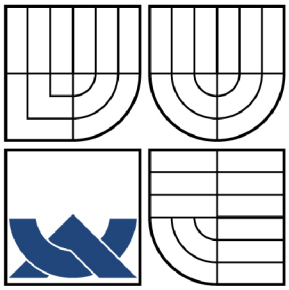
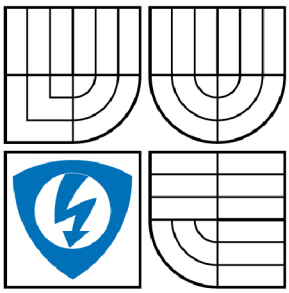


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

LABORATORNÍ ÚLOHA INFRASTRUKTURY VEŘEJNÝCH KLÍČŮ

LAB OF PUBLIC KEY INFRASTRUCTURE

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

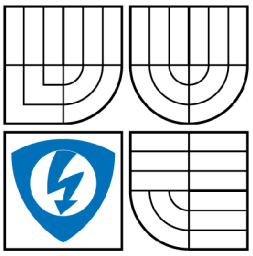
AUTOR PRÁCE — Bc. PETR SLAVÍK

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. KAREL BURDA, CSc.

SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Petr Slavík	***ID:***	84867
Ročník:	2	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Laboratorní úloha infrastruktury veřejných klíčů

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte a popište problematiku infrastruktury veřejných klíčů jak z odborného hlediska, tak i z hlediska legislativy ČR. Dále navrhněte laboratorní úlohu k vytvoření a provozování jednoduché infrastruktury veřejných klíčů. Obsahem úlohy bude vytvoření certifikační autority, vytvoření páru klíčů uživatele, vytvoření certifikátu veřejného klíče uživatele a praktické využití tohoto certifikátu při řízení přístupu. Úloha musí být uskutečnitelná na jediném počítači. Navrženou úlohu prakticky ověřte a zpracujte k ní potřebnou dokumentaci.

DOPORUČENÁ LITERATURA:

[1] Hanáček, P. - Staudek, J.: Certifikační infrastruktury veřejných klíčů, PKI. Učební text MU, Brno 2006.
[2] -: Zákon o elektronickém podpisu (č. 227/2000 Sb.). Parlament ČR, Praha 2000.

Termín zadání:	9.2.2009	***Termín odevzdání:***	26.5.2009

Vedoucí práce: doc. Ing. Karel Burda, CSc.

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Cílem této práce je seznámit čtenáře se systémem PKI od jednotlivých dílčích stavebních prvků, kterými jsou především kryptografické operace (asymetrické a symetrické šifrování, hashovací funkce, digitální podpis), přes podrobný popis jednotlivých dílčích subjektů systému PKI (certifikační autority, certifikáty, bezpečnostní protokoly, bezpečná úložiště) až po popis kompletních řešení infrastruktury veřejných klíčů (OpenSSL, Microsft CA).

Praktická část práce, tedy laboratorní úloha, studenty prakticky seznámí s metodou instalace certifikační autority postavené na systému OpenSSL, dále se zabezpečením webového serveru certifikátem vydaným vlastní CA a na závěr s možnostmi řízení přístupu uživatelů k webovému serveru prostřednictvím certifikátů jim vydaným dříve nainstalovanou CA.

Klíčová slova

Infrastruktura veřejných klíčů, PKI, OpenSSL, Zabezpečení apache, Certifikační autorita, Registrační autorita, Řízení přístupu, certifikát, USB token, SSL, TLS, digitální podpis

Abstract

The aim of this thesis is to study and describe the theme of Public Key Infrastructure (PKI). Within the scope of minute PKI characterization there is a gradual depiction of particular structural elements, which are above all represented by cryptographic operations (asymetric and symetric cryptography, hash function and digital signature); then, there are also individual PKI subjects that are dealt with, like eg. certification authority, certificates, security protocols, secure heap etc. Last but not least there are a few complete Public Key Infrastructure implementation solutions described (OpenSSL, Microsft CA).

The practical part of the thesis, a lab exercise, gives potential students the knowledge of installing OpenSSL system based certification authority. The next task educate students how to secure web server with certificate signed with own CA and also how to secure web server users‘ access control through certificates signed by the previously installed CA.

Keywords

Public Key Infrastructure, PKI, OpenSSL, Apache Security, Certificate Authority, Registration Authority, Access control, Certificate, USB token, SSL, TLS, Digital signature

SLAVÍK, P. Laboratorní úloha infrastruktury veřejných klíčů. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 56 s. Vedoucí diplomové práce doc. Ing. Karel Burda, CSc.

Prohlášení o původnosti práce

Prohlašuji, že svou diplomovou práci na téma Laboratorní úloha infrastruktury veřejných klíčů jsem vypracoval samostatně pod vedením vedoucího semestrálního projektu a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením tohoto projektu jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č.140/1961 Sb.

V Brně dne 24. května 2009 .....

podpis autora

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce doc. Ing. Karlu Burdovi, CSc., za velmi užitečnou metodickou pomoc, poskytnutí užitečných informací v oblasti bezpečnosti a šifrování a v neposlední řadě také cenných rad při zpracování diplomové práce.

V Brně dne 21.5.2009 .....

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

1. ÚVOD

Současné **počítačové sítě** nejsou již dávno uzavřenými systémy. Tvoří je naopak *komplexní informační propojení* sestávající se ze sítí intranet, serverů sítě Internet, sítí extranet a dalších dílčích prvků.

Všechny tyto součásti počítačových sítí jsou vystaveny potenciálnímu **riziku napadení**, respektive neoprávněnému průniku za účelem odcizení či modifikace dat určitého subjektu. Konkrétním terčem takovýchto útoků jsou často prostředky *elektronické komunikace*, jako např. e-maily, obchodní transakce, či data ve formě přenášených souborů.

Účinnou obranou, které je tato práce věnována, je použití *infrastruktury veřejných klíčů*. Obecně se však za účinnou ochranu proti napadení již považuje samotná *autentizace pomocí hesla*. Aby však toto heslo bylo z bezpečnostního hlediska efektivní, musí obsahovat dostatečný počet náhodně zvolených znaků. V praxi však uživatelé, ve snaze si zadávaní a zapamatování hesel usnadnit, toto kritérium často podceňují. Využívají stejná hesla pro více systémů a mnohdy volí jednoduchá, či snadno odhadnutelná hesla, čímž potenciálním narušitelům velmi usnadňují práci. Nejen tento, ale i mnohé další bezpečnostní problémy jsou právě řešeny **infrastrukturou veřejných klíčů**.

Pojem *infrastruktura veřejných klíčů* pochází z doslovného anglického překladu **Public Key Infrastructure**. V češtině se také tento termín brzy rozšířil a často se používá zkratka pocházející právě z tohoto anglického názvu - **PKI**.

Infrastruktura veřejných klíčů (PKI) je systém digitálních certifikátů, certifikačních autorit (CA) a dalších registračních úřadů, které slouží k ověřování platnosti každé strany zúčastněné v určité elektronické transakci. Jako základní stavební kámen je přitom používána tzv. **asymetrická kryptografie** s *veřejným a soukromým klíčem*.

PKI dnes tvoří *základní bezpečnostní pilíř* ve světě ICT. Příkladem jeho použití je např. zabezpečení přístupu do internetových aplikací, zabezpečení přístupu do VPN, vzdálený přístup a autentizace uživatelů, bezpečné DNS (DNSSEC), zabezpečení

elektronických transakcí, přihlašování do operačního systému, bezpečný e-mail a mnoho dalších.

Většina z výše uvedených implementací systému PKI využívá jako *propojovací prvek* mezi klienty a PKI systémem **protokol SSL/TLS**, kterému bude velká část této práce věnována a na jehož základě bude též postavena samotná výsledná laboratorní úloha.

Vypracovaná laboratorní úloha a vlastně i celá tato práce si klade za cíl **seznámit čtenáře se systémem infrastruktury veřejných klíčů** od jednotlivých dílčích stavebních prvků, kterými jsou především *kryptografické operace*, přes podrobný *popis* jednotlivých dílčích *subjektů systému PKI* (certifikační autority, certifikáty, bezpečnostní protokoly, bezpečná úložiště) až po *popis kompletních řešení* infrastruktury veřejných klíčů (OpenSSL, Microsoft CA).

První kapitola je věnována **kryptografickým metodám**, které tvoří základní stavební kámen fungování infrastruktury veřejných klíčů. Mezi tyto operace patří *hashovací funkce, asymetrické a symetrické šifrování* a především princip *digitálního podpisu* a všeobecně digitálního podepisování elektronických dokumentů, pomocí něhož jsou vytvářeny elektronické certifikáty.

Samotným **certifikátům** je věnována následující kapitola, ve které je podrobně popsán význam a použití certifikátu se zaměřením na certifikát, dle dnes používaného standardu X.509 verze 3. Samozřejmě jsou součásti popisu i jednotlivé prvky tohoto standardu a jejich význam. Závěr kapitoly je věnován tzv. *kvalifikovanému certifikátu*, který je definován českou legislativou *Zákonem o elektronickém podpisu*, a to včetně ukázky jeho praktického využití v ČR.

Navazující kapitola popisuje jednotlivé **komponenty infrastruktury veřejných klíčů**, především její hlavní subjekty, kterými jsou *certifikační autorita, registrační autorita* a *seznam odvolaných certifikátů* (CRL). V závěru kapitoly jsou popsány jednotlivé způsoby propojení tzv. *topologie* těchto *PKI systémů*.

Jelikož nejdůležitějším prvkem práce s PKI certifikáty jsou **veřejné** a především **soukromé klíče** uživatelů, které je třeba zachovat maximálně utajené a důvěryhodné, je další z kapitol věnována prostředkům pro uskladnění těchto soukromých klíčů. V této kapitole jsou podrobně popsány *USB tokeny a PKI čipové karty*, dále možná *rizika* hrozící těmto úložištím, včetně možností *obrany* před těmito riziky. V závěru kapitoly je objasněn tzv. *HSM modul*, používaný pro ochranu klíčů důležitých serverů a kořenových certifikačních autorit.

V následující kapitole jsou popsány některé praktické **implementace PKI systému**. Jelikož většina těchto implementací je založena na využití *protokolu SSL/TLS*, je tomuto protokolu věnována podstatná část této kapitoly a celý protokol je zde dopodrobna

rozebrán. V rámci protokolu jsou popsány *metody jeho využití*, jak pro autentizaci serveru, tak i pro autentizace jednotlivých klientů. Dalšími popsanými využitími systému PKI poté jsou *zabezpečení elektronické pošty*, *zabezpečení DNS* a také zabezpečení *autentizace WiMAX klientů*.

Poslední teoretická kapitola pak již charakterizuje konkrétní **dostupná řešení infrastruktury veřejných klíčů**, se zaměřením především na open source implementaci *OpenSSL* a *certifikační službu Microsoft* (tzv. MSCA) dostupnou v operačních systémech Microsoft Windows Server.

Poté následuje praktická část práce, tedy **laboratorní úloha**, ve které se studenti prakticky seznámí s metodou *instalace certifikační autority postavené na systému OpenSSL*, dále se *zabezpečením webového serveru certifikátem vydaným vlastní CA* a na závěr s možnostmi *řízení přístupu uživatelů k webovému serveru prostřednictvím certifikátů jim vydaným dříve nainstalovanou CA*.

2. KRYPTOGRAFICKÉ METODY

V dnešní době je kryptografie již nezbytnou součástí běžného života, má uplatnění jak v profesionálním světě, tak i v soukromém životě. Firmy si potřebují vyměňovat interní strategické informace, osobní údaje zaměstnanců, potřebují navzájem nerušeně komunikovat, anebo jen by neradi na svém počítači nechali nezajištěná data přístupná např. i jen své sekretářce.

Kryptografické metody obecně využívají tzv. "klíč", pomocí kterého tajná data zakódují a posléze opět rozkódují. Současně některé metody umožňují nebo i vynucují použití více klíčů různých pro zakódování a rozkódování. Původní nezašifrovaný obsah bývá ve většině publikací nazýván **otevřený text** (OT). Zakódováním či zašifrováním otevřeného textu, vzniká tzv. **šifrovaný text** (ŠT), který by měl být již opravdu nečitelný bez vlastnictví znalostního klíče. Tento **znalostní klíč** slouží k zajištění důvěrnosti chráněných dat a bez jeho vlastnictví není možné šifrovaný text dešifrovat do původní podoby otevřeného textu.

Většina moderních algoritmů je založena na matematické teorii čísel. Tzv. kryptografická transformace T je libovolné prosté zobrazení množiny celých čísel na množinu celých čísel. Kryptografický systém je pak parametrický systém kryptografických transformací T^K = (Tk: k ε K), kde k je klíč a K je prostor klíčů.

Podle použití způsobu práce s klíči se kryptografické metody dělí na symetrické a asymetrické (viz. níže).

2.1. Hashovací funkce

Hashovací funkce jsou funkce, které libovolně dlouhému vstupnímu řetězci znaků (zpráva, data apod.) přiřazují výstup pevné délky, který bývá nazýván *hash*, neboli kontrolní součet. Bývají označovány jako jednocestné funkce – výpočet hashe není výpočetně náročný, nalézt však původní text by však mělo být v reálném čase nemožné. Hashovací funkce zajišťuje, že jakkoli by se vstup změnil i o jediný bit, bude výsledek diametrálně odlišný. Bývají využívány např. pro kontrolu *integrity dat* (zajištění integrity

je proces ustavení určité míry jistoty o tom, že zpráva nebyla na cestě od odesilatele k příjemci změněna) a pro kontrolu *autenticity* (pravost, hodnověrnost, např. digitální podpis). Otisk spočtený ze zprávy, popř. ze zřetězené zprávy obsahující sdílené tajemství bývá označován MAC (Message Authentication Code). MAC se velice často využívá v protokolech SSL/TLS, protokolu IPSec apod. Mezi nejznámějšími hashovacími funkce lze uvézt **MD5** (otisk 16 B)**, SHA1** (otisk 20 B)**, SHA2** (64 B), či hashovací funkce **SNMAC**, která je vyvíjena naším předním českým kryptologem Dr. Vlastimilem Klímou.

2.1.1. Hashovací funkce MD5

Zkratka MD5 vychází z původního anglického názvu Message-Digest algorithm 5. Její délka je 128bitů (16 bajtů) a je převážně využívána pro kontrolu integrity dat, nebo ukládání hesel. Algoritmus MD5 nefunguje na principu šifrování, ani data jakkoliv nepozměňuje, nýbrž z nich vytvoří jistý *otisk* o délce 128 bitů, podle kterého je možné poznat integritu, či autenticitu dat. Tab. 2.1 názorně demonstruje rozdílnost hashe hesla, které se liší pouze ve velikosti počátečního znaku.

Tab. 2.1: Ukázka hashování

Heslo	**MD5 Hash**
admin123	0192023a7bbd73250516f069df18b500
Admin123	e64b78fc3bc91bcbc7dc232ba8ec59e0

V roce 2004 byly objeveny zásadní bezpečnostní rizika v algoritmu MD5 (již dříve zmíněným českým kryptologem Dr. Klímou), díky kterým bylo umožněna poměrně rychlá zpětná analýza hashe do původního řetězce (v řádu minut).

2.2. Symetrická kryptografie

Z historického hlediska bylo šifrování používáno již před více než tisícem let například za vlády G. J. Césara, který používal tzv. *monoalfabetickou šifru*. Jejím principem je změna jednotlivých znaků abecedy za jiné. Nejčastěji bývá založena na pouhém posunutí abecedy o x znaků. V případě Césarovy šifry (viz. Tab. 2.2) se jednalo právě o posun abecedy o 3 znaky.

Tab. 2.2: Césarova šifra

Otevřený text	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Šifrovaný text	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Tento způsob šifrování patří do skupiny tzv. Symetrické kryptografie. Ta pracuje s jedním klíčem, který je společný všem komunikujícím stranám a je využíván pro šifrování i dešifrování textu. Obrovskou výhodou tohoto způsobu šifrování je její rychlost a možnost práce s velkým objemem dat. Nevýhodou je ovšem skutečnost, že ten kdo data zašifroval, je umí i dešifrovat a komunikují-li spolu dvě strany, je nutné, aby si klíč bezpečně předaly důvěrnou cestou. Bezpečnost šifry také závisí na kvalitě použitého klíče - musí být dostatečně komplexní a dostatečně náhodný, jinak je šifra snadno prolomitelná. V případě zabezpečení *komunikace* s větším počtem klientů touto metodou je nevýhodou nutnost vysokého počtu klíčů a jejich následná správa. Do symetrické šifry spadají např. šifrovací algoritmy **DES** (používán před téměř 30 lety, dnes již překonán), různé **transpoziční šifry** a v současné době nejvíce používaný symetrický algoritmus **AES** a **3DES**.

2.3. Asymetrická kryptografie

Hlavním rysem asymetrického šifrování je existence dvou klíčů – **veřejného** a **soukromého (privátního) klíče**. Jedná se o takzvaný *klíčový pár*, který má specifickou vlastnost, že text zašifrovaný jedním klíčem lze dešifrovat pouze druhým klíčem z páru. Uživatel může svůj veřejný klíč libovolně šířit, avšak svůj soukromý klíč musí uchovávat v dokonalé tajnosti. Obecně však platí, že asymetrické šifrovací algoritmy jsou výpočetně náročnější než symetrické algoritmy. Nejznámější asymetrickým mechanismem je algoritmus **RSA**.

2.3.1. Algoritmus RSA

Jedná se o nepokořený algoritmus založený na složitosti faktorizace velkých prvočísel, který je pojmenovaný podle svých tvůrců – Rivestu, Shamirovi a Alemanovi. Algoritmus funguje tak, že nejdříve vygeneruje dvojici klíčů. Toto generování probíhá za pomocí náhodných čísel p a q, která jsou zároveň prvočísly. Tato prvočísla jsou následně násobena a dostáváme číslo n = p . q. Veřejný klíč e je generován jako náhodné číslo, které nemá nic společného se součinitelem (p-1)(q-1). Soukromý klíč vzniká podle vztahu $d = e^{-1} \bmod ((p-1)(q-1))$. Samotné šifrování a dešifrování poté probíhá následně a je jedno, který z klíčů se použije na šifrování, či dešifrování [4]:

šifrování: $ŠT = OT^e \bmod(n)$ ŠT = šifrovaný text

dešifrování: $OT = ŠT^d \bmod(n)$ OT = otevřený text

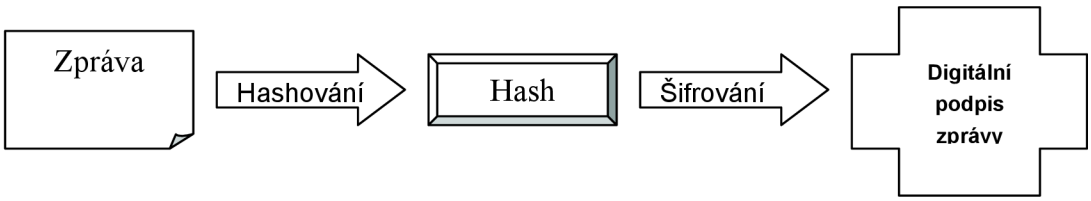
2.4. Elektronická obálka

Šifrování je operací, do které vstupují data určená k zašifrování a šifrovací klíče (popř. inicializační vektory). V případě využití asymetrické kryptografie využívající náročné matematické postupy však může být doba trvání výpočtu dlouhá. Řešením tohoto problému je elektronická obálka. Odesilatel zašifruje zprávu symetrickým klíčem (rychlá operace) a k takto šifrované zprávě přibalí tzv. *Recipient info* obsahující klíč použitý při symetrickém šifrování. Tento krátký klíč je však zašifrován asymetrickým šifrováním. Výsledek je velice rychlý a efektivní. Další výhodou elektronické obálky může být v případě zasílání hromadné zprávy více uživatelům možnost přibalit každému adresátovi jeho veřejný klíč.

2.5. Digitální podpis

Digitální podpis je mechanismus, kterým se zajišťuje ***nepopiratelnost*** dat (pravost dokumentu). Digitální podpis je vytvářen ve dvou krocích [5]:

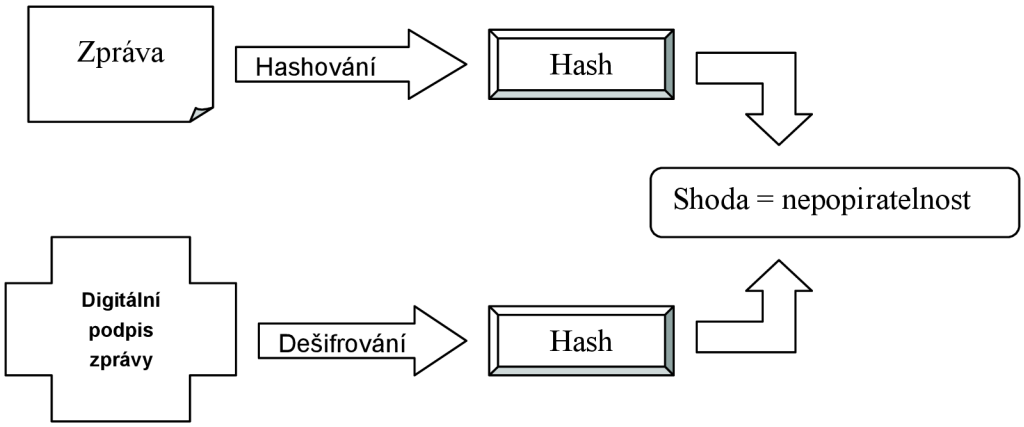
1. **Vypočte se hash (otisk) dokumentu** - Pro použití digitálního podpisu potřebujeme nejprve nějakou známou hashovaní funkci (např. MD5 nebo SHA-1). Známou v tom smyslu, aby všichni adresáti, kteří budou chtít ověřit pravost naší zprávy tuto funkci znali (resp. ji znal program, který ověření provede). Hashovací funkce udělá z naší zprávy tzv. otisk (angl. "message digest") nebo se výsledek také dá nazvat jakýmsi kontrolním součtem zprávy. Tento otisk má vždy stejnou délku bez ohledu na délku vstupní zprávy (128 či 160 bitů).
2. **Výsledný hash se šifruje soukromým klíčem uživatele** – Soukromým klíčem šifrovaný hash ze zprávy se nazývá digitální podpis zprávy.



Obr. 2.1: Schéma vzniku digitálního podpisu

Podpis pak přiložíme k původní zprávě, kterou podepisujeme, a zprávu i s touto přílohou odešleme. Příjemce zprávu otevře a provede její ověření (verifikace)

1. **Vypočte se hash (otisk) dokumentu** - Pomocí stejné hashovací funkce zakóduje její obsah.
2. **Dešifrování digitálního podpisu** - Pomocí veřejného klíče odesílatele příjemce dále rozkóduje obsah digitálního podpisu. Je-li tento rozkódovaný obsah totožný s hashem přijaté zprávy, je identita odesílatele potvrzena, jelikož nikdo jiný, než vlastník soukromého klíče nemohl digitální podpis s touto vlastností vytvořit.



Obr. 2.2: Schéma verifikace zprávy pomocí digitálního podpisu

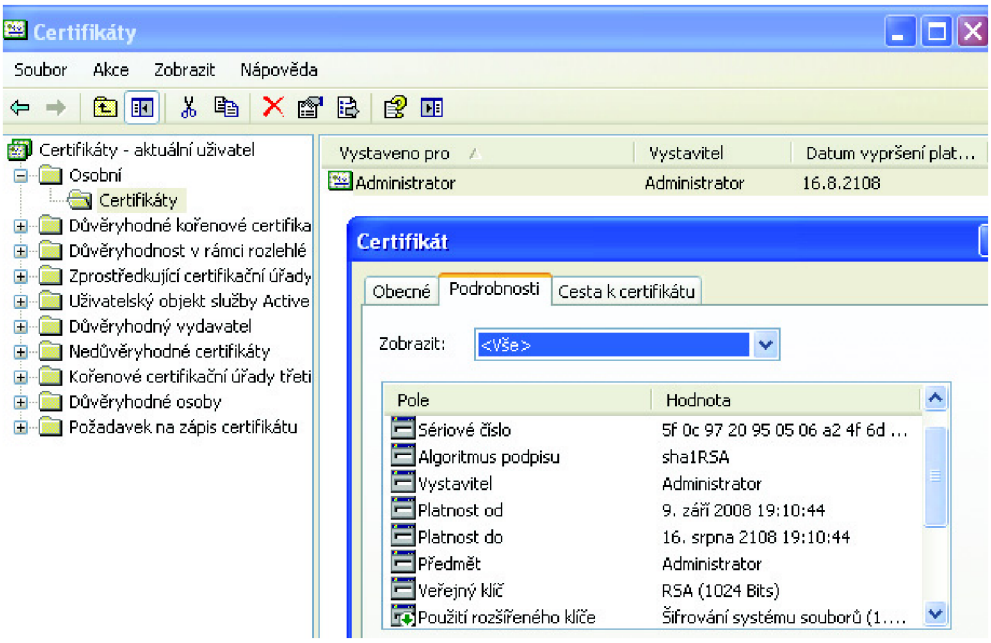
Hashovací funkce se používá z důvodu, aby přikládaný digitální podpis nebyl příliš velký. Pokud by odesílatel svým soukromým klíček kódoval celou zprávu, digitální podpis by byl minimálně jednou tak velký a tedy finální zpráva s podpisem by zvětšila objem minimálně na dvojnásobek.

Digitální podpis provádí důkaz pravosti na základě vlastnictví soukromého klíče. Soukromý klíč je tedy cenným *aktivem*. Je tedy nutné, abychom si své soukromé klíče dobře střežili pomocí bezpečných prostředků k ukládání těchto druhů aktiv. Neopatrnost ochrany soukromého klíče lze přirovnat k podepisování bianco šeků.

3. CERTIFIKÁT

Řešením problému správy, distribuce a uchování klíčů je využití tzv. **certifikátu veřejného klíče**, zkráceně nazývaného certifikát. Certifikát se často přirovnává k občanskému průkazu či pasu. Zatímco občanský průkaz se vydává v tištěné podobě, certifikát je **digitálně podepsanou** datovou strukturou, jejíž základní **součástí je veřejný klíč** držitele certifikátu. V mnoha zemích se dokonce dnes již vydávají občanské průkazy ve tvaru čipové karty, které v sobě obsahují certifikáty držitele karty. Certifikáty obsahují obvykle veřejný klíč, jméno a další údaje zajišťující jednoznačnou identifikaci subjektu, kterému byl tento certifikát vydán. Běžně také obsahují **datum počátku a ukončení platnosti**, jméno vydávající certifikační autority apod. Vydavatelem certifikátu může být kdokoli disponující příslušnou technologií. Pro vydávání ověřených důvěryhodných certifikátů však musí být vydavatel specializovaným poskytovatelem certifikačních služeb (PCS), častěji označován jako certifikační autorita (CA).

Existuje několik norem definujících strukturu certifikátu (X.509, EDI, WAP, apod.). V prostředí Internetu se využívá **norma X.509** popsaná v doporučení RFC-3280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Toto doporučení je odvozeno od ITU X.509. Původní verze standard X.509 vydaná v roce 1988 označená jako verze 1, se dnes používá v pozměněné formě verze 3.



Obr. 3.1: Ukázka certifikátu standardu X.509 v prostředí MS Windows XP

Tab. 3.1: Příklad klientského certifikátu

Certificate:

Data:

Version: 3 (0x2)

Serial Number: 1 (0x1)

Signature Algorithm: md5WithRSAEncryption

Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
OU=Certification Services Division,
CN=Thawte Server CA/emailAddress=server-certs@thawte.com

Validity

Not Before: Aug 1 00:00:00 1996 GMT

Not After : Dec 31 23:59:59 2020 GMT

Subject: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
OU=Certification Services Division,
CN=Thawte Server CA/emailAddress=server-certs@thawte.com

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (1024 bit)

Modulus (1024 bit):

00:d3:a4:50:6e:c8:ff:56:6b:e6:cf:5d:b6:ea:0c:
68:75:47:a2:aa:c2:da:84:25:fc:a8:f4:47:51:da:
85:b5:20:74:94:86:1e:0f:75:c9:e9:08:61:f5:06:
6d:30:6e:15:19:02:e9:52:c0:62:db:4d:99:9e:e2:

```
            6a:0c:44:38:cd:fe:be:e3:64:09:70:c5:fe:b1:6b:
            5d:c3:58:e1:c0:e4:d9:5b:b0:b8:dc:b4:7b:df:36:
            3a:c2:b5:66:22:12:d6:87:0d
        Exponent: 65537 (0x10001)
    X509v3 extensions:
        X509v3 Basic Constraints: critical
            CA:TRUE
Signature Algorithm: md5WithRSAEncryption
    07:fa:4c:69:5c:fb:95:cc:46:ee:85:83:4d:21:30:8e:ca:d9:
    a8:6f:49:1a:e6:da:51:e3:60:70:6c:84:61:11:a1:1a:c8:48:
    3e:59:43:7d:4f:95:3d:a1:8b:b7:0b:62:98:7a:75:8a:dd:88:
    4e:4e:9e:40:db:a8:cc:32:74:b9:6f:0d:c6:e3:b3:44:0b:d9:
    8a:6f:9a:29:9b:99:18:28:3b:d1:e3:40:28:9a:5a:3c:d5:b5:
    e7:20:1b:8b:ca:a4:ab:8d:e9:51:d9:e2:4c:2c:59:a9:da:b9:
    b2:75:1b:f6:42:f2:ef:c7:f2:18:f9:89:bc:a3:ff:8a:23:2e:
    70:47
```

Význam jednotlivých hodnot certifikátu [1]:

Version (verze)– určuje podle které normy X.509 byl příslušný certifikát vydán.

Serial Number (Sériové číslo) – musí být jednoznačné v rámci jedné CA. Spolu s *Issuer* tvoří jednoznačný identifikátor certifikátu.

Signature Algorithm (Algoritmus podpisu) – Algoritmy, které CA použila k podpisu certifikátu. V Tab. 3.1 je to hashovací funkce MD5 a asymetrický algoritmus RSA.

Issuer (vystavitel) + **Subject** (předmět) – obsahují položky identifikátorů objektů označovaný DN (Distinguished name). Mezi tyto objekty patří např. Country (C), Organization (O), Common name (CN), atd.

Validity (platnost) – každý certifikát je časově omezen (obvykle jeden rok).

Subject Public Key (veřejný klíč) – obsahuje veřejný klíč vlastníka. Je složen ze dvou základních informací - vlastní hodnoty klíče a algoritmu, kterým byl klíč tvořen.

3.1. Klientské certifikáty

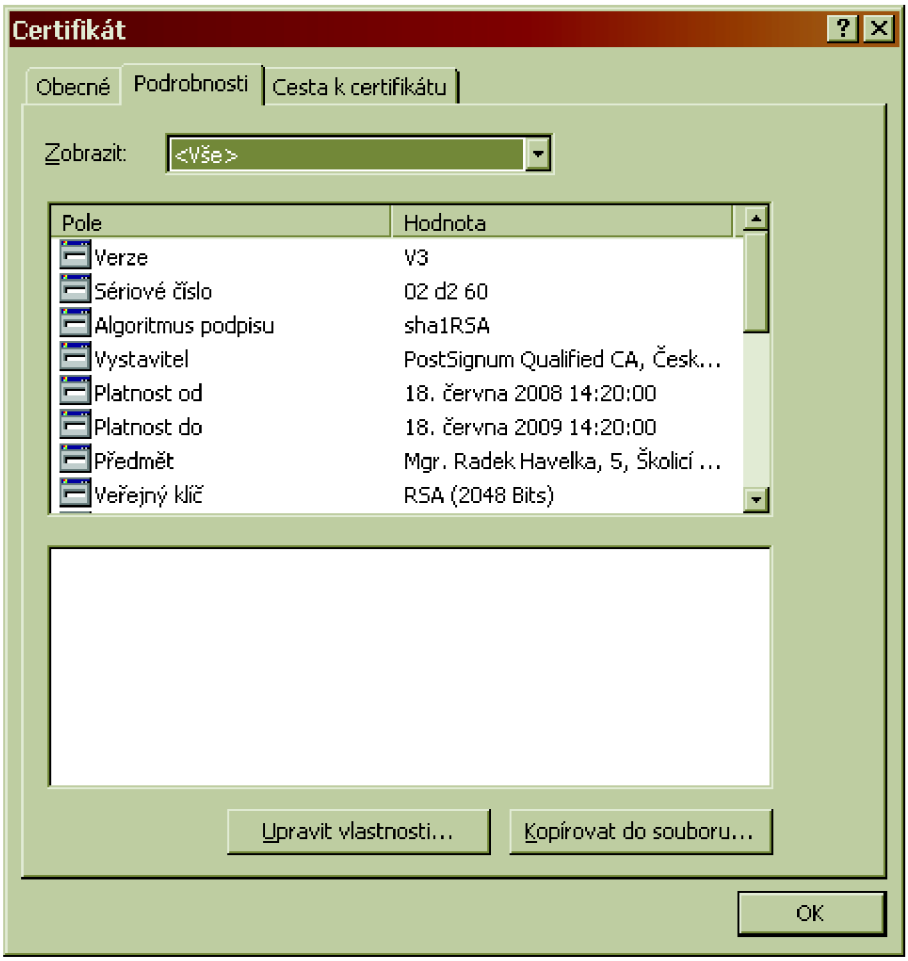
Většina certifikačních autorit vydává více různých typů certifikátů podle různých potřeb svých klientů. Nejrozšířenějším typem pro bezpečnou komunikaci po Internetu je **komerční certifikát**. Toto označení se používá pro certifikáty, které nejsou spojeny se zákonem o elektronickém podpisu. To, že komerční certifikáty nejsou předmětem zákonné úpravy, znamená, že jejich použití v praxi není nijak omezeno a je jen na dohodě komunikujících stran k jakému účelu budou použity. Nevýhodou je, že úroveň CA není nikým kontrolována a to je třeba brát v úvahu při výběru důvěryhodné CA. Komerční certifikáty se nejčastěji používají v oblasti komerčních aplikací a elektronickém bankovnictví (SSL certifikáty, se kterými je nejčastěji zabezpečována komunikace mezi uživatelem a serverem).

3.2. Kvalifikované certifikáty

Certifikáty vydávané v souladu se zákonem o elektronickém podpisu se nazývají **kvalifikované certifikáty (QCA)**. Tyto certifikáty jsou určeny výhradně pro **elektronický podpis**. Toto omezení použití kvalifikovaných certifikátů je dáno českou legislativou – Zákonem o elektronickém podpisu (Zákon č. 227/2000 Sb., o elektronickém podpisu a o změně některých dalších zákonů ve znění zákona č. 226/2002 Sb. – více v [15]). Tyto zákony vychází z jednotné legislativy EU. Cílem zákona je **nahradit rukou psaný podpis** elektronickým podpisem.

Kvalifikovaný certifikát obsahuje identifikaci držitele certifikátu založenou na oficiální identifikaci svého držitele. Certifikační autorita vždy zná konkrétné osobu, které certifikát vydala. Bezpečnost a důvěryhodnost CA vydávajících kvalifikované certifikáty je do jisté míry standardizována a kontrolována příslušnými úřady. V ČR se momentálně nachází tyto akreditované CA:

- První certifikační autorita, a.s. (I.CA, komerční i kvalifikovaný) - http://www.ica.cz/
- Česká pošta, s.p. (PostSignum, komerční i kvalifikovaný) - http://qca.postsignum.cz/
- eIdentity, a.s. (ACAeID, komerční i kvalifikovaný) - https://www.eidentity.cz/app



Obr. 3.2: Ukázka kvalifikovaného certifikátu od CA PostSignum

Mezi kvalifikované certifikáty patří i tzv. *kvalifikované systémové certifikáty*. Jedná se v podstatě o serverové certifikáty splňující podmínky zákona o elektronickém podpisu. Jejich použití je především spojeno s elektronickými podatelnami. O použití kvalifikovaného systémového certifikátu se spíše hovoří jako o elektronické značce.

3.3. Příklad použití kvalifikovaného certifikátu

Při zastupování firmy danou osobou se využívá tzv. **kvalifikovaný zaměstnanecký certifikát**. Jde o formu zaručeného elektronického podpisu, kdy je certifikační autoritou ověřována kromě totožnosti také vazba fyzické osoby na jejího zaměstnavatele. Obsahem kvalifikovaného certifikátu je potom jednak jméno a příjmení žadatele a zároveň název příslušné organizace. Tento certifikát umožňuje ověřit, že datovou zprávu podepsala osoba uvedená na tomto kvalifikovaném certifikátu ve smyslu zákona 227/2000 Sb. O elektronickém podpisu.

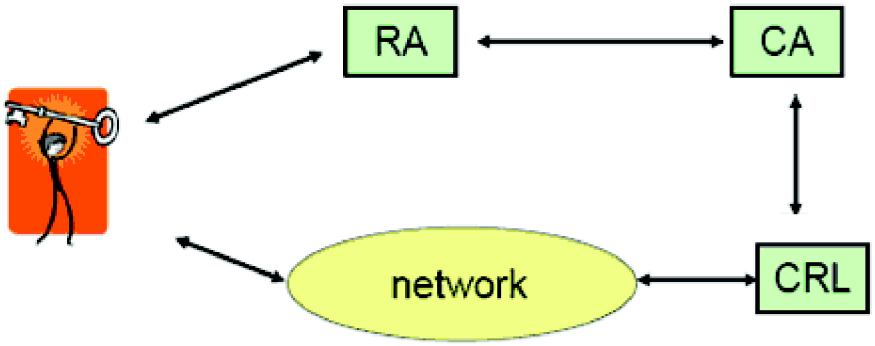
Právě takového kvalifikovaného zaměstnaneckého certifikátu využívá např. Český Podací Ověřovací Informační Národní Terminál (Czech POINT) – jedná se o projekt, který by měl zredukovat přílišnou byrokracii ve vztahu občan – veřejná správa. Dříve musel občan navštívit několik úřadů k vyřízení jednoho problému. Czech POINT slouží jako asistované místo výkonu veřejné správy, umožňující komunikaci se státem prostřednictvím jednoho místa tak, aby „obíhala data ne občan".

Jelikož tento systém může poskytovat i *velice citlivá osobní data* (např. Rejstřík trestů, apod.) je nutné maximálně zajistit bezpečnost pro přístup do tohoto systému. Momentálně je tato bezpečnost řešena povolením přístupu pouze pro vyškolené zaměstnance daných obecních a městských úřadů, vlastnících USB token **SafeNet iKey 4000** (o USB tokenech podrobněji v kapitole 5.2.5-USB token). Do systému se tedy dostane jen držitel tokenu, který je po vsunutí zařízení do USB dále autentizován prokázáním znalostí kódu PIN. Jeho certifikát také musí být administrátorem úřadu **spárován (schválen)** s daným úřadem v systému Czech POINT administrátorem systému. V případě potřeby tedy také může administrátor zablokovat přístup danému tokenu (uživateli).

Obr. 3.3: Administrace přístupu do portálu Czech POINT na základě QCA

4. STRUKTURA SYSTÉMU PKI

Hlavními subjekty infrastruktury veřejných klíčů jsou **certifikační autorita**, **registrační autorita** a **seznam odvolaných certifikátů**. Rolí certifikační autority v rámci bezpečné elektronické komunikace je být třetí nezávislou stranou. Nezávislost CA na komunikačních stranách jí umožňuje vystupovat v roli arbitra. Základními funkcemi CA je tedy vydávání certifikátů a seznamu zneplatněných certifikátů (CRL - Certificate Revocation List). Z toho plyne, že jádrem PKI je certifikační autorita CA. Jí mohou být podřízeny další CA a tzv. registrační autority (RA). Úkolem RA je fyzické ověření údajů žadatele o certifikát.



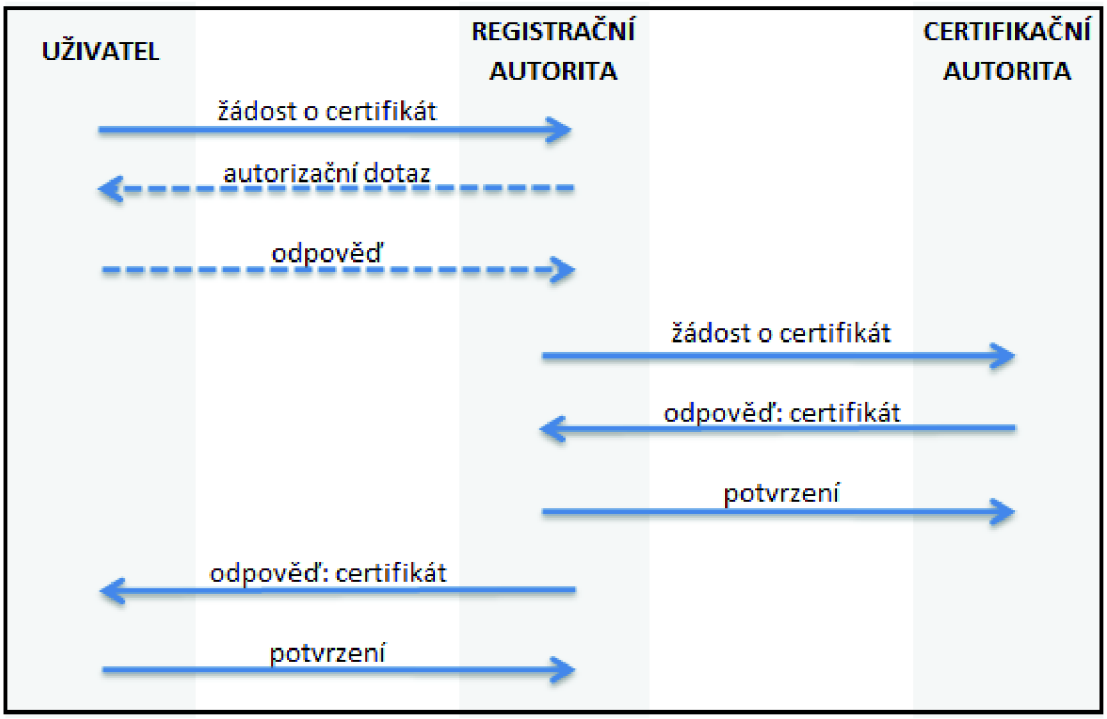
Obr. 4.1: Struktura PKI [2]

4.1. Certifikační autorita (CA)

Certifikační autorita je subjekt, který vydává digitální certifikáty. Na základě principu přenosu důvěry tak lze za předpokladu (ověřené) důvěryhodnosti certifikační autority předpokládat i důvěryhodnost jimi vydaného (podepsaného) certifikátu. Všechny certifikáty jsou podepisovány soukromým klíčem CA. Tento klíč je tedy největším aktivem CA a je tedy nutné jej odpovídajícím způsobem chránit. Jako ochrana se proto používají tzv. HSM (viz. kap. 5 - Prostředky bezpečného ukládání PKI aktiv).

4.2. Registrační autorita (RA)

Úkolem RA je fyzické ověření údajů žadatele o certifikát. Jsou často realizovány podobně jako bankovní přepážky. Mohou být však realizovány i jako servery a žadatel s nimi komunikuje elektronicky. Na RA se dostavují žadatelé o certifikáty se svými žádostmi, kde RA může ověřit na základě např. občanského průkazu jejich totožnost. RA následně zprostředkovává vydání certifikátu a následně i jeho předání žadateli.



Obr. 4.2: Dialog vydávání certifikátu

4.3. Seznam odvolaných certifikátů (CRL)

Seznam zneplatněných certifikátů (Certificate Revocation List) udržuje CA a zapisuje do něj certifikáty, které byly zneplatněny ještě dříve, než je uvedeno na certifikátu (například z důvodu kompromitace tajného klíče uživatele). CRL by měl být veřejně dostupný buďto skrze webové rozhraní, popřípadě skrze protokol LDAP.

Tento seznam udržuje CA a zapisuje do něj certifikáty, které byly zneplatněny ještě dříve, než je uvedeno na certifikátu (např. z důvodu kompromitace tajného klíče uživatele).

4.4. Další volitelné subjekty systému PKI

Atributová autorita (AA) – přiděluje privilegia uživatelům vydáváním tzv. atributových certifikátů

Repositář – zpravidla plní dvě funkce. Tvoří privátní databázi pro zálohování platných klíčů a pro archivování klíčů, kterým uplynula doba platnosti. Druhou fcí je LDAP Server (*Lightweight Directory Access Protocol Server*), který má na starosti publikování seznamů certifikátů (např. CLR).

Vydavatel certifikačních politik (PMA) – definuje technické a procedurální požadavky na vytváření a ověřování digitálních certifikátů pro potřeby nějaké třídy aplikací, resp. Skupiny uživatelů.

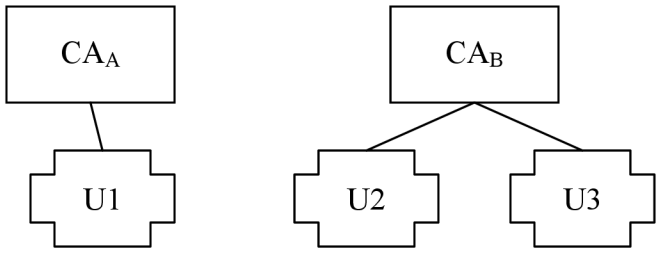
Schvalovatel certifikačních politik (PAA) – provádí hodnocení politik a vypracovává jejich „hodnotící zprávy".

Tyto funkce zpravidla obsahuje základní topologie nazývána jako tradiční **obecný model** [7].

4.5. Topologie propojení PKI

4.5.1. Obecný model PKI

Základní PKI architektura se skládá z **jediné CA**, která zajišťuje všechny související služby (speciálně vydávání certifikátů, publikování CRL, apod.) pro všechny uživatele tohoto PKI. Všichni uživatelé této architektury důvěřují této jediné CA. Každá certifikační cesta začíná veřejným klíčem tohoto poskytovatele – CA. Nevýhodou je, že při velkém počtu uživatelů může být problém s dostupností CRL, popřípadě může být problém se zajištěním klientské podpory jednotlivých aplikací, které PKI využívají.



Obr. 4.3: Obecný model PKI (izolované CA)

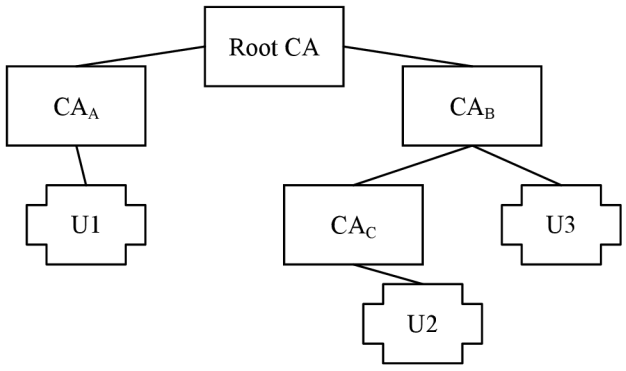
Podívejme se na situaci na Obr. 4.3, kde jsou znázorněny dvě izolované PKI. Uvažujme situaci, kdy spolu komunikují (pod tímto pojmem zde a dále budeme chápat vytváření a ověřování elektronických podpisů a dále i šifrování resp. autentizaci) uživatelé těchto izolovaných PKI. Jako příklad uvažujme situaci, kdy uživatel U1 **potřebuje ověřit certifikát** uživatele U3.

Při ověřování podpisu uživatel U1 zjistí, že certifikát U3 vydala CA_B a tento certifikát také podepsala. Ke správnému ověření tedy potřebuje U1 ověřit podpis i CA_B. Tato PKI však nemá s jeho CA_A žádný vztah, z toho důvodu se bude jevit U3 uživateli U1 jako nedůvěryhodný. Řešením může být, že se uživatel U1 **rozhodne důvěřovat certifikátům** vydaným CA_B a nainstaluje si její certifikát do svého úložiště důvěryhodných CA, popřípadě se rozhodne důvěřovat pouze konkrétnímu certifikátu - tedy U3. S tímto postupem je spojena celá řada problémů bezpečnostního a procesního charakteru.

Existuje proto možnost, kdy PKI umožňuje jedné CA certifikovat jinou CA. To znamená, že certifikující CA říká uživateli, který důvěřuje jí vydaným certifikátům, že může věřit certifikátům vydaným certifikovanou CA. Tento systém se nazývá **Hierarchická struktura.**

4.5.2. Hierarchická struktura

Problém při ověřování certifikátů vydaných různými CA by neměl být řešen na úrovni uživatelů (viz. předchozí případ), ale na úrovni správců CA. Existuje řada způsobů, jak řešit vztah důvěry mezi jednotlivými CA. Nejznámější je **budování vztahu nadřízenosti** a podřízenosti jednotlivých autorit. V tomto případě se PKI se konstruuje tak, že existuje jedna výchozí autorita, která se nazývá **kořenovou certifikační autoritou** (root CA). Tato autorita (obecně) nevydává certifikáty koncovým uživatelům, ale pouze jiným certifikačním autoritám („tzv. podepisuje veřejný klíč CA“). Takovéto certifikační autority se pak nazývají podřízené CA. Kořenová autorita vydá certifikát sama sobě, který si i sama sobě podepíše („*self signed certificate*“).

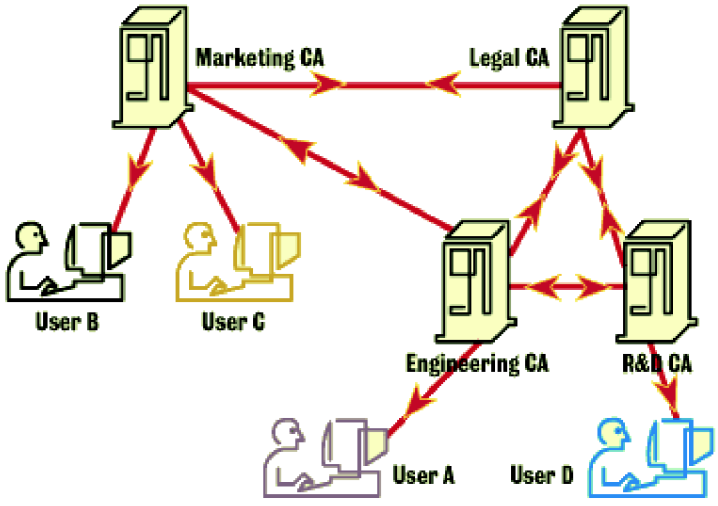


Obr. 4.4: Jednoduchá hierarchická struktura

Na Obr. 4.4 se nachází jednoduché PKI, kde však CA_B a CA_B již nejsou izolované, ale mají společnou nadřízenou certifikační kořenovou autoritu (*Root CA*). Uživatelé tohoto PKI **vždy důvěřují vydavateli** svého vlastního certifikátu a v jeho nadřízenou certifikační autoritu. Fáze ověřování důvěryhodnosti od uživatelského certifikátu, přes CA, která certifikát vydala, až po kořenovou autoritu se nazývá **certifikační cesta**. Použitelnost hierarchické PKI je však silně omezené, protože mnohé organizace hierarchicky organizované nejsou.

4.5.3. Síťová struktura (mesh)

V síťové PKI se nezávislé CA certifikují vzájemně podle nějaké zavedené obecné sítě vztahů důvěry mezi nimi. Tato struktura vzniká jednak tehdy, kdy není možné se dohodnout na vztahu podřízenosti a nadřízenosti jednotlivých CA nebo není možné takovýto vztah budovat. Typické pro tuto strukturu je, že jednotlivé certifikační autority si jsou navzájem dvoustranně důvěryhodné. Není zde žádná dominantní (kořenová) certifikační autorita. Hlavní výhodou této struktury je, že lze jednoduše přidat celou novou PKI strukturu. Klient zná veřejný klíč jemu blízké CA (obvyklé té lokální, která mu vydala certifikát) a předložené certifikáty ověřuje po **certifikační cestě**, která vede zpět do jemu blízké CA. Pokud taková cesta neexistuje, pak předloženému certifikátu nedůvěřuje. Při vytváření systému smluv o vzájemném uznávání certifikátů se může ukázat zásadním problém, že chybí koordinační centrum – vůdčí autorita, která by jednotlivé smluvní vztahy ošetřovala, zajišťovala a udržovala.



Obr. 4.5: Ukázka síťové (mesh) topologie [12]

4.5.4. Kombinovaná struktura

Kompromisním řešením architektury PKI je kombinovaná PKI, obvykle tvořená několika vhodně propojenými hierarchickými PKI. Jejich kořenové CA náleží do vhodné síťové PKI. V kombinované PKI lze přitom zavádět vztahy vzájemné důvěry i mezi některými (nekořenovými) CA přímo. Centrální kořenová autorita je pochopitelně velmi atraktivním cílem pro útoky a navíc musí její veřejný klíč všichni uživatelé PKI znát. Vhodně nastavené „příčky“ v topologii kombinované PKI mohou délky certifikačních cest podstatně zkrátit. Příkladem kombinované struktury je například tzv. bridge CA.

Bridge CA organizuje systém vzájemné důvěry mezi dalšími strukturami PKI (hierarchickými, mash) nebo přímo mezi jednotlivými dříve izolovanými CA. Důvěryhodným způsobem udržuje a rozesílá přehled o stavu všech CA v bridge. Výhodou je **veliká flexibilita**, snadno se přidává další struktura PKI, při kompromitaci některé CA nejsou ohroženy ostatní CA nebo celé struktury PKI v bridge, snižuje se délka nutné certifikační cesty atd. V roce 2002 byl spuštěn projekt na vytvoření European Bridge/Gateway CA. Tento projekt je zaměřený na problematiku uznávání a důvěryhodnosti elektronických osvědčení vydávaných různými certifikačními orgány (CA) v rámci EU. Tato aktivita se jeví jako velmi slibná, například vzhledem k **neexistenci kořenové CA pro celou ČR**.

5. PROSTŘEDKY BEZPEČNÉHO UKLÁDÁNÍ PKI AKTIV

Správa soukromých klíčů je jedním ze základních problémů, se kterými se oblast PKI potýká. Klíče používané v PKI jsou velmi dlouhé řetězce znaků a na rozdíl od hesel si je člověk nemůže zapamatovat. Musí být proto uloženy v nějaké elektronické podobě tak, aby jej mohla přečíst aplikace, kterou uživatel používá. Nejčastěji se dnes klíče ukládají na lokální disk počítače, buď ve formě samostatného souboru, nebo ve specializovaném úložišti, které poskytuje aplikace, příp. operační systém.

Technik jak získat příslušná data z lokálního disku je celá řada, od použití počítačových virů, přes zneužití různých chyb v aplikacích běžících na daném počítači, až k technikám sociálního inženýrství, které zřejmě právě zažívají renesanci v podobě tzv. phishingu.

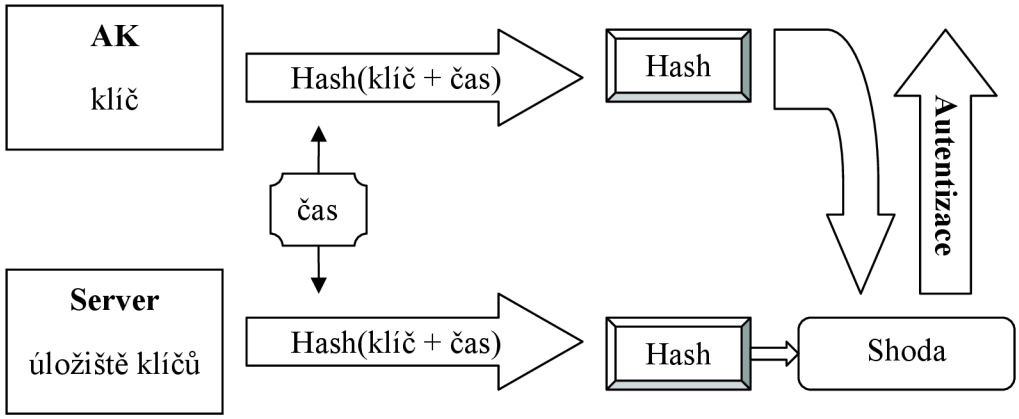
Další problematickou oblastí správy klíčů je fakt, že **zabezpečení souboru s klíčem** je z velké části v rukách samotného uživatele. Navíc v oblasti PKI neexistují mechanismy, které by spolehlivě zajistily, že soubor s klíčem je patřičně ochráněn, tj. že použité heslo je dostatečně silné, aby odolalo běžným útokům, že jsou správně nastavena přístupová práva k souboru s klíčem apod.

Výrazné zvýšení bezpečnosti by přineslo uložení soukromých klíčů na bezpečnější místo tak, aby neležely přímo na disku stroje, ale místo toho byly na nějakém jiném médiu, které se použije pouze v případě potřeby. Nejpoužívanějším druhem těchto médií jsou autentizační kalkulátory a hardwarové klíče (čipové karty, USB tokeny, HSM).

5.1. Autentizační kalkulátory

Autentizační kalkulátor (AK) je zařízení často podobné kapesní kalkulačce, které na základě vstupních údajů vygeneruje jedinečný **jednorázový autentizační nebo autorizační kód**. Autentizační kalkulátory zpravidla umí některý z kvalitních algoritmů pro výpočet otisku. Do AK se ukládá sdílené tajemství (mezi serverem a AK), které je po

zadání dat (nejčastěji čas, či počet vygenerovaných hesel) zřetězeno. Z tohoto řetězce je dále vytvořen otisk, který slouží jako jednorázové heslo. Vzhledem k tomu, že toto heslo bývá následně uživatelem opisováno, nepoužívají se zde standardní hashovací funkce (MD5, SHA1) jelikož by uživatel musel zadávat příliš dlouhé řetězce a docházelo by k překlepům apod. Namísto toho využívají výrobci AK **vlastní vytvořené** a patentované **hashovací funkce** jejichž algoritmus není veřejně znám.



Obr. 5.1: Schéma průběhu autentizace pomocí autentizačního kalkulátoru

5.2. Hardwarové klíče

Hardwarovým klíčem se rozumí technické zařízení, které poskytuje bezpečnostní funkce spojené s ukládáním soukromých klíčů, tajných klíčů, sdílených tajemství a jiných aktiv držitele hardwarového klíče. Na rozdíl od autentizačních kalkulátorů je hardwarový klíč **přímo propojen s počítačem** pomocí daného rozhraní (sériový port, USB, PCI, …). Hardwarové klíče jsou zpravidla uzpůsobeny tak, že jej **soukromý klíč nikdy neopouští**. Hardwarový klíč tedy zajišťuje nejčastěji tyto funkce:

- Generuje dvojici veřejný-soukromý klíč,
- generuje podklady k žádosti o certifikát,
- vydaný certifikát lze uložit opět do hardwarového klíče,
- hardwarový klíč provádí šifrování požadovaných dat pomocí soukromého klíče uloženého v hardwarovém klíči.

K hardwarovým klíčům patří čipové karty, USB tokeny a hardware security moduly (HSM).

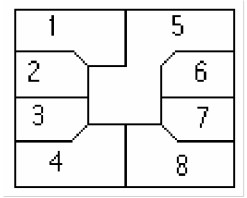
5.2.1. Čipové karty

Čipové karty prošly dlouhodobým vývojem a existuje mnoho různých druhů. Pod standardem ISO jsou označovány jako IIC (Integrated Circuit Card). Jejich základní rozdělení je dělí se na paměťové a mikroprocesorové (**Smart card**) a dále dle způsobu komunikace s rozhraním na kontaktní a bezkontaktní.

Čipová karta je zpravidla plastiková karta, která má ve svém těle vložen čip obsahují jak chráněný prostor, do kterého lze uložit soukromý klíč s certifikátem, tak i samostatný procesor, který je schopen s těmito klíči pracovat a provádět s nimi základní kryptografické operace.

Bezkontaktní, neboli radio-frekvenční karty, komunikují díky integrované anténě prostřednictvím elektromagnetických vln a není potřeba je zasouvat do čtečky. Z tohoto důvodu jsou vhodné pro masovou identifikaci fyzického přístupu (vstup do budov apod.).

V prostředí PKI se však spíše vyžívají kontaktní čipové karty, které mají dle specifikace ISO 7816-2 osm kontaktů. Nejjednodušší čipové karty jsou osazené pouze paměťovými registry (telefonní karty). **Procesorové karty** mají kromě paměti i jednočipový procesor schopný vykonávat příkazy.



C1	Vcc = 5V	C5	Zem
C2	Reset	C6	Vpp (EEPROM)
C3	Hodiny	C7	I/O
C4	Rezerva	C8	Rezerva

Obr. 5.2: Kontakty čipové karty dle ISO 7816-2

Speciální podskupinou procesorových čipových karet jsou **PKI čipové karty**. Jsou to procesorové čipové karty schopné provádět příkazy nejenom asymetrické kryptografie, ale i symetrické kryptografie a často i výpočet otisku (tzv. hash). PKI čipové karty zpravidla mají kryptografické koprocesory pro urychlení kryptografických operací. PKI čipové karty mohou být nejenom kontaktní, ale i bezkontaktní.

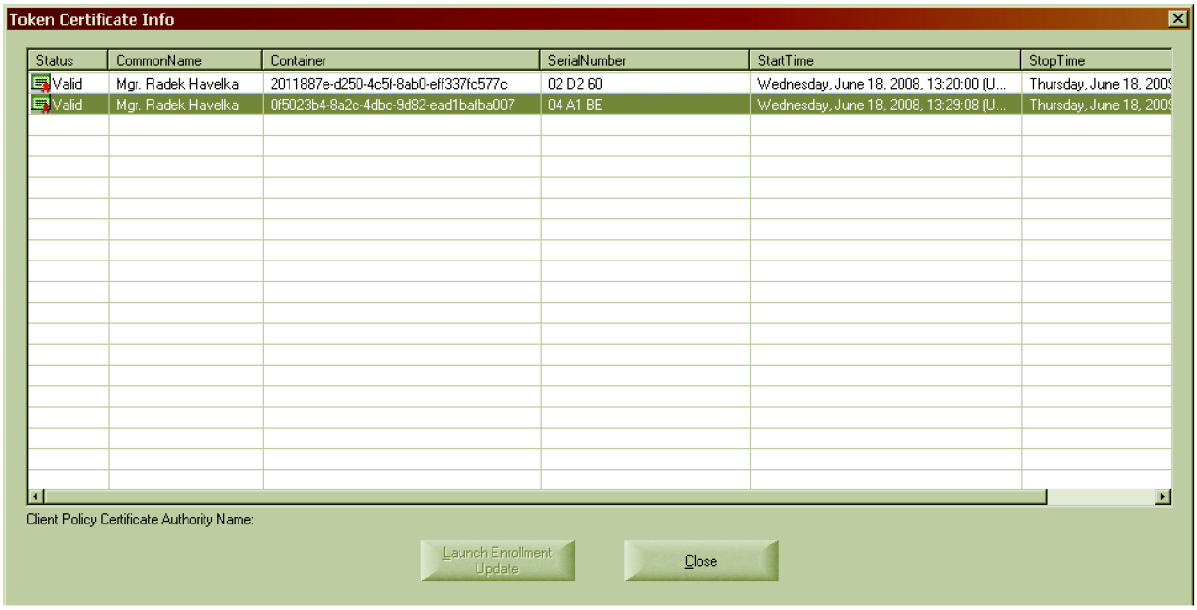
Z důvodu ochrany před kompromitací soukromého klíče probíhá často generování dvojice veřejný/soukromý klíč přímo samotnou PKI čipovou kartou a soukromý klíč je okamžitě uložen uvnitř paměti karty.

Karta je k počítači připojena pomocí čtečky zapojené přes USB nebo sériový port, pomocí níž komunikují aplikace s kartou. Aplikace tak nepoužívají přímo soukromý klíč, ale předávají kartě data, která jsou **zpracována procesorem přímo na tokenu**. Výsledek je

vrácen zpět aplikaci. Klíč tak nikdy neopustí kartu a není jej možné nijak zkopírovat. Přístup ke kartě je autentizován, tj. aplikace se musí procesoru na kartě nejprve prokázat znalostí příslušného PINu, který zadá uživatel. Je tak zabráněno zneužití informací z karty v případě její ztráty. Většina karet je konstruována tak, že se po zadání určitého počtu chybných PINů zablokuje a jedinou možností jak ji zprovoznit je její nová inicializace, která však nevratně smaže všechny informace na kartě. Vedle čipových karet s čtečkami také existují čipové tokeny připojitelné do USB, které kombinují funkcionalitu karty a čtečky v jednom kusu hardware.

5.2.2. Struktura dat uložených v paměti čipových karet

Data uložená v paměti karty tvoří souborovou strukturu tvořenou kořenovým adresářem **MF (Master File)**, ve kterém mohou být podadresáře **DF (Dedicated File)** nebo datové soubory **EF (Elementary File)**. DF mohou obsahovat již jen datové soubory (EF), nikoliv další podadresáře. Na rozdíl od běžné adresářové struktury standardních souborových systémů (FAT, NTFS, apod.) však neexistuje obdoba příkazu DIR a uživatel tedy nemá možnost procházet strukturou paměti. V případě PKI čipů jsou v některých EF uloženy soukromé klíče, v jiných veřejné klíče, případně certifikáty. Držitele karty však nezajímá, ve kterém EF hledat konkrétní soukromý klíč, ale bude chtít mít k dispozici dvojici soukromý/veřejný klíč, případně celý certifikát. Proto PKI čipy mají kromě fyzické vrstvy (MF, DF, EF) ještě strukturu logickou tvořenou tzv. kontejnery. Příslušející soukromé klíče, veřejné klíče, včetně případných certifikátů tvoří jeden konkrétní kontejner. Tyto kontejnery lze většinou zobrazit pomocí utilit dodávaných výrobcem daného zařízení.



Obr. 5.3: Token Certificate Info z utility dodávané k USB SafeNet iKey 4000

Většina současných dodavatelů kryptografických aplikací na čipových kartách a USB tokenech pracuje se svým (proprietárním) formátem aplikace. Takovou kartu pak nelze použít pro jinou aplikaci, než poskytuje vydavatel karty. Z těchto důvodů byl zaveden mezinárodní standard v ČR zaveden jako **ČSN ISO/IEC 7816-15 (PKCS#15)** definující standardní kryptografickou aplikaci **CIA (Cryptography Information Application).** Standard specifikuje, jak mají být na čipové kartě uloženy kryptografické údaje, definuje strukturu aplikací v kryptografických tokenech (čipové karty, kryptografické tokeny, sw. tokeny), identifikátory souborů a jejich obsah.

PKCS#15 definuje:

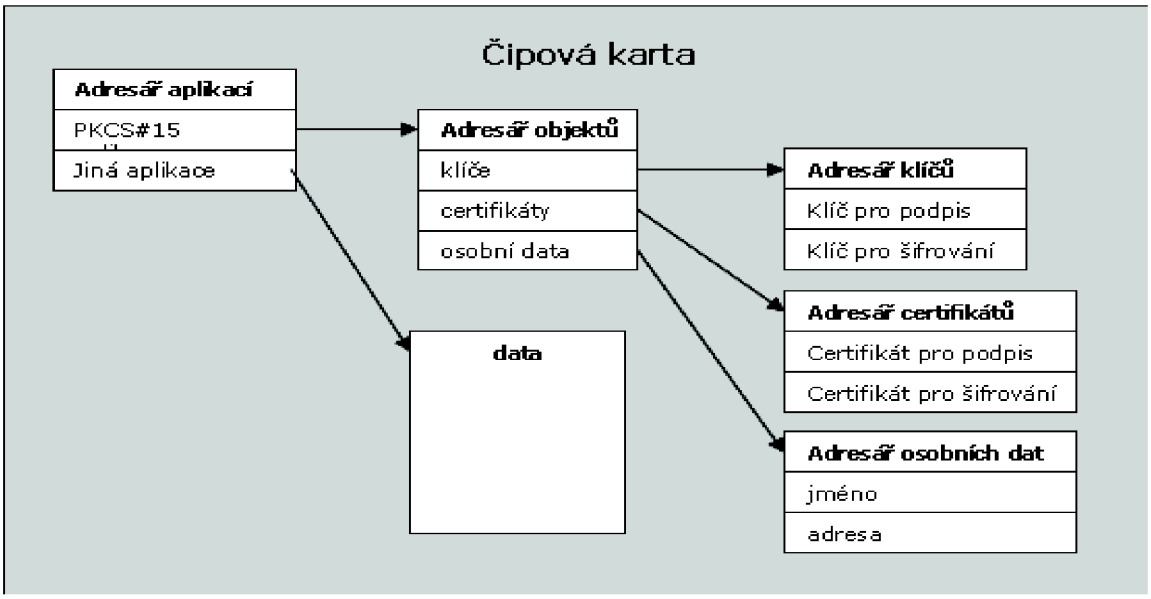
⇒ klíče - symetrický klíč, soukromý klíč, veřejný klíč, …

⇒ certifikáty - X.509, ostatní

⇒ autentizační data - PIN, biometrické údaje, …

⇒ ostatní data (specifická pro aplikace)

PKCS#15 dále dělí objekty na kartě na soukromé a veřejné, přístup k soukromým objektům je chráněn nějakou autentizační procedurou.



Obr. 5.4: Struktura ISO/IEC 7816-15/PKCS#15

5.2.3. Rozhraní pro komunikaci mezi systémem a aplikací

Kryptografický čip zajišťuje pomocí vlastního operačního systému komunikaci s nadřazeným systémem, autentizaci a volání jednotlivých kryptografických operací. Komunikace mezi nadřazeným systémem a danou aplikací pak probíhá pomocí tzv.

middleware vrstvy (vrstvy pracující mezi tokenem a aplikací). Nejpoužívanějšími standardy middleware vrstev jsou [9]:

- **PKCS#11 -** Nejpoužívanější rozhraní nezávislé na systému. Využívají jej především nativní utility karty, aplikace z rodiny Mozilla, Entrust a další. Toto rozhraní je též používánu v prostředí Linuxu.
- **MS CAPI** - Microsoft crypto API. Token je registrován jako další CSP (crypto service provider). Zpřístupňuje operace tokenu v nativních Windows aplikacích jako např.: IExplorer, MS Outlook, MS Office, OpenVPN for Win.
- **Java API -** Zpřístupňuje operace tokenu přímo pro JAVA prostředí. Většinou je JRE (JAVA runtime environment) schopno použít volání systému a zprostředkovat přístup k certifikátům prostřednictvím MS CAPI.

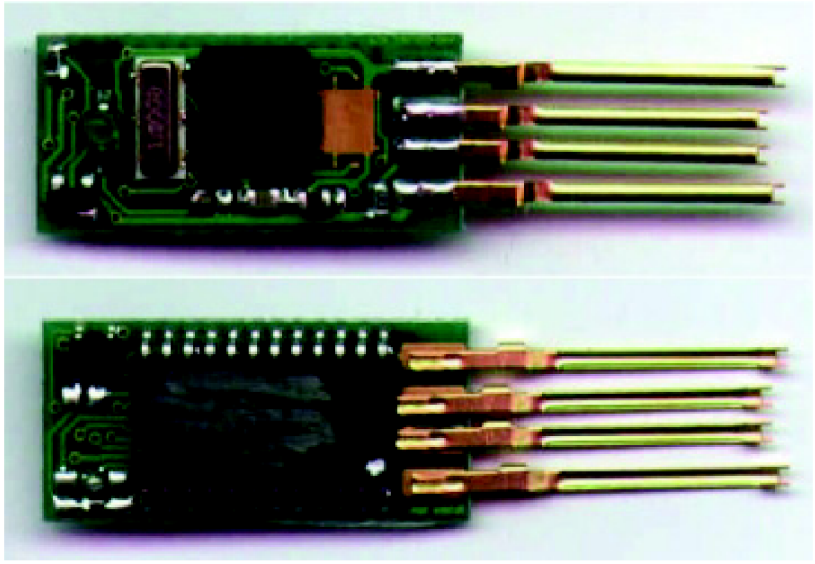
Pro každý typ tokenu výrobce dodává příslušné knihovny. Přímo v knihovnách je dané rozložení jednotlivých datových struktur na kartě (tzv. schéma karty). Proto je nutné použít pro daný typ tokenu příslušnou knihovnu.

5.2.4. Příklad využití PKI čipových karet

PKI čipové karty se často využívají ve velkých společnostech např. pro autentizaci uživatele do operačního systému či pro přístup do firemní sítě pomocí VPN. V těchto společnostech je zpravidla **databáze uživatelů propojena přímo s certifikační autoritou** dané společnosti (například pomocí active directory). Ve velkých korporacích by autentizace pomocí hesla měla být v dnešní době již spíše přežitek. Uživatelé zadávající své uživatelské jméno a heslo, jsou noční můrou každého správce sítě, který dbá na bezpečnost. Od verze Windows 2000 umožňují operační systémy společnosti Microsoft autentizaci uživatele pomocí PKI čipových karet (nazývaných Smart Card). Přihlášení uživatele do systému tak probíhá pouhým zasunutím jeho vlastní PKI karty do čtečky karet. Zpravidla se nastavuje dodatečná autentizace pomocí znalosti PIN kódu. Stejný postup lze použít i v případě použití USB tokenu (viz.dále). Navíc existují tzv. hybridní a duální PKI čipové karty umožňující komunikovat přes kontaktní i bezkontaktní rozhraní. Ve velkých korporacích bývají standardně využívány systémy pro řízení přístupu osob do objektů/budov/místností popř. docházkový systém právě pomocí čipových karet. Pomocí duální PKI čipové karty lze tedy snadno vyřešit problematiku řízení přístupu po budově i do operačních systémů. Dále lze tímto vyřešit další nešvar mnohých uživatelů. Ti se často vzdálí od svého počítače na delší dobu, aniž by se odhlásili, či alespoň uzamknuli stanici. Lze nastavit, co se stane po vysunutí čipové karty ze slotu. Uživatel odcházející z kanceláře si vezme svoji čipovou kartu potřebnou např. pro otevření dveří a tím se stanice buď automaticky uzamkne, či dojde přímo k odhlášení uživatele.

5.2.5. USB token

USB tokeny se podobají USB flash diskům, ale vnitřní architektura je zcela totožná s dříve popsanými čipovými kartami (například v modelu iKey 2032 je úplně stejný čip jako v čipové kartě Datakey model 330), tj. obsahují vlastní procesor a není možné přistupovat přímo k citlivým datům na tokenu.



Obr. 5.5: Vnitřní uspořádání USB tokenu

Oproti PKI čipovým kartám mají USB tokeny např. následující výhody:

- **Není potřeba čtečka čipových karet, díky čemuž jsou USB tokeny**:

  ⇒ *Mobilnější*: není zapotřebí mít na každém PC připojenou čtečku, ale stačí USB port.

  ⇒ *Levnější*: pro účely PKI stojí čtečka cca 2000 Kč a čipová karta cca 800Kč. Celkem na jednoho uživatele cca 2800Kč jen pokud jde o hardware. USB token pro přihlašování do Windows, VPN, zabezpečení digitálního podpisu, či šifrování dat lze pořídit v ceně kolem 1.000,- Kč.

  ⇒ *Není potřeba instalovat ovladače* pro čtečku a ovladače pro čipovou kartu, stačí jedny ovladače pro USB token => méně softwaru, méně problémů, …

- USB tokeny lze připnout na svazek klíčů uživatelů, což má nespornou výhodu. Uživatelé mají token svázán s osobním majetkem (se svými klíči), na který si dávají daleko větší pozor než na majetek firmy, tj. uživatelé tokeny méně ztrácejí, méně je zapomínají doma, atd.

- Čip je u USB tokenů schovaný ve skořápce, takže je chráněn proti lidskému potu, rozlitému čaji, poškrábání, …

– Čipovou kartu lze snadněji fyzicky zlomit, či jinak poškodit.

– USB tokeny mají LED diodu, která indikuje nejen napájení USB portu, ale může například blikat, pokud dochází k šifrování uvnitř tokenu.

5.2.6. Dostupná komerční řešení tokenů

V současné době na trhu působí dva nejvýznamnější výrobci USB tokenů – SafeNet a Aladin. Nejrozšířenějším na trhu s hardwarovými klíči je dnes společnost SafeNet, nabízející tokeny s označením iKey. Jejich produkty se dají pořídit v běžných obchodech zabývajících se prodejem PC a IT periferií. Naopak společnost Aladin se rozhodla jít cestou prodeje pouze přes vybrané obchodní partnery (v ČR momentálně 4společnosti), kteří zákazníkům zpravidla zasílají nabídky „ušité“ na míru pro jejich projekty, mnohdy včetně nabídky jejich implementace.

Jednotlivé verze tokenu se liší především v kapacitě úložišť (20-64kB) a podporovaných kryptografických algoritmů. Druhým rozdílem bývá kompatibilita mezi různými aplikacemi a podpora operačních systémů. V Tab. 5.1 je uvedeno základní srovnání jednotlivých dostupných řešení USB tokenů [9].

Tab. 5.1: Srovnání komerčně dostupných tokenů

	iKey 2032	iKey Rainbow 3000	iKey 4000	Aladin eToken
OS čipu	STARCOS 2.3	STARCOS 2.3	STARCOS 2.3	CardOS 4.2
Paměť pro data	32kB	20kB	64kB	64kB
Max. délka klíčů	RSA 2048-bit	RSA 1024-bit	RSA 2048-bit	RSA 2048-bit
Algoritmy	RSA, DSA, DH, DES/3DES, MD5, SHA-1, RC2, RC4	RSA, MD5, MD2, SHA-1, MD160, RC2, RC4, DES/3DES	RSA, DH, 3DES, AES, SHA-1	RSA, DES/3DES, SHA-1
Generováni klíče na kartě	ANO	ANO	ANO	ANO
PIN	4-20 znaků	1-8 znaků	4-20 znaků	1-20 znaků
PUK	Nemá	1-8 znaků	1-8 znaků	Nemá
Orient. cena	1 220 Kč	1 200 Kč	1 900 Kč	nedostupná

5.2.7. Bezpečnost hardwarových tokenů

Aby mohl nějaký hardwarový token bezpečně poskytnout autentizaci uživatele a autorizaci jeho operací, je nutné, aby především on sám byl navržen s ohledem na požadovanou míru bezpečnosti. V této části budou popsány některé otázky a problémy, které se týkají oblasti zabezpečení právě HW tokenů.

Největším problémem je samozřejmě lidský faktor. V praxi jsem se jako IT administrátor osobně několikrát setkal se situací, kdy uživatel odkládá svůj osobní token na „bezpečné“ místo, za které považuje prostor pod monitorem v kanceláři. Dokonce jsem i viděl nálepku

na takto uloženém tokenu, na níž byl napsán PIN pro případ, že jej uživatel zapomene. V tomto případě bohužel nepomohou ani nejlepší bezpečnostní a kryptografické metody a je třeba zařídit větší uvědomělost uživatelů těchto tokenů.

Další již sofistikovanější bezpečnostní rizika lze rozdělit dle toho, je-li třeba mít zařízení fyzicky k dispozici, nebo jde-li o útoky spíše ***softwarové (logické)***. Logické útoky jsou velmi podobné klasickým útokům, tak jak je známe z počítačového světa - jde o objevení softwarové chyby, kvůli které jsou pak data dostupná i bez znalosti hesla, případně PINu. Tyto útoky se většinou dělí na [10]:

- **Útoky na klíče** – využívají vlastnosti pro zajištění kompatibility se staršími zařízeními nebo znalosti vztahů mezi klíči, kdy při (částečné) znalosti klíče je možné odvodit nebo dopočítat jiný (citlivější) klíč.
- **Nedostatečná kontrola parametrů** – je definován jako neočekávaná posloupnost transakcí, jejímž cílem je oklamat bezpečnostní modul tak, aby zpřístupnil tajemství způsobem, který odporuje bezpečnostní politice zařízení. Mezi zástupce těchto útoků patří útok s decimalizační tabulkou nebo útoky využívající možnosti odvodit části PINu nebo najít vhodnou kolizi.
- **Nevynucení politiky** – zneužívá API, která neobsahují a nevyhodnocují žádnou bezpečnostní politiku – např. PKCS #11, které je navrženo jako rozhraní mezi aplikacemi a jednouživatelskými bezpečnostními zařízeními a přesto bývá často používáno jako hlavní API i u mnohých kryptografických modulů.

Obranou před softwarovými útoky by mohlo dle [11] být nejlepším řešením mít softwarovou část zařízení s formálním důkazem správnosti. Verifikace kódu je však komplikovaná záležitost a automatizované techniky jsou v současnosti použitelné spíše na kratší bloky kódu. To však navádí k první důležité zásadě – vytvářet API po malých, nějakým způsobem ověřitelných částech.

Fyzické případy se liší obtížností a náročností na vybavení útočníka. Rozdělujeme tyto základní kategorie:

- **Neinvazivní metody –** využívají skutečnosti, že ne všechna zařízení bývají navržena s dostatečnou odolností vůči extremním výkyvům či vlivům okolního prostředí. Takovéto stavy pak mohou způsobovat, že se obvody začnou chovat chybně a dochází k únikům kryptografického materiálu. Ačkoliv je tu jisté riziko zničení testovaného obvodu, zpravidla nezanechávají neinvazivní metody žádné stopy po své činnosti.
- **Invazivní metody –** zařízení se nejprve rozebere až na samotný čip, odstraní se krycí vrstvy z čipu a útočník se následně pomocí speciálního hardwaru, mikroskopů a mikrosond napojí na sběrnici, případně vyčítá data přímo z paměti. Tyto metody patří

mezi nejnáročnější na vybavení (už kvůli nutné míře potřebných znalostí i miniaturním rozměrům současných čipů). Nejčastěji jsou tyto metody používány zejména pro čipové karty.

- **Semiinvazní metody** – čip je rozebrán jen částečně, obvykle pouze zbaven vrchní vrstvy nebo plastového krycího pouzdra (Obr. 5.5) a dále je na něj působeno některým druhem záření, obvykle UV, elektromagnetickým či silným světelným zdrojem. Tento druh útoků je finančně dostupný a potřebné znalosti jsou nižší, než u invazivních útoků. Semiinvazivní útoky jsou často používány pro útoky na USB zařízení - jejich velikost je dostatečná na to, aby nebylo nutné používat mikroskopy a často si při jejich výrobě sami výrobci pomáhají různými testovacími obvody, které pak nedostatečně odstraňují. To vede ke zjednodušení situace při získávání klíčů a jiných citlivých dat uložených na takovýchto zařízeních.

Obranou před fyzickými útoky může být snížení rizika úniku informací pomocí datových remanencí. K tomu se doporučuje nepracovat s kryptografickými klíči, hesly a dalšími citlivými informacemi v SRAM (druh používané paměti v tokenech), přesunovat je mezi lokacemi v paměti a provádět bezpečné mazání původních míst. Dále je vhodná kombinace SRAM čipů s obvody na měření okolní teploty, společně s nějakým dalším, reakčním obvodem.

Proti UV útokům je možné se bránit kombinací paměťových obvodů s materiály citlivými na UV záření taková, že nadměrné UV záření dokáže bezpečně zničit veškerá data, která jsou v paměťovém čipu uložena.

5.2.8. Hardware Security Module (HSM)

HSM moduly (občas označovány i Host Security moduly) jsou speciálními úložišti pro PKI kryptografický materiál. Soukromé klíče důležitých serverů (např. soukromý klíč samotné certifikační autority) **nebývají ukládány na USB tokenech**, ale právě v těchto specializovaných „černých skříňkách“, které jsou navíc vybaveny speciální fyzickou bezpečností umožňující např. zašifrování soukromých klíčů symetrickou šifrou, nebo dokonce vymazání kryptografického materiálu v případě mechanického pohybu s touto skříňkou.

Obr. 5.6: HSM modul pro umístění do 19" rozvaděče [14]

Hlavním přínosem HSM je však ochrana kryptografického materiálu před nepovolanými osobami – zejména správcům serverů, ke kterým je HSM připojen. Použití odděleného HSM totiž umožňuje **oddělit správce serveru od bezpečnostního administrátora**, který jediný má přístup ke kryptografickému materiálu.

6. PRAKTICKÉ IMPLEMENTACE PKI

Využití infrastruktury veřejných klíčů (PKI) je obecně považováno za jednu z nejúčinnějších metod technologického zabezpečení základních atributů elektronické konfigurace (**integrita, důvěrnost, autenticita, zodpovědnost**) postavených na využití kryptografie.

S využitím metody PKI mohou být přenášené zprávy vybaveny při odeslání elektronickým podpisem a zašifrovány, při přijetí je zpráva dešifrována a elektronický podpis ověřen. Vedle přenosů dat a **ověření zpráv** je možno PKI využít i pro **ověření identity uživatelů** (autentizace) a pro **ochranu komunikačních kanálů** (SSL).

V rámci této kapitoly tedy bude pospán hlavní stavební kámen většiny praktických implementací PKI, kterým je **protokol SSL/TLS**. Ten je využíván protokoly různých vrstev ISO/OSI jako jsou např. protokol HTTPS (zabezpečení webových stránek), VPN (OpenVPN), SSH, FTP, SMTP, POP3 a dalších. Protokol SSL je rovněž využíván v případě autentizace uživatelů prostřednictvím PKI.

V dalších částech kapitoly bude popsáno využití PKI pro **zabezpečení elektronické pošty** a dále pak budou pro zajímavost uvedeny relativně nové možnosti využití PKI, kterými jsou **zabezpečení systému DNS** (DNSSEC) a metoda **zabezpečení sítí WiMAX**.

6.1. Zabezpečení komunikace šifrováním a autentizace komunikujících stran (SSL/TLS)

Protokol SSL byl vyvinut pro použití na webových stránkách, kdy záleží na utajení přenášených dat. Například v aplikacích internetového bankovnictví či při zadávání uživatelského jména a hesla na webových stránkách. SSL je zkratka Secure Sockets Layer. Je to protokol/vrstva, vložená mezi vrstvu transportní (např. TCP/IP) a aplikační (HTTP). Protokol SSL postupem času procházel vývojem od verze 1 po SSL verze 3. **Oficiálním protokolem Internetu** se však stal až *protokol TLS* (Transport Layer Security Protocol),

také označován jako SSL verze 3.1. Z kryptografického hlediska se jedná o hybridní protokol, tj. používá symetrický i asymetrický kryptosystém. Z hlediska modelu OSI jej lze umístit do relační a prezentační vrstvy, SSL přebírá data od aplikace, v rámci prezentační vrstvy je rozdělí na *segmenty*, které *zašifruje*. Před přenosem v rámci relační vrstvy naváže spojení a provede vzájemnou autentizaci stanic. Poté využívá služeb transportní vrstvy k samotnému přenosu dat (tcp i udp na portu 443). Spojením HTTP + SSL vznikl název **protokolu HTTPS**.

Princip SSL spočívá ve vytvoření zabezpečeného kanálu mezi klientem (prohlížeč) a serverem. SSL umožňuje bezpečnou výměnu informací při inicializaci TCP/IP spojení, při kterém si klient a server ustavují konkrétní bezpečnostní parametry pro následný provoz a vzájemně se autentizují certifikáty. Po ustavení těchto parametrů je veškerá komunikace (http požadavky i http odpovědi) plně šifrována a to včetně URL, které klient požaduje, webových formulářů apod.

SSL protokol v sobě zahrnuje dva subprotokoly. Jedná se ***SSL Handshake Protocol*** a ***SSL Record Protocol.*** Pomocí SSL Record Protocol se **definuje formát**, jakým budou data přenášena a SSL Handshake Protocol zajišťuje **úvodní výměnu informací**.

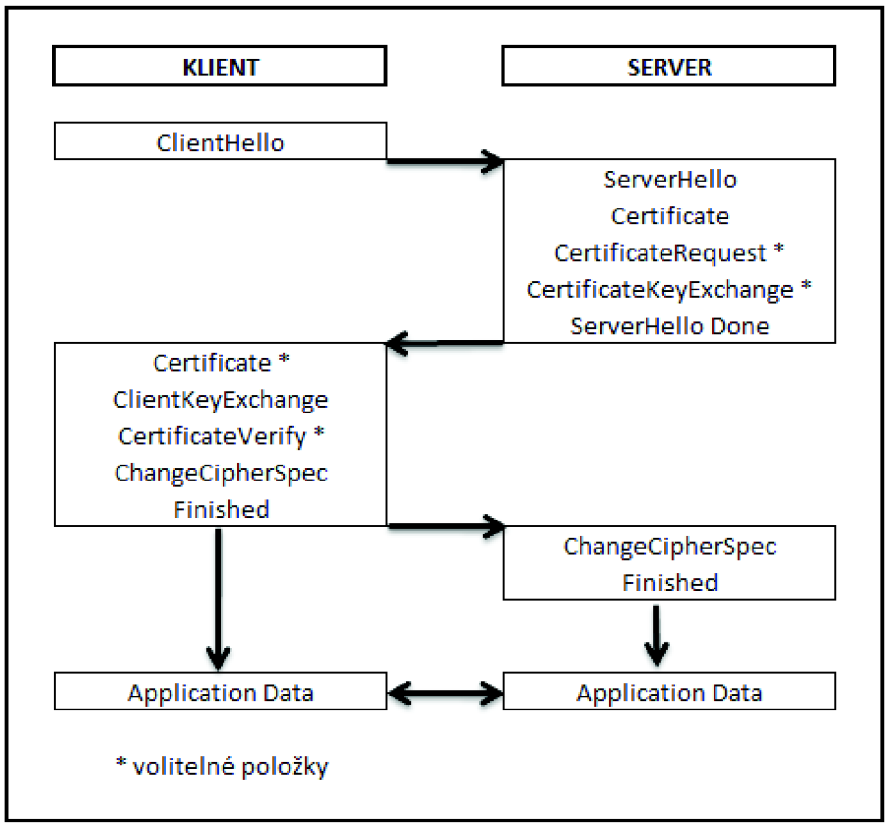
6.1.1. SSL Handshake Protocol

Protokol SSL používá kombinaci symetrického a asymetrického šifrování. Ve fázi handshake se používá **šifrování asymetrické**, které poskytuje lepší vlastnosti pro autentizaci. Jeho hlavní nevýhodou je, že je relativně pomalé a náročné na strojový čas ve srovnání se šifrováním symetrickým. Proto je použito jen ve fázi handshake a pro další přenos se používá šifrování symetrické. SSL Handshake **zajišťuje autentičnost** komunikujících a vyjednání klíčů, je také nazýván *key-exchange protocol*. Zajišťuje **ustavení bezpečné cesty** (session) mezi dvěma účastníky. Připravuje tedy parametry pro Record protokol.

Fáze handshake probíhá v následujících krocích:

1. Klient odešle verzi SSL, informace o šifrách, které používá a náhodně generovaná data.
2. Server odešle svoji verzi SSL, informace o šifrách, které používá, náhodně generovaná data a **svůj certifikát.**
3. Klient pomocí **získaného certifikátu ověří důvěryhodnost serveru**, pokud tak nelze učinit, je o tom informován uživatel k rozhodnutí, zda spojení ukončit.
4. Klient vytvoří z dat dosavadní komunikace tzv. **premaster secret**, zašifruje ho pomocí veřejného klíče serveru získaného z certifikátu a odešle ho serveru

5. Server použije svůj privátní klíč k **dešifrování premaster secret** a vytvoří z něj master secret, ze kterého si vygeneruje klíč sezení
6. Klient si rovněž vytvoří **master secret** a z něj vygeneruje **klíč sezení**
7. Klient zašle zprávu serveru, že další data **budou šifrována klíčem sezení** a odešle zašifrované sdělení, že fáze handshake byla u něj skončena
8. Server rovněž zašle, že další data budou šifrována klíčem sezení a odešle zašifrované potvrzení, že fáze **handshake byla ukončena** a může být **zahájen Record Protocol**



Obr. 6.1: Posloupnost fáze handshake

6.1.2. SSL Record Protocol

Jak uvádí doc. Burda v [3] Record protocol provádí zapouzdření dat aplikačních protokolů - nejnižší SSL úroveň. Z hierarchicky vyšší OSI vrstvy (např. od HTTP) je převzata zpráva Z, která je zpracována tímto postupem:

1. **Fragmentace** Z na segmenty SS o maximální délce 214 bajtů.
2. Bezeztrátová **komprimace** segmentu SS do podoby S.
3. Výpočet **autentizačního kódu** h segmentu S, kdy h = F(S, Autentizační_klíč) a F je hashovací funkce.

4. Vytvoření bloku B = S || h.
5. Zašifrování bloku B symetrickým klíčem sezení.

Zašifrovaný blok je opatřen služebními daty a předán transportní vrstvě k přenosu TCP protokolem.
Na přijímací straně se provedou inverzní operace v opačném pořadí. Příjemce zároveň kontroluje jím vypočítané autentizační kódy s přijatými.

6.1.3. Autentizace serveru pomocí SSL/TLS protokolu

SSL umožňuje i komunikaci bez jakékoliv autentizace (tzv. plně anonymní komunikace). Ta se však v praxi téměř nepoužívá. V praxi aplikace téměř vždy **vyžadují autentizaci serveru**. K autentizaci využívají server i klient certifikáty. Jestliže strana nezašle svůj certifikát druhé straně, znamená to, že se nechce autentizovat.

V případě autentizace serveru zašle server klientovi svůj certifikát, ze kterého **klient vyjme veřejný klíč** serveru, kterým pak šifruje předběžné sdílené tajemství, které následně zašle serveru. Pomocí soukromého klíče server dešifruje toto předběžné sdílené tajemství, čímž se **server zároveň i autentizuj**e (nikdo jiný než server nemá příslušný dešifrovací soukromý klíč).

V terminologii Handshake protokolu (HP) server zasílá svůj certifikát klientovi ve zprávě certifikát (*Certificate*).

Také může nastat situace, kdy certifikát serveru umožňuje jeho autentizaci, ale neumožňuje přenos předběžného tajemství. V tom případě server musí vygenerovat dočasná párová data, která však musí spárovat se svým certifikátem – podepíše tato data svým klíčem – vzniká tzv. ***dočasný veřejný klíč***.

V tomto případě v terminologii HP protokolu server zasílá klientovi svůj certifikát (*Certificate*) a dočasný veřejný klíč (*CertificateKeyExchange*).

6.1.4. Autentizace klienta pomocí SSL/TLS protokolu

Autentizace klienta je výrazně jednodušší. Pokud se chce klient autentizovat, zašle serveru postupně zprávy *Certificate* (obsahuje klientův certifikát) a *CertificateVerify*. CertificateVerify zašle elektronický podpis TLS z předchozí komunikace protokolem HP, který mj. obsahuje náhodná čísla generovaná klientem i serverem (*client_random* a *server_random*).

Server následně provede verifikaci tohoto elektronického podpisu TLS (nikdo jiný než klient nemá k dispozici soukromý klíč, kterým byl elektronický podpis TLS vytvořen).

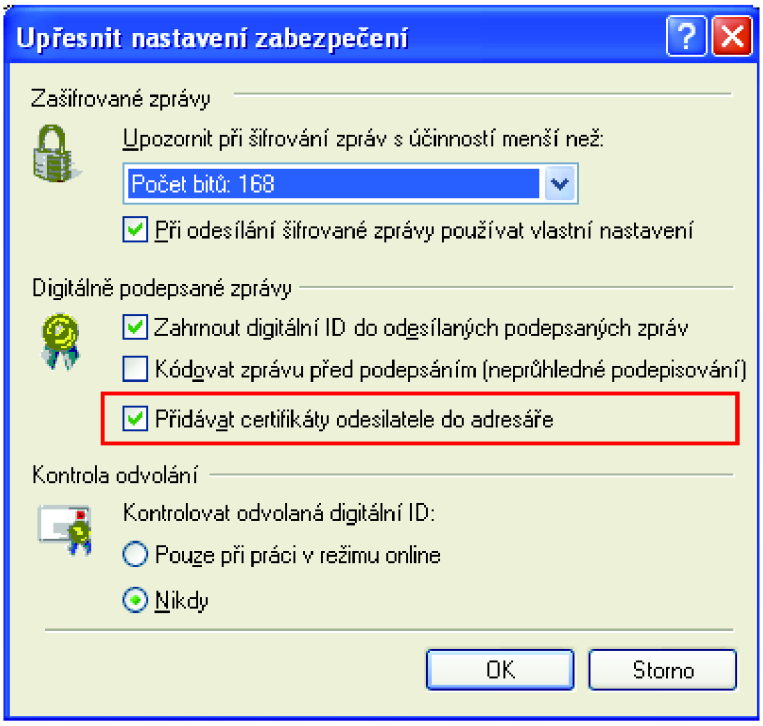
6.2. Bezpečnost elektronické pošty E-mail

Standardizovaný protokol pro bezpečnost elektronické pošty je označován jako *S/MIME* (Secure Multipurpose Internet Mail Extensions). Je definován v RFC-2632 až RFC-2634, respektive RFC-3851. Nedílnou součástí tohoto protokolu je podpora pro kryptografické operace (hash, symetrická i asymetrická kryptografie) využívající i certifikáty a CRL. Protokol umožňuje zasílat elektronicky podepsaná data (zajištěna integrita a nepopiratelnost), tak zašifrovaná a uzavřená v elektronické obálce.

Podle standardu S/MIME musí být adresa elektronické pošty uvedena v použitém certifikátu, avšak certifikát může obsahovat i více poštovních adres najednou. Aplikace odesilatele i příjemce pak mj. kontrolují shodu adresy uvedené v certifikátu a ve zprávě.

Vlastní použití bezpečné elektronické pošty po instalaci certifikátu do aplikace je jednoduché. Většina grafických poštovních aplikací umožňuje podepsat, nebo zašifrovat zprávu jednoduše pouhým kliknutím na příslušnou ikonu v průběhu komponování zprávy. Při podepisování zprávy potřebuje aplikace soukromý klíč odesilatele. Složitější je to v případě šifrování zprávy, kdy aplikace potřebuje veřejný klíč příjemce. Jednou možností jak jej získat je **stáhnout certifikát příjemce z webových stránek** certifikační autority, což může být často problém, jelikož neexistuje žádná povinnost zveřejňování certifikátů.

Druhou a jednodušší možností získání certifikátu je jednoduše napsat plánovanému příjemci **podepsanou zprávu se žádostí o certifikát**. Příjemce poté může pomocí veřejného klíče odesilatele, který získal z přijatého certifikátu zašifrovat svůj certifikát a odeslat jej zpět odesilateli. Popřípadě může příjemce původnímu odesilateli odpovědět opět jen nešifrovanou, avšak vlastním soukromým klíčem podepsanou zprávou obsahující jeho certifikát s veřejným klíčem. Pro následnou další komunikaci je vhodné, aby si oba účastníci uložili veřejné klíče k příslušným kontaktům v adresáři. Některé poštovní aplikace umožňují toto ukládání automaticky (např. MS Office Outlook), u některých (např. Outlook Express) je třeba tuto funkci nejdříve povolit.



Obr. 6.2: Nastavení klienta Outlook Express

6.3. Bezpečné DNS (DNSSEC)

DNSSEC je **rozšíření doménového systému DNS**, které umožní ověřit pravost informací získaných z DNS a předcházet tak např. phishingu či spamu. Principem DNS je překlad jmenných internetových adres, jako například www.vutbr.cz na IP adresy, pomocí nichž dokáží zajistit zobrazování webových stránek, odesílání e-mailů, telefonování po internetu a další běžné internetové služby. DNSSEC zvyšuje bezpečnost při používání DNS tím, že zabraňuje podvržení falešných, pozměněných či neúplných údajů o doménových jménech.

DNSSEC je tedy systém doménových jmen chráněný ověřovacími certifikáty. Držitel domény vygeneruje dvojici soukromého a veřejného klíče. Svým soukromým klíčem pak **elektronicky podepíše technické údaje**, které o své doméně do DNS vkládá. Pomocí veřejného klíče je pak možné ověřit pravost tohoto podpisu. Aby byl tento klíč dostupný všem, publikuje jej držitel ke své doméně u nadřazené autority, kterou je pro všechny domény v rámci ČR registr domén .cz (CZ.NIC).

6.4. Zabezpečení autentizace WiMAX klientů

Standardu 802.16 definující bezdrátové sítě WiMAX definuje mj. *protokol PKM* (Privacy Key Management) sloužící k bezpečné autentizaci klientů pomocí PKI certifikátů. V průběhu autentizace základnová stanice **autentizuje klienta na základě digitálního certifikátu X.509**, který klientská stanice obdrží při výrobě od výrobce. Certifikát obsahuje veřejný klíč a MAC adresu. Stanice musí certifikát předložit pro ověření základnovou stanicí a po úspěšné autorizaci dostane autorizační klíč zašifrovaný ověřeným veřejným klíčem.

7. DOSTUPNÁ ŘEŠENÍ SYSTÉMU PKI

Tato kapitola popisuje dostupná řešení systému PKI. Pokud chceme provozovat komerční certifikační autoritu, můžeme si zakoupit některý z komerčních produktů na trhu např. *RSA KEON, Entrust, Baltimore uniCERT*, a další. Jedná se však o **velice nákladná** a **rozsáhlá řešení** vhodná pro rozsáhlé organizace, popřípadě organizace vyžadující vysokou úroveň zabezpečení (armáda apod.).

U nás se v praxi nejčastěji používají řešení v podobě OpenSSL (především pro zabezpečení webových serverů) a certifikační služba dodávaná jako součást Microsoft Windows Server (MSCA).

7.1. OpenSSL

OpenSSL je multi-platformní open source implementace protokolů SSL a TLS. Knihovny systému jsou napsány v jazyce C, je však možné jej provozovat v prostředí jazyka JAVA. Z domovských stránek *www.openssl.org* je možné získat zdrojové kódy pro různé operační systémy, ať už jsou to **unixové systémy** (Solaris, Linux, Mac OS X a operační systémy BSD), tak i pro **systémy Microsoft**.

OpenSSL je možné používat dvěma způsoby. Buďto jako utilitu příkazové řádky pro provádění kryptografických operací anebo jako knihovnu (API) pro využití v dalších aplikacích. OpenSSL může být použit pro [13]:

- vytváření RSA, DH a DSA klíčů
- vytváření X.509 certifikátů
- šifrování a dešifrování
- SSL/TLS – testování klienta a serveru
- podepisování, šifrování a následná verifikace a dešifrování S/MIME emailů

Hlavní aplikací je program openssl, který spouští jednotlivé dílčí aplikace. Syntaxe příkazu je následovná:

```
openssl aplikace parametry
```

Příklad:

```
openssl req -config openssl.cnf -new -nodes -keyout private/jvonasek.key
-out jvonasek.csr -days 365
```

Pomocí programu openssl lze spustit následující aplikace [5]:

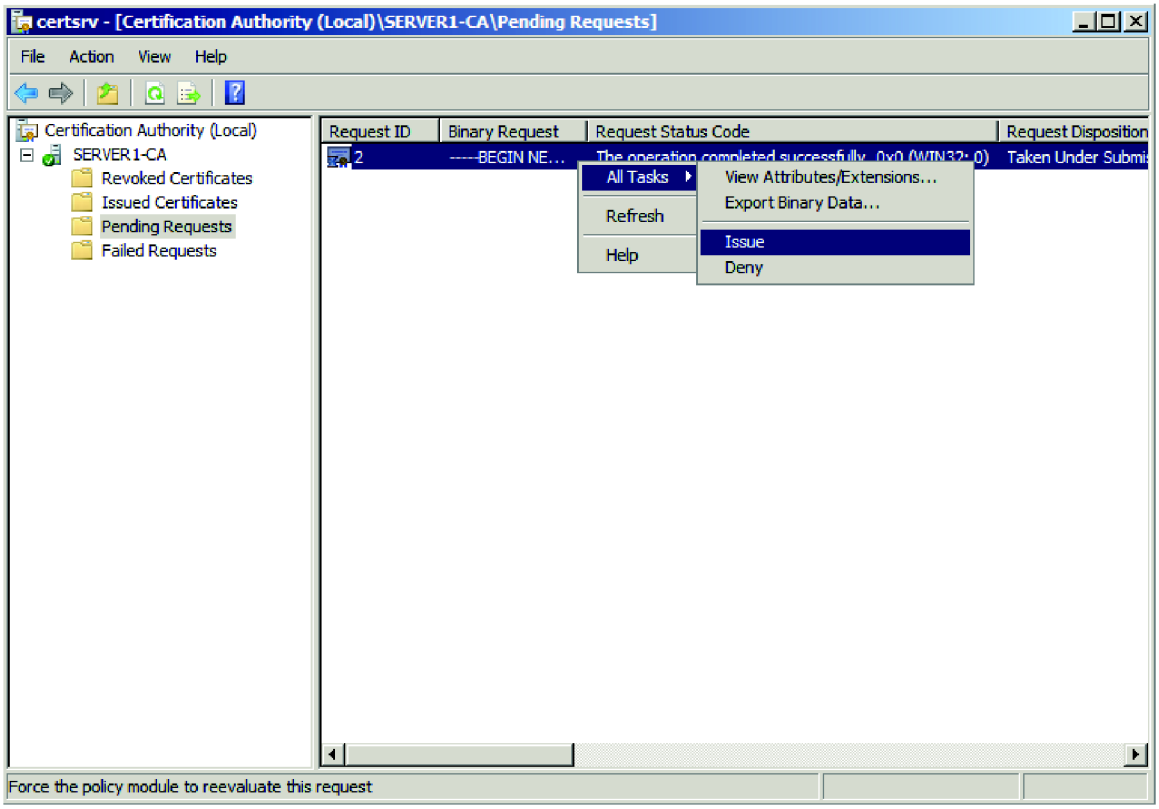
Tab. 7.1: Přehled aplikací OpenSSL

Aplikace	Popis
ca	Podepisování žádosti o certifikáty a CRL, udržování "vydaných" certifikátů
ciphers	Seznam podporovaných šifer
crl	Práce s CRL
dgst	Výpočet otisku
dh	Manipulace s Diffie-Hellman
dhparam	Generování Diffie-Hellman parametrů
dsa	Manipulace s DSA
dsaparam	Generování DSA parametrů
ec	Manipulace s EC (Eliptické křivky)
ecparam	Generování EC parametrů
enc	Šifrování/dešifrování a kódování/dekódování Base64
gendh	Generování párových dat Diffie-Hellman
gendsa	Generování párových dat DSA
genrsa	Generování párových dat RSA
passwd	Práce s hesly
rand	Generování pseudonáhodných čísel
req	Generování žádosti o certifikáty dle PKCS#10
rsa	Manipulace s RSA klíči
smime	Práce s e-maily dle normy S/MIME
verify	Verifikace certifikátu
version	Výpis aktuální verze OpenSSL
x509	Práce s certifikáty dle normy X.509 včetně jejich podepisování

7.2. Certifikační autorita Microsoft (MSCA)

V operačních systémech Windows Server je certifikační služba součástí používanou pro vytváření a správu MSCA. Pomocí certifikační služby operačního systému Windows lze vytvořit MSCA, která přijímá žádosti o certifikáty, ověřuje informace uvedené v žádostech a totožnost žadatelů, vystavuje certifikáty, zneplatňuje certifikáty a publikuje CRL.

Procesy pracující s certifikáty v systému Windows používají jako **standardní formát certifikátů X.509v3**. Certifikáty jsou MSCA vydávány na základě informací uvedených v žádosti o certifikát a nastavení zadaných v šabloně certifikátu. Šablona certifikátu je sada pravidel a nastavení, které mají být použity pro přijatou žádost o certifikát. Pro každý typ certifikátu, který může být vydán MSCA pro rozlehlou síť, je třeba nakonfigurovat šablonu certifikátu. Šablony certifikátů lze vytvářet a upravovat MSCA pro rozlehlé sítě v systémech Windows Server Enterprise a Datacenter Edition. Jsou ukládány v adresáři služby Active Directory a mohou být použity všemi MSCA v doménové struktuře. To správci umožňuje vybrat jednu nebo více výchozích šablon instalovaných spolu s MSCA nebo vytvářet šablony přizpůsobené pro určité úkoly nebo role.



Obr. 7.1: Snap-in MMC konzole pro správu CA

MSCA může plnit následující funkce:

- Zápis žádostí uživatelů o certifikáty, přijaté pomocí webu nebo modulu snap-in MMC (Microsoft Management Console), případně transparentním automatickým zápisem. Tato funkce plní roli RA.
- Omezení rozhodnutí požadovaných po žadateli o certifikát pomocí šablon certifikátů v závislosti na zásadách definovaných CP CA.
- Využití adresářové služby Active Directory pro publikování důvěryhodných nadřízených certifikátů, vystavených certifikátů a CRL.
- Implementace přihlašování do domény operačního systému Windows pomocí karty Smart Card.

8. POPIS LABORATORNÍ ÚLOHY

Vytvořená laboratorní úloha sestává ze dvou částí. První část práce tvoří teoretický úvod, ve kterém si studenti nejdříve obnoví některé znalosti v oblasti **kryptografických metod**, které jsou nezbytné pro pochopení celé problematiky a to především teorii vzniku elektronického podpisu a podepisování dokumentů. Další část je věnována popisu elektronického certifikátu včetně souvislosti certifikátu s **českou legislativou** – tzv. kvalifikovaný certifikát. V rámci certifikátů jsou studenti detailněji seznámeni se strukturou certifikátu dle standardu X.509.

V následné části teoretického seznámení s infrastrukturou veřejných klíčů se studenti seznámí se **strukturou celého systému PKI** a rolemi jednotlivých prvků tohoto systému (CA, RA, CRL).

Poslední část teoretické části laboratorní úlohy popisuje praktické implementace a využití infrastruktury veřejných klíčů. Největší prostor je zde věnován podrobnému popisu **protokolu SSL/TLS,** který tvoří základní stavební kámen právě většiny praktických implementaci. Mimo popisu dílčích podprotokolů (Handshake Protocol a Record Protocol) je v závěru popsán samotný princip využití protokolu SSL/TLS jak pro **autentizaci serveru** (známo především z HTTPS), tak i pro **autentizaci jednotlivých klientů** pomocí vlastních certifikátů.

Druhou část laboratorní práce tvoří praktická část, ve které si studenti prakticky ověří získané teoretické znalosti v praxi na operačním systému Linux CentOS.

8.1. Praktická část laboratorní úlohy

V rámci laboratorní úlohy je nezbytné, aby studenti měli k dispozici dva operační systémy. Jeden operační systém (Linux), který bude fungovat jako server a druhý operační systém (Windows), který budou používán jako klient pro připojování se k serveru a ověření správné konfigurace. Vzhledem k tomu, že v rámci laboratoře má každý student k dispozici pouze jeden počítač, je nutné **pracovní prostředí virtualizovat**. Pro tyto účely

je tedy možno využít stávající instalaci, která se momentálně na učebnách nachází – MS Windows XP jako klient a hostitelský systém, ve kterém bude prostřednictvím VMWare Player emulováno druhé virtuální PC, na kterém bude nainstalován OS Linux. Pro tento účel je možno použít buďto pro tuto úlohu speciálně předpřipravený image, popřípadě je možné pokračovat v práci na systému z předchozích laboratorních cvičení.

V rámci laboratorní úlohy studenti plní následující zadané úkoly:

1. **Pomocí OpenSSL nainstalujte vlastní certifikační autoritu.**
2. **Zabezpečte webový server protokolem SSL/TLS. Pro generování klíčů i žádosti o certifikát použijte opět OpenSSL. Žádost o certifikát bude podepsána prostřednictvím Vámi nainstalované certifikační autority.**
3. **Pro uživatele vytvořte pár klíčů, certifikát veřejného klíče a nastavte webový server tak, aby přístup na server prostřednictvím HTTPS protokolu byl umožněn pouze na základě autentizace uživatele pomocí certifikátu a jeho klíče.**

8.1.1. Instalace certifikační autority pomocí OpenSSL

V první fázi laboratorní úlohy si studenti vybudují svoji vlastní jednoduchou kořenovou certifikační autoritu postavenou na systému OpenSSL. Vybudování CA spočívá ve vytvoření vlastní adresářové struktury, do které budou následně zapisovaný certifikáty, klíče, žádosti o certifikáty a zapisován CRL. Dalším krokem je nastavení konfiguračního souboru *openssl.cnf*

Po tomto nakonfigurování a přípravě pracovního prostředí je posledním krokem vygenerování páru klíčů a vlastního tzv. self-signed certifikátu.

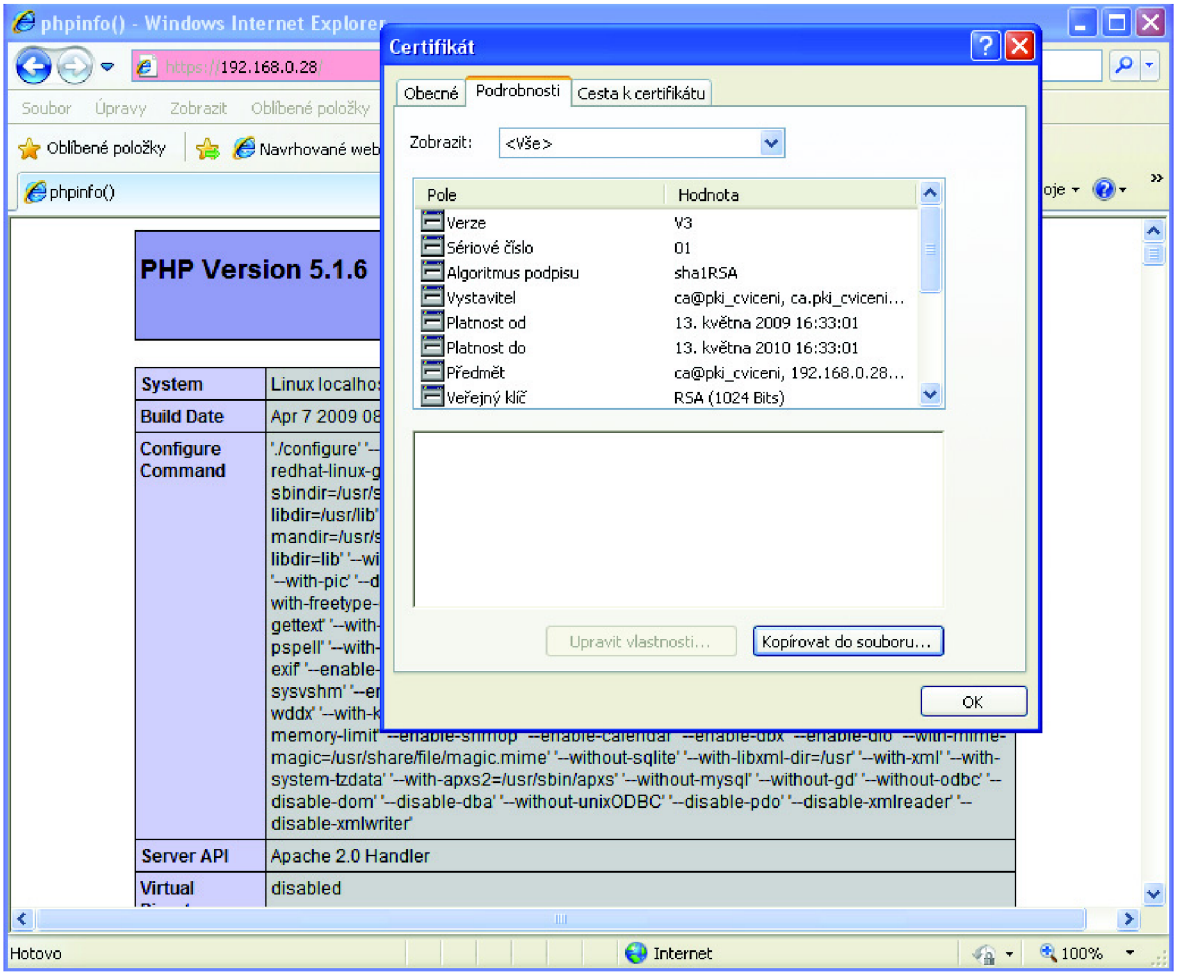
V rámci této kapitoly se studenti prakticky seznámí s nezbytnými prvky potřebnými pro chod CA, dále se seznámí se strukturou příkazů pro ovládání aplikace openssl přes příkazovou řádku.

8.1.2. Konfigurace zabezpečení webového serveru

Standardní instalace systému Linux s webovým serverem apache již zpravidla obsahuje možnost připojování klientů k webovému serveru prostřednictvím protokolu SSL/TLS na portu 443 pomocí certifikátu a soukromého klíče, který je vygenerován již v průběhu instalace.

Pro lepší praktické seznámení s touto problematikou si však studenti v rámci tohoto úkolu vytvoří zabezpečení webového serveru pomocí nově vygenerovaného páru klíčů a veřejného certifikátu, podepsaného certifikační autoritou vytvořenou v rámci prvního

úkolu. Správnost provedení úkolu bude ověřeno zabezpečeným přístupem z hostitelských Windows k běžícímu webovému serveru a zobrazením aktuálně používaného certifikátu protokolem HTTPS.

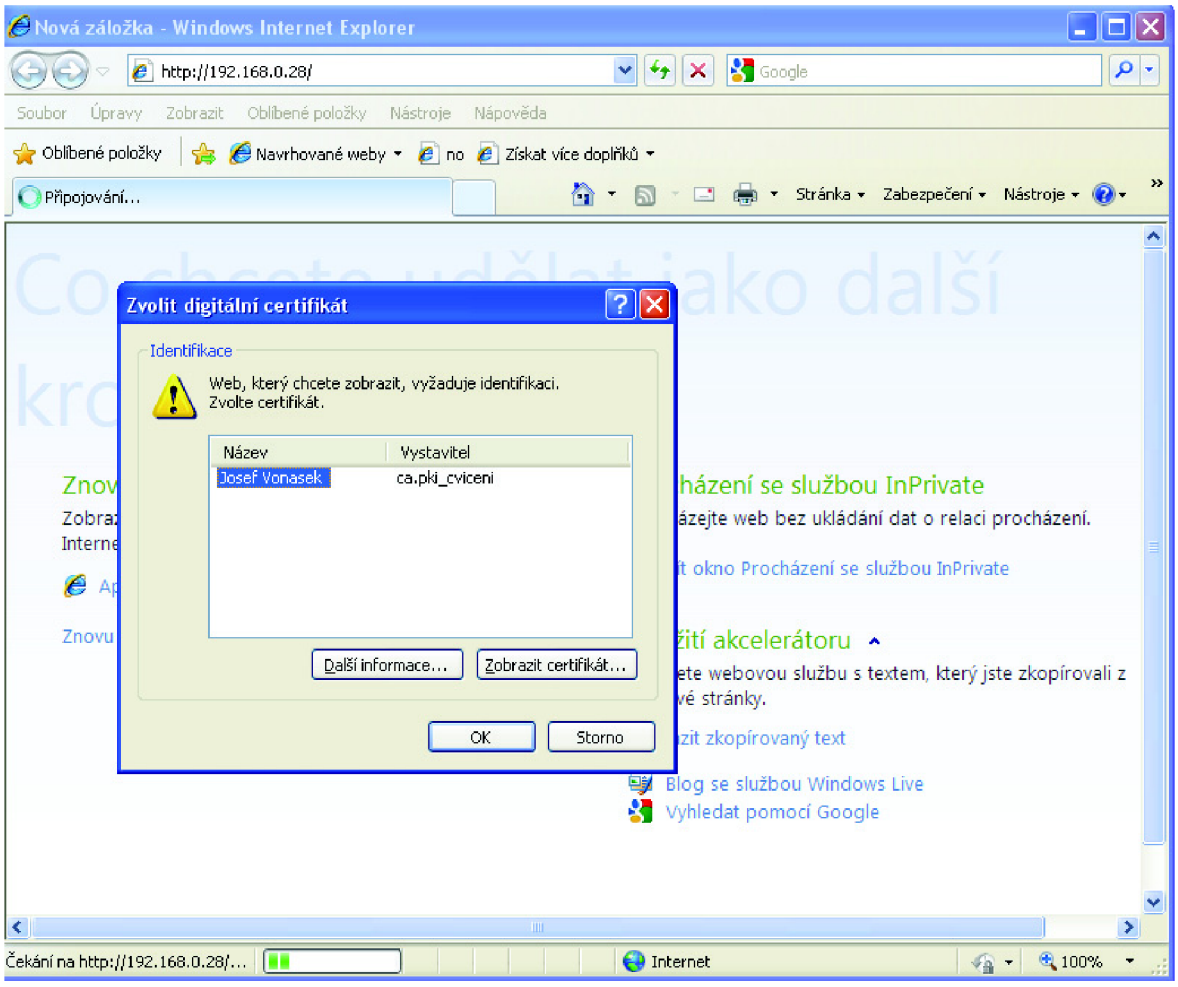


Obr. 8.1: Zobrazení certifikátu webového serveru

V rámci tohoto úkolu se studenti seznámí s možnostmi zabezpečení webového serveru apache, dále s postupem generování a podepisování žádostí o certifikát. Také se seznámí s možnostmi konfigurace mod_ssl modulu, který zajišťuje přenos dat mezi serverem a prohlížečem přes HTTPS protokol.

8.1.3. Autentizace uživatelů pomocí PKI

V závěrečném úkolu student nakonfiguruje mod_ssl modul tak, aby vyžadoval autentizaci klientů prostřednictvím certifikátů podepsaných CA instalovanou v prvním úkolu. Podobně jako v předchozím úkolu v případě serveru si vygenerují pro uživatele pár klíčů spolu s žádostí o certifikát, kterou poté jako administrátor CA podepíší. Podepsaný certifikát spolu se soukromým klíčem **vyexportují do formátu pksc#12**, který je určen právě pro zálohování certifikátů a privátních klíčů. Tento vyexportovaný certifikát s párem klíčů spolu s certifikátem kořenové CA si nainstalují do klientských Windows a opět pomocí webového prohlížeče ověří správnost nastavení. Pokud je vše v pořádku, je klient autentizován pro přístup na webové stránky, v opačném případě mu je přístup na stránky zamítnut.



Obr. 8.2: Autentizace uživatele prostřednictvím certifikátu

V tomto úkolu se studenti seznámí s dalšími možnostmi konfigurace mod_ssl modulu a s možnostmi exportu certifikátu s klíči do formátu pkcs#12. Volitelně jako nástavbový úkol je v návodu popsán mod_rewrite modul a jeho direktivy nezbytné pro nastavení webového serveru pro vyžadování spojení pouze za pomoci SSL/TLS protokolu.

8.2. Výukový webový portál

Speciálně pro účely této laboratorní úlohy byl vytvořen výukový webový portál pro seznámení se s tématikou PKI. Na této stránce studenti naleznou veškeré teoretické informace týkající se infrastruktury veřejných klíčů, které jsou uvedeny i v tištěném návodu k laboratorní úloze. Dále je zde popsán detailní návod pro zadané úkoly v operačním systému GNU/Linux včetně popisu významu jednotlivých příkazů. Oproti tištěnému návodu webový portál navíc obsahuje podrobného názorného průvodce na zprovoznění certifikační autority pod systémem **Microsoft Windows Server 2008**, včetně základní správy této certifikační autority (schvalování x zamítnutí žádostí o certifikát apod.).

Obr. 8.3: Náhled webového portálu pro výuku PKI

9. ZÁVĚR

Cílem této práce bylo prostudovat a popsat problematiku **infrastruktury veřejných klíčů**. V rámci podrobného popisu této problematiky je postupně popsána tématika od nejmenších dílčích prvků, přes jednotlivé komponenty tvořící tyto systémy, až po hotová řešení PKI.

Konečným výstupem studia dané problematiky v obecné teoretické části práce je vlastní **laboratorní úloha**, která je postavena na třech hlavních bodech.

Prvním bodem je **teoretické seznámení** se jednak se základními pojmy infrastruktury veřejných klíčů, dále s popisem jednotlivých subjektů (CA, RA, CRL) tvořících PKI a také s praktickými implementacemi využití tohoto systému.

Druhým a hlavním bodem je **praktická část** laboratorní úlohy, ve které si studenti v průběhu tří zadaných úkolů ověří získané teoretické znalosti.

Pro splnění úkolů je nezbytné, aby každý student měl k dispozici *dva operační systémy* – **Linux** (server) a **Windows** (klient). Jelikož má každý student k dispozici **pouze jedno PC**, je třeba jeden z operačních systémů virtualizovat. K tomuto účelu může byt použita standardní instalace Windows XP, která se nachází na učebnách Ústavu telekomunikací, kde bude výuka probíhat. Tato instalace již zpravidla obsahuje *VMWare Player*, pro který je v rámci této diplomové práce připraven virtuální operační systém CentOS. Úloha je přesto koncipována tak, že je možné použít jiný virtuální systém Linux vytvořený a používaný studenty již v průběhu předchozích cvičení.

V prvním praktickém úkolu laboratorní úlohy studenti vybudují svoji vlastní jednoduchou kořenovou certifikační autoritu postavenou na systému *OpenSSL*. Vybudování CA spočívá ve vytvoření vlastní adresářové struktury, do které budou následně zapisovaný certifikáty, klíče, žádosti o certifikáty a zapisován CRL. Dalším dílčím úkolem je nastavení konfiguračního souboru openssl.cnf . Po tomto nakonfigurování a přípravě pracovního prostředí je posledním krokem vygenerování páru klíčů a vlastního tzv. *self-signed certifikátu*.

Druhý praktický úkol seznámí studenty se *zabezpečením webového serveru* prostřednictvím certifikátu podepsaným jimi vytvořenou certifikační autoritou. V rámci tohoto úkolu student vygeneruje pár klíčů s žádostí o certifikát pro webový server, kterou jako správce CA následně

podepíší. Takto podepsaný certifikát spolu se soukromým klíčem následně propojí s webovým serverem pomocí *mod_ssl* modulu.

V posledním úkolu se studenti obeznámí s možnostmi *autentizace klientů* při řízení přístupu na webové stránky pomocí jejich *certifikátů*. Pro splnění tohoto úkolu student vygeneruje pro uživatele pár klíčů spolu s žádostí o certifikát, kterou poté opět jako administrátor CA podepíše. Takto podepsaný certifikát spolu s klíči vyexportuje do formátu pkcs#12 pro následný import klíčů s certifikátem do klientských Windows. Pomocí mod_ssl modulu nakonfiguruje nutnost autentizace klientů pomocí certifikátů podepsaných jimi vytvořenou certifikační autoritou. Správnost řešení úkolů demonstruje vyučujícímu připojením se z klientských Windows k webovému serveru zabezpečeným HTTPS spojením a zároveň se při vstupu na stránky autentizuje vlastním certifikátem.

Třetím bodem laboratorní úlohy je **výukový webový portál** vytvořený pro případné další prostudování problematiky infrastruktury veřejných klíčů. Tento portál je součástí předpřipraveného virtuálního operačního *systému Linux* a je na něm nastaven jako výchozí webový dokument (DocumentRoot). Oproti klasickému tištěnému *návodu* s podrobným postupem k vypracování zadaných úloh a popisem jednotlivých kroků obsahuje navíc webový portál *podrobného průvodce pro nasazení certifikační autority* v prostředí Microsoft Windows Server, konkrétně ve verzi Windows Server 2008. Tento webový portál je napsán v jazyce *PHP* a může být kromě virtuálního operačního systému Linux umístěn pro případné výukové účely na kterýkoliv veřejný školní webový server, popřípadě bude také umístěn na stránkách **http://pki.petrslavik.com**.

V rámci *laboratorní úlohy* se tak studenti **teoreticky seznámí se systémem infrastruktury veřejných klíčů** a tyto získané znalosti ihned **ověří v praxi**. Snáz při tomto empirickém postupu studenti detailně pochopí, jak funguje *zabezpečení webových serverů* a především jakým způsobem pracovat s *klíči a certifikáty* ať už serverových, tak i jednotlivých klientů. Pokud se studenti dostanou v průběhu své praktické práce do nesnází v jakémkoli z jejích dílčích kroků, mají připraveno mnoho podpůrných materiálů a pomocných vodítek, zejména ve formě *detailně popsaného návodu* popřípadě uceleného *webového portálu*, jež jim pomohou zadané úkoly zdárně dokončit.

SEZNAM POUŽITÉ LITERATURY A ZDROJŮ

[1] BUDIŠ, P.: Elektronický podpis : a jeho aplikace v praxi. 1. vyd. Olomouc : Anag, 2008. 157 s. ISBN 978-80-7263-465-1.

[2] BURDA, K.: Návrh, správa a bezpečnost počítačových sítí : (přednáška. 1). 2009, s. 32.

[3] BURDA, K.: Návrh, správa a bezpečnost počítačových sítí: 10: Síťová bezpečnost). 2009, s. 14-21.

[4] DOSEDĚL, T.: Počítačová bezpečnost a ochrana dat. 1. vyd. Brno : Computer Press, 2004. 182 s. ISBN 80-251-0106-1.

[5] DOSTÁLEK, L., VOHNOUTOVÁ, M.: Velký průvodce infrastrukturou PKI a technologií elektronického podpisu. 1. autoriz. vyd. Brno : Computer Press, 2006. 534 s. ISBN 80-251-0828-7.

[6] ENGELSCHALL, Ralf S.. OpenSSL : The Open Source toolkit for SSL/TLS [online]. c1999-2005 [cit. 2009-03-13]. Anglicky. Dostupný z WWW: <http://www.openssl.org/>.

[7] HANÁČEK, P. - STAUDEK, J.: Certifikační infrastruktury veřejných klíčů, PKI. Učební text MU, Brno 2006., 40 s.

[8] HUNT, C. . Linux - síťové servery. [s.l.] : Soft Press, 2003. 672 s. ISBN 8086497593.

[9] JINDRA, P.: USB token v prostředí CESNET CA. Technická zpráva CESNETu [online]. 2005, roč. 34 [cit. 2008-12-03].

[10] LORENC, V., MATYÁŠ, V.: Autentizační HW a možná vylepšení. Zpravodaj ÚVT MU. ISSN 1212-0901, 2007, roč. XVIII, č. 1, s. 17-20.

[11] LORENC, V., MATYÁŠ, V.: Odolnost kryprografického hardwaru s ohledem na nasazení. XXVIII. konference : Sborník příspěvků. 2006, 1, č. 1, s. 21-33.

[12] NEWMAN, D.:. PKI: Build, buy or bust?. Network World [online]. 2001 [cit. 2009-05-14]. Dostupný z WWW: <http://www.networkworld.com/research/2001/1210feat.html>.

[13] RAMÓN , H.. Linux : praktická bezpečnost. [s.l.] : Grada Publishing, 2003. 440 s. ISBN 80-247-0652-0.

[14] WIKIPEDIA PROJECT: Hardware Security Module : Wikipedia, the free encyclopedia [online]. 2009 [cit. 2009-05-16]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Hardware_Security_Module>.

[15] :- Zákon o elektronickém podpisu (č. 227/2000 Sb.). Parlament ČR, Praha 2000.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

DN	Distinguished name (identifikátor objektů)
DNS	Domain Name Systém
DNSSEC	Domain Name System Security Extensions
EF	Elementary File
FTP	File Transfer Protocol
HP	Handshake Protocol
HSM	Hardware Security Module
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICT	Information and Communication Technologies
ISO	International Organization for Standardization
ITU	International Telecommunication Union
LDAP	Lightweight Directory Access Protocol
MD5	Message-Digest algorithm 5
MF	Master File
MSCA	Microsoft Certificate Authority
OSI	Open Systems Interconnection
OT	Otevřený text
PKI	Public Key Infrastructure
PKM	Privacy Key Management
POP3	Post Office Protocol version 3
QCA	Kvalifikovaný certifikát
RA	Registration Authority (registrační autorita)
RSA	Asymetrický šifrovací algoritmus (Rivest, Shamir, Adleman)
S/MIME	Secure Multipurpose Internet Mail Extensions
SHA-1	Secure Hash Algorithm
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
SSL	Secure Sockets Layer
ŠT	Šifrovaný text
TLS	Transport Layer Security
USB	Universal Serial Bus
VPN	Virtual private network

SEZNAM PŘÍLOH

Laboratorní úloha infrastruktury veřejných klíčů

1 Teoretický úvod

Infrastruktura veřejných klíčů (PKI) je systém digitálních certifikátů, certifikačních autorit (CA) a dalších registračních úřadů, které slouží k ověřování platnosti každé strany zúčastněné v určité elektronické transakci. Jako základní stavební kámen je přitom používána tzv. **asymetrická kryptografie** s *veřejným a soukromým klíčem*.

Úkolem PKI je zajistit správu veřejných klíčů a certifikátů veřejných klíčů tak, aby pomocí asymetrických šifrovacích technik bylo možné poskytovat bezpečnostní služby, jako jsou autentizace, zajištění nepopiratelnosti původu a důvěrnosti.

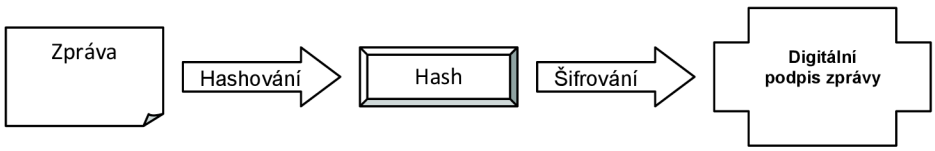
PKI se v praxi využívá k těmto účelům:
Elektronický podpis - datové soubory (daňové přiznání, platební příkazy), elektronická pošta, datová on-line komunikace (SSL, IPSec).
Identifikace a autentizace uživatelů - bezpečné přihlašování k PC, serverům, či aplikacím pomocí čipové karty, USB tokenu.
Šifrování - soubory, elektronická počta, datová on-line komunikace (SSL, TLS), tvorba VPN

1.1 Digitální podpis

Digitální podpis je mechanismus, kterým se zajišťuje ***nepopiratelnost*** dat (pravost dokumentu). Digitální podpis je vytvářen ve dvou krocích

1. Vypočte se hash (otisk) dokumentu - Pro použití digitálního podpisu potřebujeme nejprve nějakou známou hashovaní funkci (např. MD5 nebo SHA-1). Známou v tom smyslu, aby všichni adresáti, kteří budou chtít ověřit pravost naší zprávy, tuto funkci znali (resp. ji znal program, který ověření provede).

2. Výsledný hash se šifruje soukromým klíčem uživatele – Soukromým klíčem šifrovaný hash ze zprávy se nazývá digitální podpis zprávy.

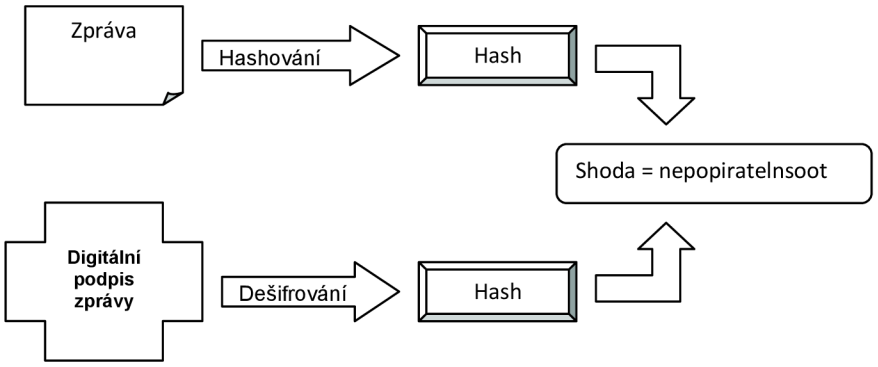


Obr. 1.1: Schéma vzniku digitálního podpisu

Podpis pak přiložíme k původní zprávě, kterou podepisujeme, a zprávu i s touto přílohou odešleme. Příjemce zprávu otevře a provede její ověření (verifikace)

1. Vypočte se hash dokumentu - Pomocí stejné hashovací funkce zakóduje její obsah.

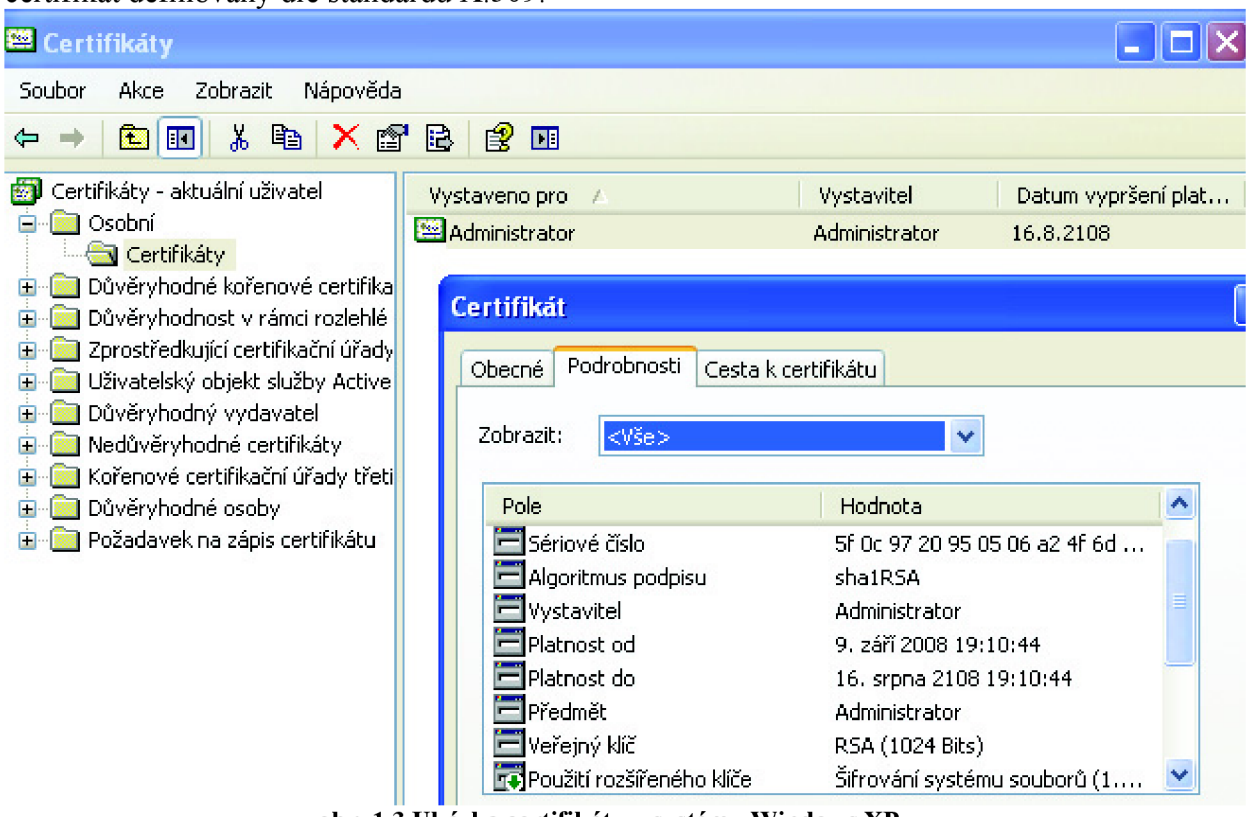
2. Dešifrovaní digitálního podpisu - Pomocí veřejného klíče odesílatele příjemce dále rozkóduje obsah digitálního podpisu. Je-li tento rozkódovaný obsah totožný s hashem přijaté zprávy, je identita odesílatele potvrzena, jelikož nikdo jiný, než vlastník soukromého klíče nemohl digitální podpis s touto vlastností vytvořit.



Obr. 1.2: Schéma verifikace zprávy pomocí digitálního podpisu

1.2 Digitální certifikát

Problémem asymetrické kryptografie je způsob, jak ověřit pravost zveřejněných veřejných klíčů. K tomu slouží ***digitální či elektronický certifikát***. Digitální certifikát je elektronická obdoba cestovního pasu nebo občanského průkazu. Jedná se v podstatě o uživatelův veřejný klíč plus další údaje popisující držitele certifikátu (jméno, bydliště, organizace apod.) To vše je zašifrováno (elektronicky podepsáno) privátním klíčem Certifikační autority, jejíž veřejný klíč je všeobecně znám a dostupný z nezaměnitelných zdrojů. V případě PKI se používá certifikát definovaný dle standardu X.509.



obr. 1.3 Ukázka certifikátu v systému Windows XP

V ČR je v rámci zákona č. 227/2000 Sb., o elektronickém podpisu a o změně některých dalších zákonů ve znění zákona č. 226/2002 Sb. definován tzv. kvalifikovaný certifikát QCA. Kvalifikovaný certifikát obsahuje identifikaci držitele certifikátu založenou na oficiální identifikaci svého držitele. Certifikační autorita vždy zná konkrétné osobu, které certifikát vydala. Bezpečnost a důvěryhodnost

CA vydávajících kvalifikované certifikáty je do jisté míry standardizována a kontrolována příslušnými úřady. V ČR se momentálně nachází tyto akreditované CA:

– První certifikační autorita, a.s. (I.CA, komerční i kvalifikovaný) - http://www.ica.cz/

– Česká pošta, s.p. (PostSignum, komerční i kvalifikovaný) - http://qca.postsignum.cz/

– eIdentity, a.s. (ACAeID, komerční i kvalifikovaný) - https://www.eidentity.cz/app

1.3 Struktura digitálního certifikátu X.509

Existuje několik norem definujících strukturu certifikátu (X.509, EDI, WAP, apod.). V prostředí Internetu se využívá norma X.509 popsaná v doporučení RFC-3280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Toto doporučení je odvozeno od ITU X.509. Původní verze standard X.509 vydaná v roce 1988 označená jako verze 1, se dnes používá v pozměněné formě verze 3 – viz. ukázka certifikátu verze 3:

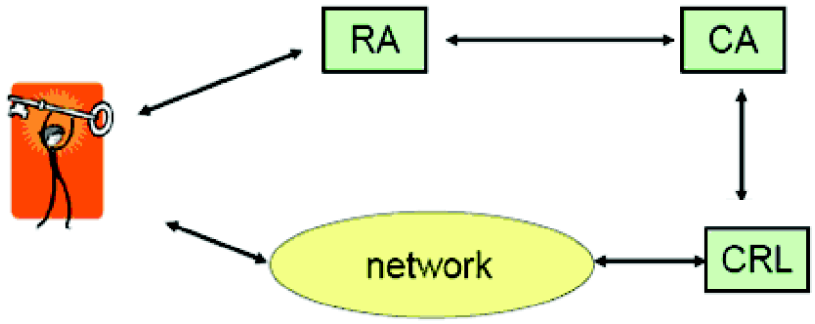
```
[root@localhost CA]# openssl x509 -noout -text -in certs/jvonasek.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 2 (0x2)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=CZ, ST=Morava, L=Brno, O=Vutbr, OU=CA,
CN=ca.pki_cviceni/emailAddress=ca@pki_cviceni
        Validity
            Not Before: May 14 11:20:55 2009 GMT
            Not After : May 14 11:20:55 2010 GMT
        Subject: C=CZ, ST=Morava, L=Brno, O=Vutbr, OU=Feec, CN=Josef
Vonasek/emailAddress=jvonasek@pki_cviceni
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:a7:78:c5:1e:88:2f:3b:76:45:67:a5:f7:fe:2d:
                    83:fc:1f:6f:e8:db:95:f9:e0:51:97:a1:4c:2c:3a:
                    3f:83:44:fd:86:1c:1a:a7:4d:4f:a7:bb:88:6c:a7:
                    8e:8f:ca:90:ba:96:f5:a2:cc:04:02:5c:44:d9:c3:
                    b9:ab:9d:fc:90:89:b6:18:49:ee:c0:07:df:b6:19:
                    16:72:55:f8:94:be:c4:a0:11:bc:6e:b1:e5:88:a3:
                    4e:33:6b:42:80:2d:24:11:75:05:a1:95:4b:ed:36:
                    7a:62:95:f0:d5:f8:f3:a2:e3:22:fa:3d:bf:7c:7c:
                    35:c9:fd:14:e7:4e:ee:0a:71
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                4B:1C:48:DE:6B:32:2C:C4:36:BB:C1:4D:9B:3C:65:65:A9:D9:C3:FA
            X509v3 Authority Key Identifier:

keyid:6B:89:85:D7:F6:13:4B:7F:B9:B4:99:52:D3:BC:BA:82:5D:77:79:94

    Signature Algorithm: sha1WithRSAEncryption
```

```
        9c:1f:14:2e:8f:f1:ab:8b:8e:00:be:b1:5a:e4:33:30:c0:de:
        15:c4:a6:ec:98:79:c9:d1:55:d7:72:24:63:d3:6b:e1:18:a4:
        f7:4d:30:f2:c4:ea:31:2a:86:4f:83:b9:be:b0:ab:d7:90:9e:
        9e:b0:44:ac:01:ad:3f:7b:1a:d9:a8:43:cb:51:f8:ab:73:42:
        4b:4c:bd:d1:ab:a0:80:eb:47:37:47:30:ca:7e:45:eb:32:74:
        c9:4d:56:4c:66:e2:22:43:c0:65:1e:f3:c2:59:b7:a9:aa:86:
        94:26:48:a6:0b:17:20:81:11:8f:40:6e:03:fc:b2:4c:e0:c6:
        9e:67
```

Význam jednotlivých hodnot certifikátu:
Version (verze)– určuje podle které normy X.509 byl příslušný certifikát vydán.
Serial Number (Sériové číslo) – musí být jednoznačné v rámci jedné CA. Spolu s *Issuer* tvoří jednoznačný identifikátor certifikátu.
Signature Algorithm (Algoritmus podpisu) – Algoritmy, které CA použila k podpisu certifikátu. V tomto případě je to hashovací funkce SHA-1 a asymetrický algoritmus RSA.
Issuer (vystavitel) + **Subject** (předmět) – obsahují položky identifikátorů objektů označovaný DN (Distinguished name). Mezi tyto objekty patří např. Country (C), Organization (O), Common name (CN),..
Validity (platnost) – každý certifikát je časově omezen (obvykle jeden rok).
Subject Public Key (veřejný klíč) – obsahuje veřejný klíč vlastníka. Je složen ze dvou základních informací - vlastní hodnoty klíče a algoritmu, kterým byl klíč tvořen.

1.4 Struktura systému PKI

Hlavními subjekty infrastruktury veřejných klíčů jsou **certifikační autorita**, **registrační autorita** a **seznam odvolaných certifikát**ů. Rolí certifikační autority v rámci bezpečné elektronické komunikace je být třetí nezávislou stranou. Nezávislost CA na komunikačních stranách jí umožňuje vystupovat v roli arbitra. Základními funkcemi CA je tedy vydávání certifikátů a seznamu zneplatněných certifikátů (CRL - Certificate Revocation List). Z toho plyne, že jádrem PKI je certifikační autorita CA. Jí mohou být podřízeny další CA a tzv. registrační autority (RA). Úkolem RA je fyzické ověření údajů žadatele o certifikát.



obr. 1.4 Struktura PKI

Certifikační autorita (CA) – certifikační autorita je subjekt, který vydává digitální certifikáty. Na základě principu přenosu důvěry tak lze za předpokladu (ověřené) důvěryhodnosti certifikační autority předpokládat i důvěryhodnost jimi vydaného (podepsaného) certifikátu. Všechny certifikáty jsou podepisovány soukromým klíčem CA. Tento klíč je tedy největším aktivem CA a je tedy nutné jej odpovídajícím způsobem chránit. Jako ochrana se proto u serverů používají tzv. HSM moduly.
Registrační autorita (RA) – úkolem RA je fyzické ověření údajů žadatele o certifikát. Jsou často realizovány podobně jako bankovní přepážky. Mohou být však realizovány i jako

servery a žadatel s nimi komunikuje elektronicky. Na RA se dostavují žadatelé o certifikáty se svými žádostmi, kde RA může ověřit na základě např. občanského průkazu jejich totožnost. RA následně zprostředkovává vydání certifikátu a následně i jeho předání žadateli.
Certificate Revocation List (CRL) - seznam zneplatněných certifikátů. Tento seznam udržuje CA a zapisuje do něj certifikáty, které byly zneplatněny ještě dříve, než je uvedeno na certifikátu (např. z důvodu kompromitace tajného klíče uživatele). CRL by měl být veřejně dostupný buďto skrze webové rozhraní, popřípadně skrze protokol LDAP.
Soukromý klíč k certifikátu je tedy cenným *aktivem*. Je tedy nutné, abychom si své soukromé klíče dobře střežili pomocí bezpečných prostředků k ukládání těchto druhů aktiv. Neopatrnost ochrany soukromého klíče lze přirovnat k podepisování bianco šeků. Jako úložiště pro tyto klíče se doporučuje používat PKI čipové karty, nebo USB tokeny. Tyto zařízení umožňují provádět veškeré kryptografické operace pomocí vlastních čipů a soukromý klíč, tak tedy nikdy neopouští tato úložiště. Přístup k němu je navíc chráněn pomocí PIN kódu.

1.5 Praktické implementace PKI

Využití infrastruktury veřejných klíčů (PKI) je obecně považováno za jednu z nejúčinnějších metod technologického zabezpečení základních atributů elektronické konfigurace (integrita, důvěrnost, autenticita, zodpovědnost) postavených na využití kryptografie.
S využitím metody PKI mohou být přenášené zprávy vybaveny při odeslání elektronickým podpisem a zašifrovány, při přijetí je zpráva dešifrována a elektronický podpis ověřen. Vedle přenosů dat a ověření zpráv je možno PKI využít i pro ověření identity uživatelů (autentizace) a pro ochranu komunikačních kanálů (SSL).
V rámci této kapitoly tedy bude pospán hlavní stavební kámen většiny praktických implementací PKI, kterým je protokol SSL/TLS. Ten je využíván protokoly různých vrstev ISO/OSI jako jsou např. protokol HTTPS (zabezpečení webových stránek), VPN (OpenVPN), SSH, FTP, SMTP, POP3 a dalších. Protokol SSL je rovněž využíván v případě autentizace uživatelů prostřednictvím PKI.

1.5.1 Zabezpečení komunikace šifrováním a autentizace komunikujících stran (SSL/TLS)

Protokol SSL byl vyvinut pro použití na webových stránkách, kdy záleží na utajení přenášených dat. Například v aplikacích internetového bankovnictví či při zadávání uživatelského jména a hesla na webových stránkách. SSL je zkratka Secure Sockets Layer. Je to protokol/vrstva, vložená mezi vrstvu transportní (např. TCP/IP) a aplikační (HTTP). Protokol SSL postupem času procházel vývojem od verze 1 po SSL verze 3. Oficiálním protokolem Internetu se však stal až protokol TLS (Transport Layer Security Protocol), také označován jako SSL verze 3.1. Z kryptografického hlediska se jedná o hybridní protokol, tj. používá symetrický i asymetrický kryptosystém. Z hlediska modelu OSI jej lze umístit do relační a prezentační vrstvy, SSL přebírá data od aplikace, v rámci prezentační vrstvy je rozdělí na segmenty, které zašifruje. Před přenosem v rámci relační vrstvy naváže spojení a provede vzájemnou autentizaci stanic. Poté využívá služeb transportní vrstvy k samotnému přenosu dat (tcp i udp na portu 443). Spojením HTTP + SSL vznikl název protokolu HTTPS.
Princip SSL spočívá ve vytvoření zabezpečeného kanálu mezi klientem (prohlížeč) a serverem. SSL umožňuje bezpečnou výměnu informací při inicializaci TCP/IP spojení, při kterém si klient a server ustavují konkrétní bezpečnostní parametry pro následný provoz a vzájemně se autentizují certifikáty. Po ustavení těchto parametrů je veškerá komunikace (http požadavky i http odpovědi) plně šifrována a to včetně URL, které klient požaduje, webových formulářů apod.

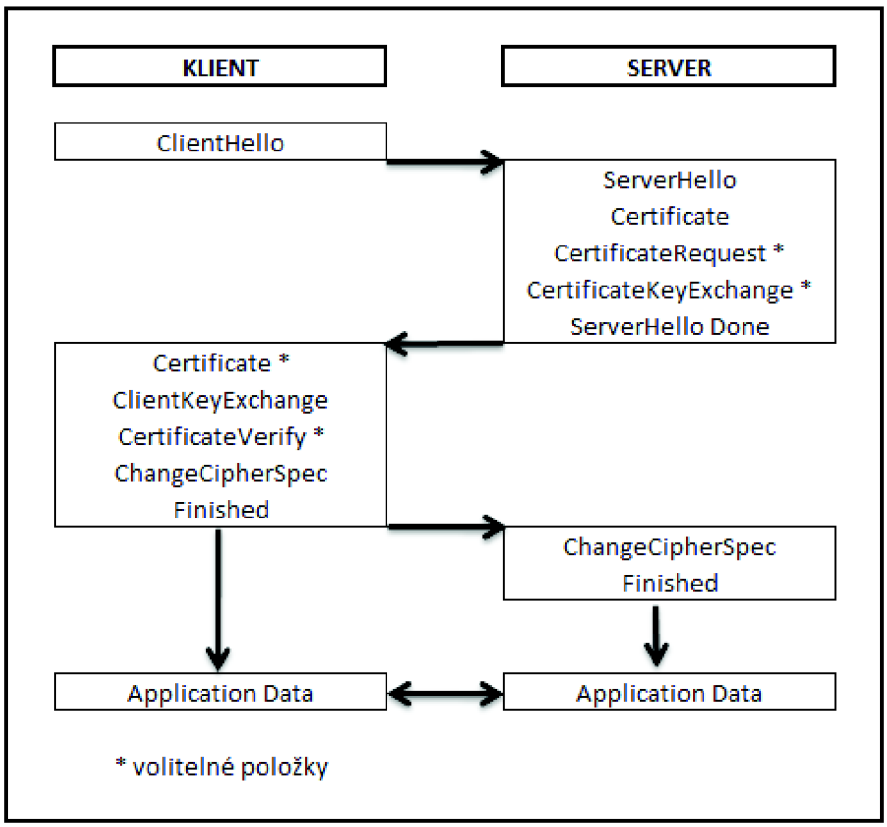
SSL protokol v sobě zahrnuje dva subprotokoly. Jedná se SSL Handshake Protocol a SSL Record Protocol. Pomocí SSL Record Protocol se definuje formát, jakým budou data přenášena a SSL Handshake Protocol zajišťuje úvodní výměnu informací.

SSL Handshake Protocol

Protokol SSL používá kombinaci symetrického a asymetrického šifrování. Ve fázi handshake se používá šifrování asymetrické, které poskytuje lepší vlastnosti pro autentizaci. Jeho hlavní nevýhodou je, že je relativně pomalé a náročné na strojový čas ve srovnání se šifrováním symetrickým. Proto je použito jen ve fázi handshake a pro další přenos se používá šifrování symetrické. SSL Handshake zajišťuje autentičnost komunikujících a vyjednání klíčů, je také nazýván key-exchange protocol. Zajišťuje ustavení bezpečné cesty (session) mezi dvěma účastníky. Připravuje tedy parametry pro Record protokol.

Fáze handshake probíhá v následujících krocích:

1. Klient odešle verzi SSL, informace o šifrách, které používá a náhodně generovaná data.
2. Server odešle svoji verzi SSL, informace o šifrách, které používá, náhodně generovaná data a svůj certifikát.
3. Klient pomocí získaného certifikátu ověří důvěryhodnost serveru, pokud tak nelze učinit, je o tom informován uživatel k rozhodnutí, zda spojení ukončit.
4. Klient vytvoří z dat dosavadní komunikace tzv. premaster secret, zašifruje ho pomocí veřejného klíče serveru získaného z certifikátu a odešle ho serveru
5. Server použije svůj privátní klíč k dešifrování premaster secret a vytvoří z něj master secret, ze kterého si vygeneruje klíč sezení
6. Klient si rovněž vytvoří master secret a z něj vygeneruje klíč sezení
7. Klient zašle zprávu serveru, že další data budou šifrována klíčem sezení a odešle zašifrované sdělení, že fáze handshake byla u něj skončena
8. Server rovněž zašle, že další data budou šifrována klíčem sezení a odešle zašifrované potvrzení, že fáze handshake byla ukončena a může být zahájen Record Protocol



obr. 1.5 Posloupnost fáze handshake

SSL Record Protocol

Record protocol provádí zapouzdření dat aplikačních protokolů - nejnižší SSL úroveň.
Z hierarchicky vyšší OSI vrstvy (např. od HTTP) je převzata zpráva Z, která je zpracována tímto postupem:

1. **Fragmentace** Z na segmenty SS o maximální délce 214 bajtů.
2. Bezeztrátová **komprimace** segmentu SS do podoby S.
3. Výpočet **autentizačního kódu** h segmentu S, kdy h = F(S, Autentizační_klíč) a F je hashovací funkce.
4. Vytvoření bloku B = S || h.
5. Zašifrování bloku B symetrickým klíčem sezení.

Zašifrovaný blok je opatřen služebními daty a předán transportní vrstvě k přenosu TCP protokolem.
Na přijímací straně se provedou inverzní operace v opačném pořadí. Příjemce zároveň kontroluje jím vypočítané autentizační kódy s přijatými.

1.5.2 Autentizace serveru pomocí SSL/TLS protokolu

SSL umožňuje i komunikaci bez jakékoliv autentizace (tzv. plně anonymní komunikace). Ta se však v praxi téměř nepoužívá. V praxi aplikace téměř vždy **vyžadují autentizaci serveru**. K autentizaci využívají server i klient certifikáty. Jestliže strana nezašle svůj certifikát druhé straně, znamená to, že se nechce autentizovat.

V případě autentizace serveru zašle server klientovi svůj certifikát, ze kterého **klient vyjme veřejný klíč** serveru, kterým pak šifruje předběžné sdílené tajemství, které následně zašle serveru. Pomocí soukromého klíče server dešifruje toto předběžné sdílené tajemství, čímž se **server zároveň i autentizuj**e (nikdo jiný než server nemá příslušný dešifrovací soukromý klíč).
V terminologii Handshake protokolu (HP) server zasílá svůj certifikát klientovi ve zprávě certifikát (*Certificate*)

1.5.3 Autentizace klienta pomocí SSL/TLS protokolu

Autentizace klienta je výrazně jednodušší. Pokud se chce klient autentizovat, zašle serveru postupně zprávy *Certificate* (obsahuje klientův certifikát) a *CertificateVerify*.
CertificateVerify zašle elektronický podpis TLS z předchozí komunikace protokolem HP, který mj. obsahuje náhodná čísla generovaná klientem i serverem (*client_random* a *server_random*).
Server následně provede verifikaci tohoto elektronického podpisu TLS (nikdo jiný než klient nemá k dispozici soukromý klíč, kterým byl elektronický podpis TLS vytvořen).

1.5.4 Bezpečnost elektronické pošty E-mail

Standardizovaný protokol pro bezpečnost elektronické pošty je označován jako *S/MIME* (Secure Multipurpose Internet Mail Extensions). Je definován v RFC-2632 až RFC-2634, respektive RFC-3851. Nedílnou součástí tohoto protokolu je podpora pro kryptografické operace (hash, symetrická i asymetrická kryptografie) využívající i certifikáty a CRL. Protokol umožňuje zasílat elektronicky podepsaná data (zajištěna integrita a nepopiratelnost), tak zašifrovaná a uzavřená v elektronické obálce.
Podle standardu S/MIME musí být adresa elektronické pošty uvedena v použitém certifikátu, avšak certifikát může obsahovat i více poštovních adres najednou. Aplikace odesilatele i příjemce pak mj. kontrolují shodu adresy uvedené v certifikátu a ve zprávě.
Vlastní použití bezpečné elektronické pošty po instalaci certifikátu do aplikace je jednoduché. Většina grafických poštovních aplikací umožňuje podepsat, nebo zašifrovat zprávu jednoduše pouhým kliknutím na příslušnou ikonu v průběhu komponování zprávy. Při podepisování zprávy potřebuje aplikace soukromý klíč odesilatele. Složitější je to v případě šifrování zprávy, kdy aplikace potřebuje veřejný klíč příjemce. Jednou možností jak jej získat je **stáhnout certifikát příjemce z webových stránek** certifikační autority, což může být často problém, jelikož neexistuje žádná povinnost zveřejňování certifikátů.
Druhou a jednodušší možností získání certifikátu je jednoduše napsat plánovanému příjemci **podepsanou zprávu se žádostí o certifikát**. Příjemce poté může pomocí veřejného klíče odesilatele, který získal z přijatého certifikátu zašifrovat svůj certifikát a odeslat jej zpět odesilateli. Popřípadě může příjemce původnímu odesilateli odpovědět opět jen nešifrovanou, avšak vlastním soukromým klíčem podepsanou zprávou obsahující jeho certifikát s veřejným klíčem. Pro následnou další komunikaci je vhodné, aby si oba účastníci uložili veřejné klíče k příslušným kontaktům v adresáři. Některé poštovní aplikace umožňují toto ukládání automaticky (např. MS Office Outlook), u některých (např. Outlook Express) je třeba tuto funkci nejdříve povolit.

2 Vlastní zadání laboratorní úlohy:

1. Pomocí OpenSSL nainstalujte vlastní certifikační autoritu
2. Zabezpečte webový server protokolem SSL/TLS. Pro generování klíčů i žádosti o certifikát použijte opět OpenSSL. Žádost o certifikát bude podepsána prostřednictvím Vámi nainstalované certifikační autority.
3. Pro uživatele vytvořte pár klíčů, certifikát veřejného klíče a nastavte webový server tak, aby přístup na server prostřednictvím HTTPS protokolu byl umožněn pouze na základě autentizace uživatele pomocí certifikátu a jeho klíče.

Většina distribucí systému Linux dnes standardně obsahuje všechny potřebné balíčky pro zprovoznění webového serveru a certifikační autority – apache (popř. httpd), openssl a mod_ssl. Tento návod je specifikován pro předpřipravenou distribuci CentOS 5.2 (už.jméno root, heslo „password“). V jiných distribucích mohou být menší odlišnosti např. v umístění konfiguračních souborů.

V první fázi je nutné ověřit, zdali jsou všechny balíčky nainstalovány.

```
[root@localhost ~]# rpm -q apache httpd openssl mod_ssl
balíček apache není nainstalován
httpd-2.2.3-22.el5.centos
openssl-0.9.8e-7.el5
mod_ssl-2.2.3-22.el5.centos
```

V případě, že některý z balíčku chybí, lze je doinstalovat buďto pomocí *rpm –i* nebo příkazem *yum install*.

```
[root@localhost ~]# [root@localhost ~]# yum install mod_ssl

Loading mirror speeds from cached hostfile
 * base: centos.politechnika.lublin.pl
 * updates: centos.politechnika.lublin.pl
 * addons: centos.politechnika.lublin.pl
 * extras: centos.politechnika.lublin.pl
Setting up Install Process
Parsing package install arguments
Package  1:mod_ssl-2.2.3-22.el5.centos.i386  already  installed  and  latest
version
Nothing to do
```

V tomto případě je již balíček nainstalován, proto „Nothing to do“

2.1 Vytvoření certifikační autority

Prvním krokem bude příprava prostředí pro umístění CA, klíčů, žádostí, certifikátů, apod. Standardně je v distribucich CentOS OpenSSL nainstalovano v **/etc/pki/CA** Změníme tedy pracovní adresář do tohoto umístění.

```
[root@localhost ~]# cd /etc/pki/CA
```

Vytvoříme adresář, ve kterém budou ukládány certifikáty

```
[root@localhost CA]# mkdir certs
```

Vytvoříme adresář, ve kterém budou ukládány odvolané certifikáty (CRL)

```
[root@localhost CA]# mkdir crl
```

Vytvoříme adresář, ve kterém budou ukládány serverové certifikáty ve formátu PEM

```
[root@localhost CA]# mkdir newcerts
```

Dále vytvoříme soubor index.txt, který je CA používán pro údržbu databáze certifikátů a musí být vytvořen prázdný.

```
[root@localhost CA]# touch index.txt
```

Každý certifikát vydaný touto CA obsahuje sériové číslo. Přídáme tedy text 01 do souboru **/etc/pki/CA/serial a /etc/pki/CA/crlnumber** . Tyto soubory používá CA pro přiřazování sériových čísel.

```
[root@localhost CA]# echo "01" > serial
[root@localhost CA]# echo "01" > crlnumber
```

Hlavním konfiguračním souborem OpenSSL je soubor openssl.cnf . Ten se standardně nachází v **/etc/pki/tls/** a obsahuje základní defaultní nastavení. Abychom nemuseli pro naši CA vytvářet tento konfigurační soubor celý ručně, zkopírujeme si tento defaultně nastavený openssl.cnf do adresáře s naší CA.

```
[root@localhost CA]# cp /etc/pki/tls/openssl.cnf openssl.cnf
```

Pomocí textového editoru jednoduše upravíme tento soubor a to především nastavíme správné cesty k adresářům, které jsou definovány v sekci [CA_default]. Jelikož se openssl.cnf nachází ve stejném adresáři, kde se budou nacházet i klíče a certifikáty, je vhodné změnit defaultně nastavenou položku dir na aktuální adresář. Dále nastavíme cestu i k námi vytvořenému klíči, CRL a certifikátu v položkách certificate, crl a private_key.

```
[ CA_default ]

dir             = .                # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
#unique_subject = no                    # Set to 'no' to allow creation of
                                        #  several  ctificates  with  same
subject.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/certs/ca.crt       # The CA certificate
serial          = $dir/serial           # The current serial number
crlnumber       = $dir/crlnumber        # the current crl number
                                        # must be commented out to leave a
V1 CRL
crl             = $dir/crl/ca.crl         # The current CRL
private_key     = $dir/private/ca.key    # The private key
RANDFILE        = $dir/private/.rand    # private random number file
```

Z hlediska bezpečnosti je vhodné znemožnit jakýkoliv přístup k tomuto souboru komukoli jinému, kromě vlastníka.

```
chmod 0600 openssl.cnf
```

Nyní je již vše připraveno a můžeme pomocí příkazu openssl vygenerovat klíče a certifikát pro tuto certifikační autoritu.

```
[root@localhost CA]# openssl req -config openssl.cnf -new -x509 -extensions
v3_ca -keyout private/ca.key -out certs/ca.crt -days 3650
Generating a 1024 bit RSA private key
..............++++++
.........++++++
writing new private key to 'private/ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:CZ
State or Province Name (full name) [Berkshire]:Morava
Locality Name (eg, city) [Newbury]:Brno
Organization Name (eg, company) [My Company Ltd]:Vutbr
Organizational Unit Name (eg, section) []:CA
Common Name (eg, your name or your server's hostname) []:ca.pki_cviceni
Email Address []:ca@pki_cviceni
```

Tento příkaz vytváří dle nastavení v openssl.cnf soukromý klíč certifikační autority ca.key a zároveň vytváří její „self-signed“ certifikát ca.crt ve formátu X.509, jehož platnost je nastavena na 3650 dni (10let).

Také zabezpečíme soukromý klíč serveru, aby jej mohl číst pouze root.

```
[root@localhost CA]# chmod 0400 private/ca.key
```

2.2 Zabezpečení webového serveru

2.2.1 Vytvoření certifikátu pro webový server

Obdobným způsobem jako se vytvářely klíče a certifikát pro CA nyní vygenerujeme certifikát a klíče pro webový server.

```
[root@localhost CA]# openssl req -config openssl.cnf -new -nodes -keyout
private/pki_cviceni.key -out pki_cviceni.csr -days 365
Generating a 1024 bit RSA private key
............++++++
............++++++
writing new private key to 'private/pki_cviceni.key'
-----
```

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:CZ
State or Province Name (full name) [Berkshire]:Morava
Locality Name (eg, city) [Newbury]:Brno
Organization Name (eg, company) [My Company Ltd]:Vutbr
Organizational Unit Name (eg, section) []:Secure Web Server
Common Name (eg, your name or your server's hostname) []:192.168.0.28
Email Address []:ca@pki_cviceni

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: [ENTER]
An optional company name []: [ENTER]
```

Tímto příkazem jsme vygenerovali soukromý klíč, který bude používán webovým serverem k dešifrování komunikace přijaté od webových klientů. Také jsme současně vytvořili žádost o certifikát pki_cviceni.csr s platností 1rok , která je uložena ve složce CA. Dalším krokem tedy bude podepsání této žádosti a vydání certifikátu.

```
[root@localhost CA]# openssl ca -config openssl.cnf -policy policy_anything
-out certs/pki_cviceni.crt -infiles pki_cviceni.csr
Using configuration from openssl.cnf
Enter pass phrase for ./private/ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 1 (0x1)
        Validity
            Not Before: May 13 14:33:01 2009 GMT
            Not After : May 13 14:33:01 2010 GMT
        Subject:
            countryName               = CZ
            stateOrProvinceName       = Morava
            localityName              = Brno
            organizationName          = Vutbr
            organizationalUnitName    = Secure Web Server
            commonName                = 192.168.0.28
            emailAddress              = ca@pki_cviceni
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                11:3D:43:E7:B0:DF:FA:C4:70:06:8D:80:AC:A0:64:D6:F5:41:F4:7C
            X509v3 Authority Key Identifier:

keyid:6B:89:85:D7:F6:13:4B:7F:B9:B4:99:52:D3:BC:BA:82:5D:77:79:94

Certificate is to be certified until May 13 14:33:01 2010 GMT (365 days)
Sign the certificate? [y/n]:y
```

```
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Nyní jsme pomocí soukromého klíče CA (samozřejmě po zadání jeho hesla) podepsali požadavek a vytvořili tak certifikát pro webový server pki_cviceni.crt

Posledním krokem je vygenerování a podepsání seznamu odvolaných certifikátů (CRL)

```
[root@localhost CA]# openssl ca -config openssl.cnf -gencrl -out crl/ca.crl
Using configuration from openssl.cnf
Enter pass phrase for ./private/ca.key:
```

2.2.2 Propojení webového serveru s CA

Posledním nezbytným krokem je propojení webového serveru s připravenou certifikační autoritou. Nejdříve si ověříme, zdali webový server je opravdu spuštěný

```
[root@localhost CA]# service httpd status
httpd (pid 3556 3555 3554 3553 3552 3551 3550 3549 3514) běží...
```

Propojení webového serveru s CA zajišťuje nádstavbový modul mod_ssl. Jeho nastavení je v distribuci CentOS standardně uloženo v souboru /etc/httpd/conf.d/ssl.conf
Standardně je komunikaci prostřednictvím SSL/TLS povolena, stačí tedy v konfiguračním souboru nastavit cestu k veřejnému (SSLCertificateFile) a soukromému (SSLCertificateKeyFile) klíči serveru. Pomocí editoru tedy upravíme následujicí položky v ssl.conf

```
[root@localhost CA]# cd /etc/httpd/conf.d
[root@localhost conf.d]# vi ssl.conf
```

Upravit cestu ke klíčům u následujících položek:

```
SSLCertificateFile /etc/pki/CA/certs/pki_cviceni.crt
```

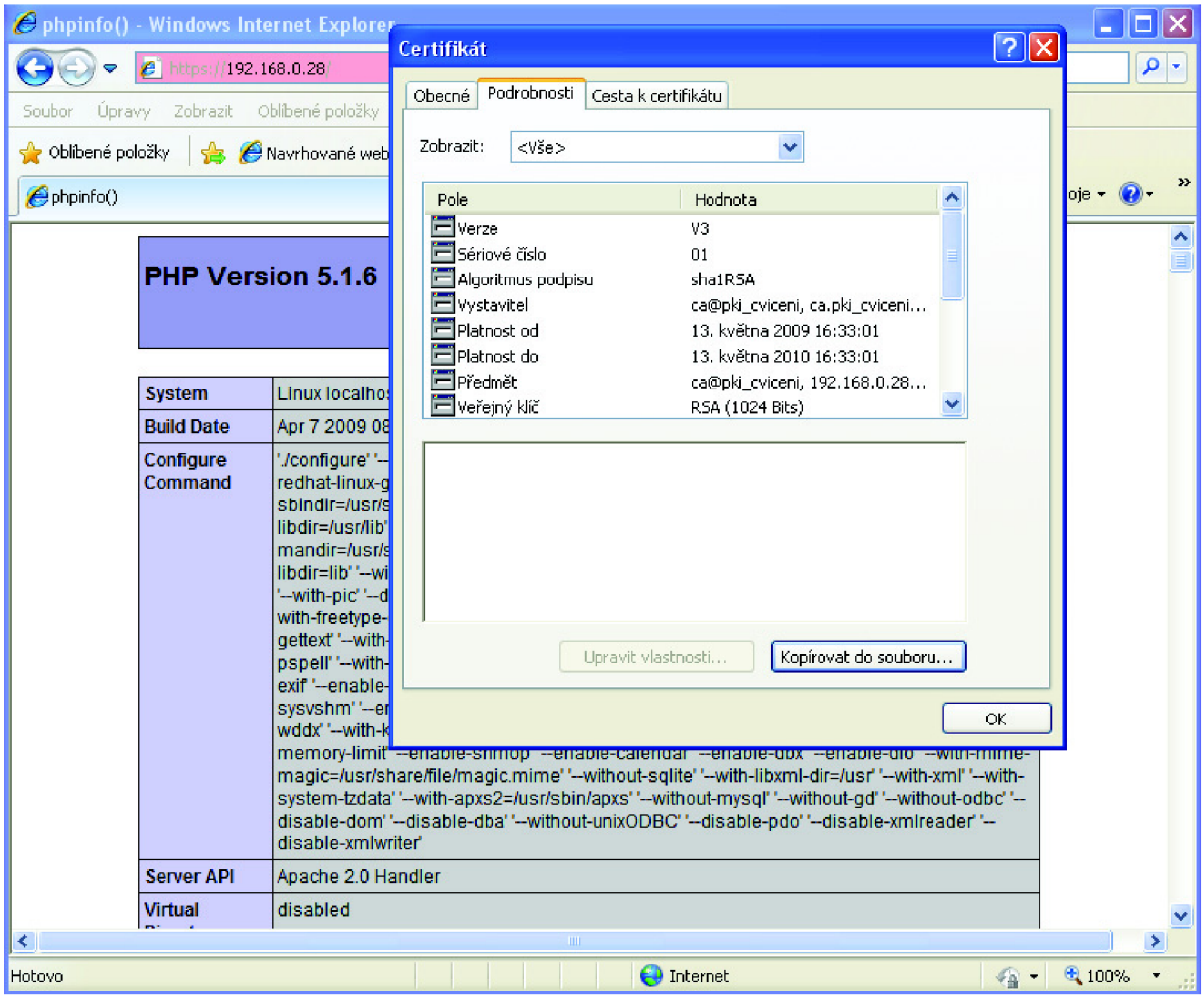
```
SSLCertificateKeyFile /etc/pki/CA/private/pki_cviceni.key
```

Posledním krokem je zrestartování httpd démona s novým nastavením.

```
[root@localhost conf.d]# service httpd restart
Ukončuji httpd:                                            [  OK  ]
Spouštím httpd:                                            [  OK  ]
```

V tuto chvíli by měl být webový server připraven obsluhovat požadavky na zabezpečenou komunikaci. Funkčnost ověříme přístupem na vlastní webové stránky přes protokol https.

Pozor! Vzhledem k tomu, že certifikát byl podepsán vlastní certifikační autoritou, která je samozřejmě pro prohlížeče nedůvěryhodná, bude prohlížeč upozorňovat na problém s nedůvěryhodným certifikátem. Řešením je buďto certifikát dočasně přijmout, nebo přidat vydávající CA do důvěryhodných certifikačních autorit.



2.3 Řízení přístupu k webovým stránkám pomocí PKI

2.3.1 Vytvoření páru klíčů a certifikát veřejného klíče pro uživatele

Z hlediska bezpečnosti je nezbytné, aby si každý uživatel generoval pár klíčů se žádostí o certifikát sám prostřednictvím vlastního PC a k registrační autoritě se dostavil jen s žádostí o certifikát. Tím se zamezuje případnému získání jeho soukromého soukromého klíče třetí osobou. V praxi se toto generování provádí buďto prostřednictvím webové aplikace napojené přímo na RA, na níž je po vygenerování klíčů přímo odeslána žádost o certifikát, nebo ještě lépe prostřednictvím USB tokenů, nebo čipových karet. Výrobci těchto zařízení dodávají spolu s tímto hardware zpravidla i software k jejich správě. Pomocí tohoto softwaru si pak uživatel může vygenerovat pár klíčů, přičemž soukromý klíč **nikdy** neopouští úložiště těchto zařízení a všechny kryptografické operace probíhají přímo prostřednictvím jejích čipů.
V rámci demonstrace si však pro výukové účely vystačíme s generováním klíčů prostřednictvím serveru a jejich následným exportem do formátu p12, speciálně určeném právě pro tyto účely.

V první řadě si pomocí již známého příkazu vygenerujeme pár klíčů spolu s žádostí o certifikát s platností 1 rok.

```
[root@localhost CA]# openssl req -config openssl.cnf -new -nodes -keyout private/jvonasek.key -out jvonasek.csr -days 365
```

```
Generating a 1024 bit RSA private key
....++++++
......++++++
writing new private key to 'private/jvonasek.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:CZ
State or Province Name (full name) [Berkshire]:Morava
Locality Name (eg, city) [Newbury]:Brno
Organization Name (eg, company) [My Company Ltd]:Vutbr
Organizational Unit Name (eg, section) []:Feec
Common Name (eg, your name or your server's hostname) []:Josef Vonasek
Email Address []:jvonasek@pki_cviceni

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Stejně jako v případě žádosti o certifikát pro webový server tuto žádost prostřednictvím soukromého klíče CA podepíšeme a tento certifikát uložíme k ostatním certifikátům do adreáře certs.

```
[root@localhost CA]# openssl ca -config openssl.cnf -policy policy_anything
-out certs/jvonasek.crt -infiles jvonasek.csr   Using configuration from
openssl.cnf
Enter pass phrase for ./private/ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 2 (0x2)
        Validity
            Not Before: May 14 11:20:55 2009 GMT
            Not After : May 14 11:20:55 2010 GMT
        Subject:
            countryName               = CZ
            stateOrProvinceName       = Morava
            localityName              = Brno
            organizationName          = Vutbr
            organizationalUnitName    = Feec
            commonName                = Josef Vonasek
            emailAddress              = jvonasek@pki_cviceni
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                4B:1C:48:DE:6B:32:2C:C4:36:BB:C1:4D:9B:3C:65:65:A9:D9:C3:FA
            X509v3 Authority Key Identifier:

keyid:6B:89:85:D7:F6:13:4B:7F:B9:B4:99:52:D3:BC:BA:82:5D:77:79:94

Certificate is to be certified until May 14 11:20:55 2010 GMT (365 days)
Sign the certificate? [y/n]:y
```

```
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Certifikát X509 zobrazíte pro kontrolu v textové podobě následujícím příkazem:

```
[root@localhost CA]# openssl x509 -noout -text -in certs/jvonasek.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 2 (0x2)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer:     C=CZ,     ST=Morava,     L=Brno,     O=Vutbr,     OU=CA,
CN=ca.pki_cviceni/emailAddress=ca@pki_cviceni
        Validity
            Not Before: May 14 11:20:55 2009 GMT
            Not After : May 14 11:20:55 2010 GMT
        Subject:  C=CZ,  ST=Morava,  L=Brno,  O=Vutbr,  OU=Feec,  CN=Josef
Vonasek/emailAddress=jvonasek@pki_cviceni
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:a7:78:c5:1e:88:2f:3b:76:45:67:a5:f7:fe:2d:
                    83:fc:1f:6f:e8:db:95:f9:e0:51:97:a1:4c:2c:3a:
                    3f:83:44:fd:86:1c:1a:a7:4d:4f:a7:bb:88:6c:a7:
                    8e:8f:ca:90:ba:96:f5:a2:cc:04:02:5c:44:d9:c3:
                    b9:ab:9d:fc:90:89:b6:18:49:ee:c0:07:df:b6:19:
                    16:72:55:f8:94:be:c4:a0:11:bc:6e:b1:e5:88:a3:
                    4e:33:6b:42:80:2d:24:11:75:05:a1:95:4b:ed:36:
                    7a:62:95:f0:d5:f8:f3:a2:e3:22:fa:3d:bf:7c:7c:
                    35:c9:fd:14:e7:4e:ee:0a:71
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                4B:1C:48:DE:6B:32:2C:C4:36:BB:C1:4D:9B:3C:65:65:A9:D9:C3:FA
            X509v3 Authority Key Identifier:

keyid:6B:89:85:D7:F6:13:4B:7F:B9:B4:99:52:D3:BC:BA:82:5D:77:79:94

    Signature Algorithm: sha1WithRSAEncryption
        9c:1f:14:2e:8f:f1:ab:8b:8e:00:be:b1:5a:e4:33:30:c0:de:
        15:c4:a6:ec:98:79:c9:d1:55:d7:72:24:63:d3:6b:e1:18:a4:
        f7:4d:30:f2:c4:ea:31:2a:86:4f:83:b9:be:b0:ab:d7:90:9e:
        9e:b0:44:ac:01:ad:3f:7b:1a:d9:a8:43:cb:51:f8:ab:73:42:
        4b:4c:bd:d1:ab:a0:80:eb:47:37:47:30:ca:7e:45:eb:32:74:
        c9:4d:56:4c:66:e2:22:43:c0:65:1e:f3:c2:59:b7:a9:aa:86:
        94:26:48:a6:0b:17:20:81:11:8f:40:6e:03:fc:b2:4c:e0:c6:
        9e:67
```

V tuto chvíli je tedy certifikát s klíči připraven pro další práci. Abychom jej mohli použít v hostitelském PC, je nutné pár klíčů spolu s podepsaným certifikátem vyexportovat do formátu pkcs#12. Vzhledem k tomu, že soubor bude obsahovat i soukromý klíč, je nutné práci s ním zabezpečit autentizací heslem. Exportovaný soubor bude uložen v adresáři export.

```
[root@localhost CA]# mkdir export
[root@localhost CA]# openssl pkcs12 -export -in certs/jvonasek.crt -inkey
private/jvonasek.key -out export/jvonasek.p12
Enter Export Password:
Verifying - Enter Export Password:
```

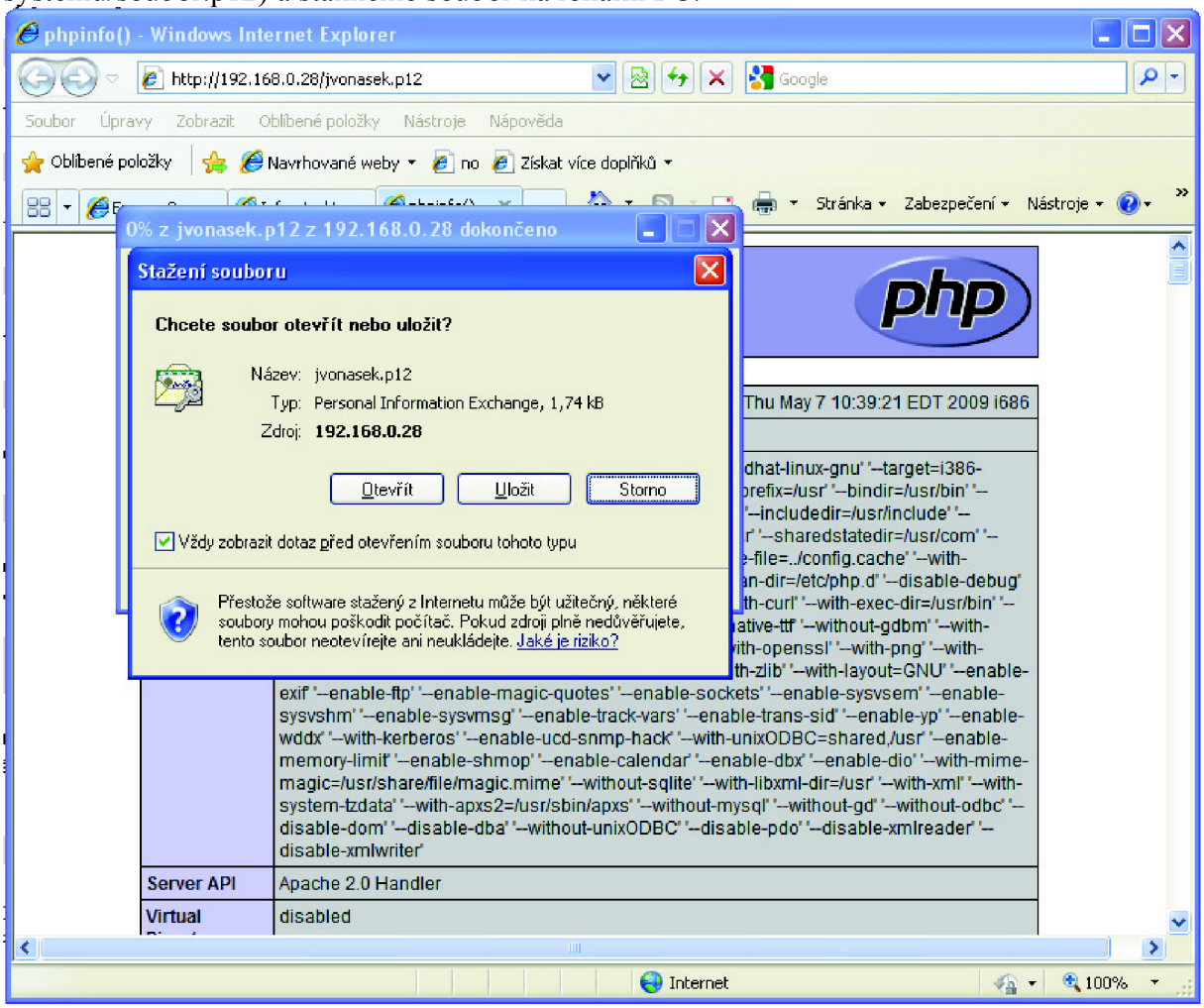
Nyní je již soukromý klíč spolu s certifikátem obsahujícím veřejný klíč připravený k exportu do jiného PC. Soubor jvonasek.p12 nyní přeneseme do hostitelského operačního systému dle možností buďto pomocí flash disku, nebo prostřednictvím ftp. Pokud nemáte jinou možnost je možné soubor překopírovat do adresáře, kde se nachází webové stránky a odtud jej pak stáhnout prostřednictvím http protokolu.

```
[root@localhost CA]# cp /etc/pki/CA/export/jvonasek.p12
/var/www/html/jvonasek.p12
```

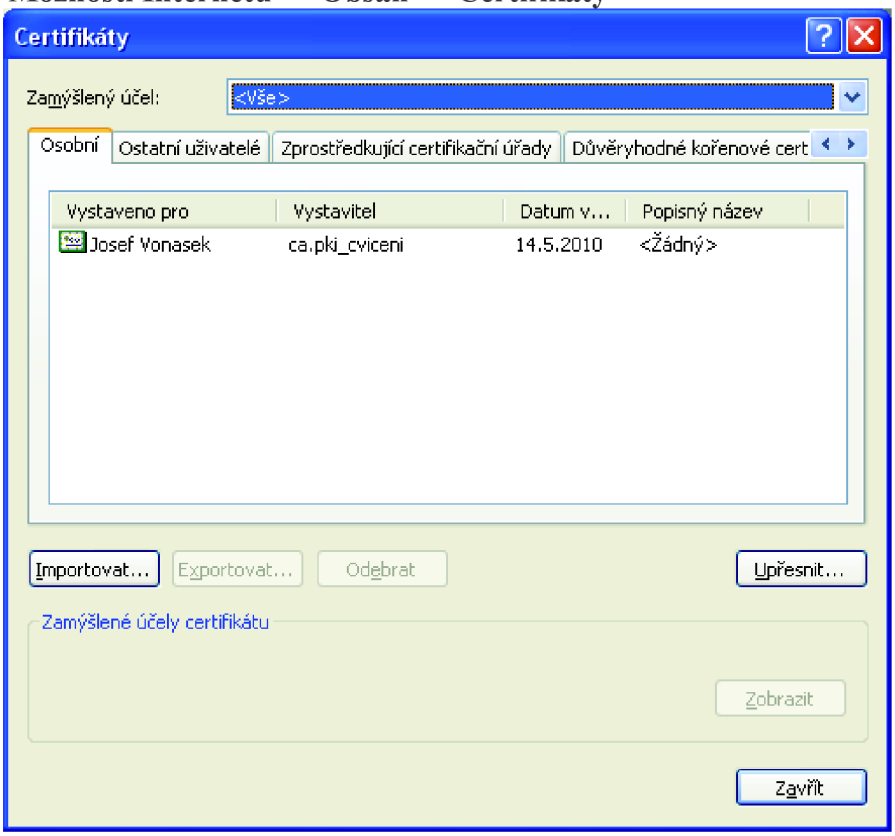
Současně si zpřístupníme i certifikát kořenové CA, abychom ji mohli na klientském PC přidat mezi důvěryhodné certifikační autority čímž se do budoucna zabrání při přístupu přes https protokol varování, že server používá nedůvěryhodný certifikát.

```
[root@localhost CA]# cp /etc/pki/CA/certs/ca.crt /var/www/html/ca.crt
```

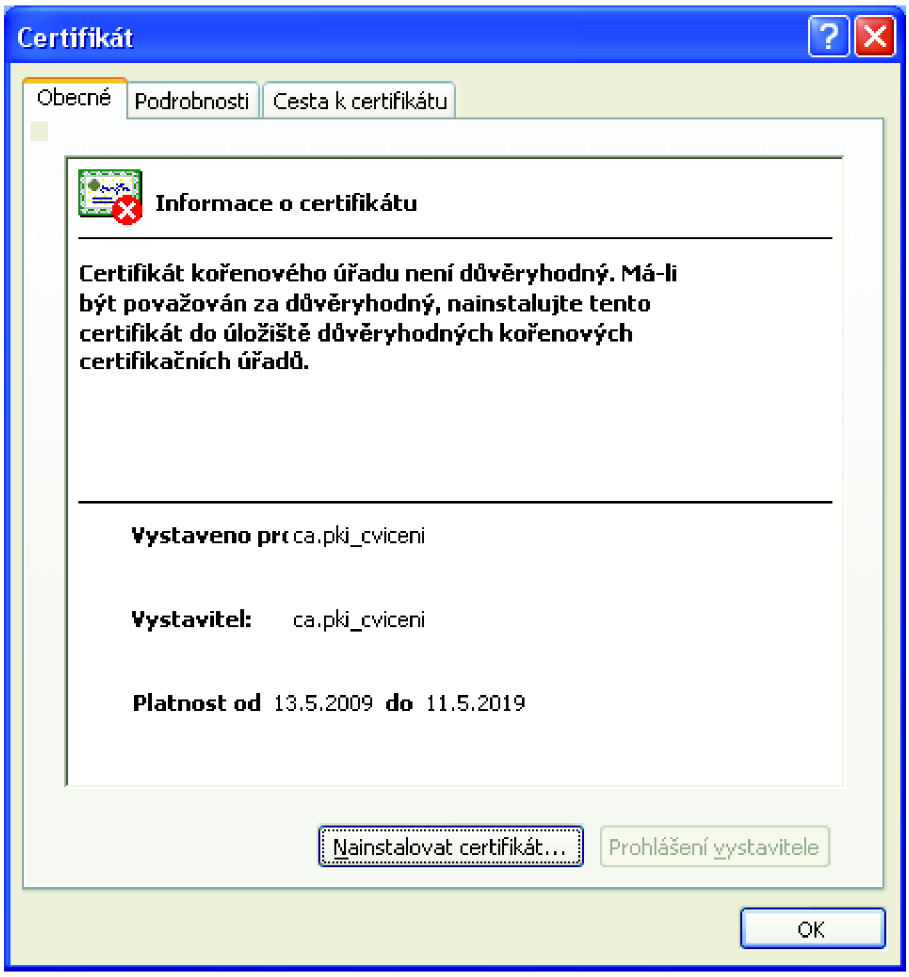
Nyní na moment opustíme virtuální systém a přepneme se do hostitelských Windows. Do prohlížeče zadáme přímou cestu k uloženému certifikátu (IP adresu virtuálního systému/soubor.p12) a stáhneme soubor na lokální PC.



Po stažení a otevření souboru se spustí Průvodce importem certifikátu. Pokračujeme tlačítkem **další**, potvrdíme umístění souboru **další**. V následujícím okně zadáme heslo k soukromému klíči, které jsme zvolili v průběhu exportu a zatrhneme Povolit silnou ochranu soukromého klíče. Pokud vyžadujeme, aby při každém použití soukromého klíče bylo požadováno ověření tohoto hesla, zvolíme i možnost „povolit silnou ochranu klíče“. Posledním krokem je volba úložiště certifikátu, do kterého bude certifikát naimportován. V tomto případě můžeme ponechat automatickou volbu, systém Windows rozpozná, že se jedná o osobní certifikát.
Úspěšné naimportování certifikátu ověříme spuštěním aplikace Internet Explorer v nabídce **Nástroje → Možnosti Internetu → Obsah → Certifikáty**



Nyní ještě přidáme kořenovou certifikační autoritu mezi důvěryhodné certifikační autority. Do prohlížeče zadáme přímou cestu k uloženému certifikátu (IP adresu virtuálního systému/ca.crt) a stáhneme soubor na lokální PC. Po spuštění certifikátu zvolíme možnost Nainstalovat certifikát a na závěr potvrdíme, že této certifikační autoritě opravdu důvěřujeme.



2.3.2 Řízení přístupu k webovému serveru

Nastavení přístupu k webovému serveru je stejně jako nastavení samotného zabezpečení webového serveru protokolem https uloženo v souboru /etc/httpd/conf.d/ssl.conf
Řízení přístupu zajišťuje direktiva *SSLVerifyClient*, dále je nezbytné nastavit cestu k CA, která podepisuje certifikáty uživatelům, kteří budou přistupovat k serveru a také cestu k seznamu zneplatněných certifikátů CRL.

```
[root@localhost CA]# cd /etc/httpd/conf.d
[root@localhost conf.d]# vi ssl.conf
```

```
SSLCACertificateFile /etc/pki/CA/certs/ca.crt //odkomentovat a změnit cestu
SSLCARevocationFile /etc/pki/CA/crl/ca.crl     //doplnit
```

```
SSLVerifyClient require
SSLVerifyDepth  1
```

Nyní jsme nastavili cesty k CA a CRL a nastavili nutnost autentizace certifikátem při spojení pomocí protokolu https. Direktiva *SSLVerifyDepth* hodnotou 1 určuje, že certifikát klienta bude ověřováň pouze u nejbližší certifikační autority, kterou server zná (SSLCACertificateFile).

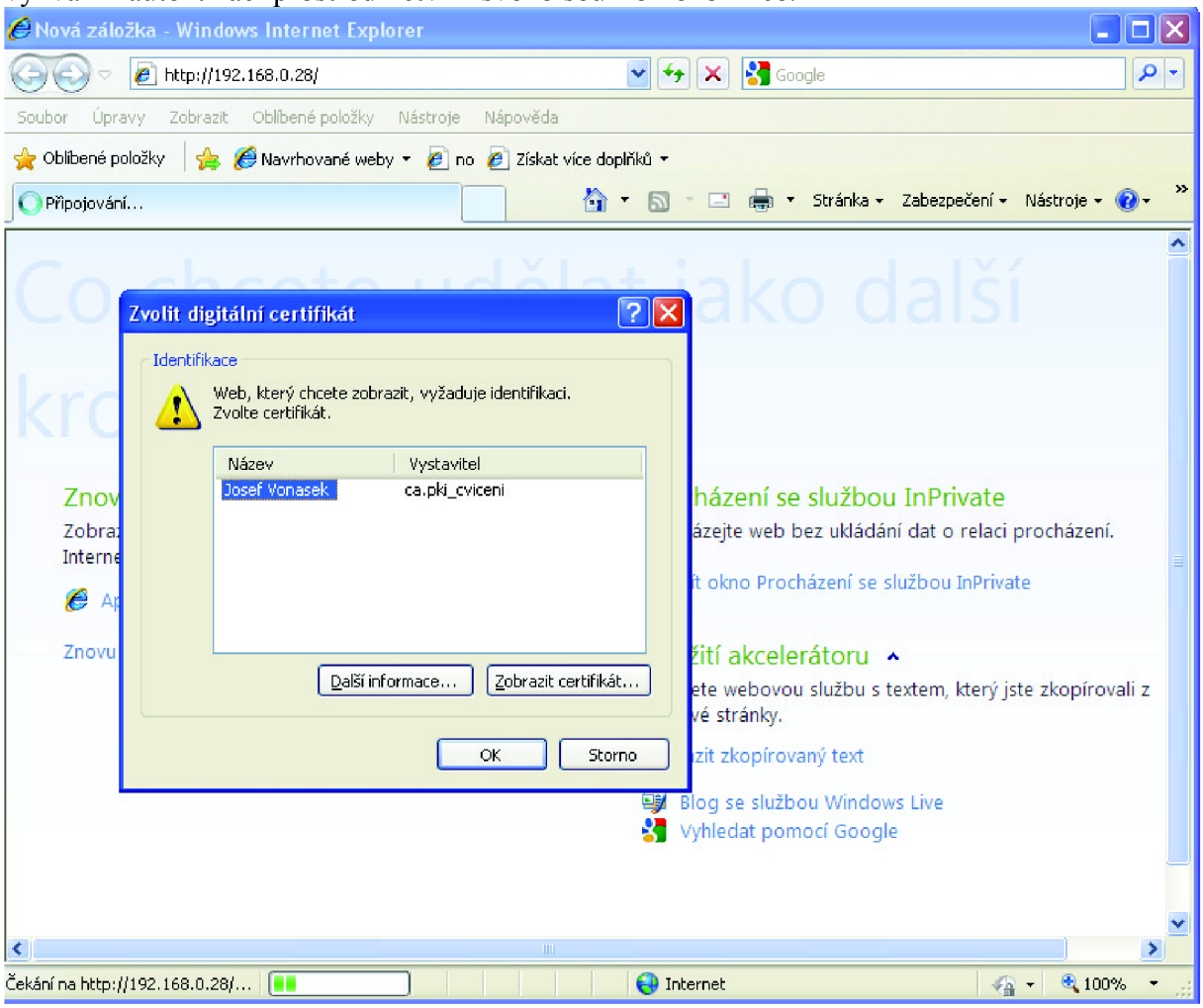
Pokud budete požadovat, aby přístup na stránky byl umožněn výhradně pomocí zabezpečeného protokolu https a nikoliv protokolu http (80), je nezbytné nastavit pravidlo přepisující všechny http požadavky, na požadavky https na portu 443. Toto umožňuje modul mod_rewrite. Ten je možno nastavit přímo v httpd.conf přidáním následujících direktiv.

```
<IfModule mod_rewrite.c>
      RewriteEngine on
      RewriteCond %{SERVER_PORT} !^443$
      RewriteRule ^.*$ https://%{SERVER_NAME}%{REQUEST_URI} [L,R]
</IfModule>
```

Po těchto změnách nastavení je opět nutné zrestartovat httpd démona, aby se tyto změny aplikovaly.

2.4 Ověření korektní instalace

V tuto chvíli tedy webový server obsluhuje všechny požadavky pouze prostřednictvím SSL/TLS protokolu a současně je vstup na webové stránky umožnen pouze držitelům certifikátu podepsaným námi nainstalovanou certifikační autoritou.
Z klientské stanice nyní zadáme IP adresu webového serveru. Požadavek bude ihned přesměrován na port 443 a obsloužen zabezpečeným https protokolem. Současně je uživatel vyzván k autentizaci prostřednictvím svého soukromého klíče.



PŘÍLOHA B: OBSAH PŘILOŽENÉHO DVD

Navod /
- Laboratorni_uloha.doc
- Laboratorni_uloha.pdf

VMWare IMG /
- CentOS_5_2.nvram
- CentOS_5_2.vmdk
- CentOS_5_2.vmsd
- CentOS_5_2.vmx
- CentOS_5_2.vmxf

WWW /
- **images /**
- **linux /**
- **obecne /**
- **server2008 /**
- **vyuziti /**
- default.css
- index.php
- linux.php
- obecne.php
- odkazy.php
- server2008.php
- vyuziti.php

/
- DP_LabUloha_PKI.pdf