

# EEC-484/584 Computer Networks

## Lecture 25

Wenbing Zhao

wenbing@ieee.org

(Lecture notes are based on materials supplied by  
Dr. Louise Moser at UCSB and Prentice-Hall)

## Outline

- Review of last lecture
- Today's topics
  - Authentication Protocols
  - Email Security
- **Final Exam: Dec 14, 6:00-8:00pm**
  - Chapters 7 and 8
- **Final hard deadline for project 1 and project 2 (improvement only): Dec 14, midnight**
- Records of your course work will be posted on the course Web site. If there is any mistake, please contact me as soon as possible

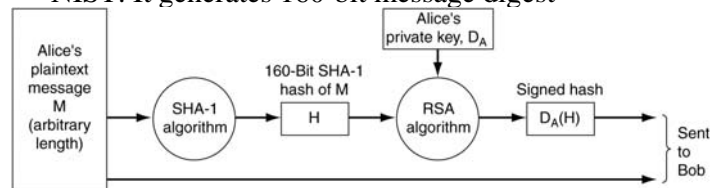
9 December 2005

EEC484/584

Wenbing Zhao

## Message Digests

- **MD5** is the fifth in a series of message digests designed by Ronald Rivest (1992). MD5 generates a 128-bit fixed value
- **SHA-1: Secure Hash Algorithm 1**, developed by National Security Agency (NSA) and blessed by NIST. It generates 160-bit message digest



9 December 2005

EEC484/584

Wenbing Zhao

## The Birthday Attack

- How many messages need to be generated to find a collision for an  $m$ -bit message digest?
  - Only  $2^{m/2}$ , **NOT**  $2^m$ , using the **birthday attack**
- **Birthday attack**. Generally, if there is some mapping between inputs and outputs with  $n$  inputs and  $k$  possible outputs,
  - There are  $n(n-1)/2$  input pairs
  - If  $n(n-1)/2 > k$ , the chance of having at least one match is pretty good.
  - Thus, approximately, a match is likely for  $n > \sqrt{k}$

9 December 2005

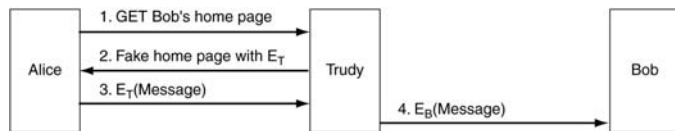
EEC484/584

Wenbing Zhao

## Problems with Public-Key Management

5

- If Alice and Bob do not know each other, how do they get each other's public keys to start the communication process ?
  - It is essential Alice gets Bob's public key, not someone else's
- A way for Trudy to subvert public-key encryption



9 December 2005

EEC484/584

Wenbing Zhao

## Certificates

6

- **Certification Authority (CA):** an organization that certifies public keys
  - It certifies the public keys belonging to people, companies, or even attributes
  - CA does not need to be on-line all the time
- A possible certificate and its signed hash

```

    I hereby certify that the public key
    19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A
    belongs to
    Robert John Smith
    12345 University Avenue
    Berkeley, CA 94702
    Birthday: July 4, 1958
    Email: bob@superdupernet.com

    _____
    SHA-1 hash of the above certificate signed with the CA's private key
    
```

9 December 2005

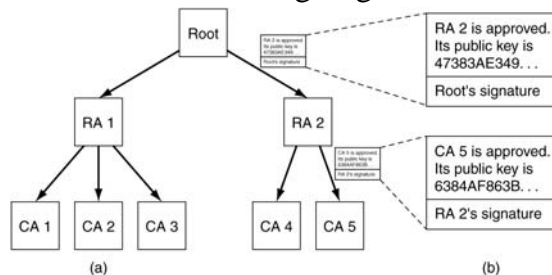
EEC484/584

Wenbing Zhao

## Public-Key Infrastructures

7

- Hierarchical PKI
- A **chain of trust/certification path:**  
A chain of certificates going back to the root



9 December 2005

EEC484/584

Wenbing Zhao

## IPsec

8

- **IPsec (IP security):** a solution for Internet security
- The complete IPsec design is a framework for multiple services, algorithms and granularities.
  - The major services are **secrecy, data integrity, and protection from replay attacks** (intruder replays a conversation).
  - All of these are based on **symmetric-key cryptography** because high performance is crucial.

9 December 2005

EEC484/584

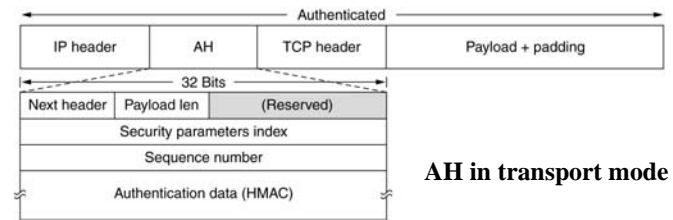
Wenbing Zhao

# IPsec

- **Security Association (SA):** A simplex connection between two end points and has a security identifier associated with it.
  - If secure traffic is needed in both directions, two security associations are required.
  - Security identifiers are carried in packets traveling on these secure connections and are used to look up keys and other relevant
- IPsec can be used in either of two modes
  - Transport mode and Tunnel mode

# IPsec – Authentication Header

- **AH (Authentication Header):** It provides integrity checking and antireplay security, but not secrecy (i.e., no data encryption)



AH in transport mode

# IPsec - AH header

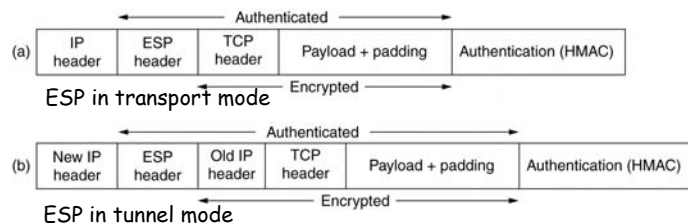
- The *Sequence number* field is used to number all the packets sent on an SA. **Every packet gets a unique number, even retransmissions**
  - The purpose of this field is to detect **replay attacks**
  - These sequence numbers may not wrap around. If all  $2^{32}$  are exhausted, a new SA must be established to continue communication

# IPsec - AH header

- The *Authentication data*, which is a variable-length field that contains the payload's digital signature
  - When the SA is established, the two sides negotiate which signature algorithm they are going to use and the shared key to use
  - **HMAC (Hashed Message Authentication Code):** Compute the hash over the packet plus the shared key. The shared key is not transmitted.
  - The integrity check covers some of the fields in the IP header, namely, those that do not change as the packet moves from router to router

## IPsec – ESP Header

- **ESP (Encapsulating Security Payload):** provides both authentication and confidentiality guarantee for packets. Can be used for both transport mode and tunnel mode



## 802.11 Security

- **WEP (Wired Equivalent Privacy):** a data link-level security protocol prescribed by 802.11 standard
  - It is designed to make the security of a wireless LAN as good as that of a wired LAN
- When 802.11 security is enabled, each station has a secret key shared with the base station
  - WEP encryption uses a stream cipher based on the RC4 algorithm
  - In WEP, RC4 generates a keystream that is XORed with the plaintext to form the ciphertext

## Authentication Protocols

- **Authentication** is the technique by which a process verifies that its communication partner is who it is supposed to be and not an imposter
  - Verifying the identity of a remote process in the face of a malicious, active intruder is surprisingly difficult and requires complex protocols based on cryptography
- Not to be confused with **authorization**
  - Authorization is concerned with what that process is permitted to do

## General Model for Authentication Protocols

- Alice starts out by sending a message either to Bob or to a trusted **KDC (Key Distribution Center)**, which is expected to be honest
- Several other message exchanges follow in various directions
- As these messages are being sent Trudy may intercept, modify, or replay them in order to trick Alice and Bob or just to gum up the works

## General Model for Authentication Protocols

17

- When the protocol has been completed, Alice is sure she is talking to Bob and Bob is sure he is talking to Alice
- Furthermore, in most of the protocols, the two of them will also have established a secret **session key** for use in the upcoming conversation
  - For each new connection, a new, randomly-chosen session key should be used
- Public-key cryptography is widely used for the authentication protocols themselves and for establishing the session key

9 December 2005

EEC484/584

Wenbing Zhao

## General Model for Authentication Protocols

18

- **Why use a session key ?**
  - For performance reasons, all data traffic is encrypted using symmetric-key cryptography (typically AES or triple DES)
  - To minimize the amount of traffic that gets sent with the users' secret keys or public keys
  - To reduce the amount of ciphertext an intruder can obtain
  - To minimize the damage done if a process crashes and its core dump falls into the wrong hands. Hopefully, the only key present then will be the session key
  - All the permanent keys should have been carefully zeroed out after the session was established

9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Protocols

19

- Authentication Based on a Shared Secret Key
- Establishing a Shared Key: Diffie-Hellman
- Authentication Using a Key Distribution Center
- Authentication Using Kerberos
- Authentication Using Public-Key Cryptography

9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Based on a Shared Secret Key

20

- Two-way authentication using a **challenge-response** protocol
  - **Challenge-response:** one party sends a random number to the other, who then transforms it in a special way and then returns the result
  - **Nonces:** random numbers used just once in challenge-response protocols
  - Assume that Alice and Bob already share a secret key,  $K_{AB}$

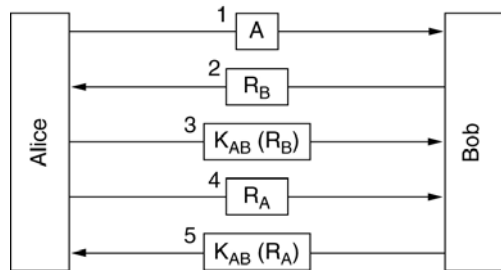
9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Based on a Shared Secret Key

21



9 December 2005

EEC484/584

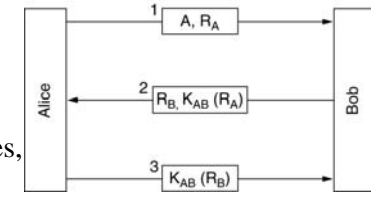
Wenbing Zhao

## Authentication Based on a Shared Secret Key

22

- A shortened two-way authentication protocol. Is this new protocol an improvement over the original one ?

- It is shorter
- But it is also wrong
- Under certain circumstances, Trudy can defeat this protocol by using what is known as a **reflection attack**



9 December 2005

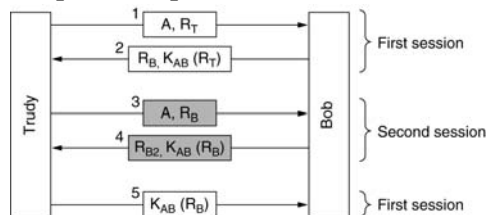
EEC484/584

Wenbing Zhao

## Reflection Attack

23

- **The reflection attack:** Trudy can break it if it is possible to open multiple sessions with Bob at once



- This attack can be defeated by encrypting  $R_B$  with  $K_{AB}$  in message 2

9 December 2005

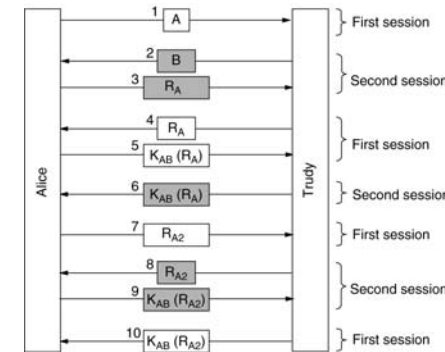
EEC484/584

Wenbing Zhao

## Reflection Attack

24

- A reflection attack on the original protocol can happen if Alice were a general-purpose computer that also accepted multiple sessions, rather than a person at a computer



9 December 2005

EEC484/584

Wenbing Zhao

## General Rules for Authentication Protocols Design

25

- Have the initiator prove who she is before the responder has to
  - In the previous case, Bob gives away valuable information before Trudy has to give any evidence of who she is
- Have the initiator and responder use different keys for proof, even if this means having two shared keys,  $K_{AB}$  and  $K'_{AB}$
- Have the initiator and responder draw their challenges from different sets. For example, the initiator must use even numbers and the responder must use odd numbers
- Make the protocol resistant to attacks involving a second parallel session in which information obtained in one session is used in a different one

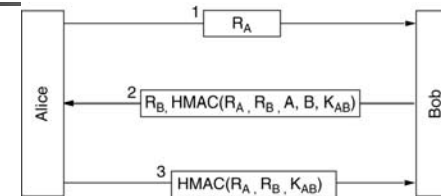
9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using HMACs

26



- She knows she is talking to Bob because Trudy does not know  $K_{AB}$  and thus cannot figure out which HMAC to send
- Trudy cannot subvert this protocol because **she cannot force either party to encrypt or hash a value of her choice**

9 December 2005

EEC484/584

Wenbing Zhao

## Establishing a Shared Key: The Diffie-Hellman Key Exchange

27

- Diffie-Hellman key exchange:** protocol that allows strangers to establish a shared secret key
  - Two large numbers,  $n$  and  $g$ , where  $n$  is a prime,  $(n - 1)/2$  is also a prime and certain conditions apply to  $g$ . These numbers may be public

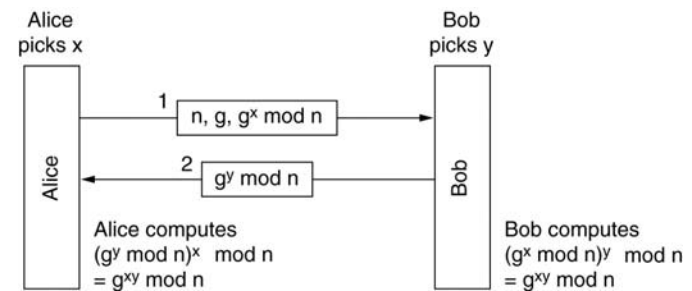
9 December 2005

EEC484/584

Wenbing Zhao

## Establishing a Shared Key: The Diffie-Hellman Key Exchange

28



9 December 2005

EEC484/584

Wenbing Zhao

## Establishing a Shared Key:

29

### The Diffie-Hellman Key Exchange

- Example:  $n = 47$  and  $g = 3$ . Alice picks  $x = 8$  and Bob picks  $y = 10$ . Both of these are kept secret
  - Alice's message to Bob is  $(47, 3, 28)$  because  $3^8 \bmod 47$  is 28. Bob's message to Alice is  $(17)$
  - Alice computes  $17^8 \bmod 47$ , which is 4
  - Bob computes  $28^{10} \bmod 47$ , which is 4
  - Alice and Bob have independently determined that the secret key is now 4
  - Trudy has to solve the equation  $3^x \bmod 47 = 28$ , which can be done by exhaustive search for small numbers like this, but not when all the numbers are hundreds of bits long
  - All currently-known algorithms simply take too long, even on massively parallel supercomputers

9 December 2005

EEC484/584

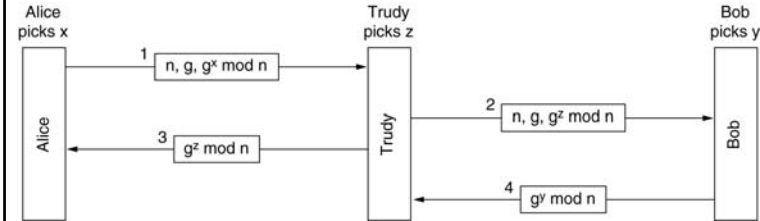
Wenbing Zhao

## Establishing a Shared Key:

30

### The Diffie-Hellman Key Exchange

- The **bucket brigade** or **man-in-the-middle attack**
  - When Bob gets the triple  $(47, 3, 28)$ , how does he know it is from Alice and not from Trudy? There is no way he can know. Unfortunately, Trudy can exploit this fact to deceive both Alice and Bob



9 December 2005

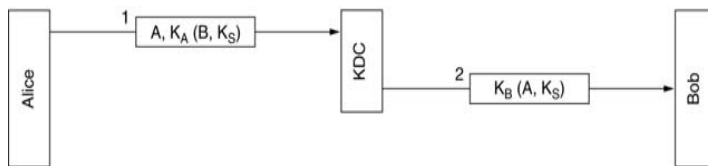
EEC484/584

Wenbing Zhao

## Authentication Using a Key Distribution Center

31

- Each user has a single key shared with the KDC. Authentication and session key management now goes through the KDC
- The following protocol is subject to **replay attack**



9 December 2005

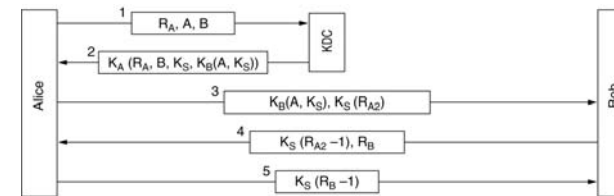
EEC484/584

Wenbing Zhao

## Authentication Using a Key Distribution Center

32

- Needham-Schroeder authentication protocol:** a multiway challenge-response protocol
  - By having each party both generate a challenge and respond to one, the possibility of any kind of replay attack is eliminated



9 December 2005

EEC484/584

Wenbing Zhao

## Needham-Schroeder Authentication Protocol

33

- **Message 1:**  $R_A$  is a nonce
- **Message 2:**
  - $K_B(A, K_S)$  is ticket Alice will send to Bob
  - $R_A$ : so that message 2 is not a replay
  - $B$ : so that if Trudy replaces  $B$  with her id in message 1, it will be detected
  - Ticket is encrypted using Bob's key  $K_B$  so that Trudy cannot replace it with something else on the way back to Alice

9 December 2005

EEC484/584

Wenbing Zhao

## Needham-Schroeder Authentication Protocol

34

- **Message 3:** a new nonce  $R_{A2}$  is used
- **Message 4:** Bob sends back  $K_S(R_{A2}-1)$  instead of  $K_S(R_{A2})$  so that Trudy cannot steal  $K_S(R_{A2})$  from message 3 and replay it here
- **Message 5:** to convince Bob he is talking to Alice and no replays are being used

9 December 2005

EEC484/584

Wenbing Zhao

## Needham-Schroeder Authentication Protocol

35

- **Weakness:**
  - If Trudy ever manages to obtain an old session key in plaintext, she can initiate a new session with Bob by replaying the message 3 corresponding to the compromised key and convince him that she is Alice
    - Trudy needs to know  $K_S$  because she has to respond to Bob's challenge in message 5

9 December 2005

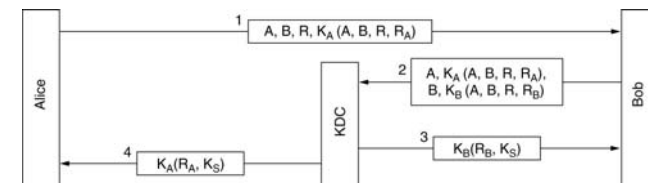
EEC484/584

Wenbing Zhao

## Authentication Using a Key Distribution Center

36

- **The Otway-Rees authentication protocol** (slightly simplified)
  - Addressed the weakness in Needham-Schroeder authentication protocol



9 December 2005

EEC484/584

Wenbing Zhao

## Otway-Rees Authentication Protocol

- **Message 1:** Alice generate two random numbers
  - $R$ : used as a common identifier
  - $R_A$ : used by Alice to challenge
- **Message 2:**
  - $K_A(A, B, R, R_A)$  obtained from encrypted part of message 1
  - $K_B(A, B, R, R_B)$  constructed by Bob using  $R$  from plaintext part of message
  - Both contain the common identifier  $R$ , and contain a challenge

## Otway-Rees Authentication Protocol

- **Message 3 and 4:**
  - The KDC checks to see if the  $R$  in both parts is the same.
    - It might not be because Trudy tampered with  $R$  in message 1 or replaced part of message 2.
  - If the two  $R$ s match, the KDC believes that the request message from Bob is valid
  - It then generates a session key and encrypts it twice, once for Alice (message 4) and once for Bob (message 3).
    - Each message contains the receiver's random number, as proof that the KDC, and not Trudy, generated the message.

## Authentication Using Kerberos

- **Kerberos:** an authentication protocol based on a variant of Needham-Schroeder
  - Used in many real systems (including Windows 2000)
  - Named after a multiheaded dog in Greek mythology that used to guard the entrance to Hades (presumably to keep undesirables out)
  - Designed at M.I.T. to allow workstation users to access network resources in a secure way
  - Its biggest difference from Needham-Schroeder is its assumption that all clocks are fairly well synchronized
  - The protocol has gone through several iterations. V4 is the version most widely used in industry

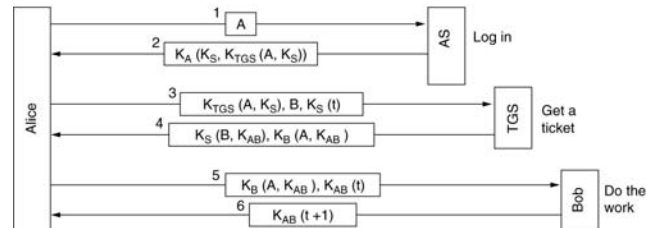
## Authentication Using Kerberos

- **Kerberos** involves three servers in addition to Alice (a client workstation):
  - **Authentication Server (AS):** verifies users during login. AS is similar to a KDC in that it shares a secret password with every user
  - **Ticket-Granting Server (TGS):** issues "proof of identity tickets". The TGS's job is to issue tickets that can convince the real servers that the bearer of a TGS ticket really is who he or she claims to be
  - **Bob the server:** actually does the work Alice wants performed

## Authentication Using Kerberos

41

### The operation of Kerberos V4



9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using Kerberos

42

- **Message 1:** Alice sits down at an arbitrary public workstation and types her name. The workstation sends her name to the AS in plaintext
- **Message 2:** A session key and a ticket,  $K_{TGS}(A, K_S)$ , intended for the TGS
  - These items are packaged together and encrypted using Alice's secret key, so that only Alice can decrypt them
  - Only when message 2 arrives does the workstation ask for Alice's password
  - The password is then used to generate  $K_A$  in order to decrypt message 2 and obtain the session key and TGS ticket inside it
  - At this point, the workstation overwrites Alice's password to make sure that it is only inside the workstation for a few milliseconds at most

9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using Kerberos

43

- **Message 3:** After she logs in, Alice tells the workstation that she wants to contact Bob the file server
  - The workstation then sends message 3 to the TGS asking for a ticket to use with Bob.
  - $K_{TGS}(A, K_S)$ : key element in this request, encrypted with the TGS's secret key and used as proof that the sender really is Alice
  - Trudy can copy message 3 and try to use it again, but she will be foiled by the encrypted timestamp,  $t$ , sent along with it. Trudy cannot replace the timestamp with a more recent one, because she does not know  $K_S$

9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using Kerberos

44

- **Message 4:** The TGS responds by creating a session key,  $K_{AB}$ , for Alice to use with Bob. Two versions of it are sent back.
  - The first is encrypted with only  $K_S$ , so Alice can read it
  - The second is encrypted with Bob's key,  $K_B$ , so Bob can read it

9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using Kerberos

- **Message 5:** Alice sends  $K_{AB}$  to Bob to establish a session with him. This exchange is also timestamped
- **Message 6:** The response is proof to Alice that she is actually talking to Bob, not to Trudy
  - Again  $K_{AB}(t+1)$  is returned instead of  $K_{AB}(t)$  to prevent replay attack
- After this series of exchanges, Alice can communicate with Bob under cover of  $K_{AB}$

## Authentication Using Kerberos

- If she later decides she needs to talk to another server, Carol, she just repeats message 3 to the TGS, only now specifying  $C$  instead of  $B$ 
  - The TGS will promptly respond with a ticket encrypted with  $K_C$  that Alice can send to Carol
  - Carol will accept as proof that it came from Alice

## Authentication Using Kerberos

- The point of all this work is that now Alice can access servers all over the network in a secure way and **her password never has to go over the network**
- However, note that **each server does its own authorization**. When Alice presents her ticket to Bob, this merely proves to Bob who sent it. Precisely what Alice is allowed to do is up to Bob

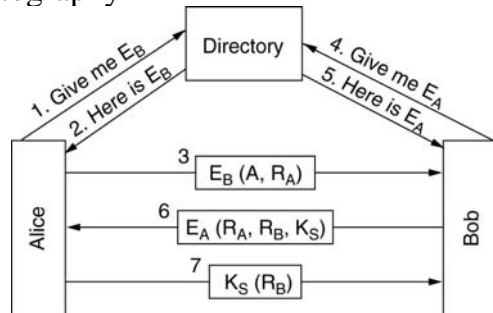
## Authentication Using Kerberos

- Kerberos supports multiple **realms**
  - Each realm has its own AS and TGS
  - To get a ticket for a server in a distant realm, Alice would ask her own TGS for a ticket accepted by the TGS in the distant realm
  - If the distant TGS has registered with the local TGS (the same way local servers do), the local TGS will give Alice a ticket valid at the distant TGS.
  - She can then do business over there, such as getting tickets for servers in that realm.
  - Note, however, that for parties in two realms to do business, each one must trust the other's TGS

## Authentication Using Public-Key Cryptography

49

- Mutual authentication using public-key cryptography



9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using Public-Key Cryptography

50

- **Message 1:** Alice asks a directory server for Bob's certificate (containing Bob's public key)
- **Message 2:** An X.509 certificate containing Bob's public key is sent to Alice
- **Message 3:** When Alice verifies that the signature is correct, she sends Bob a message containing her identity and a nonce

9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using Public-Key Cryptography

51

- **Message 4:** When Bob receives this message, he asks the directory server for Alice's public key
- **Message 5:** An X.509 certificate containing Alice's public key is sent to Bob
- **Message 6:** Bob then sends Alice a message containing Alice's  $R_A$ , his own nonce,  $R_B$ , and a proposed session key,  $K_S$

9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using Public-Key Cryptography

52

- **Message 7:**
  - When Alice gets message 6, she decrypts it using her private key. The fact that  $R_A$  is in message 6 proves that
    - The message must have come from Bob, since Trudy has no way of determining  $R_A$
    - Furthermore, it must be fresh and not a replay, since she just sent Bob  $R_A$
  - Alice agrees to the session by sending back message 7
  - When Bob sees  $R_B$  encrypted with the session key he just generated, he knows Alice got message 6 and verified  $R_A$

9 December 2005

EEC484/584

Wenbing Zhao

## Authentication Using Public-Key Cryptography

53

- What can Trudy do to try to subvert this protocol?
  - She can fabricate **message 3** and trick Bob into probing Alice, but Alice (from **message 6**) will see an  $R_A$  that she did not send and will not proceed further
  - Trudy cannot forge **message 7** back to Bob because she does not know  $R_B$  or  $K_S$  and cannot determine them without Alice's private key

9 December 2005

EEC484/584

Wenbing Zhao

## E-Mail Security

54

- PGP– Pretty Good Privacy
- PEM – Privacy Enhanced Mail
- S/MIME

9 December 2005

EEC484/584

Wenbing Zhao

## PGP – Pretty Good Privacy

55

- **PGP (Pretty Good Privacy)**: e-mail security package that provides privacy, authentication, digital signatures, and compression, all in an easy-to-use form
  - Created by Zimmermann, released in 1991
  - Zimmermann is a privacy advocate whose motto is: *If privacy is outlawed, only outlaws will have privacy*
  - The complete package, including all the source code, is distributed free of charge via the Internet
  - Due to its quality, price (zero), and easy availability on UNIX, Linux, Windows, and Mac OS platforms, it is widely used today

9 December 2005

EEC484/584

Wenbing Zhao

## PGP – Pretty Good Privacy

56

- PGP encrypts data by using a block cipher called **IDEA (International Data Encryption Algorithm)**
  - It was devised in Switzerland at a time when DES was seen as tainted and AES had not yet been invented.
  - It uses 128-bit keys
- Key management uses **RSA**
- Data integrity uses **MD5**
- Compression uses the **ZIP program**, which uses the Ziv-Lempel algorithm (Ziv and Lempel, 1977)
  - Compression saves bandwidth
  - It also wipes out the frequency information contained in the plaintext. In effect, it converts the plaintext into junk

9 December 2005

EEC484/584

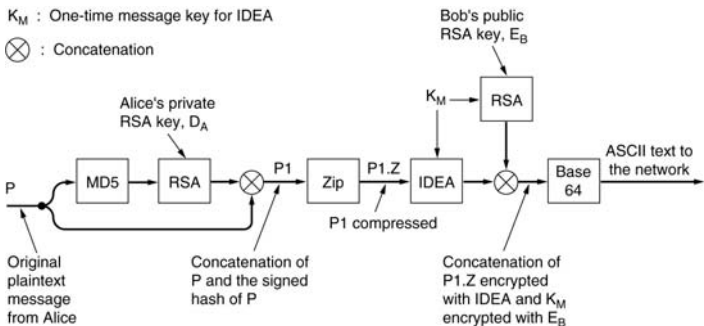
Wenbing Zhao

# PGP – Pretty Good Privacy

## PGP in operation for sending a message

$K_M$  : One-time message key for IDEA

⊗ : Concatenation



# PGP – Pretty Good Privacy

## Alice sends an email P to Bob using PGP:

- Both Alice and Bob have private ( $D_X$ ) and public ( $E_X$ ) RSA keys. Assume that each one knows the other's public key
- PGP first hashes Alice's message, P, using MD5, and then encrypts the resulting hash using her private RSA key,  $D_A$
- The encrypted hash and the original message are concatenated into a single message, P1, and compressed using the ZIP program, the output of this step is P1.Z.

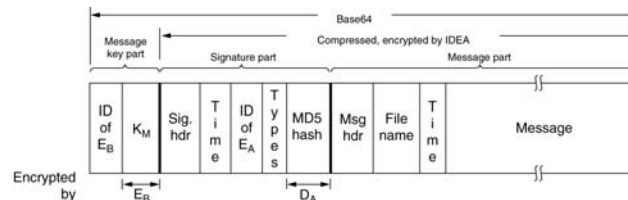
# PGP – Pretty Good Privacy

- Next, PGP prompts Alice for some random input. Both the content and the typing speed are used to generate a 128-bit IDEA message key,  $K_M$
- $K_M$  is now used to encrypt P1.Z with IDEA in cipher feedback mode
- In addition,  $K_M$  is encrypted with Bob's public key,  $E_B$ . These two components are then concatenated and converted to base64

# PGP – Pretty Good Privacy

## The format of a classic PGP message

- RSA is only used in two places: to encrypt the 128-bit MD5 hash and to encrypt the 128-bit IDEA key



## PGP – Pretty Good Privacy

- **Key management:** supports both PGP public key ring mechanism and X.509
- **PGP public key ring mechanism:** Each user maintains two data structures locally, a private key ring and a public key ring
- The **public key ring** contains public keys of the user's correspondents
  - These are needed to encrypt the message keys associated with each message
  - Each entry on the public key ring contains not only the public key, but also its 64-bit identifier and an indication of how strongly the user trusts the key

## PGP – Pretty Good Privacy

- The **private key ring** contains one or more personal private-public key pairs
  - Multiple pairs per user is permitted so that users can change their public keys periodically or when one is thought to have been compromised, without invalidating messages currently in preparation or in transit
  - Each pair has an **identifier (low-order 64 bits of the public key)** associated with it so that a message sender can tell the recipient which public key was used to encrypt it
  - The private keys on disk are encrypted using a special (arbitrarily long) password to protect them against sneak attacks

## S/MIME

- **S/MIME (Secure/MIME):** IETF's next venture into e-mail security.
- Like PEM, it provides authentication, data integrity, secrecy, and nonrepudiation
- It also is quite flexible, supporting a variety of cryptographic algorithms
- S/MIME integrates well with MIME, allowing all kinds of messages to be protected.
- A variety of new MIME headers are defined, for example, for holding digital signatures

## Exercises

- P8.30, P8.32, P8.33, P8.35, P8.39