



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
Одсек за рачунарство и аутоматику
Катедра за рачунарску технику и рачунарске
комуникације



ДИПЛОМСКИ - МАСТЕР РАД

Кандидат: Дарко Видаковић

Број индекса: E10399

Тема рада: **Једно решење сигурног преноса
података у оквиру система mLunch**

Ментор рада: проф. Др. Никола Теслић

Нови Сад, Мај 2008.



UNIVERZITET U NOVOM SADU • **FAKULTET TEHNIČKIH NAUKA**
21000 NOVI SAD, Trg Dositeja Obradovića 6

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj, RBR :			
Identifikacioni broj, IBR :			
Tip dokumentacije, TD :	Monografska publikacija		
Tip zapisa, TZ :	Tekstualni štampani materijal		
Vrsta rada, VR :	Diplomski-master rad		
Autor, AU :	Darko Vidaković		
Mentor, MN :	prof. Dr. Nikola Teslić		
Naslov rada, NR :			
Jezik publikacije, JP :	Srpski		
Jezik izvoda, Ji :	Srpski		
Zemlja publikovanja, ZP :	Srbija		
Uže geografsko područje, UGP :	Vojvodina		
Godina, GO :	2008		
Izdavač, IZ :			
Mesto i adresa, MA :			
Fizički opis rada, FO : (poglavlja/strana/citata/tabela/slika/grafika/priloga)			
Naučna oblast, NO :	Elektrotehnika i računarska tehnika		
Naučna disciplina, ND :			
Predmetna odrednica/Ključne reči, PO :			
UDK			
Čuva se, ČU :	U biblioteci FTN, Novi Sad, Trg Dositeja Obradovića 6		
Važna napomena, VN :			
Izvod, IZ :			
Datum prihvatanja teme, DP :			
Datum odbrane, DO :			
Članovi komisije, KO :	Predsednik:	prof. Dr. Miodrag Temerinac	
	Član:	prof. Dr. Nebojša Pjevalica	Potpis mentora
	Član, mentor:	prof. Dr. Nikola Teslić	



UNIVERSITY OF NOVI SAD • **FACULTY TECHNICAL SCIENCES**
21000 NOVI SAD, Trg Dositeja Obradovića 6

KEY WORDS DOCUMENTATION

Accession number, ANO :			
Identification number, INO :			
Document type, DT :		Monographic's publication	
Type of record, TR :		Word printed record	
Contents code, CC :		Diploma-Master work	
Author, AU :		Darko Vidaković	
Mentor, MN :		Ph. D.E.E. Nikola Teslić	
Title, TI :			
Language of text, LT :		Serbian	
Language of abstract, LA :		Serbian	
Country of publication, CP :		Serbia	
Locality of publication, LP :		Vojvodina	
Publication year, PY :		2008	
Publisher, PB :			
Publication place, PP :			
Physical description, PD : (chapters/pages/ref./tables/pictures/graphs/appendixes)			
Scientific field, SF :		Electrical engineering and computer science	
Scientific discipline, SD :			
Subject/Key words, S/KW :			
UC			
Holding data, HD :		Library of FTN, Novi Sad, Trg Dositeja Obradovića 6	
Note, N :			
Abstract, AB :			
Accepted by the Scientific Board on, ASB :			
Defended on, DE :			
Defended Board, DB :	President:	Ph. D.E.E. Miodrag Temerinac	
	Member:	Ph. D.E.E. Nebojša Pjevalica	Mentor's sign
	Member, Mentor:	Ph. D.E.E. Nikola Teslić	

SKRAČENICE:

GPRS	- General Packet Radio Service
WiFi	- Wireless Fidelity (IEEE 802.11a/b/g/n wireless networking)
HTTP	- Hyper Text Transfer Protocol
TCP	- Transmission Control Protocol
IP	- Internet Protocol
IEEE	- Institute of Electrical & Electronics Engineers
SQL	- Structured Query Language (database query language)
PHP	- Hypertext Preprocessor (HTML-embedded scripting language)
GSM	- Global System for Mobile Communications
SMS	- Short Message Service
MMS	- Multimedia Messaging Service
EDGE	- Enhanced Data for GSM Evolution
SDK	- Software Development Kit
RAM	- Random-Access Memory
ROM	- Read-Only Memory
QVGA	- Quarter Video Graphics Array
USB	- Universal Serial Bus
URL	- Uniform Resource Locator (world wide web address)
ARP	- Address Resolution Protocol
IDS	- Intrusion detection system
DMZ	- Demilitarized zone
NIDS	- Network-based Intrusion Detection System
PBIDS	- Protocol based Intrusion Detection System
APIDS	- Application-based Intrusion Detection System
PIDS	- Perimeter Intrusion Detection System
HIDS	- Host-based Intrusion Detection System
IPS	- Intrusion Prevention System
DES	- Data Encryption Standard
AES	- Advanced Encryption Standard
IDEA	- International Data Encryption Algorithm
RSA	- Rives-Shamir-Adleman
DSA	- Digital Signature Algorithm
SSL/TLS	- Secure Sockets Layer/Transport Layer Security
PKI	- Public Key Infrastructure
VPN	- Virtual Private Network
IETF	- Internet Engineering Task Force
SRP	- Secure Remote Password Protocol
PSK	- Pre-shared Key
MD5	- Message-Digest algorithm 5
SHA	- Secure Hash Algorithm
MAC	- Message Authentication Code
DNS	- Domain Name Server
CA	- Certificate Authority
RA	- Registration Authority
MIME	- Multipurpose Internet Mail Extensions

SADRŽAJ:

1. UVOD	7
1.1 Osnovni zadaci rada.....	7
2. ARHITEKTURA MLUNCH SISTEMA.....	9
2.1 Klijentska strana komunikacije	10
1.1.1 Klijent – korisnik PC	10
1.1.2 Klijent – mobilni skener uređaj	10
2.2 Serverska strana komunikacije	10
2.3 Razvjno okruženje	11
3. SIGURAN PRENOS PODATKA.....	13
3.1 Sigurnosni rizici u prenosu podataka.....	14
3.2 Mere zaštite računarskih mreža i metode sigurnog prenosa podataka	14
3.2.1 Sistem za otkrivanje neovlašćenog pristupa računarskoj mreži ^{[3] [10]}	15
3.2.2.1 Kriptografija simetričnog ključa ^[9]	16
3.2.2.2 Kriptografija javnog ključa ^[5]	17
3.2.2.3 Kriptoanaliza ^{[8] [7]}	18
3.2.3 Kriptografski protokoli.....	20
3.2.3.1 Princip rada kriptografskih protokola ^{[11] [12]}	20
3.2.3.2 Faze uspostave sigurne veze kod kriptografskih protokola – protokol rukovanja..	21
3.2.3.3 Sigurnost kriptografskih protokola ^{[11] [12]}	23
3.2.4 Sistem za izadavanje digitalnih ključeva i identifikaciju korisnika ^[6]	24
3.2.5 Virtuelna privatna mreža ^[13]	24
3.2.5.1 Kategorizacija virtuelnih privatnih mreža na osnovu nivoa sigurnosti ^[14]	25
4. REALIZACIJA.....	26
4.1 Realizacija virtuelne privatne mreže – OpenVPN ^{[15] [16]}	27
4.1.1 Tehnike upravljanja tokom saobraćaja unutar virtuelne privatne mreže.....	27
4.1.2 Stvaranje digitalnih sertifikata potrebnih unutar OpenVPN mreže.....	28
4.1.2.1 Stvaranje sertifikata i privatnog ključa entiteta koje izdaje sertifikate.....	28
4.1.2.2 Stvaranje sertifikata i privatnog ključa za servera.....	29
4.1.2.3 Stvaranje sertifikata i privatnog ključa za klijente	30
4.1.2.4 Inicijalizacija Diffie Hellman parametara	30
4.1.2.5 Ključne datoteke	30
4.1.3 Implementiranje konfiguracionih datoteka servera i klijenata	31
4.1.3.1 Konfiguraciona datoteka OpenVPN servera	31
4.1.3.2 Konfiguraciona datoteka OpenVPN klijenta.....	33
4.1.4 Dodatne mere sigurnosti unutar OpenVPN sistema	34
4.1.4.1 Tls-auth direktiva.....	34
4.1.4.2 ProtoUdp direktiva	35
4.1.4.3 Dužina javnih ključeva	35
4.1.4.4 Dužina simetričnih ključeva	35
4.1.4.5 Čuvanje privatnog ključa izdavaoca sertifikata.....	35
4.1.5 Izbegavanje mogućih napada “Čovek u sredini”.....	35
4.2 Realizacija sistema za siguran prenos podataka implementacijom kriptografskih protokola i sistema za proveru identiteta korisnika.....	36
4.2.1 Konfiguracija Apache web servera za rad sa sistemom za proveru identiteta korisnika..	37
4.2.2 Realizacija i integrisanje sistema za proveru identiteta korisnika i kriptografskih protokola u sklopu skener aplikacije klijentskog uređaja.....	40
4.2.2.1 ContainerForm moduo.....	41
4.2.2.2 MainForm, GetRequestForm i SyncForm moduli.....	42

4.2.2.3	OptionsForm, SSLOptionsForm, OptionsStorage i FileIO moduli.....	43
4.2.2.4	HTTPBaseClass, CertificateCheck, CertInfo i CertInfoForm moduli	46
5.	TESTIRANJE I VERIFIKACIJA	50
5.1	Testiranje otpornosti sistema na sigurnosne napade.....	50
5.2	Analiza protoka podataka i odziva sistema	52
6.	ZAKLJUČAK	54
7.	LITERATURA.....	55
8.	PRILOG	56
8.1	Primeri sigurnosnih napada tipa “čovjek u sredini” ^[1]	56
8.1.1	Mitnikov napad ^[2]	56
8.1.2	Metod maskiranja adrese web stranice ^[4]	58
8.1.3	Metod maskiranja paketa za rezoluciju fizičkih adresa računara ^[1]	58
8.2	Rad sa 51. konferencije ETRAN-a	59

1. UVOD

Ubrzani razvoj tehnologije i međuračunarskih komunikacija neminovno nameće potrebu za sigurnim prenosom podataka. Pod sigurno prenetim podatkom se podrazumeva podatak ili informacija koja je uspešno prenesena između dva ili više entiteta, a da pri tom ostali entiteti unutar mreže za prenos nemaju pristup tim informacijama. Pojava korisničkih aplikacija koje koriste globalnu Internet mrežu za prenos osetljivih i poverljivih informacija (elektronsko bankarstvo, informacioni sistemi, sistemi za razmenu elektronske pošte...) je samo ubrzala razvoj mrežnih kriptografskih protokola za siguran prenos podatka kao i sistema za uspešnu identifikaciju učesnika komunikacije u obliku digitalnih sertifikata i ključeva.

Proteklih godina razvoj u oblasti računarske tehnike i računarskih komunikacija se uveliko preneo u domen prenosivih mobilnih uređaja koji se zbog svojih osobina oslanjaju na bežičnu komunikaciju. Upravo u ovakvom vidu komunikacije usko grlo predstavlja siguran prenos informacije uzimajući u obzir da je informacija koja se prenosi preko etra dostupna skoro svima unutar dometa samog uređaja i same mreže za prenos podatka. Jedan primer informacionog sistema koji se oslanja na bežičnu komunikaciju između klijenta i servera jeste mLunch sistem za elektronsko naručivanje obroka u restoranima. Kod ovog sistema klijenstska strana koja se nalazi na mobilnom uređaju (Pocket PC) ima potrebu za sigurnim prenosom i sinhronizacijom podatka sa serverom zbog činjenice da se za prenos podatka koristi bežična GPRS ili WiFi mreža.

1.1 Osnovni zadaci rada

Osnovni zadatak rada je razvoj programskog rešenja koje omogućuje siguran i pouzdan prenos podataka između klijenta i servera unutar mLunch informacionog sistema za elektronsko naručivanje obroka. Pod klijentskom stranom se podrazumeva mobilni uređaj koji se nalazi u restoranu i koji je zasnovan na Pocket PC Windows Mobile 5.0 platformi dok se server nalazi na PC računaru u kome se nalaze sve potrebne informacije o naručivanju i naplati obroka. Pod informacijama koje se prenose između uređaja – klijenta i servera se podrazumevaju:

- Informacije vezane za proveru statusa narudžbine datog korisnika
- Potvrda o ručavanju datog korisnika u datom restoranu
- Sinhronizaciju baze korisnika unutar sistema
- Informacije vezane za pregled broja narudžbina za trenutni dan

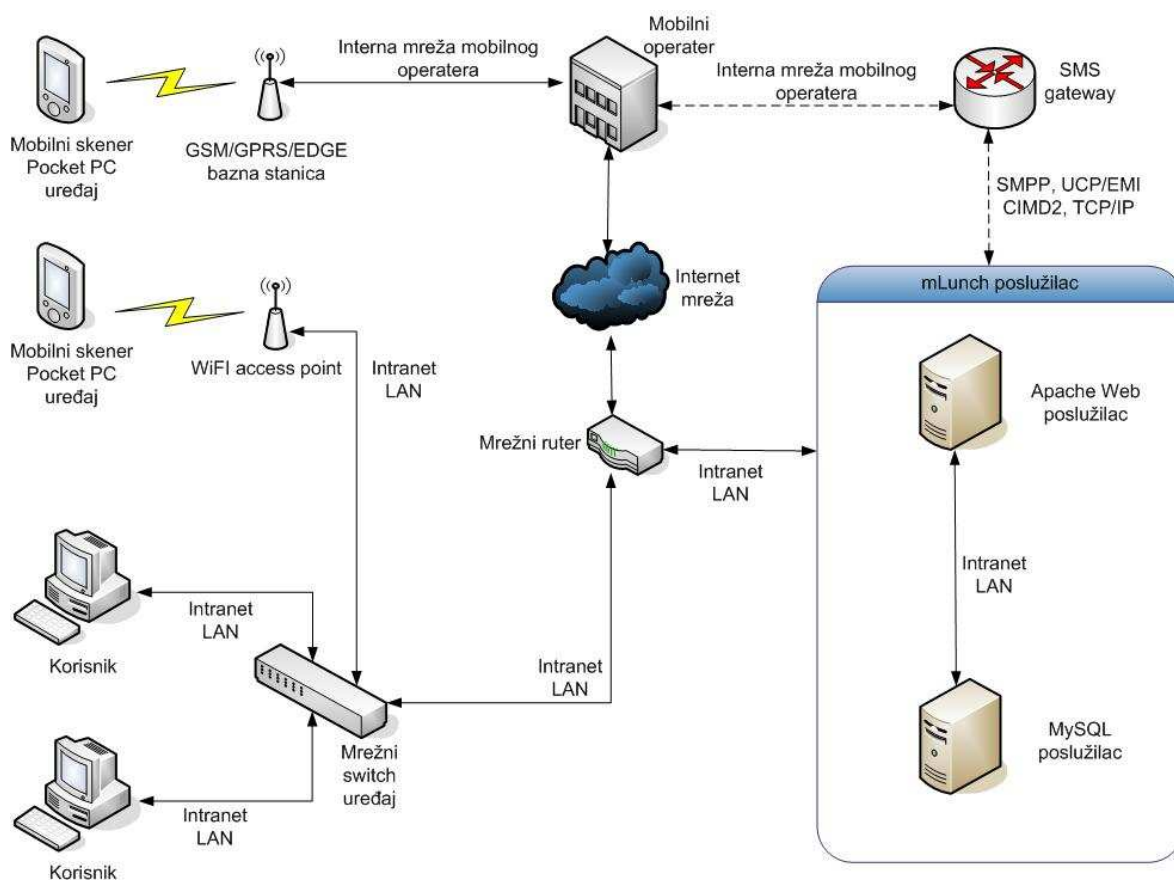
Programsko rešenje je potrebno realizovati u sklopu mLunch aplikacije na mobilnom Pocket PC uređaju sa operativnim sistemom Windows Mobile 5.0 kao i na strani servera u sklopu Apache web servera na Widnows XP operativnom sistemu.

Rad je organizovan u VIII poglavlja:

- o Poglavlje “Uvod” sadrži opis zadatka rada.
- o Poglavlje “Arhitektura mLunch sistema” opisuje rad inforamcionog sistema mLunch sa stanovišta arhitekture mrežnog povezivanja i komunikacije klijenata i servera, sa osvrtom na rizike u pogledu privatnosti i prenosa podataka unutar sistema.
- o Poglavlje “Siguran prenos podataka” se bavi sigurnosnim problemima koji nastaju u modernim informacionim sistemima. Drugi deo ovog poglavlja prikazuje neke od tehnika i mera zaštite privatnosti i samih podataka unutar ovih sistema.
- o Poglavlje “Realizacija” daje opis realizacije dva rešenja za siguran i pouzdan prenos podataka koja su primenjena u ovom radu: VPN mreža i integracija HTTPS protokola uz pomoć SSL/TLS i PKI infrastrukture na klijentskoj i serverskoj strani mLunch sistema.
- o Poglavlje “Testiranje i verifikacija” se bavi testiranjem i proverom ispravnosti rada klijentske aplikacije mobilnog skener uređaja.
- o Poglavlje “Zaključak” daje kratki osvrt na realizovani sistem i ukazuje na potencijalna ograničenja i dalji razvoj klijentske aplikacije i samog sistema.
- o Poglavlje “Literatura” sadrži prikaz literature koja je korištena prilikom izrade rada.
- o Poglavlje “Dodatak” sadrži dodatne informacije i dokumenta koji su povezani sa temom ovog rada.

2. ARHITEKTURA MLUNCH SISTEMA

Unutar sistema mLunch razlikuju se klijentska i serverska strana komunikacije. Kao i kod gotovo svih informacionih sistema, uloga servera je da odgovara na upite od strane klijenata kao i da čuva, uklanja ili menja podatke koji se preko ovih upita prosleđuju. Na slici 1. je prikazana arhitektura sistema sa stanovišta mrežnog povezivanja i prenosa podataka.



Slika 1 : Arhitektura mreže za prenos podataka mLunch sistema

2.1 Klijentska strana komunikacije

U okviru komunikacije unutar mLunch sistema razlikujemo dva tipa klijenata:

- Klijent – korisnik, koji sistemu pristupa putem PC računara radi naručivanja obroka
- Klijent – bar kod skener uređaj, koji sistemu pristupa putem mobilnog uređaja radi provere i potvrde naručenog jela

1.1.1 Klijent – korisnik PC

Korisnik je u mogućnosti da pristupanjem korisničkom portalu koji se nalazi na serveru vrši naručivanje obroka za prestojeće dane. Komunikacija između korisnika i servera se odvija preko HTTP protokola koji se oslanja na TCP protokol nižeg nivoa za sam prenos podatka unutar Intranet mreže. Takođe postoji mogućnost da se ova komunikacija odvija i putem globalne Internet infrastrukture.

1.1.2 Klijent – mobilni skener uređaj

Skener uređaj se koristi na strani restorana i njegova uloga je identifikacija korisnika koji je prethodno naručio obrok u tom restoranu preko korisničkog portala. Kao i u prethodnom slučaju i ovde se koristi HTTP protokol sa TCP protokolom za prenos podatka. Gotovo u većini slučajeva mobilni skener uređaji se ne nalaze u okviru lokalne Intranet mreže samog servera tako da se sva komunikacija obavlja putem Internet mreže, dok se za fizički način povezivanja koristi bežična mreža (npr. mreže mobilnih operatera – GPRS ili bežične računarske mreže zasnovane na IEEE 802.11 a/b/g standardu).

2.2 Serverska strana komunikacije

Sa strane servera sistem se može podeliti na više delova koji se mogu nalaziti unutar jednog ili više servera:

- Apache web server – nadležan za komunikaciju HTTP protokolom
- MySQL server – realizuje bazu podatka kao i mehanizam upita i ažuriranja iste

U sklopu Apache servera nalazi se integrisani PHP 4.0 skriptni jezik koji se koristi za kreiranje dinamičkih internet stranica uz pomoć kojeg je realizovan portal za naručivanje jela od strane korisnika. Takođe, PHP se koristi i za kreiranje odgovora na upite skener uređaja prilikom provere i potvrde narudžbina. Još jedna uloga servera je da na zahtev administratora sistema izda i pošalje poruku putem SMS-a ili MMS-a odgovarajućem korisniku u obliku sličice dvodimenzionalnog bar koda koji identifikuje datog korisnika.

Server je u sklopu svoje arhitekture povezan i sa klijent - korisnicima u okviru Intranet mreže i sa klijent - skener uređajima u okviru Internet mreže i mreže mobilnog operatora. U uslovima eksperimentalnog rada mLunch sistema i malog broja korisnika, slanje poruka putem SMS-a se obavlja preko GSM/GPRS modema koji je povezan sa mLunch serverom. Ukoliko su zahtevi servera veći usled velike količine korisnika i većeg broja zahteva za slanjem identifikacionih poruka (sličica), mLunch server se može povezati direktno na SMS kapiju (*eng. SMS gateway*) radi bržeg slanja SMS poruka kao i mogućnosti slanja velikog broja poruka u isto vreme.

2.3 Razvjno okruženje

Za potrebe realizacije ovog zadatka korišćen je mobilni Pocket PC uređaj Motorola Symbol MC70 (slika 2.) zasnovan na Windows Mobile 5.0 operativnom sistemu sa podrškom za Compact .NET 2.0 framework aplikacije. Razvoj klijentske aplikacije mLunch sistema realizovan je u Microsoft Visual Studio 2005 razvojnom okruženju sa dodatkom Windows Mobile 5.0 SDK (Pocket PC) koji u sebi sadrži odgovarajući emulator mobilnog uređaja koji omogućuje testiranje i razvoj aplikacija bez posedovanja samog uređaja.



Slika 2 : Motorola Symbol MC70 Pocket PC

Sa stanovišta fizičke arhitekture uređaja, Motorola Symbol MC70 sadrži sledeće bitnije karakteristike:

- Procesor Intel® XScale™ sa radnim taktom od 624 Mhz
- Operativni sistem Microsoft Windows Mobile 5.0
- 64MB RAM i 128MB ROM uz mogućnost proširenja memorije uz pomoć SD memorijskih fleš (*eng. flash*) kartica.
- Transflective color 3.5" QVGA ekran osetljiv na dodir (*eng. touchscreen*) sa rezolucijom od 240 x 320 piksela i 64000 boja.
- Linearni 1D i 2D bar kod skener optičke rezolucije 640 x 480 piksela – *grayscale*
- Podrška za GSM/EDGE (850, 900, 1800 and 1900 MHz)
- Bežična komunikacija Tri-mode IEEE® 802.11a/b/g
- Mogućnost povezivanja sa ostalim uređajima preko Bluetooth, RS-232 i USB 1.1 porta

Pored standardnih funkcija koje podržava svaki Pocket PC uređaj, MC70 se odlikuje i dodatkom u vidu bar kod čitača, lociranim na gornjoj strani uređaja. Ovaj čitač je u mogućnosti

da dekoduje jednodimenzionalne i dvodimenzionalne bar kodove koje predhodno prepozna sa ekrana korisničkog mobilnog telefona. Uređaj se povezuje bežičnim putem sa mobilnom GSM/GPRS mrežom nekog od mobilnih operatera i komunikaciju sa serverom ostvaruje preko iste. Ovakav način povezivanja se odlikuje visokom fleksibilnošću jer daje mogućnost pristupu mLunch sistemu sa bilo kog mesta gde posotji signal mobilnog operatera. Za potrebe ovog zadatka korišćena je mobilna mreža Telenor Srbija operatera i njihova GPRS/EDGE mreža za prenos podatka.

Na strani servera se nalazi PC računar sa Windows XP operativnim sistemom u kojem je instaliran Apache Web server i PHP skriptni jezik za kreiranje dinamičkih Internet prezentacija. U sklopu mLunch servera je i MySQL server zadužen za manipulaciju bazom podatka, tj. podacima sistema kao što su podaci o korisnicima i njihovim narudžbinama. Radi slanja dvodimenzionalnih bar kodova na mobilne telefone korisnika mLunch server je proširen USB GSM/GPRS modemom koji sistemu daje mogućnost slanja SMS/MMS poruka.

3. SIGURAN PRENOS PODATKA

Jedan od osnovnih problema informacionih sistema jeste siguran prenos osetljivih informacija putem računarskih mreža. U zavisnosti od konfiguracije same mreže razlikuju se i kriterijumi sigurnosti. Generalno, kod informacionih sistema kod kojih se nalaže pristup informacijama koje su izvan dometa lokalne mreže sistema posebno se obraća pažnja sigurnom protoku informacija. Mreža za prenos podatka unutar mLunch informacionog sistema se može podeliti na dva segmenta:

- Internet – globalna svetska računarska mreža
- Intranet – lokalna mreža unutar informacionog sistema (objekta, kompanije...)

Sa slike 1. se može zaključiti da se klijenti, koji mLunch sistemu pristupaju radi naručivanja jela, kao i sam mLunch server nalaze unutar Intranet lokalne mreže dok se skener uređaji, koji su locirani u restoranima, na sistem povezuju putem globalne Internet mreže. Analizirajući protok podatka unutar mLunch sistema dolazi se do zaključka da najveći rizik po sigurnost podataka leži u delu mreže za prenos koja se oslanja na globalnu Internet mrežu. Konkretno, deo komunikacije između mobilnog skener uređaja i mLunch servera koji se odvija preko mobilne mreže operatera i Interneta predstavlja najveći rizik po sigurnost mLunch sistema. Kriterijumi sigurnosti na ovom delu mreže moraju biti strogi i tačno određeni ne bi li se postigao zadovoljavajući nivo sigurnosti informacija koje se prenose.

Razmena informacija između klijenata - korisnika koji vrše naručivanje jela kao i međusobna komunikacija Apache i MySQL servera unutar mLunch servera se odvija unutar lokalne Intranet mreže. Pod pretpostavkom da je Intranet mreža isprojektovana tako da daje veliki stepen sigurnosti podatka koji se prenose unutar nje, kriterijumi sigurnog prenosa podatka u Intranet segmentu mreže su manje strogi od kriterijuma koji su vezani za Internet segment mreže.

Pored fizičke mreže, za pouzdan prenos podataka unutar računarske mreže i sistema mLunch korisiti se globalno prihvaćen TCP protokol. Još jedan od razloga upotrebe ovog protokola jeste i činjenica da Apache Web server koristi HTTP protokol višeg nivoa za primanje upita i slanje odgovora klijentima mLunch sistema. HTTP protokol se direktno oslanja na TCP protokol radi prenosa podatka.

3.1 Sigurnosni rizici u prenosu podataka

Postoje različiti oblici napada na sigurnost mreža za prenos podataka. Generalna vrsta napada nosi ime *Man in the middle attack* (napad tipa "Čovek u sredini"). Ideja iza ovog vida napada je pozicioniranje napadača između dve strane koje učestvuju u komunikaciji, pristup podacima koji se prenose, izmena podatka i prosleđivanje odredišnoj strani. Izraz *Man in the middle* se koristi još od 1994. godine. Postoji nekoliko varijacija ove vrste napada, ali generalna definicija se može predstaviti kao "Sigurnosni prekršaj u računarskoj komunikaciji u kojoj maliciozni napadač preuzima i po mogućnosti menja sadržaj podataka koji se prenose mrežom". Bez identifikacije strana u komunikaciji (eng. *authentication*) klijenti nisu u stanju da utvrde da li su žrtve *Man in the middle* napada.

3.2 Mere zaštite računarskih mreža i metode sigurnog prenosa podatka

Zaštita računarskih mreža od neovlašćenog pristupa kao i siguran prenos podatka kroz globalnu računarsku mrežu kao što je Internet su jedna od glavnih tema i briga svakog ozbiljnijeg računarskog sistema. Razvoj širakopojasnog pristupa Internetu, kao i povećanje broja korisnika iz dana u dan, proporcionalno povećava i rizik od sigurnosnih prekršaja i nanošenja štete sistemima koji rukuju osetljivim informacijama. Kako su metode napada na sigurnost računarskih mreža postale sve složenije i raznovrsnije tako su napredovale i alatke i sistemi za prevenciju tih napada. Može se slobodno reći da se najveća pažnja prilikom implementacije nekog informacionog sistema posvećuje projektovanju računarskih mreža i sigurnosnih modula unutar aplikacija koje tom sistemu pristupaju. Postoje različite tehnike zaštite sistema koje u zavisnosti od implementacije i potreba mogu biti u sklopu jednog sistema ili kao posebni sistemi (uređaji) unutar računarske mreže.

Potreba za adekvatnom zaštitom računarskih mreža i sistema nameće određene zahteve prilikom implementacije informacionog sistema i same računarske mreže. Neki od tih zahteva su:

- Adekvatno projektovanje mrežne infrastrukture koja zadovoljava potrebe informacionog sistema.
- Instalacija efikasnog antivirus i firewall softvera (ili hardvera) u sve računare koji čine informacioni sistem, kao i redovno održavanje istih.
- Implementacija sistema za otkrivanje neovlašćenog pristupa računarskoj mreži – IDS (eng. *intrusion detection sistem*).
- Integrisanje kriptografskih protokola SSL/TLS za siguran prenos podatka u aplikacijama na strani servera i na strani klijenata.
- Uvođenje PKI (eng. *public key infrastructure*) sistema za identifikaciju korisnika uz pomoć izdavanja digitalnih ključeva i sertifikata (eng. *certificate*) svim stranama koje učestvuju u komunikaciji unutar informacionog sistema
- Implementacija virtuelne privatne mreže VPN (eng. *Virtual Private Network*).

3.2.1 Sistem za otkrivanje neovlašćenog pristupa računarskoj mreži ^{[3] [10]}

Sistem za otkrivanje neovlašćenog pristupa računarskoj mreži - IDS se koristi kod ranog otkrivanja sumnjivog ponašanja unutar računarske mreže koje može dovesti do narušavanja sigurnosti iste. Pod sumnjivim ponašanjem se podrazumevaju napadi na osetljive usluge (servise) koji su implementirani u računarskim sistemima unutar mreže, prosleđivanje pogrešnih podataka aplikacijama, napadi na servere radi pridobijanja administratorskih privilegija i neovlašćenog upravljanja sistemom kao i instaliranje neovlašćenog softvera u obliku virusa, trojanaca i slično.

IDS je podeljen u više podsistema:

- Senzori – zaduženi za otkrivanje i stvaranje sigurnosnih događaja.
- Konzola – koristi se za praćenje događaja i upozorenja, kao i kontrolu senzora.
- Jezgro – čuva događaje koji su otkriveni od strane senzora u vidu baze podataka i koristi kolekciju pravila za stvaranje upozorenja na sigurnosne događaje unutar mreže.

Postoji nekoliko načina na koje se IDS sistem može podeliti u zavisnosti od vrste i lokacije senzora i metodologije stvaranja sigurnosnih upozorenja od strane jezgra sistema. U većini prostih implementacija IDS sistema sve tri komponente sistema se nalaze u okviru istog uređaja tj. sistema.

U mrežno baziranim IDS sistemima – NIDS (eng. *Network-based intrusion detection system*) senzori se postavljaju u delovima mreže koji pripadaju tzv. demilitarizovanoj zoni računarske mreže – DMZ (eng. *Demilitarized zone*) ili na samoj granici lokalne i globalne mreže (ruteri, kapije – *gateway* i sl.). Senzori preuzimaju sav mrežni saobraćaj koji spolja ulazi u lokalnu računarsku mrežu i analiziraju sadržaj svakog paketa. Kod ovih sistema se koriste PIDS (eng. *Perimeter Intrusion Detection System*) podsistemi za praćenje transpornih protokola. Druga vrsta sistema su HIDS sistemi (eng. *Host-based intrusion detection system*). Kod ove vrste sistema senzori se sastoje od softverskih agenata koji imaju ulogu praćenja aktivnosti individualnog računarskog sistema na kome se nalaze. Mešanjem ove dve vrste IDS sistema nastaju i hibridne verzije IDS sistema.

Još neke od podela IDS sistema:

- NIDS – realizovan kao nezavisna platforma koja identifikuje sumnjivo ponašanje i neovlašćen pristup mreži tako što analizira mrežni saobraćaj i posmatra učesnike u komunikaciji. NIDS sistem se obično povezuje na mrežni uređaj (hub, switch) koji ima mogućnost prisluškivanja komunikacionih portova (eng. *port mirroring, network tap*).
- PBIDS (eng. *Protocol based intrusion detection system*) se sastoji od agenata koji praktično stoje ispred servera i posmatraju i analiziraju komunikaciju između povezanih uređaja (korisnik/PC). Za jednog web servera, kao što je npr. Apache, ovakav IDS sistem bi u praksi posmatrao HTTP protokol uz razumevanja svrhe i načina korišćenja datog servera.
- APIDS (eng. *Application-based intrusion detection system*) se sastoji od agenata koji su obično smešteni unutar jedne grupe servera. Unutar takve grupe ovi agenti imaju zadatak da analiziraju komunikaciju preko protokola koji su specifični za sam sistem servera. Na primeru jednog web servera koji je povezan sa SQL serverom baze podataka ovakav IDS sistem bi analizirao SQL protokol i sve transakcije koje nastaju između ovih servera.
- HIDS – se sastoji od agenata koji su smešteni unutar samog računarskog sistema (npr. PC računara) i čiji je zadatak identifikacija neovlašćenog korišćenja sistema analizom sistemskih poziva, aplikacionih logova, izmena unutar sistema datoteka (eng. *file system*) i ostalih aktivnosti.
- Hibridni IDS – nastaje kombinacijom jednog ili više modela IDS sistema. Agenti unutar računarskog sistema (HIDS) kombinuju svoje podatke sa agentima unutar

mrežnog IDS sistema (NIDS) kako bi stvorili kompletnu sliku informacionog sistema i efikasnije zaštitili isti od napada i neovlašćenog korišćenja.

Pored navedene podele koja je zasnovana na implementaciji i funkcionalnosti IDS sistema, postoji još jedna podela koja IDS sisteme, po načinu reagovanja sistema na sumnjivo ponašanje unutar mreže, deli na pasivne i reaktivne IDS sisteme. U slučaju pasivnih IDS sistema senzori detektuju potencijalno neovlašćen pristup mreži, zapisuju osobine i karakter tog incidenta i šalju obaveštenje o tome konzoli ili nadležnom licu. Kod reaktivnih IDS sistema (koji takođe nose naziv IPS – eng. *Intrusion prevention system*) sistem pored detekcije i upozorenja o incidentu preuzima odgovarajuće sigurnosne mere u vidu prekida komunikacije sa izvorom problema (npr. programiranje firewall aplikacije da privremeno ili zauvek zaustavlja komunikaciju sa datim korisnikom). Reagovanje ove vrste IDS sistema na potencijalni rizik se može pokrenuti automatski ili reagovanjem samog operatora sistema.

Gledano sa strane sigurnosti računarskih mreža IDS sistemi su slični firewall aplikacijama, ali neke suštinske razlike postoje. Firewall aplikacije ili uređaji sprečavaju napade na sigurnost koji dolaze iz spoljašnjosti i ograničavaju pristup spoljnim računarskim mrežama kao odgovor na napade i ne ukazuju na napade iz unutrašnjosti mreže. Sa druge strane, IDS sistem uočava sumnjivo ponašanje unutar mreže i zatim šalje odgovarajuće signale upozorenja. IDS sistemi takođe reaguju na napade čiji su izvori unutar računarskog sistema i računarskih mreža.

3.2.2 Kriptografija ^[5]

Kriptografija (eng. *cryptography*) – disciplina usko vezana za matematiku i računarsku nauku koja se bavi proučavanjem metoda skrivanja informacija. U ranijem dobu, kriptografija se uglavnom bavila procesom pretvaranja informacija koje su u obliku običnog teksta (eng. *plaintext*) u neshvatljiv niz znakova - slova (eng. *ciphertext*). Ovaj proces se zove enkripcija (eng. *encryption*) tj. šifrovanje informacija. Obrnuti proces, tj. dešifrovanje – dekrpicija (eng. *decryption*) je proces pretvaranja neshvatljivog niza znakova u originalni niz znakova tj. razumljivu informaciju. Takozvani “sifer” (eng. *cipher* ili *cypher*) predstavlja par algoritama koji se koriste u procesu šifrovanja i dešifrovanja informacija. Zajedno sa odgovarajućim algoritmima, cipher sadrži i odgovarajući ključ (eng. *key*). Ključ predstavlja tajni parametar (u idealnom slučaju poznat samo učesnicima u komunikaciji) prilikom prenosa informacija. Ključevi su veoma bitni faktori u komunikaciji, jer cipher algoritmi sa slabim ili nepromenljivim ključevima su dosta laki za probijanje i kao takvi manje korisni za siguran prenos podataka. Pre nego što je predstavljen PKI princip, cipher-i su se često koristili direktno u procesu šifrovanja i dešifrovanja informacija bez dodatnih procedura provere identiteta i integriteta strana koje učestvuju u komunikaciji.

Moderna kriptografija je podeljena u nekoliko oblasti primene i proučavanja. Neke od njih su:

- Kriptografija simetričnog ključa (eng. *symetric-key cryptography*)
- Kriptografija javnog ključa (eng. *public-key cryptography*)
- Kriptoanaliza (eng. *cryptoanalysis*)

3.2.2.1 Kriptografija simetričnog ključa ^[9]

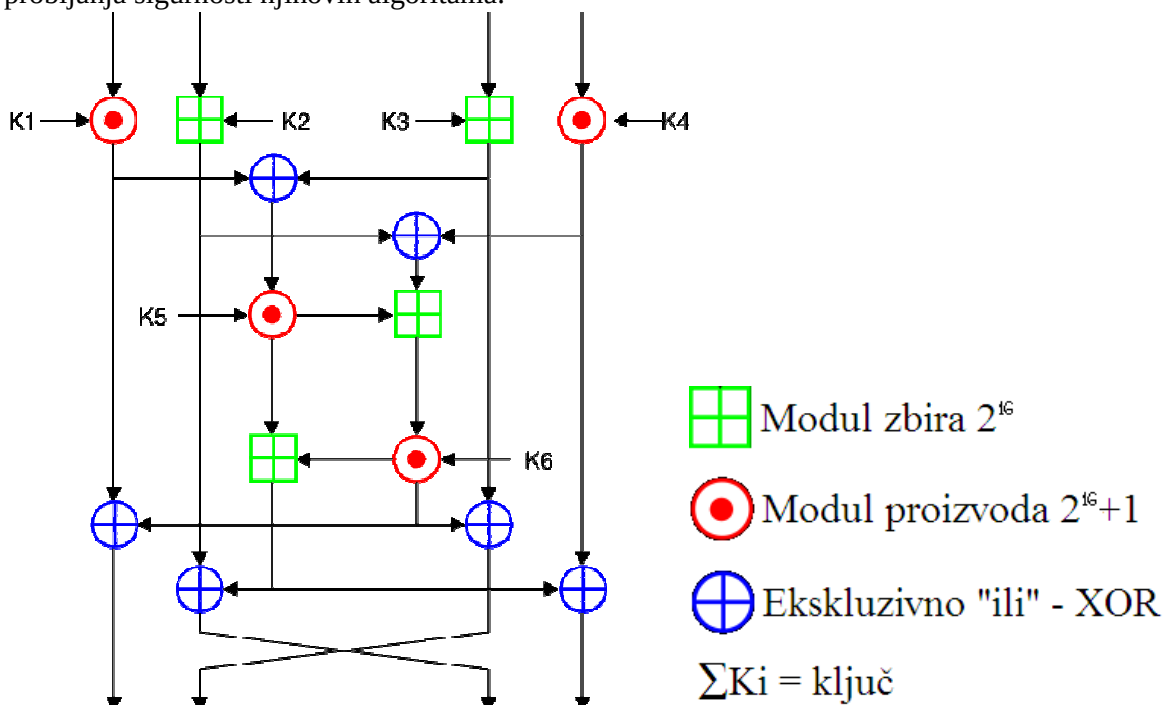
Kriptografija simetričnog ključa označava metode šifrovanja informacija u kojima obe strane komunikacije (onaj koji šalje i onaj koji prima podatke) dele isti ključ (u retkim slučajevima ključevi su različiti, ali se prostom matematičkom funkcijom jedan ključ može izvesti iz drugog). Ovo je bio jedini javno poznat način šifrovanja informacija sve do juna 1976. godine.

U modernom proučavanju kriptografije algoritmi (*ciphers*) simetričnog ključa su povezani uglavnom sa proučavanjem tzv. *block-cipher* i *stream-cipher* algoritama. Block cipher algoritmi za ulazne podatke uzimaju određeni blok podataka i ključ, a na izlazu daju blok šifrovanih podataka (*ciphertext*) iste dužine kao i ulaz. Pošto su informacije skoro uvek veće od

veliĉine jednog bloka, nalaŹe se potreba za metodama povezivanja sukcesivnih blokova. Nekoliko metoda je veĉ razvijeno pod nazivom "reŹimi rada" (eng. *mode of operations*) i one se sa posebnom paŹnjom posmatraju (sa strane sigurnosti) prilikom korišćenja blok cipher algoritama unutar kriptosistema.

DES (eng. *Data Encryption Standard*) i AES (eng. *Advanced Encryption Standard*) su primeri block-cipher dizajna koji su predstavljeni kao kriptografski standardi od strane Ameriĉke vlade. DES algoritam je kasnije povuĉen pojavom dosta jaĉeg AES algoritma, meĊutim poboljšana verzija DES algoritma pod imenom *triple-DES* ostaje jedan od popularnijih standarda koji se danas koristi u raznim aplikacijama. TakoĊe, jedan od popularnijih blok cipher algoritma je i IDEA (eng. *International Data Encryption Algorithm*) razvijen od strane Xuejia Lai-a profesora univerziteta Shanghai Jiao Tong i DŹejmsa Meseja (eng. *James Massey*) profesora tehniĉkog iniverziteta u Cirihi (*ETH Zurich*). IDEA je nastao 1991. godine sa idejom da zameni veĉ standardizovani DES algoritam. Jedan nivo enkripcione funkcije IDEA algoritma je prikazan na slici 3.

Mnogo drugih block-cipher standarda je razvijeno i objavljeno sa razliĉitim nivoima kvaliteta. Dosta takvih standarda se pokazalo kao nezadovoljavajuĉim sa sigurnosne strane zbog probijanja sigurnosti njihovih algoritama.



Slika 3 : Blok dijagram jednog nivoa unutar IDEA algoritma

U poreĊenju sa block-cipher algoritmima, stream-cipher algoritmi stvaraju proizvoljno dugaĉke nizove znakova koji se sastoje od kljuĉa i znakova informacija kombinovanih bit po bit. Kod stream-cipher metoda, izlazni niz znakova se stvara u zavisnosti od unutrašnjeg stanja koje se menja u toku rada samog algoritma. Promena stanja algoritma je kontrolisana od strane samog kljuĉa, a kod nekih verzija stream-cipher algoritma i od strane samih podataka. RC4 je primer dobro poznatog algoritma ove vrste.

3.2.2.2 Kriptografija javnog kljuĉa ^[5]

U kriptosistema koji se oslanjaju na kriptografiju simetriĉnog kljuĉa isti kljuĉ se koristi i u procesu šifrovanja i u procesu dešifrovanja podataka sa mogućnošću da se kljuĉevi menjaju od jedne do druge grupe podatka. Usko grlo u funkcionisanju algoritama simetriĉnog kljuĉa prestavlja rukovanje kljuĉevima (eng. *key management*), koje treba da omogući siguran prenos informacija. U idealnim okolnostima svaki par uĉesnika u komunikaciji mora da deli razliĉiti

ključ za svaki smer komunikacije, samim tim broj ključeva potrebnih za uspešan i siguran prenos podatka je jednak kvadratu broja korisnika unutar mreže za prenos. Veliki broj korisnika kao i još veći broj ključeva zahteva kompleksan sistem rukovanja ključevima koji bi osigurao tajnost svih ključeva i sigurnost podatka u samoj komunikaciji. Problem utvrđivanja tajnog ključa između dve strane u komunikaciji, koja nije osigurana, predstavlja značajnu prepreku za korisnike kriptografije.

Whitfield Diffie i Martin Hellman su u svom radu koji je objavljen 1976. godine, u već utvrđenu oblast kriptografije, uveli pojam “javnog ključa” (takođe poznat i pod terminom “asimetrični ključ” – eng. *asymmetric key*) pod kojim se podrazumevala upotreba dva matematički povezana ključa – javni ključ (eng. *public key*) i privatni ključ (eng. *private key*). Sistem javnog ključa se zasniva na činjenici da privatni ključ nije moguće matematički izvesti iz javnog ključa iako su oni međusobno povezani.

Unutar kriptosistema koji su zasnovani na kriptografiji javnog ključa javni ključ se može slobodno deliti i poznat je svim učesnicima u komunikaciji, dok se njegov upareni privatni ključ mora čuvati u tajnosti. Uz pomoć javnog ključa se vrši šifrovanje podatka dok se privatni ključ koristi za dešifrovanje istih. Pošto se predhodna uspostava i razmena ključeva dešava u mreži za prenos koja nije osigurana, Diffie i Hellman su uveli poseban protokol za razmenu ključeva koji nosi njihovo ime *Diffie-Hellman key exchange protocol*.

Jedan od najpopularnijih predstavnika cipher algoritama u kriptografiji javnog ključa jeste RSA algoritam (ime algoritma je skraćenica koja se dobija uzimanjem prvog slova prezimena profesora MIT instituta koji su doprineli stvaranju ovog algoritma: Dr. Ronald L. Rives, Dr. Adi Shamir, Dr. Leonard Adleman). RSA je jedan od prvih algoritama koji je bio pogodan za proces digitalnog potpisivanja kao i proces šifrovanja podatka unutar kriptografije javnog ključa. Pod pojmom digitalnog potpisa se podrazumeva potpis u obliku digitalnih sertifikata koji se lako stvaraju od strane korisnika ali se teško falsifikuju. Digitalni potpisi se mogu vezati za sadržaj koji se potpisuje i kao takvi se ne mogu prenositi sa jednog dokumenta na drugi, tj. sam prenos se može uočiti. U okviru sistema digitalnih potpisa koriste se dva algoritma: jedan za sam proces potpisivanja dokumenta gde se privatni ključ koristi za obradu podataka unutar dokumenta i drugi za verifikaciju gde se koristi upareni javni ključ radi provere ispravnosti potpisa. RSA i DSA (eng. *Digital Signature Algorithm*) su dva najpopularnija algoritma za digitalno potpisivanje dokumenata. Digitalni potpisi predstavljaju centralni deo funkcionisanja PKI struktura i sistema mrežne sigurnosti (SSL/TLS, VPN mreže, itd.).

3.2.2.3 Kriptoanaliza ^{[8][7]}

Cilj kriptoanalize je pronalaženje slabosti ili nesigurnosti u kriptografskim sistemima. Kriptoanaliza se može sporvesti od strane napadača u pokušaju preuzimanja kontrole nad informacionim sistemom, ali i od strane administratora u procesu pronalaženja potencijalnih slabosti sistema. Poslednjih godina kriptografski algoritmi i protokoli se temeljno proveravaju i testiraju pre nego što se za njih tvrdi da nude zadovoljavajući nivo sigurnosti sistema.

Tvrđnja da se svaki metod šifrovanja može probiti, tj. nasilno dešifrovati je pogrešna tvrdnja koja se može čuti u svetu kriptografije. U vezi sa ovim, naučnik Claude Shannon je dokazao da se tzv. *one-time pad cipher* algoritam ne može probiti ukoliko je ključ dovoljno dugačak (jednak ili duži od same poruke koja se šifruje), sastavljen od stohastičke raspodele znakova koji se nikada ne ponavljaju i kao takav čuva od dometa potencijalnog napadača. Većina cipher algoritama, osim *one-time pad* algoritma, se mogu probiti primenom dovoljne procesne snage, tj. tzv. brutalnom silom (eng. *brute force attack*), međutim vreme trajanja i trud primene brutalne sile su eksponencijalno zavisni od veličine ključa koji se koristi u procesu šifrovanja i neuporedivo veći od normalne upotrebe cipher algoritma. U ovakvim slučajevima efektivna sigurnost u prenosu podatka se može postići ukoliko se dokaže da trud koji je potrebno uložiti za probijanje nekog algoritma veći od mogućnosti samog napadača. Iz ovoga proizilazi činjenica da se za svaki cipher algoritam mora dokazati da ne postoji efektivna metoda (isključujući primenu brutalne sile koja traži dosta vremena) koja bi mogla da probije algoritam

šifrovanja podatka. Pošto do dan danas nijedan od pristunih algoritama nema dovoljno dokaza o neprobojnosti istih, *one-time pad cipher* algoritam je jedini teorijski algoritam koji se ne može probiti.

U svetu kriptografije postoje razne vrste napada na sigurnost kriptosistema. Podela ovih napada se bazira na informacijama o tome šta potencijalni napadač zna i koje su njegove sposobnosti.

- *Ciphertext-only attack* – napad koji se vrši nad nerazumljivim šifrovanim delom podatka. Kod ove vrste napada napadač ima pristup samo podacima koji su nerezumljivi tj. šifrovani. Moderni kriptosistemi su uglavnom efektivno imuni na ovu vrstu napada.
- *Known-plaintext attack* – napad nad poznatim podacima koji nisu šifrovani. Napadač ima pristup originalnim podacima (*plaintext*) kao i njihovim šifrovanim ekvivalentima (*ciphertext*).
- *Chosen-plaintext attack* – napad nad odabranim podacima koji nisu šifrovani. Napadač sam bira određeni deo informacija koje nisu šifrovane i na osnovu njih može naučiti njihov odgovarajući šifrovani ekvivalent. Jedan od primera ove vrste napada je tzv. *gardening* napad koju je koristila Britanska vojska u Drugom svetskom ratu.
- *Chosen-ciphertext attack* – napad nad odabranim šifrovanim podacima. Napadač sam bira određeni deo informacija koje su šifrovane i na osnovu njih pronalazi njihove ekvivalente u obliku nešifrovanih podatka. Ovaj napad je zapravo obrnuti proces od *chosen-plaintext attack* napada.

Veoma važno je napomenuti da u dosta slučajeva uspeh napadača zavisi i od grešaka koje se javljaju u dizajnu nekog od protokola koji se koriste u kriptosistemima.

U kriptooanalizi sistema koji koriste kriptografiju simetričnog ključa pretežno se traže napadi usmereni ka *block-cipher* i *stream-cipher* algoritmima. Na primer, običan napad primenom brutalne sile na DES algoritam zahteva jedan poznat nešifrovani podatak (*plaintext*) i 2^{55} operacija dekrpcije. Sa druge strane napad upotrebom linearne kriptooanalize (eng. *linear cryptanalysis*) na DES algoritam zahteva 2^{43} poznatih nešifrovanih podatka i približno 2^{43} DES operacija.

Kod kriptografije javnog ključa jačina kriptografskih algoritama je direktno srazmerna matematičkoj težini proračuna unutar algoritma. Najpoznatija matematička rešenja koja se koriste kod algoritama javnog ključa jesu faktORIZACIJA celih brojeva (RSA algoritam je usko vezan za problem faktORIZACIJE celih brojeva) i diskretne logaritamske funkcije. Za rešavanje ovih matematičkih problema većina algoritama unutar kriptografije javnog ključa koriste numeričke metode proračunavanja. Na primer, najpoznatiji numerički algoritmi za rešavanje diskrenih logaritama zasnovanim na eliptičnim krivama su dosta vremenski zahtevniji od algoritama za rešavanje problema faktORIZACIJE celih brojeva ako se uzme u obzir da su ulazni podaci algoritama (ključ, podaci, itd.) približno isti. Iz ovoga se može zaključiti da bi se postigao podjednak nivo otpornosti na napade, tehnike enkripcije koje se oslanjaju na faktORIZACIJU celih brojeva moraju koristiti duže ključeve od tehnika koje koriste algoritme eliptične krive. Zbog ove činjenice kriptosistemi javnog ključa koji su zasnovani na algoritmima eliptičnih kriva su popularni još od vremena njihovog nastanka, sredinom devedesetih godina.

Dok se čista kriptooanaliza zasniva na pronalaženju slabosti unutar samih algoritama, druge vrste napada na kriptoseisteme se oslanjaju na njihovu praktičnu primenu u raznim uređajima. Ove vrste napada nose naziv *side-channel attacks*. Ukoliko napadač ima pristup, npr. količini vremena koje je potrebno samom uređaju da šifrue određenu količinu podatka ili da prijavi grešku u unosu lozinke ili PIN broja, on je onda u mogućnosti da koristi tzv. “vremenske napade” (eng. *timing attack*) ne bi li probio algoritam koji je sam po sebi otporan na kriptooanalizu. Napadač je takođe u mogućnosti da proučava obrazac i dužinu poruka radi izdvajanja korisnih informacija. Ova vrsta napada je poznata pod nazivom “analiza saobraćaja”

(eng. *traffic analysis*) i može biti veoma korisna svesnom napadaču. Naravno napadi na osoblje koje ima veze sa kriptosistemima (socijalni inženjering – eng. *social engineering*) radi pridobijanja važnih informacija predstavlja najproduktivniji način neovlašćenom pristupu informacijama i probijanja kriptosistema.

3.2.3 Kriptografski protokoli

TLS (*Transport-Layer Security*) kao i njegov predhodnik SSL (*Secure Sockets Layer*) su kriptografski protokoli koji daju mogućnost sigurne komunikacije Internet mrežom aplikacijama kao što su internet pretraživači, aplikacije za slanje i primanje elektronske pošte, aplikacije za slanje brzih i kratkih poruka, itd. Postoje male razlike između SSL i TLS, međutim funkcionalnost ova dva protokola je uglavnom ista.

TLS protokol, koji je razvijen od strane IETF (*Internet Engineering Task Force*) i predstavljen u obliku RFC 2246 dokumenta, pruža mogućnost aplikacijama da komuniciraju putem globalne Internet mreže na siguran način, pri tome sprečavajući moguće prisluškivanje, menjanje ili falsifikovanje podatka od strane trećih lica. TLS takođe pruža mogućnost dokazivanja autentičnosti (eng. *authentication*) i čuvanja privatnosti na globalnoj mreži uz pomoć kriptografije. U najčešćim slučajevima dokazivanje autentičnosti se vrši samo za servere dok se autentičnost klijenta ne proverava. Ovim se klijentu (bilo da je to čovek ili računar) daje do znanja sa kime tačno komunicira. Sledeći nivo sigurnosti – u kome obe strane komunikacije (klijent i server) znaju sa kime komuniciraju se zove uzajamno dokazivanje autentičnosti (eng. *mutual authentication*). Uzajamno dokazivanje autentičnosti zahteva implementaciju PKI sistema, uglavnom na klijentskoj strani komunikacije, ili nekih drugih sistema provere autentičnosti (kao što su TLS-PSK i TLS-SRP).

U TLS protokolu postoje tri faze rada:

- *Peer negotiation for algorithm support* - pregovaranje komunikacionih strana o odgovarajućem kriptografskom algoritmu
- *Key exchange and authentication* - razmena ključeva i dokaz autentičnosti
- *Symmetric cipher encryption and message authentication* - šifrovanje podatka uz pomoć simetričnih cipher algoritama

U toku prve faze, klijent i server pregovaraju o nizu cipher algoritama (eng. *cipher suites*) i odlučuju koji od njih će biti korišćen u toku komunikacije. Takođe, u toku pregovaranja se odlučuje i o načinu razmene ključeva i kodova autentičnosti poruka (eng. *Message Authentication Codes* - MAC). Algoritmi koji se koriste za razmenu ključeva i algoritmi dokazivanja autentičnosti su uglavnom algoritmi koji pripadaju kriptografiji javnog ključa. Kodovi autentičnosti poruka se stvaraju uz pomoć kriptografskih “heš” (eng. *hash*) funkcija.

Primer mogućih algoritama:

- Za razmenu ključeva: RSA, Diffie-Hellman, DSA, SRP, PSK
- Algoritmi kriptografije simetričnog ključa: RC4, Triple DES, AES ili Camellia.
- Za stvaranje kodova autentičnosti poruka (*hash* funkcije): HMAC-MD5 ili HMAC-SHA

3.2.3.1 Princip rada kriptografskih protokola ^{[11][12]}

Pregovaranje i uspostava sigurne veze između TLS klijenta i servera započinje procedurom koja se zove “rukovanje” (eng. *handshake*). Za vreme rukovanja, klijent i server se dogovaraju o raznim parametrima koji će se koristiti za uspostavljanje sigurne veze. Generalno procedura izgleda ovako:

- *Handshake* počinje onog trenutka kada se klijent poveže sa serverom koji podržava TLS protokol tražeći od njega uspostavu sigurne veze. U toku ove operacije, klijent takođe šalje serveru listu cipher algoritama koje podržava kao i *hash* funkcije.

- Server nakon primanja liste cipher algoritama i *hash* funkcija iz nje bira najači i najsigurniji algoritam koji on takođe podržava i klijenta obaveštava o tome.
- Server šalje klijentu svoju identifikaciju u vidu digitalnog sertifikata. Sertifikat obično sadrži ime servera, ime organizacije koja je izdala i potpisala dati sertifikat (eng. *certificate authority* - CA) i javni ključ servera. Klijent može dodatno kontaktirati organizaciju koja je izdala datom serveru digitalni sertifikat – CA i uz pomoć nje potvrditi da je sertifikat autentičan pre nastavka komunikacije sa serverom.
- Da bi se uspešno stvorili ključevi sesije koji se koriste za sigurnu komunikaciju, klijent na osnovu javnog ključa servera šifrue nasumice izabrani broj i takav šalje serveru. Server je jedini koji dati broj može dešifrovati (uz pomoć svog privatnog ključa), ovim se postiže to da nasumice izabrani broj bude poznat samo klijentu i serveru, a sakriven od strane trećeg lica
- Na osnovu nasumice izabranog broja, obe strane komunikacije stvaraju odgovarajuće ključeve za datu sesiju koje koriste za proces šifrovanja i dešifrovanja podatka.

Ovim se ujedno i završava proces rukovanja i započinje sigurna komunikacija sve do njenog prekida. Ukoliko se bilo koji od gore navedenih koraka ne obavi uspešno, TLS *handshake* se označava kao neuspešan i sigurna komunikacija se ne ostvaruje između klijenta i servera.

3.2.3.2 Faze uspostave sigurne veze kod kriptografskih protokola – protokol rukovanja

Jedinica za prenos podatka unutar TLS protokola se zove *record* i ona enkapsulira sve podatke koji se prenose putem ovog protokola. Svaki *record* se može komprimovati (kompresovati), popunjavati, proširivati MAC kodom i šifrovati u zavisnosti od trenutnog stanja ostvarene veze. Svaki *record* u sebi sadrži polje koje označava tip sadržaja (eng. *content type*), polje dužine samog sadržaja i TLS verziju.

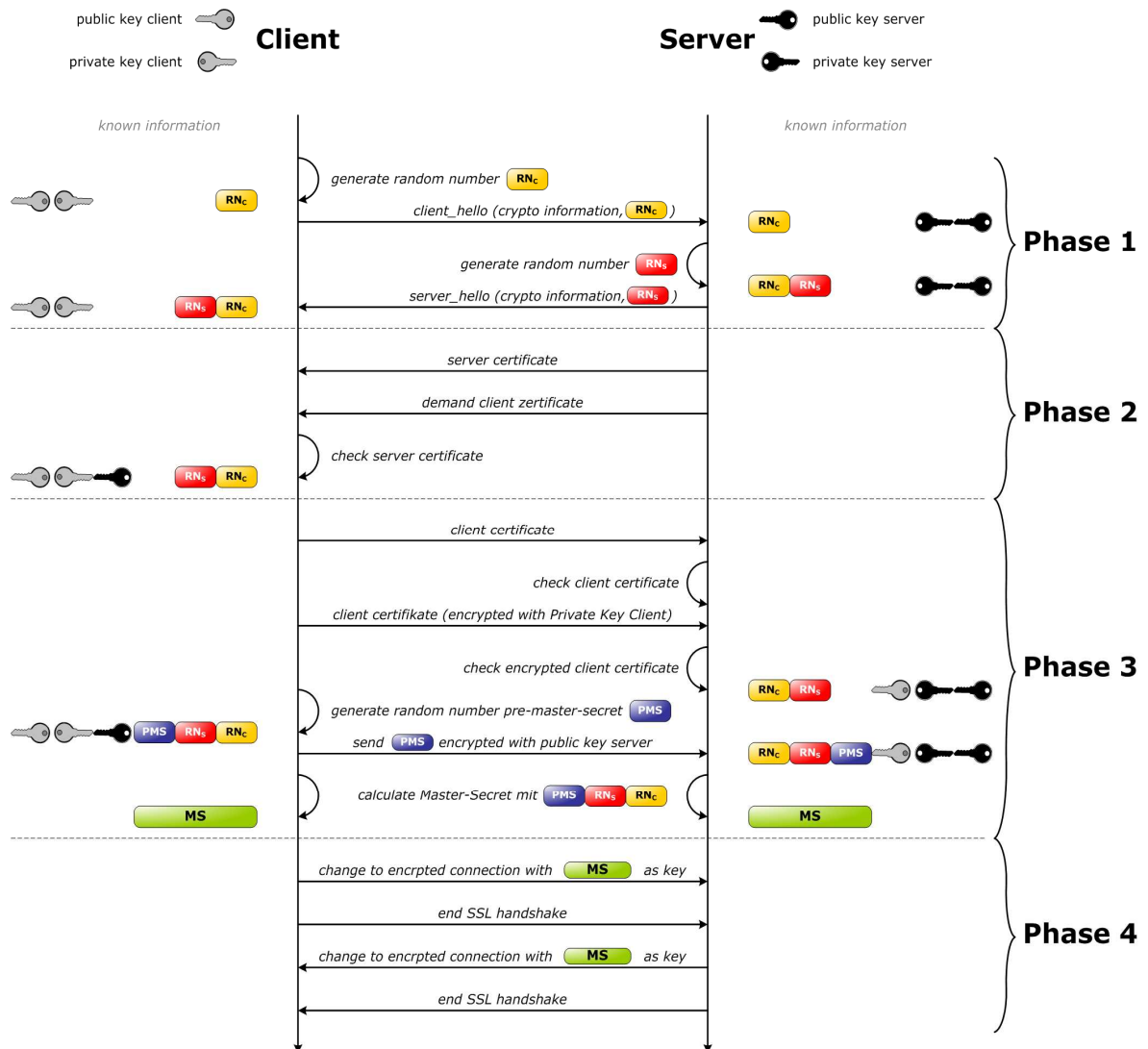
Prilikom uspostave TLS veze, *record* u sebi enkapsulira jedan drugi protokol, a to je protokol rukovanja (*handshake protocol*) čija je oznaka tipa sadržaja (*content type*) 22.

Primer jednog TLS rukovanja (slika 4.):

- Klijent koji započinje komunikaciju šalje poruku **ClientHello** zajedno sa najvišom verzijom TLS protokola koju podržava, nasumice odabrani broj, listu podržanih cipher nizova i metoda komprimovanja koje podržava.
- Server odgovara sa **ServerHello** porukom koja u sebi sadrži izabranu verziju TLS protokola, nasumice odabrani broj, izabrani cipher algoritam i metodu komprimovanja iz liste koju je klijent ponudio.
- Server šalje svoj **sertifikat** (u zavisnosti od odabranog cipher algoritma, ovaj korak se može preskočiti). Trenutno najzastupljeniji format sertifikata je X.509, ali se takođe može koristiti i format *OpenPGP* koji je trenutno u razvoju.
- Server može zahtevati klijentski sertifikat, radi ostvarenja uzajamno poverljive veze, porukom **CertificateRequest**.
- Server šalje **ServerHelloDone** poruku kojom označava kraj procesa rukovanja sa njegove strane.
- Klijent odgovara sa **ClientKeyExchange** porukom u okviru koje se nalaze *PreMasterSecret*, javni ključ ili ništa od navedenog (u zavisnosti od odabranog cipher algoritma).
- Klijent i server stvaraju zajednički tajni kod – glavna tajna (eng. *master secret*) na osnovu nasumice izabranih brojeva i *PreMasterSecret* koda. Svi ostali podaci vezani za ključeve enkripcije nastaju iz koda glavne tajne upotrebom pažljivo implementirane *pseudorandom* funkcije.
- Klijent šalje poruku **ChangeCipherSpec** sa kojom obaveštava servera da je sva komunikacija od ove tačke pa nadalje šifrovana na osnovu prethodno dogovorenih

parametara. **ChangeCipherSpec** je sam po sebi protokol na nivou *record-a* sa oznakom tipa sadržaja 20.

- Na kraju, klijent šalje šifrovanu poruku **Finished** u kojoj navodi kodove *hash* i MAC funkcija prethodnih poruka rukovanja.
- Server pokušava da dešifruje klijentsku **Finished** poruku kao i da proveriti *hash* i MAC kodove. Ukoliko se naiđe na bilo kakav problem u procesu dešifrovanja i provere, TLS rukovanje se označava kao neuspešno i veza između servera i klijenta se prekida.
- Ukoliko se prethodni korak obavi bez problema, server šalje klijentu poruku **ChangeCipherSpec** i svoju šifrovanu **Finished** poruku. Ovog puta klijent vrši dešifrovanje i proveru date poruke.
- U ovom trenutku rukovanje je završeno uspešno i oznaka tipa sadržaja se postavlja na 23 (*Application protocol*).



Slika 4 : SSL/TLS rukovanje (*handshake*) sa obostranom proverom autentičnosti

3.2.3.3 Sigurnost kriptografskih protokola ^{[11] [12]}

SSL/TLS protokol sadrži veliki broj sigurnosnih mera:

- Klijenti su u mogućnosti da koriste javne ključeve izdavaoca sertifikata (*certificate authority* - CA) radi provere ispravnosti sertifikata servera. Ukoliko se digitalnom potpisu, unutar sertifikata, dokaže autentičnost klijent prihvata sertifikat servera kao validan sertifikat koji je izdat od strane CA.
- Klijent proverava da li se dati CA nalazi u njegovoj internoj listi verodostojnih (eng. *trusted*) CA.
- Klijent proverava vreme trajanja sertifikata. Provera autentičnosti sertifikata se prekida ukoliko se trenutni datum ne nalazi unutar perioda trajanja sertifikata i takav sertifikat se ne prihvata.
- Da bi se ostvarila efikasna zaštita od napada na privatnost primenom tehnike "čovek u sredini" (eng. *Man in the middle attack*), klijent proverava stvarno DNS (eng. *domain name server*) ime servera sa DNS imenom koje je upisano u samom sertifikatu servera (ova dva imena se moraju poklapati). Ova provera se obično vrši na nivou aplikacije i nije definisana u okviru TLS specifikacije.
- Zaštita od promene verzije protokola (eng. *downgrade of the protocol*) na prethodnu (manje sigurnu) verziju sa slabijim cipher algoritmima.
- Sortiranje poruka aplikativnog nivoa u sekvencijalne nizove i korišćenje tih nizova za primenu MAC kodova.
- Korišćenje tzv. odabiraka poruka (eng. *message digest*) u kombinaciji sa ključem (tako da samo vlasnik ključa može proveravati MAC). Ovo je specificirano u RFC 2104 dokumentu (važi samo za TLS).
- Poruka kojom se završava rukovanje (**Finished** poruka) šalje *hash* kodove svih prethodno razmenjenih poruka u procesu rukovanja.
- Pseudorandom funkcija razdvaja ulazne podatke na dva dela i svaki od delova obrađuje sa različitim algoritmom (MD5 i SHA-1), rezultati ova dva algoritma se kasnije spajaju logičkom XOR funkcijom. Ovim se postiže zaštita ukoliko se jedan od ova dva algoritama probije tj. poseduje sigurnosni propust (važi samo za TLS).
- Jedno od unapređenja SSL protokola verzije 3 u osnovu na verziju 2 je dodavanje cipher algoritama zasnovanih na SHA-1 algoritmu i dodavanje podrške za proveru autentičnosti sertifikata. Dodatna unapređenja su bolji tok protokola rukovanja kao i povećana otpornost na *Man in the middle* napade.

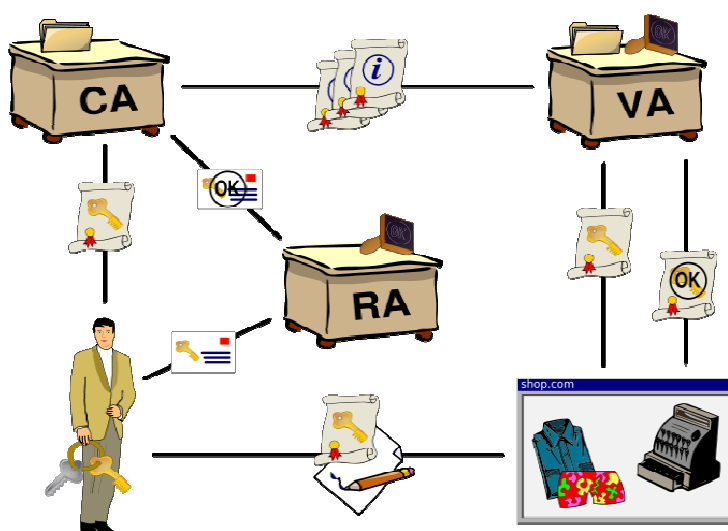
Nekoliko nedostataka SSL protokola verzije 2:

- Korišćenje istih ključeva za proveru autentičnosti i šifrovanje podataka.
- MAC kodovi su nepotrebno oslabljeni u tzv. "izvoznom režimu rada" (eng. *export mode*) usled ograničenja unutar izvozne regulative Sjedinjenih Američkih Država (simetrični ključevi su mogli da budu dugački maksimalno 40 bita u aplikacijama kao što su Internet Explorer i Netscape).
- SSL v2 poseduje slabu konstrukciju MAC kodova i oslanja se isključivo na MD5 *hash* funkcije.
- SSL v2 ne poseduje zaštitu procesa TLS/SSL rukovanja, što znači da se *Man in the middle* napadi koji vrše obaranje verzije protokola (*downgrading of protocol*) ne mogu otkriti na vreme.
- SSL v2 koristi TCP *close* direktivu za označavanje kraja podatka. Ovim se ostavlja mogućnost za tzv. napade skraćivanja (eng. *truncation attack*): napadač uspešno falsifikuje **TCP FIN** poruku čime obaveštava primaoca podatka o kraju prijema koji se zapravo nije desio (SSL v3 rešava ovaj problem tako što šalje eksplicitno poruku o kraju prenosa podatka).

3.2.4 Sistem za izdavanje digitalnih ključeva i identifikaciju korisnika ^[6]

U jednom kriptografskom okruženju sistem za izdavanje i identifikaciju korisnika - PKI ima ulogu povezivanja javnih ključeva sa odgovarajućim identitetom korisnika uz pomoć izdavaoca sertifikata – CA. Identitet korisnika mora biti jedinstven za svaki CA. Povezivanje javnog ključa i korisnika se ostvaruje kroz proces registracije i izdavanja koji, u zavisnosti od nivoa sigurnosti, može biti obavljen od strane softvera unutar CA ili od strane čoveka. PKI deo sistema koji osigurava ovakvo povezivanje se zove rukovalac registracije – RA (eng. *Registration Authority*). Za svakog korisnika, identitet, javni ključ, njihova povezanost, uslovi važenja i ostali atributi su zaštićeni od falsifikovanja u obliku sertifikata javnih ključeva koje izdaje CA.

Skraćenica PKI se ponekad pogrešno koristi za označavanje algoritama javnog ključa (eng. *public key algorithms*) koji ne zahtevaju upotrebu CA.



Slika 5 : Dijagram PKI sistema

PKI sistem pruža mogućnost provere autentičnosti računara koji učestvuju u komunikaciji bez prethodne razmene informacija kao i mogućnost korišćenja javnih ključeva koji se nalaze unutar sertifikata za šifrovanje podataka. Generalno, PKI se sastoji od klijentskog softvera, serverskog softvera, hardvera (npr. pametne kartice – eng. *smart cards*), ugovora o sigurnom radu i operativnih procedura. Sertifikat javnog ključa entiteta koji je potpisuje ostale sertifikate (CA) se takođe može koristiti od strane trećeg lica, radi provere digitalnog potpisa poruke koja se prenosi i koji je nastao na osnovu privatnog ključa CA. Zapravo, PKI pruža stranama koje učestvuju u komunikaciji visok nivo poverljivosti i integriteta poruka koje se prenose kao i dokaz autentičnosti korisnika bez potrebe za prethodnom razmenom bilo kakvih tajnih podataka. Ispravnost PKI sistema je ograničena praktičnim problemima kao što su neizvesno povlačenje sertifikata od strane CA (eng. *certificate revocation*), uslovi postavljeni od strane CA za izdavanje sertifikata, promenljivost regulativa pravosudnih zakona i poverenje.

3.2.5 Virtuelna privatna mreža ^[13]

Virtuelna privatna mreža – VPN je komunikaciona mreža koja se nalazi unutar druge fizičke mreže za prenos podataka. Jedna od najčešćih primena VPN mreža jeste sigurna komunikacija kroz nesigurnu mrežu kao što je Internet, ali isto tako, VPN ne mora da se oslanja na eksplicitne sigurnosne mere kao što su međusobna provera autentičnosti ili šifrovanje

informacija. VPN se npr. može koristiti za međusobno razdvajanje grupa korisnika koje se nalaze na fizičkoj mreži sa jakim sigurnosnim karakteristikama.

3.2.5.1 Kategorizacija virtuelnih privatnih mreža na osnovu nivoa sigurnosti ^[14]

U pogledu sigurnosti, VPN mreža mora imati sopstvene sigurnosne mehanizme ili fizička mreža na koju se VPN oslanja mora biti osigurana. Pristup samoj VPN mreži mora biti ostvaren kroz mehanizme provere autentičnosti korisnika. Neki od sigurnosnih modela VPN mreža su:

- **Provera autentičnosti pre ostvarenja same VPN veze** – poznati verodostojni korisnik, ponekad samo uz korišćenje sigurnosnih uređaja (npr. *smart card*), može dobiti odgovarajuće sigurnosne privilegije za pristup nekom informacionom sistemu unutar VPN mreže. Serveri se ponekad takođe proveravaju pre nego što postanu deo VPN mreže.
- **Verodostojne VPN mreže** (eng. *Trusted Delivery Networks*) – ne koriste kriptografiju unutar samog VPN sistema, već se oslanjaju na sigurnost same fizičke mreže za prenos podataka na koju se oslanjaju.
- **Sigurnosni mehanizmi unutar VPN mreže** – Sigurne VPN mreže koriste kriptografske VPN protokole (eng. *cryptographic tunneling protocols*) radi postizanja poverljivosti (sprečavanja prisluškivanja mreže), provere autentičnosti korisnika (sprečavanje maskiranja identiteta – eng. *identity spoofing*) i integriteta poruka (sprečavanje neovlašćenog menjanja poruka). Ukoliko se pažljivo izaberu, implementiraju i koriste, ovi mehanizmi mogu da pruže zavidan nivo sigurnosti u VPN mreži koja se prostire kroz fizički neosiguranu mrežu. Pod sigurnosnim VPN protokolima se podrazumeva:
 - o IPsec (*IPSecurity*) – najčešće korišćen kroz mreže tipa IPv4 i kao obavezan deo IPv6.
 - o SSL/TLS koji se koristi za siguran prenos unutar čitave VPN mreže (kao što je slučaj kod OpenVPN projekta) ili za osiguravanje *web proxy* aplikacije.
 - o OpenVPN – otvoreni VPN standard. Predstavlja varijaciju VPN mreže bazirane na SSL/TLS protokolu sa mogućnošću rada i preko UDP protokola. Projekat poseduje klijentski i serverski softver za većinu operativnih sistema.
 - o L2TPv3 (*Layer 2 Tunneling Protocol version 3*) – novo izdanje VPN sigurnosnog protokola.
 - o VPN Quarantine – Klijentski PC računar na jednom kraju VPN mreže može biti pretnja sigurnosti mreže, stoga se on postavlja u tzv. “VPN karantin” tj. izoluje se od ostatka mreže. Odluka o potpunom odsecanju tog korisnika iz mreže ili njegovom povratku zavisi od administratora mreže.
 - o MPVPN (*Multi Path Virtual Private Network*) – protokol razvijen od strane *Ragula Systems Development Company*.
- **Sigurnost i pokretljivost (mobilnost)** – Mobilne VPN mreže su dizajnirane za potrebe mobilnih i bežičnih korisnika. One u sebi sadrže standardne tehnike provere autentičnosti i šifrovanja podatka radi ostvarenja sigurnog prenosa informacija. Zbog mobilnosti, mobilne VPN mreže su dizajnirane tako da zadovolje potrebe korisnika koji su u stalnom pokretu i koji zahtevaju pristup osetljivim podacima kroz različite tipove bežičnih veza. Jedna od osnovnih karakteristika ovakvih mreža jeste održavanje VPN veze uprkos prekidima koji mogu nastati u bežičnim komunikacijama (nestanak signala, prelazak sa jednog predajnika na drugi, itd.) dokle god je to moguće. Ovim se postiže tzv. *roaming*, tj. mogućnost kretanja korisnika iz jedne zone pokrivenosti signala u drugu bez gubitka ostvarene VPN veze.

4. REALIZACIJA

Klijentska skener aplikacija mLunch sistema je aplikacija koja ima zadatak da omogući brzu identifikaciju osobe (korisnika) koja je naručila obrok u restoranu (uz pomoć skeniranja dvodimenzionalnog bar koda), kao i pregled osnovnih informacija vezanih za naručene obroke za zadati dan u datom restoranu. Informacije se prenose u obliku XML datoteka koje se stvaraju kao odgovori na upite klijenta prema serveru i poslužuju na strani servera u sklopu Apache web usluge (servisa).

Pošto se na osnovu arhitekture mLunch sistema nameće upotreba Internet mreže za prenos informacija od servera do klijentskog skener uređaja, neophodno je implementirati sigurnosne mehanizme na obe strane komunikacije (klijent i server) radi postizanja zadovoljavajućeg nivoa sigurnosti rada informacionog sistema. U prethodnom poglavlju su razmatrani sigurnosni mehanizmi uz pomoć kojih se može ostvariti siguran prenos podatka, neki od tih mehanizama je neophodno implementirati unutar mLunch sistema. Uspostavljanje sigurnosnih mehanizama zahteva implementaciju određenih funkcionalnosti i na strani servera i na strani klijenta.

U ovom radu su razmatrana dva načina uspostavljanja sigurne komunikacije unutar mLunch sistema:

1. Implementacija VPN mreže unutar Internet mreže sa odgovarajućim sigurnosnim mehanizmima koje pruža ovaj vid povezivanja. Celokupna komunikacija između servera i klijenta se odvija unutar VPN mreže.
2. Zbog činjenice da klijent i server koriste HTTP protokol za razmenu informacija, implementacija SSH/TLS protokola zajedno sa PKI sistemom za razmenu sertifikata, digitalnih potpisa i javnih ključeva na klijentskoj i serverskoj strani, stvara uslove za primenu HTTPS protokola. HTTPS je zapravo HTTP protokol sadržan unutar SSL/TLS transportnog protokola za siguran prenos podatka.

Za potrebe realizacije VPN mreže, korišćeno je besplatno *open source* softversko rešenje projekta pod imenom OpenVPN verzija 2.0 (Internet prezentacija: www.openvpn.org), dok su za potrebe implementacije PKI sistema i SSL/TLS protokola korišćenje alatke i moduli prilagođeni programskom jeziku C# (C - Sharp) dostupni unutar Compact .NET 2.0 Framework okruženja za razvoj mobilnih aplikacija unutar Windows Mobile 5 SDK paketa.

4.1 Realizacija virtuelne privatne mreže – OpenVPN ^{[15] [16]}

OpenVPN je softversko rešenje otvorenog koda (eng. *open source*) uz pomoć kojeg se uspešno realizuje VPN mreža zasnovana na SSL protokolu. OpenVPN je u mogućnosti da se lako prilagodi velikom broju fizičkih arhitektura računarskih mreža za prenos podatka, pružajući podršku za daljinsko upravljanje i nadzor mrežom, osiguravanje bežičnih WiFi mreža, kao i kontrolu i rasterećenje toka saobraćaja (eng. *load balancing*).

Za razliku od mnogih VPN rešenja koja su dosta složena za implementaciju i nadgledanje OpenVPN, zahvaljujući svom prostom i efikasnom dizajnu, nudi relativno laku implementaciju, kontrolu i nadzor mreže bez potrebe za visokim stepenom obučenosti administratora VPN mreže. Sigurnosni model OpenVPN sistema je zasnovan na implementaciji SSL/TLS protokola na drugom ili trećem niovu ISO-OSI mrežnog modela, sa dodatkom provere autentičnosti korisnika primenom PKI sistema (digitalni sertifikati, pametne kartice...).

U okviru OpenVPN instalacionog paketa sadržan je softver zadužen za uspešnu realizaciju VPN servera na Linux ili Windows operativnom sistemu za PC računare. Serverski deo OpenVPN paketa se instalira na strani mLunch servera kao zaseban server ili integrisan u istom računaru zajedno sa Apache i MySql serverom. Klijentski softver OpenVPN paketa je takođe prilagođen PC platformi i gotovo rešenje postoji za Linux i Windows operativne sisteme. Klijentska aplikacija za mobilni skener uređaj razvijana je za Windows Mobile 5 operativni sistem, za koji još uvek ne postoji zvanična OpenVPN aplikacija za pristup i komunikaciju sa OpenVPN serverom. Ovaj problem se rešava upotrebom OpenVPN klijenta za Pocket PC platformu (autor Krzysztof Sobolewski) koji je razvijen nezavisno od OpenVPN projekta na osnovu izvornog koda koji je dostupan na osnovu OpenVPN GNU licence i prilagođen Windows Mobile platformi (<http://ovpnppc.ziggurat29.com/ovpnppc-main.htm>).

Nakon uspešne instalacije OpenVPN servera i klijenta, na datim uređajima se stvara tzv. virtuelni mrežni uređaj (eng. *network interface*) koji predstavlja apstrakciju VPN mreže. Pre početka rada mreže neophodna su detaljna podešavanja OpenVPN softvera radi postizanja što efikasnijeg i sigurnijeg rada iste. OpenVPN poseduje veliki broj pogodnosti za stvaranje raznovrsnih konfiguracija VPN mreža, u ovom radu su opisani koraci i konfiguracije dovoljni za uspešnu realizaciju mLunch sistema kroz VPN mrežu.

4.1.1 Tehnike upravljanja tokom saobraćaja unutar virtuelne privatne mreže

VPN mreža realizovana uz pomoć OpenVPN softvera pruža dve mogućnosti upravljanjem tokom podatka:

- Premošćenje saobraćaja (eng. *bridging*) – kod koje se VPN server ponaša kao most koji povezuje klijente iz različitih delova VPN mreže, slično mrežnim povezivanjem poznatijim pod imenom *bridge* (most).
- Preusmeravanje saobraćaja (eng. *routing*) – kod koje VPN server preuzima ulogu preusmeravanja saobraćaja iz jednog dela mreže (eng. *subnet*) na drugi, slično ruter mrežnim uređajima. VPN server postaje kapija (eng. *gateway*) klijentima VPN mreže.

Tehnika preusmeravanja saobraćaja se pokazala kao efikasnija i lakša za korišćenje i podešavanje u odnosu na tehniku premošćavanja. Preusmeravanje takođe pruža mogućnost za selektivnu kontrolu pristupa korisnika mreži, što je posebno značajno za mLunch sistem.

Razlozi upotrebe tehnike premošćenja saobraćaja:

- Ukoliko VPN mreža mora da omogućiti prenos protokola koji nisu zasnovani na IP protokolima (kao što je npr. IPX).
- Korišćenje mrežnih aplikacija koje se oslanjaju na emisiju paketa (eng. *network broadcast*).
- Pristup deljenju datoteka putem Windows operativnog sistema (eng. *file sharing*) bez potrebe za dodatnim Samba ili WINS serverom.

Pošto se aplikacije unutar mLunch sistema oslanjaju na TCP/IP protokol bez mogućnosti i potrebe za *broadcast* funkcijom, upotreba tehnike premošćenja saobraćaja nije neophodna.

4.1.2 Stvaranje digitalnih sertifikata potrebnih unutar OpenVPN mreže

Prvi korak u procesu konfiguracije OpenVPN mreže jeste uspostava PKI sistema. PKI se sastoji od:

- Posebnog sertifikata (poznatog i kao javni ključ) i privatnog ključa za OpenVPN servera i njegove klijente.
- Sertifikata izdavaoca sertifikata (*Certificate Authority* - CA) koji se koristi za potpisivanje i izdavanje sertifikata servera i klijenata.

OpenVPN podržava obostranu proveru autentičnosti, što znači da OpenVPN klijenti proveravaju autentičnost OpenVPN servera i isto tako server proverava autentičnost klijenata pre nego što se uspostavi uzajamno poverenje i otpočne sa sigurnim prenosom podataka.

Provera autentičnosti se vrši tako što klijent i server uzajamno proveravaju da li su dati sertifikati digitalno potpisani od strane CA nakon čega se dodatno proveravaju informacije koje su sadržane u sertifikatima (kao što su: *common name*, tip sertifikata – server ili klijent, itd.).

Upotrebom PKI sistema, OpenVPN poseduje sledeće pogodnosti:

- Server je u obavezi da pamti samo svoj sertifikat/ključ i ne mora da zna sertifikate svih mogućih klijenata koji se mogu povezati sa njim.
- Server će prihvatiti samo one klijente čiji su sertifikati potpisani od strane CA. Server je u stanju da sprovodi proveru sertifikata bez potrebe za pristupom privatnom ključu CA entiteta (najosetljiviji podatak u čitavom PKI sistemu). Usled toga čuvanje privatnog ključa CA entiteta se može odvijati na drugom računaru koji, radi veće sigurnosti, može biti izolovan od svih računarskih mreža.
- Ukoliko je privatni ključ nekog od klijenata na bilo koji način kompromitovan (ključ je ukraden, nije više tajen), on se proglašava nevažećim i dodaje u CLR listu (*Certificate Revocation List*). CRL lista pruža mogućnost za efikasno odbacivanje nevažećih sertifikata bez potrebe za ponovnom instalacijom čitavog PKI sistema.
- Server može da uvede dodatne provere klijenata zasnovane na poljima unutar sertifikata, kao što je polje *common name* (ime klijenta).

4.1.2.1 Stvaranje sertifikata i privatnog ključa entiteta koje izdaje sertifikate

OpenVPN sadrži nekoliko skripti sadržanih unutar izvršnih datoteka tipa *bat* (eng. *batch file*). Ove skripte su zadužene za brzo i jednostavno stvaranje sertifikata i ključeva neophodnih za PKI menadžment uz pomoć OpenSSL izvršnih datoteka koje su uključene u OpenVPN projekat. Pre početka stvaranja bilo kog sertifikata potrebno je inicijalizovati konfiguraciju PKI okruženja (podešavanje *vars.bat* i *openssl.cnf* datoteke) pozivanjem sledeće skripte iz DOS komandnog prozora:

```
init-config
```

Nakom inicijalizacije konfiguracije, u novonastaloj datoteci *vars.bat*, neophodno je podesiti sledeće promenjive: *KEY_COUNTRY*, *KEY_PROVINCE*, *KEY_CITY*, *KEY_ORG* i *KEY_EMAIL*.

Sledeći korak je inicijalizacija PKI sistema uz pomoć sledećih skripti:

```
vars
clean-all
build-ca
```

Pozivanjem skripte *build-ca.bat* pokreće se proces stvaranja sertifikata i privatnog ključa CA entiteta uz pomoć interaktivne OpenSSL direktive:

```
C:\Program Files\OpenVPN\easy-rsa>build-ca.bat
Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to 'ca.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) [RS]:
State or Province Name (full name) [NA]:
Locality Name (eg, city) [Novi Sad]:
Organization Name (eg, company) [FTN]:
Organizational Unit Name (eg, section) [KRT]:
Common Name (eg, your name or your server's hostname) []: KRT-CA
Email Address [ca-admin@uns.ns.ac.yu ]:
```

Prilikom popunjavanja informacija potrebnih za stvaranje sertifikata vrednosti koje se nalaze u uglastim zagradama su one vrednosti koje su prethodno popunjene u *vars.bat* datoteci i ne moraju se eksplicitno popunjavati. Jedini podatak koji se mora popuniti jeste *Common Name* i on predstavlja ime entiteta (CA, server ili klijent) kome se sertifikat izdaje. Ukoliko se mimo ugrađenih PKI provera, unutar klijenata implementira provera imena domena na kome se server nalazi *Common Name* podatak je obavezan i može imati ključnu ulogu u sprečavanju *Man In The Middle* napada na sigurnost sistema.

4.1.2.2 Stvaranje sertifikata i privatnog ključa za servera

Stvaranje digitalnog sertifikata i privatnog ključa za OpenVPN servera se izvršava pozivom sledeće skripte:

```
build-key-server mlunch
```

Kao i kod stvaranja CA sertifikata, većina parametara se može izostaviti pošto su navedeni u datoteci *vars.bat* osim parametra *Common Name* kod koga se mora upisati ime servera, npr “mlunch”. Da bi dati sertifikat bio potpisan od strane CA, moraju se potvrdno odgovoriti sledeća pitanja koja su ponudjena na kraju popunjavanja parametara:

```
Sign the certificate? [y/n] y
1 out of 1 certificate requests certified, commit? [y/n] y
```

4.1.2.3 Stvaranje sertifikata i privatnog ključa za klijente

Stvaranje sertifikata i privatnih ključeva za svakog klijenta ponaosob je slično kao i kod stvaranja sertifikata za servera. Primer za tri klijenta:

```
build-key klijent1
build-key klijent2
build-key klijent3
```

Klijentski privatni ključ se dodatno može zaštititi lozinkom gde se umesto *build-key* skripte koristi skripta *build-key-pass*.

Posebna pažnja se treba posvetiti *Common Name* poljima prilikom stvaranja klijentskih sertifikata pošto se klijenti međusobno razlikuju upravo po ovom imenu i ono mora biti jedinstveno za svakog klijenta.

4.1.2.4 Inicijalizacija Diffie Hellman parametara

Diffie Hellman parametri se moraju inicijalizovati na strani OpenVPN servera:

```
build-dh
```

4.1.2.5 Ključne datoteke

Stvoreni sertifikati i privatni ključevi koji su nastali upotrebom skripti koje su opisane u prethodnim koracima se nalaze u poddirektorijumu *keys*. U sledećoj tabeli su data objašnjenja i značenja svake stvorene datoteke u ovom procesu:

Tabela 1 : Ključne datoteke za funkcionisanje PKI unutar OpenVPN sistema

Ime datoteke	Potrebna od strane	Svrha datoteke	Tajno čuvanje
ca.crt	servera i svih klijenata	CA sertifikat	NE
ca.key	računara zaduženog za potpisivanje sertifikata	CA ključ	DA
dh{n}.pem	servera samo	DH parametri	NE
mlunch.crt	servera samo	sertifikat servera	NE
mlunch.key	servera samo	ključ servera	DA
klijent1.cer	klijent1 samo	sertifikat klijent1	NE
klijent1.key	klijent1 samo	ključ klijent1	DA
klijent2.cer	klijent2 samo	sertifikat klijent2	NE
klijent2.key	klijent2 samo	ključ klijent2	DA
klijent3.cer	klijent3 samo	sertifikat klijent3	NE
klijent3.key	klijent3 samo	ključ klijent3	DA

Datoteke koje se tajno čuvaju mogu se dodatno hardverski zaštititi od neovlašćenog korišćenja upotrebom smart kartica ili na neki drugi način. Da bi PKI sistem bio pouzdan, neophodno je tajno čuvanje svih datoteka koje u sebi sadrže ključ, a najvažniji ključ za ceo PKI sistem jeste ključ CA entiteta, jer se uz pomoć njega potpisuju i izdaju svi ostali sertifikati (serverski i klijentski).

4.1.3 Implementiranje konfiguracionih datoteka servera i klijenata

OpenVPN aplikacije servera i klijenata se mogu pokretati iz komandne linije, ali se isto tako mogu pokretati i kao pozadinski servisi unutar Windows XP operativnog sistema. Podešavanja i način rada VPN mreže se u potpunosti može konfigurisati uz pomoć tzv. konfiguracionih datoteka, nezavisno od načina pokretanja aplikacije. Ove datoteke se posebno stvaraju za servera (koji ujedno nameće pravila i način rada VPN mreže) i posebno za svakog klijenta.

Kao mogući šablon za početak stvaranja konfiguracionih datoteka se mogu iskoristiti primeri konfiguracija koji se nalaze unutar OpenVPN paketa (direktorijum **sample-config-files**). Ovi primeri se kasnije mogu prilagoditi konkretnim potrebama VPN mreže koja se implementira.

4.1.3.1 Konfiguraciona datoteka OpenVPN servera

U ovom radu OpenVPN server je implementiran na istom PC računaru na kome se nalazi i mLunch server. Neophodno je napomenuti da se pre pokretanja ostalih servera na ovom računaru mora pokrenuti OpenVPN server ne bi li se inicijalizovala i spremila za rad VPN mreža preko koje će klijenti i server kasnije komunicirati.

Konfiguraciona datoteka OpenVPN servera (*server.opvn*) unutar mLunch sistema sadrži sledeće stavke:

- Lokalna IP adresa servera preko koje se ostvaruje veza za klijentima. Ovaj parametar može da se izostavi ukoliko računar servera ima samo jednu mrežnu spregu (mrežni uređaj) ili ako je poželjno da server radi sa svim mrežnim uređajima.

```
local 192.168.1.1
```

- Port (prolaz) na kojem server očekuje uspostavu veze od strane klijenata. Za trenutnu konfiguraciju odabran je port 4444.

```
port 4444
```

- Transportni mrežni protokol preko koga se realizuje VPN mreža. OpenVPN podržava TCP i UDP protokol. Za potrebe implementirane VPN mreže odabran je UDP protokol.

```
proto udp
```

- Odabir tehnike kontrole toka saobraćaja unutar VPN mreže. U poglavlju 4.1.1 je su objašnjene dve tehnike, tehnika premošćenja (virtuelni mrežni uređaj **tap**) i preusmeravanja (virtuelni mrežni uređaj **tun**) saobraćaja i skrenuta je pažnja da je za potrebe mLunch sistema odgovarajuća tehnika preusmeravanja saobraćaja.

```
dev tun
```

- Digitalni sertifikat CA entiteta koji je potpisao sertifikate servera i klijenata, sertifikat i ključ samog servera. Ove datoteke moraju biti prisutne unutar servera i datoteka sa ključem servera mora biti tajno čuvana.

```
ca ca.crt
cert mlunch.crt
key mlunch.key # This file should be kept secret
```

- Datoteka koja u sebi sadrži Diffie Hellman parametre.

```
dh dh1024.pem
```

- Serverska uloga PC računara kao i raspon virtuelnih IP adresa koje će se koristiti unutar realizovane VPN mreže. Prva adresa u nizu se rezerviše za servera dok se ostale raspoređuju klijentima. Za potrebe mLunch sistema rezervisan je raspon od 10.8.0.0/24 IP adresa (subnet).

```
server 10.8.0.0 255.255.255.0
```

- Definisanje datoteke u kojoj se čuva lista trajnih asocijacija klijent – IP adresa. Ova lista omogućava klijentima da uvek dobiju istu virtuelnu IP adresu od strane DHCP servera ukoliko se desi neočekivani otkaz mreže.

```
ifconfig-pool-persist ipp.txt
```

- Konfigurisanje tabele preusmeravanja (eng. *routing table*) za potrebe preusmeravanja saobraćaja. Ova stavka je veoma bitna ukoliko se mLunch server nalazi na više od jednog računara, tj. OpenVPN server se nalazi na različitom računaru u odnosu na ostatak sistema. Putanje koje se ovom stavkom konfigurišu se, preko DHCP servera, predaju klijentima kojima je ovaj podatak potreban.

```
push "route 10.8.0.0 255.255.255.0"
```

- Konfigurisanje provere ispravnosti VPN veze slanjem tzv. “ping” zahteva klijentima na koje oni treba da odgovore u što skorijem vremenu. U trenutnoj konfiguraciji ping se šalje svakih 10 sekundi i ukoliko klijent ne odgovori za 300ms server će smatrati da je veza sa tim klijentom prekinuta. Period čekanja odgovora na ping zahtev se bira u zavisnosti od konfiguracije i osobina fizičke mreže koja se koristi za ostvarenje VPN mreže. Ukoliko je mreža lokalna ili širokopojasna (eng. *broadband*) vreme čekanja može biti do 120ms, ukoliko je u pitanju mobilna mreža tipa GPRS ovo vreme može da se poveća i do 300ms.

```
keepalive 10 300
```

- Definisanje kriptografskog cipher algoritma koji će se koristiti za siguran prenos podataka. Klijenti moraju da koriste isti algoritam kao i server da bi se uspostavila sigurna veza.

```
cipher DES-EDE3-CBC # Triple-DES
```

- OpenVPN nudi mogućnost komprimovanja podataka radi smanjenja opterećenja mreže. Ova stavka ima poseban značaj u mrežama u kojim usko grlo predstavlja mali protok podatka u jedinici vremena (npr. GPRS, dial-up i slično).

```
comp-lzo
```

- Za potrebe praćenja rada VPN mreže definiše se datoteka u kojoj se zapisuju sve važnije informacije kao i eventualne greške koje se javljaju u toku rada OpenVPN servera. Dodatno se može definisati nivo ispisa detalja o greškama (od 0 do 9 gde je 0 bez ispisa, a 9 detaljan ispis).

```
log openvpn.log  
verb 3
```

Nakon uspešnog stvaranja konfiguracione datoteke OpenVPN servera, isti se može pokrenuti iz DOS komandnog prozora na sledeći način:

```
openvpn server.ovpn
```

Uspesno startovanje OpenVPN servera na ekranu prikazuje sledeće informacije:

```
Mon Mar 24 12:25:20 2008 OpenVPN 2.0.9 Win32-MinGW [SSL] [LZO] built on Oct
1 2006
Mon Mar 24 12:25:20 2008 MANAGEMENT: TCP Socket listening on 127.0.0.1:7505
Mon Mar 24 12:25:20 2008 Diffie-Hellman initialized with 1024 bit key
Mon Mar 24 12:25:20 2008 TLS-Auth MTU parms [ L:1574 D:138 EF:38 EB:0 ET:0
EL:0]
Mon Mar 24 12:25:20 2008 TAP-WIN32 device [VPN] opened:
\\.\Global\{F2CA480C-C251-4CB9-8F08-178FBC3B42F5}.tap
Mon Mar 24 12:25:20 2008 TAP-Win32 Driver Version 8.4
Mon Mar 24 12:25:20 2008 TAP-Win32 MTU=1500
Mon Mar 24 12:25:20 2008 Notified TAP-Win32 driver to set a DHCP IP/netmask
of 10.8.0.1/255.255.255.0 on interface {F2CA480C-C251-4CB9-8F08-
178FBC3B42F5} [DHCP-serv: 10.8.0.0, lease-time: 31536000]
Mon Mar 24 12:25:20 2008 Sleeping for 10 seconds...
Mon Mar 24 12:25:30 2008 NOTE: FlushIpNetTable failed on interface [65540]
{F2CA480C-C251-4CB9-8F08-178FBC3B42F5} (status=259) : No more data is
available.
Mon Mar 24 12:25:30 2008 Data Channel MTU parms [ L:1574 D:1450 EF:42
EB:135 ET:32 EL:0 AF:3/1 ]
Mon Mar 24 12:25:30 2008 UDPv4 link local (bound): [undef]:4444
Mon Mar 24 12:25:30 2008 UDPv4 link remote: [undef]
Mon Mar 24 12:25:30 2008 MULTI: multi_init called, r=256 v=256
Mon Mar 24 12:25:30 2008 IFCONFIG POOL: base=10.8.0.2 size=253
Mon Mar 24 12:25:30 2008 IFCONFIG POOL LIST
Mon Mar 24 12:25:30 2008 PocketPC,10.8.0.2
Mon Mar 24 12:25:30 2008 Initialization Sequence Completed
```

4.1.3.2 Konfiguraciona datoteka OpenVPN klijenta

Jedan od zahteva ovog rada je da se klijent nalazi na mobilnom uređaju zasnovanom na Windows Mobile 5.0 operativnom sistemu. Ranije je napomenuto da OpenVPN projekat nema zvanično podršku za Windows Mobile 5.0. Ovaj problem je rešen upotrebom OpenVPN klijentom za Pocket PC koji je razvijen na osnovu izvornog koda originalnog OpenVPN klijenta na Windows PC platformi. Pocket PC verzija aplikacije je još uvek u procesu razvoja i trenutno je dostupna njena beta verzija koja zadovoljava potrebe mLunch sistema.

Konfiguraciona datoteka OpenVPN klijenta koji je pokrenut na mobilnom skener uređaju sa Windows Mobile 5.0 operativnim sistemom sadrži sledeće stavke:

- Adresu OpenVPN servera i port na kome dati server čeka na klijentske konekcije.

```
remote 192.168.1.1 4444
```

- Protokol preko kojeg će se ostvarivati VPN konekcija. Ovaj podatak mora biti isti kao i kod servera.

```
proto udp
```

- Definisanje klijentske uloge datog uređaja.

```
client
```

- Odabir tehnike kontrole saobraćaja. Ovaj podatak mora biti isti kao i kod servera.

```
dev tun
```

- Provera tipa serverskog sertifikata. U trenutnoj konfiguraciji se prihvataju samo oni sertifikati servera koji su tipa *server*.

```
ns-cert-type server
```

- Podešavanje cipher algoritma. Ovaj parametar mora biti isti kao i kod servera.

```
cipher DES-EDE3-CBC
```

- Komprimovanje podataka.

```
comp-lzo
```

- Putanje do CA sertifikata, sertifikata i ključa klijenta.

```
ca "\\Application\\Program Files\\OpenVPN\\crypto\\ca.crt"
cert "\\Application\\Program Files\\OpenVPN\\crypto\\client.crt"
key "\\Application\\Program Files\\OpenVPN\\crypto\\client.key"
```

Nakon uspešnog stvaranja klijentske konfiguracione datoteke može se pokrenuti OpenVPN klijentska aplikacija i odpočeti proces povezivanja sa OpenVPN server na osnovu date datoteke. Ukoliko se uspostavi uspešna veza sa serverom može se preći na pokretanje mLunch servera na strani servera i skener aplikacije na strani klijenta. Od ovog trenutka se sva komunikacija između servera i klijenta odvija kroz siguran VPN tunel.

4.1.4 Dodatne mere sigurnosti unutar OpenVPN sistema

Jedno od nepisanih pravila u projektovanju sigurnosnih mehanizama i sistema sa sigurnim prenosom podataka jeste da se nikada ne sme verovati samo jednoj sigurnosnoj funkciji sistema čiji bi neuspeh mogao da kompromituje rad informacionog sistema. OpenVPN nudi nekoliko mehanizama dodatne sigurnosti sa kojima se pojave sigurnosnih pretnji smanjuju na minimum.

4.1.4.1 Tls-auth direktiva

Tls-auth direktiva u konfiguraciji OpenVPN mreže stvara dodatne HMAC potpise (signature) za sve pakete u procesu SSL/TLS rukovanja radi ojačanja integriteta uspostave sigurne veze. Svaki UDP paket koji u sebi sadrži pogrešnu HMAC signaturu se odbacuje bez dodatne obrade. **Tls-auth** direktiva između ostalog dopunjava SSL/TLS protokol i uvodi zaštitu od sledećih sigurnosnih pretnji:

- DoS (eng. *Denial of Service*) napadi na OpenVPN UDP port servera.
- Skeniranje UDP portova radi otkrivanja porta na kome server sluša dolazne veze klijenata.
- Ranjivost implementacije SSL/TLS protokola na tzv “prelivanje bafera” (eng. *buffer overflow*)
- Uspostava SSL/TLS veze od strane neautorizovanih klijenata. SSL/TLS protokol ovakve uspostave može otkriti i odbaciti, ali **tls-auth** direktiva je u mogućnosti da obezbedi ranije otkrivanje i odbacivanje.

Za korišćenje **tls-auth** direktive neophodno je stvaranje ključa u obliku deljenje tajne (eng. *shared secret key*) koji se koristi kao dodatak standardnim RSA ključevima i sertifikatima. Ovaj ključ se stvara pozivom sledeće komande:

```
openvpn --genkey --secret ta.key
```

Kao rezultat se dobija statički ključ koji je zapisan unutar datoteke *ta.key*. Ova datoteka se mora predati serveru i klijentima (preko već uspostavljenog sigurnog kanala) i može se nalaziti na istom mestu gde su RSA ključevi i sertifikati.

Da bi se dati ključ mogao koristiti neophodno je dodati **tls-auth** direktivu u konfiguracione datoteke i servera i klijenta:

- Server:

```
tls-auth ta.key 0
```

- Klijent:

```
tls-auth ta.key 1
```

4.1.4.2 ProtoUdp direktiva

Iako OpenVPN nudi mogućnost korišćenja i UDP i TCP protokola kao nosećih transportnih protokola za ostvarenje VPN mreže, odabir UDP protokola pruža bolju zaštitu od DoS napada i skeniranja portova u odnosu na TCP protokol.

4.1.4.3 Dužina javnih ključeva

Dužine javnih RSA ključeva prilikom stvaranja istih se kontroliše uz pomoć parametra `KEY_SIZE` unutar *vars.bat* datoteke. Ovaj parametar se mora podesiti pre stvaranja sertifikata, ključeva servera, klijenata i CA. Inicijalna konfiguracija OpenVPN softvera postavlja `KEY_SIZE` na vrednost 1024 koja se kasnije može lako promeniti na 2048 bez bitnijeg negativnog uticaja na funkcionalnost VPN mreže.

4.1.4.4 Dužina simetričnih ključeva

OpenVPN konfiguracija koristi Blowfish cipher algoritam dužine 128 bita za potrebe šifrovanja podataka. Međutim, cipher algoritmi unutar OpenVPN mreže mogu biti svi koji su ponuđeni unutar OpenSSL biblioteke i kao takvi mogu biti i duži što se tiče broja bita ključa (npr. 256 bitni AES cipher algoritam).

4.1.4.5 Čuvanje privatnog ključa izdavaoca sertifikata

Jedna od prenosti upotrebe X509 PKI sistema jeste to da se datoteka sa privatnim ključem CA entiteta (*ca.key*) ne mora nalaziti na OpenVPN serveru (koji je najčešće meta napada na sigurnost sistema). Kod konfiguracije sistema gde se zahteva visok nivo sigurnosti podataka, za potpisivanje sertifikata uz pomoć CA ključa se definiše poseban računar koji je dobro fizički obezbeđen i isključen sa mreže za prenos podataka. Prenosni medijumi (floppy diskovi, USB flash diskovi, CD diskovi, itd.) se mogu koristiti za prenos ključa sa jednog računara na drugi ukoliko je to potrebno. Ovakve sigurnosne mere čine krađu privatnog CA ključa veoma teškom.

4.1.5 Izbegavanje mogućih napada “Čovek u sredini”

Da bi se izbegla eventualna pojava “Čovek u sredini” vrste napada, kod koje uspešno prijavljeni klijent pokušava da se poveže sa drugim klijentom predstavljajući se lažno kao OpenVPN server, neophodno je uvesti proveru sertifikata servera od strane klijenata. OpenVPN softver nudi pet različitih načina uz pomoć kojih se može dodatno proveriti autentičnost servera:

- Stvaranje sertifikata klijenata i servera sa dodatnim poljima koja opisuju namenu ključa i proširenu namenu ključa (eng. *key usage* i *extended key usage*). U specifikaciji RFC 3280 se daje mogućnost sledećih atributa koji se mogu predstaviti u TLS vezama:

Uloga	Namena ključa	Proširena upotreba ključa
Klijent	digitalni potpis	TLS web provera autentičnosti klijenta
	razmena ključa	
	digitalni potpis i razmena ključa	
Server	digitalni potpis i upotreba ključa	TLS web provera autentičnosti servera
	digitalni potpis i razmena ključa	

Prilikom stvaranja sertifikata servera OpenVPN skripta **build-key-server** postavljanjem određenih atributa određuje dati sertifikat da može da se koristi samo za potrebe servera. Nakon ovoga dovoljno je u konfiguracionoj datoteci klijenta dodati sledeću direktivu:

```
remote-cert-tls server
```

- Stvaranje serverskog sertifikata uz pomoć **build-key-server** skripte će postaviti atribut **nsCertType=server**. Nakon ovoga potrebno je dodati sledeću direktivu u konfiguraciji klijenata:

```
ns-cert-type server
```

Ovom direktivom se klijentima nalaže da odbacuju sve OpenVPN servere koji u svom sertifikatu nisu postavili atribut **nsCertType=server**, čak i u slučaju da je dati sertifikat uredno potpisan od strane validnog CA sertifikata i ključa.

- Korišćenje **tls-remote** direktive u konfiguraciji klijenata radi prihvatanja ili odbijanja servera na osnovu polja *Common Name* unutar sertifikata.
- Korišćenje **tls-verify** direktive u konfiguraciji klijenata radi prihvatanja ili odbijanja servera na osnovu ugrađenih X509 informacija serverskog sertifikata.
- Potpisivanje serverskog sertifikata sa jednim CA entitetom, a klijentskih sa drugim. U ovom slučaju CA direktiva u konfiguraciji klijenata treba da ukazuje na CA sertifikat kojim je potpisan sertifikat servera, dok CA direktiva u konfiguraciji servera treba da ukazuje na CA sertifikat kojim se potpisuju sertifikati klijenata.

4.2 Realizacija sistema za siguran prenos podataka implementacijom kriptografskih protokola i sistema za proveru identiteta korisnika

Pored VPN rešenja, siguran prenos podataka unutar mLunch sistema se može ostvariti i direktnom implementacijom PKI sistema zajedno sa SSL/TLS protokolom na strani klijenta i servera. Na ovaj način se izbegava potreba za dodatnim softverom (kao što je OpenVPN) koji je zadužen za siguran prenos podataka jer se mehanizmi provere autentičnosti i sigurnog prenosa podataka nalaze u samoj aplikaciji mobilnog skener uređaja i Apache web servera. Takođe ovakav način realizacije sigurnog prenosa podataka smanjuje količinu uloženog truda za podešavanje mLunch sistema za rad u sigurnom okruženju, a sa druge strane smanjuje potrošnju resursa koji su u datom trenutku na raspolaganju sistemu.

Realizacija PKI sistema i SSL/TLS protokola unutar servera se ostvaruje putem konfiguracionih datoteka Apache web servera uz obavezni dodatak SSL modula. Modul je razvijen od strane Apache razvojnog tima na osnovu otvorenog koda OpenSSL projekta. Na klijentskoj strani PKI sistem i SSL/TLS protokol su realizovani unutar posebnog modula unutar postojećeg projekta skener aplikacije za mobilni uređaj. Razvojno okruženje u kome je projekat realizovan je *Microsoft Visual Studio 2005* sa dodatkom *Windows Mobile 5.0 SDK for Pocket PC*, a programski jezik u kome je projekat napisan je C# (C Sharp).

4.2.1 Konfiguracija Apache web servera za rad sa sistemom za proveru identiteta korisnika

Slično kao i kod konfiguracije OpenVPN servera i Apache web server se konfiguriše i priprema za rad uz pomoć stvaranja konfiguracionih datoteka. Direktorijum u kome je instaliran Apache web server sadrži poddirektorijum pod imenom *conf* u kome se nalaze sve potrebne konfiguracione datoteke. Unutar datog direktorijuma se nalaze dve relevantne datoteke koje su bitne za rad Apache web servera sa PKI sistemom, a to su:

- **httpd.conf** – opšta podešavanja Apache web poslužioca
- **ssl.conf** – podešavanja vezana za PKI sistem i SSL/TLS protokol

Unutar **httpd.conf** datoteke se nalaze opšta podešavanja rada Apache web servera kao što su prolaz (port) na kome će server da prihvata dolazne veze klijenata, putanja do direktorijuma u kome se nalaze datoteke internet stranica koje će se posluživati klijentima, definisanje dodatnih modula koji će se koristiti prilikom rada servera (PHP skriptni jezik, SSL modul i slično) itd. Nakon instaliranja softverskog paketa Apache automatski se stvara **httpd.conf** datoteka sa standardnim skupom konfiguracionih direktiva. Za potrebe mLunch servera potrebne su minimalne izmene ove datoteke, tj. dodavanje određenih direktiva radi osposobljavanja Apache servera za rad sa PHP jezikom i SSL/TLS protokolom unutar PKI sistema.

Dodavanja modula PHP skriptnog jezika i PKI sistema za siguran prenos podataka se ostvaruje pozivom sledećih direktiva unutar **httpd.conf** datoteke:

```
LoadModule ssl_module modules/mod_ssl.so
PHPIniDir "C:\Program Files\PHP\"
LoadModule php5_module "C:\Program Files\PHP\php5apache2_2.dll"
```

Potrebno je izvršiti inicijalizaciju SSL protokola dodatnim direktivama jer se radi o statički uvazanom SSL modulu na platformi (Windows) koja ne sadrži ekvivalent Linuxovom /dev/random generatoru stohastičkih brojeva. Takođe je potrebno navesti putanju do **ssl.conf** datoteke u kojoj se nalaze dodatna podešavanja SSL modula.

```
<IfModule ssl_module>
SSLMutex default
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
SSLSessionCache none
Include conf/ssl.conf
</IfModule>
```

Posle uspešne inicijalizacije Apache web servera uz pomoć pravilno stvorene **httpd.conf** datoteke, potrebno je podesiti PKI sistem i SSL/TLS protokol uz pomoć **ssl.conf** datoteke. U ovoj datoteci se kao i kod prethodne zadaju određene direktive koje upravljaju sa radom PKI sistema, HTTPS protokola kao i protokola koji su zaduženi za siguran prenos podataka.

Za uspešnu realizaciju HTTPS protokola tj. PKI sistema u okviru mLunch informacionog sistema **ssl.conf** datoteka sadrži sledeće direktive:

- Port (prolaz) na kome server čeka usposave sigurne veze od strane klijenata. Nepisano pravilo je da se koristi port 443 zbog toga što veliki broj klijentskih aplikacija koje rade sa HTTPS protokolom podrazumevaju dati port kao standardni prilikom uspostave veze sa serverom.

Listen 443

- Dodavanje podrške za MIME tipove koji su potrebni za uspešnu obradu digitalnih sertifikata i lista sa nevažećim sertifikatima (CRL – *Certificate Revocation List*).

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

- Konfigurisanje keš memorije u vidu datoteke u kojoj se čuvaju ostvarene SSL sesije i definisanje vremena isteka uspostave SSL sesije.

```
SSLSessionCache dbm:logs/ssl-scache.log
SSLSessionCacheTimeout 300
```

- Definisanje semafora (*mutex*) koji se koristi za međusobno isključivanje i sinhronizaciju niti unutar jezgra SSL modula.

```
SSLMutex default
```

- Konfigurisanje generatora nasumice izabranih brojeva (PRNG – *Pseudo Random Number Generator*). Pošto se Apache server nalazi na Windows platformi koristi se ugrađeni generator unutar SSL modula, inače se preporučuje upotreba `/dev/random` ukoliko je server na Linux operativnom sistemu.

```
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
```

- Direktiva za pokretanje ili zaustavljanje SSL modula za datu konfiguraciju.

```
SSLEngine on
```

- Podešavanje liste niza cipher algoritama koje je server spreman da prihvati od strane klijenata.

```
SSLCipherSuite
ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
```

- Definisanje putanje do datoteke koja sadrži sertifikat servera potpisan od strane validnog CA entiteta. Sertifikat unutar datoteke mora biti kodovan u PEM binarnom formatu.

```
SSLCertificateFile conf/ssl/mLunchServer.crt
```

- Definisanje putanje do datoteke koja sadrži privatni ključ poslužioca.

```
SSLCertificateKeyFile conf/ssl/mLunchServer.key
```

- Definisanje putanje do datoteke koja sadrži digitalni sertifikat CA entiteta preko kojeg je potpisan sertifikat servera. Kao i kod sertifikata servera i ovde je potrebna PEM kodovana datoteka.

```
SSLCACertificateFile conf/ssl/ca.crt
```

- Definisanje putanje do datoteke koja sadrži listu nevažećih sertifikata (CRL). Datoteka može u sebi da sadrži jednu ili više sertifikata kodovanih u PEM formatu onih klijenata čiji su sertifikati proglašeni nevažećim.

```
SSLCARevocationPath conf/ssl.crl
```

- Konfigurisanje provere autentičnosti klijenata. Ova direktiva se poziva u slučajevima kada je potrebno proveriti digitalne sertifikate klijenata pre nego što im se omogući sigurna komunikacija sa serverom. Pored same provere definiše se i dubina provere, tj. broj CA entiteta unutar CA lanca sa gde će se proveravati klijentski sertifikat.

```
SSLVerifyClient require  
SSLVerifyDepth 1
```

- Konfiguracija opcija jezgra SSL modula. Definisanje tipova datoteka koje server treba da opslužuje putem sigurne veze i HTTPS protokola. Pored ovih opcija definiše se i skup standardnih promenljivih koje su potrebne za optimalan rad SSL modula.

```
<Files ~ "\.(cgi|shtml|phtml|php3?)$">  
    SSLOptions +StdEnvVars  
</Files>  
<Directory "cgi-bin">  
    SSLOptions +StdEnvVars  
</Directory>
```

- Podešavanje SSL/TLS protokola. Definisanje direktiva koje regulišu proces upravljanja i završetka SSL sesije. Jedan od osnovnih problema kraja SSL sesije jeste odstupanje od standarda i ne slanje SSL close direktive od strane klijentske aplikacije ili web čitača (eng. *browser*). Takođe, potrebno je sprečiti moguće napade na sigurnost SSL veze u vidu spuštanja verzije protokola (eng. *SSL downgrade*) odgovarajućom direktivom.

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

Stvaranje digitalnih sertifikata i ključeva servera, klijenata i CA je identično kao i kod OpenVPN sistema i veoma lako se mogu koristiti isti ili slični sertifikati unutar VPN mreže i Apache SSL modula.

Nakon uspešne konfiguracije SSL modula, Apache web server je spreman za prihvatanje i ostvarenje sigurne veze sa klijentima (web čitača ili klijentske skener aplikacije mLunch sistema).

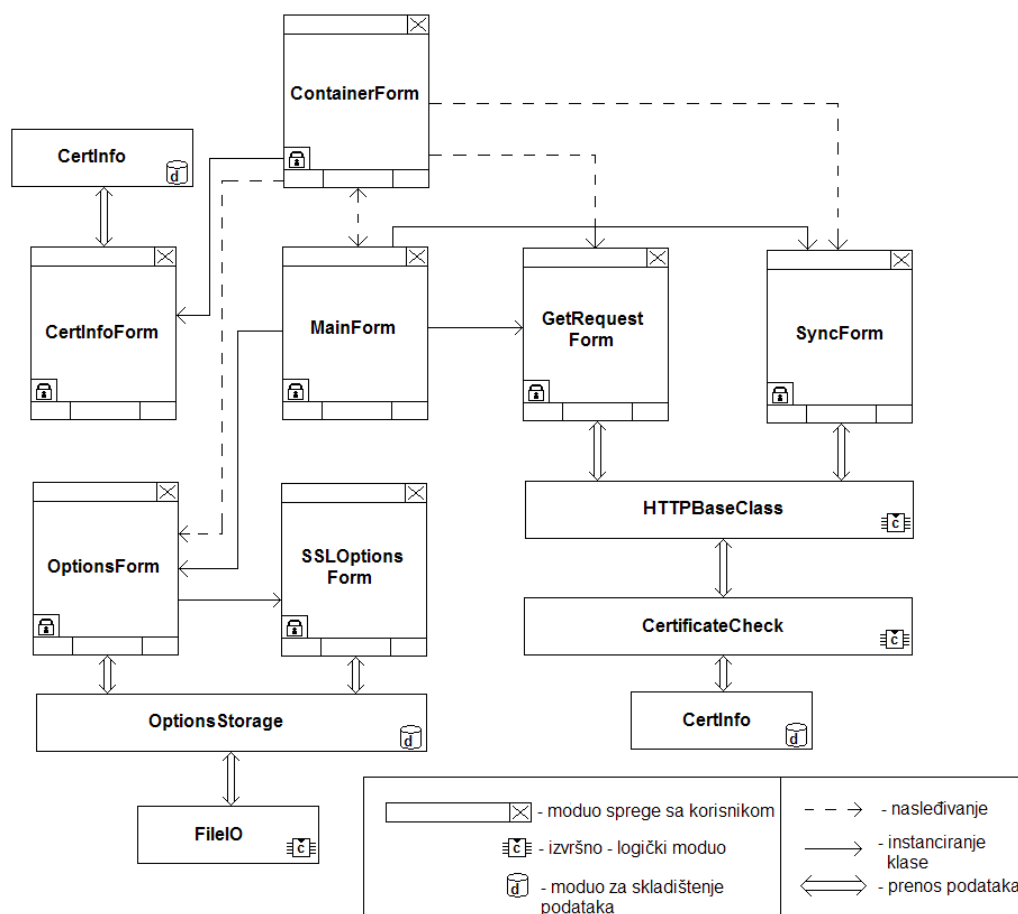
4.2.2 Realizacija i integrisanje sistema za proveru identiteta korisnika i kriptografskih protokola u sklopu skener aplikacije klijentskog uređaja

Osposobljavanje klijentske skener aplikacije mLunch sistema za rad sa PKI infrastrukturom i protokolima za siguran prenos podataka odpočinje pripremom postojećih programskih modula aplikacije za rad u novom okruženju kao i uvođenjem novih modula koji će vršiti operacije PKI sistema i SSL/TLS protokola. Programski moduli su realizovani kao posebne datoteke (koje sadrže jednu ili više definicija i realizacija klasa) u okviru projekta skener aplikacije pisane u C# programskom jeziku. Na osnovu funkcionalnosti, module možemo podeliti na:

- Module za spregu sa korisnikom (eng. *UI module*)
- Module za skladištenje podataka (eng. *data module*)
- Izvršno – logičke module (eng. *data processing module*)

Svaki od navedenih modula je podjednako važan i ima unapred definisanu ulogu u radu klijentske skener aplikacije: moduo za spregu sa korisnikom sadrži definicije klasa i instance objekata zaduženih za iscrtavanje objekata na ekranu (prozor, dugme, polje za upis teksta, slika i sl.), prikupljanje i prikaz informacija korisniku; moduo za skladištenje podataka kao zadatak ima prikupljanje i čuvanje podataka koji su u datom trenutku relevantni za dalje funkcionisanje aplikacije; izvršno – logički moduo predstavlja jezgro aplikacije, njegova uloga je logičko upravljanje i izvršavanje zadataka koji se zadaju aplikaciji na osnovu ulaznih podataka (koji se unose od strane klijenta ili se prikupljaju od servera).

Na slici br. 6 je prikazan blok dijagram povezanosti modula koji učestvuju u realizaciji PKI sistema i SSL/TLS protokola unutar klijentske skener aplikacije.



Slika 6 : Blok dijagram povezanosti modula radi ostvarenja PKI sistema i SSL protokola

4.2.2.1 ContainerForm moduo

ContainerForm je moduo za spregu sa korisnikom (UI moduo) koji je realizovan unutar datoteke *ContainerForm.cs*. Sadrži klasu *ContainerForm* koja je zadužena za iscrtavanje standardnog prozora koji u donjem levom uglu sadrži ikonu u obliku katanca koja prikazuje trenutno stanje SSL veze i PKI sistema. Ovaj moduo se nasleđuje od strane ostalih UI modula radi prikaza ikone i statusa SSL veze. Ikonica može označavati tri osnovna stanja:

- Koristi se SSL veza - 
- Ne koristi se SSL veza - 
- Postoji problem sa PKI sistemom i SSL vezom - 

Klikom na SSL ikonicu se otvara novi prozor (*CertInfoForm*) koji daje detaljne informacije i parametre o trenutnoj ostvarenoj SSL vezi.

Metode koje se koriste za iscrtavanje ikonice i reagovanje na klik na ikonicu su sledeće:

```
protected void UpdateSslIcon();
public void DisplayWarningIcon();
private void sslNotifyPanel_Click(object sender, EventArgs e);
```

UpdateSslIcon je funkcija koja na osnovu odabrane opcije upotrebe SSL veze iscrtava odgovarajuću ikonicu.

```
protected void UpdateSslIcon()
{
    if(OptionsStorage.EnableSSL)
        alpicSslNotify.RelativeImageFilePath = "secure.png";
    else
        alpicSslNotify.RelativeImageFilePath = "notsecure.png";
    alpicSslNotify.Show();
    alpicSslNotify.Visible = false;
}
```

DisplayWarningIcon je zadužena za iscrtavanje ikonice koja označava postojanje problema u PKI sistemu i SSL komunikaciji.

```
public void DisplayWarningIcon()
{
    alpicSslNotify.RelativeImageFilePath = "warning.png";
    alpicSslNotify.Show();
    alpicSslNotify.Visible = false;
}
```

sslNotifyPanelClick funkcija ima dvostruku ulogu : jedna je da ukoliko nema problema u toku SSL komunikacije klikom na ikonicu stanja SSL veze otvori novi prozor u kome se prikazuju osnovne informacije SSL protokola i PKI sistema (*CertInfoForm*); druga je da ukoliko postoje problemi sa SSL vezom i PKI sistemom stvori odgovarajući tekst o datom problemu i ispiše ga u okviru sa porukom (*MessageBox*) kada korisnik klikne na ikonicu sa oznakom greške.

```

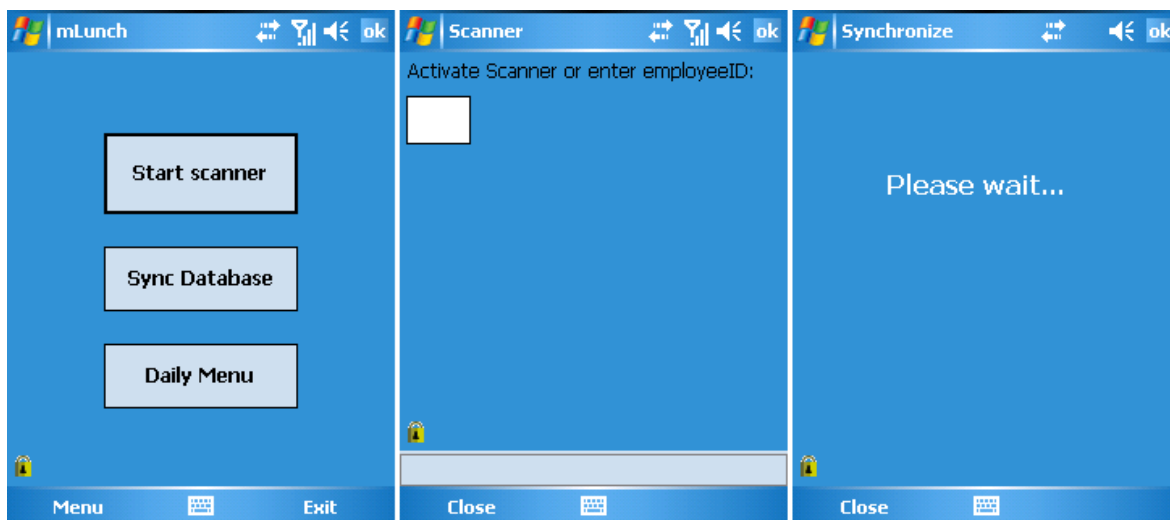
private void sslNotifyPanel_Click(object sender, EventArgs e)
{
    if (alpicSslNotify.RelativeImageFilePath != "warning.png")
    {
        //dv: Display ssl certificate info form
        CertInfoForm newCertInfoForm = new CertInfoForm();
        newCertInfoForm.ShowDialog();
    }
    else
    {
        //dv: Display warning messagebox
        string warNotMachCN = "- Server address (" +
OptionsStorage.ServerAddress + ") doesn't mach with certificate owner (" +
CertInfo.Name + ").";
        string warUntrustedROOT = "- Certificate is self signed, or
CA root (" + CertInfo.IssuerName + ") is untrusted, please import valid CA
root certificate.";
        string warString = "\n\n";
        //dv: Check SSLOptions flags for displaying notifications
        if(OptionsStorage.CARootWarn)
            if
((CertInfo.SSLErrorString.IndexOf("CertUNTRUSTEDROOT")) >= 0 )
                warString += warUntrustedROOT + "\n\n";
            if(OptionsStorage.CertOwnerWarn)
                if
((CertInfo.SSLErrorString.IndexOf("CertCN_NO_MATCH")) >= 0)
                    warString += warNotMachCN + "\n\n";
                if(OptionsStorage.CARootWarn ||
OptionsStorage.CertOwnerWarn)
                    if (warString.Length > 0)
                        MessageBox.Show("Server's security certificate
warning(s):" + warString,
                                "Server certificate
warning", MessageBoxButtons.OK,
                                MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);
    }
}

```

4.2.2.2 MainForm, GetRequestForm i SyncForm moduli

MainForm moduo predstavlja sam početak aplikacije tj. iscrtaava početni meni iz kojeg se mogu izabrati sve moguće funkcije aplikacije. *GetRequestForm* moduo iscrtaava ekran i inicijalizuje bar-kod skener radi kasnije identifikacije korisnika i preuzimanja podatka sa servera. *SyncForm* je moduo koji iscrtaava ekran sa opcijama za sinhronizaciju baze podataka (slike korisnika, informacije o naručenim obrocima za dati dan, i sl.) sa serverom. Ova tri modula su moduli za spregu sa korisnikom i kao takvi nasleđuju *ContainerForm* klasu ranije pomenutog *ContainerForm.cs* modula radi prikaza i iscrtavanja stanja SSL veze, pošto se SSL komunikacija može koristiti u slučaju sinhronizacije podataka sa serverom (*SyncForm*), kao i prilikom potvrde i identifikacije korisnika (*GetRequestForm*).

Na slici br.7 je prikazan izgled ekrana koji se iscrtavaju upotrebom *MainForm*, *GetRequestForm* i *SyncForm* modula, u donjem levom uglu se nalazi SSL ikonica koja je nasleđena iz *ContainerForm* modula.



Slika 7 : MainForm, GetRequestForm i SyncForm prozori

4.2.2.3 OptionsForm, SSLOptionsForm, OptionsStorage i FileIO moduli

OptionsForm je moduo za spregu sa korisnikom koji je zadužen za prikupljanje informacija o podešavanju rada aplikacije. Unutar *OptionsForm* klase se mogu podesiti mrežni parametri aplikacije kao što je adresa mLunch servera, upotreba SSL veze i dodatna podešavanja SSL parametara klikom na dugme “SSL Options” (čime se stvara novi objekat klase *SSLOptionsForm*). Pored parametara vezanih za komunikaciju moguće je podesiti i parametar izgleda aplikacije kao što je boja pozadine prozora aplikacije.

OptionsForm klasa u sebi sadrži dve metode za čitanje i upis parametara aplikacije u strukturu *OptionsStorage*:

```
private void OptionsForm_Load(object sender, EventArgs e);
private void OptionsForm_Closing(object sender, CancelEventArgs e);
```

OptionsForm_Load je funkcija koja se poziva prilikom inicijalizacije prozora za podešavanje parametara (*OptionsForm*) i njena uloga je da prikupi odgovarajuće parametre iz strukture u kojoj se čuvaju parametri aplikacije (*OptionsStorage*) i da ih učita u odgovarajuće komponente ekrana.

```
private void OptionsForm_Load(object sender, EventArgs e)
{
    //dv: Load settings to form controls
    serverAddress.Text = OptionsStorage.ServerAddress;
    checkBoxSSL.Checked = OptionsStorage.EnableSSL;
    ShowSSLOptionsButton();
    this.BackColor = OptionsStorage.BgFormColor;
    Appearance.BackColor = OptionsStorage.BgFormColor;
    Network.BackColor = OptionsStorage.BgFormColor;
}
```

OptionsForm_Closing se poziva prilikom zatvaranja prozora za podešavanje parametara aplikacije i ona je zadužena za upis unetih podešavanja nazad u strukturu za čuvanje parametara aplikacije (*OptionsStorage*). Pored upisa u *OptionsStorage* strukturu, ova funkcija takođe trajno čuva parametre aplikacije u datoteci “mLunchSettings.txt” uz pomoć modula *FileIO*.

```

private void OptionsForm_Closing(object sender, CancelEventArgs e)
{
    //dv:
    //If close menu item is clicked
    //close the form without saving
    //parameters, otherwise save parameters.
    if (!this.isClose)
    {
        OptionsStorage.EnableSSL = checkBoxSSL.Checked;
        OptionsStorage.ServerAddressFull = serverAddress.Text;
        OptionsStorage.ServerAddress = serverAddress.Text;
        OptionsStorage.BgFormColor = tempBgColor;
        //dv: Write settings to file
        FileIO fileSettings = new FileIO("mLunchSettings.txt", true);
        fileSettings.WriteSettings();
        fileSettings.Close();
    }
}

```

SSLOptionsForm je moduo za spregu sa korisnikom uz pomoć kojeg je moguće podesiti reagovanje aplikacije na neke od upozorenja prilikom provere digitalnog sertifikata servera. Konkretno, moguće je omogućiti ili onemogućiti ispis upozorenja korisniku ukoliko se dese sledeća dva slučaja:

- Ne postoji CA sertifikat koji je potpisao dati sertifikakt servera – *CA Root Warnings*
- *CommonName* parametar serverskog sertifikata se ne poklapa sa DNS imenom samog servera – *Certificate owner warnings*

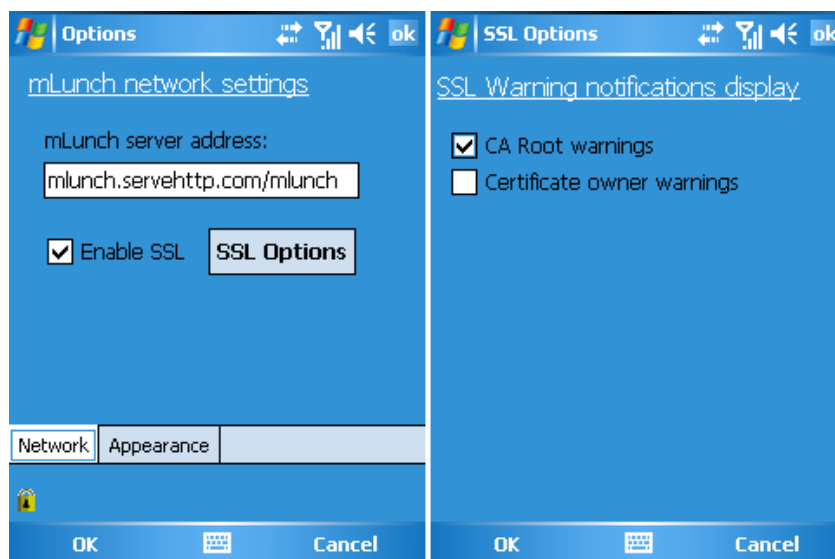
Odabir prikaza datih upozorenja se vrši uz pomoć komponenti prozora *CheckBox* koje se mogu uključiti ili isključiti u zavisnosti od odabira korisnika.

Kao i kod *OptionsForm* klase i ovde se koriste dve metode za čuvanje odabira prikaza upozorenja na date događaje unutar *OptionsStorage* strukture:

```

private void SSLOptionsForm_Load(object sender, EventArgs e);
private void SSLOptionsForm_Closing(object sender, CancelEventArgs e);

```



Slika 8 : *OptionsForm* i *SSLOptionsForm* prozori

OptionsStorage je moduo koji pripada grupi modula za skladištenje podatka (*data module*). Unutar njega je definisana statička klasa *OptionsStorage* koja sadrži statičke metode i atribute potrebne za skladištenje podešavanja aplikacije. Informacije koje se čuvaju unutar ovog modula su sledeće:

- Adresa mLunch servera
- Puna adresa mLunch servera – sa http ili https prefiksom
- Upotreba SSL veze
- Tekst upozorenja o SSL grešci
- Prikaz upozorenja o ne postojanju ispravnog CA sertifikata
- Prikaz upozorenja o neispravnom parametru *CommonName* unutar serverskog sertifikata
- Boja pozadine prozora aplikacije
- Prikaz poruka o greškama u mrežnoj komunikaciji

Atributi u kojima se smeštaju date informacije su:

```
private static string m_serverAddress;
private static string m_serverAddressFull;
private static bool m_enableSSL = true;
private static string m_sslWarnStr = "";
private static bool m_CARootWarn = true;
private static bool m_CertOwnerWarn = true;
private static Color m_bgFormColor;
private static bool m_ShowNetExcMsg = true;
```

Metode koje se koriste za čitanje i upis podataka u odgovarajuće atribute su realizovani putem GET i SET direktiva koje su specifične za C# programski jezik. Dat je prikaz definicije statičkih metoda za upis i čitanje podataka kao što su adresa mLunch servera i upotreba SSL veze. Po istom principu su realizovane i metode za preostale atribute.

```
public static string ServerAddressFull
{
    get
    {
        if (m_enableSSL) return m_serverAddressFull;
        else return m_serverAddressFull;
    }
    set
    {
        if (m_enableSSL) m_serverAddressFull = "https://" + value + "/";
        else m_serverAddressFull = "http://" + value + "/";
    }
}

public static string ServerAddress
{
    get { return m_serverAddress; }
    set { m_serverAddress = value; }
}

public static bool EnableSSL
{
    get { return m_enableSSL; }
    set { m_enableSSL = value; }
}
```

FileIO moduo je predstavnik grupe izvršnih modula (*data processing module*). Njegova uloga je da otvori datoteku u kojoj će upisati, ili iz koje će se pročitati informacije o podešavanju aplikacije. Informacije se iz učitane datoteke upisuju u *OptionsStorage* strukturu, a obrnuto upisuju u datoteku iz iste strukture. Moduo sadrži klasu *FileIO* u čijem se konstruktoru otvara zadata datoteka za čitanje ili upis.

```
//dv: Open for read or write (pMode=true - write; pMode=false - read)
public FileIO(string pFileName, bool pMode)
{
    fileName = pFileName;
    if (pMode)
    {
        newFileStream = File.Open(pFileName, FileMode.Create);
        outputFileWrite = new StreamWriter(newFileStream);
        mode = pMode;
    }
    else
    {
        newFileStream = File.Open(pFileName, FileMode.Open);
        outputFileRead = new StreamReader(newFileStream);
        mode = pMode;
    }
}
```

Ulazni parametri konstruktora:

- `string pFileName` – putanja i ime datoteke koja se koristi
- `bool pMode` – režim rada nad datotekom (upis ili čitanje)

Ostale metode koje su realizovane su:

- `public void WriteLine(string pText);` - upis jedne linije teksta u datoteku
- `public void WriteSettings();` - upis svih parametara u datoteku (uz pomoć *WriteLine metode*) iz *OptionsStorage* strukture
- `public void ReadSettings();` - čitanje svih parametara iz datoteke u *OptionsStorage* strukturu
- `public string ExtractSetting(string pSettingStr);` - preuzimanje informacije iz učitane linije teksta
- `public string ReadLine();` - čitanje jedne linije teksta iz datoteke
- `public void Close();` - zatvaranje prethodno otvorene datoteke
- `public bool Exists();` - provera da li datoteka postoji

4.2.2.4 HTTPBaseClass, CertificateCheck, CertInfo i CertInfoForm moduli

Ova grupa modula međusobnom saradnjom vrše proveru sertifikata servera i ukoliko je provera uspešna ostvaruje sigurnu SSL vezu sa mLunch serverom. Proces uspostave sigurne veze otpočinje sa izvršnim modulom *HTTPBaseClass* u kojem se stvara HTTPS zahtev (ukoliko je uključena opcija upotrebe SSL veze) serveru putem *HttpWebRequest* strukture. Pre nego što se dati zahtev inicijalizuje potrebno je redefinisati klasu *Certificatepolicy* koja je sadržana unutar *ServicePointManager* strukturi koja se nalazi u sistemskom paketu *System.Net Compact .NET 2.0* okruženja. Klasa koja će predstavljati redefiniciju *Certificatepolicy* klase je sadržana unutar

izvršnog modula *CertificateCheck*. Nakon ovog koraka sledi stvaranje samog zahteva putem *HttpRequest* strukture kojoj se prosleđuje odgovarajuća adresa tj. URI mLunch servera.

```
System.Net.ServicePointManager.CertificatePolicy = new
CertificateCheck(true);
HttpRequest webrequest = (HttpRequest)WebRequest.Create(uri);
webrequest = (HttpRequest)WebRequest.Create(uri);
webrequest.KeepAlive = false;
webrequest.Method = RequestMethod;
webrequest.Timeout = 30000;
```

CertificateCheck moduo proverava sertifikat servera koju mu se prosledi uz pomoć redefinisane metode *CheckValidationResult*.

```
public bool CheckValidationResult(ServicePoint sp, X509Certificate cert,
WebRequest req, int problem)
{
    //dv: Search for problem code in problemCode array
    int problemCode = Array.BinarySearch(CertInfo.certProblemCodeArray,
problem);
    //dv: If problem is found find it's string value
    if ((problemCode == 8) || (problemCode == 13))
    {
        CertInfo.SSLErrorString +=
CertInfo.certProblemStringArray.GetValue(problemCode) + "\n";
        CertInfo.SSLWarning = true;
        if (problemCode == 8)
            CertInfo.CAWarn = true;
        else
            CertInfo.CAWarn = false;
    }
    else if (problemCode >= 0)
    {
        CertInfo.SSLErrorString =
CertInfo.certProblemStringArray.GetValue(problemCode) + "\n";
        accept = false;
    }
    CertInfo.IssuerName = cert.GetIssuerName();
    CertInfo.Name = cert.GetName();
    CertInfo.ValidFrom = cert.GetEffectiveDateString();
    CertInfo.ValidTo = cert.GetExpirationDateString();
    CertInfo.SerialNumber = cert.GetSerialNumberString();

    CertInfo.Format = cert.GetFormat();
    CertInfo.HashCode = Convert.ToString(cert.GetHashCode());
    CertInfo.HashValue = cert.GetCertHashString();

    CertInfo.PublicKey = cert.GetPublicKeyString();
    CertInfo.KeyAlgorithm = cert.GetKeyAlgorithm();
    CertInfo.KeyAlgorithmParams = cert.GetKeyAlgorithmParametersString();

    return accept;
}
```

Ulazni parametri *CheckValidationResult* strukture:

- *ServicePoint* sp - trenutna *ServicePoint* struktura
- *X509Certificate* cert - sertifikat servera u X509 v3. formatu
- *WebRequest* req - *WebRequest* zahtev koji je inicirao SSL vezu
- *int* problem - numerička oznaka problema sa sertifikatom

Prilikom provere sertifikata servera unutar *CheckValidationResult* funkcije vrši se popunjavanje strukture *CertInfo* sa podacima iz serverskog sertifikata. *CertInfo* strukutra je sadržana unutar *data* modula *CertInfo.cs*. Kao i kod *OptionsStorage* modula *CertInfo* klasa je statička struktura sa statičkim atributima i metodama. Atributi su zapravo podaci iz sertifikata koji se preko odgovarajućih metoda (GET i SET direktiva) mogu sačuvati ili pročitati.

Parametri sertifikata koji se čuvaju u modulu *CertInfo* se nalaze unutar sledećih atributa:

```
private static bool m_sslWarning = false;
private static bool m_sslError = false;
private static string m_sslErrStr = "";
private static bool m_CAWarn = false;
private static string m_IssuerName = "";
private static string m_Name = "";
private static string m_ValidFrom = "";
private static string m_ValidTo = "";
private static string m_SerialNumber = "";
private static string m_Format = "";
private static string m_HashCode = "";
private static string m_HashValue = "";
private static string m_PublicKey = "";
private static string m_KeyAlgorithm = "";
private static string m_KeyAlgorithmParams = "";
```

Pored parametara *CertInfo* struktura sadrži i dva niza:

- `public static Array certProblemCodeArray;` - Niz od 16 numeričkih kodova mogućih problema prilikom provere sertifikata
- `public static Array certProblemStringArray;` - Niz od 16 tekstualnih reprezentacija mogućih problema prilikom provere sertifikata

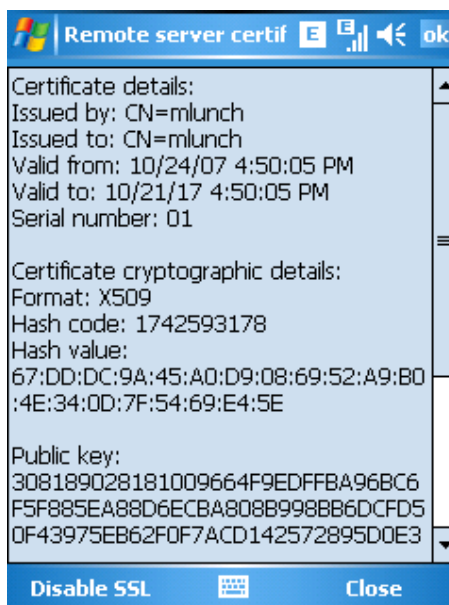
Niz *certProblemCodeArray* i *certProblemStringArray* se preslikavaju jedan na drugi, tj. svaki član iz jednog niza poseduje ekvivalentnog člana u drugom nizu.

Prikaz informacija o trenutno ostvarenoj SSL vezi i parametrima sertifikata servera se ostvaruje preko modula za spregu sa korisnikom *CertInfoForm*. Prikaz informacija je organizovan u vidu prozora u kome se nalaze sve relevantne informacije o serverskom sertifikatu kao što su:

- Informacije o sertifikatu
 - Ime CA entiteta koji je potpisao dati sertifikat
 - Ime servera kojem je sertifikat izdat
 - Datum početka stupanja na snagu sertifikata
 - Datum kraja važenja sertifikata
 - Serijski broj sertifikata
- Kriptografski detalji sertifikata
 - Format zapisa sertifikata
 - Heš kod (*hash code*)
 - Heš vrednost (*hash value*)
 - Javni ključ
 - Algoritam ključa
 - Parametri algoritma ključa

Prikaz i inicijalizacija *CertInfoForm* prozora otpočinje pritiskom na ikonicu stanja SSL veze u donjem levom uglu prozora aplikacije. Ukoliko je SSL veza u upotrebi u *CertInfoForm* prozoru će biti prikazani gore navedeni parametri SSL veze i sertifikata servera, a u suprotnom

će korisnik biti obavešten o tome da se SSL veza ne koristi sa mogućnošću da se klikom na određeno dugme “*Enabe SSL*” SSL veza aktivira. Na slici br. 9 je prikazan izgled *CertInfoForm* prozora sa informacijama o trenutnoj vezi i sertifikatu mLunch servera.



Slika 9 : *CertInfoForm* prozor

5. TESTIRANJE I VERIFIKACIJA

Proces testiranja i verifikacije je podeljen na dve faze. Svaka faza se odnosi posebno za realizaciju sigurnog prenosa podataka preko VPN mreže i posebno za realizaciju HTTPS protokola unutar aplikacije u mobilnom uređaju. Prva faza predstavlja testiranje uspostave sigurne veze između mLunch servera i mobilnog klijenta i otpornost ostvarene sigurne veze na sigurnosne pretnje kao što su ranije napomenuti “čovjek u sredini” napadi na sigurnost informacionog sistema i računarske mreže. Druga faza testiranja je testiranje performansi sistema, tj. poređenje rezultata rada sistema sa i bez upotrebe sigurnog prenosa podataka.

Za potrebe testiranja sigurnosti i performansi VPN mreže i realizovanog HTTPS protokola unutar mLunch sistema korišćena su dva softverska paketa:

- Cain
- Wireshark

Cain je softverski paket koji u sebi sadrži veliki broj alati za testiranje sigurnosti prenosa podataka kroz računarske mreže, a pored toga sadrži i mnoge alatke za analizu i probijanje lozinki koje se čuvaju u šifrovanom obliku (npr. *MD5* ili *SHA* algoritam). Za proces testiranja sigurnosti prenosa podataka u ovom radu je najvažnija mogućnost Cain softverskog paketa da vrši i analizira sigurnosne napade tipa “čovjek u sredini” između dva ili više računara u računarskoj mreži. Cain je potpuno besplatan i može se preuzeti sa internet stranice www.oxid.it.

Wireshark je aplikacija za prikupljanje, čuvanje i analizu svih mrežnih paketa koji prolaze kroz računar na kome se ova aplikacija nalazi. Uz pomoć Wireshark aplikacije je moguće testiranje performansi prenosa podataka samim time što se svaki preneti paket podataka može razložiti na delove i posebno analizirati.

Oba pomenuta softverska paketa u sebi sadrže WinPcap 4.1 *open source* dodatak koji se koristi u mnogim besplatnim alatima za analizu mrežnog saobraćaja. WinPcap (*Windows Packet Capture*) predstavlja jezgro ovih alati.

5.1 Testiranje otpornosti sistema na sigurnosne napade

Prva faza testiranja je realizovana preko unapred definisanih testnih slučajeva koji se izvršavaju i kasnije ocenjuju kao uspešni, neuspešni ili neodređeni. Za potrebe testiranja otpornosti na sigurnosne napade “čovjek u sredini” izdvojen je poseban PC računar koji se nalazi na istoj računarskoj mreži gde se nalaze mLunch server i klijentski mobilni uređaj. Prvo je testirana mogućnost prisluškivanja prenesenih podataka između klijenta i servera čija je veza

ostvarena preko OpenVPN mreže, a zatim uz pomoć HTTPS protokola. U tabelama br. 2 i br. 3 su prikazani testni slučajevi kao i rezultati testiranja otpornosti sistema na napade “Čovek u sredini”.

Tabela 2 : Testiranje otpornosti mLunch sistema na napade tipa “Čovek u sredini” za OpenVPN rešenje.

mLunch VPN – test otpornosti na napade tipa “Čovek u sredini”

Opis testova					Rezultat
Test ID	Kratak opis	Preduslovi	Akcija	Očekivani rezultat	Rezultat
001	Čovek u sredini – bilo koja metoda napada (napadač nije unutar VPN mreže)	1. Server i klijent uspešno prijavljeni na VPN mrežu 2. Napadač nema pristup VPN mreži	Izvršenje Mitnikovog napada, napada maskiranjem web stranice i napad maskiranjem ARP paketa na sigurnost mLunch sistema	Zbog činjenice da napadač nema pristup VPN mreži, Čovek u sredini napadi se ne mogu uspešno ostvariti.	Uspešan
002	Čovek u sredini – Mitnikov napad (napadač je unutar VPN mreže)	1. Server i klijent uspešno prijavljeni na VPN mrežu 2. Napadač ima pristup VPN mreži	Izvršenje Mitnikovog napada na sigurnost mLunch sistema	Arhitektura mLunch servera ne dozvoljava izvršenje ove vrste napada. Sync-flood napad se ignoriše i ne predstavlja opasnost po sigurnost sistema.	Uspešan
003	Čovek u sredini – Maskiranje adrese web stranice (napadač je unutar VPN mreže)	1. Server i klijent uspešno prijavljeni na VPN mrežu 2. Napadač ima pristup VPN mreži	Izvršenje napada metodom maskiranja web stranice na sigurnost mLunch sistema	Provera domena servera pored provere ispravnosti serverskog sertifikata ne dozvoljava da se ovaj napad realizuje. Odgovarajuća poruka upozorenja se ispisuje kod klijenta.	Uspešan
004	Čovek u sredini – Maskiranje ARP paketa (napadač je unutar VPN mreže)	1. Server i klijent uspešno prijavljeni na VPN mrežu 2. Napadač ima pristup VPN mreži	Izvršenje napada metodom maskiranja ARP paketa na sigurnost mLunch sistema	Napadač ima pristup šifrovanim podacima koji se razmenjuju između klijenta i servera, ali ne postoji način da ih dešifruje (nesdostatak privatnog ključa sertifikata).	Uspešan

Table 3 : Testiranje otpornosti mLunch sistema na napade tipa "Čovek u sredini" za HTTPS rešenje.

Opis testova					Rezultat
Test ID	Kratak opis	Preduslovi	Akcija	Očekivani rezultat	Rezultat
005	Čovek u sredini – Mitnikov napad	Napadač ima pristup mreži za prenos podataka na kojoj su klijent i server.	Izvršenje Mitnikovog napada na sigurnost mLunch sistema	Arhitektura mLunch servera ne dozvoljava izvršenje ove vrste napada. Sync-flood napad se ignoriše i ne predstavlja opasnost po sigurnost sistema.	Uspešan
006	Čovek u sredini – Maskiranje adrese web stranice	Napadač ima pristup mreži za prenos podataka na kojoj su klijent i server.	Izvršenje napada metodom maskiranja web stranice na sigurnost mLunch sistema	Provera domena servera pored provere ispravnosti serverskog sertifikata ne dozvoljava da se ovaj napad realizuje. Odgovarajuća poruka upozorenja se ispisuje kod klijenta.	Uspešan
007	Čovek u sredini – Maskiranje ARP paketa	Napadač ima pristup mreži za prenos podataka na kojoj su klijent i server.	Izvršenje napada metodom maskiranja ARP paketa na sigurnost mLunch sistema	Napadač ima pristup šifrovanim podacima koji se razmenjuju između klijenta i servera, ali ne postoji način da ih dešifruje (nesдостatak privatnog ključa sertifikata).	Uspešan

Dalje testiranje otpornosti PKI sistema na falisifikaciju, lažno predstavljanje i probijanje ključeva koji se koriste za šifrovanje podataka nije potrebno jer se podrazumeva da su softverski paketi OpenSSL, OpenVPN, PKI i SSL/TLS moduli unutar VPN realizacije i realizacije HTTPS protokola u samom klijentu testirani i verifikovani na osnovu relevantnih RFC dokumenata i testova koje su vršeni prilikom realizacije istih.

5.2 Analiza protoka podataka i odziva sistema

Druga faza testiranja obuhvata upotrebu Wireshark aplikacije na strani mLunch servera. Testni slučajevi su osmišljeni tako da se prvo analiziraju paketi koji se prenose bez uspostave sigurne veze, a kasnije upotrebom VPN rešenja i HTTPS rešenja. U svim slučajevima se mere parametri rada sistema kao što su količina prenetih podataka, odziv sistema, potrošnja resursa, i slično. Analizom paketa u kojima su sadržane informacije koje se prenose unutar VPN mreže pored korisnih, šifrovanih podataka, upotreba SSL/TLS protokola kao i VPN tehnika upravljanjem tokom saobraćaja uvodi pojavu tzv. viška podataka (eng. *overhead*) koji se dodaje na ukupne korisne podatke.

Višak podataka u nivou sigurnog prenosa podataka

I (1)	II (20)	III (16)	IV (4)
-------	---------	----------	--------

I – oznaka paketa, veličine 1 bajt.

II – HMAC-SHA1 signatura , veličine 20 bajta.

III – vektor inicijalizacije, veličine 16 bajta.

IV – broj sekvence, veličine 4 bajta.

Ukupna veličina viška podataka koji su sadržani u jednom paketu nivoa sigurnog prenosa (eng. *security layer overhead*) je $1 + 20 + 16 + 4 = 41$ bajta.

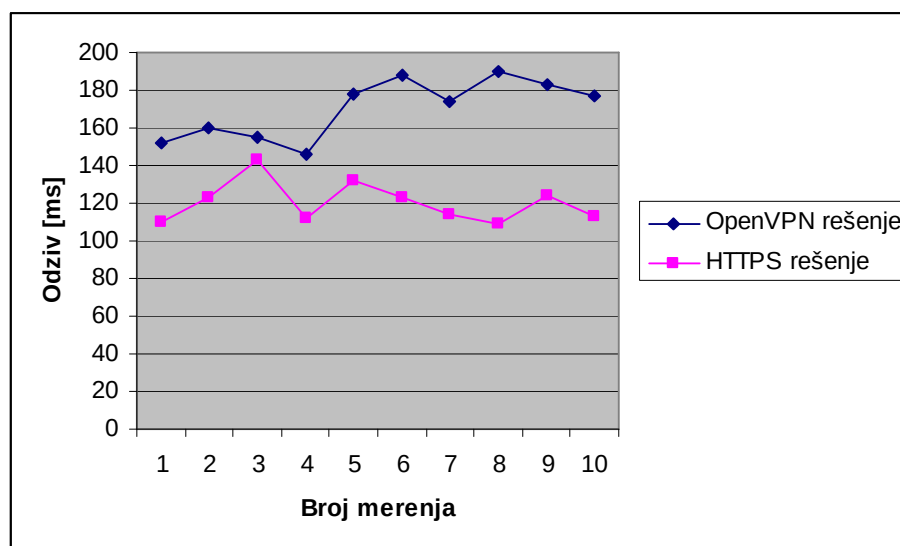
Višak podataka u transportnom nivou VPN mreže

IP zaglavlje	UDP zaglavlje
--------------	---------------

Ukupna veličina viška podatka usled korišćenja VPN mreže za prenos podataka je 28 bajta (IP zaglavlje + UDP zaglavlje).

Za OpenVPN realizaciju ukupni višak podataka po svakom paketu iznosi $41 + 28 = 69$ bajta. Ovaj višak se može ublažiti primenom *LZO* kompresije paketa koju nudi OpenVPN rešenje. U slučaju HTTPS realizacije, pod viškom se podrazumevaju samo podaci iz nivoa sigurnog prenosa podataka, tj. 41 bajt.

Odziv sistema, tj. brzina odgovora servera na zahtev klijenata sa strane mrežne infrastrukture se meri slanjem tzv. ping zahteva. Ping zahtev je paket određene veličine (najčešće 32 bajta) nasumice odabranih podataka koji klijent šalje serveru, server prilikom prihvatanja ping zahteva taj isti paket salje nazad klijentu. Klijent meri vreme koje je proteklo od trenutka slanja ping paketa do trenutka prihvatanja istog (izraženo u milisekundama). Što je vreme kraće, to je odziv sistema brži. Na slici br. 10 je prikazan grafik vremenskog odziva servera mLunch sistema prema klijentu.



Slika 10 : Grafik vremenskog odziva mLunch servera prema klijentu.

Sa datog grafika se može zaključiti da je odziv servera unutar VPN mreže nešto sporiji od odziva unutar HTTPS rešenja. Ovi rezultati su očekivani zbog činjenice da se unutar OpenVPN mreže svaki paket dodatno obrađuje, preusmerava do odredišta i dodatno kompresuje radi smanjenja viška podataka što kao posledicu ima malo kašnjenje u odzivu. Kod rešenja koje uključuje HTTPS realizaciju dodatna obrada paketa ne postoji pa je stoga i odziv sistema brži.

6. ZAKLJUČAK

U ovom radu su realizovane dve metode sigurnog prenosa podataka u okviru informacionog sistema za elektronsko naručivanje obroka mLunch. Osnovni ciljevi ovog rada su ukazivanje na sigurnosne pretnje sa kojima se susreću moderni informacioni sistemi, pregled sigurnosnih pretnji kao i najefikasnije metode zaštite i dostizanja zadovoljavajućeg nivoa sigurnosti unutar sistema.

Rešenja koja su odabrana i realizovana unutar mLunch sistema – VPN mreža i HTTPS protokol, predstavljaju najefikasnije metode koje se primenjuju u savremenim informacionim sistemima radi očuvanja privatnosti podataka koji se prenose. VPN je realizovan na osnovu softverskog paketa otvorenog koda (*Open Source*) OpenVPN koji se odlikuje velikom pouzdanošću, raličitim metodatama zaštite podataka kao i lakim konfigurisanjem za rad u bilo kom mrežnom okruženju. Realizacija HTTPS protokola uz pomoć implementacije PKI sistema i SSL/TLS protokola unutar klijentske aplikacije predstavlja drugačije, ali podjednako dobro rešenje koje otklanja potrebu za dodatnim softverom na strani klijenta i servera, smanjenim utroškom resursa i potrebom za veoma malim početnim konfigurisanjem aplikacije.

U toku realizacije ovog rada naišlo se na jedno ograničenje koje je vezano za implementaciju HTTPS protokola unutar klijentske aplikacije mobilnog uređaja. Ograničenje se se ogleda u nemogućnosti implementacije provere klijentskog sertifikata od strane mLunch servera usled nedostatka programske podrške *Compact .NET 2.0 framework* okruženja u kojem je razvijana klijentska aplikacija. Za razliku od gotovog rešenja unutar OpenVPN paketa koji nudi ovu mogućnost, provera klijentskog sertifikata u okviru implemntiranog HTTPS protokola će biti moguća tek nakon pojave *Compact .NET 3.0 framework* koje će sadržati alate (funkcije) za uspešnu realizaciju provere klijentskog sertifikata unutar PKI sistema.

Potencijalni razvoj mLunch sistema bi mogao da uvede sledeće sigurnosne mehanizme unutar mLunch sistema:

- Integrisanje dodatne sigurnosti digitalnih privatnih ključeva koji se koriste za šifrovanje (enkripciju) podataka (npr. primenom pametnih kartica – eng. *smart cards*).
- Implementacija sistema za rano otkrivanje i sprečavanje neovlašćenog pristupa mreži za prenos podataka mLunch sistema – IDS.
- Uvođenje dodatnih sigurnosnih protokola za proveru identiteta korisnika sistema.
- Realizovanje provere digitalnog sertifikata klijentskog skener uređaja uz pomoć *Compact .NET Framework 3.0*.

7. LITERATURA

- [1] An Example of a Man-in-the-middle Attack Against Server Authenticated SSL-Sessions, Mattias Eriksson, June 2006
- [2] A Weakness in the 4.2BSD Unix TCP/IP Software, Robert T. Morris, AT&T Bell Laboratories, February 1985
- [3] Network Intrusion Detection, Stephan Northcutt and Judy Novac
- [4] Web Spoofing: An Internet Con Game, Edward W. Felten, Dirk Balfanz, Drew Dean, and Dan S. Wallach. Technical Report 540-96 (revised Feb. 1997), Department of Computer Science, Princeton University.
- [5] Multi-user cryptographic techniques, Whitfield Diffie and Martin Hellman, June 8 1976.
- [6] A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, R. Rivest, A. Shamir, L. Adleman, April 197
- [7] A Note on 'Non-Secret Encryption', Clifford Cocks, CESG Research Report, 20 November 1973
- [8] Safeguarding cryptographic keys, G. Blakley, June 1979.
- [9] "The Data Encryption Standard (DES)", Bruce Schneier, 15 June 2000
- [10] Guide to Intrusion Detection and Prevention Systems (IDPS), NIST CSRC special publication SP 800-94, released February 2007.
- [11] SSL 3.0 Specification, Webpage retrieved on 4 April 2008, <http://wp.netscape.com/eng/ssl3/>
- [12] TLS specification (RFC 2246), Webpage retrieved on 4 April 2008, <http://www.ietf.org/rfc/rfc2246.txt>
- [13] IP Based Virtual Private Networks, RFC 2764, B. Gleeson, February 2000
- [14] Provider Provisioned Virtual Private Network (VPN) Terminology, RFC4026, L. Andersson and T. Madsen, March 2005
- [15] www.openvpn.org
- [16] OpenVPN for PocketPC, <http://ovpnppc.ziggurat29.com/ovpnppc-main.htm>

8. PRILOG

U ovom poglavlju su navedene dodatne informacije i dokumenti koji su povezani sa temom ovog rada.

8.1 Primeri sigurnosnih napada tipa “čovjek u sredini” ^[1]

U naredna tri poglavlja su predstavljena tri najpozantije i najopasnije tehnike napada na sigurnost računarskih mreža tipa “čovjek u sredini” (eng. *Man In The Middle*).

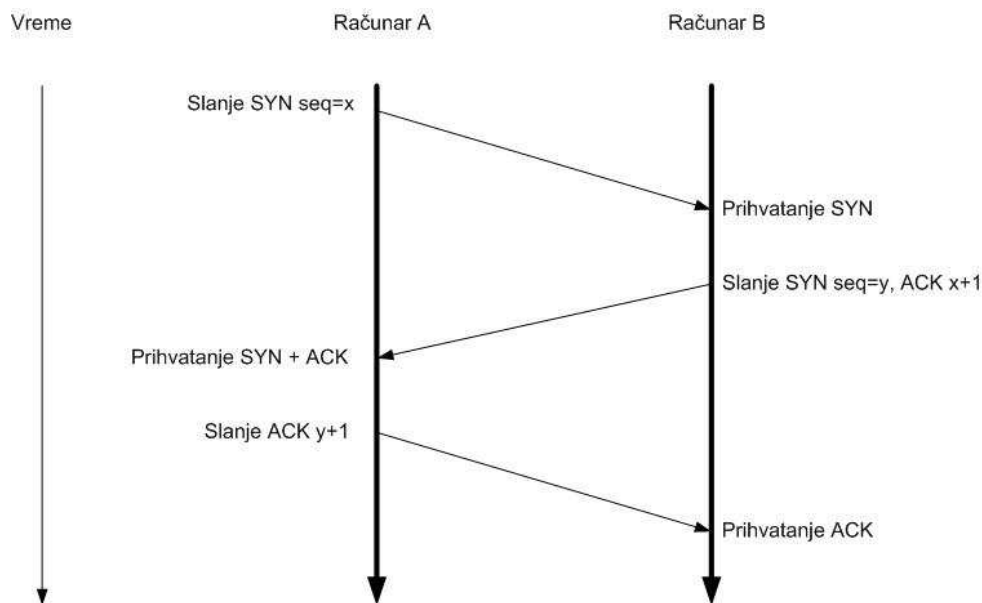
8.1.1 Mitnikov napad ^[2]

Mitnikov napad je jedna vrsta *Man in the middle* napada na sigurnost računarske mreže. Napad se bazira na iskorišćavanju slabosti u osnovnom dizajnu TCP/IP protokola. Napad se izvršava iz nekoliko koraka:

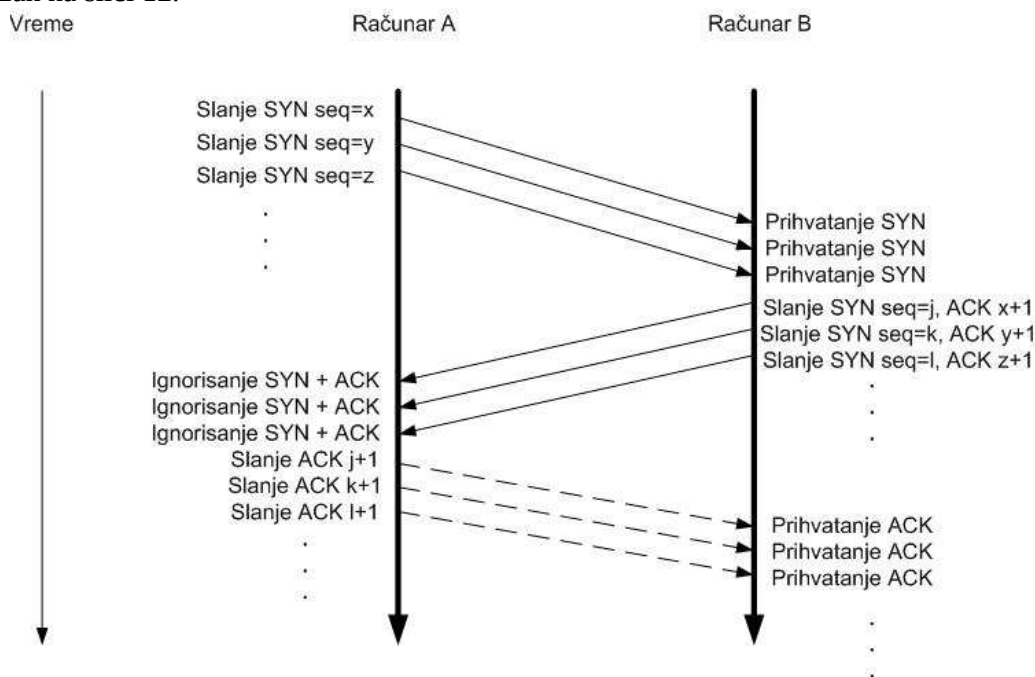
- Identifikacija slabo osigurane veze između dva računara i prikupljanje neophodnih informacija za ostvarenje napada.
- Učutkivanje pravog servera koji se zamenjuje serverom samog napadača.
- Sam čin preuzimanja i prosleđivanja podatka kao i lažnog predstavljanja.

Mitnikov napad identifikuje određene usluge (service) koji su pokrenuti na strani servera na koga će kasnije biti izvršen napad. Identifikacija servisa se vrši alatkama koje se normalno koriste od strane administratora mreže. Informacije koje se prikupljaju korišćenjem ovih alatki na prvi pogled izgledaju beznačajno i bezopasno, ali u kombinaciji sa odgovarajućim predznanjem napadača ovakve informacije se mogu iskoristiti za uspešno preuzimanje sistema (eng. *Hijacking*). Današnji sistemi i serveri su dosta sigurniji po pitanju izlaganja ovoj vrsti napada tako da se ovaj napad retko može primetiti u praksi.

Nakon što su informacije o uslugama servera prikupljene, one se analiziraju i odgovarajući metod napada se planira. Nekoliko sekundi pre nego što se krene sa postupkom preuzimanja sistema, Mitnikov metod pokušava da potpuno odseče servera iz komunikacije, tj. da ga “učutka”. Ovo se postiže tehnikom koja se zove SYN-flooding. SYN-flooding napad se bazira na proceduri uspostave TCP veze (eng. *Three way handshake* – slika 11.) između dve strane u komunikaciji.

Slika 11 : TCP uspostava veze (*three way hadnshake*)

Kada se procedura uspostave veze završi, obe strane dobijaju informaciju da druga strana komunikacije prima podatke ispravno. Napad se vodi činjenicom da se u toku komunikacije na Internetu određeni paketi gube i da se po isteku određenog vremenskog perioda moraju ponovo poslati. Posle slanja početnog SYN paketa i primanja SYN/ACK odgovora od druge strane treći korak, a to je slanje potvrde SYN/ACK, se nikad ne izvrši tako da druga strana ima utisak da potvrda za početni SYN paket nikad nije stigla i da se mora opet poslati. Slanjem velikog broja SYN paketa serveru uz ignorisanje potvrdnih paketa, sve moguće veze ka serveru se mogu prebaciti u stanje u kome server čeka na odgovor druge strane (strane napadača) koji nikada nije poslat. Pošto se odgovori na SYN pakete ignorišu, lažna IP adresa se može koristiti radi izbegavanja identifikacije ili lažnog predstavljanja napadača. Dijagram SYN-flood napada je prikazan na slici 12.



Slika 12 : SYN flood metod

Kada se jedna strana komunikacije učutka, preuzimanje sistema može da odpočne. Ovo se izvršava tako što se predviđa broj sekvence TCP/IP paketa i po toj sekvenci vrši dalje slanje podataka. Jedna od stvari koje treba imati na umu ovde jeste da sa strane servera nije bilo indikacije da je druga strana komunikacije promenila IP adresu. Upravo ovu slabost u računarskoj komunikaciji je Mitnikov napad iskorišćava. IP adresa je deo IP-sloja koji nije u korelaciji sa brojem sekvence TCP paketa. Ovo zanči da IP-sloj ne može da zna kojoj TCP vezi (sesiji) dati paket pripada, čime se daje mogućnost menjanja IP adrese u toku samog napada.

Nakon uspešnog preuzimanja sistema, napadač ima potpunu kontrolu nad protokom informacija između servera i klijenata.

8.1.2 Metod maskiranja adrese web stranice ^[4]

Drugačiji pristup *Man in the middle* vrsti napada se može predstaviti u obliku napada maskiranjem adrese web stranice (eng. *Web spoofing*). Ovom vrstom napada se može kreirati lažna kopija (eng. *shadow copy*) internet prezentacija svetske mreže (eng. *World Wide Web - WWW*).

Metoda koja se koristi za pozicioniranje napadača između korisnika i prezentacije koju korisnik posećuje se zove prepisivanje URL-a (eng. *URL rewriting*). Jezgro alatke koja se koristi za ovakvu vrstu napada se sastoji od web aplikacije koja preuzima stvarni sadržaj Internet stranice koju korisnik želi da poseti, tj. pravi izmenjenu kopiju iste i takvu šalje nazad korisniku. Jedini preduslov za uspešno sprovođenje ovakve vrste napada je uspešno navođenje korisnika na datu lažnu kopiju web stranice.

Napad se sastoji iz sledećih koraka:

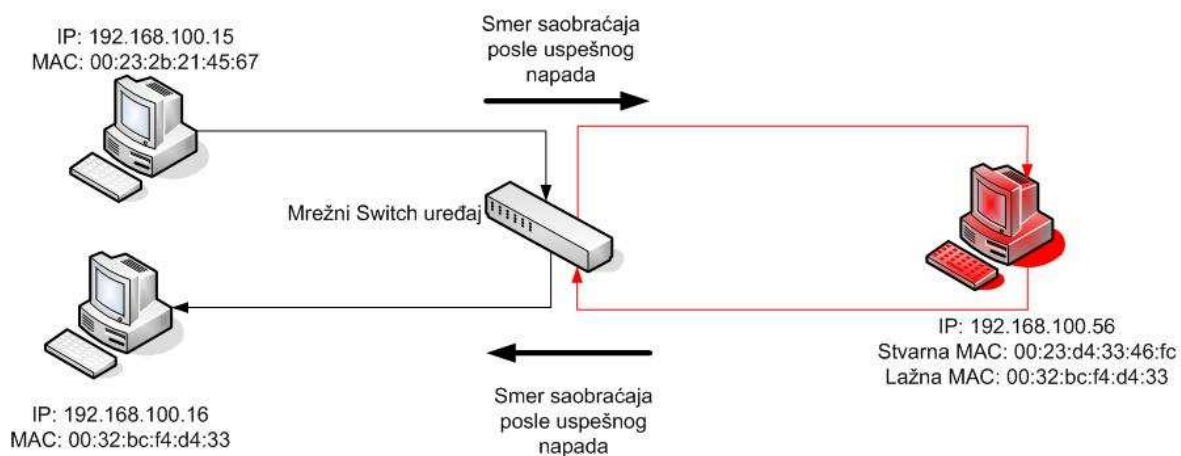
- Korisnik se navodi na lažnu kopiju internet stranice klikom na putanju (eng. *link*) do nje. Ime putanje može da bude npr. <http://home.netscape.com> ali u realnosti ona pokazuje na <http://www.napadac.com/http://home.netscape.com>. Adresa internet stranice koju korisnik želi da poseti je zapravo nalepljena na adresu web aplikacije samog napadača tako da aplikacija tačno zna koju adresu je korisnik hteo da poseti.
- Web aplikacija zatim preuzima sadržaj originalne stranice i menja sve putanje na njoj tako da one uvek pokazuju na adresu napadačeve web aplikacije sa dodatim pravim putanjama na kraju. Ovim se postiže to da korisnik stalno posećuje stranice koje su zapravo servirane na napadačevoj web aplikaciji. Ostale informacije se takođe mogu menjati po potrebi napadača.
- Modifikovana stranica se šalje korisniku.

Ovakva prosta web aplikacija kreira iluziju kopija Internet prezentacija. Međutim, ove kopije nisu validne jer se njihov sadržaj preuzima i menja od strane napadača u realnom vremenu. Mana ovakvog napada se ogleda u činjenici da napad nije kompletno transparentan sa strane korisnika tj. žrtve napada pošto se URL adresa napadača prikazuje unutar aplikacije za pretraživanje internet stranica (eng. *Browser*). Međutim, malo iskusniji napadač može zaobići prikazivanje prave adrese korišćenjem java skripti koje su podržane od većine današnjih pretraživača.

8.1.3 Metod maskiranja paketa za rezoluciju fizičkih adresa računara ^[1]

ARP protokol se koristi za prevođenje IP adrese u fizičku MAC adresu računara unutar lokalne mreže (mreža koja je bazirana na Switch mrežnim uređajima). Pošto je ova vrsta *Man in the middle* napada ograničena na lokalne mreže, rizik ovakvog napada ne obuhvata Internet mrežu već samo deo ili segment Intranet lokalne mreže.

Da bi se uspešno prisluškivao saobraćaj u ovakvoj mreži napadač odašilje lažne ARP odgovore u kojima navodi da napadačev računar sadrži MAC adresu svih ostalih računara unutar mreže. Ovim se postiže to da sav saobraćaj koji je namenjen datoj MAC adresi prolazi kroz računar napadača koji prisluškuje dati saobraćaj i po potrebi prosleđuje izmenjene podatke. Grafički prikaz ove vrste napada prikazan je na slici 13.



Slika 13 : Metod maskiranja ARP paketa

8.2 Rad sa 51. konferencije ETRAN-a

U prilogu se nalazi rad pod nazivom “Prikaz arhitekture Sony Streamman usluge za prenos multimedijalnih audio sadržaja putem Interneta i mobilnih mreža” objavljen na 51. Konferenciji ETRAN-a, održanoj u Herceg Novom - Igalu, 4 - 8. juna 2007. godine.

