Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 812 19 Bratislava

Infraštruktúra PKI a vybavenie pre koncových používateľov

Tímový Projekt

Tím číslo 4 PSS:
Bc. Peter Mulinka
Bc. Radomír Škrib
Bc. Jozef Hamar
Bc. Tomáš Smolek

Rok: 2004 / 2005

Vedúci projektu: Doc. Ing. Ladislav Hudec, Csc.

Obsah

1 HandySign

1.1 Úvod

Podpisovanie dokumentov je bežne riešené pomocou nejakého externého zariadenia, z ktorého nie je možné zistit privátny kľúč. Takéto zariadenie čaká na dáta, ktoré má podpísať. Ak ich dostane, podpíše ich a vráti elektronický podpis. Tento postup by sa dal simulovať tak, že externé zariadenie by bol mobilný telefón podporujúci javu a k počítaču by sa pripájal pomocou kábla, IRDA alebo bluetooth.

1.2 Analýza

Zariadenie, ktoré bude držať privátne kľúče a ktoré bude podpisovať údaje musí byť nejakým spôsobom pripojiteľné k počítaču. V dnešnej dobe sa mobilné telefóny dokážu pripojiť k PC štyrma spôsobmi:

- Sériový kábel
- IRDA
- Bluetooth
- Internet

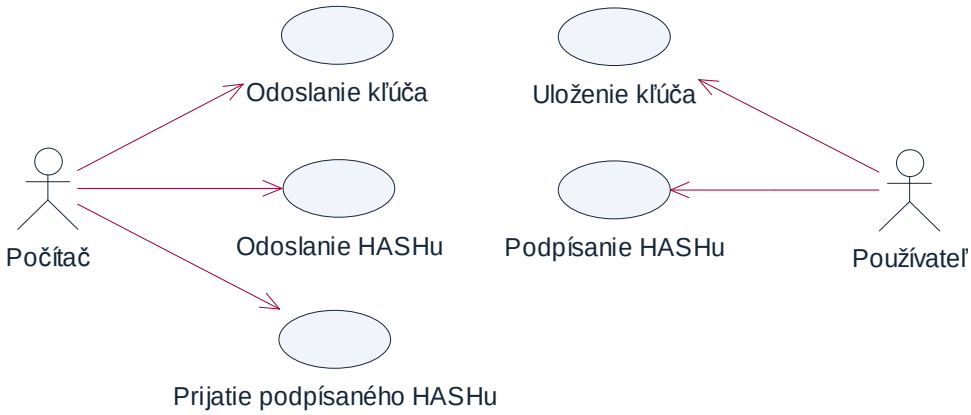
Pre naše spojenie (PC – mobilný telefón) by bola najlepšia architektúra peer-to-peer. Je najlepšia čo sa týka bezpečnosti a aj najjednoduchšia. Bezpečnosť by sa ešte zvýšila keby k danému spojeniu nemal nikto iný prístup ako počítač a samotný mobilný telefón. Týmto pádom odpadajú možnosti Internet a Bluetooth. V Internete číha na podpis a privátny kľúč príliš veľa nebezpečenstva, už len samotný komunikačný protokol by bol veľkou témou pre tímový projekt. Bluetooth je server-client architektúra. Síce môže byť kryptovaný, ale rovnako ako všetky WiFi technológie NIE JE bezpečný.

Sériový kábel a IRDA sú ideálne riešenia spojenia. Naraz môžu byť spojené iba dve zariadenia. Nikto nemôže odpočúvať komunikáciu medzi nimi. Sú jednoducho implementovateľné a väčšina mobilných telefónov ich má. Pri IRDA by sa dalo ešte namietať, že je to predsa len svetlo, ktoré žiari aj za mobilný telefón, avšak reálny komunikačný dosah IRDA je 1m. Oproti sériovému káblu má IRDA výhodu v tom, že je štandardizované a teda

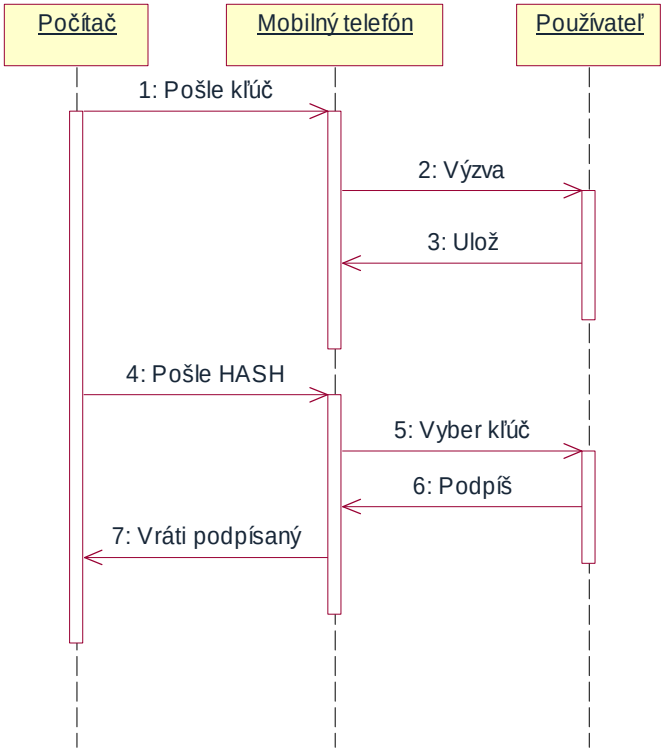
všetky mobilné telefóny a IRDA prijímače na PC komunikujú rovnakým protokolom. Z tohto dôvodu naša architektúra použije na spojenie IRDA port.

1.2.1 Prípady použitia

Aplikácia pre mobilný telefón by mala byť jednoduchá, ľahko použiteľná. Mala by ponúkať akési skladisko privátnych kľúčov a podpisovanie pomocou týchto kľúčov. Prípady použitia pre takúto aplikáciu sú nižšie (Obrázok 1.2-1).



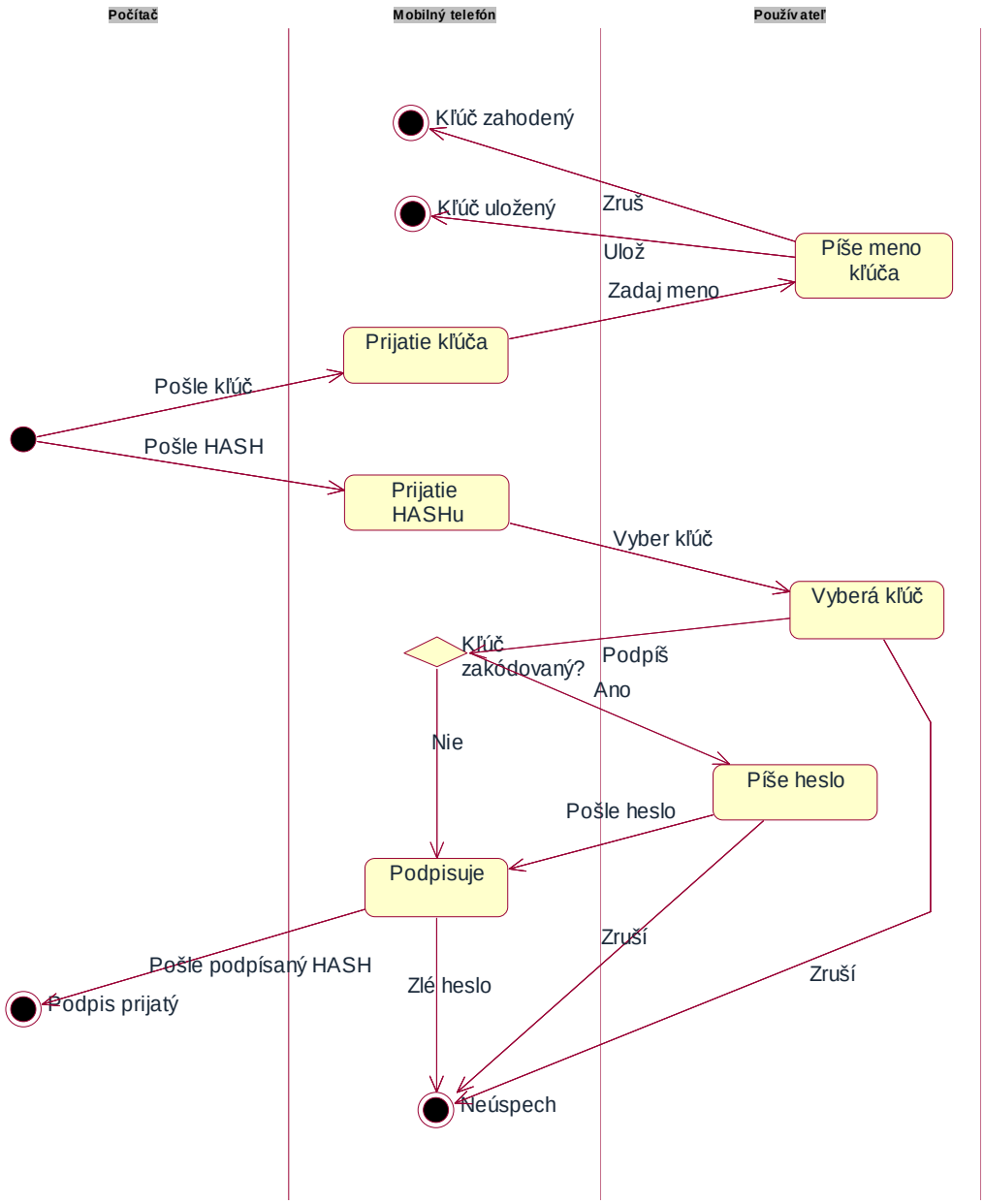
Obrázok 1.2-1 - Prípady použitia HandySign



Obrázok 1.2-2 - Sekvencia interakcie

S aplikáciou interagujú dvaja hráči – počítač a používateľ. Medzi počítačom a programom je obojsmerná komunikácia. Počítač musí vedieť odoslať programu kľúč aj HASH a mal by očakávať návrat podpísaného HASHu. Aplikácia by mala pri každej udalosti vytvorenej počítačom komunikovať s používateľom. Interakcia s používateľom na základe udalostí spôsobených počítačom je vyššie (Obrázok 1.2-2).

1.2.2 Aktivity programu



Obrázok 1.2-3 – Diagram aktivít

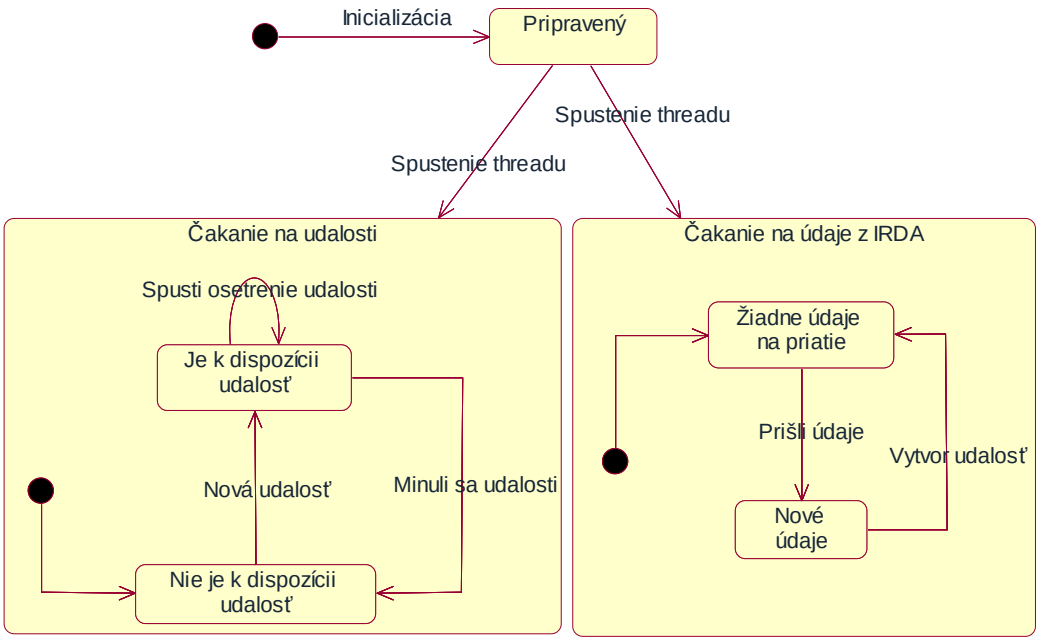
Vyššie na obrázku (Obrázok 1.2-3) je zobrazený stavový diagram chovania aplikácie HandySign. Všetko sa začína keď počítač vyšle privátny kľúč alebo HASH cez IRDA na mobilný telefón.

Telefón po prijatí kľúča vyzve používateľa aby zadal meno pod ktorým ho uloží do úložiska kľúčov. Používateľ má možnosť ho zamietnuť alebo potvrdiť nové meno.

Ak počítač vyšle HASH, telefón vyzve používateľa aby vybral kľúč na podpísanie. Podobne ako v predchádzajúcom prípade má používateľ možnosť odmietnuť podpísanie, alebo vybrať vhodný kľúč z úložiska. Ak je kľúč zašifrovaný, HandySign vyzve používateľa aby zadal heslo. Zase má používateľ na výber možnosť stornovania podpisu. Ak sa programu podarí dešifrovať kľúč pomocou hesla, podpíše HASH a odošle ho spať cez IRDA na počítač. Ak sa nepodarilo dešifrovať kľúč, znamená to, že heslo nebolo zadané správne. V takomto prípade sa stornuje podpis. Opakovaný pokus o zadávanie hesla a podpísanie HASHu nieje povolené, možné je ho však dosiahnuť novým odoslaním HASHu. Z hľadiska bezpečnosti je toto lepšie riešenie, lebo znemožnuje brute force technikám zistiť heslo.

1.3 Návrh

1.3.1 Životný cyklus programu



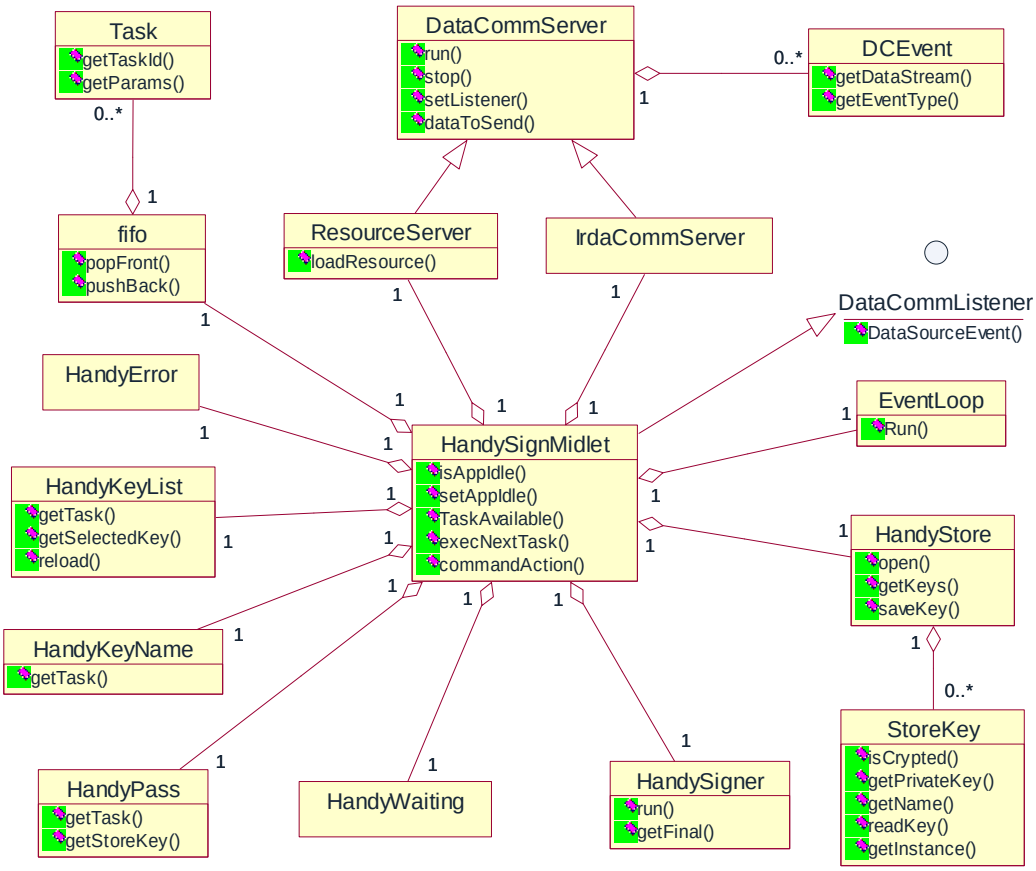
Obrázok 1.3-1 - Vnútorné vlákna

Program sa má správať na základe veľmi jednoduchých pravidiel, po spustení má očakávať príkazy od používateľa a údaje prichádzajúce na IRDA port telefónu. Ak program práve vykonáva obsluhu udalosti, nesmie začať vykonávať inú, avšak údaje môže v pozadí aj naďalej prijímať. Takéto riešenie si vyžaduje najmenej dve vlákna. Jedno čo by obsluhovalo

prijímanie a odosielanie dát z IRDA portu a druhé, ktoré by sa staralo o ošetrenie udalostí. Udalosť môže byť príkaz od používateľa ale aj príchod nových dát z portu. Súbeh týchto dvoch vlákien zobrazuje Obrázok 1.3-1.

1.3.2 Triedy

Pre užívateľské prostredie má java pre mobilné telefóny (J2ME) dobre vyvinuté triedy. Tieto použijeme ako predkov pre triedy HandyError, HandySigner, HandyWaiting, HandyPass, HandyKeyName a HandyKeyList. Triedy dediace po DataCommServer sú určené pre prijímanie a odosielanie dát. Na obrázku nižšie je zobrazený kompletný diagram tried a vzťahov medzi nimi. Ich významy sú objasnené v nasledujúcom texte.



Obrázok 1.3-2 - Class diagram

HandySignMidlet

Je to hlavná trieda, ktorá zastrešuje všetky operácie. Drží si refrencie na inštancie všetkých tried v programe okrem tried Task, DCEvent, StoreKey. Implementuje

rozhranie DataCommListener, ktoré vnucuje metódu DataSourceEvent. Pre všetky GUI objekty sa automaticky stáva commandListenerom, teda ak používateľ vyvolá niektorú z funkcií z používateľského rozhrania, ošetrí ju práve trieda HandySignMidlet.

DataCommListener

Toto rozhranie zabezpečuje, že trieda, ktorá ho implementuje, bude mať metódu DataSourceEvent. Metóda DataSourceEvent je volaná príslušným DataCommServer-om alebo jeho potomkami ak sa na ich vstupe objavili údaje.

DataCommServer

Trieda je implementovaná ako vlákno, ktoré čaká na údaje. Ak dostane nové údaje, vyvolá DataSourceEvent jeho DataCommListenera. Odosieľať údaje je možné pomocou metódy dataToSend. Táto metóda iba pridá požiadavku na odoslanie údajov do zoznamu požiadaviek v DataCommServer-i. Samotné odoslanie a prijímanie údajov sa koná vo vlákne aby program nebol zdržovaný týmito blokujúcimi operáciami.

DCEvent

Požiadavky v triede DataCommServer sa ukladajú do poľa takýchto tried. Metóda getEventType vráti typ požiadavky napríklad požiadavka na odoslanie dát alebo prijatie kľúča či HASHu. Samotné dáta sú uložené v prúde, ktorý vráti metóda getDataStream.

ResourceServer

Táto trieda je vytvorená len pre testovacie účely. Simuluje nejaké fyzické zariadenie, ktoré odosiela údaje do programu. Údaje musia byť uložené ako zdroje vo vnútri jar balíčka a ich načítanie je možné pomocou funkcií z používateľského rozhrania.

IrdaCommServer

Je priamym potomkom triedy DataCommServer. Prepisuje iba niektoré metódy svojho predka zabezpečujúce samotné prijatie údajov alebo odoslanie. Jej komunikačný port je IRDA.

Task

Podobá sa na DCEvent, avšak táto trieda je prispôsobená potrebám triedy HandySignMidlet.

Fifo

Táto trieda je jednoduchým first-in-first-out bufferom. Slúži na uloženie tried Task.

HandyError

Je súčasťou používateľského rozhrania, zobrazuje chybové hlásenia.

HandyKeyList

Ak program dostane HASH, spúšťa práve obrazovku tvorenú touto triedov. Trieda HandyKeyList vyberie všetky kľúče z úložiska kľúčov HandyStore, zoradí ich do listu a zobrazí. Používateľovi dá na výber či chce použiť niektorý z dostupných kľúčov alebo zrušiť podpisovanie.

HandyKeyname

Ak program dostane kľúč, musí najprv zistiť pod akým menom ho má uložiť. Toto zabezpečí trieda HandyKeyName. Používateľovi dá na výber či chce kľúč uložiť, alebo ho zahodiť.

HandyPass

Potom ako si používateľ vyberie kľúč na podpisovanie aplikácia zistí či nieje šifrovaný. V prípade že je, trieda HandyPass vypýta od používateľa heslo. Používateľ môže tiež zrušiť podpisovanie.

HandyWaiting

Táto trieda zastrešuje hlavné menu v aplikácii a tiež určuje či je aplikácia v stave, keď môže vykonať nasledujúcu úlohu (trieda Task). Keď je trieda HandyWaiting zobrazená na display aplikácia čaká na dáta alebo na príkazy používateľa.

HandySigner

Je implementovaný ako vlákno a GUI prvok súčastne. Kvôli veľkej dĺžke čakania na podpísanie HASHu a možnosti zrušiť podpisovanie beží proces podpisovania v samostatnom vlákne, ktoré priebežne o svojom stave informuje používateľa na obrazovke.

StoreKey

Trieda dokáže načítať kľúč z formátu PEM či už šifrovaného alebo nie. Preparsuje ho a získa privátny kľúč vo forme aká je potrebná pre pre podpisovanie.

HandyStore

Je primárne určené ako úložisko kľúčov. Drží ich nezašifrovane, teda ak treba bezpečne používať toto úložisko, je nutné aby boli kľúče vo formáte PEM zašifrované.

EventLoop

Je to vlákno s nekonečným cyklom. V cykle sa zisťuje či je aplikácia pripravená na ošetrenie ďalšej udalosti, ak áno, tak ju táto trieda vyberie z fifo a spustí pre ňu ošetrujúcu funkciu.

1.3.3 Kľúče

Program by mal vedieť načítať privátne kľúče uložené vo formáte PEM. Formát by mal dokonale poznať, preto ho v nasledujúcom texte rozoberieme.

Začiatok súboru PEM, ktorý obsahuje privátny RSA kľúč je vždy rovnaký a to:

```
-----BEGIN RSA PRIVATE KEY-----
```

Ak sa za prvým riadkom nachádza nasledovný riadok, znamená to že privátny kľúč je zakódovaný:

```
Proc-Type: 4,ENCRYPTED
```

Ak je privátny kľúč zakódovaný, súbor musí ešte obsahovať riadok identifikujúci algoritmus šifrovania:

```
DEK-Info: DES-EDE3-CBC,4046AC1454B02220
```

Alebo

```
DEK-Info: DES-CBC,4046AC1454B02220
```

Kde identifikátor DES-EDE3-CBC znamená, že ide o šifrátor s algoritmom 3DES a so zreťazenými blokmi. Ak sa za reťazcom DEK-Info: nachádza identifikátor DES-CBC ide iba o šifrátor s algoritmom DES a so zreťazenými blokmi. Reťazec za identifikátormi šifrátora je takzvaná soľ, ktorá je hexadecimálne zakódovaná do reťazca znakov.

Nasleduje už iba samotný privátny kľúč v BASE64 kóde.

Koniec súboru PEM s privátnym RSA kľúčom vyzerá nasledovne:

```
-----END RSA PRIVATE KEY-----
```

1.3.4 Komunikačný protokol

Snahou pri návrhu komunikačného protokolu bola spraviť čím jednoduchší a efektívnejší komunikačný protokol. Týmto protokolom musia byť prenášané dva druhy dát z počítača na mobilný telefón: privátny kľúč fo formáte PEM (aj s hlavičkou), HASH a jeden druh dát smerom od mobilného telefónu k počítaču a to podpísaný HASH. Protokol je textovo orientovaný, to znamená, že znaky vstupujúce doňho musia byť vypísateľné a riadky musia byť ukončené buď dos štýlom alebo unix štýlom. PEM súbor je prevažne v kódovaní BASE64, čo ho robí textovým súborom. HASH a jeho podpísaná verzia by sa pred prenosom mala zakódovať kódovaním BASE64 a ukončiť koncom riadka.

Mobilný telefón načítava údaje ktoré mu prídu na vstupný port a ak sa mu podarí načítať riadok podobajúci sa na prvý riadok PEM súboru opisovaného vyššie, tak automaticky načíta zvyšné riadky primárneho kľúča až po posledný, ukončovací riadok PEM súboru. Všetky ostatné údaje berie ako HASH, ktorý treba podpísať.

Ak počítač dostane údaje zo vstupného portu, musí to byť podpísaný HASH, ktorý naposledy poslal mobilnému telefónu. Aplikácia bežiaca na počítači by mala rátať s tým, že požívateľ môže zrušiť podpisovanie, teda žiadny podpísaný HASH sa nevráti. Pre tento prípad by mala mať tlačítko na zrušenie čakania na podpísaný HASH.

1.4 Implementácia

Program HandySign používa na dešifrovanie privátnych kľúčov a podpísanie HASHu knižnicu bouncycastle verzia 2.29. Pre odosielanie a prijímanie dát z IRDA portu mobilného telefónu bola použitá knižnica SMTK verzie 2.10.52. Java J2ME verzie 1.4_8 bola použitá pre implementáciu ostatných funkcií programu.

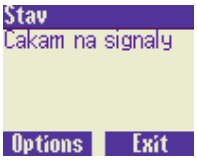
Pri implementácií sme narazili na niekoľko naozaj zložitých problémov. Knižnica bouncycastle je síce často používaná, veľká a obsahuje asi všetky možné spôsoby šifrovania, podpisovania, HASHovania... ale napriek tomu je len veľmi slabo zdokumentovaná. Na dôvažok, pre mobilné zariadenia je táto knižnica ešte značne osekaná a chýbali v nej práve tie triedy, ktoré by silne pomohli pri vývoji aplikácie HandySign. Na internete všetci uvádzajú, že privátny kľúč v súbore PEM je zapísany ako trieda PrivateKeyInfo z tejto knižnice, avšak všetci sa mýlia, asi to nikto z nich naozaj neskúšal, je nutné použiť triedu RSAPrivateKeyStructure.

Štandard J2ME nie je v dostatočnej miere podporovaný výrobcami mobilných zariadení, dokonca niektorý výrobcovia ho ani nedodržujú ani čo sa týka základov. Väčšina hardvérových doplnkov mobilného zariadenia je v štandarde spomenutá len ako odporúčanie, z ktorého si výrobcovia zjavne neberú príklad. Každy si implementuje serialove porty, IRDA, súborový systém... podľa svojho, z čoho vyplýva značná nekompatibilita javy medzi mobilnými telefónmi (poznámka autora: toľko o prenositeľnosti a multiplatformovosti javy). Aj keď na telefóne fyzicky je IRDA port, neznamená to však ešte, že je programovateľný cez javu. Prakticky sme našli iba jeden typ mobilného telefónu Siemens s55, ktorý umožnoval ovládať IRDA port cez javu. Ale ani tento mobil neostal bez vlastných interpretácií štandardov. Pre použitie IRDA portu bolo nutné použiť knižnicu SMTK dodávanú samotným Siemens-om, ktorá prakticky implementovala IRDA rovnakým spôsobom ako je odporúčané s drobnými zmenami, avšak táto knižnica spôsobila, že aplikácia SMTK už nie je spustiteľná na iných typoch mobilných telefónov a tiež zabránila obfuscovaniu výsledného java byte-kódu. Takto má výsledná aplikácia niečo okolo 500kB, čo je pre mobilný telefón s55 polovica diskového priestoru.

1.5 Používateľská príručka

Ako väčšina mobilných aplikácií aj HandySign je veľmi jednoduchý na použitie. Jeho otestovanie je možné aj bez pripojenia na počítač, má zabudované testovacie vzorky

zakódovaných a nezakódovaných privátnych kľúčov a jeden SHA1 hash. Keď sa program spustí, používateľ sa dostane na nasledujúcu obrazovku:



Obrázok 1.5-1 - Hlavná obrazovka

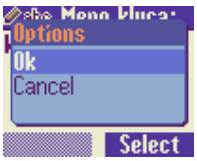
Tu je možné vyvolať menu alebo vypnúť aplikáciu. V tomto stave aplikácia počúva aj na IRDA porte telefónu. Ak vyvoláme hlavné menu, zobrazia sa nám tri položky:

Obrázok 1.5-2 Hlavné menu

„Test: kluc“ je testovacia vzorka nezašifrovaného kľúča, „Test: kodovany“ je testovacia vzorka zašifrovaného kľúča (heslo je „handysign“) a „Test: sha1“ je testovacia vzorka sha1 HASHu. Ak by sme zvolili jeden z kľúčov, zobrazí sa nám nasledujúca obrazovka:

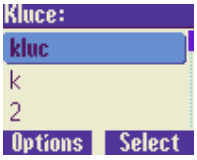
Obrázok 1.5-3 - Zadávanie mena pre kľúč

Aplikácia od nás žiada aby sme vložili meno kľúča, pod ktorým ho má uložiť do úložiska kľúčov.



Obrázok 1.5-4 - Potvrdenie mena pre kľúč

Options menu nám umožní potvrdiť uloženie kľúča ale aj jeho zahodenie. Hneď po potvrdení sa dostaneme do hlavnú obrazovku (Obrázok 1.5-1). Keby sme si v hlavnom menu zvolili položku „Test: sha1“ aplikácia nám ponúkne na výber všetky kľúče čo má v úložisku:



Obrázok 1.5-5 - Zoznam kľúčov

Ak si vyberieme zašifrovaný kľúč, aplikácia si od nás vypýta heslo:

Obrázok 1.5-6 - Zadávanie hesla

Ak bolo heslo správne, aplikácia začne s podpisovaním. Podpisovanie môže trvať aj niekoľko minút vzhľadom na pomalý procesor v mobilnom telefóne s55, pre ktorý je HandySign určený.

Obrázok 1.5-7 - Podpisovanie

Po potvrdení sa aplikácia pokúsi odoslať podpis na IRDA a zároveň sa používateľ dostane na hlavnú obrazovku (Obrázok 1.5-1).

2 Aplikácia na podpisovanie

Podpisovanie dokumentu je realizované zo zameraním na jednoduché používanie. Z daného dôvodu je aplikácia, používaná na podpisovanie, umiestnená priamo v textovom editore. Z dôvodu voľného šírenia a veľkej rozšírenosti medzi používateľmi sme ako textový editor vybrali aplikáciu „OpenOffice“. Danú aplikáciu je možné si stiahnuť v rôznych jazykových verziách priamo zo stránky www.openoffice.org.

Priamo v editore v hlavnom panely nástrojov je umiestnená ikonka, na ktorú keď sa klikne aktivuje sa aplikácia, ktorá nám zabezpečí bezpečné vytvorenie elektronického podpisu k danému textu, ktorý práve vytvárame.

Na začiatku je potrebné zadať meno a adresár na uloženie súboru, ktorý sa bude podpisovať. Dané riešenie je použité z dôvodu, že pôvodný dokument sa môže v priebehu času stále meniť. Elektronický podpis bude vytvorený k dokumentu, ktorý aktuálne uložíme a nebude prepísaný pri ďalšom editovaní dokumentu. Taktiež sa zachová dokument ako doklad pre neskoršie overenie. Doporučuje sa používať adresár, kde by boli uložené všetky dokumenty, ktoré boli určené na podpis a s nimi by boli uložené aj odpovedajúce elektronické podpisy.

2.1 Implementácia

Aplikácia využíva jazyk Visual Basic, ktorý je podporovaný v balíčku office od firmy Microsoft. Daný jazyk tiež používa program OpenOffice ako makrá, ktoré sa dajú priamo spúšťať. V OpenOffice bola vytvorená tzv. knižnica „Podpisovanie“, ktorá obsahuje modul „Module1“ a dialóg „DialogPodpis“. Modul obsahuje viacero makier potrebných pre aplikáciu. Hlavné makro „Main“ sa volá pri kliknutí na ikonku. V danom makre sa následné otvorí dialógové okno s jednotlivými prvkami pre ovládanie.

Po stlačení tlačítka „Podpísať“ v dialógovom okne sa uloží práve vytváraný dokument na vybrané miesto. Predtým sa overí, či zadaný adresár existuje a dokument je prekonvertovaný na formát rtf. V kóde si skopírujeme inštanciu otvoreného dokumentu a ten prekonvertujeme a uložíme. Danú činnosť popisujú nasledovné riadky, kde *Filename* je meno súboru, pod ktorým ho chceme uložiť:

```
oDoc = ThisComponent
```

```
oDoc2 = CreateUnoStruct( "com.sun.star.beans.Property" )

oDoc2 = oDoc

oDoc2.storeToUrl( sFilename, Array( MakePropertyValue(
"FilterName", "Rich Text Format" ) ) )
```

Následne sa z uloženého dokumentu vypočíta hash, pomocou funkcie z knižnice „diCrHash.dll“. Deklarácia potrebnej funkcie vyzerá naspedovne:

```
Declare Function SHA1_FileHexHash Lib "diCrHash.dll" (ByRef
sDigest As String, ByVal sFilename As String, ByVal sMode As
String) As Long
```

Daná knižnica je štandartná Win32 DLL, ktorá je kompatibilná z všetkými verziami Windowsov (95,98,Me,NT4,2K,XP). Nie je to žiadný „COM“ ani „Active-X“, takže nepotrebuje žiadne špeciálne používanie. Nevyžaduje žiadnu inú knižnicu a je nezávislá od Cryptographic API.

Na komunikáciu s mobilným telefónom sa využíva komunikácia pomocou virtuálneho sériového portu. Daný port je možné doinštalovať aj do nových windowsov pomocou patchu *IrCOMM*. V danom patchy sa nadstavý na ktorom porte bude prebiehať komunikácia pomocou IrDA. Na začiatku je potrebné si daný port otvoriť pomocou funkcie *OPen*, ktorá nám v podstate zavolá funkciu:

```
m_hPort=CreateFile(strPort,GENERIC_READ|GENERIC_WRITE,0,0,OPEN
_EXISTING, 0, 0)
```

Daná funkcia vytvorí *Handle* na komunikačný port a následne nadstaví všetky potrebné parametre, ktoré sa tykajú prenosu. Hash sa potom na mobilný telefón posiela pomocou funkcie:

```
port.Send(hash, TIMEOUT)
```

Daná funkcia následne v cykle posiela dáta. Čas pre posielanie musí byť časovo obmedzený a musí rešpektovať správanie sa portu.

```
WriteFile(m_hPort, pData, dwCount, dwBytesWritten, NULL);
```

Je to z dôvodu, že dáta nebudeme posielať naraz ako paket, ale postupne ako pri sériovej komunikácií. Taktiež mobilný telefón nemusí začať hneď prijímať údaje, ale až po nijakom čase.

Podobne ako funkcia *Send* funguje aj funkcia na prijímanie dát s IrDA.

```
port.WaitForResponse(strResponse, TIMEOUT)
```

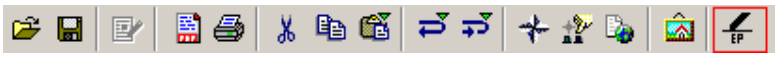
Pri úspešnom načítaní budú v premennej strResponze uložené dáta prijaté s IrDA. Dané dáta začne mobilný telefón posielať až po úspešnom zašifrovaní (podpísaní) vstupných dát (hashu). Z daným časom musí aplikácia počítať.

Následne do vybraného adresára v dialógu bude vytvorený elektronický podpis v tvare XML, ktorý bude obsahovať pôvodný hash aj podpísaný hash a ďalšie potrebné atribúty.

Aplikácia na inštaláciu je realizovaná ako dokument OpenOffice, kde sú uložené potrebné makrá. Do dokumentu je dané tlačítko, ktoré volá makro „InstallAddon“. Dané makro vytvorí priamo v aplikácií „OpenOffice“ novú knižnicu „Podpisovanie“ a prekopíruje do nej potrebné makrá a dialógy. Taktiež vytvorí ikonku v panely nástrojov.

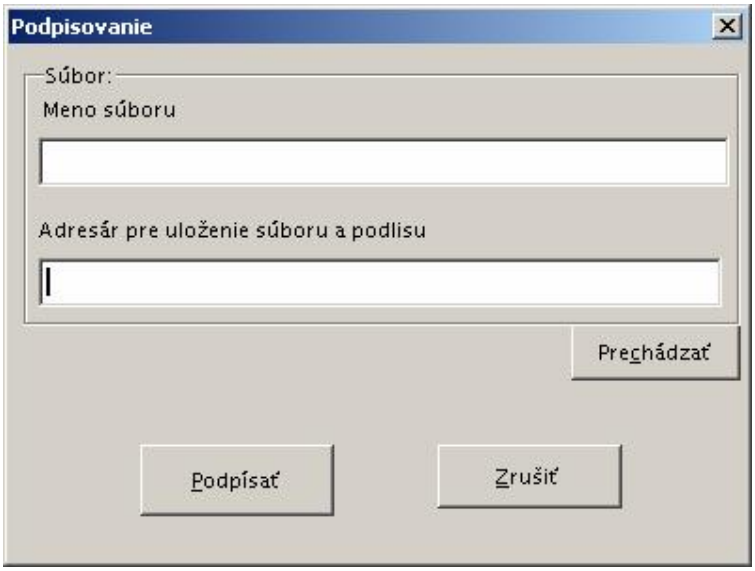
2.2 Používateľská príručka

Keď máme vytvorený textový sobor v editore OpenOffice, klikneme na ikonu na podpísanie dokumentu.



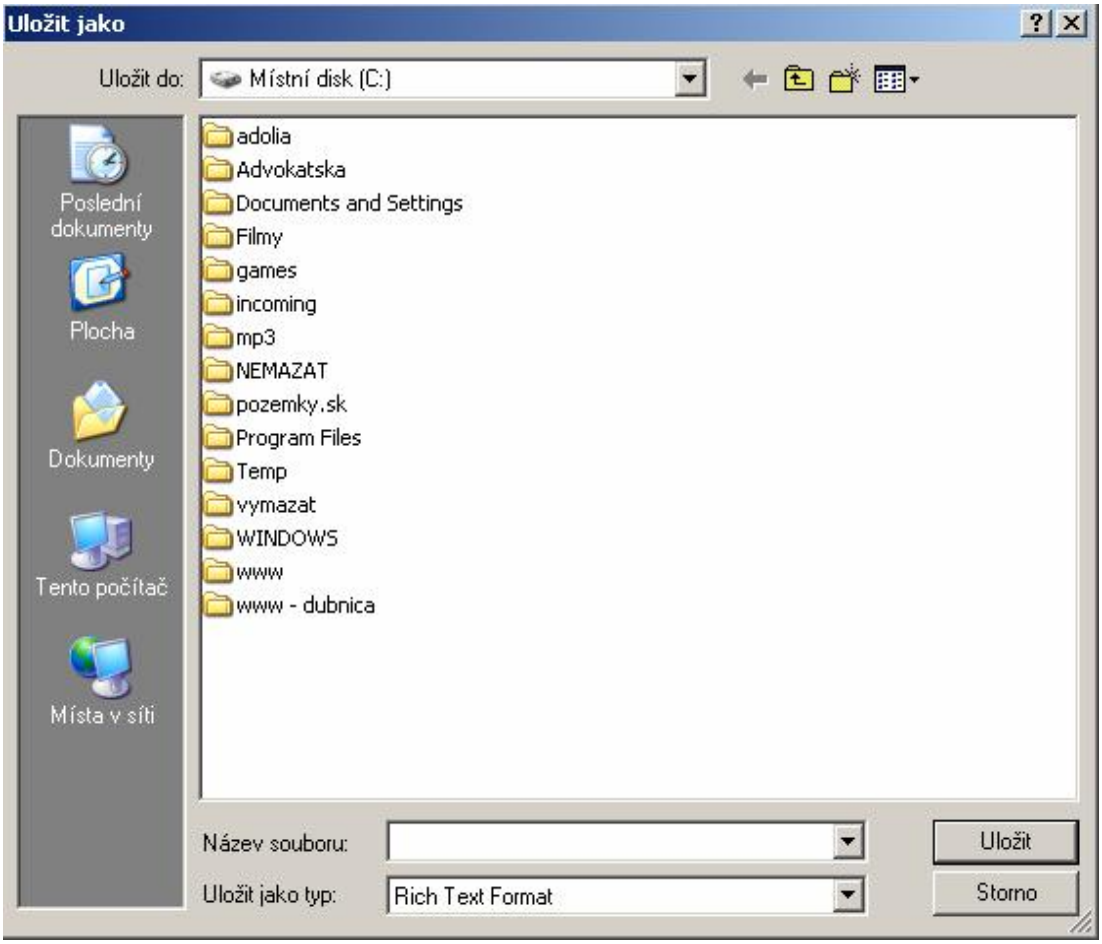
Obrázok č. 2.2-1 Ikonka v panely nástrojov (červený rámček)

Otvorí sa nám dialógové okno, kde zadáme meno a adresár, pre uloženie súboru a podpisu.



Obrázok č. 2.2-2 Dialóg, ktorý sa zobrazí po kliknutí na ikonku podpísať

Následne do editovacích polí zadáme meno súboru, pod ktorým bude súbor uložený a adresár, kde sa uloží dokument a elektronický podpis. Máme tiež možnosť naplniť dané polia pomocou štandartného dialógu na uloženie súboru. Daný modálny dialóg vyvoláme kliknutím na tlačítko „Prechádzať“. Po výbere adresára a zadaní mena súboru v dialógu kliknite na tlačítko uložiť. Následne sa naplnia editovacie polia v hlavnom dialógu. Súbor sa preddefinovane uloží vo formáte „Rich Text Format“ s príponou „rtf“. Daný súbor je povolené používať na podpisovanie a je to štandardný súbor, ktorý sa dá otvoriť vo väčšine textových editorov.



Obrázok č. 2.2-3 Dialóg na uloženie súboru

Po zadaní mena a adresára môžete kliknúť na ikonku „Podpísať“. Následne sa dokument uloží na vybrané miesto a vypočíta sa z neho hodnota hashu, ktorá sa pošle pomocou infračerveného portu na mobil.

3 Aplikácia na generovanie a ukladanie kľúčov

3.1 Implementácia

Aplikácia na generovanie a podpisovanie bola implementovaná ako Win32 aplikácia vo Visual Basic, ale využíva knižnicu, ktorá by fungovala aj pod C/C++. Na generovanie kľúčou bola použitá knižnica „diCrPKI.dll“, kde boli volané dané funkcie:

```
lngRes = RSA_MakeKeys(strPublicKeyFile, strPrivateKeyFile,
nBits, nExpFermat, nTests, nCount, strPassword, strSeed,
nSeedLen, nOptionFlags)

lngRet = RSA_MakeKeys (strNewCertFile, strEPKFile, nCertNum,
nYearsValid, strDistName, strEmail, KeyUsageFlags,
strPassword, nOptions)

lngRes = RSA_ReadEncPrivateKey(strPrivateKey, nKeyMaxLen,
strEpkFileName, strPassword, nOptions) As Long

lngRes = RSA_SavePrivateKeyInfo(strOutputFile, strPrivateKey,
nOptions) As Long

lngRes = RSA_SavePublicKey(strOutputFile, strPublicKey,
nOptions) As Long

lngRes = RSA_ReadPublicKey(strPublicKey, nKeyMaxLen,
strKeyFileName, nOptions) As Long

lngRes = RSA_SavePublicKey(strOutputFile, strPublicKey,
nOptions) As Long
```

Vo funkcií, ktorá sa volá po stlačení tlačítka „*generuj kľúče*“ sa zavolá funkcia *RSA_MakeKeys* s odpovedajúcimi parametrami vyhovujúcimi PKI. Funkcia uloží dva súbory, v jednom je uložený privátny a v druhom súkromný kľúč.

Pri posielaní súkromného kľúču na mobilný telefón sa reťazec najprv uloží ako PEM súbor a až ten je posielaný cez IrDA. Komunikácia bola bližšie popísaná pri aplikácií na podpisovanie. Po skončení sa súkromné kľúče z disku vymažú.

Po stlačení tlačítka „*Uložit*“ sa vytvorí certifikát verejného kľúča, ktorý sa podpíše našim práve vygenerovaným súkromným kľúčom. Na to slúži funkcia *RSA_MakeKeys*.

3.2 Používateľská príručka

Po spustení aplikácie *„kľúče“* sa nám zobrazí dialóg, v ktorom máme možnosť vygenerovať a uložiť nový kľúčový pár. Výzor dialógu je na obrázku:

Obrázok č. 3.2-1 Aplikácia na vygenerovanie a uloženie kľúčového páru

Po stlačení tlačítka sa nám vygeneruje nový kľúčový pár. V editovaciom okne je náhľad daného kľúčového páru.

Po stlačené tlačítka *„Pošli na mobil*“ aplikácia pošle cez port, ktorý sme vybrali v comboxe, súkromný vygenerovaný kľúč. Samozrejme, že mobilný telefón musí byť v stave čakajúcom na signály. Port, ktorý sme vybrali musí súhlasiť s portom, ktorý je pridelený v operačnom systéme na komunikáciu cez IrDA. Ak používate Windows 2000 alebo XP je potrebné doinštalovať potrebný ovládač, ktorý nám do COM portou pridá virtuálny ovládač na IrDA.

Keď stlačíte tlačítko „*Ulož*“, vyskočí Vám dialógové okno, kde si zadáte meno a cestu súboru pod ktorým si chcete uložiť novo vygenerovaný verejný kľúč. Budete ho potrebovať pre ďalšie šírenie. Váš osobný (privátny) kľúč bude uložený iba na mobilnom telefóne z dôvodu, aby bolo zabránené možnému odcudzeniu.

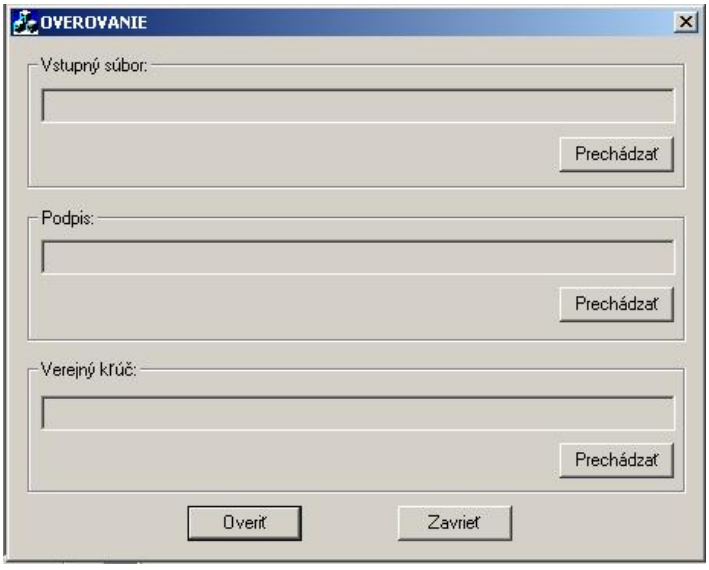
4 Aplikácia na overenie

4.1 Implementácia

Aplikácia na generovanie a podpisovanie bola implementovaná ako Win32 aplikácia vo Visual Basic. Sú tu použité funkcie s predchádzajúcich aplikácií. Zo vstupného súboru vypočítame Hash ako v aplikácií na podpisovanie v OpenOffice. Z certifikátu verejného kľúča dostaneme kľúč, ktorým dešifrujeme podpísaný Hash, uložený v XML podpise. Zistíme, či sa dešifrovaný reťazec zhoduje s pôvodným Hashom dokumentu, ktorý sme vypočítali a zároveň sa musí zhodovať aj z dešifrovaným Hashom uloženým v XML podpise. Ako je všetko v poriadku vyhlásime, že daný podpis patrí k danému súboru a je podpísaný súkromným kľúčom, ktorý je reverzný k verejnému kľúču, ktorý sme vybrali. Inak vyhlási nezrovnalosť.

4.2 Používateľská príručka

Na overenie správnosti fungovania elektronického podpisu slúži aplikácia „*Overovanie*“. Po zadaní súboru, ktorý chceme overiť „textový súbor OpenOfficu“ (*.sxw), certifikátu verejného kľúča (*.cer) a elektronického podpisu do formáte XML, stlačíme tlačítko „*Overit*“. Následne nám aplikácia vypíše, či daný elektronický podpis je správny. Teda či priložený podpis je podpis k danému súboru. Výzor aplikácie je na obrázku:



Obrázok č. 4.2-1Aplikácia na overenie elektronického podpisu

5 OpenCA

5.1 Úvod

Projekt OpenCA je výsledok snahy vyvinúť robustnú OpenSource Certifikačnú Autoritu. OpenCA podporuje väčšinu dnešných protokolov používaných v PKI. OpenCA je založené na viacerých OpenSource projektoch. Využíva OpenLDAP, OpenSSL, Apache, Apache mod_ssl. Vývoj projektu je rozdelený do dvoch hlavných častí. Prvá má za úlohu neustále študovať a zlepšovať bezpečnostnú schému PKI. Druhá získané poznatky implementuje do softvéru Certifikačnej Autority.

OpenCA ako také sa skladá z CGI skriptov, ktoré využívajú jazyk perl. Pre ukladanie údajov sa používa databáza MySQL. Ako webserver slúži Apache.

OpenCA sa skladá z viacerých funkčných blokov. Ide o verejný portál pre koncových používateľov, registračnú autoritu, a samotný systém certifikačnej autority. Ďalšie dve dôležité časti tvoria manažment registračnej autority a manažment certifikačnej autority. Verejný portál a registračná autorita bývajú zväčša nainštalované na jednom fyzickom počítači, avšak certifikačná autorita by mala byť nainštalovaná na zvlášť systéme, bez pripojenia do siete a na zabezpečenom mieste. Požiadavky od používateľov prijíma registračná autorita a po formálnej kontrole ju podsúva na podpis certifikačnej autorite. Za normálnych podmienok by to robil administrátor formou média, na ktorom by preniesol žiadosť o certifikát na server certifikačnej autority. V prípade LiveCD OpenCA však sú všetky tri časti OpenCA nainštalované na jednom mieste a je zabezpečená ich vzájomná komunikácia cez databázu.

5.2 Inštalácia

Pôvodne sme zamýšľali inštalovať OpenCA priamo zo zdrojových kódov na operačnom systéme Linux. Aktuálna verzia OpenCA bola v tom čase openca-0.9.2.1.tar.gz. operačný systém bol Debian Testing s v tom čase aktuálnymi verziami balíčkov. Počiatočné problémy s chýbajúcimi závislosťami (knižnice perl, ldap) sme síce vyriešili a zdrojové kódy sa nám podarilo skompilovať, následné testovanie systémom make test však skončilo s nezdokumentovanou chybou. Po dvoch týždňoch neúspešného pátrania sme sa rozhodli pre alternatívu.

Ako nosný systém sme použili LiveCD distribúciu založenú na KNOPPIX LiveCD s funkčnou inštaláciou OpenCA. V prípade KNOPPIX-u ide o verziu 3.2. KNOPPIX ako taký je založený na distribúcii Debian. OpenCA je nainštalované vo verzii 0.9.2. Pre pohodlnejšiu prácu sme systém pred inicializáciou nainštalovali na pevný disk. Tým sme odstránili problém, ktorým bola nutnosť bootovať z LiveCD po každom reštarte počítača. Vytvorenie perzistentnej kópie domovského adresára by síce vyriešilo mnohé problémy, ale neodstránilo by napríklad nutnosť nastavovania siete, bezpečnostného firewallu a pod. Napriek tomu, že používanie LiveCD distribúcie nepatrí medzi najčistejšie riešenia sa nám podarilo dosiahnuť stav podobný bežnej linuxovej distribúcie.

Použitie systému založeného na LiveCD sa v neskorších fázach projektu ukázalo ako výhodný fakt. Po niektorých neúspešných pokusoch a skúšaniach sa dalo jednoducho vrátiť na úplný začiatok. Projekt OpenCA sa totiž neskôr ukázal ako rozsiahli projekt, ktorého správne pochopenie z implementačného a konfiguračného hľadiska je nad rámec tohto projektu. Množstvo konfiguračných súborov, úzke previazanie s webovým serverom Apache a databázou MySQL neprispieva k prehľadnosti riešenia. Taktiež sa na systéme tlačia registračná autorita spolu s certifikačnou, ktoré by za normálnych okolností boli na oddelených systémoch.

5.3 Konfigurácia

V danom konkrétnom prípade nebola potrebná žiadna dodatočná konfigurácia OpenCA. Bolo však potrebné nakonfigurovať samotný systém, aby bol použiteľný ako plnohodnotný serverový systém.

Po inštalácii na pevný disk počítača bolo treba nakonfigurovať sieťové rozhranie, prideliť permanentnú IP adresu, povoliť na firewally prihlasovanie cez SSH. Napriek tomu, že LiveCD OpenCA má nainštalovanú Certifikačnú autoritu na jednom systéme spolu s registračnou, je zachovaná aspoň minimálna bezpečnosť a na rozhranie certifikačnej autority sa vďaka nataveniam webového servera Apache dá dostať len z lokálneho počítača. Z dôvodu pohodlnosti a lepšieho manažovania sme túto bezpečnostnú vlastnosť vypli. Testovací systém je totiž umiestnený mimo bežného dosahu a bolo by neefektívne chodiť kvôli každej operácii na certifikačnej autorite fyzicky k serveru. Keďže nejde o komerčnú inštaláciu, ale o akademickú testovaciu prevádzku, nepovažujeme tento krok za narušenie bezpečnosti, pretože okrem napadnutia certifikačnej autority nehrozia jej zneužitím žiadne materiálne ani

iné škody. Konkrétne sa táto vlastnosť vypínala v konfiguračnom súbore Apacha */etc/apache/conf.d/ openca.conf.*

OpenCA samotné je nainštalované v adresári */usr/local*, kde sa nachádzajú adresáre *openca* a *openra*. Verejné rozhranie pre používateľov je súčasťou registračnej autority. V týchto adresároch sa nachádzajú všetky súbory OpenCA. Konfiguračné súbory sa nachádzajú rôzne podľa zamerania.

5.4 Inicializácia

Po nainštalovaní OpenCA sú všetky hlavné časti prístupné pod URL http://localhost/pup, http://localhost/ra a http://localhost/ca. Ďalšie dve časti slúžiace na manažment sa dajú nájsť pod http://localhost/ra-node a http://localhost/ca-node.

Pre používanie je treba inicializovať certifikačnú autoritu a registračnú autoritu. V prvom kroku sa inicializuje databáza, ktorá bude slúžiť ako miesto ukladania certifikátov, CRL, žiadostí, atď. Následne sa pristúpi k vygenerovaniu kľúčového páru certifikačnej autority. Ďalej sa vytvorí žiadosť o certifikát certifikačnej autority. Na základe tejto žiadosti vydá certifikačná autorita certifikát sama sebe, tzv. ***self-signed*** certifikát. Ďalej je potrebné vytvoriť certifikát tzv. CA administrátorovi. Postupuje sa podobne ako pri žiadosti o normálny certifikát. Treba vygenerovať kľúčový pár, vytvoriť žiadosť o certifikát a nechať si vygenerovať certifikát. Podobne sa vygeneruje aj RA administrátorský certifikát. Ako posledný krok treba importovať konfiguráciu v ***ra-node*** administrátorskej časti.

Podrobnejší opis spolu s presným návodom kde sa postup opisuje krok za krokom sa dá nájsť na priloženom CD.

5.5 Lokalizácia

Lokalizáciu OpenCa je potrebné vykonať v nasledovných krokoch:

1. Preloženie textov, ktoré sa používajú v Perle
   1.1. Je potrebné vytvoriť nový adresár
   usr/local/openca/openca/lib/locale/xy_XY/LC_MESSAGES,
   usr/local/openra/openca/lib/locale/xy_XY/LC_MESSAGES (kde XY predstavuje nazov jazyka, napr. sk_SK)

1.2. Treba skopírovať src/common/lib/locale/openca.pot do src/common/lib/locale/xy_XY/openca.po

1.3. Preložiť openca.po (reťazce ako napriklad __FILE__, musia ostať nepreložené, pretože ich OpenCA dynamicky nahradzuje), nastaviť správnu kódovú stránku v hlavičke súboru

1.4. Vykonať príkaz msgfmt (msgfmt openca.po -o openca.mo) a vysledný súbor openca.mo skopírovať do adresára usr/local/openca/openca/lib/locale/xy_XY/LC_MESSAGES, usr/local/openra/openca/lib/locale/xy_XY/LC_MESSAGES

2. Preloženie textov, používaných v Javascript-e a VBscript-e:

2.1. Treba vytvoriť nasledovné adresáre /usr/local/openra/httpd/httdocs/ra/scripts/xy_XY, /usr/local/openra/httpd/httdocs/pub/scripts/xy_XY, /usr/local/openca/httpd/httdocs/ca/scripts/xy_XY

2.2. Do týchto adresárov treba skopírovať všetky súbory, ktoré sa nachádzajú v adresáry C, ktorý je vždy o úroveň vyššie

2.3. Nakoniec treba preložiť všetky texty v týchto súboroch.

3. Aktivovanie jazyka:

3.1. Do všetkých sekcií v častiach Languages (súbor usr/local/openca/openca/etc/menu.xml) treba pridať nový jazyk s odkazom na adresár s prekladom a s definovanou kódovou stránkou.

4. Reštartovanie servera s OpenCA.

Pre správne zobrazenie diakritiky v prehliadači je potrebné, aby bola nastavená kódová stránka na central european win-1250.

6 Záver

Cieľom projektu bola analýza štruktúry systému PKI, overenie funkčnosti projektu OpenCA, edukačná web prezentácia a aplikácia na podpisovanie a overovanie dokumentov. Analýza PKI a edukačná web prezentácia bola súčasťou zimného semestra a pojednáva sa o nej vo výstupnom dokumente zimného semestra.

V letnom semestri sme navrhli a naimplementovali všetky potrebné aplikácie na ktorých budeme prezentovať činnosť PKI. Ako bezpečné úložisko súkromného kľúča sme si zvolili mobilný mobilný telefón, s ktorým komunikujeme pomocou Infračerveného portu pripojeného na počítač. Z daným riešením sme mali značné problémy a vyriešenie vzájomnej komunikácie mobilného telefónu a PC nám zabralo najviac času. Bohužiaľ sa nám ešte nepodarilo odstrániť všetky problémy spojené s komunikáciou. Veríme, že do prezentácie aplikácie už budú dané problémy odstránené.

Priloha A - Obsah CD nosiča

Ponuka	dokumenty\ponuka.doc
Analýza PKI	dokumenty\analyza.doc
Posudok k analýze konkurenčného tímu	dokumenty\posudok.doc
Posudok konkurenčného tímu k analýze	dokumenty\posudok_OdXchoice.pdf
Reakcia na posudok konkurenčného tímu	dokumenty\reakcia_na_posudok.doc
Posudok na prezentáciu konkurenčného týmu	dokumenty\Posudok_nas_na_Web_Choice.doc
Posudok konkurenčného týmu k prezentácii	dokumenty\posudok_Xchoice_na_prezentaciu.pdf
Reakcia na posudok k prezentácii	dokumenty\reakcia_na_Xchoice_posudok_prezentacie.pdf
Všetky zápisnice	dokumenty\zapisnice
Návod na inštaláciu a konfiguráciu OpenCA	dokumenty\OpenCA-LiveCD.html
Dokumentácia k implementácii	dokumenty\implementacia.doc
Dokumentácia k implementácii Webovej aplikácie	dokumenty\prezentacia.doc
Prílohy k zápisom o stretnutí	dokumenty\rôzne
Webová prezentácia ilustrujúca princípy PKI	prezentacia\
Lokalizované súbory	zdrojove_kody\lokalizacia
Zdrojový kód s aplikáciou pre mobilný telefón	zdrojove_kody\java_mobil\handy_sign
Aplikácia pre OpenOffice	zdrojove_kody\openoffice

Priloha B - Riadenie projektu

Záznam o 1. stretnutí tímového projektu

číslo stretnutia	1
miesto stretnutia	D109
čas stretnutia	12.10.2004 - 9.15
zúčastnení	Jozef Hamar, Tomáš Smolek, Radomír Škrib
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

1. dohoda na termínoch stretnutí vrátane nasledujúceho stretnutia
2. Hudec: podrobnejšia špecifikácia požiadaviek, oznámenie o celkových predpokladaných črtách systému
3. zvolenie vedúceho (Jozef Hamar) a zástupcu vedúceho (Tomáš Smolek)
4. Jožo: zadelenie funkcií jednotlivým členom týmu

Zadelenie funkcií

funkcia	**jozef**	**peter**	**rado**	**tomáš**
vedúci	x			
výroba zápisnice			x	
analytik	x	x		
tvorba tímovej stránky				x
dizajn web stránky		x		
Programátor - flash			x	

Záznam o 2. stretnutí tímového projektu

číslo stretnutia	2
miesto stretnutia	D109
čas stretnutia	16.10.2004 - 9.15
zúčastnení	Jozef Hamar, Tomáš Smolek, Radomír Škrib, Martin Mačica, Peter Mulinka
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

5. diskusia o TP s vedúcim projektu
6. zmena v zadelení funkcií (viď tabuľka)
7. prezentácia rôznych návrhov stránok nášho tímu (Peter Mulinka), výber jedného návrhu
8. nasadenie vybratého návrhu na nám určené webové miesto (Peter Mulinka)
9. diskusia o vývojovom nástroji pre fázu 1 TP, zadelená úloha na analýzu možností, ktoré poskytuje Flash (Martin Mačica, Radomír Škrib)
10. úlohy: podrobné naštudovanie problematiky Jozef Hamar, Tomáš Smolek

Zadelenie funkcií

funkcia	**jozef**	**peter**	**rado**	**tomáš**	**martin**
vedúci tímu	x			z	
tvorba zápisnice			x		z
analytik	x	z		x	
tvorba tímovej stránky		x		z	
Webovská časť projektu		z		x	
Programátor - flash			x		x
Dokumentácia	x	x	x	x	xx

Záznam o 3. stretnutí tímového projektu

číslo stretnutia	3
miesto stretnutia	D109
čas stretnutia	23.10.2004 - 9.15
zúčastnení	Jozef Hamar, Tomáš Smolek, Radomír Škrib, Martin Mačica, Peter Mulinka
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

11. diskusia o TP s vedúcim projektu, doplnenie znalostí (Jozef Hamar)
12. diskusia o prvej časti TP – výber vývojového prostriedku (Macromedia Flash MX 2004)
13. úlohy na nasledovné týždne:

- špecifikácia návrhov pre prvú časť TP (Jožo, Tomáš)
- vypracovanie návrhov vo vybranom vývojovom prostriedku (Rado, Martin)

14. spravenie poriadku v súboroch na tímovom ftp (Tomáš)
5. aktualizovanie tímovej stránky (Peter)

Záznam o 4. stretnutí tímového projektu

číslo stretnutia	4
miesto stretnutia	D109
čas stretnutia	2.11.2004 - 9.15
zúčastnení	Jozef Hamar, Tomáš Smolek, Radomír Škrib, Martin Mačica, Peter Mulinka
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

15. diskusia o TP s vedúcim projektu, doplnenie znalostí (Jozef Hamar)
16. diskusia o prvej časti TP – prezentácia návrhov
17. úlohy na nasledovný týždeň:

- dokončenie analýzy (Jožo, Tomáš)
- vypracovanie návrhov vo vybranom vývojovom prostriedku (Rado, Martin)
- dokončenie a odovzdanie prvej časti dokumentácie (všetci)
- vyhľadanie vhodných obrázkov potrebných pre WEB aplikáciu demonštrujúcu princípy a činnosť PKI (Rado, Martin)

4. aktualizovanie tímovej stránky, úvod do dokumentácie (Peter)

Záznam o 5. stretnutí tímového projektu

číslo stretnutia	5
miesto stretnutia	D109
čas stretnutia	9.11.2004 - 9.15
zúčastnení	Jozef Hamar, Tomáš Smolek, Radomír Škrib, Martin Mačica, Peter Mulinka
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

18. diskusia o TP s vedúcim projektu, doplnenie znalostí (Jozef Hamar)
19. kontrola úloh zadelených na prechádzajúcom stretnutí (všetky splnené)
20. rozdelenie úloh, ktoré treba spraviť do 12.11. (odovzdanie dokumentácie)
21. úlohy ktoré treba vypracovať do konca týždňa:

- vyhotovenie analytickej časti dokumentácie (Jožo, Tomáš)
- vyhotovenie obrázkov potrebných v dokumentácii (Rado)
- napísanie úvodu do prvej časti dokumentácie (Peter)
- záverečné dokončenie dokumentácie (Martin)

22. aktualizovanie tímovej stránky (Peter)
23. prebratia a analýza práce konkurenčného týmu

Záznam o 6. stretnutí tímového projektu

číslo stretnutia	6
miesto stretnutia	D109
čas stretnutia	16.11.2004 - 9.15
zúčastnení	Jozef Hamar, Tomáš Smolek, Radomír Škrib, Martin Mačica, Peter Mulinka
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

24. diskusia o TP s vedúcim projektu
25. kontrola úloh zadelených na prechádzajúcom stretnutí
    - vyhotovenie analytickej časti dokumentácie (Jožo, Tomáš) – splnené
    - vyhotovenie obrázkov potrebných v dokumentácii (Rado) – splnené
    - napísanie úvodu do prvej časti dokumentácie (Peter) – splené
    - záverečné dokončenie a odovzdanie dokumentácie (Martin, Jožo)
26. rozdelenie úloh, ktoré treba spraviť do ďalšieho stretnutia

- vyhotovenie a odovzdanie posudku k projektu konkurenčného týmu (Martin, Jožo, Tomáš) – termín 19.11.
- scenáre (hrubý návrh) k webovskej aplikácii (Rado) – termín 23.11.
- hrubý návrh stránky kde pobeží webovská aplikácia (Peter) – termín 23.11.

27. aktualizovanie tímovej stránky (Peter)

Záznam o 7. stretnutí tímového projektu

číslo stretnutia	7
miesto stretnutia	D109
čas stretnutia	23.11.2004 - 9.15
zúčastnení	Jozef Hamar, Tomáš Smolek, Radomír Škrib, Peter Mulinka
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

28. diskusia vedúcim projektu o tom čo má obsahovať webovská aplikácia
29. preberanie a dopĺňanie navrhnutých scenárov, navrhnuté ďalšie scenáre
30. kontrola úloh zadelených na prechádzajúcom stretnutí
    - vyhotovenie a posudku k projektu konkurenčného týmu (Jožo, Tomáš) – termín 19.11. – splnené
    - posudku k projektu konkurenčného týmu (Martin) – termín 19.11. – nesplnené, odovzdaný až 22.11.
    - scenáre (hrubý návrh) k webovskej aplikácii (Rado) – termín 23.11. – splnené
    - hrubý návrh stránky kde pobeží webovská aplikácia (Peter) – termín 23.11. – splnené
31. rozdelenie úloh, ktoré treba spraviť do ďalšieho stretnutia

- reakcia na posudok od konkurenčného týmu (Peter) – termín 26.11. 10:00
- odovzdanie reakcie na posudok (Martin) – termín 26.11.
- úprava návrhu stránky pre webovskú aplikáciu, návrh menu, tvorba slovníku pojmov (Peter) – termín 30.11.
- doplnenie hrubého návrhu webovskej aplikácie (Rado) – termín 25.11.
- naprogramovanie prototypu webovskej aplikácie (Rado, Martin) – termín 30.11
- návrh scenárov pre overovanie certifikátov a CRL (Jožo, Tomáš)
- návrh scenára pre časovú pečiatku (Peter)

32. aktualizovanie tímovej stránky (Peter)

Záznam o 8. stretnutí tímového projektu

číslo stretnutia	8
miesto stretnutia	D109
čas stretnutia	30.11.2004 - 9.15
zúčastnení	Jozef Hamar, Tomáš Smolek, Radomír Škrib, Peter Mulinka, Martin Mačica
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

33. prezentácia prototypov prezentácie vedúcemu projektu
34. preberanie a dopĺňanie navrhnutých scenárov
35. kontrola úloh zadelených na prechádzajúcom stretnutí
    - reakcia na posudok od konkurenčného tímu (Peter) – termín 26.11. 10:00 – splnené
    - odovzdanie reakcie na posudok (Martin) – termín 26.11. – nebolo treba
    - úprava návrhu stránky pre webovskú aplikáciu, návrh menu, tvorba slovníku pojmov (Peter) – termín 30.11 – splnené
    - doplnenie hrubého návrhu webovskej aplikácie (Rado) – termín 25.11 – splnené
    - naprogramovanie prototypu webovskej aplikácie (Rado, Martin) – termín 30.11 – Rado – splnené, Martin – čiastočne splenené
    - návrh scenárov pre overovanie certifikátov a CRL (Jožo, Tomáš) – nesplnené
    - návrh scenára pre časovú pečiatku (Peter) – nesplnené
36. rozdelenie úloh, ktoré treba spraviť do ďalšieho stretnutia
- naprogramovanie všetkých scenárov (Martin, Rado) – termín 7.12.
- práca na stránke pre webovskú aplikáciu, integrovanie flashu (Peter) – termín 7.12.
- návrh scenárov pre overovanie certifikátov a CRL (Jožo, Tomáš) – termín 4.12.
- návrh scenára pre časovú pečiatku (Peter) – termín 4.12.

37. aktualizovanie tímovej stránky (Peter)

Záznam o 9. stretnutí tímového projektu

číslo stretnutia	9
miesto stretnutia	D109
čas stretnutia	7.12.2004 - 9.15
zúčastnení	Jozef Hamar, Radomír Škrib, Peter Mulinka, Martin Mačica
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

38. prezentácia prototypov webovej prezentácie vedúcemu projektu
39. preberanie a dopĺňanie navrhnutých scenárov, konečná úprava, definovanie štruktúry stránky webovej prezentácie
40. kontrola úloh zadelených na prechádzajúcom stretnutí

- naprogramovanie všetkých scenárov (Rado) – termín 7.12. – väčšina spravená
- naprogramovanie všetkých scenárov (Martin) – termín 7.12. – väčšina spravená
- práca na stránke pre webovskú aplikáciu, integrovanie flashu (Peter) – termín 7.12. – splnené
- návrh scenárov pre overovanie certifikátov a CRL (Jožo, Tomáš) – termín 4.12. – splnené
- návrh scenára pre časovú pečiatku (Peter) – termín 4.12. – splnené

41. rozdelenie úloh, ktoré treba spraviť do ďalšieho stretnutia

- **Peto**
  - skrátiť úvod
  - pridať banner
  - zmeniť štruktúru stránky, pridať posúvanie späť – dopredu, rolovanie a úprava obsahu
  - napísať inštalačnú a používateľskú príručku
- **Rado**
  - spraviť flash pre časovú pečiatku, ocsp, upraviť existujúce flashe (nahradiť „vygenerovať“, upraviť hierarchiu CA – RA, upraviť flash integrita, vymeniť archív súkromných kľúčov za CRL, pri odvolaní spísať možnosti,)
  - napísanie dokumentácie k implementácii
- **Tomáš**
  - k nasledovným flashom urobiť doprovodné texty: šifrovanie, certifikácia a CRL (napr. k žiadosti o vydanie popis PKCS10, popísať vlastnosti hashu, popísať štruktúru žiadosti a certifikátu atď)
- **Martin**
  - spraviť prezentáciu pre časť CRL, použitie a overenie certifikátov
  - obsah prezentácie, zhotovenie dokumentácie

- **Jožo**
  - doprovodné texty k nasledovným flashom: použitie a overovanie certifikátov, zneplatnenie certifikátov
  - prečítať a zaradiť OCSP

Záznam o 11 stretnutí tímového projektu

číslo stretnutia	11
miesto stretnutia	D109
čas stretnutia	28.02.2005 - 12.15
zúčastnení	Jozef Hamar, Radomír Škrib, Peter Mulinka, Martin Mačica, Tomáš Smolek
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

42. diskusia o tom čo treba ďalej spraviť
43. spojazdnenie, konfigurovanie a vytváranie CA
44. zaregistrovanie používateľov
45. kontrola úloh ktoré boli zadané na predchádzajúcom cvičení
    - **Peťo**
      - zmeniť formáty dokumentov na web stránke – spravené
      - premenovať dokumenty na webe (to čo je na CD musí byť aj na webe) – spravené
      - pridať na web posudok a reakciu na posudok – spravené
    - **Tomáš**
      - nájsť kde má openca uložené texty (kvôli prekladu) – spravené
    - **Martin**
      - upratať FTP – spravené
      - pozrieť Open Office (kvôli možnej implementácii openca) – spravené
    - **Jožo**
      - dať linku na openca – spravené
46. rozdelenie úloh, ktoré treba spraviť do ďalšieho stretnutia
    - **Peťo, Tomáš**
      - pozrieť si na internete ako sa kódujú aplikácie na mobile
    - **Martin**
      - nájsť a pozrieť nejaký plugin do Open Office (kvôli možnej implementácii openca)
    - **Všetci**
      - vyskúšať si prácu z CA a vytvoriť si vlastný certifikát

Záznam o 12 stretnutí tímového projektu

číslo stretnutia	4
miesto stretnutia	D109
čas stretnutia	14.03.2005 - 12.15
zúčastnení	Jozef Hamar, Radomír Škrib, Peter Mulinka, Martin Mačica, Tomáš Smolek
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

47. diskusia o tom čo treba ďalej spraviť
48. kontrola úloh ktoré boli zadané na predchádzajúcom cvičení
    - **Peťo**
        - pozrieť vývoj aplikácie na mobil – spravené
    - **Martin**
        - nájsť a pozrieť nejaký plugin do Open Office (kvôli možnej implementácii openca) – spravené
    - **Rado**
        - nájsť v OpenCA texty na preloženie, ak sa bude dať niečo preložiť – spravené
    - **Jožo**
        - preinštalovať OpenCA – spravené
49. rozdelenie úloh, ktoré treba spraviť do ďalšieho stretnutia
    - **Peťo**
        - pozrieť vývoj aplikácie na mobil cez IR
    - **Martin**
        - pozrieť Open Office a spraviť jednoduchý príklad
    - **Rado**
        - pozrieť sa na preklad textov
    - **Tomáš**
        - pozrieť posielanie na IR, nainštalovať Open Office

Záznam o 13 stretnutí tímového projektu

číslo stretnutia	13
miesto stretnutia	D109
čas stretnutia	21.03.2005 - 8.15
zúčastnení	Jozef Hamar, Radomír Škrib, Peter Mulinka, Tomáš Smolek
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

50. diskusia o tom čo treba ďalej spraviť, testovanie prekladu openca
51. kontrola úloh ktoré boli zadané na predchádzajúcom cvičení
    - **Peťo**
        - pozrieť vývoj aplikácie na mobil cez IR – splnené
    - **Martin**
        - pozrieť Open Office a spraviť jednoduchý príklad – nesplnené
    - **Rado**
        - pozrieť sa na preklad textov – splnené
    - **Tomáš**
        - pozrieť posielanie na IR, nainštalovať Open Office – splnené
52. rozdelenie úloh, ktoré treba spraviť
    - **Peťo**
        - pozrieť vývoj aplikácie na mobil cez IR – termín 30.3.2005
    - **Martin**
        - pozrieť Open Office a spraviť jednoduchý príklad – termín 30.3.2005
    - **Rado**
        - spraviť nejakú časť prekladu OpenCA – termín 30.3.2005
    - **Tomáš**
        - spraviť jednoduchý program v MFC s komponentmi na šifrovanie, dešifrovanie, overenie certifikátu, pozrieť si hlavne prácu s týmito komponentmi – termín 30.3.2005
    - **Jozef**
        - skúsiť rozchodiť posielanie mailov pod OpenCA – termín 4.4.2005

Záznam o 14 stretnutí tímového projektu

číslo stretnutia	14
miesto stretnutia	D109
čas stretnutia	04.04.2005 – 10:00
zúčastnení	Jozef Hamar, Radomír Škrib, Peter Mulinka, Tomáš Smolek
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

53. diskusia o tom čo treba ďalej spraviť, prezentácia jednoduchej aplikácie na mobile
54. kontrola úloh ktoré boli zadané na predchádzajúcom cvičení
    - **Peťo**
        - pozrieť vývoj aplikácie na mobil cez IR – termín 30.3.2005 - splnené
    - **Martin**
        - pozrieť Open Office a spraviť jednoduchý príklad – termín 30.3.2005 - nesplnené
    - **Rado**
        - spraviť nejakú časť prekladu OpenCA – termín 30.3.2005 - splnené
    - **Tomáš**
        - spraviť jednoduchý program v MFC s komponentmi na šifrovanie, dešifrovanie, overenie certifikátu, pozrieť si hlavne prácu s týmito komponentmi – termín 30.3.2005 - splnené
    - **Jozef**
        - skúsiť rozchodiť posielanie mailov pod OpenCA – termín 4.4.2005 – nepodarilo sa
55. rozdelenie úloh, ktoré treba spraviť
    - **Peťo**
        - pokračovať vo vývoji aplikácie na mobil cez IR – termín 20.4.2005
    - **Martin**
        - pozrieť Open Office a spraviť jednoduchý príklad – termín 6.4.2005
    - **Rado**
        - dokončiť preklad OpenCA – termín 11.4.2005
    - **Tomáš**
        - spraviť program v MFC s komponentmi na šifrovanie, dešifrovanie, overenie certifikátu – termín 20.4.2005
        - nainštalovať Martinom dodaný Open Office (ak nedodá, tak vlastný) – termín 11.4.2005
    - **Jozef**
        - preklad dokumentu popisujúceho prácu s OpenCA – termín 20.4.2005

- **Všetci**
  - napísať dokumentáciu ku svojej časti – termín 20.4.2005

Záznam o 15 stretnutí tímového projektu

číslo stretnutia	15
miesto stretnutia	D109
čas stretnutia	11.04.2005 – 10:00
zúčastnení	Jozef Hamar, Radomír Škrib, Tomáš Smolek
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

56. diskusia o tom čo treba ďalej spraviť, prerozdelenie úloh, keďže Martin ukončil štúdium, jeho záväzky preberá Tomáš
57. kontrola úloh, ktoré boli zadané na predchádzajúcom cvičení
    - **Peťo**
        - pokračovať vo vývoji aplikácie na mobil cez IR – termín 20.4.2005
    - **Martin**
        - pozrieť Open Office a spraviť jednoduchý príklad – termín 6.4.2005 – nesplnené
    - **Rado**
        - dokončiť preklad OpenCA – termín 11.4.2005 – splnené na 50%
    - **Tomáš**
        - spraviť program v MFC s komponentmi na šifrovanie, dešifrovanie, overenie certifikátu – termín 20.4.2005
        - nainštalovať Martinom dodaný Open Office (ak nedodá, tak vlastný) – termín 11.4.2005 – splnené
    - **Jozef**
        - preklad dokumentu popisujúceho prácu s OpenCA – termín 20.4.2005
58. rozdelenie úloh, ktoré treba spraviť
    - **Peťo**
        - pokračovať vo vývoji aplikácie na mobil cez IR – termín 20.4.2005
    - **Rado**
        - dokončiť preklad OpenCA – termín 20.4.2005
    - **Tomáš**
        - spraviť program v MFC s komponentmi na šifrovanie, dešifrovanie, overenie certifikátu – termín 20.4.2005
        - spraviť Martinovu časť – vytvorenie hešu z dokumentu v OpenOffice
    - **Jozef**
        - preklad dokumentu popisujúceho prácu s OpenCA – termín 20.4.2005
    - **Všetci**
        - napísať dokumentáciu ku svojej časti – termín 20.4.2005

Záznam o 16 stretnutí tímového projektu

číslo stretnutia	16
miesto stretnutia	D109
čas stretnutia	18.04.2005 – 10:00
zúčastnení	Jozef Hamar, Radomír Škrib, Peter Mulinka
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

59. diskusia o tom čo treba spraviť do 21.4., keďže sa odovzdáva prvá verzia produktu, prerozdelenie úloh, preloženie ďalších stretnutí na piatok
60. rozdelenie úloh, ktoré treba spraviť
    - **Peťo**
        - pokračovať vo vývoji aplikácie na mobil cez IR – termín 20.4.2005
        - pomôcť Jožovi s finalizovaním dokumentácie
    - **Rado**
        - dokončiť preklad OpenCA – termín 25.4.2005
        - spraviť obsah CD a napísať k nemu dokumentáciu – termín 20.4.2005
    - **Tomáš**
        - spraviť program v MFC s komponentmi na šifrovanie, dešifrovanie, overenie certifikátu – posúva sa na neurčito, záleží či nato bude čas
        - spraviť Martinovu časť – vytvorenie hešu z dokumentu v OpenOffice
    - **Jozef**
        - preklad dokumentu popisujúceho prácu s OpenCA – termín 20.4.2005
        - dať dokopy dokumentáciu, ktorá sa bude odovzdávať 21.4.2005
    - **Všetci**
        - napísať dokumentáciu ku svojej časti – termín 20.4.2005

Záznam o 17 stretnutí tímového projektu

číslo stretnutia	17
miesto stretnutia	D109
čas stretnutia	06.05.2005 – 10:00
zúčastnení	Jozef Hamar, Radomír Škrib, Peter Mulinka, Tomáš Smolek
vedúci	Doc. Ing. Ladislav Hudec, PhD.
vypracoval	Radomír Škrib
overil	Jozef Hamar

Záznam stretnutia:

61. diskusia o tom čo treba ešte spraviť
62. rozdelenie úloh, ktoré treba spraviť
    - **Peťo**
        - pokračovať vo vývoji aplikácie na mobil cez IR – termín 20.4.2005 - splnené
        - pomôcť Jožovi s finalizovaním dokumentácie - splnené
    - **Rado**
        - dokončiť preklad OpenCA – termín 25.4.2005 - splnené
        - spraviť obsah CD a napísať k nemu dokumentáciu – termín 20.4.2005 - splnené
    - **Tomáš**
        - spraviť program v MFC s komponentmi na šifrovanie, dešifrovanie, overenie certifikátu – posúva sa na neurčito, záleží či nato bude čas
        - spraviť Martinovu časť – vytvorenie hešu z dokumentu v OpenOffice – splnené
    - **Jozef**
        - preklad dokumentu popisujúceho prácu s OpenCA – termín 20.4.2005 - splnené
        - dať dokopy dokumentáciu, ktorá sa bude odovzdávať 21.4.2005 – splnené
    - **Všetci**
        - napísať dokumentáciu ku svojej časti – termín 20.4.2005 - splnené
63. rozdelenie úloh, ktoré treba spraviť
    - **Tomáš**
        - spraviť program, ktorý pošle hash na mobil – termín 16.5.2005
    - **Všetci**
        - spraviť finálnu dokumentáciu ku svojej časti – termín 16.5.2005

Počas letného semestra náš tým opustil jeden z členov, Bc. Martin Mačica. Tento člen bol zodpovedný za aplikáciu, ktorá mala na starosti integrovanie do systému Open Office a komunikáciu s mobilným telefónom. Tymto krokom nám zapríčinil viaceré časové sklzy. Bolo potrebné prehodnotiť priority projektu. Z toho dôvodu sme znížili prioritu časti, ktorá mala na starosti overovanie už podpísaného dokumentu. Člen týmu, Bc. Tomáš Smolek bol nútený prebrať zadania, ktoré mal pôvodne na starosti p. Mačica.

A toho dôvodu sa vo výslednom projekte nenachádza riešenie časti overovania dokumentu.