Graduate Course on **Computer Security**

Lecture 4: Authentication Protocols

Iliano Cervesato `iliano@itd.nrl.navy.mil`

ITT Industries, Inc @ NRL - Washington DC

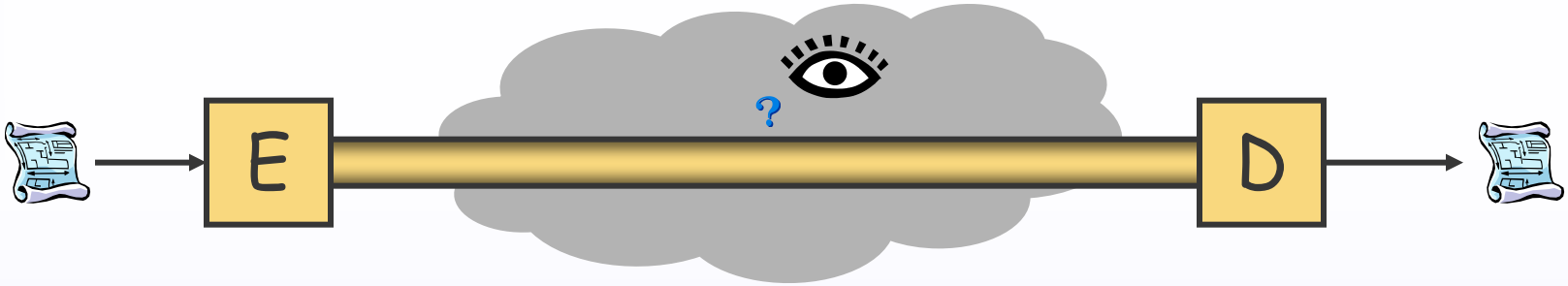
http://www.cs.stanford.edu/~iliano/

Outline

"Cryptography is not broken,
it is circumvented"
[Shamir]

- Authentication Protocols
  - Challenge-response
  - Key generation
  - Key distribution
- Attacks
  - Man-in-the-middle
  - Type flaw
  - Parallel session
  - Binding
  - Encapsulation
  - Implementation-dependent
- Design principles

Security Protocols



Encryption provides virtual trusted channels

- Security protocols
  - How to establish, maintain and use these channels
  - Authentication protocols
    - How to establish channel in the first place
      - Negotiate parameters of channel
      - Ensure that channel is still trusted
  - Other types of protocols
    - Using trusted channels for specific purposes
      - Electronic commerce (e-cash, e-auctions, …)
      - Electronic voting
      - Electronic contract signing, …

This lecture, also 5, 7, 8, 9

Lecture 10

Authentication Protocols

- Challenge-response
  - Verify somebody is at the other end of channel
- Key generation
  - Establish channel
- Key distribution
  - Bind channel ends with requesters
- Key translation
  - Use indirect channels

These aspects can be combined

Some Notation

We abstract from the cryptographic algorithms used

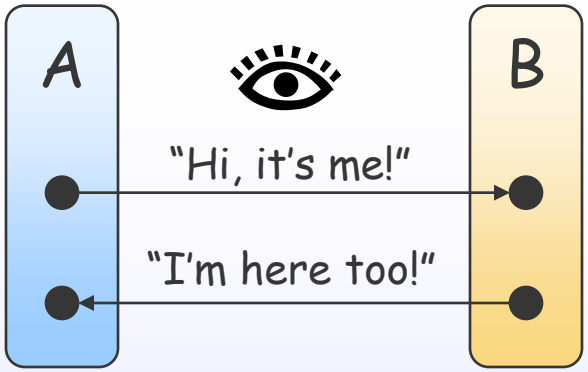
- Encryption: $\{m\}_k$
  - In particular shared-key encryption
  - Public-key encryption sometimes written $\{\{m\}\}_k$
- Authentication: $[m]_k$
  - In particular for MACs
  - Digital signatures sometimes written $[[m]]_k$
  - Usually includes both message and digest
- Decryption/verification not modeled explicitly

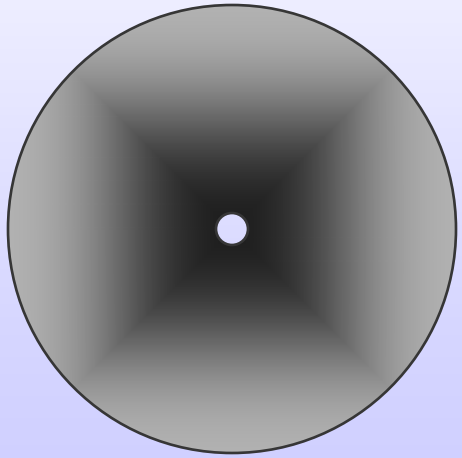
Our Heros

- Generic principals
  - A (Alice)
  - B (Bob)
  - C (Charlie), …
- Servers
  - S (Sam)
  - … specialized names
    - Trusted-Third Party – TTP
    - Certification Authority –CA
    - Key Distribution Center – KDC
    - …
- Attacker
  - I (intruder)
  - Also known as
    - E (Eve – eavesdropper, enemy)
    - M (Mallory – malicious)
    - Trudy, ….

Challenge-Response Protocols

- Given trusted channel
  - A checks if B is there
    - Sends challenge to B
    - Waits for response
  - Get B to use the channel
    - By decrypting the challenge
    - By encrypting the response
    - … or both
- Used to
  - Test a newly established channel
  - Verify a previously used channel
- Usually part of bigger protocols
- Also called *authentication test*





A's view

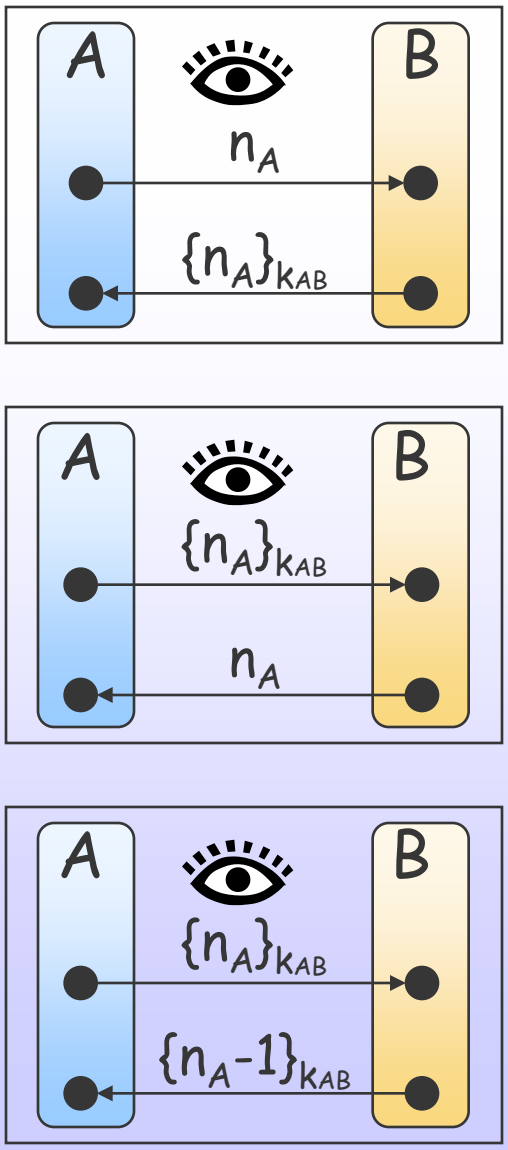
Guarantying Freshness

- Reusing challenges is dangerous
  - Waste subsequent transmissions
  - Replay of favorable messages
    - If channel used to transmit keys
    - and a previous key k was compromised,
    - then I can force A to reuse k
- Response should be fresh
  - Nonces
  - Timestamps
  - Sequence numbers
  - Fresh key (with care!)

Nonces

Random sequence of bits

- Typically 32-128 bit long
- Generated fresh by originator as challenge
  - Unpredictable
  - Checked in response
- Not checked by recipient
  - Impractical to memorize them
- Never reused
  - But may contribute to keys
    - E.g. by hashing

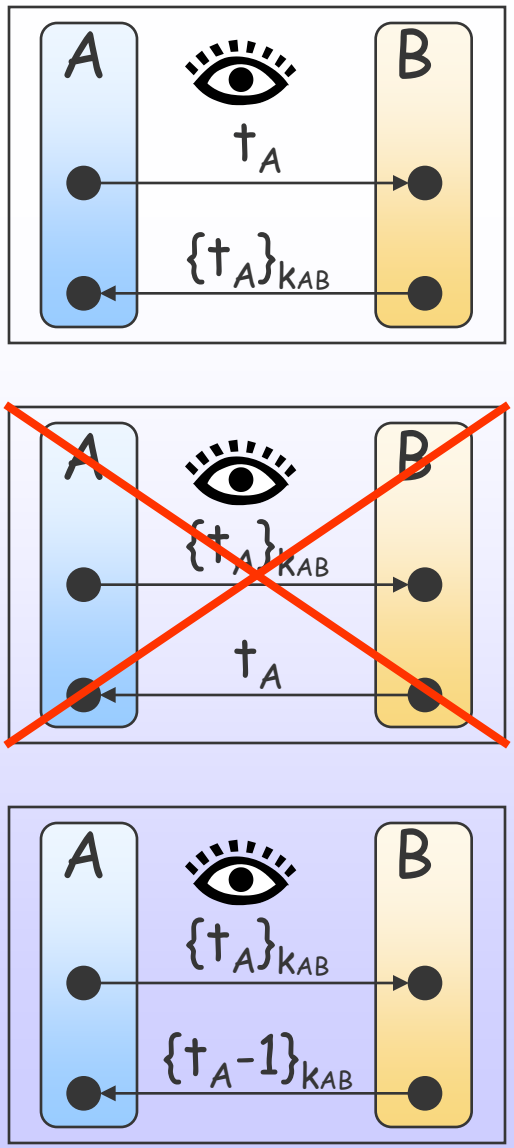


Challenge-response exchanges using nonces

Timestamps

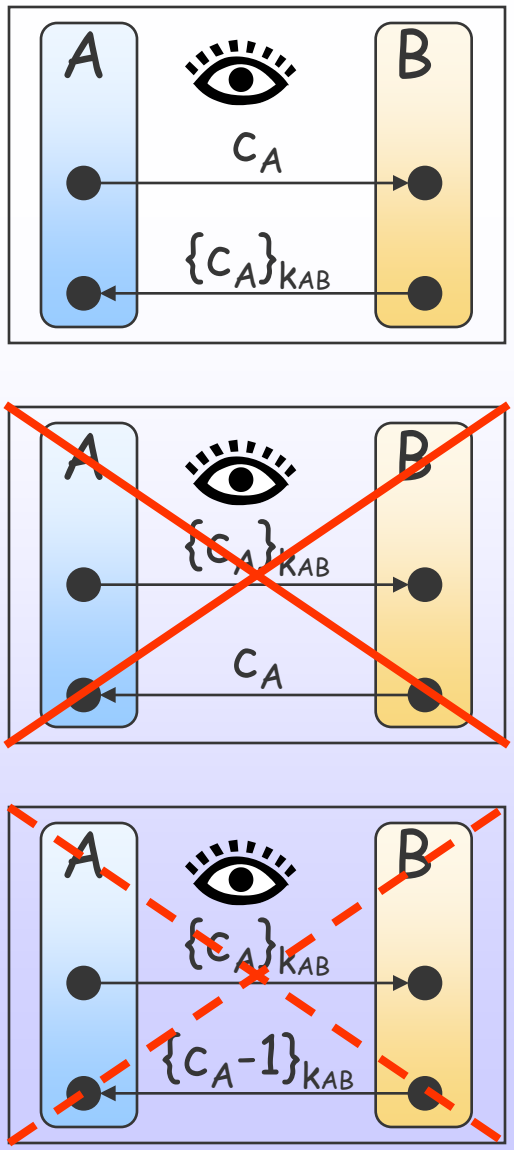
Current time in local computer

- E.g. in milliseconds
- Checkable by recipient
  - Element of predictability
  - Recipient must keep most recent timestamps to avoid replay
- Requires common time reference
  - Allow for clock skew
  - Use secure synchronized clocks
- Supports for service time-out



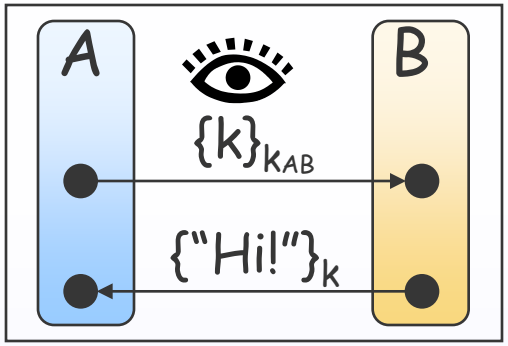
Sequence Numbers

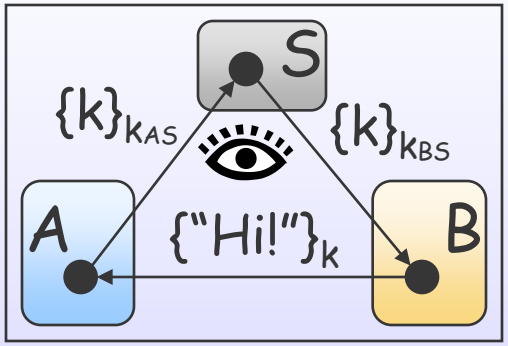
- Originator maintains counter
  - Incremented by 1 after each challenge
  - Must be bound with data that identifies channel
- Recipient memorizes most recent value
  - Rejects values that are too old
- Similar to timestamp but
  - Local to originator or even channel
  - Cannot be used for timeout



Keys

- Initiator generates key k
  - Sends it encrypted
- Recipient responds using k
  - Other mechanisms needed to guaranty freshness to recipient
- Often done through third-party
- Achieves key distribution at the same time





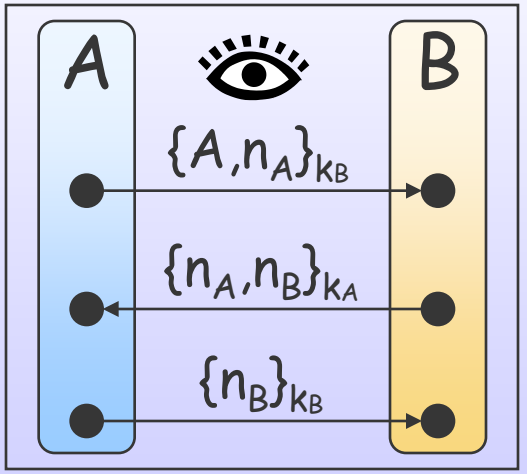
More on Keys

- Long-term keys
  - Exist before the protocol begins
  - Do not change across protocol executions
- Session keys (or short-term keys)
  - Generated as part of the protocol
  - Validity guaranteed till protocol is completed
    - Could be released when protocol terminates
    - Could be cryptographically weak
- Session (or run)
  - Protocol execution from start to finish

Authentication

Assurance to be talking with the expected principal

- Challenge-response is a fundamental mechanism
  - Ensure freshness
  - If channel is trusted, authenticates recipient to initiator
- Mutual authentication
  - Both party believe they are talking to each other
  - Done through double challenge-response
    - Typically 3 messages



Needham-Schroeder public-key protocol (fragment)

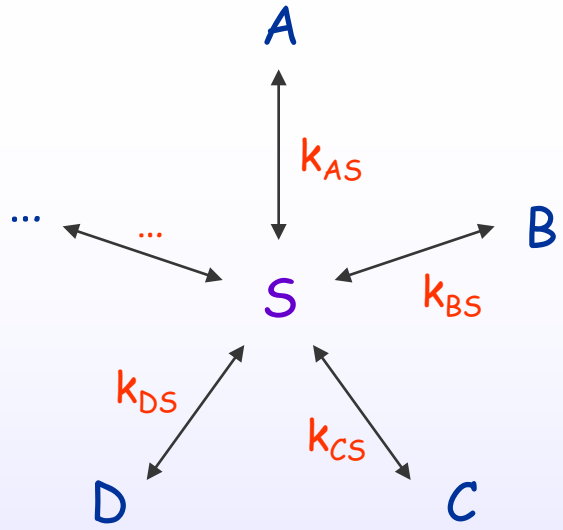
Key Generation Protocols ...

A wants to establish channel with B

- Shared-key infrastructure
  - Principals shares a key with a KDC
- Public-key infrastructure
  - Principals have published encryption keys
- Diffie-Hellman
  - Principals know group and generator

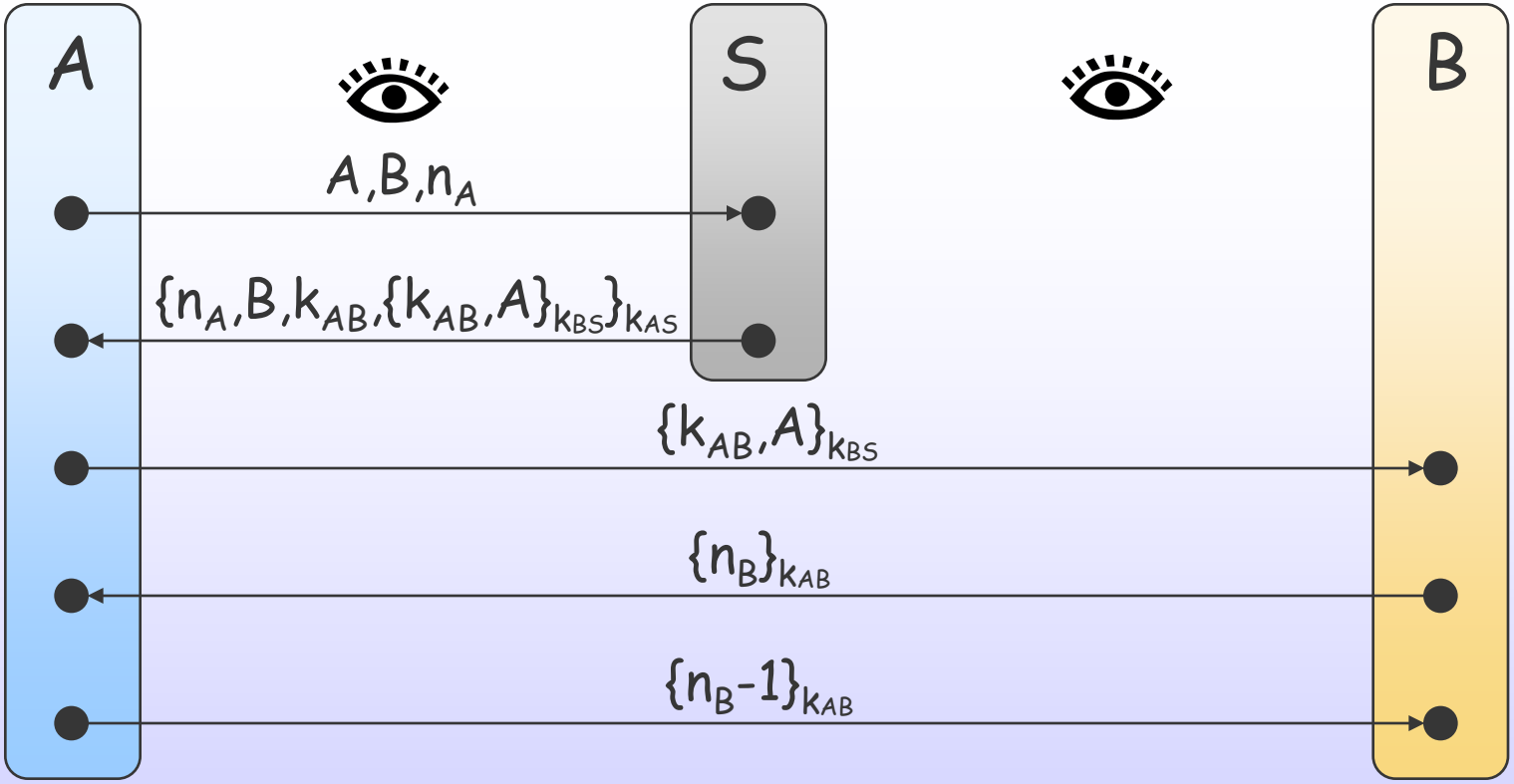
… with Shared-Key Infrastructure

- Each principal has a shared key with KDC S



- Ask S to create channel
  - Create new key k
  - Distribute k to A and B using k_{AS} and K_{BS}
- Examples
  - Needham-Schroeder shared-key protocol
  - Otway-Rees, Yahalom, Woo-Lam, …

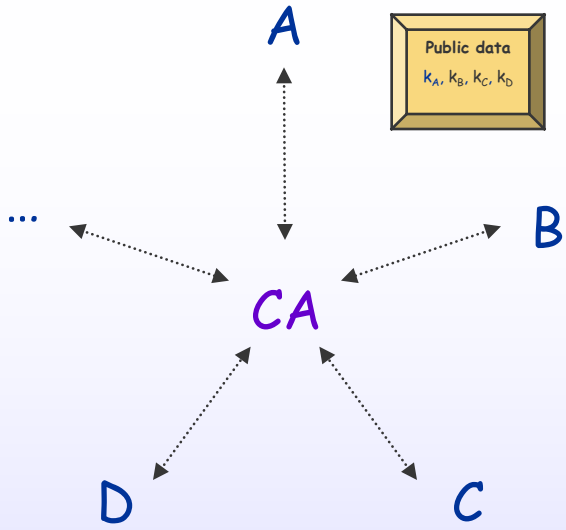
Needham-Schroeder Shared Key



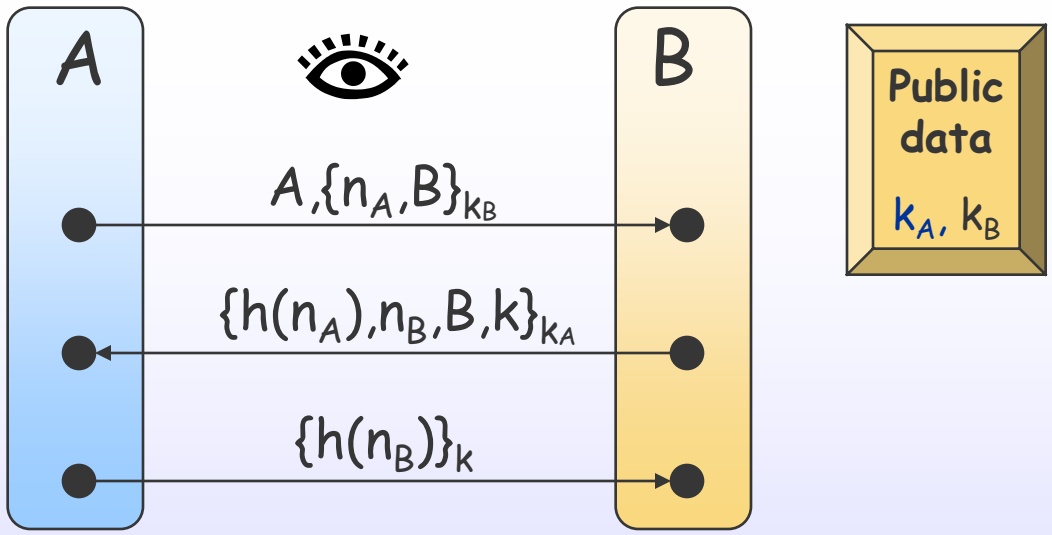
- S creates k_{AB}
- 2 challenge response authenticate A and B

... with Public-Key Infrastructure

- Each principal has a certified public key available to others
- A and B use k_B and k_A to communicate securely
- Examples
  - Bilateral key exchange protocol
  - ...



Bilateral Key Exchange Protocol

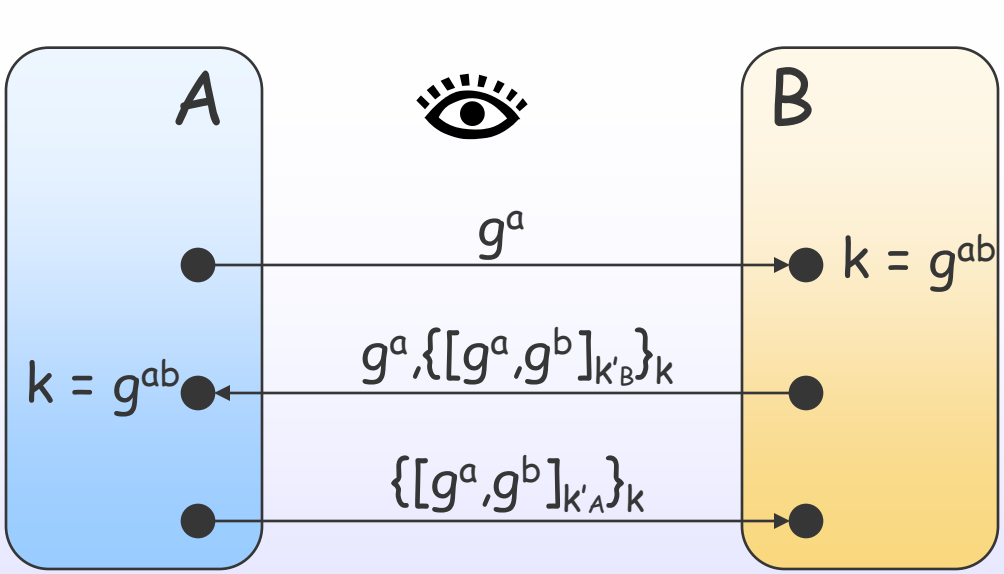


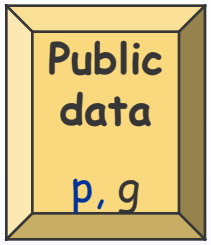
- h is a hash function
- Certificates could be included
- Includes 2 challenge response exchanges

... with Diffie-Hellman

- Diffie Hellman alone cannot guarantee authentication
- Minimum infrastructure required
  - Public key infrastructure for signatures
- Examples
  - Station-to-station protocol
  - Found as option in many big protocols
    - IPSEC, ISAKMP, ...

Station-to-Station Protocol



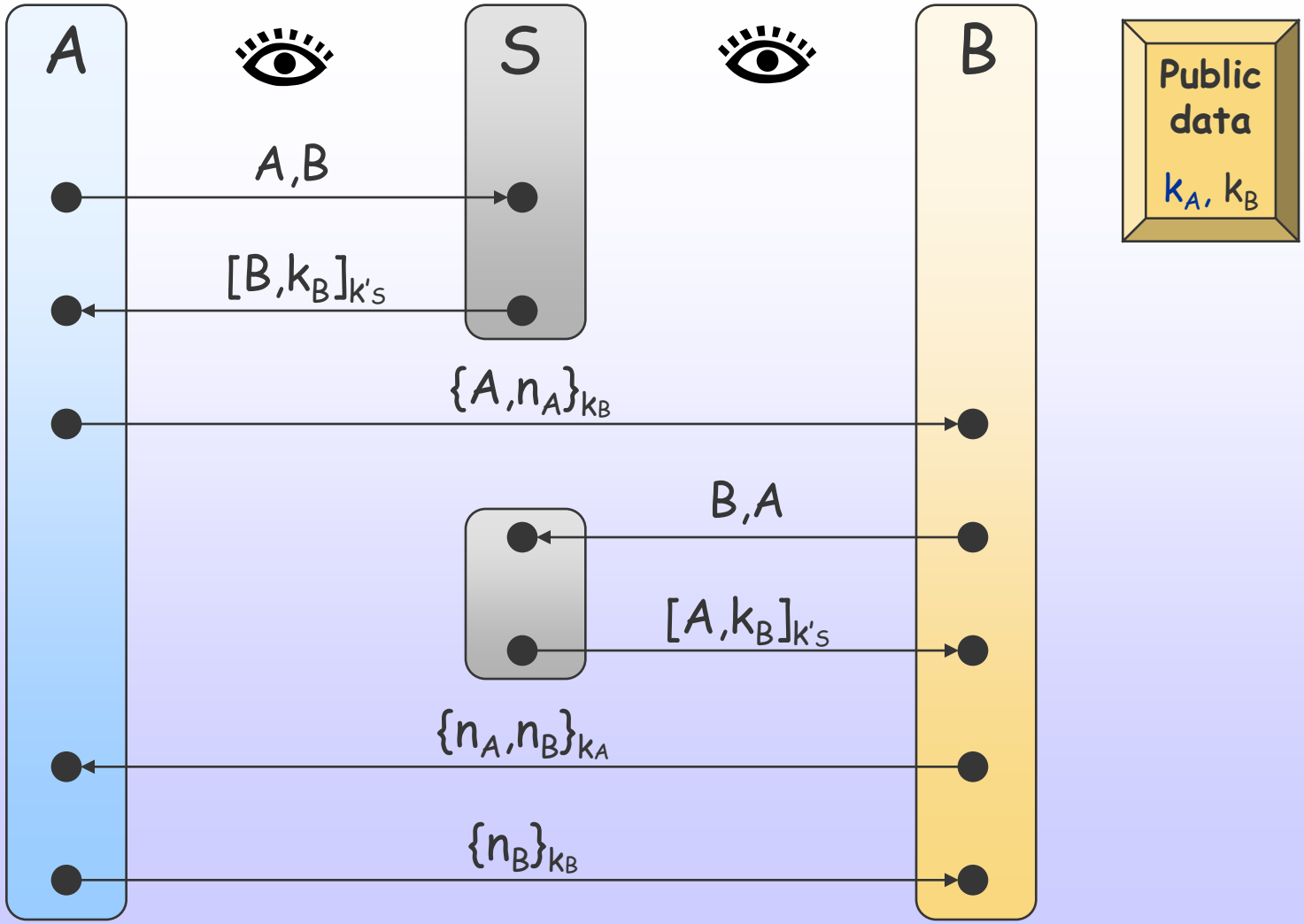


- This is an authenticated Diffie-Hellman
- g^a and g^b used for challenge response
  - Achieves mutual authentication

Key Distribution Protocols

- A and B possess public keys
  - Registered with certification authority
  - Certificates not available
- Request signed certificates from CA
- Examples
  - Needham-Schroeder public-key protocol
    - S acts as key database and CA
    - A and B use nonces for mutual authentication
  - ...

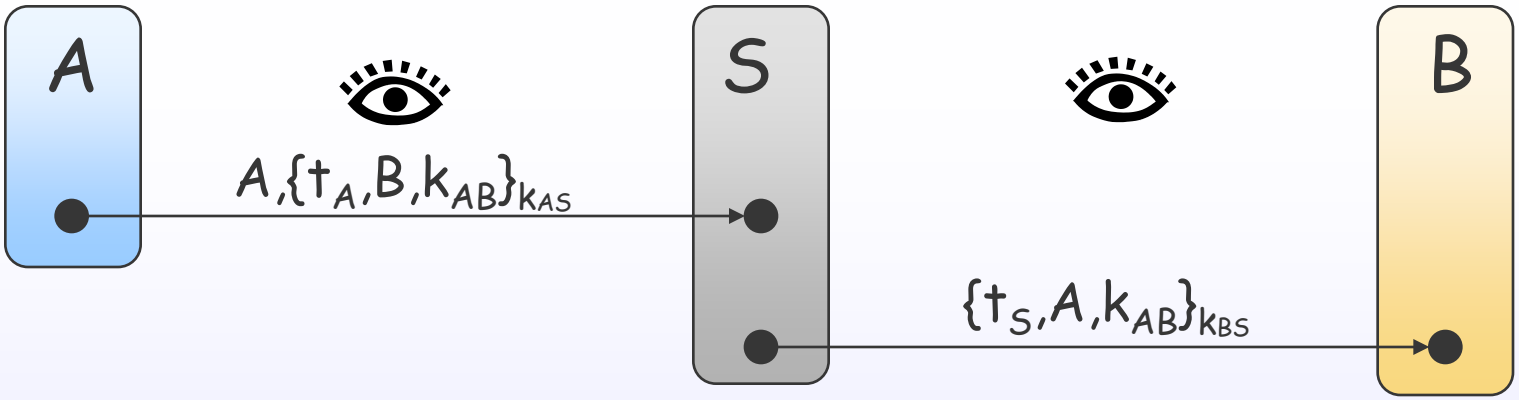
Needham-Shroeder Public Key



Key Translation Protocols

- A wants to send message to B
  … but no server is around to create keys
- A exploits existing channels with a trusted third party S
  - A send m to S encrypted with k_{AS}
  - S forwards m to B encrypted with k_{BS}
  - Timestamps or other mechanisms used for authentication
    - S must be trusted to manipulate them correctly
- Examples
  - Wide-Mouthed Frog protocol

Wide-Mouthed Frog Protocol



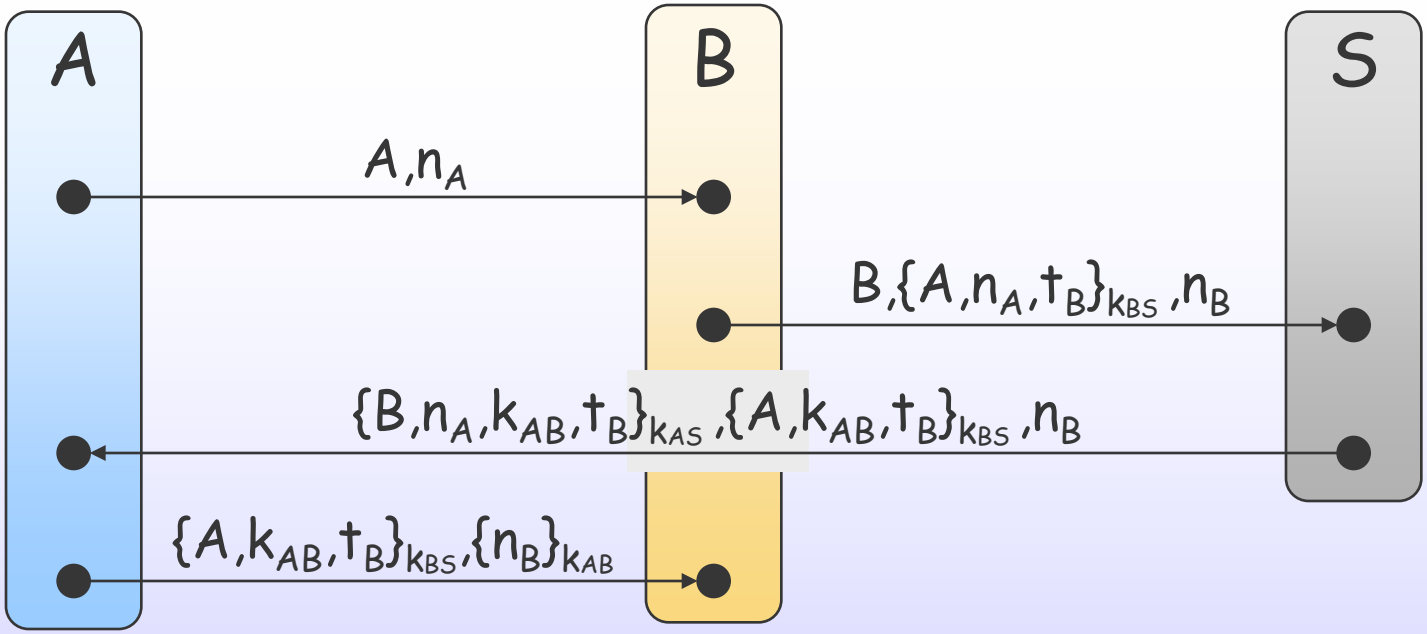
- A generates the key k_{AB}
- S provides trusted timestamping
  - With t_A, A authenticates to S
  - With t_S, S authenticates to B
- A authenticates to B indirectly
- No authentication in the reverse direction

Subprotocols

Useful to add structure to protocols

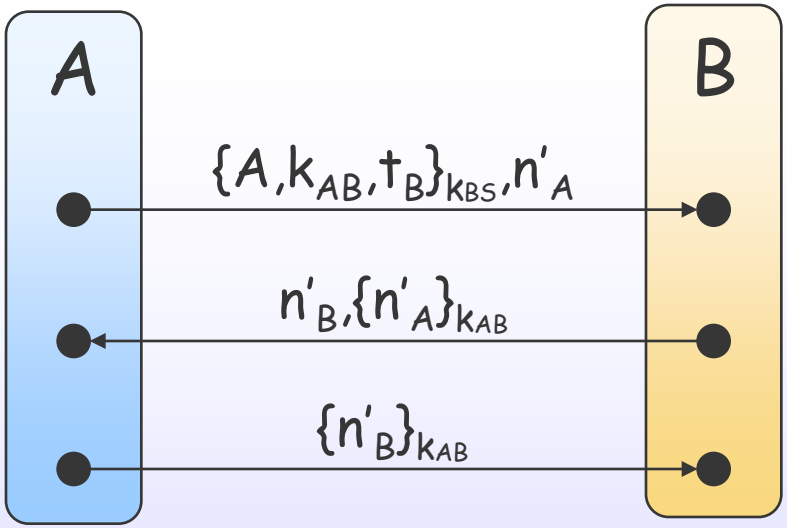
- Deterministic choice of continuation
  - Protocol behaves differently on different inputs
  - Protocols responds to optional requests
- Non-deterministic continuation
  - Protocol flips a coin
  - Protocol can request optional behavior
- Repeated parts
  - Repetitive behavior after initial phase
  - E.g. Neuman-Stubblebine, Kerberos, …

Neuman-Subblebine – Initial Part



- $\{A, k_{AB}, t_B\}_{K_{BS}}$ is A's ticket to access B's service
- n_A and n_B mutually authenticate A and B

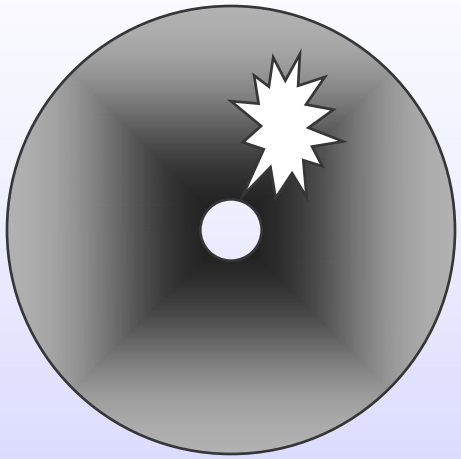
Neuman-Stubbl. – Repeated Part



- A uses ticket to access B's service
  - … until it expires
- n'_A and n'_B reauthenticate A and B

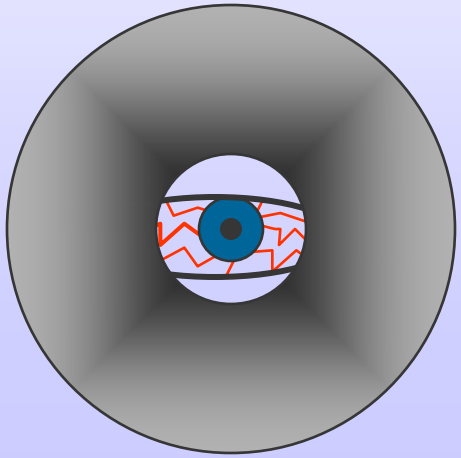
Attacks

Almost all previous protocols have flaws!



- Intruder can break secrecy of the channel

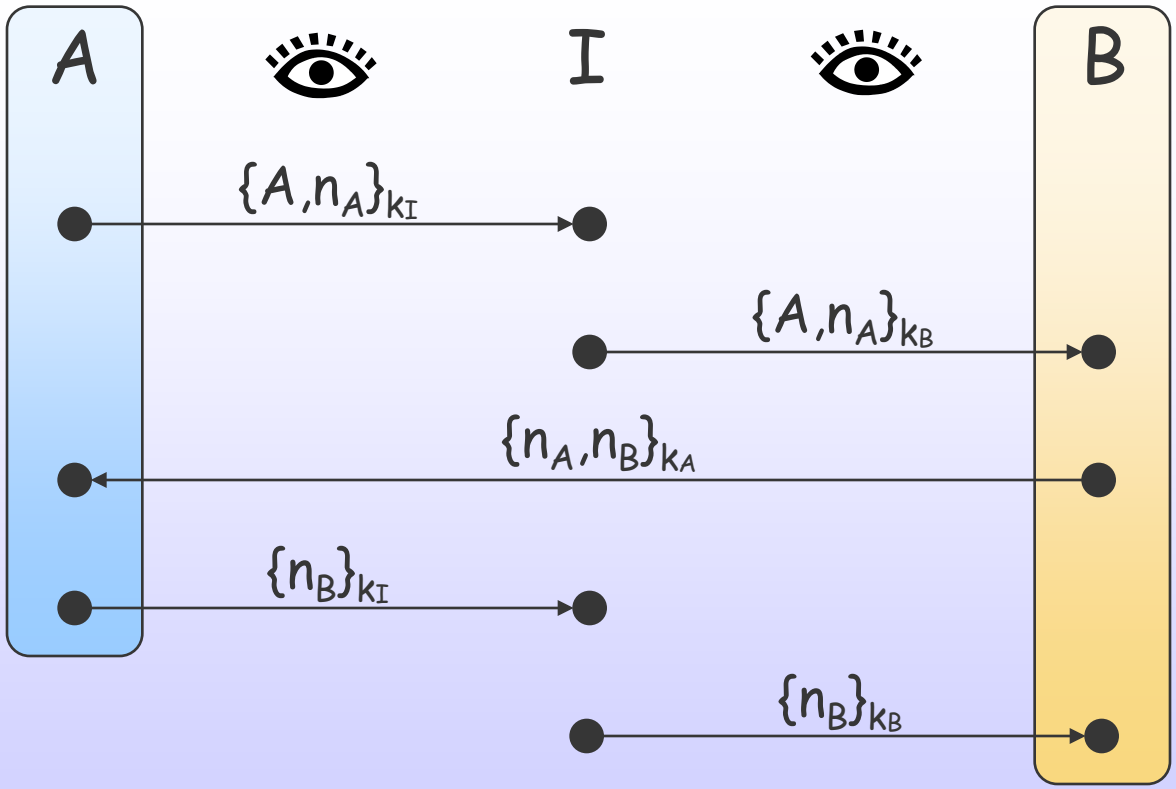
- Intruder can break authentication



Lowe's Attack on NS-PK

$A \rightarrow B: \{A, n_A\}_{k_B}$
$B \rightarrow A: \{n_A, n_B\}_{k_A}$
$A \rightarrow B: \{n_B\}_{k_B}$

NS-PK [3-5]



Public data

k_A, k_B, k_I

Attack discovered 17 years after protocol was published

(Exchanges with S have been omitted)

Man-In-The-Middle Attack

- A wants to talk to B
  - I has replaced k_B with k_I in S's database
  - I acts as a key translator
  - In the end
    - A thinks to be talking to B, but she is talking to I
    - B thinks to be talking to A, but he is talking to I
- A really wants to talk to I
  - I cheats and acts as key translator
  - In the end
    - A knows she talking to I
    - B thinks to be talking to A, but he is talking to I

What happened?

- Protocol assumptions were not specified
  - Intruder is (also) a principal
    - What are the intruder's capabilities anyway?
  - Initial knowledge of principals
  - Meaning of notation
    - Who can access what? How?
- Protocol goals were not specified
  - Failure of mutual authentication …
  - … but A has authenticated I
    - Many people do not agree that this is an attack!

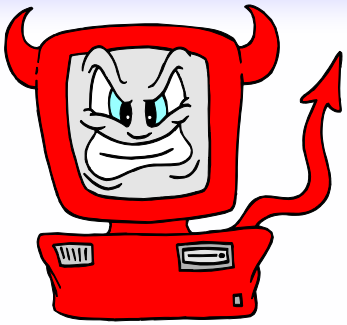
Protocol Specifications

Describe what the protocol does

- For doing implementation
- For doing verification

- 3 aspects
  - Assumptions
    - Initial knowledge
    - Maintained state
    - Environment
    - Intruder
  - Messages exchanged
  - Goals
- Much more in Lecture 7 …

The Dolev-Yao Intruder

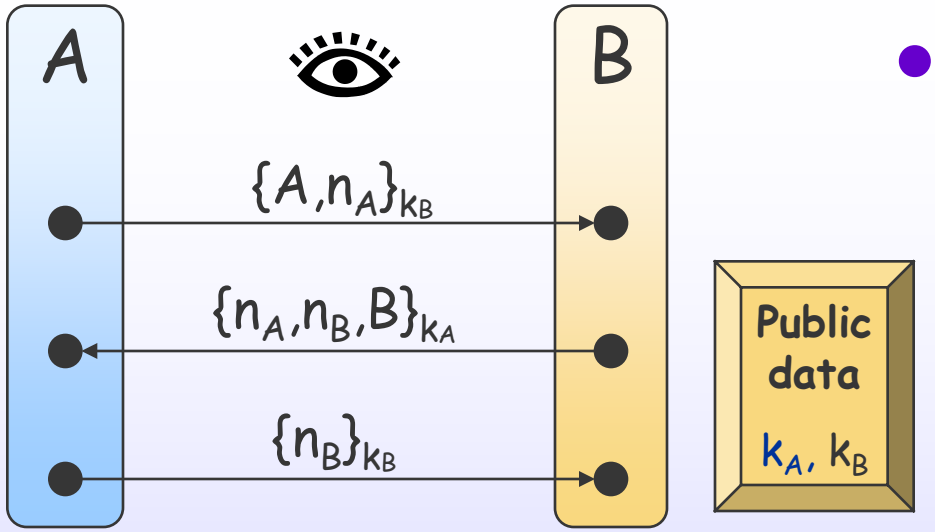


Standard attacker model

- Intercept / Emit messages
- Decrypt / Encrypt *with known key*
- Split / Form pairs
- Look up public information
- Generate fresh data

- Not fully realistic but convenient
- Much more in Lecture 8 …

Lowe's Fix to NS-PK



- Goals
  - Mutual authentication
  - Freshness of nonces
  - Secrecy of nonces

- Assumptions
  - Dolev-Yao intruder
  - I is a principal
  - Principals know public data
  - Public data is correct
  - Private keys uncompromised

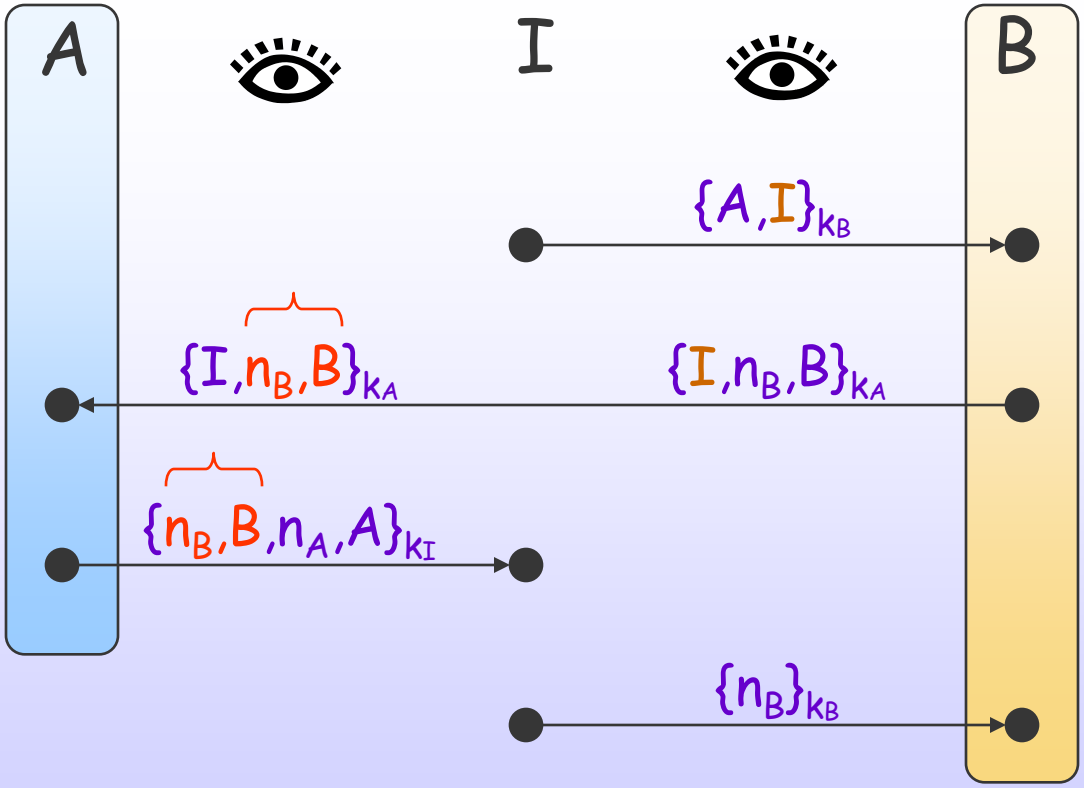
Millen's Attack on NSL

$A \rightarrow B: \{A, n_A\}_{k_B}$

$B \rightarrow A: \{n_A, n_B, B\}_{k_A}$

$A \rightarrow B: \{n_B\}_{k_B}$

Needham-Schroeder-Lowe



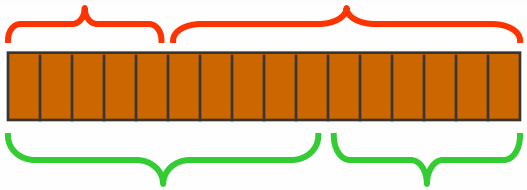
Confusion 1:
name/nonce

Confusion 2:
pair/nonce

B is fooled!

"Unlikely type violation"

Type-Flaw Attacks



- Functionalities seen as "types"
  - Names
  - Nonces
  - Keys, …
- Violation
  - Recipient accepts message as valid …
  - … but imposes different interpretation on bit sequence than sender
- Type flaw/confusion attack
  - Intruder manipulates message
  - Principal led to misuse data

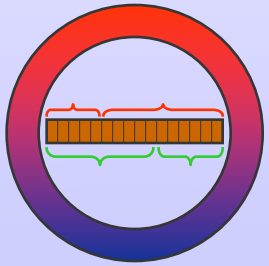
The Dolev-Yao Model of Security

An abstraction for reasoning about protocols

- Not to be confused with the Dolev-Yao intruder … although related
- More on Dolev-Yao model later
- Much more in Lecture 7

- Data are atomic constants
  - No bits
  - Subject to symbolic manipulations

- Tension between type violations and Dolev-Yao model
  - A possible solution in Lecture 8



Some Other Common Attacks

- Freshness
  - I forces stale data in challenge-response
- Parallel session
  - I combines messages from different sessions
- Binding
  - I subverts the public database
- Encapsulation
  - I uses another principal for encryption or decryption
- Cipher-dependent
  - I exploits properties of cryptographic algorithms used
- … and many more

Freshness Attacks

$A \rightarrow S: A,B,n_A$

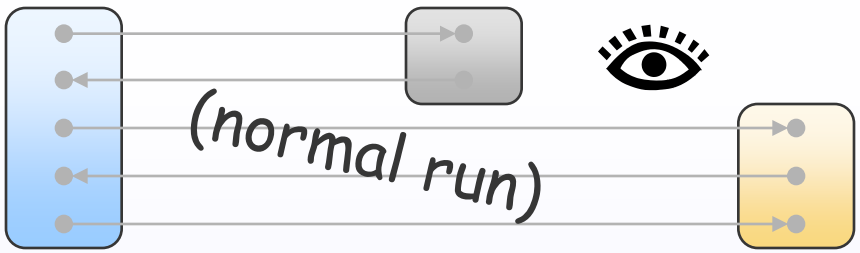
$S \rightarrow A: \{n_A,B,k_{AB}, \{k,n_A\}_{kBS}\}_{kAS}$

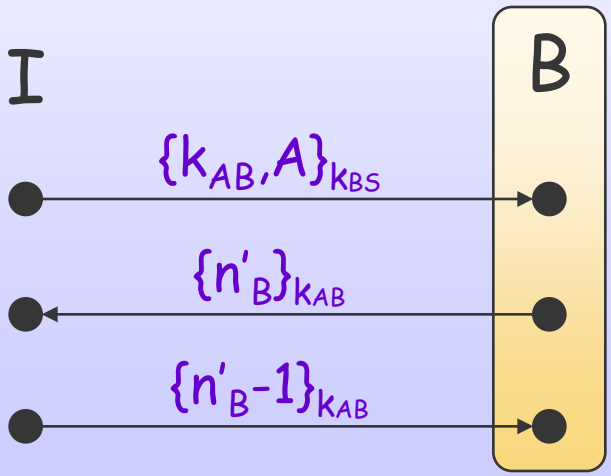
$A \rightarrow B: \{k_{AB},A\}_{kBS}$

$B \rightarrow A: \{n_B\}_{kAB}$

$A \rightarrow B: \{n_B-1\}_{kAB}$

Needham-Schroeder Shared-Key





- I records exchange
- Replays messages in subsequent run

I

  - k_{AB} is a not fresh
    - But B does not know
  - Next messages over k_{AB} are known to I

Parallel Session Attacks

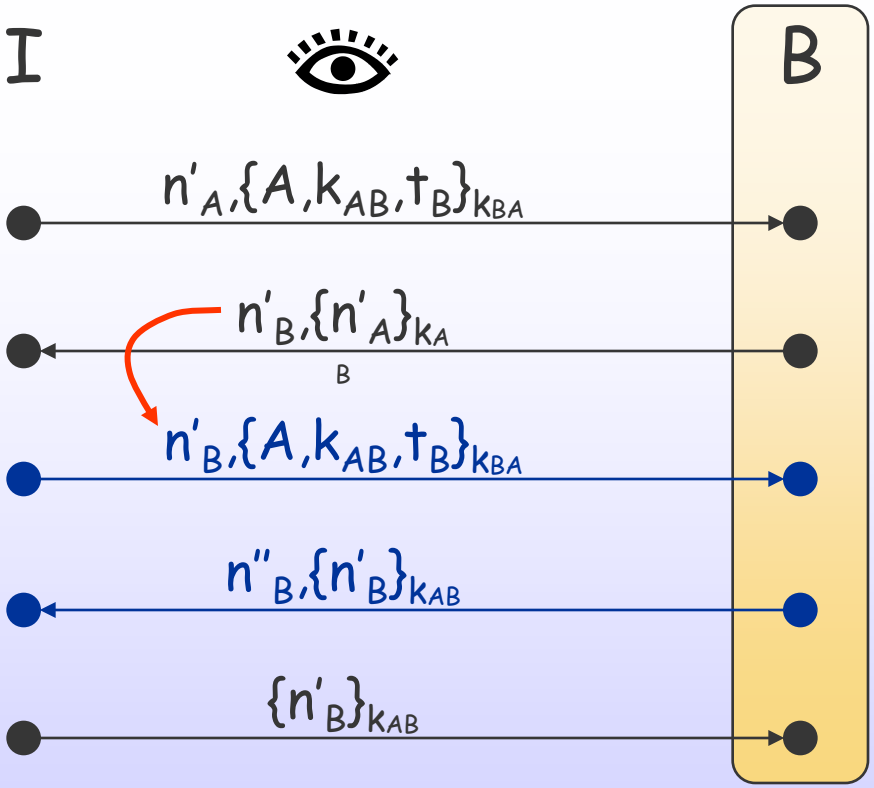
$A \rightarrow B: n'_A, T$

$B \rightarrow A: n'_B, \{n'_A\}_{k_{AB}}$

$A \rightarrow B: \{n'_B\}_{k_{AB}}$

where $T = \{A, k_{AB}, t_B\}_{k_{BS}}$

Neuman-Stubblebine - phase II



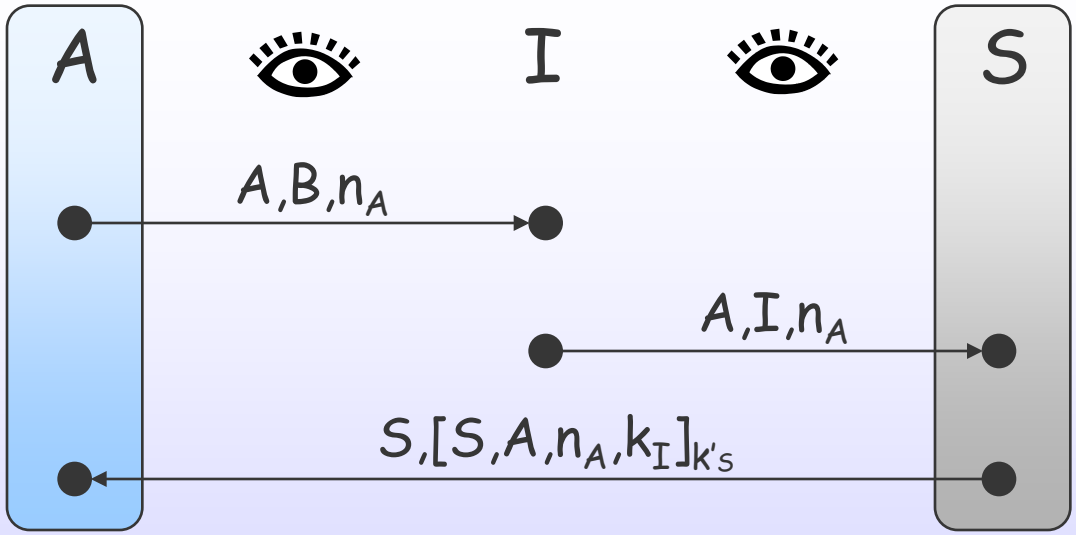
- B things he has authenticated A
- A has not even participated

- I combines messages from 2 sessions

Binding Attacks

$A \rightarrow S: A,B,n_A$

$S \rightarrow A: S,[S,A,n_A,k_B]_{k's}$



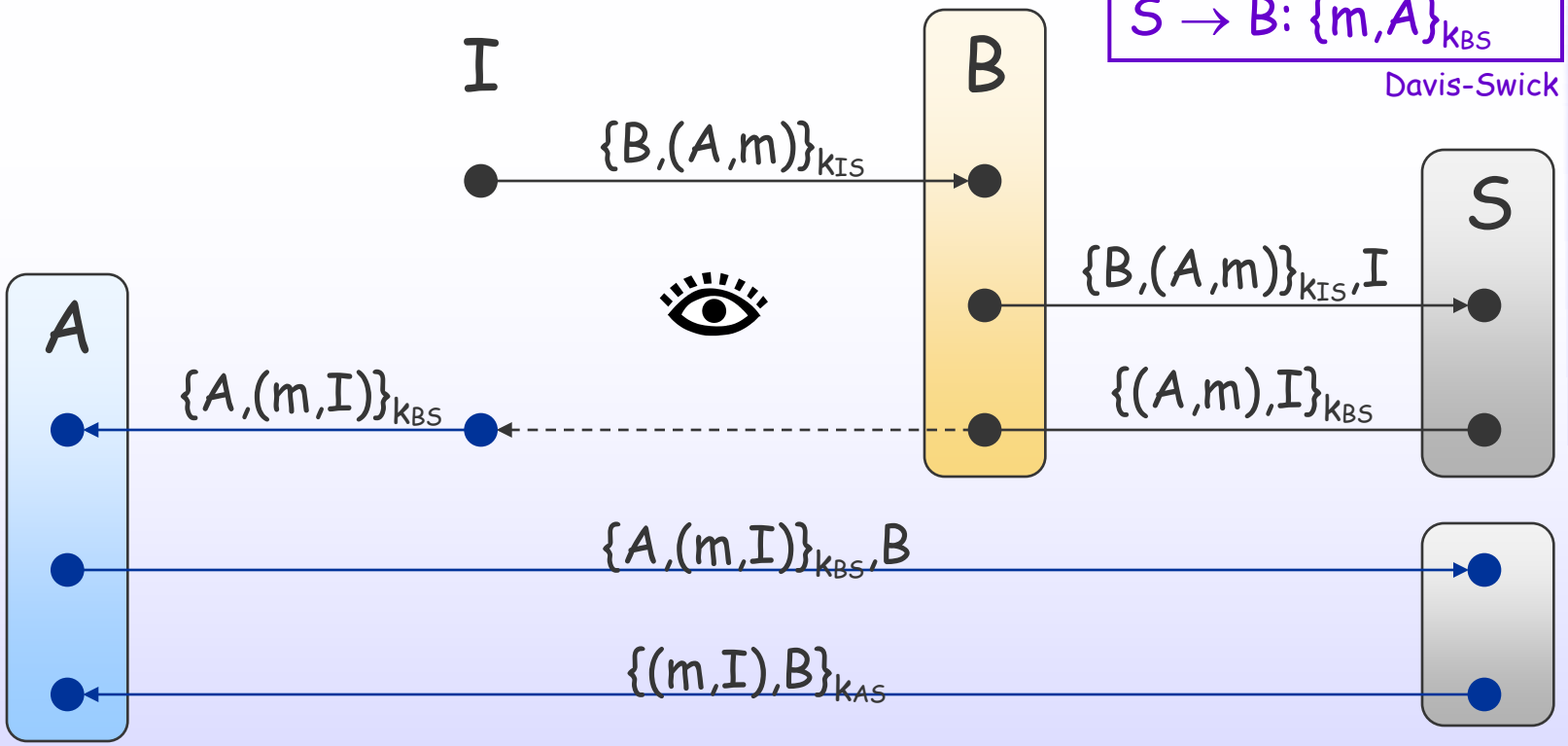
- I convinces A that B's public key is k_I

- I overwrites replies from CA
- I may also overwrite public tables

Encapsulation Attacks

$A \rightarrow B: \{B,m\}_{k_{AS}}$
$B \rightarrow S: \{B,m\}_{k_{AS}},A$
$S \rightarrow B: \{m,A\}_{k_{BS}}$

Davis-Swick



- I uses other principals as cryptographic oracles

- A believes message (m,I) comes from B
- m may include key material

Cipher-Based Attacks

$A \rightarrow S: A,B,n_A$

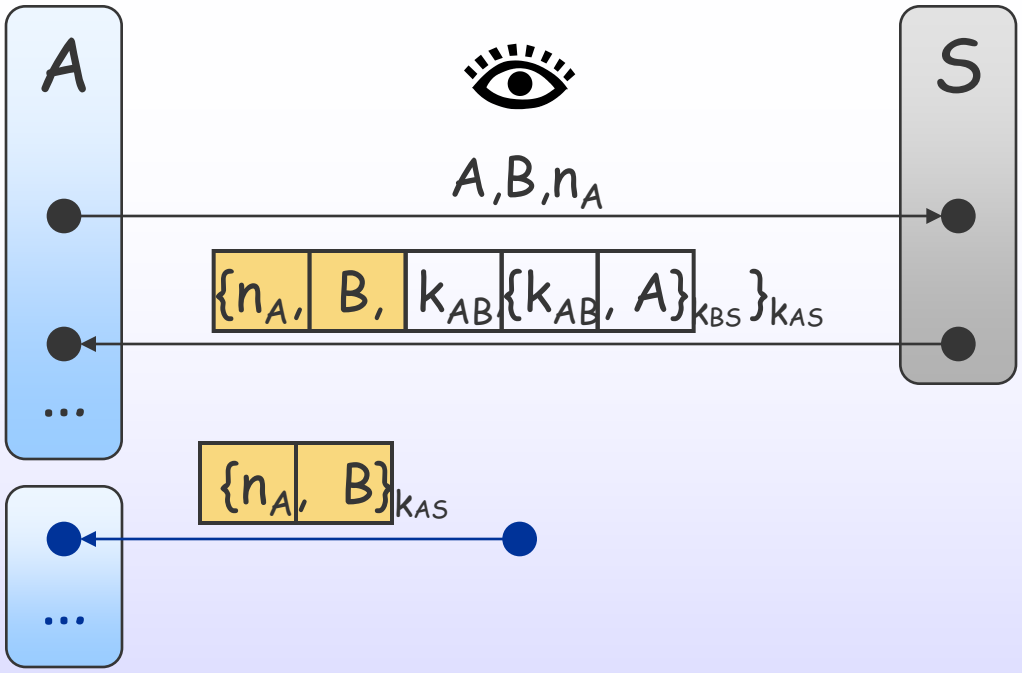
$S \rightarrow A: \{n_A,B,k, \{k_{AB},n_A\}_{kBS}\}_{kAS}$

$A \rightarrow B: \{k_{AB},A\}_{kBS}$

$B \rightarrow A: \{n_B\}_{kAB}$

$A \rightarrow B: \{n_B-1\}_{kAB}$

Needham-Schroeder Shared-Key



- Prefix of CBC is valid

Here also

- Parallel session
- Type flaw

- I exploits particular cipher in use
- I exploits implementation of cipher

Black-Box Cryptography

Most attacks are independent from details of cryptography

Another aspect of Dolev-Yao model

- No first-class notion of ciphertext
- $\{m\}_k$ is a term
- m accessible in $\{m\}_k$ only if k is known
  - No guessing of bits

- Bridging the gap between
  - cryptographic algorithms and
  - Dolev-Yao model

  Several proposal, no definite solution
    - Not covered in this course

Further Issues

- Mixing protocols
  - Protocols may appear safe in isolation
  - … but have nasty interactions when mixed
    - Several protocols coexist in a system
- Composing protocols
  - In parallel
  - In sequence

  Modularity would help
    - Little composability

Getting Protocols Right

- Testing
  - Not a solution!
    - Assumes statistical distribution of errors
    - Security is about worst-case scenario
- Formal verification
  - Hard – See lectures 7, 8, 9
- Attack-free construction
  - Rules-of-thumb
  - Formal criteria
  - A few automated tools

Authent.
Ch.-Resp.
Key gen.
Key Distr.
M-I-T-M
Type flaw
Other
Design

Design Principles

[Abadi,Needham]

- Aimed at
  - Avoiding many mistakes
  - Simplifying protocols
  - Simplifying formal analysis
- Tested on many published examples
- Works beyond authentication
- Attempted
  - Formalizations
  - Automations

"Prudent Engineering Practice"

- Every message should say what it means
  - Include identity of principal if important for meaning
    - See Needham-Schroeder Public Key
  - Be clear as to why encryption is being done
    - Encryption is not synonymous with security
    - Double encryption is no cause for optimism
- Be clear about
  - trust relations protocol depends on
  - properties assumed about nonces
    - Good for freshness, not always association
- A principal may not knows the contents of encrypted material he signed
- … and a few more

In Summary [Abadi]

- Be explicit
  - Include sufficient proof of freshness
  - Include sufficient names
  - Do not count on context
  - Use evident classifications
- Do not send secret data on public channels
- Distinguish secret input from public inputs
- Secrets should be strong enough for data they protect
- Do not expect attackers to obey rules
- Cryptography does not imply security

Fail-Stop Protocols

[Syverson]

Tempering any message causes abort of the protocol

- No further message sent

- Authentication is automatic
- Active attacker cannot force secret to be released

- Extensible Fail-Stop Protocols
  - If appending message always yield fail-stop
    - Immune from replay
    - Closed w.r.t. sequential and parallel composition

Constructing a Fail-Stop Protocol

- Each message contains header with
  - Identity of sender and receiver
  - Protocol identifier
  - Sequence number
  - Freshness identifier
- Each message encrypted with shared key between sender and recipient
- Honest principals
  - Follow protocol
  - Ignore unexpected messages
  - Halts if expected message does not arrive in time

Readings

- Dieter Gollmann, *Authentication – Myths and Misconception*, 2001
- J. Clark and J. Jacob, *A Survey of Authentication Protocol Literature: Version 1.0*, 1997
- M. Abadi and R. Needham, *Prudent Engineering Practice for Cryptographic Protocols*, 1994
- L. Gong and P. Syverson, *Fail-Stop Protocols, an Approach to Designing Secure Protocols*, 1994

Exercises for Lecture 4

- Find a parallel session attack on the handshake
  - $A \rightarrow B : \{n_A\}_{k_{AB}}$
  - $B \rightarrow A : \{n_A+1\}_{k_{AB}}$
- Fix the key distribution protocol on slides 41
- Find a type flaw attack on the Yahalom protocol
  - $A \rightarrow B : A, n_A$
  - $B \rightarrow S : B, \{A, n_A, n_B\}_{k_{BS}}$
  - $S \rightarrow A : \{B, k_{AB}, n_A, n_B\}_{k_{AS}}, \{A, k_{AB}\}_{k_{BS}}$
  - $A \rightarrow B : \{A, k_{AB}\}_{k_{BS}}, \{n_B\}_{k_{AB}}$

Next ...

- Case Study I: Kerberos V