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

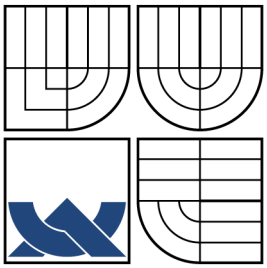
INFRASTRUKTURA VEŘEJNÝCH KLÍČŮ

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

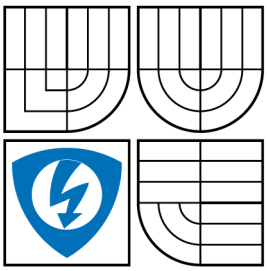
AUTOR PRÁCE
AUTHOR

BC. ONDŘEJ BĚDAJÁNEK

BRNO 2008



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

INFRASTRUKTURA VEŘEJNÝCH KLÍČŮ
TITLE OF STUDENT'S THESIS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — BC. ONDŘEJ BĚDAJÁNEK
AUTHOR

VEDOUCÍ PRÁCE — ING. RADIM PUST
SUPERVISOR

BRNO 2008

ZDE VLOŽIT LIST ZADÁNÍ

Z důvodu správného číslování stránek

ZDE VLOŽIT PRVNÍ LIST LICENČNÍ SMOUVY

Z důvodu správného číslování stránek

ZDE VLOŽIT DRUHÝ LIST LICENČNÍ SMOUVY

Z důvodu správného číslování stránek

ABSTRAKT

Diplomová práce se zabývá infrastrukturou veřejných klíčů, ve které jsou popsány principy a funkce certifikační autority. V rámci diplomové práce, byla vytvořena vlastní certifikační autorita pod operačním systémem Linux. Pro vytváření, vydávání a zneplatňování certifikátů bylo naprogramováno webové rozhraní v PHP.
V diplomové práci jsou popsány konfigurační soubory pro OpenVPN, které jsou následně využity při zabezpečení bezdrátové sítě WiFi.

KLÍČOVÁ SLOVA

pki, certifikační autorita, registrační autorita, digitální podpis, openvpn, infrastruktura veřejných klíčů, wifi, linux, apache, php, kryptografie, symetrické šifrování, asymetrické šifrování

ABSTRACT

The subject of my thesis dscribes function and principles of the public key infrastructure as well as certificate authority. Under the operation system Linux was created self signed certificate authority. Web interface was devoloped in PHP for the purpose of the generation, distribution and rejection certificates.
Configuration files for OpenVPN are included in the thesis and wireless security is achived by OpenVPN.

KEYWORDS

pki, certificate authority, registration authority, digital signature, openvpn, public key infrastructure, wifi, linux, apache, php, cryptography, symetric encryption, asymetric encryption

BĚDAJÁNEK, O. *Infrastruktura veřejných klíčů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. Počet stran 67. Počet stran příloh 2. Vedoucí diplomové práce Ing. Radim Pust.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Infrastruktura veřejných klíčů“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce 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 této diplomové práce 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 ....

(podpis autora)

Poděkování

Chtěl bych poděkovat svému vedoucímu Ing. Radimu Pustovi za příkladné vedení při psaní diplomové práce v průběhu celého semestru. Jeho odborné rady mi byly nápomocí při návrhu a řešení struktury diplomové práce.

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

V současné době je na vzestupu používání bezdrátových technologií pro přístup koncových uživatelů k domácím či podnikovým sítím.

Bezdrátové sítě poskytují, jak mobilitu pro koncové uživatele tak i dostatečnou přenosovou kapacitu. Tyto vlastnosti zapříčinili rychlé rozšíření bezdrátových sítí do veřejného i soukromého sektoru.

Spolu s rozšířením bezdrátových sítí vznikla potřeba bezpečného přístupu a přenosu dat v těchto sítích. Pokud neoprávněná osoba získá přístup do bezdrátové sítě získává většinou i přístup ke vše aktivům této sítě (servery, tiskárny, koncové stanice).

Historicky první protokol pro zabezpečení bezdrátové komunikace byl Wired Equivalent Privacy(WEP)[1]. Vzhledem k velkému množství bezpečnostních slabin[2] u tohoto protokolu, je dnes doporučeno používat zabezpečení s WPA2 (IEEE 802.11i). Tato konfigurace nám zajití bezpečnost podle nejnovějších standardů pro bezdrátovou komunikaci.

Další možností jak dosáhnout zabezpečené komunikace v bezdrátové síti je použití Infrastruktury veřejných klíčů(PKI). PKI nám zajistí integritu, důvěrnost, autenticitu a zodpovědnost. Tyto čtyři základní pojmy jsou správným předpokladem pro bezpečně fungující přenosový systém.

[1] http://standards.ieee.org/getieee802/802.11.html
[2] http://www.aircrack-ng.org

1 INFRASTRUKTURA VEŘEJNÝCH KLÍČŮ

Infrastruktura veřejných klíčů, často označovaná zkratkou PKI, je systém digitálních certifikátů, certifikačních úřadů a dalších registračních úřadů, které slouží k verifikaci a ověření platnosti všech stran zúčastněných v určité elektronické transakci, přičemž je použita asymetrická kryptografie.

Využití infrastruktury veřejných klíčů je obecně považováno za jednu z nejúčinnějších metod technologického zabezpečení základních atributů elektronické konfigurace (integrita, důvěrnost, autenticita, zodpovědnost) postavených na využití kryptografie[1].

Shrnutím výše uvedených poznámek se ukazuje, že digitální certifikáty nám poskytují jeden velice důležitý prvek bezpečnosti – důvěru. Základní dnes používaný přístup k vytváření digitálních certifikátů je popsán v normě ITU (International Telecommunications Union) X.509.

1.1 Digitální certifikát

Je dokument vydaný důvěryhodnou institucí. V tomto dokumentu je obsaženo tvrzení, že k určité osobě (přesněji - k určitému jednoznačnému jménu – distinguished name) patřící veřejný klíč má určitou konkrétní (číselnou) hodnotu. Filosofie opřená o služby důvěryhodné třetí strany není nová, je to vlastně analogie notářského ověření papírového dokumentu. Příjemce papírového dokumentu ověřuje razítko notáře a interpretuje ho např. jako důkaz, že osoba podepsaná v dokumentu učinila tento podpis v přítomnosti důvěryhodné strany – notáře.
Vzhledem k digitálnímu certifikátu sehrává roli (která je paralelní k roli notáře) důvěryhodné strany instituce nazývaná certifikační autoritou.

Výše byl zmíněn veřejný klíč patřící určité osobě. Konkrétní číselná hodnota tohoto klíče je obsažena v digitálním certifikátu. K tomuto veřejnému klíči patří soukromý klíč a tím již může výlučné disponovat pouze majitel tohoto klíče. Existuje pak řada prakticky používaných cest, které na základě této dvojice klíčů asymetrické kryptografie umožňují vytvářet cesty pro důvěryhodné transakce (digitální podpis, přenos klíčů pro symetrickou kryptografii atd.).
Uživatel musí mít důvěru v legitimnost takto získaného veřejného klíče. V opačném případě by mohl narušitel buď zaměnit veřejný klíč ležící někde v adresáři nebo by se

[1] definice Literatura [1]

mohl vydávat za někoho jiného. Pro tyto účely slouží právě certifikáty. Digitální certifikát označuje vlastníka veřejného klíče. Dovoluje verifikaci tvrzení, že daný veřejný klíč patří skutečně danému jedinci. Certifikáty pomáhají chránit se před možností, že někdo falzifikuje klíč s cílem vydávat se za někoho jiného. Ve své nejjednodušší podobě obsahují certifikáty veřejný klíč a jméno.
Obecně užívané certifikáty obsahují :

- dobu vypršení platnosti
- jméno certifikační autority, která vydala certifikát
- pořadové číslo
- informaci o tom jak klíč má být používán
- nejdůležitější je digitální podpis vydavatele certifikátu

Certifikáty nesmí být možné padělat, musí být získány bezpečnou cestou a vytvářeny musí být tak, aby potenciální narušitel je nemohl zneužít. Vydání certifikátu musí rovněž probíhat bezpečným způsobem, musí být odolné proti možným útokům. Pokud by něčí soukromý klíč byl ztracen či kompromitován, pak ostatní uživatelé musí být včas varováni a nesmí již déle šifrovat zprávy neplatným veřejným klíčem nebo akceptovat zprávy podepsané tímto zkompromitovaným soukromým klíčem. Uživatelé musí své klíče mít bezpečně uloženy, na druhé straně musí mít tyto klíče k dispozici pro jejich legitimní používání. Klíče mají platit pouze do doby než vyprší jejich platnost. Doba platnosti musí být vhodně zvolena a bezpečně publikována. Je třeba rovněž vzít do úvahy, že některé dokumenty budou mít zapotřebí ověřit platnost podpisu i po uplynutí doby platnosti daného veřejného klíče.

Nejrozšířenější akceptovaný formát pro certifikáty je definován mezinárodní normou ITU X.509. Tyto certifikáty mohou být pak čteny či psány libovolnou aplikací vytvořenou ve shodě s X.509. Normu X.509 využívá řada protokolů, např. PEM, PKCS, S-HTTP a SSL. Certifikační autorita je organizace (důvěryhodná třetí strana), která podepisuje uživatelův veřejný klíč a jeho jméno (případně další doplňkové údaje jako doba platnosti) svým vlastním soukromým klíčem. Certifikát lze ověřit veřejným klíčem certifikační autority. Pokud chtějí nyní dva partneři spolu komunikovat, mohou se vzájemně autentizovat ověřením digitálního podpisu druhé strany veřejným klíčem partnera a posléze ověřením partnerova veřejného klíče verifikací digitálního podpisu certifikátu užitím veřejného klíče certifikační autority. Stačí pak důvěřovat veřejnému klíči certifikační autority. Tímto způsobem je redukován počet veřejných klíčů, kterým každý s uživatelů musí důvěřovat.

Certifikační autority často také provádí verifikaci klíčů, aby bylo zajištěno správné vygenerování klíčů. Je jim důvěřováno, že správně provedou verifikaci. Na druhou stranu jim není sdělována žádná utajovaná informace (např. jiné uživatelovy utajované klíče).

Při větším počtu uživatelů jedna certifikační autorita nestačí. Veřejný klíč jedné certifikační autority může být certifikován jinou certifikační autoritou. Vytváří se tak sítě certifikačních autorit, které mohou mít různou hierarchickou strukturu.
Pro konkrétní certifikační autoritu je důležité jak postupuje při vydávání certifikátů, zejména pak, jak prověřuje oprávnění žadatele o certifikát. Některé certifikační autority mohou požadovat při identifikaci uživatele velmi málo, ale například banky nebudou zajisté chtít věřit certifikátům s nízkou úrovní jistoty. Každá certifikační autorita musí zveřejnit své požadavky na identifikaci klienta a svoji politiku v této oblasti (dokumenty CP – certifikační politika a CPS – certifikační prováděcí směrnice). Další strany tak mohou posoudit úroveň spolehlivosti certifikátů dané certifikační autority.

Důležitým souvisejícím pojmem je seznam odvolaných certifikátů (CRL – Certification Revocation List), což je seznam veřejných klíčů, které byly odvolány dříve než skončila doba jejich platnosti. Je řada důvodů, pro které mohl být klíč odvolán a umístěn v CRL. Klíč mohl být kompromitován. Klíč mohl být určen pro zaměstnance firmy, který mezitím z firmy odešel. Při ověřování podpisu je nutné si ověřit zda příslušný klíč není umístěn v CRL. CRL je provozován certifikační autoritou a obsahuje informaci o odvolaných klíčích, které byly původně certifikovány touto certifikační autoritou. Jsou zde umístěny pouze klíče, jejichž původní doba platnosti nevypršela (klíče s vypršenou dobou platnosti nesmí být akceptovány v žádném případě).

Vůbec jestliže certifikát má sloužit jako základní prvek důvěry, pak je třeba říci, že strana, která se o tento certifikát opírá, by měla učinit všechny dostupné kroky k tomu, aby si ověřila platnost tohoto certifikátu. K tomu je třeba minimálně zjistit, zda doba platnosti certifikátu nevypršela, zda daný certifikát byl platný v době jeho použití (nebyl na CRL) a zda byly platné i všechny návazné certifikáty příslušného řetězce certifikačních autorit. Toto za opírající se stranu často učiní jí používaný software (např. Web. prohlížeč), je však v jejím zájmu ověřit si zda vše proběhlo správnou cestou.

1.2 Vytvoření digitálního certifikátu:

Můžeme požádat o vydání certifikátů certifikační autoritu, musíme jí k tomu postoupit některé informace. Jaké to budou údaje, záleží na politice certifikační autority. Obvykle také CA vydává certifikáty různých tříd, kde se tyto požadavky v jednotlivých třídách různí (zpřísňují). Tak např. pro tzv. demo-certifikáty stačí zaslat požadavek obsahující vaši mailovou adresu. Naopak pro udělení certifikátu nejvyšších tříd je nezbytná Vaše osobní přítomnost na tzv. registrační autoritě, kam přinesete také příslušné požadované dokumenty.

Můžeme si také sami vyrobit a podepsat digitální certifikát (tzv. self-signed certificate), ale je pak otázka, kdo bude tomuto certifikátu důvěřovat. Ověřující osoba bude samozřejmě vyžadovat, aby se k ní tento certifikát dostal důvěryhodnou cestou a v daném kontextu je toto možné vlastně jen osobním předáním. Je zjevné, že tento postup (při vyšším počtu zainteresovaných stran) není příliš praktický.

1.3 PKI (Public Key Infrastructure)

S tímto pojmem se v návaznosti na problematiku digitálních certifikátů setkáváme velmi často. Pojem PKI je používán často ve velice různorodých souvislostech. Je to komplexní název pro celou řadu činností. Přesnější formulace lze získat pro:

- cíle PKI – ustavit a ošetřovat důvěryhodné prostředí v síti.
- prostředky PKI – to jsou služby řídící práci s klíči a digitálními certifikáty (jako šifrování a digitální podpisy).

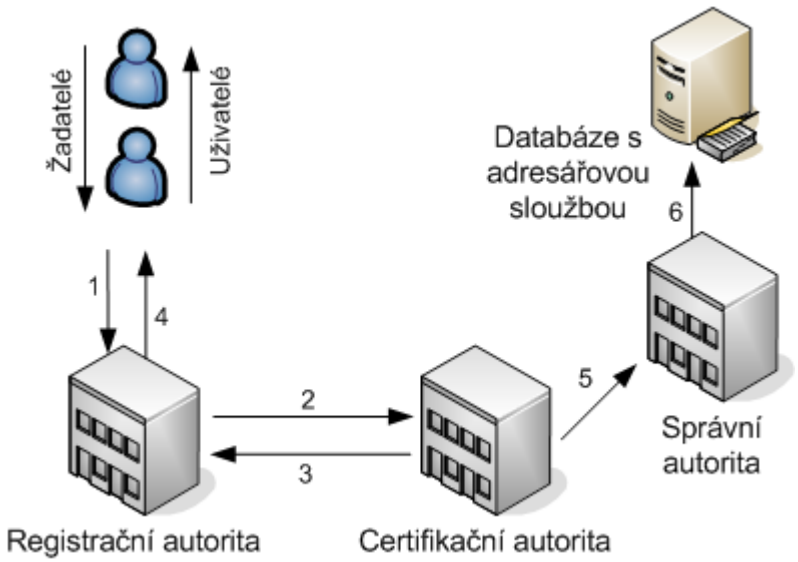
Jako komponenty PKI jsou v literatuře uváděny následující:

- Certifikační autorita (CA)
  Vydává a podepisuje certifikáty, s klientem převážně komunikuje Registrační autorita.
- Registrační autorita (RA)
  Komunikuje s klientem, prověřuje klientovu totožnost. Shledá-li RA vše v pořádku, pak žádost podepíše klíčem RA a předá k vyřízení CA.
- Certificate revocation List(CRL)
  CRL je list se zamítnutými certifikáty. Tyto certifikáty byly z patřičného důvodu zamítnuty a nejsou již nadále platné.
- Řízení práce s digitálními certifikáty (management)
  Management životních cyklů umožnuje řízení důležitých procesů spojených s uživatelskými účty, čipovými kartami a certifikáty. Mezi procesy patří založení uživatelského účtu v centrální databázi, jeho přiřazení do správných bezpečnostních skupin.
- Adresáře (databáze certifikátů), slouží jako uložiště.
- Certifikační politiky a prováděcí směrnice (CP a CPS), zde se definuje mechanismus prověřovaní totožnosti u žadatele.
- Navazující aplikace (např. šifrování mailů atd.)

V praxi jsou dnes již používána různá řešení PKI. Je proto vhodné uvést určitá kriteria, která umožňují srovnávat jednotlivá tato řešení.

- flexibilita, interoperabilita řešeni
- bezpečnost CA, RA

- snadnost použití (user friendly)
- snadnost úprav vzhledem k změnám (počtu uživatelů, hardwaru atd.)
- podpora bezpečnostní politiky organizace



Obr. 1.1: Postup při žádosti o certifikát

Legenta k obrázku 1.1:
Zákazní přijde na Registrační autoritu s žádostí o certifikát*(1)*. Registrační autorita prověří totožnost žadatele. Shledá-li vše v pořádku, pak žádost podepíše klíčem RA a předá k vyřízení CA*(2)*. CA verifikuje podpis RA a vystaví certifikát, který předá zpět RA*(3)* společně s certifikátem CA.
Registrační autorita předá žadateli*(4)*:

- Vydaný certifikát
- Certifikát CA
- Může se předat i CRL

CA předá vydaný certifikát správní autoritě*(5)*, která jej uloží do adresáře*(6)*. Uživatelé Internetu mohou certifikát získat např. Secure LDAP atd.

2 OPENVPN

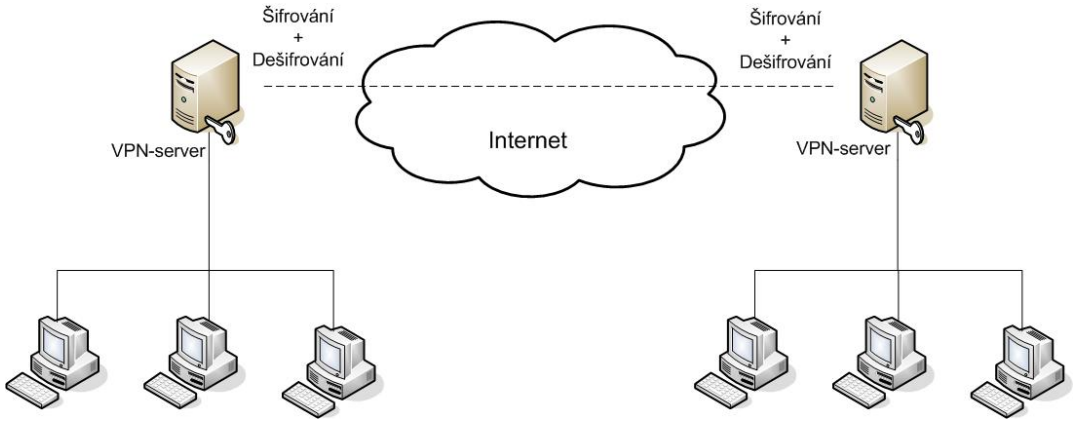
2.1 Seznámení s OpenVPN

Pro svou diplomovou práci jsem si vybral OpenVPN software, který podporuje vytvoření PKI. OpenVPN je plně konfigurovatelné open source řešení. OpenVPN se převážně využívá při vytváření zabezpečeného vzdáleného připojení, propojení dvou na sobě nezávislých sítí, bezdrátové sítě. OpenVPN můžeme použít na :

- Vytvoření zabezpečeného spojení v rámci vnitřní podnikové sítě
- Při vzdáleném přístupu zaměstnanců k firemní podnikové síti
- Vytvoření zabezpečeného spojení v rámci vnějších na sobě nezávislých sítí

Zabezpečení OpenVPN je založeno na SSL[1]. OpenVPN pracuje na OSI vrstvě 2 nebo 3. Jako síťové rozhraní používá SSL/TLS protokol, který podporuje autentizaci klienta na základě certifikátů, smart karet atd. OpenVPN není webová aplikace, ale konzolová aplikace. Toto jsou základní charakterizující vlastnosti.

Na následujícím příkladě demonstrujeme základní strukturu a použití.



Obr. 2.1: Obecné schéma OpenVPN.

Obrázek 2.1 ukazuje schéma, na kterém jsou obě strany permanentně připojeny k internetu. Servery jsou nakonfigurovány tak, aby chránily vnitřní síť před neautorizovaným přístupem z vnější sítě (Internet). Na každém serveru musí běžet program, který dokáže blokovat nebo povolit námi specifikovaný datový tok. Takový program se nazývá firewall. OpenVPN musí být správně nakonfigurováno ve firewallu daného serveru tak ,aby příchozí a odchozí datový tok pro OpenVPN mohl procházet.

[1]Standard pro bezpečnou komunikaci

Jakmile je OpenVPN a firewall nastaven můžeme mluvit o plnohodnotné virtuální privátní síti. Obě sítě jsou spojeny přes Internet a jejich topologie odpovídá klasické místní lokální síti. Aby byla komunikace bezpečná, použijeme šifrování.

Veškerý datový tok musí být zašifrován předtím než je odeslán, o to se stará OpenVPN.

Následující situaci si můžeme představit jako vytvoření bezpečných zašifrovaných tunelů skrz Internet, kterými pak proudí data z jedné sítě do druhé.

2.2 Popis sítových protokolů a vrstev pro OpenVPN

Síťový model OSI:

1. Fyzická vrstva: Zajišťuje vysílání a přijímání na hardwarové úrovni.
2. Linková vrstva: Řadí přenášené rámce, stará se o parametry přenosové linky.
3. Síťová vrstva: Směrování, adresování.
4. Transportní vrstva: Stará se o bezchybný přenos.
5. Relační vrstva: Vytváří spojení mezi koncovými aplikacemi.
6. Prezentační vrstva: Překlad mezi aplikačními a sítovými formáty.
7. Aplikační vrstva: Zajišťuje specifické aplikační protokoly.

Toto síťové seskupení je hierarchické, každá vrstva slouží nadřazené vrstvě. Pokud vše proběhne v pořádku na fyzické vrstvě tak řízení je předáno vrstvě další tj. linkové.
Tímto způsobem se pokračuje až do dosažení aplikační vrstvy.
Inernet je převážně založen na Internet Protocol(IP)

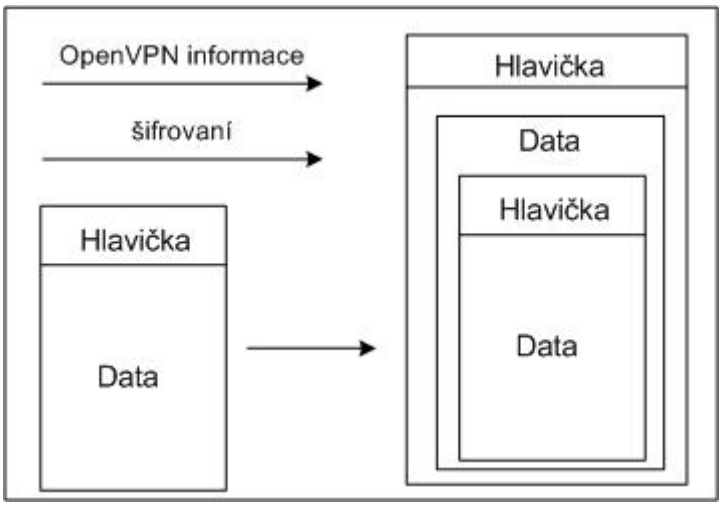
Sítový model IP:

1. Linková vrstva: Spojení OSI vrstev 1 a 2 (Fyzická a linkové vrstvy).
2. Síťová vrstva: Zahrnuje Síťovou vrstvu z OSI modelu.
3. Transportní vrstva: Obsahuje protokoly TCP a UDP
4. Aplikační vrstva: Zahrnuje vrstvy 5 až 7 z OSI modelu.

Síťové pakety jsou složeny ze dvou částí: hlavička a data. Hlavička obsahuje informace o odesílateli, příjemci a relevantní informace pro přenos. Tyto pakety se nazývají *ethernetové rámce* (vrstva 2) nebo *IP datagramy* (vrstva 3).

OpenVPN jednoduše vezme tyto ethernetové rámce nebo IP datagramy a zabalí je do jiného paketu obr. 2.2

- Celý síťový paket (rámec, datagram) obsahující hlavičku a data je zabalen do nového paketu.
- Všechny data, včetně informací jako odesílatel, příjemce jsou šifrovány.
- Nový paket má novou hlavičku obsahující informace o OpenVPN.



Obr. 2.2: Zabalení ethernetového rámce/IP datagramu .

2.3 VPN Bezpečnost

Veškerá síťová bezpečnost vychází ze čtyř základních bodů:
Důvěrnost, Integrita, Autentičnost, Nepopíratelnost.

1. Confidentiality (důvěrnost):
   Přenášená data musí být dostupná pouze autorizované osobě. Abychom toto splnili, musíme použít šifrování. Symetrické šifrování používá velmi rychlý blokový algoritmus při šifrování a dešifrování. Obě dvě strany používají stejný klíč pro šifrování a dešifrování. Největší slabina symetrického šifrování je v distribuci klíčů. Běžné symetrické šifry jsou DES, 3DES, Blowfish, AES, RC5, RC6, Serpant a IDEA.

2. Integrita:
   Pomocí VPN posíláme citlivá data přes veřejnou síť (Internet). Kdokoliv ve veřejné síti může pozměnit nebo jinak modifikovat námi odesílaná data. Chceme se ujistit, že data která příjme příjemce, jsou opravdu ta data, která jsme odeslali. K zajištění integrity přenášených dat se používají kontrolní součty neboli Hash funkce. Hash funkce je nevratná matematická operace, která vezme zprávu o jakékoliv délce a zakóduje ji na pevně stanovenou délku. Výsledek si můžeme představit jako výtah ze zprávy, který je ale zakódovaný. Každá zpráva má pouze jeden hash a v ideálním případě by nikdy neměli mít dvě rozdílné zprávy stejný hash. Malá změna ve zprávě změní celý hash.

   Předtím než odešleme naši zprávu, vypočítáme z této zprávy hash. Poté odešleme hash spolu s naší zprávou. Jakmile druhá strana příjme naši zprávu, vypočítá hash z přijaté zprávy. Pokud se vypočítaný hash rovná s hash, který

jsem poslali, tak příjemce ví, že zpráva nebyla cestou změněna. Běžně používané hash funkce jsou MD5, SHA1

3. Autentičnost:
   Pomocí asymetrického šifrování zajistíme autentičnost a nepopíratelnost. Chceme mít jistotu, že osoba se kterou mluvíme, je opravdu ta osoba, za kterou se vydává. K dosažení autentičnosti se používá asymetrické šifrování spolu s infrastrukturou veřejných klíčů. Obojí zahrnuje vytvoření páru klíčů (soukromý a veřejný), které jsou na sobě matematicky závislé. Data, která jsou zašifrovaná jedním klíčem je možné dešifrovat pouze druhým klíčem a naopak. První klíč je pojmenován jako veřejný klíč a je poskytnut veřejnosti. Druhý klíč je pojmenován jako soukromý klíč a musí být držen v tajnosti.

   K ověření autentičnosti se využívá vlastnost, že zpráva je zašifrovaná soukromým klíčem odesílatele. Jelikož je veřejný klíč dostupný, můžeme veřejný klíč použít k dešifrování zprávy. Pokud vše souhlasí je takto ověřena totožnost odesílatele.

4. Nepopíratelnost:
   Dále chceme zajistit, aby odesílatel, který odeslal zprávu nemohl později říci, že danou zprávu neodeslal, nepopíratelnost. Nepopíratelnost vychází z předchozí definice, protože osoba která odeslala zprávu musela znát odesílatelův soukromí klíč. A to je pouze odesílatel sám.

V praxi použijeme digitální podpis:
Když potřebujeme poslat zprávu, tak nejprve vypočítáme hash z této zprávy. Následně hash digitálně podepíšeme tak, že hash zašifrujeme naším soukromím klíčem. Zpráva i hash je následně spojena a symetricky zašifrovaná, dosažení důvěrnosti. Zpráva i hash jsou odeslány na druhou stranu tunelu, kde je symetrické šifrování dešifrováno. Příjemce poté dešifruje hash pomocí veřejného klíče odesílatele. Pokud vše souhlasí máme autentičnost a nepopíratelnost. Příjemce poté vypočítá hash ze zprávy a porovná ho s tím který dostal. Pokud se oba hashe rovnají tak víme, že zpráva nebyl změněna, tím jsem zajistili integritu. Nejpoužívanější asymetrický algoritmus je RSA.

OpenVPN spojení

OpenVPN pracuje na principu vytvoření virtuálního tunelu přes Internet nebo jakoukoliv síť. K vytvoření tohoto tunelu se používá symetrické šifrování. Obě strany tunelu používají stejný klíč pro šifrování a dešifrování. Symetrické šifrování je velice rychlé a je zde velice mnoho algoritmů, které můžeme použít (AES, Blowfish,

3DES).
Symetrické šifrování má dva základní problémy:

1. Key exchange = jak doručíme klíče na obě dvě strany tunelu
2. Autentičnost = jak víme, že si vyměňujeme klíče se správnou osobou

Máme několik možností jak klíče vyměnit. Použijeme telefon nebo email s Pretty Good Privacy (PGP)

Základem správné kryptografie je výměna klíčů tak často jak je to jen možné, s tím že frekvence výměny klíčů nezpůsobí velké vytížení systému nebo administrátora. Tento pojem můžeme najít pod anglickým názvem **Perfect Forward Secrecy**. Pokud je klíč prolomen pro jednu sérii přenosů, tak neohrozí žádné další.

Pokud budeme chtít provádět výměnu klíčů jednou za hodinu, tak využití telefonu nebo emailu není moc praktické.

OpenVPN řeší tento problém pomocí použití certifikátů. Každý systém má svůj veřejný a tajný klíč. Veřejný klíč je známý a soukromý klíč musí být držen v tajnosti. Jak už bylo výše řečeno, soukromý klíč se požívá k dokázání autentičnost a nepopíratelnosti. Pokud zašifrujeme zprávu pomocí našeho soukromého klíče tak druhá strana může potvrdit naši autentičnost tím, že použije k dešifrování náš veřejný klíč. Pokud máme vice jak 10 uživatelů, tak vzniká problém s rozšíření distribuce klíčů navzájem mezi těmito systémy. Každý systém by musel obsahovat 10 různých klíčů.

Řešením je Certifikační autorita (CA). CA zkontroluje identifikační údaje žadatele a potvrdí, že osoba je ta, za kterou se vydává. CA jednoduše podepíše žadatelův veřejný klíč svým soukromým klíčem. Tímto krokem odpadá povinnost naimportovat do systému 10 klíčů, stačí naimportovat veřejný klíč CA. Jinými slovy pokud důvěřujeme CA, tak důvěřujeme i tomu komu důvěřuje CA. Když dostaneme certifikát od druhé strany, tak tento certifikát bude obsahovat podpis vytvořený soukromým klíčem CA. My použijeme veřejný klíč CA k dešifrování tohoto podpisu, tím ověříme platnost certifikátu.

Nyní, když máme 10 uživatelů, kteří byly podepsáni CA, tak můžeme autentizovat tyto uživatele, tak že zkontrolujeme CA podpis na jejich certifikátech s veřejným klíčem CA. Tím se vyřeší problém rozšiřitelnosti.

OpenVPN Key Generation

Jakmile SSL/TSL handshake autentizuje obě strany, vytvoří se 4 rozdílné klíče. HMAC odesílající klíč, HMAC přijímací klíč, šifrovací/dešifrovací odesílající klíč a šifrovací/dešifrovací přijímací klíč.

K ochraně dat se používá symetrické šifrování, abychom mohli použít symetrické šifrování musíme mít na obou koncích spojení stejné klíče. Tím pádem potřebujeme doručit tyto klíče. IPSec používá systém pro výměnu klíčů, který se nazývá Internet Key Exchange (IKE). OpenVPN používá standardní RSA/DH handshake

Fáze 1:

Klient pošle hello, tím pozdraví a zároveň oznámí začátek handshake. Uvnitř je seznam šifer, které klient podporuje a jeden z parametrů pro vytvoření klíčů RSA nebo Diffie-Hellmann.

Fáze 2:

Server vybere šifru ze seznamu od klienta a pošle ji zpět se svým certifikátem, který obsahuje veřejný klíč serveru podepsaný soukromým klíčem CA. Toto je důležitý krok. Jakmile dostane klient certifikát od serveru, tak použije veřejný klíč CA k ověření zda-li byl certifikát opravdu vytvořen pomocí soukromého klíče CA. Pokud ověření souhlasí, tak máme jistotu, že komunikujeme s autentizovaným serverem. Pokud by nebyl proveden tento krok, vznikla by bezpečnostní díra pro útok man in the middle(muž uprostřed). Server zašle klientovi svůj parametr k vytvoření RSA.

```
Klient    ->    Fáze 1    ->    Server
Klient    <-    Fáze 2    <-    Server
Klient    ->    Fáze 3    ->    Server
Klient    <-    Fáze 4    <-    Server
```

Fáze 3:

Klient zašle svůj certifikát, který je také digitálně podepsaný soukromým klíčem CA. Server použije veřejný klíč CA k ověření autentičnosti klienta. Klient vygeneruje a pošle pre-master secret, tzv Client Key Exchange. Toto je základní bod v handshake a z něho vychází následující kroky. Pre-master secret je poslední parametr v RSA a je zašifrován veřejným klíčem serveru. Jakmile server dostane tento parametr, tak obě strany mají všechny nezbytné informace k vypočítání stejného symetrického klíče. Pre-master secret může být dešifrován pouze soukromým klíčem serveru.

Další bod je Change Cipher Spec, který znamená, že se použije šifra vybraná ve fázi 2, veškerá další komunikace bude šifrovaná. HMAC finish step je zde důležitý,neboť klient pošle hash celého handshake. Tím je zajištěna integrita. Běžný

útok je, že útočník pozmění seznam podporovaných šifer, tak že odstraní silné šifry. Důsledkem je spojení vytvořené slabou šifrou, které může být prolomeno. HMAC finish step detekuje takový útok a ukončí spojení.

Generování klíčů:
V tomto bodě obě strany vygenerují symetrický klíč, který je nezbytný pro bezpečnou komunikaci. Pre-master secret je použit k vytvoření master secret. Master secret blok klíčů, který vznikne zřetězením hash funkcí. Veškeré další klíče jsou pak odvozeny od tohoto bloku klíčů.

Fáze 4: Server přejde do šifrovacího módu z fáze 2 Change Cipher Spec. Také provede HMAC finish, tím ověří platnost veškerých odeslaných a přijatých dat. Tímto se vytvořil zašifrovaný tunel mezi dvěma účastníky, který používá symetrické šifrování.

Symetrické šifry: Jakmile se dostaneme do tohoto bodu, nastává minimální rozdíl mezi IPSec a SSL/TSL. Neboť šifrování závisí na šifře, pro kterou se rozhodneme. Oba dva systémy jak IPSec tak SSL/TLS nám dovolí vybrat velké množství podporovaných šifer.

Když vybíráme z symetrických šifer měli bychom se vyhnout DES. DES používá 56-bitový klíč, který se nedá již považovat za bezpečný. Také by se neměl používat 3DES, který má délku klíče 112 bitů, který je stále prohlášen za bezpečný, ale zbytečně využívá mnoho systémového času.

Další požadavek na šifrování je, aby se šifrovaný text náhodně měnil. Pokud se útočníkovi podaří získat velký objem dat zašifrovaných prostým klíčem, může se pokusit o prolomení. Tato situace může nastat, když je vytvořen tunel mezi dvěma sítěmi po delší dobu. OpenVPN mění klíče v základu každou hodinu, ale pro použití kratšího intervalu se použije mód CBC. Cipher Block Changing mode(CBC). Každý blok otevřeného textu je xorován s předcházejícím zašifrovaným blokem předtím než je celý zašifrován. Tím pádem, žádné množství zašifrovaného textu nemůže vést ke zjištění klíče.

Pokud nezvolíme jinak tak OpenVPN používá v základu velmi dobrou šifru bf-cbc (Bluefish cipher block changing) s klíčem 128bitů. Blowfish je velice silný algoritmus a zároveň velice rychlý. AES zajišťuje lepší šifrování na úkor šifrovacího výkonu.

```
Generovano příkazem: "openssl speed", CPU 1.8GHz C2D
type             16 bytes     64 bytes    256 bytes   1024 bytes   8192 bytes
md2               1738.73k     3696.04k     5104.38k     5663.17k     5819.05k
```

mdc2	0.00	0.00	0.00	0.00	0.00
md4	19798.60k	68502.56k	195231.54k	360394.41k	477094.96k
md5	16620.93k	54876.18k	145053.10k	244395.73k	306791.88k
hmac(md5)	17463.90k	54943.59k	145315.54k	245628.58k	304981.06k
sha1	15395.78k	46297.60k	105496.31k	154597.49k	182096.88k
rmd160	12784.34k	35362.49k	73203.84k	99879.17k	111523.71k
rc4	222702.94k	260464.14k	274607.46k	275295.41k	278417.61k
des cbc	38490.54k	40087.70k	40103.93k	40665.64k	40532.85k
des ede3	14261.25k	14472.94k	14354.98k	14526.66k	14500.14k
idea cbc	0.00	0.00	0.00	0.00	0.00
seed cbc	0.00	0.00	0.00	0.00	0.00
rc2 cbc	16886.77k	17516.69k	17488.85k	17646.02k	17615.68k
rc5-32/12 cbc	0.00	0.00	0.00	0.00	0.00
blowfish cbc	62307.39k	68600.49k	69767.82k	69281.15k	69577.60k
cast cbc	33531.62k	35373.44k	35960.67k	36167.96k	35859.43k
aes-128 cbc	39582.09k	66891.89k	81744.12k	85789.75k	88066.99k
aes-192 cbc	37277.02k	59609.94k	70447.50k	73945.60k	74606.66k
aes-256 cbc	47482.40k	59082.45k	63136.10k	64512.81k	64669.80k
sha256	8826.72k	20329.55k	35276.90k	43478.46k	46354.07k
sha512	5522.85k	22170.86k	36289.28k	51707.53k	58342.83k

HMAC/Hashing

Předali jsme klíče a používáme symetrické šifrování, můžeme začít přenášet data. Útočník nemůže číst námi přenášená data, ale může je modifikovat. Útočník nebude vědět jaké data změnil, ale tato změna může mít negativní efekt na přijímací straně. Zabezpečení proti pozměnění data se nazývá integrita. Dále chceme dosáhnout nepopíratelnosti.

K zajištění integrity se používá hash. Příjemci je poslána zpráva spolu s hashem dané zprávy. Příjemce vypočítá kontrolní hash z přijaté zprávy. Porovná oba hashe, pokud se rovnají tak zpráva nebyla změněna.
Co zabraní útočníkovi, aby jednoduše odebral náš hash ze zprávy, pozměnil zprávu a udělal nový hash?
Použijeme Hash Message Authentication Code (HMAC). Předtím než vytvoříme ze zprávy hash, připojíme ke zprávě secret key. Tím pádem klíč bude spolu se zprávou zahašován. Druhá strana příjme zprávu a připojí ke zprávě secret key a provede výpočet hashe ze zprávy a klíče dohromady. Pokud se vypočítaný hash a připojený hash ke zprávě rovnají, je ověřena autentičnost a nepopíratelnost. Pokud útočník pozmění zprávu, tak odstraní starý hash a vypočítá nový hash. Útočník ale nezná náš secret key, takže když příjemce vypočítá hash z přijaté zprávy a svého tajného klíče tak dostane jiný hash než ten, který je připojený ke zprávě.
Secret key je stejný pro obě strany a byl domluven v RSA handshake.

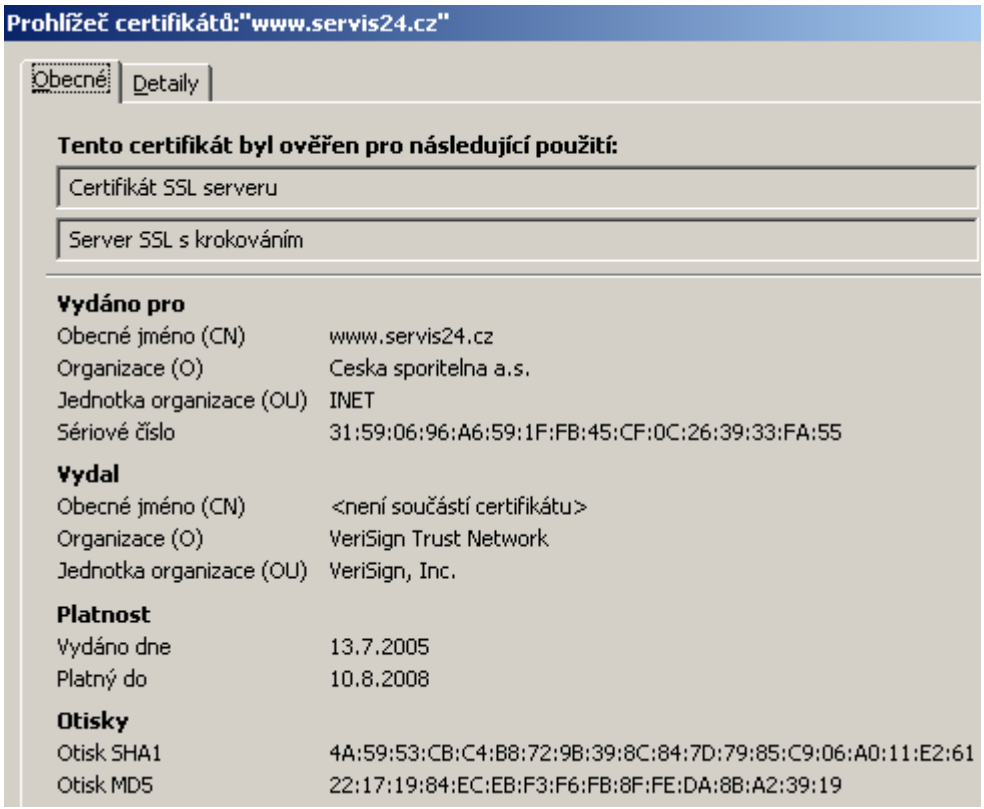
Pomocí HMAC jsem zajistili integritu a nepopíratelnost, protože pouze odesílatel zná secret key.

Shrnutí SSL/TSL Protokol Transport Layer Security (TLS) a jeho předchůdce, Secure Sockets Layer (SSL), jsou kryptografické protokoly, poskytující možnost zabezpečené komunikace na Internetu pro služby jako WWW, elektronická pošta, OpenVPN a jiné.

1. Klient používá veřejný klíč CA k ověření jejího podpisu v serverovém certifikátu. Lze-li digitální podpis CA ověřit, klient přijme serverový certifikát jako platný certifikát vydaný důvěryhodnou CA.
2. Klient ověřuje, zda je vydávající certifikační autorita na seznamu důvěryhodných CA.
3. Klient kontroluje dobu životnosti serverového certifikátu. Autentizační proces se zastaví, pokud doba jeho platnosti vypršela.
4. K ochraně před útoky typu Man-in-the-Middle porovnává klient aktuální DNS jméno serveru se jménem z certifikátu.
5. Ochrana před několika známými útoky (včetně man-in-the-middle), jako je snaha o použití nižší (méně bezpečné) verze protokolu nebo slabšího šifrovacího algoritmu.
6. Opatření všech aplikačních záznamů pořadovými čísly a používání těchto čísel v MAC.
7. Používání ověřovacího kódu zprávy rozšířeného o klíč, takže jen vlastník klíče dokáže MAC ověřit. Definováno v RFC 2104. Jen v TLS.
8. Zpráva ukončující inicializaci (Finished) obsahuje hash všech zpráv vyměněných v rámci inicializace oběma stranami.
9. Pseudonáhodná funkce rozděluje vstupní data na poloviny a zpracovává každou z nich jiným hashovacím algoritmem (MD5 a SHA-1), pak je XORuje dohromady. To poskytuje ochranu, pokud by byla nalezena slabina jednoho z algoritmů.
10. SSL v3 je proti SSL v2 vylepšeno přidáním šifer založených na SHA-1 a podporou autentizace certifikáty. Další vylepšení SSL v3 zahrnují lepší inicializační protokol a vyšší odolnost proti útokům typu man-in-the-middle[2].

[2]Definice SSL/TLS Literatura[1]

Na následujícím příkladě je ukázáno využití SSL/TSL pro WWW:
1)Certifikát je podepsaný důvěryhodnou CA.
Pokud zadáme http://servis24.cz (internetové bankovnictví České spořitelny) jsme automaticky přesměrováni na https://servis24.cz
Následující screenshoty ukazují jaké zabezpečení používá Česká spořitelna obr. 2.3

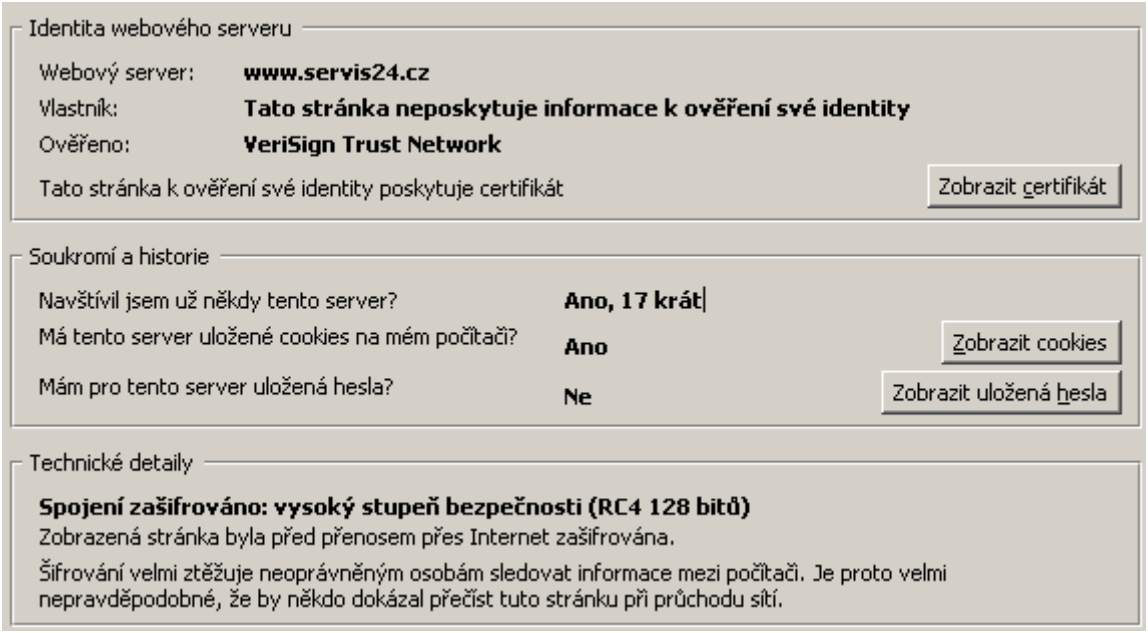


Obr. 2.3: Ověření certifikátu

1. **Identita webové stránky byla ověřena**. Jak na to webový prohlížeč přišel? V základu jsou v každém prohlížeči integrovány důvěryhodné Certifikační autority (CA). V tomto případě se jedná o VeriSign Trust Network. Česká spořitelna má certifikát podepsaný od VeriSign Trust Network. Tím, že důvěřujeme VeriSign Trust Network, důvěřujeme i České spořitelně.

Obecné informace nám o zabezpečení serveru řeknou obr. 2.4

1. K čemu se certifikát používá: Certifikát pro server
2. Pro koho byl certifikát vydán: www.servis24.cz
3. Kdo certifikát vydal: VeriSign Trust Network



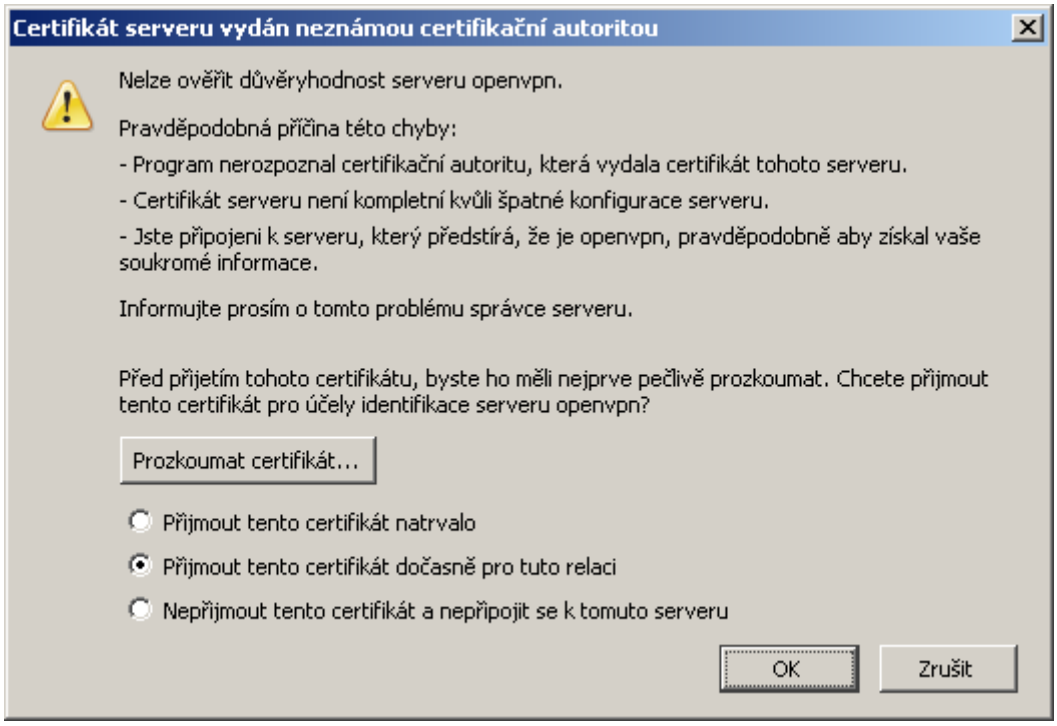
Obr. 2.4: Detail certifikátu

4. Platnost certifikátu: Platný do 10.8.2008
5. Otisky (Fingerprint) certifikátu: Charakterizující otisk MD5, SHA1. Algoritmus podpisu certifikátu je SHA1

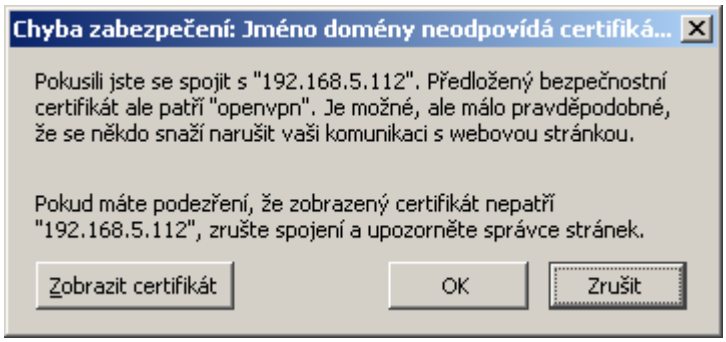
2)Certifikát je podepsaný sám sebou.

Certifikáty podepsány sami sebou se používají za účelem testování nebo v rámci lokální sítě, kdy podepsání CA není potřeba.

- V tomto případe nás bude prohlížeč varovat, protože tento certifikát není podepsaný žádnou důvěryhodnou CA obr. 2.5
- Prohlížeč nás varuje před špatným doménovým jménem obr. 2.6



Obr. 2.5: Certifikát serveru vydán neznámou certifikační autoritou



Obr. 2.6: Jméno certifikátu neodpovídá doméně

3 INSTALACE A KONFIGURACE SYSTÉMU

3.1 Instalace operačního systému Debian

Debian je svobodný operační systém (OS) určený k provozu na mnoha různých typech počítačů. Operační systém se skládá ze základního programového vybavení a dalších nástrojů, kterých je k provozu počítače třeba. Vlastním základem OS je jádro. Jelikož Debian používá jádro Linux a většina základních systémových programů byla vytvořena v rámci projektu GNU, nese systém označení GNU/Linux. [1]

Pro naši potřebu nám postačí instalační obraz "netinst", tento obraz je ke stáhnutí na stránkách http://www.debian.com/CD/netinst/ . Samotná velikost instalačního obrazu je 180MB, po nainstalování operačního systému je zabráno zhruba 700MB.

Instalaci popíši v základních krocích, pro podrobnější informace použijte instalační příručku http://www.debian.com/releases/stable/i386/ .

1. Vypálíme stáhnutý instalační obraz na CD-ROM, pro snadné nabootování. Po nabootování nás přivítá instalační průvodce, který nám pomůže s instalací.
2. Nastavení instalačního programu a rozpoznání hardwaru
3. Rozdělení disku a výběr přípojných bodů
4. Nastavení systému
5. Instalace základního systému
6. Instalace dodatečného softwaru
7. Nastavení zavádění systému
8. Dokončení instalace

Námi vytvoření uživatelé:

Přihlašovací jméno	Heslo	Práva
root	toor	superuživatel
xbedaj	xbedaj	normální uživatel

Superuživatel slouží ke konfiguraci systému, normální uživatel slouží k používání systému. Nyní máme plnohodnotný operační systém, přejdeme proto k instalaci a konfiguraci OpenVPN.

[1] Převzato ze stránek http://www.debian.com

3.2 Instalace, konfigurace OpenVPN

3.2.1 Instalace OpenVPN

Pro instalaci OpenVPN můžeme zvolit již předkompilovaný balíček nebo zkompilujeme balíček sami.

Zvolíme první možnost a pokračujeme příkazy:
apt-get update => provede aktualizaci balíčků
apt-get install openvpn => provede instalaci balíčků

OpenVPN se nainstaluje do **/usr/share/doc/openvpn/examples/easy-rsa/2.0**
Výpis z adresáře:

```
backup      build-inter     build-key-pkcs12  build-req-pass  keys
build-ca    build-key       build-key-server  clean-all       list-crl
build-dh    build-key-pass  build-req         inherit-inter   Makefile
openssl-0.9.6.cnf.gz        README            vars            openssl.cnf
revoke-full whichopensslcnf pkitool           sign-req
```

Všechny programy, které potřebujeme pro nakonfigurování OpenVPN jsou zde. My využijeme program "*pkitool*", který slouží k vytvoření certifikátů pro CA, server, clienty. *Pkitool* využijeme i ve webovém rozhraní, proto se budeme zabývat převážně jím.

Pro správné nakonfigurování "*pkitool*" je potřeba editovat "*vars*".
export KEY_SIZE=1024 => Určuje velikost klíče. Pro testovací účel může ponechat, pro reálné nasazení doporučení na 2048[2]
export CA_EXPIRE=3650[3] => Za kolik dnů vyprší platnost CA klíče
export KEY_EXPIRE=3650 => Za kolik dnů vyprší platnost certifikátů

```
Změníme poslední řádky:
"These are the default values for fields
 which will be placed in the certificate.
 Don't leave any of these fields blank."

export KEY\_COUNTRY="CZ"
```

[2] http://www.keylength.com/en/8/
[3] v reálném nasazení se volí délka 365 dnů

```
export KEY\_PROVINCE="Morava"
export KEY\_CITY="Brno"
export KEY\_ORG="VutbrFEKT"
export KEY\_EMAIL="xbedaj01@stud.feec.vutbr.cz"
.....................................................
KEY\_COUNTRY="CZ"       => Dva znaky charakterizující stát
KEY\_PROVINCE="Morava"  => Kraj
KEY\_CITY="Brno"        => Město
KEY\_ORG="VutbrFEKT"    => Organizace
KEY\_EMAIL="xbedaj01@stud.feec.vutbr.cz" => email
```

Postup pro vytvoření certifikátů je následující

1. *source ./vars* => nastaveni zdroje
2. *./build-db* => vytvoření DH algoritmu
3. *./pkitool –initca* => Vytvoření certifikační autority
4. *./pkitool –sever* mujserver => Vytvoření certifikátu pro server s názvem "mujserver"
5. *./pkitool klient1* => Vytvoření certifikátu pro klienta s názvem "klient1"
6. *./pkitool klient2* => Vytvoření certifikátu pro klienta s názvem "klient2"

V těchto krocích jsem vytvořili Diffie-Helman[4], certifikáty pro Certifikační autoritu, server a dva klienty.

Příklad přidání nového uživatele s názvem "*klient3*" :
source ./vars ; *./pkitool klient3*

[4]http://en.wikipedia.org/wiki/Diffie-Hellman

3.3 Instalace Apache, PHP, HTTPS

Instalace Apache2 + openssl

 # apt-get install apache2 openssl ssl-cert

Instalace PHP5

 # apt-get install libapache2-mod-php5 php5-cli php5-common php5-cgi

V předchozích krocích jsme nainstalovali Apache2 a PHP5, pro podporu HTTPS protokolu musíme vygenerovat certifikát pro server. Jak již bylo výše zmíněno, námi vytvořený certifikát bude nepodepsaný věrohodnou Certifikační autoritou.

Pro vytvoření certifikátu použijeme openssl s následující syntaxí:

 # openssl req $@ -new -x509 -days 365 -nodes -out /etc/apache2/apache.pem -keyout /etc/apache2/apache.pem

```
 Generating a 1024 bit RSA private key
.............................................++++++
.........................................++++++
writing new private key to ‘/etc/apache2/apache.pem’
```

Vyplníme podle následujícího příkladu:

```
Country Name (2 letter code) [AU]:CZ
State or Province Name (full name) [Some-State]:Czech republic
Locality Name (eg, city) []:Brno
Organization Name (eg, company) [Internet Widgits Pty Ltd]:vutbr
Organizational Unit Name (eg, section) []: openvpn
Common Name (eg, YOUR name) []: openvpn
Email Address []: xbedaj01@stud.feec.vutbr.cz
```

Certifikát je vytvořen, nyní musíme nastavit správná přístupová práva k souboru "*.pem*" . Práva nastavíme pouze pro čtení a zápis vlastníkovi.

 # chmod 600 /etc/apache2/apache.pem

Standardně Apache2 naslouchá pouze na portu 80. Aby naslouchal na portu 443, na kterém běží SSL spojení, změníme následující soubor.

/etc/apache2/ports.conf

Listen 443

Nastavení SSL modulu pro Apache2

 # a2enmod ssl

Modul je zaveden, pro promítnutí změny provedeme restart Apache2

 # /etc/init.d/apache2 restart

Configurace SSL certifikátu, Virtual Hosts

Editujeme soubor */etc/apache2/sites-available/default*

```
NameVirtualHost *:443
  <VirtualHost *:443>
        SSLEngine on
        SSLCertificateFile /etc/apache2/apache.pem
        ServerAdmin webmaster@localhost

   DocumentRoot /var/www/
        <Directory />
                Options FollowSymLinks
                AllowOverride None
        </Directory>

   atd......
```

Důležité jsou první 4 řádky.

1. port na kterém bude naslouchat server
2. port na kterém bude naslouchat Virtual Host
3. zapnutí SSL módu, pro Virtual Host
4. definování certifikátu pro server

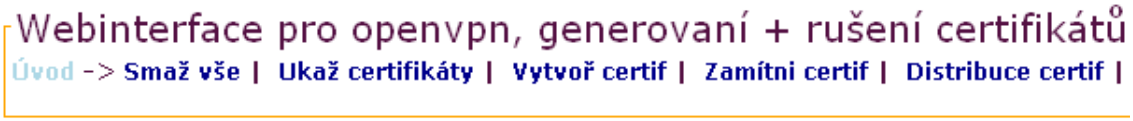
Aby se projevilo nové nastavení serveru, provedeme reload.

 # /etc/init.d/apache2 reload

4 WEBINTERFACE OPENVPN

Uživatelské rozhraní pro vytváření, zamítnutí a distribuci certifikátů je naprogramováno v PHP. V první části se zaměřím na praktické použití tohoto rozhraní, v druhé části ukážu funkce, které toto rozhraní ovládají.

4.1 Uživatelské rozhraní

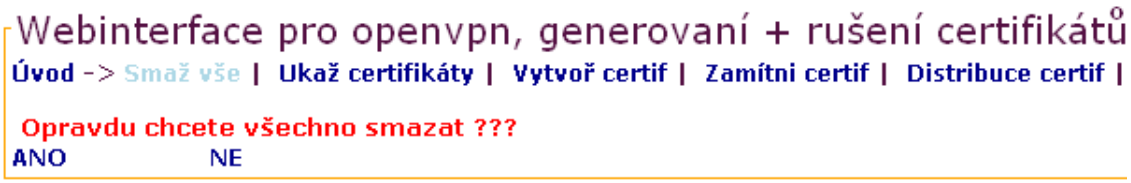


Obr. 4.1: Úvod do webového rozhraní

Na obr. 4.1 vidíme webové rozhraní, které je naprogramováno pro :

- Smazání všech certifikátu a klíčů, které jsme vytvořili
- Přehled o již vytvořených certifikátů
- Vytvoření certifikátů
- Zamítnutí, zneplatnění certifikátu
- Distribuce certifikátů koncovým uživatelům

4.2 Smazání všech certifikátů



Obr. 4.2: Zrušení všech certifiátů

Pokud odpovíme na otázku na obr. 4.2 kladně, smažou se všechny certifikáty a klíče pro ně vytvořené. Tato funkce se použije, při potřebě vygenerovat všechny certifikáty nanovo.
Standardní umístění všech certifikátů je v */usr/share/doc/openvpn/examples/easy-rsa/2.0/keys*. Smazání je provedeno skriptem *clean-all* od OpenVpn. Tento skript

smaže veškerý obsah v adresáři */usr/share/doc/openvpn/examples/easy-rsa/2.0/keys*. Proto tuto funkci používejte s rozvahou.

4.3 Vytvoření certifikátů

Tato sekce slouží k vytvoření certifikátů pro Certifikační Autoritu, Server a pro klienty.

4.3.1 Vytvoření DH

Obr. 4.3: Vytvoření DH

```
NOTE: If you run ./clean-all, I will be doing a rm -rf on /usr/share
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
.....................................+.............+.+.......
```

Obr. 4.4: Vytvoření DH, průběh

Vytvoření DH obr. 4.3 se provádí pouze jednou, trvá zhruba 2 minuty v závislosti na výkonnosti CPU. Příkaz pro vytvoření DH: *./build-dh*

4.3.2 Vytvoření CA

Obr. 4.5: Vytvoření certifikátů pro CA

```
NOTE: If you run ./clean-all, I will be doing a rm -rf on /usr/share/doc/ope
Using CA Common Name: CA
Generating a 1024 bit RSA private key
.....................................................++++++
...................++++++
writing new private key to 'ca.key'
-----
```

Obr. 4.6: Vytvoření certifikátů pro CA, průběh

Certifikační autorita se vytváří také pouze jednou obr. 4.5. Námi vytvořenou CA podepíšeme následující certifikáty jak pro server tak pro klienty.
Příkaz pro vytvoření CA: *./pkitool –initca CA*

4.3.3 Vytvoření certifikátu pro server

Obr. 4.7: Vytvoření certifikátů pro server

```
Generating a 1024 bit RSA private key
.......++++++
.............++++++
writing new private key to 'server.key'
-----
Using configuration from /usr/share/doc/openvpn/examples/easy-rsa/2.0/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'CZ'
stateOrProvinceName   :PRINTABLE:'CZ'
localityName          :PRINTABLE:'Brno'
organizationName      :PRINTABLE:'VutbrFEKT'
commonName            :PRINTABLE:'server'
emailAddress          :IA5STRING:'xbedaj01@stud.feec.vutbr.cz'
Certificate is to be certified until May 14 14:10:12 2018 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
```

Obr. 4.8: Vytvoření certifikátů pro server, průběh

Certifikátů pro server obr. 4.7. můžeme vytvořit více než jeden, v našem případě vytvoříme jeden. Máme jenom jeden server, proto vytváříme pouze jeden certifikát. Průběh vytváření je patrný. Údaje pro vytvoření certifikátu se berou z výše zmiňovaného souboru *vars*.
Příkaz pro vytvoření certifikátu: *./pkitool –server server*

4.3.4 Vytvoření certifikátu pro klienta

Obr. 4.9: Vytvoření certifikátů pro klienta

```
Generating a 1024 bit RSA private key
......++++++
........++++++
writing new private key to 'xbedaj01.key'
-----
Using configuration from /usr/share/doc/openvpn/examples/easy-rsa/2.0/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'CZ'
stateOrProvinceName   :PRINTABLE:'CZ'
localityName          :PRINTABLE:'Brno'
organizationName      :PRINTABLE:'VutbrFEKT'
commonName            :PRINTABLE:'xbedaj01'
emailAddress          :IA5STRING:'xbedaj01@stud.feec.vutbr.cz'
Certificate is to be certified until May 14 14:26:32 2018 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
```

Obr. 4.10: Vytvoření certifikátů pro klienta, průběh

Certifikáty pro klienty obr. 4.9. vytvoříme hned čtyři:

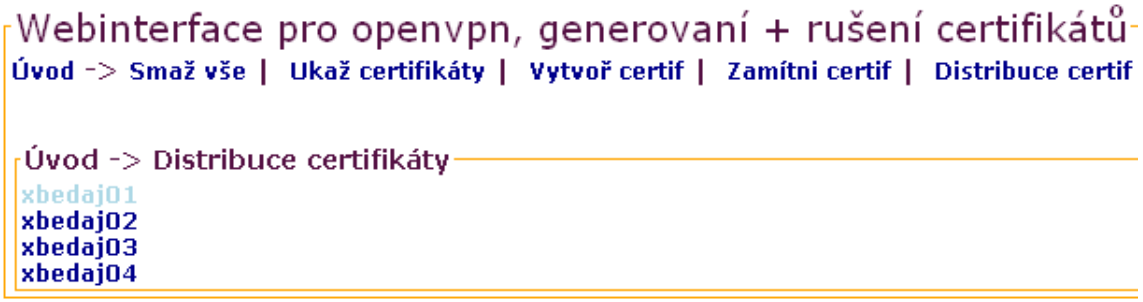
1. xbedaj01
2. xbedaj02
3. xbedaj03
4. xbedaj04

Z následujího obr. 4.9 je zřejmý způsob jak se vytváří certifikáty.
Příkaz pro vytvoření certifikátu pro klienty: *./pkitool xbedaj01*

Nyní jsme vytvořili jeden certifikát pro Certifikační autoritu, jeden certifikát pro server a čtyři certifikáty pro klienty.
V další části se podíváme na distribuci certifikátů ke koncovému uživateli.

4.4 Distribuce certifikátů ke koncovým uživatelům

V této části webového rozhraní se budeme zabývat distribucí certifikátů ke koncovým uživatelům. Z následujícího schéma je patrné, že pouze klientské certifikáty jsou určený pro distribuci. V našem případě je OpenVPN i Apache2 nainstalován na stejném počítači, tím odpadá distribuce certifikátu pro CA a server.



Obr. 4.11: Distribuce klientských certifikátů

Požadavek na přenos certifikátu ke koncovému uživateli:

- Přenos musí být šifrovaný, protokol HTTPS
- Server musí být autentizován vůči uživateli
- Uživatel musí být autentizován vůči serveru

Přenos musí být šifrovaný, protokol HTTPS

Podporu pro šifrovaný přenos jsem provedli v kapitole 2.3

Server musí být autentizován vůči uživateli

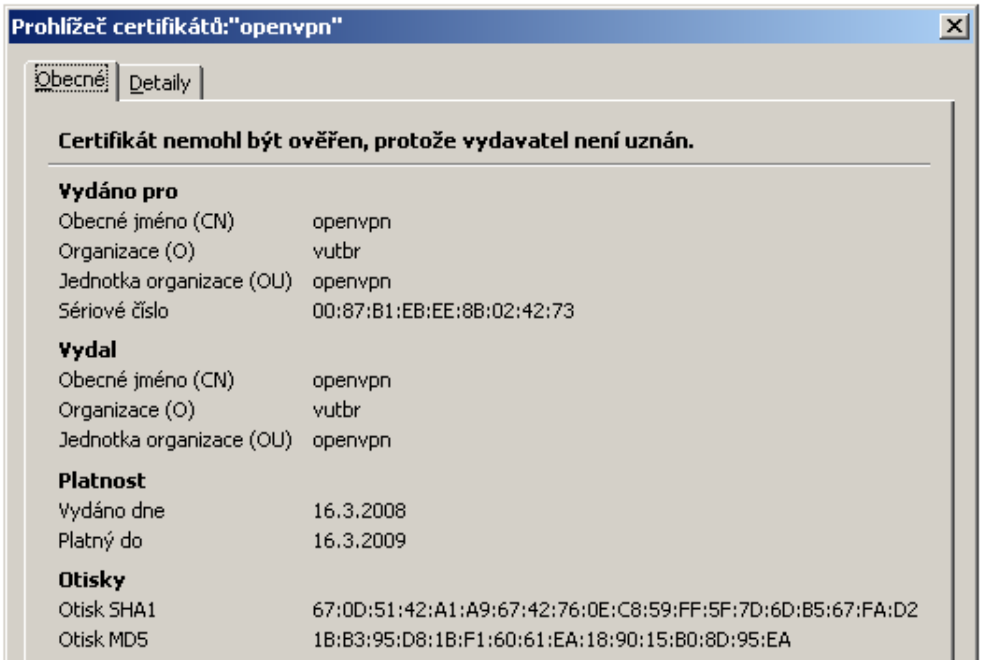
Uživatel musí mít možnost ověřit totožnost serveru. To znamená, server je opravdu ten server, za který se vydává obr. 4.12.

Ověření probíhá ze strany uživatele, který ví fingerprint neboli otisk serveru:

SHA1 Fingerprint=67:0D:51:42:A1:A9:67:42:76:0E:C8:59:FF:5F:7D:6D:B5:67:FA:D2

Teto otisk serveru porovná s otiskem ze svého webového prohlížeče:

Pokud se oba otisky shodují může uživatel certifikát serveru naimportovat do webového prohlížeče. Tímto krokem prohlásí, že certifikátu důvěřuje. Tímto je autentizace serveru dokončena.



Obr. 4.12: Porovnání otisku serveru s otiskem od administrátora

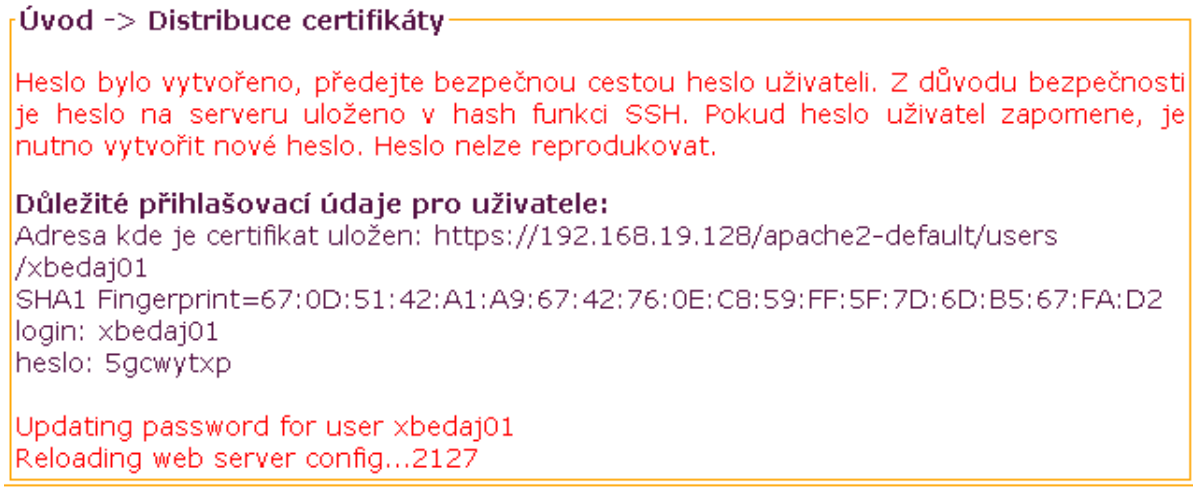
Uživatel musí být autentizován vůči serveru

Autentizace uživatele vůči serveru se provádí zadáním uživatelského jména a hesla:
login: xbedaj01 heslo: 5gcwytxp

Obr. 4.13: Vytvoření heslo pro uživatele xbedaj01

Z následujícího obr. 4.14. je důležité toto upozornění:
Heslo bylo vytvořeno. Předejte bezpečnou cestou heslo uživateli. Z důvodu bezpečnosti je heslo na serveru uloženo v hash funkci SSH1. Pokud heslo uživatel zapomene, je nutno vytvořit nové heslo. Heslo nelze reprodukovat.

Dále je zde, cesta k certifikátu a klíči, výše zmiňovaný otisk serveru, login a heslo pro uživatele.
V předchozích krocích proběhlo několik funkcí, které zajišťují tuto distribuci certifikátů. Proto se podíváme na tyto funkce podrobněji.



Obr. 4.14: Otisk serveru, login a heslo pro koncového uživatele

1. **Funkce pro vytvoření hesla:**

```
function generatePassword ($length = 8)
{
//zaciname s prazdnym retezcem
   $password = "";
//definumeme mozne znaky
   $possible = "0123456789bcdfghjkmnpqrstvwxyz";

//nastavení citace
   $i = 0;

//pridávání nahodnych znaku, dokud neni pozadovana delka hesla
   while ($i < $length) {

//vybere se nahodny znak
   $char = substr($possible, mt_rand(0, strlen($possible)-1), 1);

//do hesla se prida pouze znak, ktery tam jeste neni
    if (!strstr($password, $char)) {
      $password .= $char;
      $i++;
    }
  }
//hotovo!
  return $password;
}
```

2. **Zapsání hesla do databáze pro server Apache2:**
   htpasswd -bs /home/domain/public_html/membersonly/.htpasswd xbedaj01 5gcwy-

txp

K tomuto účelu slouží *htpasswd*, kde přepínač znamená:

"*-b* = zadej heslo z příkazové řádky"

"*-s* = Nastavení SSH šifrování"

Nově vytvořené heslo vždy přepíše staré heslo. Toto je výchozí soubor pro Apache2 při autentizaci uživatelů.

3. **Konfigurace souboru *./etc/apache2/sites-available/default***
   Default soubor je konfigurační soubor sloužící pro nastavení, jak podpory SSL modu, tak i autentizaci uživatelů:

```
<Directory /var/www/apache2-default/users/$this->username>
 AuthName \"Zadejte login a heslo.\"
 AuthType Basic
 AuthUserFile /home/domain/public_html/membersonly/.htpasswd
 AuthGroupFile /dev/null
 require user $this->username

 Options Indexes FollowSymLinks MultiViews
 AllowOverride None
 Order allow,deny
 allow from all
 RedirectMatch ^/$ /apache2-default/
</Directory>

\emph{$this->username} je v našem případě xbedaj01
```

   Toto nastavení zajistí, že obsah adresáře */var/www/apache2-default/users/xbedaj* uvidí pouze uživatel.

   Přihlašovací jméno: *xbedaj01* heslo: *5gcwytxp*

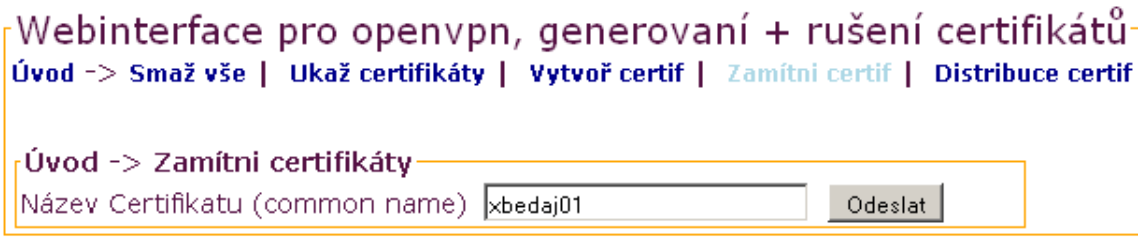
   soubor "default" je automaticky upravován v závislosti na platnosti uživatelských certifikátů.

   Tzn. po vytvoření certifikátu je uživatel automaticky přidán, ale jakmile je certifikát zamítnut, tak se uživatel automaticky odstraní.

4.5 Zamítnutí certifikátů

Administrátor použije funkci zamítnutí certifikátu, v případě:

1. Došlo k odcizení certifikátu, ukradení USB klíčenky, notebooku atd.
2. Ukončení pracovní dohody mezi zaměstnancem, zaměstnavatelem. Odepření přístupu bývalého zaměstnance k firemní síti.



Obr. 4.15: Zamítnutí certifikátů

```
Using configuration from /usr/share/doc/openvpn/examples/easy-rsa/2.0/openssl.cnf
Revoking Certificate 10.
Data Base Updated
Using configuration from /usr/share/doc/openvpn/examples/easy-rsa/2.0/openssl.cnf
xbedaj01.crt: /C=CZ/ST=CZ/L=Brno/O=VutbrFEKT/CN=xbedaj01/emailAddress=xbedaj01@stud.feec.vutbr.cz
error 23 at 0 depth lookup:certificate revoked

Reloading web server config...2127
```

Příkaz pro zamítnutí certifikátu: *./revoke-full xbedaj01*
Jakmile se provedou tyto příkazy vytvoří se soubor *crl.pem*". CRL obsahuje všechny certifikáty, které byly zamítnuty. Tento soubor je potřeba přehrát do konfiguračního souboru pro OpenVPN, pokud chceme aby se změna projevila okamžitě musíme reloadovat nastavení OpenVPN */etc/init.d/openvpn reload*". Jinak se změna projeví až při dalším přihlášení.

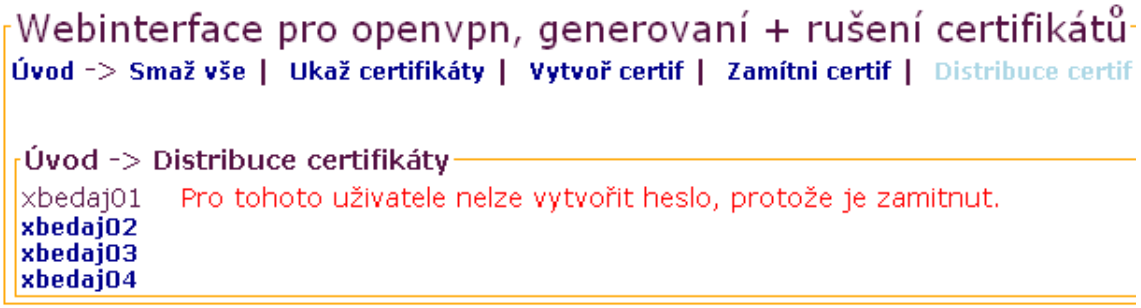
Modelový případ

Zaměstnanci s uživatelským jménem *xbedaj01* byl ukraden notebook, uživatel zde měl nainstalován software OpenVPN se vzdáleným přístupem do sítě. Certifikát nebyl chráněn heslem, a tak útočníkovi nebrání nic v přístupu do firemní sítě.
Uživatel informuje administrátora, který neprodleně zamítne certifikát *xbedaj01* obr. 4.15. CRL přehraje do konfiguračního adresáře a provede reload OpenVPN. Po těchto třech krocích je certifikát zneplatněn a útočník se již nemůže připojit k firemní síti.

Větší bezpečnosti dosáhneme při zabezpečení certifikátu heslem. Tzn. kdo nezná heslo, nemůže použít tento certifikát. Pokud nám útočník odcizí notebook s OpenVPN, tak se nepřipojí do firemní sítě pokud nezadá heslo.
Zde je ovšem možné použít rozdílné metody jak získat heslo:

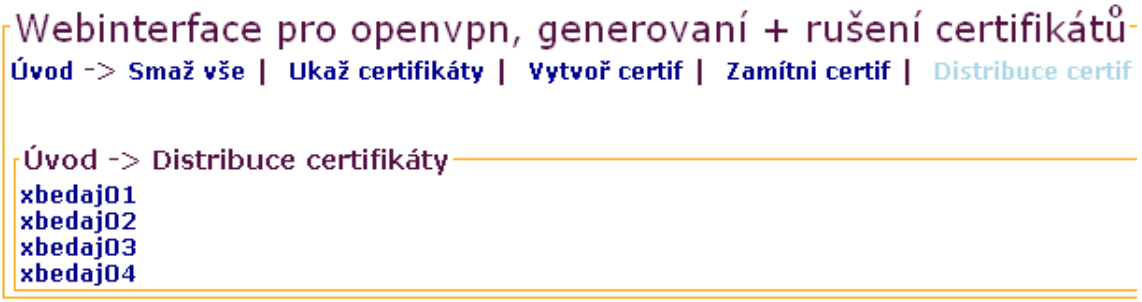
- Sociální inženýrství
- Hádání hesla
- Bruteforce

I když je certifikát zajištěn heslem, musíme okamžitě informovat administrátora, který provede výše zmíněný postup pro zamítnutí certifikátu.



Obr. 4.16: Zamítnutý certifikát nelze distribuovat

Jak ale vyřešíme přístup pro daného uživatele, který je již zvyklý na své přihlašovací jméno?
Jednoduše vytvoříme nový certifikát se stejným jménem *xbedaj01*. Nový certifikát přepíše starý certifikát. Tento nový certifikát dáme našemu uživateli. Uživatel nic nepozná, ale v OpenVPN je starý certifikát zamítnut a nový povolen obr. 4.17.



Obr. 4.17: Obnovený certifikát lze znovu distribuovat

Vytvořili jsem nový certifikát se stejným jménem, proto je certifikát znovu připraven pro distribuci.

4.6 Přehled vytvořených certifikátů

4.6.1 Přehled vytvořených certifikátů, CA

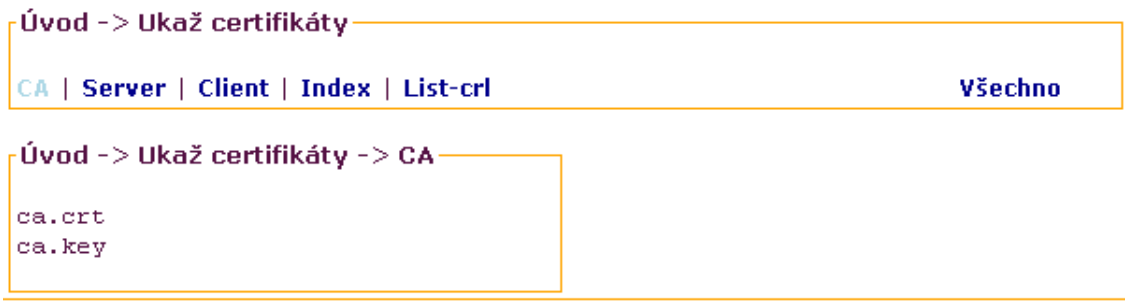
Obr. 4.18: Přehled certifikátů

Na obr. 4.18 se můžeme podívat na náhled certifikátů pro CA. Příkaz pro vypsání CA je:

```
function showCa(){
 $this->show = "cd /usr/share/doc/openVPN/examples/easy-rsa/2.0/keys/
                2>&1; ls ca.* 2>&1";
 $this->output = shell_exec($this->show);

 echo "<div style='width: 300px;'>
        <form>
         <fieldset style='border: 1px solid orange; padding: 3px'>
                      <legend><b>Úvod -> Ukaž certifikáty -> CA </b></legend>";
 echo     "<pre>$this->output</pre>";
 echo   "</fieldset></form></div>";
        }
```

Příkaz vypíše všechny soubory v adresáři "/usr/share/doc/openVPN/examples/easy-rsa/2.0/keys/", které začínají na "ca." a vytvoří orámování kolem výpisu.



Obr. 4.19: Přehled certifikátů CA

Z obr. 4.19 je patrné, že jsme vytvořili pro CA certifikát a klíč.

4.6.2 Přehled vytvořených certifikátů, Server

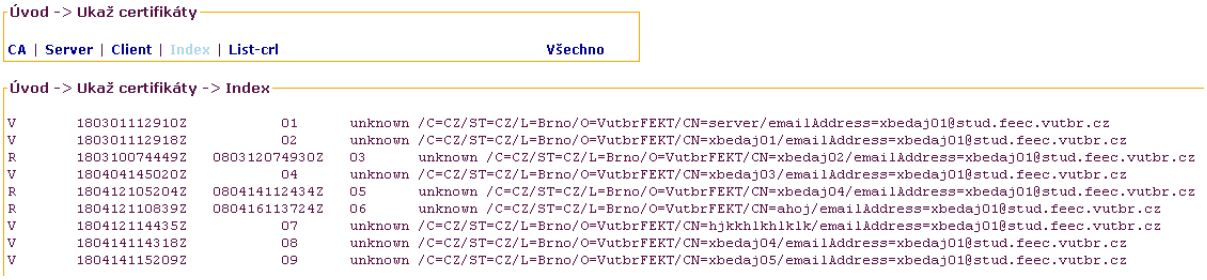
Obr. 4.20: Přehled certifikátů Server

Příkaz pro výpis certifikátu pro server je analogický k předchozímu výpisu CA. Změna je v použití příkazu *ls*.
*ls server.**

4.6.3 Přehled vytvořených certifikátů, Client

Obr. 4.21: Přehled certifikátů Client

Příkaz pro výpis všech klientů je analogický k předchozím. Změna je v použití přepínače *grep*. Ten nám zajistí výpis pouze námi požadovaných klientů.
ls | grep -v index | grep -v .pem | grep -v server | grep -v ca | grep -v serial



Obr. 4.22: Přehled certifikátů Index

4.6.4 Přehled vytvořených certifikátů, Index

Na obr. 4.22 je celkový přehled všech vytvořených a zamítnutých certifikátů. Vysvětlíme si význam jednotlivých symbolů na třech řádcích.

V 180301112910Z 01 /C=CZ/ST=CZ/L=Brno/O=VutbrFEKT/CN=server/emailAddress=xbedaj01@stud.feec.vutbr.cz

V 180301112918Z 02 /C=CZ/ST=CZ/L=Brno/O=VutbrFEKT/CN=xbedaj01/emailAddress=xbedaj01@stud.feec.vutbr.cz

R 180310074449Z 080312074930Z 03 /C=CZ/ST=CZ/L=Brno/O=VutbrFEKT/CN=xbedaj02/emailAddress=xbedaj01...

1. Symbol *V* je zkratka pro *Valid* neboli říká, že certifikát je platný, následuje pořadové číslo a informace o certifikátu. Důležitá informace v certifikátu je *CN* zkratka pro *Common Name*, určuje název certifikátu. Podle těchto informací najdeme ten certifikát, který hledáme

2. Druhý řádek má stejný význam jako předchozí, liší se pouze v *CN*

3. Symbol *R* je zkratka pro *Revoked* neboli oznamuje nám, že certifikát byl zamítnut. V tomto řádku je ještě údaj o čase, kdy byl certifikát zamítnut.

Příkaz pro výpis *Indexu*: pouze zobrazení souboru *index.txt* tento soubor vytváří a spravuje OpenVPN.

4.6.5 Přehled CRL - Certification Revocation List

Certification Revocation List obr. 4.23 obsahuje seznam všech zamítnutých certifikátů. Na výpise vidíme, že certifikáty s pořadovým číslem *03 05 06 0A* byli zamítnuty. Tyto certifikáty dohledáme ve výše zmíněném indexu, který obsahuje jak pořadové číslo vytvoření certifikátu tak jméno certifikátu.

Obr. 4.23: Přehled certifikátů CRL

Příkaz pro výpis *CRL*: výpis je zobrazen pomocí skriptu *./list-crl*" tento soubor vytváří a spravuje OpenVPN.

5 KONFIGURAČNÍ SOUBORY PRO OPENVPN

Pomocí webového rozhraní jsem vytvořili certifikáty. V této části nastavíme OpenVPN na straně serveru a na straně jednotlivých klientů. Budeme předpokládat, že OpenVPN v režimu server poběží na linuxové platformě. A klienti se budou připojovat na server z Windows XP SP2 nebo Windows Vista. Pro klienty, kteří se připojují z Linuxu je postup analogický.

5.1 Konfigurace na straně serveru(server.conf)

```
port 1194                       #VPN Port
local 192.168.5.112             #místní adresa

proto tcp                       #Protocol pro VPN
dev tun                         #Volba TUN  nebo TAP

ca keys/ca.crt                  #Umístění klíčů a certif.
cert keys/server.crt
key keys/server.key             #secret file

crl-verify keys/crl.pem         #CRL list

dh keys/dh1024.pem              #Diffie hellman parametr.

server 10.8.1.0 255.255.255.0   #VPN DHCP Pool

push "redirect-gateway"         #Presmeruje branu

keepalive 10 120                #keepalive

cipher AES-128-CBC              #def. šifry
comp-lzo                        #komprese

user nobody                     #openVPN omezení práv
group nogroup
persist-key
persist-tun

status openVPN-status.log       #status log
verb 3                          #verbosita
```

5.2 Konfigurace na straně klienta(xbedaj01.oVPN)

```
#Určení klienta
client

#Určení protokolu
proto tcp

#Protekce proti Man in the middle
ns-cert-type server

#Typ routování
dev tun

#ip adresa serveru, kam se připojíme
remote 192.168.19.128

#Uchování stavu, při restartu
persist-key
persist-tun

#Klíče a certifikáty
ca xbedaj01/ca.crt
cert xbedaj01/xbedaj01.crt
key xbedaj01/xbedaj01.key

#Určení šifry
cipher aes-128-cbc

#Komprese
comp-lzo

verb 3

mute 20
```

5.3 Vysvětlení konfiguračních souboru pro server a klienta:

TAP/TUN jedná se o univerzální síťový ovladač, který se skládá ze dvou částí:
TAP - šifruje na linkové vrstvě
TUN - šifruje na síťové vrstvě
dev tun - šifrujeme na síťové vrstvě

Certifikáty na straně serveru:
ca keys/ca.crt - zde je uvedena cesta k certifikátu CA, tento certifikát je veřejný
cert keys/server.crt - cesta k certifikátu server, certifikát je veřejný
cert keys/server.crt - cesta k certifikátu server, certifikát je veřejný
key keys/server.key - cesta ke klíči server, klíč je soukromý
Crl-verify keys/crl.pem - cesta k CRL, Certification Revocation List
keepalive 10 120 - server testuje spojení s klientem, každých 10 sekund provede ping na klienta. Pokud nedojde do 120 sekund odpověď tak rozpojení spojení.
Cipher AES-128-CBC - námi definovaná šifra
comp-lzo - Informujeme Openvpn aby se použila komprese
user nobody - omezení práv pod Linuxem
group nogroup

Klient :
Zde je nastavení obdobné jako u serveru, až na :
ns-cert-type server = Zkontroluje, zda-li byl certifikát vydán pro server. Zabraňuje tak MIM

Veřejné certifikáty mohou být distribuovány, soukromé klíče musí být drženy v tajnosti !

5.4 Spouštění OpenVPN, server

Standardní adresář pro konfiguraci OpenVPN je : */etc/openVPN/*
Zde umístíme konfigurační soubor, nazveme ho *server.conf* a do toho souboru vložíme výše zmíněný kód z bodu 4.1
Dále vytvoříme adresář *keys* : *mkdir /etc/openVPN/keys*
Do adresáře *keys* zkopírujeme následující soubory, tak aby výpis adresáře vypadal následovně: *ca.crt crl.pem dh1024.pem server.crt server.key*
OpenVPN spustíme příkazem: **openVPN server.conf**

```
debian:/etc/openVPN# openVPN server.conf
Sat May 17 22:34:53 2008 OpenVPN 2.0.9 i486-pc-linux-gnu [SSL] [LZO] [EPOLL] built on Sep 20 2007
Sat May 17 22:34:53 2008 Diffie-Hellman initialized with 1024 bit key
Sat May 17 22:34:53 2008 TLS-Auth MTU parms [ L:1560 D:140 EF:40 EB:0 ET:0 EL:0 ]
Sat May 17 22:34:53 2008 TUN/TAP device tun0 opened
Sat May 17 22:34:53 2008 ifconfig tun0 10.8.1.1 pointopoint 10.8.1.2 mtu 1500
Sat May 17 22:34:53 2008 route add -net 10.8.1.0 netmask 255.255.255.0 gw 10.8.1.2
Sat May 17 22:34:53 2008 Data Channel MTU parms [ L:1560 D:1450 EF:60 EB:135 ET:0 EL:0 AF:3/1 ]
Sat May 17 22:34:53 2008 GID set to nogroup
Sat May 17 22:34:53 2008 UID set to nobody
Sat May 17 22:34:53 2008 Listening for incoming TCP connection on 192.168.19.128:1194
Sat May 17 22:34:53 2008 TCPv4_SERVER link local (bound): 192.168.19.128:1194
Sat May 17 22:34:53 2008 TCPv4_SERVER link remote: [undef]
Sat May 17 22:34:53 2008 MULTI: multi_init called, r=256 v=256
Sat May 17 22:34:53 2008 IFCONFIG POOL: base=10.8.1.4 size=62
Sat May 17 22:34:53 2008 MULTI: TCP INIT maxclients=1024 maxevents=1028
Sat May 17 22:34:53 2008 Initialization Sequence Completed
```

Server je nakonfigurován, čeká na připojení klienta.

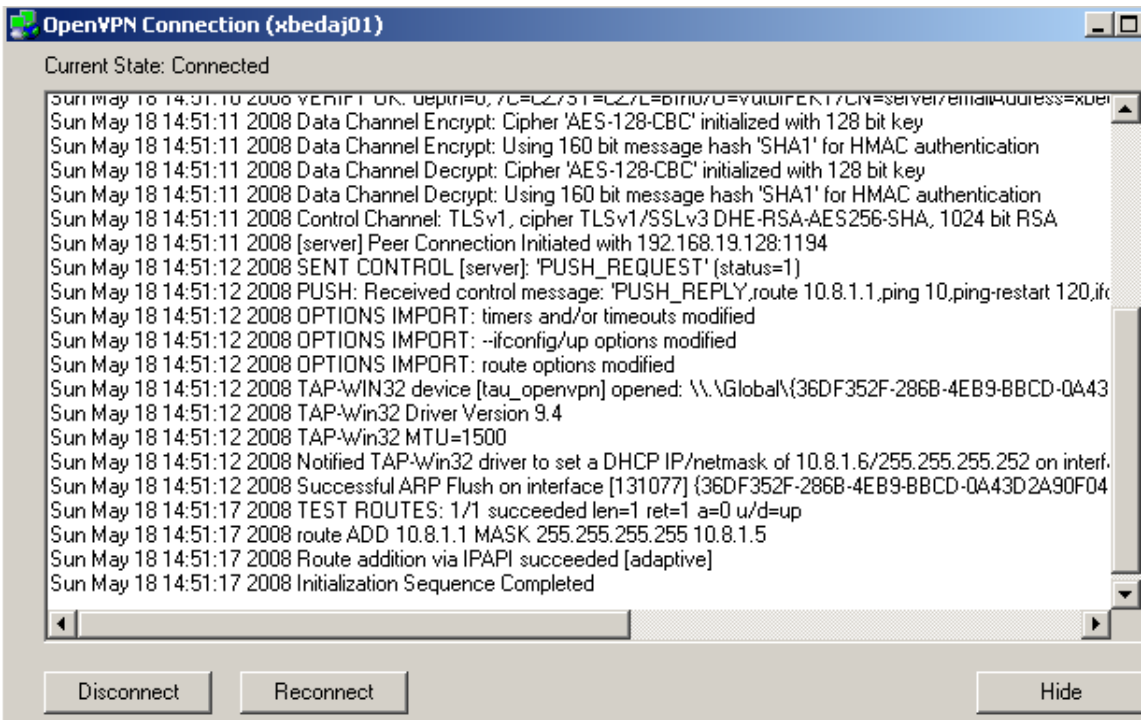
5.5 Spouštění OpenVPN, klient

Pro přihlášení klienta zvolíme operační systém Windows XP, při přihlášení pod Linuxem postupujeme jako při konfiguraci serveru. Rozdíl je pouze v konfiguračním souboru a certifikátech.
Software pro přihlášení stáhneme z oficiálních stránek OpenVPN. Link pro stažení *http://openVPN.net/release/openVPN-2.1_rc7-install.exe*
Po nainstalování softwaru spustíme OpenVPN GUI, toto je grafické rozhraní pro konfiguraci a přihlášení k OpenVPN serveru.

Standardní instalace je provedena do *"C:\Program Files\OpenVPN\"*
Adresář pro konfiguraci je: *C:\Program Files\OpenVPN\config* zde vytvoříme konfigurační soubor pro klienta *xbedaj01.oVPN*. Tento soubor bude obsahovat konfiguraci OpenVPN k připojení k serveru. Tzn. zkopírujeme kód z bodu 4.2 do souboru *xbedaj01.oVPN*.

Konfigurační soubor jsem vytvořili, potřebujeme ještě nahrát certifikát a klíč klienta. Vytvoříme nový adresář *C:\Program Files\OpenVPN\config\xbedaj01* do kterého nahrajeme soubory *ca.crt xbedaj01.crt xbedaj01.key*

Jakmile jsem provedly všechny předchozí kroky vyzkoušíme připojení k OpenVPN serveru obr. 5.1.



Obr. 5.1: OpenVPN GUI

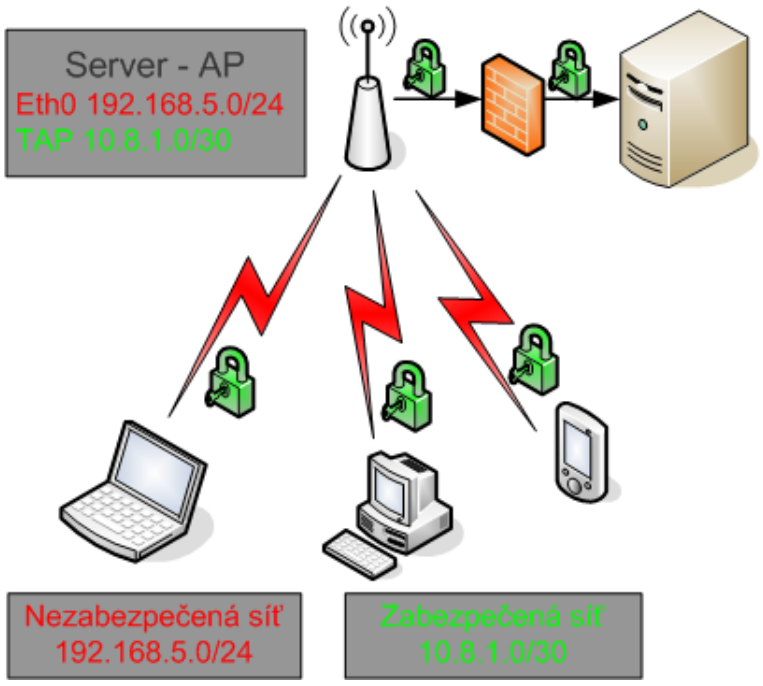
Vidíme, že připojení je úspěšné *Initialization Sequence Completed*
Funkčnost připojení ověříme pomocí příkazu *ping*.

Příkaz PING na 10.8.1.1 s délkou 32 bajtů:

```
Odpověď od 10.8.1.1: bajty=32 čas=2ms TTL=64
Odpověď od 10.8.1.1: bajty=32 čas=3ms TTL=64
Odpověď od 10.8.1.1: bajty=32 čas=1ms TTL=64
Odpověď od 10.8.1.1: bajty=32 čas=2ms TTL=64
```

6 ZABEZPEČENÍ WLAN POMOCÍ OPENVPN

Naším cílem je zabezpečit WLAN (Wireless LAN) pomocí OpenVPN. Budeme vycházet z následujícího schéma zapojení obr. 6.1



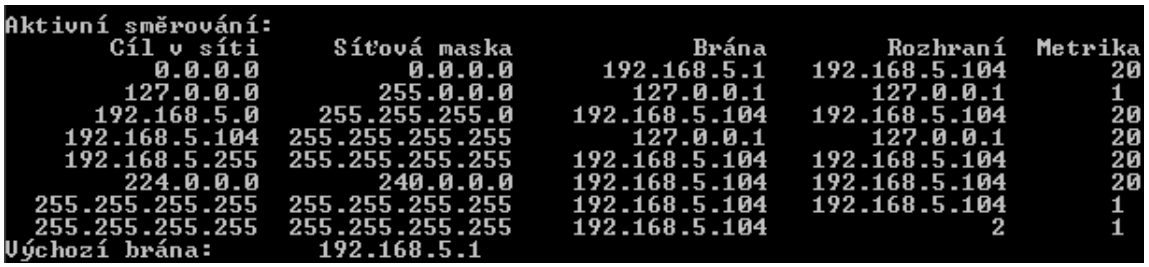
Obr. 6.1: Zabezpečení bezdrátové sítě

Na serveru jsem vytvořili dvě sítě. První síť má síťové rozhraní *eth0*. Druhá síť má rozhraní *TAP*, které bylo vytvořeno pomocí OpenVPN.

- *Síť 192.168.5.0/24 je nezabezpečená, do této sítě se může připojit každý.*
- *Síť 10.8.1.0/30 je zabezpečená, do této sítě se může připojit pouze autentizovaný uživatel.*

Síť *192.168.5.1/0* slouží pouze jako přístupový bod, tzn. uživatel který se připojí do této sítě může komunikovat pouze s *AP (Access Point)*. Dál do vnitřní sítě se již nedostane, správná konfigurace firewallu.

Síť *10.8.1.0/30* vytváří tunel mezi koncovým uživatelem ze sítě *192.168.5.0/24* a serverem, na kterém je nainstalováno OpenVPN. Do sítě *10.8.1.0/30* se může připojit pouze uživatel, který má platný certifikát (viz. vytváření certifikátů). Tomuto uživateli je dále poskytnuta vnitřní síť, která se ukrývá za firewallem.
Při konfiguraci OpenVPN serveru *v bodě 4.1* jsem použili direktivu *push "redirect-gateway"*. Toto nastavení na serveru, zajistí koncovému uživateli přesměrování výchozí brány na OpenVPN bránu.



```
Aktivní směrování:
       Cíl v síti      Síťová maska          Brána       Rozhraní  Metrika
         0.0.0.0          0.0.0.0    192.168.5.1   192.168.5.104       20
       127.0.0.0        255.0.0.0      127.0.0.1       127.0.0.1        1
     192.168.5.0    255.255.255.0  192.168.5.104   192.168.5.104       20
   192.168.5.104  255.255.255.255      127.0.0.1       127.0.0.1       20
   192.168.5.255  255.255.255.255  192.168.5.104   192.168.5.104       20
       224.0.0.0        240.0.0.0  192.168.5.104   192.168.5.104       20
 255.255.255.255  255.255.255.255  192.168.5.104   192.168.5.104        1
 255.255.255.255  255.255.255.255  192.168.5.104               2        1
Výchozí brána:       192.168.5.1
```

Obr. 6.2: Tabulka směrování před připojením k OpenVPN

```
Aktivní směrování:
       Cíl v síti      Síťová maska          Brána       Rozhraní  Metrika
         0.0.0.0          0.0.0.0      10.8.1.5       10.8.1.6        1
        10.8.1.1  255.255.255.255      10.8.1.5       10.8.1.6        1
        10.8.1.4  255.255.255.252      10.8.1.6       10.8.1.6       30
        10.8.1.6  255.255.255.255    127.0.0.1      127.0.0.1       30
  10.255.255.255  255.255.255.255      10.8.1.6       10.8.1.6       30
       127.0.0.0        255.0.0.0    127.0.0.1      127.0.0.1        1
     192.168.5.0    255.255.255.0  192.168.5.104   192.168.5.104       20
   192.168.5.104  255.255.255.255    127.0.0.1      127.0.0.1       20
   192.168.5.112  255.255.255.255    192.168.5.1   192.168.5.104        1
   192.168.5.255  255.255.255.255  192.168.5.104   192.168.5.104       20
       224.0.0.0        240.0.0.0      10.8.1.6       10.8.1.6       30
       224.0.0.0        240.0.0.0  192.168.5.104   192.168.5.104       20
 255.255.255.255  255.255.255.255      10.8.1.6       10.8.1.6        1
 255.255.255.255  255.255.255.255  192.168.5.104   192.168.5.104        1
Výchozí brána:         10.8.1.5
===========================================================================
```

Obr. 6.3: Tabulka směrování po připojením k OpenVPN

Na obr. 6.3 vidíme změnu výchozí brány z *192.168.5.1* na *10.8.1.5*. Tato změna zajistí přesměrování všech požadavků, které se netýkají lokálního nastavení, do námi definovaného rozhraní OpenVPN. Tímto může být zajištěn například přístup na internet nebo přístup do firemní sítě.
Veškerá komunikace je nyní šifrována. Těmito kroky jsem zabezpečili přístup a přenos dat pro bezdrátovou síť.

7 ZÁVĚR

V první části diplomové práce jsem podrobněji popsal infrastrukturu veřejných klíčů. Zaměřil jsem se na rozdíl mezi symetrickými a asymetrickými kryptografickými systémy. Zmínil jsem výhody a nevýhody při distribuci šifrovacích klíčů mezi koncovými systémy. Uvedl jsem teoretické bezpečné délky klíčů a jejich rychlosti při šifrování.

Nainstaloval jsem operační systém Linux Debian 4.0. Na tomto systému jsem nakonfiguroval webový server Apache2 s podporou pro zabezpečený přenos pomocí protokolu HTTPS. Dohrál jsem přídavný modul PHP5, který je využíván při programování webového rozhraní a nastavil jsem OpenVPN.

Naprogramoval jsem webové rozhraní, ve kterém se dá vytvořit, zneplatnit a distribuovat certifikát pro Certifikační autoritu, server nebo koncového uživatele.

Uvedl jsem modelové příklady pro nastavení Openvpn, jak na straně serveru, tak na straně klienta.

V poslední řadě jsme ukázal, jak se dá použít infrastruktura veřejných klíčů k zabezpečení bezdrátové sítě WiFi.

V použití infrastruktury veřejných klíčů vidím budoucnost. Dnes již i malé firmy mají potřebu vybudovat bezpečné vzdálené připojení pro externí pracovníky. Díky kombinaci synchronních a asynchronních kryptografických systémů vznikají bezpečné, ale přesto rychlé šifrovací systémy.
Osobně mě tato tématika velice zajímá a rád bych pokračoval v tomto oboru i po ukončení magisterského studia.

LITERATURA

[1] Dostálek, Libor; Vohnoutová, Marta *Velký pruvodce infrastrukturou PKI a technologií elektronického podpisu.* Brno : Computer Press, 2006 – 534 s. : ISBN: 80-251-0828-7

[2] Barken, Lee. *OpenVPN and the SSL VPN Revolution* GSEC v.1.4B 8.8.2004

[3] Markus Feilner. *OpenVPN: Building and Integrating Virtual Private Networks.* Packt Publishing Ltd ISBN 190481185X

[4] Jaroslav Pinkava, AEC spol. s r.o. & Norman Data Defense Systems, CZ. *Crypto-World* Informační sešit GCUCMP Ročník 3, číslo 9/2001 15. září 2001

[5] RNDr. Libor Dostalek. *Certifikáty a certifikační autority* Cpress 1997

[6] *Openvpn* [online]. c2008 , 2008 [cit. 2008-05-20]. Dostupný z WWW: <http://openvpn.net/>.

[7] *Tutorial Apache Web Login Authentication* [online]. 2000 , 2000, 2001, 2005, 2006 [cit. 2008-05-15]. Dostupný z WWW: <http://www.yolinux.com/TUTORIALS/LinuxTutorialApacheAddingLoginSiteProtection.h

[8] *Install and Configure Apache2 with PHP5 and SSL Support in Debian Etch* [online]. 2006 , 2006,2007 [cit. 2008-05-10]. Dostupný z WWW: <http://www.debianadmin.com/install-and-configure-apache2-with-php5-and-ssl-support-in-debian-etch.html>.

SEZNAM PŘÍLOH

A INDEX.PHP

Zde je naprogramováno základní grafické rozhraní pro webové prostředí, ve kterém se vytváří, zamítají a distribuují certifikáty. Pro příklad uvedu hlavní funkce v programu. Celý program je přiložen v příloze.

```
//INCLUDE FUNKCI ZE SOUBORU OPENVPN.PHP
include("openvpn.php");

switch ( htmlspecialchars($_GET["switch"], ENT_QUOTES)) {
//LAYOUT K VYPSANI CERTIFIKATU
  case ukaz_certifikaty: $ondra->layoutShowCert();
                break;

//LAYOUT K VYTVORENI CERTIFIKATU
        case vytvor_certifikaty: $ondra->layoutCreateCert();
                break;

//LAYOUT K ZAMITNUTI CERTIFIKATU
        case zamitni_certifikaty: $ondra->formRevoke();
                break;

//LAYOUT K DISTRIBUCI CERTIFIKATU
        case distribuce_certifikaty: $ondra->layoutDistribuceCert();
                break;
}
```

B OPENVPN.PHP

Zde jsou naprogramovány veškeré funkce, které jsou použity ve webovém prostředí.

```
function dh
function layoutShowCert
function layoutCreateCert
function layoutDistribuceCert
function getFingerprint
function formCa
function formServer
function formClient
function formRevoke
function createCa
function createServer
function createClient
    .
    .
    .
function generatePassword
```

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

WEP soukromí ekvivalentní drátovým sítím – Wired Equivalent Privacy

WPA Wi-Fi chráněný přístup – Wi-Fi Protected Access

PKI Infrastruktura veřejných klíčů – public key infrastructure

ITU Mezinárodní telekomunikační úřad – International Telecommunications Union

PEM Rozšířené zabezpečení – Privacy Enhanced Mail

PKCS Kryptografie veřejných klíčů – Public Key Cryptography Standards

S-HTTP Zabezpečený HTTP – Secure hypertext transfer protocol

SSL Zabezpečená soketová vrstva – Secure Sockets Layer

TSL Zabezpečená transportní vrstva – Transport Sockets Layer

CPS Certifikační prováděcí směrnice – Certification Practice Statement

CP Certifikační politika – Certificate politics

CRL List zamítnutých certifikátů – Certification Revocation List

CA Certifikační autorita – Certificate authority

RA Registrační autorita – Registration authority

IP Internet protokol – Internet protocol

OSI Síťový model – Open Systems Interconnection

TCP Transportně kontrolní protokol – Transmission Control Protocol

UDP UDP – User Datagram Protocol

DES Datový šifrovací standard – Data Encryption Standard

AES Pokročilý šifrovací standard – Advanced Encryption Standard

HMAC Hash s klíčem – keyed-Hash Message Authentication Code

RSA Rivest, Shamir, Adleman – Rivest, Shamir, Adleman

DH Diffie-Hellman – Diffie-Hellman

MD5 Digitální otisk zprávy – Message-Digest algorithm 5

SHA1 Bezpečný hash algoritmus – Secure Hash Algorithm