

---

# CSE331: Introduction to Networks and Security

Lecture 28

Fall 2006



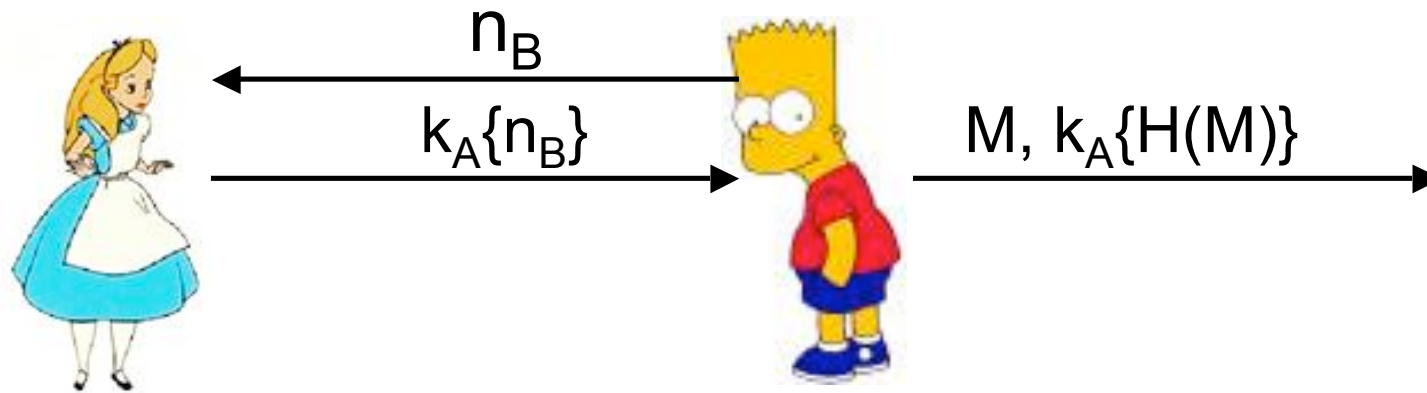
# Announcements

---

- Project 3 is due Monday, November 20th
- Plan for today:
  - Digital Signatures

# Multiple Use of Keys

- Risky to use keys for multiple purposes.
- Using an RSA key for both authentication and signatures may allow a chosen-text attack.
- B attacker/verifier,  $n_B = H(M)$  for some message  $M$ .



B, pretending to be A

# General Principles

---

- Don't do anything more than necessary until confidence is built.
  - Initiator should prove identity before the responder does any “expensive” action (like encryption)
- Embed the intended recipient of the message in the message itself
- Principal that generates a nonce is the one that verifies it
- Before encrypting an untrusted message, add “salt” (i.e. a nonce) to prevent chosen plaintext attacks
- Use asymmetric message formats (either in “shape” or by using asymmetric keys) to make it harder for roles to be switched

# Physical Signatures

---

- Consider a paper check used to transfer money from one person to another
- Signature confirms authenticity
  - Only legitimate signer can produce signature
- In case of alleged forgery
  - 3<sup>rd</sup> party can verify authenticity
- Checks are cancelled
  - So they can't be reused
- Checks are not alterable
  - Or alterations are easily detected

# Digital Signatures: Requirements I

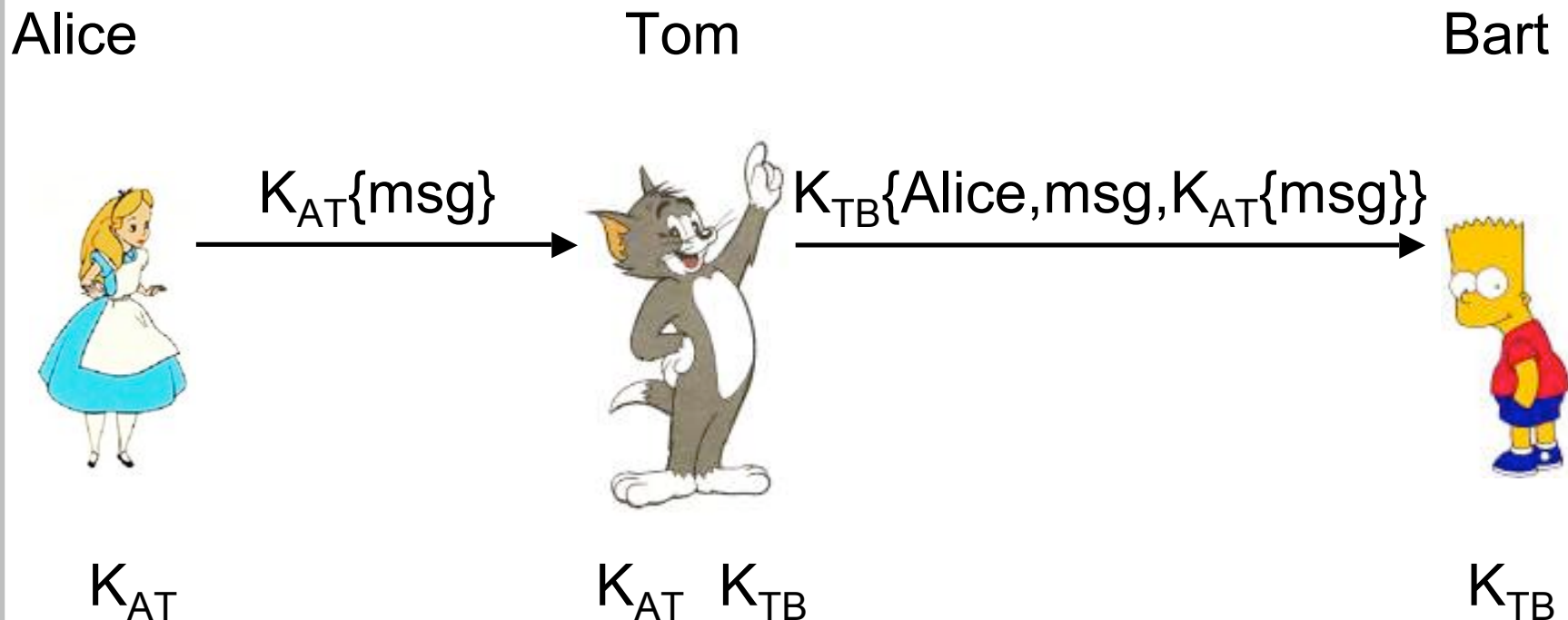
- A mark that only one principal can make, but others can easily recognize
- Unforgeable
  - If P signs a message M with signature  $S_P\{M\}$  it is impossible for any other principal to produce the pair  $(M, S_P\{M\})$ .
- Authentic
  - If R receives the pair  $(M, S_P\{M\})$  purportedly from P, R can check that the signature really is from P.

## Digital Signatures: Requirements II

---

- Not alterable
  - After being transmitted,  $(M, S_P\{ M\})$  cannot be changed by P, R, or an interceptor.
- Not reusable
  - A duplicate message will be detected by the recipient.
- Nonrepudiation:
  - P should not be able to claim they didn't sign something when in fact they did.
  - (Related to unforgeability: If P can show that someone else could have forged P's signature, they can repudiate ("refuse to acknowledge") the validity of the signature.)

# Digital Signatures with Shared Keys



Tom is a trusted 3<sup>rd</sup> party (or arbiter).

**Authenticity:** Tom verifies Alice's message, Bart trusts Tom.

**No Forgery:** Bart can keep  $msg$ ,  $K_{AT}\{msg\}$ , which only Alice (or Tom, but he's trusted not to) could produce



# Preventing Reuse and Alteration

---

- To prevent reuse of the signature
  - Incorporate a *timestamp* (or sequence number)
- Alteration
  - If a block cipher is used, recipient could splice-together new messages from individual blocks.
- To prevent alteration
  - Timestamp must be part of each block
  - Or... use *cipher block chaining*

## Digital Signatures with Public Keys

---

- Assumes the algorithm is *commutative*:
  - $D(E(M, K), k) = E(D(M, k), K)$
- Let  $K_A$  be Alice's public key
- Let  $k_A$  be her private key
- To sign msg, Alice sends  $D(msg, k_A)$
- Bart can verify the message with Alice's public key
  
- Works! RSA:  $(m^e)^d = m^{ed} = (m^d)^e$

# Digital Signatures with Public Keys

Alice



$k_A, K_A, K_B$

Bart



$k_B, K_B, K_A$

$k_A\{msg\}$



- No trusted 3<sup>rd</sup> party.
- Simpler algorithm.
- More expensive
- No confidentiality

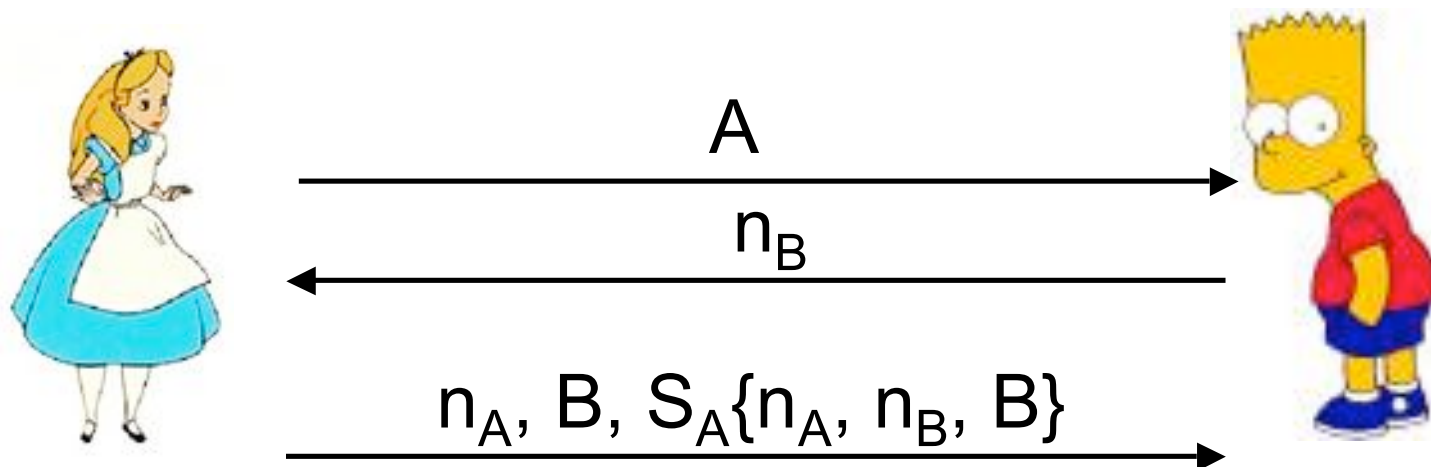
## Variations on Public Key Signatures

---

- Timestamps again (to prevent replay)
  - Signed certificate valid for only some time.
- Add an extra layer of encryption to guarantee confidentiality
  - Alice sends  $K_B\{k_A\{msg\}\}$  to Bart
- Combined with hashes:
  - Send  $(msg, k_A\{MD5(msg)\})$

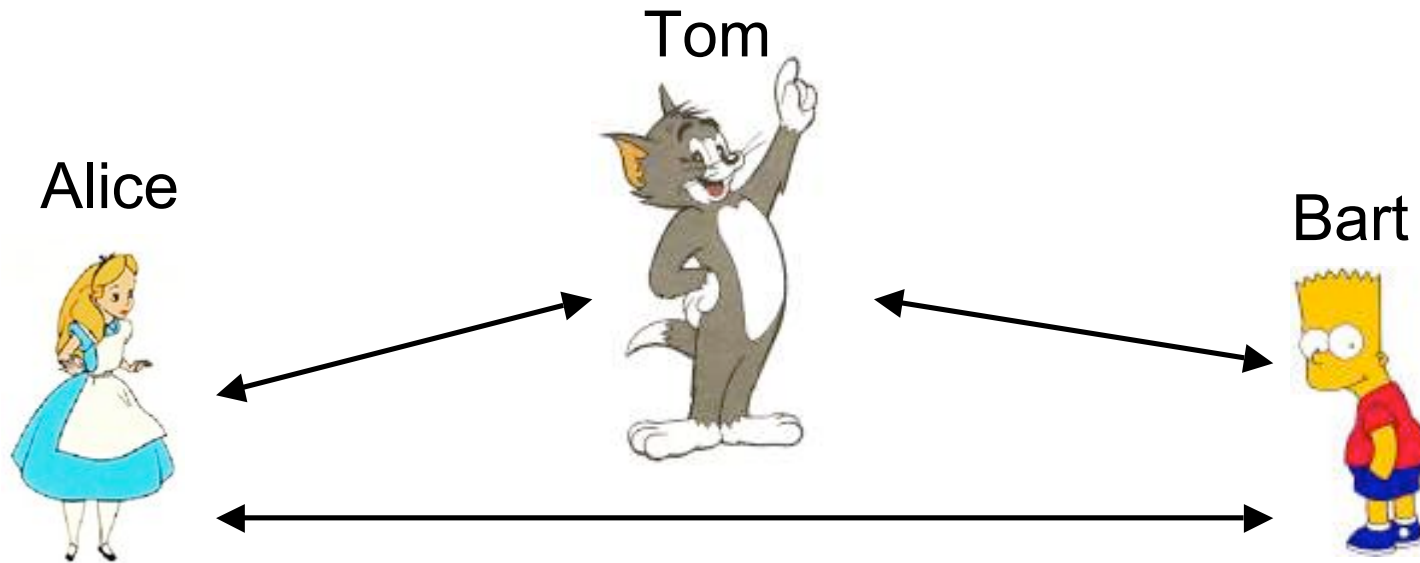
## Unilateral Authentication: Signatures

- $S_A\{M\}$  is A's signature on message M.
- Unilateral authentication with nonces:



The  $n_A$  prevents chosen plaintext attacks.

# Arbitrated Protocols

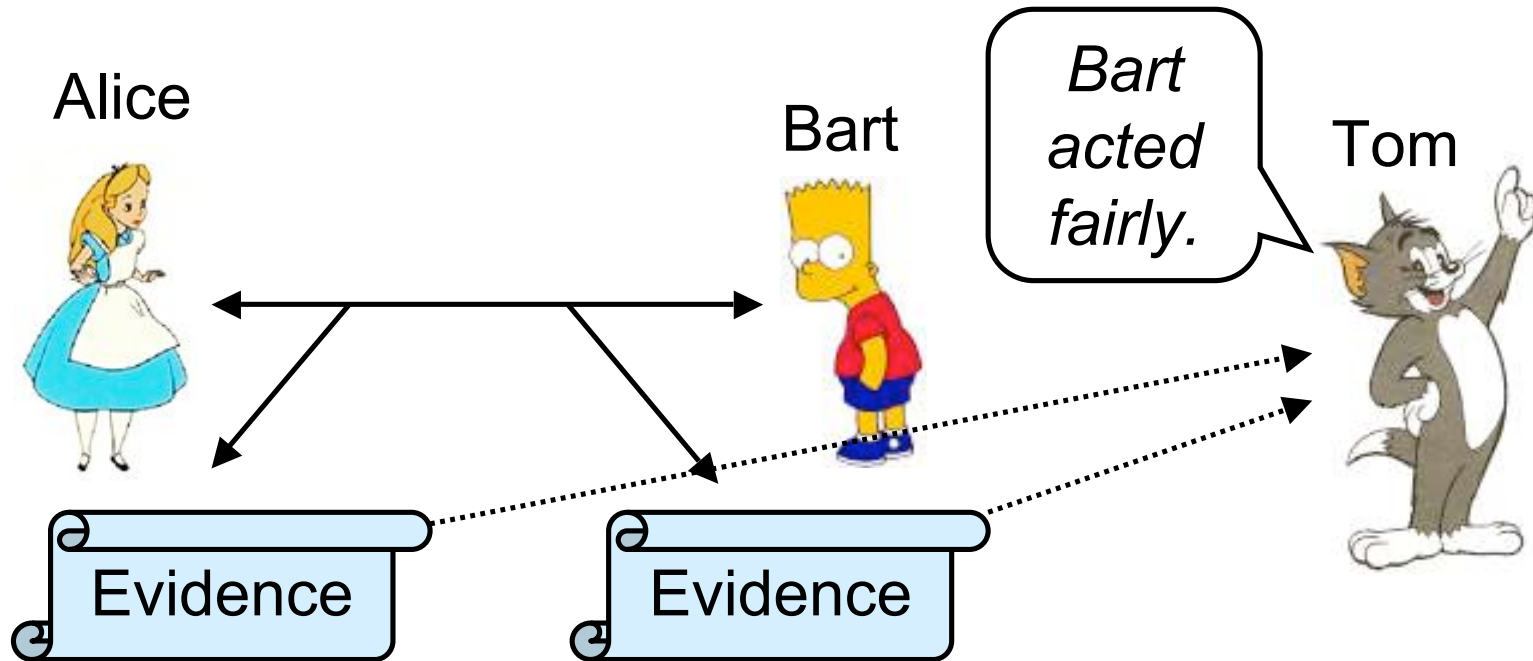


- Tom is an *arbiter*
  - Disinterested in the outcome (doesn't play favorites)
  - Trusted by the participants (Trusted 3<sup>rd</sup> party)
  - Protocol can't continue without T's participation

# Arbitrated Protocols (Continued)

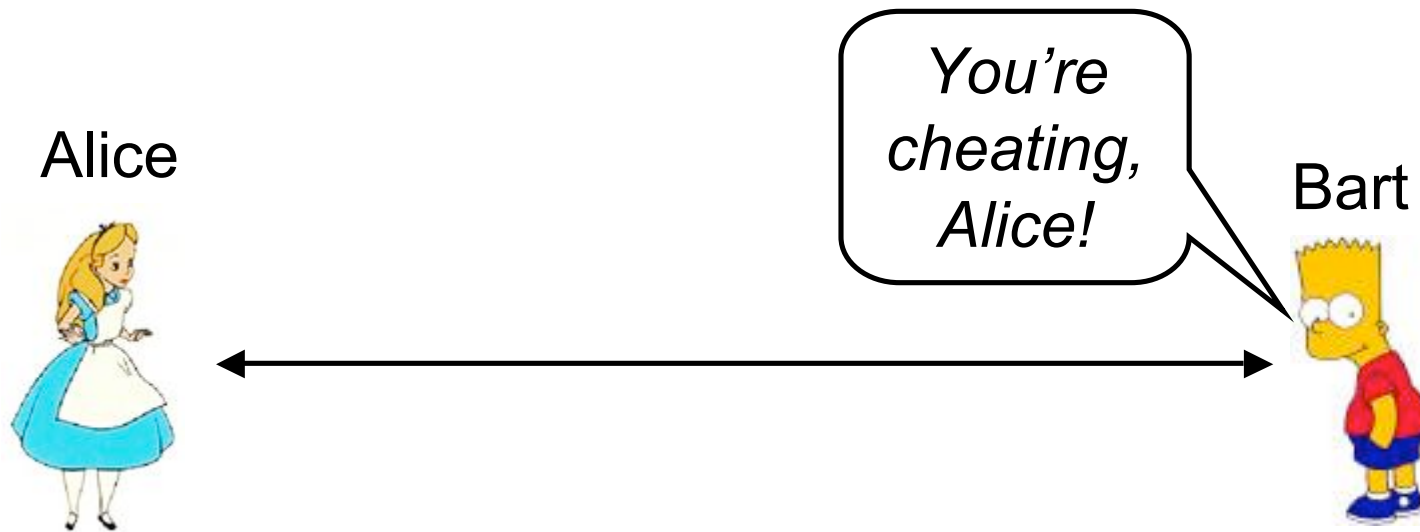
- Real-world examples:
  - Lawyers, Bankers, Notary Public
- Issues:
  - Finding a trusted 3<sup>rd</sup> party
  - Additional resources needed for the arbitrator
  - Delay (introduced by arbitration)
  - Arbitrator might become a bottleneck
  - Single point of vulnerability: attack the arbitrator!

# Adjudicated Protocols



- Alice and Bart record an *audit log*
- Only in exceptional circumstances do they contact a trusted 3<sup>rd</sup> party. (3<sup>rd</sup> party is not always needed.)
- Tom as the *adjudicator* can inspect the evidence and determine whether the protocol was carried out fairly

# Self-Enforcing Protocols



- No trusted 3<sup>rd</sup> party involved.
- Participants can determine whether other parties cheat.
- Protocol is constructed so that there are no possible disputes of the outcome.

# Examples We've Seen

---

- Arbitrated Protocol
  - Shared key digital signature algorithm
  - Trusted 3rd party provided authenticity
- Adjudicated Protocol
  - Public key digital signature algorithm
  - Bart can keep Alice's digitally signed message
    - Trusted 3<sup>rd</sup> party provided non-repudiation