Network Security
and Network Management
#9

Overview

- Encryption
- Authentication
- Message integrity
- Key distribution & Certificates
- Transport Layer Security (TLS)
- IPsec
- Network Management

Security, Privacy and Trust

- Security
  - Host
  - Network ← Focus here
- Trust is more than security
  - Security,
  - Privacy,
  - Robust to failures,
  - Reliability,
  - Usability
- Tussle between security and privacy
  - For example: Tor "Tor is a network of virtual tunnels that allows people and groups to improve their privacy and security on the Internet. It also enables software developers to create new communication tools with built-in privacy features. Tor provides the foundation for a range of applications that allow organizations and individuals to share information over public networks without compromising their privacy. " From http://www.torproject.org/about/overview.html.en

Network Security: Motivation

- Networks are essential components of organizational processes
- Large investments in time and money in network infrastructures
- Information is a valuable resource that must be protected.

Network Security

- Different environments have different security concerns
- Security considerations
  - What do you want to protect?
  - How much do you want to spend?
  - How much does it cost to recover losses?
- Resource: US Cybersecurity and Infrastructure Security Agency (CISA)

Financial Impact of major security breach

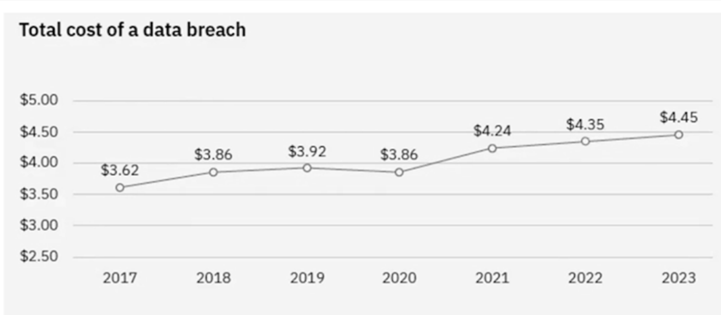


Figure 1. Measured in USD millions

Source: https://thehackernews.com/2023/12/cost-of-data-breach-report-2023.html

What is network security?

Confidentiality: only sender, intended receiver should "understand" message contents

- sender encrypts message
- receiver decrypts message

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

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

Access and Availability: services and data must be accessible and available to users

Desired Properties of a Secure System

- Assurance – the system works
- Non-repudiation – cannot deny the new car was ordered
- Proof of submission – proof the check is in the mail
- Proof of delivery – proof the utility got the check
- Traffic confidentiality – no one can tell when you sent the check
- Anonymity – no one knows who paid your bill
- Audit (logging) – someone can tell when and to whom the message was sent
- Accounting – you get a bill
- Sequence Integrity – bills are paid in the order received
- Trusted third parties – a judge

From: "Information Security" G. Minden

Network Security: Tools

- Limit network access
- Limit access to the physical infrastructure
- Segment the system
- Fiber transmission facilities
  (Because fiber hard to tap)
- Encryption
- Authentication protocols

Threats: What

- Interruption
- Interception
- Modification
- Masquerade
- DoS
  - Example of recent DoS Attack-https://www.bankinfosecurity.com/record-setting-ddos-attack-hits-financial-service-firm-a-17345
- Ransomware

Protecting systems from ransomware

- FBI has listed a series of recommended mitigations for keeping systems protected from ransomware, e.g.,
  - Updating applications and operating systems periodically,
  - Keeping all data backed up offline,
  - Implementing network segmentation
  - Implementing least privileged policies,
  - Previewing logs and auditing user accounts,
  - Implementing multi-factor authentication,
  - Disabling unused protocols.

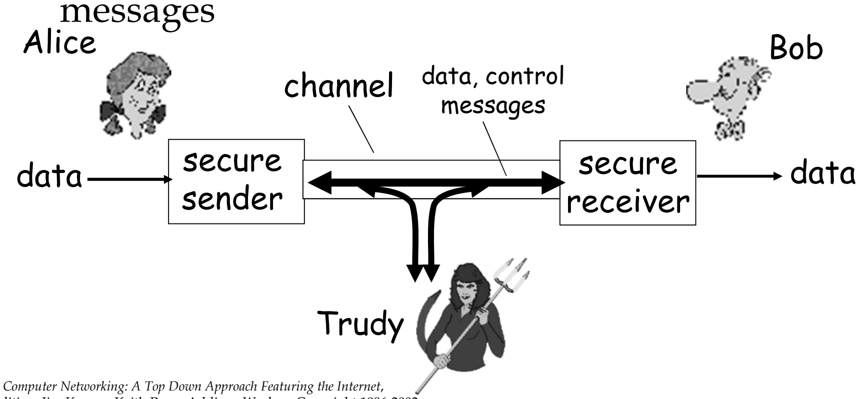
Modified from: https://www.securityweek.com/fbi-publishes-indicators-compromise-ranzy-locker-ransomware

Threats: How

- Passive threats
  - Interception
  - Release of contents
  - Traffic analysis, e.g., learn the location of the headquarters
- Active threats
  - Denial of services
  - Modification
  - Masquerade (Authenticity)
  - Ransomware

Framework for Discussion:
Friends and enemies: Alice, Bob, Trudy

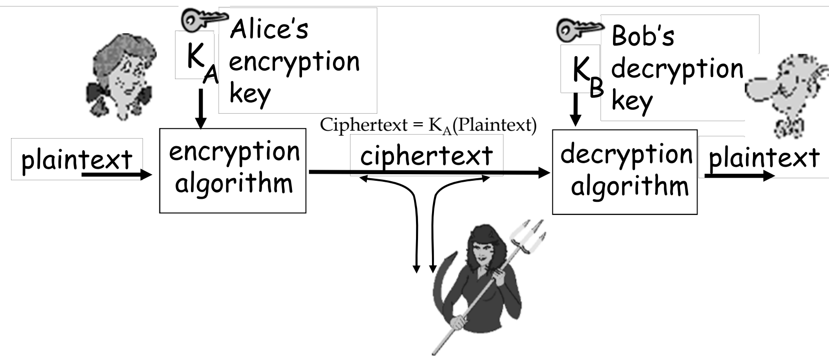
- Bob, Alice want to communicate "securely"
- Trudy (intruder) may intercept, delete, add messages



Who might Bob, Alice be?

- Web browser/server for electronic transactions (e.g., on-line purchases)
- On-line banking client/server
- DNS servers
- Routers exchanging routing table updates
- OpenFlow messages
- Other examples?

The language of cryptography



→**<u>Symmetric key</u>** crypto: sender, receiver keys *identical*

→**<u>Public-key</u>** crypto: encryption key *public*, decryption key *secret* (private)

Network Security
Encryption

- Encryption does:
  - Provides secrecy
  - Prevents tampering
  - Prevents forgery
- Encryption does not:
  - Keep attacker from deleting/encrypting files
  - Keep attacker from denying service
    (Distributed Denial of Service - DDoS)
- Security is more than encryption

Encryption Issues

- Secrecy of the key
- Preventing successful key search
- Breaking the encryption algorithm
- No back doors, i.e., ways to decrypt the file without knowing the key
- Give a partial decrypt message the ability to decrypt the entire file
- Attacks:
  - Ciphertext-only
  - Known-plaintext
  - Chosen plain text

Symmetric key cryptography

substitution cipher: substituting one thing for another

- monoalphabetic cipher: substitute one letter for another

```
plaintext:  abcdefghijklmnopqrstuvwxyz

ciphertext: mnbvcxzasdfghjklpoiuytrewq
```

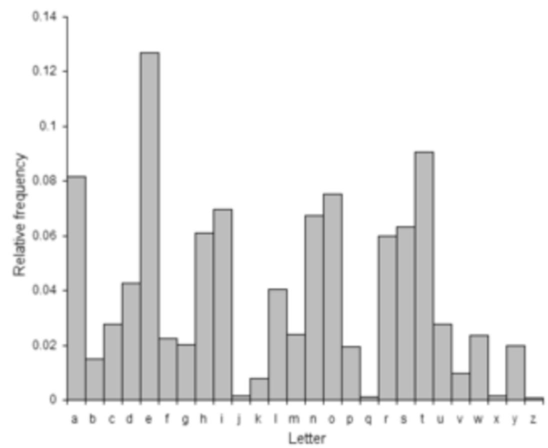
E.g.:
```
 Plaintext: bob. i see you. alice
ciphertext: nkn. s icc wky. mgsbc
```

Q: How hard to break this simple cipher?:

- brute force (how hard?)

The frequency of the letters of the alphabet in English

E	11.1607%	56.88	M	3.0129%	15.36
A	8.4966%	43.31	H	3.0034%	15.31
R	7.5809%	38.64	G	2.4705%	12.59
I	7.5448%	38.45	B	2.0720%	10.56
O	7.1635%	36.51	F	1.8121%	9.24
T	6.9509%	35.43	Y	1.7779%	9.06
N	6.6544%	33.92	W	1.2899%	6.57
S	5.7351%	29.23	K	1.1016%	5.61
L	5.4893%	27.98	V	1.0074%	5.13
C	4.5388%	23.13	X	0.2902%	1.48
U	3.6308%	18.51	Z	0.2722%	1.39
D	3.3844%	17.25	J	0.1965%	1.00
P	3.1671%	16.14	Q	0.1962%	(1)



From: https://www3.nd.edu/~busiforc/handouts/cryptography/letterfrequencies.html

Network Security
Encryption

- A simple encryption algorithm: the one-time pad

$$C = P \oplus K \qquad \oplus = EOR$$

Where P is a binary representation of the plain text and K is a random binary key the same length in bits as P.

Let
$P = 011 \quad and \quad K = 101$
then
$C = 110.$
To decrypt
$P = C \oplus K$

Network Security
Encryption: Problems with one-time pad

- Key can be used only once
- The key must be *random*
- The key must be of the same length as the plaintext
- Encryption, Episode 1- SIGSALY: AT&T Labs

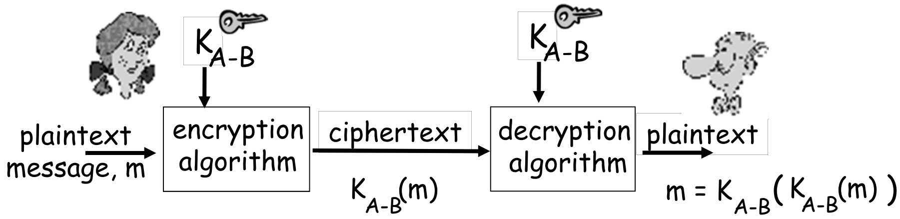
Pioneering cryptanalyst

- Elizebeth Smith Friedman

  (August 26, 1892 – October 31, 1980)
- https://en.wikipedia.org/wiki/Elizebeth_Smith_Friedman
- Book

  Code Girls: The Untold Story of the American Women Code Breakers of World War II by Liza Mundy

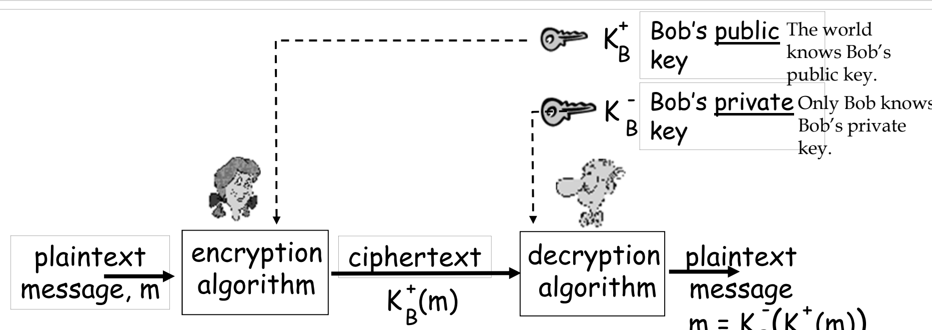
Symmetric key cryptography



symmetric key crypto: Bob and Alice share (know) same (symmetric) key: K_{A-B}

- e.g., key is knowing substitution pattern in mono alphabetic substitution cipher
- <u>Problem:</u> how do Bob and Alice agree on key value?

Public key cryptography



One key used for encryption while a different one is used for decryption

Public key encryption algorithms

Requirements:

1. need $K_B^+(\cdot)$ and $K_B^-(\cdot)$ such that $K_B^-(K_B^+(m)) = m$

2. given public key K_B^+, it should be impossible to compute private key K_B^-

RSA: Rivest, Shamir, Adleman algorithm is commonly used

An important property of RSA algorithm:

The following property will be ***very*** useful later:

$$\underbrace{K_B^-(K_B^+(m))}_{\text{use public key first, followed by private key}} = m = \underbrace{K_B^+(K_B^-(m))}_{\text{use private key first, followed by public key}}$$

Result is the same!

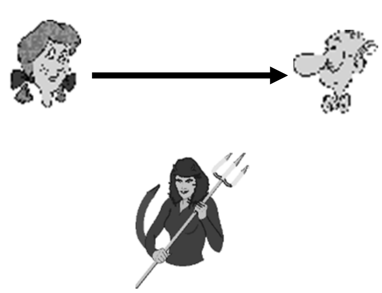
Authentication

- Process of proving one's identity
- Consider real-time interaction
- Approach: look at a series of Authentication Protocols

Authentication

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

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

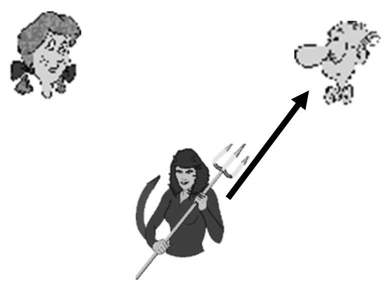


Failure scenario??

Authentication

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

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



in a network,
Bob can not "see"
Alice, so Trudy simply
declares
herself to be Alice

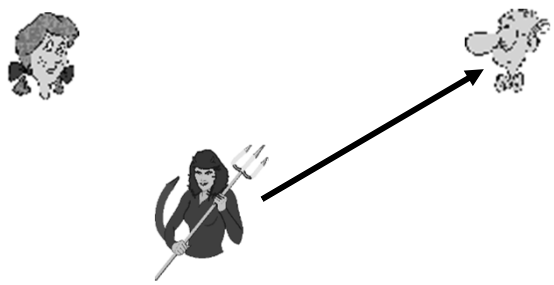
Authentication: another try

Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address

Failure scenario??

Authentication: another try

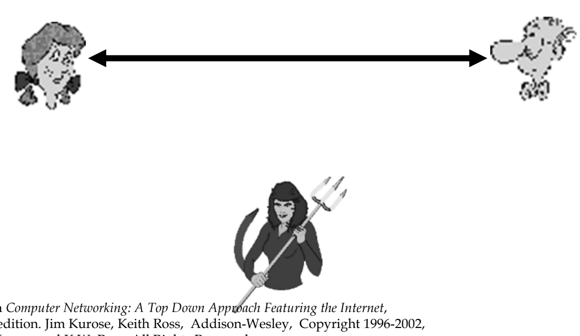
Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address



Trudy can create a packet "spoofing" Alice's address

Authentication: another try

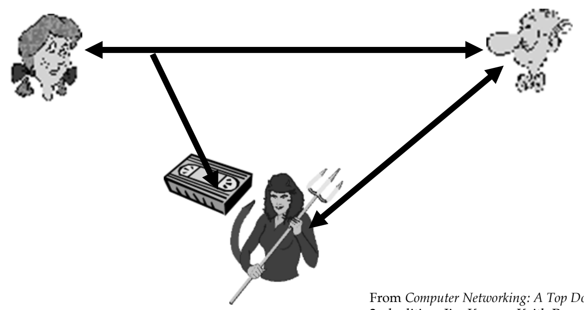
Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



Failure scenario??

Authentication: another try

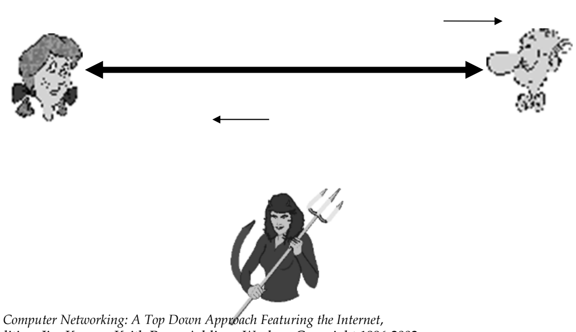
Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



playback attack: Trudy records Alice's packet and later plays it back to Bob

Authentication: yet another try

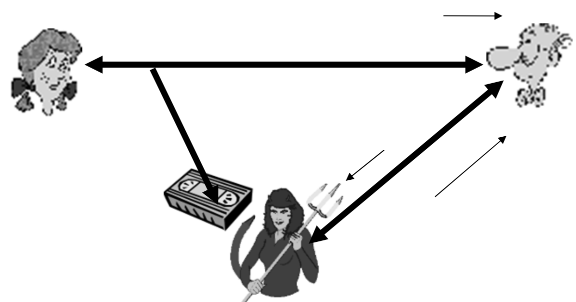
Protocol ap3.1: Alice says "I am Alice" and sends her *encrypted* secret password to "prove" it. Alice and Bob share a private Key $K_{A-B}(m)$



Failure scenario??

Authentication: another try

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



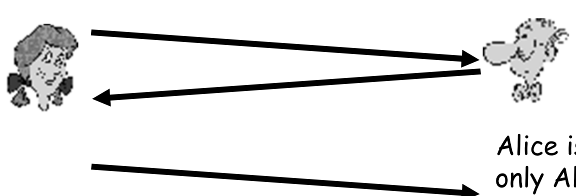
record and playback still works!

Authentication: yet another try

Goal: avoid playback attack

Nonce: number (R) A nonce used only *once –in-a-lifetime*

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

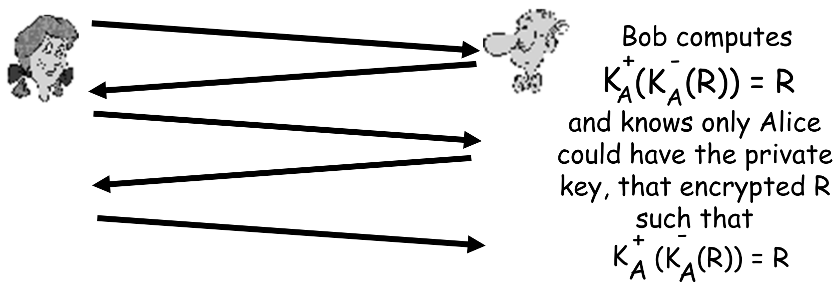


Alice is live, and only Alice knows key to encrypt nonce, so it must be Alice!

Authentication: ap5.0

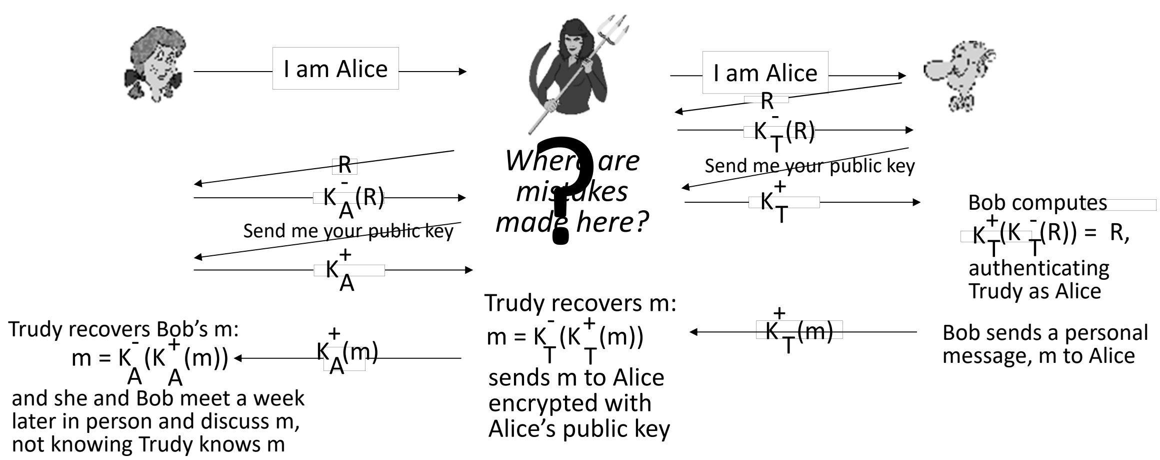
ap4.0 requires shared symmetric key
- can we authenticate using public key techniques?

ap5.0: use nonce, public key cryptography



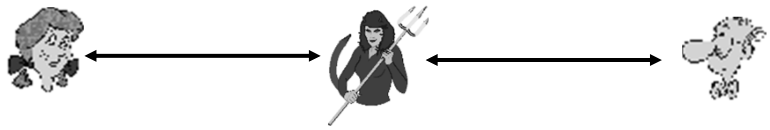
Authentication: ap5.0 – there's still a flaw!

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



ap5.0: security hole

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



Difficult to detect:

- Bob receives everything that Alice sends, and vice versa. (e.g., so Bob, Alice can meet one week later and recall conversation)
- Problem is that Trudy receives all messages as well!
- To defeat this Bob needs a secure way (**trusted third party**) of getting Alice's public key K_A^+

Trusted Intermediaries
(a trusted third party)

Symmetric key problem:

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

Solution:

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

Public key problem:

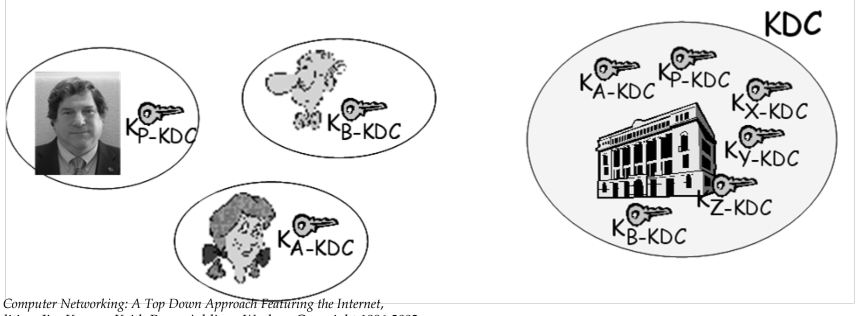
- When Alice obtains Bob's public key (from web site, e-mail, diskette), how does she know it is Bob's public key, not Trudy's?

Solution:

- trusted certification authority (CA)

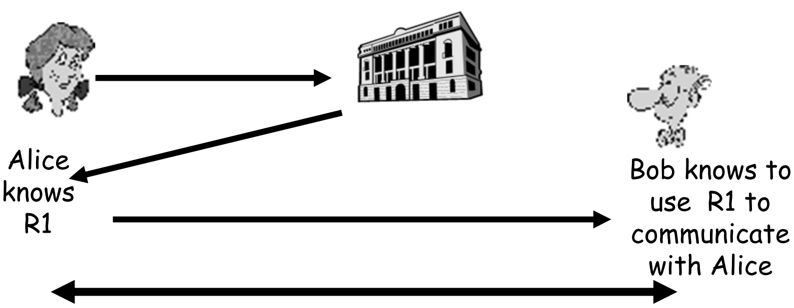
Key Distribution Center (KDC)

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



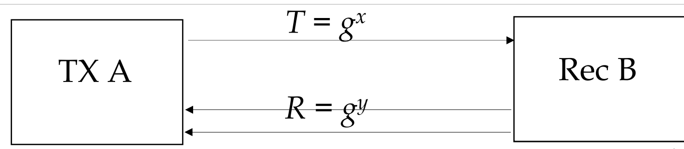
Key Distribution Center (KDC)

<u>*Q:*</u> How does KDC allow Bob, Alice to determine shared symmetric secret key to communicate with each other?



Alice and Bob communicate: using R1 as *session key* for shared symmetric encryption

Key Generation: Diffie-Hellman Exchange



$K = R^x \bmod p$ $= g^{xy} \bmod p$

$K = T^y \bmod p$ $= g^{xy} \bmod p$

- Generate keys instead of distributing keys
- Diffie-Hellman exchange to *create* a shared key
- A & B pick p a large prime #, and generator $g < p$
  - A picks x and sends $T = g^x$ to B; B picks y and sends $R = g^y$
  - Secret key is $K = (g^x)^y = (g^y)^x$ which are calculated by A & B
- Eavesdropper that obtains p, g, T, R cannot obtain x and y because $x = logT$ and $y = logR$ are extremely difficult to solve

Network Security
Kerberos (The three headed watch dog of Hades)

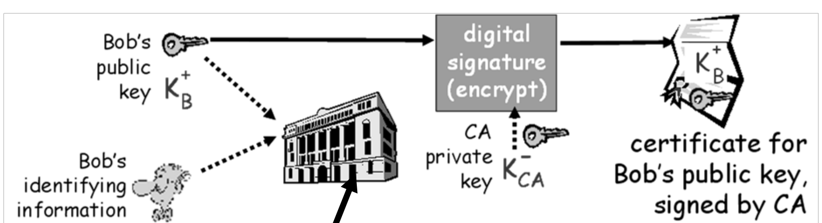
- Kerberos is an authentication system that uses a KDC
- Add on to an existing network protocol
- Users get access to services via KDC

Certificate Authorities (CAs)

- Public key authentication
  - Suppose you want to support EECS with 1,000 accounts
  - Requires 1,000 public keys
  - You have to remember 999 public keys of others in the department
  - You have to learn about 250 new public keys per year and forget about 250 public keys per year
  - Public key cryptography requires you maintain 999 public keys of others in the department
  - If you want to change your keys (either compromised or you're paranoid) you have to notify 999 others, privately
- Since public keys are 'public' you can publish on a 'directory service' or <u>Certification Authority</u> (CA)

From: "Information Security" G. Minden

What is a Certificate



Certificate Authority
a
Trusted
Institution

Certificate
Contains
Bob's
Public Key

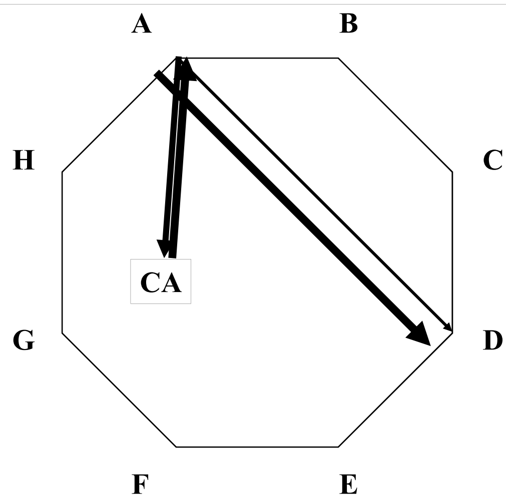
From *Computer Networking: A Top Down Approach Featuring the Internet*, 2nd edition. Jim Kurose, Keith Ross, Addison-Wesley, Copyright 1996-2002,

Certificates

Username: Alice
Public Key: A97E2345CD76ACB62...
Expires: 31-Dec-2000 23:59 Z
Authority: The University of Kansas EECS
Signed: 213458ABEDCDEB63C2B1FFF8695...

- Alice obtains certificate from EECS Department
- Alice presents certificate to Bob stating her identity
- Bob checks certificate signature against EECS CA public key
- If signature matches, Bob accepts Alice’s certificate and her public key
- Bob only needs to know CA’s public key to operate

Certificate Authority



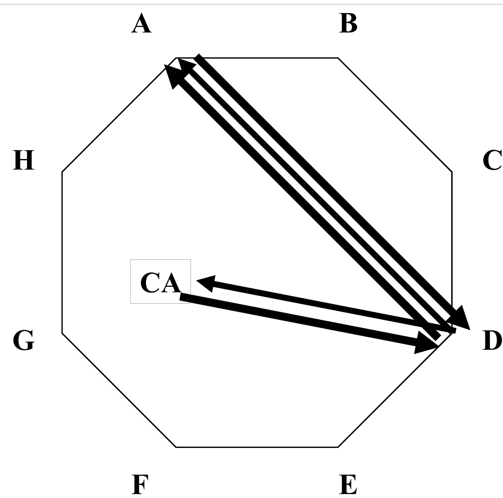
1. A wants to communicate with D

2. A requests D’s public key

3. CA sends D’s public key

4. A uses D’s public key to encrypt data and communicate

Certificate Authority



1. D wants to authenticate A

2. D requests A's public key

3. CA sends A's public key

4. D sends challenge to A encrypted with A's public key

5. A decrypts challenge with private key and responds

Certificate Authority

- You maintain one public key with CA
- You can change public key at any time
- Use public/private key pair for communications, no session key
- CA only has public key so cannot impersonate any user
- CA is a single point of failure
- CA is system bottleneck
- Do you trust the public key the CA send you?

Certificate Management

- CA need not be on-line, certificates can be generated on the CA but distributed via 'sneaker net'
- If CA were not available, it does not prevent system from operating
  - New users cannot be added
  - Established users eventually timeout
- Certificates are not security-sensitive
  - If I have a copy of your certificate, I cannot impersonate you because I do not have your private key
  - A saboteur cannot write bogus certificates because they do not have the CA's private key
- A compromised CA cannot decrypt private conversations since it does not have the private keys and/or session keys

Certificate Revocation

- Certificates carry an expiration date
  - Expiration dates can be extended by re-issuing the certificate without changing the public/private key
- CAs issue revocation lists (CRLs) when someone leaves the system or a key is compromised
- Services need to check CRL before honoring certificate

Common Certificate Authorities

- Symantec (used to be VeriSign)
- Comodo,
- Let's Encrypt (non-profit CA)

Integrity: Digital Signatures

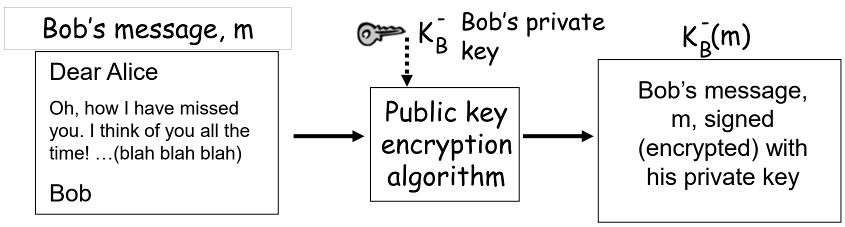
Cryptographic technique analogous to hand-written signatures.

- sender (Bob) digitally signs document, establishing he is document owner/creator.
- verifiable, non-repudiable, non-forgeable: recipient (Alice) can prove to someone that Bob, and no one else (including Alice), must have signed document

Digital Signatures

Simple digital signature for message m:

- Bob signs m by encrypting with his private key K_B^-, creating "signed" message, $K_B^-(m)$



Digital Signatures

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

Alice thus verifies that:

- ✓ Bob signed m.
- ✓ No one else signed m.
- ✓ Bob signed m and not m′.

Non-repudiation:

- ✓ Alice can take m, and signature $K_B^-(m)$ to court and prove that Bob signed m.

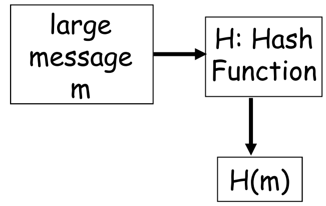
Problem: Computationally expensive to public-key-encrypt long messages

Message Digests

Solution: Use Hash Function

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

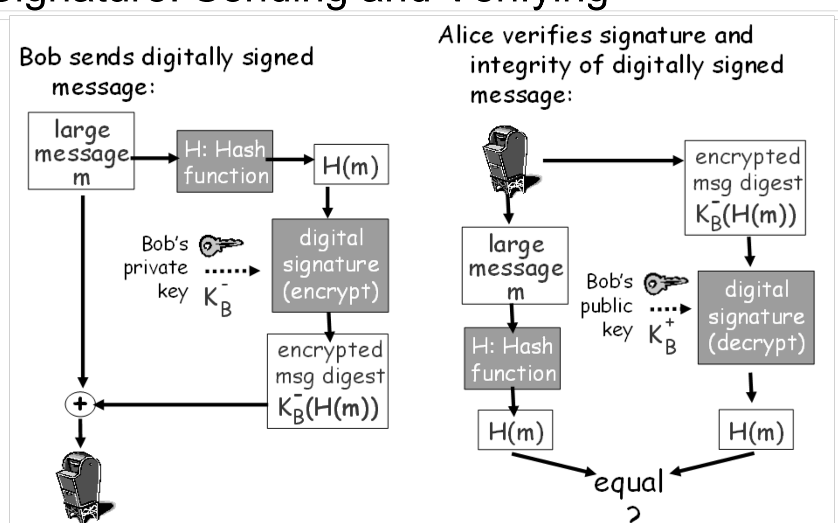
- apply hash function H to *m*, get fixed size message digest, *H(m)*.



Hash function properties:

- many-to-1
- produces fixed-size msg digest (fingerprint)
- given message digest x, computationally infeasible to find m such that x = H(m)

Digital Signature: Sending and Verifying



Network Security at which layer?

- Application
  - All payload is protected;
  - Maximum user control;
  - In general, OS does not see payload;
  - Source/destination/services visible;
  - Header is in the clear
- Transport
  - Applications share security infrastructure; service can be hidden;
  - Applications still need modifications to access security features;
  - ‘standards’ process (IETF)→Transport Layer Security (TLS)

From: “Information Security” G. Minden

Network Security at which layer?

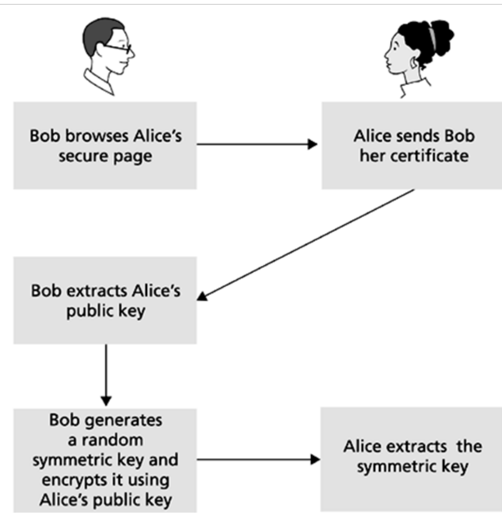
- Network
  - Multiple transport and applications share security mechanisms;
  - Can do some source/destination hiding in VPNs; difficult to handle per-user properties (non-repudiation, traffic flows);
  - ‘standards’ process (IETF)→IPsec
- Physical/Data Link
  - Difficult to implement;
  - Faster; subject to transmission errors (synchronization);
  - Key management difficult;
  - Dedicated point-to-point links

From: “Information Security” G. Minden

Transport Layer Security (TLS)

- transport layer security to any TCP-based app using TLS services.
- used between Web browsers, servers for e-commerce (https).
- security services:
  - server authentication
  - data encryption
  - client authentication (optional)
- server authentication:
  - TLS-enabled browser includes public keys for trusted CAs.
  - Browser requests server certificate, issued by trusted CA.
  - Browser uses CA's public key to extract server's public key from certificate.
- check your browser's security menu to see its trusted CAs.

High-level overview of SSL

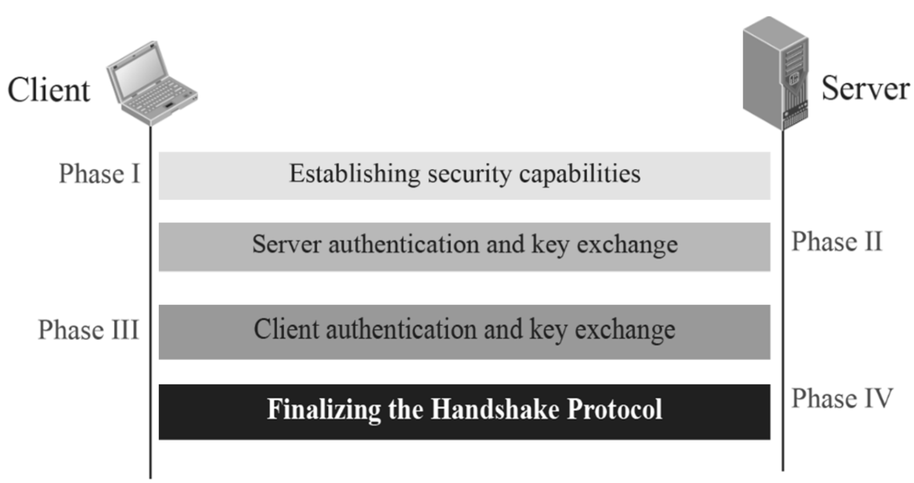


TLS (continued)

Encrypted TLS session:

- Browser generates *symmetric session key*, encrypts it with server's public key, sends encrypted key to server.
- Using private key, server decrypts session key.
- Browser, server know session key
  - All data sent into TCP socket (by client or server) encrypted with session key.
- TLS can be used for non-Web applications, e.g., Internet Message Access Protocol (IMAP), e.g., e-mail
- Client authentication can be done with client certificates.
- The TCP header and payload are encrypted by TLS.

Handshake Protocol



From: Data Communications and Networking 5th Edition by Behrouz A. Forouzan and Computer and Communication Networks, 2nd Edition. Nader F. Mir. Prentice Hall

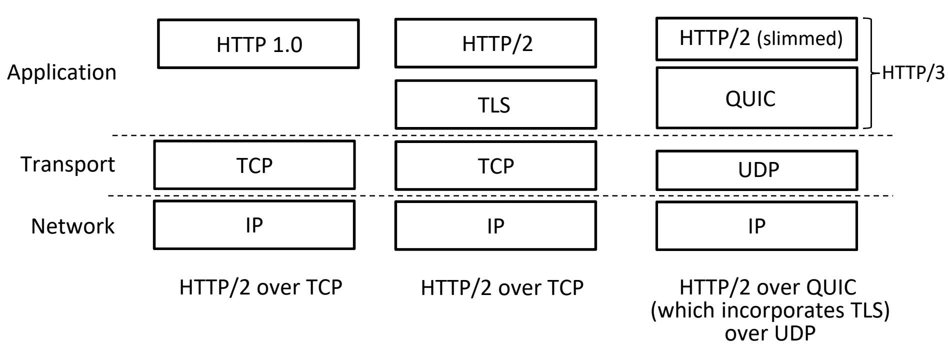
Quick UDP Internet Connections (QUIC)

- QUIC is transport layer protocol developed by Google to improve web page loading times and security. It operates on top of UDP
- Encrypts data by default, providing confidentiality and integrity. This encryption protects against eavesdropping and tampering with the transmitted data.
- Supports forward secrecy, which means that even if an attacker were to compromise the server's private key, they would not be able to decrypt past communications.
- Includes mechanisms for endpoint authentication to ensure that clients are communicating with the intended server and vice versa, mitigating the risk of person-in-the-middle attacks
- Allows multiple streams of data to be sent over a single connection, which improves efficiency. Each stream is independently encrypted, providing isolation between different streams and enhancing security
- "QUIC is an application-layer protocol providing encrypted, reliable, congestion-controlled data transfer between two endpoints." From COMPUTER NETWORKINGA Top-Down Approach EIGHTH EDITION

Modified from ChatGPT

Transport-layer security (TLS)

- TLS provides an API that *any* application can use
- an HTTP view of TLS:



Modified from: 8th edition Jim Kurose, Keith Ross Pearson, 2020

IETF IPSec

- Provides security at Network layer
- Provides per-flow or per-connection security

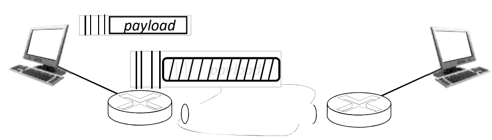
- **Original IP Packet**
  - **IP Header**
  - **TCP Header**
  - **Payload**

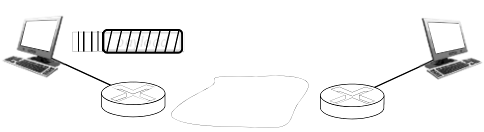
IPHdr	TCPHdr	Data

From: "Information Security" G. Minden

IP Sec

- provides datagram-level encryption, authentication, integrity
  - for both user traffic and control traffic (e.g., BGP, DNS messages)
- two "modes":



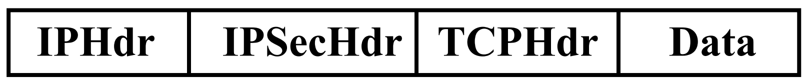


transport mode:
- *only* datagram *payload* is encrypted, authenticated

tunnel mode:
- entire datagram is encrypted, authenticated
- encrypted datagram encapsulated in new datagram with new IP header, tunneled to destination

Modified from: 8th edition Jim Kurose, Keith Ross Pearson, 2020

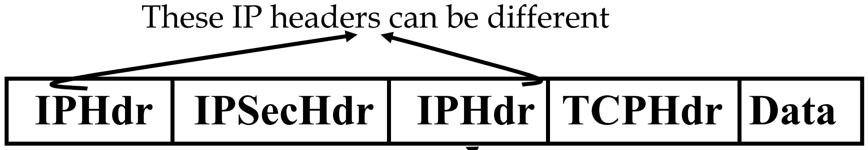
IPSec Modes – Transport Mode



- **Transport Protection**
  - **IP Header**
  - **IPSec Header TCP Header**
  - **Protected Payload**

IPSec Modes – Tunnel Mode

These IP headers can be different



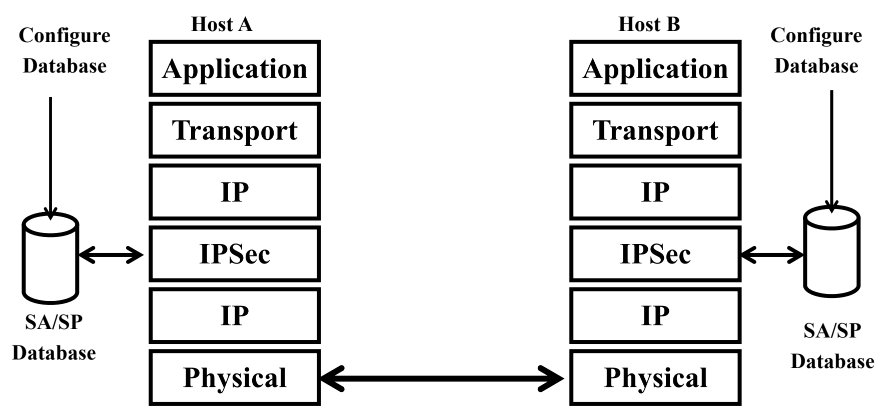
- **Tunnel mode**
  - **IP Header**
  - **IPSec Header IP Header (protected)**
  - **TCP Header (protected)**
  - **Payload (protected)**

IPSec Policy

- Need to associate a key with a transmitted packet
- Called a Security Association (SA)
  - Unidirectional
- SA Policies
  - Select defined flows or connections
  - Reside in a Security Policy Database (SPDB)
  - Several actions: discard, bypass, protect
  - Applicable policy selected by ‘selectors’
  - SA Policies point to SA

From: “Information Security” G. Minden

IPSec Model



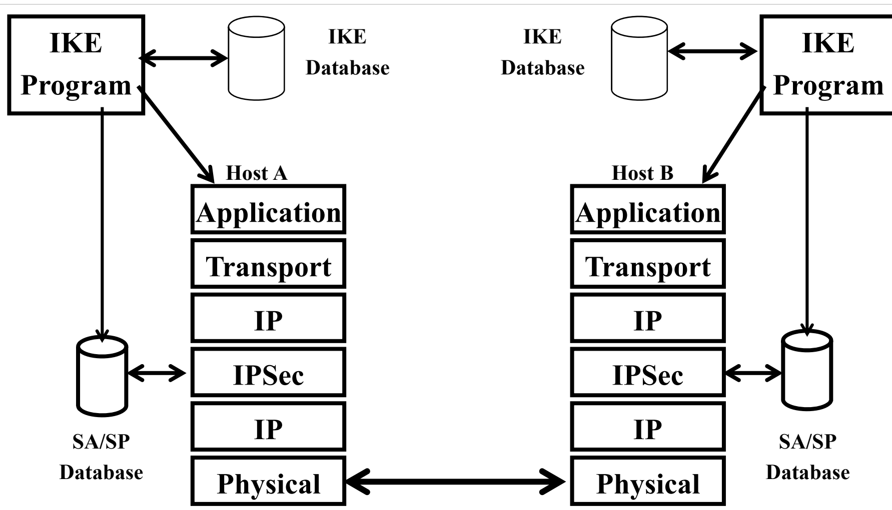
SA= Security Association
SP=Security Policy

From: “Information Security” G. Minden

IPSec: Key Management-- Internet Key Exchange (IKE)

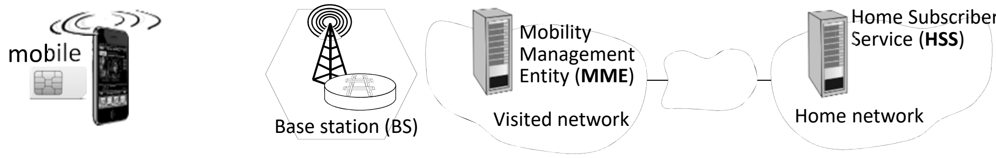
- IKE facilitates secure and automated key management.
- It ensures that both the client and server agree on a set of security parameters, including
  - encryption algorithms,
  - integrity algorithms,
  - Diffie-Hellman groups,
  - authentication methods,
- IPsec peers can be two end systems, two routers/firewalls, or a router/firewall and an end system

IPSec: Key Management-- Internet Key Exchange (IKE)



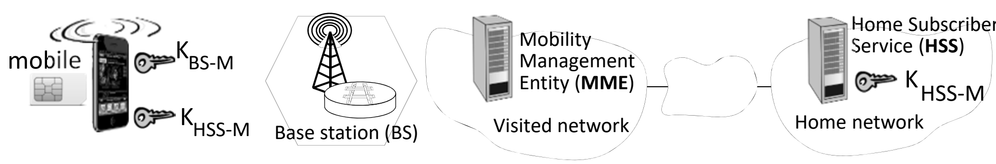
From: "Information Security" G. Minden

Authentication, encryption in 4G LTE



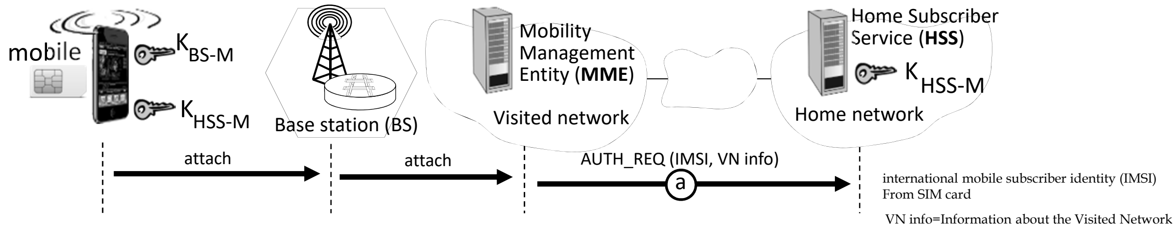
- arriving mobile must:
  - associate with BS: (establish) communication over 4G wireless link
  - Authenticate mobile to network, and authenticate network to the moble
  - Two-way authentication known as **mutual authentication** (WiFi requires the same functionality)
- notable differences from WiFi
  - mobile's SIMcard provides global identity, contains shared keys
  - services in visited network depend on (paid) service subscription in home network

Authentication, encryption in 4G LTE



- mobile, BS use derived session key K_{BS-M} to encrypt communications over 4G link
- MME in visited network + HHS in home network, together play role of WiFi AS
  - ultimate authenticator is HSS
  - trust and business relationship between visited and home networks

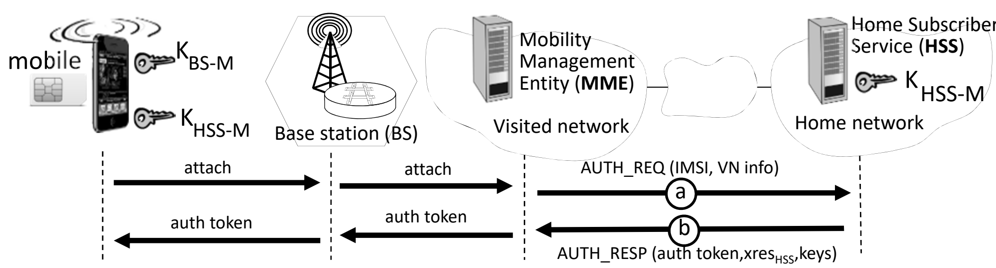
Authentication, encryption in 4G LTE



ⓐ authentication request to home network HSS

- mobile sends attach message (containing its IMSI, visited network info) relayed from BS to visited MME to home HHS
- IMSI identifies mobile's home network

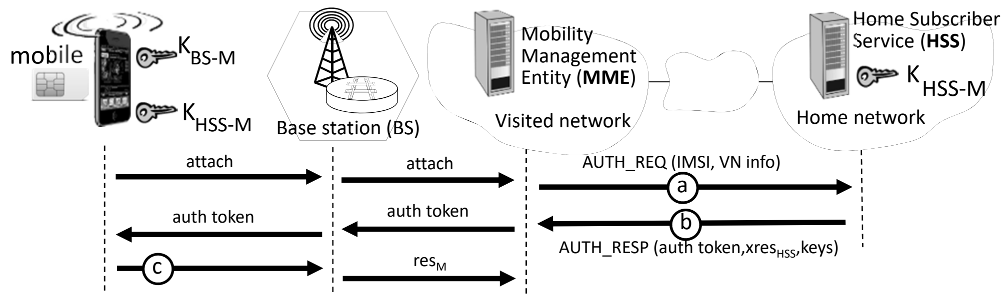
Authentication, encryption in 4G LTE



ⓑ HSS use shared-in-advance secret key, K_{HSS-M}, to derive authentication token, *auth_token*, and expected authentication response token, *$xres_{HSS}$*

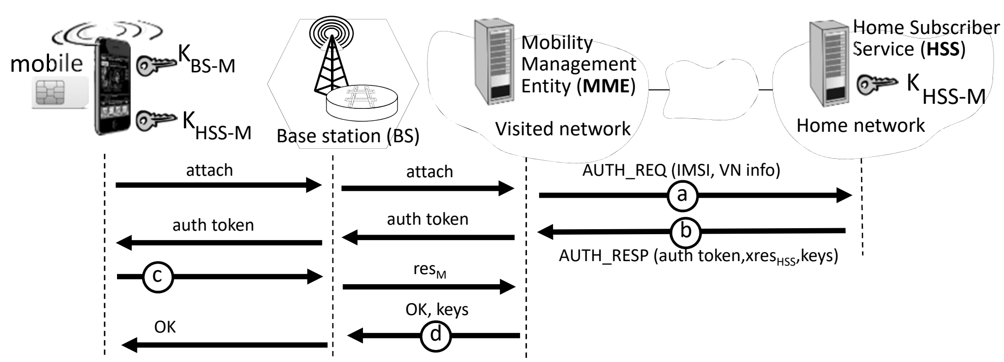
- *auth_token* contains info encrypted by HSS using K_{HSS-M} , allowing mobile to know that whoever computed *auth_token* knows shared-in-advance secret
- mobile has authenticated network
- visited HSS keeps *$xres_{HSS}$* for later use
- Here a token is a nonce

Authentication, encryption in 4G LTE



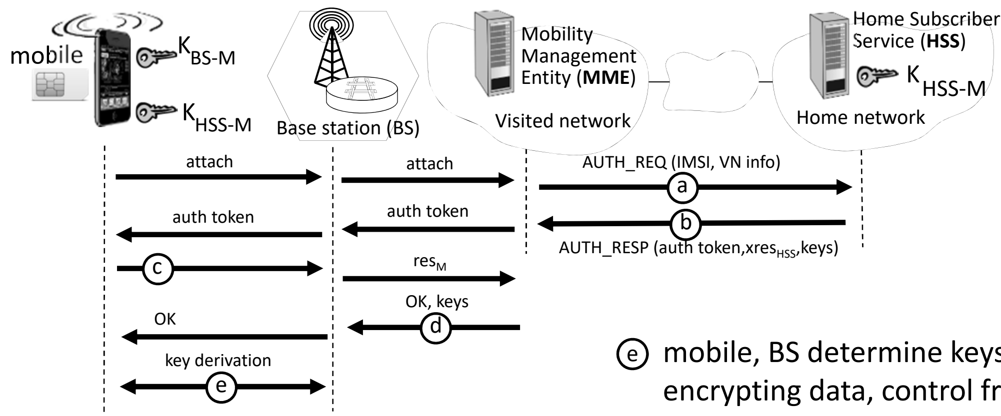
ⓒ authentication response from mobile:
- mobile computes res_M using its secret key to make same cryptographic calculation that HSS made to compute $xres_{HSS}$ and sends res_M to MME

Authentication, encryption in 4G LTE



ⓓ mobile is authenticated by network:
- MMS compares mobile-computed value of res_M with the HSS-computed value of $xres_{HSS}$. If they match, mobile is authenticated ! (why?)
- MMS informs BS that mobile is authenticated, generates keys for BS

Authentication, encryption in 4G LTE



e mobile, BS determine keys for encrypting data, control frames over 4G wireless channel

- AES can be used

Firewalls: why

prevent denial of service attacks:

- SYN flooding: attacker establishes many bogus TCP connections, no resources left for "real" connections

prevent illegal modification/access of internal data

- e.g., attacker replaces CIA's homepage with something else

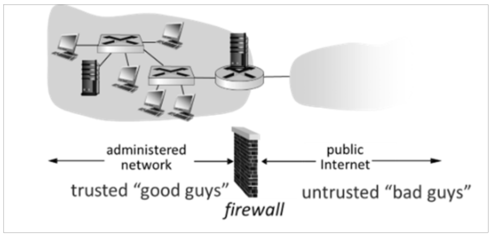
allow only authorized access to inside network

- set of authenticated users/hosts

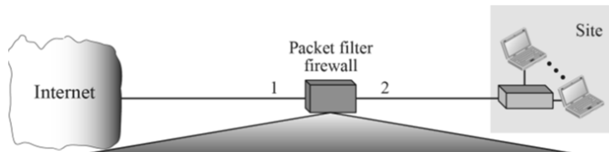
three types of firewalls:

- stateless packet filters
- stateful packet filters
- application gateways

Firewalls- Stateless packet filtering



- Internal network connected to Internet via router firewall
- Router filters packet-by-packet, decision to forward/drop packet based on:
  - source IP address, destination IP address
  - TCP/UDP source and destination port numbers
  - ICMP message type
  - TCP SYN and ACK bits



Interface	Source IP	Source port	Destination IP	Destination port
1	131.34.0.0	*	*	*
1	*	*	*	23
1	*	*	194.78.20.8	*
2	*	*	*	80

*=any

- On interface 1 all packets coming from 131.34.0.0 will be blocked
- On interface 1 all packets with destination port 23 will be blocked
- On interface 1 all packets with destination IP 194.78.20.8 will be blocked
- On interface 2 all packets with destination port 80 will be blocked

From: Data Communications and Networking 5th Edition by Behrouz A. Forouzan and Computer and Communication Networks, 2nd Edition. Nader F. Mir. Prentice Hall

From: *Computer Networking: A Top Down Approach Featuring the Internet*, 2nd edition. Jim Kurose, Keith Ross, Addison-Wesley, Copyright 1996-2002, J .F Kurose and K.W. Ross, All Rights Reserved

Stateless packet filtering: more examples

Policy	Firewall Setting
no outside Web access	drop all outgoing packets to any IP address, port 80
no incoming TCP connections, except those for institution's public Web server only.	drop all incoming TCP SYN packets to any IP except 130.207.244.203, port 80
prevent Web-radios from eating up the available bandwidth.	drop all incoming UDP packets - except DNS and router broadcasts.
prevent your network from being used for a smurf DoS attack.	drop all ICMP packets going to a "broadcast" address (e.g. 130.207.255.255)
prevent your network from being tracerouted	drop all outgoing ICMP TTL expired traffic

*The Smurf DoS attack is a distributed DoS where large numbers of ICMP packets with the intended victim's spoofed source IP are broadcast to a computer network using an IP broadcast address.

Modified from: 8th edition Jim Kurose, Keith Ross Pearson, 2020

Access Control Lists (ACL)

ACL: table of rules, applied top to bottom to incoming packets: (action, condition) pairs: looks like OpenFlow forwarding (Ch. 4)!

action	source address	dest address	protocol	source port	dest port	flag bit
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	---
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	----
deny	all	all	all	all	all	all

Stateful packet filtering

- *stateless packet filter:* heavy handed tool
  - admits packets that "make no sense," e.g., dest port = 80, ACK bit set, even though no TCP connection established:

action	source address	dest address	protocol	source port	dest port	flag bit
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK

- *stateful packet filter:* track status of every TCP connection
  - track connection setup (SYN), teardown (FIN): determine whether incoming, outgoing packets "makes sense"
  - timeout inactive connections at firewall: no longer admit packets

Stateful packet filtering

ACL augmented to indicate need to check connection state table before admitting packet

action	source address	dest address	proto	source port	dest port	flag bit	check connection
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any	
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK	X
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	---	
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	----	X
deny	all	all	all	all	all	all	

Security Summary

→Basic techniques
- Cryptography (symmetric and public)
- Authentication
- Message integrity
- Key distribution

→Used in many different security scenarios
- Secure transport (TLS)
- IPSec
- Link layer
  - Encryption in 4G
  - WiFi (see book)

→Firewalls

What is network management?

- autonomous systems (aka “network”): 1000s of interacting hardware/software components
- other complex systems requiring monitoring, configuration, control:
  - jet airplane, nuclear power plant, others?
- Network management is accomplished through the “Management Plane”
  - Data, Control, management planes
- Network management is coordinated at a Network Operations Center (NOC)
- Key Terms
  - Management Information Base (MIB)
  - Simple Network Management Protocol (SNMP)

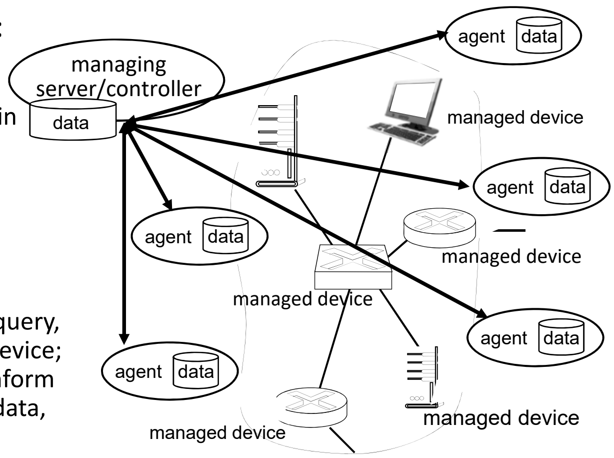
"Network management includes the deployment, integration and coordination of the hardware, software, and human elements to monitor, test, poll, configure, analyze, evaluate, and control the network and element resources to meet the real-time, operational performance, and Class/Quality of Service requirements at a reasonable cost."

Network Operations Center (NOC)

Components of network management

Managing server: application, typically with network managers (humans, in NOC) in the loop

Network management protocol: used by managing server to query, configure, manage device; used by devices to inform managing server of data, events.



Managed device: equipment with manageable, configurable hardware, software components. Typically has a management interface, physical or virtual.

Data: device "state" configuration data, operational data, device statistics

Network operator approaches to management

CLI (Command Line Interface)

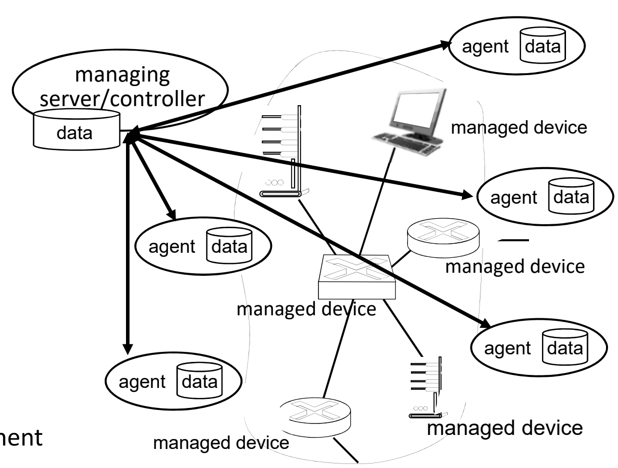
- operator issues (types, scripts) direct to individual devices (e.g., vis ssh)

SNMP/MIB

- operator queries/sets devices data (MIB) using Simple Network Management Protocol (SNMP)

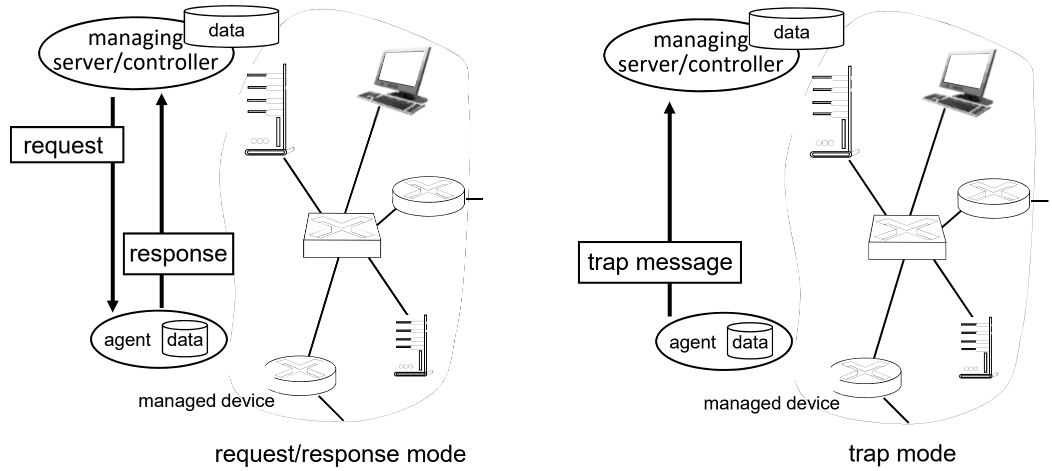
NETCONF:

- more abstract, network-wide, holistic
- emphasis on multi-device configuration management.
- NETCONF: communicate YANG-compatible actions/data to/from/among remote devices
- Yet Another Next Generation, YANG, is a data modeling language used in network management and device configuration.



SNMP protocol

Two ways to convey MIB info, commands:



SNMP protocol: message types

Message type	Function
GetRequest GetNextRequest GetBulkRequest	manager-to-agent: "get me data" (data instance, next data in list, block of data).
SetRequest	manager-to-agent: set MIB value
Response	Agent-to-manager: value, response to Request
Trap	Agent-to-manager: inform manager of exceptional event

SNMP protocol: message formats

message types 0-3

Get/set header				Variables to get/set				
PDU type (0-3)	Request ID	Error Status (0-5)	Error Index	Name	Value	Name	Value	….

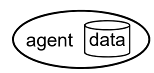
message type 4

	Trap header					Trap info		
PDU type 4	Enterprise	Agent Addr	Trap Type (0-7)	Specific code	Time stamp	Name	Value	….

SNMP PDU

SNMP: Management Information Base (MIB)

- managed device's operational (and some configuration) data



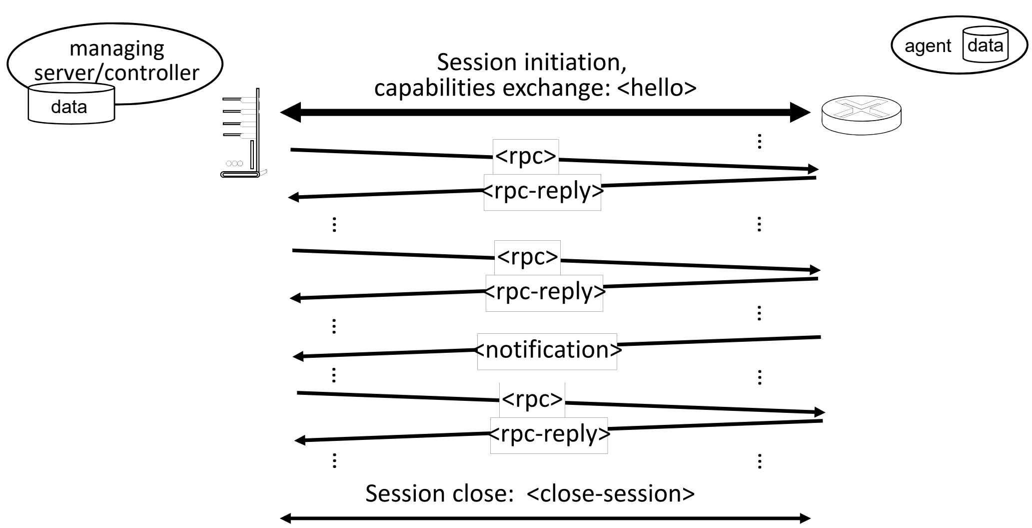
- gathered into device MIB module
  - 400 MIB modules defined in RFC's; many more vendor-specific MIBs
- Structure of Management Information (SMI): data definition language
- example MIB variables for UDP protocol:

Object ID	**Name**	**Type**	**Comments**
1.3.6.1.2.1.7.1	UDPInDatagrams	32-bit counter	total # datagrams delivered
1.3.6.1.2.1.7.2	UDPNoPorts	32-bit counter	# undeliverable datagrams (no application at port)
1.3.6.1.2.1.7.3	UDInErrors	32-bit counter	# undeliverable datagrams (all other reasons)
1.3.6.1.2.1.7.4	UDPOutDatagrams	32-bit counter	total # datagrams sent
1.3.6.1.2.1.7.5	udpTable	SEQUENCE	one entry for each port currently in use

NETCONF overview

- goal: actively manage/configure devices network-wide, emphasis on configuration management
- operates between managing server and managed network devices
  - actions: retrieve, set, modify, activate configurations
  - atomic-commit actions over multiple devices
  - query operational data and statistics
  - subscribe to notifications from devices
- remote procedure call (RPC) paradigm
  - NETCONF protocol messages encoded in XML
  - exchanged over secure, reliable transport (e.g., TLS) protocol

NETCONF initialization, exchange, close



Selected NETCONF Operations

NETCONF	Operation Description
<get-config>	Retrieve all or part of a given configuration. A device may have multiple configurations.
<get>	Retrieve all or part of both configuration state and operational state data.
<edit-config>	Change specified (possibly running) configuration at managed device. Managed device <rpc-reply> contains <ok> or <rpcerror> with rollback.
<lock>, <unlock>	Lock (unlock) configuration datastore at managed device (to lock out NETCONF, SNMP, or CLIs commands from other sources).
<create-subscription>, <notification>	Enable event notification subscription from managed device

Network Management Enables

- Fault Detection and Troubleshooting
- Performance Optimization
- Security Management
- Configuration and Change Management
- Scalability and Capacity Planning
- Compliance and Reporting (logging)