Principles of Network Security (cont.)

Kai Shen

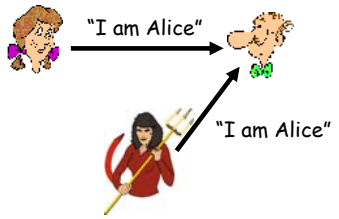
Principles of Network Security

- Confidentiality: cryptography
- Authentication
- Integrity
- Key distribution and certification

Authentication: version 1.0

Authentication: Bob wants Alice to "prove" her identity to him.

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

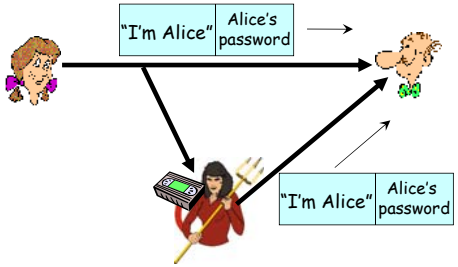


Failure scenario??

Trudy can simply declare herself to be Alice

Authentication: version 2.0

Protocol ap2.0: Alice says "I am Alice" and sends her secret password to "prove" it.



Failure scenario??

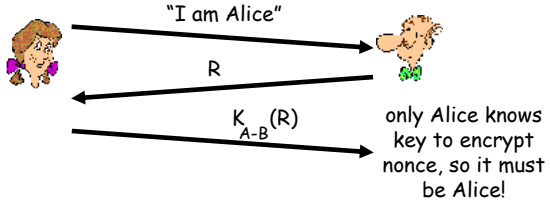
playback attack: Trudy records Alice's packet and later plays it back to Bob

Authentication: version 3.0

Goal: avoid playback attack

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

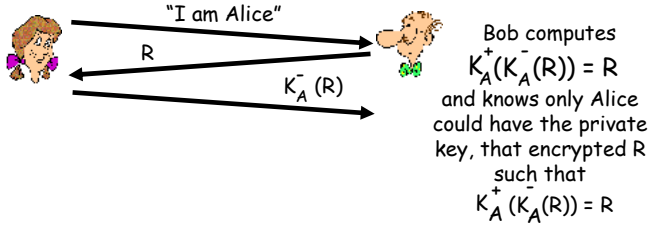
ap3.0: Bob sends Alice a nonce, R. Alice must return R, encrypted with shared secret key



Authentication: version 4.0

ap3.0 requires shared symmetric key. Key distribution can be a problem.

ap4.0: use nonce, public key cryptography.



Principles of Network Security

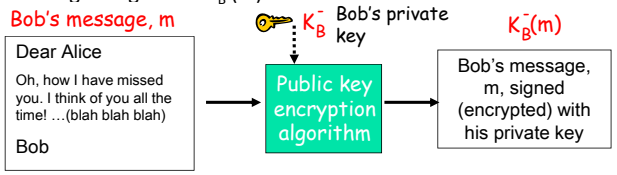
- Confidentiality: cryptography
- Authentication
- Integrity
- Key distribution and certification

Integrity

- Digital Signatures:
  - cryptographic technique to ensure document integrity.
  - analogous to hand-written signatures.
- Sender (Bob) digitally signs document, establishing he is document owner/creator.
- The recipient (Alice) receives the document and the digital signature.
- The recipient can be sure that the document is
  - verifiable: Bob signed the document.
  - nonforgeable: the document hasn't been changed since Bob signed it.

Digital Signatures

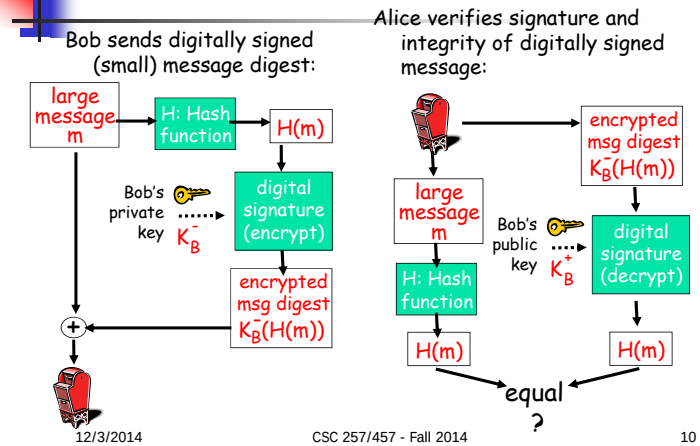
- Bob signs m by encrypting with his private key, creating a digital signature $K_B^-(m)$



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

Problem: computationally expensive to public-key-encrypt long messages.

Signed Message Digest



Message Digests

- Apply a hash function H to m, get a much smaller message digest H(m).
- Public-key-encrypt the message digest to generate the digital signature $K_B^-(H(m))$.

Good/bad hash functions?

- Hint: given a hash function, it is possible for many messages sharing the same digest.

Internet Checksum: Poor Hash Function for Generating Message Digests

Given a message and its Internet checksum, it is easy to find another message with same checksum.

message	ASCII format
I O U 1	49 4F 55 31
0 0 . 9	30 30 2E 39
9 B O B	39 42 D2 42
	B2 C1 D2 AC

message	ASCII format
I O U 9	49 4F 55 39
0 0 . 1	30 30 2E 31
9 B O B	39 42 D2 42
	B2 C1 D2 AC

different messages but identical checksums!

Hash function property: given digest x for message m, computationally infeasible to find another message m' that shares the same digest. Pre-image resistance. Collision resistance.

Good Hash Functions for Generating Message Digests

- MD5
  - computes 128-bit message digest in 4-step process.
  - appears difficult to construct message m whose MD5 hash is equal to x.
- SHA-1
  - [NIST, FIPS PUB 180-1]
  - 160-bit message digest
- SHA-2
  - More bits: SHA256, SHA512

http://en.wikipedia.org/wiki/Cryptographic_hash_function

Principles of Network Security

- Confidentiality: cryptography
- Authentication
- Integrity
- Key distribution and certification

Key Distribution and Certification

Symmetric key distribution problem:

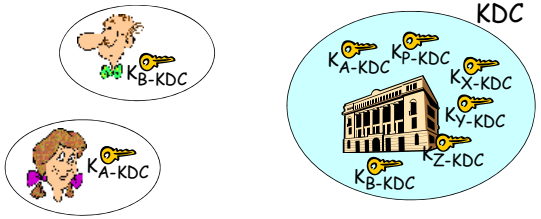
- How do Alice and Bob establish shared secret key over network without Trudy's knowledge?

Public key distribution 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?

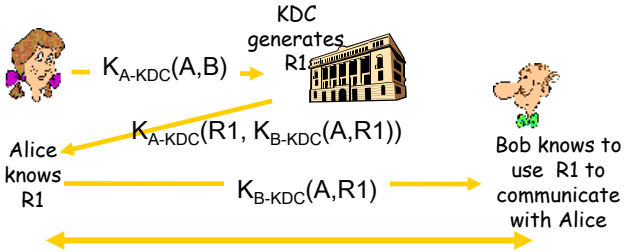
Secret Key Distribution: Key Distribution Center (KDC)

- 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 using KDC

Q: 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

Security Vulnerability with Public Key Distribution

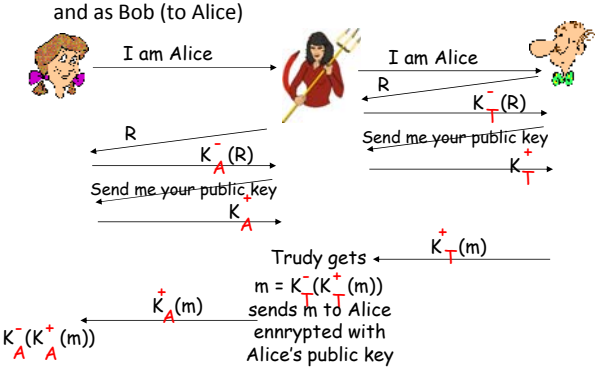
A case example for public key-based authentication.

Bob computes $K_A^+(K_A^-(R)) = R$ and knows only Alice could have the private key, that encrypted R such that $K_A^+(K_A^-(R)) = R$

What if Bob doesn't know Alice's public key ahead of time?

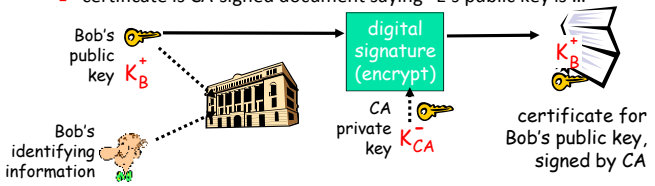
Security vulnerability when public keys are not well known

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



Public Key Distribution: Certification Authorities

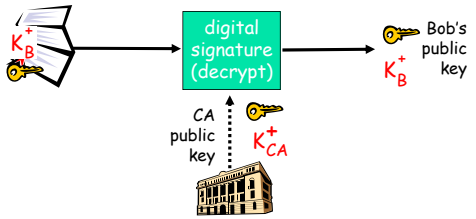
- Certification authority (CA): trustable by everyone; everyone knows its public key.
- E (person, router) registers its public key with CA.
  - E provides "proof of identity" to CA.
  - CA creates certificate binding E to its public key.
  - certificate is CA-signed document saying "E's public key is …"



Certification Authorities (cont.)

When Alice wants to verify Bob's public key:

- gets Bob's certificate (Bob or elsewhere).
- apply CA's public key to Bob's certificate, verify Bob's public key.



Key Certification Methods

- Public key certificate signed by a certification authority
- Peer certification:
  - If A knows B personally, they can verify each other's public keys using offline means and sign them;
- Certificate chain leading to a certificate authority
  - CA signs A's public key certificate
  - A signs B's public key certificate
  - B signs C's public key certificate
  - If you trust all signers (CA, A, B in this case), then you can trust the certificate.

Summary: Principles of Network Security

Cryptography:

- symmetric keys: protocols? weakness?
- public keys: protocol? weakness?

Confidentiality:

- only sender, intended receiver should "understand" message contents

Authentication:

- sender, receiver want to confirm identity of each other

Message Integrity:

- sender, receiver want to ensure message not altered (in transit, or afterwards)

Key Distribution and Certification:

- problem and solution for symmetric / public keys

Disclaimer

- Parts of the lecture slides contain original work of James Kurose, Larry Peterson, and Keith Ross. The slides are intended for the sole purpose of instruction of computer networks at the University of Rochester. All copyrighted materials belong to their original owner(s).