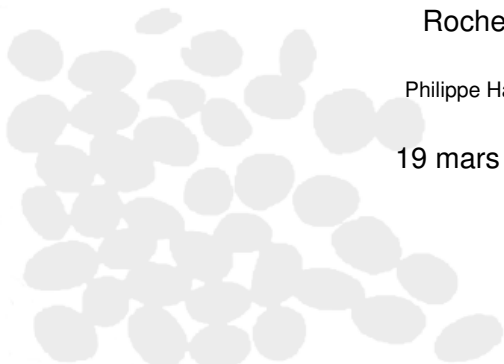


Chiffrement

Rochelug

Philippe Harrand

19 mars 2009



Chiffrement

Définitions

Authentification

Confidentialité

Confidentialité ET authentification

SSL / TLS

Chiffrement

Définitions

Authentification

Confidentialité

Confidentialité ET authentification

SSL / TLS

Chiffrement

Définitions

Authentification

Confidentialité

Confidentialité ET authentification

SSL / TLS

Chiffrement

Définitions

Authentification

Confidentialité

Confidentialité ET authentification

SSL / TLS

Chiffrement

Définitions

Authentification

Confidentialité

Confidentialité ET authentification

SSL / TLS

Chiffrement

▶ algorithme public

- ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
- ▶ Déchiffrement \Rightarrow restituer le document original

▶ Utilisation de **clefs**

▶ Chiffrement Symétrique

- ▶ Chiffrement / Déchiffrement \Rightarrow même clef
- ▶ Connue de tous les protagonistes

▶ Chiffrement Asymétrique

- ▶ Chiffrement \Rightarrow Clef privée
- ▶ Déchiffrement \Rightarrow Clef publique
- ▶ Ou l'inverse

▶ Chiffrement symétrique moins gourmand en ressources et en bande passante



Chiffrement

- ▶ **algorithme public**
 - ▶ **Chiffrement** ⇒ rendre incompréhensible un document en utilisant un **algorithme public**
 - ▶ Déchiffrement ⇒ restituer le document original
- ▶ Utilisation de **clefs**
- ▶ **Chiffrement Symétrique**
 - ▶ Chiffrement / Déchiffrement ⇒ même clef
 - ▶ Connue de tous les protagonistes
- ▶ **Chiffrement Asymétrique**
 - ▶ Chiffrement ⇒ Clef privée
 - ▶ Déchiffrement ⇒ Clef publique
 - ▶ Ou l'inverse
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante

Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ **Déchiffrement \Rightarrow restituer le document original**
- ▶ Utilisation de **clefs**
- ▶ Chiffrement Symétrique
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ Connue de tous les protagonistes
- ▶ Chiffrement Asymétrique
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ Ou l'inverse
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante

Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ **Utilisation de clefs**
 - ▶ Chiffrement Symétrique
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ Connue de tous les protagonistes
 - ▶ Chiffrement Asymétrique
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ Ou l'inverse
 - ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante

Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ Utilisation de **clefs**
- ▶ **Chiffrement Symétrique**
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ Connue de tous les protagonistes
- ▶ Chiffrement Asymétrique
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ Ou l'inverse
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante



Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ Utilisation de **clefs**
- ▶ Chiffrement Symétrique
 - ▶ **Chiffrement / Déchiffrement \Rightarrow même clef**
 - ▶ Connue de tous les protagonistes
- ▶ Chiffrement Asymétrique
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ Ou l'inverse
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante



Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ Utilisation de **clefs**
- ▶ Chiffrement Symétrique
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ **Connue de tous les protagonistes**
- ▶ Chiffrement Asymétrique
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ Ou l'inverse
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante



Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ Utilisation de **clefs**
- ▶ Chiffrement Symétrique
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ Connue de tous les protagonistes
- ▶ **Chiffrement Asymétrique**
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ Ou l'inverse
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante



Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ Utilisation de **clefs**
- ▶ Chiffrement Symétrique
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ Connue de tous les protagonistes
- ▶ Chiffrement Asymétrique
 - ▶ **Chiffrement \Rightarrow Clef privée**
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ Ou l'inverse
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante



Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ Utilisation de **clefs**
- ▶ Chiffrement Symétrique
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ Connue de tous les protagonistes
- ▶ Chiffrement Asymétrique
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ **Déchiffrement \Rightarrow Clef publique**
 - ▶ Ou l'inverse
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante



Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ Utilisation de **clefs**
- ▶ Chiffrement Symétrique
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ Connue de tous les protagonistes
- ▶ Chiffrement Asymétrique
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ **Ou l'inverse**
- ▶ Chiffrement symétrique moins gourmand en ressources et en bande passante



Chiffrement

- ▶ algorithme public
 - ▶ Chiffrement \Rightarrow rendre incompréhensible un document en utilisant un algorithme public
 - ▶ Déchiffrement \Rightarrow restituer le document original
- ▶ Utilisation de **clefs**
- ▶ Chiffrement Symétrique
 - ▶ Chiffrement / Déchiffrement \Rightarrow même clef
 - ▶ Connue de tous les protagonistes
- ▶ Chiffrement Asymétrique
 - ▶ Chiffrement \Rightarrow Clef privée
 - ▶ Déchiffrement \Rightarrow Clef publique
 - ▶ Ou l'inverse
- ▶ **Chiffrement symétrique moins gourmand en ressources et en bande passante**



Hachage

Appliqué à un fichier ou à un texte, le hachage produit un mot

- ▶ **de taille fixée par l'algorithme utilisé**
- ▶ irréversible (l'original ne peut pas être reconstitué)
- ▶ unique (2 originaux différents ne produisent jamais le même mot)

Hachage

Appliqué à un fichier ou à un texte, le hachage produit un mot

- ▶ de taille fixée par l'algorithme utilisé
- ▶ **irréversible (l'original ne peut pas être reconstitué)**
- ▶ unique (2 originaux différents ne produisent jamais le même mot)

Hachage

Appliqué à un fichier ou à un texte, le hachage produit un mot

- ▶ de taille fixée par l'algorithme utilisé
- ▶ irréversible (l'original ne peut pas être reconstitué)
- ▶ unique (2 originaux différents ne produisent jamais le même mot)

Signature

- ▶ **Message envoyé accompagné d'un haché (MD5 ou SHA1)**
⇒ **SIGNATURE**
- ▶ Chiffré avec la clef privée de l'expéditeur
- ▶ Le destinataire recalcule le hachage à partir du message envoyé en clair
- ▶ La signature envoyée est déchiffrée avec la clef publique de l'expéditeur
- ▶ Si le haché calculé est égal à celui envoyé, cela prouve que le message n'a pas été modifié pendant le transfert et que l'expéditeur est la personne dont on a utilisé la clef publique
- ▶ Mais le contenu du message a pu être lu

Signature

- ▶ Message envoyé accompagné d'un haché (MD5 ou SHA1)
⇒ SIGNATURE
- ▶ **Chiffré avec la clef privée de l'expéditeur**
- ▶ Le destinataire recalcule le hachage à partir du message envoyé en clair
- ▶ La signature envoyée est déchiffrée avec la clef publique de l'expéditeur
- ▶ Si le haché calculé est égal à celui envoyé, cela prouve que le message n'a pas été modifié pendant le transfert et que l'expéditeur est la personne dont on a utilisé la clef publique
- ▶ Mais le contenu du message a pu être lu

Signature

- ▶ Message envoyé accompagné d'un haché (MD5 ou SHA1)
⇒ SIGNATURE
- ▶ Chiffré avec la clef privée de l'expéditeur
- ▶ **Le destinataire recalcule le hachage à partir du message envoyé en clair**
- ▶ La signature envoyée est déchiffrée avec la clef publique de l'expéditeur
- ▶ Si le haché calculé est égal à celui envoyé, cela prouve que le message n'a pas été modifié pendant le transfert et que l'expéditeur est la personne dont on a utilisé la clef publique
- ▶ Mais le contenu du message a pu être lu

Signature

- ▶ Message envoyé accompagné d'un haché (MD5 ou SHA1)
⇒ SIGNATURE
- ▶ Chiffré avec la clef privée de l'expéditeur
- ▶ Le destinataire recalcule le hachage à partir du message envoyé en clair
- ▶ **La signature envoyée est déchiffrée avec la clef publique de l'expéditeur**
 - ▶ Si le haché calculé est égal à celui envoyé, cela prouve que le message n'a pas été modifié pendant le transfert et que l'expéditeur est la personne dont on a utilisé la clef publique
 - ▶ Mais le contenu du message a pu être lu

Signature

- ▶ Message envoyé accompagné d'un haché (MD5 ou SHA1)
⇒ SIGNATURE
- ▶ Chiffré avec la clef privée de l'expéditeur
- ▶ Le destinataire recalcule le hachage à partir du message envoyé en clair
- ▶ La signature envoyée est déchiffrée avec la clef publique de l'expéditeur
- ▶ Si le haché calculé est égal à celui envoyé, cela prouve que le message n'a pas été modifié pendant le transfert et que l'expéditeur est la personne dont on a utilisé la clef publique
- ▶ Mais le contenu du message a pu être lu

Signature

- ▶ Message envoyé accompagné d'un haché (MD5 ou SHA1)
⇒ SIGNATURE
- ▶ Chiffré avec la clef privée de l'expéditeur
- ▶ Le destinataire recalcule le hachage à partir du message envoyé en clair
- ▶ La signature envoyée est déchiffrée avec la clef publique de l'expéditeur
- ▶ Si le haché calculé est égal à celui envoyé, cela prouve que le message n'a pas été modifié pendant le transfert et que l'expéditeur est la personne dont on a utilisé la clef publique
- ▶ Mais le contenu du message a pu être lu

Certificats

- ▶ **Les clefs publiques peuvent être stockées chez des "tiers de confiance"**
- ▶ Ceux-ci délivrent les clefs publiques accompagnées d'informations sur leurs propriétaires
- ▶ le tout signé avec les clefs privées des dits tiers
- ▶ Les clefs publiques de ces tiers de confiance sont authentifiées par des "autorités racines"
- ▶ Vous devez faire confiance aux "autorités racines"
- ▶ A l'intérieur d'un intranet, vous pouvez diffuser de tels certificats "racine" qui n'auront bien sûr de valeur qu'à l'intérieur de l'entreprise...
- ▶ /usr/share/ssl/certs/ca-bundle.crt

Certificats

- ▶ Les clefs publiques peuvent être stockées chez des "tiers de confiance"
- ▶ **Ceux-ci délivrent les clefs publiques accompagnées d'informations sur leurs propriétaires**
 - ▶ le tout signé avec les clefs privées des dits tiers
 - ▶ Les clefs publiques de ces tiers de confiance sont authentifiées par des "autorités racines"
 - ▶ Vous devez faire confiance aux "autorités racines"
 - ▶ A l'intérieur d'un intranet, vous pouvez diffuser de tels certificats "racine" qui n'auront bien sûr de valeur qu'à l'intérieur de l'entreprise...
 - ▶ /usr/share/ssl/certs/ca-bundle.crt

Certificats

- ▶ Les clefs publiques peuvent être stockées chez des "tiers de confiance"
- ▶ Ceux-ci délivrent les clefs publiques accompagnées d'informations sur leurs propriétaires
- ▶ **le tout signé avec les clefs privées des dits tiers**
- ▶ Les clefs publiques de ces tiers de confiance sont authentifiées par des "autorités racines"
- ▶ Vous devez faire confiance aux "autorités racines"
- ▶ A l'intérieur d'un intranet, vous pouvez diffuser de tels certificats "racine" qui n'auront bien sûr de valeur qu'à l'intérieur de l'entreprise...
- ▶ /usr/share/ssl/certs/ca-bundle.crt

Certificats

- ▶ Les clefs publiques peuvent être stockées chez des "tiers de confiance"
- ▶ Ceux-ci délivrent les clefs publiques accompagnées d'informations sur leurs propriétaires
- ▶ le tout signé avec les clefs privées des dits tiers
- ▶ **Les clefs publiques de ces tiers de confiance sont authentifiées par des "autorités racines"**
 - ▶ Vous devez faire confiance aux "autorités racines"
 - ▶ A l'intérieur d'un intranet, vous pouvez diffuser de tels certificats "racine" qui n'auront bien sûr de valeur qu'à l'intérieur de l'entreprise...
 - ▶ /usr/share/ssl/certs/ca-bundle.crt

Certificats

- ▶ Les clefs publiques peuvent être stockées chez des "tiers de confiances"
- ▶ Ceux-ci délivrent les clefs publiques accompagnées d'informations sur leurs propriétaires
- ▶ le tout signé avec les clefs privées des dits tiers
- ▶ Les clefs publiques de ces tiers de confiances sont authentifiées par des "autorités racines"
- ▶ **Vous devez faire confiance aux "autorités racines"**
- ▶ A l'intérieur d'un intranet, vous pouvez diffuser de tels certificats "racine" qui n'auront bien sûr de valeur qu'à l'intérieur de l'entreprise...
- ▶ /usr/share/ssl/certs/ca-bundle.crt

Certificats

- ▶ Les clefs publiques peuvent être stockées chez des "tiers de confiance"
- ▶ Ceux-ci délivrent les clefs publiques accompagnées d'informations sur leurs propriétaires
- ▶ le tout signé avec les clefs privées des dits tiers
- ▶ Les clefs publiques de ces tiers de confiance sont authentifiées par des "autorités racines"
- ▶ Vous devez faire confiance aux "autorités racines"
- ▶ A l'intérieur d'un intranet, vous pouvez diffuser de tels certificats "racine" qui n'auront bien sûr de valeur qu'à l'intérieur de l'entreprise...
- ▶ /usr/share/ssl/certs/ca-bundle.crt

Certificats

- ▶ Les clefs publiques peuvent être stockées chez des "tiers de confiances"
- ▶ Ceux-ci délivrent les clefs publiques accompagnées d'informations sur leurs propriétaires
- ▶ le tout signé avec les clefs privées des dits tiers
- ▶ Les clefs publiques de ces tiers de confiances sont authentifiées par des "autorités racines"
- ▶ Vous devez faire confiance aux "autorités racines"
- ▶ A l'intérieur d'un intranet, vous pouvez diffuser de tels certificats "racine" qui n'auront bien sûr de valeur qu'à l'intérieur de l'entreprise...
- ▶ [/usr/share/ssl/certs/ca-bundle.crt](#)

Confidentialité

- ▶ Le message est chiffré par l'expéditeur avec la clef publique du destinataire
 - ▶ Seul le détenteur de la clef privée pourra le déchiffrer
 - ▶ N'importe qui peut obtenir la clef publique
 - ▶ Le destinataire ne sait donc pas qui a envoyé le message

Confidentialité

- ▶ Le message est chiffré par l'expéditeur avec la clef publique du destinataire
- ▶ **Seul le détenteur de la clef privée pourra le déchiffrer**
- ▶ N'importe qui peut obtenir la clef publique
- ▶ Le destinataire ne sait donc pas qui a envoyé le message

Confidentialité

- ▶ Le message est chiffré par l'expéditeur avec la clef publique du destinataire
- ▶ Seul le détenteur de la clef privée pourra le déchiffrer
- ▶ **N'importe qui peut obtenir la clef publique**
- ▶ Le destinataire ne sait donc pas qui a envoyé le message

Confidentialité

- ▶ Le message est chiffré par l'expéditeur avec la clef publique du destinataire
- ▶ Seul le détenteur de la clef privée pourra le déchiffrer
- ▶ N'importe qui peut obtenir la clef publique
- ▶ Le destinataire ne sait donc pas qui a envoyé le message

Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul a pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul à pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ **Puis le message est chiffré avec la clef publique du destinataire**
- ▶ Le destinataire est le seul à pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ **Le destinataire est le seul a pouvoir déchiffrer le message**
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul a pouvoir déchiffrer le message
- ▶ **Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer**
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul à pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ **Coûteux en ressources : deux chiffrements/déchiffrements**
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul à pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ **A réserver aux messages uniques**

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul à pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul à pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ **Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"**
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul à pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ **Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser**
- ▶ Clef de session envoyée par la méthode 1



Confidentialité ET authentification

▶ Méthode 1

- ▶ Le message est haché par l'expéditeur et "signé" avec sa clef privée
- ▶ Puis le message est chiffré avec la clef publique du destinataire
- ▶ Le destinataire est le seul a pouvoir déchiffrer le message
- ▶ Le haché étant déchiffré avec la clef publique de l'expéditeur, il est le seul à avoir pu l'envoyer
- ▶ Coûteux en ressources : deux chiffrements/déchiffrements
- ▶ A réserver aux messages uniques

▶ Méthode 2 :

- ▶ Le chiffrement du dialogue est réalisé avec une clef symétrique temporaire dite "clef de session"
- ▶ Changer la clef de session avec une fréquence supérieure au volume de données nécessaire pour la casser
- ▶ **Clef de session envoyée par la méthode 1**



Généralité

- ▶ **SSL/TLS** ⇒ **authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité**
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ Diffusé en 1994
- ▶ Racheté par l'**IETF** en 2001
- ▶ Rebaptisé **Transport Layer Security**
- ▶ SSL ⇒ V3
- ▶ TLS ⇒ V1.0
- ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
- ▶ TLS est négocié par les applicatifs
- ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer créé par Netscape**
 - ▶ Diffusé en 1994
 - ▶ Racheté par l'IETF en 2001
 - ▶ Rebaptisé Transport Layer Security
 - ▶ SSL \Rightarrow V3
 - ▶ TLS \Rightarrow V1.0
 - ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
 - ▶ TLS est négocié par les applicatifs
 - ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ **Diffusé en 1994**
- ▶ Racheté par l'IETF en 2001
- ▶ Rebaptisé Transport Layer Security
- ▶ SSL \Rightarrow V3
- ▶ TLS \Rightarrow V1.0
- ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
- ▶ TLS est négocié par les applicatifs
- ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ Diffusé en 1994
- ▶ **Racheté par l'IETF en 2001**
- ▶ Rebaptisé Transport Layer Security
- ▶ SSL \Rightarrow V3
- ▶ TLS \Rightarrow V1.0
- ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
- ▶ TLS est négocié par les applicatifs
- ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ Diffusé en 1994
- ▶ Racheté par l'**IETF** en 2001
- ▶ **Rebaptisé Transport Layer Security**
- ▶ SSL \Rightarrow V3
- ▶ TLS \Rightarrow V1.0
- ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
- ▶ TLS est négocié par les applicatifs
- ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ Diffusé en 1994
- ▶ Racheté par l'**IETF** en 2001
- ▶ Rebaptisé **Transport Layer Security**
- ▶ **SSL \Rightarrow V3**
- ▶ TLS \Rightarrow V1.0
- ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
- ▶ TLS est négocié par les applicatifs
- ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ Diffusé en 1994
- ▶ Racheté par l'**IETF** en 2001
- ▶ Rebaptisé **Transport Layer Security**
- ▶ SSL \Rightarrow V3
- ▶ **TLS \Rightarrow V1.0**
- ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
- ▶ TLS est négocié par les applicatifs
- ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ Diffusé en 1994
- ▶ Racheté par l'**IETF** en 2001
- ▶ Rebaptisé **Transport Layer Security**
- ▶ SSL \Rightarrow V3
- ▶ TLS \Rightarrow V1.0
- ▶ **SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)**
- ▶ TLS est négocié par les applicatifs
- ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ Diffusé en 1994
- ▶ Racheté par l'**IETF** en 2001
- ▶ Rebaptisé **Transport Layer Security**
- ▶ SSL \Rightarrow V3
- ▶ TLS \Rightarrow V1.0
- ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
- ▶ **TLS est négocié par les applicatifs**
- ▶ Openssl implémente les deux

Généralité

- ▶ SSL/TLS \Rightarrow authentification mutuelle serveur et client, chiffrement, vérification de l'intégrité
- ▶ **Secure Socket Layer** créé par Netscape
- ▶ Diffusé en 1994
- ▶ Racheté par l'**IETF** en 2001
- ▶ Rebaptisé **Transport Layer Security**
- ▶ SSL \Rightarrow V3
- ▶ TLS \Rightarrow V1.0
- ▶ SSL est utilisé sur des ports TCP spéciaux (443 https, 995 pop3s, 636 ldaps)
- ▶ TLS est négocié par les applicatifs
- ▶ **Openssl implémente les deux**

Fonctionnement

- ▶ Le client se connecte au serveur et lui demande de s'authentifier
- ▶ Le client envoie également la liste des cryptosystèmes supportés
- ▶ Le serveur envoie un certificat au client, contenant sa clef publique, signée par une autorité de certification (CA)
- ▶ Ainsi que le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible
- ▶ Le client vérifie la validité du certificat
- ▶ Crée une clef pseudo-aléatoire, la chiffre à l'aide de la clé publique du serveur
- ▶ Le serveur déchiffre la clef de session avec sa clef privée

Fonctionnement

- ▶ Le client se connecte au serveur et lui demande de s'authentifier
- ▶ **Le client envoie également la liste des cryptosystèmes supportés**
- ▶ Le serveur envoie un certificat au client, contenant sa clef publique, signée par une autorité de certification (CA)
- ▶ Ainsi que le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible
- ▶ Le client **vérifie la validité du certificat**
- ▶ Crée une clef pseudo-aléatoire, la chiffre à l'aide de la clé publique du serveur
- ▶ Le serveur déchiffre la clef de session avec sa clef privée

Fonctionnement

- ▶ Le client se connecte au serveur et lui demande de s'authentifier
- ▶ Le client envoie également la liste des cryptosystèmes supportés
- ▶ **Le serveur envoie un certificat au client, contenant sa clef publique, signée par une autorité de certification (CA)**
 - ▶ Ainsi que le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible
 - ▶ Le client **vérifie la validité du certificat**
 - ▶ Crée une clef pseudo-aléatoire, la chiffre à l'aide de la clé publique du serveur
 - ▶ Le serveur déchiffre la clef de session avec sa clef privée

Fonctionnement

- ▶ Le client se connecte au serveur et lui demande de s'authentifier
- ▶ Le client envoie également la liste des cryptosystèmes supportés
- ▶ Le serveur envoie un certificat au client, contenant sa clef publique, signée par une autorité de certification (CA)
 - ▶ Ainsi que le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible
- ▶ Le client vérifie la validité du certificat
- ▶ Crée une clef pseudo-aléatoire, la chiffre à l'aide de la clé publique du serveur
- ▶ Le serveur déchiffre la clef de session avec sa clef privée

Fonctionnement

- ▶ Le client se connecte au serveur et lui demande de s'authentifier
- ▶ Le client envoie également la liste des cryptosystèmes supportés
- ▶ Le serveur envoie un certificat au client, contenant sa clef publique, signée par une autorité de certification (CA)
- ▶ Ainsi que le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible
- ▶ **Le client vérifie la validité du certificat**
- ▶ Crée une clef pseudo-aléatoire, la chiffre à l'aide de la clé publique du serveur
- ▶ Le serveur déchiffre la clef de session avec sa clef privée



Fonctionnement

- ▶ Le client se connecte au serveur et lui demande de s'authentifier
- ▶ Le client envoie également la liste des cryptosystèmes supportés
- ▶ Le serveur envoie un certificat au client, contenant sa clef publique, signée par une autorité de certification (CA)
- ▶ Ainsi que le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible
- ▶ Le client **vérifie la validité du certificat**
- ▶ **Crée une clef pseudo-aléatoire, la chiffre à l'aide de la clé publique du serveur**
- ▶ Le serveur déchiffre la clef de session avec sa clef privée

Fonctionnement

- ▶ Le client se connecte au serveur et lui demande de s'authentifier
- ▶ Le client envoie également la liste des cryptosystèmes supportés
- ▶ Le serveur envoie un certificat au client, contenant sa clef publique, signée par une autorité de certification (CA)
- ▶ Ainsi que le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible
- ▶ Le client **vérifie la validité du certificat**
- ▶ Crée une clef pseudo-aléatoire, la chiffre à l'aide de la clé publique du serveur
- ▶ **Le serveur déchiffre la clef de session avec sa clef privée**

