TP3 - Étude et sécurisation de la messagerie électronique

Le courrier électronique est le premier service qui a été développé historiquement. Aujourd'hui, en 2024, plus de 361 milliards d'emails (ou courriels) seraient envoyés quotidiennement au niveau mondial, soit pas loin de 4,2 millions d'emails par seconde. Toutefois, on estime que la majorité de tous ces emais (voir la très grande majorité, avec des estimations d'environ 85%) de tous ces emails sont du SPAM (du courrier indésirable) ou malicieux. Par exemple, les défenses du service Gmail de Google, basées sur l'intelligence artificielle, empêchent plus de 99,9% des SPAMs, du hameçonnage (phishing) et des logiciels malveillants d'atteindre les boîtes de réception, bloquant pas loin de 15 milliards d'emails chaque jour.

Du point de vue protocolaire, dans un client de messagerie (par exemple outlook ou thunderbird) quand on configure un compte de messagerie électronique, auquel est associé une boîte aux lettres (BAL), il faut préciser deux serveurs : le serveur sortant qui à sert aux envois et l'entrant qui héberge la BAL. Pour chaque serveur il faut bien entendu préciser un identifiant et un mot de passe. Au niveau protocole, les envois se font via le protocole SMTP alors que l'accès à la BAL peut se faire via POP ou IMAP.

Comme d'habitude, il peut être nécessaire d'être connecté en super-utilisateur (`root`). Seules les grandes lignes des commandes seront décrites, pour avoir la syntaxe complète d'une commande on vous invite à utiliser le manuel : `man [commande]` (`[ ]` indique que c'est optionnel), exemple : `man ls`.

IMPORTANT : changer le mot de passe de root !!

1 Configuration réseau

Dans un premier temps, il s'agit de mettre en place une configuration réseau qui permettra d'atteindre le serveur de messagerie qui sera utilisé lors de ce TP.

ATTENTION : dans la suite, le caractère `X` correspondra au numéro de votre machine (écrit au crayon en bas à droite de l'écran), tandis que `Y` dénotera le numéro de la machine d'un(e) de vos camarades.

Les manipulations à effectuer sont :

1. se connecter en tant que `root` dans une console ou un terminal ;
2. arrêter le processus `dhclient` (il peut remettre le fichier `resolv.conf` d'origine) via la commande `kill -9` suivi du PID du processus à "tuer". Pour obtenir le PID de `dhclient`, il faut procéder comme suit :

   ```
   ps -aef | grep dhclient
   ```

3. modifier la configuration réseau de la machine pour que l'adresse IP de votre machine soit `172.20.20.X` avec comme préfixe `24` et pour passerelle la machine dont l'adresse IP est `172.20.20.254`. `ATTENTION` : vous devez utiliser la commande `ip` pour faire cela, voir-ci dessous, en vous basant sur les commandes vues lors du TP sur le routage. Vous devez ainsi ajouter deux commandes `ip` à la suite des deux commandes données sur la page suivante.

```
ifdown eno1
ip link set eno1 up
```

`ip addr` ... (à compléter)

`ip route` ... (à compléter)

4. modifier le fichier `/etc/resolv.conf` comme suit :

```
domain tpreseauX.org
search tpreseauX.org
nameserver 172.20.20.250
```

5. vérifier que le serveur de nom (DNS) utilisé est dorénavant une machine dans la salle de TP. Pour cela exécuter la commande ci-dessous (si elle est absente, installer le paquet `dnsutils`) et répondre aux questions posées :

```
nslookup tpreseau.tpreseau16.org
```

— Que fait cette commande ?
— Quelle est l'adresse IP du serveur de nom / DNS utilisé ?
— Le numéro de port réservé pour un serveur DNS ?

2 La messagerie électronique

Après une description sommaire du fonctionnement de la messagerie électronique, nous verrons comment configurer un client de messagerie qui sera en charge de la gestion du courrier électronique d'un utilisateur associé au domaine `tpreseau16.org`. Naturellement, tout le monde aura une adresse email différente, en l'occurrence de la forme `userX@tpreseau16.org`. Ensuite, vous étudierez le protocole utilisé pour envoyer des messages, ainsi qu'un protocole permettant de consulter une **B**oîte **A**ux **L**ettres distante.

2.1 Brève description de son fonctionnement

- **Itinéraire suivi par un message**
  — Considérons deux utilisateurs :
    — *bugs bunny* ayant `bugsbunny@acme.com` pour adresse électronique ;
    — *droopy* ayant `droopy@texavery.com` pour adresse électronique.
  — Les serveurs dont chacun dispose sont :
    — pour *bugs bunny* → `smtp.acme.com` et `pop.acme.com` ;
    — pour *droopy* → `smtp.texavery.com` et `pop.texavery.com`.
  — *Bugs bunny* envoie un message à *droopy*.
  La figure ci-dessous nous permet d'observer le cheminement du message.
  Remarques :
    — `BAL` dénote la boîte aux lettres ;
    — le serveur POP peut être remplacé par un serveur IMAP.

  Description du cheminement

  1. *bugs bunny* rédige un courrier élec. avec son client de messagerie préféré (ex : `thunderbird`) et clique sur `Envoyer` ;
  2. le serveur `smtp.acme.com` constate que le destinataire n'est pas dans son domaine :
     — recherche d'un serveur de messagerie du domaine `texavery.com` ;

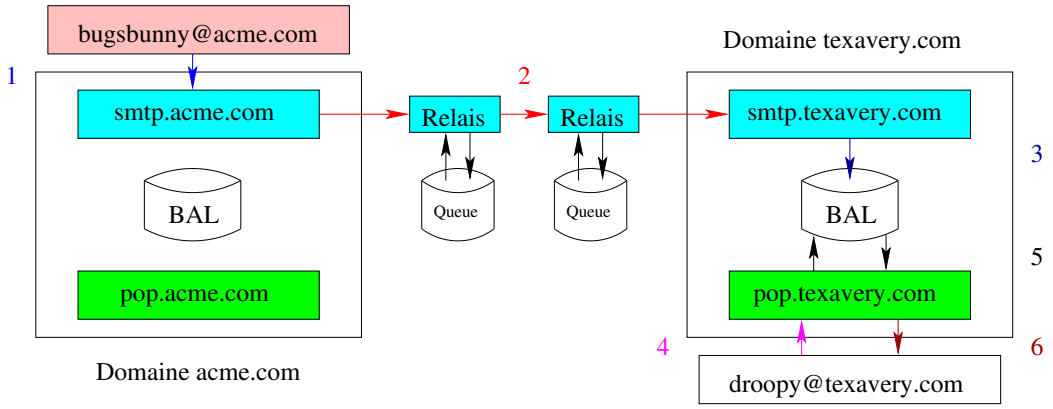

FIGURE 1 – Cheminement du message envoyé par `bugsbunny@acme.com`.

— utilisation du DNS qui fournit un enregistrement de type `MX` → adresse IP d'une machine hébergeant un serveur de messagerie ;
le message est alors envoyé à `smtp.texavery.com`

3. le serveur `smtp.texavery.com` constate que le destinataire fait partie de son domaine → message stocké dans la BAL ;
4. `droopy` consulte sa messagerie via son outil préféré qui interroge le serveur `pop.texavery.com` ;
5. le serveur `pop.texavery.com` consulte la BAL et constate la présence d'un message pour `droopy` ;
6. le message est envoyé au client de messagerie qui peut alors demander sa suppression (ou non) de la BAL

- **Protocoles mis en jeu**
  — Notion de port
  Plusieurs applications réseaux peuvent s'exécuter sur une machine, que celle-ci soit cliente ou serveur. Chaque application, de type client / serveur, utilise les services de la couche transport TCP ou UDP et est identifiée par un numéro de port du côté serveur. Ainsi, les applications `ssh` et `telnet` utilisent respectivement les numéros de port 22 et 23. Un port est représenté par une valeur entière sur 16 bits, ce qui implique un maximum de $2^{16} = 65536$ ports différents :
  — les ports 0 à 1023 sont les ports **réservés**. Ils sont spécifiés par l'***I**nternet **A**ssigned **N**umbers **A**uthority* et permettent d'accéder aux services standards : messagerie électronique (protocole SMTP, port 25), serveur web (HTTP, port 80), etc. ;
  — les ports suivants sont les ports **utilisateurs**. Ils sont disponibles pour des services applicatifs quelconques (jeu en réseau, partage de fichiers, etc.).
  La correspondance entre nom de service et numéro de ports est donnée dans le fichier `/etc/services`.
  — ***S**imple **M**ail **T**ransfer **P**rotocol*
  — Protocole de transport des messages sur l'Internet ;
  — serveur à l'écoute au port 25 ;
  — assure l'acheminement jusqu'à une BAL :

1. analyse la partie droite de l'adresse élec. du destinataire
2. si :
   - le domaine le concerne (courrier local) → recherche la BAL adéquate en

analysant la partie gauche de l'adresse ;
• le domaine ne le concerne pas → interroge le DNS pour obtenir l'adresse IP d'un serveur de messagerie le gérant

3. le message est :
• soit rangé dans la BAL ;
• soit le message est envoyé

— principales commandes du protocole

Commande	Description
`HELO <adresse IP>`	Ouverture, répond `OK`
`MAIL FROM:`	Spécification de l'adresse de l'expéditeur
`RCPT TO:`	Spécification de l'adresse du destinataire
`DATA`	Partie de l'en-tête et corps du courrier
`HELP`	Récupération de la liste des commandes
`QUIT`	Quitter la connexion en cours

— ***P**ost **O**ffice **P**rotocol*
— Protocole de dialogue entre un client de messagerie et une BAL ;
— serveur à l'écoute au port 110 ;
— accède à une BAL à la fois ;
— permet à l'utilisateur de gérer le courrier dans sa BAL ;
— ***I**nternet **M**essage **A**ccess **P**rotocol* est une alternative (protocole multi BAL)

Les manipulations à effectuer sont :

1. consulter le contenu du fichier `/etc/services` ;
2. installer le scanner de port `nmap` (à moins qu'il soit déjà installé)

```
apt update
apt install nmap
```

3. que vous apprennent les commandes suivantes ?

```
nmap 172.20.20.0/24
nmap -sV cours-info.iut-bm.univ-fcomte.fr
nmap localhost
host -t mx tpreseau16.org
```

4. ouvrir un nouveau terminal et se connecter via `ssh` avec le compte `userX` sur la machine `172.20.20.16`. Le mot de passe à utiliser est `CqriT` ;
5. lancer la commande suivante dans ce terminal

```
tail -f /var/log/mail.info
```

Celle-ci permettra d'observer ce que fait le serveur de messagerie ;

6. ouvrir un nouveau terminal et se connecter comme précédemment, puis envoyer un courrier électronique à un de vos camarades (`userY@tpreseau16.org`) avec `mail` comme cela :

```
$ mail userY@tpreseau16.org
Cc:
Subject: Messagerie locale
Hello World!!
```

```
How are you?
Good Bye
<Ctrl+D>
```

observer le journal dans le terminal où vous avez lancé `tail` (en principe tout devrait s'être bien passé, regarder plus particulièrement `message-id=`, `from=` et `to=`) ;

7. si quelqu'un vous a envoyé un message vous devriez le retrouver dans votre boîte aux lettres. Par exemple, un message de `user10` dans le BAL de `user14` donnerait :

```
$ mail
"/var/mail/user14": 1 message 1 nouveau
>N  1 User10              ven. avril 28 13  15/470   Messagerie locale
?
```

Vous pouvez consulter le manuel pour obtenir plus d'informations sur `mail`, ou accéder à l'aide avec `?` ;

8. quitter `mail` en tapant `q` ;
9. se déconnecter de la machine `172.20.20.16` ;
10. forger directement un courrier électronique via les commandes du protocole SMTP, en utilisant `telnet` pour vous connecter directement sur le port 25 du serveur de messagerie.

2.2 Configuration et utilisation du client de message Thunderbird

ATTENTION : tous ce qui suit doit être fait sur votre machine.

Nous allons maintenant utiliser `thunderbird` pour envoyer des messages. De plus, il sera configuré pour consulter la boîte aux lettres avec le protocole POP. Enfin, pour observer le déroulement de ce protocole, nous utiliserons l'analyseur de protocole `wireshark`.

Les manipulations à effectuer sont :

1. garder sur la machine `172.20.20.16` uniquement le terminal montrant avec `tail` le journal, fermer les autres connexions ;
2. vérifier que `thunderbird` est installé (sinon l'installer)

   ```
   apt install thunderbird
   ```

3. utiliser la commande `host -t mx tpreseau16.org` pour obtenir le nom du serveur de messagerie à utiliser lors de la configuration du compte de messagerie dans `thunderbird` ;
4. lancer `thunderbird`, puis dans `Set Up Your Existing Email Account` compléter les champs comme suit :
   — `Your full name` → `UserX`
   — `Email address` → `userX@tpreseau16.org`
   — `Password` → `CqriT`
   puis cliquer sur `Continue` ;
5. vérifier que les champs `Incoming`, `Outcoming` et `Username` sont bien remplis automatiquement

   Dans quel sens (*In* / *Out*) les communications seront-elles chiffrées ?
6. cliquer sur `Done`
7. une fenêtre avec en titre *Warning !* en blanc sur fond rouge s'est normalement affichée, quel est son rôle ?

8. mettre une croix dans `I understand the risks`, puis cliquer sur `Confirm` suivi de `Finish` ;
9. rédiger un email et essayer de l'envoyer à l'un de vos camarades via `Write` ;
10. un fenêtre s'ouvre, après lecture cliquer sur `OK`, puis `Confirm Security Exception`, l'envoi devrait alors pouvoir se dérouler correctement en recliquant sur `Send` ;
11. démarrer l'analyseur de protocole `wireshark`. Quel est le filtre de capture à utiliser pour capturer les messages du protocole POP ?
12. relever le courrier dans votre compte de messagerie (si besoin, le mot de passe à utiliser est celui du compte) ;
13. arrêter l'analyseur de protocole et observer le résultat. Vous devriez constater que votre identifiant et le mot de passe associé ont transités en clair vers le serveur de messagerie. Vous pourrez également observer les commandes du protocole POP envoyées par le client de messagerie ;
14. démarrer à nouveau l'analyseur afin de capturer l'envoi d'un email et constater la différence avec la relève du courrier.

3 Sécurisation des échanges avec `openssl` et S/MIME

L'utilisation de S/MIME suppose la création d'une paire de clés (publique, privée) et d'un certificat associé. D'autre part, ce certificat doit être délivré par une autorité de certification reconnue par le client de messagerie. Pour ce faire, nous utiliserons OpenSSL et nous générerons notre propre autorité de certification. De ce fait, nous créerons un certificat auto-signé.

OpenSSL est une boîte à outils cryptographiques, offrant notamment une commande en ligne (`openssl`) qui permet :

— de créer des clés RSA, DSA (signature) ;
— de créer des certificats X.509 ;
— le calcul d'empreintes ;
— le chiffrement et le déchiffrement ;
— la signature et le chiffrement de courrier.

Toutes les fonctionnalités sont décrites dans le manuel (`man openssl`), la syntaxe générale de la commande `openssl` étant :

```
openssl <commande> <options>
```

En principe, tous les paquets nécessaires sont déjà installés.

ATTENTION : vous ne devez pas essayer les exemples...

3.1 Gestion de sa paire de clés RSA

3.1.1 Génération d'une clé privée RSA

— Syntaxe :
`openssl genrsa -out <fichier_cle_privee> <taille>`
Une paire de clés est sauvegardée dans *fichier_ cles*, avec *taille* qui exprime la taille du modulus de la clé.

— Exemple :
`openssl genrsa -out maClePrivee.pem 4096`

3.1.2 Visualisation de la clé privée RSA

— Syntaxe :
`openssl rsa -in <fichier_cle_privee> -text -noout`

3.1.3 Protéger l'accès à sa clé privée en chiffrant le fichier

Après sa création, la paire de clés est en clair. Il est cependant plus prudent de la chiffrer. Plusieurs algorithmes symétriques sont disponibles pour cela dont : DES (option `-des`), DES3 (`-des3`) et IDEA (`-idea`).

— Syntaxe :
`openssl rsa -in <fichier_cle_privee> -<algo_sym> -out <fichier_cle_privee>`

— Exemple :
`openssl rsa -in meClePrivee.pem -des3 -out maClePrivee.pem`

3.1.4 Exporter la clé publique associée à la clé privée

La clé publique peut être communiquée à n'importe qui. Le fichier `maClePrivee.pem` contient la clé privée ne peut donc être communiqué.

— Syntaxe :
`openssl rsa -in <fichier_cle_privee> -pubout -out <fichier_cle_publique>`

— Exemple :
`openssl rsa -in maClePrivee.pem -pubout -out maClePublique.pem`

Les manipulations à effectuer sont :

1. créer une clé privée RSA avec pour nom de fichier `maClePrivee.pem` ;
2. visualiser votre clé ;
3. empêcher l'utilisation par quelqu'un de votre clé en chiffrant le fichier avec l'algo. `DES3` ;
4. exporter la clé publique associée à votre clé privée.

3.2 Chiffrement/déchiffrement et signature de fichiers

3.2.1 Chiffrement et déchiffrement de données avec RSA

— Syntaxe :
`openssl rsautl -encrypt -in <fichier_entree> -inkey <fichier_cle_privee> -out <fichier_sortie>`
avec *fichier_ entree* qui contient les données à chiffrer ; *fichier_ cle_ privee* est la clé privée ou alors la clé publique pour laquelle il faut ajouter l'option `-pubin` ; *fichier_ sortie* est le fichier chiffré. Pour déchiffrer, il suffit de remplacer *encrypt* par *decrypt*.
Dans le cas où on veut envoyer un fichier chiffré à quelqu'un, on utilise la clé publique de cette personne pour chiffrer, alors qu'elle utilisera sa clé privée pour le déchiffrer :

— chiffrement
`openssl rsautl -encrypt -in <fichier_entree> -pubin -inkey <fichier_cle_publique> -out <fichier_sortie>`

— déchiffrement
`openssl rsautl -decrypt -in <fichier_entree> -inkey <fichier_cle_privee> -out <fichier_sortie>`

RSA fait du chiffrement asymétrique, pour utiliser un chiffrement symétrique :

— chiffrement
`openssl enc -<algo> -in <fichier_entree> -e -out <fichier_sortie>`
— déchiffrement
`openssl enc -<algo> -in <fichier_entree> -d -out <fichier_sortie>`
algo désigne le cryptosystème symétrique choisi, par exemple `des3` (cf. `man openssl`).

Les manipulations à effectuer sont :

1. on crée un fichier avec `echo 'Un peu de blabla' > my_file` ;
2. chiffrer le fichier avec votre clé publique ;
3. le déchiffrer avec la clé privée correspondante.

3.2.2 Signature de fichiers

Uniquement de petits documents peuvent être signés directement, sinon il faut calculer une empreinte grâce à la commande `dgst` :

```
openssl dgst <hachage> -out <empreinte> <fichier_entree>
```

hachage désigne une méthode comme MD5 (`-md5`) ou SHA256 (`-sha256`).
— Syntaxe de la signature :
`openssl rsautl -sign -in <empreinte> -inkey <fichier_cle_privee>`
`-out <signature>`
— Syntaxe de la vérification de la signature :
`openssl rsautl -verify -in <signature> -pubin -inkey <fichier_cle_publique>`
`-out <empreinte>`
Il reste à comparer l'empreinte "signée" avec l'empreinte du fichier (appelée aussi le condensé). Par exemple avec la commande `diff`.

Les manipulations à effectuer sont :

1. signer le fichier `my_file` ;
2. vérifier la signature.

3.3 Certificats

Dans cette partie, vous allez tout d'abord générer une autorité de certification qui permettra de signer votre certificat. Ensuite, vous créerez un certificat pour votre adresse email, puis nous verrons comme utiliser tout cela dans thunderbird.

On suppose que vous disposez d'une paire de clés RSA dans le fichier `maClePrivee.pem`, protégée avec un mot de passe. Sinon, faire les manipulations des sections 3.1.1 et 3.1.3.

3.3.1 Création de l'autorité de certification

Les manipulations à effectuer sont :
Pour générer une autorité de certification avec

```
cd ~tpreseau
/usr/lib/ssl/misc/CA.pl -newca
```

Valider avec `enter`. Comme *pass phrase* vous utiliserez `CqriT`, concernant les informations demandées (`X` est toujours le numéro de votre machine) :

— Pays → `FR` ;
— Etat ou province → `FC` ;
— Ville ou localité → `Belfort` ;
— Organisation → `IUT-X` ;
— Unité → `Info-X` ;
— Nom → `userX` ;
— Email → rien.

Laisser les `'extra' attributes` vides. Ces informations pourraient être définies via un fichier de configuration. Si tout s'est bien passé, un répertoire `demoCA` a été créé dans le compte utilisateur où vous vous trouvez. Les fichiers dont nous aurons besoin dans la suite sont :

— `private/cakey.pem` → la paire de clés de l'autorité ;
— `crlnumber` → contient le numéro de série du prochain certificat à générer (le copier en `serial`) ;
— pour obtenir le certificat de l'autorité → `cacert.pem` (d'une validé de 365 jours)

```
cd ~
openssl req -new -x509 -days 365 -key ~/demoCA/private/cakey.pem -out ~/demoCA/cacert.pem
```

3.3.2 Création d'une requête de certificat

On utilise la commande `req`.

Les manipulations à effectuer sont :

1. `cd ~tpreseau` pour se placer dans le *home directory* de `tpreseau` ;
2. `mkdir my_certificate`, puis `cd my_certificate`
3. on crée la demande de certificat (la paire de clés est supposé être à la racine du compte)

   ```
   openssl req -new -key ../maClePrivee.pem -out maRequete.csr
   ```

   Reprendre les mêmes informations que précédemment pour générer la demande de certificat, en ajoutant l'adresse électronique `userX@tpreseau16.org`.
4. consulter le contenu de la demande via

   ```
   openssl req -in maRequete.csr -text -noout
   ```

3.3.3 Création et signature du certificat par l'autorité

Les manipulations à effectuer sont :

1. récupérer le certificat de l'autorité de certification (`cacert.pem`), sa paire de clés (`cakey.pem`), le fichier `serial`. Voici à titre d'exemple comment récupérer le premier fichier

   ```
   cp ../demoCA/cacert.pem .
   ```
2. regarder la durée de validité du certificat de l'autorité de certification

   ```
   openssl x509 -in cacert.pem -text -noout
   ```
3. créer le certificat

   ```
   openssl x509 -days 10 -CA cacert.pem -CAkey cakey.pem -CAserial serial
   -req -in maRequete.csr -out monCertificatX.crt
   ```
4. vérifier la validité de la signature du certificat

   ```
   openssl verify -CAfile cacert.pem monCertificatX.crt
   ```

5. exporter la clé publique, la clé privée et le certificat de la l'autorité de certification dans un fichier de type `pkcs12` utile pour thunderbird

```
openssl pkcs12 -export -in monCertificatX.crt -inkey ../maClePrivee.pem
-certfile cacert.pem -out monCertificatX.p12
```

3.3.4 Signature et chiffrement de messages électroniques

— Localisation des certificats
Vous pouvez consulter les certificats, en ajouter, etc. en sélectionnant le compte de messagerie, puis `Account Settings` en haut à droite, suivi de `End-To-End Encryption`.
— Importation des certificats
`Manage S/MIME Certificates` ouvre le gestionnaire de certificats, celui-ci comporte différents onglets :
  — `Your Certificates`
    — On y importe son fichier de type `pkcs12`.
  — `People`
    — On y importe le certificat d'un correspondant.
  — `Authorities`
    — On y importe le certificat d'une autorité de certification.

Les manipulations à effectuer sont :

1. lancer `thunderbird`, puis dans `Set Up Your Existing Email Address` compléter les champs `Your full name`, `Email address` et `Password` ;
2. importer votre fichier de type `pkcs12` et vérifier qu'`IUT-X` apparaît comme autorité de certification ;
3. demander à un(e) camarade de vous envoyer le certificat de son autorité de certification (son fichier `cacert.pem`), ainsi que son certificat (son fichier `monCertificatY.crt`) ;
4. importer le certificat de l'autorité de certification de votre camarade (son fichier `cacert.pem`) ;
5. importer le certificat de votre camarade (son fichier `monCertificatY.crt`) ;
6. dans l'onglet `Authorities`, éditer (`Edit Trust`) le certificat de chacune des deux autorités et cocher la case concernant l'acceptation de l'autorité pour les messages électroniques ;
7. vérifier la validité de votre certificat et de celui de votre camarade en les éditant ;
8. finalement, si vous avez tout fait correctement, vous devriez pouvoir envoyer et recevoir des messages cryptés et signés de / vers votre camarade (`userY@tpreseau16.org`).