Announcements

• Project 2 Due Monday, Nov. 1

• 2\textsuperscript{nd} Midterm Scheduled for Monday, Nov. 15
Recap: Challenge Response

- Protocol doesn’t reveal the secret.
- **Challenge/Response**
  - Bart requests proof that Alice knows the secret
  - Alice requires proof from Bart
  - $R_A$ and $R_B$ are randomly generated numbers
(Flawed) Optimized Version

- Why not send more information in each message?
- This seems like a simple optimization.
- But, it’s broken… how?
Attack: Marvin can Masquerade as Alice

- Marvin pretends to take the role of Alice in two runs of the protocol.
  - Tricks Bart into doing Alice's part of the challenge!
  - Interleaves two instances of the same protocol.
Lessons

• Protocol design is tricky and subtle
  – “Optimizations” aren’t necessarily good

• Need to worry about:
  – Multiple instances of the same protocol running in parallel
  – Intruders that play by the rules, mostly
Threats

- **Transferability**: B cannot reuse an identification exchange with A to successfully impersonate A to a third party C.

- **Impersonation**: The probability is negligible that a party C distinct from A can carry out the protocol in the role of A and cause B to accept it as having A’s identity.
Assumptions

- A large number of previous authentications between A and B may have been observed.

- The adversary C has participated in previous protocol executions with A and/or B.

- Multiple instances of the protocol, possibly instantiated by C, may be run simultaneously.
Attacks Not Addressed

• Identification affirms that communication with the expected party occurred at a given point in time.

• Two active attacks are not addressed:
  – *Usurpation*: The session beginning with the identification is “usurped” by the attacker.
  – *Grand Chess Master Problem*: A man-in-the-middle relays messages between two parties without changing them. (End-to-end authentication, not hop-by-hop)
Challenge-Response

• Background.
  – Random numbers (nonces).
  – Sequence numbers.
  – Timestamps.
• Symmetric keys.
  – With timestamps or random numbers.
• Asymmetric keys.
  – With encryption or signature.
Replay

• *Replay*: the threat in which a transmission is observed by an eavesdropper who subsequently reuses it as part of a protocol, possibly to impersonate the original sender.
  – Example: Monitor the first part of a telnet session to obtain a sequence of transmissions sufficient to get a log-in.

• Three strategies for defeating replay attacks
  – Nonces
  – Timestamps
  – Sequence numbers.
Nonces: Random Numbers

- **Nonce**: A number chosen at random from a range of possible values.
  - Each generated nonce is valid only once.
- In a challenge-response protocol nonces are used as follows.
  - The verifier chooses a (new) random number and provides it to the claimant.
  - The claimant performs an operation on it showing knowledge of a secret.
  - This information is bound inseparably to the random number and returned to the verifier for examination.
  - A timeout period is used to ensure “freshness”.

Time Stamps

- The claimant sends a message with a timestamp.
- The verifier checks that it falls within an acceptance window of time.
- The last timestamp received is held, and identification requests with older timestamps are ignored.
- Good only if clock synchronization is close enough for acceptance window.
Sequence Numbers

• Sequence numbers provide a sequential or monotonic counter on messages.
• If a message is replayed and the original message was received, the replay will have an old or too-small sequence number and be discarded.
• Cannot detect forced delay.
• Difficult to maintain when there are system failures.
Unilateral Symmetric Key

- Unilateral authentication with nonce.

\[ K_{AB}\{n, B\} \]
Mutual Symmetric Key

• Using Nonces:

\[ K_{AB}\{n_A, n_B, B\} \]

\[ K_{AB}\{n_A, A\} \]
Mutual Public Key Decryption

- Exchange nonces

\[ K_B \{ n_A, A \} \quad K_A \{ n_A, n_B \} \]

\[ n_B \]
Unilateral Digital Signatures

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

$$n_A, B, S_A\{n_A, n_B, B\}$$

The $n_A$ prevents chosen plaintext attacks.
Mutual Digital Signatures

• Using random numbers.

\[ n_B \]

\[ n_A, B, S_A\{n_A, n_B, B\} \]

\[ A, S_B\{n_A, n_B, A\} \]
Primary Attacks

• Replay.
• Interleaving.
• Reflection.
• Forced delay.
• Chosen plaintext.
Primary Controls

- **Replay:**
  - use of challenge-response techniques
  - embed target identity in response.

- **Interleaving**
  - link messages in a session with chained nonces.

- **Reflection:**
  - embed identifier of target party in challenge response
  - use asymmetric message formats
  - use asymmetric keys.
Primary Controls, continued

• Chosen text:
  – embed self-chosen random numbers ("confounders") in responses
  – use "zero knowledge" techniques.

• Forced delays:
  – use nonces with short timeouts
  – use timestamps in addition to other techniques.
Usurpation Attacks

- Identification protocols provide assurances corroborating the identity of an entity only at a given instant in time.
- Techniques to assure ongoing authenticity:
  - Periodic re-identification.
  - Tying identification to an ongoing integrity service. For example: key establishment and encryption.
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