP; move to HTTPS for login

SSL_strip attack: prevent the upgrade [Moxie'08]



<a href=https://…> ⇒ <a href=http://…>

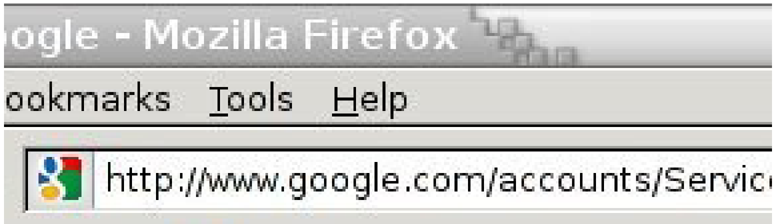
Location: https://… ⇒ Location: http://…

(redirect)

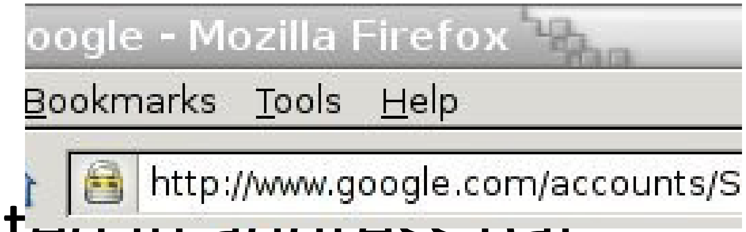
<form action=https://… > ⇒ <form action=http://…>

Tricks and Details

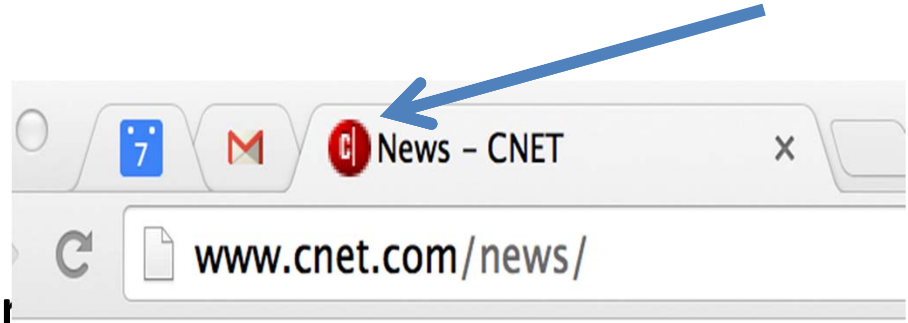
Tricks: drop-in a clever fav icon (older browsers)



⇒



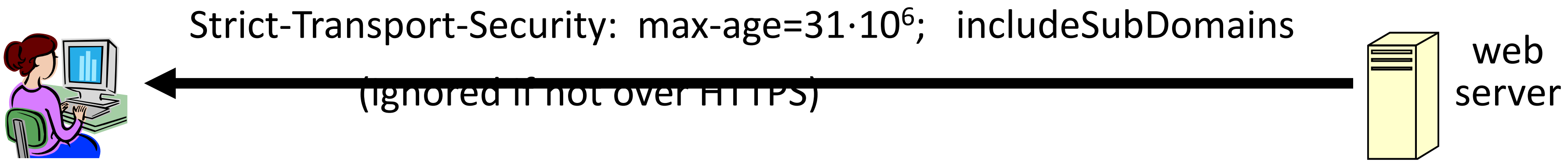
→ fav icon no longer presented in address bar



More tricks: inject "Set-cookie" headers to delete
existing session cookies in browser. Force login.

Number of users who detected HTTP downgrade: 0

Defense: Strict Transport Security (HSTS)



Strict-Transport-Security: max-age=$31 \cdot 10^6$; includeSubDomains

(ignored if not over HTTPS)

Header tells browser to always connect over HTTPS

Subsequent visits must be over HTTPS (self signed certs result in an error)

- Browser refuses to connect over HTTP or if self-signed cert
- Requires that entire site be served over HTTPS

HSTS flag deleted when user “clears private data” : security vs. privacy

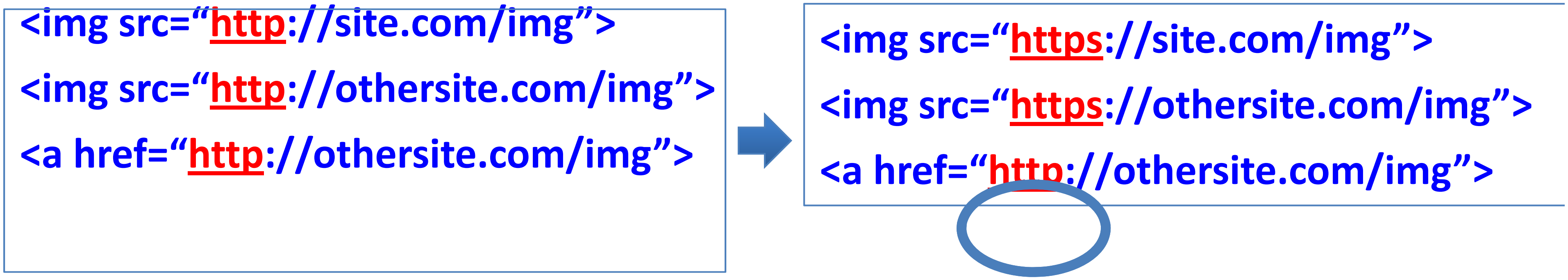
CSP: upgrade-insecure-requests

The problem: many pages use **<img src="http://site.com/img">**

- Makes it difficult to migrate site to HTTPS

Solution:

Content-Security-Policy: upgrade-insecure-requests



```
<img src="http://site.com/img">
<img src="http://othersite.com/img">
<a href="http://othersite.com/img">
```

```
<img src="https://site.com/img">
<img src="https://othersite.com/img">
<a href="http://othersite.com/img">
```

Always use protocol relative URLs **<img src="//site.com/img">**

2. Certificates: wrong issuance

2011: Comodo and DigiNotar CAs hacked, issue certs for Gmail, Yahoo! Mail, …

2013: TurkTrust issued cert. for gmail.com (discovered by pinning)

2014: Indian NIC (intermediate CA trusted by the root CA IndiaCCA) issue certs for Google and Yahoo! domains

Result: (1) India CCA revoked NIC's intermediate certificate

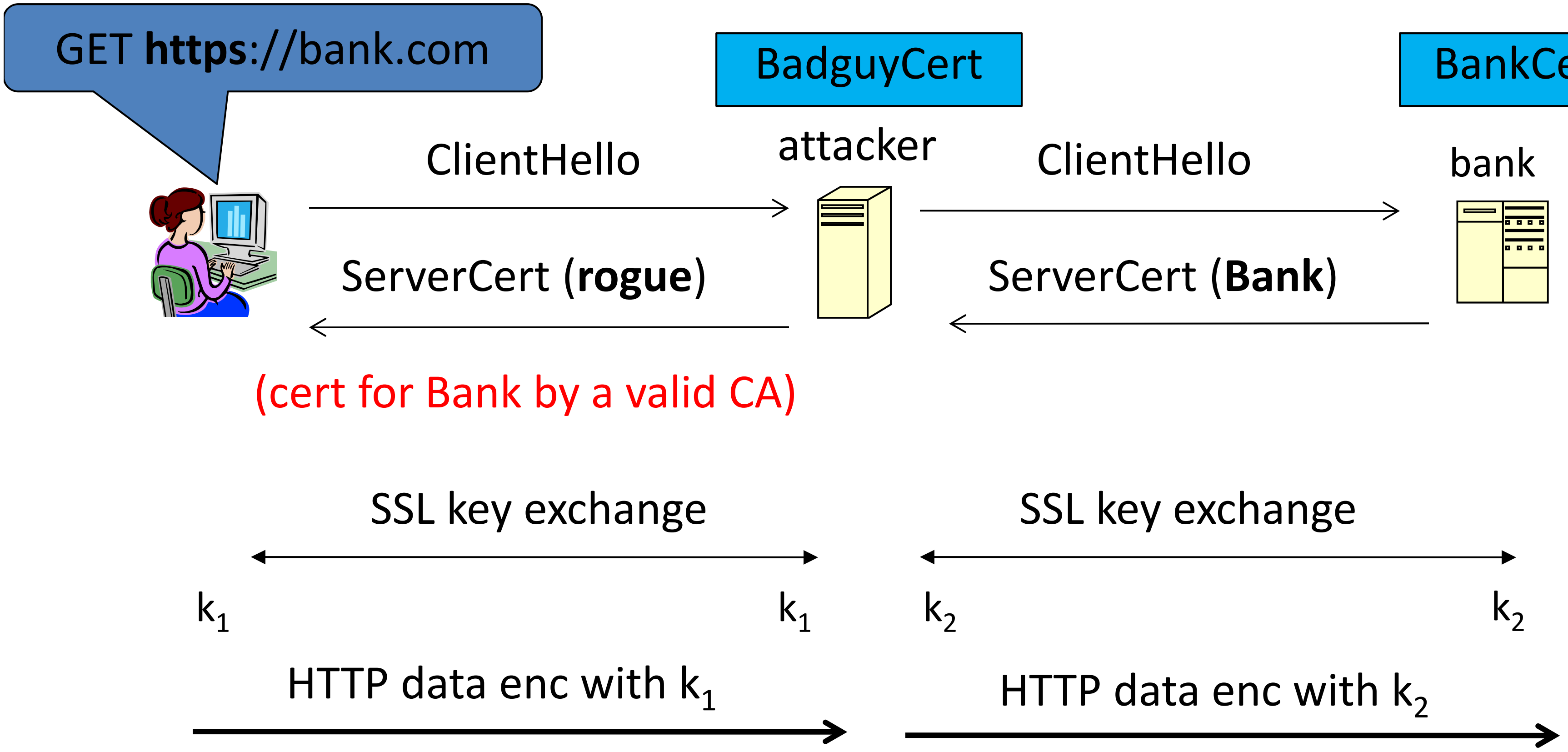
(2) Chrome restricts India CCA root to only seven Indian domains

2015: MCS (intermediate CA cert issued by CNNIC) issues certs for Google domains

Result: CNNIC root no longer recognized by Chrome

⇒ enables eavesdropping w/o a warning on user's session

Man in the middle attack using rogue cert



GET **https**://bank.com

BadgyuyCert

BankCert

ClientHello

attacker

ClientHello

bank

ServerCert (**rogue**)

ServerCert (**Bank**)

(cert for Bank by a valid CA)

SSL key exchange

SSL key exchange

k_1 k_1 k_2 k_2

HTTP data enc with k_1

HTTP data enc with k_2

Attacker proxies data between user and bank.
Sees all traffic and can modify data at will.

What to do? (many good ideas)

1. HTTP public-key pinning, TACK
   - Let a site declare CAs that can sign its cert (similar to HSTS)
   - on subsequent HTTPS, browser rejects certs issued by other CAs
   - TOFU: Trust on First Use

2. Certificate Transparency: [LL'12]
   - idea: CA's must advertise a log of <u>all</u> certs. they issued
   - Browser will only use a cert if it is published on log server
     - Efficient implementation using Merkle hash trees
   - Companies can scan logs to look for invalid issuance

3. Mixed Content: HTTP and HTTPS

Page loads over HTTPS, but contains content over HTTP

(e.g. <script src="http://.../script.js>)

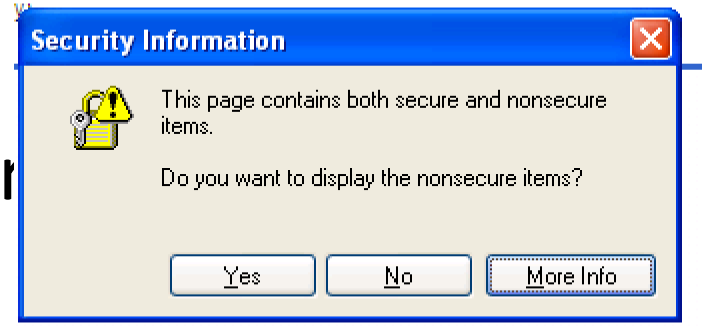
never write this

⇒ Active network attacker can hijack session

ng script en-route to

br

IE7:

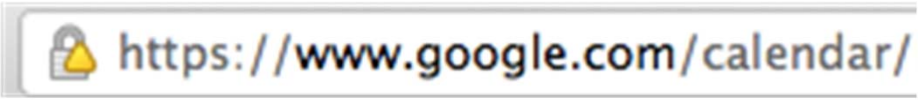


Security Information

This page contains both secure and nonsecure items.

Do you want to display the nonsecure items?

Chrome:



https://www.google.com/calendar/

Chrome policy: CSS, script, frame: blocked; images, XHR: allowed

4. Peeking through SSL: traffic analysis

- Network traffic reveals length of HTTPS packets
  - TLS supports up to 256 bytes of padding

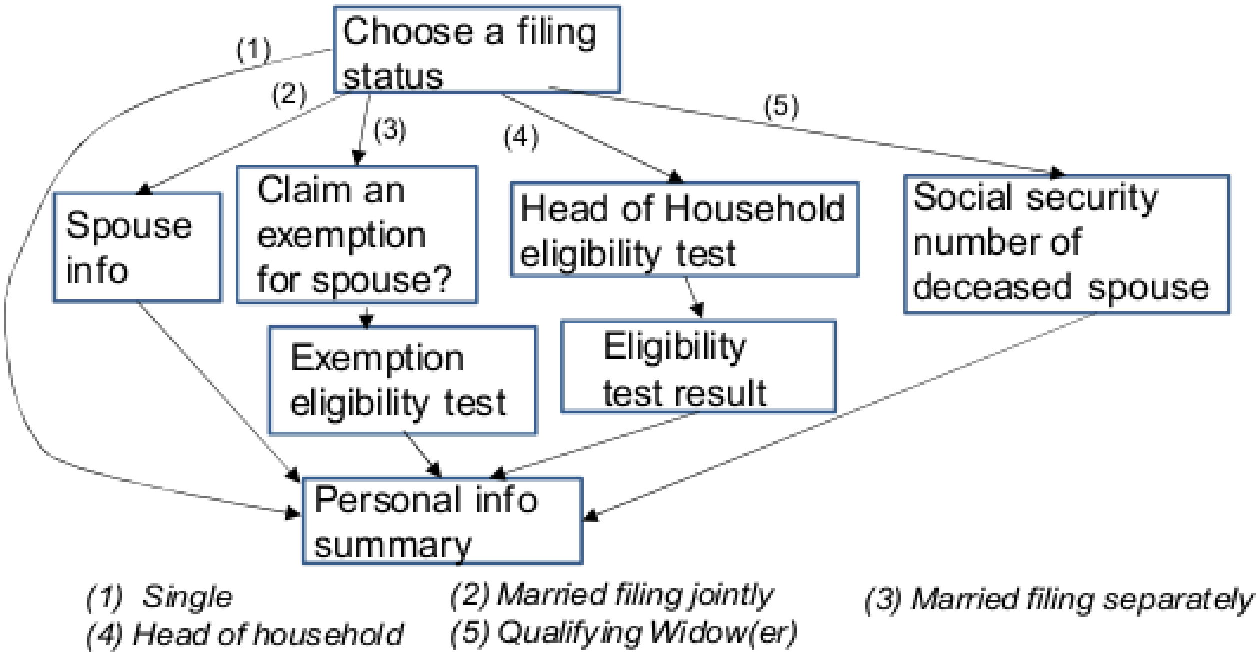
- AJAX-rich pages have lots and lots of interactions with the server

- These interactions expose specific internal state of the page

BAM!

Chen, Wang, Wang, Zhang, 2010

Peeking through SSL: an example [CWWZ'10]


Choose a filing status
(1)
(2)
(3)
(4)
(5)
Spouse info
Claim an exemption for spouse?
Head of Household eligibility test
Social security number of deceased spouse
Exemption eligibility test
Eligibility test result
Personal info summary
(1) Single
(2) Married filing jointly
(3) Married filing separately
(4) Head of household
(5) Qualifying Widow(er)

Vulnerabilities in an online tax application

No easy fix. Can also be used to ID Tor traffic

Peeking through SSL: compression

[DR'12]

HTTPS: supports compressing data before encryption (16KB records)

Attacker: wants to recover Gmail session cookie (say)

- Places Javascript on some site that issues request:

```
GET  gmail.com/__AAAAAAAAAAAAA….AAAAAA
Cookie:   session=__A 6Bh63g53ig4
Host: gmail.com
```

16KB

1st byte of cookie is "A" ⇒ record will compress more than when not

- Script tries all possibilities to expose 1st byte. Moves to 2nd bytes …

What to do: do not use compression with HTTPS

Peeking through SSL: weak algs.

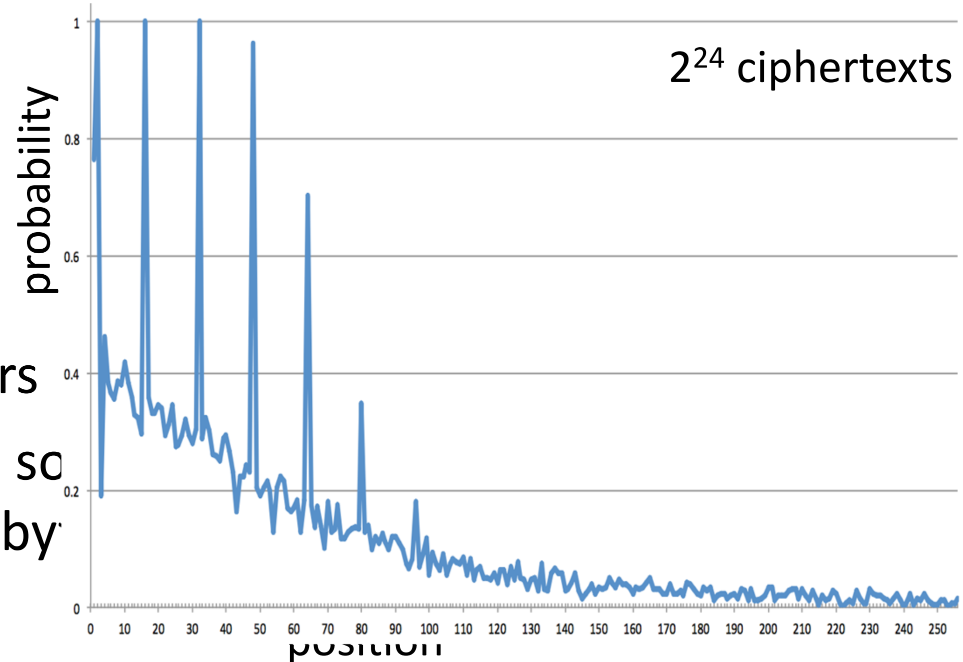
[ABPPS'13]

RC4: a stream cipher commonly used in HTTPS

(fast, other options in TLS 1.0 are problematic)

Bad news: [MS'01, M'02, ABPPS'13]

RC4 does not hide plaintext well



[Source: ABPPS'13]

What to do:

- Push for TLS 1.2 support in browsers
- If must use RC4, pad HTTP headers sc that nothing important in first 512 by

Module 11: DDoS

Denial of Service

What is network DoS?

- Goal: take out a large site with little computing work

- How: **Amplification**
  - Small number of packets ⇒ big effect

- Two types of amplification attacks:
  - DoS bug:
    - Design flaw allowing one machine to disrupt a service
  - DoS flood:
    - Command bot-net to generate flood of requests

DoS can happen at any layer

- This lecture:
  - Sample Dos at different layers (by order):
    - Link
    - TCP/UDP
    - Application
  - Generic DoS solutions
  - Network DoS solutions

- Sad truth:
  - Current Internet not designed to handle DDoS attacks

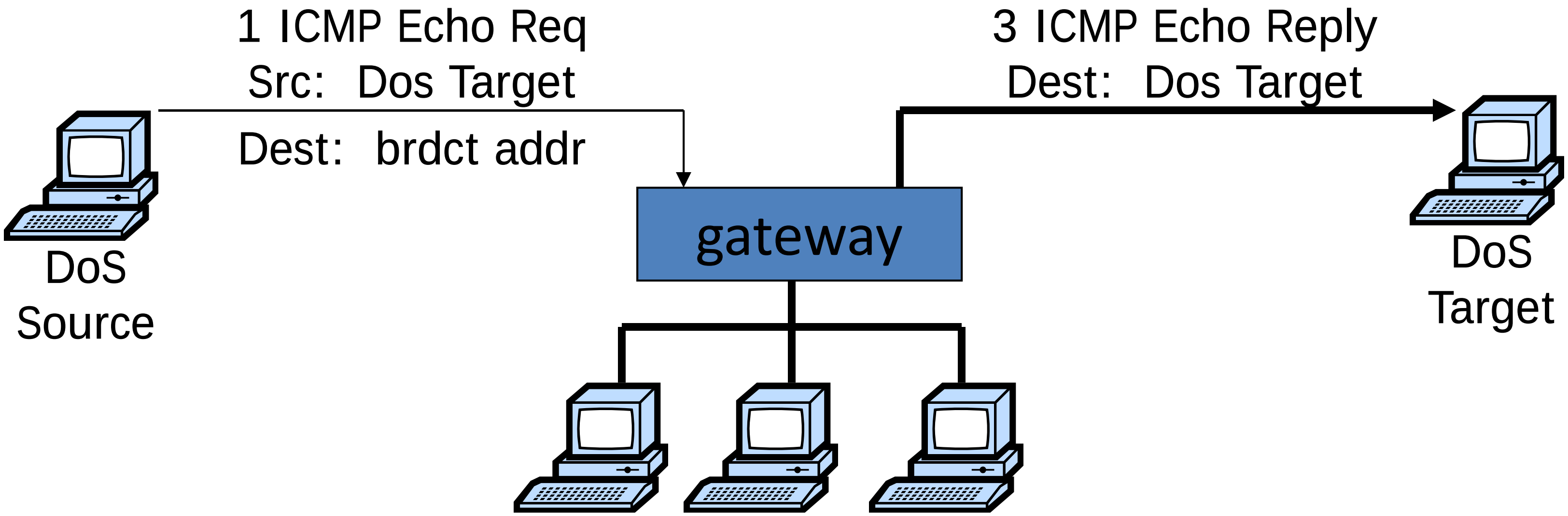
Warm up: 802.11b DoS bugs

- Radio jamming attacks: trivial, not our focus.

- Protocol DoS bugs: [Bellardo, Savage, ’03]
  - NAV (Network Allocation Vector):
    - 15-bit field. Max value: 32767
    - Any node can reserve channel for NAV seconds
    - No one else should transmit during NAV period
    - ... but not followed by most 802.11b cards

  - De-authentication bug:
    - Any node can send deauth packet to AP
    - Deauth packet unauthenticated
    - ... attacker can repeatedly deauth anyone

Smurf amplification DoS attack

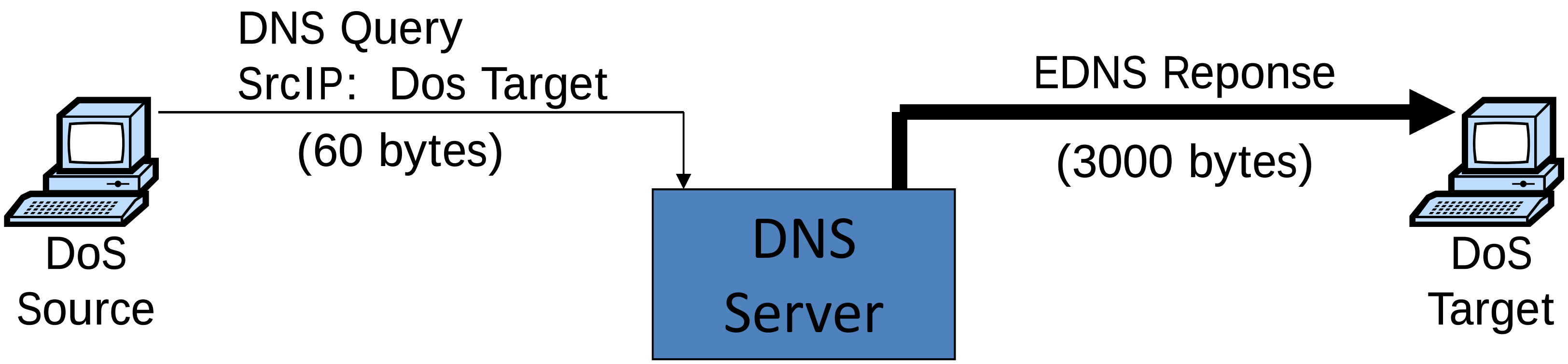


- Send ping request to broadcast addr (ICMP Echo Req)
- Lots of responses:
  - Every host on target network generates a ping reply (ICMP Echo Reply) to victim

Prevention: reject external packets to broadcast address

Modern day example (Mar ’13)

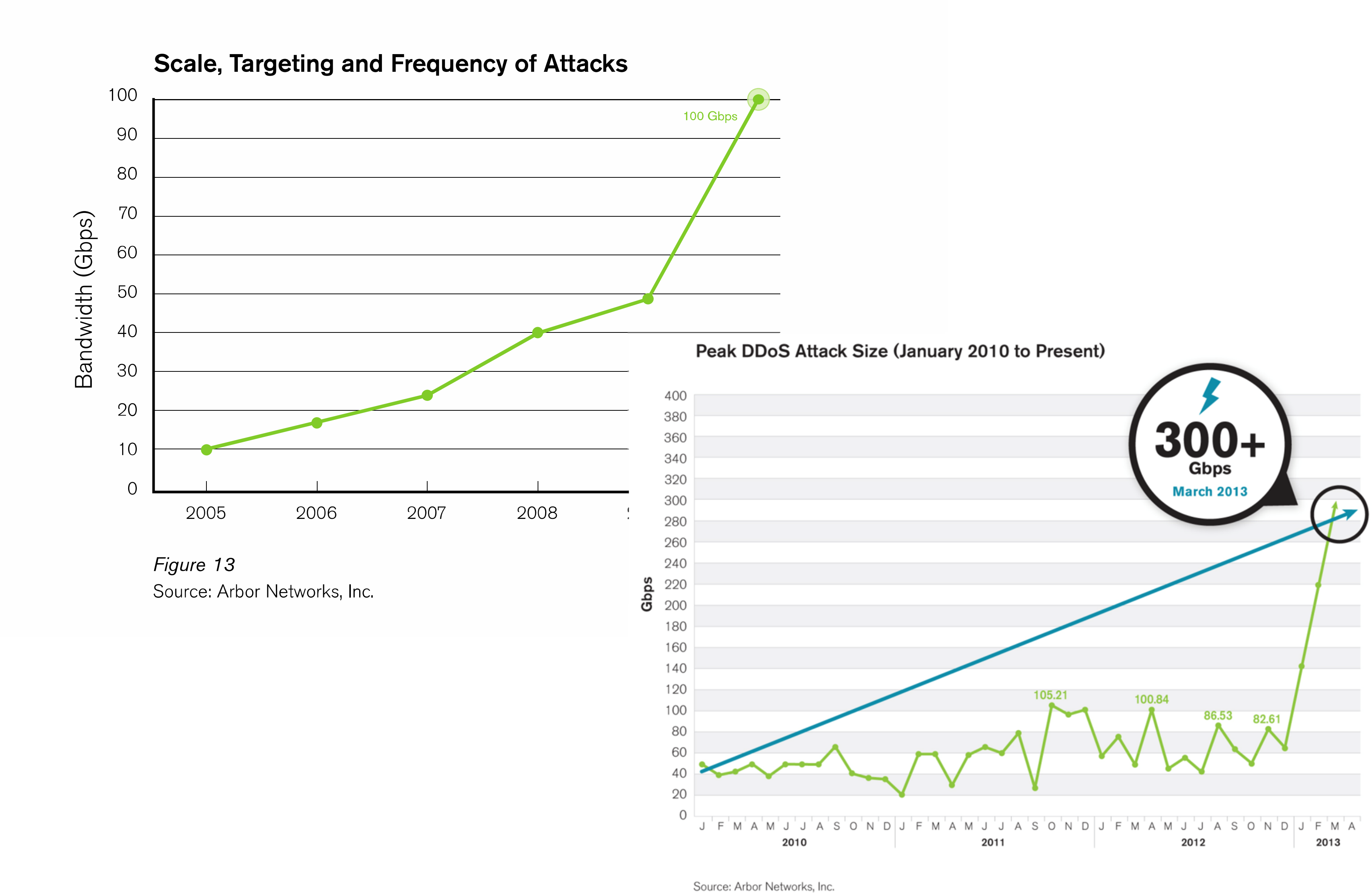
DNS Amplification attack: (×50 amplification)



2006: 0.58M open resolvers on Internet (Kaminsky-Shiffman)

2014: 28M open resolvers (openresolverproject.org)

⇒ 3/2013: DDoS attack generating 309 Gbps for 28 mins.



Scale, Targeting and Frequency of Attacks

Figure 13

Source: Arbor Networks, Inc.

Peak DDoS Attack Size (January 2010 to Present)

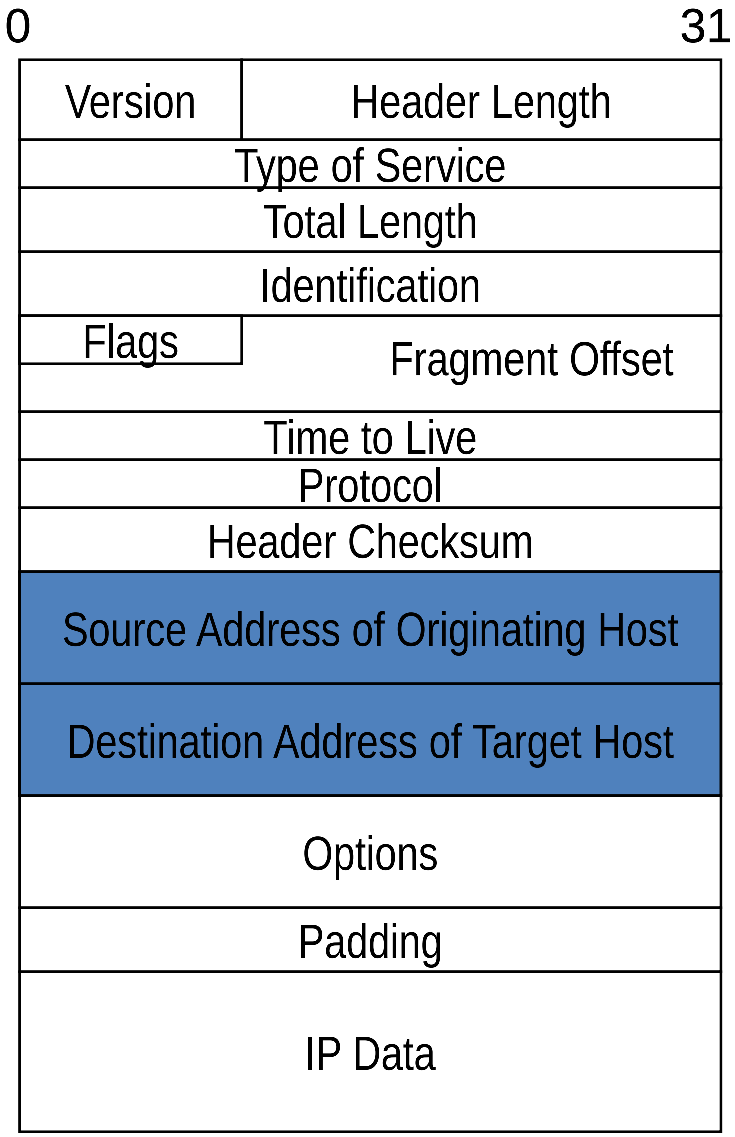
300+ Gbps March 2013

Source: Arbor Networks, Inc.

Feb. 2014: 400 Gbps via NTP amplification (4500 NTP servers)

Review: IP Header format

- Connectionless
  - – Unreliable
  - – Best effort



0

31

Version

Header Length

Type of Service

Total Length

Identification

Flags

Fragment Offset

Time to Live

Protocol

Header Checksum

Source Address of Originating Host

Destination Address of Target Host

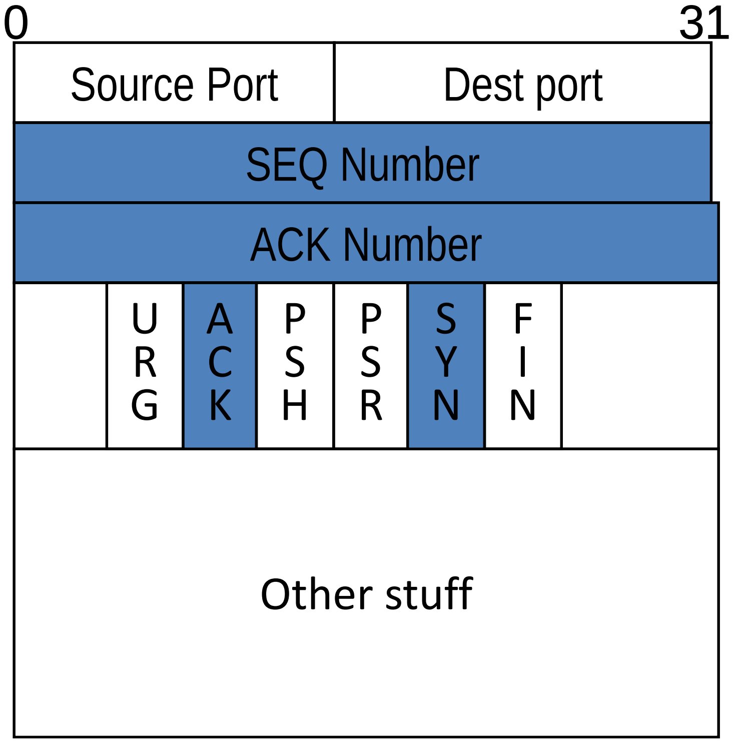
Options

Padding

IP Data

Review: TCP Header format

- TCP:
  - Session based
  - Congestion control
  - In order delivery



0

31

Source Port

Dest port

SEQ Number

ACK Number

URG

ACK

PSH

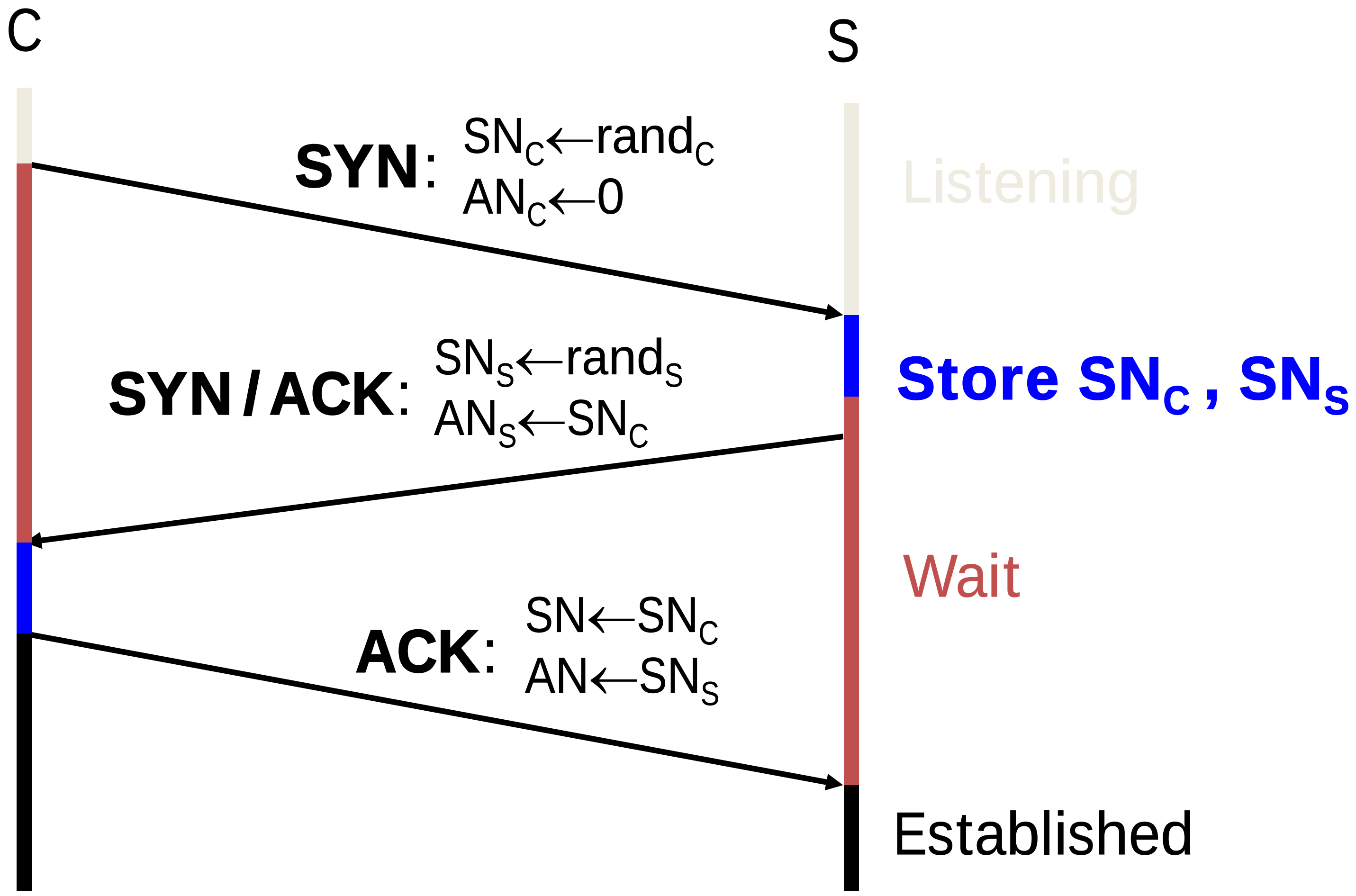
PSR

SYN

FIN

Other stuff

Review: TCP Handshake



C

S

SYN: $SN_C \leftarrow rand_C$ $AN_C \leftarrow 0$

Listening

SYN / ACK: $SN_S \leftarrow rand_S$ $AN_S \leftarrow SN_C$

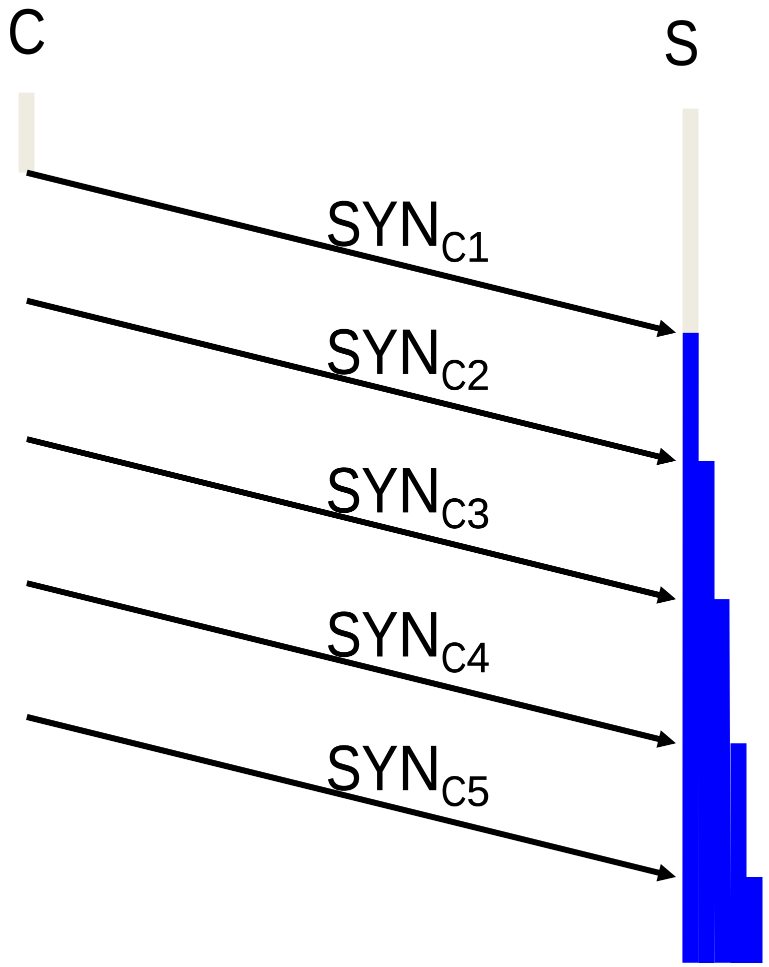
Store SN_C , SN_S

Wait

ACK: $SN \leftarrow SN_C$ $AN \leftarrow SN_S$

Established

TCP SYN Flood I: low rate (DoS bug)



Single machine:

- SYN Packets with **random source IP addresses**
- Fills up backlog queue on server
- No further connections possible

SYN Floods (phrack 48, no 13, 1996)

OS	Backlog queue size
Linux 1.2.x	10
FreeBSD 2.1.5	128
WinNT 4.0	6

Backlog timeout: 3 minutes

⇒ Attacker need only send 128 SYN packets every 3 minutes.

⇒ Low rate SYN flood

A classic SYN flood example

- MS Blaster worm (2003)
  - Infected machines at noon on Aug 16th:
    - SYN flood on port 80 to **windowsupdate.com**
    - 50 SYN packets every second.
      - each packet is 40 bytes.
    - Spoofed source IP: a.b.X.Y where X,Y random.
- MS solution:
  - new name: windowsupdate.microsoft.com
  - Win update file delivered by Akamai

Low rate SYN flood defenses

- Non-solution:
  - Increase backlog queue size or decrease timeout

- Correct solution (when under attack) :
  - **Syncookies**: remove state from server
  - Small performance overhead

Syncookies

[Bernstein, Schenk]

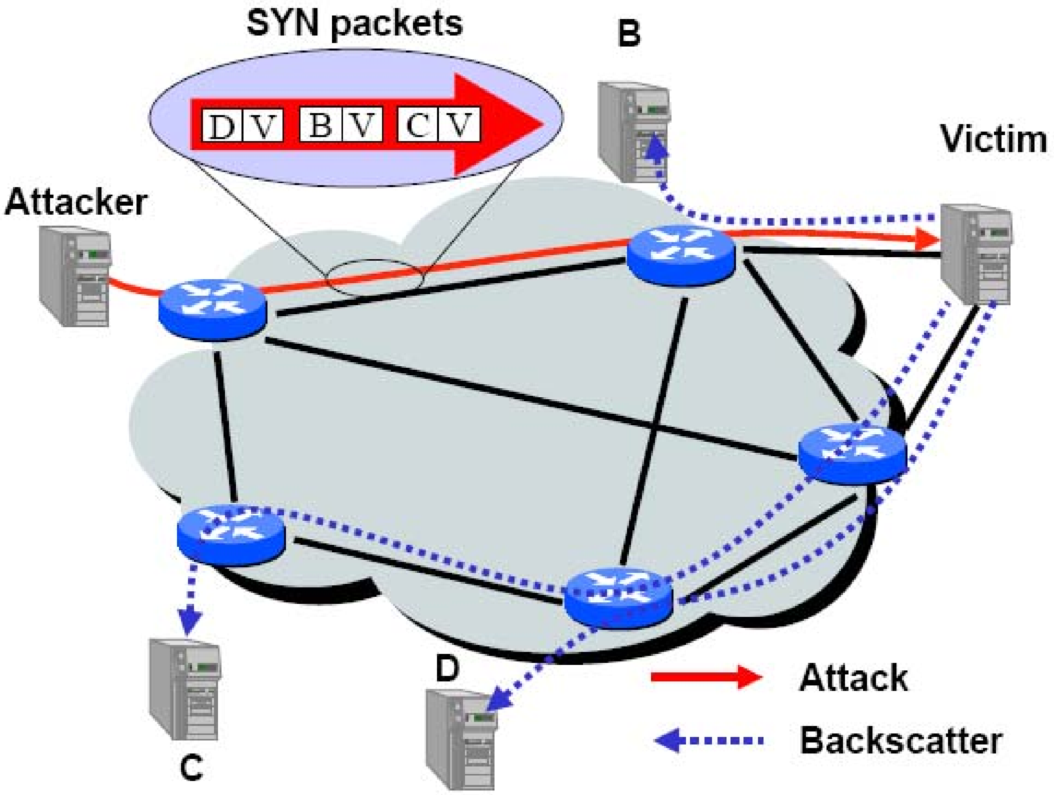
- Idea: use secret key and data in packet to gen. server SN

- Server responds to Client with SYN-ACK cookie:
  - T = 5-bit counter incremented every 64 secs.
  - L = MAC_{key} (SAddr, SPort, DAddr, DPort, SN_C, T) [24 bits]
    - key: picked at random during boot
  - SN_S = (T . mss . L) (|L| = 24 bits)
  - **Server does not save state** (other TCP options are lost)

- Honest client responds with ACK (AN=SN_S , SN=SN_C+1)
  - Server allocates space for socket only if valid SN_S

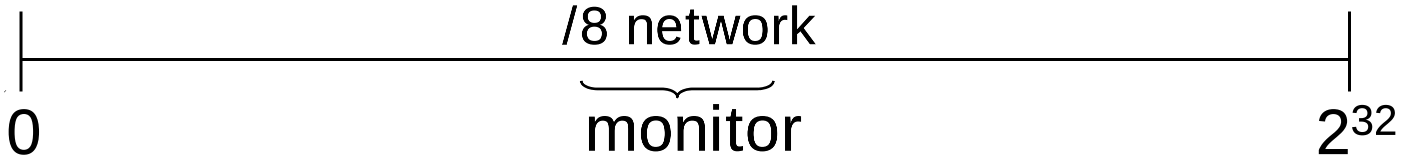
SYN floods: backscatter [MVS' 01]

- SYN with forged source IP ⇒ SYN/ACK to



Backscatter measurement [MVS' 01]

- Listen to unused IP addresss space (darknet)



- Lonely SYN/ACK packet likely to be result of SYN attack

- 2001: **400** SYN attacks/week
- 2013: **773** SYN attacks/24 hours (arbor networks ATLAS)
  - – Larger experiments: (monitor many ISP darknets)
    - Arbor networks

Estonia attack (ATLAS

- Attack types detected:
  - – 115 ICMP floods, 4 TCP SYN floods
- <u>Bandwidth</u>:
  - – 12 attacks: **70-95 Mbps for over 10 hours**
- All attack traffic was coming from outside Estonia
  - – Estonia's solution:
    - Estonian ISPs blocked all foreign traffic until attacks stopped

    => DoS attack had little impact inside Estonia

SYN Floods II: Massive flood (e.g BetCris.com '03)

- Command bot army to flood specific target: (DDoS)

  - **20,000** bots can generate **2Gb/sec** of SYNs (2003)

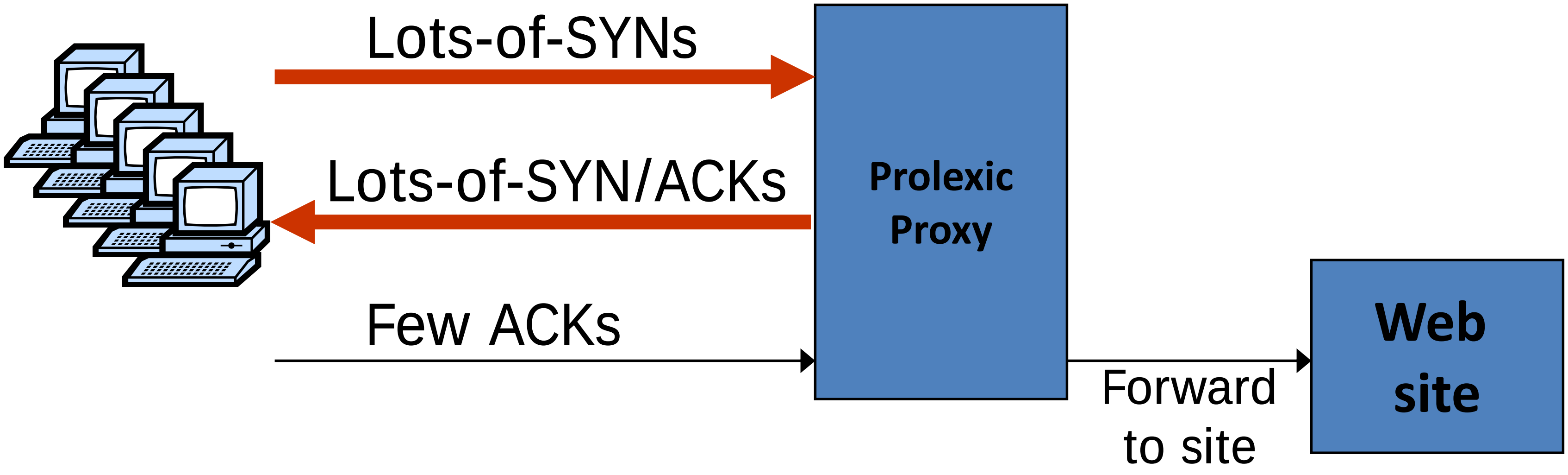
  - At web site:
    - Saturates network uplink or network router
    - Random source IP ⇒

      attack SYNs look the same as real SYNs

  - What to do ???

Prolexic / CloudFlare

- Idea: only forward established TCP connections to site



Other junk packets

Attack Packet	**Victim Response**	**Rate: attk/day [ATLAS 2013]**
TCP SYN to open port	TCP SYN/ACK	773
TCP SYN to closed port	TCP RST	
TCP ACK or TCP DATA	TCP RST	
TCP RST	No response	
TCP NULL	TCP RST	
ICMP ECHO Request	ICMP ECHO Response	**50**
UDP to closed port	ICMP Port unreachable	387

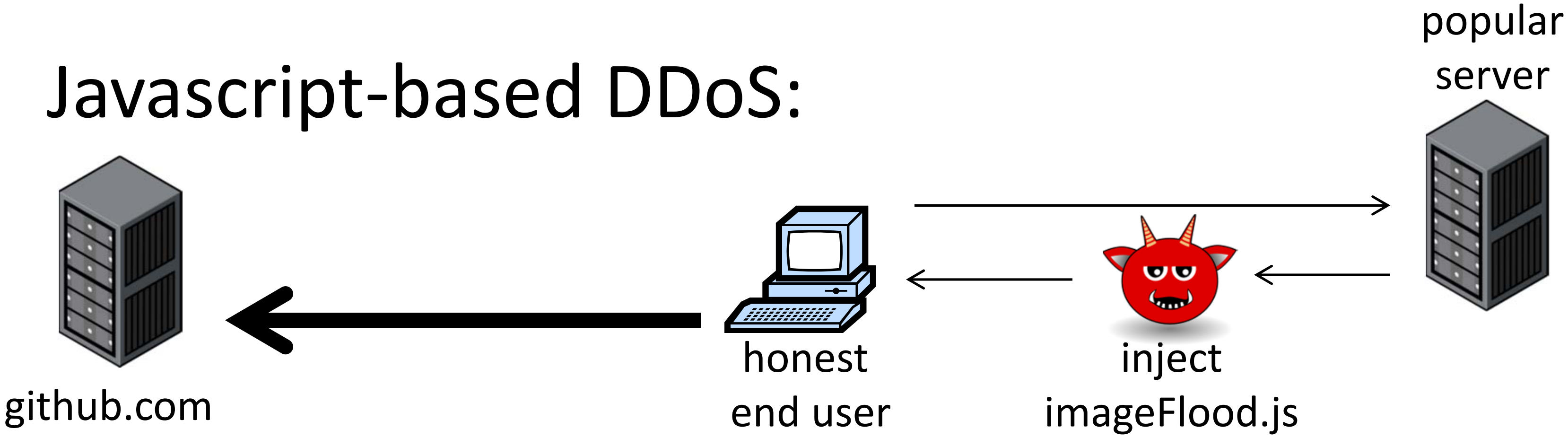
Proxy must keep floods of these away from web site

Stronger attacks: TCP con flood

- Command bot army to:
  - Complete TCP connection to web site
  - Send short HTTP HEAD request
  - Repeat
- Will bypass SYN flood protection proxy
- ... but:
  - Attacker can no longer use random source IPs.
    - Reveals location of bot zombies
  - Proxy can now block or rate-limit bots.

A real-world example: GitHub (3/2015)

Javascript-based DDoS:



imageFlood.js

```
function imgflood() {
  var TARGET = 'victim-website.com/index.php?’
  var rand = Math.floor(Math.random() * 1000)
  var pic = new Image()
  pic.src = 'http://'+TARGET+rand+'=val'
}
setInterval(imgflood, 10)
```

Would HTTPS prevent this DDoS?

DNS DoS Attacks (e.g. bluesecurity ’06)

- DNS runs on UDP port 53
  - DNS entry for victim.com hosted at victim_isp.com

- DDoS attack:
  - flood victim_isp.com with requests for victim.com
  - **Random source IP address** in UDP packets

- Takes out entire DNS server: (collateral damage)
  - bluesecurity DNS hosted at Tucows DNS server
  - DNS DDoS took out Tucows hosting many many sites

- What to do ???

DNS DoS solutions

- Generic DDoS solutions:
  - – Later on. Require major changes to DNS.

- DoS resistant DNS design: (e.g. CloudFlare)
  - – **CoDoNS**: [Sirer' 04]
    - Cooperative Domain Name System
  - – P2P design for DNS system:
    - DNS nodes share the load
    - Simple update of DNS entries
    - Backwards compatible with existing DNS

DoS via route hijacking

- YouTube is 208.65.152.0**/22** (includes 2^{10} IP addr)

  youtube.com is 208.65.153.238, ...

- Feb. 2008:
  - Pakistan telecom advertised a BGP path for

    208.65.153.0**/24** (includes 2^{8} IP addr)
  - Routing decisions use most specific prefix
  - The entire Internet now thinks

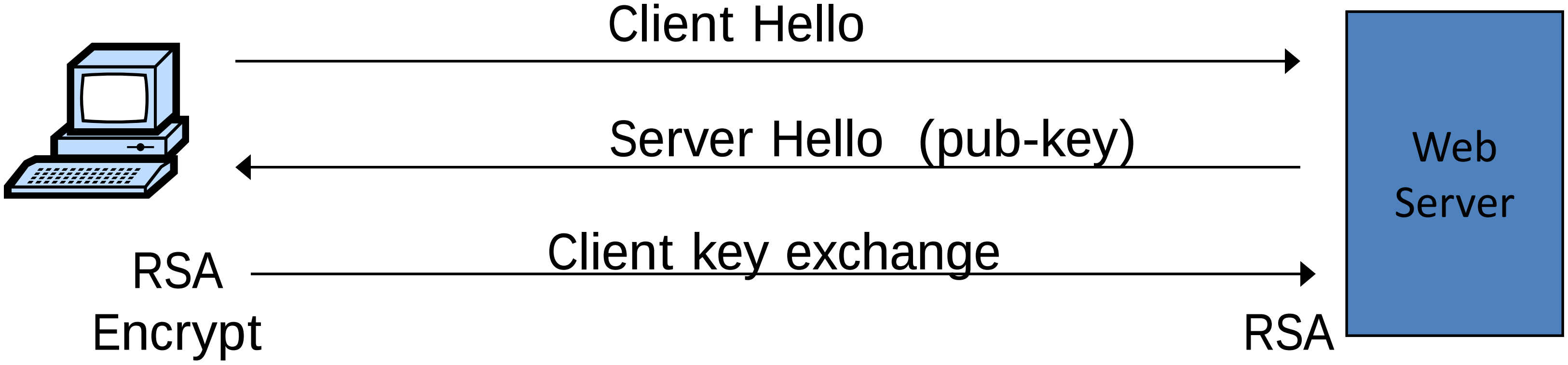
    208.65.153.238 is in Pakistan

◆ Outage resolved within two hours

... but demonstrates huge DoS vuln. with no solution!

DoS at higher layers

- SSL/TLS handshake [SD' 03]



Client Hello

Server Hello (pub-key)

Client key exchange

Web Server

RSA Encrypt

RSA Decrypt

  - RSA-encrypt speed ≈ 10× RSA-decrypt speed

  ⇒ Single machine can bring down ten web servers

- Similar problem with application DoS:
  - Send HTTP request for some large PDF file

  ⇒ Easy work for client, hard work for server.

DoS Mitigation

1. Client puzzles

- Idea: slow down attacker

- Moderately hard problem:
  - Given challenge C find X such that

    $$\mathbf{LSB_n\left(SHA\text{-}1(\ C \ \|\ X\ )\ \right) \ = \ 0^n}$$

  - Assumption: takes expected 2^n time to solve
  - For n=16 takes about .3sec on 1GhZ machine
  - Main point: checking puzzle solution is easy.

- During DoS attack:
  - Everyone must submit puzzle solution with requests
  - When no attack: do not require puzzle solution

Examples

- TCP connection floods (RSA ‘99)
  - – Example challenge: C = TCP server-seq-num
  - – First data packet must contain puzzle solution
    - Otherwise TCP connection is closed

- SSL handshake DoS: (SD’03)
  - – Challenge C based on TLS session ID
  - – Server: check puzzle solution before RSA decrypt.

- Same for application layer DoS and payment DoS.

Benefits and limitations

- Hardness of challenge: n
  - Decided based on DoS attack volume.

- Limitations:
  - Requires changes to both clients and servers
  - Hurts low power legitimate clients during attack:
    - Clients on cell phones and tablets cannot connect

Memory-bound functions

- CPU power ratio:
  - high end server / low end cell phone = 8000

  ⇒ Impossible to scale to hard puzzles

- Interesting observation:
  - Main memory access time ratio:
    - high end server / low end cell phone = 2

- Better puzzles:
  - Solution requires many main memory accesses
    - Dwork-Goldberg-Naor, Crypto '03
    - Abadi-Burrows-Manasse-Wobber, ACM ToIT '05

2. CAPTCHAs

- Idea: verify that connection is from a human

STRAIGHT
STRAIGHT
EARLY
RAIL
STORE
STORE
BEAUTIFUL
BEAUTIFUL
EARLY
STORE

- Applies to application layer DDoS [Killbots ’05]
  - During attack: generate CAPTCHAs and process request only if valid solution
  - Present one CAPTCHA per source IP address.

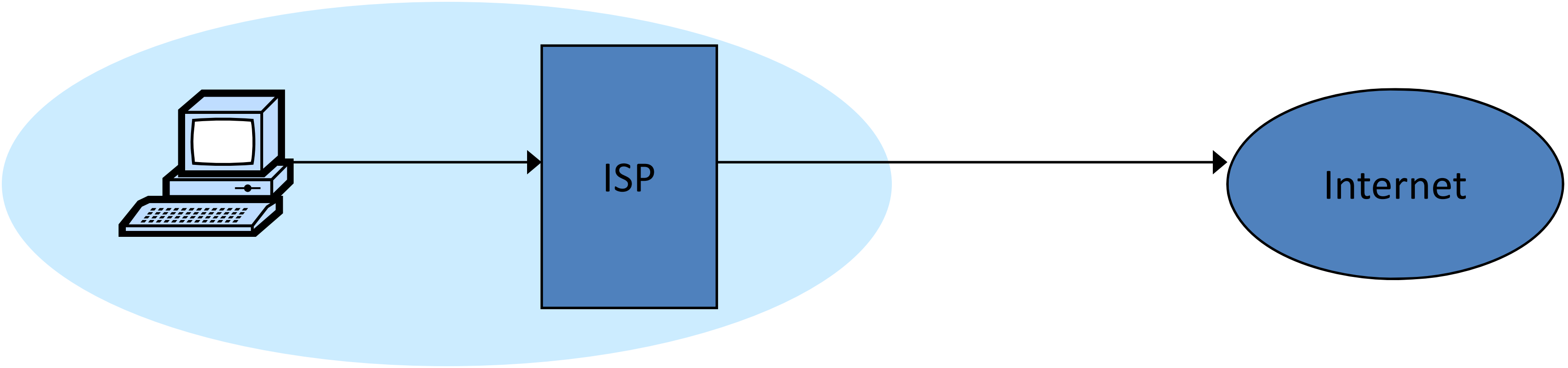
3. Source identification

Goal: identify packet source

Ultimate goal: block attack at the source

1. Ingress filtering (RFC 2827, 3704)

- Big problem: DDoS with spoofed source IPs



- Ingress filtering policy: ISP only forwards packets
  with legitimate source IP (see also SAVE protocol)

Implementation problems

ALL ISPs must do this. Requires global trust.

- If 10% of ISPs do not implement ⇒ no defense
- No incentive for deployment

2014:

- 25% of Auto. Systems are fully spoofable (spoofer.cmand.org)
- 13% of announced IP address space is spoofable

Recall: 309 Gbps attack used only 3 networks (3/2013)

2. Traceback [Savage et al. ’00]

- Goal:
  - Given set of attack packets
  - Determine path to source

- How: change routers to record info in packets

- Assumptions:
  - Most routers remain uncompromised
  - Attacker sends many packets
  - Route from attacker to victim remains relatively stable

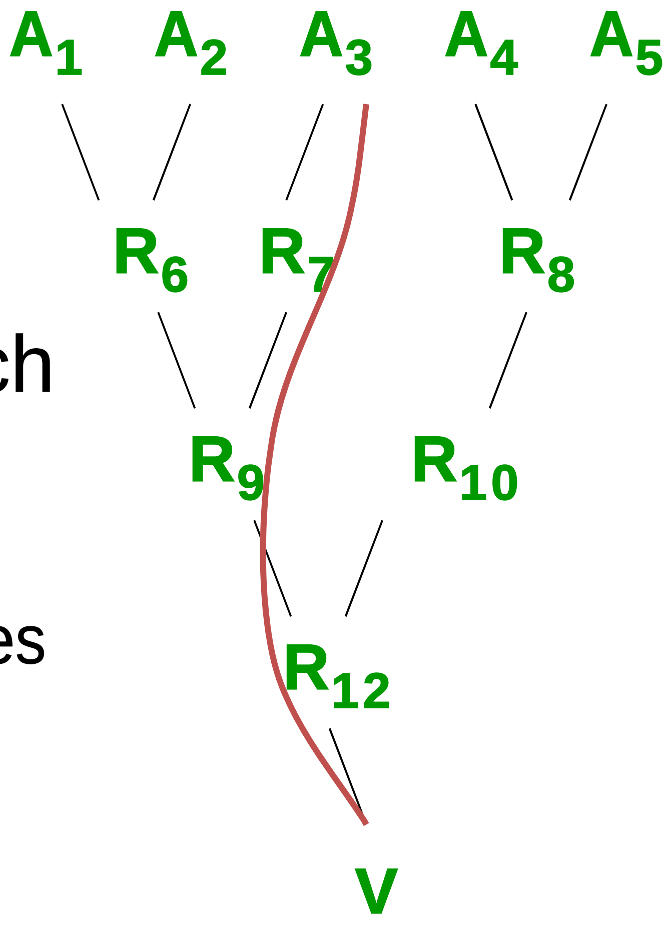
Simple method

- Write path into network packet
  - Each router adds its own IP address to packet
  - Victim reads path from packet
- Problem:
  - Requires space in packet
    - Path can be long
    - No extra fields in current IP format
      - Changes to packet format too much to expect

Better idea

- DDoS involves many packets on same path

- Store one link in each packet
  - Each router probabilistically stores own address
  - Fixed space regardless of path length



Edge Sampling

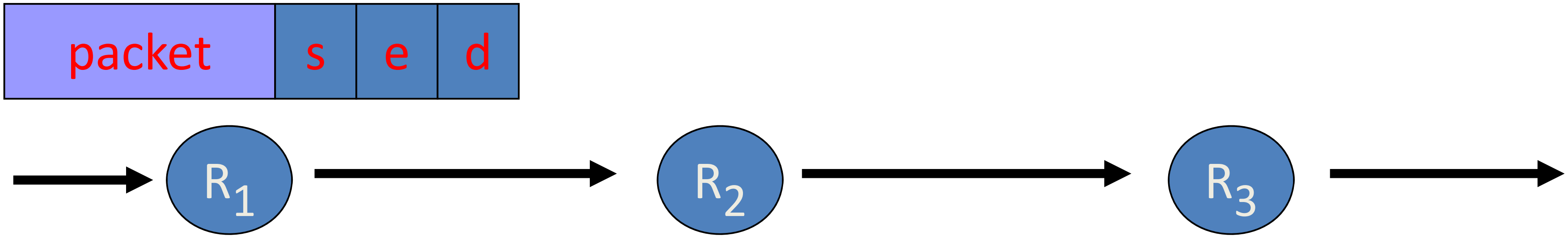
- Data fields written to packet:
  - Edge: *start* and *end* IP addresses
  - Distance: number of hops since edge stored

- Marking procedure for router R

```
if coin turns up heads (with probability p) then
        write R into start address
        write 0 into distance field
else
        if distance == 0 write R into end field
        increment distance field
```

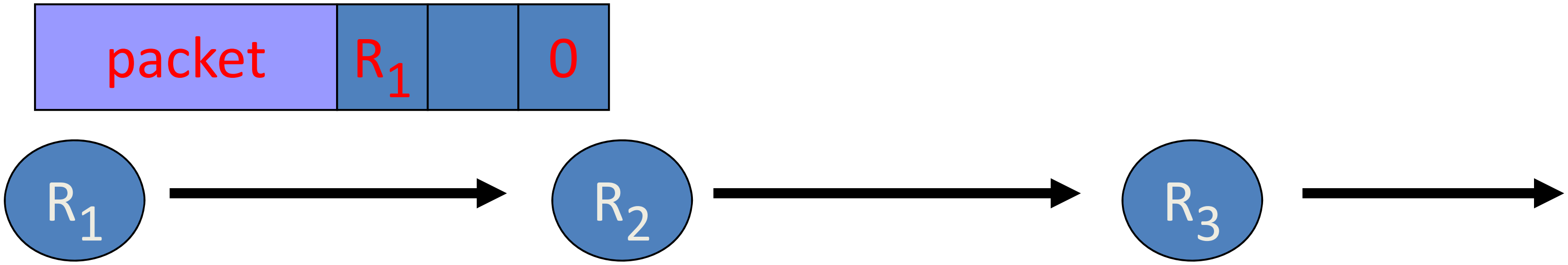
Edge Sampling: picture

- Packet received
  - – R_1 receives packet from source or another router
  - – Packet contains space for start, end, distance



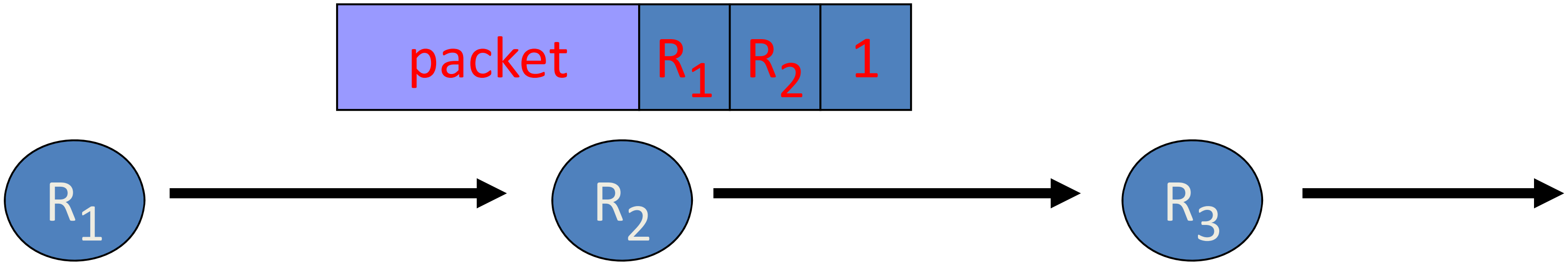
Edge Sampling: picture

- Begin writing edge
  - R_1 chooses to write start of edge
  - Sets distance to 0



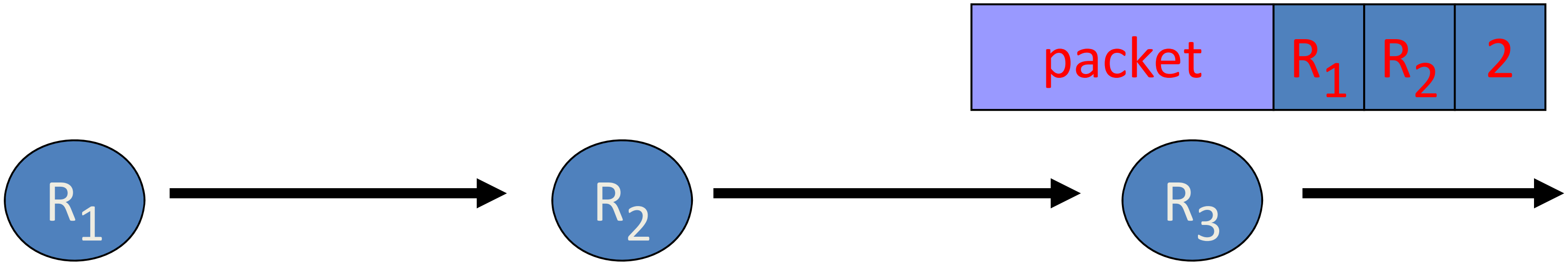
Edge Sampling

- Finish writing edge
  - R_2 chooses not to overwrite edge
  - Distance is 0
    - Write end of edge, increment distance to 1



Edge Sampling

- Increment distance
  - R_3 chooses not to overwrite edge
  - Distance >0
    - Increment distance to 2



Path reconstruction

- Extract information from attack packets

- Build graph rooted at victim
  - Each (start,end,distance) tuple provides an edge

- # packets needed to reconstruct path

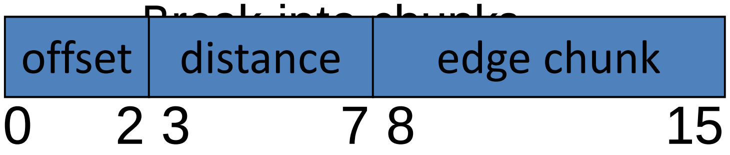
$$E(X) < \frac{\ln(d)}{p(1-p)^{d-1}}$$

where p is marking probability, d is length of path

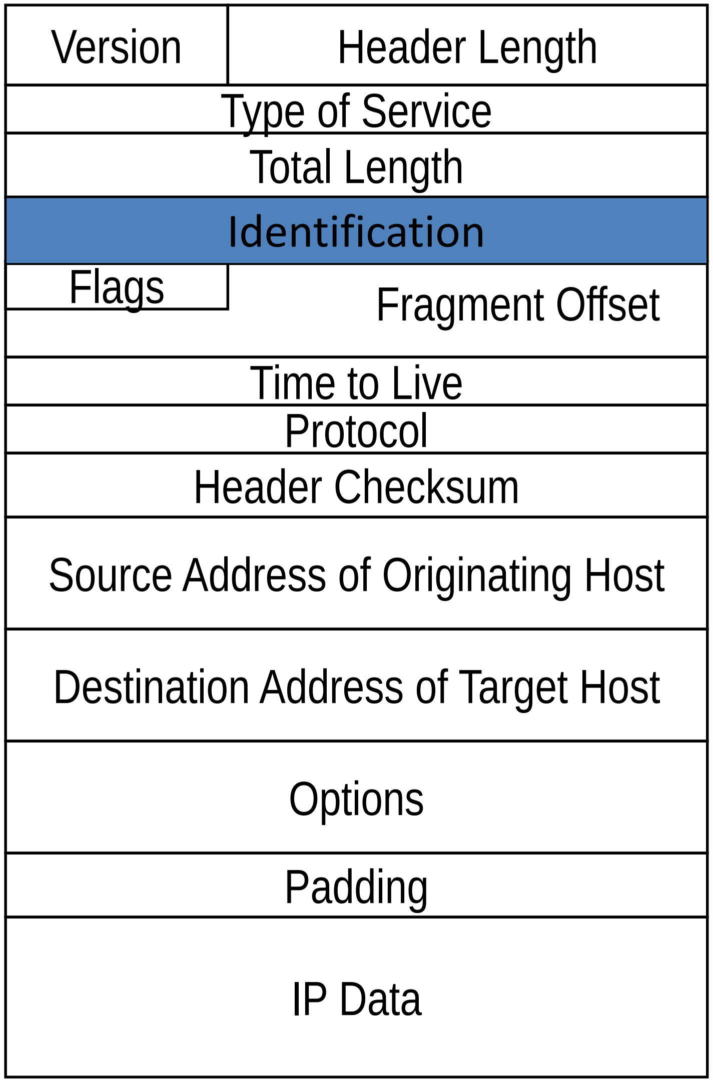
Details: where to store edge

- Identification field
  - Used for fragmentation
  - Fragmentation is rare
  - 16 bits
- Store edge in 16 bits?

Break into chunks



○



More traceback proposals

- Advanced and Authenticated Marking Schemes for IP Traceback
  - Song, Perrig. IEEE Infocomm ’01
  - Reduces noisy data and time to reconstruct paths

- An algebraic approach to IP traceback
  - Stubblefield, Dean, Franklin. NDSS ’02

- Hash-Based IP Traceback
  - Snoeren, Partridge, Sanchez, Jones, Tchakountio, Kent, Strayer. SIGCOMM ‘01

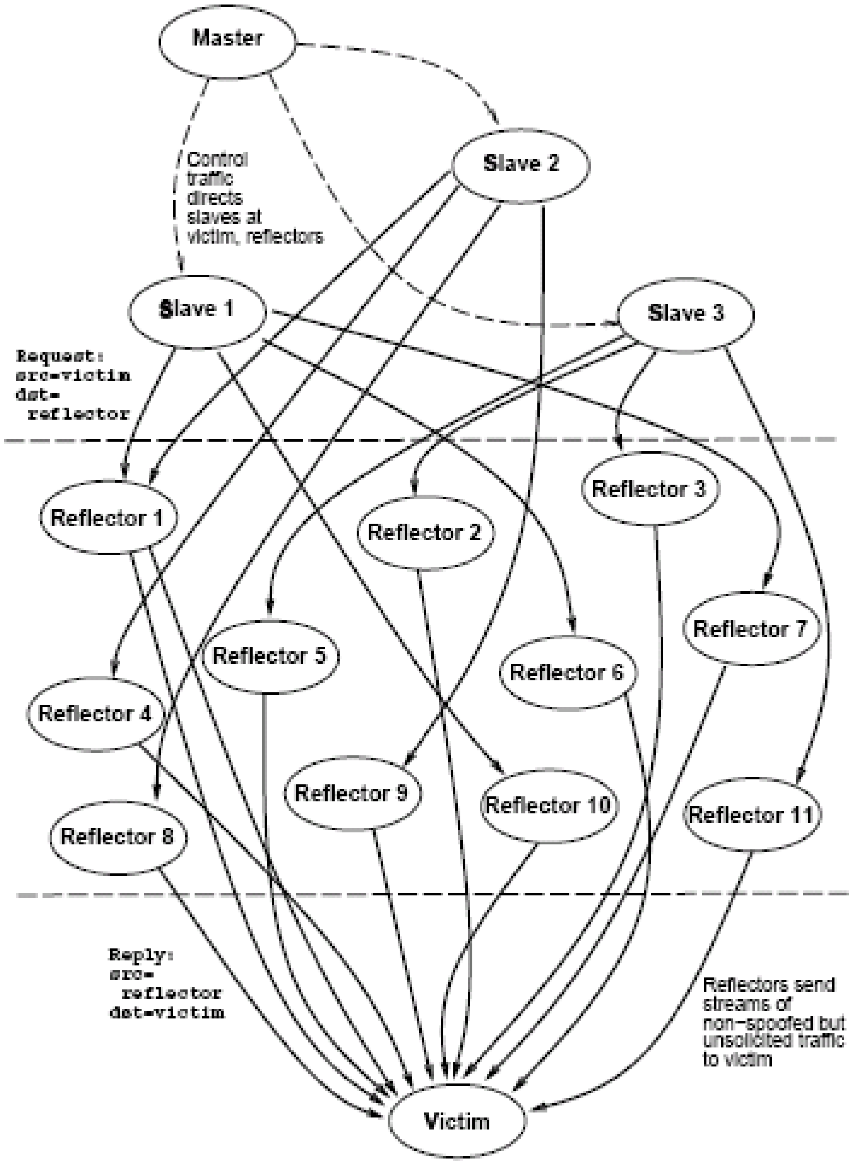
Problem: Reflector attacks [Paxson ’01]

- **Reflector**:
  - A network component that responds to packets
  - Response sent to victim (spoofed source IP)

- Examples:
  - DNS Resolvers: UDP 53 with victim.com source
    - At victim: DNS response
  - Web servers: TCP SYN 80 with victim.com source
    - At victim: TCP SYN ACK packet
  - Gnutella servers

DoS Attack

- Single Master
- Many bots to generate flood
- Zillions of reflectors to hide bots
  - – Kills traceback and pushback methods


Master
Control traffic directs slaves at victim, reflectors
Slave 2
Slave 1
Slave 3
Request: src=victim dst= reflector
Reflector 1
Reflector 2
Reflector 3
Reflector 7
Reflector 5
Reflector 6
Reflector 4
Reflector 9
Reflector 10
Reflector 11
Reflector 8
Reply: src= reflector dst=victim
Reflectors send streams of non−spoofed but unsolicited traffic to victim
Victim

Capability based defense

Capability based defense

- Anderson, Roscoe, Wetherall.
  - Preventing internet denial-of-service with capabilities. SIGCOMM ‘04.
- Yaar, Perrig, and Song.
  - Siff: A stateless internet flow filter to mitigate DDoS flooding attacks. IEEE S&P ’04.
- Yang, Wetherall, Anderson.
  - A DoS-limiting network architecture. SIGCOMM ’05

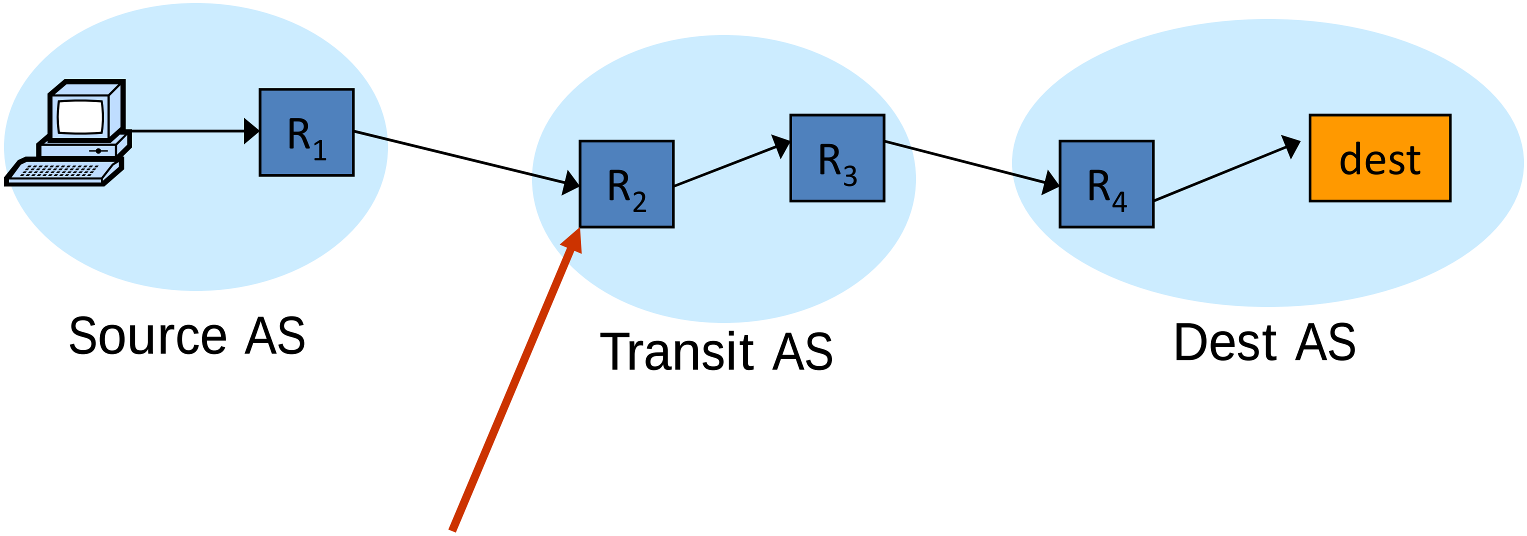
Capability based defense

- Basic idea:
  - Receivers can specify what packets they want

- How:
  - Sender requests capability in SYN packet
    - Path identifier used to limit # reqs from one source
  - Receiver responds with capability
  - Sender includes capability in all future packets
  - **Main point**: Routers only forward:
    - Request packets, and
    - Packets with valid capability

Capability based defense

- Capabilities can be revoked if source is attacking
  - Blocks attack packets close to source



Attack packets dropped

Pushback Traffic Filtering

Pushback filtering

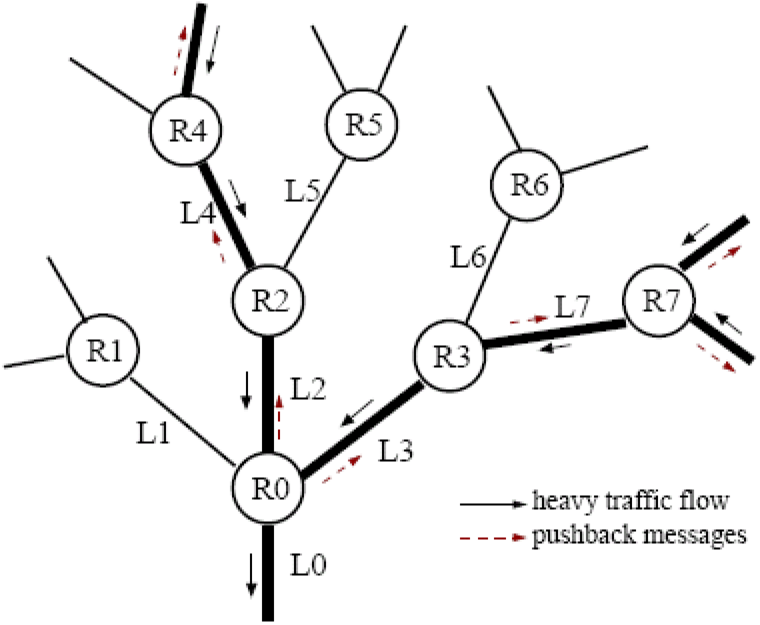
- Mahajan, Bellovin, Floyd, Ioannidis, Paxson, Shenker. Controlling High Bandwidth Aggregates in the Network. *Computer Communications Review* '02.

- Ioannidis, Bellovin. Implementing Pushback: Router-Based Defense Against DoS Attacks. *NDSS* ' 02

- Argyraki, Cheriton. Active Internet Traffic Filtering: Real-Time Response to Denial-of-Service Attacks. USENIX '05.

Pushback Traffic Filtering

- Assumption: DoS attack from few sources



- Iteratively block attacking network segments.

Overlay filtering

Overlay filtering

- Keromytis, Misra, Rubenstein.
  SOS: Secure Overlay Services. SIGCOMM ‘02.

- D. Andersen. Mayday.
  Distributed Filtering for Internet Services.
  Usenix USITS ‘03.

- Lakshminarayanan, Adkins, Perrig, Stoica.
  Taming IP Packet Flooding Attacks. HotNets ’03.

Take home message:

- Denial of Service attacks are real.
  Must be considered at design time.

- Sad truth:
  - Internet is ill-equipped to handle DDoS attacks
  - Commercial solutions: CloudFlare, Prolexic

- Many good proposals for core redesign.