Université de Mons-Hainaut
Rapport de stage
Deuxième licence en informatique

Responsable en entreprise :
Mr. Christophe PONSARD

Réalisation d'une application eID

Jérôme JONCKERS
<jerome.jonckers@umh.ac.be>

Année académique 2006-2007

Table des matières

1 Introduction

Comment imaginer obtenir une certification en informatique sans avoir approché une entreprise du secteur ?

Cela semble utopique, et, à tout le moins, fort peu sérieux.

A l'issue de la première année de licence, il est important d'être correctement informé de ce que sont réellement les métiers de l'informatique. Ceux-ci sont nombreux, et touchent à de multiples domaines, allant de la recherche pure à la mise en service d'applications pratiques, en passant par la réalisation de logiciels applicatifs, la gestion de bases de données, les métiers de l'Internet,...

Le monde de l'application concrète, la mise au service du plus grand nombre des outils proposés par les dernières réalisations technologiques, l'usage de l'informatique dans la vie courante sont des domaines qui offrent de larges perspectives.

Le CETIC m'a accepté comme stagiaire, et m'a amené à concevoir une application pratique utilisant les caractéristiques de la carte d'identité électronique, en mettant en avant la sécurité qu'elle présente.

2 Présentation de l'entreprise

En 2001 à l'initiative de la Faculté Polytechnique de Mons (FPMs), des Facultés Universitaires Notre-Dame de la Paix de Namur (FUNDP) et de l'Université Catholique de Louvain (UCL), le *CETIC, Centre d'Excellence en Technologies de l'Information et de la Communication*, a été créé grâce aux fonds structurels européens Objectif 1 dont a bénéficié la province du Hainaut.
Aujourd'hui, le CETIC est un centre de recherche appliquée agréé par la Région wallonne. Ses activités de recherche sont financées grâce au soutien de la Région wallonne, de la Commission européenne et des entreprises privées.

Le CETIC est actif en recherche appliquée en génie logiciel, en technologies GRID et en systèmes électroniques. Il est un agent de connexion, de transfert de technologies entre recherche universitaire et entreprises.

Le centre se considère comme étant « au service des entreprises ». Il se positionne comme un partenaire actif en transfert de technologies en génie logiciel et en électronique. Dans le cadre de ses activités de recherche, le CETIC s'appuie régulièrement et fortement sur ses relations avec les laboratoires universitaires.

Les domaines spécifiques de recherche se partagent entre :
- La qualité des processus et des produits logiciels.
- L'ingénierie des cahiers des charges informatiques.
- Les systèmes répartis.
- L'ingénierie des bases de données.
- Les systèmes électroniques.
- Le logiciel libre.

Le CETIC est particulièrement actif au niveau du 6ème Programme-Cadre de l'Union Européenne, en participant à des projets intégrés et des réseaux d'excellence.

L'entreprise organise régulièrement des évènements (journée d'information, d'étude, groupes de discussion,...) visant à faire connaître le résultat de ses recherches et à créer une interface entre le monde de la recherche universitaire et le monde du travail.

Le CETIC est à l'écoute des besoins d'innovation technologique des entreprises, et propose de les accompagner dans leurs projets en leur fournissant un soutien personnalisé.

Fig. 1 – Le CETIC

Il se destine également au développement de prototypes de nouveaux produits et services destinés à être exploités commercialement à travers des entreprises existantes ou à mettre en place.

Au sein du CETIC, l'atelier *FAUST* a pour objectif d'assurer le développement de cahiers des charges de haute qualité, notamment dans le domaine de systèmes critiques. Il s'intéresse particulièrement aux activités supportées, à leurs composants et à la manière dont ils s'intègrent harmonieusement entre eux et avec l'environnement logiciel existant.

Des systèmes logiciels complexes sont introduits dans de plus en plus de domaines de notre monde actuel : automobiles, électroménager, contrôle industriel, transactions financières, appareils médicaux... De plus en plus souvent leur rôle est critique : leur dysfonctionnement peut mettre en danger des vies ou des organisations.

Conscient qu'assurer un développement de qualité est impératif, l'atelier FAUST[1] s'occupe principalement de l'ingénierie des exigences. Il tente ainsi d'éviter que des projets ne dérapent à cause d'une mauvaise prise en charge des étapes critiques.

L'objectif de l'atelier FAUST est de proposer une suite d'outils permettant de mieux soutenir les activités importantes lors de cette phase :

- la **vérification** : s'assurer que les exigences sont consistantes, non ambiguës, robustes par rapport à des menaces,... ;
- la **validation** : s'assurer que les exigences correspondent à ce que les parties prenantes (commanditaires, utilisateurs) souhaitent ;
- l'**acceptation** : s'assurer que le système délivré correspond bien aux exigences formulées dans le cahier des charges ;

C'est au sein de cette structure que le stage s'est déroulé.

FIG. 2 – Le CETIC

[1] http ://faust.cetic.be

3 Le stage

3.1 Problème posé

3.1.1 Présentation de la carte d'identité électronique

L'implémentation de la carte d'identité électronique (eID) fait partie d'un projet e-Government de simplification administrative et de modernisation des services publics belges.

La carte d'identité électronique reste avant tout la preuve officielle de l'identité d'une personne, mais elle permet aussi l'identification électronique à distance. On peut faire la distinction entre l'identification simple et l'authentification qui livre une preuve effective de l'identité du titulaire. Une autre fonctionnalité est le fait, pour chaque titulaire, de disposer d'une signature électronique à valeur légale lui permettant de signer des documents électroniquement. Il en résulte un service rapide et orienté client sans mettre les données privées du titulaire en danger.

La carte d'identité est remise au citoyen par l'intermédiaire des administrations communales, avec lesquelles le citoyen est en contact étroit. Les administrations communales sont connectées électroniquement au gouvernement fédéral, aux sociétés agréées et aux autorités de certification participant au projet eID à divers niveaux. Le Registre National a un rôle clé aussi bien dans l'organisation que dans la surveillance du système.

3.1.2 La carte eID

Cette nouvelle pièce d'identité a le format d'une carte de banque sur laquelle sont imprimées les informations de base d'identité telles que les coordonnées de la personne et sa photo. La carte eID contient également une puce reprenant les mêmes informations ainsi que deux certificats numériques. Ces informations sont invisibles, uniquement lisibles électroniquement :

- les mêmes informations que celles lisibles sur la carte ;
- l'adresse du titulaire de la carte ;
- les clés d'identité et de signature ;
- les certificats d'identité et de signature ;
- le prestataire de service de certification ;
- l'information nécessaire à l'authentification de la carte ;
- l'information nécessaire à la protection des données visibles de manière électronique figurant sur la carte ;
- l'information nécessaire à l'utilisation des certificats qualifiés y afférents ;

FIG. 3 – Un modèle de carte d'identité électronique

Les certificats numériques standard permettent aux citoyens de signer électroniquement et de s'identifier lors de transactions électroniques. Ils conviennent également aux différentes applications e-Gov ainsi qu'à une utilisation à des fins privées (envoi de recommandés électroniques, e-mails signés, ...), authentification via des webbrowsers (e-Banking, e-Contracting,...), etc. L'émergence de ces applications combinée à la reconnaissance légale de la signature électronique remplaceront rapidement et en toute sécurité une partie des flux de documents papier et leur équivalent électronique.

Il est à noter que les mécanismes de sécurité de la carte eID sont tels que, à tout moment, les informations de la puce eID sont identiques aux informations du Registre National concernant le titulaire de la carte. De cette façon la carte eID contient l'adresse correcte, officielle du citoyen et l'orthographe correcte des prénoms et nom du titulaire de la carte.

3.1.3 La sécurité

De nos jours, la plupart des gens utilisent un nom d'utilisateur et un mot de passe pour s'identifier sur un site Web ou sur une application en ligne. On connaît les limites de ce procédé dont le niveau de sécurité est insuffisant pour des applications critiques comme le home-banking.

La sécurisation carte eID, qui par définition est strictement individuelle et ne peut être utilisée que par une seule personne, est basée sur le système PKI signifiant "Public Key Infrastructure".

Ce système de sécurisation des services de communication électronique est basé sur le principe des "clés". L'utilisateur dispose d'une paire de clés (une clé "publique" et une clé "privée") avec lesquelles il peut sécuriser et authentifier ses communications électroniques.

La sécurisation et l'authentification via la paire de clés numériques sont basées sur le principe de la cryptographie asymétrique : ce qui est "chiffré" avec la clé privée peut seulement être "déchiffré" avec la clé publique correspondante, et inversement. La clé privée peut seulement être utilisée par le titulaire du couple de clés et doit donc être conservée en sécurité. La clé publique doit pouvoir être utilisée par n'importe quelle personne de contact du titulaire de la clé privée, et donc être accessible publiquement dans une banque de données.

3.1.4 Les certificats

Si les paires de clés sont utilisées pour authentifier la communication électronique (c'est-à-dire pour garantir que la communication provient d'une source déterminée et n'est pas modifiée par rapport au moment de l'envoi par cette source), elles sont liées de manière univoque à un ou plusieurs certificats, où figurent l'identité et/ou une ou plusieurs qualités ou attributs de cette personne.

Les paires de clés numériques mises à disposition par une infrastructure PKI peuvent être utilisées pour différentes finalités, telles que le cryptage de messages (pour en garantir la confidentialité et l'intégrité), l'authentification lors de la consultation de sites web, l'utilisation de signatures électroniques ayant valeur probante en vertu du Code civil,... De manière générale, on préfère néanmoins, pour des raisons de sécurité, que la paire de clés utilisée pour apposer une signature électronique ayant valeur probante ne soit pas également utilisée pour le cryptage ou l'authentification dans l'accès aux sites web. Ceci est exactement la raison pour laquelle deux certificats différents ont été mis sur la carte d'identité électronique.

Lors de la délivrance de certificats, on peut distinguer deux rôles :
- celui de l'autorité d'enregistrement (AE) : l'AE est le « guichet » où le certificat est demandé ; elle vérifie si l'identité ou la compétence communiquée est correcte ; si tel est le cas, elle approuve la demande et en informe l'autorité de certification ;
- celui de l'autorité de certification (AC) : l'AC produit un certificat sur la base de l'information qu'il a reçue de l'AE, certificat qu'il lie à une paire de clés et qui indique ce que la paire de clés prouve désormais ;

Par ailleurs, on distingue également le rôle de Directory Service (DS), qui veille à la publication des clés publiques et du statut des certificats qui y sont liés et qui sont émis par une autorité de certification et éventuellement du contenu de ces certificats et ce, de manière contrôlée.

3.1.5 Le projet

Le fil conducteur du stage est l'utilisation simple des caractéristiques de la carte d'identité électronique « eID ».

La conception d'une application tirant parti des données contenues dans la puce des eID impose des contraintes de sécurité. Celles-ci devront être rencontrées et solutionnées de manière optimale.
Pour cela, il faudra, dans un premier temps, résoudre le problème de configuration d'un serveur afin qu'il puisse traiter des demandes de connexions sécurisées, en utilisant l'eID.
Ce serveur nécessite une configuration particulière, en reverse-proxy.

La réalisation d'une *FAQ*[2] est également proposée afin d'apporter des réponses pratiques et précises aux difficultés que pourraient rencontrer les utilisateurs.

L'application à réaliser consiste en l'implémentation d'un programme de « chat » sécurisé, à accès restreint et permettant à tout utilisateur connecté de vérifier l'identité des personnes avec qui il discute. Celle-ci sera testée sur le serveur préalablement configuré.

[2] Foire Aux Questions

3.2 Les outils

Une première difficulté consiste à déterminer les outils, les méthodes à utiliser pour développer un programme utilisant l'eID comme moyen d'identification.

En effet, peu d'applications existent et sont utilisables actuellement. Il faut visiter le site de l'UCVW (Union des Villes et Communes de Wallonie)[6] pour trouver un dossier traitant du sujet.

Pour pouvoir utiliser sa carte d'identité (afin de s'authentifier sur Internet ou pour pouvoir signer des documents ou des mails), il faut :

- Posséder une carte d'identité sur laquelle les fonctionnalités d'authentification et de signature sont actives (les personnes qui ont désactivé ces fonctions lors de la réception de la carte devront les faire activer dans leur commune au moyen du code *PUK*[3]) ;
- Avoir connaissance du code *PIN*[4] de la carte (les personnes qui ont oublié leur code PIN devront se présenter dans leur commune pour le réinitialiser grâce au code PUK) ;
- Disposer d'un ordinateur ;
- Disposer d'un lecteur de carte prévu à cet effet. Les modèles de carte à puce pouvant différer, vous ne pourrez pas lire la carte d'identité électronique sur un lecteur de carte bancaire qui n'a pas été prévu pour cela. ;
- Disposer d'une connexion Internet (si l'usage qui est fait de la carte n'est pas local) ;
- Installer le logiciel permettant à l'ordinateur de communiquer avec le lecteur de carte. Ce logiciel est appelé middleware. ;

3.2.1 Middleware

Le middleware est un logiciel adapté à la carte d'identité électronique belge qui permet de communiquer avec le lecteur de carte. Il est mis à disposition par *FEDICT*[5]. Il est parfois également appelé Run-time.
Le middleware se base sur le projet libre OpenSC qui a été adapté pour la carte d'identité électronique belge. Le code adapté a été publié et a été en partie inclus dans le projet standard. Les librairies OpenSC sont aussi utilisées pour les autres cartes d'identité électroniques adoptées par certains pays. La version la plus récente du middleware est la 2.5.9. Elle est téléchargeable. Le téléchargement est proposé pour 3 systèmes d'exploitation (Windows, Linux, Mac). Un guide d'installation y est fourni pour chacun des systèmes.

[3] PUK : Personal Unlocking Code
[4] PIN : Personal Identification Number
[5] Le Service public fédéral Technologie de l'Information et de la Communication (http ://www.fedict.be)

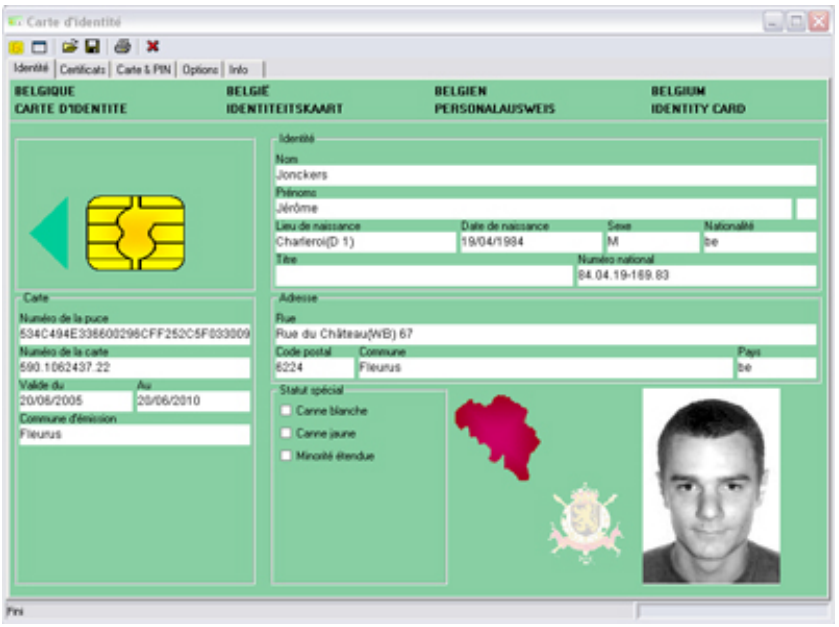


FIG. 4 – Middleware Belgique-eID

3.2.2 HTTPS

Pour incorporer l'authentification sur un site Internet afin de vérifier l'identité de la personne qui se connecte, le serveur web doit demander une authentification via le certificat d'authentification de la carte d'identité électronique.

Pour réaliser cela, il est nécessaire d'effectuer les deux opérations suivantes :

1. Le serveur web doit être configuré pour fonctionner en HTTPS[6].

   La couche de sécurisation est réalisée grâce à la technologie SSL[7], qui repose sur un procédé de cryptographie par clé publique, c'est-à-dire celui sur lequel sont basés les certificats.
   Le principe de cette sécurisation consiste, après une étape d'authentification, à établir un canal de communication sécurisé (chiffré) entre le client et le serveur.
   En d'autres mots, les données sont cryptées lorsqu'elles transitent par Internet et ne sont donc pas lisibles par un tiers qui intercepterait la communication.

2. La configuration du serveur https doit spécifier qu'un certificat client est requis.

   Le fait de demander un certificat client fait partie d'un processus d'authentification des personnes qui accèdent au site web.
   Le fonctionnement reste assez identique par rapport au point précédent hormis que le serveur exige un certificat client. Le navigateur client propose de choisir un certificat personnel dans le magasin de certificats. L'utilisateur doit donc avoir placé son certificat d'authentification dans le magasin du navigateur.
   Si le certificat de la carte est choisi, le navigateur doit accéder à la carte d'identité électronique pour utiliser la clé privée qui est sur celle-ci afin de renvoyer une signature accompagnée du

[6] HTTP sécurisé
[7] Secure Socket Layer

certificat d'authentification au serveur.
Lorsque le serveur web reçoit la signature accompagnée du certificat d'authentification client, il déchiffre la signature avec le certificat, récupère les données présentes dans celui-ci et doit vérifier sa validité (OCSP[8] ou CRL[9]) avant d'initier la connexion https.
Si le statut des certificats n'est pas valide, le serveur renvoie une page d'erreur, ou un message est directement affiché par le navigateur.
Si le statut est valide, la page demandée est renvoyée.

3.2.3 Apache

Le logiciel Apache HTTP Server, mieux connu sous le nom Apache, est un serveur http, logiciel servant des requêtes respectant le protocole de communication client-serveur HyperText Transfer Protocol (HTTP), qui a été développé pour le World Wide Web.
Un ordinateur sur lequel fonctionne un serveur HTTP est appelé serveur Web.

Le Projet Apache est un logiciel d'implémentation d'un serveur web. Les codes sources de ce logiciel sont totalement gratuits et disponibles sur le web. Le projet n'a pas été engagé par une seule individualité mais par un véritable groupe de travail constitué de volontaires du monde entier qui ont participé non seulement dans son développement mais aussi dans la constitution de sa documentation. Ces volontaires sont connus sous le nom de Apache Group. Il faut également ajouter la contribution de centaines d'utilisateurs qui ont apporté idées, codes et documentations supplémentaires pour le projet.
Apache 1.0 a été distribué au 1er décembre 1995 et totalement gratuitement. La version actuelle est Apache 2.0.

Apache peut également faire office de proxy intermédiaire, que l'on nomme proxy inverse ou reverse proxy. Par ce biais, il est possible d'interdire le relayage des requêtes vers certains serveurs distants, ou même de gérer plusieurs serveurs web à partir d'une seule adresse (serveurs en grappe). Il est également possible d'acheminer d'autres protocoles que HTTP (telnet, news, etc.).

Il existe plusieurs moyens de sécuriser l'accès aux ressources d'un serveur Apache :
- authentification des utilisateurs par les modules standards (mod_auth) et les modules tiers (LDAP, Smb, Kerberos, Radius, Oracle, etc.) ;
- sécurisation des requêtes par le protocole SSL (mod_ssl + certificat) ;

La version d'Apache utilisée pour ce stage est une version modifiée par le FEDICT, incluant les outils nécessaires à l'ajout des en-têtes eID ainsi que la validation OCSP des certificats.

3.2.4 SSL

SSL (Secure Socket Layer) est un système qui permet d'échanger des informations entre 2 ordinateurs de façon sûre.

SSL assure 3 choses :
- Confidentialité : il est impossible d'espionner les informations échangées ;
- Intégrité : il est impossible de truquer les informations échangées ;
- Authentification : il permet de s'assurer de l'identité du programme, de la personne ou de l'entreprise avec laquelle on communique ;

[8] Protocole de vérification en ligne de certificat : Online Certificate Status Protocol
[9] Liste de certificats révoqués : Certification Revocation List

SSL est un complément à TCP/IP et permet (potentiellement) de sécuriser n'importe quel protocole ou programme utilisant TCP/IP.
SSL a été créé et développé par la société Netscape et RSA Security. On trouve désormais des versions Opensource ainsi qu'un protocole libre similaire, TLS.

Les principaux avantages de SSL sont :
- SSL est standardisé ;
- Il existe une version libre de SSL, OpenSSL utilisable sans payer de royalties ;
- OpenSSL est opensource, ce qui signifie que tout un chacun peut contrôler et vérifier le code source, sans danger puisque le secret réside dans les clés de chiffrement, pas dans l'algorithme lui-même ;
- SSL a été cryptanalysé, plus analysé que tous ses concurrents. SSL a été passé en revue par de nombreux spécialistes en cryptographique. On peut donc le considérer comme sûr ;
- Il est très répandu et l'on peut facilement créer des programmes qui dialogueront avec d'autres programmes utilisant SSL ;

SSL consiste en 2 protocoles :
- SSL Handshake protocol : avant de communiquer, les 2 programmes SSL négocient des clés et des protocoles de chiffrement communs ;
- SSL Record protocol : une fois négociés, ils chiffrent toutes les informations échangées et effectuent divers contrôles ;

Pour protéger les communications, SSL utilise :
- un système de chiffrement asymétrique (comme RSA ou Diffie Hellman) ;
- un système de chiffrement symétrique (DES, 3DES, IDEA, RC4...) en utilisant les clés de session pour chiffrer les données ;
- un système de signature cryptographique des messages (HMAC, utilisant MD5, SHA...) pour s'assurer que les messages ne sont pas corrompus ;

C'est lors de la négociation SSL que le client et le serveur choisissent des systèmes communs (chiffrement asymétrique, symétrique, signature et longueur de clé).
Dans un navigateur, la liste des systèmes utilisés peut être visualisée en plaçant le curseur sur le petit cadenas qui apparaît dans une page en HTTPS.

Lors d'une négociation SSL, il faut s'assurer de l'identité de la personne avec qui on communique. Comment être sûr que le serveur auquel on parle est bien celui qu'il prétend être ?
C'est là qu'interviennent les certificats. Au moment de se connecter sur un serveur web sécurisé, ce dernier envoie un certificat contenant le nom de l'entreprise, son adresse, etc. C'est une sorte de pièce d'identité.

Pour vérifier l'authenticité de cette pièce d'identité, les PKI (Public Key Infrastructure), des sociétés externes (auxquelles vous faites implicitement confiance), vont vérifier l'authenticité du certificat.
Ces PKI signent cryptographiquement les certificats des entreprises.

SSL peut être utilisé pour sécuriser pratiquement n'importe quel protocole utilisant TCP/IP. Certains protocoles ont été spécialement modifiés pour supporter SSL :
- HTTPS : c'est HTTP+SSL. Ce protocole est inclus dans pratiquement tous les navigateurs, et vous permet (par exemple) de consulter vos comptes bancaires par le web de façon sécurisée ;
- FTPS est une extension de FTP (File Transfer Protocol) utilisant SSL ;

– SSH (Secure Shell) : c'est une sorte de telnet (ou rlogin) sécurisé. Cela permet de se connecter à un ordinateur distant de façon sûre et d'avoir une ligne de commande. SSH possède des extensions pour sécuriser d'autres protocoles (FTP, POP3 ou même X Windows) ;

Il est possible de sécuriser des protocoles en créant des tunnels SSL. Une fois le tunnel créé, vous pouvez faire passer n'importe quel protocole dedans (SMTP, POP3, HTTP, NNTP, ...). Toutes les données échangées sont automatiquement chiffrées. On peut faire cela avec des outils comme STunnel ou SSH.

Une fois que la fonctionnalité d'authentification est implémentée sur un site internet, le travail commence seulement ! Il faut en effet encore développer et mettre en place les services à proposer au citoyen via le site internet.

3.2.5 Java

Java est un langage qui connaît un grand succès. Une multitude de petits (ou gros) détails font que finalement Java est un choix judicieux :

– Java est un langage orienté objet : la brique de base du programme est donc l'objet, instance d'une classe ;
– Java est portable et un programme, une fois compilé fonctionnera aussi bien sous des stations Unix, que sous Windows ou autre. Une des caractéristiques principales est que le code Java est compilé pour une machine dite virtuelle (c'est-à-dire qui n'a pas forcément d'existence physique, mais son concept peut être reproduit sur une machine cette fois-ci réelle). Le code machine résultant est nommé ByteCode. Lors de l'exécution celui-ci est transformé en un code machine compréhensible par le microprocesseur utilisé ;
– D'un point de vue langage, Java intègre tout ce que l'on sait faire de mieux en matière de langage de programmation, tout en évacuant les difficultés apportées par d'autres langages dont il est issu (C et C++), comme la gestion de la mémoire qui n'est plus à charge du programmeur, ... ;
– Java est multithreadé, distribué, robuste et sûr ;

L'API[10] de Java est très riche et différents packages permettent d'accéder au réseau, aux entrées/sorties, aux différents composants graphiques ...

– Pour la mise en oeuvre d'interfaces graphiques, deux packages principaux vous sont offerts :
  * L'AWT[11] fournit des méthodes d'accès à toutes les ressources graphiques de la machine. Les éléments graphiques utilisés sont ceux du système d'exploitation hôte. Cependant, les concepteurs de l'AWT se sont assurés que tous les comportements proposés au sein de cette librairie soient supportés par tous les environnements pouvant supporter Java.
  * La librairie Swing, quant à elle, propose des composants graphiques totalement pris en charge et dessinés par la JVM[12]. Cela permet de garantir que l'applicatif aura le même look visuel, ce quel que soit le système d'exploitation hôte. En conséquence, SWING est plus riche que l'AWT, mais aussi moins efficace en terme de temps d'exécution.
– Le package java.applet permet la réalisation d'applets (application pouvant s'exécuter dans un navigateur Internet.Il doit être utilisé conjointement avec l'AWT, en effet, dans un document HTML, l'applet occupera une zone graphique.

[10] Application Programming Interface
[11] Abstract Window Toolkit
[12] Java Virtual Machine : machine virtuelle permettant d'interpréter et d'exécuter le bytecode Java.

- Le package java.beans est réservé à la programmation orientée composants. Les Java Beans sont des composants logiciels réutilisables et visuellement manipulables dans un atelier de programmation.
- Le package java.lang fournit un maximum de classes liées au langage : manipulation de chaînes de caractères, manipulation des types de bases, récupération de métaclasses, prise en charge des flots standards de l'application. On y trouve aussi la classe Object , classe mère de tout objet Java. Ce package est par défaut directement accessible, contrairement aux autres packages qu'il faut explicitement importer.
- Le package java.io est un ensemble de classes permettant la gestion des opérations d'entrées/sorties.
- Le package java.sql regroupe les fonctionnalités permettant d'accéder à des bases de données diverses. L'utilisation en est très simple, quoique des bases en SQL[13] sont requises.
- Le package java.rmi (Remote Method Invocation) permet de mettre en place des applications réparties. Dans de telles applications, le code n'est pas localisé sur une unique machine, mais au contraire, déployé sur plusieurs.

Le langage Java trouve ses origines dans les années 1990. A cette époque, quelques ingénieurs de SUN Microsystems ont commencé à parler d'un projet d'environnement indépendant du hardware pouvant facilement permettre la programmation d'appareils aussi variés que les téléviseurs, les magnétoscopes,...

En 1995, un premier essai, Oak, est renommé Java et est soumis à la communauté Internet grandissante. Une machine virtuelle, un compilateur ainsi que de nombreuses spécifications sont données gratuitement et Java attaque une conquête fulgurante. Aujourd'hui, après de nombreuses améliorations Java n'est plus uniquement une solution liée à Internet, et de plus en plus de sociétés (ou de particuliers) utilisent ce langage pour leurs développements.

3.2.6 Eclipse

Eclipse est un environnement de développement intégré extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en oeuvre n'importe quel langage de programmation. L'application est écrite en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.
Sa spécificité vient du fait de son architecture, toutes les fonctionnalités de cet atelier logiciel sont développées en tant que plug-in. Deux raisons principales ont amené à considérer Eclipse comme socle pour des applications clientes :

- la qualité d'Eclipse : l'environnement de développement Eclipse s'est imposé par sa fiabilité et la qualité de ses interfaces graphiques. L'utilisation massive de l'environnement de développement Eclipse par la communauté des développeurs Java ainsi que son utilisation comme base de produits commerciaux (WebSphere Studio, SAP NetWeaver Studio...) ont permis d'éprouver le framework Eclipse.
- la disponibilité en open source du framework Eclipse.

3.2.7 CVS

CVS, Concurrent Versions System est un outil d'aide au développement de logiciels. Très présent dans le monde des programmeurs Open Source, il est utile à la communauté des développeurs pour plusieurs raisons.

[13] Structured Query Language

Tout d'abord, comme cela transparaît dans son nom, CVS permet une gestion efficace et riche des différentes versions pour un projet logiciel. Cela passe notamment par la mise en place d'un suivi, et par conséquent d'un historique, pour l'ensemble des fichiers appartenant au projet.
Il est également à noter que la gestion de versions se fait autant au niveau de l'ensemble du projet qu'au niveau de chaque fichier pris séparément.

L'un des autres points forts de CVS est de permettre et de favoriser un développement en équipe. En effet, il permet un stockage centralisé du code source sur un serveur et gère les accès concurrents sur les fichiers de développement. Ce qui distingue CVS d'autres outils de développement collaboratif (notamment RCS) est la possibilité pour les développeurs d'accéder en même temps à un même fichier pour le modifier, avec une prise en charge des modifications lorsque celles-ci ne génèrent pas de conflits.

Enfin, comme tout outil de gestion de configuration, CVS s'intègre au processus qualité et permet non seulement d'introduire des règles pour une équipe, mais également de conserver un historique complet de ce qui a été fait.

3.3 Le travail

3.3.1 Configuration du serveur

La première opération consiste à configurer le serveur Apache en « reverse proxy », c'est-à-dire jouant un rôle de passerelle entre le serveur web applicatif et le navigateur client. Le serveur "reverse proxy" utilise la couche SSL pour créer une communication https, demande un certificat client au navigateur et vérifie si ce certificat est valide (OCSP ou CRL). Les données intéressantes du certificat client sont passées au serveur applicatif via un en-tête html rajouté par le serveur Apache.

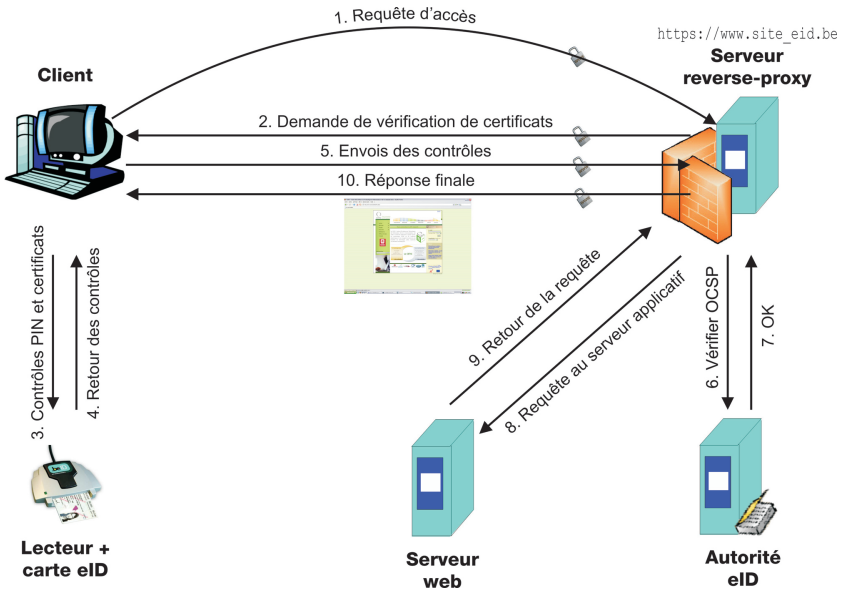


FIG. 5 – Schéma d'une requête sécurisée eID

Le schéma ci-dessus montre clairement le chemin suivi entre la requête initiale et la réponse finale. Le processus est interrompu au niveau du serveur Apache si la réponse de la vérification OCSP est négative.

Après avoir réuni le matériel nécessaire (lecteur de carte eID, une carte eID,...) l'installation du serveur Apache est entamée, en suivant les indications fournies par le site de l'Union des Villes et Communes de Wallonie.

L'installation est réalisée sur un ordinateur travaillant sous une version de Linux Mandrake.

La version 2.0.54 du serveur web Apache est téléchargée. La préférence est donnée à la version modifiée par le Fedict pour inclure la fonctionnalité OCSP. Le SSL utilisé est OpenSSL, version 0.9.8b.

– Installation d'Apache

  L'archive est décompressée dans un répertoire. Après s'être placé dans ce répertoire, l'utilisateur lance les commandes d'installation suivantes :

```
./configure --prefix=/usr/local/apache2
```

```
        --enable-modules="ssl proxy headers rewrite"
        --with-ssl=/usr/local/ssl
make
make install
```

L'installation se déroule normalement.

Pour pouvoir utiliser le système sans devoir commander un certificat auprès d'une autorité de certification, il est nécessaire de créer un certificat de test. Ce dernier ne doit en aucun cas être utilisé pour un service en production. La procédure est détaillée sur le site de l'UVCW, mais manque de précision. Celle-ci a donc été complétée comme suit :

– Création du certificat serveur

On génère la clé privée avec la commande :

```
openssl genrsa 1024 > server.key
```

La sortie attendue est :

```
Generating RSA private key, 1024 bit long modulus
.............++++++
.....++++++
e is 65537 (0x10001)
```

Ceci a pour effet de créer une clé SSL (fichier server.key) : c'est la clé privée.

A partir de cette clé, un fichier de demande de signature de certificat (CSR : Certificate Signing Request) va être créé.

On génère la demande de certificat avec la commande :

```
openssl req -new -key server.key > server.csr
```

Le système va demander de saisir des champs. Il faudra les remplir en adaptant les données, sauf le champ « Common Name », qui doit être identique au nom d'hôte du serveur virtuel.

```
Country Name (2 letter code) [GB]:BE
State or Province Name (full name) [Berkshire]:Belgium
Locality Name (eg, city) [Newbury]:Gosselies
Organization Name (eg, company) [My Company Ltd]:CETIC
Organizational Unit Name (eg, section) []:ied.CETIC
Common Name (eg, your name or your server's hostname) []:pt-mda.cetic.be
Email Address []:
```

Ce n'est pas la peine de saisir d'autres « extra attributes ».

Ceci a pour effet de créer le formulaire de demande de certificat (fichier server.csr) à partir de la clé privée préalablement créée. Ce fichier contient la clé publique à certifier.

– Création du certificat de l'autorité de certification

On génère la clé privée de l'autorité de certification avec la commande :

```
openssl genrsa -des3 1024 > ca.key
```

Cette commande a pour effet de créer la clé privée de l'autorité de certification. Ici il vaut mieux ajouter l'option -des3 qui introduit l'usage d'une « passphrase ».
Cette passphrase sera demandée à chaque utilisation de la clé de l'autorité de certification.

On crée ensuite, à partir de la clé privée, un certificat x509 pour une durée de validité d'un an, et auto-signé :

```
openssl req -new -x509 -days 365 -key ca.key > ca.crt
```

Il faut saisir la passphrase puisque « ca.key » est utilisé. On donne les renseignements concernant cette fois l'autorité de certification. Le nom de « Common Name » doit être différent de celui qui a été donné pour la clé premièrement créée.

C'est le certificat d'autorité de certification qui va permettre de signer les certificats créés.

– Signature du certificat serveur par le CA (Certificate Authority)

On signe la demande de certificat par la commande :

```
openssl x509 -req -in server.csr -out server.crt -CA ca.crt
             -CAkey ca.key -CAcreateserial -CAserial ca.srl
```

Le certificat signé est le fichier « server.crt ».

La sortie attendue est :

```
Signature ok
subject=/C=BE/ST=Belgium/L=Gosselies/O=CETIC/OU=ied.CETIC/CN=pt-mda.cetic.be
Getting CA Private Key
Enter pass phrase for ca.key :
```

– Configuration d'Apache

Il faut ensuite configurer le serveur Apache, en modifiant le fichier de configuration.

Quelques lignes importantes sont détaillées ci-dessous. Le fichier de configuration est détaillé en annexe (Annexe A).

`<VirtualHost _default_:numero_de_port>`
» définition d'un virtualHost sur le numéro de port sur lequel le serveur écoute

`SSLEngine on`
» switch du moteur SSL (doit être mis à « on » pour que l'authentification SSL soit appliquée)

`SSLOptions +StrictRequire +StdEnvVars +ExportCertData`
» définition des options SSL

`<Directory />SSLRequireSSL</Directory>`
» accès strictement interdit si SSL n'est pas utilisé

`SSLVerifyClient require`
» indique qu'un certificat client est requis

`SSLVerifyDepth 2`
» indique le nombre de certificats à vérifier

`SSLCACertificateFile chemin/trusted-clients.pem`
» fichier au format PEM contenant les certificats CA qui permettent d'authentifier le certificat client

`SSLUseOCSP on`
» active la validation OCSP

`SSLForceValidation on`
» refuse le certificat si le répondeur OCSP n'est pas accessible

`SSLCertificateFile chemin/server.crt`
» fichier contenant le certificat serveur

`SSLCertificateKeyFile chemin/server.key`
» fichier contenant la clé privée correspondant au certificat serveur

Il reste enfin à paramétrer les fonctions de reverse proxy.

`RequestHeader set SSL\_CLIENT\_S\_DN "\%{SSL\_CLIENT\_S\_DN}e"`
» le premier nom de variable est le nom de la variable ajoutée aux variables d'environnement du serveur applicatif, le deuxième nom de variable est le nom d'une variable SSL du serveur apache. On peut ajouter à chaque fois une nouvelle ligne avec le nom de la variable SSL à transférer. Cette commande remplace tout header qui porterait le même nom.

L'installation est terminée, les certificats et clés sont créés. Il faut maintenant tester le serveur, et corriger quelques anomalies inévitablement rencontrées.

La période d'installation, de configuration et de tests commence. Ceux-ci sont effectués à partir d'un PC client, un portable, sur lequel est installé le middleware, fourni par le FEDICT, ainsi qu'un

lecteur de cartes eID (cardreader).

Ces tests dureront quelques jours et permettront de corriger quelques dysfonctionnements dus à la configuration du serveur Apache. Après quelques jours de travail intense, la connexion entre le PC client et le serveur se déroule normalement.

3.3.2 Elaboration d'une FAQ

Ces différentes étapes de tests et de configuration ont mis en évidence une série de questions pour lesquelles aucune réponse directe n'était disponible.

Il a donc paru opportun de créer une FAQ, recueil des questions et problèmes rencontrés durant la première phase du stage.
Dès qu'un problème est résolu, la solution trouvée est inscrite en réponse à la question. Si le problème ne trouve pas de solution, la question reste sans réponse.

Cette FAQ est censée être mise en commun avec tous les développeurs ou utilisateurs de programme basés sur l'eID. Elle pourra donc évoluer au fûr et à mesure des travaux réalisés autour de la carte eID. Elle pourrait trouver sa place sur le site de l'UCVW[6], par exemple.

Le contenu actuel de la FAQ est présenté en annexe (Annexe B). Vous trouverez, ci-dessous, l'un ou l'autre exemple des problèmes qui s'y trouvent.

Quelle est l'architecture générale d'une application eID en ligne ?
Client / PC-SC / Reverse Proxy / Serveur applicatif

J'ai le problème « cannot initialise, library already loaded ».
Il faut malheureusement redémarrer le navigateur
Pas d'autre solution connue pour le moment

Comment récupérer les entêtes eID en PHP ?
$entetes = getallheaders() ;
$entete1 = $entetes["SSL_ CLIENT_ S_ DN"] ;
$entete2 = $entetes["SSL_ CLIENT_ M_ SERIAL"] ;
$entete3 = $entetes["SSL_ CLIENT_ CERT"] ;
$entete4 = $entetes["SSL_ CLIENT_ VERIFY"] ;
Il ne reste ensuite qu'à les traiter correctement en php.
Note : l'en-tête « SSL_ CLIENT_ CERT » indique le résultat de l'authentification du client sur le serveur apache. Elle peut donc être utilisée comme un premier moyen de sécurité pour tester si le client passe bien par le reverse-proxy et vérifier que le statut de son authentification est bien « SUCCESS ».

La pertinence d'un tel travail a été telle qu'il a été décidé de poursuivre l'enrichissement de la FAQ pendant toute la durée du stage.

3.3.3 Réalisation d'une application eID

Après la mise au point et les tests de communication, il fallait trouver une application à réaliser pour utiliser le système. Utiliser un applet Java est une décision naturelle. En effet, le middleware contient un package Java permettant de réaliser assez facilement les accès à l'eID.

Le choix s'est porté sur un « chat » sécurisé. Cette application permet de montrer l'intérêt de l'utilisation de l'eID, dans diverses situations.

- L'application est sécurisée, par définition, puisqu'elle utilise tous les moyens de sécurisation de l'eID.
- Elle est assez simple à réaliser, du moins pour un utilisateur averti.
- Elle permet de mettre en évidence des qualités d'identification : même si l'interlocuteur utilise un pseudo, il est possible de connaître son identité, son âge, ou tout autre renseignement que le développeur aura inscrit dans le programme.
- Elle permet de contrôler les accès, ou de les réduire. On peut ainsi limiter l'accès aux personnes majeures, ou en fonction du sexe de l'utilisateur, ou en fonction de sa localisation géographique.

Exemple de code Java accédant aux données de la carte d'identité :

```
//Initialisation du lecteur de cartes
BEID_Long CardHandle = new BEID_Long();
BEID_Status oStatus = eidlib.BEID_Init("", 0, 0, CardHandle);

//Récupération des données d'identité
idData = new BEID_ID_Data();
BEID_Certif_Check CertCheck = new BEID_Certif_Check();
BEID_Status oStatus = eidlib.BEID_GetID(idData, CertCheck);

//Récupération des données de l'adresse
addrData = new BEID_Address();
BEID_Certif_Check CertCheck = new BEID_Certif_Check();
BEID_Status oStatus = eidlib.BEID_GetAddress(addrData, CertCheck);

//Récupération de la photo
BEID_Bytes Picture = new BEID_Bytes();
oStatus = eidlib.BEID_GetPicture(Picture, CertCheck);
FileOutputStream oFile = new FileOutputStream("photo.jpg");
oFile.write(Picture.getData());
```

L'absence d'un jeu de cartes test, reçues après la fin du stage, n'a pas permis de vérifier totalement les accès : certificats révoqués, cartes abîmées, ...

Néanmoins, à l'issue du stage, l'application fonctionne correctement et a d'ailleurs été présentée lors de la journée des entreprises, à laquelle le CETIC a participé. Trois machines équipées de lecteurs de cartes ont permis aux visiteurs de tester le chat sécurisé.

Un manuel utilisateur pour ce chat a été rédigé (Annexe C).

Note : pour permettre à l'applet d'accéder aux données de la carte d'identité (via le lecteur de cartes) et donc lui accorder certains privilèges, il est nécessaire de signer l'archive contenant l'applet (Annexe D).

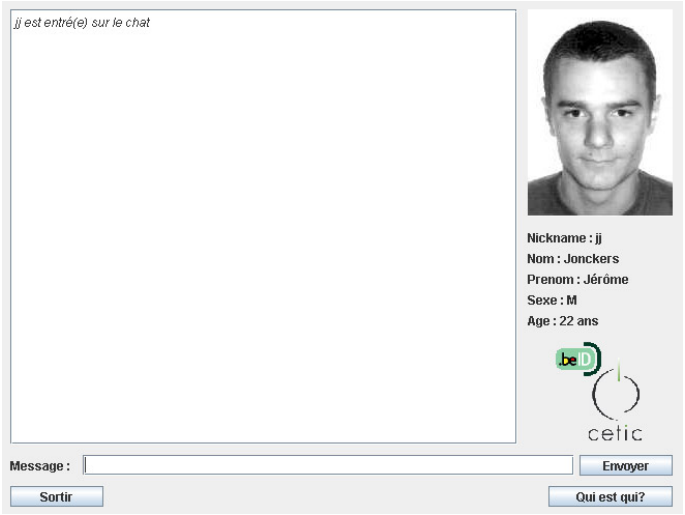


FIG. 6 – Ecran de chat

FIG. 7 – Ecran d'accès interdit

3.3.4 Journée découverte entreprise et 5 ans du CETIC

Profitant de son 5e anniversaire, le CETIC a ouvert ses portes durant un week-end, et l'application de chat eID a été présentée.
Plus de 300 visiteurs ont visité l'entreprise. Canal Z, télévision spécialisée, a diffusé un reportage sur celle-ci et ses activités, nottamment l'eID.

C'est en prévision de cette animation que diverses affichettes ont donc été réalisées afin de présenter le travail effectué.
Cela a occupé les dernières journées du stage.

Les divers documents produits sont présentés en annexe (Annexe E).

4 Conclusion

Dans le bain... ! Telle est l'impression première qui permet de caractériser ce stage. En parallèle avec la somme d'informations, de connaissances acquises au long des trois premières années d'université, la découverte d'un service informatique d'une entreprise permet de mesurer ses certitudes avec les surprises de la vie professionnelle.

Après quelques jours de tâtonnements, une inquiétude légitime quant au résultat à obtenir en un laps de temps finalement assez court, la mise au travail est somme toute simple et facile. Cela est certainement du à l'aide du maître de stage,... ainsi qu'à l'ambiance de l'entreprise.

Certes, il s'agissait d'une entreprise cherchant à mettre la recherche au service des services de production, et l'employé y bénéficie d'une relative liberté, mais les défis y sont nombreux, et l'apprentissage du métier ne se limite pas à reproduire les mêmes routines, à longueur de journée.

Appréhender les impératifs d'un travail de production, répondre positivement aux challenges proposés a été très constructif. La découpe du travail en petites entités, à chacune desquelles est fixée une durée déterminée, a permis le respect des timings et m'a ouvert les yeux sur des méthodes de travail auxquelles je ne pensais pas.

Ce stage a donc été particulièrement enrichissant, tant au niveau des connaissances qu'au niveau humain.

Mettre en pratique des éléments théoriques manipulés ici et là dans les séances d'exercices, dans les projets, dans une ambiance agréable, tout en sachant pouvoir bénéficier du soutien efficace d'un maître de stage rassure, et permet d'aborder la dernière année de formation avec une relative confiance en soi, dans ce cas précis en particulier.

5 Remerciements

Un stage réussi repose sur le concours de multiples intervenants. Je tiens ici à les remercier sincèrement.

Je me dois de remercier Monsieur Pierre Guisset, directeur du CETIC, pour avoir accepté de m'offrir un stage dans son entreprise.

Je voudrais particulièrement mettre en évidence l'accueil de Monsieur Christophe Ponsard, de la cellule « génie logiciel » du CETIC, qui, dès la première prise de contact, s'est montré intéressé par ma demande de stage. Il m'a guidé et conseillé tout au long de mon séjour dans l'entreprise. Je lui suis également redevable de mon intégration dans la société. Cela a facilité le stage et l'a rendu infiniment enrichissant. Je l'en remercie profondément.

Je suis très heureux d'avoir pu, pendant ce mois de septembre, partager le travail, l'ambiance des services du CETIC. J'en exprime ma gratitude à l'ensemble de son personnel.

Références

[1] Cay S. Hortsmann and Gary Cornell. *Au coeur de Java 2, notions fondamentales.* CampusPress, Paris, 2004. Traduit de l'américain par : Christiane Silhol et Nathalie Le Guillou de Penanros.

[2] Cay S. Hortsmann and Gary Cornell. *Au coeur de Java 2, fonctions avancées.* CampusPress, Paris, 2005. Traduit de l'américain par : Marie-Cécile Baland et Nathalie Le Guillou de Penanros.

[3] Danny De Cock, *"WebHome < Main < Godot's TWiki"*, sur http ://homes.esat.kuleuven.be/ decockd/wiki/bin/view.cgi/Main/WebHome

[4] Le CETIC, *"CETIC - Centre d'Excellence en Technologie de l'Information et de la Communication"*, sur http ://www.cetic.be

[5] Le Portail Fédéral, *"eID"*, sur http ://eid.belgium.be.

[6] Union des Villes et Communes de Wallonie, *"Articles E-communes : Dossier carte d'identité électronique (06-2006)"*, sur http ://www.uvcw.be/e-communes/eid.

A Fichier de configuration Apache

```
httpd.conf

ServerRoot "/cetic/home/externe/jj/install/apache2"

<IfModule !mpm_winnt.c>
<IfModule !mpm_netware.c>
#LockFile logs/accept.lock
</IfModule>
</IfModule>

<IfModule !mpm_netware.c>
<IfModule !perchild.c>
#ScoreBoardFile logs/apache_runtime_status
</IfModule>
</IfModule>


<IfModule !mpm_netware.c>
PidFile logs/httpd.pid
</IfModule>

Timeout 300

KeepAlive On

MaxKeepAliveRequests 100

KeepAliveTimeout 15


<IfModule prefork.c>
StartServers         5
MinSpareServers      5
MaxSpareServers     10
MaxClients         150
MaxRequestsPerChild  0
</IfModule>


<IfModule worker.c>
StartServers         2
MaxClients         150
MinSpareThreads     25
MaxSpareThreads     75
ThreadsPerChild     25
MaxRequestsPerChild  0
</IfModule>

<IfModule perchild.c>
NumServers           5
StartThreads         5
MinSpareThreads      5
MaxSpareThreads     10
MaxThreadsPerChild  20
MaxRequestsPerChild  0
</IfModule>

<IfModule mpm_winnt.c>
ThreadsPerChild 250
MaxRequestsPerChild  0
</IfModule>

<IfModule beos.c>
StartThreads               10
MaxClients                 50
MaxRequestsPerThread       10000
</IfModule>

<IfModule mpm_netware.c>
ThreadStackSize      65536
StartThreads           250
MinSpareThreads         25
MaxSpareThreads        250
MaxThreads            1000

                                    1
```

```
httpd.conf

MaxRequestsPerChild      0
MaxMemFree             100
</IfModule>

<IfModule mpmt_os2.c>
StartServers           2
MinSpareThreads        5
MaxSpareThreads       10
MaxRequestsPerChild    0
</IfModule>


Listen *:8080


#ExtendedStatus On

### Section 2: 'Main' server configuration

<IfModule !mpm_winnt.c>
<IfModule !mpm_netware.c>


User nobody
Group #-1
</IfModule>
</IfModule>


ServerAdmin you@example.com


ServerName pt-mda.cetic.be


UseCanonicalName off


DocumentRoot "/cetic/home/externe/jj/install/apache2/htdocs"


<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>


<Directory "/cetic/home/externe/jj/install/apache2/htdocs">

    Options Indexes FollowSymLinks

    AllowOverride None

    Order allow,deny
    Allow from all

</Directory>


UserDir public_html

#
# Control access to UserDir directories.  The following is an example
# for a site where these directories are restricted to read-only.
#
#<Directory /home/*/public_html>
#    AllowOverride FileInfo AuthConfig Limit Indexes
#    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
#    <Limit GET POST OPTIONS PROPFIND>
#        Order allow,deny
#        Allow from all

                                    2
```

httpd.conf

```
#    </Limit>
#    <LimitExcept GET POST OPTIONS PROPFIND>
#        Order deny,allow
#        Deny from all
#    </LimitExcept>
#</Directory>


DirectoryIndex index.html index.html.var

AccessFileName .htaccess

<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>

TypesConfig conf/mime.types

DefaultType text/plain

<IfModule mod_mime_magic.c>
    MIMEMagicFile conf/magic
</IfModule>

HostnameLookups Off

#EnableMMAP off

#EnableSendfile off

ErrorLog logs/error_log

LogLevel info

LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

#LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I
%O" combinedio

CustomLog logs/access_log common

#CustomLog logs/referer_log referer
#CustomLog logs/agent_log agent

#CustomLog logs/access_log combined

ServerTokens Full

ServerSignature On

Alias /icons/ "/cetic/home/externe/jj/install/apache2/icons/"

<Directory "/cetic/home/externe/jj/install/apache2/icons">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

AliasMatch ^/manual(?:/(?:de|en|es|fr|ja|ko|ru))?(/.*)?$
"/cetic/home/externe/jj/install/apache2/manual$1"

<Directory "/cetic/home/externe/jj/install/apache2/manual">
    Options Indexes
    AllowOverride None
    Order allow,deny
    Allow from all
```

httpd.conf

```
    <Files *.html>
        SetHandler type-map
    </Files>

    SetEnvIf Request_URI ^/manual/(de|en|es|fr|ja|ko|ru)/ prefer-language=
    $1
    RedirectMatch 301 ^/manual(?:/(de|en|es|fr|ja|ko|ru)){2,}(/.*)?$
    /manual/$1$2
</Directory>

ScriptAlias /cgi-bin/ "/cetic/home/externe/jj/install/apache2/cgi-bin/"

<IfModule mod_cgid.c>

#Scriptsock            logs/cgisock
</IfModule>

<Directory "/cetic/home/externe/jj/install/apache2/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>

IndexOptions FancyIndexing VersionSort

AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip

AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*

AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
AddIcon /icons/p.gif .pl .py
AddIcon /icons/f.gif .for
AddIcon /icons/dvi.gif .dvi
AddIcon /icons/uuencoded.gif .uu
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core

AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^

DefaultIcon /icons/unknown.gif

#AddDescription "GZIP compressed document" .gz
#AddDescription "tar archive" .tar
#AddDescription "GZIP compressed tar archive" .tgz

ReadmeName README.html
HeaderName HEADER.html

IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t

# DefaultLanguage nl
# Catalan (ca) - Croatian (hr) - Czech (cs) - Danish (da) - Dutch (nl)
# English (en) - Esperanto (eo) - Estonian (et) - French (fr) - German (de)
# Greek-Modern (el) - Hebrew (he) - Italian (it) - Japanese (ja)
# Korean (ko) - Luxembourgeois* (ltz) - Norwegian Nynorsk (nn)
```

4

```
httpd.conf

# Norwegian (no) - Polish (pl) - Portugese (pt)
# Brazilian Portuguese (pt-BR) - Russian (ru) - Swedish (sv)
# Simplified Chinese (zh-CN) - Spanish (es) - Traditional Chinese (zh-TW)

AddLanguage ca .ca
AddLanguage cs .cz .cs
AddLanguage da .dk
AddLanguage de .de
AddLanguage el .el
AddLanguage en .en
AddLanguage eo .eo
AddLanguage es .es
AddLanguage et .et
AddLanguage fr .fr
AddLanguage he .he
AddLanguage hr .hr
AddLanguage it .it
AddLanguage ja .ja
AddLanguage ko .ko
AddLanguage ltz .ltz
AddLanguage nl .nl
AddLanguage nn .nn
AddLanguage no .no
AddLanguage pl .po
AddLanguage pt .pt
AddLanguage pt-BR .pt-br
AddLanguage ru .ru
AddLanguage sv .sv
AddLanguage zh-CN .zh-cn
AddLanguage zh-TW .zh-tw

LanguagePriority en ca cs da de el eo es et fr he hr it ja ko ltz nl nn no
pl pt pt-BR ru sv zh-CN zh-TW

ForceLanguagePriority Prefer Fallback

AddCharset ISO-8859-1  .iso8859-1  .latin1
AddCharset ISO-8859-2  .iso8859-2  .latin2 .cen
AddCharset ISO-8859-3  .iso8859-3  .latin3
AddCharset ISO-8859-4  .iso8859-4  .latin4
AddCharset ISO-8859-5  .iso8859-5  .latin5 .cyr .iso-ru
AddCharset ISO-8859-6  .iso8859-6  .latin6 .arb
AddCharset ISO-8859-7  .iso8859-7  .latin7 .grk
AddCharset ISO-8859-8  .iso8859-8  .latin8 .heb
AddCharset ISO-8859-9  .iso8859-9  .latin9 .trk
AddCharset ISO-2022-JP .iso2022-jp .jis
AddCharset ISO-2022-KR .iso2022-kr .kis
AddCharset ISO-2022-CN .iso2022-cn .cis
AddCharset Big5        .Big5       .big5

AddCharset WINDOWS-1251 .cp-1251   .win-1251
AddCharset CP866       .cp866
AddCharset KOI8-r      .koi8-r .koi8-ru
AddCharset KOI8-ru     .koi8-uk .ua
AddCharset ISO-10646-UCS-2 .ucs2
AddCharset ISO-10646-UCS-4 .ucs4
AddCharset UTF-8       .utf8

AddCharset GB2312      .gb2312 .gb
AddCharset utf-7       .utf7
AddCharset utf-8       .utf8
AddCharset big5        .big5 .b5
AddCharset EUC-TW      .euc-tw
AddCharset EUC-JP      .euc-jp
AddCharset EUC-KR      .euc-kr
AddCharset shift_jis   .sjis

#AddEncoding x-compress .Z
#AddEncoding x-gzip .gz .tgz

AddType application/x-compress .Z
AddType application/x-gzip .gz .tgz

                                    5
```

httpd.conf

```
#AddHandler cgi-script .cgi

#AddHandler send-as-is asis

#AddHandler imap-file map

AddHandler type-map var

#AddType text/html .shtml
#AddOutputFilter INCLUDES .shtml

#
# Action lets you define media types that will execute a script whenever
# a matching file is called. This eliminates the need for repeated URL
# pathnames for oft-used CGI file processors.
# Format: Action media/type /cgi-script/location
# Format: Action handler-name /cgi-script/location
#

#
# Customizable error responses come in three flavors:
# 1) plain text 2) local redirects 3) external redirects
#
# Some examples:
#ErrorDocument 500 "The server made a boo boo."
#ErrorDocument 404 /missing.html
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"
#ErrorDocument 402 http://www.example.com/subscription_info.html
#

#
# Putting this all together, we can internationalize error responses.
#
# We use Alias to redirect any /error/HTTP_<error>.html.var response to
# our collection of by-error message multi-language collections.  We use
# includes to substitute the appropriate text.
#
# You can modify the messages' appearance without changing any of the
# default HTTP_<error>.html.var files by adding the line:
#
#   Alias /error/include/ "/your/include/path/"
#
# which allows you to create your own set of files by starting with the
# /cetic/home/externe/jj/install/apache2/error/include/ files and copying
them to /your/include/path/,
# even on a per-VirtualHost basis.  The default include files will display
# your Apache version number and your ServerAdmin email address regardless
# of the setting of ServerSignature.
#
# The internationalized error documents require mod_alias, mod_include
# and mod_negotiation.  To activate them, uncomment the following 30 lines.

#    Alias /error/ "/cetic/home/externe/jj/install/apache2/error/"
#
#    <Directory "/cetic/home/externe/jj/install/apache2/error">
#        AllowOverride None
#        Options IncludesNoExec
#        AddOutputFilter Includes html
#        AddHandler type-map var
#        Order allow,deny
#        Allow from all
#        LanguagePriority en cs de es fr it ja ko nl pl pt-br ro sv tr
#        ForceLanguagePriority Prefer Fallback
#    </Directory>
#
#    ErrorDocument 400 /error/HTTP_BAD_REQUEST.html.var
#    ErrorDocument 401 /error/HTTP_UNAUTHORIZED.html.var
#    ErrorDocument 403 /error/HTTP_FORBIDDEN#.html.var
    ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var
#    ErrorDocument 405 /error/HTTP_METHOD_NOT_ALLOWED.html.var
#    ErrorDocument 408 /error/HTTP_REQUEST_TIME_OUT.html.var
```

```
httpd.conf

#    ErrorDocument 410 /error/HTTP_GONE.html.var
#    ErrorDocument 411 /error/HTTP_LENGTH_REQUIRED.html.var
#    ErrorDocument 412 /error/HTTP_PRECONDITION_FAILED.html.var
#    ErrorDocument 413 /error/HTTP_REQUEST_ENTITY_TOO_LARGE.html.var
#    ErrorDocument 414 /error/HTTP_REQUEST_URI_TOO_LARGE.html.var
#    ErrorDocument 415 /error/HTTP_UNSUPPORTED_MEDIA_TYPE.html.var
#    ErrorDocument 500 /error/HTTP_INTERNAL_SERVER_ERROR.html.var
#    ErrorDocument 501 /error/HTTP_NOT_IMPLEMENTED.html.var
#    ErrorDocument 502 /error/HTTP_BAD_GATEWAY.html.var
#    ErrorDocument 503 /error/HTTP_SERVICE_UNAVAILABLE.html.var
#    ErrorDocument 506 /error/HTTP_VARIANT_ALSO_VARIES.html.var


BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0

BrowserMatch "Microsoft Data Access Internet Publishing Provider" redirect-
carefully
BrowserMatch "^WebDrive" redirect-carefully
BrowserMatch "^WebDAVFS/1.[012]" redirect-carefully
BrowserMatch "^gnome-vfs" redirect-carefully

#<Location /server-status>
#    SetHandler server-status
#    Order deny,allow
#    Deny from all
#    Allow from .example.com
#</Location>SSLProtocol -all +TLSv1 +SSLv3

#<Location /server-info>
#    SetHandler server-info
#    Order deny,allow
#    Deny from all
#    Allow from .example.com
#</Location>


SSLCertificateFile /cetic/home/externe/jj/install/apache2/certs/ca.crt
SSLCertificateKeyFile /cetic/home/externe/jj/install/apache2/certs/ca.key
SSLSessionCache shmcb:logs/ssl_cache_shm
SSLSessionCacheTimeout 600
SSLPassPhraseDialog builtin

<VirtualHost _default_:8080>
    SSLEngine on
    SSLOptions +StrictRequire +StdEnvVars +ExportCertData
    <Directory />
     SSLRequireSSL
    </Directory>
    SSLProtocol -all +TLSv1 +SSLv3
    SSLCipherSuite HIGH:MEDIUM:!aNULL:+SHA1:+MD5:+HIGH:+MEDIUM

    SSLVerifyClient require
    SSLVerifyDepth 2
    SSLCACertificateFile conf/ssl/trusted-clients.pem

    SSLUseOCSP on
    SSLForceValidation on
    SSLCertificateFile
    /cetic/home/externe/jj/install/apache2/certs/server.crt
    SSLCertificateKeyFile
    /cetic/home/externe/jj/install/apache2/certs/server.key
    ErrorLog logs/pt-mda-error_log
    CustomLog logs/pt-mda-access_log common
    LogLevel error

</VirtualHost>

<IfModule mime.c>

                                    7
```

httpd.conf

```
 AddType application/x-x509-ca-cert.crt
 AddType application/x-pkcs7-crl.crl
</IfModule>

<IfModule mod_proxy.c>
 ProxyRequests off
 ProxyPass / http://192.168.0.134/cetic-eid/
 ProxyPassReverse / http://192.168.0.134/cetic-eid/
</IfModule>


 RequestHeader set SSL_CLIENT_S_DN "%{SSL_CLIENT_S_DN}e"
 RequestHeader set SSL_CLIENT_M_SERIAL "%{SSL_CLIENT_M_SERIAL}e"
 RequestHeader set SSL_CLIENT_CERT "%{SSL_CLIENT_CERT}e"
 RequestHeader set SSL_CLIENT_VERIFY "%{SSL_CLIENT_VERIFY}e"

 SetEnvIf User-Agent ".*MSIE.*" \
 nokeepalive ssl-unclean-shutdown \
 downgrade-1.0 force-response-1.0
```

8

B FAQ

1 Généralité

– Qu'elle est l'architecture générale d'une application eID en ligne ?

Client / PC-SC / Reverse Proxy / Serveur applicatif

– A quel niveau se fait/font la/les vérification(s) de certificats ?

2 Reverse-Proxy

– Pourquoi ai-je besoin de recompiler apache ?

Parce que apache va mettre des en-têtes en plus et vérifier la validité OCSP

– Comment recompiler apache ?

– Comment configurer apache en HTTPS ? Doit-on passer par la configuration d'un virtualHost ?

– Qu'est ce que OCSP ? Comment l'activer et l'utiliser ?

OCSP signifie "Online Certificate Status Protocol". Ce protocole va vérifier la validité des certificats. Pour l'activer, il faut ajouter la ligne "SSLUseOCSP on" dans le fichier de configuration (httpd.conf) de Apache.

– Pourquoi tant d'étapes lors de la création de certificats ? Que fait chacune ?

3 Applicatif serveur

– Comment récupérer les entêtes eID en PHP ?

```
$entetes = getallheaders();
$entete1 = $entetes["SSL_CLIENT_S_DN"];
$entete2 = $entetes["SSL_CLIENT_M_SERIAL"];
$entete3 = $entetes["SSL_CLIENT_CERT"];
$entete4 = $entetes["SSL_CLIENT_VERIFY"];
```

Il ne reste ensuite qu'à les traiter correctement en php.
Note : l'en-tête « SSL_CLIENT_CERT » indique le résultat de l'au-

thentification du client sur le serveur apache. Elle peut donc être utilisée comme un premier moyen de sécurité pour tester si le client passe bien par le reverse-proxy et vérifier que le statut de son authentification est bien « SUCCESS ».

– Comment résoudre les problèmes d'accent ?

pending, UTF8
Exemple de problème : la chaîne « `J\xC3\xA9r\xC3\xB4me` » devrait s'afficher « Jérôme », elle reste pourtant elle-même, malgré un décodage utf8.

Possibilité de résolution :
Voici une fonction en php, qui convertit les représentations des caractères spéciaux posant problème en leur représentation utf8, pour un affichage correct :

```
/*Cette fonction prend une chaine "en-tête" envoyée par le reverse proxy,
et la retourne, convertie pour un affichage correct en php.
Argument : la chaîne "en-tête"
Retourne : la chaîne convertie
*/
function adapt_utf8($chaine){
$ascii = array(
"\\xC3\\x80","\\xC3\\x81","\\xC3\\x82","\\xC3\\x83","\\xC3\\x84","\\xC3\\x85",
"\\xC3\\x86","\\xC3\\x87","\\xC3\\x88","\\xC3\\x89","\\xC3\\x8A","\\xC3\\x8B",
"\\xC3\\x8C","\\xC3\\x8D","\\xC3\\x8E","\\xC3\\x8F","\\xC3\\x90","\\xC3\\x91",
"\\xC3\\x92","\\xC3\\x93","\\xC3\\x94","\\xC3\\x95","\\xC3\\x96","\\xC3\\x97",
"\\xC3\\x98","\\xC3\\x99","\\xC3\\x9A","\\xC3\\x9B","\\xC3\\x9C","\\xC3\\x9D",
"\\xC3\\x9E","\\xC3\\x9F","\\xC3\\xA0","\\xC3\\xA1","\\xC3\\xA2","\\xC3\\xA3",
"\\xC3\\xA4","\\xC3\\xA5","\\xC3\\xA6","\\xC3\\xA7","\\xC3\\xA8","\\xC3\\xA9",
"\\xC3\\xAA","\\xC3\\xAB","\\xC3\\xAC","\\xC3\\xAD","\\xC3\\xAE","\\xC3\\xAF",
"\\xC3\\xB0","\\xC3\\xB1","\\xC3\\xB2","\\xC3\\xB3","\\xC3\\xB4","\\xC3\\xB5",
"\\xC3\\xB6","\\xC3\\xB7","\\xC3\\xB8","\\xC3\\xB9","\\xC3\\xBA","\\xC3\\xBB",
"\\xC3\\xBC","\\xC3\\xBD","\\xC3\\xBE","\\xC3\\xBF");

$trad_ascii = array(
"\xC3\x80","\xC3\x81","\xC3\x82","\xC3\x83","\xC3\x84","\xC3\x85","\xC3\x86",
"\xC3\x87","\xC3\x88","\xC3\x89","\xC3\x8A","\xC3\x8B","\xC3\x8C","\xC3\x8D",
"\xC3\x8E","\xC3\x8F","\xC3\x90","\xC3\x91","\xC3\x92","\xC3\x93","\xC3\x94",
"\xC3\x95","\xC3\x96","\xC3\x97","\xC3\x98","\xC3\x99","\xC3\x9A","\xC3\x9B",
"\xC3\x9C","\xC3\x9D","\xC3\x9E","\xC3\x9F","\xC3\xA0","\xC3\xA1","\xC3\xA2",
"\xC3\xA3","\xC3\xA4","\xC3\xA5","\xC3\xA6","\xC3\xA7","\xC3\xA8","\xC3\xA9",
"\xC3\xAA","\xC3\xAB","\xC3\xAC","\xC3\xAD","\xC3\xAE","\xC3\xAF","\xC3\xB0",
"\xC3\xB1","\xC3\xB2","\xC3\xB3","\xC3\xB4","\xC3\xB5","\xC3\xB6","\xC3\xB7",
"\xC3\xB8","\xC3\xB9","\xC3\xBA","\xC3\xBB","\xC3\xBC","\xC3\xBD","\xC3\xBE",
```

```
"\xC3\xBF");

$chaine_retour = utf8_decode(str_replace($ascii,$trad_ascii,$chaine));
return $chaine_retour;
}
```

4 Applicatif client

– Comment lire les données sur la carte via une applet ?

L'applet doit, avant de commencer toute opération avec la carte, charger la librairie eidlib. Il faut ensuite initialiser la liaison avec le lecteur de cartes. Les données peuvent ensuite être récupérées dans 2 objets :

- le premier pour les données d'identité (objet de type BEID_ID_data)
- le second pour les données d'adresse (objet de type BEID_Address)

Les données se récupèrent ensuite via les méthodes définies pour ces objets (voir documents eID Belgium).

Exemple avec code :

1. Phase d'initialisation de l'applet :

```
idData = null;
addrData = null;

//chargement de la librairie eidlib
java.lang.System.loadLibrary("eidlibj");

//initialisation avec le lecteur de cartes
BEID_Long CardHandle = new BEID_Long();
BEID_Status oStatus = eidlib.BEID_Init("", 0, 0, CardHandle);
```

2. Démarrage de l'applet

```
//l'objet idData contiendra les données d'identité
idData = new BEID_ID_Data();
BEID_Certif_Check CertCheck = new BEID_Certif_Check();

//Obtention des données d'identité sur la carte
BEID_Status oStatus = eidlib.BEID_GetID(idData, CertCheck);

//l'objet addrData contiendra les données de l'adresse
addrData = new BEID_Address();
BEID_Certif_Check CertCheck = new BEID_Certif_Check();

//Obtention des données de l'adresse sur la carte
```

```
BEID_Status oStatus = eidlib.BEID_GetAddress(addrData, CertCheck);
```

3. Récupération de données d'identité ou d'adresse

Il suffit maintenant de récupérer les données souhaitées via les méthodes prévues (voir API) à ces objets, exemples :

```
//Récupération du nom de famille dans l'objet idData via la méthode getName()
String nom = idData.getName() ;

//Récupération de la rue dans l'objet addrData via la méthode getStreet()
String rue = addrData.getStreet() ;
```

– Puis-je récupérer la photo ? Comment ?

Oui, le principe est identique à la récupération des données d'identité ou d'adresse, mais on passe ici via un objet BEID_Bytes pour récupérer la photo et pouvoir la traiter ensuite comme un fichier ou dans un objet Image.
Exemple avec code :

```
//objet de type BEID_Bytes qui contiendra la photo
BEID_Bytes Picture = new BEID_Bytes();
BEID_Certif_Check CertCheck = new BEID_Certif_Check();
BEID_Status oStatus = new BEID_Status();

//Obtention des données de la photo sur la carte
oStatus = eidlib.BEID_GetPicture(Picture,CertCheck);

//création d'un fichier « photo.jpg » à partir des données de la photo récupérée.
FileOutputStream oFile = new FileOutputStream("photo.jpg");
oFile.write(Picture.getData());
oFile.close();
```

– Pourquoi dois-je signer mon applet ?

Accès à une ressource en dehors de la VM

– Comment lire sur la carte en javascript ?

Il faut tout d'abord charger une applet possédant les méthodes nécessaires à la lecture sur la carte eID (l'archive eidlib.jar du middelware possède déjà un applet de ce type : be.belgium.eid.BEID_Applet.class).
Une fois un nom attribué à l'applet (paramètre name dans la balise applet), les récupérations des données s'effectuent comme dans un applet Java, mais dans des balises Javascript.
Note : les méthodes définies pour l'applet du middleware sont disponibles dans le document eID Belgium.

4

Exemple avec code :

```
//balise applet à appeler dans votre page web
<applet
codebase = "emplacement_de_votre_archive_eidlig.jar"
archive  = "eidlib.jar"
code  = "be.belgium.eid.BEID_Applet.class"
name  = "BEIDtest"
width = "0"
height  = "0"
hspace  = "0"
vspace  = "0"
align = "middle">
<param name="Reader" value="">
<param name="OCSP" value="0">
<param name="CRL" value="0">
</applet>
```

Il ne reste ensuite qu'à utiliser les méthodes adéquates (voir API) dans des balises Javascript.
Exemple avec code :

```
<script language="javascript">
function GetData()
{
/*récupération du nom sur la carte grâce à la méthode
getName() et copie du résultat dans un champ
de formulaire de la page appelé « name »*/

document.getElementById('name').value = document.BEIDcetic.getName();

/*récupération du premier prénom sur la carte grâce
à la méthode getFirstName1() et  copie du résultat
dans un champ de formulaire de la page appelé « firstName »*/
document.getElementById('firstName').value = document.BEIDcetic.getFirstName1()
}
</script>
```

– Comment traiter les erreurs liées à la carte ? Existe-t-il une documentation des codes d'erreurs et leur signification ?

– J'ai le problème « cannot initialise, library already loaded »

Il faut malheureusement redémarrer le navigateur.
Pas d'autre solution connue pour le moment.

– Où dois-je héberger mon applet ?

– Comment gérer une connexion d'une applet vers un serveur d'application ?

Une solution est de gérer cela via des sockets et l'envoi de messages entre l'applet et un serveur.
Le serveur principal crée une instance d'un serveur particulier pour chaque client connecté. L'applet chargée sur le poste de l'utilisateur est l'applet Client.

– Lorsque l'applet Java utilise un objet certcheck (vérification de certificat), comportement différent selon le browser :
- Firefox : fenêtre vide de demande d'authentification supplémentaire - IE : fenêtre de demande d'authentification supplémentaire contenant les certificats présents dans le navigateur

5 Tests

– Où puis-je trouver un kit de test ?

www.eid-shop.be

– Autre truc de test :

Appletviewer local ne nécessite pas de signer l'applet pour le test

6 Divers

– Où puis-je trouver d'autres références ?

C Manuel utilisateur du chat

USER MANUAL

1 Lancement du serveur

- Pour lancer le serveur sans restriction d'accès, il suffit d'exécuter la commande classique java sur le fichier Serveur.class.

  > java Serveur

- Pour lancer le serveur avec restriction(s) d'accès, il faut lancer le serveur comme précédemment mais en lui passant certain(s) paramètre(s) :
  - Restriction sur la ville : `-ville NOMDELAVILLE` (où NOMDELAVILLE est le nom de la ville sur laquelle restreindre l'accès)
  - Restriction sur le sexe : `-sexe M ou F` (où M ou F est le sexe autorisé à accéder au CHAT)
  - Restriction sur l'âge minimum : `-agemin AGEMINIMUM` (où AGEMINIMUM est l'âge minimal (un entier) des personnes autorisées à accéder au chat.
  - Restriction sur l'âge maximum : `-agemax AGEMAXIMUM` (où AGEMAXIMUM est l'âge maximal (un entier) des personnes autorisées à accéder au chat.

  Un ou plusieurs de ces paramètres peuvent être ajoutés à la suite de la commande de lancement du serveur. Exemple : > java Serveur –ville Gosselies –agemin 12 –agemax 18 (l'accès est seulement autorisé pour les personnes habitant Gosselies, âgées entre 12 et 18 ans).

  Une fois le serveur lancé, avec ou sans restriction, la console indiquera dès qu'un client se connecte ou se déconnecte.

2 Lancement d'un client

- Le client est une applet. Différents paramètres doivent lui être passés :
  - ipServeur : l'adresse IP du serveur applicatif
  - demo : true ou false (indique si le chat est en mode démo ou non)
  - hauteur : la hauteur de l'applet (recommandé : 600 si démo, 550 sinon)
  - largeur : la largeur du client (recommandé : 800 si démo, 600 sinon)

  Exemple d'insertion de l'applet dans une page web :

```
<applet codebase="http://cetic-eid/chat-eid"
archive="Client.jar,eidlib.jar" code="chateid.client.Client.class"
width="800"  height="600">
        <param name="ipServeur" value="192.168.0.134">
        <param name="hauteur" value="600">
        <param name="largeur" value="800">
        <param name="demo" value="true">
        </applet>
```

3 Manipulations du client

- Ecran d'accueil

  Au départ, un écran de demande de nickname est introduit. Tant qu'aucun nickname ne sera entré, cet écran est maintenu :

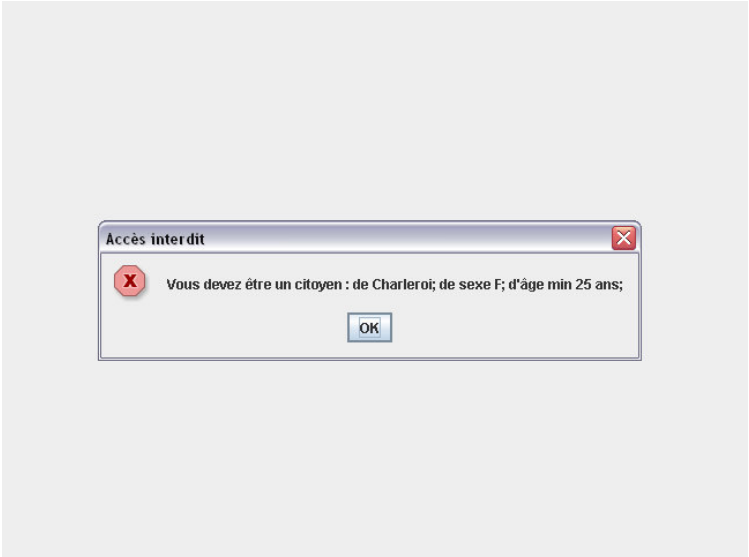
Demande de nickname en mode normal

Demande de nickname en mode démo

Une fois que le nickname choisi est introduit, et que soit le bouton « Entrer » est pressé, soit la touche « Enter » appuyée, le client tente de se connecter au serveur.

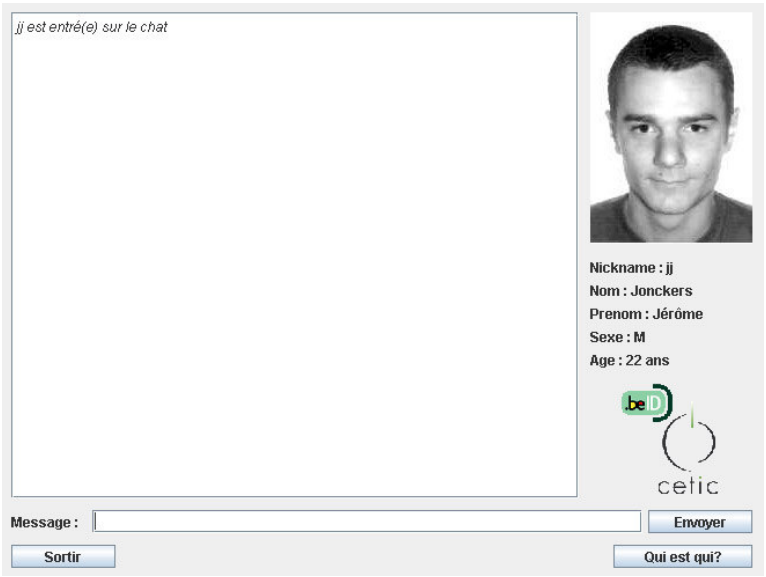
Introduction du nickname

Si la connexion est refusée, à cause d'une restriction, un message d'erreur avec le motif du refus apparaît et l'écran de demande de nickname est à nouveau affiché.



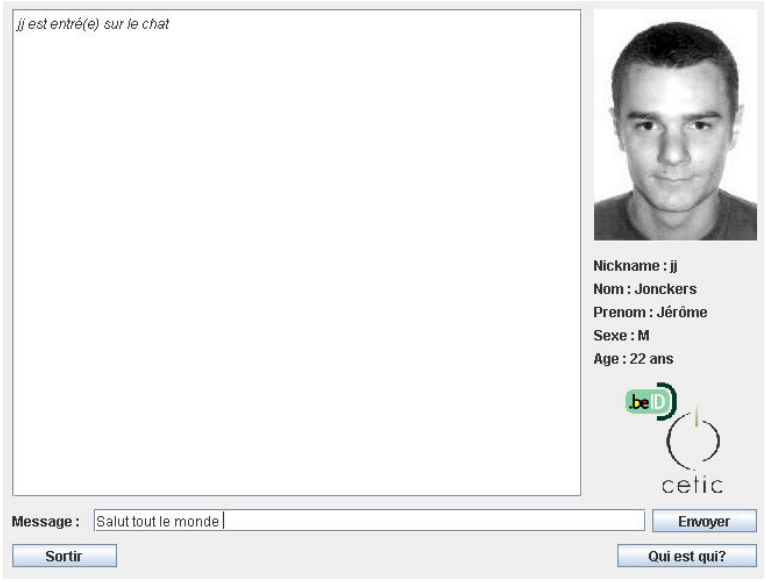
Accès interdit : le client ne répond pas aux critères de restriction Ville, Sexe et Age minimum

Si la connexion est acceptée, le client entre sur le chat. L'écran de chat, le champ de message, la photo, différents renseignements concernant le client apparaissent. Tous les clients connectés sur le chat reçoivent un message signalant que le nouveau client vient d'entrer.

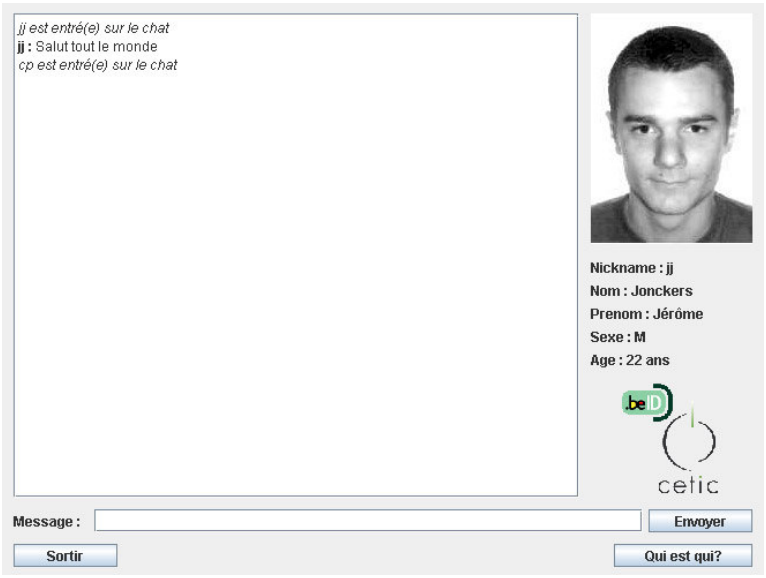


Client entrant sur le chat

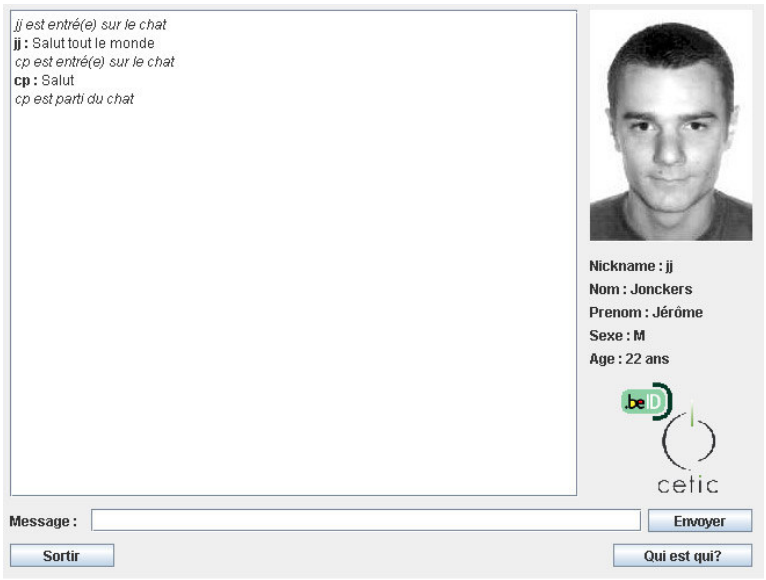
Le client peut donc maintenant composer ses messages et lire les messages reçus de la part des autres clients présents sur le chat.



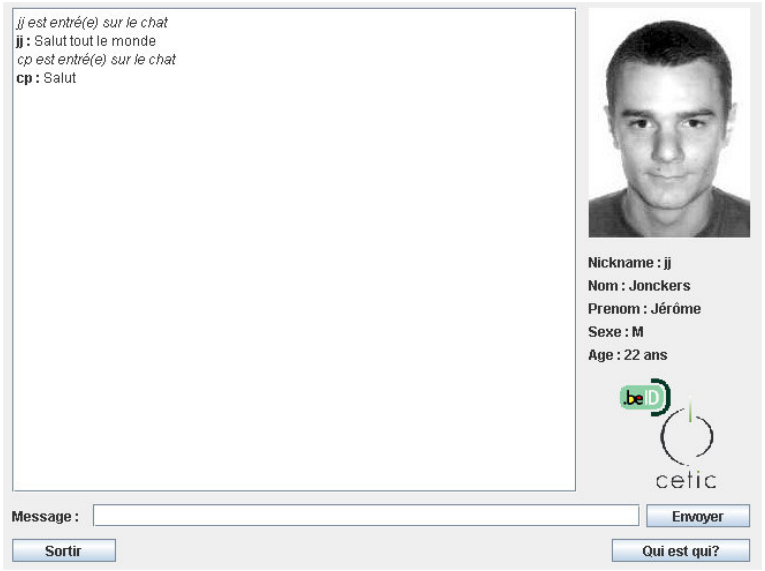
Client composant un message



Le client est averti de l'entrée d'un autre utilisateur sur le chat

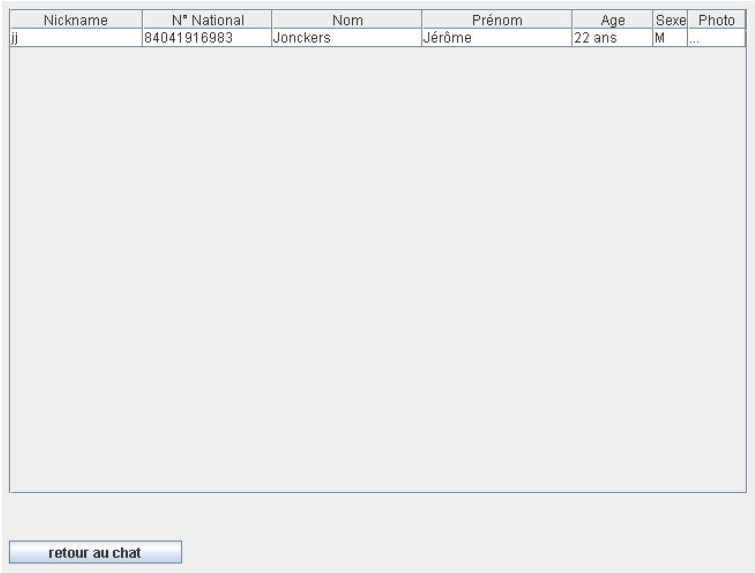


Le client est averti dès qu'un autre utilisateur quitte le chat



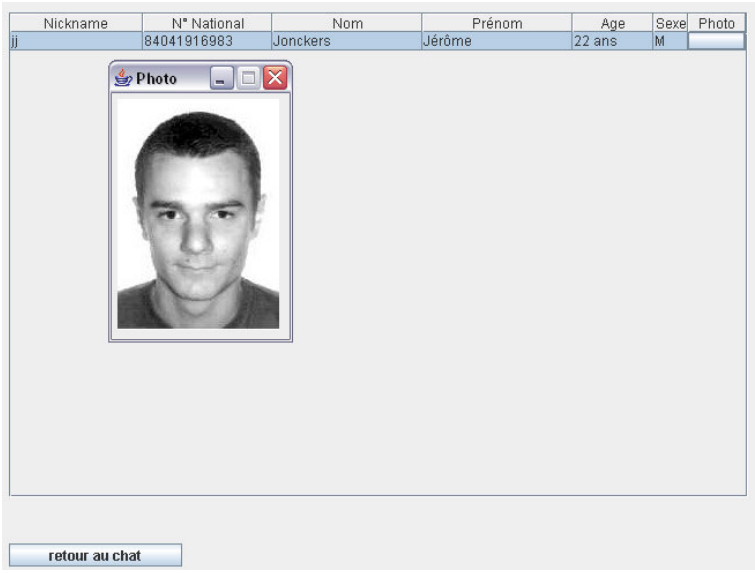
Le client lit les messages envoyés par d'autres utilisateurs

En cliquant sur le bouton « Qui est qui ? », un nouvel écran apparaît. Celui-ci est composé d'un tableau, reprenant l'ensemble des utilisateurs connectés au chat et quelques renseignements au sujet de chacun : nom et prénom véritables, âge, sexe, numéro national et photo.



Ecran « Qui est qui ? »

Si le client clique sur la case « Photo » d'un utilisateur présent dans la liste, une nouvelle fenêtre contenant la photo de cet utilisateur apparaît.



Demande d'une photo

Pour revenir à l'écran de chat, le client clique sur le bouton « retour au chat ».

Enfin, pour quitter le chat et se déconnecter, le client clique sur le bouton « sortir » de l'écran de conversation du chat.

D Signer une archive jar

DOC : SIGNER UNE ARCHIVE JAR

1 Créer une archive JAR

- Compiler le fichier java (Applet.java) :

  javac Applet.java

- Créer une archive JAR (Applet.jar) contenant les fichiers compilés souhaités (ici Applet.class) :

  jar cvf Applet.jar Applet.class

2 Signature de l'archive JAR

- Création de la clé et spécification de l'alias

  keytool –genkey –keystore cetic.store –alias cetic

  (cetic.store est le nom du magasin de certificats choisi, cetic est le nom de l'alias qui sera utilisé pour signer les archives)

  Sortie attendue :

  Tapez le mot de passe du Keystore : ceticeid

  Quels sont vos prénom et nom ?

  [Unknown] : Cetic user

  Quel est le nom de votre unité organisationnelle ?

  [Unknown] : eid.CETIC

  Quelle est le nom de votre organisation ?

  [Unknown] : CETIC

  Quel est le nom de votre ville de résidence ?

  [Unknown] : Gosselies

  Quel est le nom de votre état ou province ?

  [Unknown] : Belgium

  Quel est le code de pays ou deux lettres pour cette unité ?

```
 [Unknown]: BE

Est-ce CN=Cetic user, OU=eid.CETIC, O=CETIC, L=Gosselies, ST=Belgium,
C=BE ?

 [non]: oui

Spécifiez le mot de passe de la clé pour <cetic>

     (appuyez sur Entrée s'il s'agit du mot de passe du Keystore): ceticeid
```

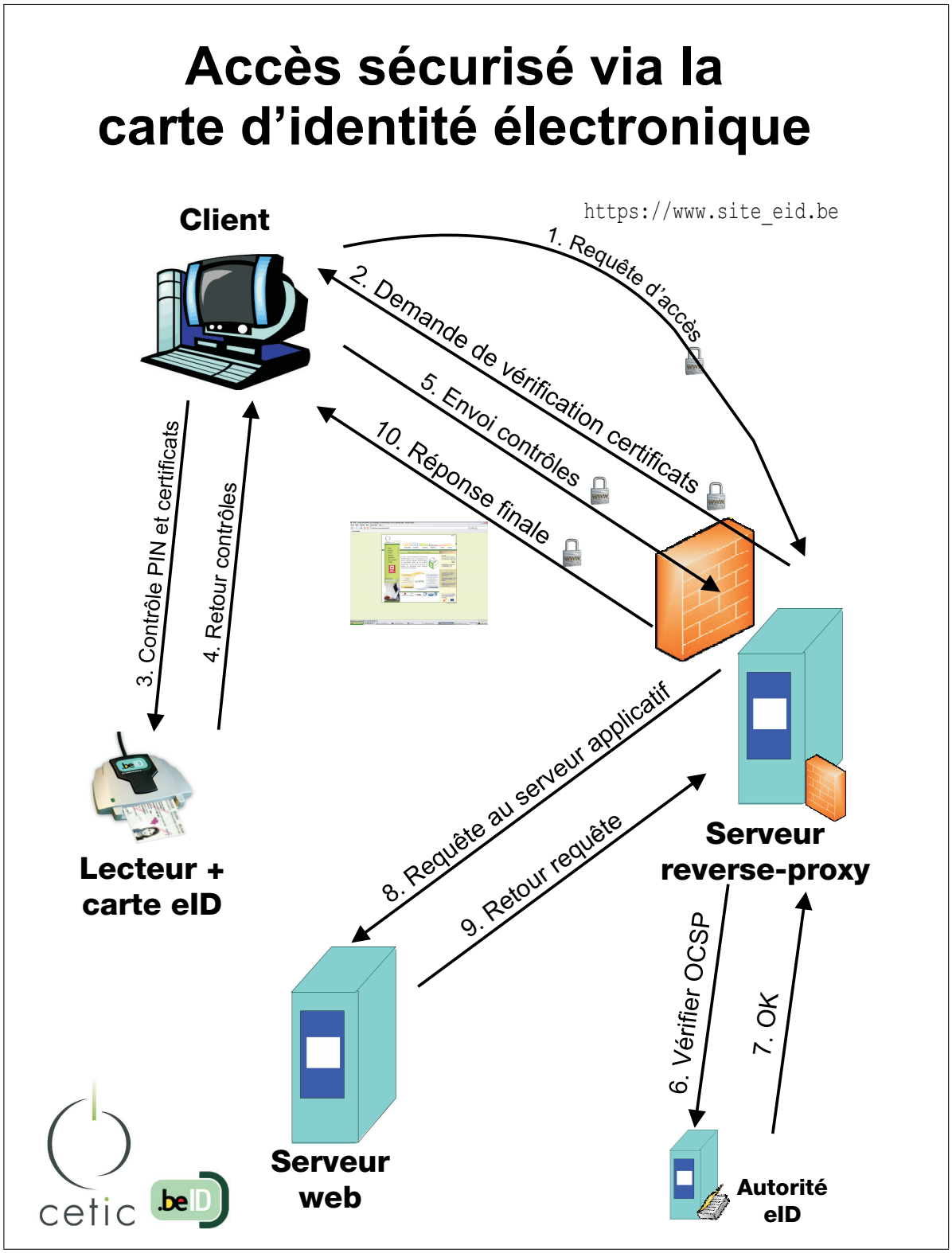
Après avoir introduit les données, il ne reste qu'à signer l'archive :

jarsigner –keystore cetic.store Applet.jar cetic

Le mot de passe choisi sera redemandé lors de la signature.

Dans cet exemple, le fichier JAR est signé à l'aide de la clé racine autosignée.

E Affiches de présentation eID



Applications utilisant la carte d'identité électronique

- Contrôle d'accès physique: parc à conteneurs, bibliothèque, police,…
- Authentification électronique: login sur une station de travail, SSH, SFT
- Signatures digitales: emails, fichiers PDF, …
- Application eGov:
  - accès web au registre national
  - déclarations : tax-on-web, TVA
  - commande de documents,
  - E-vote ?
- Chat sécurisé sur Internet
  - Kids
  - Seniors !
  - Chat « communal »
- eCommerce: eTicketing, transactions commerciales, …
- Applications e-Banking

La carte d'identité électronique

La puce

- Informations d'identité imprimées sur la carte
- Rue et numéro de résidence
- Code postal et localité
- Rang de la personne (prince, ...)
- Un statut spécial (aveugle, ...)
- Le type de documents (carte de citoyen belge, européen, ...)
- La photo
- Infos relatives à la puce ...

+

2 certificats personnels
- Authentification
- Signature

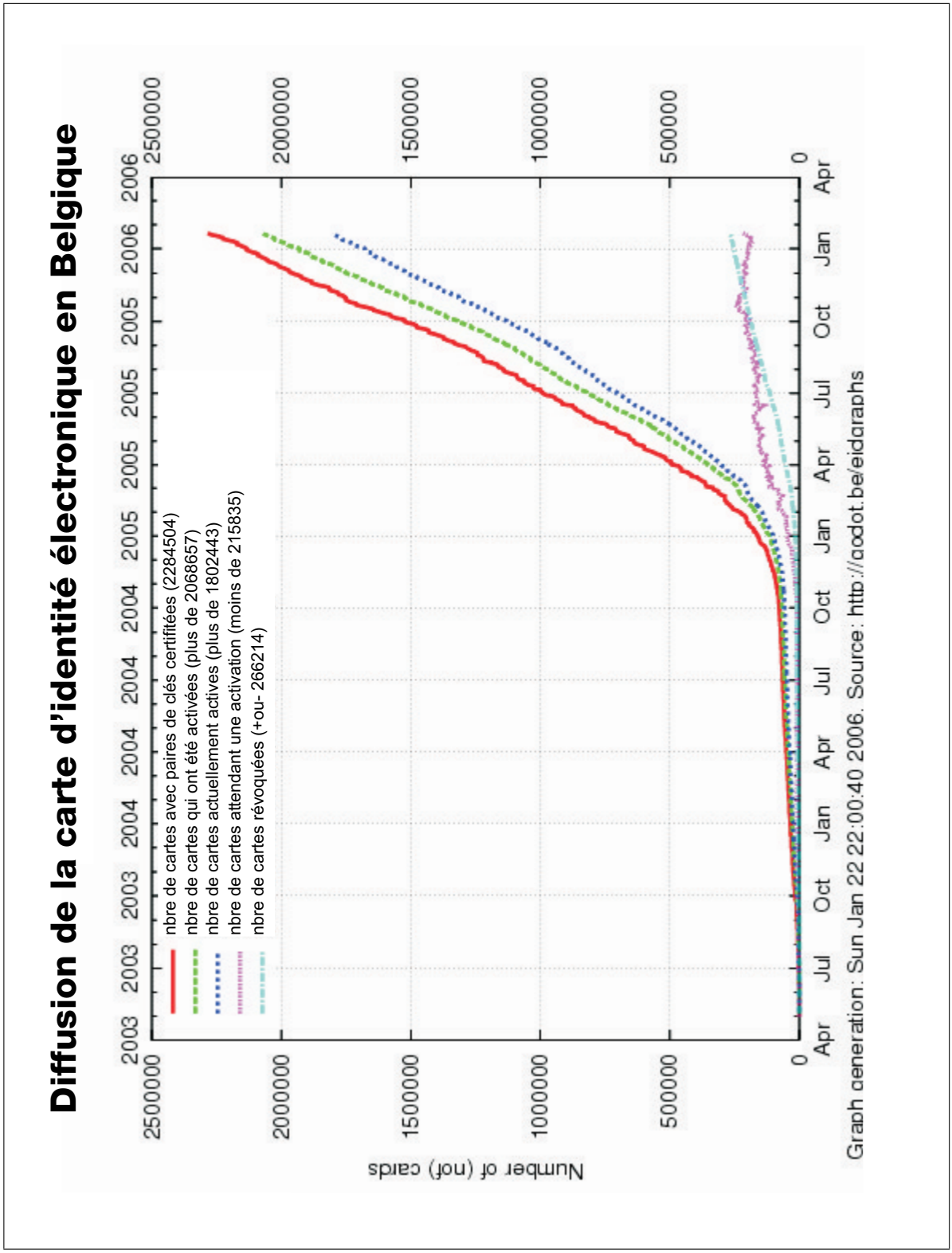
2 certificats d autorité
- Belgium CA
- Citizen CA

Au verso

N° national : 84.04.19-169.83

- 84.04.19 : date de naissance inversée
- 169 : si pair : F, si impair : H
- .83

.beID cetic


Diffusion de la carte d'identité électronique en Belgique
nbre de cartes avec paires de clés certifiées (2284504)
nbre de cartes qui ont été activées (plus de 2068657)
nbre de cartes actuellement actives (plus de 1802443)
nbre de cartes attendant une activation (moins de 215835)
nbre de cartes révoquées (+ou- 266214)
Number of (not) cards
2500000
2000000
1500000
1000000
500000
0
2003 2003 2003 2004 2004 2004 2004 2005 2005 2005 2005 2006 2006 2006 2006
Apr Jul Oct Jan Apr Jul Oct Jan Apr Jul Oct Jan Apr
Graph generation: Sun Jan 22 22:00:40 2006. Source: http://aodot.be/eidgraphs