

Security Attributes of Key Agreement Protocols

Anish Mathuria

DA-IICT, Gandhinagar
anish_mathuria@daiict.ac.in

Key Establishment Protocols

- Cryptographic protocols for deriving a common key between two or more parties that is "secret" from adversaries
- Use the derived keys to encrypt/authenticate the communication within the session
- Two classes
 - » **Key Agreement:** the shared key isn't determined until after the protocol is over
 - » **Key Transport:** the shared key is chosen before the protocol is performed

Motivation

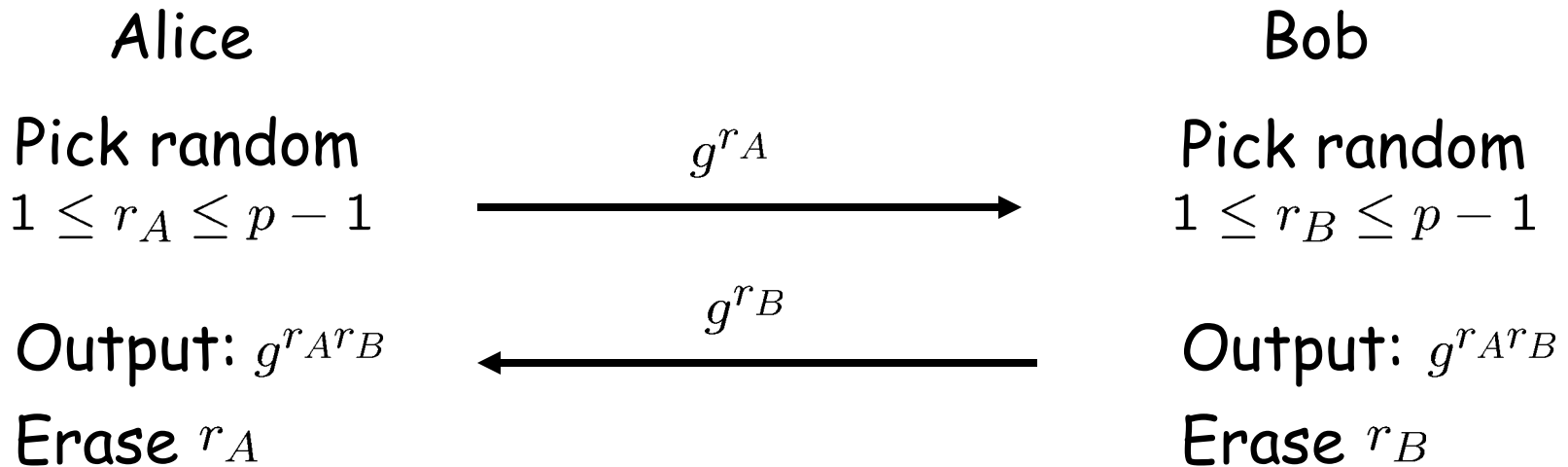
- Key agreement protocols have traditionally been among the hardest protocols to design
- Many different security attributes to consider
- This talk illustrates some challenges of design for key agreement protocols

Basic Security Attribute

- A two party protocol is "secure" if whenever one party accepts a key with another party only that party can possibly compute the key
 - » Above security property is commonly called "implicit key authentication" [Menezes, Oorschot, Vanstone]
 - » Parties are assumed to be honest
- A "secure" protocol must achieve at least implicit key authentication

Basic (unauthenticated) Diffie-Hellman

- Public: prime p (1024 bit), generator g of group Z_p^*



- Diffie-Hellman problem: Input: (g^a, g^b) , Output: g^{ab}
- Easy if we can compute x from g^x
- No better way known \rightarrow **implicit key authentication (only if attacker is passive)**

Adversary

- Monitors/controls/modifies traffic
- May corrupt parties: learns long-term secrets
- May learn session-specific information: state/keys

Varieties of AKA

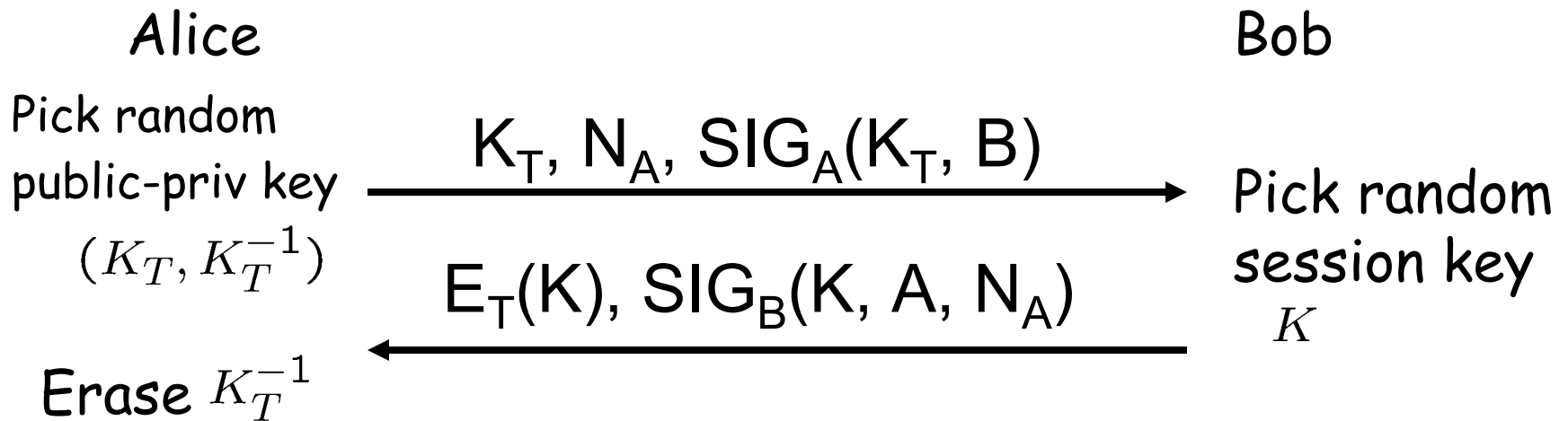
- two party, n-party
- different ways to authenticate exchange
 - » Long-term public-private keys for signature or encryption, plus “public-key infrastructure”
 - » Long-term public keys in the calculation of the shared key
 - » Implicitly-certified (identity-based) public keys
 - public keys are constructed from public data instead of being transported in certificates
 - » Long-term pre-shared keys
 - » Passwords

Forward Secrecy

- Even if long-term secret keys are compromised, it should not compromise any previously established session keys
 - » Deemed important for “off-the-record” conversations [Borisov, Goldberg, Brewer, 04]
- Most protocols possessing forward secrecy use Diffie-Hellman key agreement
- Key agreement is not necessary for achieving forward secrecy

Non-key agreement FS example

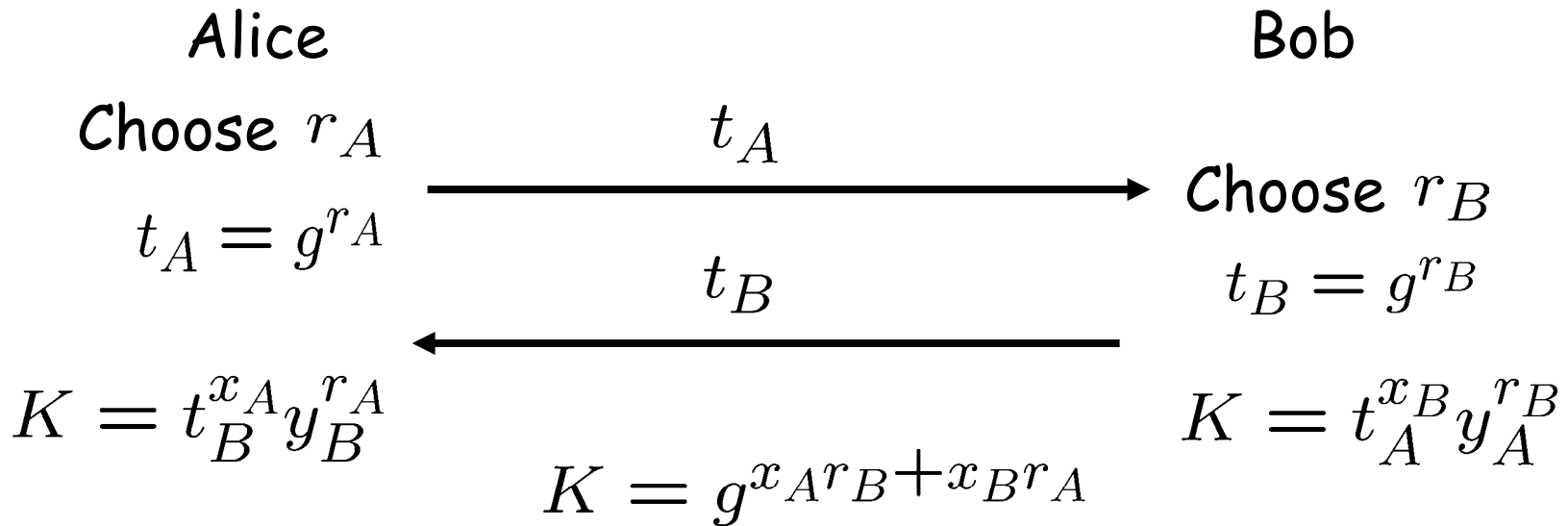
- Assume: Alice and Bob's public signature keys are certified by some mutually trusted certificate authority



- SIG is a non-message recovering signature
- Authenticity of K_T established by A 's long-term signature key
- Secrecy of K protected by A 's ephemeral public encryption key

MTI A(0) Protocol

- A and B pick long-term secrets x_A and x_B
- The corresponding "public keys", $y_A = g^{x_A}$ and $y_B = g^{x_B}$ are certified by some certificate authority



- Either x_A or x_B is required to calculate the shared secret → protection against mitm

Attack on Forward Secrecy

- E sees g^{r_A}, g^{r_B}
- Assume E cannot work out r_A or r_B
- Assume E knows x_A and x_B
- Can E compute $K = t_B^{x_A} y_B^{r_A} = t_A^{x_B} y_A^{r_B}$

$$\begin{aligned} K &= g^{x_A r_B + x_B r_A} \\ &= (g^{r_B})^{x_A} (g^{r_A})^{x_B} \\ &= t_B^{x_A} t_A^{x_B} \end{aligned}$$

Known Session-Key Security

- A protocol run should result in a unique secret session key
- If past session keys are compromised, it should not allow adversary to compromise future session keys

Modified ElGamal Signature [Nyberg-Rueppel]

- Sender Alice, Receiver Bob
- B's (long-term) private key $1 \leq x_A \leq p - 1$
- B's (long-term) public key $(p, g, y_A = g^{x_A})$
- To sign a message $m \in Z_p^*$, A does
 - » Choose randomly a secret $1 \leq k \leq p - 1$
 - » Compute: $r = mg^{-k}$
$$s = (k - x_A r) \pmod{p - 1}$$
- The signed message is (r, s)
- B can recover message as: $g^s y_A^r r = m$
- Verification: check redundancy of m

Nyberg-Rueppel Protocol

Alice

Bob

Pick random

$$1 \leq r_A, k \leq p - 1$$

$$r = g^{r_A} g^{-k}$$

$$s = k - x_A r \pmod{p - 1}$$

(r, s)



Verify: $g^s y_A^r r$

Output:

$$K = y_B^{r_A}$$

Output:

$$K = (g^s y_A^r r)^{x_B}$$

- Note: (r, s) is Alice's signature on g^{r_A}

Known Session-Key Attack

- If E knows old session key K , she can impersonate Alice

Eve (as Alice)

Bob

Choose u

$(r, s + u)$

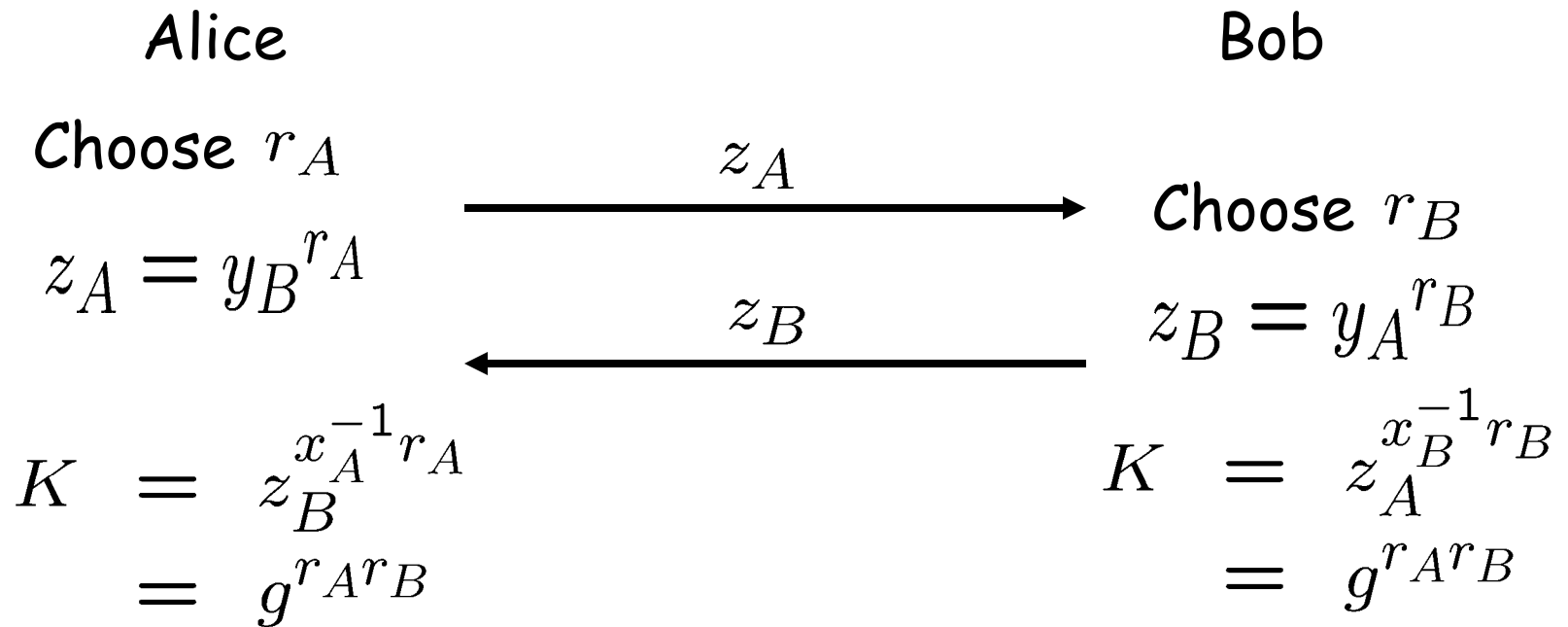


Output: $K \cdot y_B^u$

Key compromise impersonation resilience

- If entity A 's long-term private key is compromised, an adversary E is able to impersonate A
- But this should not enable E to impersonate other legitimate entities to A (called **key compromise impersonation attack**)
- Attack not applicable to protocols which use signatures to authenticate the exchange

MTI C(0)

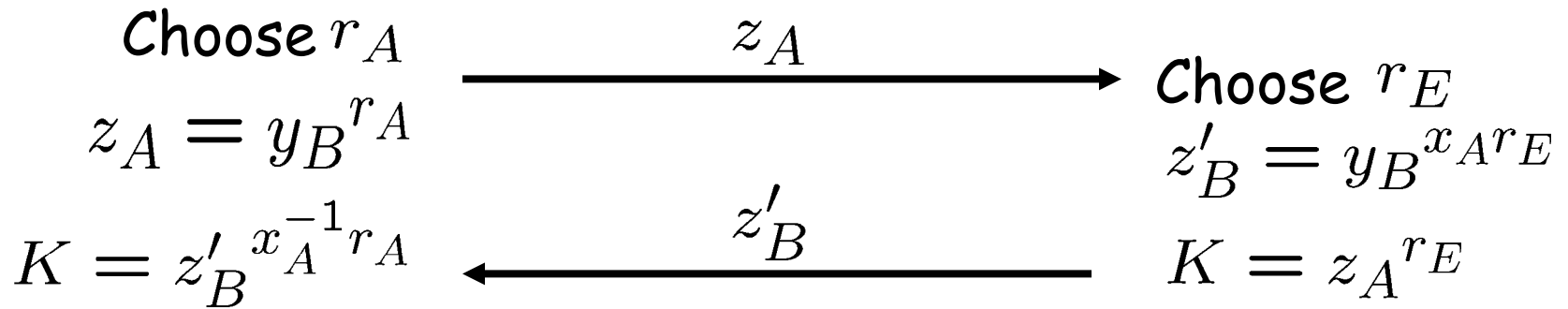


- Protection against mitm
- Forward secrecy

KCI Attack

Alice

Eve (as Bob)



- Assume E has long-term private key x_A of A
- A believes $K = y_B^{r_A r_E}$ is only known to A and B
- But E can compute K as $z_A^{r_E}$

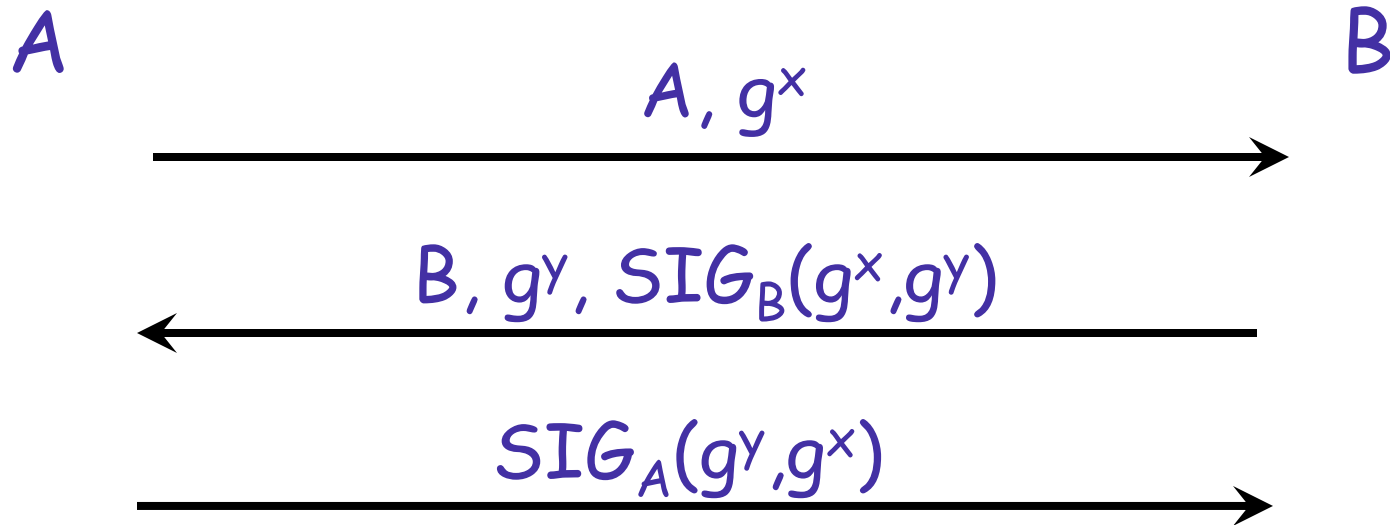
Unknown key-share resilience

- No honest entity X (A or B) can be coerced into sharing a key with another entity without X 's knowledge
- **Unknown key share attack against B**
 - » When A (correctly) believes the key is shared with B , but B believes the same key is shared with some entity $C \neq A$
- **Unknown key share attack against A**
 - » When B (correctly) believes the key is shared with A , but A believes the same key is shared with some entity $C \neq B$

Bank Example

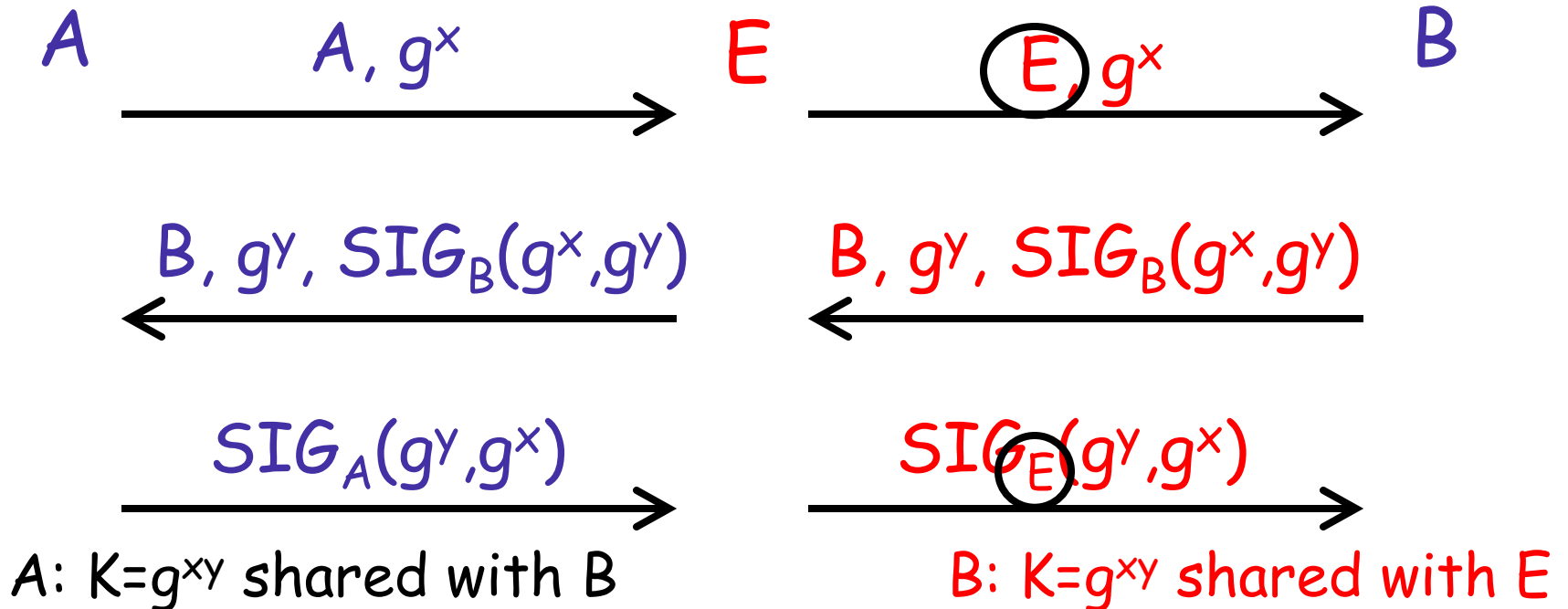
- Suppose B is a bank and A and C are two account holders [adopted from Diffie et al]
- A sends to B
 - “Please transfer a million rupees into my account”
- If B has an unknown “key share” with A, then C can get credit for the deposit made by B (without knowing K); both A and B may be hurt
- N.B.: Attack not a major concern if application messages are made sufficiently explicit
 - » e.g., “Transfer into account# 3456”

Basic Signature-based Protocol



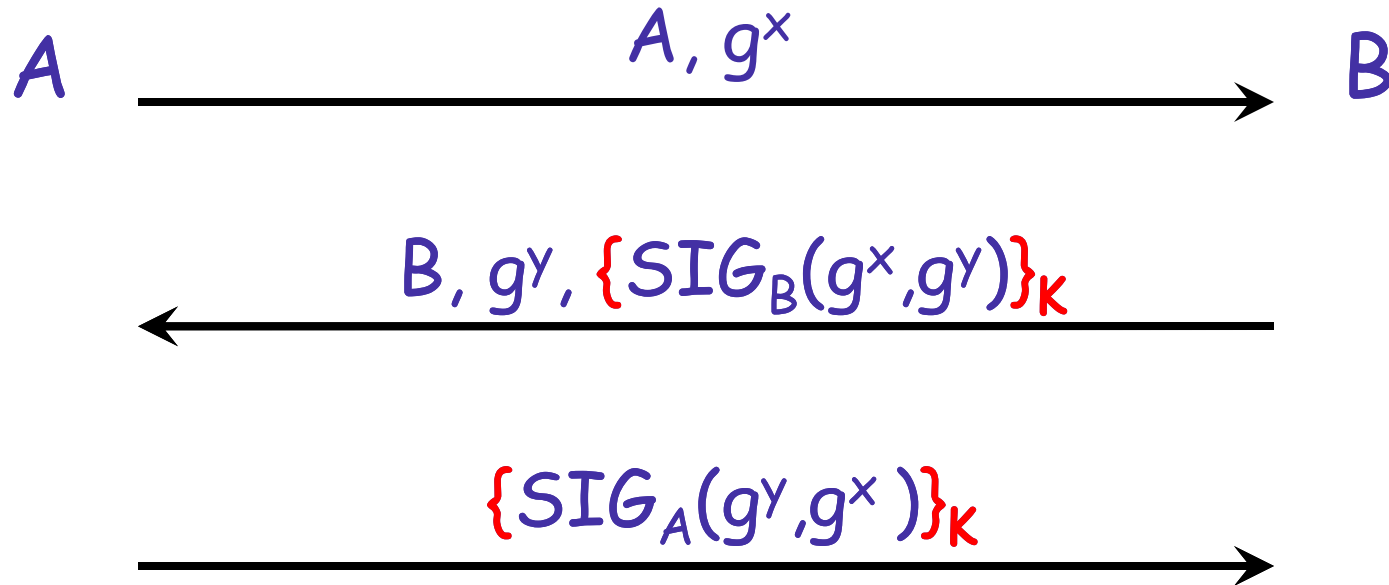
- Each party signs its own DH value
→ Protection against mitm attack
- Each party additionally signs its peer's DH value
→ Protection against leakage of DH exponents

Unknown key share attack



Eve replaces A's certificate with her own (legitimate) certificate

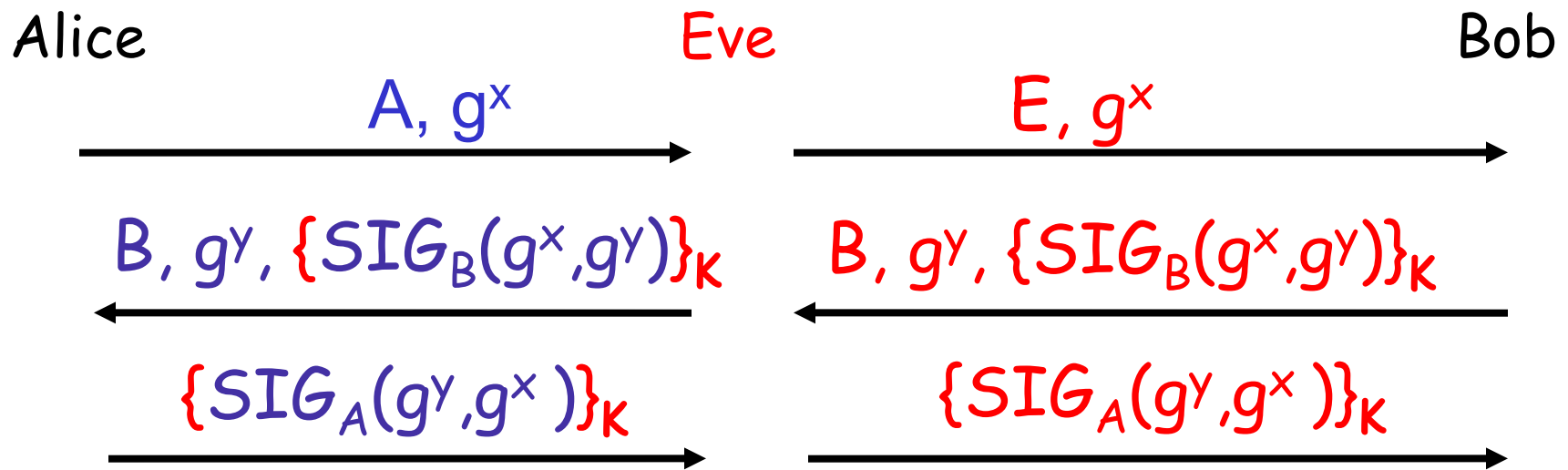
STS with Encrypted-Signature



- Each party proves knowledge of $K=g^{xy}$
- Without knowing K , Eve cannot send the encrypted signature required for attack

Unknown key-share Attack

- if E can register A's public-key under her name she can mount the same attack

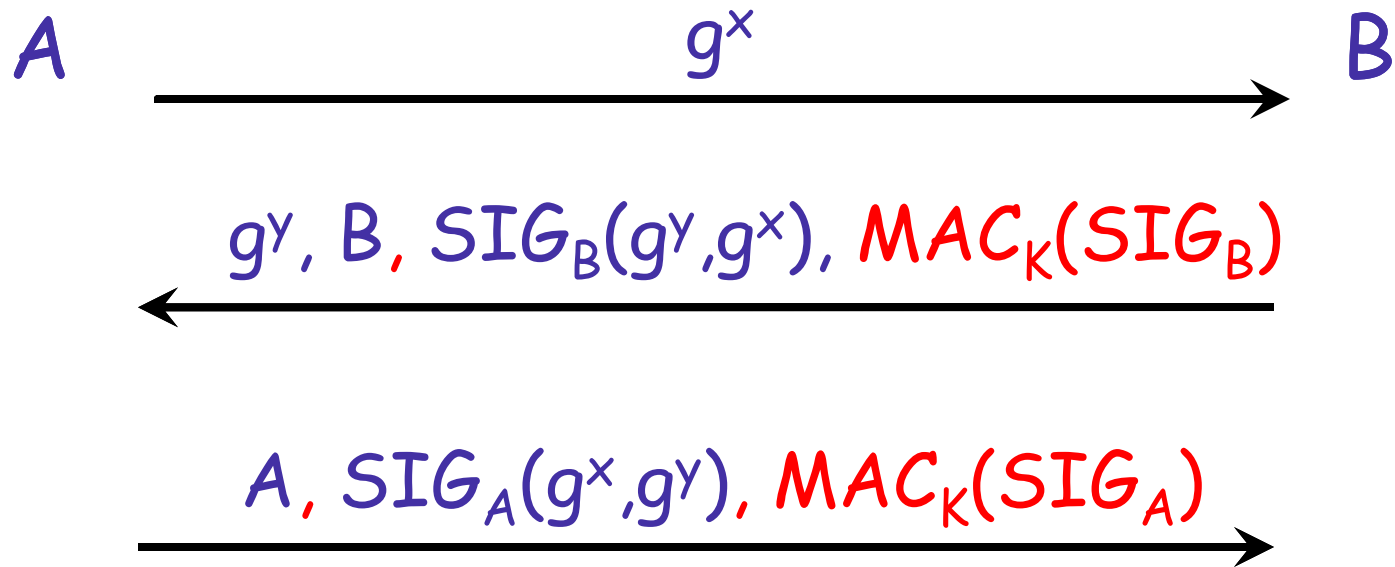


- B believes the last message was signed by E

Some countermeasures

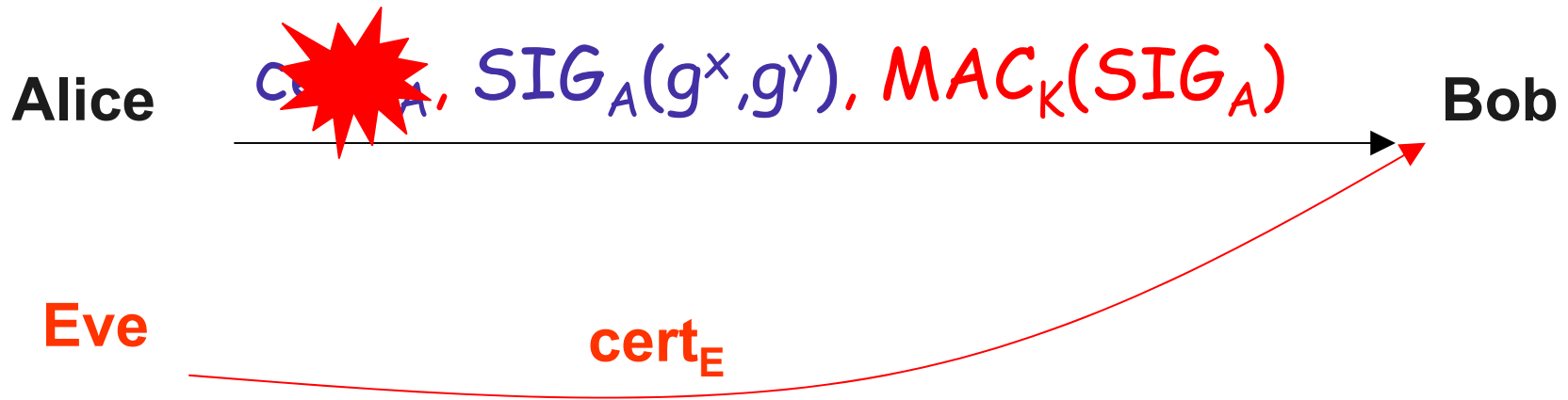
- Certificate issuer checks for duplicate public keys
- Issuer requires proof of possession of the corresponding private key
- Better defense is to include peer identities in the calculation of the derived key

STS with MACed-Signature



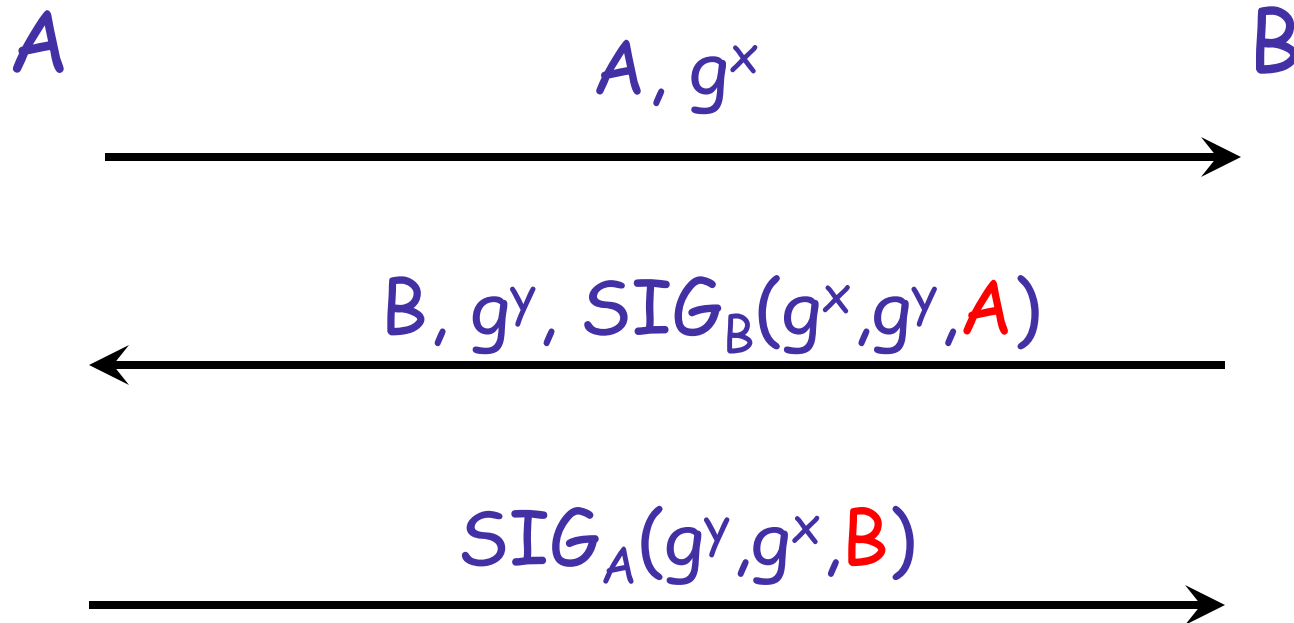
- If E can find a new public-priv key so that $SIG_A(g^x, g^y) = SIG_E(g^x, g^y)$, she can mount an unknown key share attack

Unknown key share attack



- Not prevented by priv-key PoP because Eve *knows* the private key
- Note Eve must obtain the certificate *during the protocol run*

ISO Protocol

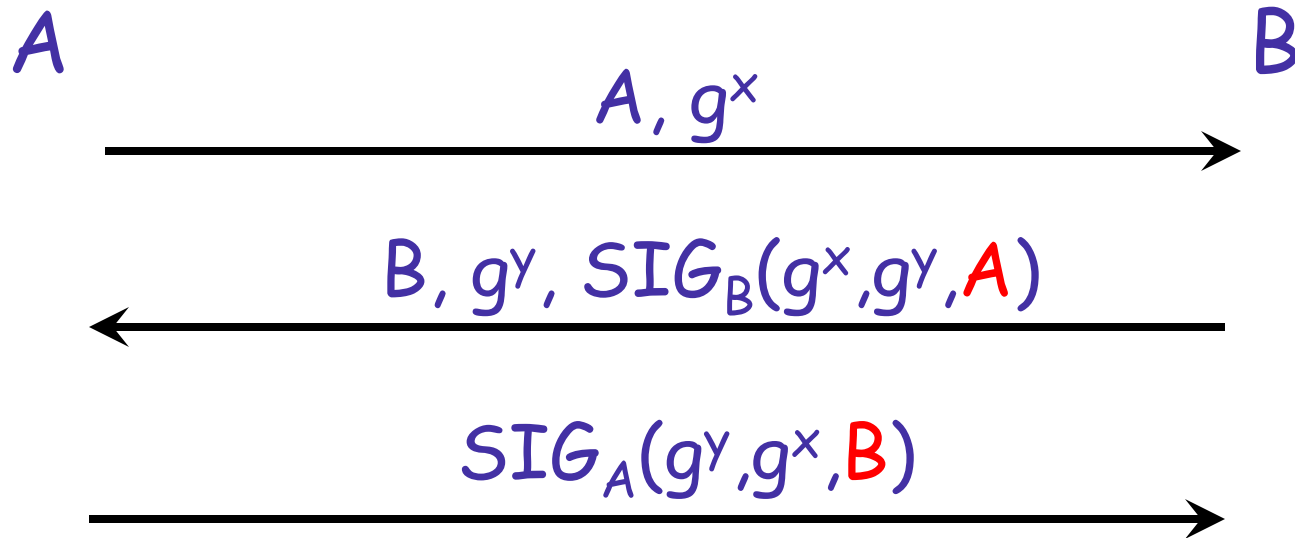


- Signatures include identity of the peer
→ Protection against uks attack
- Attack not prevented if signature is over the identity of sender (instead of peer)

Identity Protection

- Parties can use shared key to encrypt their id's
 - » passive attacker will not know the id's
 - » active attacker may still learn one or both id's
- Best possible: both id's protected against passive attacks but only one against active attacks
- Whose id should get active defense?
 - » Initiator: roaming user (e.g. hide location)
 - » Responder: avoid probing attacks (who are you?)

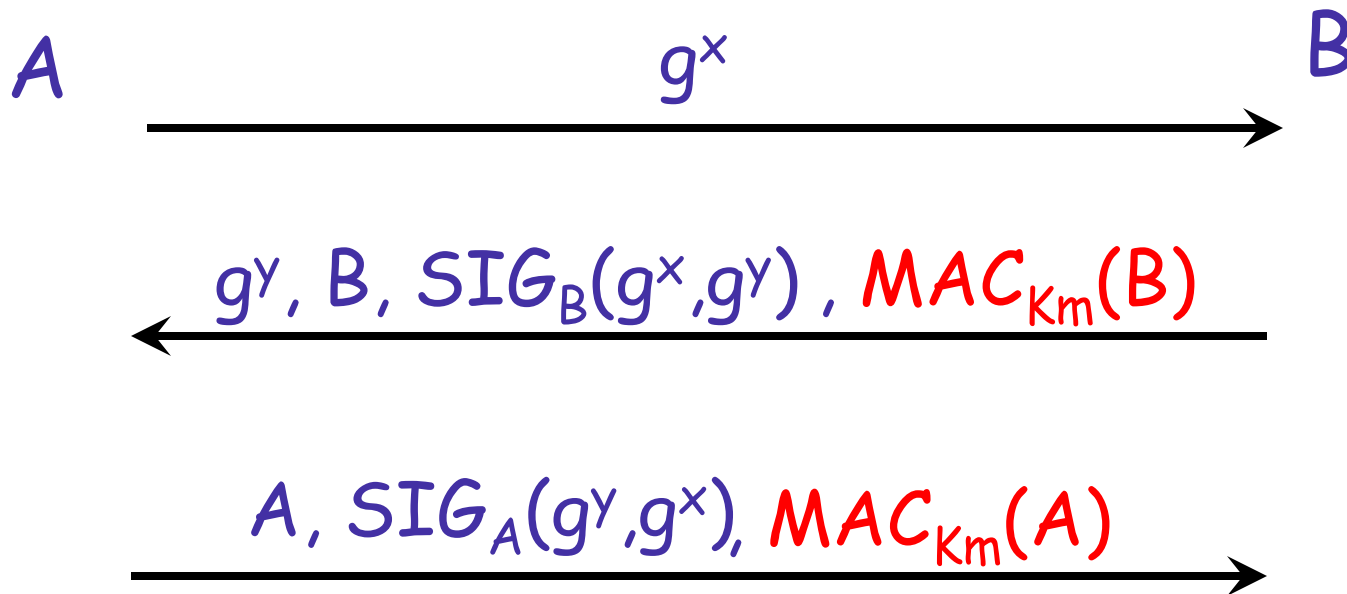
ISO Protocol



- B needs to know A's identity before he can authenticate to A; same for A →
Protection against active attackers is not possible

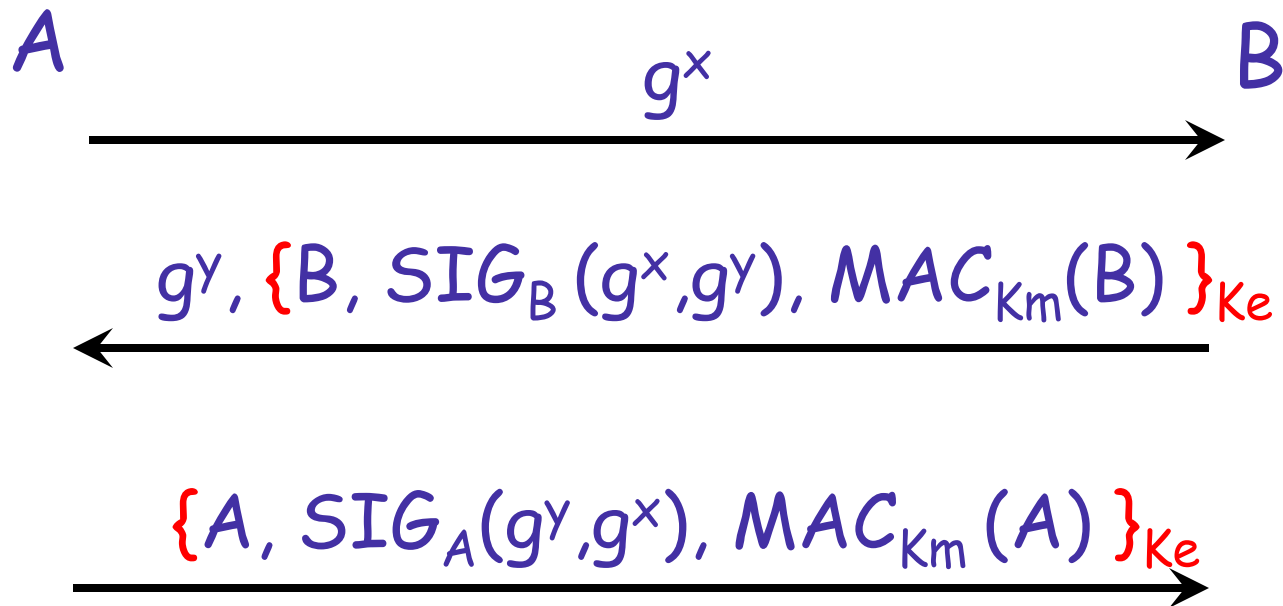
SIGMA Protocol

- Does not require knowing the peer's id for own authentication → Suited for id protection



K_m is a key derived from g^{xy}

SIGMA-I: protect initiator's id

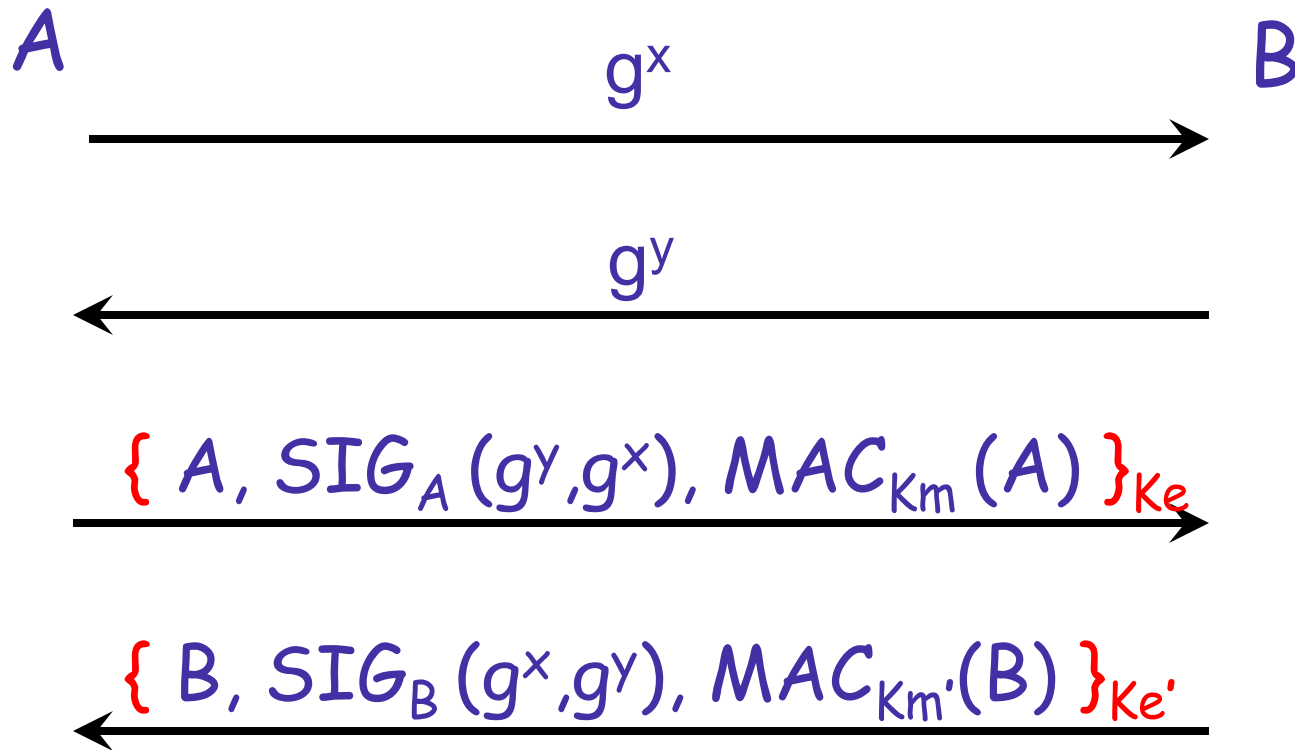


* K_e and K_m derived from g^{xy}

Responder (B) identifies first

→ Initiator's (A) id protected

SIGMA-R: protect responder's id



Initiator (A) identifies first

→ Responder's (B) id protected

Conclusions

- Security properties remain interesting topic for further study
 - » **Dos Resistance** [Aiello et al 04]
 - » **"Full" forward secrecy** [Krawczyk 05]
 - » **Bilateral UKS resilience** [Chen and Tang 07]
- Some interesting stuff
 - » **Manual authentication:** What if pre-shared keys; public keys for signatures or encryption are not available? [Vaudenay 05; Cagalj, Capkun, Hubaux 06]
 - » **Impossibility result:** No group key agreement protocol of certain type is secure, for >3 participants [Pereira and Quisquater 04]

Conclusions (contd.)

- We did not talk about identity based protocols, protocols for computationally limited devices, security proofs, ...
- For more information
 - » Boyd and Mathuria, Protocols for Authentication and Key Establishment, 03
 - » Menezes, van Oorschot and Vanstone, Handbook of Applied Cryptography, 97
 - » Krawczyk, SIGMA: The SIGn-and-MAC Approach to Authenticated Diffie-Hellman ..., Crypto 03