



Univerzitet
Singidunum

Departman za poslediplomske studije

SAVREMENE INFORMACIONE TEHNOLOGIJE

MASTER STUDIJE

- Master rad -

**APLIKACIJA ZA ŠIFROVANJE
FAJLOVA NA *WEB-U***

Mentor:

Prof.dr. Mladen Veinović

Kandidat:

Nebojša Asenijević

Sadržaj:

Contents

1.SERVERI.....	5
1.1 Upotreba.....	6
1.2 Hardver servera	7
1.3 Operativni sistemi servera.....	9
1.4 Serveri na Internetu.....	11
2 ŠTA JE WEB APLIKACIJA	12
2.1 Istorija	12
2.2 Interfejs	13
2.3 Struktura.....	14
2.4 Pisanje Web aplikacija	15
3 WEB APLIKACIJA	16
3.1 Šifrovanje	16
3.2 Dešifrovanje	20
4 MOGUĆNOSTI U KREIRANJU APLIKACIJE	22
4.1 Klijentska strana	22
4.2 Serverska strana	23
4.3 Scenario realizacije.....	24
5.IIS	25
5.1 Obrada zahteva	27
5.2 Karakteristike	32
5.3 Bezbednost	32
6 SSL	34
6.1 SSL protokol i njegova namena.....	34
6.2 SSL i TCP/IP	35
6.3 TLS	36
6.4 Struktura SSL-a	36

6.4.1 Faza rukovanja (Handshake)	38
6.4.2 Autentifikacija servera	42
6.4.3 Autentifikacija klijenta	43
6.4.4 Faza prenosa podataka (data transfer)	45
6.4.5 SSL Record protokol	45
6.4.6 Pasivni napadi na SSL Record Protocol	48
6.4.7 Aktivni napadi na SSL Record Protocol	49
6.4.8 Napadi na izbor kriptografskog paketa	49
6.4.9 Napadi prilikom razmene ključeva	49
6.4.10 Napadi na verziju protokola	50
6.4.11 SSL poruke	51
6.5 Efikasnost	53
6.5.1 SSL brzina	53
6.6 Primena SSL-a	55
6.6.1 SSL Proxy servis	56
6.6.2 SSL VPN servis	56
6.6.3 SSL Mail servis	57
6.6.4 SSL za FTP servis	57
6.7 Testiranje SSL i njegove ocene	58
7 NABAVLJANJE I INSTALIRANJE IIS-A	60
7.1 Postavljanje aplikacije	63
7.2 Podešavanje SSL-a	63
8 ZAKLJUČAK	65

Sažetak

U ovom master radu izvršena je prezentacija web aplikacije koja šifrjuje podatke, kao i kompletna njena implementacija. Aplikacija je radjena u Microsoft Visual Studio 2008, u proframskom jeziku C#. Pošto je to web aplikacija za njeno upravljanje koristi se Microsoft Internet Information Services 7.0, a glavna funkcija mu je pružanje SSL usluge.

Tokom rada biće prikazana kompletna aplikacija i rad servera i servisa kao što je IIS. Analizirani su neki scenariji izvođenja aplikacije, kao i analiza bezbednost rada SSL-a kao glavnog aktera u obezbeđivanju sigurnog protoka informacija.

Ključne reči: *web aplikacija, server, IIS, SSL.*

Abstract

In this master thesis presentation of web application for data czphering has been mada, and also it's full implementation. Application has been developed in Microsoft Visual Studio 2008 in C# programming language. As it is web application, Microsoft Internet Information Services 7.0 is used for its managment, and its main purpose is ussage of SSL.

During the work we shall familiar our self with application, server work and services like IIS. Analysis of some scenarios of application performace has been made, and also security of SSL analysis, as main thing in securing the safty of informations.

Key words: *web aplication, server IIS, SSL.*

1 SERVERI

U informatici izraz “server” može da se odnosi na razne termine: fax server, DNS server, fajl server, server baze podataka, proksi server, server za štampače...

Ali kada se kaže server uglavnom se misli na nešto od sledećeg:

- Kompjuterski program koji radi kao servis, služeći potrebama ili zahtevima drugih programa, koji može biti, ali i ne mora, pokrenut sa istog računara(u ovom kontekstu to je „klijent“)
- Fizički računar koji vrši jedan ili više takvih servisa, koji služi potrebama programa koji rade na računarima u istoj mreži
- Softver/hardver sistem kao što su baze podataka, fajl serveri ili server za štampače

Server računar je računar, ili više računara, poji zajedno povezuje druge računare ili neke druge električne uređaje. Oni često pružaju osnovne usluge preko mreže, bilo da su to privatni korisnici unutar neke mreže ili javni korisnici preko interneta. Na primer, kada unesete upit u pretraživač, upit se sa vašeg računara šalje preko interneta do servera koji sardži relevantne veb stranice. Dobijene rezultate server vraća nazad na vaš računar.

Mogi serveri imaju određenu funkcionalnost i služe samo toj svrsi, kao što su veb serveri, serveri baze podataka, serveri za štampače. Npr enterprajz serveri se koriste samo u poslovne svrhe.



Slika1.Prikaz servera

1.1 Upotreba

U informacionim tehnologijama termin „server“ se koristi prilično široko. Uprkos mnogim server-brendiranim proizvodima, kao što su serverske verzije hardvera, softvera ili operativnih sistema, u teoriji bilo koji kompjuterizovani proces koji deli resurse jednom ili više klijentskih procesa je server. Da bi se ovo bolje ilustrovalo uzmimo na primer fajl šering (file sharing). Iako ga postojanje fajlova na mašini ne klasifikuje kao server, sam mehanizam deljenja fajlova klijentu od strane operativnog sistema ga čini serverom.

Slično, posmatrajmo posmatrajmo veb server aplikaciju, kao što je multiplatformski “Apache http Server”. Ovaj veb server softver se može pokrenuti na bilo kom računaru. Na primer, dok laptop ili personalni računar nisu obično poznati kao server, oni mogu da u ovakvim situacijama ispunjavaju ulogu jednog, i samim tim biti etiketirani kao server. U ovom slučaju se vidi da sama upotreba računara kao servera može da ga klasifikuje kao server uopšte.



Slika2. Soba sa serverima

U smislu hardvera, reč server obično označava modela računara namenjen za hostovanje softverskih aplikacija pod teškim zahtevima mrežnog okruženja. U ovoj konfiguraciji klijent-server jedna ili više mašina, bilo da je to računar ili računarski uređaj, dele informacije jedni sa drugim, sa tim da se jedan ponaša kao domaćin za drugoga.

Iako je svaki personalni računar sposoban da se ponaša kao server, predodređeni server će imati karakteristike koje ga čine podobnijim za takvo radno okruženje.

Ove karakteristike mogu da uključuju brži processor, RAM povećanih visokih performansi, i obično više nego jedan veliki hard disk. Mnogo očiglednije razlike uključuju očigledniju redundantnos u napajanju i mrežnim konekcijama.

Usled velike upotrebe *posvećenog hardvera* nastala je potreba za samostalnim serverskim uređajima. To su računarski uređaji koji rade kao server. Konstruisani su tako da korisnik nema potrebe da zna detalje njegovog operativnog sistema ili komande koje on koristi. Serverski uređaji su prekonfigurisani od strane proizvođača, i koriste tzv *plug-and-*

play tehnologiju. Naime, uređaj se jednostavno priključi na mrežu, i uključi se. Pojavljuje se jednostavan korisnički meni koji kroz veoma jednostavne korake dozvoljava korisniku da odabere aktivne komponente i izvrši najosnovnija podešavanja. Kada se uređaj butuje, što traje par minuta, spreman je za upotrebu. Primeri ovakvih uređaja su *Google Mini* (indeksiranje velikih sajtova), *ASA Firewall* (štiti mrežu od spoljnjih napada), i razni drugi uređaji koji omogućavaju usluge vezane za servere kao što su skladišni server, *FTP*, *DNS*, *VPN*, *e'mail* itd.

Moderni operativni sistemi kao što su *Microsoft Windows* ili *Linux distribucije*, se dizajniraju tako da koriste klijent-server arhitekturu. OS-mi pokušavaju da dozvole velikom broju softvera da rade na komponentama računara. Oni često dozvoljavaju programerima da pišu aplikacije koje su visokih performansi i nezavisne od hardvera tako što izbegavaju standardne pozive OS-a hardveru.

Ovi operativni sistemi mogu da izvršavaju programe u pozadini, a ti programi se zovu *servisi*, za Windows ili *demoni* za Linux. Oni se izvršavaju u pozadini i bez kontrole korisnika. Mogu čekati u stanju hibernacije dok ne budu potrebni, kao što radi npr *Apache HTTP Server* softver. Pošto se svaki softver koji omogućava uslugu može nazivati „server“, moderni personalni računari mogu da se dožive kaošuma servera i računara koji rade u „*paraleli*“.

Sam Internet je takođe šuma servera i klijenata. Prostim zahtevom za *web* stranom koja se nalazi negde na par kilometara, zahteva zadovoljavanje hrpe protokola koja obuhvataju mnoge primere hardvera i softvera servera. Najneznatniji od njih su ruteri, modemi, DNS i drugi razni serveri koji nam omogućavaju *world wide web*.

1.2 Hardver servera

Hardverski zahtevi servera variraju u zavisnosti od serverskih aplikacija. Apsolutna brzina procesora nije toliko kritična za server kao što je za personalni računar. Dužnost servera da obezbedi usluge mnogim korisnicima preko mreže, vodi do različitih zahteva, kao što su brze mrežne konekcije i visoki ulazno/izlazni opseg. Pošto se serverima obično pristupa preko mreže, oni mogu da rade u „bezglavom“ modu, tj bez monitora ili ulaznih uređaja. Procesi koji nisu potrebni za funkcionisanje servera se ne koriste. Mnogi serveri nemaju *GUI* (interfejs za grafiku) jer ne nepotreban i koristi resurse koji se mogu alocirati negde drugde. Slično, audio i USB uređaji mogu biti izostavljeni.



Slika3. Izgled servera iznutra

Serveri često rade jako dugo bez prekida a dostupnost veoma često mora biti na visokom nivou, što čini pouzdanost i dugovečnost hardvera veoma bitnim. Iako serveri mogu biti napravljeni od običnih računarskih delova, enterprajz serveri za “kritične misije” su uglavnom veoma otporni na greške i koriste specijalizovani hardver sa veoma niskom “stopom rušenja” (frekvencija pada sistema) u nameri da maksimizuju vreme ispravnog rada, jer i za najkraći pad sistema može da košta više nego kupovina i instaliranje sistema zajedno. Na primer, dovoljno je da vreme nesipravnog rada bude samo par minuta kod server koji je u nacionalnoj berzi, da bi mogao da se opravda trošak zamene celokupnog sistema sa nečim što je više pouzdano. Serveri mogu da objedine hard diskove koji su brži i većeg kapaciteta, velike kompjuterske ventilatore ili vodeno hlađenje za smanjenje visoke radne temperature, i neprekidno napajanje da bi se osigurao nesmetani rad servera u slučajevima nestanka struje. Ove komponente nude bolje performanse i pouzdanost po odgovarajućoj visokoj ceni. Hardverska redundantnost je u veoma širokoj upotrebi. To je dupliciranje kritičnih komponenti sistema, kao što su napajanja i hard diskovi, poređanih tako da jedan od njih zakaže drugi se automatski aktivira. Takođe se koriste ECC memorijski uređaji koji detektuju i ispravljaju greške, jer druge memorije mogu da dovedu do gubljenja podataka.

Da bi povećali pouzdanost većina servera koristi memoriju sa detekcijom greške i ispravljanjem, redundantne diskove i redundantno napajanje koji su uglavnom *hot swap* tj. mogu se zameniti na server koji radi bez njegovog gašenja. Iako servere koriste kvalifikovani inženjeri, njihovi operativni sistemi su više podešeni za stabilnost i performance nego što su lakši za korisničku upotrebu u čemu primetno prednjače Linux distribucije.

Kako je serverima potrebno stabilno napajanje, dobar pristup internet, povećana bezbednost i uz to su veoma bučni, potrebno je da se drže u za to određenim serverskim centrima ili specijalnim prostorijama. Ovo zahteva smanjivanje potrošnje energije jer višak

energije stvara veću toplotu i temperature u prostoriji može probiti prihvatljive granice. Obično su serverske prostorije opremljene sa klima uređajima. Kućišta servera su obično pljosnata i široka tako da su pogodna za skladištenje više uređaja jedan do drugog u serverskom ormaru.

Za razliku od običnih računara serveri se obično mogu konfigurisati, isključivati, uključivati, restartovati daljinski korišćenjem *out-of-band* upravljanjem. Mnogim serverima je potrebno dosta vremena da se pokrene hardver i učita operativni sistem. To je zbog činjenice da rade opširan test memorije pre dizanja sistema i verifikaciju i startovanje daljinskih servisa upravljanja. Onda kontroleri hard diskova startuju diskove sekvencijalno radije nego sve odjednom da ne bi preopteretili sistem napajanja. Posle toga počinje predprovera *RAID* sistema zbog ispravnog izvođenja redodantnosti. Uobičajeno je da se mašina startuje po nekoliko minuta, ali joj restartovanje nije potrebno mesecima pa čak i godinama.

1.3 Operativni sistemi servera

Serverski orijentisani operativni sistemi imaju sklonost ka određenim zajedničkim karakteristikama koje ih čine pogodnijim za serversko okruženje, kao što su:

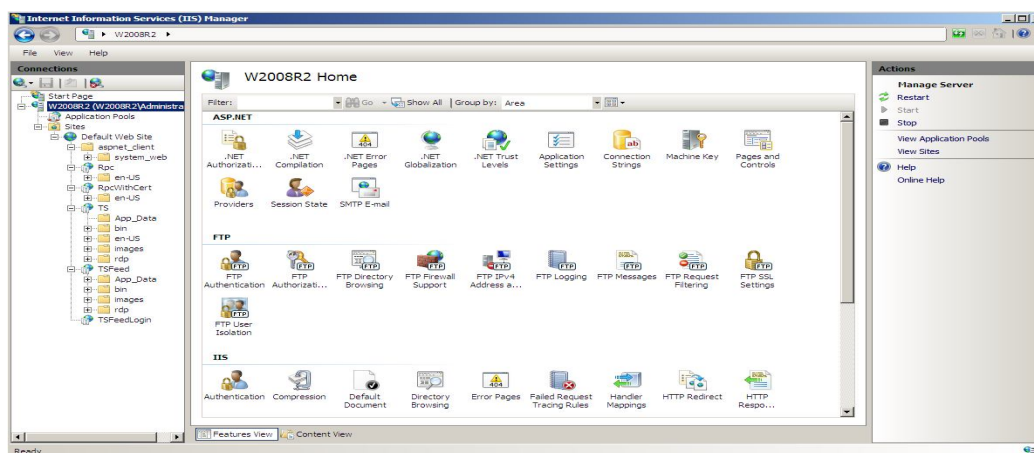
- *GUI* nije dostupan ili je opcijalan
- Mogućnost da se rekonfiguriše ili unapredi hardver i softver, do nekog stepena, bez restartovanja
- Unapređeni kapaciteti za bekapovanje zbog čestih i redovnih on line bekapa kritičnih podataka
- Transparentan transfer podataka između particija ili uređaja
- Fleksibilne i napredne mrežne mogućnosti
- Automatske mogućnosti kao što su *daemons* u Unix-u i servisi u Windows-u
- Jaka bezbednost sistema sa unapređenom zaštitom za korisnike, resurse, podatke i memoriju

Serverski orijentisani operativni sistemi u mnogo slučajeva da međusobno deluju sa hardverskim senzorima u cilju detektovanja stanja kao što su pregrevanje, zakazivanje procesora i diska i na osnovu toga obavesti operatora ili sami odklone kvar.

Dok serveri moraju da obezbede ograničen opseg usluga ka uglavnom velikom broju korisnika, desk top računari moraju da izvrše širok opseg funkcija koje zahteva jedan korisnik, samim tim zahtevi operativnog sistema servera su različiti od onih na desk top računarima. Neki operativni sistemi imaju sličan korisnički interfejs za serverske i desk top verzije.



Slika4. Linux-oliki OS



Slika5. Windows Server

Windows i Mac OS operativni sistemi nisu toliko rasprostranjeni među serverima, dok su neki uglavno zastupljeni među *mainframe* računarima kao što je z/OS. Dominantni operativni sistemi za servere se Unix-oliki ili kerneli otvorenog koda kao što je *Linux*.

Uspori servera baziranih na mikroprocesorima olakšan je razvojem Unixa koji radi na x86 procesorskoj arhitekturi. Porodica *Microsoft Windows* operativnih sistemata kodje radi na x86 arhitekturi hardvera, i od *Windows NT*-a oni su takodje na raspolaganju za serversku upotrebu.

Dok uloga servera i desktopa još uvek ostaje različita, napredci u pouzdanosti, hardveru i operativnim sistemima smanjili su razliku između ove dve klase. Danas, mnogi desktop i serverski operativni sistemi imaju sličan osnovni kod, a razlikuju se uglavnom u konfiguraciji. Promena ka *web* aplikacijama i *middleware* platformama je takodje smanjila zahteve za specijalizovanim aplikacionim serverima.

1.4 Serveri na Internetu

Skoro cela sktruktura na internetu je zasnovana na klijent-server modelu. Usmeravanje saobraćaja na internetu se vrši preko rutera, DNS servera i *root nameserver*-a. Neki od susluga koje vrse su:

- *World wide web*
- *Domain Name System*
- *E-mail*
- *FTP file transfer*
- *Chat*
- Glasovna komunikacija
- Strimovanje audio i video zapisa
- Igranje preko mreže
- Serveri baza podataka

Bukvalno svaka akcija koju uradi prosečni internet korisnik zahteva jednu ili više interakcija jednog ili više servera.

2 ŠTA JE WEB APLIKACIJA

Web aplikacija je aplikacija kojoj se pristupa preko Interneta ili intraneta. Sam termin takođe može da znači i računarski softverska aplikacija koja je *hostvovana* u brovserski kontrolisanom okruženju (*Java* aplet) ili programirana u brovserski podržanom jeziku (npr *JavaScript* u kombinaciji sa *HTML*-om) i oslanja se na običan web brouzing da renderuje izvršnu aplikaciju.

Web aplikacije su popularne zbog sverpisutnosti *web browser-a* i pogodnost upotrebe *web browser-a* kao klijenta. Oni se takodje zovu i *tin* klijenti (tanki klijenti). Mogućnost da se *web* aplikacija unapredi i održava bez distribuiranja i instaliranja softvera na hiljadama klijentski računara, je ključni razlog za njihovu popularnost.

2.1 Istorija

U ranijim kompjuterskim modelima, tj. klijent-server, većina aplikacija je deljena izmedju koda na serveru i koda instaliranog na svakom klijentu lokalno. Drugim rečima, aplikacija je imala svoj klijentski program koji je služio kao njen korisnički interfejs i morao je da bude instaliran na svakom korisničkom računaru. Unapređivanje koda na serverskoj strani aplikacije bi zahtevalo i unapredjivanje na klijentskoj strani koda koji se odvojeno instalira na svakoj radnoj stanici, što dovodi do povećanja troškova u podršci i smanjuje produktivnost.

Nasuprot tome, *web* aplikacije koriste *web* dokumenta koji se pišu u standardnom formatu kao što je *HTML* (i od skoro *XHTML*), koji je podržan od strane mnogih *web browsera*. Generalno, svaka pojedinačna *web* stranica se dostavlja klijentu kao statički dokument, ali niz stranica mogu da pruže interaktivno iskustvo, kako se korisnikov ulaz vraća kroz elemente *web* forme usađenim u stranicu. Tokom sesije, *web browser* interpretira i prikazuje strane, i ponaša se kao univerzalni klijent za bilo koju *web* aplikaciju.

Netscape je 1995 god, predstavio skript jezik na klijentskoj strani pod nazivim *JavaScript*, koji je dozvoljavao programerima da dodaju neke dinamičke elemente u korisnički interfejs koji je radio na klijentskoj strani. Do tada, svi podaci su morali da se šalju do servera na obradu, a rezultati su morali da se vraćaju klijentu kroz statičke *HTML* strane.

Tokom 1996 god *Macromedia* je predstavila *Flash*, plejer vektorske animacije koji je mogao da se doda *browser-u* kao *plug-in* da bi se usadile animacije u *web* strane. To je

dozvoljavalo upotrebu skript jezika u programskim interakcijama na klijentskoj strani bez potrebe da se komunicira sa serverom.

Tokom 1999 god koncept „web aplikacija“ je predstavljen u *Java* jeziku u *Servlet Specification* verziji 2.2. U to vreme i *JavaScript* i *XML* su već bili razvijeni, ali *Ajax* još nije izmišljen i *XMLHttpRequest (XHR)* je tek bio predstavljen u *Internet Explorer 5* kao *ActiveX* objekt.

Tokom 2005 *Ajax* je napravljen i aplikacije kao što je *Gmail* su činile da klijentska strana bude sve više i više interaktivnija.

2.2 Interfejs

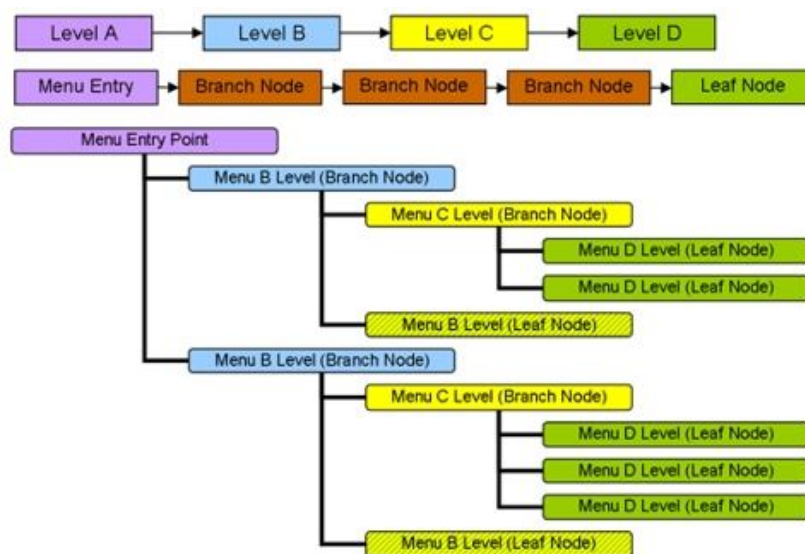
Metode specifične za aplikaciju kao što su crtanje na ekranu, puštanje muzike, pristup mišu i tastaturi, sada mogu da se ostvare zahvaljujući *Java*, *JavaScript*, *DHTML*, *Flash*, *Silverlight* i drugim tehnologijama. Mnogi servisi su radili zajedno da bi sklopili ove u poznatiji interfejs koji prihvata izgled operativnog sistema. Generalna svrha tehnika kao što je *drag and drop* je takođe podržana od strane ovih tehnologija.

Slika6. Web aplikacija

Web programeri često koriste šifrovanje na klijentskoj strani da bi dodali funkcionalnost, posebno kada se pravi interaktivni doživljaj kome ne treba ponovno učitavanje strane. U skorije vreme, razvijene su tehnologije koje koordinisu šifrovanje sa tehnologijama na serverskoj strani kao što je *PHP*. *Ajax* je tehnologija koja koristi kombinaciju ostalih tehnologija i ona je primer tehnologije koja stvara interaktivni doživljaj.

2.3 Struktura

Aplikacije se obično „lome na manje komade“ koji se nazivaju „tiers“ (nivoi), gde svaki *nivo* ima svoju ulogu. Tradicionalne aplikacije se sastoje od samo jednog nivoa, koji se nalazi na klijentskom računaru, ali *web* aplikacije same po sebi imaju *n* slojeva. Iako si mnoge varijacije moguće, najuobičajenija struktura je „*troslojna*“. U osnovnoj formi tri sloja se zovu *prezentacija*, *aplikacija* i *skladištenje*. *Web browser* je u prvom sloju (prezentacija). *Engine* (pogon aplikacije) koji koristi neku *web* tehnologiju (kao što su *ASP*, *ASP.NET CGI*, *ColodFusion*, *JSP/Java*, *PHP*, *Perl*, *Python*, *Ruby* ili *Struts2*) je srednji sloj (logika aplikacije). Baza podataka je treći sloj. *Web browser* šalje zahtev u srednji sloj, koji obrađuje zahtev praveći upite i *update*-e bazi i generiše korisnički interfejs.



SLika7. Web drvo

Za kompleksnije aplikacije troslojne aplikacije nisu najbolje rešenje. Tu treba tražiti rešenje u *n*-slojnom pristupu, kde je najveća dobit razbijanje poslovne logike, koja se nalazi u aplikacionom sloju, u model koji je više finije naštelovan model. takođe je moguće dodati integracioni sloj koji razdvaja sloj podataka (bazu) od ostalih slojeva obezbeđujući interfejs koji je lak za upotrebu da bi se pristupilo podacima.

Postoje i mišljenja da se *web* aplikacija gleda koa dvoslojna arhitektura. Može da bude „pametna“ klijent koji radi sav posao i upite i „glupi“ server, ili „glupi“ klijent koji se oslanja na „pametni“ server. Klijent i rado sa prezentacionim slojem, server bi držao bazu (skladišni sloj), a logika posla (aplikacioni sloj) bi bila na jednom od njih ili na oba. Iako ovo povećava skalabilnost aplikacija i odavaja prikaz i bazu, i dalje ne dozvoljava pravu specijalizaciju slojeva pa će većina aplikacija prerasti ovaj model.

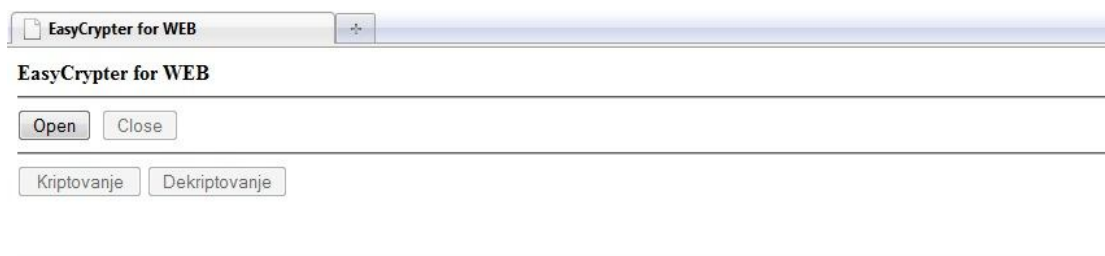
2.4 Pisanje Web aplikacija

Postoje mnoga *framework* okruženja za pisanje mrežnih aplikacija koji olakšavaju ubrzani razvoj aplikacija koja dozvoljavaju programeru da napiše program na više nivoa. Dodatak je i to što “drvo” da je potencijalnu mogućnost pisanja aplikacija za internet operativni sistem, iako za sada nema mnogo vidljivih platformi za ovaj model. Upotreba *framework*-a za pisanje mrežnih aplikacija često smanjuje broj grešaka u programu, tako što čini kod jednostavnijim i dozvoljavajući delu tima da se koncentriše samo na *framework*. U aplikacijama koje su konstantno izložene pokušajima hakovanja, problemi u bezbednosti mogu biti prouzrokovani greškama u programu. *Framework* takođe može da promoviše najbolju praksu za tu upotrebu, kao što je *GET after POST*.

3 WEB APLIKACIJA

Ovo poglavlje predstavlja uvid u napravljenu aplikaciju za šifrovanje fajlova. Aplikacija je izradjena u *Visual Studio*-u 2010, korišćenjem jezika *C#*. Cilj rada aplikacije je da unošenjem putanje do željenog fajla i izborom metoda šifrovanja, ona napravi šifrovanu kopiju tog istog fajla i po potrebi napravi i ključ.

3.1 Šifrovanje



Slika8. Početni izgled aplikacije

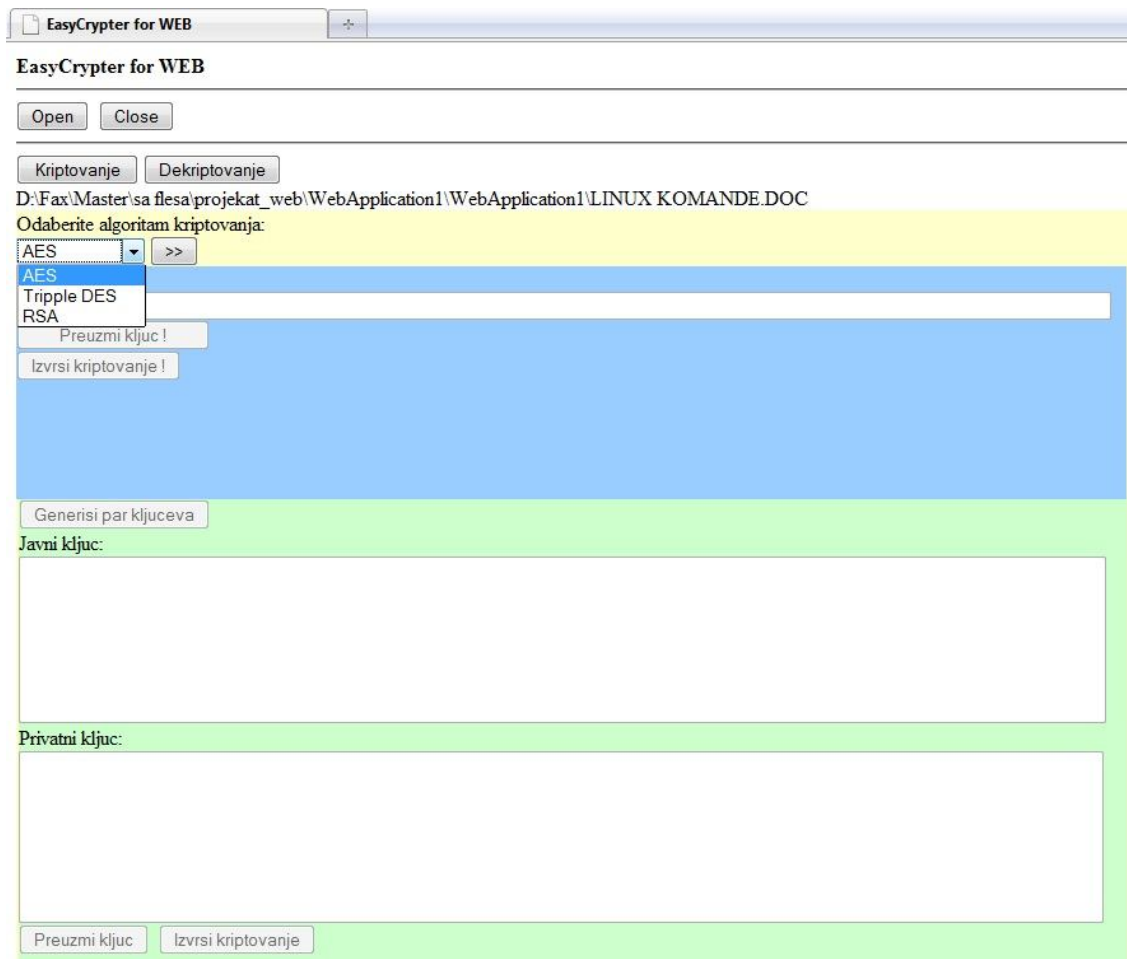
U daljem toku aplikacija je dosta intuitivna, jer su osenčene radlje ne dozvoljene. Ovde se može videti da je samo dozvoljena akcija *Open* koja dozvoljava da se dodje do fajla pomoću *Browse* opcije.



Slika9. Posle odabira fajla

Kada smo izabrali fajl, on se *upload*-uje na server, a aplikacija ispisuje put do zadatog fajla i daje nam opcije za šifrovanje ili dešifrovanje. Ostatak posla se obavlja na serveru, a klijent samo dobija vizuelnu potvrdu na svom ekranu. U sledećim koracima biće prikazano samo šifrovanje fajla, jer se dešifrovanje vrši na isti način samo koracima unazad.

Po izboru opcije „Šifrovanje“ aplikacija nam pokazuje nove opcije koje ćemo moći da koristimo, u zavisnosti od slučaja da li se koriste metode za šifrovanje simetričnim ili javnim ključem.



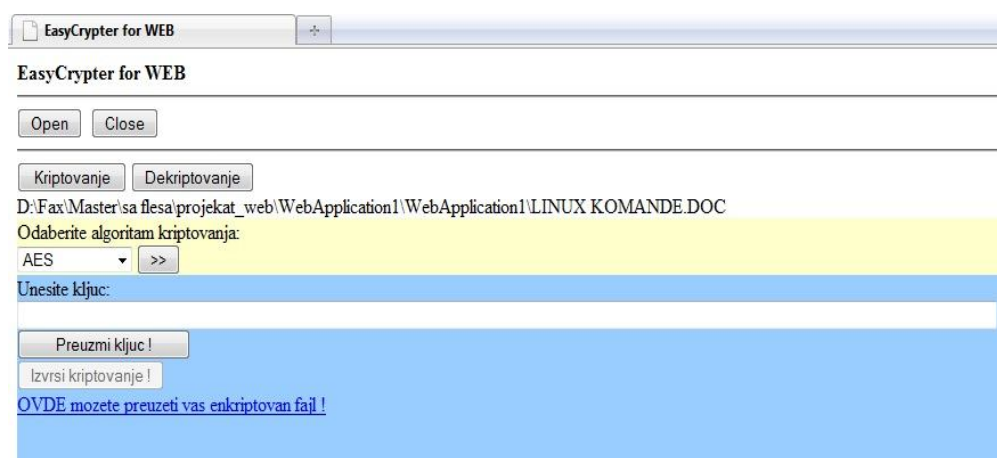
Slika10. Izbor metoda šifrovanja

U ovom segmentu vršimo izbor metode šifrovanja. Za potrebe ovog rada izabraćemo metodu *AES*. Po samom izboru metode sa simetričnim šifrovanjem biće označen samo jedan deo aplikacije u kome ćemo moći dalje da se krećemo. Izbor se vrši tako što iz padajućeg menija izaberemo neku od ponuđenih opcija i kliknemo na dugme označeno sa „>>“. U polju „Unesite ključ:“ unose se proizvoljni karakteri dužine veće od 8 znakova, zbog bolje jačine same šifre.



Slika11. Unos i preuzimanje ključa

Sledi deo u kome se vrši samo šifrovanje. Prvo se preuzima ključ, tj šifra koja kasnije služi za otvaranje šifrovanog fajla. Po preuzimanju ključa program će ispisati obaveštenje da je ključ preuzet. Posle svega ovoga ostaje samo da se izvrši šifrovanje sa opcijom „Izvršite šifrovanje!“.



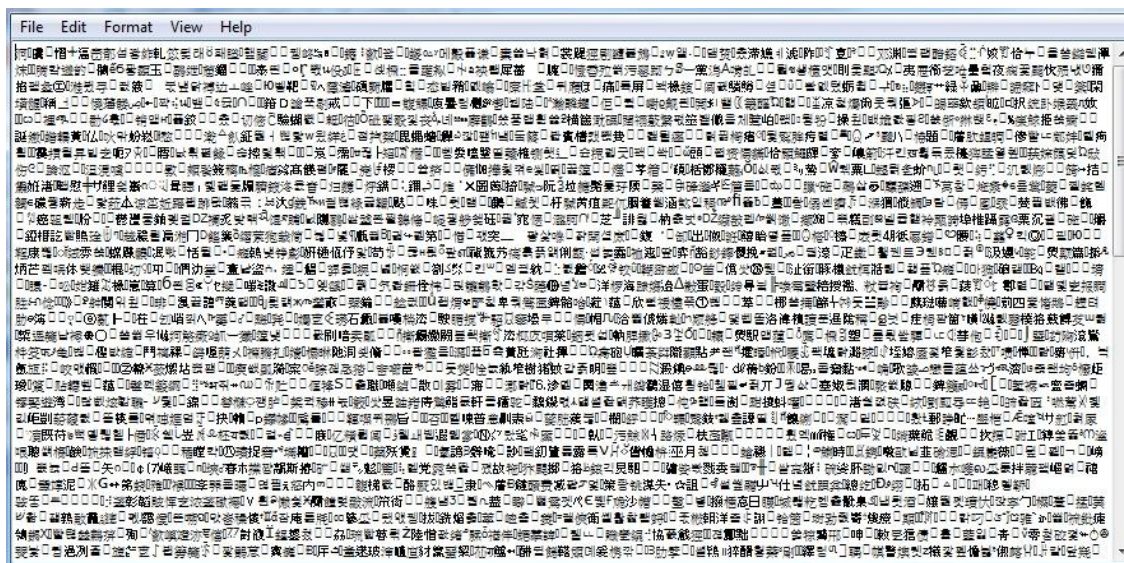
Slika12. Preuzimanje fajla

Kada se završi ceo proces program ispisuje poroku gde se može preuzeti zašifrovan fajl sa servera. Na sledeće dve slike prikazan je isti fajl, jedna slika je pre šifrovanja, a druga posle.

OSNOVNE NAREDBE

- login** – Naredba za prijavljivanje korisnika na sistem.
- logout** – Naredba logout služi za odjavljivanje korisnika sa sistema.
- pwd** – Naredba pwd prikazuje ime tekućeg direktorijuma.
- ls** – Naredba ls prikazuje spisak datoteka ili direktorijuma.
 - **-a** (prikazuje sve datoteke i direktorijume uključujući i one čija imena počinju sa ".")
 - **-R** (rekurzivno daje sadržaj svih poddirektorijuma)
 - **-l** (prikazuje spisak datoteka u proširenom obliku)
- ps** – Naredba ps prikazuje izveštaj o stanju trenutno aktivnih procesa.
 - **-A** (prikazuje SVE procese)
 - **-U** korisnik (prikazuje sve procese zadatog korisnika)
- hostname** – Naredba hostname prikazuje ili zadaje naziv sistema.
- df** – Naredba df prikazuje količinu slobodnog prostora na jednom ili više povezanih diskova ili particija.
- free** – Naredba free prikazuje izveštaj o slobodnoj i iskoriscenoj memoriji.
- passwd** – Naredba passwd menja lozinku korisnika. Ako je pokrenuta od korisnika root (administrator), može menjati lozinku bilo kog korisnika...
- tar** – Naredba tar arhivira jednu ili više datoteka ili direktorijuma.
 - **-c** (pravi novu arhivu)

Slika13. Izgld fajla pre šifrovanja



Slika 14. Izgled fajla posle šifrovanja

U slučaju da je potrebno da se šifruje korišćenjem metoda za šifrovanje sa javnim ključem koristi se metod šifrovanja RSA. Kod upotrebe ove metode postupak je malo drugačiji. Posle izabira algoritma šifrovanja RSA, postupak posle opcije „>>“ otvara se polje u kome se generišu ključevi, tj javni i privatni.

Odaberite algoritam kriptovanja:

RSA >>>

Unesite ključ:

Preuzmi ključ!

Izvrši kriptovanje!

[OVDE mozete preuzeti vas enkriptovan fajl!](#)

Generisi par ključeva

Javni ključ:

```
<RSAKeyValue><Modulus>spMV1F2ycWSvDtwcRE1atfdrMg5aCbXsFLuGweXzK6J0QqSFBae/zJaaL6gJK1LqBqT5UJVRM0woCe
/UTjziwTyzCRk71o+adVZLyW3mi3y3qnm5iB+V+9TyO2HRmUVbjvw3vOD1on8SGE+sa5UPRPNrZ8E5joQ9ocDb4/KUFM=</Modulus>
<Exponent>AQAB</Exponent></RSAKeyValue>
```

Privatni ključ:

```
<RSAKeyValue><Modulus>spMV1F2ycWSvDtwcRE1atfdrMg5aCbXsFLuGweXzK6J0QqSFBae/zJaaL6gJK1LqBqT5UJVRM0woCe
/UTjziwTyzCRk71o+adVZLyW3mi3y3qnm5iB+V+9TyO2HRmUVbjvw3vOD1on8SGE+sa5UPRPNrZ8E5joQ9ocDb4/KUFM=
</Modulus><Exponent>AQAB</Exponent>
<P>vVugYr1sHXfRTybixitjDKC4J+mumxj46mIIHsPJ0KnU2MRdJfY3YpDPKsomagh5u2vxxxy9B6ZUeLa4u6g00w==
</P><Q>7G1h7BuH563G/IX/hIh4Xmw+kIE+HtJvGAXjsVKudEal/TwSAQ66HOG0aEyoyQhJHYBYKkWD350w1rsB01JGgQ==
</Q><DP>d5QEwkvHwgpThZUzh1zKzmIwDxCWWS2061YvD31B3gS1ZVD08SE52SK1pNg3IWZpRBOBvfurZpgCXIZWHRQ==
</DP><DQ>Ui1BpZM+ohTBjtJCHNmH701tQggTW404PhsbQf22KyCr41guER8Tmrprk7OjMrd+Kcm2YwZvY/F+DTGV00YrRgQ==
</DQ><InverseQ>IvY2PbT2ht0Hdv0eNyHfh31+WYRBGGFvup1cn1X100sFrPsTBnS9g2Z1IuvNDbmZiqR5gAeFkSK+Z1TrnBZg==
```

Preuzmi ključ Izvrši kriptovanje

Slika 15. Korišćenje RSA metode

Posle preuzimanja ključa, tj. prihvatanja generisanih ključeva, kao takvih, za šifrovanje, vrši se i samo šifrovanje. Rezultat svega toga je šifrovani fajl, javni ključ i tajni ključ, koji se mogu preuzeti pomoću linkova. Ovakav postupak je urađen za potrebe ove aplikacije.

3.2 Dešifrovanje

Dekripcija se vrši na sličan način kao i enkripcija. Izabere se opcija *Open*, samo se u ovom slučaju koristi enkriptovan fajl, a ne neki drugi. Važna stvar u ovom delu je da metod dekriptovanja mora da bude isti kao i metod kriptovanja. Ako se kojim slučajem to dogodi, desiće se grška u izvođenju programa.

Kako je u prošlom delu izabran metod AES, tako ćemo i ovaj put izabrati isti metod. Posle izbora metoda šifrovanja, pristupa se unošenju ključa kojim je fajl zaključan. Ovde takodje ne sme doći do greške jer se u suprotnom dekripcija neće izvršiti.

Kada je sve urađeno kako treba, izaćiće link koji vodi do dešifrovanog fajla na serveru.

EasyCrypter for WEB

Open Close

Kriptovanje Dekriptovanje

D:\Fax\Master\sa fles\projekat_web\WebApplication1\WebApplication1\enkriptovan.enc

Odaberite algoritam dekripcije:

AES

Unesite vas kljuc za dekripciju (> od 8 znakova !):

nekarecdzaod8znakova

Slika16. Dešifrovanje fajla metodom AES

EasyCrypter for WEB

Open Close

Kriptovanje Dekriptovanje

D:\Fax\Master\sa fles\projekat_web\WebApplication1\WebApplication1\enkriptovan.enc

Odaberite algoritam dekripcije:

RSA

Uploadujte vas kljuc za dekripciju:

Browse... OK

Izvrši dekriptovanje

Slika17. Izgled dešifrovanja metodom RSA

Dešifrovanje metodom asimetričnih ključeva se vrši tako što se posle izbora fajla, i izbora enkripcije, u ovom slučaju *RSA*, unosi putanja do tajnog ključa kojim se izvršava dekripcija.

4 MOGUĆNOSTI U KREIRANJU APLIKACIJE

Što se tiče pravljenja aplikacije za mrežno šifrovanje, postoje različiti scenariji u njenom kreiranju. Mogućnosti se razlikuju u delu rada aplikacije na klijentskoj i serverskoj strani, odnosno koji deo koda će se izvršiti server, a koji deo koda će izvršiti klijentski računar. To možda ne bi bilo toliko bitno da se ne radi o programu koji je zadužen za bezbednost i sigurnost transporta fajlova. Problem bezbednosti se može ispoljiti na više načina.

4.1 Klijentska strana

Klijentska strana je dosta ne bezbedna kada se radi o ovakvom tipu aplikacije. Postoje razni problemi koji se mogu dogoditi ovde.

- Konfiguracija računara

U današnje vreme postoje više vrsta računara. Neki su brendirani, neki su sklapani od različitih delova, neki su stari, neki novi. Što se tiče brendiranih i novih računara, kod njih ne postoje problemi u konfiguraciji, jer se predpostavlja da oni ispunjavaju uslove za izvršavanje aplikacija. Problem su ove drugi, tj stariji i ne brendirani računari.

Kod ne brendiranih računara javlja se problem komponentama mašine. Iako su sve to delovi koji se ugrađuju u brendirane računare, ipak, ti delovi nisu 100% testirani i neki od njih mogu da zakažu. Recimo pinovi procesora mogu biti iskrivljeni, pa je moguća neispravnost u radu računara. Mrežni delovi ne ispunjavaju bezbednosne standarde i može doći do greške u slanju podataka. Memorija računara može neobjašnjivo da izgubi podatke u toku rada i samim tim da ugrozi ceo proces šifrovanja...

Stari računari, kao takvi su zastareli i mogu proizvesti razne probleme. Kako dugo rade, prašina se može uvući u neke delove računara i remetiti rad bitnih komponenti, kao što su procesor, memorija i razni ventilatori. Samim tim što su stari, tako su im i komponente zastarele i ne ispunjavaju uslove za rad aplikacije. Spori procesori usporavaju rad aplikacije, a može doći do „zakucavanja“, odnosno prestanka rada procesora u sred izvršenja aplikacije. Mrežne kratice mogu gubiti kontakt i tako prekidati komunikaciju klijent – server, koja je ključna za rad aplikacije.

- Bezbednost klijentskog računara

Klijentski računari su relativno nebezbedni za rad, jer su podložni raznim virusima, trojancima, crvima i kad-kad hakerskim napadima.

Virusi su uvek ozbiljna pretnja radu računara jer mogu „inficirati“ fajl koji želimo da šifrujemo i tako ga ne neamerno pošaljemo drugim korisnicima. Crvi mogu napraviti veliku štetu u mrežnoj komunikaciji i oštetiti celu mrežu, pa samim tim i rad aplikacije. Trojanci mogu dozvoliti hakerske napade i krađu baš onih podataka koje želimo da zaštitimo.

- Operativni sistem klijenta

Operativni sistemi u mnogome utiču na kompletan rad računara, pa i svake aplikacije koja se nalazi na sistemu. Greške u operativnom sistemu takođe utiču na aplikaciju koja u tom momentu radi na računaru. Na primer, ako koristimo našu aplikaciju, i u momentu kada se željeni fajl čita se računara i prebacuje na server, dodje do nekakve greške u operativnom sistemu, desiće se da fajl ne bude prebačen na server ili u još gore slučaju da se ošteti i tako oštećen prebaci na server. Kada budemo hteli da dekriptujemo fajl, rezultat neće biti kakv smo želeli, jer će se oštećeni fajl vratiti u prvobitno stanje i biće pun grešaka, a možda i skroz nesipravan.

Najgora stvar kod rada sa operativnim sistemima je pad sistema. Tada dolazi do totalnog gubljenja svih podataka na particiji sistema, a u slučaju da su bitni podaci na toj particiji, i do njihovog, možda, neprovratnog gubljenja.

4.2 Serverska strana

Globalno govoreći serveri su mnogo bezbedniji i sigurniji za rad. To je zato što im je namena da premaše „obične“ računare po njihovim karakteristikama.

- Konfiguracija servera

Po konfiguraciji svaki server je bolji od običnog računara, tj pod uslovom da su u istoj generaciji, pa čak i koju ispod. Za servere se prave posebne serije matičnih ploča, procesora pa čak se i koriste brže memorije, jer po svojoj strukturi server mora da radi više, brže i bolje od skog kućnog pandana. Ove komponente se izrađuju po drugačijim standardima, i za njih važe mnogo rigoroznije mere kontrole, da bi mogle da se nose sa postavljenim zadacima.

- Bezbednost servera

Ovde server dolazi do svoje pune nadmoćnosti. Za servere se prave posebni vidovi zaštite, jer poenta je da napadač ili pretnja uopšte ne dopre do servera, kako bi bio sprečen da napravi bilo kakvu štetu. Za tu priliku se prave posebne vrste *firewall*-ova koji su mnogo napredniji od onih koji se mogu naći kod običnih računara. To ide čak dotle da se prave

posebne hardverske komponente koje služe za odbranu servera. Pored svega toga server ima i svoje unutrašnje vidove zaštite kao što su razni sigurnosni servisi, lozinke, anti virusi, anti melveri...

Još jedna bitna stvar je da su serverski OS mnogo stabilniji od ostalih i da nisu skloni „*crash*“ –u, tj padanju sistema.

- Operativni sistemi servera

Za razliku od običnih operativnih sistema, serverski OS-mi su u osnovi pravljeni da obezbede platforme za više korisnika, čestu upotrebu, kritična izvršavanja i uglavnom koriste mrežne aplikacije. Kao takvima fokusiraju se na bezbednost, stabilnost i saradnju.

Osnovne funkcije servera su mrežna usluga kao što su, aplikacioni servisi, *e-mail* i *groupware* i terminal usluge, dok je kod običnih OS cilj korisničko okruženje.

4.3 Scenario realizacije

U aplikaciji koja je korišćena u ovom radu izveden je najbezbedniji scenario. Tokom većeg dela rada aplikacije, tj u kritičnim trenutcima aplikacija koristi seversku stranu da bi izvršavala svoje funkcije. Naime, na klijentskoj strani se obavlja deo gde se unosi putanja do željenog fajla, sve ostale funkcije obavlja server, jer je po mom mišljenju takav pristup najbezbedniji zbog činjenice da je server bolje obezbeđen, stabilniji i sigurniji od bilo koje klijentske mašine.

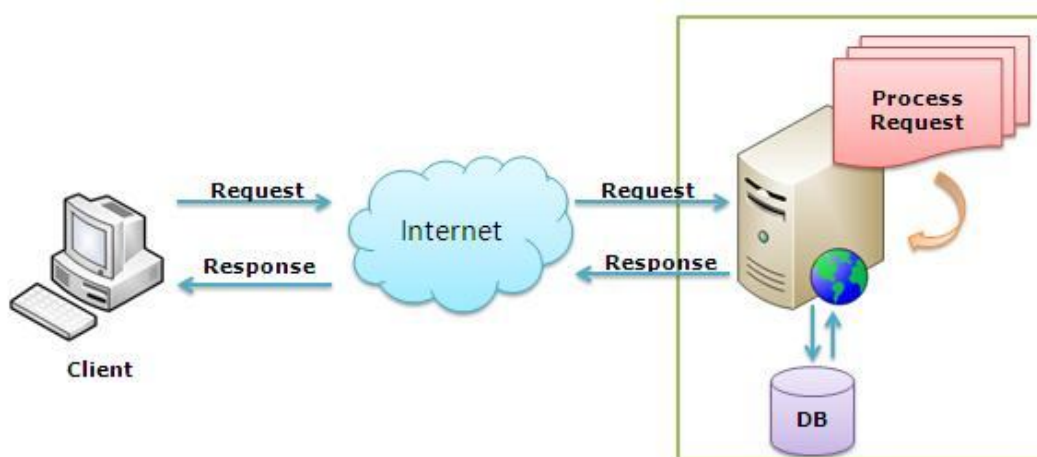
Naravno, ovo ne mora da bude obaveza u pravljenju mrežnih aplikacija, jer postoje i druge mogućnosti u opticaju, ali za potrebe ove aplikacije radjeno je na taj način.

Recimo u slučaju da se akcije rade naizmenično tj da se klijent i server stalno konsultuju, tj razmenjuju informacije, postoji više načina da dodje do greške. U tom stalnom razmenjivanju informacija može doći do greške u slanju i primanju paketa i tako se mogu oštetiti fajlovi. Može doći do pucanja veze i trajnog gubljenja fajlova. Mada, najgori mogući scenario je da se u jednom trenutku preseče veza od strane napadača i tako preusmere ili preuzmu bitni podaci.

Ako bi se aplikacija radila na klijentskoj strani može doći do napada na računar, koji je mnogo slabije obezbeđen od servera, pa bi potencijalnom napadaču bilo mnogo lakše da dodje do kritičnih podataka. Između ostalog uvek se može pronaći neki maliciozni softver koji može poremetiti rad aplikacije, ili čak zaraziti fajl koji želimo da zaštitimo i pošaljemo, pa tako nenamerno da pošaljemo trojanca ili crvaonome ko očekuje siguran i zaštićen fajl.

5.IIS

Microsoft Internet Information Services (IIS) obezbeđuje sigurnu, upravljivu platformu za razvoj, administriranje i hostovanje Web aplikacija i servisa. IIS 6.0, 7.0 i 7.5 obezbeđuju redizajniranu arhitekturu *World Wide Web Publishing* servisa, koja može da obezbedi bolje performanse, pouzdanost i bezbednost za WEB sajt. IIS je jedan od najmoćnijih servera koji služi za hostovanje i pokretanje web aplikacija. On je drugi po svojoj upotrebi u svetu, odmah iza *Apache HTTP Server*-a. IIS podržava sledeće protokole FTP, FTPS, SMTP, NNTP, HTTP/HTTPS. IIS služi za hostovanje sajtova, ali takođe može i da služi kao FTP sajt.



Slika18. Komunikacija u IIS-u

Verzije IIS-a:

- IIS 7.5 - Windows Server 2008 R2 i Windows 7
- IIS 7.0 - Windows Server 2008 i Windows Vista
- IIS 6.0 - Windows Server 2003 i Windows XP Professional x64
- IIS 5.1 - Windows XP Professional
- IIS 5.0 - Windows 2000 Server
- IIS 4.0 - Windows NT 4.0 Option Pack
- IIS 3.0 - Windows NT 4.0 SP2
- IIS 2.0 - Windows NT 4.0
- IIS 1.0 - Windows NT 3.51 (addon)

Microsoftov Visual studio ima sopstveni ASP.NET *engine* koji služi za pokretanje web aplikacija, tako da ne dolazi do problema prilikom pokretanja ASP:NET aplikacija iz Visual Studia. Ceo koncept “Web servera” stupa na snagu kada želimo da postavimo svoju

internet stranicu tako da i drugi mogu da joj pristupe. Web server služi da obezbedi odgovor na svaki zahtev koji uputi klijentska strana. Tako da kada javi više korisnika sa svojim zahtevima, server mora da opsluži njihove zahteve i da im odgovori. IIS je jedan od najmoćnijih web servera koji se koristi za hostovanje ASP.NET aplikacija. IIS ima svoj ASP.NET procese sa kojima upravlja ASP.NET zahtevima.



Slika19. Procesi u Web Serveru

Kao što se na slici vidi postupak je sledeći: prvo klijent šalje zahtev web serveru (IIS), on proverava zahtev i prosleđuje ga ASP.NET procesima. Engine ASP.NET-a će obraditi zahtev i vratiti ga klijentu istim putem. Tako da jedna od najvećih uloga IIS-a je obrada svakog pojedinačnog zahteva. Obrada stranice može da uključi i spoljne komponente. Ako kôd treba povezati sa SQL serverom radi čitanja nekih podataka, učitavaju se potrebne komponente. Ako kôd koristi neki program za crtanje dijagrama, učitava se ta komponenta. Ili, ako kôd poziva neku vašu komponentu, i ona se učitava.

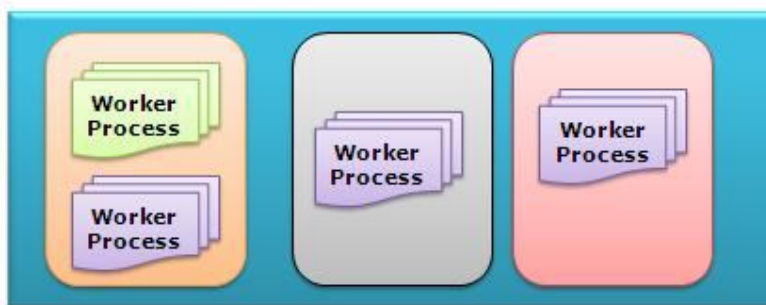
5.1 Obrada zahteva

Da bi razumeli šta se događa unutar same obrade moramo da razdvojimo 2 glavna koncepta:

- Worker Process
- Application Pool

Worker Process - radni proces (w3wp.exe) pokreće ASP.NET aplikacije u IIS-u. Ovaj proces je odgovoran za upravljanje svih zahteva i odgovora koji dolaze sa klijentske strane. Sva funkcionalnost ASP.NET-a radi pod okrivljem radnog procesa. Kada zahtev stigne so servera sa klijentske strane, radni proces je tu da bi generisao zahtev i odgovor. Jednostavno rečeno radni proces je srce ASP.NET web aplikacije koja radi pod IIS-om.

Application pool - aplikacioni bazen je skladište radnog procesa. Oni se koriste da bi razdvojili setove radnih procesa koji dele istu konfiguraciju. Aplikacioni bazen omogućava bolju sigurnost, pouzdanost i dostupnost za svaku web aplikaciju. Radni procesi služe kao procesne granice koji razdvajaju svaki aplikacioni bazen tako da kada jedan radni proces ili aplikacija imaju problem ili pripremaju naredni impuls, druge aplikacije ili radni procesi nisu obuhvaćeni sa predhodnima. Ovime se postiže da određena web aplikacija ne utiče na druge web aplikacije pošto su konfigurisane u drugi aplikacioni bazen. Aplikacioni bazen sa više radnih procesa se zove "Web bašta".

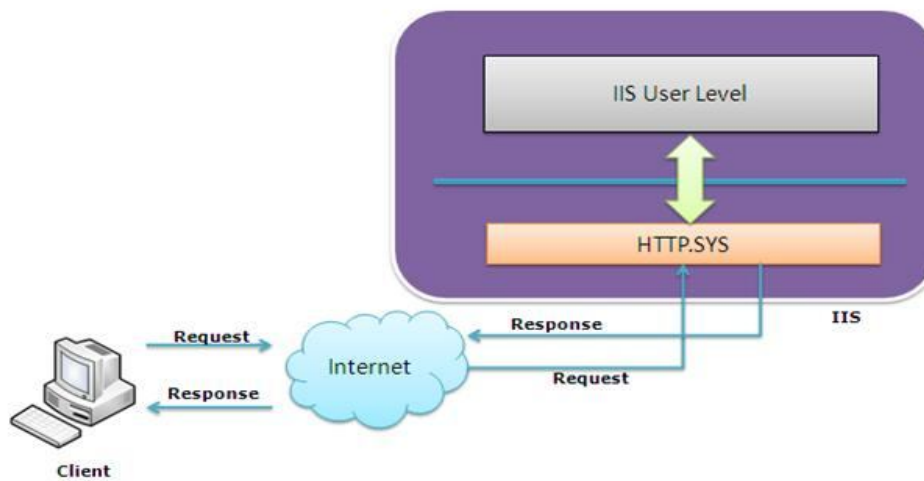


Slika20. Worker Process i Application Pool

Ako pogledamo arhitekturu IIS 6.0, možemo ga podeliti u dva sloja:

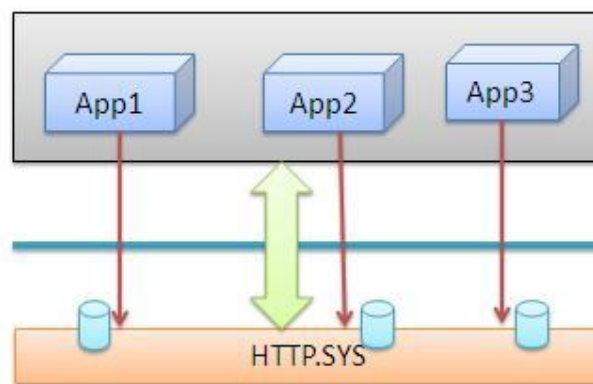
1. Kernel mod
2. Korisnički mod

Kernel mod se prvi put pojavljuje u IIS 6.0 i on sadrži HTTP.SYS . Tako da kada god zahtev dođe sa klijentske strane, on prvo dolazi do HTTP.SYS.



Slika21. Kernel mod

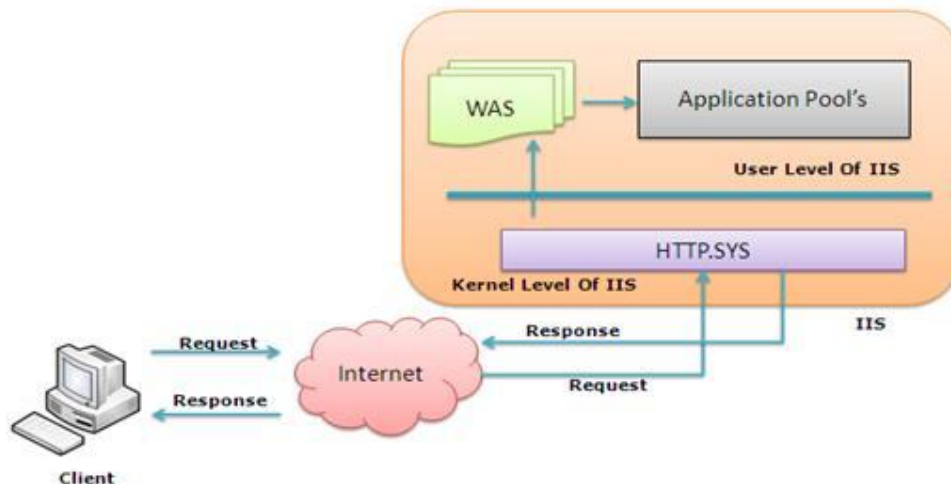
HTTP.SYS služi za prosleđivanje zahteva određenom aplikacionom bazenu. Ali, postavlja se pitanje, kako on zna kome treba da prosledi zahtev. Ovo nije neki nasumični postupak. Kada god kreiramo novi aplikacioni bazen, generise se njegov ID i on se registruje u HTTP.SYS. Tako da, kad god HTTP.SYS primi zahtev od neke web aplikacije, on proveri aplikacioni bazen i u zavisnosti od te provere on prosleđuje zahtev.



Slika22. Rad Kernel moda

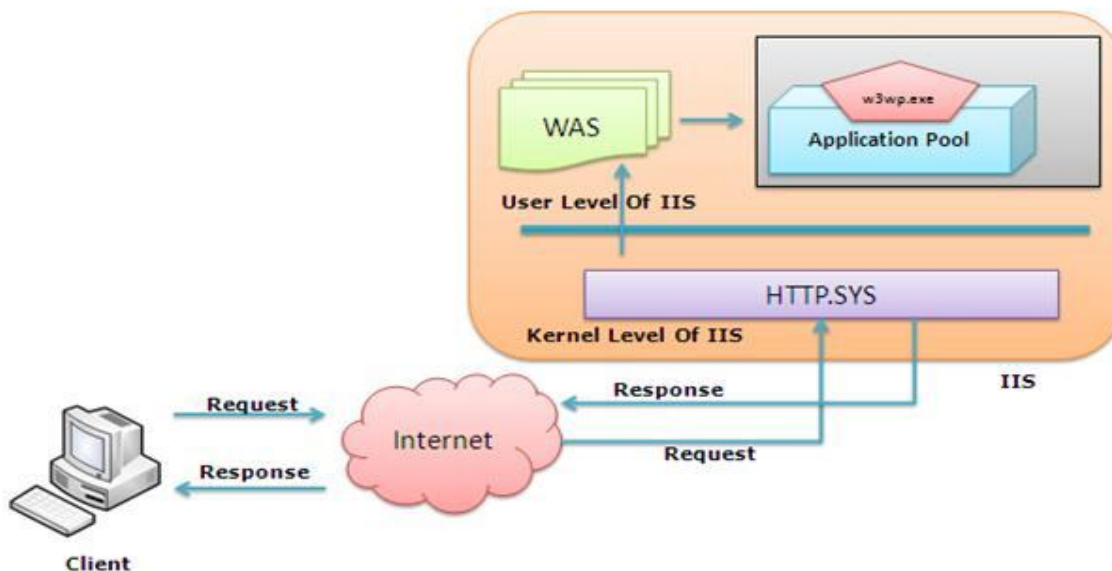
Ovo je bio prvi korak u IIS obradi zahteva.

U Korisničkom nivou IIS-a, postoji *Web Admin Services* – WAS (mrežni administratorski servisi), koji služi da preuzme zahtev od HTTP.SYS i prosledi ga odgovarajućem aplikacionom bazenu.



Slika23. WAS

Kada aplikacioni bazen primi zahtev on jednostavno prosledi zahtev radnom procesu (w3wp.exe). Radni proces potraži URL zahteva u nameri da učita ispravan ISAPI nastavak. ISAPI nastavci su načina na koji IIS upravlja zahtevima za različita sredstva. Kada se ASP.NET instalira, on automatski instalira svoj ISAPI nastavak (aspnet_isapi.dll) i doda mapiranje u IIS.



Slika24 Rad WAS-a

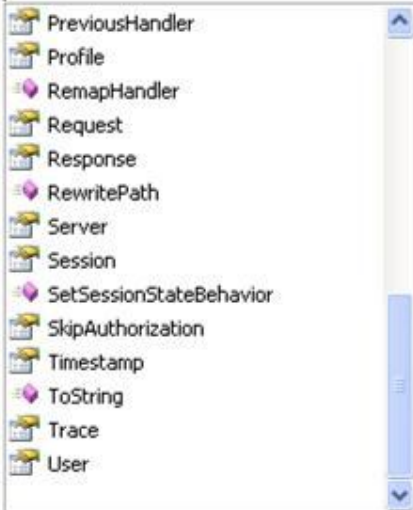
Kada radni proces učitava `aspnet_isapi.dll`, on pokreće `HttpRuntime`, koji je ulazna tačka aplikacije. `HttpRuntime` je klasa koja poziva `ProcessRequest` metodu da počne obradu.

```
HttpRuntime.ProcessRequest(|
void HttpRuntime.ProcessRequest(HttpWorkerRequest wr)
Drives all ASP.NET Web processing execution.
wr: An System.Web.HttpWorkerRequest for the current application.
```

Slika25. *HttpRuntime*

Kada je ovaj metod pozvan, kreira se nova instanca `HttpContext`, kojom se pristupa kroz `HttpContext.Current` properties. Ovaj objekat još uvek ostaje živ tokom životnog toka objekta zahteva. Upotrebom `HttpContext.Current` možemo pristupiti nekim drugim objektima, kao što su *Request*, *Response*, *Session* itd.

```
HttpContext.Current.|
```

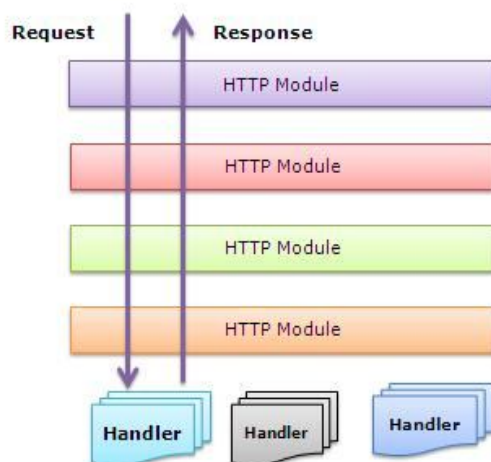


Slika23. *HttpContext*

Pošto `HttpRuntime` učitava `HttpApplication` objekat uz pomoć `HttpApplicationFactory` kalse. Svaki zahtev bi trebao da prođe kroz odgovarajući `HttpModul` da bi stigao do `HTTPHandler`-a. Ovu listu modula konfiguriše `HttpApplication`.

Sada stupa koncept nazvan “**HTTPPipeline**”. Zove se cevovod zato što sadrži set `HttpModules` (za `Web.config` i `Machine.config` nivo) koji presreću zahtev na putu za `HttpHandler`. `HttpModules` su klase koje imaju pristup dolazećim zahtevima. Možemo

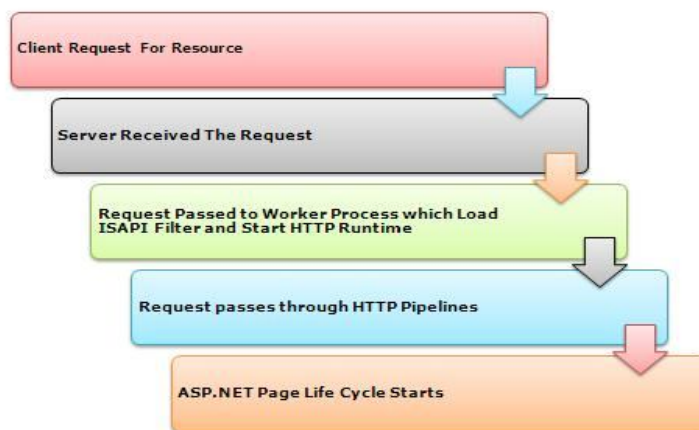
kreirati svoj HTTPModul ako postoji potreba za upravljanjem bilo čime, tokom dolazećih zahteva i odgovora.



Slika24. HTTPPipeline

HTTP hendleri su krajnje tačke u HTTP cevovodu. Svi zahtevi koji prolaze kroz HTTPModule bi trebalo da stignu do HTTPHandler-a. Onda HTTP handler generiše izlaz za traženi resurs. Tako da kada mi tražimo bilo koje *aspx* web stranice, on vraća odgovarajući HTML izlaz.

Svi zahtevi sada prolaze od HttpModules-a do odgovarajućih HTTPHandler-a. Onda pčinju metod i životni ciklus ASP.NET-a. Ovim se završava IIS obrada zahteva i počinje životni ciklus ASP.NET strane.



Slika25. Životni ciklus strane

5.2 Karakteristike

Arhitektura IIS 7 je modularna. Moduli se takođe nazivaju ekstenzije i mogu biti dodate ili uklonjene pojedinačno, tako da budu instalirani samo moduli koji su potrebni za specifične funkcije. IIS 7 ima u sebi osnovne module koji spadaju u punu instalaciju. Ovi moduli imaju odvojene karakteristike tako da server može da obrađuje zahteve. Osnovni moduli su:

- HTTP moduli – koriste se da bi izvršavali zadatke koji su specifični za HTTP, kao što su odgovori na informacije i upite koji su poslani u klijentskom zaglavlju, redirekcija zahteva i vraćanje HTTP grešaka
- Sigurnosni moduli – koriste se za zadatke koji se tiču bezbednosti, kao što su specifikacija autentifikacionih šema, izvršavanje URL autorizacije, i filtriranje zahteva
- Sadržajni moduli – koriste se za zadatke koji se tiču sadržaja, kao što su obrada zahteva za statična polja, vraćanje podrazumevane strane kada klijent ne specifikuje resursu zahtevu, i izlistavanje sadržaja direktorijuma
- Kompresioni moduli – koriste se za zadatke vezanih za kompresiju, kao što su kompresija odgovora, primena Gzip kompresiono transfernog kodiranja na odgovor, i primenjivanje pred kompresije statičkog sadržaja
- Moduli za keširanje – koriste se za zadatke vezanih za keširanje, kao što su skladištenje obrađenih informacija u memoriji servera i upotreba keširanog sadržaja u narednim zahtevima za isti resurs
- Moduli za logovanje i dijagnostiku – koriste se za zadatke vezanih za logovanje i dijagnostiku, kao što su praćenje informacija i obrađivanje statusa kao HTTP.szs za logovanje, prijavljivanje događaja, i praćenje zahteva koji se trenutno izvršavaju u radnim procesima

5.3 Bezbednost

Ranije verzije IIS-a su imale brojne ranjivosti, naročito CA-2001-13 koja je dovela do neslavnog Code Red crva. Međutim i verzija 6.0 i 7.0 trenutno nemaju problema sa nekom specifičnom ranjivošću. U verziji 6.0 Microsoft se opredelio da izmeni ponašanje pre instaliranih ISAPI hendlera, od kojih su mnogi bili krivci za ranjivosti u ranijim verzijama, tako što će smanjiti “površinu napada” IIS-a. Kao dodatak IIS 6.0 je dodata karakteristika koja se zove “*Web Service Extensions*” koja sprečava IIS da pokrene bilo kakvu aplikaciju bez izričite dozvole od administratora.

U tekućem izdanju, komponente se tretiraju kao moduli tako da samo potrebne komponente budu instalirane i time smanjena površina napada. Dodate su i mogućnosti kao što je “Filtriranje zahteva”, koji služi da odbija sumnjive URL, iz liste koje je definisao korisnik.

Po podrazumevanim kriterijumima, verzije 5.1 i niže, pokretale su sajtove pod sistemskim nalogom, tj. pod podrazumevanim Windows nalogom koji ima super-korisnička prava. Od 6.0 svi procesi koji rade sa zahtevima su premešteni pod Network Servis nalog, koji ima mnogo manje privilegije, tako da kada bi postojala ranjivost kodu, to ne bi ugrozilo ceo sistem. IIS 6.0 takođe poseduje novi kernel HTTP sa strožim HTTP parserom za zahteve i kešom za odzive kako za statičke tako i za dinamičke sadržaje.

6 SSL

SSL protokol je kreiran sa namerom da omogući praktičan, široko primenljiv, konekciono orijentisan mehanizam na aplikativnom nivou, za bezbijednu komunikaciju između klijenta i servera. SSL je u stanju da obezbijedi šifrovanje podataka, autentikaciju servera i klijenta i integritet poruke za jednu TCP/IP konekciju. Kako je SSL ugrađen u gotovo sve pretraživače i Web servere, instaliranje digitalnih sertifikata automatski uključuje i SSL mogućnosti.

6.1 SSL protokol i njegova namena

SSL (engl. *Secure Sockets Layer*) predstavlja protokol koji omogućava siguran kanal (sigurnu komunikaciju) između dva uređaja uz mogućnost identifikacije uređaja s kojim komunicirate. SSL je takođe metoda enkripcije podataka putem transportnog protokola poput TCP-a. Razvijen je od Netscape-a, 1994.god. SSL verzija 3.0 objavljena je 1996. godine, a njegov naslednik TLS predstavljen je 1999.godine od IETF-a.

Siguran kanal o kojem je ovde reč je transparentan što znači da podaci koji su poslani s jedne strane nepromenjeni stižu do druge strane. Ova karakteristika omogućava jednostavnu implementaciju SSL-a u postojeće standarde. To praktično znači da bi uz malu modifikaciju svaki protokol koji radi preko TCP-a (engl. *Transmission Control Protocol*) mogao raditi i preko SSL-a.

Kao što smo rekli SSL je razvijen u Netscape-u, vodećoj firmi web pretraživača. Netscapeu je trebao način sigurnog komuniciranja koji bi mogli iskoristiti u svojim aplikacijama. Prvenstveno se to odnosilo na Web, a tu su bili i mail i ostali servisi. Web kao najpopularniji među spomenutim servisima je prvi zahtevao pažnju. Tipična situacija koja je zahtevala primenu sigurnog kanala bila je kupovina preko Interneta u kojoj broj kreditne kartice predstavlja informaciju koju treba da štiti. Dakle, prvi zadatak koji je SSL trebao ispuniti je poverljivost. Informacije koje se šalju između klijenta i servera trebaju da ostanu poznate samo njima.

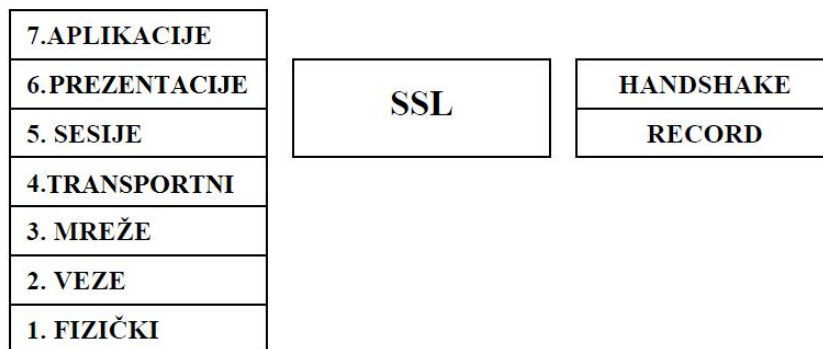
Slučaj kupovine preko Interneta otkriva i druge zadatke koje je SSL morao ispuniti. Autentifikacija – korisnik je morao biti siguran da komunicira s onim serverom s kojim želi ostvariti komunikaciju, a ne nekim trećim koji bi mogao zloupotребiti datu informaciju. Sledeća bitna karakteristika je spontanost. To znači da je korisnik mogao spontano izabrati web server s kojim želi izvršiti određenu transakciju tj. poslovati s onim s kim dosad nije imao nikakvog poslovnog iskustva. Treći cilj i zadatak je bio povezati SSL sa HTTP-om (engl. *Hyper Text Transfer Protocol*) koji je najviše korišćen na Internetu. Odatle proizlazi zahtev za transparentnošću i pouzdanošću tog protokola. Kasnije je SSL bio proširen na druge protokole.

Novija verzija SSL-a sadržavala je ranije navedene zahteve, ali protokol je uprkos tome bio poprilično složen. U sledećoj verziji ispravljeni su neki sigurnosni propusti. Poboljšana je sigurnosna razmene kriptografskih algoritama, pa je proširen i broj dostupnih algoritama.

Zašto izabrati baš SSL? Spomenimo samo nekoliko podataka. Prema Forrester Research-u u 2001. godini vrednost on-line prodaje iznosila je 41 milijardu dolara, u 2003. godini 95.7 milijardi dolara i u prethodnoj 2008. godini neverovatnih 229 milijardi američkih dolara i to predstavlja oko 10% ukupne prodaje u SAD. SSL kao što smo rekli može da štiti i druge protokole poput Usenet news-a, POP, SMTP, LDAP itd.

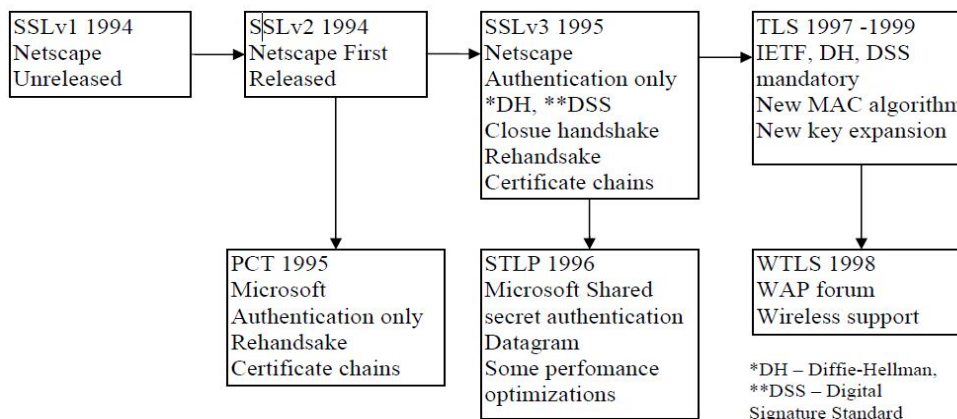
6.2 SSL i TCP/IP

Sve verzije SSL-a i TLS-a (engl. *Transport Layer Security*) kao njegovog naslednika dele zajednički cilj- uspostavljanje sigurnog kanala komunikacije preko kojega će aplikacijski programi slati sigurno svoje podatke. Stvaraoci SSL-a uzor su imali u TCP-u s kojim SSL deli sličnu semantiku.



Slika26. Položaj SSL sloja

SSL u svom radu pretpostavlja da je niži sloj pouzdan tj. da će podaci koji su poslani preko mreže sigurno stići do svog odredišta u onom obliku u kojem su i poslani. Taj zadatak prepušta drugim protokolima poput TCP-a. U praksi se ne koristi UDP protokol koji predstavlja nespojivi (connectionless) protokol i ne osigurava istu sigurnost kao TCP. Ipak u novije vreme Microsoft–ov STLP (engl. *Secure Transport Layer Protocol*) i Wireless Application Forum–ov WAP1999 su verzije SSL-a koje bi trebale korektno raditi preko UDP-a. U praksi SSL radi na dobro poznatom portu 443 (HTTPS – SSL secured HTTP).



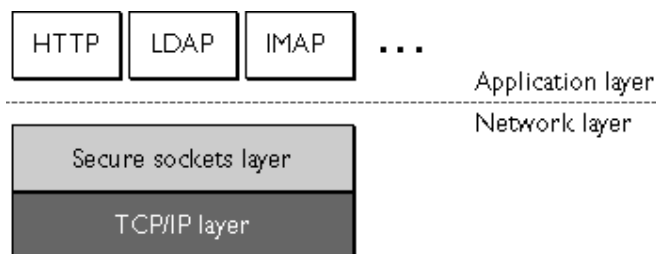
Slika27. Razvoj SSL-a

6.3 TLS

Godine 1996. Od strane IETF-a (engl. *Internet Engineering Task Force*) osnovana je grupa za izradu TLS-a, nasljednika SSL-a. Zadatak te grupe bio je uskladiti Microsoft–ov i Netscape–ov SSL. Microsoft je u međuvremenu proizveo STLP (engl. *Secure Transport Layer Protocol*) koji je bio modifikacija SSL-a i doneo je neke promene u njegovom dizajnu. STLP je imao podršku za UDP, povećao je brzinu transakcija i koristio je veći ključ pri autentifikaciji strana. Na mnogim od sastanaka dogovoreno je da će TLS imati podršku za Diffie Hellman algoritam za ključeve te DSS (engl. *Digital Signature Standard*) za autentifikaciju i Triple DES za enkripciju. Problem je bio u tome što je Netscape implementirao samo RSA algoritam za autentifikaciju i razmenu ključeva. Triple DES je predstavljao i problem što se tiče zakona u SAD–u koji su zabranjivali izvoz jake kriptografije. No odlučeno je da će IETF napraviti protokol po principima dobrog i sigurnog protokola, što je uključivalo Triple DES. U okvirima ovog seminara, kada govorim o SSL–u reč je o verziji 3.0. Iako je razvoj SSL–a stao na toj verziji, za sam TLS se često govori da je on zapravo verzija 3.1 SSL pa kad govorim o SSL–u isto se odnosi i za TLS.

6.4 Struktura SSL-a

Protokol se sastoji iz dva sloja. Donji sloj, koji se smešta na vrh nekog transportnog protokola (npr. TCP) se naziva *SSL Record Protocol* i definiše formate koji se koriste pri prenosu podataka. U njega je enkapsuliran *SSL Handshake Protocol*, koji omogućava serveru i klijentu da autentikuju jedan drugog, kao i da dogovore algoritam koji će se upotrebiti i kriptografske ključeve koji će se koristiti, pre nego što aplikativni protokol pošalje ili primi prve bajtove podataka.



Slika28. Položaj SSL u odnosu na TCP i aplikativne protokole višeg nivoa

Prilikom svake sesije generišu se ključevi za tu sesiju (session keys). Veličina ovih ključeva može biti 40 ili 128 bita. Najveći broj pretraživača podržava 40-bitne SSL sesije, dok je kod novijih omogućena upotreba 128-bitne SSL sesije.

Za realizaciju svojih funkcija SSL protokol koristi različite metode kriptovanja:

- za razmenu sesijskog ključa se koriste KEA ili RSA za ključnu listu;
- posle inicijalne razmene podataka i definisanja ključa, za šifrovanje podataka se koristi neki od simetričnih kriptografskih algoritama (3DES, DES, RC4, ...);
- za autentifikaciju drugog učesnika u komunikaciji se koristi neki od asimetričnih kriptografskih algoritama (RSA, DSS, ...);
- za proveru integriteta poruke se koristi keyed MAC(engl. Message Authentication Code), a za njegovo izračunavanje neka od hash funkcija (SHA-1, MD5, ...).

Koji algoritam će se koristiti za koju svrhu, protokol definiše prilikom inicijalne razmene podataka, na osnovu lista podržanih protokola kod oba učesnika u komunikaciji.

Glavna uloga SSL-a bila je zaštiti HTTP promet. SSL radi tako da se ostvari veza preko TCP-a, a potom se uspostavlja sigurna veza na višem sloju (SSL). Preko njega se odvija dalja komunikacija.

Najvažnija komponente sloje poruka su faze rukovanja i faze slanja podatka. U fazi rukovanja dolazi do autentifikacije strana pa onda i uspostave kriptovane komunikacije dok u fazi slanja podataka korisnički podaci putuju između aplikacija. SSL Record protocol ima zadatak da cepa i smešta te podatke.

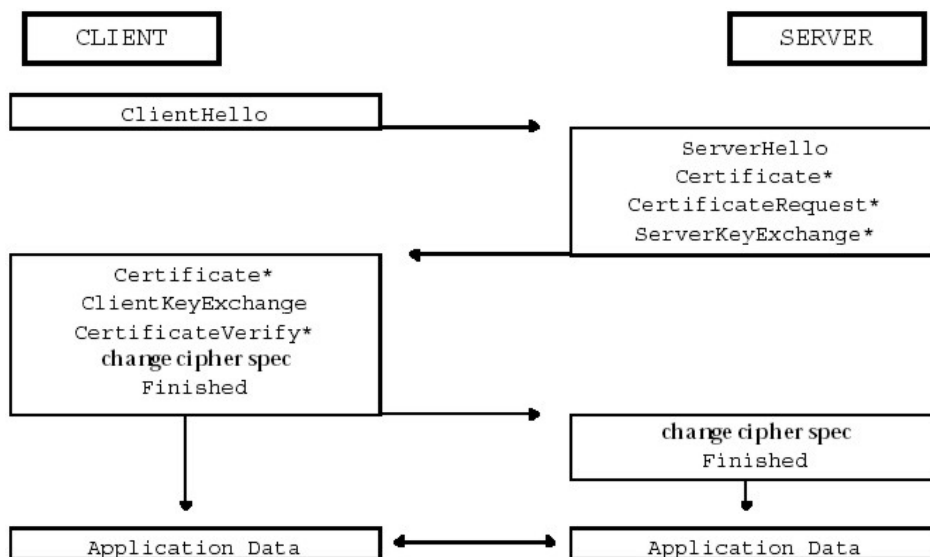
Prilikom početka komunikacije potrebno je web serveru reći da se pripremi za SSL vezu. Postupak uspostavljanja sigurne veze podeljen je u dva dela:

- rukovanje (handshake) i
- prenos podataka (data transfer).

U fazi rukovanja dolazi do autentifikacije servera i odabira kriptografskih algoritama i ključeva kojim će se štiti kanal. Faze rukovanja moraju biti završene pre nego započne

razmena podataka između aplikacija na višem sloju. Kada je faza rukovanja gotova, podaci se dele u sigurne blokove i takvi enkriptovani šalju kroz kanal.

6.4.1 Faza rukovanja (Handshake)



Slika29. Dijagram handshake protokola

Kriptografske parametre sesije proizvodi SSL Handshake Protocol. Kada SSL klijent i server započnu komunikaciju, oni se saglase o verziji protokola, selektuju kriptografske algoritme, mogu da autentikuju jedan drugoga i da razmene tajne ključeve pomoću tehnika sa javnim ključevima. Ovi procesi se odvijaju na sledeći način:

Hello request – ova poruka može biti poslata od strane servera u bilo kom momentu, ali će biti ignorisana od strane klijenta ako je protokol u toku. Ova poruka je jednostavno prikazana i predstavlja spremnost servera na uspostavljanje komunikacije sa klijentom kada ovaj bude spreman na to.

Client hello – ovo je prva poruka koju klijent šalje serveru kada želi da inicira komunikaciju. Ovakva poruka može takođe biti poslata kao odgovor na hello request ili ako iz nekih razloga klijent želi da obnovi parametre postojeće konekcije. Poruka se sastoji iz sledećih delova:

- *client_version* – najviša verzija SSL protokola koju podržava klijent.
- *random bytes* – 28 bajtova iz internog generatora slučajnih brojeva.
- *SessionID* – identifikator sesije promenljive dužine (od 0 do 32 bita). Ako je veći od nule, ukazuje na sesiju ranije uspostavljenu između istog servera i klijenta, čije parametre klijent želi da koristi.

CipherSuite list – sadrži kombinacije kriptografskih algoritama koje podržava klijent, poređani po redosledu želja klijenta. Svaka stavka u listi sadrži algoritam za razmenu ključa i CipherSpec oznaku za kryptoalgoritam. Ako u listi nema prihvatljive varijante za server, on vraća alarm handshake_failure i prekida konekciju.

CompressionMethod – kompresioni algoritmi koje podržava klijent, po redosledu koji najviše odgovara klijentu. Ako server ne podržava bilo koji od njih, sesija ne može da bude uspešna.

Server hello – ako prilikom obrade client hello poruke server nije naišao na problem i odgovorio sa handshake_failure porukom, onda odgovara sa server hello porukom. Poruka se sastoji iz sledećih delova:

- *server_version* – najviša verzija SSL protokola koju podržava server, u odnosu na ono što je ponudio klijent.
- *random bytes* – 28 bajtova iz internog generatora slučajnih brojeva servera. Mora se razlikovati od slučajnog niza koji je poslao klijent.
- *SessionID* – identitet sesije koji odgovara započetoj konekciji. Ako ovo polje u client hello poruci nije prazno, server traži konekciju sa tom identifikacijom u svojoj memoriji i, ako je nađe i ako može da je koristi, server vraća isti broj kroz ovo polje, što znači da je prihvatio zahtev klijenta. U suprotnom, ovo polje sadrži drugačiju vrednost koja označava novu identifikaciju sesije.

CipherSuite – odabrani kriptografski paket na osnovu ponude klijenta.

CompressionMethod – odabrani kompresioni algoritam na osnovu liste klijenta.

Server certificate – ako je potrebno da se izvrši autentikacija, što je generalno slučaj, server šalje svoj sertifikat odmah nakon server hello poruke. Tip sertifikata mora odgovarati selektovanom algoritmu razmene ključeva iz seta odabranih kryptoalgoritama, i obično je to X.509.v3 sertifikat (ili modifikovani X.509).

Server key exchange message – ova poruka se šalje u slučaju da server ne poseduje sertifikat, ili ako se sertifikat servera koristi samo za potpisivanje.

Certificate request – u nekim situacijama server može tražiti sertifikat od klijenta, ako je to predviđeno selektovanim paketom kryptoalgoritama.

Server hello done – ovu poruku server šalje da bi pokazao kraj server hello i ostalih pridruženih poruka. Posle slanja ove poruke server čeka na odgovor klijenta.

Client certificate – ako je server zatražio sertifikat od klijenta, on se prvo javlja ovom porukom. Ako klijent ne poseduje sertifikat, javlja se sa porukom no certificate. Na osnovu ove poruke server može nastaviti dalju konekciju ili je prekinuti, ako je sertifikacija klijenta neophodna.

Client key exchange message – izbor poruka zavisi od toga koji je algoritam sa javnim ključevima selektovan.

RSA encrypted premaster secret message – ako se koristi RSA za razmenu ključeva i autentikaciju, klijent generiše 48-bajtni pre-master ključ, kriptuje ga javnim ključem iz sertifikata servera ili privremenim RSA ključem iz server key exchange poruke, i šalje rezultat kao encrypted premaster secret poruku.

Fortezza key exchange message – Pod Fortezza DMS, klijent izvlači TEK (engl. *Token Encryption Key*) korišćenjem Fortezza's KEA (engl. *Key Exchange Algorithm*). Klijentove KEA kalkulacije koriste javni ključ iz sertifikata servera zajedno sa privatnim parametrima iz oznake klijenta. Klijent šalje javne parametre potrebne serveru da bi, uz korišćenje svojih privatnih parametara, generisao TEK. Klijent generiše ključ za sesiju, pakuje ga korišćenjem TEK i šalje rezultat serveru. Zatim klijent generiše IV korišćenjem TEK i takođe ga šalje. Na kraju klijent generiše slučajni 48-bajtni premaster secret, kriptuje ga korišćenjem TEK i šalje rezultat.

Client Diffie-Hellman public value – ova struktura sprovodi klijentovu Diffie-Hellman javnu vrednost, ako već nije uključena u klijentov sertifikat (u tom slučaju se ne šalje opet).

Certificate verify – poruka koja se koristi da potvrdi eksplicitnu verifikaciju klijentovog sertifikata, a šalje se samo posle klijentovog sertifikata koji ima mogućnost potpisivanja (svi sertifikati sem onih koji sadrže fiksne Diffie-Hellman parametre).

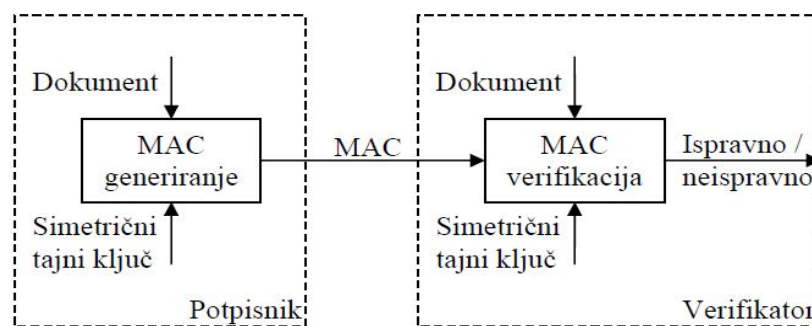
Finished – poruka koja se šalje neposredno posle change cipher specs poruke radi verifikacije da su procesi razmene ključeva i autentikacije protekli uspešno. Ova poruka je prva koja je zaštićena, a posle nje učesnici u komunikaciji mogu početi da šalju poverljive podatke.

Svrha faze rukovanja (handshake) je trostruka. Prvo što klijent i server moraju uraditi je dogovor oko algoritama koji će se koristiti u komunikaciji. SSL podržava više algoritama. Potrebno je naravno da klijent i server poseduju iste. Drugi zadatak je odabir ključeva koji će se koristiti u komunikaciji. Treći zadatak je autentifikacija klijenta (servera). Ceo proces izgleda po koracima ovako:

- Klijent šalje serveru listu algoritama koje podržava, zajedno sa slučajnim (random) brojem koji će se koristiti u procesu generisanja ključeva.
- Server bira šifru sa liste i šalje je nazad sa sertifikatom koji služe za njegovu autentifikaciju, a sadrži njegov javni ključ. Takođe, šalje i slučajni broj koji služi za generisanje ključeva
- Klijent proverava serverov sertifikat i izvlači javni ključ. Zatim, generiše slučajni string tj. ključ koji se zove `pre_master_secret` i enkriptuje ga sa serverovim javnim ključem.
- Klijent i server nezavisno rade enkripciju i MAC (engl. *Message Authentication Code*) ključeve iz `pre_master_secret` i slučajno generisanih brojeva.
- Klijent šalje MAC iz svih poruka poslatih serveru.
- Server šalje MAC iz svih poruka poslatih klijentu.

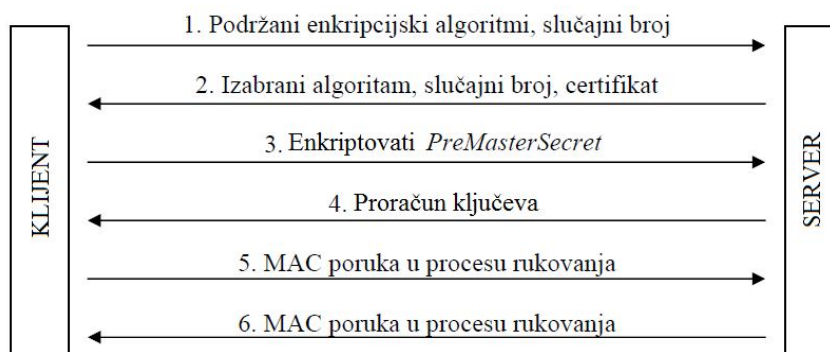
Dakle, prva dva koraka omogućavaju razmenu mogućih algoritama i razmenu ključeva. Klijent šalje listu algoritama koje podržava, a server odabira jednog od njih. U koracima dva i tri dolazi do razmene ključeva, *pre_master_secret* je generisan od klijenta, enkriptovan javnim ključem od servera i poslat njemu. Od trećeg koraka zavisi sigurnost cele transakcije. U trećem koraku klijent javnim ključem servera enkriptuje ključ koji će deliti za komunikaciju. Server svojim tajnim ključem dekriptuje zajednički ključ. Ostatak procesa rukovanja je samo osiguranje da se razmena odvija sigurno. U koracima pet i šest dolazi do razmene MAC poruka koje služe da bi se proverila verodostojnost svih poruka u procesu rukovanja, jer se MAC računa tako da se hash funkcijom deluje na telo (body) poruke uz neke dodatne podatke kao broj sekvence, dužina poruke, itd. uz *MAC_Write_Secret*. Za hash funkciju bitno je naglasiti da je jednosmerna.

MAC_Write_Secret je izveden iz *MasterSecret* poruke i za napadače je gotovo nemoguće kopirati MAC za neku poruku. MAC, dakle pruža sigurnost da poruka u komunikacijskom kanalu nije promenjena. Funkcije koje se koriste za izračunavanje MAC-a su MD5 i SHA1. Slika 5 pokazuje to grafički:



Slika30. Proračun MAC-a

Gornji postupak započinjanja SSL sesije, grafički prikazuje sledeća slika:



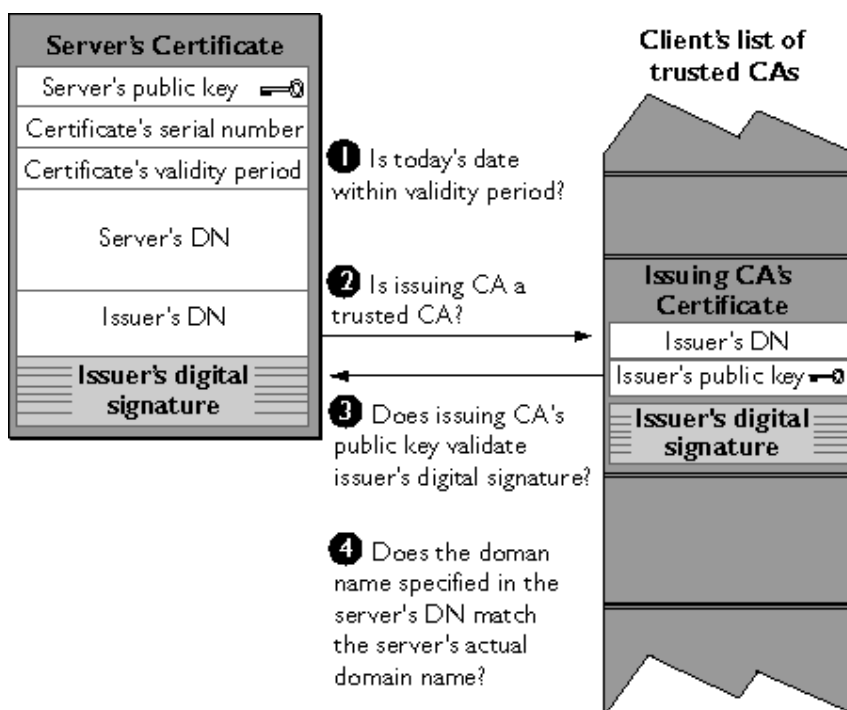
Slika31. Postupak započinjanja SSL sesije

U stvarnosti proces razmene poruka je malo detaljniji i svaki od gore navedenih koraka se sastoji on jedne ili više poruka rukovanja (handsake messages). Korak jedan odgovara ClientHello poruci, korak dva je serija poruka koja se sastoji od ServerHello poruke, Certificate poruke u kojoj se šalje certifikat i ServerHelloDone kojom se završava taj korak. Treći korak odgovara poruci ClientKeyExchange. Koraci pet i šest odgovaraju Finished porukama. CipherSuite poruka donosi obaveštenje serveru koje enkripcijske algoritme klijent podržava, a ista poruka sa serverove strane odabere jedan od tih „paketa“.

SSL podržava veliki niz raznih algoritama i nivoa sigurnosti, uključujući i algoritme koji pružaju minimalni nivo sigurnosti ili čak nikakvu. Spomenimo da se 40 bitni ključ lako probija, pa treba voditi računa o izboru dužine ključa. Preporuka standarda je da se ne koristi anoniman Diffie Hellman standard za digitalne potpise, a poznato je da se RSA 512 bitni ključ koji štiti certifikat pokazao nedovoljno sigurnim.

6.4.2 Autentifikacija servera

Autentifikaciju servera klijent vrši na osnovu sertifikata, digitalno potpisanog tajnim ključem sertifikacionog tela, koji mu server pošalje u toku handshake protokola.



Slika32. Autentifikacija servera

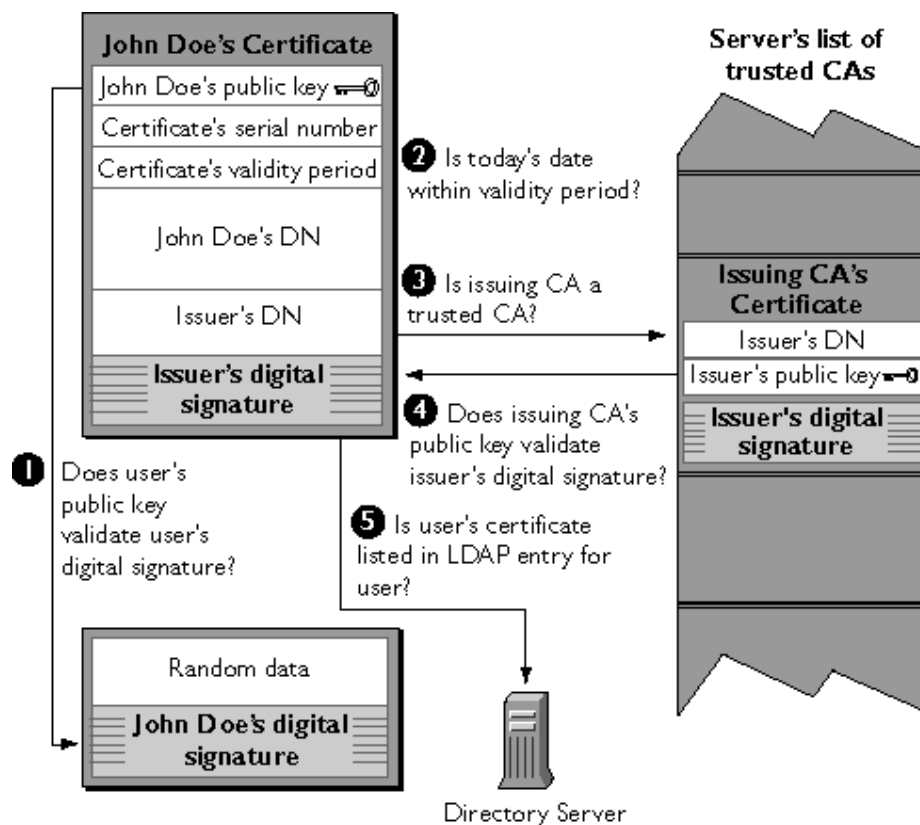
Autentifikacija servera ima četiri koraka:

- Proverava se da li se tekući datum i vreme nalaze u periodu validnosti u sertifikatu servera.
- Proverava se da li se sertifikaciono telo koje je potpisalo sertifikat servera, ili neko u hijerarhijskom lancu iznad njega, nalazi na spisku klijentovih sertifikacionih tela od poverenja.
- Klijent koristi javni ključ iz sertifikata sertifikacionog tela za proveru validnosti digitalnog potpisa sertifikata servera. Sertifikat sertifikacionog tela se nalazi u listi klijentovih sertifikacionih tela od poverenja.
- Proverava se da li se server zaista nalazi na adresi koja je u imenu domena DN (engl. Domain Name) specificirana u sertifikatu servera.

U slučaju da je u bilo kom koraku konstatovana nepravilnost, dalje provere i konekcija sa serverom se prekidaju, a korisniku klijentske aplikacije se šalje obaveštenje o nemogućnosti autentifikacije servera. Ako su sve provere pozitivne, klijent smatra da je pouzdano utvrđen identitet servera i nastavlja sa izvršavanjem protokola.

6.4.3 Autentifikacija klijenta

Server može biti konfigurisan da zahteva autentifikaciju klijenta, ili kriptografsku validaciju identiteta klijenta. U tom slučaju, u toku handshake protokola, nakon slanja ServerHello poruke i svog sertifikata, server šalje zahtev za sertifikatom klijenta. Na ovaj zahtev, klijent šalje serveru svoj sertifikat i, zasebno, još jedan blok podataka. Ovaj blok podataka klijent generiše tako što nad slučajnim bajtovima, generisanim i poslatim serveru u toku Client hello poruke, primeni hash funkciju, a zatim dobijenu vrednost kriptuje svojim privatnim ključem.



Slika33. Autentifikacija klijenta

Autentifikacija klijenta se odvija u pet koraka.

- Klijent proverava da li digitalni potpis servera može biti verifikovan javnim ključem koji se nalazi u sertifikatu.
- Proverava se da li se tekući datum i vreme nalaze u periodu validnosti u sertifikatu klijenta.
- Proverava se da li se sertifikaciono telo koje je potpisalo sertifikat klijenta, ili neko u hijerarhijskom lancu iznad njega, nalazi na spisku serverovih sertifikacionih tela od poverenja.
- Server koristi javni ključ iz sertifikata sertifikacionog tela za proveru validnosti digitalnog potpisa sertifikata klijenta. Sertifikat sertifikacionog tela se nalazi u listi serverovih sertifikacionih tela od poverenja.
- Ovaj korak je jedan od načina da sistem administrator opozove sertifikat korisnika, čak i ako to prolazi testove u svim ostalim koracima. *Netscape* sertifikat servera može automatski ukloniti odbijen sertifikat od ulaska korisnika u LDAP direktorijum. Svi serveri koji su postavljeni da obavljaju ovaj korak će tada odbijati da potvrde sertifikat ili da uspostave vezu. Ako je korisnik sertifikata u direktorijumu identičan sa sertifikatom korisnika koji je predstavljen u SSL handshake, veza se uspostavlja.

6.4.4 Faza prenosa podataka (data transfer)

Tek kada se završi faza rukovanja, može se započeti faza razmene podataka. U ovoj fazi server i klijent razmenjuju enkriptovane podatke. Svakoj poruci dodat je MAC sa kojim se proverava ispravnost poruke. Ključevi koji se koriste pri enkripciji poruke i MAC-a dobijeni su u koraku razmene ključeva. Protokol koji brine o razmeni je Record Protocol.

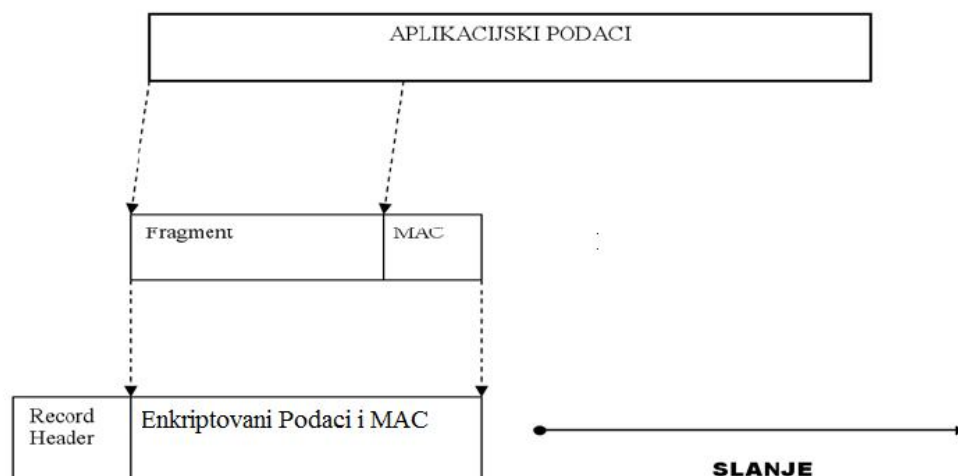
6.4.5 SSL Record protokol

SSL Record protokol prihvata podatke od protokola viših nivoa i vrši njihovu fragmentaciju na SSLPlaintext zapise od 214 bajtova ili manje. Svi zapisi se zatim kompresuju korišćenjem kompresionog algoritma koji je definisan za tekuću sesiju u SSL Handshake protokolu. Kompresioni algoritam pretvara SSLPlaintext u SSLCompressed strukturu. Prilikom kompresije ne sme doći do gubljenja podataka, niti do povećanja veličine zapisa za više od 1024 bajta. Ako se prilikom dekompresije dobije zapis čija veličina prelazi 214 bajtova, generiše se poruka fatalne greške (fataly error).

Svi zapisi se štite šifrovanjem i korišćenjem MAC algoritama definisanim u aktivnom CipherSpec. U toku SSL Handshake protokola, dve strane u komunikaciji razmene tajne poruke, koje se koriste za šifrovanje zapisa i izračunavanje MAC kodova. Šifrovanje i MAC funkcije pretvaraju SSLCompressed strukturu u SSLCiphertext, pri čemu se MAC vrednosti izračunavaju pre kriptovanja.

Kada koristimo SSL želimo imati sigurnu, verodostojnu i autentifikovanu komunikaciju. Zadatak procesa rukovanja je opisan gore, a sam proces slanja stvarnih aplikacijskih podataka zadatak je SSL Record protokola.

SSL Record protokol radi tako da segmentira podatke u fragmente koji se svaki zasebno enkriptuju i šalju. Na strani primaoca svaki takav fragment se posebno prima i dekriptuje. Pre nego što se pošalje, fragment se mora zaštititi od napada. Da bi se osigurala autentifikacija poruka, uz fragment se šalje i njegov MAC koji se mora verifikovati na strani primaoca da bi bio primljen. MAC je dodat fragmentu i takav paket se šalje enkriptovan. Tom paketu je dodat i Record header. Postupak je pojašnjen na slici 5.



Slika34. Funkcija SSL Record protokola

Ovakav postupak slanja podataka često se naziva „enkapsulacija“ paketa. On omogućava višim protokolima da nesmetano rade svoj posao. Na ovom mestu možemo ponoviti zahteve koje takav protokol treba ostvariti. Uz opšte zahteve koji se postavljaju pred jedan siguran sistem za SSL Record Protocol moramo navesti :

- kriptografsku sigurnost – temeljni zadatak koji se postavlja na sistem,
- interoperabilnost – nezavisni programeri bi trebali razvijati SSL aplikacije koje će međusobno dobro raditi,
- proširivost – mogućnost nadogradnje SSL-a (sprečava razvoj novih protokola – „open source“)
- efikasnost – racionalna upotreba dostupnih računarskih resursa.

Record Header sadrži informacije o tipu podatka koji se nalazi u paketu, njegovu dužinu i SSL verziju. To je potrebno kako bi se komunikacija mogla nesmetano odvijati.

Što se tiče tipa podataka, SSL podržava četiri, i to su:

- podaci aplikacija (application_data),
- protokol za vanredne događaje (alert),
- protokol za rukovanje (handshake) i
- protokol za promenu načina šifriranja (change_cipher_spec).

Application_data su podaci aplikacija koji koriste SSL, a ostala tri protokola služe za upravljanje sa vezom. Tako se na primer alert koristi kod signalizacije grešaka, handshake kod tzv. rukovanja, a change_cipher_spec za promenu enkripcionih ključeva kojima se štiti veza.

Dakle, uz Record header jedan „SSL paket“ (ili zapis) sadrži enkriptovane podatke i MAC. MAC se izračunava pomoću sledeće formule:

$$\text{hash}(\text{MAC_write_secret} + \text{pad2} + \text{hash}(\text{MAC_write_secret} + \text{pad1} + \text{seq_number} + \text{length} + \text{content}))$$

MAC_write_secret predstavlja zajednički glavni tajni ključ, pad1 niz ASCII znakova kojima se povećava sigurnost MAC-a, sadržaj je proizvoljan. Pad2 predstavlja drugi niz znakova, a seq_number predstavlja sekvencijski broj poruke.

Protokol za vanredne događaje sastoji se od dva dela:

- nivo
- opis

Zapisani su kao 8 bitni brojevi. SSLv3.0 definiše 13 različitih vanrednih događaja. Prikazani su u tabeli 2. Postoje dva nivoa alert protokola. Jedan je warning koji upozorava na grešku, ali ne prekida vezu, dok fatal trenutno prekida vezu.

Nivo	Ime	Opis
1	Upozorenje (Warning)	SSL upozorenje da problem nije kritičan
2	Kritično (Fatal)	SSL vanredni kritični događaji trenutno prekidaju trenutnu vezu..

Tabela 1: Nivoi Alert protokola

Kada je otkrivena greška, strana koja je otkrila grešku šalje poruku drugoj strani. U trenutku slanja i primanja takve poruke, obadve strane zatvaraju sesiju (vezu). Server i klijent u tom slučaju imaju obavezu da zaborave bilo kakve podatke o sesiji, identifikatore sesije, ključeve i ostale tajne koje su vezane uz sesiju.

Detaljniji prikaz poruka koji je definisao SSL standard za alert protokol možemo videti u tabeli 2 (13 različitih opisa vanrednih događaja). Takvi događaji, tj. poruke se šifruju i komprimuju.

roj	B	Ime	Opis
	0	close_notify	Indicira da pošaljalac više ne namerava slati podatke. Ukoliko je ovaj opis poslan sa upozorenjem, veza može biti obnovljena; ukoliko je pak poslan uz kritičan nivo događaja, veza ne može biti obnovljena
0	1	unexpected_message	Primljena je neočekivana poruka. Ovaj opis se ne bi smeo dogoditi; indicira grešku u jednoj od SSL implementacija (klijentskoj ili serverskoj). Kritičan
0	2	bad_record_mac	Pošaljalac je primio zapis sa neispravnim autentikacijskim kodom poruke (MAC). Kritičan
0	3	decompression_failure	Informacija u zapisu se ne može ispravno dekompresovati. Kritičan
0	4	handshake_failure	Indicira da pošaljalac nije u mogućnosti prihvatiti skup sigurnosnih parametara (npr. pošaljalac nije zadovoljan s primaočevim algoritmima za šifrovanje i njihovom snagom). Kritičan
1	4	no_certificate	Događa se kao odgovor na zahtev za uverenjem ukoliko ne postoji odgovarajuće uverenje
2	4	bad_certificate	Događa se ukoliko je zahtev za uverenjem neuspešan (uverenje je neispravno ili je neispravan digitalni potpis)
3	4	unsupported_certificate	Događa se ukoliko pošaljalac ne podržava određeni tip uverenja
4	4	certificate_revoked	Događa se ukoliko pošaljalac primi uverenje koje je već ranije povučeno
5	4	certificate_expired	Događa se ukoliko pošaljalac primi uverenje koje je isteklo
6	4	certificate_unknown	Događa se ukoliko se prilikom obrade uverenja dogodi greška
7	4	illegal_parameter	Događa se ukoliko pošaljalac ustanovi da je neka vrednost u protokolu za rukovanje nedozvoljene vrednosti. Kritičan

Tabela 2: Vanredni događaji SSL v3.0

6.4.6 Pasivni napadi na SSL Record Protocol

Pasivni napadi se svode na prisluškivanje i analizu saobraćaja. Pre nego što krene prenos kriptovanih podataka, učesnici u komunikaciji razmene dosta otvorenog teksta koji može prihvatiti neko ko prisluškuje komunikacije jednog od učesnika. Međutim, kako se koriste jaki algoritmi za šifrovanje, a takođe je i razmena ključeva dobro rešena, podaci prikupljeni na ovaj način ne mogu biti iskorišćeni na značajan način.

Analiza saobraćaja se bavi nekriptovanim poljima paketa, atributima paketa, dužinama pojedinih polja itd. Na ovaj način se može doći do nekih značajnih podataka, kao što su identitet Web servera, dužina URL-a i dužina html podataka koje je Web server vratio. Ovi podaci se ne mogu izvući u slučajevima kada se za šifrovanje koriste blok algoritmi, jer je tada različita dužina otvorenog teksta i šifara.

6.4.7 Aktivni napadi na SSL Record Protocol

Za razliku od IPsec, SSL se pokazao prilično otporan na aktivne napade. Jedan od najčešćih tipova napada na koje se *IPsec* pokazao ranjiv je cut-and-paste napad. Kod njega, iz paketa koji sadrži osjetljive podatke iseče se šifrat i zalepi na šifrat nekog drugog paketa, tako da primalac može da dođe u situaciju da, po prijemu i dekripciji, deo podataka pošalje nazad na mrežu u otvorenom obliku. To se, na primer, može desiti ako primalac neki segment paketa protumači kao URL adresu i pokuša da na nju pošalje neki paket. SSL 3.0 prevazilazi ovaj problem tako što koristi nezavisne ključeve za svaku konekciju, kao i strogu autentikaciju za sve kriptovane pakete.

Druga vrsta napada koja pogađa *IPsec* je short-block. Ova vrsta napada se koristi kada poslednji blok neke poruke sadrži jedan bajt otvorenog teksta, a ostatak bloka je popunjen nasumice. Napad se sastoji u tome da se pogađa nepoznati bajt otvorenog teksta tako što se poslednji blok šifrata zameni sa nekim drugim blokom šifrata za koji znamo otvoreni tekst. Ako je pretpostavljeni blok pogrešan, paket će na prijemu biti odbačen; ako je dobar, primalac će poslati ACK pošiljaocu, na osnovu čega napadač može doći u poziciju da rekonstruiše otvoreni tekst. SSL je zaštićen od ovog tipa napada jer on praktično ne generiše kratke blokove.

6.4.8 Napadi na izbor kriptografskog paketa

Veoma zanimljiva slabost SSL 2.0 protokola je omogućila ovu vrstu napada koji je pogađao samo Amerikance: napadač je mogao, editovanjem liste podržanih kriptografskih paketa, da neopaženo isforsira korišćenje slabijih algoritama koji se isporučuju uz verzije protokola namenjene izvozu. Verzija 3.0 ispravlja ovu slabost tako što zahteva autentifikaciju svih poruka u SSL Handshake protokolu.

6.4.9 Napadi prilikom razmene ključeva

Prilikom razmene ključeva kod SSL 3.0 postoje određene slabosti koje mogu biti iskorišćene za preduzimanje napada. Na primer, server može poslati, u okviru server key exchange poruke, parametre javnih ključeva, potpisane sertifikatom servera. Međutim, ovaj potpis ne štiti polje koje određuje koji algoritam za razmenu ključeva će biti korišćen. Modifikacijom ovog polja, klijent može biti naveden da umesto seta *Diffie-Hellman* parametara prihvati informaciju da je server potpisao set kratkotrajnih RSA parametara.

Praćenjem dalje komunikacije između servera i klijenta, napadač matematički može izračunati `pre_master_secret`, a sa tim i sve ostale ključeve i doći do sadržaja poruke.

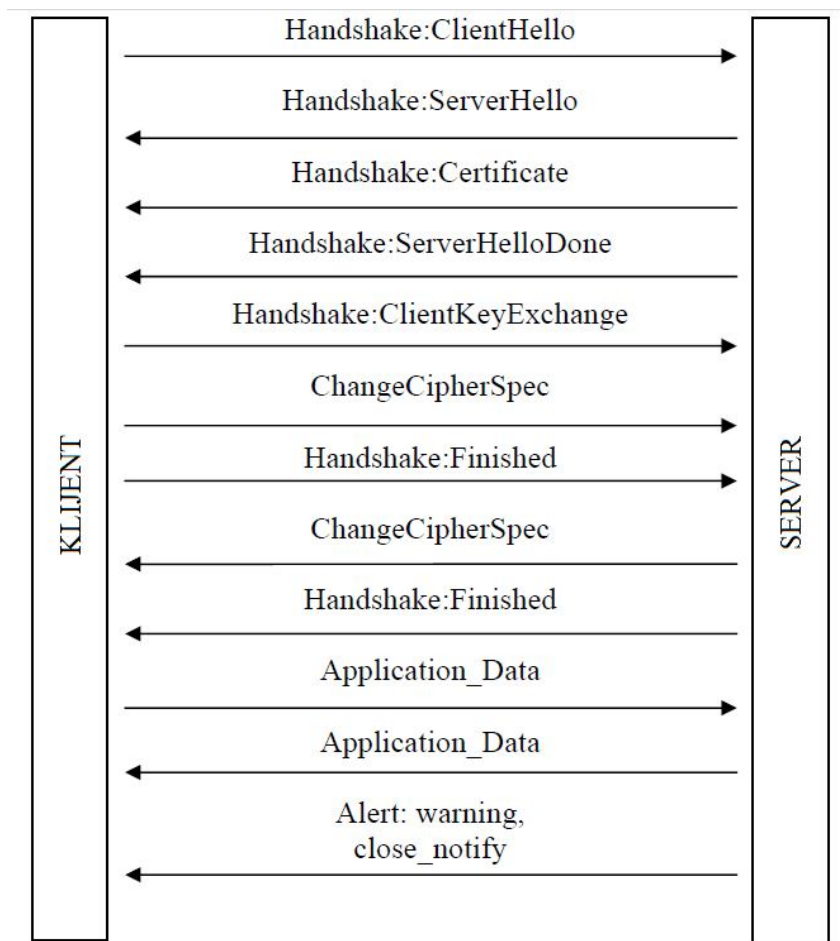
Još jedna slabost prilikom razmene ključeva kod SSL 3.0 primećena je kod anonimnih konekcija. Ako server bude doveden u situaciju da prihvati anonimnu razmenu ključeva, onda server ne traži signaturu kriptografskih parametara, što onom ko je takvu sesiju inicirao može omogućiti praćenje svih poslednjih anonimnih razmena ključeva.

6.4.10 Napadi na verziju protokola

Već je rečeno da je u verziji 2.0 bilo dosta slabosti koje su ispravljene verzijom 3.0. Kako verzija 3.0 poštuje vertikalnu kompatibilnost, tj. može da radi sa aplikacijama koje koriste verziju 2.0, jednostavna podloga za dalje napade se stvara tako što se drugi učesnik u komunikaciji ubedi da komunicira sa verzijom 2.0. Napadaču je dovoljno da kreira client hello poruku na taj način da izgleda kao poruka iz verzije 2.0 i da počne da koristi sve slabosti stare verzije.

6.4.11 SSL poruke

Detaljniji prikaz jedne SSL konekcije izgledao ovako:



Slika35. Detaljni prikaz jedne SSL konekcije

ClientHello poruka sastoji se od:

- *ProtocolVersion client_version* poruke u kojoj je sadržana najnovija verzija SSL-a koju klijent podržava,
- *Random* - poruka koje su slučajna struktura za povećanje sigurnosti, sastoje se od 32-bitne vremenske funkcije i 28 okteta generisanih od strane sigurnog generatora slučajnih brojeva i SessionID session_id koji predstavlja identifikaciju ID. Polje je prazno za novu vezu, a ukoliko nije, to znači da klijent pokušava obnoviti neku prethodnu vezu.
- *CipherSuite cipher_suites* je spisak načina šifrovanja koje klijent podržava i na kraju

- *CompressionMethod* spisak načina kompresije koje klijent podržava. Kod kompresije je bitno naglasiti da zbog autorskih prava u sve verzije SSL-a nije uključena kompresija. Uz nju dolazi i 32 byte-a slučajnih podataka (*Client.random*)
- *ServerHello* poruka sadrži sledeće informacije:
- *ProtocolVersion client_version* - SSL verzija koju koristi klijent,
- *Random* - slučajna struktura,
- *SessionID session_id* – identifikacija veze (ovo polje nikad nije prazno). Ukoliko se podudara sa *session_id* poljem iz *ClientHello* poruke to znači da će prethodna SSL veza biti obnovljena. Inače *session_id* sadrži ID nove veze.
- *CipherSuite* - način šifrovanja odabran od servera za tekuću vezu.
- *CompressionMethod* - način kompresije odabran od servera.

Certificate: ukoliko server poseduje javni sertifikat on ga prosleđuje klijentu. Klijent takođe može koristiti ovu poruku da pošalje svoj sertifikat serveru ukoliko mu to server zatraži. Kada se to dogodi klijent šalje *CertificateVerify* poruku u kojoj enkriptuje neku zadatu poruku i šalje je serveru da bi ovaj mogao proveriti da li klijent poseduje svoj privatni ključ. Ova se poruka sastoji od:

- imena izdavaoca sertifikata,
- imena entiteta za koji je taj sertifikat izdat,
- javnog ključa tog entiteta,
- vremenske komponente koja služi kao neki „rok trajanja“ sertifikata. Iako taj rok trajanja može biti beskonačan u praksi se obično iz sigurnosnih razloga on ograničava.

Ovakav postupak identičan je ukoliko klijent poseduje sertifikat. Važnost sertifikata je u tome da je nama kao kupcu nekog proizvoda ili usluge na Internetu jako važno da smo sigurni da je server upravo onaj za kojeg se predstavlja. Kao posrednik u ovom poslu potrebne su usluge treće strane, CA (engl. Certificate Authority) ustanove. Njena uloga je u tome da „potpiše“ svojim javnim ključem sertifikat koji koristi naš server. Potpis CA garantuje da je sertifikat našeg servera verodostojan. Često se koristi sledeća jednačina da se pokaže proces:

$$\text{Poverenje} = \text{Autentifikacija} + \text{Enkripcija} + \text{CA}.$$

ServerKeyExchange: server šalje ovu poruku samo ukoliko ne poseduje sertifikat ili se ono koristi za digitalne potpise, a moguće je i da je zabranjen izvoz ključeva većih od 512 bita. U tom slučaju server generiše privremeni par javnih/tajnih ključeva i koristi ovu poruku za slanje tih ključeva.

CertificateRequest: poruka kojom server traži od klijenta da mu pošalje svoj javni sertifikat, služi za autentifikaciju klijentske strane.

CertificateVerify: služi da server proveri da li klijent zna tajni ključ. Ukoliko ga ne zna neće moći potpisati poruku koju će server proveriti s javnim ključem. U TLS-u svi sertifikati su X.509 oblika.

ClientKeyExchange: služi za razmenu ključeva između servera i klijenta, moguća su 3 oblika poruka i to su:

- Prvi je za Diffie-Hellman-ovu razmenu ključeva
- Drugi je za RSA ServerRSAparams parametri
- Treći je za Foretezza/DMS Server Foretzza Parametri.

ChangeCipherSpec: poruka kojom se vrši odabir načina šifrovanja.

Alert:Warning close_notify: poruka kojom se zatvara veza.

6.5 Efikasnost

Faza rukovanja je skupa. Njena skupoća proizlazi iz vremena koje je potrebno da bi se jedna SSL sesija u potpunosti obradila. Takav proces zahteva određeno procesorsko vreme. Dešifrovanje poruka *ClientKeyExchange* obavlja se na serveru što usporava fazu rukovanja. Tvorcii SSL su prilikom stvaranja postavili i zahtev za efikasnošću protokola. Zbog toga server vezi dodeljuje netransparentan ključ i sprema podatke u vezi. Kasnije kada se klijent ponovo spoji, može sa serverom koristiti prethodne ključeve. Na taj način ubrzava se ceo proces. U ovom procesu ostvaren je zahtev koji su tvorcii SSL imali pri definisanju zahteva kod faze stvaranja SSL – efikasnost. Informacije koje se čuvaju o svakoj sesiji su *pre_master_secret* koji se čuva u nekom vremenu.

6.5.1 SSL brzina

Koliko je zapravo brz ili možda spor SSL? Njegovu brzinu možemo uporediti simetričnim kriptografskim sistemom. Sledeća tabela prikazuje brzinu u Mbps:

Mašina	R C4 (Mbps)	3 DES (Mbps)
Athlon 600MHz (Linux)	5 41	40
Intel P III 450MHz (Linux)	4 08	30

IBM RS6000 43P/140 330MHz	1 92	17
Sparc Ultra 5 (Solaris 7)	1 76	14
SGI Indy (IRIX 6)	6 3	5

Tabela 3: Brzina SSL-a na različitim procesorima

Upoređujući tu tabelu sa tabelom asimetričnih krypto-sistema dolazimo do sledećih podataka. Na tabeli 4. prikazan je broj prijava u sekundi. Vidimo da na SGI Indy mašini SSL uspeva da ostvari tek 13 prijava. Nažalost, nedovoljno brzo za jedan ozbiljan on-line business (do 100 prijava u sekundi).

Mašina	Ope rativni sastav	Br .prijava/s
Athlon 600MHz (Linux)	Lin ux	10 0
Intel P III 450MHz (Linux)	Lin ux	73
IBM RS6000 43P/140 330MHz	Sola ris 7	27
Sparc Ultra 5 (Solaris 7)	AIX 4.3	27
SGI Indy (IRIX 6)	IRI X 6.4	13

Tabela 4: Broj prijava u sekundi

Treba napomenuti da pri obradi ovih zahteva dolazi do 100% iskorišćenja procesora. To znači da svi ostali servisi pokrenuti na računaru ostaju bez resursa. Načini na koji bi mogli ubrzati proces SSL prijava je da eventualno kupimo bržu mašinu, koja će koristiti brži procesor ili čak više njih u multitaskingu. Ubrzanje je moguće ukoliko koristimo HTTP1.1/Keepalive opciju u web serveru, i ukoliko klijent i server skladište sesiju u kešu (cache).

Dizajneri SSL-a sami su naglasili da SSL ne može sprečiti česte sigurnosne propuste. Naveli su i nekoliko tih propusta koje treba imati u vidu prilikom uvođenja SSL:

- Prvi savet je da su američki izvozni zakoni zabranjivali izvoz jake enkripcije. Svi RSA ključevi iznad 512 bita bili su zabranjeni. No taj zakon se nije odnosio na RSA ključeve koji su se koristili za operacije potpisivanja (gde treba koristiti veće ključeve jer 512 bita nije dovoljno sigurno). U nemogućnosti izbora većeg ključa preporuka je što češće menjati te ključeve.
- Drugi savet je da se obrati pažnja prilikom odabira funkcije za kreiranje slučajnog broja. Treba koristiti proverenu metodu koja će zaista nuditi nepredvidivi niz bitova.

- Treći savet odnosi se na pitanje sertifikata, CA treba izabrati sa pažnjom i mora imati mogućnost ukidanja sertifikata.

Fortezza, što na italijanskom jeziku znači tvrđava je enkripcijski algoritam koji je razvila američka agencija pod imenom NSA (engl. *National Security Agency*), a popularan je u državnim krugovima, policiji i vojsci. Koristi se u mobilnoj telefoniji. Popularan je i zbog dostupnosti hardvera koji obavlja dešifrovanje Fortezza algoritama. Naravno, takav hardver je dobrodošao jer značajno može ubrzati zahtevan proces SSL komunikacije. Ipak treba imati na umu da ga je razvila NSA, a onda izraziti i sumnju u „sigurnost“ takvog algoritma jer njegova struktura je tajna.

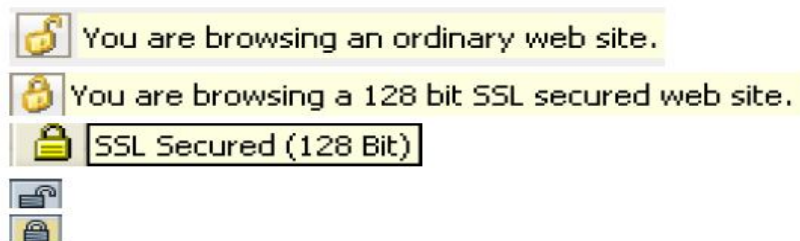
Za izbegavanje „*man in the middle*“ tipa napada, izumitelji predlažu korišćenje sertifikata. Prilikom faze rukovanja server je obavezan da pošalje svoj sertifikat koji je potpisan od Certificate Authority-a jer to garantuje autentičnost.

6.6 Primena SSL-a

SSL je stvoren prvenstveno kao odgovor na sverastuće potrebe za zaštićenim prenosom podataka putem Interneta. SSL je gotovo neophodan za sigurnost e-trgovine. Uprkos postojanju i drugih rešenja sigurnosti na Internetu kao što su: Secure-MIME protokol, Secure HTTP, Secure Shell, Private Enhanced Mail, MIME Object Security Services, Private Communication Technology, SSL se pokazao kao najbolje rešenje za osiguravanje elektronskih transakcija. Dalji uspeh e-poslovanja najviše zavisi od potrošačevog poverenja u funkcionisanju SSL-a. Ipak, kao i svi ostali sistemi, sigurnost SSL-a zavisi od svih karika u lancu koji ga čine: klijenta, servera, pa i izdavača potvrda. Svi ovi učesnici treba da brinu o sigurnosti i redovno održavati svoje sisteme kako bi se ostavila što manja mogućnost smetnji u komunikacijama i krađe, kako informacija tako i novca.

U budućnosti, SSL će moći obavljati mnogo više transakcija u kraćem vremenu. Dužina ključeva će rasti, algoritmi će biti uspešniji, sve u cilju razvoja sigurnosti informacija na Internetu.

Pristupljanjem stranicama zaštićenim SSL-om, menja se i izgled pretraživača (Browser). Iako je ikonografija različita od pretraživača do pretraživača, svaki od njih daje do znanja korisniku da li se nalazi na običnoj stranici ili na zaštićenoj.



Slika36. Prikaz ikonografija obične ili zaštićene web stranice

Implementacija SSL-a je moguća hardverski i softverski.

Možemo napraviti svoj lični SSL sertifikat za šifrovane komunikacije sa programom koji se zove OpenSSL.

OpenSSL je besplatan kriptografski alat koji implementira sigurnosne protokole SSL verzije 2 i 3 i TLS v1, kao i ostale kriptografske standarde koji se odnose na ove protokole. OpenSSL omogućava da se sa komandne linije pozivaju razne kriptografske funkcije ugrađene u zbirke datoteka programskog paketa OpenSSL. Možemo ga besplatno preuzeti sa Web stranice <http://www.openssl.org/>.

6.6.1 SSL Proxy servis

Ako želite siguran rad web servera, SSL Proxy može biti vaše rešenje. Na vezi između klijenta i servera se dodaje SSL podrška, ili obratno, ukoliko imate obični telnet klijent i želite da se spojite sigurnim putem. SSL Proxy je otprilike ista stvar kao Stunnel, ali sa manje mogućnosti za podešavanje.

6.6.2 SSL VPN servis

VPN SSL (engl. Virtual Private Network Secure Sockets Layer) je oblik VPN koji se može koristiti sa standardnim web pretraživačem. Za razliku od tradicionalnih IPsec VPN (engl. Internet Protocol Security), SSL VPN ne zahtijeva instalaciju specijalnog klijentskog softvera na računaru krajnjeg korisnika. Virtuelne privatne mreže (VPN) omogućavaju siguran komunikacioni mehanizam za podatke i druge informacije koje se prenose između dve krajnje tačke.

SSL VPN sastoji se od jednog ili više VPN uređaja na koje se korisnik spaja pomoću svog web pretraživača. Nudi svestranost, jednostavnost upotrebe i preciznu kontrolu za izbor korisnika na različitim računarima, pristup sredstvima iz mnogih lokacija.

Postoji dva glavna tipa SSL VPN:

- **SSL VPN Portal:** Ovaj tip SSL VPN omogućava jednu SSL vezu sa web lokacije tako da krajnji korisnik sigurno može pristupiti na više mrežnih servisa. Sajt poziva portal jer su to jedina vrata (jedna stranica) koja dovodi do mnogih drugih resursa. Udaljeni korisnik pristupa SSL VPN Gateway koristeći bilo koji moderni web pretraživač, sam sebe identifikuje ili poveže koristeći metodu autentifikacije podržane od strane gateway i onda se predstavlja sa web stranicom koja radi kao portal za ostale usluge.
- **SSL VPN Tunel:** Ovaj tip SSL VPN omogućava sigurnost web pretraživača kako bi pristup većem broju mrežnih usluga, uključujući i program i protokole koji se zasnivaju na web-u, kroz tunel koji se izvodi pod SSL. SSL VPN tunel zahteva da web pretraživač može da obradi aktivan sadržaj koji im omogućava da obezbede funkcionalnost koja nije dostupna SSL-VPN Portalom. Primeri aktivnog sadržaja uključuju Javu, JavaScript, Active X ili Flash aplikacije i plug-inove.

6.6.3 SSL Mail servis

Postoje dva načina za pristup svojim email poruka:

- **Webmail** je novouvedeni servis koji korisnicima omogućava da primaju i šalju poruke korišćenjem samo web pretraživača. Omogućen je download i upload attachmenta, kao i napredne funkcije kao što su kalendar, provera pravopisa i sl. Pristup ovom servisu je preko secure HTTP protokola (HTTP over SSL) tako da je sav saobraćaj između vašeg pretraživača i servera kriptovan. Iz tog razloga je dobro da se pre upotrebe webmail-a na sistem instalira ETF-ov CA digitalni sertifikat kojim je verifikovan digitalni sertifikat samog servera. Ako koristite pristupanje preko shell naloga, možete koristiti programe za pristup mailu koji rade u terminalskom okruženju kao što je program pine. Pine vam omogućava da pored ostalog, pristupate i news serveru i čitate UseNet konferencije.
- **Najčešći način pristupa mailu** je preko klijentskih mail programa koji se izvršavaju na vašem računaru, kao što su Outlook Express ili Netscape/Mozilla Mail. U ovom slučaju imate mogućnost da birate način na koji vaš mail klijentski program pristupa vašim porukama sa mail servera. Možete koristiti SPOP3 koji je isti protokol koji ste i do sada koristili za ovakav pristup mailu, samo što je u ovoj varijanti sva komunikacija između vašeg programa i servera kriptovana.

6.6.4 SSL za FTP servis

Na temelju nacрта IETF od Juna 1997, SSL FTP (engl. File Transfer Protocol) je opisan kao nastavak za FTP protokol koji je omogućio da bude osiguran, overena FTP komunikacija se odvija putem kanala podataka korišćenjem SSL protokola.

FTPS (takođe poznat kao Secure FTP i FTP-SSL) je nastavak na najčešći FTP servis koji dodaje podršku za TLS i SSL kriptografske protokole.

FTPS ne treba mešati sa SFTP, siguran prenos datoteka kompatibilnih podsistema SSH (engl. Secure Shell) protokola. SSH protokol omogućava da se prijavite na druge računare preko mreže premeštanjem datoteka i izvršavanjem naredbi. Koristeći SSH klijent, moguće je stvoriti siguran tunel koji štiti FTP autentičnost, što ga čini sigurnijim.

6.7 Testiranje SSL i njegove ocene

Rigorozne provere pre svakog izdavanja dozvola i sertifikata su obavezne da bi krajnji korisnici mogli verovati svaki put kad vide EV SSL (engl. Extended Validation) sertifikat. Novi internet pretraživači (počevši od verzije Internet Explorera 7, Firefoxa 3, Opere 9.5, Safarija 3.2, Chroma...) imaju podršku za EV SSL sertifikat, a krajnjim korisnicima je prikaz provere EV SSL sesije znatno vidljiviji nego stari SSL.

Web pretraživač	Operativni sistem	Najbolja šifra	Ocena
FireFox v3.05	Windows XP	AES 256-bit	Good
FireFox v3.05	Windows Vista & 7	AES 256-bit	Good
FireFox v3.01	Mac OSX 10.6	AES 256-bit	Good
Safari v3.2.1	Windows XP	AES 128-bit	Fair
Safari v3.2.1	Windows Vista & 7	AES 128-bit	Good
Safari v.3.2.1	Mac OSX 10.5.5	AES 128-bit	Good
Safari	iPhone v2.2	AES 128-bit	Good
Chrome v1.0.154.43	Windows XP	AES 128-bit	Fair
Chrome v1.0.154.43	Windows Vista & 7	AES 128-bit	Good
Internet Explorer v7	Windows Vista	AES 128-bit	Good
Internet Explorer v8	Windows 7	AES 128-bit	Good
Internet Explorer v7	Windows XP	AES 128-bit	Fair
Internet Explorer	Windows XP	AES 128-bit	Fair

v8		bit	ir
Internet Explorer v6	Windows XP	AES 128-bit	Fa ir
Opera v10.10	Windows XP	AES 256-bit	G ood
Opera v10.10	Windows Vista & 7	AES 256-bit	G ood

Tabela 5: Spisak poznatijih web pretraživača, kao i njegove ocene za SSL/TLS enkripciju

Koje enkripcione tehnike su podržane od strane modernih e-mail klijentskih programa?

Ako ste korisnici WebMail interfejsa, vaš odgovor zavisi od vašeg web pretraživača koji koristite, dok su u tabeli 6 navedeni nekoliko popularnih programa koji su testirani i njihove ocene za dati program.

E-mail program	Operativni sistem	Ocena	Rezultati
Mozilla Thunderbird v2.0.0.19	Windows XP	G ood	256-bit AES
Mozilla Thunderbird v2.0.0.19	Windows Vista & 7	G ood	256-bit AES
Thunderbird v2.0.0.19	Mac OSX v10.6	G ood	256-bit AES
Outlook 2010	Windows Vista & 7	G ood	128-bit AES (naveden je 256-bit, ali se ne koristi)
Outlook 2007	Windows XP	Fa ir	128-bit RC4 je najbolje podržan
Outlook 2007	Windows Vista & 7	G ood	128-bit AES (naveden je 256-bit, ali se ne koristi)
Outlook 2003	Windows XP	Fa ir	128-bit RC4 je najbolje podržan
Mail.app	Mac OSX v10.6	G ood	128-bit AES (naveden je 256-bit, ali se ne koristi)
Mail.app	Mac OSX v10.5	G ood	128-bit AES (naveden je 256-bit, ali se ne koristi)
Mail.app	iPhone v2.2	G ood	128-bit AES (naveden je 256-bit, ali se ne koristi)
Eudora v7.1.0.9	Windows XP	G ood	256-bit AES
Eudora v7.1.0.9	Windows Vista & 7	G ood	256-bit AES
Eudora v8.0.1.4	MAC OSX v10.5	G ood	256-bit AES
Entourage v12.1.5	Mac OSX v10.5	Fa ir	DES

Tabela 6: Spisak poznatih e-mail klijentskih programa, kao i njegove ocene za SSL/TLS enkripciju

7 NABAVLJANJE I INSTALIRANJE IIS-A

IIS se dobija uz sve operativne sisteme iz Microsofta, kako serverske verzije tako i one za personalnu upotrebu. U slučaju da je potrebna novija verzija IIS-a, ona se može skinuti sa sajta Microsoft-a. Postupak instaliranja IIS-a uz već postojeći operativni sistem je sledeća:

Klikom na dugme Start izabere se Control Panel.



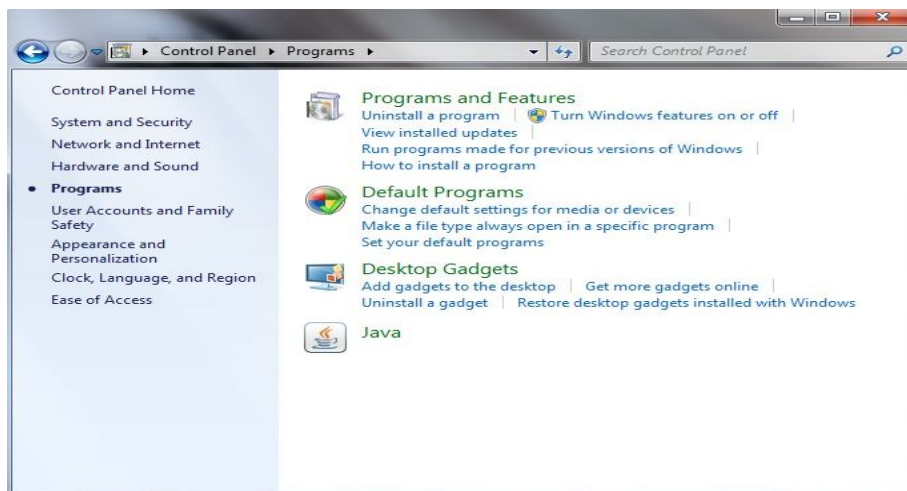
Slika37. Control panel

U Control Panelu se izabere opcija Programs. Ovakav raspored se opcija je karakteristican za operativni sistem Windows 7, dok se u ranijoj verziji tj. Windows Vista opcije nešto drugačije rasporedjne, mada lako se pronalaze i isto se nazivaju tako da nije velika ralika u samom postupku.



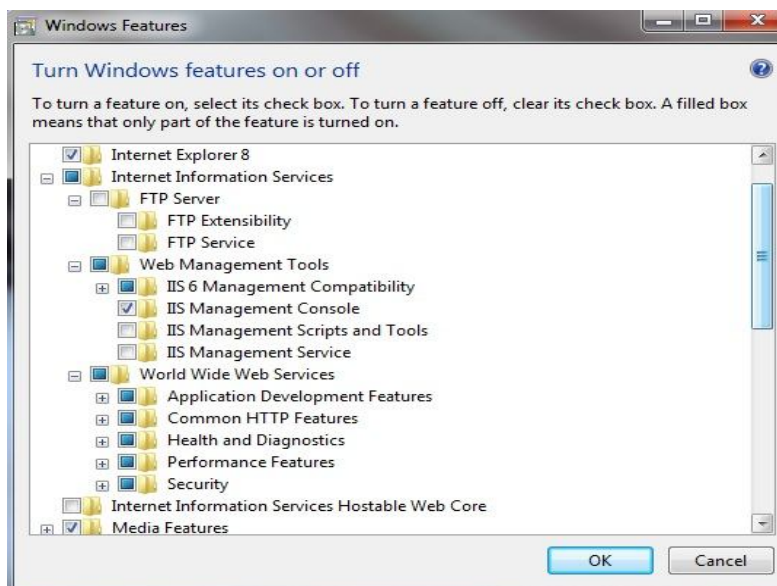
Slika38 Izaberite Programs

Klikne se na opciju Turn Windows features on or off.



Slika39. Klik na Turn windows features

Posle toga će se pojaviti prozor koji pokazuje da Windows proverava svoje uključene kopcije i izlistava spisak svega što možemo pokrenuti i instalirati.

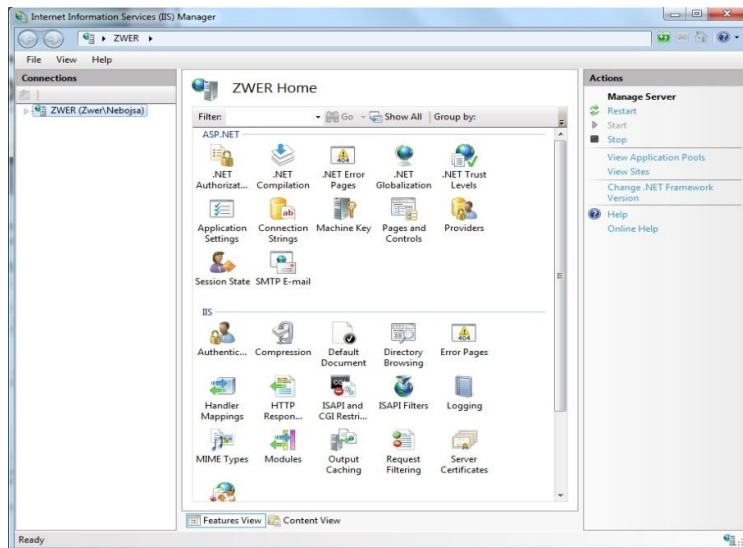


Slika 40. Izbor komponenti

Ako kliknemo na Internet Information Service on će potamniti polje u plavo, time pokazujući da je sam izabrao neke osnovne karakteristike za instalaciju IIS-a. Međutim pošto je IIS modularan mi sami možemo izabrati šta želimo da bude uključeno u instalaciju a šta ne, a takodje da napravimo neke svoje module. Ako proširimo opciju Internet Information Services videćemo da ona podrazumeva još 3 servisa koja možemo aktivirati. U zavisnosti od

namene IIS-a izabraćemo mogućnosti koje nama trebaju i započeti instalaciju klikom na OK dugme.

Po zavretku instalacije otvoriće se IIS Manager u kome se odvija dalji rad u IIS-u.



Slika41. IIS Manager

Po završetku instalacije IIS-a, kada se u browser-u unese URL <http://localhost> treba da se pojavi slika sa početnom stranom IIS-a.



Slika42. IIS

7.1 Postavljanje aplikacije

Pokretanje i puštanje u rad aplikacije se vrši detaljnim podešavanjima za svaku stranu posebno. Ova podešavanja zavise od toga šta želimo da bude na sajtu, da li će to da bude običan statički sajt, sajt sa bazom podataka ili aplikacija, kao što je i ovde slučaj. Takođe je važno napomenuti da se svi servisi ili dodatci mogu naknadno instalirati u zavisnosti od potreba sajta. Tu se takođe podrazumevaju i podešavanja vezana za SSL.

Sva ova podešavanja, a i više mogu se naći na zvaničnom sajtu Microsoft Internet Information Service sajtu <http://www.iis.net/>.

Za potrebe ove aplikacije i sa obzirom na to da de je ovo samo test, biće prikazano jednostavno ubacivanje web sajta preko menadžera.

U IIS menadžeru, sa leve strane (slika 40), postoji drvo koje se može proširiti. Desnim klikom na folder *Sites*, dibijamo mogućnost da napravimo novi sajt. U samom podešavanju biramo ime sajta, aplikacioni bazen u kome će se nalaziti i putanju do lokacije sajt na našem računaru.

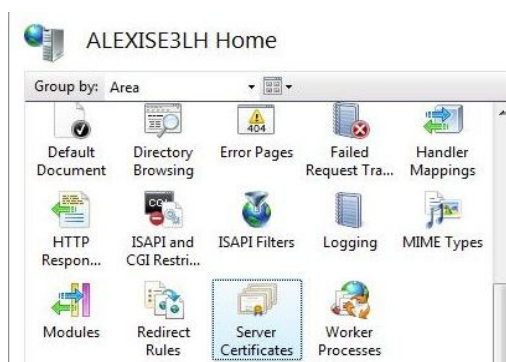
Ovde možemo i izvršiti testiranje našeg sajt i postaviti podešavanja za povezivanje sertifikata, ali najbolje je ne dirati te opcije, već kasnije ih detaljno podesiti korišćenjem IIS menadžera.

7.2 Podešavanje SSL-a

Za podešavanje SSL-a potrebno je imati certifikate koje omogućuju bezbedno korišćenje SSL-a i *https* protokola. Takve certifikate izdaju različite organizacije koje se bave bezbedošću sistema. Izdavanje nekog sertifikata traje oko 10 dana, jer mora podnosioc zahteva prolazi kroz kontrolu da bi se utvrdilo da nije zlonameran korisnik i da obavlja legalne radnje.

Za potrebe ovog rada koristićemo sertifikat koji ćemo sami napraviti jer se takvi sertifikati prave u slučaju nekog testiranja ili vežbe, kao što je i ovde slučaj. Podešavanje ćemo vršiti u IIS menadžeru.

Sertifikati se konfigurišu na tri različita mesta. Prvo mesto je na serverskom nivou, a dva se nalaze na sajtu koji želimo da konfigurišemo. Da bi podesili sertifikat na serverskom nivou kliknućemo na karticu “*Server certifikates*”. Tu biramo opciju “*Create Self-Signed Certificate*”, da bi smo nastavili testiranje, jer ostale opcije vode ka zvaničnim sertifikacionim organizacijama.



Slika43 Serverski sertifikat



Slika 44. Test sertifikat

Potrebno je samo da mu damo ime i pojaviće se kao validni sertifikat, ali samo na serverskom nivou, jer nije ratifikovan kod validnih organizacija. Kad je to uradno možemo da nastavimo ka vezivanju sertifikata za sajt. Desnim klikom na ime našeg sajta pojaviće se opcija “*Edit Bindings*”. Ovaj editor nam omogućava da pravimo, menjamo i brišemo veze sa sertifikatima. I to je drugo mesto za regulisanje sertifikata.

Pošto je SSL konekcija sigurna konekcija, ona ne korsiti standardni *http* protokol nego sigurni *https* protokol. Samim tim naš port više nije standardni 80 , već sigurnosni 443. Ostalo je još samo da u novoj vezi podesimo da koristimo test sertifikat.

Ako bi smo sada pokušali da udjemo na našu stranu lokalhosta, *browser* bi izbacio grešku, sa ispisom da postoji greška sa sertifikatom. Mi ćemo to zanemariti i ući na sajt. Tu ćemo videti da je URL adresa crvene boje i da dalje postoji greška u sertifikatu. To znači da je veza sigurna i šifrovana ali da je sertifikat nevalidan. Ako želimo da sprovedemo naš sertifikat u delo, moramo uraditi i treći korak u podešavanju sertifikata.

U IIS menadžeru postoji opcija “*SSL Setings.*”, koja se nalazi pod svim podešavanjima i lako ju je uočiti. Ovde možemo izričito da tražimo da naš sajt koristi SSL, pa čak i 128 bitnu enkripciju. Kada prihvatimo ova podešavanja, upsi se vrši u konfiguracioni fajl koji se odnosi na podrazumevani sajt.

Postupak dobijanja sertifikata za sajt koje želimo da postavimo je dosta drugačiji i podrazumeva proceduru slanja zahteva ovlašćenim organizacijama.

8 ZAKLJUČAK

U ovom master radu realizovana je aplikacija za šifrovanje i dešifrovanje fajlova, kao web aplikacija. Zatim je razmtrano kompletno okruženje za postavljanje i rad aplikacije. Za realizaciju su korišćene Microsoft tehnologije. Podignut je Internet Information Server, a sama aplikacija je napisana u Microsoft Visual Studio okruženju. Rad je podeljen na 8 glava koje obuhvataju sledeće. Prva glava je sadrži objašnjenje o serverima. Druga glava objašnjava šta su to web aplikacije. U trećoj glavi nalazi se prikaz napravljene aplikacije. Četvrta glava pruža uvid u jeoš neke mogućnosti pri kreiranju aplikacije. U petoj glavi analiziran je IIS. U šestoj glavi je izvršena analiza SSL-a. U sedmoj glavi je objašnjeno kako da se instalira i podesi IIS. Osma glava je zaključak sa literaturom.

Zaštita privatne svojine je odouvek bila značajan poduhvat, pa tako i danas u eri naprednih komunikacija i poslovanja, podaci koje posedujemo su od velikog značaja. Svedoci smo svakodnevnih ugrožavanja podataka od strane raznih napadača, hakera, organizovanih kriminalnih grupa, koji mogu podatke koje imamo da zlonamerno iskoriste.

Najveći problem ovog dela IT industrije je taj što je svaki korak koji se čini u nameri da se zaštite podaci, preduhitren od strane hakera, napadača i njihovih finansijera, jer su mogućnosti za zloupotrebu privatnih podataka jako veliki. Kada se nadje novi sistem za zaštitu podataka, napadači već uveliko izbacuju neku novu verziju zlonamernog softvera pomoću koga brobijaju zaštitu i stižu do podataka.

Različitim pristupima (zakonodavnim, socijalnim i tehnološkim) države, kompanije i organizacije za Internet bezbednost troše ogromne količine novca i resurse za istraživanja, eksperimente, stvaranje programa i alata za zaštitu, tehnika koje otkrivaju zlonamerne napade, propagadne materijale za osveščivanje korisnika, donošenje novih zakona protiv nelegalnog softvera. Sve ovo može doneti veći broj radnih mesta i doprineti dinamici IT industrije, i to je jedna dobra strana pojave zlonamernih korisnika. Loših strana ima mnogo više i one se ogledaju u konstantnoj bojznosti običnih korisnika socijalnih mreža i kompanija, da neko može doći u posed njihovim privatnim i poverljivim podacima i iskoristiti ih za ucene, finansisku dobit ili neki drugi način. Svi korisnici Interneta su podjednako ugroženi.

Da li će u narednom periodu biti sve više nelegalnih programa i sve što uz njih ide ili sve manje, zavisi pre svega od društava različitih država i IT industrije i njihovoj smelosti da vide kolika je opasnost sve većeg nereda u najvećoj informatičkoj mreži – Internetu.

Literatura:

1. Internet adresa: http://en.wikipedia.org/wiki/Server_%28computing%29
2. Internet adresa: http://en.wikipedia.org/wiki/Web_application
3. Internet adresa:
http://webtrends.about.com/od/webapplications/a/web_application.htm
4. Bruce Schneier, Jonh Wiles & Sons - Applied Cryptography, John Wiley & Sons, 1996.
5. Pleskonjić, Maček, Đorđević, Šarić - Sigurnost računarskih sistema, Mikroknjiga 2007
6. Andrew S. Tanenbaum - Računarske mreže, Mikroknjiga 2005
7. Internet adresa:
<http://www.codeproject.com/KB/aspnet/ExploringIIS.aspx>
8. Internet adresa:
<http://www.codeproject.com/KB/aspnet/IIS7ASPNet.aspx>
9. Internet adresa:
<http://www.dotnetfunda.com/articles/article821-beginners-guide-how-iis-process-aspnet-request-.aspx>
10. Internet adresa:
http://www.petri.co.il/configure_ssl_on_your_website_with_iis.htm
11. John Paul Mueller - IIS 7 Implementation and Administration, Wiley Publishing 2007
12. Dr. Shahram Khosravi - Professional IIS 7 and ASP.NET Integrated Programing – Wiley Publishing 2008