Version : **2021.01**

Dernière mise-à-jour : 2021/03/02 14:27

LRF405 - Cryptologie

Contenu du module

Le Problématique

Le **sniffing** des paquets de données est possible sur un réseau utilisant une technologie de diffusion tel un réseau **Ethernet**. En effet certains protocoles ne cryptent pas les données avant de les envoyer dont :

- Telnet,
- Rlogin,
- Ftp,
- Pop3.

Un *sniffeur* est un logiciel qui captent les paquets circulant sur un réseau de type diffusion afin de les analyser. Le logiciel les plus utilisé est :

- Tcpdump.

LAB #1 - Utilisation de tcpdump

Le logiciel **tcpdump** sert à écouter le réseau en interceptant les paquets.

Options de la commande

Les options de cette commande sont :

```
[root@centos7 ~]# tcpdump --help
tcpdump version 4.9.2
libpcap version 1.5.3
OpenSSL 1.0.2k-fips  26 Jan 2017
Usage: tcpdump [-aAbdDefhHIJKlLnNOpqStuUvxX#] [ -B size ] [ -c count ]
        [ -C file_size ] [ -E algo:secret ] [ -F file ] [ -G seconds ]
        [ -i interface ] [ -j tstamptype ] [ -M secret ] [ --number ]
        [ -Q|-P in|out|inout ]
        [ -r file ] [ -s snaplen ] [ --time-stamp-precision precision ]
        [ --immediate-mode ] [ -T type ] [ --version ] [ -V file ]
        [ -w file ] [ -W filecount ] [ -y datalinktype ] [ -z postrotate-command ]
```

```
        [ -Z user ] [ expression ]
```

Utilisation

L'option -i

Pour écouter sur une **interface spécifique**, utilisez l'option **-i** :

```
[root@centos7 ~]# tcpdump -i enp0s3
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:32:57.800710 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2207704927:2207705115, ack
23445380, win 40096, length 188
01:32:57.801785 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 188, win 65535, length 0
01:32:57.805318 IP centos7.fenestros.loc.45319 > google-public-dns-a.google.com.domain: 62187+ PTR? 2.2.0.10.in-
addr.arpa. (39)
01:32:57.862866 IP google-public-dns-a.google.com.domain > centos7.fenestros.loc.45319: 62187 NXDomain 0/0/0 (39)
01:32:57.873506 IP centos7.fenestros.loc.54658 > google-public-dns-a.google.com.domain: 6854+ PTR? 15.2.0.10.in-
addr.arpa. (40)
01:32:57.934593 IP google-public-dns-a.google.com.domain > centos7.fenestros.loc.54658: 6854 NXDomain 0/0/0 (40)
01:32:57.947943 IP centos7.fenestros.loc.53477 > google-public-dns-a.google.com.domain: 63054+ PTR? 8.8.8.8.in-
addr.arpa. (38)
01:32:57.948649 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 188:464, ack 1, win 40096, length
276
01:32:57.958724 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 464, win 65535, length 0
01:32:58.025768 IP google-public-dns-a.google.com.domain > centos7.fenestros.loc.53477: 63054 1/0/0 PTR google-
public-dns-a.google.com. (82)
01:32:58.026959 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 464:1476, ack 1, win 40096, length
1012
01:32:58.027640 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 1476:1632, ack 1, win 40096, length
156
01:32:58.028108 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 1476, win 65535, length 0
```

```
01:32:58.028146 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 1632, win 65535, length 0
01:32:58.028313 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 1632:1788, ack 1, win 40096, length
156
01:32:58.028822 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 1788:2152, ack 1, win 40096, length
364
01:32:58.029240 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 1788, win 65535, length 0
01:32:58.029273 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 2152, win 65535, length 0
01:32:58.029710 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2152:2516, ack 1, win 40096, length
364
01:32:58.030217 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 2516, win 65535, length 0
01:32:58.030773 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2516:2776, ack 1, win 40096, length
260
01:32:58.034485 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 2776, win 65535, length 0
...
```

Notez qu'à la fin, un résumé vous est présenté, par exemple :

```
...
^C
767 packets captured
767 packets received by filter
0 packets dropped by kernel
```

L'option -x

Pour écouter sur une interface spécifique et voir le contenu en Hexadécimal, utilisez les options -i et **-x** :

```
[root@centos7 ~]# tcpdump -i enp0s3 -x
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:34:55.540011 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2208166651:2208166839, ack
23445916, win 40096, length 188
```

```
    0x0000:  4510 00e4 851c 4000 4006 9cd7 0a00 020f
    0x0010:  0a00 0202 0016 8e08 839d f2fb 0165 c19c
    0x0020:  5018 9ca0 18e7 0000 96d1 7d89 0022 fb31
    0x0030:  dec8 d6f4 1227 ceb3 4df9 1ed8 691b b0e4
    0x0040:  4561 6454 7f80 8928 1704 99a9 d4fe e565
    0x0050:  7d2e e55f eadc 7e0e 352c 2f65 dd2a ff02
    0x0060:  3a66 d1fa 3a15 9a4e 0054 91fe 7fe5 ce35
    0x0070:  7df4 371a 6363 1302 4037 2a7f dde1 3d76
    0x0080:  6ce5 069a 5855 cff6 18f5 dcf3 afff e525
    0x0090:  ec03 c13d c23c 9d4b e0b9 0661 4f59 cd11
    0x00a0:  cffb 1447 d6e6 7ab7 d7a8 d357 8fa6 ac7a
    0x00b0:  bcf5 257f 41dd 5064 ba0b 189b 1563 06fa
    0x00c0:  5364 9d6a 4f7c e99c 7a27 c3fc 3a2e 2618
    0x00d0:  1357 7e4c b62a 9e44 8350 71fa e9b8 a17f
    0x00e0:  a2be c731
01:34:55.540618 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 188, win 65535, length 0
    0x0000:  4500 0028 061f 0000 4006 5ca1 0a00 0202
    0x0010:  0a00 020f 8e08 0016 0165 c19c 839d f3b7
    0x0020:  5010 ffff cf4e 0000 0000 0000 0000
^C01:34:55.543115 IP centos7.fenestros.loc.47568 > google-public-dns-a.google.com.domain: 37698+ PTR?
2.2.0.10.in-addr.arpa. (39)
    0x0000:  4500 0043 8ca3 4000 4011 91e8 0a00 020f
    0x0010:  0808 0808 b9d0 0035 002f 1c5f 9342 0100
    0x0020:  0001 0000 0000 0000 0132 0132 0130 0231
    0x0030:  3007 696e 2d61 6464 7204 6172 7061 0000
    0x0040:  0c00 01

3 packets captured
21 packets received by filter
0 packets dropped by kernel
```

L'option -X

Pour écouter sur une interface spécifique et voir le contenu en Hexadécimal et en ASCII, utilisez les options -i et **-X** :

```
[root@centos7 ~]# tcpdump -i enp0s3 -X
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:35:56.671522 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2208168643:2208168831, ack
23446048, win 40096, length 188
    0x0000:  4510 00e4 8522 4000 4006 9cd1 0a00 020f  E....."@.@.......
    0x0010:  0a00 0202 0016 8e08 839d fac3 0165 c220  .............e..
    0x0020:  5018 9ca0 18e7 0000 d8cc 9fe4 3055 351c  P...........0U5.
    0x0030:  96a0 b2c0 650e 44e7 0016 b72e c990 4929  ....e.D.......I)
    0x0040:  0d05 e79b ef5b ccef aafc 607b b9a2 a714  .....[....`{....
    0x0050:  e1a0 36a1 b5f3 4a0a 5cdd 90bd 96cf b75d  ..6...J.\......]
    0x0060:  efa5 cc0b d195 d69a e933 48f2 74a5 ca6b  .........3H.t..k
    0x0070:  c803 fe4e 5832 2253 d6d9 178b c63a df8e  ...NX2"S.....:..
    0x0080:  62a3 6e67 91f8 cc50 f330 51da a285 1c3a  b.ng...P.0Q....:
    0x0090:  b074 a98f 95d6 f528 03e2 308a 31e8 5aa2  .t.....(..0.1.Z.
    0x00a0:  988b e3b1 444e ab24 e1b0 3eb8 ffa7 ae3f  ....DN.$..>....?
    0x00b0:  332f 221d 3bec d17a 1c8f 741a eaaf 5444  3/".;..z..t...TD
    0x00c0:  ea38 320e ea7f e9d2 035a 2531 e8f1 0757  .82......Z%1...W
    0x00d0:  0259 70b1 464b 51ea 9e6b 1b93 48ea 6f25  .Yp.FKQ..k..H.o%
    0x00e0:  d918 9027                                ...'
01:35:56.672680 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 188, win 65535, length 0
    0x0000:  4500 0028 06cc 0000 4006 5bf4 0a00 0202  E..(....@.[.....
    0x0010:  0a00 020f 8e08 0016 0165 c220 839d fb7f  .........e......
    0x0020:  5010 ffff c702 0000 0000 0000 0000       P.............
^C01:35:56.674310 IP centos7.fenestros.loc.57986 > google-public-dns-a.google.com.domain: 33529+ PTR?
2.2.0.10.in-addr.arpa. (39)
    0x0000:  4500 0043 188e 4000 4011 05fe 0a00 020f  E..C..@.@.......
    0x0010:  0808 0808 e282 0035 002f 1c5f 82f9 0100  .......5./._....
    0x0020:  0001 0000 0000 0000 0132 0132 0130 0231  .........2.2.0.1
    0x0030:  3007 696e 2d61 6464 7204 6172 7061 0000  0.in-addr.arpa..
    0x0040:  0c00 01                                  ...
```

```
3 packets captured
13 packets received by filter
0 packets dropped by kernel
```

L'option -w

Pour écouter sur une interface spécifique et envoyer la sortie dans un fichier, utilisez les options -i et **-w** :

```
[root@centos7 ~]# tcpdump -i enp0s3 -w log.dump
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C45 packets captured
45 packets received by filter
0 packets dropped by kernel

[root@centos7 ~]# ls -l log.dump
-rw-r--r--. 1 tcpdump tcpdump 8685 Jun  9 01:37 log.dump
```

Important - Pour générer le trafic, réactualisez simplement cette page web. Arrêtez la sortie de la commande à l'aide des touches **^C**.

Notez que le fichier log.dump est au format **libpcap** et non au format texte. Il est donc inutile d'essayer de lire son contenu :

```
[root@centos7 ~]# file log.dump
log.dump: tcpdump capture file (little-endian) - version 2.4 (Ethernet, capture length 262144)
```

L'option -v

Tcpdump peut être utilisé avec un de trois modes verbose.

Mode	Option
Light verbose	-v
Medium verbose	-vv
Full verbose	-vvv

```
[root@centos7 ~]# tcpdump -i enp0s3 -v
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:39:13.094781 IP (tos 0x10, ttl 64, id 39762, offset 0, flags [DF], proto TCP (6), length 164)
^C    centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], cksum 0x18a7 (incorrect -> 0xd270), seq
2210981047:2210981171, ack 23448932, win 40096, length 124

1 packet captured
11 packets received by filter
0 packets dropped by kernel
```

Filtrage à l'écoute

Tcpdump peut effectuer du filtrage lors de l'écoute.

Pour uniquement écouter les paquets en provenance de l'adresse IP 192.168.1.11, utilisez la condition **src host** :

```
# tcpdump src host 192.168.1.11 [Entrée]
```

Pour uniquement écouter les paquets en provenance de l'adresse IP 192.168.1.11 et vers l'adresse 192.168.1.2, utilisez les conditions src host et **dst host** :

```
# tcpdump src host 192.168.1.11 and dst host 192.168.1.2 [Entrée]
```

Pour uniquement écouter les paquets d'un port précis, utilisez la condition **port** :

```
# tcpdump -i eth0 port 80 [Entrée]
```

Pour uniquement écouter les paquets d'un protocole précis, utilisez une condition telle **ip**, **icmp**, **arp**, **rarp**, **udp** ou **tcp**:

```
# tcpdump -i eth0 udp [Entrée]
```

Pour uniquement écouter les paquets d'une taille inférieure à 100 octets, utilisez la condition **less** :

```
# tcpdump -i eth0 less 100 [Entrée]
```

Pour uniquement écouter les paquets d'une taille supérieure à 100 octets, utilisez la condition **great** :

```
# tcpdump -i eth0 greater 100 [Entrée]
```

L'utilisation des ses options et conditions peut être combinée pour donner des commandes telles :

```
# tcpdump -i eth0 -X src host 192.168.1.11 and dst host 192.168.1.2 and port 21 and ftp [Entrée]
```

Les Contre-Mesures

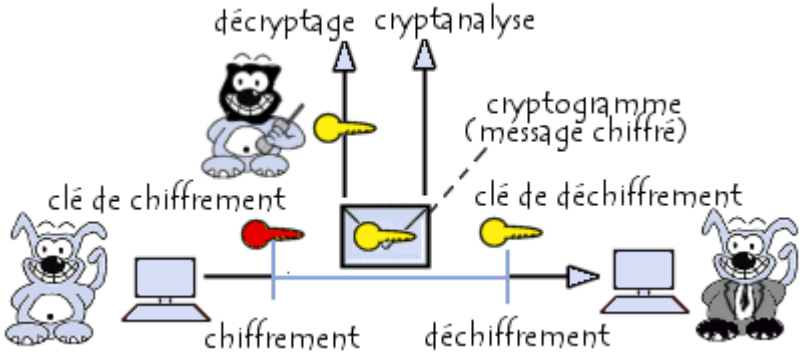
Les contre-mesures incluent l'utilisation du chiffrement sous la forme de SSH, de SCP et d'OpenVPN.

Introduction à la cryptologie

Définitions

- **La Cryptologie**
  - La science qui étudie les aspects scientifiques de ces techniques, c'est-à-dire qu'elle englobe la cryptographie et la cryptanalyse.
- **La Cryptanalyse**
  - Lorsque la clef de déchiffrement n'est pas connue de l'attaquant on parle alors de cryptanalyse ou cryptoanalyse (on entend souvent aussi le terme plus familier de cassage).
- **La Cryptographie**

    - Un terme générique désignant l'ensemble des techniques permettant de chiffrer des messages, c'est-à-dire permettant de les rendre inintelligibles sans une action spécifique. Les verbes crypter et chiffrer sont utilisés.
- **Le Décryptement ou Décryptage**
    - Est le fait d'essayer de déchiffrer illégitimement le message (que la clé de déchiffrement soit connue ou non de l'attaquant).



La Cryptographie

La cryptographie apporte quatre points clefs:

- La confidentialité
    - consiste à rendre l'information inintelligible à d'autres personnes que les acteurs de la transaction.
- L'intégrité
    - consiste à déterminer si les données n'ont pas été altérées durant la communication (de manière fortuite ou intentionnelle).
- L'authentification
    - consiste à assurer l'identité d'un utilisateur.
- La non-répudiation
    - est la garantie qu'aucun des correspondants ne pourra nier la transaction.

La cryptographie est basée sur l'arithmétique. Il s'agit, dans le cas d'un texte, de transformer les lettres qui composent le message en une succession de chiffres (sous forme de bits dans le cas de l'informatique), puis ensuite de faire des calculs sur ces chiffres pour:

- Procéder au chiffrement
    - Le résultat de cette modification (le message chiffré) est appelé cryptogramme (Ciphertext) par opposition au message initial, appelé message en clair (Plaintext)
- Procéder au déchiffrement

Le chiffrement se fait à l'aide d'une clef de chiffrement. Le déchiffrement nécessite une clef de déchiffrement.

On distingue deux types de clefs:

- Les clés symétriques:
  - des clés utilisées pour le chiffrement ainsi que pour le déchiffrement. On parle alors de chiffrement symétrique ou de chiffrement à clé secrète.
- Les clés asymétriques:
  - des clés utilisées dans le cas du chiffrement asymétrique (aussi appelé chiffrement à clé publique). Dans ce cas, une clé différente est utilisée pour le chiffrement et pour le déchiffrement.

Le Chiffrement par Substitution

Le chiffrement par substitution consiste à remplacer dans un message une ou plusieurs entités (généralement des lettres) par une ou plusieurs autres entités. On distingue généralement plusieurs types de cryptosystèmes par substitution :

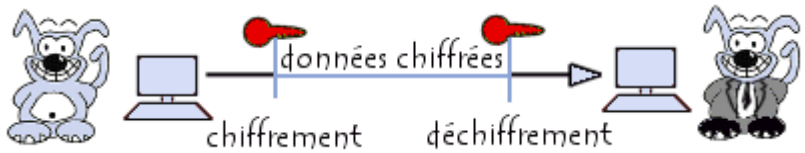
- La substitution **monoalphabétique**
  - consiste à remplacer chaque lettre du message par une autre lettre de l'alphabet
- La substitution **polyalphabétique**
  - consiste à utiliser une suite de chiffres monoalphabétique réutilisée périodiquement
- La substitution **homophonique**
  - permet de faire correspondre à chaque lettre du message en clair un ensemble possible d'autres caractères
- La substitution de **polygrammes**
  - consiste à substituer un groupe de caractères (polygramme) dans le message par un autre groupe de caractères

Algorithmes à clé secrète

Le Chiffrement Symétrique

Ce système est aussi appelé le système à **Clef Secrète** ou à **clef privée**.

Ce système consiste à effectuer une opération de chiffrement par algorithme mais comporte un inconvénient, à savoir qu'il nécessite un canal sécurisé pour la transmission de la clef de chiffrement/déchiffrement.



Important - Le système de Méthode du Masque Jetable (One Time Pad) fût mis au point dans les années 1920. Il utilisait une clef générée aléatoirement à usage unique.

Les algorithmes de chiffrement symétrique couramment utilisés en informatique sont:

- **Data Encryption Standard** (DES),
- **Triple DES** (3DES),
- **RC2**,
- **Blowfish**,
- **International Data Encryption Algorithm** (IDEA),
- **Advanced Encryption Standard** (AES).

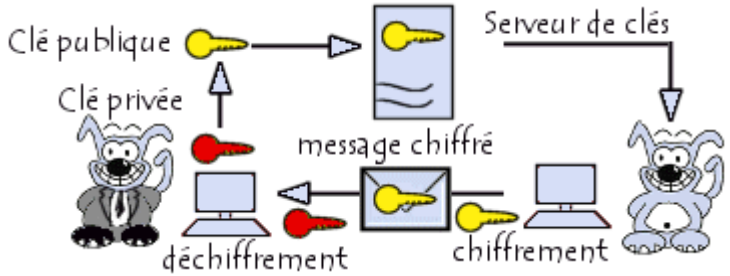
Algorithmes à clef publique

Le Chiffrement Asymétrique

Ce système est aussi appelé **Système à Clef Publique**.

Ce système consiste à avoir deux clefs appelées des **bi-clefs**:

- Une clef **publique** pour le chiffrement
- Une clef **secrète** ou **privée** pour le déchiffrement



- L'utilisateur A (celui qui déchiffre) choisit une clef privée.
- A partir de cette clef il génère plusieurs clefs publiques grâce à un algorithme.
- L'utilisateur B (celui qui chiffre) choisit une des clefs publiques à travers un canal non-sécurisé pour chiffrer les données à l'attention de l'utilisateur A.

Ce système est basé sur ce que l'on appelle une **fonction à trappe à sens unique** ou **one-way trap door**.

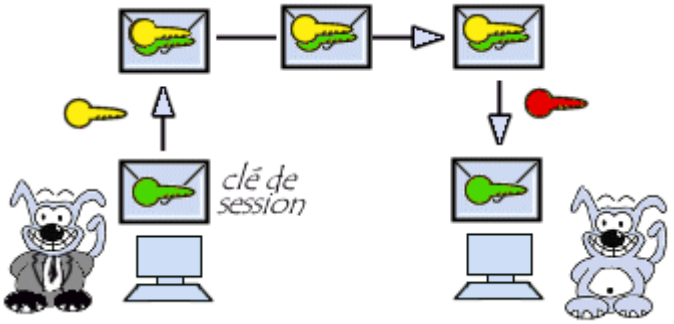
Il existe toutefois un problème – s'assurer que la clef publique récupérée est bien celle qui correspond au destinataire !

Les algorithmes de chiffrement asymétrique couramment utilisés en informatique sont:

- **Digital Signature Algorithm** (DSA)
- **Rivest, Shamir, Adleman** (RSA)

La Clef de Session

Ce système est un compromis entre le système symétrique et le système asymétrique. Il permet l'envoie de données chiffrées à l'aide d'un algorithme de chiffrement symétrique par un canal non-sécurisé et a été mis au point pour palier au problème de lenteur de déchiffrement du système asymétrique.

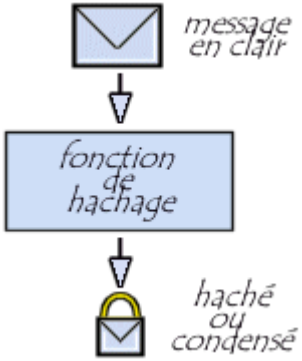


Ce système fonctionne de la façon suivante :

- L'utilisateur A chiffre une clef privée générée aléatoirement, appelée une « clef de session », en utilisant une des clefs publiques de l'utilisateur B.
- L'utilisateur A chiffre les données avec la clef de session.
- L'utilisateur B déchiffre la clef de session en utilisant sa propre clef privée.
- L'utilisateur B déchiffre les données en utilisant la clef de session.

Fonctions de Hachage

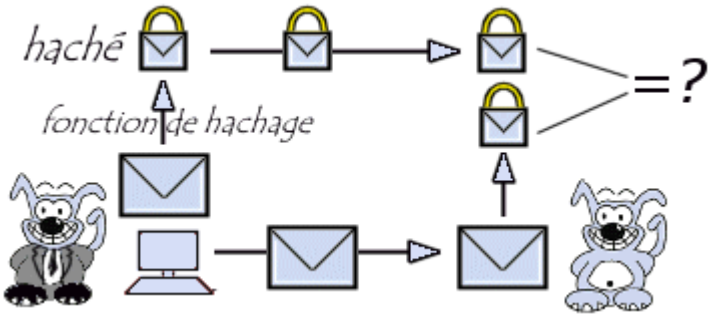
La fonction de **hachage**, aussi appelée une fonction de **condensation**, est à **sens unique** (one way function). Il « condense » un message en clair et produit un haché unique.



Les deux algorithmes de hachage utilisés sont:

- **Message Digest 5** (MD5)
- **Secure Hash Algorithm** (SHA)

Lors de son envoie, le message est accompagné de son haché et il est donc possible de garantir son intégrité:

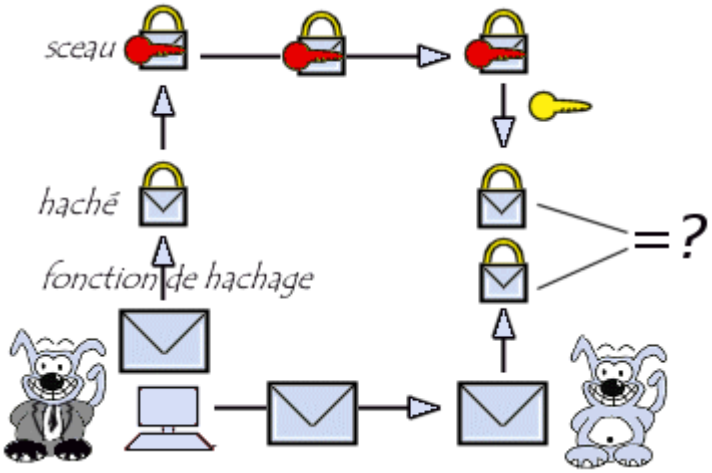


- A la réception du message, le destinataire ou l'utilisateur B calcule le haché du message reçu et le compare avec le haché accompagnant le document.
- Si le message ou le haché a été falsifié durant la communication, les deux empreintes ne correspondront pas.

> **Important** - Ce système permet de vérifier que l'empreinte correspond bien au message reçu, mais ne permet pas de prouver que le message a bien été envoyé par l'utilisateur A.

Signature Numérique

Pour garantir l'authentification du message l'utilisateur A va chiffrer ou **signer** le haché à l'aide de sa clé privée. Le haché signé est appelé un **sceau**.



- L’utilisateur A envoie le sceau au destinataire.
- A la réception du message L’utilisateur B déchiffre le sceau avec la clé publique de l’utilisateur A.
- Il compare le haché obtenu au haché reçu en pièce jointe.

Ce mécanisme de création de sceau est appelé **scellement**.

Ce mécanisme est identique au procédé utilisé par SSH lors d'une connexion

PKI

On appelle **PKI** (Public Key Infrastucture, ou en français **infrastructure à clé publique (ICP)**, parfois **infrastructure de gestion de clés (IGC)**) l’ensemble des solutions techniques basées sur la cryptographie à clé publique.

Les cryptosystèmes à clés publiques permettent de s'affranchir de la nécessité d'avoir recours systématiquement à un canal sécurisé pour s'échanger les clés. En revanche, la publication de la clé publique à grande échelle doit se faire en toute confiance pour assurer que :

- La clé publique est bien celle de son propriétaire ;
- Le propriétaire de la clé est digne de confiance ;
- La clé est toujours valide.

Ainsi, il est nécessaire d'associer au bi-clé (ensemble clé publique / clé privée) un certificat délivré par un **tiers de confiance** : l'infrastructure de gestion de clés.

Le tiers de confiance est une entité appelée communément autorité de certification (ou en anglais Certification authority, abrégé CA) chargée d'assurer la véracité des informations contenues dans le certificat de clé publique et de sa validité.

Pour ce faire, l'autorité signe le certificat de clé publique à l'aide de sa propre clé en utilisant le principe de signature numérique.

Le rôle de l'infrastructure de clés publiques est multiple et couvre notamment les champs suivants :

- enregistrer des demandes de clés en vérifiant l'identité des demandeurs ;
- générer les paires de clés (clé privée / clé publique) ;
- garantir la confidentialité des clés privées correspondant aux clés publiques ;
- certifier l'association entre chaque utilisateurs et sa clé publique ;
- révoquer des clés (en cas de perte par son propriétaire, d'expiration de sa date de validité ou de compromission).

Une infrastructure à clé publique est en règle générale composée de trois entités distinctes :

- L'autorité d'enregistrement (AE ou RA pour Recording authority), chargée des formalité administratives telles que la vérification de l'identité des demandeurs, le suivi et la gestion des demandes, etc.) ;
- L'autorité de certification (AC ou CA pour Certification Authority), chargée des tâches techniques de création de certificats. L'autorité de certification est ainsi chargée de la signature des demandes de certificat (CSR pour Certificate Signing Request, parfois appelées PKCS#10, nom du format correspondant). L'autorité de certification a également pour mission la signature des listes de révocations (CRL pour Certificate Revocation List) ;
- L'Autorité de dépôt (Repository) dont la mission est de conserver en sécurité les certificats.

Certificats X509

Pour palier aux problèmes liés à des clefs publiques piratées, un système de certificats a été mis en place.

Le certificat permet d’associer la clef publique à une entité ou une personne. Les certificats sont délivrés par des Organismes de Certification.

Les certificats sont des fichiers divisés en deux parties :

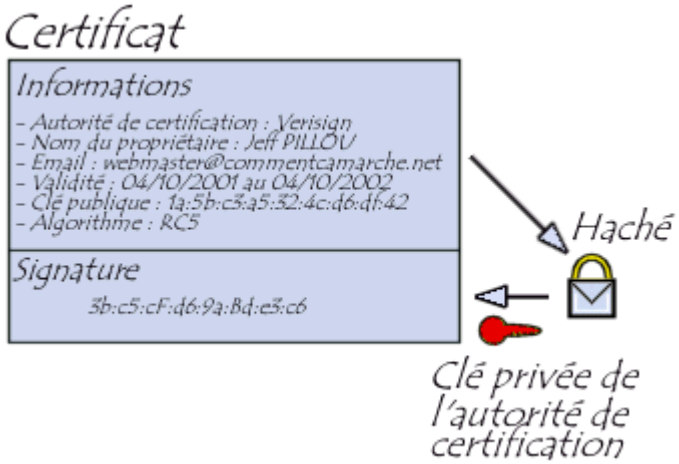
- La partie contenant les informations
- La partie contenant la signature de l'autorité de certification

La structure des certificats est normalisée par le standard **X.509** de l'**Union internationale des télécommunications**.

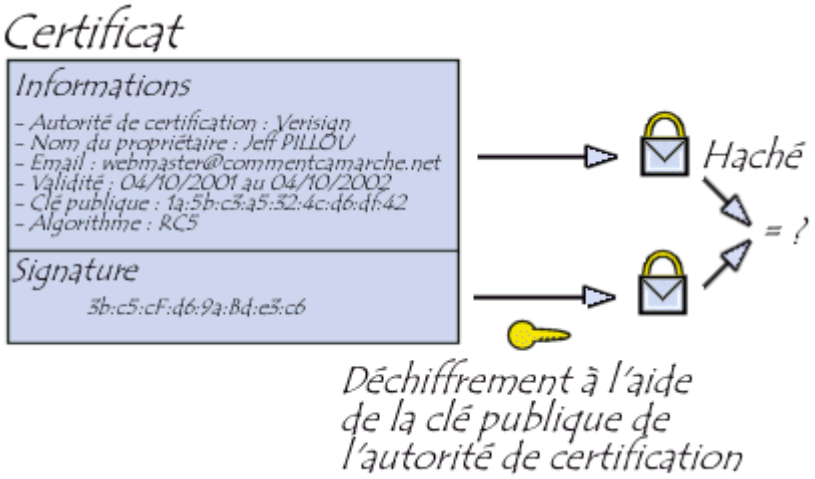
Elle contient :

- Le nom de l'autorité de certification
- Le nom du propriétaire du certificat
- La date de validité du certificat
- L'algorithme de chiffrement utilisé
- La clé publique du propriétaire

Le Certificat est signé par l'autorité de certification:



La vérification se passe ainsi:



LAB #2 - Utilisation de GnuPG

Présentation

GNU Privacy Guard permet aux utilisateurs de transférer des messages chiffrés et/ou signés.

Installation

Sous RHEL/CentOS 7, le paquet gnupg peut être installé de base :

```
[root@centos7 ~]# whereis gpg
gpg: /usr/bin/gpg /usr/share/man/man1/gpg.1.gz
```

Sinon il convient de l'installer.

Utilisation

Pour initialiser GnuPG, saisissez la commande suivante :

```
[root@centos7 ~]# gpg
gpg: directory `/root/.gnupg' created
gpg: new configuration file `/root/.gnupg/gpg.conf' created
gpg: WARNING: options in `/root/.gnupg/gpg.conf' are not yet active during this run
gpg: keyring `/root/.gnupg/secring.gpg' created
gpg: keyring `/root/.gnupg/pubring.gpg' created
gpg: Go ahead and type your message ...
^C
gpg: signal Interrupt caught ... exiting
```

Pour générer les clefs, saisissez la commande suivante :

```
[root@centos7 ~]# gpg --gen-key
gpg (GnuPG) 2.0.22; Copyright (C) 2013 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
```

```
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0)
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: I2TCH
Email address: infos@i2tch.eu
Comment: Test key
You selected this USER-ID:
    "I2TCH (Test key) <infos@i2tch.eu>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.

We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key 3B5BBC1B marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   2  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 2u
pub   2048R/3B5BBC1B 2018-06-09
      Key fingerprint = F3F9 D48D A422 6373 E105  548D C60A 69D9 3B5B BC1B
```

```
uid                  I2TCH (Test key) <infos@i2tch.eu>
sub   2048R/954BFAE5 2018-06-09
```

La liste de clefs peut être visualisée avec la commande suivante :

```
[root@centos7 ~]# gpg --list-keys
/root/.gnupg/pubring.gpg
------------------------
pub   2048R/3B5BBC1B 2018-06-09
uid                  I2TCH (Test key) <infos@i2tch.eu>
sub   2048R/954BFAE5 2018-06-09
```

Important - Indiquez la passphrase **fenestros**.

Pour importer la clef d'un correspondant dans sa trousse de clefs il convient d'utiliser la commande suivante :

```
# gpg --import la-clef.asc
```

Pour exporter sa clef publique, il convient d'utiliser la commande suivante :

```
[root@centos7 ~]# gpg --export --armor I2TCH > ~/mykey.asc
[root@centos7 ~]# cat mykey.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQENBFsbJT0BCAC/dUhSAq0ETWDOII+rN2rLHs3JDC/LRk92/svf7r1C7gqFiUzo
PTqX5Bi+EiYI5pLk+hLEQUGm/H1fLGi4Q5mqekI0QSrUo8kxz1d8H6y8bsksGecI
U/8N1WJZnVdzSdSHqPv0ow23U75yC/iuX6teMHBx0xOIANpfw/OhnaKs5eDkhTlU
TZAaSzkfRVssVyaGrSCZmsypeNUaa+oWIvdHjPSBiY5JeV69tuYPoP67sgTA+CE5
VBHDCuYEn+Kkvz+JS4xvQMj/cvBUywr9JJfihCB8yUeiOaQjTQjqonCjROJ712R7
LsR8q/5/ac7m7EjbaBJxr7jx+Jp0F0v2hS87ABEBAAG0IUkyVENIIChUZXN0IGtl
```

```
eSkgPGluZm9zQGkydGNoLmV1PokBOQQTAQIAIwUCWxslPQIbAwcLCQgHAwIBBhUI
AgkKCwQWAgMBAh4BAheAAAoJEMYKadk7W7wb2E0IAI0773sLKjXz+QjWOSUWb3nS
oYD7m0VgrWTbu2fO8MSYID6Eixo+xOhPQ35DiWGxrCn6TEYB2fOvmbr+7N/l7MHj
ELKVGJFmHpk61J53kACDWwxYo1RSCdp8Cii/6JZdSkmCO1e5TxsSV9k6SW4iefF3
T/1cUIi3bqL+S4iq/zC/bMTTGJOGsQWGBaSkTP9Whgm4DVLlSmZFV/SH87EzfW77
eDw50g3hxkBG8q0n9pe5Xztxh/7Y/kuLHRwHCc+V+7159rjr6TzsaOB0hMrMIQuX
nIOHJ3LLbiCw6vA3BfBUDTA1Bf/0VOM9PXsxlpn/MrJSJnbV2KPiYD5LZvvhyRi5
AQ0EWxslPQEIALpz1NN7gEiZH1yBz1BzhZNjr6Fvz6kNmfL85+HoidU8ZbXRXwYK
Yxm9jNBi4rvqd6eOFHuBrBCJgLqK28VmEPFGw1EONjWLBqz3jSOhepc6A0KAlOLf
4hUaTds7t6lqHfWzwj8RQTBsUOjABKpJKWeD1PYygQAt3MfSY9xoH7EN5rt8dOIj
qLw3o29mfFwNBlJDdUdxAxrDFyLFcqS53xjVifPmTFqOz7yRm8Nrq6QCVaWM6tGo
c5WgxxyFR9Ki8tsj14OLEq2tCyNe80C0CVw2qHL+RUBt3+lLo/Zg3uBWST6C2kSn
uIzdfPonAR6YA6GmeAfsuA+8tdnxwZ8claUAEQEAAYkBHwQYAQIACQUCWxslPQIb
DAAKCRDGCmnZO1u8G80mB/0X+hRRz2KL7wESlBgb1mnswagfNh2/oxUD0KYaA6nk
E89jWlsoH+UqOSQ9tGhKJdZEMIRvjnN/crWc4L/T521sBDKeQBAX+qCpQTBxub56
8ey70mU+0qvjj6fPdzEJB3qdXrRSG4dvsGIJ7clKQCYdGwMnc0UZnqsFYskoOBXa
QUTb2FaHdVDXd088xzB0/30VjczpC05OoyAAp68pHz8RIAm1sTtsSPzMujofJF29
G8QpZ08yieZBT7QZDK9f2KSqP1Ee/GgarkwYgQ7hmDAzdRu2RMe6pF2CxBnNnKe9
OHBQuwotU9EdNDChHkrQ9hfCzJoPNMzNdKmgl1sbGu8p
=VNCb
-----END PGP PUBLIC KEY BLOCK-----
```

Cette clef peut ensuite être jointe à des messages électroniques ou bien déposée sur un serveur de clefs tel http://www.keyserver.net.

Signer un message

Créez maintenant un message à signer :

```
[root@centos7 ~]# vi message.txt
[root@centos7 ~]# cat message.txt
# ~/message.txt
Ceci est un message de test pour GnuPG
```

Pour signer ce message en format binaire, il convient d'utiliser la commande suivante :

```
[root@centos7 ~]# gpg --default-key I2TCH --detach-sign message.txt

You need a passphrase to unlock the secret key for
user: "I2TCH (Test key) <infos@i2tch.eu>"
2048-bit RSA key, ID 3B5BBC1B, created 2018-06-09

[root@centos7 ~]# ls -l | grep message
-rw-r--r--. 1 root    root          55 Jun  9 03:03 message.txt
-rw-r--r--. 1 root    root         287 Jun  9 03:05 message.txt.sig
[root@centos7 ~]# cat message.txt.sig
[illegible]
[illegible]
[illegible]
```

```
[root@centos7 ~]# cat message.txt.asc
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2.0.22 (GNU/Linux)

iQEcBAABAgAGBQJbGyg8AAoJEMYKadk7W7wbgMEIAJBB86wQS4hSDj5w1u9wFiQQ
NM9QfO77WDc5r5NAYcXWWfOm1LtNNqMvoeM6Q0STf6IjSOgc4SPFsD1Uv+UKQ5GN
GnpQHmg3s8otPXBvreRMP9eDbLzpaf0OhA2C+5zkiRzIYfJq3EGJ32d3UdxtrOkh
ID8G3xVvaNE2b0+UZvegdGiDDcISZjJVzWC/aaIWC8dp4OAqSm9cCRqD3FWIfa0n
TiyK1m274CT3iclp9ckjWyebNvsJhn3zCMk9mQb9bK5bsi0ygwTdkomuLEAbvtiD
iDBDiCw6pKs9Xkxv4qLCyZhGqZ6qgFB8rw9dwYN7QGwqnL9uueal0l1ErpcWdsM=
=4ebo
-----END PGP SIGNATURE-----
[root@centos7 ~]#
```

Pour signer ce message **dans le message lui-même** en format ascii, il convient d'utiliser la commande suivante :

```
[root@centos7 ~]# gpg --default-key I2TCH --clearsign message.txt

You need a passphrase to unlock the secret key for
user: "I2TCH (Test key) <infos@i2tch.eu>"
2048-bit RSA key, ID 3B5BBC1B, created 2018-06-09

File `message.txt.asc' exists. Overwrite? (y/N) y

[root@centos7 ~]# ls -l | grep message
-rw-r--r--. 1 root    root          55 Jun  9 03:03 message.txt
-rw-r--r--. 1 root    root         592 Jun  9 03:10 message.txt.asc
-rw-r--r--. 1 root    root         287 Jun  9 03:05 message.txt.sig

[root@redhat ~]# cat message.txt.asc
[root@centos7 ~]# cat message.txt.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1
```

```
# ~/message.txt
Ceci est un message de test pour GnuPG
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2.0.22 (GNU/Linux)

iQEcBAEBAgAGBQJbGykZAAoJEMYKadk7W7wbgZ0H/AmXUCY4pNxxvVcE2WLmH2n/
TpkGsckyy/nulasN7+Gr4J//ja5ZkKDb3YVxz0RqirkBwtWPmaSCzjR3VeKmaZ36
a4mR26ySnSDTPQ/ezY7RAKoV90PP+1slDKjQQBssSk9TPMO1jqoU3TqN2e0ZUiBC
G5eQSV6+E2+qCmMPm0BddRkW8oSptaY+iKJUbMpiRnfY5yDOSo02cpSh3grMgZzC
eRQ6kP3EcyqS8tbxreLu7hvYvliAxu8epDvZkFNpTCLlLYOKjdLoxeEwXstxl7S3
Am6p5sVsrZTj9aU7n2N4FyL83bYPhbQmmzVxmIZPZXlkgRlVsg2gr/dyJFACr6A=
=TYcP
-----END PGP SIGNATURE-----
```

Pour vérifier la signature d'un message signé en mode ascii, il convient d'utiliser la commande :

```
[root@centos7 ~]# gpg --verify message.txt.asc
gpg: Signature made Sat 09 Jun 2018 03:10:49 CEST using RSA key ID 3B5BBC1B
gpg: Good signature from "I2TCH (Test key) <infos@i2tch.eu>"
```

Important - Pour vérifier la signature d'un message signé en mode ascii et produit en dehors du message lui-même, il convient d'utiliser la commande :

```
# gpg --verify message.txt.asc message.txt
```

Chiffrer un message

Pour chiffrer un message, il faut disposer de la clef publique du destinataire du message. Ce dernier utilisera ensuite sa clef privée pour déchiffrer le message. Il convient de préciser le destinataire du message, ou plus précisément la clef publique à utiliser, lors d'un chiffrement :

```
gpg --recipient <destinataire> --encrypt <message>
```

- <*destinataire*> représente toute information permettant de distinguer sans ambigüité une clef publique dans votre trousseau. Cette information peut-être le nom ou l'adresse email associé à la clef publique que vous voulez utiliser,
- <*message*> représente le message à chiffrer.

Par exemple pour chiffrer un message en mode binaire, il convient de saisir la commande suivante :

```
[root@centos7 ~]# gpg --recipient I2TCH --encrypt message.txt
[root@centos7 ~]# ls -l | grep message
-rw-r--r--. 1 root    root          55 Jun  9 03:03 message.txt
-rw-r--r--. 1 root    root         592 Jun  9 03:10 message.txt.asc
-rw-r--r--. 1 root    root         382 Jun  9 03:14 message.txt.gpg
-rw-r--r--. 1 root    root         287 Jun  9 03:05 message.txt.sig
[root@centos7 ~]# cat message.txt.gpg
[illegible]
```

Et pour chiffrer un message en mode ascii, il convient de saisir la commande suivante :

```
# gpg --recipient I2TCH --armor --encrypt message.txt [Entrée]
```

```
[root@centos7 ~]# gpg --recipient I2TCH --armor --encrypt message.txt
File `message.txt.asc' exists. Overwrite? (y/N) y
```

```
[root@centos7 ~]# ls -l | grep message
-rw-r--r--. 1 root    root          55 Jun  9 03:03 message.txt
-rw-r--r--. 1 root    root         616 Jun  9 03:17 message.txt.asc
-rw-r--r--. 1 root    root         382 Jun  9 03:14 message.txt.gpg
-rw-r--r--. 1 root    root         287 Jun  9 03:05 message.txt.sig
[root@centos7 ~]# cat message.txt.asc
-----BEGIN PGP MESSAGE-----
Version: GnuPG v2.0.22 (GNU/Linux)

hQEMA+6qJB+VS/rlAQf+IHOy5iOoTjfcOrfvfRVCK35igYBePZYwjD5Mop6LxVem
J7AomE7zXNl+698LE66W7S5QnBj21SCKsnyr5Aq64gV+0bKquyiRxYehLaAhHL8o
/FzQSqsFoFFrL8RtzjfpWuO7M2pBuQffcpBjMEIGNpiTTdWxdyvRJnbNmb22qljQ
7QPlV0GpgzBUTG3EbtOWJcmTr+AYC0iPTTP/2z1PEL/7dXF+47K5eVaijgXHBT/X
i5a2kOU3Fwupjst0Os3gUq0XxPvfriJ9I45Fk5KO+9Egmia3wlq019XNa2IUE2lf
uYpK9EL4OaDcHGYHqvhJrn7g4lMcLlBUYVrluVaAPtJtAWlJqO/1MBOMHa6h9bTB
wZ5uRzsj4dVPX7vJHPW9fDfK5FOkIZHirlEgOLLcuAXnkvX2NQ8BuQcyg6ZRzHTj
UMHDk4fJfwUtdcY8eWtFTLQrn8iiW5MBWMH2qcaRnd83ieH/wqdPYdzolWLcDA==
=G95a
-----END PGP MESSAGE-----
```

Pour décrypter un message il convient d'utiliser la commande suivante :

```
[root@centos7 ~]# gpg --decrypt message.txt.asc

You need a passphrase to unlock the secret key for
user: "I2TCH (Test key) <infos@i2tch.eu>"
2048-bit RSA key, ID 954BFAE5, created 2018-06-09 (main key ID 3B5BBC1B)

gpg: encrypted with 2048-bit RSA key, ID 954BFAE5, created 2018-06-09
      "I2TCH (Test key) <infos@i2tch.eu>"
# ~/message.txt
Ceci est un message de test pour GnuPG
```

LAB #3 - Mise en place de SSH et SCP

Introduction

La commande **ssh** est le successeur et la remplaçante de la commande **rlogin**. Il permet d'établir des connexions sécurisées avec une machine distante. SSH comporte cinq acteurs :

- Le **serveur SSH**
  - le démon sshd, qui s'occupe des authentifications et autorisations des clients,
- Le **client SSH**
  - ssh ou scp, qui assure la connexion et le dialogue avec le serveur,
- La **session** qui représente la connexion courante et qui commence juste après l'authentification réussie,
- Les **clefs**
  - **Couple de clef utilisateur asymétriques** et persistantes qui assurent l'identité d'un utilisateur et qui sont stockés sur disque dur,
  - **Clef hôte asymétrique et persistante** garantissant l'identité du serveur er qui est conservé sur disque dur
  - **Clef serveur asymétrique et temporaire** utilisée par le protocole SSH1 qui sert au chiffrement de la clé de session,
  - **Clef de session symétrique qui est générée aléatoirement** et qui permet le chiiffrement de la communication entre le client et le serveur. Elle est détruite en fin de session. SSH-1 utilise une seule clef tandis que SSH-2 utilise une clef par direction de la communication,
- La **base de données des hôtes connus** qui stocke les clés des connexions précédentes.

SSH fonctionne de la manière suivante pour la la mise en place d'un canal sécurisé:

- Le client contacte le serveur sur son port 22,
- Les client et le serveur échangent leur version de SSH. En cas de non-compatibilité de versions, l'un des deux met fin au processus,
- Le serveur SSH s'identifie auprès du client en lui fournissant :
  - Sa clé hôte,
  - Sa clé serveur,
  - Une séquence aléatoire de huit octets à inclure dans les futures réponses du client,
  - Une liste de méthodes de chiffrage, compression et authentification,
- Le client et le serveur produisent un identifiant identique, un haché MD5 long de 128 bits contenant la clé hôte, la clé serveur et la séquence aléatoire,
- Le client génère sa clé de session symétrique et la chiffre deux fois de suite, une fois avec la clé hôte du serveur et la deuxième fois avec la clé serveur. Le client envoie cette clé au serveur accompagnée de la séquence aléatoire et un choix d'algorithmes supportés,

- Le serveur déchiffre la clé de session,
- Le client et le serveur mettent en place le canal sécurisé.

SSH-1

SSH-1 utilise une paire de clefs de type RSA1. Il assure l'intégrité des données par une **Contrôle de Redondance Cyclique** (CRC) et est un bloc dit **monolithique**.

Afin de s'identifier, le client essaie chacune des six méthodes suivantes :

- **Kerberos**,
- **Rhosts**,
- **RhostsRSA**,
- Par **clef asymétrique**,
- **TIS**,
- Par **mot de passe**.

SSH-2

SSH-2 utilise **DSA** ou **RSA**. Il assure l'intégrité des données par l'algorithme **HMAC**. SSH-2 est organisé en trois **couches** :

- **SSH-TRANS** – Transport Layer Protocol,
- **SSH-AUTH** – Authentification Protocol,
- **SSH-CONN** – Connection Protocol.

SSH-2 diffère de SSH-1 essentiellement dans la phase authentification.

Trois méthodes d'authentification :

- Par **clef asymétrique**,
  - Identique à SSH-1 sauf avec l'algorithme DSA,
- **RhostsRSA**,
- Par **mot de passe**.

Options de la commande

Les options de cette commande sont :

```
[root@centos7 ~]# ssh --help
unknown option -- -
usage: ssh [-1246AaCfGgKkMNnqsTtVvXxYy] [-b bind_address] [-c cipher_spec]
           [-D [bind_address:]port] [-E log_file] [-e escape_char]
           [-F configfile] [-I pkcs11] [-i identity_file]
           [-J [user@]host[:port]] [-L address] [-l login_name] [-m mac_spec]
           [-O ctl_cmd] [-o option] [-p port] [-Q query_option] [-R address]
           [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
           [user@]hostname [command]
```

L'authentification par mot de passe

L'utilisateur fournit un mot de passe au client ssh. Le client ssh le transmet de façon sécurisée au serveur ssh puis le serveur vérifie le mot de passe et l'accepte ou non.

Avantage:

- Aucune configuration de clef asymétrique n'est nécessaire.

Inconvénients:

- L'utilisateur doit fournir à chaque connexion un identifiant et un mot de passe,
- Moins sécurisé qu'un système par clef asymétrique.

L'authentification par clef asymétrique

- Le **client** envoie au serveur une requête d'authentification par clé asymétrique qui contient le module de la clé à utiliser,
- Le **serveur** recherche une correspondance pour ce module dans le fichier des clés autorisés **~/.ssh/authorized_keys**,

    - Dans le cas où une correspondance n'est pas trouvée, le serveur met fin à la communication,
    - Dans le cas contraire le serveur génère une chaîne aléatoire de 256 bits appelée un **challenge** et la chiffre avec la **clé publique du client**,
- Le **client** reçoit le challenge et le décrypte avec la partie privée de sa clé. Il combine le challenge avec l'identifiant de session et chiffre le résultat. Ensuite il envoie le résultat chiffré au serveur.
- Le **serveur** génère le même haché et le compare avec celui reçu du client. Si les deux hachés sont identiques, l'authentification est réussie.

Installation

Pour installer/mettre à jour le serveur **sshd**, utilisez **yum** :

```
[root@centos7 ~]# yum install openssh-server
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: centos.mirror.fr.planethoster.net
 * extras: ftp.ciril.fr
 * updates: centos.mirrors.ovh.net
Package openssh-server-6.6.1p1-25.el7_2.x86_64 already installed and latest version
Nothing to do
```

Important - Pour les stations de travail, installez le client : **openssh-clients**.

Options de la commande

Les options de la commande sont :

```
SYNOPSIS
     sshd [-46DdeiqTt] [-b bits] [-C connection_spec] [-f config_file] [-g login_grace_time] [-h host_key_file]
```

```
[-k key_gen_time] [-o option] [-p port]  [-u len]
```

Configuration

Serveur

La configuration du serveur s'effectue dans le fichier **/etc/ssh/sshd_config** :

```
[root@centos7 ~]# cat /etc/ssh/sshd_config
#   $OpenBSD: sshd_config,v 1.93 2014/01/10 05:59:19 djm Exp $

# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

# The default requires explicit activation of protocol 1
#Protocol 2
```

```
# HostKey for protocol version 1
#HostKey /etc/ssh/ssh_host_key
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key

# Lifetime and size of ephemeral version 1 server key
#KeyRegenerationInterval 1h
#ServerKeyBits 1024

# Ciphers and keying
#RekeyLimit default none

# Logging
# obsoletes QuietMode and FascistLogging
#SyslogFacility AUTH
SyslogFacility AUTHPRIV
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#RSAAuthentication yes
#PubkeyAuthentication yes

# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
```

```
AuthorizedKeysFile  .ssh/authorized_keys

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#RhostsRSAAuthentication no
# similar for protocol version 2
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# RhostsRSAAuthentication and HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication yes
#PermitEmptyPasswords no
PasswordAuthentication yes

# Change to no to disable s/key passwords
#ChallengeResponseAuthentication yes
ChallengeResponseAuthentication no

# Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
#KerberosGetAFSToken no
#KerberosUseKuserok yes

# GSSAPI options
```

```
GSSAPIAuthentication yes
GSSAPICleanupCredentials no
#GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no
#GSSAPIEnablek5users no

# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
# WARNING: 'UsePAM no' is not supported in Red Hat Enterprise Linux and may cause several
# problems.
UsePAM yes

#AllowAgentForwarding yes
#AllowTcpForwarding yes
#GatewayPorts no
X11Forwarding yes
#X11DisplayOffset 10
#X11UseLocalhost yes
#PermitTTY yes
#PrintMotd yes
#PrintLastLog yes
#TCPKeepAlive yes
#UseLogin no
UsePrivilegeSeparation sandbox      # Default for new installations.
#PermitUserEnvironment no
#Compression delayed
#ClientAliveInterval 0
```

```
#ClientAliveCountMax 3
#ShowPatchLevel no
#UseDNS yes
#PidFile /var/run/sshd.pid
#MaxStartups 10:30:100
#PermitTunnel no
#ChrootDirectory none
#VersionAddendum none

# no default banner path
#Banner none

# Accept locale-related environment variables
AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL LANGUAGE
AcceptEnv XMODIFIERS

# override default of no subsystems
Subsystem   sftp    /usr/libexec/openssh/sftp-server

# Example of overriding settings on a per-user basis
#Match User anoncvs
#   X11Forwarding no
#   AllowTcpForwarding no
#   PermitTTY no
#   ForceCommand cvs server
```

Pour ôter les lignes de commentaires dans ce fichier, utilisez la commande suivante :

```
[root@centos7 ~]# cd /tmp ; grep -E -v '^(#|$)'  /etc/ssh/sshd_config > sshd_config
[root@centos7 tmp]# cat sshd_config
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
```

```
HostKey /etc/ssh/ssh_host_ed25519_key
SyslogFacility AUTHPRIV
AuthorizedKeysFile  .ssh/authorized_keys
PasswordAuthentication yes
ChallengeResponseAuthentication no
GSSAPIAuthentication yes
GSSAPICleanupCredentials no
UsePAM yes
X11Forwarding yes
UsePrivilegeSeparation sandbox      # Default for new installations.
AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL LANGUAGE
AcceptEnv XMODIFIERS
Subsystem   sftp    /usr/libexec/openssh/sftp-server
```

Pour sécuriser le serveur ssh, ajoutez ou modifiez les directives suivantes :

```
AllowGroups adm
Banner /etc/issue.net
HostbasedAuthentication no
IgnoreRhosts yes
LoginGraceTime 60
LogLevel INFO
PermitEmptyPasswords no
PermitRootLogin no
PrintLastLog yes
Protocol 2
StrictModes yes
X11Forwarding no
```

Votre fichier ressemblera à celui-ci :

```
AllowGroups adm
```

```
Banner /etc/issue.net
HostbasedAuthentication no
IgnoreRhosts yes
LoginGraceTime 60
LogLevel INFO
PermitEmptyPasswords no
PermitRootLogin no
PrintLastLog yes
Protocol 2
StrictModes yes
X11Forwarding no
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
SyslogFacility AUTHPRIV
AuthorizedKeysFile  .ssh/authorized_keys
PasswordAuthentication yes
ChallengeResponseAuthentication no
GSSAPIAuthentication yes
GSSAPICleanupCredentials no
UsePAM yes
UsePrivilegeSeparation sandbox      # Default for new installations.
AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL LANGUAGE
AcceptEnv XMODIFIERS
Subsystem   sftp    /usr/libexec/openssh/sftp-server
```

A Faire - Renommez le fichier **/etc/ssh/sshd_config** en **/etc/ssh/sshd_config.old** puis copiez le fichier **/tmp/sshd_config** vers **/etc/sshd/**. Redémarrez ensuite le service sshd. N'oubliez pas de mettre l'utilisateur **trainee** dans le groupe **adm** !

Pour générer les clefs sur le serveur saisissez la commande suivante en tant que **root**:

Lors de la génération des clefs, la passphrase doit être **vide**.

```
[root@centos7 ~]# ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/root/.ssh/id_dsa): /etc/ssh/ssh_host_dsa_key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/ssh/ssh_host_dsa_key.
Your public key has been saved in /etc/ssh/ssh_host_dsa_key.pub.
The key fingerprint is:
d5:54:d3:30:1c:f5:da:f8:21:15:1f:c8:6c:3b:b1:ff root@centos7.fenestros.loc
The key's randomart image is:
+--[ DSA 1024]----+
|            +oBB.|
|           o *.o*|
|          . o +.o|
|         .   +.+ |
|        S    .=..|
|              .o.|
|               o|
|               E|
|                |
+-----------------+
```

Important - Le chemin à indiquer pour le fichier est **/etc/ssh/ssh_host_dsa_key**. De la même façon, il est possible de générer les clefs au format **RSA**, **ECDSA** et **ED25519**.

Les clefs publiques générées possèdent l'extension **.pub**. Les clefs privées n'ont pas d'extension :

```
[root@centos7 ~]# ls /etc/ssh
moduli      sshd_config       ssh_host_dsa_key.pub  ssh_host_ecdsa_key.pub  ssh_host_ed25519_key.pub
ssh_host_rsa_key.pub
ssh_config  ssh_host_dsa_key  ssh_host_ecdsa_key    ssh_host_ed25519_key    ssh_host_rsa_key
```

Re-démarrez ensuite le service sshd :

```
[root@centos7 ~]# systemctl restart sshd.service
```

Saisissez maintenant les commandes suivantes en tant que **trainee** :

Important - Lors de la génération des clefs, la passphrase doit être **vide**.

```
[trainee@centos7 ~]$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_dsa):
Created directory '/home/trainee/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_dsa.
Your public key has been saved in /home/trainee/.ssh/id_dsa.pub.
The key fingerprint is:
97:92:85:d1:ae:97:f7:64:d2:54:45:89:eb:57:b1:66 trainee@centos7.fenestros.loc
The key's randomart image is:
+--[ DSA 1024]----+
|        ..    ..=|
|         o.  . o.|
|        ...   ..o|
|         o.. ..E.|
|        S.o..oo .|
|        .oo o.+. |
```

```
|         . . =.  |
|              .  |
|                 |
+-----------------+
[trainee@centos7 ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_rsa.
Your public key has been saved in /home/trainee/.ssh/id_rsa.pub.
The key fingerprint is:
80:4c:5a:bf:d0:2f:d1:a1:34:7c:09:a1:9c:0d:ed:2d trainee@centos7.fenestros.loc
The key's randomart image is:
+--[ RSA 2048]----+
|    +o=o..       |
|   * Xo+o.       |
|  . B.Bo.        |
|     .E=.        |
|      o.S        |
|       .         |
|                 |
|                 |
|                 |
+-----------------+
[trainee@centos7 ~]$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_ecdsa.
Your public key has been saved in /home/trainee/.ssh/id_ecdsa.pub.
The key fingerprint is:
41:5d:64:cf:d6:4a:ce:8e:a9:a8:4a:62:04:57:09:fc trainee@centos7.fenestros.loc
```

```
The key's randomart image is:
+--[ECDSA  256]---+
| .....  .. o+    |
|  ...  .  .. o . |
|. ..    .     = .|
| o  E    .   = . |
|  .     S     +  |
| .           +   |
|  o .       o .  |
| . o     . .     |
|    ..... .      |
+-----------------+
[trainee@centos7 ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_ed25519.
Your public key has been saved in /home/trainee/.ssh/id_ed25519.pub.
The key fingerprint is:
66:3a:83:d1:6d:79:46:48:88:7c:d9:65:59:bb:e6:d0 trainee@centos7.fenestros.loc
The key's randomart image is:
+--[ED25519  256--+
|   . . +..oo.    |
|    o +..o.  .   |
|     .  . . .    |
|     . . o . .   |
|    . . S + E    |
|     o = o +     |
|    . +    .     |
|       o         |
|                 |
+-----------------+
```

Important - Les clés générées seront placées dans le répertoire **~/.ssh/**.

Utilisation

La commande ssh prend la forme suivante:

```
ssh -l nom_de_compte numero_ip (nom_de_machine)
```

En saisissant cette commande sur votre propre machine, vous obtiendrez un résultat similaire à celle-ci :

```
[trainee@centos7 ~]$ su -
Mot de passe :
Dernière connexion : lundi  9 mai 2016 à 22:47:48 CEST sur pts/0

[root@centos7 ~]# ssh -l trainee localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is 19:cd:05:58:af:2c:10:82:52:ba:e3:31:df:bd:72:54.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
trainee@localhost's password: trainee
Last login: Mon May  9 23:25:15 2016 from localhost.localdomain
```

Tunnels SSH

Le protocole SSH peut être utilisé pour sécuriser les protocoles tels telnet, pop3 etc.. En effet, on peut créer un *tunnel* SSH dans lequel passe les communications du protocole non-sécurisé.

La commande pour créer un tunnel ssh prend la forme suivante :

```
ssh -N -f compte@hôte -Lport-local:localhost:port_distant
```

Dans votre cas, vous allez créer un tunnel dans votre propre vm entre le port 15023 et le port 23 :

```
[root@centos7 ~]# ssh -N -f trainee@localhost -L15023:localhost:23
trainee@localhost's password:
```

Installez maintenant le client et le serveur telnet :

```
[root@centos7 ~]# yum install telnet telnet-server
```

Telnet n'est ni démarré ni activé. Il convient donc de le démarrer et de l'activer :

```
[root@centos7 ~]# systemctl status telnet.socket
● telnet.socket - Telnet Server Activation Socket
   Loaded: loaded (/usr/lib/systemd/system/telnet.socket; disabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:telnetd(8)
   Listen: [::]:23 (Stream)
 Accepted: 0; Connected: 0

[root@centos7 ~]# systemctl start telnet.socket

[root@centos7 ~]# systemctl status telnet.socket
● telnet.socket - Telnet Server Activation Socket
   Loaded: loaded (/usr/lib/systemd/system/telnet.socket; disabled; vendor preset: disabled)
   Active: active (listening) since Mon 2016-05-09 23:40:13 CEST; 3s ago
     Docs: man:telnetd(8)
   Listen: [::]:23 (Stream)
 Accepted: 0; Connected: 0

May 09 23:40:13 centos7.fenestros.loc systemd[1]: Listening on Telnet Server Activation Socket.
May 09 23:40:13 centos7.fenestros.loc systemd[1]: Starting Telnet Server Activation Socket.
```

```
[root@centos7 ~]# systemctl enable telnet.socket
Created symlink from /etc/systemd/system/sockets.target.wants/telnet.socket to
/usr/lib/systemd/system/telnet.socket.
[root@centos7 ~]# systemctl status telnet.socket
● telnet.socket - Telnet Server Activation Socket
   Loaded: loaded (/usr/lib/systemd/system/telnet.socket; enabled; vendor preset: disabled)
   Active: active (listening) since Mon 2016-05-09 23:40:13 CEST; 36s ago
     Docs: man:telnetd(8)
   Listen: [::]:23 (Stream)
 Accepted: 0; Connected: 0

May 09 23:40:13 centos7.fenestros.loc systemd[1]: Listening on Telnet Server Activation Socket.
May 09 23:40:13 centos7.fenestros.loc systemd[1]: Starting Telnet Server Activation Socket.
```

Connectez-vous ensuite via telnet sur le port 15023, vous constaterez que votre connexion n'aboutit pas :

```
[root@centos7 ~]# telnet localhost 15023
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Kernel 3.10.0-327.13.1.el7.x86_64 on an x86_64
centos7 login: trainee
Password:
Last login: Mon May  9 23:26:32 from localhost.localdomain
[trainee@centos7 ~]$
```

Important - Notez bien que votre communication telnet passe par le tunnel SSH.

SCP

Introduction

La commande **scp** est le successeur et la remplaçante de la commande **rcp** de la famille des commandes **remote**. Il permet de faire des transferts sécurisés à partir d'une machine distante :

```
$ scp compte@numero_ip(nom_de_machine):/chemin_distant/fichier_distant /chemin_local/fichier_local
```

ou vers une machine distante :

```
$ scp /chemin_local/fichier_local compte@numero_ip(nom_de_machine):/chemin_distant/fichier_distant
```

Utilisation

Nous allons maintenant utiliser **scp** pour chercher un fichier sur le «serveur» :

Créez le fichier **/home/trainee/scp_test** :

```
[trainee@centos7 ~]$ pwd
/home/trainee
[trainee@centos7 ~]$ touch scp_test
```

Récupérez le fichier **scp_test** en utilisant scp :

```
[trainee@centos7 ~]$ touch /home/trainee/scp_test
[trainee@centos7 ~]$ scp trainee@127.0.0.1:/home/trainee/scp_test /tmp/scp_test
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
ECDSA key fingerprint is 19:cd:05:58:af:2c:10:82:52:ba:e3:31:df:bd:72:54.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '127.0.0.1' (ECDSA) to the list of known hosts.
```

```
trainee@127.0.0.1's password: trainee
scp_test                                                                                    100%    0
0.0KB/s   00:00
[trainee@centos7 ~]$ ls /tmp/scp_test
/tmp/scp_test
```

Mise en place des clefs

Il convient maintenant de se connecter sur le «serveur» en utilisant ssh et vérifiez la présence du répertoire ~/.ssh :

En saisissant cette commande, vous obtiendrez une fenêtre similaire à celle-ci :

```
[trainee@centos7 ~]$ ssh -l trainee 127.0.0.1
trainee@127.0.0.1's password:
Last login: Mon May  9 23:42:46 2016 from localhost.localdomain
[trainee@centos7 ~]$ ls -la | grep .ssh
drwx------.  2 trainee trainee 4096 May  9 23:25 .ssh
[trainee@centos7 ~]$ exit
logout
Connection to 127.0.0.1 closed.
```

Si le dossier distant .ssh n'existe pas dans le répertoire personnel de l'utilisateur connecté, il faut le créer avec des permissions de 700.
Dans votre cas, puisque votre machine joue le rôle de serveur **et** du client, le dossier /home/trainee/.ssh existe **déjà**.

Ensuite, il convient de transférer le fichier local **.ssh/id_ecdsa.pub** du «client» vers le «serveur» en le renommant en **authorized_keys** :

```
[trainee@centos7 ~]$ scp .ssh/id_ecdsa.pub trainee@127.0.0.1:/home/trainee/.ssh/authorized_keys
trainee@127.0.0.1's password: trainee
id_ecdsa.pub                                                                                100%  227
0.2KB/s   00:00
```

Connectez-vous via ssh et insérer les clefs publiques restantes dans le fichier .ssh/authorized_keys :

```
root@centos7 ~]# ssh -l trainee localhost
trainee@localhost's password: trainee
Last login: Tue May 10 01:39:33 2016 from localhost.localdomain
[trainee@centos7 ~]$ cat .ssh/id_rsa.pub >> .ssh/authorized_keys
[trainee@centos7 ~]$ cat .ssh/id_dsa.pub >> .ssh/authorized_keys
[trainee@centos7 ~]$ cat .ssh/id_ed25519.pub >> .ssh/authorized_keys
[trainee@centos7 ~]$ cat .ssh/authorized_keys
ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBG5BtOMFLrUbxD//RLELvkkA06CQvXuJqKSjSB2dlUXgaPyEJXuwHO0pxcdbr
g4qqb0f9sE75oMVowXxYgqhWDE= trainee@centos7.fenestros.loc
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQC9KOuEH5+kyihhm99Na8UTA4Gi5AfiOVeJyS3UzH7ta73ewmv7JZqaXzar1NlHcpEMkCUs2yKxHyO/yAfjb
CSdow5vfwJiuJTes+HbpvsJqKp1+OR7tf+OMgjDcajoGi7DYuybIs9QrbWgh57QclbldHQXRO+xbeUTykxcRun7AvR5uWZe4zMooBAmVVEms+l1rn
8CUi+D811jqQGSpU39PxkojTAwgbxlevT/Twy4sfeRR47UHc3AbrHb8SgyKqbx5/S9UxkbkhJjckx0s58fnAwf9nX5rKE7RdCQisRvdLeLHozq3EO
omvc7kzejefBtUDWxBEjnSeAgIP3+OEQl trainee@centos7.fenestros.loc
ssh-dss
AAAAB3NzaC1kc3MAAACBAK9/4siucBnf/NAHBMjZWIx1coA/wYVBjfudVyKArip1fVUuYqf0Ri9vTorG8KJ2zzLRbW5z7V5ZDSn4f6P5Kv7K5xVPn
e9dYQHxImkZIljpFseUW56BwCvcgTNZVLD0tYZzF+B0/Py4waJW+pnTDfZush6DYyAhVnEuxIPI4i+PAAAAFQCeCZyDRo1o41lf19qWGJTG7W+ChQ
AAAIAKtQe9QlkW4CA9kP+q4v3N07WR5TzWsvfZARjGXgrSqTo0BeQgMLwRJHeEOhdsgJ30cNbl6QXlB4G4J6dUoTiN/sY1dFbXzjzsT/MHLedsllV
fXXRQxgvN2nsbsKEUnmqEBWzgw5s6KOkGX33+0Six0E3xvOrYxkMNLP/5VT4aQwAAAIEAmOS94peBeo78yCKzCvSFnEL72dUCFFA6CGFGqgffhK1v
P5H5pG5vQxzBn9NnIXURCACF7ZxtZaxohSoB1M0/s0DfrfNIvXRMGvsJpZ9B2psTMDl9qBffIfIARnwkWKG1gC/lWaovUpDByE1wl09ZCDCnZp/16
ULJY0zvJ566Seg= trainee@centos7.fenestros.loc
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIENas3A3hmXFj1cb+lrn2NAt6g95Pla6qUFQHd1wg2y1 trainee@centos7.fenestros.loc
ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBG5BtOMFLrUbxD//RLELvkkA06CQvXuJqKSjSB2dlUXgaPyEJXuwHO0pxcdbr
g4qqb0f9sE75oMVowXxYgqhWDE= trainee@centos7.fenestros.loc
```

Lors de la connexion suivante au serveur, l'authentification utilise le couple de clefs asymétrique et aucun mot de passe n'est requis :

```
[trainee@centos7 ~]$ ssh -l trainee localhost
Last login: Tue May 10 01:50:39 2016 from localhost.localdomain
[trainee@centos7 ~]$ exit
```

```
déconnexion
Connection to localhost closed.
```

Important - Le fichier **authorized_keys** doit avoir les permissions de 600.

LAB #4 - Mise en place d'un VPN avec OpenVPN

Présentation

OpenVPN permet à des pairs de s'authentifier entre eux à l'aide :

- d'une **clé privée partagée** à l'avance,
- de **certificats** ou,
- à partir de la version 2.0 et à condition que le serveur possède un certificat, de **couples de noms d'utilisateur/mot de passe** sans besoin d'un certificat client

OpenVPN :

- utilise de manière intensive la bibliothèque d'authentification **OpenSSL** ainsi que le protocole **SSLv3/TLSv1**,
- n'est pas compatible avec IPsec ou d'autres logiciels VPN.

Configuration commune au client et au serveur

Commencez par vérifiez si le paquet **openssl** est bien installé :

```
[root@centos7 ~]# rpm -q openssl
openssl-1.0.2k-8.el7.x86_64
```

Installez ensuite le paquet openvpn :

```
[root@centos7 ~]# yum install openvpn
```

Naviguez au répertoire **/etc/openvpn** et créez la clef partagée :

```
[root@centos7 ~]# cd /etc/openvpn/
[root@centos7 openvpn]# openvpn --genkey --secret static.key
[root@centos7 openvpn]# cat static.key
#
# 2048 bit OpenVPN static key
#
-----BEGIN OpenVPN Static key V1-----
54f96ea50dbef7d5341efeda459b05ad
5af134bf915bbd867fdd6310f4f0b72b
331a82cdc6080622a7861e8c30cd0ffb
6b35c143e5c715077247270bdb610fc8
4c536f34742ba23f2bfe9ab148b3fa04
20d1f6e5a20d58db30cce56ce1ca5744
3028353a7e5e47b3f630738b71b04a1e
e388b5e986826ce481ff457157b3492e
61c147cd3d4373e283ad91c8ac44c0e8
3b593d342cd0a2600db7b3e7cd0efa89
d38dd861c1e4fc566e5e50004b102c7f
b444795e2691cd59dfbb51e79996339d
7e54d002aa4d5c63b3c155fbcc20f696
fe148128f2e94e509c39c72c117a684b
9fa8c7e159c451a7c52f42b2260d62c9
586d66a454319ba538559c143643e434
-----END OpenVPN Static key V1-----
```

L'architecture réseau sera donc la suivante :

```
                                    serveur <------------Votre réseau-------------> client
```

```
                                    |                                                   |
                                    |                                                   |
                                eth0@ifxxx                                          eth0@ifxxx
                                172.yy.0.3                                          172.yy.0.4
```

Configuration du client

Créez le fichier **/etc/openvpn/client.conf** :

```
[root@centos7 ~]# vi /etc/openvpn/client.conf
[root@centos7 ~]# cat /etc/openvpn/client.conf
remote 10.0.2.15
dev tun
port 1194
proto udp
comp-lzo
ifconfig 10.0.0.2 10.0.0.1
secret /etc/openvpn/static.key
```

Important - Trouvez la signification de chacune des directives dans ce fichier.

Arrêtez le service **firewalld** :

```
[root@centos7 ~]# systemctl stop firewalld
```

Lancez openvpn en ligne de commande et en arrière plan en spécifiant une journalisation :

```
[root@centos7 ~]# openvpn --config /etc/openvpn/client.conf > /var/log/vpn 2>&1 &
```

Vérifiez ensuite que le **socket** d'openvpn soit ouvert :

```
[root@centos7 ~]# netstat -an | grep 1194
udp        0      0 0.0.0.0:1194                0.0.0.0:*
```

Constatez ensuite la table de routage :

```
[root@centos7 ~]# netstat -ar
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Fenêtre irtt Iface
default         gateway         0.0.0.0         UG        0 0          0 enp0s3
10.0.0.1        0.0.0.0         255.255.255.255 UH        0 0          0 tun0
10.0.2.0        0.0.0.0         255.255.255.0   U         0 0          0 enp0s3
```

Notez la présence de la route via **tun0**.

Constatez ensuite le montage du tunnel en regardant le contenu du fichier de journalisation **/var/log/vpn** :

```
[root@centos7 ~]# tail /var/log/vpn
```

Configuration du serveur

Créez le fichier **/etc/openvpn/server.conf** :

```
[root@centos7 ~]# vi /etc/openvpn/server.conf
[root@centos7 ~]# cat /etc/openvpn/server.conf
dev tun
ifconfig 10.0.0.1 10.0.0.2
secret /etc/openvpn/static.key
port 1194
proto udp
user nobody
```

```
group nobody
daemon
comp-lzo
keepalive 10 60
ping-timer-rem
persist-tun
persist-key
log /var/log/vpn
verb 1
```

Important - Trouvez la signification de chacune des directives dans ce fichier.

Arrêtez le service **firewalld** :

```
[root@centos7 ~]# systemctl stop firewalld
```

Lancez openvpn en ligne de commande et en arrière plan en spécifiant une journalisation :

```
[root@centos7 ~]# openvpn --config /etc/openvpn/server.conf > /var/log/vpn 2>&1 &
[1] 7751
```

Vérifiez ensuite que le **socket** d'openvpn soit ouvert :

```
[root@centos7 ~]# netstat -an | grep 1194
udp        0      0 0.0.0.0:1194                0.0.0.0:*
```

Constatez ensuite la table de routage :

```
[root@centos7 ~]# netstat -ar
Kernel IP routing table
```

```
Destination     Gateway         Genmask         Flags   MSS Fenêtre irtt Iface
0.0.0.0         10.0.2.2        0.0.0.0         UG        0 0          0 enp0s3
10.0.0.1        0.0.0.0         255.255.255.255 UH        0 0          0 tun0
10.0.2.0        0.0.0.0         255.255.255.0   U         0 0          0 enp0s3
```

Constatez ensuite le montage du tunnel en regardant le contenu du fichier de journalisation **/var/log/vpn** :

```
[root@centos7 ~]# tail /var/log/vpn
```

Tests

Du client vers le serveur

Sur le client, utilisez la commande ping pour envoyer des paquets dans le tunnel :

```
[root@centos6 ~]# ping -c3 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=7.62 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=1.35 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.000 ms

--- 10.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2047ms
rtt min/avg/max/mdev = 0.000/2.994/7.629/3.323 ms
```

Du serveur vers le client

Sur le serveur, utilisez la commande ping pour envoyer des paquets dans le tunnel :

```
[root@centos7 ~]# ping -c5 10.0.0.2
```

```
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=2.59 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=9.08 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=7.24 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=7.03 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=4.08 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4034ms
rtt min/avg/max/mdev = 2.597/6.008/9.084/2.340 ms
```

<html>

</html>