Crypto 101

Concepts

- Fonctions de hachage
- Cryptage symétrique
- Cryptage asymétrique
- Échanges Diffie-Hellman
- Signature cryptographique

Applications

- Échange de messages confidentiels (PGP, iMessages, WhatsApp, Signal)
- Établissement de connections sécurisées (SSH, TLS/SSL, ZRTP)
- Authentification par clé publique SSH
- Validation de certificats et chaîne de confiance
- Arnaques pyramidales (Bitcoin)

Algorithmes de hachage

Fonction à un seul sens:

- `f(x) = y` est simple à calculer
- Trouver `x` à partir de `y` est quasi impossible

Donnée de taille quelconque en entrée
=> donne une *empreinte* de taille fixe en sortie.

Exemples:

- famille SHA (1, 2, ...),
- MD5,
- Whirlpool

Exemple

```
➜ echo "123456789" | sha256sum
```

6d78392a5886177fe5b86e585a0b695a2bcd01a05504b3c4e38bc8eeb21e8326

```
➜ echo "123456779" | sha256sum
```

8f9d6dbc5c656b3fd63f25e72c3ec9d7738f198238a46eeb01875ee102c34860

Petit changement => grande différence

Cryptage symétrique

> Permet à la fois de chiffrer et de déchiffrer des messages à l'aide d'une même clé. (Wikipédia)

Exemples:

- AES (Rijndael)
- ENIGMA
- Blowfish
- ROT13

Comment faire transiter la clé de manière sécurisée?

- échange physique
- canal sécurisé (comment le sécuriser?)

Cryptage asymétrique

Algorithme qui utilise 2 clés au lieu d'une: Chiffrement avec une clé, déchiffrement avec l'autre.

- 1 clé reste privée
- l'autre peut être rendue publique et communiquée

Exemples:

- RSA
- EcDSA
- ElGamal

Cryptage asymétrique

Propriétés des algorithmes asymétriques:

- À partir d'une clé, on ne peut pas déduire l'autre dans un temps raisonnable
- Généralement plus complexes et pas adaptés à de gros volumes de données
- Fonctionne dans les deux sens!
    - je crypte avec la clé publique, je ne peux décrypter qu'avec la clé privée (permet les communications confidentielles)
    - je crypte avec la clé privée, je ne peux decrypter qu'avec la clé publique (permet l'authentification d'un message)

Réaliser un échange confidentiel

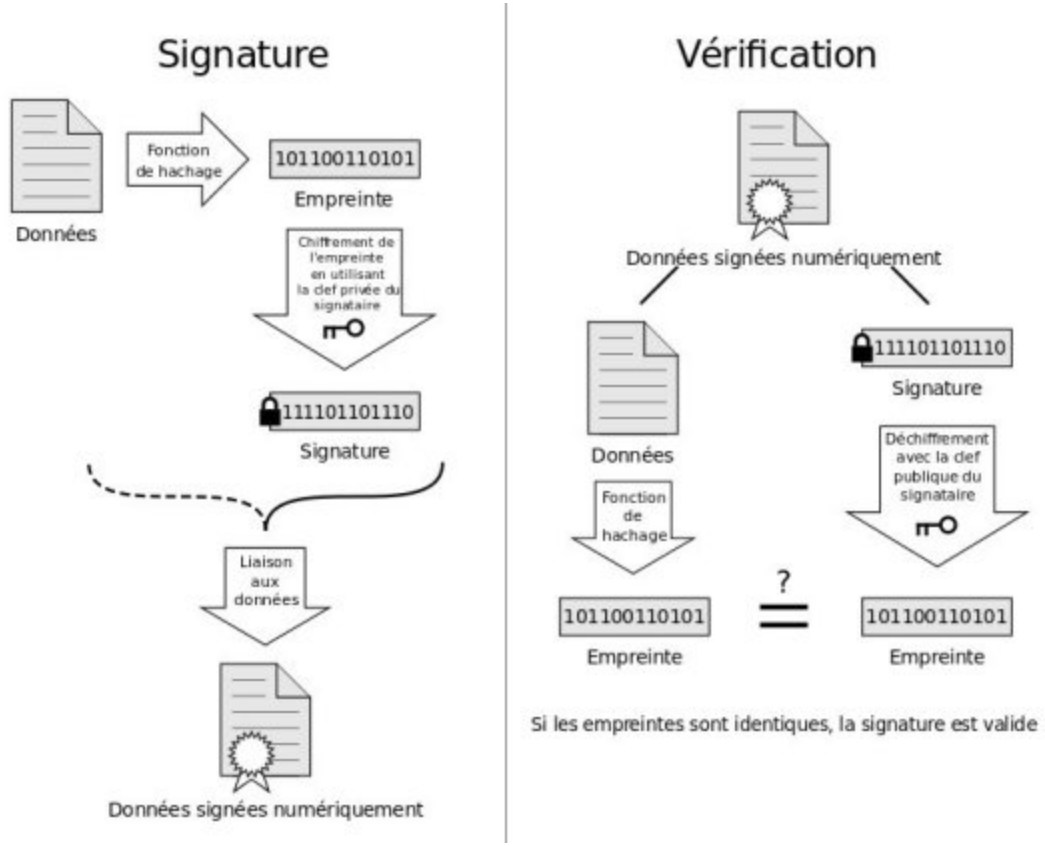
1. Je donne ma clé publique à mon correspondant sans me soucier du medium
2. Il l'utilise pour crypter son message et me l'envoie crypté sans se soucier du medium
3. Personne d'autre que moi ne peut décrypter le message, car je suis le seul à avoir la clé privée

C'est le principe de PGP et GPG.
Gros problème: comment savoir que c'est bien mon correspondant qui a écrit le message?

Solution: Signature cryptographique

Signature cryptographique

Permet de s'assurer de l'auteur d'un message.

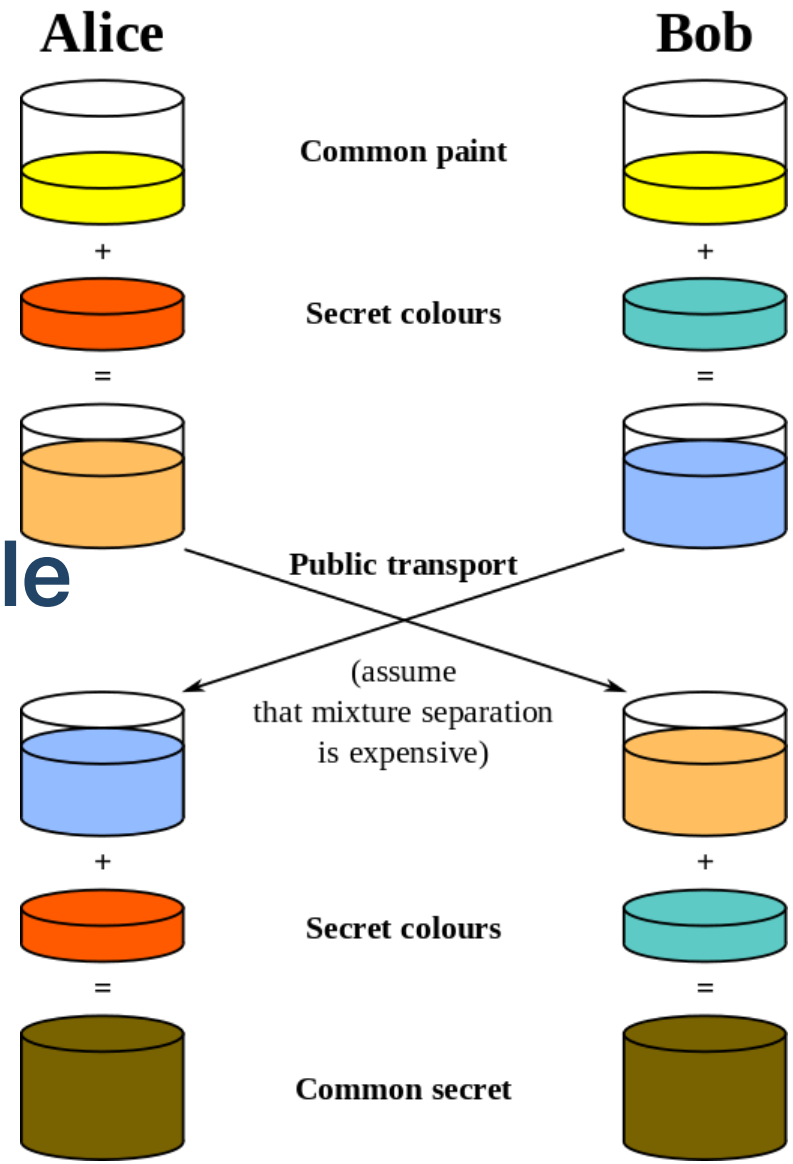


Établissement d'un canal sécurisé

Diffie-Hellman: crypto asymétrique commre RSA, formalise la manière dont 2 correspondants peuvent négocier une clé de chiffrement (**symétrique**).

Utilise le principe de chiffrement asymétrique.

Explication visuelle



Authentification par clé publique

- Quelqu'un se présente masqué à moi, mais je connais sa clé publique.
- Pour savoir si c'est bien celui qu'il dit être, je choisis un nombre aléatoire, je le chiffre avec sa clé publique, et lui envoie avec ma clé publique (on appelle ça un "Challenge")
- Si c'est bien lui, il peut décrypter mon message avec sa clé privée, récupérer le nombre aléatoire, et le réencrypter avec ma clé publique (c'est la "Réponse")
- Je reçois le message, le décrypte avec ma clé privée, et si le nombre qui est dedans est celui que j'ai envoyé, alors mon correspondant est bien celui qu'il dit être

Comment tout ça est utilisé en vrai

SSH

1. Établissement d'une connection TCP standard
2. Échange Diffie-Hellman pour obtenir un secret partagé
3. Mise en place d'une connection sécurisée chiffrée par AES en utilisant comme clé le secret échangé plus tôt
4. Authentification par clé publique du client si il présente une clé publique connue de `~/.ssh/authorized_keys` (sinon par password)

Authentification par chaîne de confiance (certificat)

Un certificat contient:

- une clé publique
- les infos du certificat (nom de domaine lié à ce certificat, etc.)
- une signature de ce certificat (rappel: signature = `chiffrage( clé privée, HASH( contenu du certificat) )`) par un tiers

Cette signature provient:

- soit d'une autorité de certification dont la clé publique est dans votre navigateur
- soit de la machine qui présente le certificat ("certificat autosigné")

Chaîne de certification

- Les navigateurs n'embarquent que les certificats des autorités dite "Racines", qui sont de gros groupes commerciaux audités ou des gouvernements.
- Ces certificats racines signent des certificats intermédiaires, et les fournissent aux autorités de certification "Tier 2", qui peuvent à leur tour signer des certificats
- Moi, braincube.com, demande à une de ces autorités de certification tier 2 de signer mon certificat avec sa clé privée, moyennant finances et preuves que je possède bien ce nom de domaine.

Anatomie d'un certificat

```
openssl s_client -connect mybraincube.com:443 -showcerts
```

```
Certificate:
    Signature Algorithm: sha256WithRSAEncryption                    <-- comment le certificat est signé et quelle est le type de clé publique
        Issuer: C=US, O=GeoTrust Inc., CN=RapidSSL SHA256 CA        <-- qui a signé ce certificat
        Validity
            Not Before: Mar 17 00:00:00 2016 GMT
            Not After : Mar 17 23:59:59 2017 GMT
        Subject: CN=*.mybraincube.com                               <-- pour qui ce certificat a été signé
        Subject Public Key Info:                                    <-- clé publique de *.mybraincube.com
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                [...]
        X509v3 extensions:
            X509v3 Subject Alternative Name:
                DNS:*.mybraincube.com, DNS:mybraincube.com          <-- pour quels DNS ce certificat est valide
            X509v3 Basic Constraints:
                CA:FALSE                                            <-- ce certificat n'est pas autorisé à signer d'autres certificats
    Signature Algorithm: sha1WithRSAEncryption                      <-- signature de RapidSSL, confirme ce qui est écrit au dessus
         [... signature ... ]
```

Voir certs.txt pour toute la chaîne.

PS: RapidSSL sucks

Établissement de connection SSL/TLS

TLS 1.3

1. Le client établit un canal sécurisé avec le serveur (en utilisant EDH)
2. Le serveur web présente son certificat au client
3. Le client vérifie la chaîne de certificats pour authentifier le serveur. Envoie un challenge encrypté avec la clé publique du serveur, et envoie sa clé publique
4. Le serveur web décrypte le challenge avec sa clé privée et le crypte avec la clé publique du client
5. Si le serveur arrive à renvoyer le challenge au client, c'est qu'il possède la clé privée.

Authentification par certificat client

Je demande au serveur web de me fournir un certificat. Pour cela:

1. Je crée un couple de clés, et je crée un certificat non signé avec la clé publique
2. J'envoie ce certificat pour signature ("CSR") au serveur web, qui me le renvoie signé.

Au prochain login, j'envoie mon certificat au serveur.

1. Il peut vérifier qu'il l'a bien signé en vérifiant la signature;
2. Il peut m'authentifier via l'authentification par clé publique, car ma clé est présente dans le certificat (Challenge/Réponse)

Questions

Trivia

Certificats bloqués

Nom du certificat	Publié par	Type	Dimension de clé	SIG ALG	Numéro de série	Échéance	Politiqu EV
*.EGO.GOV.TR	TÜRKTRUST Elektronik Sunucu Sertifikası Hizmetleri	RSA	2 048 bits	SHA-1	08 27	7 h 07, 51 s, 6 juil 2021	Non EV
*.google.com	*.EGO.GOV.TR	RSA	1 024 bits	SHA-1	0A 88 90 40 CE 12 6E 65 57 AE C2 42 7B 4A C1 FB	19 h 43, 27 s, 7 juin 2013	Non EV

Trivia

Lenovo Superfish

- Embarque dans tous les ordinateurs Lenovo un certificat racine autosigné
- Embarque aussi la clé privée (protégée par mot de passe)
- Le mot de passe est trouvé facilement, et des certificats bidons peuvent être générés
- TOUS les ordinateurs Lenovo pourront être dupés par ces certificats

Source

Trivia

Bitcoin

(explication au tableau si vous voulez)

Source

BREACH

Trivia

Exemple d'échange de clé DH (Wikipédia)

Exemple de chiffrage clé publique (RSA) (Wikipédia)

Support TLS du browser

RFC Certificats x509

Décoder un pem (mais sinon, utiliser openssl)

Chiffrement RSA (Wikipédia)

Chiffrement par courbes ellipriques (Wikipédia)

Un site qui explique tout ça vraiment bien (et en français)