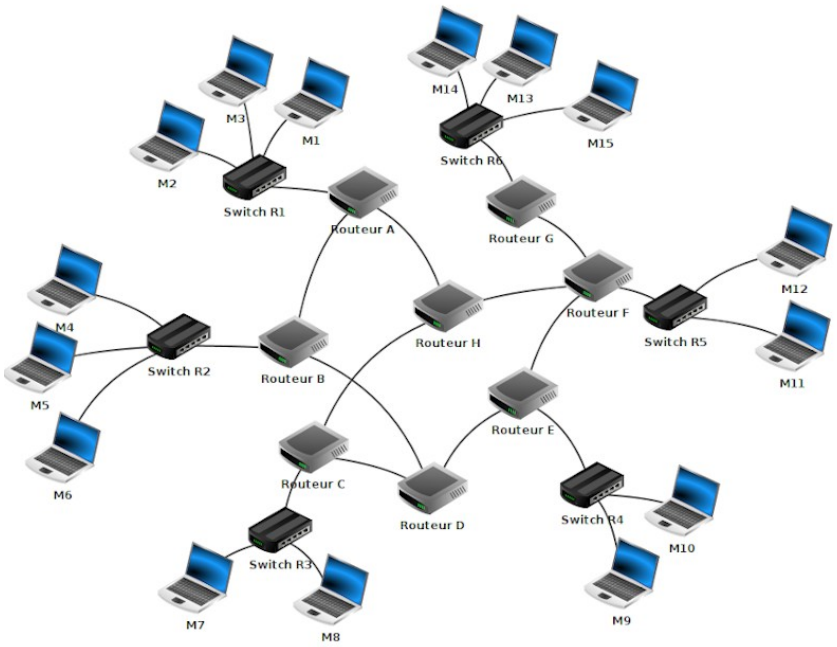
CH7 Routage et sécurisation des données



Dans ce réseau, on souhaite transférer un paquet depuis la machine M1 vers la machine M10... On constate qu'il y a ***beaucoup*** de chemins possibles !!

Pourquoi privilégier un chemin plutôt qu'un autre ? Comment diriger correctement et de façon sécurisée le paquet dans le réseau ?

Nous allons découvrir les 2 protocoles les plus utilisés qui permettent de répondre au problème (RIP et OSPF), ainsi que les méthodes de sécurisation.

Rappels sur les réseaux : https://youtu.be/s18KtOLpCg4

1 Le routage

→ **ACT1 Trouver son chemin** **TRAVAIL A RENDRE**

(a) Le rôle du routeur

Un paquet de données qui doit traverser un réseau n'a *a priori aucune idée* du chemin qu'il va devoir suivre. Il fait entièrement confiance aux indications fournies par les *routeurs*, espérant que celles-ci soient les plus fiables possibles.

La **box** domestique est un bon exemple de routeur qui possède plusieurs interfaces réseau :

- une interface est connectée au réseau de l'opérateur (FAI).
- une interface filaire (ethernet) connectée au réseau local
- une interface Wifi

Un **routeur sur internet** est un peu plus sophistiqué, possède souvent plus de ports et ressemble d'extérieur à un *switch*. Il dispose d'un logiciel interne bien plus sophistiqué afin de lui permettre de communiquer avec ses routeurs *voisins* pour l'aider à déterminer les meilleures routes à emprunter pour acheminer ses paquets.

Le routeur reçoit un paquet sur l'un de ses ports et sa mission consiste essentiellement à déterminer sur lequel de ses autres ports il doit réacheminer le paquet. Il doit le faire très vite et s'adapter à un environnement qui change (routes qui apparaissent et disparaissent).

(b) La table de routage

Vidéo explicative : https://youtu.be/sT9-IcbjqzI

A retenir :
On appelle le **protocole de routage** l'ensemble des règles qui permettent de trouver le chemin dans le réseau pour un paquet donné.
Pour choisir le bon chemin, le routeur s'appuie sur une **table de routage** : c'est une table donnant pour chaque *destination* connue par le routeur la porte de sortie à emprunter ainsi que l'efficacité de cette route. Cette efficacité sera exploitée dans les algorithmes spécifiques (voir plus loin).

Il existe 2 méthodes pour compléter la table de routage :

- le **routage statique**: chaque ligne doit être renseignée *à la main*. Cette solution est seulement envisageable pour des très petits réseaux de réseaux;
- le **routage dynamique**: tout se fait *automatiquement*, on utilise des protocoles qui vont permettre de *découvrir* les différentes routes automatiquement afin de pouvoir remplir la table de routage tout aussi automatiquement.

Voir https://www.it-connect.fr/routage-statique-et-routage-dynamique/

Exemple : Voici un table de routage simplifiée

Destination	Gateway	Port	Cost
137.194.2.0	137.194.4.254	eth1	2
137.194.4.0	137.194.4.253	eth2	2
137.194.4.8	137.194.4.251	eth3	11
137.194.4.192	0.0.0.0	eth0	1
137.194.6.0	137.194.4.254	eth4	2

Le routeur *sait* ainsi que pour atteindre la machine 137.194.2.21, il doit s'adresser au réseau 137.194.2.0/23 en redirigeant le paquet au routeur 137.194.4.254 qui est joignable sur l'interface eth1.

Dans des réseaux très complexes, chaque routeur aura une table de routage qui comportera de très nombreuses lignes (des dizaines voir des centaines...). En effet chaque routeur devra savoir vers quelle interface réseau il faudra envoyer un paquet afin qu'il puisse atteindre sa destination. Dans le cas où il existe plusieurs chemins possibles pour atteindre une même destination, le routeur va choisir le *chemin le plus court*.

2 Les protocoles RIP et OSPF

→ ACT2 La meilleure route TRAVAIL A RENDRE

Les deux protocoles dynamiques utilisés pour le transport de paquets dans les réseaux sont :
- **RIP** (*Routing Information Protocol*)
- **OSPF** (*Open Shortest Path First*)

(a) Le protocole RIP

La **métrique** utilisée dans un protocole de routage est une mesure de la *distance* qui sépare un routeur d'un réseau de destination) : c'est le nombre de sauts inter-réseaux (« hops » en anglais).

A Retenir :

Le **protocole RIP** s'appuie sur l'algorithme de Bellman-Ford, qui permet de calculer les plus courts chemins dans un graphe, par recherche de la ***distance minimale*** (ou métrique minimale).

Principe :
Chaque routeur envoie toutes les 30 secondes, à tous ses voisins (routeurs adjacents), un message contenant la liste de tous des réseaux qu'il connaît. Toutes les 30 secondes, les routeurs mettent ainsi à jour leur table de routage avec les informations reçues, notamment la métrique pour atteindre la destination.

Caractéristiques principales :
- La distance est calculée en nombre de sauts (« hops »)
- Chaque routeur n'a d'information que sur ses voisins (« next hop »)
- Distance maximale de 15 sauts (16 étant considéré comme l'infini)

Pour la métrique, voir https://fr.wikipedia.org/wiki/M%C3%A9trique_(routage)
Pour le protocole RIP, voir https://youtu.be/kzablGaqUXM

Exemple :

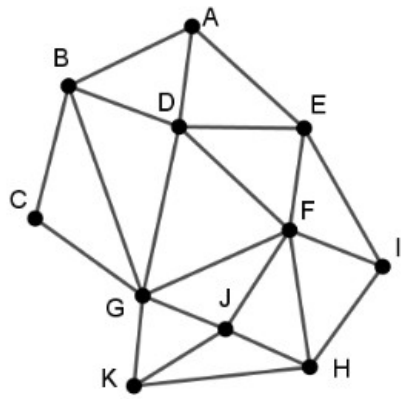


Table de routage du routeur A		
Destination	Routeur suivant	Métrique
B	B	1
C	B	2
D	D	1
E	E	1
F	D	2
G	D	2
H	D	3
I	E	2
J	D	3
K	D	3

On peut noter qu'il y a plusieurs trajets possibles, donc plusieurs tables de routage possibles. Si un routeur tombe en panne, l'information sera transmise dans les 30 secondes et les tables de routage seront mise à jour.

Le protocole RIP est rarement utilisé dans les réseaux de grande taille car l'envoi de messages génère un trafic important. On lui préfère souvent les protocoles IGRP (créé par CISCO) ou OSPF (voir ci-après). Les échanges sont plus « intelligents » dans ces protocoles, ils permettent donc de réduire l'occupation du réseau.

(b) Le protocole OSPF

Comme dans le cas du protocole RIP, il y a des échanges d'informations entre les routeurs pour les mises à jour du réseau, mais cette fois ce sont les valeurs de débits entre les routeurs qui sont mises à jour.

A Retenir :

Le protocole OSPF utilise la notion de ***coût des liaisons*** entre routeurs : plus le coût est grand et moins la liaison est intéressante). En additionnant les coûts de chaque liaisons, on obtient le ***coût total d'une route***.

Le coût est lié au débit des liaisons entre les routeurs. Le **débit d** est le nombre de bits de données qu'il est possible de faire passer dans un réseau par seconde.
Le **débit d** est donné en *bits par seconde* (bps).

La métrique s'écrit alors $cout = \frac{10^8}{d}$ Avec 1 Mbps = 10^3 kbps = 10^6 bps

Un lien de 10 Mbps aura par exemple un coût de 10.

Remarques :

- La valeur 10^8 a été choisie pour assurer un coût de 1 à une liaison *FastEthernet* à 100 Mbps
- Le débit est aussi appelé bande passante.

Exemple :

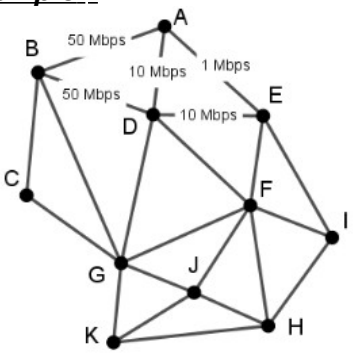


Table de routage du routeur A		
Destination	Routeur suivant	Métrique
E	E	100
E	D	10 + 10 = 20
E	B	2 + 2 + 10 = 14
D	D	10
D	B	2 + 2 = 4
...	...	...

1ère ligne de la table : A → E d_{A-E} = 1 Mbps donc coût = $10^8 / 10^6 = 100$

2nde ligne de la table : A → D → E d_{A-D} = 10 Mbps et d_{D-E} = 10 Mbps
donc coût = $10^8 / 10^7 + 10^8 / 10^7 = 20$

A Retenir :

Les étapes du protocole OSPF :

(1) Chaque routeur envoie un message court nommé « hello » à tout ses voisins toutes les 10s pour signaler sa présence et mettre à jour régulièrement la table de voisinage
(2) Ensuite chaque routeur envoie toutes ses tables à tous les routeurs du réseau, pendant cette phase chaque routeur met à jour sa table.
(3) Quand tous les routeurs ont leur table à jour, ils calculent chacun, en utilisant l'algorithme de Dijkstra, le chemin optimal vers chaque point du réseau.

Une conséquence importante, c'est que tant qu'il n'y a pas de changement de topologie du réseau (panne, ajout suppression de routeur ou de réseau), il n'y a plus d'échanges entre tous les routeurs du réseau.

Voir la vidéo pour le fonctionnement de OSPF https://youtu.be/-utHPKREZV8

3 Sécurisation et cryptage des communications

Comment se connecter au site https://gitlab.com/edaldegan/nsi_cours ?

1	L'utilisateur saisie l'adresse dans le navigateur.
2	Le navigateur demande au serveur DNS l'adresse IP du serveur web correspondant. (11112331)
3	Le navigateur établit une connexion TCP vers l'IP du serveur web sur le port 80.
4	La connexion étant établie, le client et le serveur échange des données.

Dans ce cas de figure, **sécuriser la communication**, c'est garantir que :
- *les échanges client-serveur ne sont lisibles que par la source et le destinataire.*
- *le serveur auquel le client se connecte est bien celui auquel on pense se connecter*

(a) Les méthodes de chiffrement

Pour assurer la sécurisation de la communication, on va pouvoir **chiffrer** la communication.

Un peu de vocabulaire :
- **Coder** (ou ***encoder***) = Représenter de l'information par un ensemble de signes prédéfinis.
- **Décoder** = Interpréter un ensemble de signes pour en extraire l'information qu'ils représentent.
- **Chiffrer** = Rendre une suite de symbole incompréhensible au moyen d'une ***clé de chiffrement.***
- **Déchiffrer =** Retrouver la suite de symbole originale à partir du message chiffré en utilisant la ***clé de chiffrement***. On parle de *Décrypter* lorsqu'on n'utilise pas la clé.

A Retenir :

Dans le cas où BOB envoie un message à ALICE, on retrouve les 2 méthodes de chiffrement utilisées :

- Le chiffrement symétrique

BOB et ALICE disposent tous les deux des fonctions suivantes :
C(*m, k*) est la fonction de chiffrement qui produit un message chiffré **s.**
D(*s, k*) est la fonction de déchiffrement qui produit le message en clair **m.**

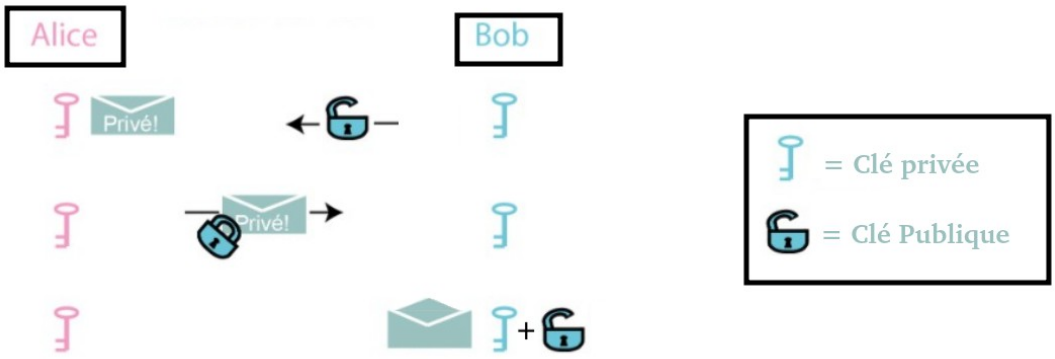
$$C : (m, k) \rightarrow s \qquad \text{et} \qquad D : (s, k) \rightarrow m$$

1 - *BOB utilise la fonction* ***C(m, k)*** *pour chiffrer le message m et envoie le résultat s à ALICE.*
2 - *ALICE utilise la fonction* ***D(s, k)*** *pour déchiffrer s reçu et retrouver le message m.*

Dans les 2 sens, on utilise **k** la clé de chiffrement, d'où le nom de *symétrique*.

- Le chiffrement asymétrique (1976, Diffie & Hellman)

BOB et ALICE disposent chacun de leur propres « cadenas + clés », un des « cadenas » va circuler sur le réseau : c'est la ***clé publique***. Une fois fermé, ce « cadenas » ne pourra s'ouvrir qu'avec la ***clé privée*** associée, qui elle n'a pas circulé sur le réseau.



1 - *BOB envoie à ALICE son cadenas (clé publique), il garde sa clé (clé privée).*

2 - *ALICE renvoie à BOB le message chiffré avec la clé publique. Personne n'en possède la clé excepté BOB.*

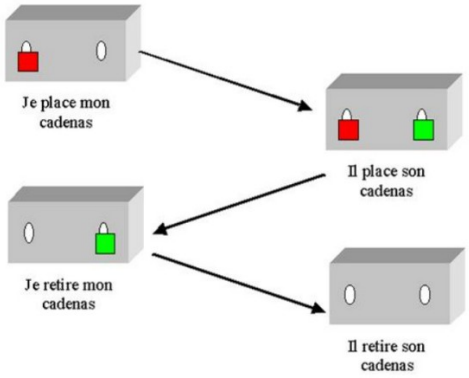
3 - *BOB reçoit le message chiffré et peut le déchiffre à l'aide de sa clé (clé privée).*

Exemples :

- Certains chiffrements symétriques

Nom	Type	Clé	Attaque possible
Chiffre de César (-50 av. J.-C.)	Monoalphabétique	Décalage de l'alphabet	- Force brute - Analyse fréquentielle
Chiffre de Vigenère (1586)	Polyalphabétique	Décalage différent pour chaque lettre	Cryptanalyse de Babbage(1854) et Kasiski (1863)
Chiffre de Vernam (1917)	Masque jetable	Aléatoire et aussi longue que le message	Incassable
AES (Advanced Encryption Standard) (2000)	Permutations itérées de blocs (10 à 14 tours)	Clés de 128 à 256 bits	Non cassé à ce jour

- Chiffrement asymétrique à cadenas :

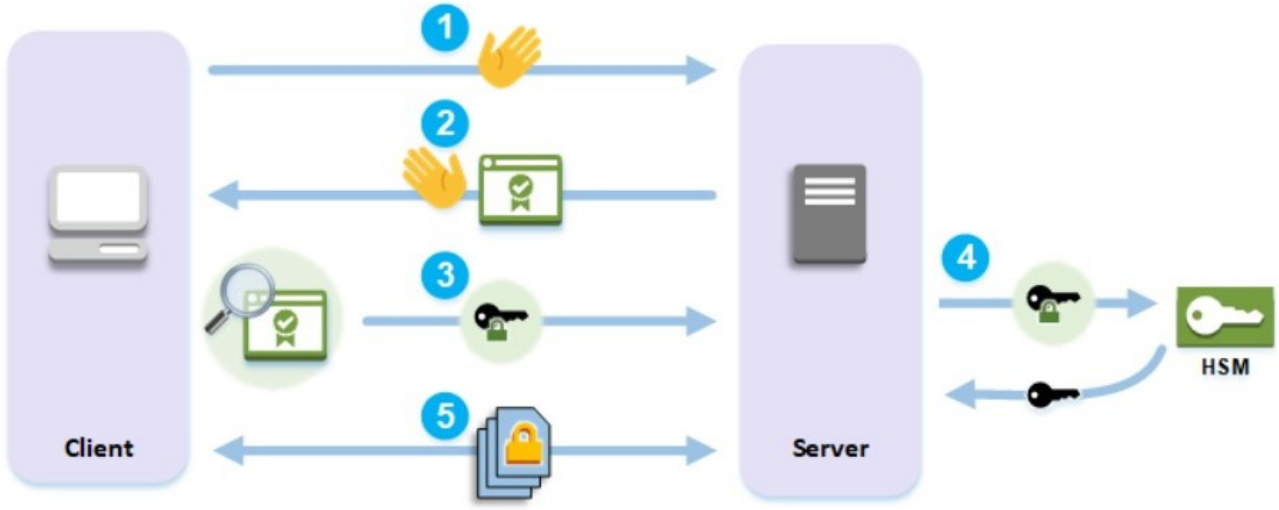


→ **ACT3 Chiffrer pour sécuriser TRAVAIL A RENDRE**

(b) Les protocoles HTTPS et TLS

La connexion au site web https://gitlab.com/edaldegan/nsi_cours utilise un protocole particulier : le HTTPS qui lui permet de sécuriser les échanges grâce à un certificat qui sert de à authentifier la source et le destinataire.

Pour établir une connexion HTTPS, votre serveur web effectue un processus de négociation avec les clients (RSA utilisé, voir https://fr.wikipedia.org/wiki/Chiffrement_RSA).
Dans le cadre de ce processus, le serveur décharge une partie du traitement cryptographique sur les HSM (Hardware Security Module).



1. Le client envoie un message Hello au serveur.
2. Le serveur répond par un message Hello et envoie son certificat de serveur.
3. Le client effectue les actions suivantes :

a. Il vérifie que le certificat du serveur SSL/TLS est signé par un certificat racine auquel il fait confiance.

b. Il extrait la clé publique du certificat du serveur.

c. Il génère un prémaster secret et le chiffre avec la clé publique du serveur.

d. Il envoie le prémaster secret chiffré au serveur.

4. Pour déchiffrer le prémaster secret du client, le serveur l'envoie au HSM. Le HSM utilise la clé privée dans le HSM pour déchiffrer le prémaster secret, puis il envoie le prémaster secret au serveur. Indépendamment, le client et le serveur utilisent tous les deux le prémaster secret et certaines informations issues des messages Hello pour calculer un secret principal.

5. Le processus de négociation se termine. Dans le reste de la session, tous les messages envoyés entre le client et le serveur sont chiffrés avec des dérivés du secret principal.

Autres sources :
https://www.cnil.fr/fr/comprendre-les-grands-principes-de-la-cryptologie-et-du-chiffrement
https://www.ssi.gouv.fr/particulier/bonnes-pratiques/crypto-le-webdoc/cryptologie-art-ou-science-du-secret/