Atelier 2 : Éléments de cryptographie

Cycle d'ateliers Internet et vie privée

27 juin 2009

Introduction

Un peu d'histoire

La cryptographie, selon le Larousse, est « l'ensemble des techniques permettant de protéger une communication au moyen d'une écriture conventionnelle secrète ». Elle est utilisée depuis longtemps, et vous même avez sans doute déjà joué avec. Il y a plus de 2000 ans, César utilisait déjà le « chiffre de César », qui consiste à décaler les lettres d'un message d'un nombre de lettres prédéfini.

> Par exemple, en décalant les lettres d'un cran, `Bonjour` devient `Cpokpvs`.

La cryptographie a été beaucoup utilisée pour des applications militaires à travers l'histoire. Pendant la seconde guerre mondiale, les allemands utilisaient la machine Enigma pour chiffrer leurs communications. En Angleterre, Alan Turing (mathématicien anglais, un des fondateurs de l'informatique), participe à casser ce code, en utilisant l'un des premiers ordinateurs.

Jusqu'en 1999 en France, la cryptographie est assimilée à une arme de guerre, et son utilisation est interdite (seule les techniques que les autorités peuvent casser sont permises). Puis, pour permettre au commerce électronique de se développer, elle devient autorisée. En revanche, son utilisation peut être considérée comme une « circonstance aggravante ». De plus, le fait de refuser de déchiffrer des messages secrets échangés peut entraîner une condamnation à deux ans de prison et 30 000 € d'amende, cette peine ayant été augmentée à trois ans d'emprisonnement et 45 000 € d'amende en 2003 dans le cadre de la Loi sur la Sécurité Intérieure.

Remarque Si la preuve que j'avais préparé un attentat se trouvais dans un courriel que la police me sommait de déchiffrer, je pense que je préférerais passer trois en en prison pour avoir refusé de donner la clef, que ma vie en prison pour avoir préparé un attentat.

Remarque Ces lois sont relativement jeunes, et nous ne savons pas exactement comment elles sont appliquée, puisque qu'il n'y a pas vraiment de jurisprudence à ce sujet.

Performances

Fiables ?

Les techniques anciennes de cryptographie (comme le chiffre de César) sont pour la plupart facile à casser. En revanche, les techniques modernes, que je vais vous présenter, sont considérées comme « fiables ».

Qu'est-ce que j'entends par fiable ?

- Premièrement, elles sont fiables **aujourd'hui**. Il est possible que demain, quelqu'un trouve une manière de casser ces méthodes. Dans ce cas, il pourra être capable, s'il a espionné des conversations chiffrées, de les décrypter.
  Par exemple, la protection des réseaux Wi-Fi nommée *WEP* était considérée comme fiable à sa création. Nous avons cassé une telle protection devant vos yeux la semaine dernière.
- Elles sont fiables si elles sont correctement utilisées : une porte blindée est inutile si je laisse ma fenêtre ouverte, et un gilet pareballes ne protège pas grand'chose s'il n'est pas fermé.
- Ces techniques sont fiable dans le sens où en théorie, il est possible de décrypter un mes-

sage chiffré sans posséder la clef, mais en pratique, le temps que ça prendrait se compte en années, milliers d'années, voire milliards d'années.

Source de fiabilité

Principe de Kerckhoffs Une des forces de ces techniques est qu'elles sont connues de tous : c'est ce qu'on appelle le principe de *Kerckhoffs*. Par exemple, la technique RSA, que je vais vous présenter, est connue et utilisée depuis 25 ans. Avec quelques connaissances en programmation, il est possible pour n'importe qui de faire un logiciel de cryptage utilisant ces techniques.
Cette méthode étant connue de tous, les spécialistes du monde entier ont pu s'y intéresser, et chercher comment la contourner, sans succès jusqu'à présent.
Si tout le monde a accès au mode de fonctionnement de ces techniques, d'où vient leur force ?

Factorisation d'entiers Cette force vient de problèmes mathématiques extrêmement difficiles à résoudre. Le RSA, par exemple, repose sur la « factorisation d'entiers ».
Étant donné le nombre 12, il est simple de le décomposer en $2 \times 2 \times 3$. De même, 111 est égal à 3×37. En revanche, comment décomposer le nombre suivant, composé de 221 chiffres ?

1 063 318 849 029 258 264 129 789 516 630 408 681 136 350 112 530 003 822 078 103 710 474 892 345 979 297 145 692 825 356 715 266 418 834 032 875 960 755 421 136 105 760 420 828 439 724 683 185 030 654 150 581 426 128 712 741 255 715 005 059 972 996 965 252 025 217 012 670 940 163 394 764 801

Le résultat et le produit des deux nombres premiers composés respectivement de 102 et 120 chiffres. Avec nos ordinateurs actuels, ce calcul serait beaucoup plus long qu'une vie humaine.

Ce problème de factorisation de entiers est étudié depuis plus de 2000 ans par les mathématiciens, et nous n'avons toujours pas trouvé de solution pratique. La cryptographie RSA, par exemple, repose sur ce problème.

Notons que si un jour une personne résout le problème, il sera possible, si les échanges chiffrés ont été enregistrés, de les déchiffrer sans trop problèmes, ce qui signifie que beaucoup de secrets militaires et commerciaux seront révélés à tous : en d'autres termes, une sacrée pagaille.

Chiffrements

But du chiffrement

Nous avons vu dans le précédent atelier qu'il est relativement simple pour quelqu'un de mal intentionné de lire les communications échangées par Internet entre deux personnes, ou d'usurper l'identité de quelqu'un.

Dans la suite de l'atelier, nous allons étudier des chiffrements, avec pour but de sécuriser des communications entre deux interlocuteurs. Pas « sécuriser », j'entends : s'assurer que personne d'autre que le destinataire du message n'est en mesure de le lire, et assurer le destinataire que personne ne se fait passer pour l'expéditeur pour délivrer le message. Ceci peut avoir pour cadre un échange de courriels, ou la consultation d'un site web.

Chiffrement symétrique

La cryptographie symétrique est un type de méthode de cryptographie dans lequel expéditeur et récepteur partagent une clef qui permet de chiffrer et déchiffrer le message. Par exemple, le chiffre de César, cité en introduction, est symétrique.

> Supposons que Bob veuille envoyer un message chiffré à Alice. Tous deux ont convenus d'une clef : le chiffre 2.
> Bob écrit son message `Bonjour`, qu'il chiffre avec cette clef en utilisant le chiffre de César (décalage de deux lettres) : `Dqplqwt`.
> Il envoie ce message chiffré à Alice, qui réalise l'opération inverse : à partir du message `Dqplqwt` qu'elle a reçu, elle décale les lettres de deux lettres en arrière, et obtient `Bonjour`.

Nous n'étudierons pas dans le détail la technique de ce genre de chiffrement, mais il est important de noter que des méthodes a priori fiables (avec toutes les restrictions que j'ai appliquées à ce mot) sont connues et utilisées.
Un avantage de ces méthodes est qu'il est rapide de chiffrer ou déchiffrer un message (contrairement au RSA que nous verrons dans la partie suivante).
Comme inconvénients, nous pouvons citer :

- Il faut une nouvelle clef pour chaque couple émetteur/récepteur : si je veux pouvoir échanger des messages cryptés avec ma sœur d'une part, et un ami d'autre part, j'ai besoin d'une clef pour chaque, sans quoi ma sœur pourra lire les messages que j'échange avec mon ami, et inversement.
- Il faut que l'émetteur et le récepteur puisse s'échanger de manière sûre cette clef.

Concrètement, ce chiffrage peut être utilisé en complément d'un chiffrement asymétrique que nous allons voir maintenant.

Enfin, notons que c'est ce chiffrement symétrique qui est utilisé pour chiffrer des données (par exemple pour protéger tout le disque dur d'un ordinateur), pour éviter que d'autres personnes y aient accès sans notre accord.

Chiffrement asymétrique

Le principe du chiffrement asymétrique est qu'il existe deux clef : une pour chiffrer le message, une pour le déchiffrer.

> Un cadenas illustre ce principe. Pour chiffrer, il suffit de fermer le cadenas à la main. Pour déchiffrer, il est nécessaire de posséder la clef.
> Ce chiffrement est dit « asymétrique » car une information différente est nécessaire pour chiffrer et déchiffrer.

Comme nous l'avons déjà expliqué, les bases mathématiques de cette méthode de chiffrement sont la factorisation d'entiers, qui est un problème extrêmement difficile, du moins pour le moment...
Dans le cas du RSA, la clef publique est un couple de nombres qui est utilisé pour appliquer une opération mathématiques aux données à chiffrer. La clef privée est un autre couple qui permet d'appliquer la transformation inverse.

Application au chiffrement

Nous appliquons ce chiffrement à l'envoi d'un courriel chiffré : ici nous voulons être capables d'envoyer un courriel que seul le destinataire est capable de lire.

Première version Voici par un exemple une version simplifiée du fonctionnement.

Bob souhaite envoyer un message chiffré à Alice, de manière à ce que personne d'autre qu'Alice ne puisse lire le contenu.
1. Alice génère une paire de clés (cadenas ouvert = clef publique, clef = clef privée). Elle met sa clef publique (cadenas ouvert) à disposition de tous.
2. Bob chiffre son message avec cette clef publique.
3. Bob envoie son message chiffré.
4. Alice déchiffre le message avec sa clef privée.

Si quelqu'un d'extérieur à ces deux personnes intercepte le message (ce qui est très simple), et souhaite le lire, il a besoin de la clef privée d'Alice. Il ne peut donc pas le déchiffrer.

Fonctionnement réel C'était une version simplifiée. En pratique, chiffrer le message complet avec ce genre de chiffrement est trop long.

Ainsi, ce qui se passe en pratique, est que les deux interlocuteurs s'échangent une clef de chiffrement symétrique, en utilisant le chiffrement asymétrique.

1. Bob génère une clef pour faire du chiffrement symétrique.
2. Bob chiffre cette clef avec la clef publique d'Alice. Il obtient une clef chiffrée.
3. Bob chiffre son message avec la clef générée en 1. Il obtient un message chiffré.
4. Bob envoie son message chiffré, et la clef chiffrée.
5. Alice reçoit le tout. Elle déchiffre la clef chiffrée avec sa clef privée. Elle obtient la clef que Bob avait générée en 1.
6. Avec cette clef, elle peut enfin déchiffrer le message.

Question importante : Comment s'assurer que je possède bien la *bonne* clef publique de mon destinataire, et que ce n'est pas un usurpateur qui m'a fourni ne fausse clef publique pour pouvoir continuer d'intercepter mes messages ?

Une méthode est de vérifier l'intégralité de la clé de la main à la main. Il existe aussi des *autorités* de certification auxquelles on peut faire confiance pour garantir la provenance d'une clé, et enfin les systèmes de « *web of trust* ». Nous verrons ces dernières solutions plus tard dans l'atelier.

Application à la signature

Maintenant, nous allons appliquer ce même chiffrement asymétrique à la signature numérique : comment certifier que je suis bien l'auteur d'un message, et que personne n'a usurpé mon identité ? *Évidemment, il est possible d'utiliser la signature en même temps que le chiffrement (que nous venons de voir). Mais pour simplifier l'explication, nous supposons ici que le message est envoyé en clair.*

L'analogie clef publique/privée avec le cadenas ouvert/clef est utile, mais elle a ses limites. Une particularité utile de la paire de clef publique/privée et que si, comme nous venons de le faire, nous pouvons chiffrer avec la clef publique, et déchiffrer avec la clef privée, il est aussi possible de faire l'inverse : chiffrer avec la clef privée, et déchiffrer avec la clef publique. Pour l'explication qui vient, la clef privée est donc le cadenas ouvert, et la clef publique est la clef de ce cadenas, qui est accessible à tous (elle est publique).

Version simplifiée Encore une fois, nous commençons avec une version simplifiée.

Bob veut envoyer un message à Alice, en certifiant son identité.

1. Il se crée une paire de clefs.
2. Il chiffre son message avec sa clef privée.
3. Il envoie son message, ainsi que le message chiffré avec sa clef privée.
4. Alice reçoit les deux messages. Elle déchiffre le message chiffré avec la clef publique de Bob, et elle le compare avec l'autre message. Si c'est le même, elle est certaine que Bob en est l'auteur, puisque seul lui est capable de le chiffrer de la sorte.

Fonctionnement réel En pratique, comme nous l'avons vu plus tôt, il serait trop lent de chiffrer le message avec la clef privée. Nous n'allons donc pas chiffrer le message, mais une empreinte de ce message.

Dans l'exemple qui suit, nous prenons comme empreinte le nombre de mots du message.

Bob veut envoyer le message `Comment vas tu ?` à Alice.
Il l'envoie, avec comme signature le nombre de mot du message (ici 3). Cette signature est transmise de façon chiffrée, en utilisant le chiffrement vu dans la précédente partie.
Alice reçoit le message de Bob. Elle compte le nombre de mots, et le compare avec ce nombre qui a été envoyé, chiffré, comme signature. S'ils concordent, c'est que le message n'a pas été modifié.

En pratique, il est assez simple de faire un message qui a le même nombre de mots qu'un autre, donc cette empreinte est mauvaise. Ce n'était qu'un exemple. Nous utilisons donc une *fonction de hachage* plus complexe, telle qu'il soit *en pratique* impossible de fabriquer un message donnant une empreinte donnée.

De telles fonctions de hachage, comme le `sha256`, sont telles qu'actuellement, il n'est pas possible, étant donné le résultat du hachage (par exemple `cd66e8ddb3dae4b9d1bd038878f7628278b2c6d7`), de générer un fichier ayant cette même empreinte.

Ainsi, si l'empreinte du message telle qu'envoyée par Bob, et celle qu'Alice a calculée à partir de son message, concordent, Alice est sûre que personne n'a usurpé l'identité de Bob. *Notons au passage qu'Alice est également sûre que le message n'a pas été modifié.*

Actuellement, les techniques de hachage utilisées couramment sont `md5`, `sha1`, `sha256`, et `512`. La première (`md5`) a été cassé récemment (en 2008), et n'est donc plus fiable. elle continue néanmoins parfois à être utilisé, car tous les éditeurs de logiciel et de protocoles ne se sont pas encore adaptés.

Certificats

Si je veux aller sur un site sécurisé (relever mon courrier sur un webmail par exemple), comment puis-je utiliser les techniques de chiffrement et signature vues précédemment ? Le cœur du problème réside à l'obtention de la clef publique du site. Ensuite, la communication pourra se faire de manière sécurisée en utilisant les techniques vues plus haut.
Pour ce faire, le site va m'envoyer sa clef publique. Comment être sûr que c'est bien la clef publique de mon serveur de messagerie, et non pas celle d'un autre ordinateur voulant se faire passer pour lui ?
Je vais expliquer ici comment les autorités de certification peuvent certifier que cette clef publique est la bonne, et ainsi certifier que l'on communique bien avec la même personne.

Mise en œuvre

Un client veut accéder au site `https://webmail.no-log.org`.

1. Le client envoie une requête au serveur.
2. Le serveur renvoie un certificat, signé numériquement. Le certificat a été signé avec la clef privée d'une autorité de certification racine, dont la clef publique est intégrée au navigateur du client.
3. Le client fait quelques vérifications, à savoir :
   - la signature du certificat est valide ;
   - la date du certificat n'est pas dépassée ;
   - le certificat se rapporte bien au site en cours.

Ce certificat contient notamment la clef publique du site web. À partir de ce moment là, le client ayant la clef publique du serveur, c'est un classique échange de données sécurisées, comme vu dans la partie précédente, à savoir :

4. Le client génère une clef de chiffrement symétrique, qu'il chiffre en utilisant la clef publique du serveur.
5. Le client envoie cette clef au serveur, avec sa requête, chiffrée avec cette clef.
6. Le serveur déchiffre cette clef symétrique avec sa clef privée.
7. Le serveur chiffre son message (le site web) avec cette clef symétrique, et l'envoie au client.
8. Le client déchiffre ce message avec la clef symétrique.

Comment le site `https://webmail.no-log.org` a-t-il obtenu son certificat ? C'est la qu'interviennent les *autorités de certification*. Ce sont des entreprises chargées de vérifier et certifier les identités d'acteurs de l'Internet.

Concrètement, le propriétaire d'un site voulant se faire certifier fournit à une autorité de certification les preuves de son identité, ainsi que sa clef publique.
L'autorité de certification vérifie l'identité du demandeur, et crée un certificat (contenant notamment l'identité du demandeur et sa clef publique), qu'il signe, et l'envoie au demandeur.
Le demandeur a donc en sa possession un certificat signé par une autorité de certification certifiant son identité et contenant sa clef publique.

Un exemple de certificat est donné en annexe, page 8.

Faux certificats

Petit problème concernant les certificats : ils utilisent pour le moment le hachage *md5* pour la signature, qui n'est plus fiable. Ainsi, en 2008, des chercheurs ont utilisé cela pour générer un faux certificat d'une autorité de certification, en utilisant la puissance de calcul de 200 PlayStation3. Du coup, tous les certificats signés par cette fausse autorité étaient faux.

Cela signifie qu'il est possible de casser toute ce beau système que je viens de décrire.

Une solution est de remplacer le hachage *md5* par un hachage plus puissant (comme le

sha256). C'est fait peu à peu, mais ce n'est pas encore systématique.

Certificats auto-signés

La certification auprès d'une autorité de certification coûte cher ; tous le monde n'a pas l'envie ou les moyens de payer un tel certificat. Il existe donc des certificats auto-signés.
Pour ces certificats, le principe est que ce n'est pas une autorité de certification qui signe le certificat, mais la personne qui l'utilise.

Il est alors nécessaire de vérifier manuellement le certificat. Ceci sera expliqué dans un prochain atelier.

Éthique

Le but premier des autorités de certification n'est pas de certifier des identités : c'est de faire de l'argent. Ces autorités sont des entreprises privées, donc leur but est de gagner de l'argent, en vendant comme service la certification d'identités. Mais vérifier une identité coûte cher.

Qu'est-ce qui nous prouve qu'ils le font bien ? Encore une fois, c'est une question de confiance... Leurs faites vous confiance ?
On peut espérer que ne serait-ce que pour maintenir leur activité, elles font bien leur travail...

Il existe néanmoins une autorité de certification à but non lucratif : CACert, qui mise sur le *web-of-trust*.

Web of trust

Les certificats sont une réponse centralisée au problème de certification de l'identité des acteurs d'Internet. Mais nous avons vu que cette centralisation de l'autorité pose des problèmes.

Une autre idée est le web of trust. C'est une certification a-centrée. Ici chaque participant participe à la certification d'autres participants.

L'idée est que je fais confiance à un certain nombre de personnes (des amis par exemple), et je signe leur clef publique avec ma clef privée. Eux font de même avec ma clef publique : ils la signent avec leur clef privée.
Supposons que je veuille communiquer avec un nouvel acteur, dont je ne possède pas la clef publique. Je télécharge sa clef publique, qu'il me propose, et je télécharge aussi sa clef publique par d'autres personnes, dont les personnes en qui j'ai confiance et qui ont signé cette clef. Si mes amis confirment que sa clef publique est la bonne, alors je peux lui faire confiance et penser qu'il est bien la personne qu'il prétend être.

Bien entendu, cette solution n'est pas parfaite. Comme inconvénients, on peut citer le fait que cette solution impose que chaque personne publie la liste des personnes en qui elle a confiance. Je considère cette information comme relevant de ma vie privée.

CACert

CACert mise sur un réseau de bénévoles pour établir une web-of-trust permettant de certifier que telle personne est bien qui elle prétend être. Chaque certification individuelle permet de capitaliser des points qui permettent ensuite d'obtenir des certificats. Ça fait beaucoup plus de monde à corrompre avant de faire un faux.

Annexes

Exemple de certificat

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 441565 (0x6bcdd)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: O=Root CA, OU=http://www.cacert.org, CN=CA Cert
              Signing Authority/emailAddress=support@cacert.org
        Validity
            Not Before: Apr 16 18:58:52 2009 GMT
            Not After : Apr 16 18:58:52 2011 GMT
        Subject: CN=*.no-log.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:da:fa:6e:95:1a:ca:cd:b9:8c:08:79:d8:d4:1f:
                    8b:ca:df:29:6b:d3:6e:72:34:00:56:cd:fe:55:1d:
                    56:b3:3d:81:7f:3a:f2:63:c6:85:af:dd:0f:87:16:
                    a0:87:50:bc:6a:f7:19:5e:80:15:82:14:41:00:9c:
                    02:1b:b4:64:ec:a2:36:45:e1:5d:0e:e4:2b:0f:d3:
                    20:eb:28:9f:69:ad:e1:2c:12:6c:b8:30:db:b6:7e:
                    ae:26:e4:50:b4:95:8e:d2:26:d6:2a:b5:e8:e0:36:
                    57:6d:7c:0a:58:44:45:d2:77:5c:fd:f6:4f:cf:a7:
                    50:d4:f8:f3:88:47:24:89:8b
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:FALSE
            X509v3 Extended Key Usage:
                TLS Web Client Authentication, TLS Web Server Authentication, Nets-
cape Server Gated Crypto, Microsoft Server Gated Crypto
            X509v3 Key Usage:
                Digital Signature, Key Encipherment
            Authority Information Access:
                OCSP - URI:http://ocsp.cacert.org/

            X509v3 Subject Alternative Name:
                DNS:*.no-log.org, othername:<unsupported>
    Signature Algorithm: sha1WithRSAEncryption
        88:e7:6f:81:fe:e0:77:24:86:c3:cc:e8:b3:10:24:7d:32:4e:
        88:da:20:f9:1b:68:69:d0:8f:2f:1c:74:ad:5e:d0:c5:bb:81:
        8d:4d:a8:93:00:f6:ba:ef:04:ef:92:1b:2e:c5:11:7f:3d:dd:
        9e:ad:d9:86:1f:ce:05:9b:88:a9:bf:91:a1:d7:89:5b:5b:da:
        23:b0:ca:45:51:94:fa:4c:b4:79:08:91:cb:64:99:c2:b9:e0:
        ff:f8:0a:32:cc:12:9a:a0:ca:98:0a:3e:76:f6:03:b6:21:01:
        cf:6a:2d:6a:39:1e:c8:70:dc:39:f2:85:73:35:98:95:6b:73:
        d1:78:0a:bd:08:8d:f4:e4:de:91:b0:52:dd:a5:c7:e9:49:16:
```

```
a1:75:63:b6:bc:a5:ec:83:20:68:91:10:87:e7:37:2e:10:d2:
38:84:4d:1c:e7:9c:91:33:6e:49:55:f9:e8:71:dd:5a:64:3a:
53:60:2c:42:08:b9:27:30:d2:ba:97:65:96:84:de:a3:6f:f2:
5c:d0:bd:c9:94:48:75:3e:27:7d:a0:a4:0e:33:e7:91:cf:34:
0d:8f:5c:99:40:91:e2:fc:45:48:e5:0c:f8:82:6b:8d:95:e1:
3f:a8:6d:52:a8:f3:38:54:16:db:0a:6b:b0:2a:3f:f7:20:0f:
83:b8:fa:73:df:6b:bd:dc:cd:41:28:a8:17:80:8c:0b:d0:b9:
ba:63:01:98:c5:12:0c:45:fd:6c:d8:90:c4:90:65:71:41:a6:
84:8d:40:ae:7f:cc:44:27:78:97:35:46:43:ad:55:ba:5c:b6:
cf:4f:43:e9:d3:aa:b8:25:d7:e2:ea:65:58:a0:91:c7:a7:ea:
c7:69:f7:de:67:b9:05:6c:30:2e:e8:bf:08:51:06:fe:e6:d0:
fd:d4:5b:b8:ad:ea:05:8e:91:0e:fb:fe:54:54:1c:51:d0:75:
5f:34:a5:b6:c0:cd:09:d0:84:50:f7:1f:0d:9e:7e:70:60:da:
1a:2c:98:58:fc:c0:95:28:a2:be:8e:b7:d3:4e:0c:6a:4d:33:
0c:7b:43:49:39:e7:09:de:54:d4:54:58:d6:10:c1:c9:dd:99:
51:12:44:f7:c4:c0:98:c2:41:61:8c:c8:c8:cb:81:df:2a:d3:
2b:03:5e:7e:60:c8:5e:a9:24:f6:fd:1f:3c:c3:ac:83:33:ea:
ed:0c:af:99:91:70:bc:e1:22:a0:ff:11:e6:5e:9b:9a:8a:82:
08:e1:2a:6b:db:9e:ae:66:52:c8:44:2c:58:75:0b:77:d5:7a:
2a:14:bd:0d:2d:2e:5b:e8:d6:65:e7:71:48:11:c3:31:08:6e:
a1:bd:e0:ad:8e:6c:18:59
```

Les choses intéressantes à voir sont :

`Subject: CN=*.no-log.org`
: Site auquel s'applique ce certificat.

`Issuer: O=Root CA, OU=http://www.cacert.org, CN=CA Cert Signing Authority`
: Autorité de certification : « CA Cert Signing Authority ».

`Validity: Not Before: Apr 16 18:58:52 2009 GMT; Not After : Apr 16 18:58:52 2011 GMT`
: Dates de validité

`Signature Algorithm: sha1WithRSAEncryption`
: Ce certificat est signé avec sha1 et un chiffrement RSA, et la clef suit.

`RSA Public Key: (1024 bit)`
: Clef publique du site