EE488
Introduction to
Cryptography Engineering

Yongdae Kim

Blind signature scheme

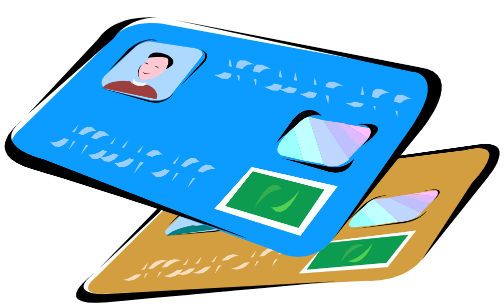
- Chaum for Electronic Cash

- Sender A; Signer B
- B's RSA public and private key are as usual. k is a random secret integer chosen by A, satisfying $0 \le k < n$
- Protocol actions
  - (blinding) A: comp $m^* = mk^e \bmod n$, to B
    Note: $(mk^e)^d = m^d k$
  - (signing) B comp $s^* = (m^*)^d \bmod n$, to A
  - (unblinding) A: computes $s = k^{-1}s^* \bmod n$

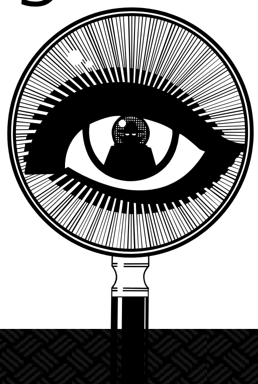
Identification

Basis of identification

- Something *known* - passwords, PINs, keys…
  - *a^*ehk3&(dAs*
- Something *possessed* - cards, handhelds…

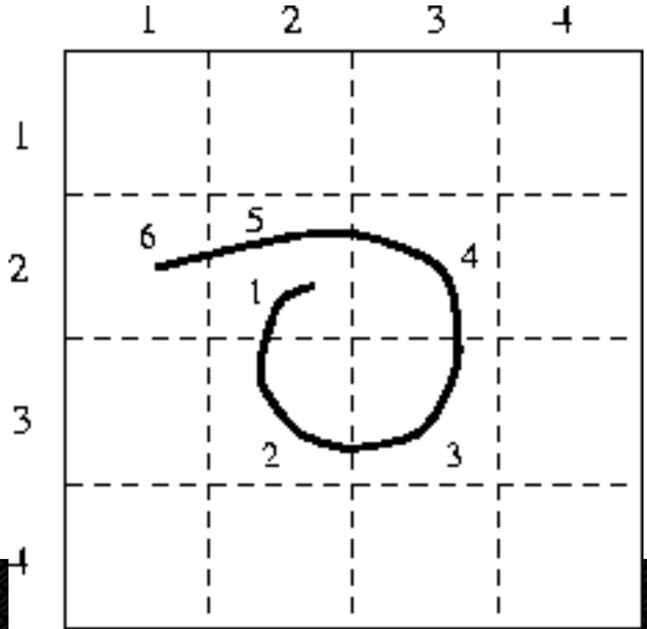
- Something *inherent* - biometrics

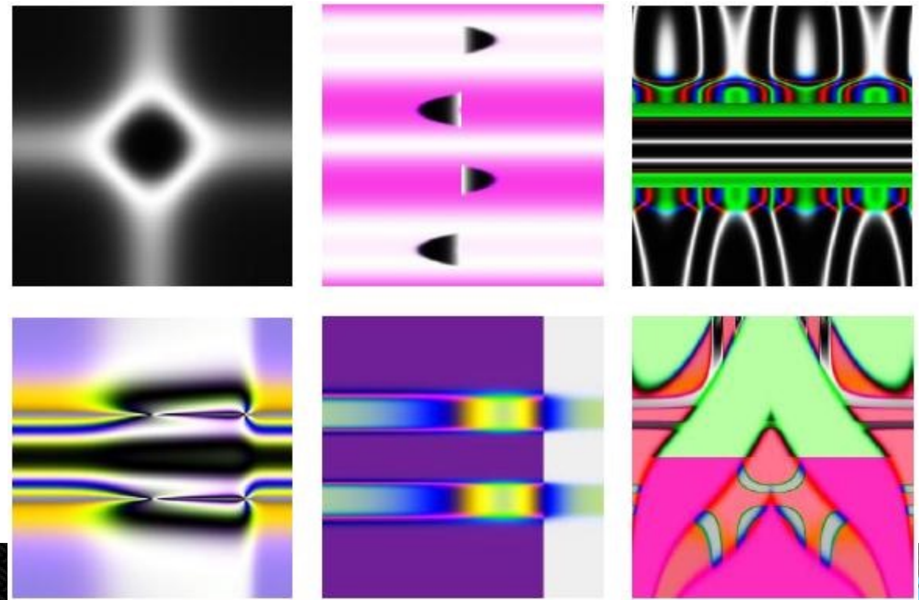
   

PINs and keys

- Long key on physical device (card), short PIN to remember
- PIN unlocks long key
- Need possession of both card and PIN
- Provides ***two-level*** security (or two-factor authentication)

Other password: graphical





Lamport's One Time Passwords

- User has a secret *w*
  - Using a OWF *h*, create the password sequence:
    $w, h(w), h(h(w)), \cdots, h^t(w)$
  - Bob knows only $h^t(w)$
  - Password for *i* -th identification is: $w_i = h^{t-i}(w)$

- Attacks
  - ***Pre-play attack*** - Eve intercepts an unused password and uses it later
  - Make sure you're giving password to the right party
  - Bob must be *authenticated*

Another one-time password

- Stores actual passwords on system side
- Alice and Bob share a password *P*
- Alice: generate *r,* send to Bob: *(r, h(r, P))*
- Check: Bob computes *h(r, P)*, from given r, and local copy of P.
- Security
  - Works only if *r* is something that will only be accepted once (else replay attack!)
  - Any other?

Challenge-response authentication

- Alice is identified by a *secret* she possesses
  - *Bob* needs to know that Alice does indeed possess this secret
  - *Alice* provides ***response*** to a time-variant ***challenge***
  - Response depends on ***both*** secret and challenge

- Using
  - Symmetric encryption
  - One way functions
  - Public key encryption
  - Digital signatures

Challenge Response using SKE

- Alice and Bob share a key *K*
- Taxonomy
  - ***Unidirectional*** authentication using ***timestamps***
  - ***Unidirectional*** authentication using ***random numbers***
  - ***Mutual*** authentication using ***random numbers***
- Unilateral authentication using timestamps
  - Alice → Bob: $E_K(t_A, B)$
  - Bob decrypts and verified that timestamp is OK
  - Parameter *B* prevents replay of same message in B → A direction

Challenge Response using SKE

- Unilateral authentication using random numbers
  - Bob → Alice: r_b
  - Alice → Bob: $E_K(r_b, B)$
  - Bob checks to see if r_b is the one it sent out
    - Also checks "*B*" - prevents reflection attack
  - r_b must be ***non-repeating***
- Mutual authentication using random numbers
  - Bob → Alice: r_b
  - Alice → Bob: $E_K(r_a, r_b, B)$
  - Bob → Alice: $E_K(r_a, r_b)$
  - Alice checks that r_a, r_b are the ones used earlier

Challenge-response using OWF

- Instead of encryption, used keyed MAC h_K
- Check: compute MAC from *known quantities,* and check with message
- SKID3
  - Bob → Alice: r_b
  - Alice → Bob: r_a, $h_K(r_a, r_b, B)$
  - Bob → Alice: $h_K(r_a, r_b, A)$

Challenge-response using PKE

- Mutual Authentication based on PK decryption
  - Alice → Bob: $P_B(r_A, B)$
  - Bob → Alice: $P_A(r_A, r_B)$
  - Alice → Bob: r_B

Challenge-response using DS

- Timestamp-based
  - Alice → Bob: $cert_A$, t_A, B, $S_A(t_A, B)$
  - Bob checks:
    - » Timestamp OK
    - » Identifier "B" is its own
    - » Signature is valid (after getting public key of Alice using certificate)
- Mutual Authentication using Signatures
  - Bob → Alice: r_B
  - Alice → Bob: $cert_A$, r_A, B, $S_A(r_A,r_B,B)$
  - Bob → Alice: $cert_B$, A, $S_B(r_A,r_B,A)$

Quiz Q&A

- Junho 1
  - differential crypt analysis
  - ZKP, lattice crypto
  - QC breaks prime factorization and DLP.
  - SEED instead of AES
- Junho 2
  - Riemanian hypothesis and factorization
- Jein
  - PKE not based on DLP or factorization
- Jaehong
  - Time, # of messages, IP instead of random number
- Beomsu
  - SKT incidents
- Chanho
  - When do we use crypto?
- Martin
  - Security vs. crypto
- Jungwoo
  - Why light weight homomorphic encryption difficult?
- Samuel
  - ML for breaking crypto

Key Establishment

Terms

- (Implicit) Key authentication
  - Assurance that no other party aside from a specifically identified second party may gain access to a secret key
- Key confirmation
  - one party is assured that a second party actually has possession of a particular secret key
- Explicit key authentication
  - both (implicit) key authentication and key confirmation
- authenticated key establishment
  - key establishment + key authentication
- Session key
  - ephemeral secret, i.e., one whose use is restricted to short time period after which all trace of it is eliminated

Assumptions, Adversaries

- Attacks
  - passive attack: adversary simply records data, analyze
  - active attack: adversary modifies or injects messages
- What are the attacker's roles?
  - deduce a session key using info gained by tapping
  - participate covertly in protocol initiated by one party, and influence it by altering messages to deduce the key
  - initiate protocol executions and combine messages from one with another so as to carry out above attacks
  - without deducing the key, deceive good party regarding the identity of the party with which it shares a key

PFS and Known Key Attacks

- perfect forward secrecy
  - break long-term key !⇒ break past session keys
  - previous traffic is locked securely in the past
  - generating session keys by DH key agreement, wherein DH exponentials are based on short-term keys
  - If long-term secrets are compromised, future session can be impersonated
- known-key attack
  - compromise of past session keys allows either a passive adversary to compromise future session keys, or impersonation by an active adversary in the future.
  - in some environments, the probability of compromise of session keys may be greater than that of long-term keys

Point-to-Point Key Update

- Key Transport with one pass
  - $A \rightarrow B: E_K(r_A)$
  - Implicit key authentication
  - Additional field
    - timestamp, sequence number: freshness
    - redundancy: explicit key authentication, message modification
    - target identifier: prevent undetectable message replay
  - Hence $A \rightarrow B: E_K(r_A, t_A, B)$
  - Mutual authentication: $B \rightarrow A: E_K(r_B, t_B, A)$: $K = f(r_A, r_B)$
- Key Transport with challenge-response
  - $B \rightarrow A: n_B$: for freshness
  - $A \rightarrow B: E_K(r_A, n_A, n_B, B)$
  - $B \rightarrow A: E_K(r_B, n_B, n_A, A)$
  - Cannot provide PFS
- Authenticated Key Update Protocol
  - $A \rightarrow B: r_A$
  - $B \rightarrow A: (B, A, r_A, r_B), h_K(B, A, r_A, r_B)$
  - $A \rightarrow B: (A, r_B), h_K(A, r_B)$
  - $W = h'_{K'}(r_B)$

Shamir's no key algorithm

- Protocol
  - $A \rightarrow B$: $K^A \bmod p$
  - $B \rightarrow A$: $(K^A)^B \bmod p$
  - $A \rightarrow B$: $(K^{AB})^{A^{-1}} \bmod p$
- Property
  - Provide key transport
  - No a priori information is required
  - Not necessarily modular exponentiation, but not one-time pad

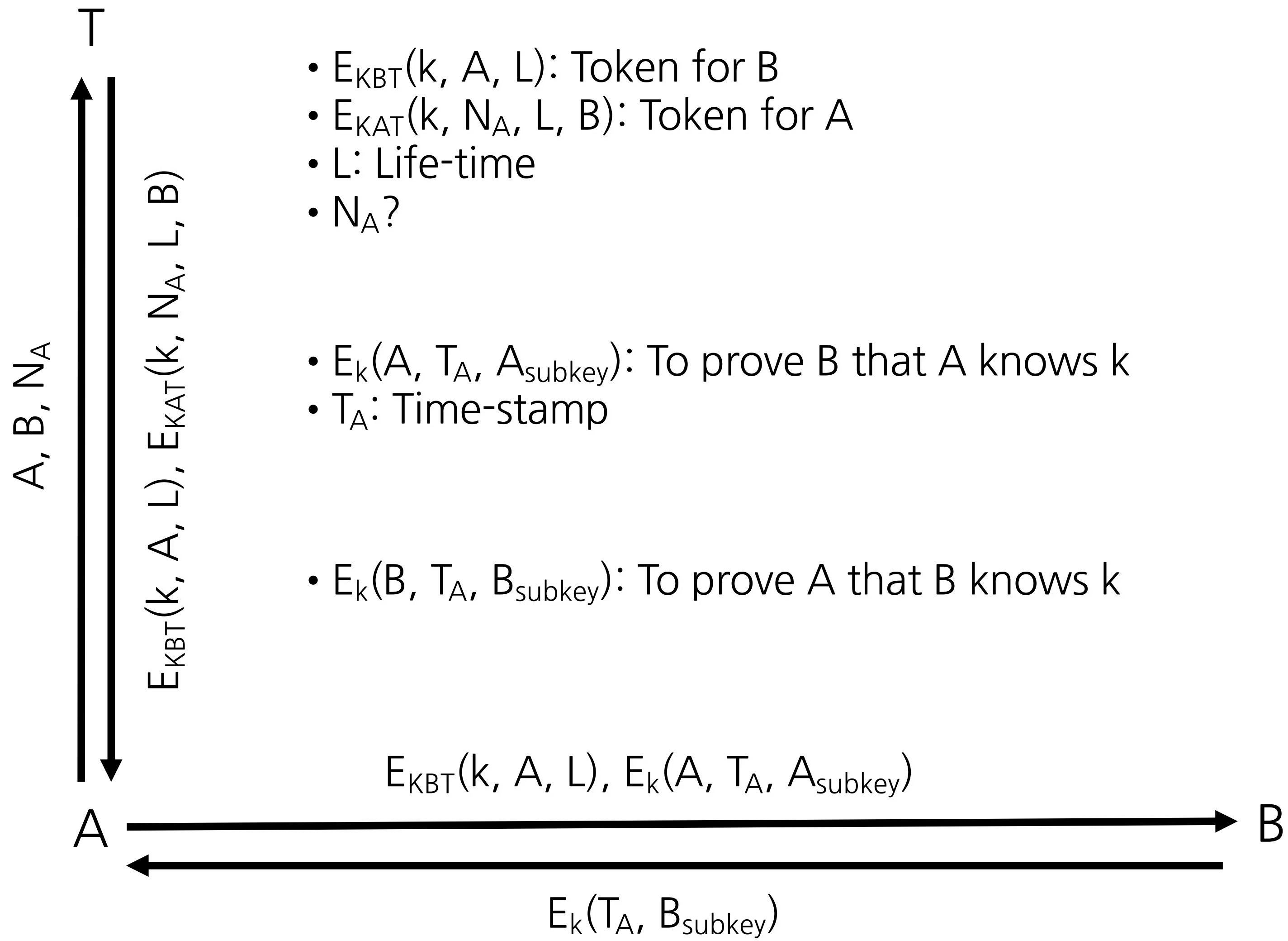
Kerberos

- Basic
  - A, B, a TTP share long-term pairwise secret keys a priori
  - TTP either plays the role of KDC and itself supplies the session key, or serves as a key translation center (KTC)
  - A and B share no secret, T shares a secret with each
  - Goal: for B to verify A's identity, establishing shared key
- Description
  - A requests for credential to allow it to authenticate itself
  - T plays the role of a KDC, returning to A a session key encrypted for A and a ticket encrypted for B
  - The ticket contains the session key and A's identity

Kerberos (cnt.)

- Protocol
  - $A \rightarrow T$: A, B, N_A — N_A: freshness
  - $T \rightarrow A$: $E_{KAT}(k, A, L)$, $E_{KBT}(k, N_A, L, B)$: — L: lifetime
  - $A \rightarrow B$: $E_{KBT}(k, A, L)$, $E_k(A, T_A, A_{subkey})$
  - $B \rightarrow A$: $E_k(T_A, B_{subkey})$ — Optional mutual authentication
- Properties
  - secure and synchronized clocks
  - If password-based, protocol is susceptible to password-guessing attack
  - A_{subkey} and B_{subkey} allow transfer of a key from A to B
  - Lifetime is intended to allow A to re-use the ticket

Kerberos

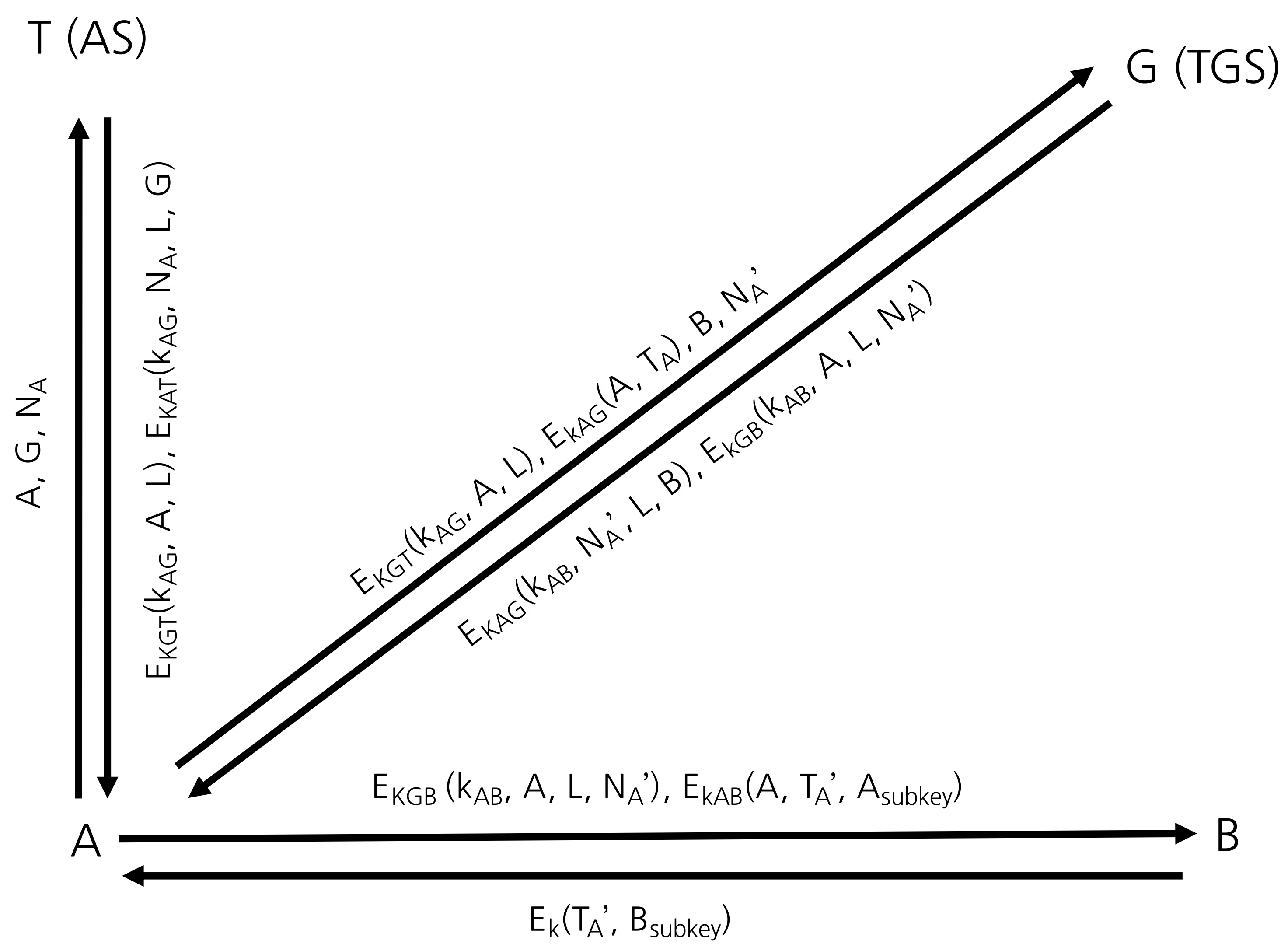


- $E_{KBT}(k, A, L)$: Token for B
- $E_{KAT}(k, N_A, L, B)$: Token for A
- L: Life-time
- N_A?

- $E_k(A, T_A, A_{subkey})$: To prove B that A knows k
- T_A: Time-stamp

- $E_k(B, T_A, B_{subkey})$: To prove A that B knows k

Kerberos (scalable)



Key Transport using PKC

- Needham-Schroeder
  - Algorithm
    - $A \rightarrow B: P_B(k_1, A)$
    - $B \rightarrow A: P_A(k_1, k_2, B)$
    - $A \rightarrow B: P_B(k_2)$
  - Properties: Mutual authentication, mutual key transport

- Modified NS
  - Algorithm
    - $A \rightarrow B: P_B(k_1, A, r_1)$
    - $B \rightarrow A: P_A(k_2, r_1, r_2)$
    - $A \rightarrow B: r_2$
  - Removing third encryption

Key Transport using PKC

- Needham-Schroeder
  - Algorithm
    - » $A \rightarrow B: P_B(k_1, A)$
    - » $B \rightarrow A: P_A(k_1, k_2, B)$
    - » $A \rightarrow B: P_B(k_2)$
- Modified NS
  - Algorithm
    - » $A \rightarrow B: P_B(k_1, A, r_1)$
    - » $B \rightarrow A: P_A(k_2, r_1, r_2)$
    - » $A \rightarrow B: r_2$
  - Removing third encryption
- Encrypting signed keys
  - $A \rightarrow B: P_B(k, t_A, S_A(B, k, t_A))$
  - Data for encryption is too large
- Encrypting and signing separately
  - $A \rightarrow B: P_B(k, t_A), S_A(B, k, t_A)$
  - Acceptable only if no information regarding plaintext data can be deduced from the signature
- Signing encrypted keys
  - $A \rightarrow B: t_A, P_B(A, k), S_A(B, t_A, P_B(A, k))$
  - Prevent the above problem
  - Can provide mutual authentication

Combining PKE and DS

- Assurances of X.509 strong authentication
  - identity of A, and the token received by B was constructed by A
  - the token received by B was specifically intended for B;
  - the token received by B has “freshness”
  - the mutual secrecy of the transferred key.
- X.509 strong authentication
  - D_A=(t_A, r_A, B, $data_1$, $P_B(k_1)$), D_B=(t_B, r_B, A, r_A, $data_2$, $P_A(k_2)$),
  - $A \rightarrow B$: $cert_A$, D_A, $S_A(D_A)$
  - $B \rightarrow A$: $cert_B$, D_B, $S_B(D_B)$
- Comments
  - Since protocol does not specify inclusion of an identifier within the scope of the encryption P_B within D_A, one cannot guarantee that the signing party actually knows (or was the source of) plaintext key

Hybrid Key Transport (PKE)

- Beller-Yacobi (4 pass)
  - Properties
    - mutual authentication, explicit key authentication
    - for applications where there is imbalance in processing power
    - identity of the weaker remains concealed from eavesdroppers
  - Algorithm
    - $B \rightarrow A$: $cert_B = (I_B, n_B, G_B)$: certificate generated with RSA
    - $A \rightarrow B$: $P_B(K) = K^3 \bmod n_B$
    - $B \rightarrow A$: $E_K(m, \{0\}^t)$: Encryption with symmetric key encryption
    - $A \rightarrow B$: $E_K((v, w), cert_A)$: DSA signature with precomputation
  - Comment
    - To achieve mutual authentication, each party carry out at least one private-key operation, and one or two public-key operations
    - careful selection of two separate public-key schemes
    - RSA PKE and ElGamal signature are cheap

Hybrid Key Transport (PKE)

- Beller-Yacobi (2 pass)
  - Algorithm (RSA vs. ElGamal again?)

Terminal A		Server B
precompute x, $v = g^x \bmod n_S$		select random challenge m
verify $cert_B$ via $P_T(G_B)$	$\leftarrow$	send m, $cert_B$
compute $(v, w) = S_A(m, I_B)$		$cert_B = (I_B, n_B, G_B)$
send $P_B(v)$, $E_v(cert_A, w)$	$\rightarrow$	recover v, set K = v
$cert_A = (I_A, u_A, G_A)$		verify $cert_A$, signature (v, w)

  - I_M: Identity of M, G_M: Certificate of M, u_A: ElGamal public key of A, n_B: RSA modulus
  - Properties: slightly weaker authentication assurances
    - » B obtains entity authentication of A and obtains a key K that A alone knows, while A has key authentication with respect to B
    - » For A to obtain explicit key authentication of B, a third message may be added whereby B exhibits knowledge through use of K on a challenge or standard message (e.g., $\{0\}^t$)

Contents

- Classification and framework
- Key transport based on symmetric encryption
- Key agreement based on symmetric techniques
- Key transport based on public-key encryption
- Key agreement based on asymmetric techniques
- Analysis of key establishment protocols

Diffie-Hellman

- Diffie-Hellman
  - Setup: prime p, generator g of Z_p*
  - $A \rightarrow B$: $g^x \bmod p$
  - $B \rightarrow A$: $g^y \bmod p$
  - Properties
    - fixed exponent: zero-pass key agreement with special certificate
    - Authentication is required

MTI/A0

- Protocol
  - $A \rightarrow B$: $g^x \bmod p$
  - $B \rightarrow A$: $g^y \bmod p$
  - A: $k = (g^y)^a PK_b{}^x = g^{ya} g^{bx} = g^{ya+bx}$
  - B: $k = (g^x)^b PK_a{}^y$
  - source-substitution attack: C is not actually able to compute k itself, but rather causes B to have false belief
    - C registers A's public key as its own
    - When A sends B, C replaces A's certificate with its own
    - C forwards B's response g^y to A
    - B concludes that subsequently received messages encrypted by $k = g^{bx+ay}$ originated from C, it is only A who knows k and can originate such messages

STS

- Algorithm
  - $A \rightarrow B$: $g^x \bmod p$
  - $B \rightarrow A$: $g^y \bmod p$, $E_k(S_B(g^y, g^x))$
  - $A \rightarrow B$: $E_k(S_A(g^x, g^y))$
- Properties
  - Encryption under key k provides mutual key confirmation plus allows the conclusion that the party knowing the key is that which signed the exponentials.

Contents

Attack strategies and classic flaws

- "man-in-the-middle" attack on unauthenticated DH
- Reflection attack
  - Original protocol
  1. $A \rightarrow B : r_A$
  2. $B \rightarrow A : E_k(r_A, r_B)$
  3. $A \rightarrow B : r_B$
  - Attack
  1. $A \rightarrow E : r_A$
  2. $E \rightarrow A : r_A$: Starting a new session
  3. $A \rightarrow E : E_k(r_A, r_A')$: Reply of (2)
  4. $E \rightarrow A : E_k(r_A, r_A')$: Reply of (1)
  5. $A \rightarrow E : r_A'$
  - prevented by using different keys for different sessions

Attack strategies and classic flaws

- Interleaving attacks
  - To provide freshness and entity authentication
  - Flawed protocol
    1. $A \rightarrow B : r_A$
    2. $B \rightarrow A : r_B, S_B(r_B, r_A, A)$
    3. $A \rightarrow B : r_A', S_A(r_A', r_B, B)$
  - Attack
    1. $E \rightarrow B : r_A$
    2. $B \rightarrow E : r_B, S_B(r_B, r_A, A)$
    3. $E \rightarrow A : r_B$
    4. $A \rightarrow E : r_A', S_A(r_A', r_B, B)$
    5. $E \rightarrow B : r_A', S_A(r_A', r_B, B)$
  - Due to symmetric messages (2), (3)

Questions?

- Yongdae Kim
  - email: yongdaek@kaist.ac.kr
  - Home: http://syssec.kaist.ac.kr/~yongdaek
  - Facebook: https://www.facebook.com/y0ngdaek
  - Twitter: https://twitter.com/yongdaek
  - Google "Yongdae Kim"