

Falzon Marc

BTS Informatique de Gestion - Session 2005

Epreuve E6 : Soutenance de projet

Déploiement d'une Infrastructure de Gestion de Clés publiques libre

1) Présentation de l'association

J'ai effectué mon stage au sein de l'association Alternatives 87, dont mon maître de stage Jean-Philippe Gaulier est le président. Depuis 1998, le but de cette association est de promouvoir le logiciel libre dans le Limousin, en participant à des événements nationaux (ex : Linux Solutions) et internationaux (ex : Rencontres Mondiales du Logiciel Libre) et aussi en en organisant (ex : Game-Over, conférences diverses). Elle gère également un Espace Publique Numérique (EPN) où le public peut venir se connecter à Internet, suivre des formations et trouver des ressources et de l'aide sur les logiciels libres.

2) Mission

Au cours de mon stage, ma mission principale a été de concevoir puis de déployer une infrastructure de gestion de clés publiques (PKI, *Public Key Infrastructure*) entièrement libre dans le sens où les logiciels mis en œuvre sont des **logiciels libres**⁽¹⁾ (donc peu de contraintes de licence) et où le service qu'elle fournira sera lui aussi libre et gratuit. Mon projet est en quelque sorte une maquette fonctionnelle qui sera réutilisé dans le futur dans des environnements de production.

Au premier abord, travailler sur un projet de conception/déploiement de PKI alors que l'on étudie en BTS IG option administrateur de réseaux locaux d'entreprises peut sembler totalement décalé, et n'ayant pas de rapport direct avec l'administration de réseaux.

Toutefois il faut savoir que la sécurité des matériels et des applications en réseau est des caractéristiques de plus en plus considérées par les responsables de services d'information dans les entreprises. Un autre aspect de la sécurité informatique en entreprise tend à être mis de plus en plus fréquemment en avant : la sécurité des données, des échanges de données et des connexions.

En effet à quoi bon sécuriser son réseau privé si les connexions des clients ou des partenaires transitent en clair sur les réseaux intermédiaires jusqu'à ce dernier ? Existe-t-il un moyen de sécuriser l'accès aux ressources internes d'une entreprise pour le personnel extérieur ? Comment peut-t-on s'assurer que l'e-mail que nous venons de recevoir provient bien de son expéditeur ou qu'il n'a pas été intercepté puis altéré au passage ? Des technologies ont été créées en réponse à ce genre de problématiques et la solution de PKI que je dois fournir tentera d'y répondre.

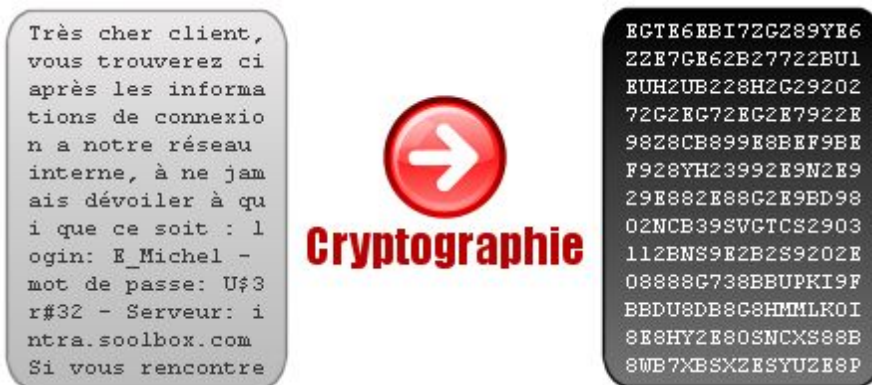
2.1) Notions préliminaires

Avant toute référence au concept global de PKI, il convient d'éclaircir certains points cruciaux à la compréhension de certains concepts et principes sur lesquels elle est elle-même basée, comme la cryptographie ou encore la signature numérique.

Cryptologie, cryptographie, chiffrement

La cryptologie est considérée comme une science mathématique qui englobe la cryptographie et la cryptanalyse.

La cryptographie est l'art et la science de garder le secret des messages. Elle est un art dans le sens où elle est pratiquée depuis de nombreux siècles (l'Empire Romain sous Jules César l'utilisait déjà) et que de nombreuses techniques (les algorithmes) ont été imaginées pour chiffrer des messages, toujours plus ingénieuses et complexes que les précédentes. Elle est reconnue comme une science à part entière c'est-à-dire depuis qu'elle est enseignée en tant que thème de recherche universitaire et est considérée comme une discipline mathématique au même titre que l'algèbre et la théorie des nombres.



La cryptanalyse est l'inverse de la cryptographie, dans le sens où elle est définie comme l'art de décrypter les messages chiffrés, de les "casser".



Il est à noter une différence entre "chiffrer" et "crypter" un message; le chiffrement est réversible grâce à un système de clés, c'est-à-dire qu'une fois chiffré un message peut être ramené à son état d'origine en appliquant (le plus souvent) le processus inverse du chiffrement, ce qui est impossible avec le cryptage.

Nous pouvons distinguer deux types de cryptographie :

Cryptographie symétrique, ou "à clé secrète"

Un système de chiffrement est dit symétrique quand la même clé est utilisée pour chiffrer et déchiffrer un message.

Il existe de nombreuses méthodes de chiffrement, nommées "algorithmes"; on peut citer parmi les algorithmes à clé les plus connus Blowfish, DES (et son successeur AES), Vigenère ou encore la machine cryptographique *Enigma* utilisée par les militaires allemands pendant la 2nde Guerre Mondiale.

Très cher client,
vous trouverez ci
après les informa
tions de connexio
n a notre réseau
interne, à ne jam
ais dévoiler à qu
i que ce soit : l
ogin: E_Michel -
mot de passe: Uf3
r#32 - Serveur: i
ntra.soolbox.com
Si vous rencontre



Chiffrement

Clé
secrète



ECTE6EBI72GZ89YE6
ZZE7CE62B27722BU1
EUH2UB228H2G29202
72G2EG72EG2E7922E
98Z8CB899E8BEF9BE
F928YH23992E9N2E9
29E882E88G2E9BD98
02NCB39SVGTCS2903
112BNS9E2B2S9202E
08888G738BBUPKI9F
BBDU8DB8G8HMLK0I
8E8HY2E80SNCXS88B
8WB7XBSX2ESYUZE8P

Il est possible de chiffrer un message sans utiliser de clé secrète, mais en effectuant une rotation de lettres dans l'alphabet comme la méthode **ROT13**⁽²⁾ (autrement appelée "méthode de César" car lui-même l'utilisait pour chiffrer les ordres destinés à ses légions). Cependant, il est fortement déconseillé d'utiliser une méthode non basée sur une clé secrète car une fois que le nombre de rotations effectuées pour chiffrer le message a été trouvé - ce qui est rapide étant donné le peu de possibilités de chiffre clé possible, à savoir les 26 lettres de l'alphabet - le message est entièrement déchiffrable.

Très cher client,
vous trouverez ci
après les informa
tions de connexio
n a notre réseau
interne, à ne jam
ais dévoiler à qu
i que ce soit : l
ogin: E_Michel -
mot de passe: Uf3
r#32 - Serveur: i
ntra.soolbox.com
Si vous rencontre



Déchiffrement

Clé
secrète

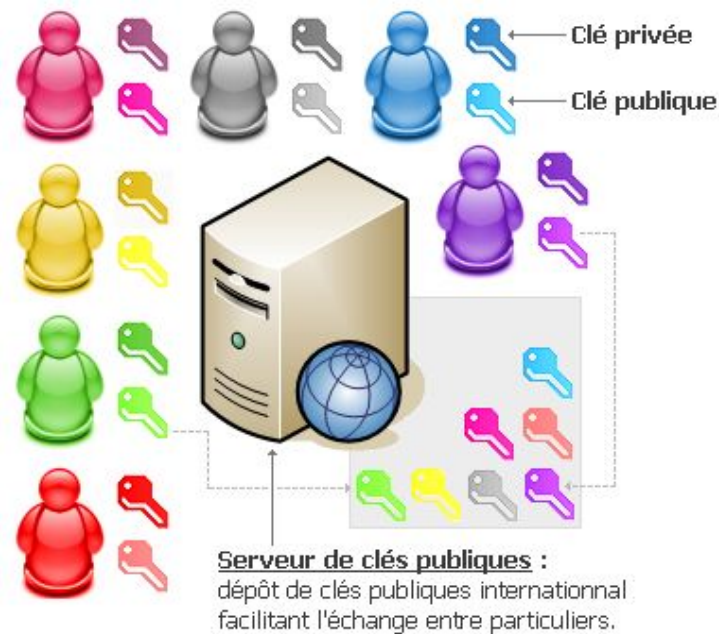


ECTE6EBI72GZ89YE6
ZZE7CE62B27722BU1
EUH2UB228H2G29202
72G2EG72EG2E7922E
98Z8CB899E8BEF9BE
F928YH23992E9N2E9
29E882E88G2E9BD98
02NCB39SVGTCS2903
112BNS9E2B2S9202E
08888G738BBUPKI9F
BBDU8DB8G8HMLK0I
8E8HY2E80SNCXS88B
8WB7XBSX2ESYUZE8P

Cryptographie asymétrique

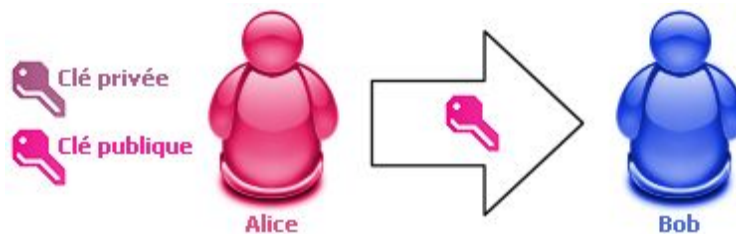
A l'inverse de la cryptographie symétrique, un système de chiffrement asymétrique est basé sur des fonctions à sens unique, c'est-à-dire qu'il est aisé de chiffrer un message grâce à cette fonction mais qu'il est très difficile voire impossible de lui rendre son état initial, à moins de posséder une information particulière : la clé privée. Là encore, le choix de l'algorithme est possible parmi DSA, RSA ou El Gamal.

Chaque utilisateur possède une paire de clés : une clé dite "publique" à partager avec ses proches et collègues si son travail implique l'échange de données chiffrées ou encore la stocker sur un serveur de clés publiques (par exemple celui de l'université du **MIT**⁽³⁾ aux Etats-Unis), et une clé dite "secrète" ou "privée" qui doit impérativement être gardée cachée et protégée.

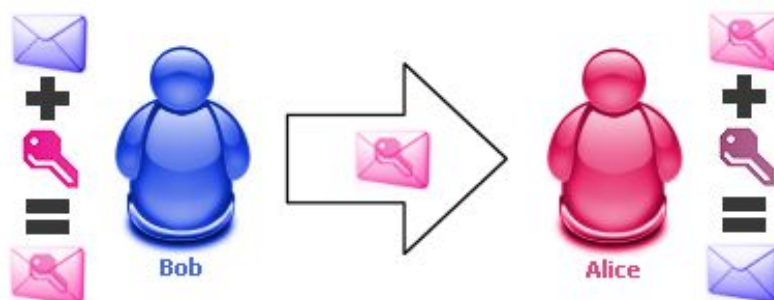


Pour illustrer le principe de fonctionnement de la cryptographie asymétrique, nous allons simuler un échange de documents chiffrés entre deux personnes : Alice et Bob.

Alice souhaite pouvoir chiffrer et déchiffrer des messages, et pour ce faire elle génère une fonction à sens unique et à brèche secrète (cad. avec moyen d'inverser le processus via une clé privée). A la suite de cela, elle communique sa clé publique et garde précieusement son information de brèche secrète : la clé privée.



Si Bob veut envoyer un message chiffré à sa seule attention, il utilisera la clé publique d'Alice pour chiffrer le message en question. Alice, elle, se servira de sa clé privée pour le déchiffrer; ainsi elle sera la seule à pouvoir lire le message, à moins que la brèche ne soit découverte entre temps.



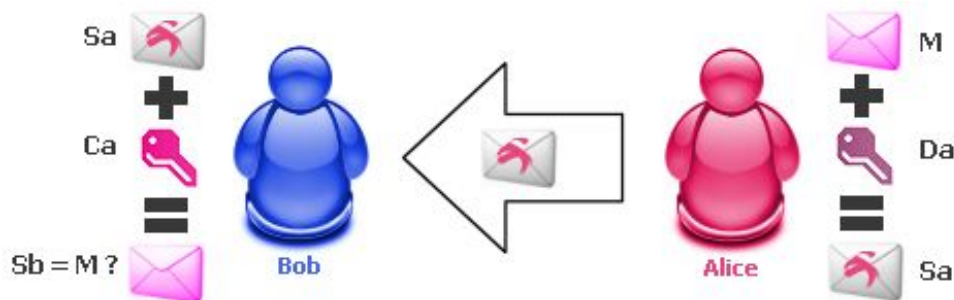
Signature numérique

Depuis **mars 2000** ⁽⁴⁾, la signature numérique de documents informatiques a la même valeur légale que la signature physique. Elle fonctionne selon les mêmes principes fondamentaux, à savoir :

- le lecteur doit être convaincu que le signataire a bien signé le document,
- la signature ne peut pas être falsifiée,
- la signature n'est pas réutilisable. Elle fait partie du document signé et ne peut être déplacée sur un autre document,
- un document signé est inaltérable. Une fois signé, on ne peut le modifier.

Pour pouvoir appliquer ces principes, la signature numérique ne peut se baser que sur la cryptographie asymétrique.

Reprenons l'exemple de nos deux personnages Alice et Bob : Alice a la possibilité de "signer" un message en le chiffrant à l'aide de sa clé privée. Si l'on considère que le message à signer est noté **M**, la fonction de chiffrement d'Alice **Ca** et sa fonction de déchiffrement **Da**, la signature se traduit par **Sa = Da(M)**, **Sa** étant une somme de contrôle, ou *hash*. Ainsi, Bob, pour s'assurer que le message qu'il vient de recevoir vient bien d'Alice, calculera la somme de contrôle **Sb** (la signature) avec la clé publique d'Alice, ce qui se traduit par **Sb = Ca(Sa)**. Si **Sb = M**, alors le message provient bien d'Alice.



Une petite précision : en réalité Alice ne signe pas le message M en entier mais seulement une "empreinte" de longueur fixe extraite par une **fonction de hachage**. Cette empreinte est unique à chaque message, et si la fonction utilisée (SHA-1, SHA-256, Whirlpool ou Tiger) est sûre cette empreinte est aussi fiable qu'en chiffrant le message dans son intégralité.

L'intérêt d'un tel procédé est que la signature d'un message est beaucoup plus rapide puisqu'il y a moins de quantité à chiffrer.

Certificat numérique

Un certificat permet d'associer une clé publique à une entité (organisation, machine, individu); il est en quelque sorte la carte d'identité de la clé publique, délivré par un organisme appelé autorité de certification (souvent notée CA pour *Certification Authority*), que l'on retrouve dans une PKI.

Techniquement, il s'agit d'un bloc d'informations organisées selon une structure particulière : la norme actuellement utilisée dans ce domaine est **X.509** (version 3), définie par l'**ITU** ⁽⁵⁾ (l'Union Internationale des Télécommunications). Elle impose que le certificat donne les informations suivantes :

- le nom de l'autorité de certification,
- le nom du propriétaire du certificat,
- la date de validité du certificat,
- l'algorithme de chiffrement utilisé,
- la clé publique du propriétaire,
- et enfin la signature numérique des données ci-dessus par une personne ou entité prenant en charge la création de ce certificat et ayant autorité de certification.

L'ensemble de ces informations (informations + clé publique du demandeur) est signé par l'autorité de certification; cela signifie qu'une fonction de hachage crée une empreinte de ces informations, puis ce condensé est chiffré à l'aide de la clé privée de l'autorité de certification, la clé publique ayant été préalablement largement diffusée afin de permettre aux utilisateurs de vérifier la signature avec la clé publique de l'autorité de certification.

2.2) Mise en place d'une solution de PKI libre

2.2.a) Principe

Une PKI (Public Key Infrastructure, "Infrastructure de Gestion de Clés Publiques") est un système semi autonome permettant de générer puis de contrôler le cycle de vie de **certificats numériques**.

D'après l'**IETF**⁽⁶⁾ (l'Internet Engineering Task Force), une PKI est scindée en 5 entités distinctes :

- **L'Autorité de Certification** (CA : *Certification Authority*) qui a pour mission de signer les demandes de certificat (CSR : *Certificate Signing Request*) et de signer les listes de révocations (CRL : *Certificate Revocation List*). Cette autorité est la plus critique.
- **L'Autorité d'Enregistrement** (RA : *Registering Authority*) qui a pour mission de générer les demandes de certificats, et d'effectuer les vérifications d'usage sur l'identité de l'utilisateur final (les certificats électroniques sont nominatifs et uniques pour l'ensemble de la PKI).
- **L'Autorité de Dépôt** (*Repository*) qui a pour mission de stocker les certificats numériques ainsi que les listes de révocation (CRL).
- **L'Autorité de séquestre** (*Key Escrow*) a un rôle particulier; en effet lorsqu'on génère des certificats de chiffrement, on a l'obligation légale de fournir aux autorités un moyen de déchiffrer les données chiffrées par un utilisateur de la PKI. C'est là qu'intervient le Séquestre, cette entité a pour mission de stocker de façon sécurisée les clés de chiffrement qui ont été générées par la PKI, pour pouvoir les restaurer le cas échéant.
- **L'Entité Finale** (EE : *End Entity*) est optionnelle, cette entité est en charge d'enregistrer les nouveaux utilisateurs finaux. Les utilisateurs des certificats (machines, personnes physiques et morales) sont font partie de l'entité finale.

Basiquement, il y a 2 cycles de vie possibles pour un certificat généré :

- Le certificat numérique expire (chaque certificat numérique contient une date de "naissance" et une date de "péremption").
- Le certificat est révoqué, pour quelque raison que ce soit (perte de la clé privée associée, etc.) et dans ce cas, l'identifiant du certificat numérique est ajouté à une liste de certificats révoqués (CRL) pour informer les utilisateurs qu'ils ne doivent plus faire confiance à ce certificat.

2.2.b) Nécessité d'une PKI

Le déploiement d'une solution de gestion de clés publiques au sein d'une organisation se justifie principalement par un besoin de rapidité dans l'authentification de documents. De nos jours les entreprises de moyenne et grande taille sont toutes informatisées : à quoi bon s'équiper d'un système de messagerie interne perfectionné ou encore d'accès VPN (*Virtual Private Network*; un sous réseau privé virtuel sécurisé au sien d'un réseau non sécurisé comme Internet) pour les sites distants si les contrats ou autres documents d'une importance capitale sont encore sur support papier et nécessitent donc une signature manuscrite, ce qui implique délais de réception, encombrement et difficultés d'archivage.

Depuis que la signature numérique a valeur légale, elle commence à faire son apparition dans les grandes entreprises. En effet ce procédé permet de sauvegarder un temps considérable grâce à la vitesse de transmission des données informatiques sur l'Internet et les Intranets : un simple e-mail signé numériquement en bonne et due forme peut faire office de contrat.

Alternatives 87, l'association pour laquelle je vais déployer la PKI nécessitait un moyen de gérer l'accès restreint à certaines parties de son site et de centraliser les clés publiques de ses membres afin de signer leurs échanges informatisés; la solution que je leur ai proposé leur convient parfaitement car les services fournis par la PKI répondent à ces besoins. Ainsi grâce à mon projet ils vont pouvoir générer un certificat électronique pour leur serveur web HTTPS et remplacer leurs protections par *login* ("identifiant")/mot de passe par une authentification basée sur les certificats des membres habilités à parcourir ces sections (lesquels se verront remettre précédemment leur propre certificat).

2.2.c) Construction et déploiement de la PKI

Une PKI, étant un agrégat d'entités distinctes, est elle-même basée sur différents éléments logiciels : nous allons voir lesquels. De plus, le challenge est de n'utiliser que des logiciels libres.

La machine sur laquelle j'ai déployé la PKI est basée sur un système d'exploitation (OS) libre GNU/Linux.

Note : j'ai compilé tous les logiciels à partir de leur code source de manière à activer des options particulières si le besoin s'en faisait sentir, comme par exemple le support de **SSL**⁽⁷⁾ (une couche de chiffrement qui s'intercale entre le réseau et l'utilisateur) par Apache.

Couche cryptographique : OpenSSL

Toutes les opérations cryptographiques (génération de clés asymétriques, chiffrement, signature numérique...) réalisées côté serveur par la PKI sont assurées par **OpenSSL**⁽⁸⁾, une collection d'utilitaires libres et de bibliothèques de fonctions qui fournissent une couche cryptographique aux systèmes UNIX/BSD.

Cette application a été compilée selon la méthode standard (*cf. note ci-dessus*).

Serveur HTTP(S) : Apache + mod_ssl + PHP

Pour pouvoir accéder depuis n'importe où aux services fournis par la PKI, nous avons besoin d'une interface web qui sera garantie par le serveur HTTP libre **Apache**⁽⁹⁾.

Avant de compiler Apache j'ai dû préparer le module de chiffrement des communications HTTP (protocole de transmission de pages web) entre l'interface web de la PKI et les utilisateurs / administrateurs. En effet, le protocole HTTP n'est pas sécurisé et il est possible d'intercepter le contenu des échanges comme les mots de passe ou pire encore, les certificats générés par la PKI.

mod_ssl⁽¹⁰⁾ est un module pour Apache qui ajoute une couche de chiffrement SSL (HTTPS) - et maintenant **TLS**⁽¹¹⁾ (le protocole successeur de SSL) - de façon à créer un tunnel virtuel sécurisé entre les 2 entités communicantes, généralement un client et un serveur HTTPS.

Il nous faut également installer le langage de script **PHP**⁽¹²⁾ qui servira à automatiser le traitement de certaines opérations de la PKI, notamment l'interfaçage avec OpenSSL.

La méthode de compilation et d'installation d'apache modifié pour le support de SSL est légèrement différente que pour une installation traditionnelle; pour plus de détails consultez les annexes présentées le jour de la soutenance.

Solution de PKI : PHPKI

La solution de PKI retenue est **PHPKI**⁽¹³⁾, un *front-end* (une surcouche) pour OpenSSL codé en **PHP**. La finalité de mon projet étant de fournir un service de génération de certificats électroniques que ce soit pour des utilisateurs (signature de données/e-mails) ou bien pour des serveurs (authentification VPN/serveurs SSL), le tout gratuitement et en utilisant uniquement des logiciels libres, cette application convient parfaitement car elle fournit une autorité de certification (CA), une autorité d'enregistrement (RA); le stockage (*Repository*) est local ainsi que la séquestre.

ATTENTION : Lors de ma première installation de PHPKI (parce qu'il y en a eu beaucoup) j'ai été confronté à quelques erreurs d'écriture sur les fichiers d'installation, et par conséquent l'application, puisque mal installée, fonctionnait peu voire pas du tout. Après avoir passé quelques jours à déboguer les sources il est apparu que PHPKI ne peut fonctionner si le "Safe Mode" de **PHP** est activé et si les "Register Globals" (la portée des variables globales) sont désactivées; j'ai donc dû adapter la configuration de PHP en conséquence, sachant toutefois que la modification de ces options sont des **failles potentielles et connues du public**. La solution à apporter en environnement de production serait de recoder certaines parties de PHPKI.

Voici une copie d'écran de la phase d'installation de PHPKI via un navigateur web; d'autres opérations post-installation ont été réalisées, principalement des réglages concernant la sécurité de l'application (réglage des permissions d'accès aux fichiers de la PKI) et des corrections de bogues internes à PHPKI.

https://bouh:8443/phpki-0.60/setup.php

PHPki: S/MIME Certificate Authority

PHPki S/MIME Certificate Authority

[ReadMe](#) [Setup](#) [About](#)

Certificate Authority Initial Setup

Root Certificate Data	
Organization *	Alternatives 87
Department/Unit *	PKI Alternatives 87
Common Name * This is embeded in certificates, and is most often displayed in e-mail clients as the <i>Issued By</i> text. This is usually the full name of your certificate authority (i.e. ACME Certificate Authority).	Autorité de Certification Alternatives 87
Technical Contact E-mail Address * Enter an e-mail address where users should send correspondence regarding your certificate authority and the certificates you issue.	pki@alternatives87.eu.org
Locality *	Limoges
State/Province *	FR
Country *	FR
Password * This password will be used to protect your root certificate private key. Do not lose or forget this password.	***** Again *****
Certificate Life * Enter the number of years you wish your root certificate to be valid.	5 Years
Certificate Authority Base URL Enter the public Web address where your Certificate Authority will reside. The address should end with a trailing slash (/) character. This address will be embeded in all certificates issued by your CA, for informational purposes.	https://bouh:8443/pki/

Configuration Options	
Location of OpenSSL Executable * Enter the location of your OpenSSL binary. The default of /usr/bin/openssl is usually ok.	/usr/bin/openssl
File Upload Prefix This is an optional prefix which will be added to root certificate and certificate revocation list file uploads. Normally the root certificate is uploaded as caroot.crt. With a prefix like "acme_", the root certificate would be uploaded as "acme_caroot.crt".	alt87_
Page Header Title This title will be displayed superimposed over the PHPki logo at the top of every page.	PKI Alternatives 87
Help Document Contact Info This text will be inserted into the online help document under the "Getting Additional Help" section. Include full contact info for the convenience of your users. Use HTML tags to improve presentation.	<pre>Contact:
 PKI Alternatives 87
 Alternatives 87
 87000 Limoges

 E-mail: pki@alternatives87.eu.org is preferred.</i>
</pre>

* Required field

PHPki v0.60 - Copyright 2003 - William E. Roadcap

2.2.d) Exploitation du système

Selon le principe d'une PKI, PHPKI est séparé en entités distinctes, mais moins bien délimités que le modèle théorique à savoir que l'application remplit bien les rôles d'autorité de certification (CA), d'enregistrement (RA), de stockage (Repository) de séquestre mais pas d'entité finale. De plus, la PKI est ici scindée en deux parties :

- l'une publique où les utilisateurs peuvent télécharger le certificat de la CA, la dernière CRL et tous les certificats qui ont été délivrés par la CA;
- l'autre est privée et son accès est restreint au niveau du serveur où l'administrateur de la PKI peut créer les certificats demandés par les utilisateurs, les renouveler et les révoquer.

Dans un premier temps nous allons jouer le rôle de l'administrateur en créant un nouveau certificat numérique pour notre serveur HTTPS.

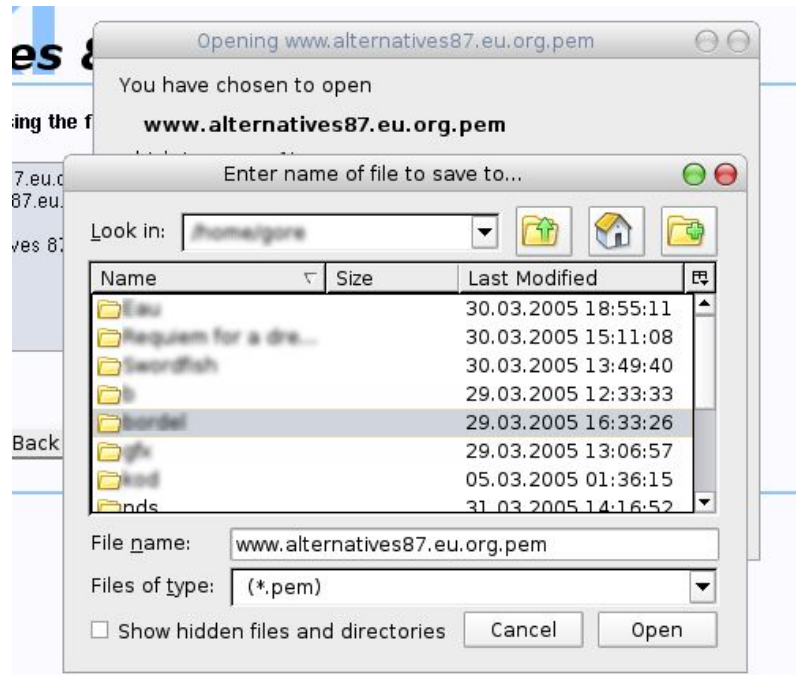
Toutes les opérations de pilotage de la PKI sont à effectuer avec un navigateur web. Nous nous rendons donc dans la section prévue à cet effet et nous remplissons les champs avec les informations requises; il est ensuite demandé de confirmer les informations fournies avant de valider la création :

The screenshot shows a web browser window with the URL `https://bouh:8443/pki/ca/request_server_cert.php`. The page title is "PHPKI 87 PKI Alternatives 87". There are navigation links for "Menu", "Public Policy", "Help", and "About". The main content is a "Server Certificate Request Form" with the following fields:

Server Certificate Request Form	
Web Server URL (www.example.com)	<input type="text" value="www.alternatives87.eu.org"/>
E-mail Address	<input type="text" value="web@alternatives87.eu.org"/>
Organization (Company/Agency)	<input type="text" value="Alternatives 87"/>
Department/Unit	<input type="text" value="Site web Alternatives 87"/>
Locality (City/County)	<input type="text" value="Limoges"/>
State/Province	<input type="text" value="FR"/>
Country	<input type="text" value="FR"/>
Certificate Password	<input type="password" value="*****"/> Again <input type="password" value="*****"/>
Certificate Life	<input type="text" value="1 Year"/>
<input type="button" value="Submit Request"/>	* All fields are required

At the bottom of the page, it says "PHPKI v0.60 - Copyright 2003 - William E. Roadcap".

Une fois validé, le certificat est généré côté serveur par OpenSSL (qui reçoit ses paramètres par PHPKI) puis il est demandé d'enregistrer le certificat sur notre machine :



Maintenant que nous avons notre certificat, il faut le copier au même endroit que sont stockés les autres certificats du serveur web et éditer le fichier de configuration dudit serveur afin de spécifier l'usage de notre nouveau certificat (cf. *détails de configuration d'Apache en annexe*).

Après redémarrage du serveur web, on obtient ceci :

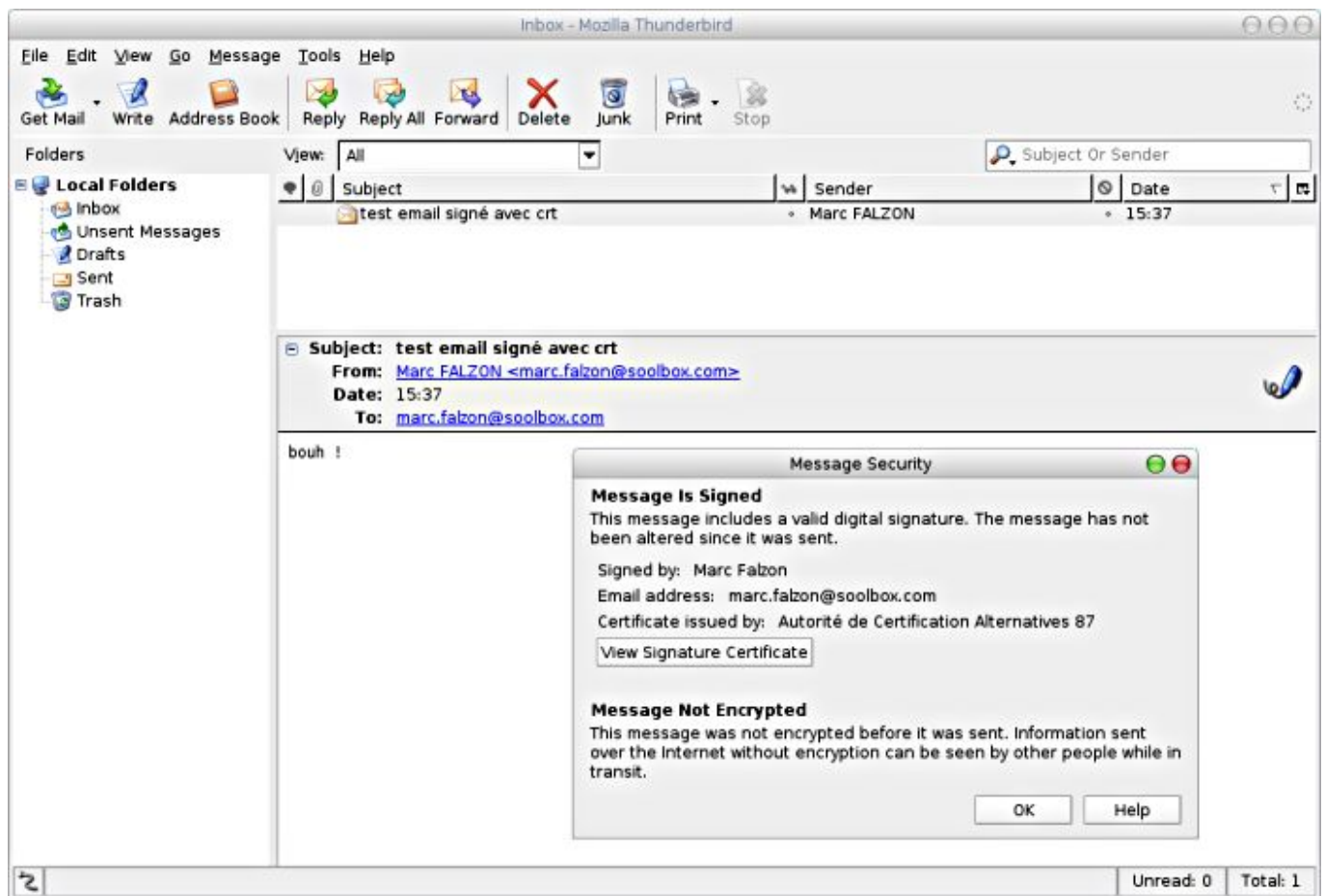


Maintenant, imaginons que nous recevons une demande d'un utilisateur nécessitant un certificat numérique pour signer ses e-mails : une fois dans la section adéquate nous suivons la même

procédure que pour un certificat de site web, à la différence qu'on demande cette fois une adresse e-mail et non plus une adresse web.

Une fois le certificat téléchargé (format **PKCS12**⁽¹⁴⁾ (*Public Key Crypto-System*, méthode de chiffrement / déchiffrement à cryptographie asymétrique) pour les certificats utilisateurs), on lance notre client mail (ici Mozilla Thunderbird) et on importe notre nouveau certificat; après avoir rappelé le mot de passe entré lors de la génération du certificat au niveau de la CA le logiciel nous informe que ce dernier est convenablement intégré à la base de données de certificats interne. Il faut également importer dans le client mail le certificat de la CA sinon les certificats ne seront pas valides (*not trusted*) : ceci fait on peut désormais envoyer et recevoir des e-mails dignes de confiance si on a importé le certificat du correspondant.

Ci-dessous une copie d'écran d'un e-mail signé envoyé à moi-même avec le client mail Mozilla Thunderbird :



En cliquant sur "View Signature Certificate" on constate en effet que l'e-mail que j'ai reçu provient bien de moi-même :



Si un certificat délivré par notre CA venait à être perdu ou volé il est impératif que son propriétaire nous en informe dans les plus brefs délais afin que l'administrateur puisse le révoquer et mettre à jour la CRL.

Pour l'exemple, nous imaginons qu'un pirate se soit infiltré sur le serveur d'Alternatives 87 et ai copié son certificat et sa clé privée; l'administrateur du serveur s'en aperçoit (trop tard) et nous demande de révoquer le certificat actuel : pour ce faire l'administrateur de la PKI se rend dans la section de gestion où il contrôle la "vie" des certificats qu'il a délivrés.

Après révocation du certificat et confirmation de sa révocation, il faut à présent régénérer la CRL pour avertir les utilisateurs de ne plus faire confiance à ce certificat : nous devons nous rendre dans la section (non visible par défaut) de génération de CRL. Ceci fait, il faut importer la nouvelle CRL dans le navigateur (et il appartient AUX UTILISATEURS de mettre à jour fréquemment leur CRL).

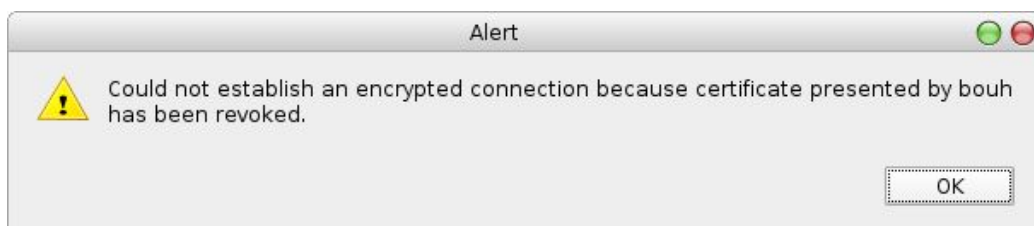
Voici une copie d'écran de liste des certificats générés par notre CA; on le certificat que nous venons de révoquer apparaît en rouge :

The screenshot shows a web browser window with the URL `https://bouh:8443/pki/ca/manage_certs.php?sortfield=issued&ascdec=A&search=&`. The page title is "PHPKi PKI Alternatives 87". The main content is a "CERTIFICATE MANAGEMENT CONTROL PANEL" with a search bar and filter options for Valid, Revoked, and Expired. Below the search bar is a table of certificates with columns for Status, Issued, Expires, User's Name, E-mail, Organization, Department, and Locality. Two certificates are listed: one with status "Revoked" and one with status "Valid".

Status	Issued	Expires	User's Name	E-mail	Organization	Department	Locality				
Revoked	05-mar-31	06-mar-31	www.alternatives87.eu.org	web@alternatives87.eu.org	Alternatives 87	Site web Alternatives 87	Limoges	Display	Renew		
Valid	05-mar-31	06-mar-31	Marc Falzon	marc.falzon@soolbox.com	Soolbox	Soolbox	Paris	Display	Download	Revoke	Renew

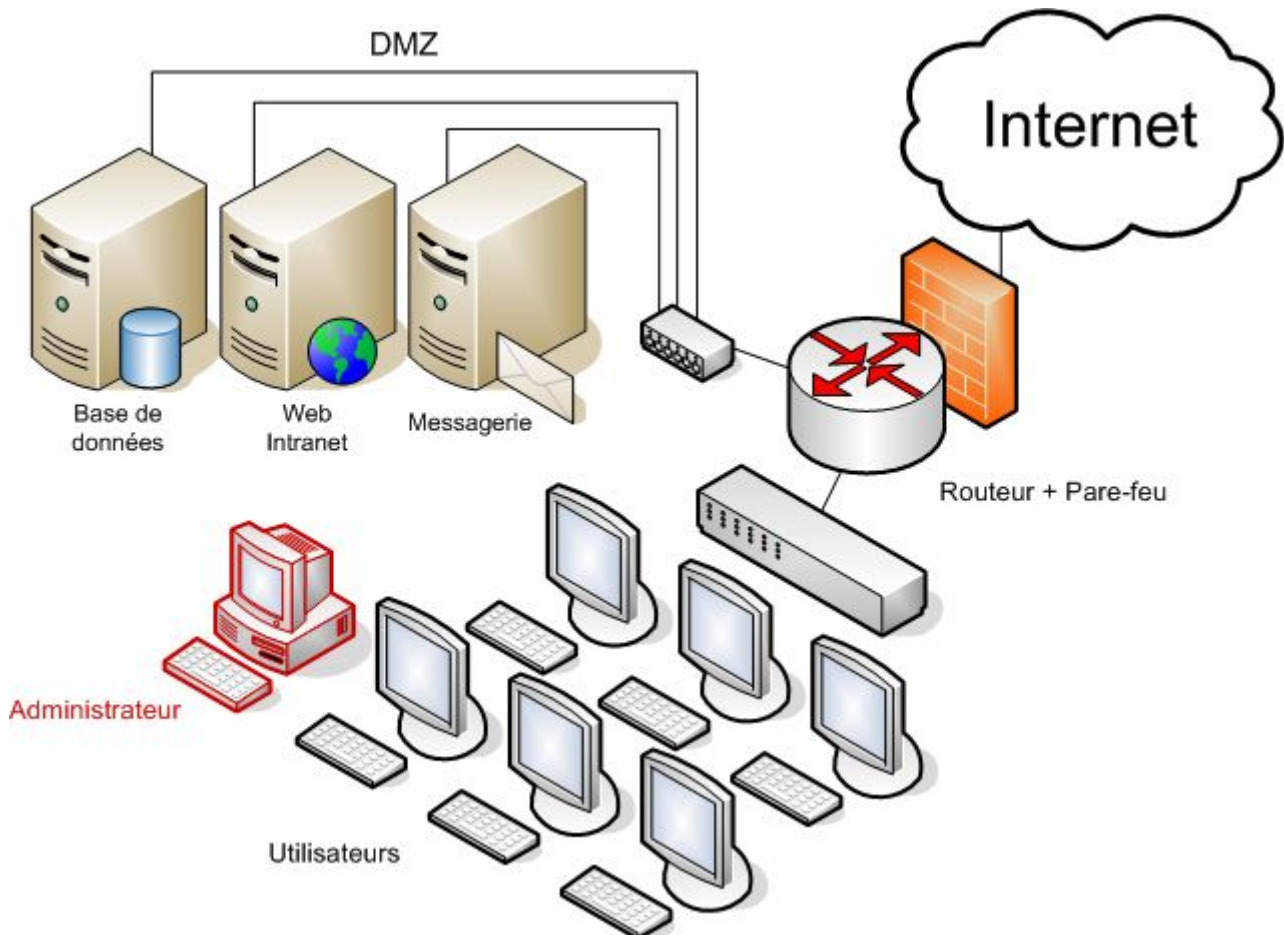
PHPKi v0.60 - Copyright 2003 - William E. Roadcap

Désormais toute tentative de connexion sécurisée avec le serveur web utilisant ce certificat se soldera par ce message :



2.2.e) Intégration d'une PKI dans un réseau

Dans le cadre de mon stage je n'ai pas eu à intégrer mon projet dans un réseau en production : sur la forme tous les tests ont été réalisés localement mais sur le fond tous les protocoles réseau habituels ont été mis en œuvre comme si la PKI avait été physiquement séparée du réseau. Sur un réseau local (LAN, pour *Local Area Network*) traditionnel l'intégration ressemblerait fort à ceci :



Si l'on considère la représentation ci-dessus comme l'architecture du réseau local (selon une topologie physique en étoile et une topologie logique Ethernet) incluant les matériels d'interconnexion de type concentrateur (*hub*) / commutateur (*switch*) et de routage (matériel ou logiciel, statique ou dynamique) d'une PME, on retrouve sur le schéma :

- Une DMZ (*DeMilitarized Zone*, un "bastion" réservé à l'usage du personnel de l'entreprise) contenant les services réseau habituels : messagerie interne (protocoles SMTP/POP3, respectivement envoi et réception de courrier électronique), une éventuelle base de données et le serveur web Intranet (protocoles HTTP/SSL, respectivement transmission de pages web et chiffrement de transmissions) hébergeant notre solution de PKI (à l'accès bien évidemment restreint à l'administrateur du service),
- un sous réseau dédié au personnel technique (administrateur(s) et administratif (utilisateurs));
- ... le tout relié par un routeur matériel ou par une passerelle logicielle (il paraît logique de mettre en place un routage statique dans ce cas de figure étant donné le peu de réseau interconnectés); ce routeur filtre éventuellement les accès venant de l'extérieur.

3) Conclusion

J'ai réussi à déployer ma solution de PKI de la manière dont je l'imaginai lors de sa conception : elle est totalement fonctionnelle dans le sens où elle est en mesure de délivrer des certificats électroniques valides mais pour un usage interne dans notre cas.

En effet la PKI ici déployée est expérimentale car nous nous autoproclamons autorité de certification : cela est valable dans un intranet ou si les personnes à qui l'on délivre des certificats nous font entièrement confiance (principe du *web of trust*). Cependant si nous voulons étendre notre portée de certification au pays puis au monde nous devons nous-même faire la demande d'un certificat (qui sera le certificat racine de notre CA) qui atteste de notre identité et nous confère le droit de délivrer des certificats à des tiers; malgré ce détail la procédure serait identique si ce n'est de renforcer la sécurité du dispositif.

Avant d'aborder mon stage je concevais assez mal la sécurité en entreprise, surtout sur le plan organisationnel. De plus je n'avais que de vagues notions en cryptographie. Ce projet m'a permis de combler ces lacunes même si le point de vue organisationnel nécessaire pour concevoir et déployer une solution de PKI "grandeur nature" en entreprise n'était pas assez concret : de ce fait malgré que les informations fournies (nom de l'organisation, e-mails et adresses web) soient authentiques le projet ressemble plus à une maquette qu'à une réelle application.

Ce service pourrait être utile dans une entreprise comme **Direct Gestion**, où j'ai effectué mon stage l'année dernière qui tient un site web de type portail d'annonces pour les agences immobilières ou directement pour les particuliers : on pourrait imaginer qu'elle crée sa propre autorité de certification afin de délivrer des certificats aux responsables d'agences immobilières pour qu'ils n'aient pas à s'authentifier à chaque fois qu'ils ont des annonces à poster sur le site, et bien évidemment pour le site web lui-même ...

4) Références

- 1) http://fr.wikipedia.org/wiki/Logiciel_libre
- 2) <http://fr.wikipedia.org/wiki/ROT13> - <http://www.rot13.com/>
- 3) <http://pgp.mit.edu/>
- 4) <http://www.legifrance.gouv.fr/WAspad/UnTexteDeJorf?numjo=JUSX9900020L>
- 5) <http://www.itu.int/>
- 6) <http://www.ietf.org/html.charters/pkix-charter.html>
- 7) <http://www.commentcamarche.net/crypto/ssl.php3>
- 8) Version utilisée : 0.9.7e - <http://www.openssl.org/>
- 9) Version utilisée : 1.3.33 - <http://httpd.apache.org/>
- 10) Version utilisée : 2.8.22 - <http://www.modssl.org/>
- 11) http://fr.wikipedia.org/wiki/Transport_Layer_Security
- 12) Version utilisée : 5.0.3 - <http://www.php.net/>
- 13) <http://sf.net/projects/phpki> - <http://phpki.sourceforge.net/phpki/> (demo)
- 14) <http://en.wikipedia.org/wiki/PKCS>