R4B10 - Cryptographie et Sécurité

Utilisation, Outils et Gestion de clés

Bruno BEAUFILS

2023/2024

1. Utilisation

2. Outils cryptographiques

3. Gestion des clés

Utilisations principales en informatique

Protéger des données

- données = suite **quelconque** d'octets
  - longueur quelconque
  - codage quelconque
    - ce que représente les octets n'a pas d'importance
    - UTF-8, ASCII, PNG, etc.
- convention terminologique
  - données à protéger = **message** ou **fichier**

Stockage

- protéger des données en cas d'accès aux fichiers
  - ajouter une protection en cas de viol de propriété

Transmission réseau

- protéger les communications en cas d'interception des paquets
  - empêcher l'accès aux données capturées
  - garantir l'intégrité des données reçues

Code d'authentification de message (*MAC*)

- Problème de l'**intégrité**
  - message peut être modifié (volontairement ou pas) entre émission et réception
    - distribution du message et d'un haché du message
  - distribuer un message et son empreinte ne suffit pas
    - interception possible des 2
    - utilisation de canaux différents complexes
- MAC = *Message Authentication Codes*
  - fonction de hachage avec une clé secrète
  - pour lire le haché il faut connaître la clé secrète
- HMAC = *Hash-based Message Authentication Code*
  - manière de transformer une fonction de hachage classique en MAC
  - Exemples : HMAC-MD5, HMAC-SHA256
- CMAC = *Chipher-based Message Authentication Code*
  - manière d'utiliser un algorithme de chiffrement symétrique en MAC
  - Exemples : CMAC-AES, CMAC-DES, etc.

Signature numérique

- Mécanisme pour garantir l'**authenticité** d'une suite d'octets
  - comme pour la signature *classique*
- Garanties
  - **authentification** : identification du signataire
  - **intégrité** : pas de modification entre la signature et la lecture
  - **non-répudiation** : le signataire ne peut pas dire ne pas l'être
- Construction
  1. créer un code d'authentification de message de hachage à clé (*HMAC*)
  2. chiffrer le HMAC avec la clé privée
  3. transmettre le résultat avec le message envoyé
- Vérification
  1. calculer le HMAC du message reçu
  2. déchiffrer le HMAC reçu avec la clé publique
  3. comparer le HMAC calculé avec celui reçu

Signature numérique

- Mécanisme pour garantir l'**authenticité** d'une suite d'octets
  - comme pour la signature *classique*
- Garanties
  - **authentification** : identification du signataire
  - **intégrité** : pas de modification entre la signature et la lecture
  - **non-répudiation** : le signataire ne peut pas dire ne pas l'être
- Construction
  1. créer un code d'authenfication de message de hachage à clé (*HMAC*)
  2. chiffrer le HMAC avec la clé privée
  3. transmettre le résultat avec le message envoyé
- Vérification
  1. calculer le HMAC du message reçu
  2. déchiffrer le HMAC reçu avec la clé publique
  3. comparer le HMAC calculé avec celui reçu

Signature numérique

- Mécanisme pour garantir l'**authenticité** d'une suite d'octets
  - comme pour la signature *classique*
- Garanties
  - **authentification** : identification du signataire
  - **intégrité** : pas de modification entre la signature et la lecture
  - **non-répudiation** : le signataire ne peut pas dire ne pas l'être
- Construction
  1. créer un code d'authenfication de message de hachage à clé (*HMAC*)
  2. chiffrer le HMAC avec la clé privée
  3. transmettre le résultat avec le message envoyé
- Vérification
  1. calculer le HMAC du message reçu
  2. déchiffrer le HMAC reçu avec la clé publique
  3. comparer le HMAC calculé avec celui reçu

Signature numérique

- Mécanisme pour garantir l'**authenticité** d'une suite d'octets
  - comme pour la signature *classique*
- Garanties
  - **authentification** : identification du signataire
  - **intégrité** : pas de modification entre la signature et la lecture
  - **non-répudiation** : le signataire ne peut pas dire ne pas l'être
- Construction
  1. créer un code d'authenfication de message de hachage à clé (*HMAC*)
  2. chiffrer le HMAC avec la clé privée
  3. transmettre le résultat avec le message envoyé
- Vérification
  1. calculer le HMAC du message reçu
  2. déchiffrer le HMAC reçu avec la clé publique
  3. comparer le HMAC calculé avec celui reçu

Signature numérique

- Mécanisme pour garantir l'**authenticité** d'une suite d'octets
  - comme pour la signature *classique*
- Garanties
  - **authentification** : identification du signataire
  - **intégrité** : pas de modification entre la signature et la lecture
  - **non-répudiation** : le signataire ne peut pas dire ne pas l'être
- Construction
  1. créer un code d'authenfication de message de hachage à clé (*HMAC*)
  2. chiffrer le HMAC avec la clé privée
  3. transmettre le résultat avec le message envoyé
- Vérification
  1. calculer le HMAC du message reçu
  2. déchiffrer le HMAC reçu avec la clé publique
  3. comparer le HMAC calculé avec celui reçu

Signature numérique

- Mécanisme pour garantir l'**authenticité** d'une suite d'octets
  - comme pour la signature *classique*
- Garanties
  - **authentification** : identification du signataire
  - **intégrité** : pas de modification entre la signature et la lecture
  - **non-répudiation** : le signataire ne peut pas dire ne pas l'être
- Construction
  1. créer un code d'authenfication de message de hachage à clé (*HMAC*)
  2. chiffrer le HMAC avec la clé privée
  3. transmettre le résultat avec le message envoyé
- Vérification
  1. calculer le HMAC du message reçu
  2. déchiffrer le HMAC reçu avec la clé publique
  3. comparer le HMAC calculé avec celui reçu

Signature numérique

- Mécanisme pour garantir l'**authenticité** d'une suite d'octets
  - comme pour la signature *classique*
- Garanties
  - **authentification** : identification du signataire
  - **intégrité** : pas de modification entre la signature et la lecture
  - **non-répudiation** : le signataire ne peut pas dire ne pas l'être
- Construction
  1. créer un code d'authenfication de message de hachage à clé (*HMAC*)
  2. chiffrer le HMAC avec la clé privée
  3. transmettre le résultat avec le message envoyé
- Vérification
  1. calculer le HMAC du message reçu
  2. déchiffrer le HMAC reçu avec la clé publique
  3. comparer le HMAC calculé avec celui reçu

Signature numérique

- Mécanisme pour garantir l'**authenticité** d'une suite d'octets
  - comme pour la signature *classique*
- Garanties
  - **authentification** : identification du signataire
  - **intégrité** : pas de modification entre la signature et la lecture
  - **non-répudiation** : le signataire ne peut pas dire ne pas l'être
- Construction
  1. créer un code d'authenfication de message de hachage à clé (*HMAC*)
  2. chiffrer le HMAC avec la clé privée
  3. transmettre le résultat avec le message envoyé
- Vérification
  1. calculer le HMAC du message reçu
  2. déchiffrer le HMAC reçu avec la clé publique
  3. comparer le HMAC calculé avec celui reçu

1. Utilisation

2. Outils cryptographiques

3. Gestion des clés

Outils principaux sous Linux

- 2 suite d'outils
  - OpenSSL `openssl(1)`
  - GnuPG `gpg(1)`
- offrent tout ce qu'il faut pour
  - utiliser la PKI TLS
  - utiliser la PKI OpenPGP
  - chiffrer et signer des messages
- **beaucoup** d'options

1. Utilisation

2. Outils cryptographiques

3. Gestion des clés

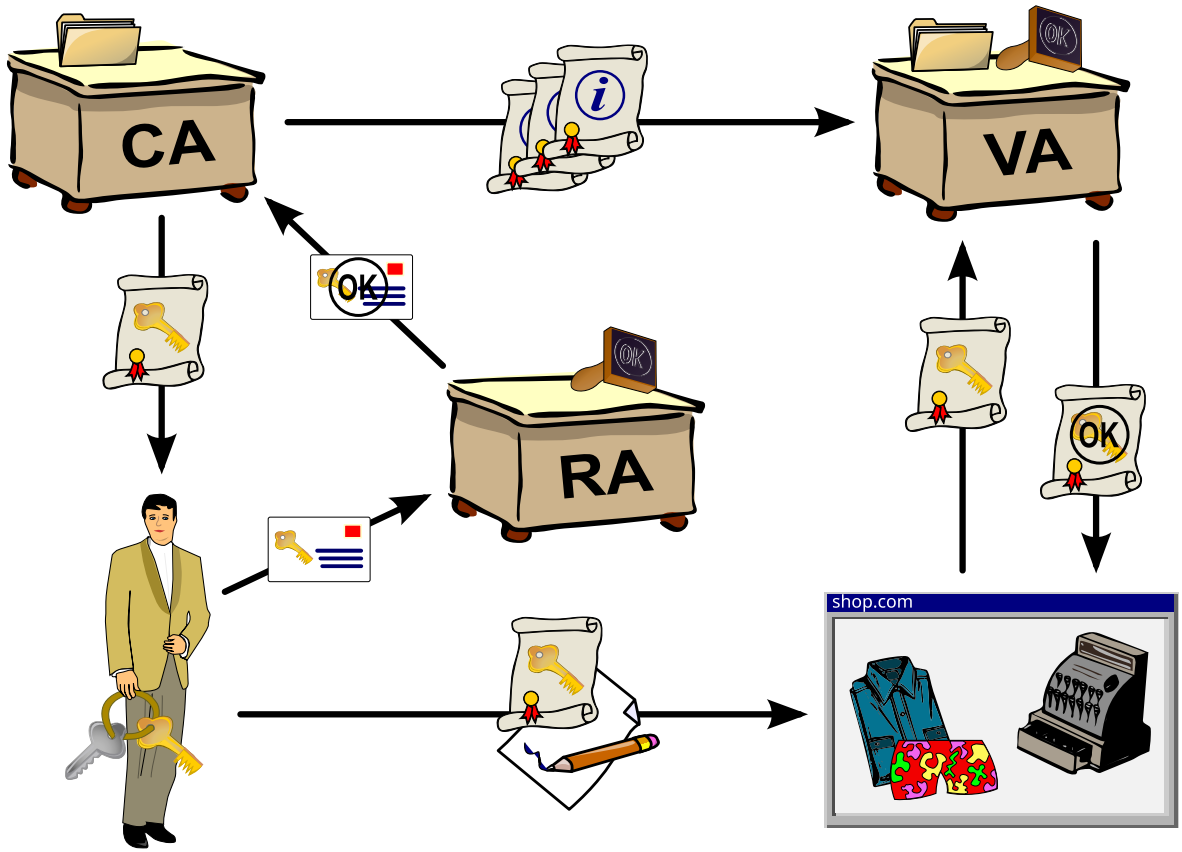
Infrastructure de gestion de clés

- *Système* permettant de gérer des clés
  - matériels
  - composants cryptographiques
  - logiciels
  - procédures humaines
  - PKI = *Public Key Infrastructure*
- Objectifs
  - lié une clé à une **identité**
  - assurer la **confiance** lors de l'utilisation d'une clé
- 2 approches différentes
  - Centralisé et hiérarchique
    - confiance donné à un **tiers** (autorité de certification)
  - Décentralisé
    - confiance donné à des **proches** (réseau de confiance)

Certificats

- fichier contenant
  - une clé publique
  - informations d'identification
  - signature
- utilisation
  - identifier
  - chiffrer

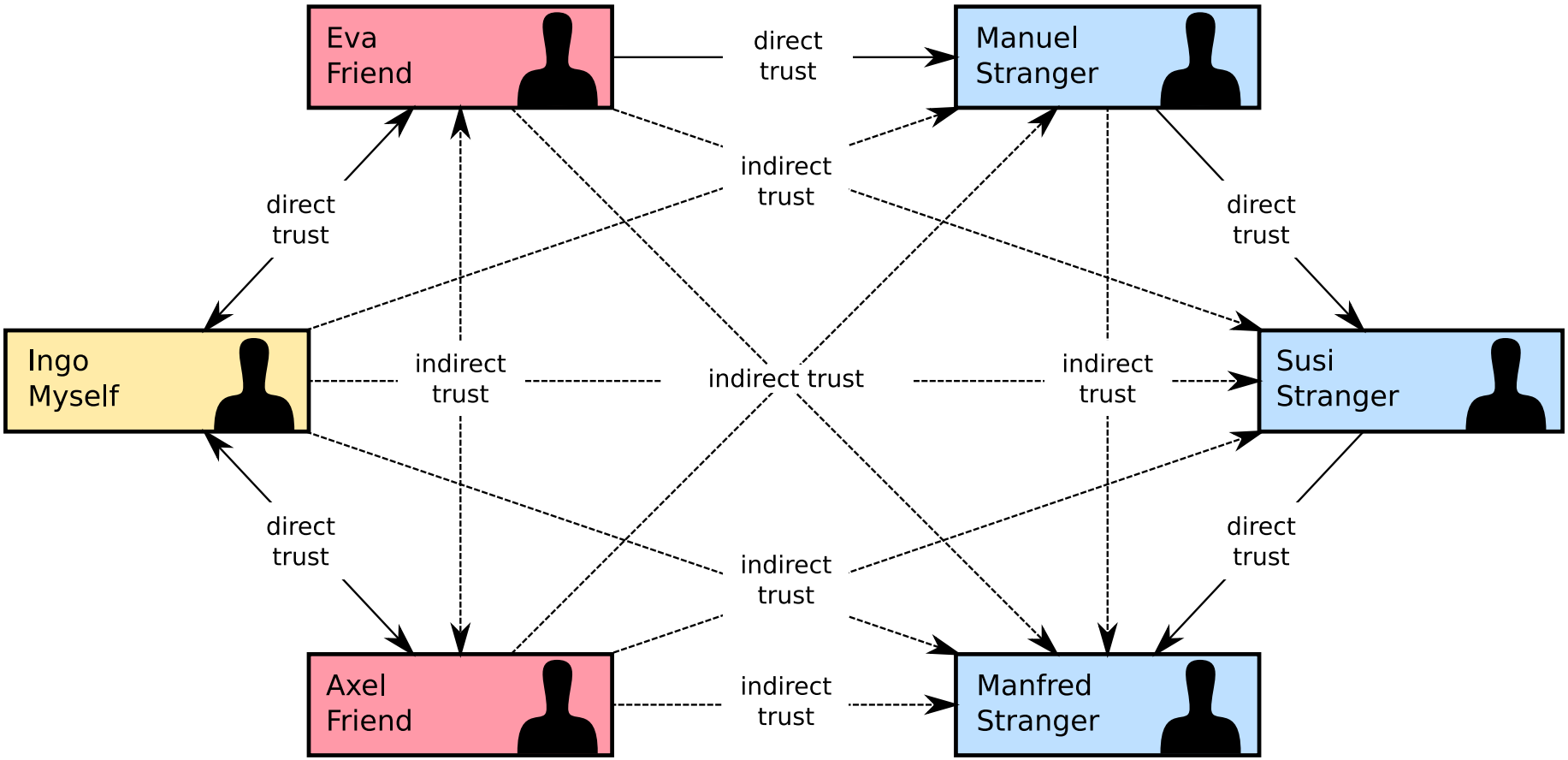
Autorités de certifications



Chris, CC BY-SA 3.0, via Wikimedia Commons

- CA : autorité de certification
- VA : autorité de validation
- RA : autorité d'enregistrement

Réseau de confiance (*web of trust*)



Kku, CC BY-SA 4.0, via Wikimedia Commons