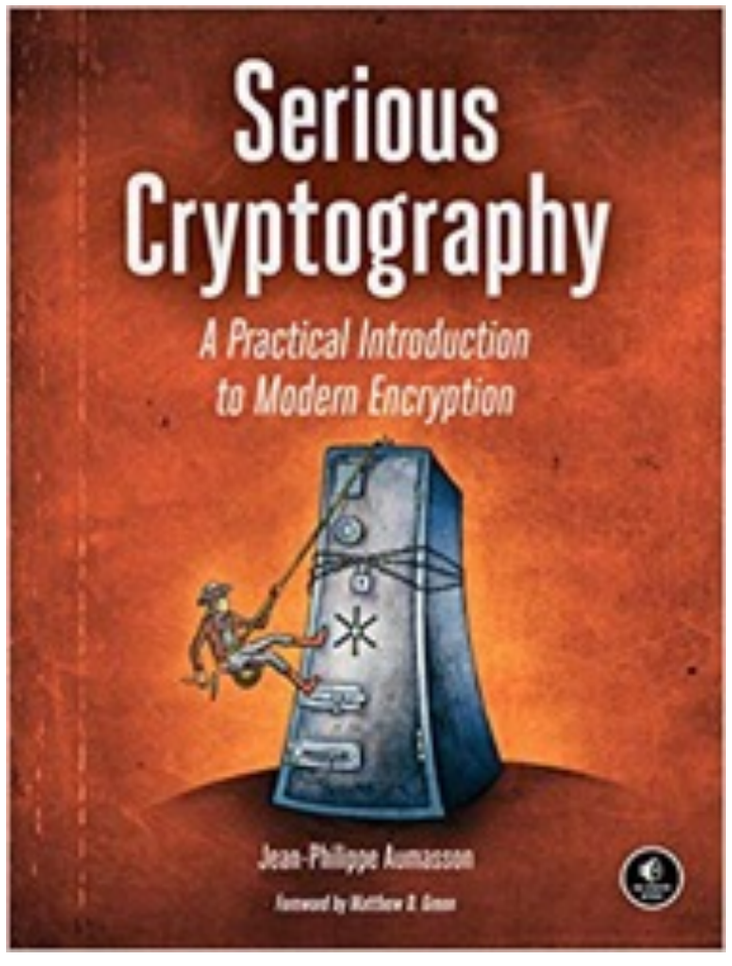
CNIT 141

Cryptography for Computer Networks



11. Diffie-Hellman

Updated 11-23-22

Topics

- The Diffie-Hellman Function
- The Diffie-Hellman Problems
- Key Agreement Protocols
- Diffie-Hellman Protocols
- How Things Can Go Wrong

1976

- Whitfield Diffie and Martin Hellman
- Published "New Directions in Cryptography"
- Revolutionized cryptography
- Specified a ***public-key distribution scheme***
  - The ***Diffie-Hellman (DH) protocol***
- The basis for public-key encryption and signatures

Key Agreement

- After exchanging a ***shared secret***
- Parties turn the secret into a ***symmetric key***
- Thus establishing a ***secure channel***

The Diffie-Hellman Function

The Group Z_p^*

- The integers 1, 2, 3, ... ***p-1***
- Where ***p*** is prime
- In DH, the two parties choose random elements ***a*** and ***b*** to be their secrets
  - From the group
- Both parties also use a number ***g***
  - Which is not a secret

Alice and Bob

- They can both calculate g^{ab} by combining public and secret information

Keep a secret		Keep b secret
Transmit $A = g^a$	⟷	Transmit $B = g^b$
Calculate $g^{ab} = B^a$		Calculate $g^{ab} = A^b$

Diffie-Hellman

- Alice calculates $A = g^a \bmod p$
  - and sends it to Bob
- Bob calculates $B = g^b \bmod p$
  - and sends it to Alice
- Alice calculates $B^a \bmod p = g^{ba} \bmod p$
- Bob calculates $A^b \bmod p = g^{ab} \bmod p$
- They now have the same ***shared secret***

Key Derivation Function (KDF)

- The shared secret is not used directly as the key
- It's passed through a KDF to create a random-looking value of the proper size
  - A kind of hash function

Safe Primes

- Not all values of ***p*** and ***g*** work
  - For highest security, both ***p*** and **(*p* - 1) / 2** should be prime
- Those are called ***safe primes***
- They don't have small subgroups
  - That would limit the shared secret to a small number of possible values

Safe Primes

- With ***safe primes*** even a ***g*** of 2 works
- But ***safe primes*** are slow to generate
  - 1000x as long as generating mere random primes

Generating 2048-bit DH

```
$ time openssl dhparam 2048
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
--snip--
-----BEGIN DH PARAMETERS-----
MIIBCAKCAQEAoSIbyA9e844q7V89rcoEV8vd/l2svwhIIjG9EPwWWr7FkfYhYkU9
fRNttmilGCTfxc9EDf+4dzw+AbRBc6oOL9gxUoPnOd1/G/YDYgyplF5M3xeswqea
SD+B7628pWTaCZGKZham7vmiN8azGeaYAucckTkjVWceHVIVXe5fvU74k7+C2wKk
iiyMFm8th2zm9W/shiKNV2+SsHtD6r3ZC2/hfu7XdOI4iT6ise83YicU/cRaDmK6
zgBKn3SlCjwL4M3+m1J+Vh0UFz/nWTJ1IWAVC+aoLK8upqRgApOgHkVqzP/CgwBw
XAOE8ncQqroJ0mUSB5eLqfpAvyBWpkrwQwIBAg==
-----END DH PARAMETERS-----
openssl dhparam 2048  154.53s user 0.86s system 99% cpu 2:36.85 total
```

- 154 seconds

Generating 2048-bit RSA

```
$ time openssl genrsa 2048
Generating RSA private key, 2048 bit long modulus
.....................................................+++
.....................................................+++
e is 65537 (0x10001)
-----BEGIN RSA PRIVATE KEY-----
--snip--
-----END RSA PRIVATE KEY-----
openssl genrsa 2048  0.16s user 0.01s system 95% cpu 0.171 total
```

- 0.17 seconds

The Diffie-Hellman Problems

Discrete Logarithm Problem

- Public value: g^a
- Secret value: a
- Recovering a from g^a is the DLP
- Diffie-Hellman's security depends on the DLP's hardness

Eavesdropper

- Attacker knows only g^a and g^b

Keep ***a*** secret

Transmit $A = g^a$

Calculate $g^{ab} = B^a$

Keep ***b*** secret

Transmit $B = g^b$

Calculate $g^{ab} = A^b$

The Computational Diffie-Hellman Problem (CDH)

- Consider an eavesdropper
- Compute the shared secret g^{ab}
  - Given only the public values g^a and g^b
  - And not the secrets a or b
- This might be easier than the DLP
- We don't know for sure

Number Sieve

- DH protocol with 2048 bit prime ***p*** provides 90 bits of security
- Same as RSA with a 2048-bit ***n***
- Fastest known attack on Computational Diffie-Hellman is the ***number field sieve***
- Similar to the fastest known attack on RSA: the "general number field sieve"

Decisional Diffie-Hellman Problem (DDH)

- Attacker knows only g^a and g^b but wants shared secret g^{ab}
- Attacker can't deduce any portion of the shared secret
- Because the shared secret appears random

Keep ***a*** secret
Transmit g^a

Attacker
wants shared
secret ga^b

Keep ***b*** secret
Transmit g^b

Decisional Diffie-Hellman Problem (DDH)

- If DDH is hard, then CDH is also hard
- DDH is less hard than CDH
- DDH hardness is a prime assumption in cryptography
  - Well-studied
- Both DDH and CDH are hard if the parameters are well-chosen

Key Agreement Protocols

A Non-DH Key Agreement Protocol

- Authenticated Key Agreement (AKA)
- Used by 3G and 4G
- To establish secure communication between a SIM card and a telecom operator
- Uses only symmetric-key operations
- Relies on a pre-shared secret ***K***

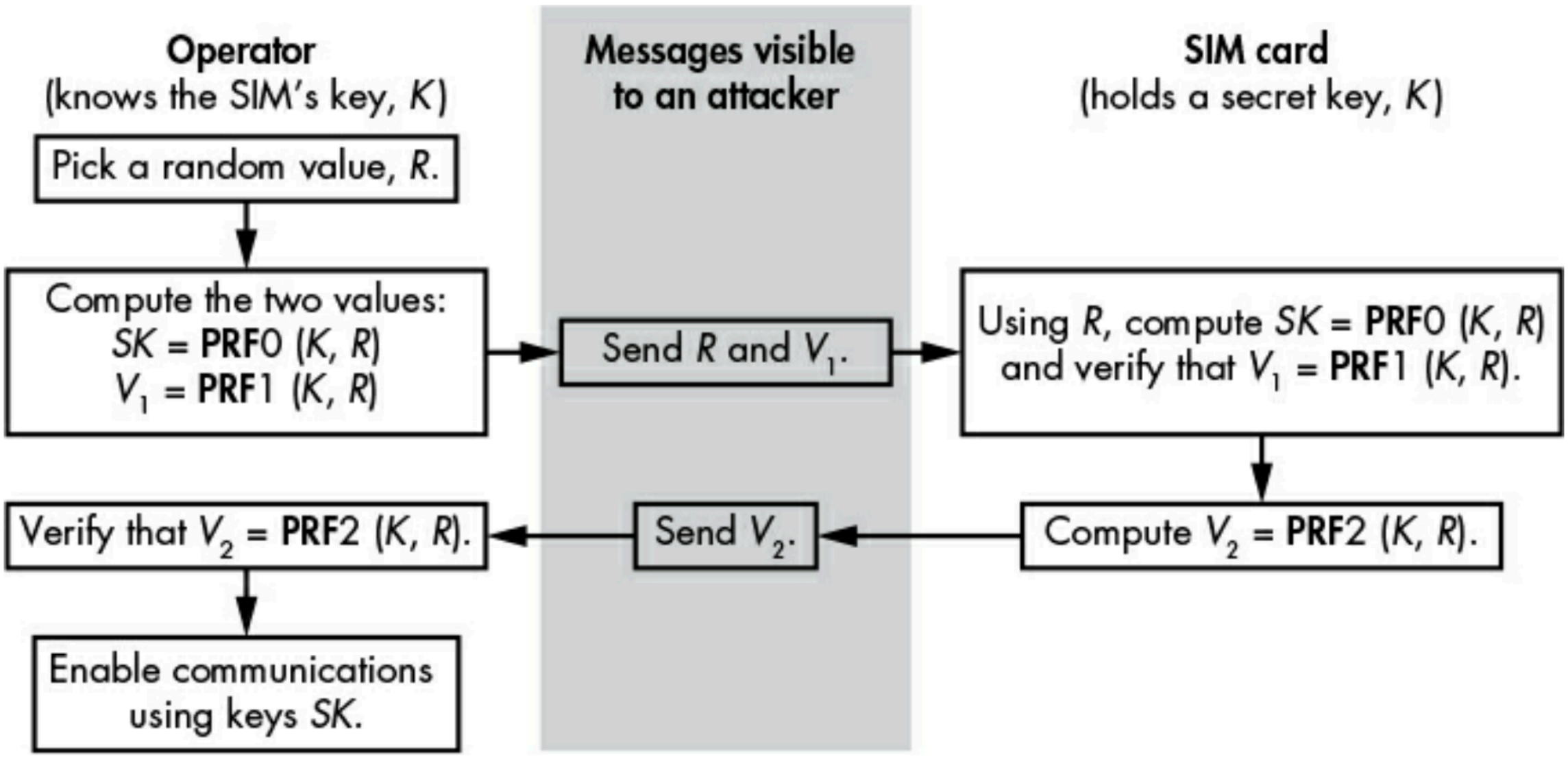


Figure 11-1: The authenticated key agreement protocol in 3G and 4G telecommunication

Replay Attack

- Attacker captures pair (***R, V***$_1$**)**
  - Sends it to SIM card to open a new session impersonating the telco
- To prevent this, protocol checks to make sure ***R*** isn't reused

Compromised K

- Attacker who gets ***K***
  - Can perform MiTM attack and listen to all cleartext communications
  - Can impersonate either party
  - Can record communications and later decrypt them using the captured ***R*** values

Attack Models for Key Agreement Protocols

- **Eavesdropper**
  - Attacker is a MiTM
  - Can record, modify, drop or inject messages
  - To stop: protocol must not leak any information about the shared secret
- **Data leak**
  - Attacker gets the session key and all temporary secrets
  - But not long-term secret ***K***

Attack Models for Key Agreement Protocols

- **Breach**
  - Attacker learns long-term key ***K***
  - Impossible to protect current session from this attack
  - But a protocol can protect other sessions

Security Goals

- **Authentication**
  - Mutual authentication: each party can authenticate to the other party
  - Authenticated Key Agreement happens when a protocol authenticates both parties

Security Goals

- **Key control**
  - Neither party can control the final shared secret
- The 3G/4G protocol lacks this property
  - Because the operator chooses ***R***
  - Which entirely determines the final shared key

Security Goals

- **Forward secrecy**
  - Even if all long-term secrets are exposed
  - Shared secrets from previous sessions are not available
  - 3G/4G protocol doesn't provide this

Performance

- Number of messages exchanged
- Message length
- Computations required
- Possibility of pre-computation
- The main cause of latency is usually ***round-trip time***
  - Computation required also counts

Performance of 3G/4G

- Exchanges two messages of a few hundred bits each
- Pre-computation is possible
  - Operator can pick many values of ***R*** in advance

Diffie-Hellman Protocols

Anonymous Diffie-Hellman

- Not authenticated
- Vulnerable to MiTM attack (next slide)

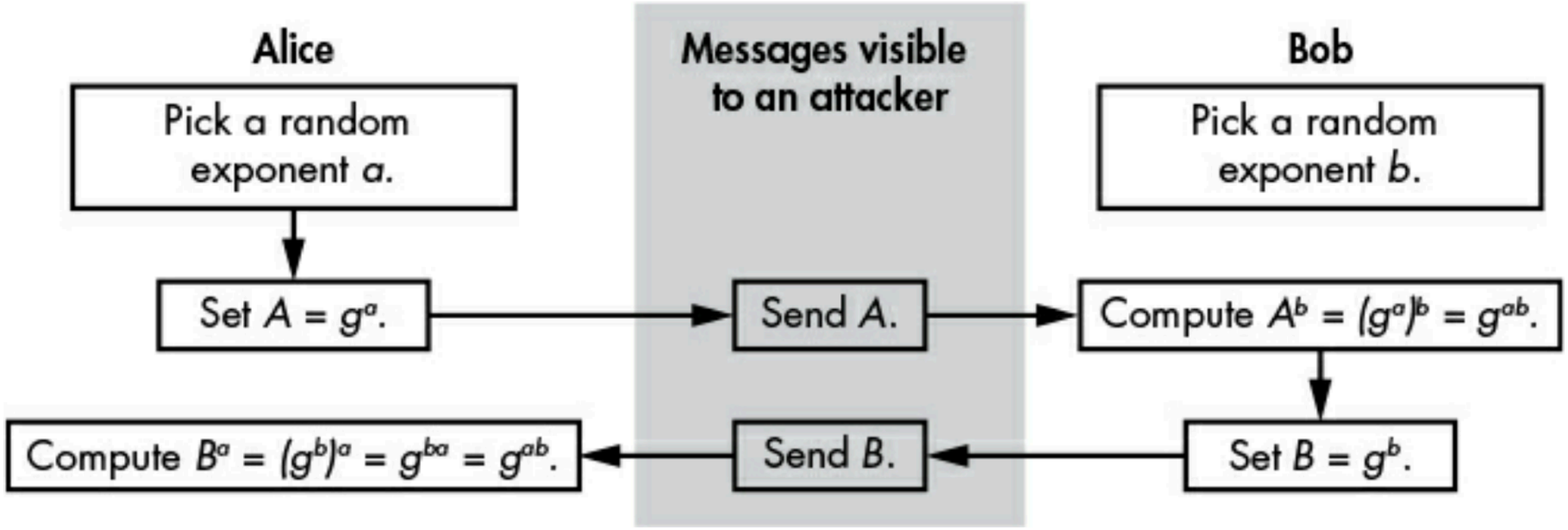
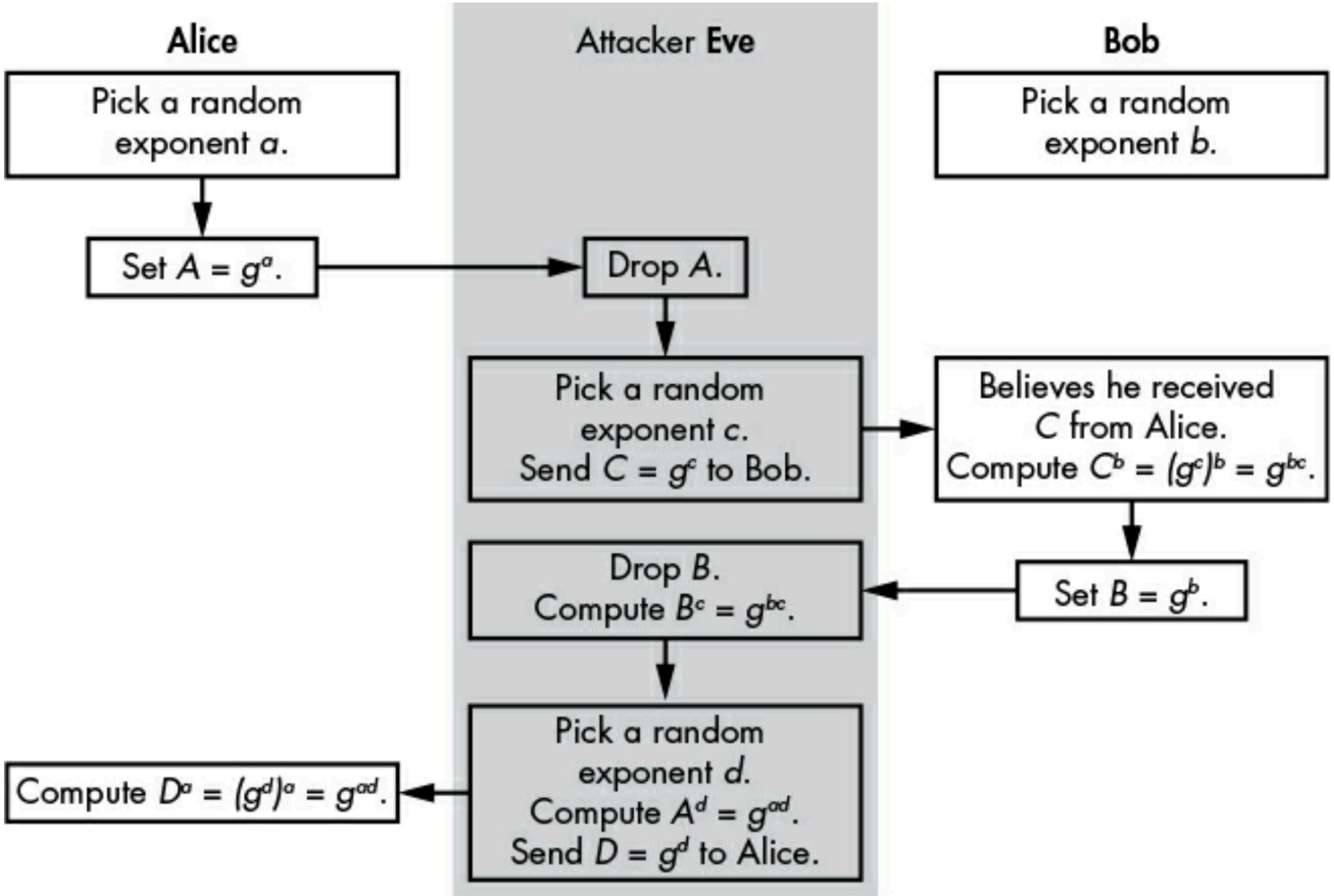


Figure 11-2: The anonymous Diffie–Hellman protocol


Alice
Attacker Eve
Bob
Pick a random exponent a.
Pick a random exponent b.
Set A = g^a.
Drop A.
Pick a random exponent c.
Send C = g^c to Bob.
Believes he received C from Alice.
Compute C^b = (g^c)^b = g^bc.
Set B = g^b.
Drop B.
Compute B^c = g^bc.
Pick a random exponent d.
Compute A^d = g^ad.
Send D = g^d to Alice.
Compute D^a = (g^d)^a = g^ad.

Authenticated Diffie-Hellman

- Uses public-key signatures to sign messages
  - With a system such as RSA-PSS (Probabilistic Signature Scheme)

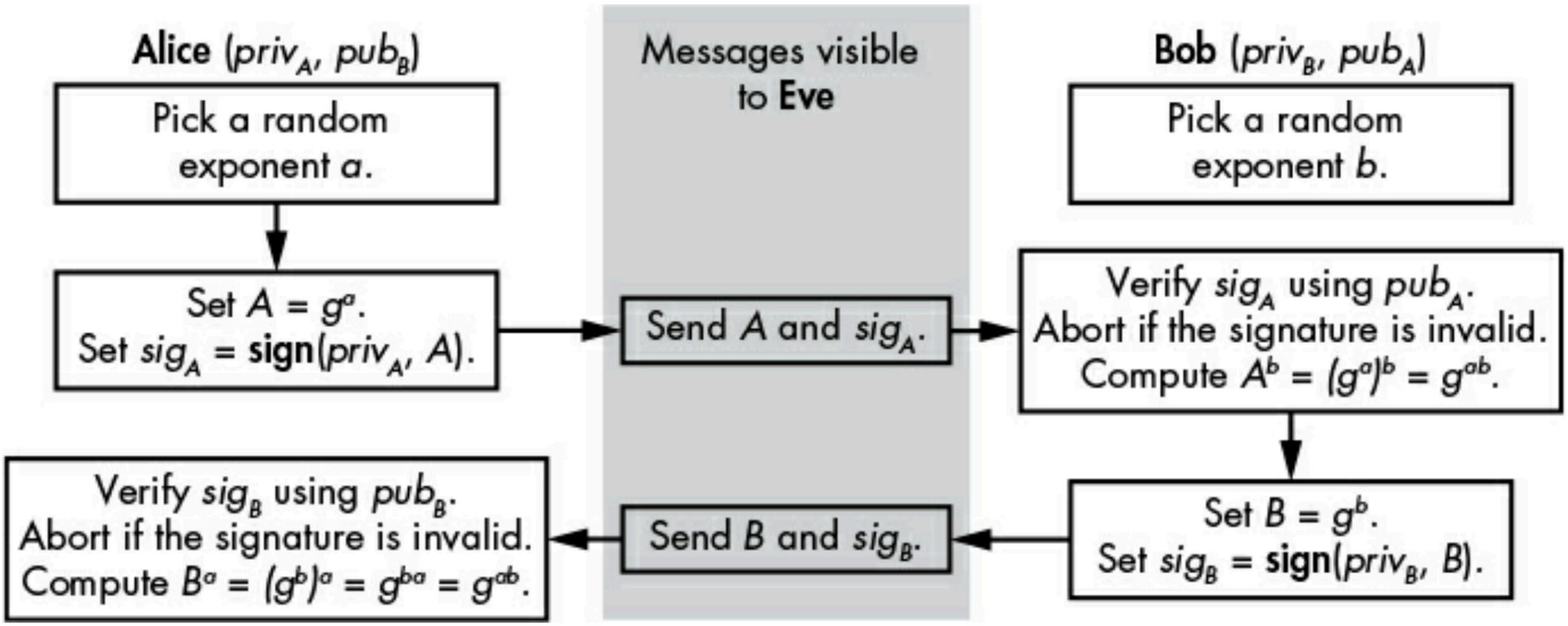


Figure 11-4: The authenticated Diffie–Hellman protocol

Security Against Eavesdroppers

- Authenticated DH stops eavesdroppers
  - Attacker can't learn the shared secret g^{ab}
- Neither party can control the shared secret

Replay

- Eve can record and replay previous values of ***A*** and $\boldsymbol{sig_A}$
  - To pretend to be Alice
- ***Key confirmation*** prevents this
  - Alice and Bob send a message to prove that they both own the shared secret

Security Against Data Leaks

- If Eve has ***a***, she can impersonate Alice
- To prevent this, integrate long-term keys into the shared secret computation

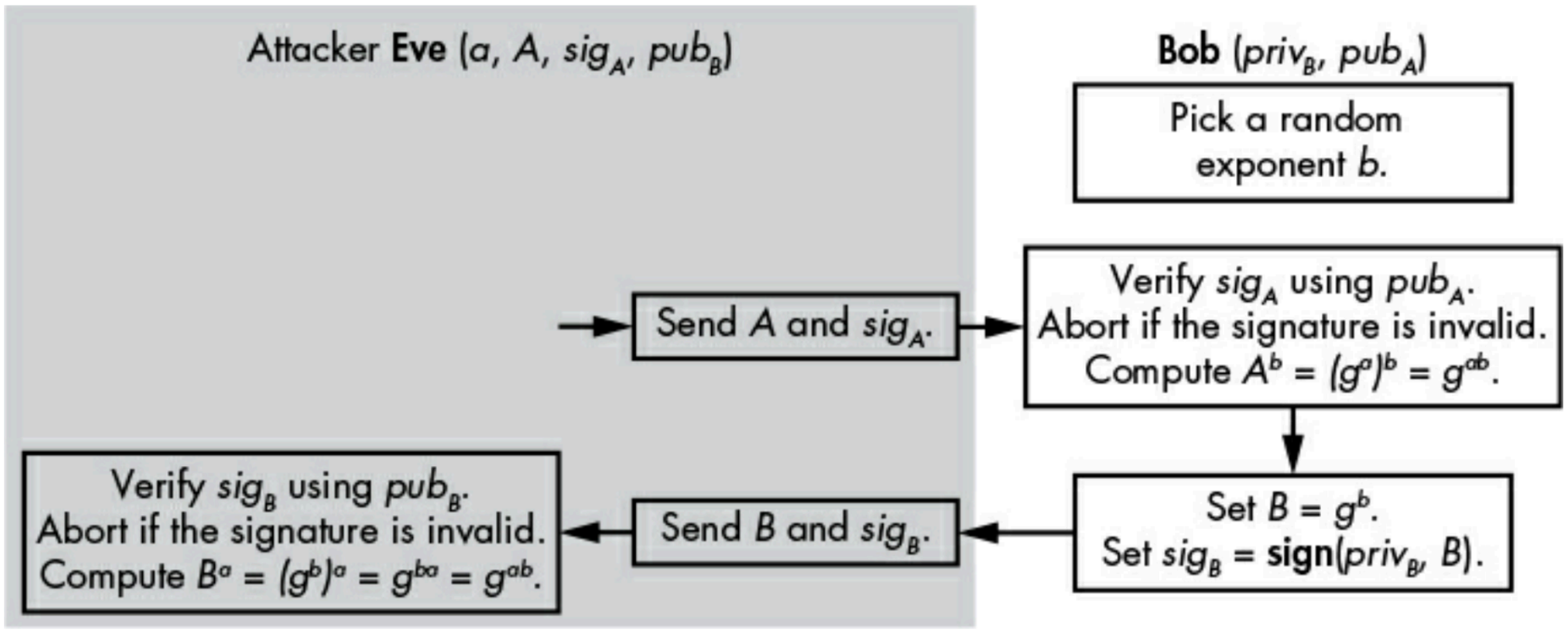


Figure 11-5: An impersonation attack on the authenticated Diffie–Hellman protocol

Memezes-Qu-Vanstone MQV

- Improved version of DH, designed in 1998
- NSA included it in Suite B
  - Designed to protect most critical assets
- More secure than authenticated DH
- Better performance

MQV

- ***x*** and ***y*** are long-term private keys
- ***X*** and ***Y*** are long-term public keys

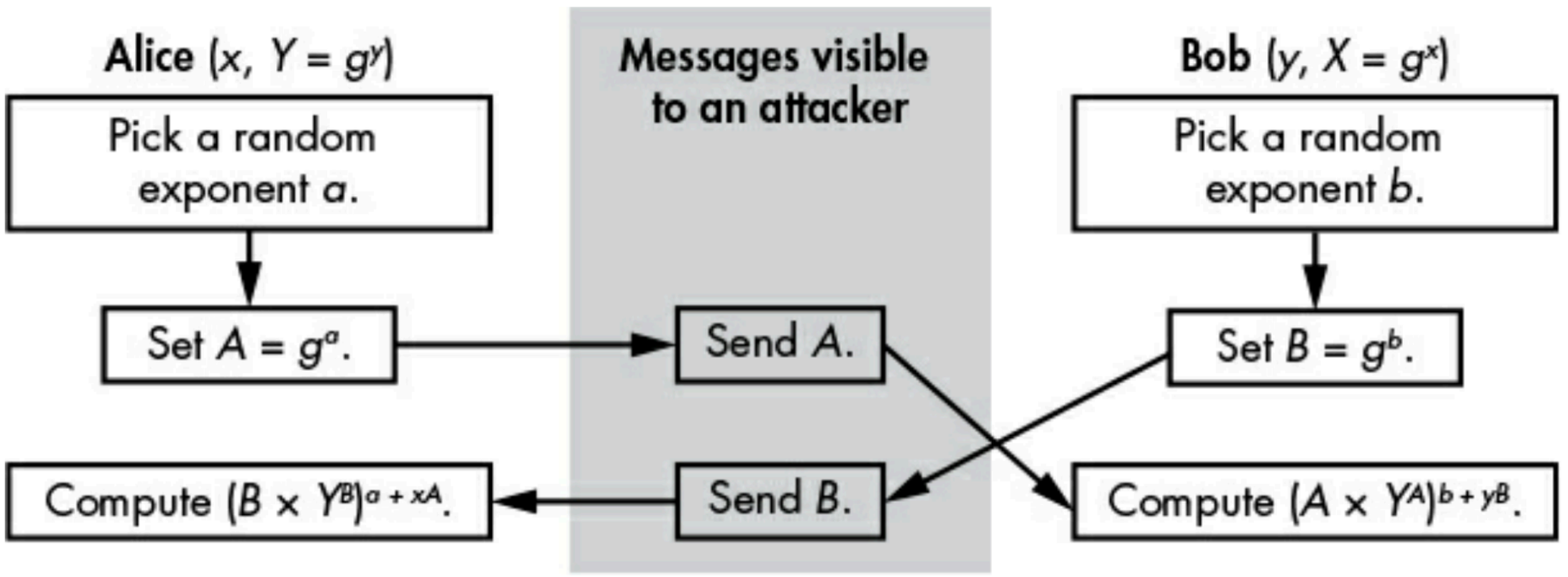


Figure 11-6: The MQV protocol

Data Leak

- Attacker who gets the ephemeral secrets ***a*** and ***b***
  - Can't find the shared secret
  - That would require knowing the long-term private keys

Breach

- Attacker gets Alice's long-term private key ***x***
- Previous sessions are still safe
  - Because they used Alice's ephemeral private keys
- There is an attack that could compromise a targeted old session
  - It can be mitigated by a key-confirmation step

MQV Rarely Used

- Was encumbered by patents
- Complex and difficult to implement
- Authenticated DH is simpler and regarded as good enough

How Things Can Go Wrong

Not Hashing the Shared Secret

- The shared secret g^{ab} is not a session key
- A symmetric key should look random
  - Every bit should be 50% likely to be 0
- But g^{ab} is in the range 1, 2, ... p
  - High-order bit more likely to be 0
- Use a KDF to convert the secret to a key

Legacy DH in TLS

- Old cipher suites uses Anonymous DH
  - TLS_DH_anon_WITH_AES_128_CBC_SHA
  - TLS_DH_ANON_AES_128_CBC_SHA1
  - TLS_DH_anon_WITH_AES_128_CBC_SHA
  - ADH-AES128-SHA
    - Link Ch 11i

Unsafe Group Parameters

- OpenSSL allowed unsafe primes ***p***
- Attacker can craft DH parameters that reveal information about the private key
- Fixed in 2016


Kahoot!