Université Grenoble Alpes

IUT1 de Grenoble

Département RESEAUX et TELECOMMUNICATIONS

Licence Professionnelle

Réseaux Informatiques, Mobilité, Sécurité RIMS

Module 25 :
TP Sécurité

CRYPTOGRAPHIE
2022-2023

1 Présentation, objectifs

Ce TP a pour objectifs la pratique de la cryptographie et la mise en œuvre d'un système de messagerie sécurisé. Le TP se divise en 3 parties, la mise en place de la plate-forme, la pratique de la cryptographie et la mise en œuvre du système de messagerie sécurisé. Dans la suite, « X » caractérise votre numéro de binôme, comme vous en avez l'habitude dans la salle de TP.

2 Mise en place de la plate-forme

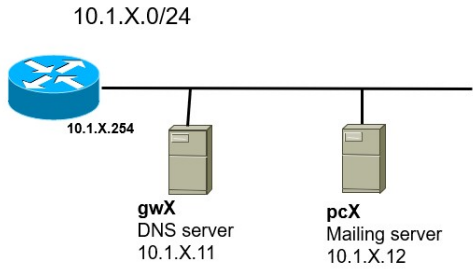
2.1 Description de la plate-forme

Pour ce TP vous utilisez 2 ordinateurs sous centOS (Community Enterprise OS, variante de la version payante Redhat Enterprise Linux) dans une machine virtuelle VM.

- L'ordinateur de gauche gèrera le DNS de votre domaine zX.rt.iut
- L'ordinateur de droite gèrera le serveur de messagerie.

Avant de lancer la machine virtuelle, il vous faut valider et configurer la carte réseau sous VirtualBox. Pour ce faire, appuyez sur "setting" dans la barre supérieure, puis dans l'adaptateur 1, sélectionnez Intel et sélectionnez le mode pont. Lorsque vous démarrerez votre machine, vous pourrez voir ensp03 (eth0) dans les paramètres réseau. Vous pourrez ensuite configurer la carte réseau dans votre machine virtuelle. Provisoirement vous mettrez le DNS sur 172.16.0.200 en attendant de pouvoir utiliser votre propre DNS, pour les tests préliminaires.

Schéma de la plate-forme :



Configuration réseau :

OS	Rôle	eth0	eth1	eth2	Hostname	Gateway
Centos 7.5	DNS Server	10.1.X.11/24	non activée	N/A	gwX.zX.rt.iut	10.1.X.254
Centos 7.5	Mailing Server	10.1.X.12/24	non activée	N/A	pcX.zX.rt.iut	10.1.X.254
Si besoin, machine physique GAUCHE		10.1.X.1/24	non activée	N/A		10.1.X.254
Si besoin, machine physique DROITE		10.1.X.2/24	non activée	N/A		10.1.X.254

Résolution DNS :

Machine	Serveur DNS primaire	Serveur DNS secondaire	Suffixe DNS par défaut
Gauche	127.0.0.1	172.16.0.200	
Droite	10.1.X.11 (lorsqu'il sera opérationnel)	N/A	

2.2 Installation de CentOS et câblage

Câblage :

Les cartes eth0 des deux machines seront brassées sur le switch du réseau étudiant 10.1.X.0.
Les autres cartes ne seront pas utilisées.
Les 2 machines sont installées sur des **machines virtuelles**.
Les configurations des deux machines sont décrites dans les paragraphes suivants.
Fichiers de configurations : cette configuration peut être effectuée avec l'interface graphique en haut et à droite, utilisez l'onglet « Sharing » en remplaçant "localhost.localdomain" par vos propres noms d'hôte et de domaine)
Vous pouvez taper la commande « hostname » dans la fenêtre Terminal pour vérifier le nom d'hôte et de domaine.
Configurer les cartes réseau en utilisant l'interface graphique, la passerelle par défaut, le serveur DNS

Si besoin, utilisez Gedit pour éditer les fichiers nécessaires.
Pour ouvrir un fichier avec gedit il faut taper : `Gedit filename`

ATTENTION ! Il se peut que vous deviez exécuter toutes les commandes en mode super utilisateur (`sudo`).

2.3 Installation et configuration du serveur DNS BIND (MACHINE DE GAUCHE) :

2.3.1 Installation de BIND

L'installation de BIND se fait à l'aide de la commande yum.

```
[root@gwX ~]# yum install bind OU [user@gwX ~]# sudo yum install bind
```

Si vous rencontrez des difficultés lors de l'installation, vous pouvez essayer de vider le cache de YUM avec la commande suivante : [root@gwX ~] # yum clean all

2.3.2 Configuration de BIND

Les fichiers de configuration sont à créer. Pour cela, inspirez-vous des fichiers en annexe ou de fichiers de configuration que vous avez déjà réalisés dans les TP Réseaux.
Le fichier de configuration devra :

- Définir les options globales : /etc/named.conf
- Déclarer la zone de recherche directe : « zX.rt.iut » dans le fichier adéquat.
- Déclarer la zone inverse : « X.1.10.in-addr.arpa » Dans le fichier adéquat.

Configurer le serveur BIND en vous aidant des fichiers exemples en annexe.
Déterminer l'emplacement des fichiers de zone dans /etc/named.conf
Avant de modifier la configuration, il est recommandé de la sauvegarder
#cp /etc/named.conf /etc/named.conf.orig

Démarrer le serveur DNS :

```
[root@gwX ~]# service named start OU [user@gwX ~]# sudo service named start
```

La commande suivante permet de démarrer votre service à chaque fois :
[root@gwX ~] # Service named enable

La commande suivante est utilisée pour vérifier l'état du service `named` :
[root@gwX ~] # service named status
La commande suivante est utilisée pour redémarrer le service `named` :
[root@gwX ~] # service named restart
Si le service ne démarre pas, vous devez vérifier vos fichiers de configuration.
Pour voir s'il y a des erreurs, on peut visualiser : `cat /var/log/messages`

2.3.3 Validation

Une fois votre serveur configuré et démarré vous devez pouvoir :

- Depuis le client :
- résoudre tous les noms DNS (ex : www.google.fr)
- résoudre les noms dans la zone zX.rt.iut (ex : pop3.zX.rt.iut ou simplement pop3).
- Depuis le serveur :
- résoudre les noms dans la zone zX.rt.iut (ex : pcX.zX.rt.iut ou simplement pcX)

Élaborer et lancer une liste de commande permettant de le montrer. Vous pouvez illustrer par exemple les outils de recherche pour Windows et Linux (cf rappels ci-dessous), l'affichage des serveurs de noms pour un nom de domaine spécifique, le transfert de zone complet ou la demande de résolution inverse.

Montrer au professeur.

RAPPELS si besoin

host *is the simplest one:*
$ host www.rtgrenoble.fr # forward resolution
$ host 152.77.173.246 # reverse resolution
dig *accepts parameters and provides the reply directly in a format reminding a zone file.*
$ dig [@IP_or name_of_DNS_server] queried_name [question_type]
nslookup *can be used as a command or interactively. Under Linux, it is deprecated and dig is recommended. On Windows, it is available by default. C:\> nslookup [-q=question_type] queried domain. [IP_or_name]*

2.4 Installation et configuration de la messagerie (MACHINE DE DROITE)

2.4.1 Installation et configuration de Postfix

Installation :

L'installation de postfix se fait à l'aide de la commande « yum ».

```
root@pcX]# yum install postfix OU user@pcX]# sudo yum install postfix
```

Configuration :

La configuration de postfix se fait en éditant le fichier /etc/postfix/main.cf
Ajouter/Modifier les paramètres dans ce fichier pour avoir les paramètres suivants :

```
myhostname = pcX. zX.rt.iut
mydomain= zX.rt.iut
inet_interfaces = all
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain
mynetworks_style = subnet
home_mailbox = /var/mail/
my origin = $mydomain
```

Il existe de nombreux autres paramètres à la configuration d'un serveur postfix. Ils permettent plus de sécurité, l'activation de fonctionnalités … dans notre cas ces quelques paramètres suffisent.
Démarrer postfix :

```
root@pcX named]# service postfix start
```

2.4.2 Installation et configuration de Dovecot

Installation :

L'installation de dovecot se fait à l'aide de la commande « yum ».

```
root@pcX]# yum install dovecot OU user@pcX]# sudo yum install dovecot
```

Configuration :

La configuration de dovecot se fait en éditant le fichier /etc/dovecot/dovecot.conf.

Ajouter/Modifier les paramètres de ce fichier pour avoir :

```
protocols = imap pop3
disable_plaintext_auth = no
```

Modifier le fichier /etc/dovecot/conf.d/10-mail.conf

```
mail_location = maildir:~/var/mail
```

Modifier le fichier **/etc/dovecot/conf.d/10-auth.conf**

```
disable_plaintext_auth = no
```

Si besoin (nécessaire pour assurer la compatibilité avec Outlook, ce dont nous n'avons en fait pas besoin ici), modifier le fichier /etc/dovecot/conf.d/10-mail.conf en décommentant.

```
pop3_uidl_format = %08Xu%08Xv
```

Il existe de nombreux autres paramètres. Par exemple le paramétrage de la directive « listen » pour un ou plusieurs protocoles pris en charge peut nous permettre d'avoir notre serveur POP3 qui ne répond que lorsqu'il est interrogé depuis le réseau 10.1.X.0/24 ou même uniquement lorsqu'il est interrogé depuis lui-même.

Cela permet d'interdire à nos utilisateurs de relever leur courrier s'ils ne sont pas connectés au réseau interne. Quels sont les protocoles pris en charge par notre serveur Dovecot ?

Démarrer dovecot :

```
root@pcX]# service dovecot start
```

2.5 Création de comptes configuration de clients de messagerie

2.5.1 Création de comptes de messagerie (MACHINE DE DROITE)

Sur le serveur, créer le groupe « etudiants » :

```
[root@pcX named]# groupadd etudiants
```

Créer 2 comptes utilisateurs : chaque étudiant disposera d'un compte utilisateur sur le serveur de messagerie (MACHINE DE DROITE).

Les noms d'utilisateurs seront sous la forme : « 2 premières lettres du prénom + 6 première lettres du nom ».

Ex :

Michael Jordan aura comme nom d'utilisateur « mijordan ».

Arnold Schwarzenegger aura comme nom d'utilisateur « arschwar »

```
Linux command to create user: sudo adduser username
```

```
[root@pcX named]# useradd -g etudiants -d /home/username -m -s /bin/bash username
```

La création d'un compte utilisateur sur le serveur CentOS crée automatiquement une boite de messagerie associée.

2.5.2 Configuration de Thunderbird (DEUX MACHINES)

Le client de messagerie utilisé sur les 2 machines est Thunderbird.

Installer Thunderbird à l'aide de la commande « yum ».

Lancer Thunderbird.
Informations de configuration :

Type de compte	Compte courrier électronique
Votre Nom	votre nom
Adresse de courrier	votrelogin@zX.rt.iut
Type de serveur de réception	POP
Nom du serveur de réception	Choisissez parmi les enregistrements DNS de la zone zX.rt.iut celui qui correspond le mieux.
Serveur d'envoi	Choisissez parmi les enregistrements DNS de la zone zX.rt.iut celui qui correspond le mieux.
Nom d'utilisateur entrant	votrelogin
Nom d'utilisateur sortant	votrelogin
Nom du compte	votrelogin@zX.rt.iut

Veillez à fournir un nom d'utilisateur et un mot de passe corrects, le mot de passe doit correspondre à celui que vous avez fourni lors de la configuration des utilisateurs locaux.
Cliquez ensuite sur la configuration manuelle.
Sélectionnez POP3.
Sélectionnez le bon nom de serveur, il doit être celui mentionné dans le fichier de configuration. Assurez-vous que le DNS fonctionne correctement et que vous pouvez effectuer un ping sur ce nom depuis le client.
Une chose importante à considérer ici est que lors de la configuration, lorsque nous cliquons sur les configurations manuelles, il est automatiquement donné un nom de serveur correspondant au domaine de votre compte de messagerie, assurez-vous de supprimer le "." avant le nom de ce serveur
Un exemple de capture d'écran est donné en annexe.

2.5.3 Validation

A ce stade vous devez pouvoir échanger des messages électroniques.
Montrer au professeur.

2.6 Mise en évidence de problèmes de sécurité

Analyse de trames :

Sur le serveur CentOS installer Wireshark à l'aide de la commande « yum ».

Lancer une capture sur l'interface eth1. Appliquer le filtre suivant : « pop.request.command ».

ATTENTION ! Il sera sûrement nécessaire de lancer Wireshark via la commande `sudo`.

Un exemple d'analyse de trames se trouve en annexe. Faire une capture de votre propre trame. Enregistrer et garder de côté.

Que pouvez-vous déduire de cette capture concernant la sécurité de votre messagerie ?

3 Utilisation d'outils de cryptographie (DEUX MACHINES)

3.1 Présentation de OpenSSL :

Le protocole SSL (Secure Socket Layer) a été développé par la société Netscape Communications Corporation pour permettre aux applications client/serveur de communiquer de façon sécurisée.
TLS (Transport Layer Security) est une évolution de SSL réalisée par l'IETF.
SSL est un protocole qui s'intercale entre TCP/IP et les applications qui s'appuient dessus.

Une session SSL se déroule en 2 temps :

- Le handshake (poignée de main) : c'est une phase au cours de laquelle le client et le serveur conviennent du système de chiffrement et de la clé qui sera utilisée par la suite. La méthode de chiffrement la plus forte connue des 2 systèmes sera utilisée.
- La communication proprement dite. Lors de cette communication, les données sont échangées de manière chiffrées et signées.

L'identification qui a lieu pendant le handshake se fait à l'aide de certificats X.509.

OpenSSL est une boite à outils cryptographiques implémentant les protocoles TLS et SSL.
OpenSSL comprend :

- Une bibliothèque de programmation en C qui permet de réaliser des applications client/serveur sécurisées (dont les communications s'appuient sur SSL/TLS).
- Un outil en ligne de commande qui permet :
    - La création de clés RSA / DSA.
    - La création de certificats X509.
    - Le calcul d'empreintes (MD5, SHA, …).
    - Le chiffrement et déchiffrement (DES, IDEA, RC2, RC4, …).
    - La réalisation de tests de clients et serveurs SSL/TLS.
    - La signature et le chiffrement de courriers (S/MIME).

Exemples d'utilisation :

Pour tous ces exemples, le répertoire de travail est le répertoire home de l'utilisateur qui manipule.

Créer un fichier nommé « texte.clair » contenant le texte :

« Chiffrer et déchiffrer des données avec openSSL n'est pas si compliqué !! »

3.2 Chiffrement symétrique RC4

Chiffrer une donnée c'est rendre son interprétation impossible à toute personne qui ne connaît pas la clé de déchiffrement. Dans le cas d'un chiffrement symétrique on utilise la même clé pour chiffrer et déchiffrer.

Chiffrer le fichier texte.clair en lui appliquant RC4 :

```
[etud1@pcX ~]$ openssl rc4 -in texte.clair -out texte.rc4
```

Recommencer l'opération avec la même clé :

```
[etud1@pcX ~]$ openssl rc4 -in texte.clair -out texte.rc42
```

Comparer ces deux fichiers de sortie :

```
diff texte.rc4 texte.rc42
```

Ces deux fichiers sont-ils identiques ? (pour comparer, on peut utiliser la commande `cmp`)

Précisions sur RC4 :

Le mot de passe n'est pas utilisé directement comme clé. Il sert à la générer avec un nombre aléatoire : « le salt » (ou graine en français). Ainsi deux messages chiffrés avec le même mot de passe ne seront pas identiques.

Envoyer le fichier texte.rc4 à votre voisin avec la clé de chiffrement.

Déchiffrer le fichier de votre voisin :

```
[etud2@gwX ~]$ openssl rc4 -d -in texte.rc4 -out texte.rc4.dec
```

Comparer avec le fichier d'origine :

```
[etud2@gwX ~]$ diff texte.clair texte.rc4. dec
```

Que constate-t-on ?

3.3 Codage base64

Reprendre les informations dans votre capture de trame POP3

Créez 2 fichiers :

- 1.b64 :

```
echo ZXR1ZDE= > /home/votrelogin/1.b64
```

- 2.b64 :

```
echo dG90bw== > /home/votrelogin/2.b64
```

Décoder ces 2 fichiers avec base64:

```
[etud2@gwX ~]$ openssl enc -base64 -d -in 1.b64 -out 1
[etud2@gwX ~]$ openssl enc -base64 -d -in 2.b64 -out 2
```

En quoi base64 n'est pas un algorithme de chiffrement ?

Que dire sur la sécurité de votre messagerie ?

3.4 Techniques de hachage

Une fonction de hachage est une fonction à sens unique. Elles permettent de calculer un code de contrôle (empreinte) liée au message à transmettre. En théorie il est impossible de déterminer le contenu d'un fichier à partir de son empreinte.

Les deux techniques de hachage les plus utilisées sont md5 (obsolète aujourd'hui) et SHA.

Hacher le fichier texte.clair avec sha1:

```
[etud2@gwX ~]$ openssl sha1 texte.clair
SHA1(texte.clair)= 34dad5a2e10810d84be7d787edc212a8
[etud2@gwX ~]$ openssl sha1 texte.clair > emptexte.clair
```

Modifier temporairement le fichier texte.clair.

Hacher à nouveau texte.clair:

```
[etud2@gwX ~]$ openssl sha1 texte.clair > emptexte.clair.2
```

Comparer les deux empreintes:

```
diff emptexte.clair emptexte.clair.2
```

Enlever les modifications apportées à texte.clair et renouveler l'opération.

Que peut-on déduire d'une comparaison d'empreintes:

- Dans le cas où elles sont identiques.
- Dans le cas où elles sont différentes ?

3.5 Mot passe Linux

Les mots de passe sont chiffrés avec l'algorithme symétrique DES. Ni le mot de passe ni même sa version chiffrée avec DES n'est stockée sur le disque dur. On stocke l'empreinte du mot de passe.

Fichier /etc/shadow:(more /etc/shadow | grep votrelogin)

```
etud2:$1$G1tcFI7A$EMH4ctom2PrQJ6HFC/A43/:15318:0:99999:7:::
```

user :-----SALT------$---------EMPREINTE---------:

Génération de l'empreinte du mot de passe « toto1 » de etud2 avec la graine (salt) G1tcFI7A

Exécutez :

```
[root@gwX etud2]# openssl passwd -1 -salt G1tcFI7A toto1
```

Quel est le mot de passe du système d'où cet exemple est tiré ?

Sur les distributions récentes, l'algorithme a été changé (6 à la place de 1) cela ne fonctionne donc plus.

3.6 Méthodes asymétriques à couple de clé publique/privée

On rencontre le plus souvent :

- Diffie-Helman.
- RSA (Rivest Shamir Adleman).
- DSA (Digital Signature Algorithm).

Générer un couple de clés :

```
[root@gwX etud2]# openssl genrsa -out username.priv 4096
```

Cette commande génère une clé privée de longueur 4096 bits.

Générer la clé publique à partir de la clé privée :

```
[root@gwX etud2]# openssl rsa -in username.priv -pubout -out username.pub
```

Chiffrer texte.clair avec la clé publique :

```
[root@gwX etud2]# openssl rsautl -inkey username.pub -pubin -in texte.clair -out texte.username.pub -encrypt
```

Déchiffrer avec la clé privée :

```
[root@gwX etud2]# openssl rsautl -inkey username.priv -in texte.username.pub -out texte.username.pub.decrypt -decrypt
```

Évidemment le chiffrement et le déchiffrement sont rarement exécutés sur la même machine.

Réaliser l'opération inverse :

On chiffre avec la clé privée et déchiffre avec la clé publique.

Quel est le résultat ? Cela parait-il logique ? Expliquer pourquoi.

3.6.1 Signature numérique

La signature numérique (parfois appelée signature électronique) est un mécanisme permettant de garantir l'intégrité d'un document électronique et d'en authentifier l'auteur, par analogie avec la signature manuscrite d'un document papier.

Un mécanisme de signature numérique doit présenter les propriétés suivantes :

- Il doit permettre au lecteur d'un document d'identifier la personne ou l'organisme qui a apposé sa signature.
- Il doit garantir que le document n'a pas été altéré entre l'instant où l'auteur l'a signé et le moment où le lecteur le consulte.

Pour cela, les conditions suivantes doivent être réunies :

- Authentique : L'identité du signataire doit pouvoir être retrouvée de manière certaine.
- Infalsifiable : La signature ne peut pas être falsifiée. Quelqu'un ne peut se faire passer pour un autre.
- Non réutilisable : La signature n'est pas réutilisable. Elle fait partie du document signé et ne peut être déplacée sur un autre document.
- Inaltérable : Un document signé est inaltérable. Une fois qu'il est signé, on ne peut plus le modifier.
- Irrévocable : La personne qui a signé ne peut le nier.

La signature électronique n'est devenue possible qu'avec la cryptographie asymétrique.

Contrairement à la signature écrite elle n'est pas visuelle, mais correspond à une suite de nombres.

Grâce à la clé publique de l'émetteur, le récepteur peut :

- Vérifier le hash reçu (option verify)
- Elaborer sa propre version du hash de l'information reçue
- Comparer les deux versions du hash.

Les lignes suivantes décrivent des exemples d'utilisation, adaptez à votre situation et échangez-vous des messages signés.

Vérifiez cette signature à l'aide de la clé publique.

```
[etud2@gwX digitalsigning]$ openssl rsautl -verify -in signature_num.sign -inkey cle_iut.pub -pubin > signature_num.emp
```

Elaborer votre propre version de l'empreinte a partir du fichier signature_num.pdf :

```
[etud2@gwX digitalsigning]$ openssl dgst -sha1 -out signature_num.emp2 signature_num.pdf
```

Comparez les empreintes, qu'en déduisez-vous ?

Dans la pratique on fait un peu plus vite :

Signature d'un fichier signature_num.pdf :

```
[etud2@gwX digitalsigning]$ openssl dgst -sha1 -sign cle_iut.priv -out signature_num.sign signature_num.pdf
```

Vérification du fichier avec sa signature :

```
[etud2@gwX digitalsigning]$ openssl dgst -sha1 -verify cle_iut.pub -signature signature_num.signsign signature_num.pdf
```

3.6.2 PGP (Pretty Good Privacy)

Simulation de PGP avec openSSL:

Vous utiliserez le couple de clés créé précédemment pour cette simulation.

Vous allez utiliser les techniques de cryptage symétrique et asymétrique. Un étudiant cryptera avec une clé symétrique de son choix (gardez-la secrète) un fichier confidentiel.

Il échangera cette clé symétrique avec son binôme grâce son couple de clé asymétriques et la clé publique de son binôme.

Le destinataire utilisera son couple de clés asymétriques et la clé publique de son binôme pour récupérer la clé secrète et déchiffrera le fichier confidentiel.

1. Échangez vos clés publiques grâce à votre messagerie.

2. L'étudiant de droite crée un fichier key.txt contenant la clé symétrique et un fichier nommé confidentiel.txt qui contient une information secrète.

Pour générer une clé symétrique : Openssl rand 128 > key.txt

3. L'étudiant de droite chiffre ce fichier avec la clé secrète key.txt. La commande suivante est utilisée pour crypter avec la clé symétrique.

```
[etud1@pcX ~]$ openssl enc -bf -in confidentiel.txt -k key.txt -out confidentiel.txt.cryptsym
```

4. Le message crypté est haché :

```
[etud1@pcX ~]$ openssl dgst -sha1 -binary confidentiel.txt.cryptsym  > confidentiel.txt.cryptsym.dgst
```

5. Le message est signé :

```
[etud1@pcX ~]$ openssl rsautl -in confidentiel.txt.cryptsym.dgst -sign -inkey etud1.priv > confidentiel.txt.cryptsym.sign
```

Générer des clés asymétriques publiques et privées pour l'autre utilisateur :

openssl genrsa -out username.priv 4096.

```
openssl rsa -in username.priv -pubout -out username.pub
```

6. L'étudiant de droite doit encore passer la clé secrète, il va la chiffrer avec la clé publique de

l'étudiant de gauche :

```
openssl rsautl -inkey username.pub -pubin -in key.txt -out key.txt.cryptasym -encrypt
```

7. L'étudiant de droite envoie à celui de gauche : *key.txt.cryptasym*, *confidentiel.txt.cryptsym.sign* et *confidentiel.txt.cryptsym* grâce à sa messagerie
8. L'étudiant de gauche va vérifier que le fichier confidentiel.txt.cryptsym n'a pas été modifié grâce à sa signature :

```
[etud2@gwX ~]$ openssl rsautl -verify -pubin -inkey etud1.pub -in
confidentiel.txt.cryptsym.sign -out dgst1
[etud2@gwX ~]$ openssl dgst -sha1 -binary confidentiel.txt.cryptsym > dgst2
[etud2@gwX ~]$ diff dgst1 dgst2
```

9. L'étudiant de gauche récupère la clé secrète grâce à sa clé privée :

```
[etud2@gwX ~]$ openssl rsautl -decrypt -inkey etud2.priv -out key.txt -in key.txt.cryptasym
```

10. Il déchiffre le message :

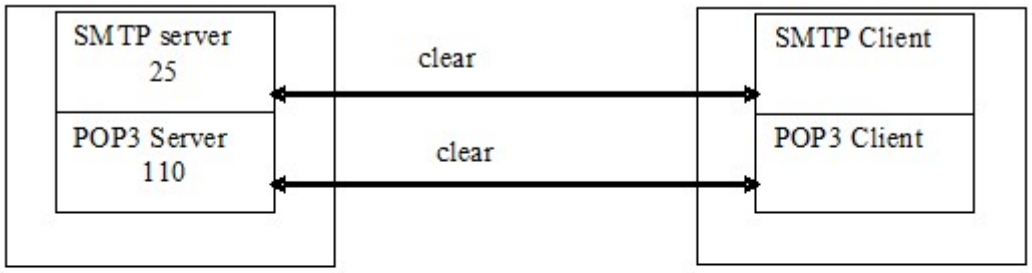
```
[etud2@gwX ~]$ openssl enc -bf -d -in confidentiel.txt.cryptsym -k key.txt -out
confidentiel.txt
```

Cette méthode simple nous permet d'envoyer des fichiers de manière sûre.
Quels sont les rôles de sécurité remplis ? (Dites à quelle(s) étape(s)).
En l'état il plane une incertitude, laquelle ?
Elaborer un schéma de la transmission sécurisée. Ce schéma est à rendre au professeur.

4 Techniques de mise en œuvre de système de messagerie sécurisé

4.1 Sécurisation de la messagerie avec stunnel

Situation classique sans sécurité :

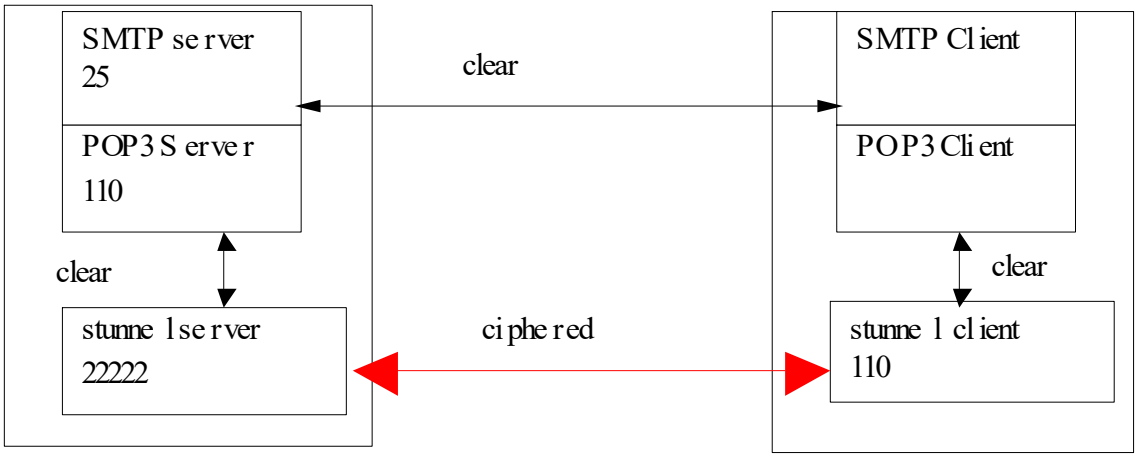


Linux station
SMTP and POP3
server
Postfix software

Windows station
SMTP and POP3
client
Outlook software

Tous les échanges entre le client et le serveur se font en clair.

Solution de sécurisation 1:



Linux station
SMTP and POP3
Serveur
Postfix software
Stunnel (on pop)

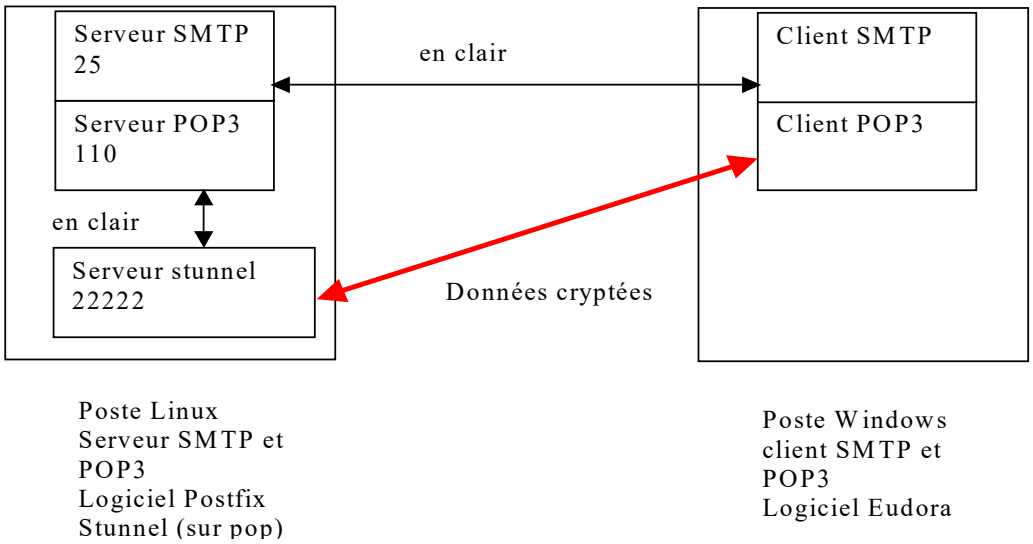
Window s station
SMTP and POP3
client
Eudora software
Stunnel Client

Le client POP3 se connecte sur le client stunnel.

Le client stunnel se connecte au serveur stunnel au moyen d'une connexion sécurisée.

Le serveur stunnel transmet la demande du client au serveur POP3. Aucun mot de passe en clair sur le réseau.

Solution de sécurisation 2:



Le client POP3 est capable de se connecter de manière sécurisée au serveur stunnel.

Le serveur stunnel relaye la demande au serveur POP3.

Choix de la solution :

La solution initiale n'est pas sécurisée donc inacceptable.

La seconde solution est sécurisée. Elle implique l'installation et la configuration d'un serveur stunnel sur le serveur mais aussi l'installation et la configuration d'un client stunnel sur tous les postes devant relever la messagerie.

La solution de sécurisation 2 est celle qui est retenue.

Configuration du serveur stunnel :

Sur le serveur :
Install stunnel
Yum install stunnel

- Créer une clé privée pour le serveur stunnel :

```
[root@gwX etud2]# openssl genrsa -out /etc/stunnel/stunnel.key 1024 OU [user@gwX etud2]# sudo openssl genrsa -out /etc/stunnel/stunnel.key 1024
```

- Créer une demande de certificat x509:

```
[root@gwX etud2]# openssl req -new -key /etc/stunnel/stunnel.key -out /etc/stunnel/stunnel.csr

Entrer la passe-phrase pour /etc/stunnel/stunnel.key:
Vous êtes sur le point d'être invité à saisir des informations qui seront intégrées à votre demande de certificat. Ce que vous êtes sur le point d'entrer est ce que l'on appelle un
```

```
Distinguished Name ou un DN. Il y a un certain nombre de champs, mais vous pouvez en
laisser certains vides. Pour certains champs, il y aura une valeur par défaut. Si vous entrez
'.', le champ sera laissé vide.
-----
Country Name (2 letter code) [GB]:FR
State or Province Name (full name) [Berkshire]:Isere
Locality Name (eg, city) [Newbury]:Grenoble
Organization Name (eg, company) [My Company Ltd]:LPRO
Organizational Unit Name (eg, section) []:RIMS
Common Name (eg, your name or your server's hostname) []:pop3.zX.rt.iut
Email Address []:root@zX.rt.iut
```

- Signer la demande de certificat :

```
[root@gwX etud2]# openssl x509 -req -days 365 -in /etc/stunnel/stunnel.csr -signkey
/etc/stunnel/stunnel.key -out /etc/stunnel/stunnel.crt

Signature ok
subject=/C=FR/ST=Isere/L=Grenoble/O=LPRO/OU=RIMS/CN=pop3.zX.rt..iut/emailAddres
s=root@zX.rt.iut
Getting Private key
```

- Configurer stunnel sur le serveur :

```
[root@gwX etud2]# cp /usr/share/doc/stunnel-4.56/stunnel.conf-sample /etc/stunnel.conf
```

Éditer ce fichier de manière à avoir les mêmes options qu'en annexe.

```
cert = /etc/stunnel/stunnel.crt
key = /etc/stunnel/stunnel.key
```

- Lancer stunnel

```
[root@gwX etud2]# stunnel /etc/stunnel.conf
```

Sur le client Thunderbird :

Changer la configuration du compte pour utiliser SSL lors de la relève.

Analyser les trames pendant une connexion avec wireshark. La sécurité est-elle complète ?

4.2 Sécurisation de SSH

Connexion SSH :

Sur le client CentOS :

- Se connecter à l'aide de SSH sur le server

```
[etud1@pcX ~]$ ssh root@gwX.zX.rt.iut
```

- Changer le mot de passe pour « WsD43%iop09=AZgh ».

-Se déconnecter et établir la connexion à nouveau.

Qu'en dites-vous ? Comment faciliter la connexion d'un administrateur ? Quel est le problème posé ?

Configuration de l'authentification par clé RSA

Prendre l'identité de l'utilisateur de la machine :

```
[etud1@pcX ~]$ su - username
```

Générer une paire de clé RSA :

```
[etud1@pcX ~]$ ssh-keygen
```

Ne mettez pas de passphrase.

Envoyer votre clé publique à votre binôme

Copier la clé publique de votre binôme dans le répertoire /root/.ssh/authorized_keys (format en annexe).

Se connecter au poste de votre binôme :

```
[etud1@pcX ~]$ ssh root@gwX.zX.rt.iut
```

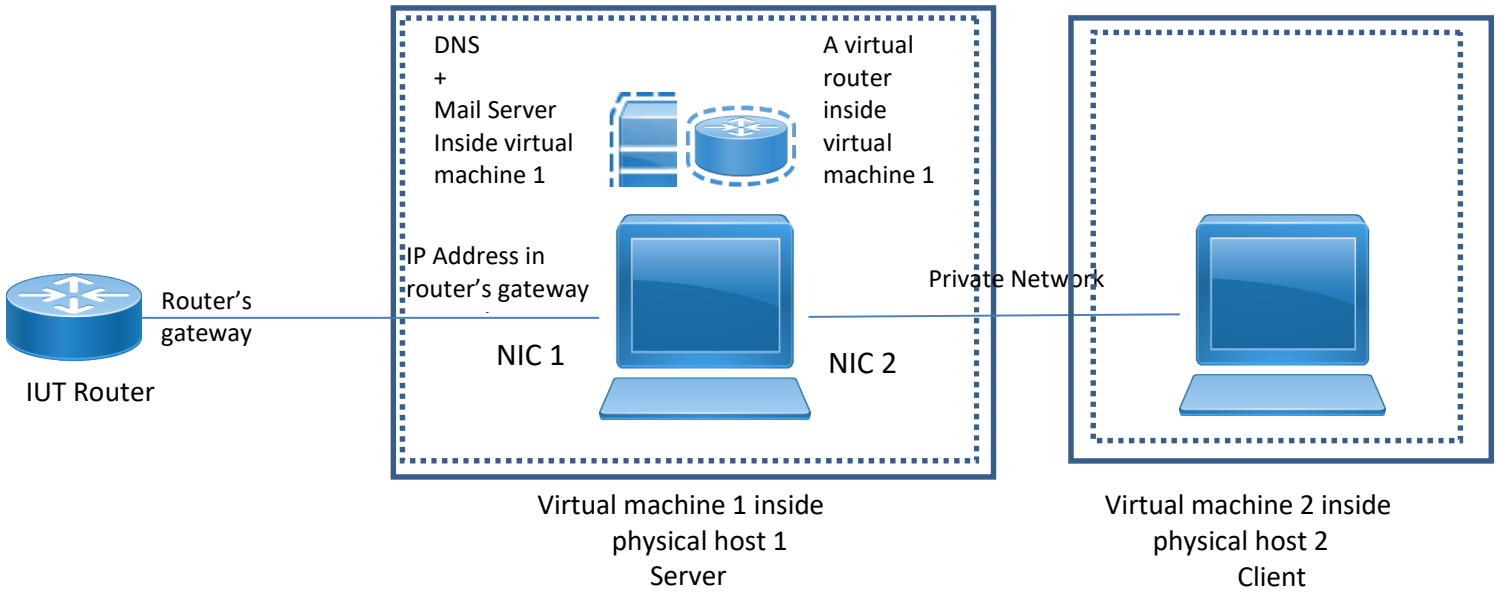
Que constatez-vous ?

A quel risque de sécurité est-on exposé ?

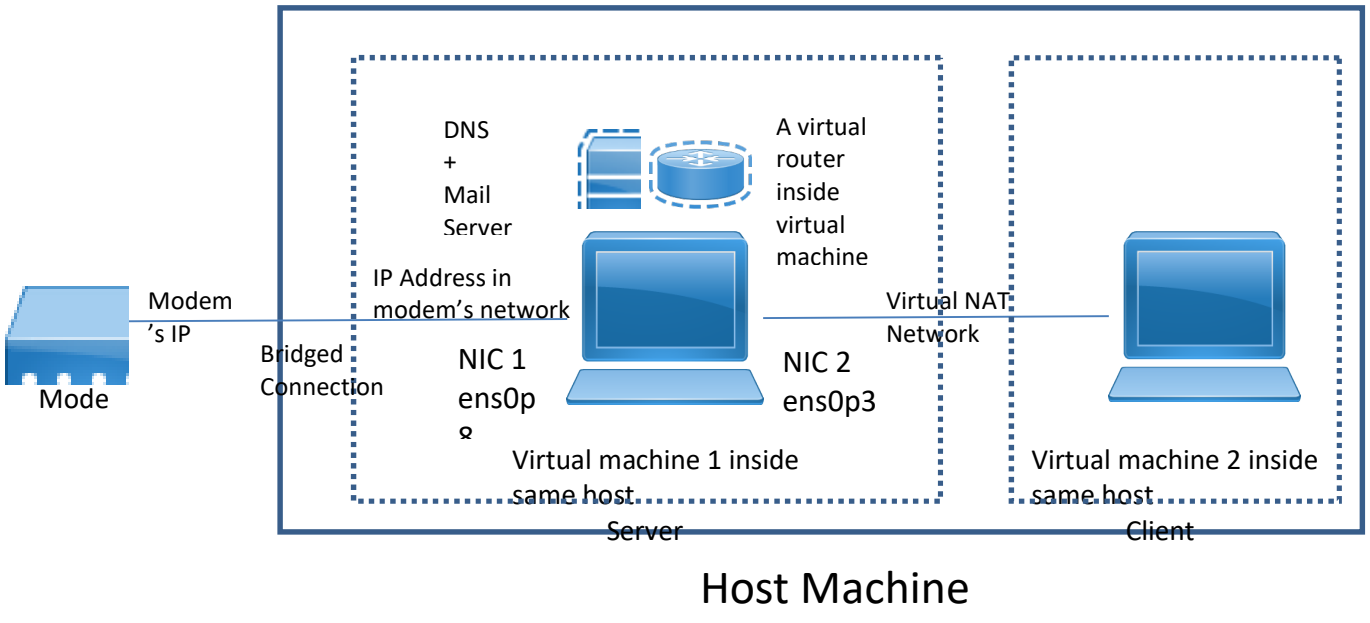
Améliorer la sécurité avec le mécanisme d'agent SSH. Son fonctionnement est largement décrit sur internet).

Annexes

Network Diagram for physical working in lab



Network Diagram for remote working:



Configuration de Bind

/etc/named.conf

```
// named.conf
//
// Provided by Red Hat bind package to configure the ISC BIND named(8) DNS
// server as a caching only nameserver (as a localhost DNS resolver only).
//
// See /usr/share/doc/bind*/sample/ for example named configuration files.
//

options {
listen-on port 53 { 127.0.0.1; }; ATTENTION A LA CONFIGURATION ICI !
directory "/var/named";
dump-file "/var/named/data/cache_dump.db";
statistics-file "/var/named/data/named_stats.txt";
memstatistics-file "/var/named/data/named_mem_stats.txt";
allow-query { 10.32.1.0/24; any; }; ATTENTION A LA CONFIGURATION ICI !
forwarders { 192.168.1.1; }; ATTENTION A LA CONFIGURATION ICI !
recursion yes;
pid-file "/run/named/named.pid";
session-keyfile "/run/named/session.key";
};

logging {
channel default_debug {
file "data/named.run";
severity dynamic;
};
};
zone "." IN {
type hint;
file "named.ca";
};

include "/etc/named.rfc1912.zones";
zone "zX.rt.iut" in {
type master;
file "direct.zX.rt.iut";
allow-update{none;};
};
zone "X.1.10.in-addr.arpa" in {
type master;
file "reverse.zX.rt.iut";
allow-update{none;};
};
```

/var/named/direct.zX.rt.iut

// Faites attention à mettre un point à la fin des parties en jaune mises en évidence sinon le service nommé ne démarrera pas.

Ces fichiers sont des exemples. Faites attention aux adresses !

Les éléments surlignés en rouge sont importants, si vous avez des erreurs, vérifiez si vous les avez écrits.

@ est remplacé par le contenu de $ORIGIN, ici : zX.rt.iut.

```
; /var/named/direct.zX.rt.iut
$TTL  30
$ORIGIN zX.rt.iut.
@           IN    SOA    gwX.zX.rt.iut. root.gwX.zX.rt.iut. (
                                        200107011; Serial
                                        8H    ; Refresh
                                        2H    ; Retry
                                        1W    ; Expire
                                        1D   ; Minimum
                                         )
                          IN    NS    gwX.zX.rt.iut.
localhost     IN    A          127.0.0.1
gwX           IN    A            10.32.1.1
e-mail        IN    CNAME    pcX
smtp          IN    CNAME    pcX
IMAP          IN    CNAME    pcX
pop3          IN    CNAME    pcX
pcX         IN    A    10.32.1.254
```

/var/named/reverse.zX.rt.iut

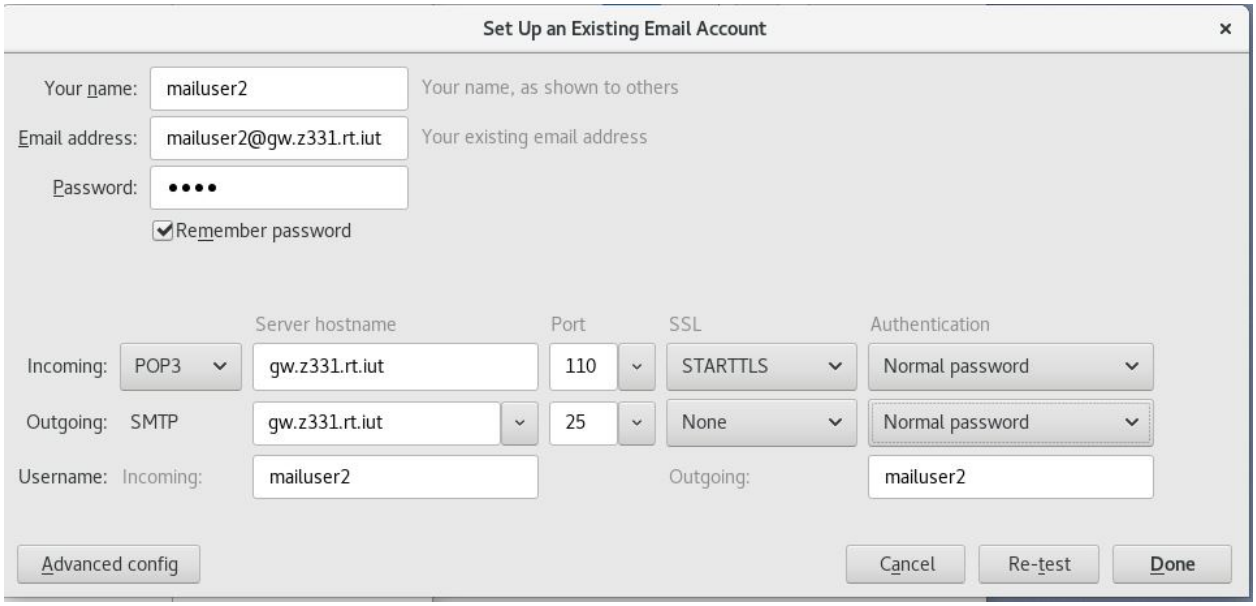
```
; /var/named/reverse.zX.rt.iut
$TTL  30
$ORIGIN 1.32.10.in-addr.arpa.  ; ATTENTION A LA CONFIGURATION ICI !
@                   IN      SOA   gwX.zX.rt.iut. root.zX.rt.iut. (
                                        1288639578; Serial
                                        8H; Refresh
                                        2H; Retry
                                        1W; Expire
                                        1D; Minimum
                                    )
                        IN      NS    gwX.zX.rt.iut.

254                 IN      PTR   gwX.zX.rt.iut.
1                   IN      PTR   pcX.zX.rt.iut.
```

Modifiez /etc/named.conf comme cela est recommandé.

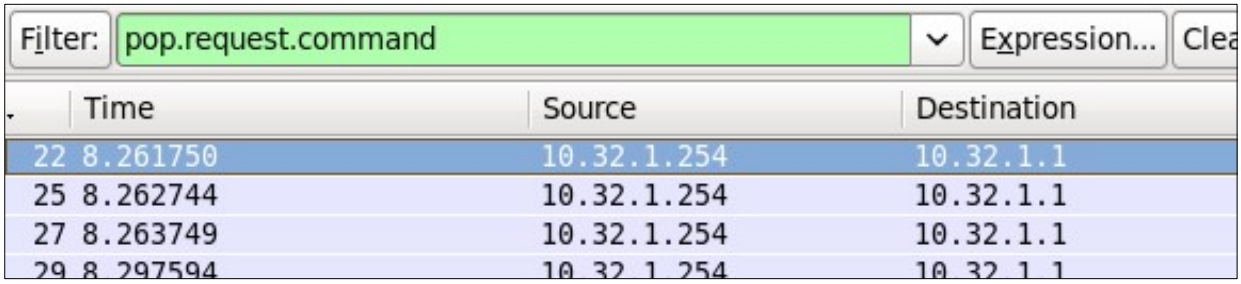
Modifiez /etc/sysconfig/named de manière à faire fonctionner named uniquement en IPv4, sinon il exécutera de nombreuses requêtes IPv6 (ce qui entraînera nombreuses erreurs dans les fichiers journaux)

OPTIONS="-4"

Thunderbird account settings (c'est simplement un exemple à adapter à notre TP) :



Example of wireshark capture.
/root/.ssh/authorized_keys



ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAtzgNCZhqVqdmRhCH6KCfbpoKL2dXK4vGjqT6ABT3LO2FMQKeKXvGOelZ
PupFU7pzessbAx9iWlpV24xSgvfYz2oYDEwR0yXdAvQSsUliZMuGZ/om+XgfUQQwILuhccvL6Ccx5k90UC5NKc5
gTXIlb8KhR4Rk8EmkD3CFuJh+RediY0sezlxgcGmPVdmVvV1YJ4LSPogPCRYojTbjPDECbLYQd/Z5GRsfFOCr4kYXx
05i9gInOET80Z8aplRX0qL2ou4ZrhOHgVxqTbFSEUbNp96BcspfogZu1xBG/KWjf9Z9DGb/pbsz/F5eol2EpmoZL8
tpj3uMVRt5C5DHSFIM4w== root@pcX.zX.rt.iut