Notes de cours
Semaine 7

Cours Turing

1 Cryptographie à clé secrète : un peu d'histoire

Dans ce premier cours sur la cryptographie, nous allons passer en revue quelques méthodes de chiffrement historiques, qui ne sont plus utilisées aujourd'hui, essentiellement à cause de la puissance des outils informatiques actuels, qui en permettent un décryptage quasi-instantané... Néanmoins, il est intéressant de se plonger dans les principes des premiers systèmes de cryptographie, car les systèmes modernes s'en inspirent fortement.

Tout au long de ce nouveau chapitre, trois personnages-clés vont nous accompagner : Alice, Bob et Eve, dans le scénario suivant : Alice cherche une méthode pour communiquer un message à Bob, tout en évitant qu'Eve, qui écoute leur conversation, soit capable de décrypter le message.

1.1 Préliminaire : l'addition modulo 26

Dans ce qui suit, nous allons manipuler l'alphabet latin, composé de 26 lettres, en nous restreignant pour simplifier aux majuscules. Sur cet alphabet, il est possible de définir une règle d'addition en identifiant chaque lettre à sa position dans l'alphabet (en commençant avec le A en position 0) :

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Ainsi, on a par exemple :

`A + A = A, A + B = B, B + B = C, B + C = D, ..., E + F = J, ...`

car

`0 + 0 = 0, 0 + 1 = 1, 1 + 1 = 2, 1 + 2 = 3, ..., 4 + 5 = 9, ...`

Mais alors, qu'advient-il par exemple de Y + D ou encore P + S ? En effet, 24 + 3 = 27 et 15 + 18 = 33, or il n'y a que 26 lettres dans l'alphabet latin. C'est pour cela qu'on définit une *addition modulo 26* sur l'ensemble des lettres, ce qui revient à dire que si le résultat de l'addition est plus grand que 25, on lui soustrait 26.

Exemples :

Y + D = B, car 24 + 3 = 27 et 27 - 26 = 1

P + S = H, car 15 + 18 = 33 et 33 - 26 = 7

L'avantage de cette règle est que l'addition de n'importe quelle paire de lettres redonne toujours une lettre.

1.2 Le chiffre de César (I^{er} siècle av. J.-C.)

La première idée pour chiffrer un message remonte à loin... Pour Alice, il s'agit simplement de choisir une des 26 lettres de l'alphabet, disons K pour l'exemple, et d'additionner celle-ci à chaque lettre du message qu'elle désire envoyer. Par exemple, pour chiffrer le message BONJOUR, elle effectue les additions :

```
   B  O  N  J  O  U  R
+  K  K  K  K  K  K  K
----------------------
=  L  Y  X  T  Y  E  B
```

et envoie donc le message chiffré LYXTYEB à Bob. Deux questions se posent alors :

1. Bob est-il capable de déchiffrer son message, et si oui, comment ?
2. Et si par hasard Eve intercepte le message envoyé par Alice, que peut-elle en faire ?

1. Pour que Bob puisse déchiffrer facilement le message transmis par Alice, il importe que celle-ci lui ait transmis *au préalable* la lettre K qu'elle a utilisé pour effectuer les additions (on appelle cette lettre la *clé secrète*). Bob peut alors effectuer les soustractions correspondantes (toujours modulo 26, ce qui veut dire qu'il rajoute 26 lorsque le résultat de la soustraction est plus petit que 0) pour retrouver le message d'origine :

```
   L  Y  X  T  Y  E  B
-  K  K  K  K  K  K  K
----------------------
=  B  O  N  J  O  U  R
```

Notez que Bob pourrait de manière équivalente *additionner* une lettre à chaque lettre du message reçu pour retrouver le message d'origine : voyez-vous laquelle ?

2. Que peut faire Eve de son côté si elle intercepte le message LYXTYEB ? Elle ne connaît pas la lettre secrète choisie par Alice, mais d'un autre côté, l'alphabet latin ne comporte que 26 lettres... Rien n'empêche donc Eve d'effectuer les mêmes soustractions que Bob, chaque fois avec une lettre différente, jusqu'à ce qu'elle tombe sur un message cohérent. Eve effectue ici ce qu'on appelle une *attaque par force brute*, ce qui veut dire qu'elle teste toutes les possibilités de chiffrement. Notez qu'avec beaucoup de malchance, il se pourrait qu'Eve tombe sur deux messages cohérents avec deux choix différents de lettre (et ne puisse donc pas décrypter le message d'origine), mais cette probabilité tombe rapidement à zéro pour de longs messages.

Alice et Bob doivent donc trouver autre chose pour assurer le secret de leurs communications...

1.3 Le chiffre par substitution (monoalphabétique)

Une variante du chiffre de César est la suivante : plutôt que d'additionner toujours la même lettre à chacune des lettres du message, Alice pourrait décider de substituer chaque lettre de l'alphabet avec une autre, par exemple :

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|	|
B	K	Z	Y	M	F	E	G	A	R	S	W	C	Q	J	O	N	V	T	H	L	I	D	U	X	P

Remarques :

- Certaines lettres peuvent rester inchangées avec ce système, comme ici le F.

- Le chiffre de César est un cas particulier du chiffre de substitution.

En utilisant ce tableau (de haut en bas) pour chiffrer le message BONJOUR, Alice obtient le message chiffré KJQRJLV, qu'elle envoie à Bob.

Comment Bob fait-il dans ce cas pour déchiffrer le message d'Alice ? A nouveau, il est nécessaire qu'Alice lui ait communiqué au préalable une clé secrète, qui n'est ici rien d'autre que le tableau ci-dessus (dans son intégralité). A l'aide du tableau, Bob peut retrouver le message d'Alice en utilisant celui-ci de bas en haut.

Se pose maintenant à nouveau la question de savoir ce qu'Eve peut faire si elle intercepte le message KJQRJLV ? La première chose à remarquer est qu'ici le nombre de clés possibles est *beaucoup* plus élevé qu'avec le chiffre de César ! Ce nombre est égal au nombre de permutations possibles des 26 lettres de l'alphabet, qui vaut $26 \cdot 25 \cdot 24 \cdots 3 \cdot 2 \cdot 1 = 26! \simeq 4 \cdot 10^{26}$. Il est donc complètement impossible pour Alice de tester toutes les possibilités en un temps raisonnable, même avec le plus performant des ordinateurs actuels... Une attaque par force brute est infaisable ici (et serait du reste bien inutile : voyez-vous pourquoi ?).

Alice et Bob auraient-ils donc trouvé le système de chiffrement idéal ? Pas si sûr... En effet, si le message envoyé par Alice est relativement long, une attaque possible d'Eve, dite aussi *attaque par analyse fréquentielle*, est la suivante : il s'agit de compter le nombre d'occurences de chaque lettre dans le message chiffré et de diviser ce nombre par le nombre total de lettres du message, pour finalement obtenir la fréquence d'apparition de chaque lettre dans le message.

Or il se trouve que dans un long texte en français, la fréquence d'apparition de chaque lettre est relativement stable et surtout connue : par exemple, le E apparaît beaucoup plus souvent que toutes les autres lettres, et les lettres A, I, S apparaissent beaucoup plus fréquemment que X, Y ou Z, par exemple. Ainsi, en calculant les fréquences d'apparition de chaque lettre dans le message chiffré, et en comparant celles-ci aux fréquences connues des lettres en français, Eve peut essayer d'établir une correspondance pour retrouver quelle lettre a été remplacée par quellle autre. Bien sûr, ce système n'est pas parfait, mais une fois que certaines lettres ont été identifiées (surtout les voyelles), il devient relativement facile pour Eve de deviner les autres lettres par déduction, en identifiant certains mots, et ainsi de retrouver le message d'origine.

Caramba, encore raté ! Essayons autre chose...

1.4 Le chiffre de Vigenère (XVI^{e} siècle)

L'idée suivante est de choisir non pas une lettre, comme dans le chiffre de César, mais une suite de lettres, comme par exemple la suite "MDR", et d'utiliser celle-ci de manière répétée pour chiffrer un message. Voici ce que cela donne sur un exemple :

```
   B  O  N  J  O  U  R  A  L  A  N  T  U  R  I  N  G
+  M  D  R  M  D  R  M  D  R  M  D  R  M  D  R  M  D
-------------------------------------------------
=  N  R  F  V  R  L  D  D  C  M  Q  K  G  U  Z  Z  J
```

Remarques :

- On a évité d'utiliser des espaces ici pour se restreindre aux 26 lettres latines, mais il est tout à fait possible d'ajouter l'espace à l'ensemble des lettres utilisées.

- Cette méthode est appelée un chiffrement par substitution *polyalphabétique*, car contrairement au cas précédent, une lettre donnée n'est pas systématiquement remplacée par une même autre lettre ; la substitution s'applique à des motifs de 3 lettres consécutives.

Muni de la clé secrète MDR, Bob est capable de déchiffrer le message reçu en effectuant les soustractions correspondantes (il faut certes qu'Alice et Bob s'accordent bien sur la position du début du message). Ce système a clairement l'avantage de la simplicité.

Et si Eve intercepte le message chiffré, que peut-elle faire ?

- Une attaque par force brute ?

Si la clé secrète comporte d lettres, le nombre de clés possibles vaut $\underbrace{26 \cdot 26 \cdot 26 \cdots 26}_{d \text{ fois}} = 26^d$,

ce qui grandit (très) vite avec d (p. ex., si $d = 10$, alors $26^d \simeq 10^{14}$). Il faut essayer autre chose...

- Une attaque par analyse fréquentielle ?

Comme mentionné plus haut, ce système de chiffrement a pour propriété qu'une lettre donnée n'est pas systématiquement remplacée par un même autre lettre. Une analyse naïve des fréquences des lettres dans le message chiffré ne donne donc rien ici. Mais une analyse des

fréquences des motifs de d lettres consécutives pourrait quant à elle donner quelque chose. Le problème pour Eve est qu'elle ne connaît même pas la longueur d de la clé...

Ceci dit, au XIX^{e} siècle (donc trois siècles plus tard...), Friederich Kasiski publie un test qui pemet de deviner la valeur de d ; ce test met fin à l'utilisation du chiffre de Vigenère.

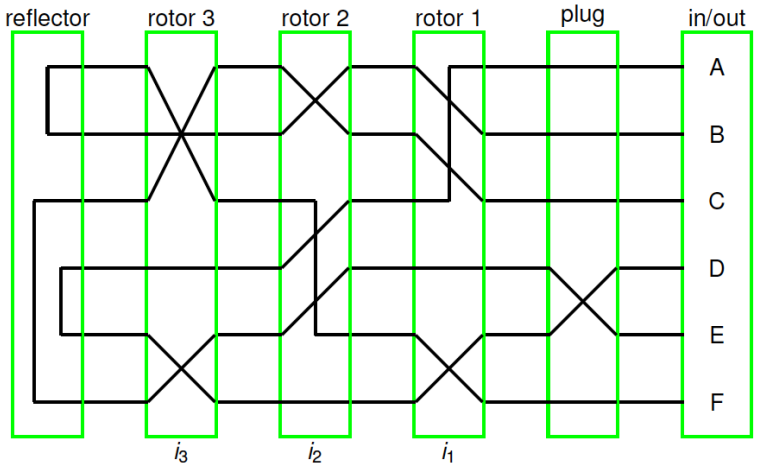
1.5 Enigma, Alan Turing et la naissance de l'informatique moderne

Le dernier exemple de système cryptographique que nous allons passer en revue aujourd'hui est celui de la machine Enigma, inventée à la fin de la première guerre mondiale et utilisée intensivement par l'armée allemande lors de la seconde. Vous en voyez ci-dessous un modèle, exposé au Musée de la Science et de la Technologie de Milan :

Le principe de base de la machine est le suivant : celle-ci est composée (entre autres) de deux claviers, et chaque frappe d'une touche du clavier du bas déclenche l'allumage d'une autre touche du clavier du haut, par un système de connexions électriques que nous allons détailler. Par exemple, pour un réglage donné, en écrivant le mot BONJOUR sur le clavier du bas, on voit s'allumer la séquence de lettres AIOZAPN sur le second clavier.

Rien qu'avec cet exemple, on voit que le chiffrement à l'œuvre ici est plus qu'un simple chiffre par substitution monoalphabétique : la lettre O est une fois remplacée par la lettre I et une autre fois par la lettre A. De même, le A dans le mot chiffré remplace une fois la lettre B et une fois la lettre O. La force du système réside dans l'introduction de 3 rotors, chacun équipé de son propre système de connexions électriques, qui tournent au fur et à mesure que le message est écrit ; ainsi, *à chaque lettre frappée sur le clavier, un autre chiffrement par substitution est utilisé !*

Voyons plus en détail comment le système fonctionne sur le schéma suivant, qui suppose pour simplifier que le clavier ne comporte que les 6 lettres A, B, C, D, E et F :



Lorsqu'une lettre est frappée sur le clavier à droite (disons la lettre A), un signal électrique est envoyé à travers les connexions successives des 3 rotors (laissons tomber pour l'instant l'étape "plug" ; nous y reviendrons), pour finir dans le réflecteur sur la gauche, qui renvoie à son tour le signal à travers les trois rotors, et finit par illuminer la lettre D. Comme déjà mentionné, avant que la prochaine lettre ne soit frappée, les rotors changent de position, ce qui change les connexions ci-dessus.

L'avantage d'introduire un réflecteur est de rendre le système symétrique : ainsi, pour déchiffrer le message reçu, il suffit de placer les rotors dans la même position initiale que lors du chiffrement, et d'écrire le message chiffré sur le clavier du bas pour voir s'allumer le message d'origine sur le clavier du haut. Vous pouvez essayer par vous-même sur le site web ci-dessous, qui offre une simulation du comportement de la machine :

`https://www.101computing.net/enigma-machine-emulator/`

Reste à expliquer un dernier détail, à savoir ce que signifie le "plug" ci-dessus. Ceci fait référence au *tableau de connexions* que vous voyez aussi au bas de l'image de la page précédente : ce tableau permettait de relier deux lettres entre elles par un fil électrique pour créer une permutation de celles-ci. Ainsi, dans le schéma ci-dessus, lorsque la lettre E est frappée au clavier, le tableau de connexions effectue d'abord une permutation avec la lettre D, puis le signal suit les connexions pour finir par atteindre la lettre C. Sans cette permutation initiale, la lettre E serait chiffrée par la lettre A. En connectant de nombreuses paires de lettres ensemble (ce nombre de paires pouvait aller jusqu'à 10), le chiffrement gagne grandement en complexité, comme nous allons le voir. Bien sûr, il est alors nécessaire d'effectuer les mêmes connexions pour déchiffrer le message. Vous pouvez aussi ajouter de telles connexions sur la machine simulée.

Forces et faiblesses d'Enigma

Voyons tout d'abord quelles sont les forces de ce système. La principale force est le nombre de combinaisons offertes par un tel système !

1. Les rotors : dans la version de base, il existe 5 rotors différents, dont 3 sont choisis pour être placés dans la machine (l'ordre choisi importe), et chaque rotor a autant de positions possibles que de lettres dans l'alphabet, soit 26. Ceci donne lieu en tout à

$$5 \cdot 4 \cdot 3 \cdot 26 \cdot 26 \cdot 26 \simeq \text{un million de combinaisons possibles } (10^6)$$

2. Le tableau de connexions : le fait de pouvoir choisir 10 paires de lettres à permuter dans le tableau de connexions donne lieu quant à lui à un nombre encore plus grand de combinaisons, environ égal à $1,5 \cdot 10^{14}$.

Donc au total, le nombre de combinaisons de la machine est de l'ordre de $1,5 \cdot 10^{20}$; un chiffre absolument faramineux ! Le système de chifffrement semble juste parfait...

Oui, mais... Plusieurs défauts se sont aussi révélés au fil du temps (sans entrer dans trop de détails) :

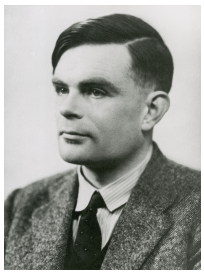
1. Le premier défaut a été de croire que le fonctionnement de la machine resterait caché des Alliés, ce qui ne fut pas le cas. Les Français, avec l'aide des Polonais, interceptent des plans de cablâge de la machine dès 1933, et c'est le mathématicien polonais Marian Rejewski qui parvient le premier à reproduire le fonctionnement de la machine.

2. Pour chaque jour, il faut que les unités communiquant entre elles se mettent d'accord sur la position des rotors et du tableau de connexions à adopter : ces positions sont consignées dans de gros cahiers transportés dans chaque unité. Certains de ces cahiers furent aussi interceptés par les Alliés pendant la guerre.

3. Dans les messages échangés entre les unités, de nombreux mots reviennent fréquemment, comme des formules de salutation au début et à la fin des messages, ou des bulletins météo. L'utilisation de ces messages répétés a pu être utilisée pour le décryptage de la machine. De plus, la fatigue et les conditions difficiles d'utilisation ont mené à certaines erreurs de transmission, ce qui a laissé échapper de précieuses informations.

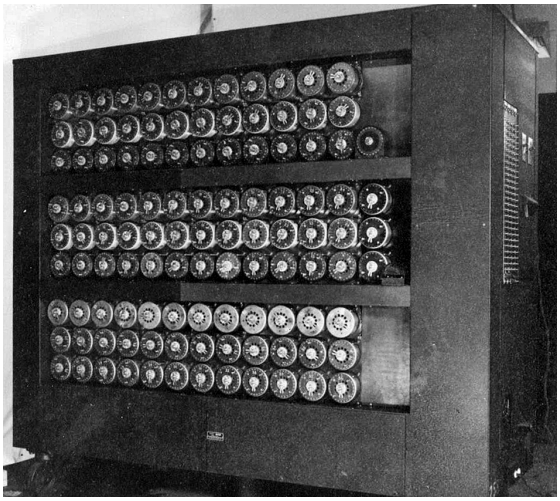
4. L'utilisation du réflecteur, si elle permet un déchiffrage aisé, a aussi pour défaut qu'une lettre n'est jamais chiffrée par elle-même, ce qui réduit (marginalement) le nombre de possibilités.

C'est en utilisant ces diverses faiblesses (entre autres) que l'unité travaillant en secret à *Bletchley Park* en Angleterre, emmenée par *Alan Turing*, a pu venir à bout du chiffrement d'Enigma.



La “bombe” d’Alan Turing

Encore une fois sans entrer dans trop de détails, mentionnons juste ici un élément clé du décryptage d’Enigma : Alan Turing et son équipe, au prix de nombreuses astuces, élaborent une méthode qui permet de faire abstraction du tableau de connexions pour le décryptage. Ainsi, le nombre de combinaisons possibles, au lieu d’être de l’ordre de 10^{20}, est ramené à un million, ce qui reste un très grand nombre de combinaisons, mais pas insurmontable... Pour essayer toutes les combinaisons possibles, Alan Turing construit une machine, qu’il appelle la *bombe*, qui permet de tester toutes ces combinaisons de manière systématique :



Sur la photo ci-dessus, vous voyez que la bombe est composée de plusieurs copies des 3 rotors d’Enigma. En un temps raisonnable, il est ainsi possible de parcourir toutes les combinaisons jusqu’à trouver la bonne.

Avec cette machine, le premier ordinateur est né : le reste appartient à l’histoire...

Les principes de Kerckhoffs et la cryptographie moderne

Mentionnons finalement une leçon retenue de cette épisode : en 1883, *Auguste Kerckhoffs* avait édicté quelques principes que tout cryptosystème devrait vérifier pour être efficace et fiable :

Le premier principe disait qu’il ne fallait pas que la sécurité du système repose sur le *secret du système en lui-même*. Ce premier principe n’a pas été respecté par Enigma... avec le résultat que l’on sait. Dans les prochaines semaines, nous verrons des systèmes de cryptographie moderne qui respectent ce premier principe !