Authentication & Key Exchange

Pierre-Alain Fouque

Université de Rennes 1

November, 2020

Agenda

1. Generality
2. Authentication
3. Key Exchange
   - Diffie-Hellman
   - Signed Diffie-Hellman
   - Signed Diffie-Hellman
   - TLS Protocol

Problem to solve

How to authenticate ?

How sharing and exchanging a secret common value ? (and why ?)

Key Exchange

Encrypt and authenticate the communications

- essential for digital payment on the internet
- ensure confidentiality and integrity
- good **performances**

Use of symmetric key cryptography (shared key)

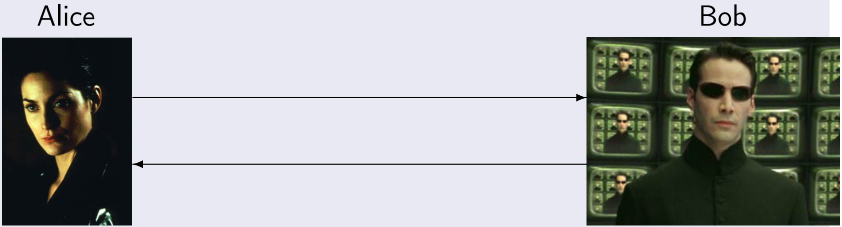
How can we compute a session key ?

Key Exchange

Several Methods to establish a (secret) key

- Key distribution : a server chooses a key and transfers it to everyone
- Key exchange : everyone participates to the negotiated key

A *client* (Alice) authenticates herself to a server (Bob)



Alice

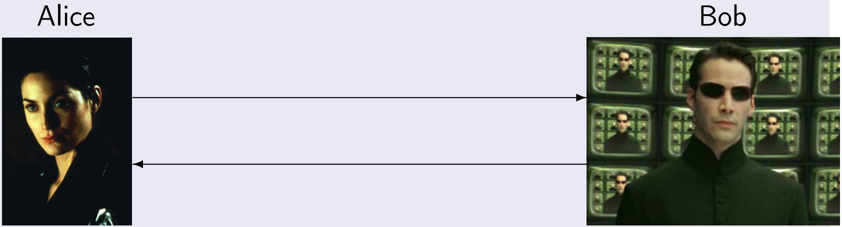
Bob

Key Exchange

Several Methods to establish a (secret) key

- Key distribution : a server chooses a key and transfers it to everyone
- Key exchange : everyone participates to the negotiated key

A *client* (Alice) authenticates herself to a server (Bob)

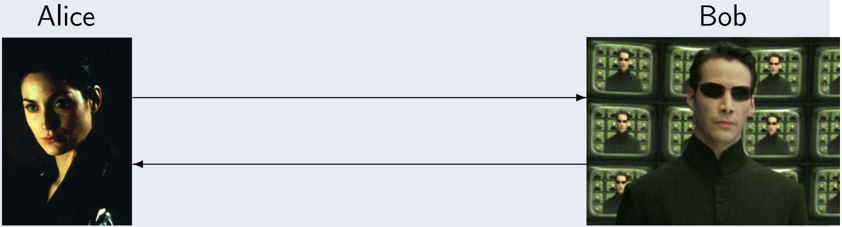


Key Exchange

Several Methods to establish a (secret) key

- Key distribution : a server chooses a key and transfers it to everyone
- Key exchange : everyone participates to the negotiated key

A *client* (Alice) authenticates herself to a server (Bob)



Alice

Bob

Key Exchange

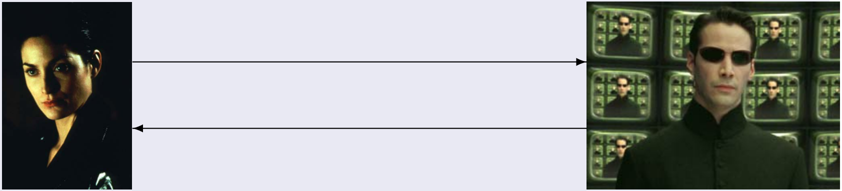
Several Methods to establish a (secret) key

- Key distribution : a server chooses a key and transfers it to everyone
- Key exchange : everyone participates to the negotiated key

A *client* (Alice) authenticates herself to a server (Bob)

Alice

Bob



K

K

Authentication

A *client* authenticates himself to a *server*

Usual situations :

- access control
- internet connexion, VPN, mobile phone, ...

General Principle

The server "tests" the client by performing it an operation that only the legitimate client can do correctly

Authentication

Example (Challenge/Response)

Decrypt a random value sent by the server

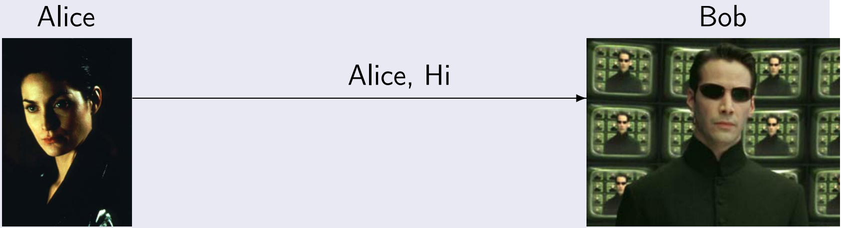
Alice

Bob

Authentication

Example (Challenge/Response)

Decrypt a random value sent by the server



Alice

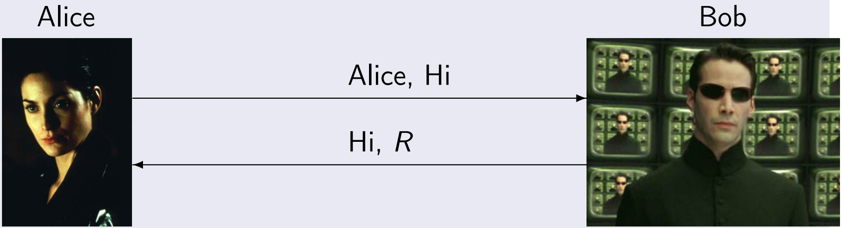
Alice, Hi

Bob

Authentication

Example (Challenge/Response)

Decrypt a random value sent by the server



Alice

Bob

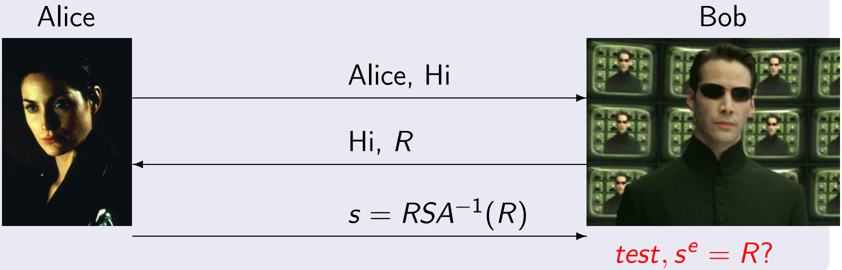
Alice, Hi

Hi, R

Authentication

Example (Challenge/Response)

Decrypt a random value sent by the server



Alice

Bob

Alice, Hi

Hi, R

$s = RSA^{-1}(R)$

$test, s^e = R?$

Authentication

Two distinct frameworks for authentication

1. No active attack : after the authentication, the server knows he is still discussing with the client
2. In case of active attack : the attacker can substitute to the authenticated client after the authentication ⇒ use a key exchange

Key Exchange

key agreement better name

Definition : key agreement

Generating a common secret (session key) by 2 entities discussing on an unsecure channel

Two possible attack scenarii :

- passive attacks : channel *listened* only
- arbitrary active attacks

Diffie-Hellman Protocol

Allows two actors to negotiate a common secret through a public channel

Alice

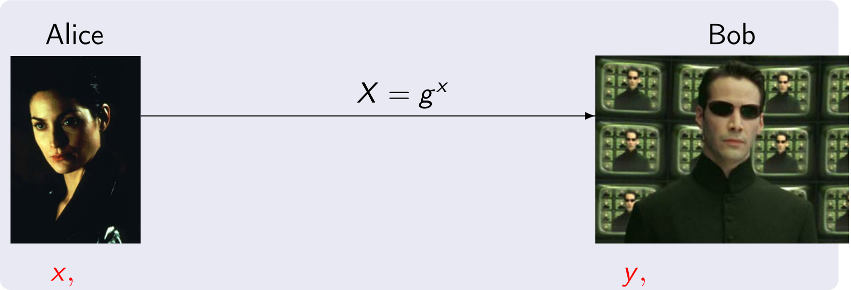
Bob

x,

y,

Diffie-Hellman Protocol

Allows two actors to negotiate a common secret through a public channel



Alice

Bob

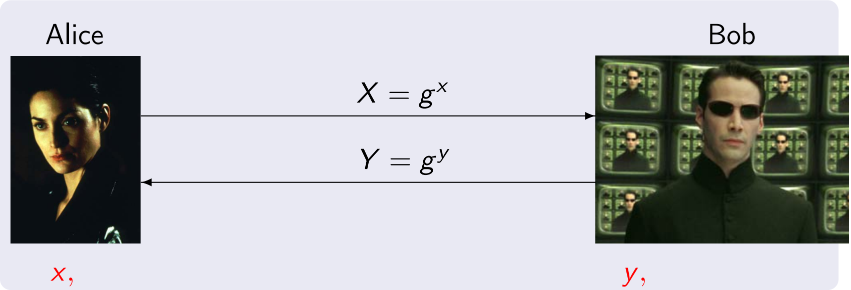
$X = g^x$

x,

y,

Diffie-Hellman Protocol

Allows two actors to negotiate a common secret through a public channel



Alice

Bob

$X = g^x$

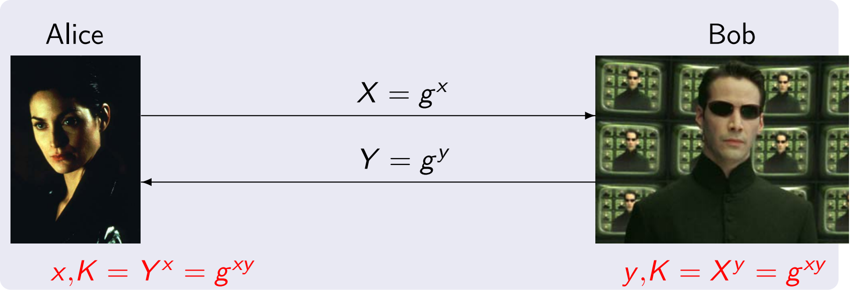
$Y = g^y$

$x,$

$y,$

Diffie-Hellman Protocol

Allows two actors to negotiate a common secret through a public channel



Alice

Bob

$X = g^x$

$Y = g^y$

$x, K = Y^x = g^{xy}$

$y, K = X^y = g^{xy}$

Common key $K = g^{xy}$

Key exchange and authentication

Without authentication : we do not check with who are we talking (we trust each other)

The interested property is then the key confidentiality

1. most important property in a key exchange
2. which confidentiality ?
   - complete : the adversary $\mathcal{A}$ cannot compute the key k
   - indistinguishability : $\mathcal{A}$ cannot distinguish k from a random element
3. which environment ?
   - what garanties if multiple parallel sessions ?
   - passive or active attacks (*concurrent* sessions) ?

Key exchange and authentication

Authentication : only the legitimate users can authenticate themselves

Authentication can be :

- unilateral (*e.g.* `https`)
- bilateral ou mutual : each party performs a verification
  - authenticated key exchange (AKE)
  - succeeds only if the identities are authenticated

Same questions on the environment and attacker model

Additional properties (integrity of the session key)

An active attacker must not be able to manipulate the exchanged messages and modify the final value of the session key

Possible issues :

- the attacker can make the protocol failed
- the attacker can force some bits of the key
- the attacker can force the value of the key
- the attacker can link the new key to previous ones (*e.g.* same unknown key established twice)

Additional properties (impossibility to control the session key)

No participant of a key exchange must be able to control the final value of the session key

Possible issues :

- a participant can force a "weak key"
- or simply a key known by someone else

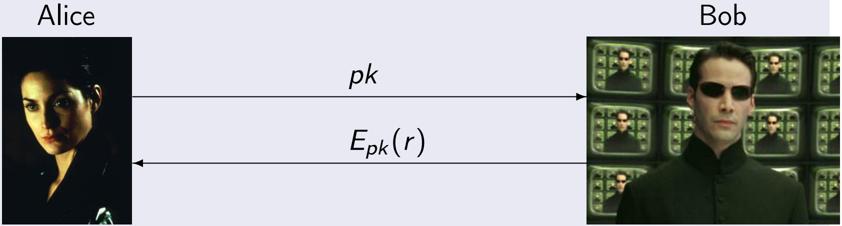
Hard to enforce in practice ...

"Session-ID" and active attacks

Simple technique to "avoid" any modification of the exchanged message in the protocol

Add in the key derivation a *session-id*, *i.e.* a quantity which depends on the set of all exchanged messages

Bad Key Exchange



Alice

Bob

pk

$E_{pk}(r)$

Sends her pk

random r

Common key $= r$

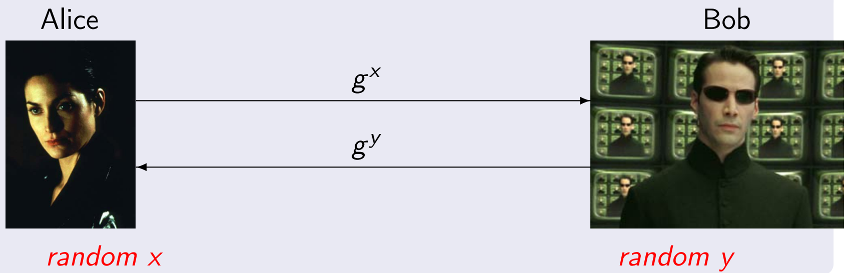
- Bob has the control on the key !
- Advantage : if Alice later proves her knowledge of the key r, she authenticates herself
- used in SSH v1 and SSL

Authenticated key exchange

Key exchange protocols proposed in 1976

- seminal article "New Directions in Cryptography" https://ee.stanford.edu/~hellman/publications/24.pdf
- based on the *one-way function* concept
- non authenticated in the basic version

Authenticated Diffie-Hellman key exchange



Alice

Bob

g^x

g^y

random x

random y

Common secret value : $K = gxy = (g^x)^y = (g^y)^x$

- mod p or over an elliptic curve
- forward secure : no long-term keys involved

Ephemeral keys and forward secrecy

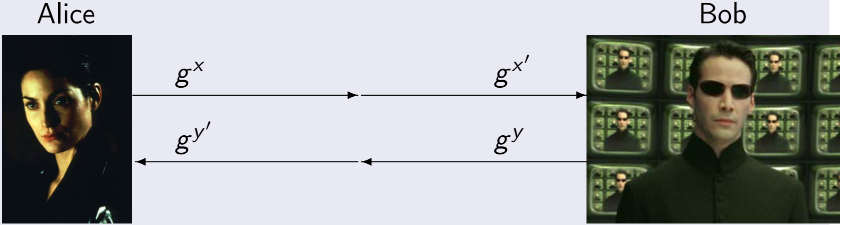
Forward secrecy : the session keys are not put in danger by leakage of long term keys

Simple mechanism to achieve it : use ephemeral public keys to protect the session key

- optionally in combinaison with long-term keys
- ephemeral secret keys have to be erased as soon as the exchange is over
- *e.g.* "Server key" in SSH v1

Diffie-Hellman Key Exchange and Active Attacks

DH protocol is vulnerable to active attacks (man-in-the-middle) :



Alice

Bob

g^x

$g^{x'}$

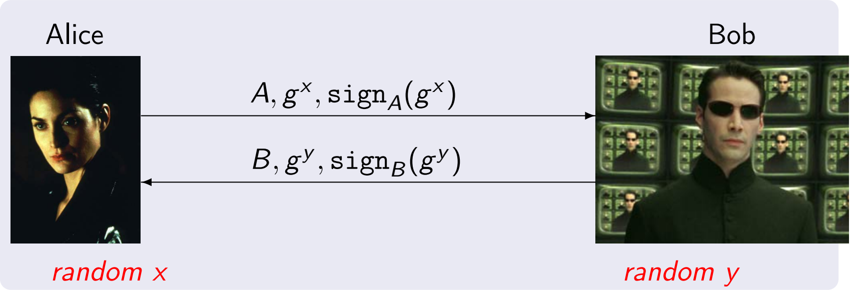
$g^{y'}$

g^y

$random\ x,\ K = g^{x'y}$

$random\ y,\ K = g^{x'y}$

Signed Diffie-Hellman version 1



Alice
Bob
$A, g^x, \mathtt{sign}_A(g^x)$
$B, g^y, \mathtt{sign}_B(g^y)$
random x
random y

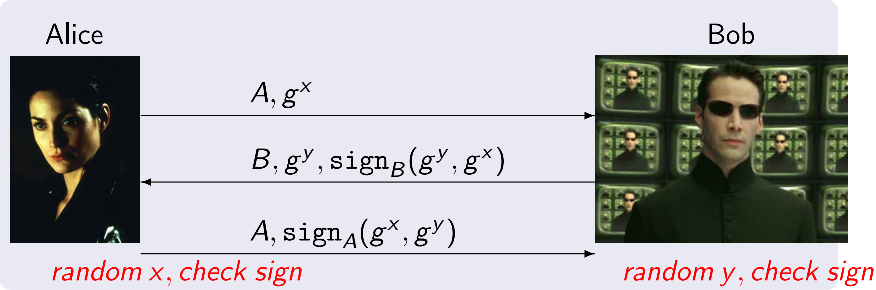
Avoid man-in-the-middle attacks :

- impossible for an attacker to forge $\mathtt{sign}_A(g^{x'})$

Replay attacks (Non fresh values) :

- if a pair $(g^x, \mathtt{sign}_A(g^{x'}))$ is obtained, we can use it to authenticate as A several times
- but we are not able to compute the corresponding key

Station-to-station protocol (Signed Diffie-Hellman version 2)



Alice

Bob

A, g^x

$B, g^y, \mathtt{sign}_B(g^y, g^x)$

$A, \mathtt{sign}_A(g^x, g^y)$

random x, check sign

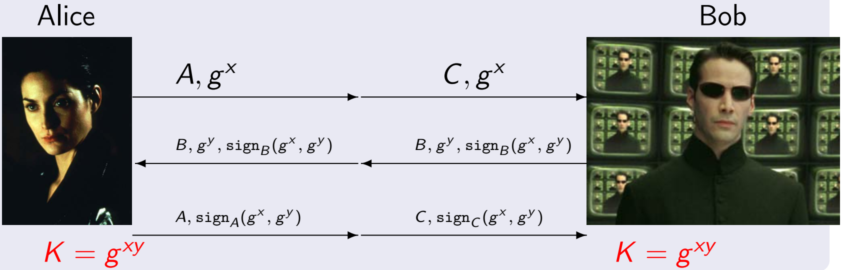
random y, check sign

Best of two worlds :

- avoid man-in-the-middle attacks : impossible for an attacker to forge $\mathtt{sign}_B(g^y, g^x)$
- avoid replay : the DH value of the correspondant is also signed

But ...

Identity Impersonation



Alice

Bob

A, g^x

C, g^x

$B, g^y, \mathtt{sign}_B(g^x, g^y)$

$B, g^y, \mathtt{sign}_B(g^x, g^y)$

$A, \mathtt{sign}_A(g^x, g^y)$

$C, \mathtt{sign}_C(g^x, g^y)$

$K = g^{xy}$

$K = g^{xy}$

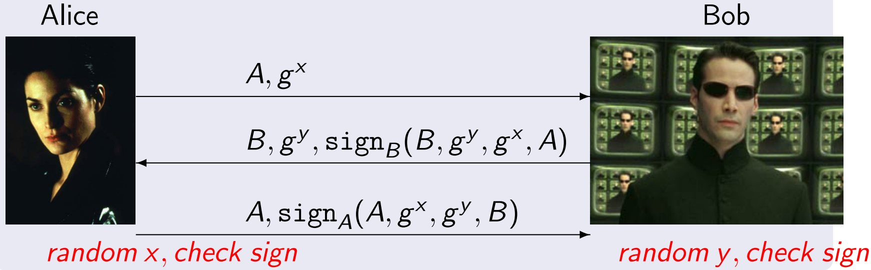
- all exchanged messages sent by Alice are seen by Bob as coming from Charlie
- Charlie does not know the key (g^{xy}) : Unknown Key-Share attack (UKS)

Identity Impersonation

Identity impersonation can have dramatic effects

- Bob is a bank ...
- Alice and Charlie are two customers
- Alice put money on her account with digital cash
- the bank puts money on Charlie account

Signed Diffie-Hellman version 3



Alice

Bob

A, g^x

$B, g^y, \mathtt{sign}_B(B, g^y, g^x, A)$

$A, \mathtt{sign}_A(A, g^x, g^y, B)$

random x, check sign

random y, check sign

Security Proof :

- forward secrecy : key integrity, protection of the long-term secrets (signature keys), ...
- **ISO/IEC 9798-3 Standard**

Introduction on SSL / TLS

SSL Protocol (Secure Socket Layer)

- developed by Netscape Inc. to secure communications
- OSI application level
- started deployed (1994) with SSL v2 ... (56 bits DES, MD5)
- 1996 : SSL v3 (SSL v2 is not secure and no more used)
- TLS 1.0 ($\approx$ SSL v3) in 1999
- TLS 1.2 (January 2008), RFC 4346
- Nowadays TLS 1.3 (August 2018), RFC 8446

Security Objectives

Develop a secure API using objects that can be viewed as sockets by the applications using them

Compatibility effort :

- redeploy existing applications
- intermediate layer under existing protocols : HTTP, POP, IMAP, ...

Scheduling of the protocol

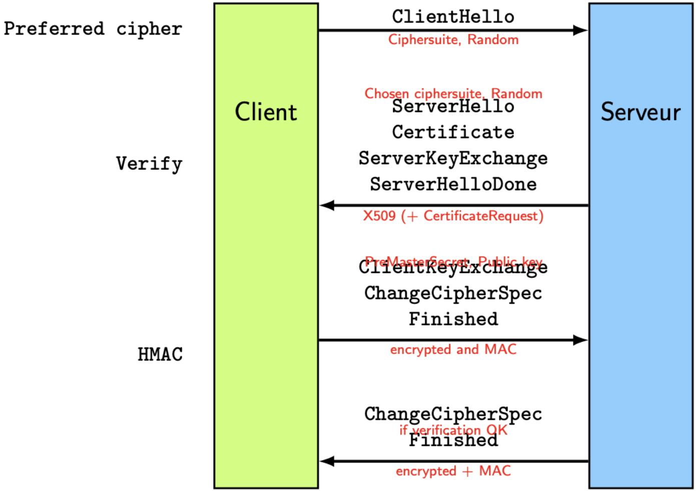
The Scheduling of a session protected by TLS is composed of 2 stages :

- Handshake Phase : Negotiation of a cipher suite (cryptographic algorithms to encrypt, authenticate, ...) and key exchange
- Record Phase : encryption and authentication of the communications

The authentication is unilateral (server authenticate only)

- the mutual authentication is heavy to deploy in practice
- variants (TLS-PSK, TLS-SRP, ...)

Scheduling of the protocol



Preferred cipher

Verify

HMAC

Client

Serveur

ClientHello

Ciphersuite, Random

Chosen ciphersuite, Random

ServerHello
Certificate
ServerKeyExchange
ServerHelloDone

X509 (+ CertificateRequest)

PreMasterSecret, Public key

ClientKeyExchange
ChangeCipherSpec
Finished

encrypted and MAC

ChangeCipherSpec

if verification OK

Finished

encrypted + MAC