OpenSC Unterstützung für Smartcards im kommerziellem Umfeld

Dr. Peter Koch

Irisstrasse 45
47249 Duisburg

0172 2470263

<Smartcard.PKoch@dfgh.net>

3./4. Mai 2006

OpenSC
die freie Smartcard Unterstützung

Dr. Peter Koch

Irisstrasse 45
47249 Duisburg

0172 2470263

<Smartcard.PKoch@dfgh.net>

3./4. Mai 2006

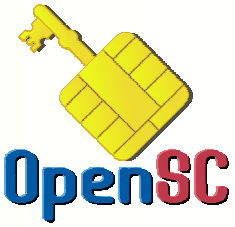
Inhalt

BuBK-Vortrag

- Vorstellung und Motivation
- Kurzvorführung
- Mini-Einführung Asymmetrische Kryptographie
- Smart Cards
  - Technische Aufbau und Funktionsweise
  - Einsatz bei digitaler Signatur und als 2-Faktor Authentifizierungs-Gegenstand
- Signaturgesetz
- Zeitstempel
- Vorstellung eines konkreten Projektes im kommerziellem Umfeld
  - Anforderungen
  - Lösung

Inhalt

freies Vortragsprogramm



- Vorstellung
- 2-Faktor Authentifizierungsverfahren
- Mini-Einführung Asymmetrische Kryptographie
- Smart Cards
  - Technische Aufbau und Funktionsweise
  - Einsatz bei digitaler Signatur und als 2-Faktor Authentifizierungs-Gegenstand
- OpenSC-Vorführung im Zusammenspiel mit anderen freien Projekten
  - OpenSSH, PuTTY
  - Samba
  - Mozilla, Firefox, Thunderbird
  - OpenSSL
  - CSP11

Projektanforderungen

- Ablösung der unsicheren Passwörter durch sicheres Authentifizierung-Verfahren
  - Bisherige Passwörter wurden schlecht gewählt und selten geändert.
  - Für unterschiedliche Anwendungen mit unterschiedlich sicheren Kommunikationswegen wurden gleiche Passwörter benutzt.
  - Extrembeispiel: Vornamen der Kinder wurde benutzt als Passwort für POP3-Abruf von Emails (Passwort-Übertragung im Klartext) und Zahlungsfreigabe von 8-stelligen EUR-Beträgen.

- Authentifizierung in kritischen Bereichen sollte nicht durch Administratoren umgangen werden können
  - In vielen (Passwort und Nicht-Passwort basierten) Systemen können die Administratoren sich beliebige Rechte zuweisen. Zum Teil sogar temporär und so, dass keine Spuren zurückbleiben.
  - Im Schadensfall gehörten Administratoren und EDV-Leiter zum Kreis der Personen, die technisch in der Lage waren, den Schaden zu verursachen.

- Vorbereitung auf elektronischen Heilberufeausweis
  - Die Ärzteschaft ist erster Personenkreis, der in Deutschland flächendeckend mit Signaturkarte ausgestattet wird.
  - Dies ermöglicht neue Service-Angebote (Direkt-Auskunft via Portal, Online-Adressänderungen, etc.)
    Die interne Nutzung von Smart Cards war hier als Vorbereitung auf die baldige Nutzung von Signaturkarten im Aussenverhältnis gedacht.

Vorstellung des OpenSC-Projektes

- Internationales Entwicklerteam
  - Genaue Liste aller Beteiligten auf www.opensc-project.org
  - In Deutschland vor allem: Andreas Jellinghaus und Nils Larsch
- Relativ unbekannt im Vergleich zu Verbreitung
  - OpenSC wird von mehreren Ländern und zum Teil unter anderem Namen als offizielle Software für die nationalen Identitätskarten eingesetzt.
    Z.B.: Belgien, Estland, Spanien
- www.opensc-project.org
  - Enthält alle Informationen zum Projekt
  - Aktuelle Snapshots aus dem Development
  - Mailing-Listen für Anwender und Entwickler

Authentifizierungsverfahren

- Passwort-Authentifizierung versus 2-Faktor Methoden
  - Idee:
    Authentifizierung durch Kenntnis eines Passwortes wird ersetzt durch Authentifizierung durch Besitz eines Gegenstandes + Kenntnis eines Passwortes
  - Vorteile:
    Gegenstand kann sich nur im Besitz einer Person befinden
    Passwort schützt im Fall des Verlustes oder bei Diebstahl des Gegenstandes

- Anforderungen an Gegenstand
  - Gegenstand ist einmalig.
  - Gegenstand ist nicht duplizierbar, auch nicht durch Systemadministratoren.
  - Besitz des Gegenstandes lässt sich von einem entfernten Standort über eine unsichere Leitung gegenüber einem Server beweisen.

- Mögliche Gegenstände
  - SecurID-Token
  - Biometrische Merkmale (Fingerabdruck, Augenhintergrund, Gesichtszüge)
  - Smart Card

SecurID-Token

- Beim SecurID-Token wird der Beweis des Besitzes dadurch geführt, dass die angezeigte 6-stellige Zahl an den Server übertragen wird

  Hierbei wird davon ausgegangen, dass:

  - nur diejenige Person die momentan angezeigt Nummer kennen kann, die sich gerade im Besitz des Tokens befindet.
  - Die oft geäußerte Behauptung, dass die Vorhersage der angezeigten Nummern nicht möglich ist, ist allerdings leider falsch.
  - Die Kenntnis der Identifikationsnummer des Tokens und des mathematischen Verfahrens, das in den Token implementiert wurde, reicht aus um die angezeigte Nummer für einen beliebigen Zeitpunkt vorherzuberechnen.
  - z.B.: Token am Schlüsselbund von Dr. Koch:
    3.5.2006 17:10-17:15 Uhr: **626040,491547,965539,748138,300678,020529**
    4.5.2006 15:10-15:15 Uhr: **840078,086617,418715,603086,489459,925685**

Biometrische Merkmale

- Der Beweis des Besitzes eines Körperteils kann gegenüber einem Computer offensichtlich nicht auf direktem Wege geführt werden, sondern nur indirekt dadurch, dass mit dem Körperteil computerauswertbare Muster erzeugt werden.
  - Fingerabdruck: Kapazitative Muster auf Fingerabdruckleser.
  - Augenhintergrund oder Gesichtszüge: Muster innerhalb Videobild.
- Problem 1:
  - Muster lassen sich auch anderweitig erzeugen:
  - Im Falle von Fingerabdrucklesern z.B. durch Tesafilm oder Gummibärchen auf dem Leser oder bei seriellen Lesern durch Abspielen zuvor aufgezeichneter Bytefolgen an der seriellen Schnittstelle.
  - Im Falle von Augenhintergrund oder bei Gesichtszügen durch Fotographien vor der Videokamera oder durch Abspielen von Videosignalen mit einem Videorecorder.
- Problem 2:
  - Mustererkennung erfolgt meist lokal, daher muss der Server dem Ergebnis der vom Client durchgeführten Mustererkennung vertrauen. Falls die lokale Software so verändert wird, dass sie sich auch ohne korrektes Muster gegenüber dem Server so verhält, als wäre ein korrektes Muster erkannt worden, so hat der Server keine Möglichkeit, das festzustellen.

Verschlüsselungsverfahren

- Symmetrische Verschlüsselung
  - Ein Klartext wird mittels eines sogenannten Schlüssels so in einen unleserlichen Buchstabensalat (Chiffrat genannt) verwandelt, dass er nur mit Kenntnis des Schlüssel wieder in den Ursprungstext zurückverwandelt werden kann.

- Asymmetrische Verschlüsselung
  - Zur asymmetrischen Verschlüsselung wird ein Schlüssel-Paar benutzt. Die beiden Schlüssel müssen gemeinsam erzeugt werden und stehen über ein mathematisches Verfahren miteinander in Beziehung. Es ist aber (in vertretbarer Zeit) nicht möglich aus einem der beiden Schlüssel den anderen zu berechnen.
  - Ein Klartext, der mittels eines der beiden Schlüssel verschlüsselt wird, kann nur mit Hilfe des anderen Schlüssels wieder entschlüsselt werden.
  - Asymmetrische Schlüssel werden pro Person benutzt. Dabei kennt jeweils nur der Besitzer des Schlüssel-Paares einen der beiden Schlüssel (den sogenannten privaten Schlüssel) und alle anderen Kommunikationspartner den anderen Schlüssel (den sogenannten öffentlichen Schlüssel).

Asymmetrische Verschlüsselung

- Einsatzmöglichkeit 1: Verschlüsselung
  - Wird eine Nachricht mit dem öffentlichen Schlüssel verschlüsselt, so kann sie nur mit dem dazugehörigen privaten Schlüssel entschlüsselt werden.
  - Im Gegensatz zur symmetrischen Verschlüsselung muss der öffentliche Schlüssel nicht auf sicherem Wege im Vorfeld ausgetauscht werden, sondern darf allgemein bekannt sein.

- Einsatzmöglichkeit 2: Digitale Unterschrift
  - Wird eine Nachricht mit dem privaten Schlüssel des Absenders verschlüsselt, so kann jeder Adressat diese Nachricht wieder entschlüsseln, da ja der öffentlicher Schlüssel allgemein bekannt ist. Eine so durchgeführte Verschlüsselung hat also überhaupt keinen Geheimhaltungseffekt, allerdings folgende zwei Nebeneffekte:
  - Wenn ein Adressat die verschlüsselte Nachricht mit einem öffentlichen Schlüssel entschlüsseln kann, ist sichergestellt, dass die Nachricht seit der Verschlüsselung nicht verändert wurde.
  - Außerdem ist sichergestellt, dass die Nachricht von derjenigen Person stammt, die den passenden privaten Schlüssel kennt, weil nur damit eine Verschlüsselung möglich war.
  - Unveränderbarkeit + Herkunft = digitale Unterschrift
  - Damit eine digitale Unterschrift nicht genauso groß ist wie das Ursprungsdokument, wird nicht das Dokument selber, sondern nur eine Prüfsumme des Dokumentes unterschrieben.

Technischer Aufbau einer Smart Card

- Technischer Aufbau
  - Eine Smart Card enthält einen kleinen Computer mit CPU und ist in der Lage asymmetrische Verschlüsselungen durchzuführen.
  - In diesem Computer ist der private und öffentliche Schlüssel mindestens eines asymmetrischen Schlüssel-Paars gespeichert. Der öffentliche Schlüssel kann ausgelesen werden. Der private Schlüssel dagegen kann nicht ausgelesen werden und kann nur INNERHALB der Karte benutzt werden. Und auch das nur, wenn vor der Benutzung die korrekte PIN der Karte eingegeben wurde.
  - Jede Smart Card enthält einen anderen privaten Schlüssel.
  - Ein Schlüsselpaar kann außerhalb einer Karte erzeugt und dann auf einer Smart Card abspeichern. Dann würde nicht nur die Karte den privaten Schlüssel kennen, sondern auch der Hersteller der Karte. In Deutschland überprüft der Gesetzgeber, dass zertifizierte Anbieter von Smart Cards nach der Programmierung der Karte den privaten Schlüssel nicht speichern.
  - Nachrichten, die mit dem öffentlichen Schlüssel einer Smart Card verschlüsselt wurden, können nur mit genau dieser Smart Card entschlüsselt werden.
  - Eine Nachricht, die sich mit dem öffentlichen Schlüssel einer Smart Card entschlüsseln lässt, kann nur vom Besitzer genau dieser Smart Card erzeugt worden sein.

  - Damit kann eine Smart Card zur sicheren Verschlüsselung und zur Erzeugung einer digitalen Unterschrift verwendet werden.

Authentifizierung mit RSA-Schlüsseln

- Asymmetrische Schlüssel ermöglichen folgendes Authentifizierungsverfahren
  - Will ein Benutzer sich am Server anmelden, so verschlüsselt der Server eine Zufallszahl mit dem öffentlichen Schlüssel des Benutzers und sendet die verschlüsselte Zahl an den Client.
  - Ist der Benutzer in der Lage die ursprüngliche Zufallszahl zurückzuschicken, muss er im Besitz des passenden privaten Schlüssels sein.
  - Durch dieses Verfahren kann ein Benutzer den Besitz einer Information (hier den Besitz des privaten Schlüssels) über eine unsichere Leitung beweisen, ohne dass die Information selber weder im Vorfeld noch bei der eigentlichen Anmeldung ausgetauscht werden muss.
  - Wird ein Schlüsselpaar benutzt, dessen privater Schlüssel sich innerhalb einer Smart Card befindet, so beweist die Tatsache, das der Benutzer im Besitz des privaten Schlüssels ist gleichzeitig, dass er im Besitz einer Smart Card ist, in der sich dieser private Schlüssel befindet.
  - Werden Signaturgesetz konforme Smart Cards eingesetzt, so ist sichergestellt, dass es nur genau eine Smart Card gibt, die den privaten Schlüssel enthält. Und es wurde bei Ausgabe dieser Smart Card vom Staat überprüft, dass die korrekte Person die Smart Card erhalten hat.

Digitale Signaturen und Zertifikate

- Funktionsweise einer digitalen Signatur
  - Eine digitale Signatur (= digitale Unterschrift) entsteht dadurch, dass die Prüfsumme einer Datei zusammen mit der Urzeit und dem Namen des Kartenbesitzers vom privaten Schlüssel einer Smart Card verschlüsselt wird.
  - Bei der Überprüfung einer digitalen Unterschrift werden die so verschlüsselten Daten mit dem öffentlichen Schlüssel der Smart Card entschlüsselt. Ist dies möglich, so hat man den Beweis, dass die digitale Unterschrift mit genau der Smart Card erzeugt wurde, dessen öffentlichen Schlüssel man für den Entschlüsselungsvorgang benutzt hat.
  - Da der Überprüfer einer digitalen Signatur im Normalfall keinen Zugriff auf die Smart Card hat, kann er nicht überprüfen, ob der öffentliche Schlüssel, mit dem er die Signatur überprüft, auch wirklich von der angegebenen Person stammt.
  - Deshalb wird zur Überprüfung einer Signatur von Person P nicht nur der öffentliche Schlüssel von Person P benötigt, sondern zusätzlich auch noch ein Beweis, dass der zu Überprüfungszwecken mitgelieferte öffentliche Schlüssel wirklich von Person P stammt.
  - Dieser Beweis wird dadurch geliefert, dass Person P sich von einer vertrauenswürdigen Stelle bestätigen lässt, dass er der Besitzer der Smart Card mit dem passenden privaten Schlüssel ist. Die vertrauenswürdige Stelle (das sogenannte Trust-Center) erstellt daraufhin eine kleine Datei, die im wesentlichen den Namen von Person P und dessen öffentlichen Schlüssel enthält, und unterschreibt diese Datei elektronisch. Eine derart durch ein Trust-Center signierte Datei wird Zertifikat genannt.
  - Das Signaturgesetz schreibt die Bedingungen vor, die ein Trust-Center erfüllen muss, damit deutsche Gerichte dieser Stelle vertrauen und Signaturen wie Papierunterschriften behandeln.

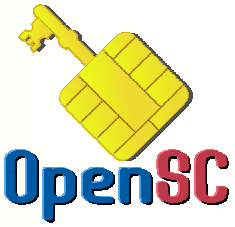
Das Signaturgesetz

- Anforderungen des deutschen Signaturgesetzes
  - Zugelassen sind nur asymmetrische Schlüssel mit ausreichender Schlüssellänge (≥ 1024 Bit)
  - Die Schlüssel müssen auf Smart Cards gespeichert werden.
  - Das Aufspielen der Schlüssel auf eine Smart Card muss in einer sicheren Umgebung erfolgen.
  - Beim Erstellen einer Signatur muss sichergestellt sein, dass der Inhalt der zu signierenden Datei dem Unterzeichnenden korrekt angezeigt wird.
    Dazu fordert das Signaturgesetz den Einsatz entsprechend geeigneter Softwarekomponenten.
  - Eventuell ist eine digitale Unterschrift, die mit nicht zertifizierten Komponenten erzeugt wurde, juristisch ähnlich wertlos wie eine Unterschrift, die beim Notar geleistet wurde, ohne dass der Notar vorher den Vertrag vorgelesen hat.
    Hierzu gibt es bisher keine eindeutigen juristischen Aussagen.
  - Welche Software-Komponenten gesetzes-konform arbeiten wird derzeit ausschließlich vom Bundesamt für Sicherheit in der Informationstechnik (BSI) geprüft (bzw. von Firmen, die vom BSI beauftragt werden, solche Prüfungen vorzunehmen).

Zeitstempel

- Bedeutung der Uhrzeit in einer digitalen Signatur
  - Eine digitale Signatur enthält neben der Prüfsumme der zu signierenden Datei und dem Namen des Unterschreibers auch die Uhrzeit, zu dem die Signatur erzeugt wurde.
  - Für den späteren Nachweis, dass die Datei nicht verändert wurde, ist genau diese Zeitangabe von Bedeutung.
  - Da die Uhrzeit in einer digitalen Signatur im Normalfall vom Computer stammt, auf dem die Signatur erzeugt wurde, kann durch Rückstellen dieser Uhrzeit leicht eine digitale Signatur mit rückdatiertem Datum erstellt werden.
  - Aus diesem Grund gibt es staatlich anerkannte Dienstleister, die Signaturen anbieten, die garantiert die korrekte Zeit enthalten. Solche Zeitstempel werden auch anonyme Unterschriften genannt, weil in diesem Fall nur die Zeitangabe innerhalb der Signatur von Bedeutung ist und nicht der enthaltene Name.
  - Zeitstempelanbieter bestätigen mit ihrer Signatur auch nicht den Inhalt einer Datei, sondern lediglich, dass zum Zeitpunkt der Signaturerstellung eine bestimmte Prüfsumme dem Dienstleister zur Signatur vorgelegt wurde.
  - Hat das signierte Dokument dann zu einem späteren Zeitpunkt immer noch genau die Prüfsumme, die vom Zeitstempel-Dienstleister signiert wurde, so kann man sicher sein, dass dieses Dokument sich immer noch im gleichen Zustand befindet wie zum Zeitpunkt der Zeitstempel-Erzeugung.
  - Ein Zeitstempelanbieter kann somit die Unverändertheit eines Dokumentes bestätigen ohne das Dokument jemals gesehen zu haben.

Projektanforderungen



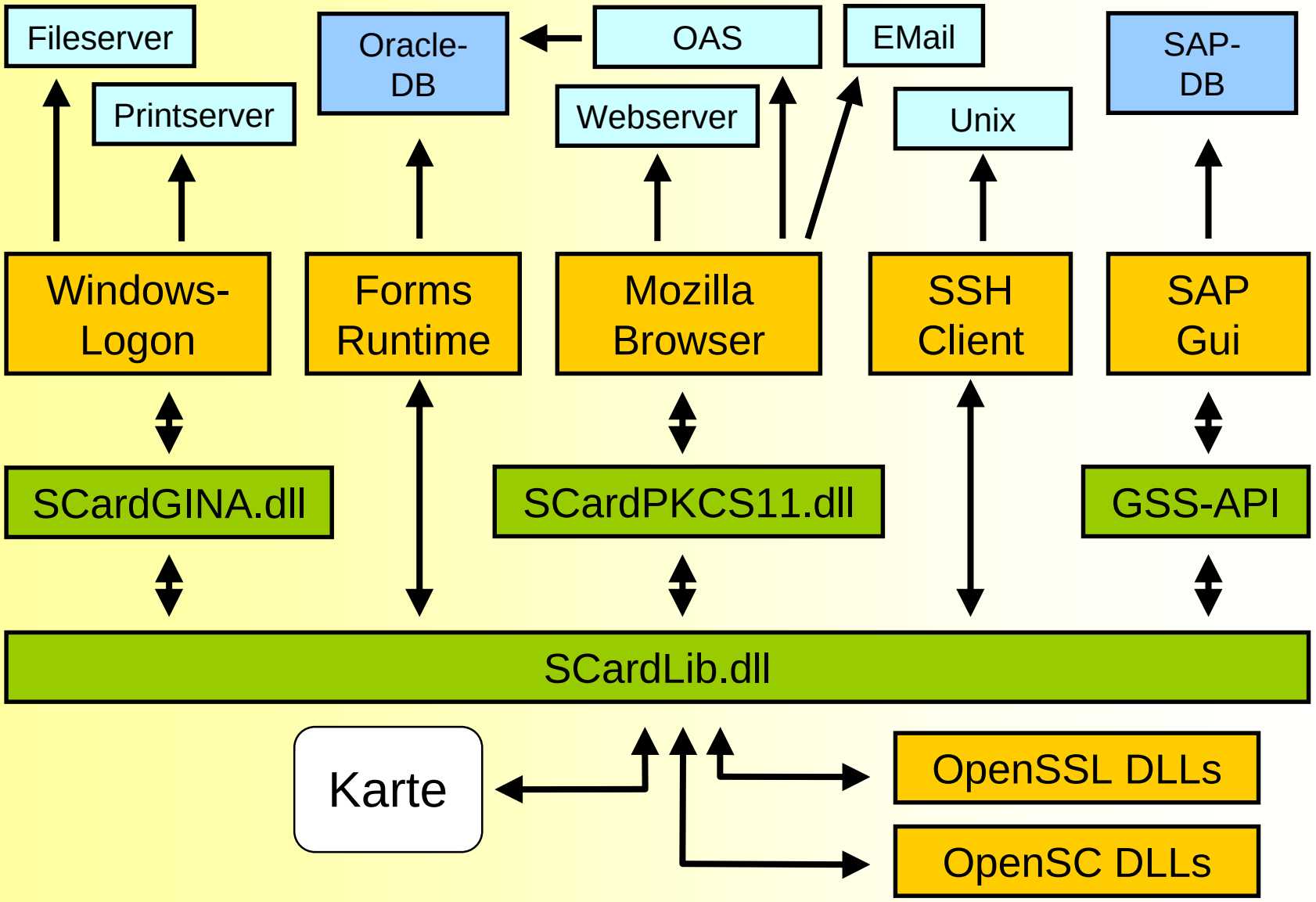
- Zertifikatsbasierte Authentifizierung
- Speicherung der geheimen Schlüsseln auf Signaturgesetz-konformen Smartkarten
- Keine Reproduktionsmöglichkeit der 2-Faktor-Gegenstände, auch nicht durch Adminstratoren und auch nicht durch EDV-Leiter
- Kein Single-Sign-On, stattdessen einmalige Freischaltung mit PIN und dann Überprüfung des Gegenstandes bei jeder Anmeldung
- Windows-Logon an Samba-PDC (optional Windows-ADS)
- Oracle-Anmeldung (Client-Server und OAS / Forms im Web)
- Secure-Shell Anmeldung an Unix-Rechnern
- SAP-Anmeldung
- Signieren von PDF-Dateien
- Benutzung der Smart Card für Zahlungsfreigaben
- PKI-Software für Zertifikatsverwaltung
- praktikable Regelungen im Verlustfall / bei Diebstahl der Smartcard
- sichere aber trotzdem praktikable Regelungen für die PIN-Vergabe (u.a. PIN-Verteilung per PIN-Brief)

Smart Card Lösung

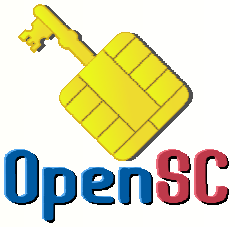
- TCOS NetKey E4-Karten mit drei Schlüsselpaaren, Signaturgesetz-konform, zusätzlicher Sender für berührungslose Zutrittskontrolle, Zeiterfassung und Bezahlfunktion
- Verwaltung der Smartkarten, der Zertifikate und der öffentlichen Schlüssel in Oracle-Tabelle.
- Verteilung der PINs mittels PIN-Brief (Ersatz-Briefe für vergessene PINs)

- Austausch der Windows-Anmelde DLL (MS-GINA.dll) gegen Smartcard-basierte Version (SCardGINA.dll).
- Anmelde-Server unter Unix zur Anbindung von Samba-Server.
- PKCS#11-DLL zur Anbindung von Mozilla (SSL-Client-Authentifizierung)
- Einsatz einer modifizierten Version des Freeware SSH-Clients PuTTY
- SAP-Gui Anbindung via GSS-API
- Oracle-Anmeldung mittels Smart Card durch Einbindung von OpenSSL und OpenSC-DLLs in die Oracle Forms Runtime-Umgebung
- POP3 Email-Abruf mit Smart Card Authentifizierung

Smart Card Lösung



OpenSC und SSH



- Eigenschaften des SSH-Protokolls
  - SSH-Verbindungen erfolgen verschlüsselt, Welches symmetrische Verschlüsselungsverfahren benutzt wird, kann vom Anwender ausgewählt werden.
  - Zur Authentifizierung stehen mehrere Verfahren zur Auswahl, unter anderem auch die Authentifizierung via asymmetrischen RSA-Schlüsseln.
  - Hierzu werden auf dem Server pro User die öffentlichen Schlüssel der berechtigten Personen hinterlegt (in der Datei .authorized_keys). Eine Anmeldung ist nur dann möglich, wenn der Client den Besitz eines passenden privaten Schlüssels nachweist.
  - Ob die privaten Schlüssel in einer Datei (Software-Key) oder auf einer Smart Card gespeichert werden ist ausschliesslich Angelegenheit des SSH-Clients.
  - Der in den Unix-Versionen von OpenSSH enthaltene SSH-Client unterstützt OpenSC und damit die Anmeldung mittels Smart Card.
  - Eine der besten SSH-Client Implementationen unter Windows ist PuTTY. PuTTY selber unterstützt OpenSC nicht und OpenSC Support ist auch nicht geplant.
    Auf dem OpenSC-Server findet man aber angepasste aktuelle PuTTY-Versionen.

- Beispiel
  - Anmeldung an produktiver Firewall mit Smart Card
  - Auslesen des öffentlichen Schlüssels im SSH-Format mittels:

```
# pkcs15-tool --read-ssh-key
ssh-rsa AAAAB3NzaC1yc2EAAAAEQAAAgQAAAIEAgZGpLOzNKiNDFB8qmAN7lAfj5
orNkJVBKZKDe5HUTspmhnA6hHikKykpXilwHZuql26DMffpakRvxNMbNBgGqTnrAO
hnj+o1NDpUnkCSFFvfy8U2u4MRSdhISfyiTZDex6DEvvs4Dqd7Tm3ptsxs/xqQKfv
+fmFgDSt1Jukk9ic= LinuxTag Demo
```

OpenSC und Samba

- Samba unterstützt die Smart Card basierte Anmeldung nicht
- GINA Bibliothek unter Windows
  - Die Anmeldung unter Windows erfolgt mittels einer sogenannten GINA-Bibliothek.
    Die mitgelieferte MS-GINA ermöglicht die Passwort-basierte Anmeldung an beliebigen Domänen (Samba und/oder MS).
    Die Smart Card basierte Anmeldung wird von der MS-GINA nur an Windows-NT Servern unterstützt.
  - Es existiert allerdings auch eine GINA, die eine Smart Card basierte Anmeldung an einer Samba Domäne ermöglicht.
    Allerdings enthält sie kommerzielle Teile und ist daher nicht frei verfügbar.
  - Für eine solche Anmeldung ist ein Zertifikat erforderlich, das im Distinguished Name des Besitzers ein x500UniqueIdentifier Attribut enthält. Letzteres gibt an, unter welchem Namen die Anmeldung an der Domäne erfolgen soll.
- Beispiel:
  - Auslesen eines Zertifikates aus einer Smart Card und Anzeige des Inhaltes mittels OpenSSL

```
# pkcs15-tool -r 45 | openssl x509 -noout -subject -nameopt multiline
subject=
    commonName                = LinuxTag 2006 Demo
    x500UniqueIdentifier      = koch
    emailAddress              = Demo@LinuxTag.org
    organizationalUnitName    = Test-Zertifikate
    organizationName          = LinuxTag
    localityName              = Wiesbaden
    countryName               = DE
```

OpenSC und Mozilla

- Mozilla unterstützt PKCS#11
  - Damit unterstützt Mozilla (und Firefox, etc.) alle kryptographischen Geräte, für die eine PKCS#11-Bibliothek existiert.
    Smart Cards sind kryptographische Geräte und OpenSC enthält sowohl eine PKCS#11 Bibliothek für Unix-Betriebssysteme als auch für Windows.

- Was ist mit der PKCS#11-Unterstützung von Mozilla möglich:
  - Client-Authentifizierung an allen Servern, mit denen via SSL kommuniziert wird, insbesondere Apache Webserver, WebDAV, Subversion Web-Frontend, etc.
  - Email Abruf via SSL-gesichertem POP3-Protokoll ohne zusätzliches Passwort (hierfür ist ein POP3-Server notwendig, der sowohl das SSL-Protokoll als auch Authentifizierung via Client-Zertifikat unterstützt)

- Beispiel
  - Kryptographie-Module und Zertifikate von Firefox anzeigen

OpenSC und OpenSSL

- OpenSC enthält eine OpenSSL-Engine
  - OpenSSL ist eine Software, mit der kryptographische Operationen durchgeführt werden können.
  - OpenSSL arbeitet im wesentlichen mit Software-Schlüsseln, kann aber auch bestimmte kryptographische Operationen durch eine externe Bibliothek durchführen lassen. Eine solche externe Bibliothek nennt sich OpenSSL-Engine.
  - OpenSC enthält eine OpenSSL-Engine, mit der alle kryptographischen Operationen, für die ein privater Schlüssel notwendig ist, durch diese Engine und damit letztendlich durch eine Smart Card durchgeführt werden können.
  - Dies hat zur Konseuenz, dass alle Operationen, die mit asymmetrischen Software-Schlüsseln von OpenSSL durchgeführt werden können, auch mit einer Smart Card möglich sind.
  - Prominente Beispiele hierfür sind das Signieren und Entschlüsseln von Nachrichten.

- Beispiel:
  - Signieren eines Zertifkates (in Datei cert.pem) mittels CA-Zertifikat (in Datei ca.pem) und privatem Software-Schlüssel (in Datei key.pem)
    ```
    openssl asdf qwer asdf qwer asdf
    ```
  - Nun das gleiche mit einem Schlüssel auf einer Smart Card
    ```
    openssl asdf qwer asdf qwer asdf
    ```

OpenSC und OpenSSL

- Beispiel:
  - Signieren eines Zertifkates (in Datei cert.pem) mittels CA-Zertifikat (in Datei ca.pem) und privatem Software-Schlüssel (in Datei key.pem)

```
openssl x509 \
        -in request.pem -req \
        -CA crt_ca.pem -CAkey key_ca.pem \
        -set_serial 1 -days 100
```

  - Nun das gleiche mit einem Schlüssel auf einer Smart Card

```
pkcs15-tool -r 45 >karte.crt
openssl x509 -engine pkcs11 \
        -in request.pem -req \
        -CA karte.crt -CAkey 2:45 -CAkeyform engine \
        -set_serial 1 -days 100
```

OpenSC und MS-Windows

- Windows benutzt CSP
  - Microsoft benutzt für den Zugriff auf kryptographische Geräte nicht den plattform-unabhängigen PKCS#11 Standard, sondern hat für diesen Zweck eine eigene Vorgehensweise entwickelt, den sogenannten Cryptographic Service Provider.
  - OpenSC enthält keinen CSP.
  - Es gibt aber ein weiteres freies Projekt (CSP11), das CSP-Aufrufe in PKCS#11-Aufrufe übersetzt.
  - Damit kann die PKCS#11-Bibliothek von OpenSC als CSP genutzt werden.
  - Prominente Anwendungen, die statt des PKCS#11-Standards einen CSP benutzen und die auf diesem Wege auch mit OpenSC funktionieren, sind der Internet Explorer und Outlook.
  - Prinzipiell sollte auf diesem Wege auch ein Smart Card Login mittels MS-GINA an einer Windows NT Domäne möglich sein.
    Bisher hat sich aber noch niemand gemeldet, der das ausprobiert hätte.