HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Tuneliranje TCP prometa kroz ICMP i HTTP protokole

CCERT-PUBDOC-2008-01-217

CARNet CERT u suradnji s LS&S

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.

Rezultat toga rada ovaj je dokument, koji je nastao suradnjom CARNet CERT-a i LS&S-a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr - nacionalno središte za **sigurnost računalnih mreža** i sustava.

***LS&S*, www.lss.hr -** laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

1. Uvod

Tuneliranje (eng. *tunneling protocol*) je postupak koji obuhvaća prijenos podataka jednog protokola – taj se naziva protokolom prijenosa podataka (eng. *payload protocol*), korištenjem nekog drugog protokola dostupnog na samoj fizičkoj mreži. Razlozi zbog kojih je postupak tuneliranja vrlo koristan uključuju, primjerice, potrebu za prijenosom podataka korištenjem neodgovarajuće računalne mreže, ali i za sigurnosni prijenos podataka nepouzdanom mrežom. Razumijevanje procesa tuneliranja zahtijeva preciznije razmatranje budući da ne odgovara u potpunosti niti modelu slojevitog protokola kao što je OSI (eng. *OSI Reference Model, Open Systems Interconnection Basic Reference Model*) niti TCP/IP (eng. *TCP - Transmission Control Protocol, IP - Internet Protocol*) modelu. Razmatranje treba započeti pojašnjenjem pojmova "podatkovni protokol" i "dostavljajući protokol". Podatkovni je izvorni protokol, inicijalni protokol u kojemu je nastala poruka, odnosno paket. Dostavljajući protokol je protokol koji se koristi za prijenos paketa, a potpuno je neovisan o podatkovnom protokolu. Enkapsulaciju protokola koja je u skladu s jednim od poznatih modela (OSI, TCP/IP) ne treba smatrati tuneliranjem. Primjer za to je HTTP protokol ugrađen u TCP protokol, koji je ugrađen u IP protokol, ugrađen pak u PPP (eng. *Point to Point*) protokol na V.92 modemima. Kao primjer mrežnog sloja enkapsuliranog u drugi mrežni sloj, može se uzeti *Generic Routing Encapsulation - GRE*, protokol koji leži nad IP (eng. *Internet Protocol*) protokolom. Spomenuti se postupak često koristi za prenošenje IP paketa s privatnim izvornim i odredišnim adresama preko Interneta korištenjem IP paketa s javnim adresama. U tom slučaju protokol prijenosa podataka usklađen je s protokolom dostave, a jedina razlika je u izvornim i odredišnim adresama korištenim u jednom i drugom protokolu. U suprotnosti s navedenim, podaci IP protokola prenošeni L2PT (*Layer 2 Tunneling Protocol*) protokolom tuneliranja, taj protokol tretiraju kao prijenosni (eng. *data link layer*). L2TP se implementira paketima UDP protokola na prijenosnom sloju korištenjem IP protokola. IP protokol može se koristiti na bilo kojoj implementaciji podatkovnog sloja (IEEE 802.2 i IEEE 802.3 - Ethernet ili PPP protokol korištenjem modemske veze). Proces tuneliranja može se proširiti sigurnosnim kodiranjem podataka kako bi zaštitio sadržaja paketa preko javne mreže poput Interneta, omogućavajući na taj način VPN (eng. *Virtual Private Network*) funkcionalnost.
Većina primjera u dokumentu načinjena je korištenjem operacijskog sustava Linux, a u nekim primjerima koristi se Windows poslužitelj. Uz odgovarajuće primjere naznačeni su i korišteni operacijski sustavi.

1.1. ICMP i HTTP protokoli

ICMP (eng. *Internet Control Message Protocol*) jedan je od najvažnijih protokola korištenih na Internetu. Središnja uloga mu je u slanju poruka o pogreškama kojima se obznanjuje učesniku s druge strane komunikacijskog kanala da je došlo do prekida rada servisa i on više nije dostupan, da neko određeno računalo s kojim se želi uspostaviti komunikacija nije dostupno ili da nije moguće doprijeti do željenog usmjerivača (eng. *router*). Prema svrsi, ICMP se razlikuje od TCP ili UDP protokola po činjenici da nije korišten za prijenos podataka kroz komunikacijski kanal i da ga većinom ne koriste izravno korisnici računala niti aplikacije, s izuzetkom poznatih *ping* i *traceroute* alata.
HTTP (eng. *HyperText Transfer Protocol*) je komunikacijski protokol korišten za prijenos podataka na *intranet* mrežama, a najpoznatija uloga mu je u prijenosu WWW (eng. *World Wide Web*) podataka. Izvorni razlog za nastajanje ovog protokola bio je omogućavanje objavljivanja i dohvaćanja tzv. *hypertext* stranica. Radi se o protokolu temeljenom na zahtjevu i odgovoru na zahtjev, odnosno arhitekturi klijent – poslužitelj. Tzv. *user agent* je klijentska aplikacija koja oblikuje zahtjeve i šalje ih poslužitelju. S druge strane komunikacijskog kanala nalazi se poslužitelj koji odgovara na zahtjeve te prosljeđuje slike, tekst ili bilo koje druge resurse klijentu. HTTP protokol ne mora nužno koristiti TCP/IP komunikaciju iako je ovaj način komunikacije najučestaliji na Internetu, budući da je zamišljen kao protokol kojeg je moguće implementirati na proizvoljnom pouzdanom (eng. *reliable transport*) protokolu komunikacijskog sloja.
Uspostava veze teče vrlo jednostavno: HTTP klijent uspostavlja TCP vezu prema određenom priključku poslužitelja (podrazumijevana vrijednost je 80) pri čemu se šalje zahtjev i za uspostavom HTTP veze. Poslužitelj u tom slučaju odgovara na zahtjev prema unaprijed određenim pravilima, a u ostalim trenucima je u stanju tzv. osluškivanja zahtjeva (eng. *listening*).

2. Tuneliranje korištenjem ICMP protokola

Tuneliranje ICMP protokolom je proces koji dopušta uspostavu TCP veze s udaljenim računalom korištenjem ICMP *echo* zahtjeva za komunikaciju u jednom smjeru te ICMP *reply* paketa za komunikaciju u drugom smjeru – što u cjelini omogućava odvijanje dvosmjerne komunikacije. Iako ovaj postupak u prvi mah može izgledati neopravdan, zapravo je koristan u određenim situacijama. Za ilustraciju korištenja ovakvog načina tuneliranja moguće je dati sljedeći primjer. Neka postoji dostupna bežična mreža koja dodjeljuje svim korisnicima ispravnu IP adresu, ali ne dopušta slanje TCP ili UDP paketa na Internet. Omogućena uspostava TCP ili UDP konekcija dozvolila bi pregled elektroničke pošte, otvaranje web stranica i sl. Međutim, korisnik uočava da je na mreži omogućeno korištenje *ping* alata – slanje i primanje ICMP *echo request* i *reply* poruka prema proizvoljnom računalu na Internetu. U tom slučaju, korištenjem tuneliranja ICMP protokolom može se ostvariti komunikacija koja inače zahtijeva TCP protokol.

2.1. Princip rada ICMP protokola

U uvodu je pojašnjeno korištenje ICMP protokola u svrhe prijavljivanja pogrešaka nastalih unutar IP paketa, dijagnostike i provjere rada usmjernika. ICMP poruke se stvaraju na IP sloju, najčešće od standardnog IP paketa, ugrađivanjem ICMP poruke u nj.
Svaki usmjernik prilikom prosljeđivanja svakog paketa umanji vrijednost parametra TTL (eng. *Time To Live*) za jedan. Ukoliko vrijednost tog parametra dosegne nulu, stvara se poruka tzv. ICMP *Time to live exceeded in transit* i odašilje prema izvorištu izvorne poruke.
Jedan od najpoznatijih primjera korištenja ICMP paketa je naredba `ping`. Njenom uporabom, uz ispravno postavljanje potrebnih parametara, moguće je ocijeniti povezanost dvaju računala na Internetu. Također, `ping` se koristi i za prikupljanje statističkih podataka, tzv. *round trip* vremena potrebnog da paket ode do određenog računala na Internetu i natrag, broja neuspješnih odgovora itd. Pokretanjem naredbe uz određivanje odredišnog računala zadavanjem njegove IP adrese, odašilje se ICMP *echo request* poruka. Ukoliko je odredišno računalu aktivno, od njega dolazi odgovor u obliku poruke ICMP *echo reply*. U protivnom, nakon isteka određenog vremena, odnosno nedobivanja odgovora, računalo se smatra nedostupnim. Primjer izvođenja `ping` naredbe za aktivno računalo dano je na sljedećem ispisu (upit se izvodi s računala pokretanog Windows operacijskim sustavom):

```
C:\>ping 192.168.1.10

Pinging 192.168.1.10 with 32 bytes of data:

Reply from 192.168.1.10: bytes=32 time<1ms TTL=128
Reply from 192.168.1.10: bytes=32 time<1ms TTL=128
Reply from 192.168.1.10: bytes=32 time<1ms TTL=128
Reply from 192.168.1.10: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Naredba `ping` za nedostupno računalo daje sljedeći odgovor:

```
C:\>ping 192.168.0.1

Pinging 192.168.0.1 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.0.1:
```

```
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Druga korisna primjena ICMP protokola je u izvedbi naredbe `traceroute`, korištene za određivanje povezanosti dvaju računala na mreži, ali s tim da daje informacije i o svim računalima koja se nalaze na putu od izvorišta do odredišta. Ona prvo šalje paket *ICMP echo request* s postavljenom vrijednošću TTL parametra na 1. Prvi čvor na kojeg naiđe ta poruka umanjuje vrijednost TTL parametra na nulu, zbog čega dolazi do odbacivanja paketa i slanja odgovarajućeg *ICMP Time to live exceeded in transit* odgovora. Sljedeći koraci uključuju slanje *request* paketa s vrijednošću TTL parametra postavljenom na 1, kako bi se ustanovilo koliko je različitih računala u neposrednoj blizini izvorišta, odnosno slanjem paketa uz postupno uvećavanje parametra za 1. Postupak se ponavlja sve do nailaska na odredišno računalo.

2.2. Osobine i zahtjevi ICMP tuneliranja

Aplikacija koja implementira tuneliranje ICMP protokolom mora zadovoljavati određene zahtjeve:

- ostvarivanje TCP komunikacije korištenjem ICMP echo request i reply paketa,
- pouzdanost uspostavljene veze (izgubljeni paketi se odašilju ponovno u slučaju gubitka),
- ostvarivanje višestrukih veza,
- primjerenu propusnost kanala (150 kb/s za primanje podataka i oko 50 kb/s za slanje je najmanje što ovakva komunikacija mora omogućiti) i
- autentikaciju radi onemogućavanja javnog korištenja posredničkog (eng. *proxy*) poslužitelja.

Za ispravan rad, odnosno ostvarivanje svega prethodno navedenog potrebno je:

- računalo dostupno na Internetu koje nije zaštićeno vatrozidom, odnosno koje omogućava primanje ICMP paketa,
- klijentsko računalo (najčešće prijenosno računalo korisnika ukoliko je riječ o bežičnoj komunikaciji),
- neograničena prava po mogućnosti na oba računala i
- operacijski sustav koji zadovoljava *posix* standard s programskom bibliotekom `libpcap` za programski pristup mehanizmu upravljanja mrežnim resursima što u konačnici omogućuje zaprimanje mrežnih paketa.

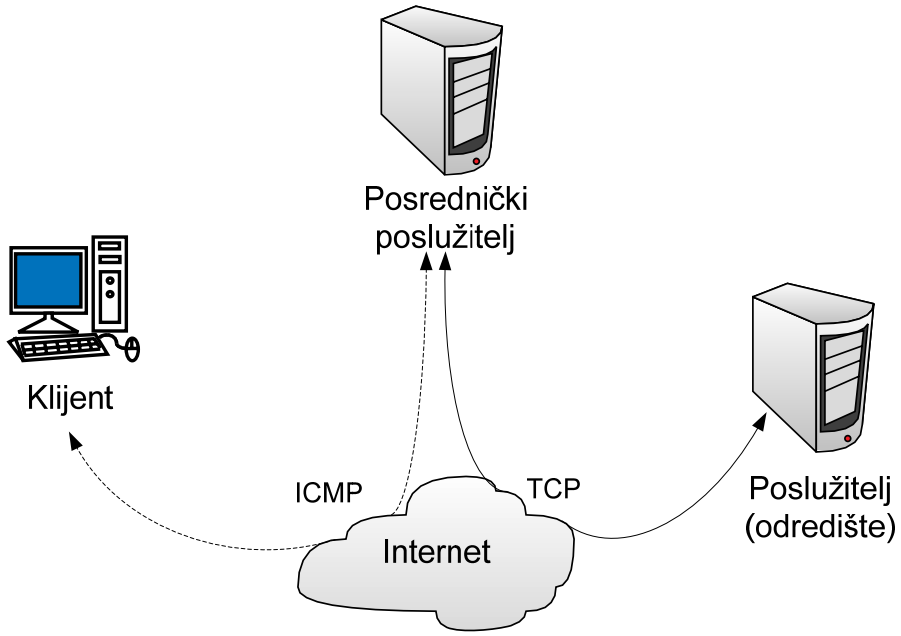
Jedna od konkretnih aplikacija koje implementiraju opisani način tuneliranja je *Ping Tunnel* ili skraćeno `ptunnel`, a dostupna je na web stranicama http://www.cs.uit.no/~daniels/PingTunnel/.

2.3. Način rada `ptunnel` aplikacije

`Ptunnel` omogućava ostvarivanje TCP komunikacije korištenjem ICMP protokola. Za razumijevanje ove vrste komunikacije potrebno je poznavanje pojmova koji uključuju klijent (eng. *client*), poslužitelj (eng. *server*) i posrednički poslužitelj (eng. *proxy*). Pri tome vrijedi sljedeća konvencija:

- klijent je računalo s kojeg korisnik želi pristupati proizvoljnim sadržajima na Internetu,
- poslužitelj je računalo kojemu bi klijent pristupao u slučaju uobičajene TCP komunikacije (odredište), a
- posrednički poslužitelj je računalo do kojega dolaze i na kome se terminiraju ping paketi (eng. *endpoint*).

Vrlo jednostavna blokovska shema takve komunikacije dana je sljedećim prikazom.

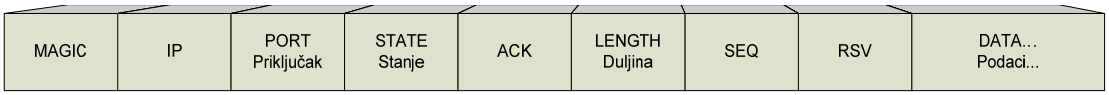


Slika 1. Shematski prikaz komunikacije kod ICMP tuneliranja

Za stvaranje stanja prikazanog slikom potrebna je mogućnost slanja i primanja *ping* paketa. Većina operacijskih sustava omogućava otvaranje tzv. *raw socket* priključaka kojima je moguće slati i primati ICMP pakete. Ipak, za otvaranje takvog načina komunikacije potrebno je posjedovati administratorske (eng. *root*) ovlasti na korištenom računalu. Klijent u postupku komunikacije participira korištenjem ICMP *echo request* paketa (tip 8) pri slanju, a posrednički poslužitelj koristi ICMP *echo reply* pakete (tip 0) za slanje odgovora. Moguć je i nešto drugačiji pristup koji podrazumijeva slanje ICMP *echo request* paketa od posredničkog poslužitelja prema klijentu, ali nema garancije da će ti paketi biti proslijeđeni klijent računalu zbog nepoznate konfiguracije mreže na kojoj se nalazi.

2.4. Opis protokola

Protokol koji određuje komunikaciju klijenta i posredničkog poslužitelja koristi četiri različita tipa poruka uz dodatna polja za pohranu rednog broja paketa (eng. *sequence number*) te polje za potvrdu primitka (eng. *acknowledgement field*). Polje nazvano `magic` iskorišteno je za razlikovanje uobičajenih `ping` zahtjeva i odgovora od onih koji se koriste u svrhe tuneliranja. Opći format paketa (ne uključujući IP ili ICMP zaglavlja) se može vidjeti na slijedećoj slici.



Slika 2. Format paketa korištenog za komunikaciju između klijenta i posredničkog poslužitelja

Polja s oznakama `ip` i `port` se koriste samo u paketima korištenim za komunikaciju klijenta i posredničkog poslužitelja. U njima se pohranjuje podatak koji određuje kamo klijent želi usmjeriti primljene pakete, a spomenute vrijednosti koriste se samo jednom – kada posrednički poslužitelj primi prvu poruku s oznakom `kProxy_start`. Sadržaj koda stanja ima dvojaku funkciju. Prva je da pokazuje koja je vrsta poruke primljena, a to može biti:

- poruka kojom je započela nova sjednica s posredničkim poslužiteljem (`kProxy_start`),
- poruka koja sadrži same podatke (`kProto_data`),
- eksplicitna potvrda zaprimljenog paketa (`kProto_ack`),
- poruka okončanja uspostavljene veze (`kProto_close`) ili
- poruka koja pripada nizu poruka u postupku autentikacije (`kProto_authenticate`).

Druga uloga se odnosi na podatke o pošiljatelju poruke. Poruka koja je poslana od strane klijenta imati će postavljenu zastavicu `kUser_flag`, dok će poruka čije izvorište je posrednički poslužitelj imati postavljenu zastavicu `kProxy_flag`. Ovaj je postupak potreban budući da će odgovor na *ping* zahtjev biti identičan odaslanom paketu, a u tom slučaju ne bi bilo moguće razlikovati pakete stvorene kao odgovor i pakete stvorene od strane posredničkog poslužitelja za tuneliranje. `Ack` i `seq` polja su međusobno povezana, a sam protokol stvoren je prema uzoru na TCP. To znači da se `ack` polje smješta redni broj posljednjeg zaprimljenog paketa. `Seq` polje se monotono povećava, a budući da je određeno sa svega 16 bita, najveća vrijednost koju može poprimiti je 2^{16}=65536. Nakon dovoljno dugog čekanja na potvrdu odlaznog paketa, druga strana komunikacijskog kanala pokušat će poslati ponovno prvi paket za kojeg nije primljena potvrda. Polje `length` predstavlja duljinu podatkovnog dijela paketa, te iznosi nula za sve slučajeve osim kada je vrijednost polja `state` postavljena na `kProto_data`. Konačno, `rsv` polje sadrži dva rezervirana okteta koja u tekućoj implementaciji služe isključivo u svrhu dopunjavanja (eng. *padding*). Po primitku `kProxy_start` zahtjeva, posrednički poslužitelj će otvoriti TCP vezu prema poslužitelju na temelju polja `ip` i `port`. Kako podaci dolaze TCP komunikacijom od poslužitelja, posrednički poslužitelj ih pretvara u odgovarajuće ICMP *echo* pakete odgovora i prosljeđuje klijentu. Klijent će izvoditi potpuno identičan postupak, s izuzetkom da će njegovi paketi uvijek biti ICMP *echo request* paketi. Tablicom 1 prikazane su vrijednosti kodova stanja i identifikatora.

Stanje	Kôd	Identifikator	Vrijednost
kProxy_start	0	kUser_flag	1 << 30
kProto_data	1	kProxy_flag	1 << 31
kProto_ack	2		
kProto_close	3		
kProto_authenticate	4		

Tablica 1. Stanja, identifikatori i njihove vrijednosti

Vrijednosti identifikatora dobivaju se tako da se u registar upiše vrijednost 1 i pomakne ulijevo 30, odnosno 31 mjesto. Pomak ulijevo označen je znakom <<.

2.5. Autentikacija

Paket *PingTunnel* podržava autentikaciju, tj. ovjeru vjerodostojnosti. Sam postupak je vrlo jednostavan. Korisnik najprije određuje ključnu riječ (zaporku, eng. *password*) ili ključni izraz (eng. *passphrase*), od kojega se potom stvara sažetak (eng. *hash*) korištenjem MD5 algoritma. Uvijek kada posrednički poslužitelj primi zahtjev za uspostavom novog tunela, odgovorit će sa izazovom ovjere vjerodostojnosti (eng. *authentication challenge*). Izazov se sastoji od vremenske oznake (eng. *timestamp*) proširene slučajnim podatkom, tako da sve zajedno bude veličine 32 okteta. Odgovor se računa prema izrazu

```
md5(challenge + md5(password))
```

pri čemu znak plus (+) označava konkatenaciju (povezivanje) dvaju znakovnih nizova (eng. *string*). Posrednički poslužitelj provjerava rezultat primjenom istog MD5 algoritma te ga nakon računanja uspoređuje s izvornim. Ukoliko ovjera vjerodostojnosti uspije, posrednički poslužitelj će dopustiti početak TCP prometa u proizvoljnom smjeru, a ako ne uspije, uspostavljena veza se prekida.

2.6. Rukovanje višestrukim konekcijama

Poslužitelj mora omogućiti uspostavljanje proizvoljnog broja veza, a ne samo jedne. Da bi to bilo ostvarivo, razvijen je postupak razlikovanja pojedinih konekcija korištenjem ICMP identifikacijskog polja. Klijent generira slučajan identifikator tijekom uspostave sjednice, a potom računalo na drugom kraju komunikacijskog kanala koristiti taj identifikator za određivanje pripadnosti paketa određenoj vezi. Mehanizam nije idealan, ali funkcionira prihvatljivo i ispravno sve dok dvije uspostavljene veze ne pokušaju koristiti isti identifikator. Autor alata navodi kako u tekućoj inačici paketa nije razvijen

mehanizam koji bi prijavljivao ovakve pogreške. Polje za pohranu rednog broja ICMP paketa se ne koristi u implementaciji paketa `ptunnel`, ponajviše zbog očekivanja da će neki od usmjerivača ispustiti paket čiji se redni broj ponavlja. Umjesto toga, zasebno polje se koristi u ove svrhe i to u sklopu `ptunnel` paketa.

2.7. Otvori za slanje i primanje

Otvori za slanje i primanje su metode kojima se kontrolira protočnost kako bi se izbjegla zagušenja kanala zbog, primjerice, sporosti primanja i obrade paketa na odredištu i prevelike brzine slanja paketa s izvorišta. Ovaj postupak se naziva kontrolom protoka podataka (eng. *flow control*). Implementacija za TCP protokol naziva se klizećim otvorom (eng. *sliding window*). Primjenjujući ovu tehniku primatelj određuje veličinu otvora za primanje – količinu podataka koju će spremati u privremeni spremnik. Ista veličina određuje i količinu podataka koju pošiljatelj može odaslati prije nego što započne proces čekanja na potvrdu prijema.

`Ptunnel` koristi jednostavni koncept otvora za nadzor slanja i primanja, mehanizam koji kontrolira broj paketa prenošenih u istom trenutku komunikacijskim kanalom. Otvor je u tekućoj inačici statički određen veličinom od 64 paketa, ali broj se može proizvoljno mijenjati korekcijom `ptunnel` postavki dostupnih u odgovarajućim datotekama zaglavlja. Povećavanje veličine otvora nužno povećava i najveći potencijalni iznos protočnosti komunikacijskog kanala. Otvori za slanje i primanje su jednostavno implementirani skupom kružnih nizova (eng. *circular array*), s pripadnim pokazivačima koji određuju sljedeće dostupno mjesto za slanje ili primanje i prvi sljedeći paket za kojeg nije zaprimljena potvrda.

2.8. Gubitak paketa

`Ptunnel` aplikacija može se nositi s gubitkom paketa. To radi tako da ponovno pošalje paket za kojeg se pretpostavlja da je izgubljen. Slanjem paketa povećava se redni broj sekvence (eng. *sequence number*), a i klijent i posrednički poslužitelj posjeduju podatak o posljednjem paketu za kojeg je zaprimljena potvrda primitka. U slučaju da se za neki paket ne primi potvrda, on se ponovo šalje. Vremensko ograničenje za primitak potvrde iznosi 1.5 sekundu. Učesnik u komunikaciji će poslati ponovo samo prvi paket za kojeg se sumnja da je izgubljen. Kada je primitak spornog paketa potvrđen, moguće je ponovno slanje sljedećeg, odnosno sljedećih, paketa, ali to ovisi o broju potvrđenih paketa. Ukoliko sljedećih nekoliko paketa bude također potvrđeno, oni će biti maknuti iz reda za slanje. Uobičajena je situacija gubitka pojedinog paketa, dok se većina ostalih ispravno isporučuje i potvrđuje. Mehanizam je razvijen tako da izbjegava nepotrebna ponovna odašiljanja paketa u najvećoj mogućoj mjeri.

2.9. Kontrola zakrčenja

Paket `ptunnel` u tekućoj inačici ne posjeduje mehanizam za kontrolu zagušenja prometa. Alat odašilje toliko `ping` paketa koliko veličina prozora dopušta, proizvoljnom brzinom. Ovaj postupak potencijalan je nedostatak čitavog mehanizma tuneliranja budući da nije izveden na odgovarajući način.

2.10. Nemogućnosti tuneliranja

Postoji nekoliko situacija kada se tuneliranje neće moći uspješno izvršiti. Te se situacije mogu svrstati u sljedeće kategorije:

1. onemogućeno prometovanje odlaznih i dolaznih `ping` paketa,
2. neispravnost uzrokovana operacijskim sustavom i
3. sve ostale potencijalne neispravnosti.

Prvi scenarij nije moguće ispraviti ni na koji način. Ako je prometovanje ICMP paketa na mreži onemogućeno, tuneliranje njihovim korištenjem nije izvedivo.

Problem drugog scenarija moguće je zaobići korištenjem vanjske programske biblioteke koja omogućuje korištenje mehanizama vezanih uz mrežne resurse. Na taj način moguće je pristupiti mrežnim paketima prije no što ih operacijski sustav obradi. Ovo je nužno na nekim platformama među

koje je uključen i Mac OS X operacijski sustav. Problem koji se nalazi u operacijskom sustavu, primjerice , može biti ne isporučivanje ICMP *request* paketa tzv. *raw* priključcima nego njihova obrada izravno u jezgri operacijskog sustava. Zaobilazni postupak uključuje već spomenuto korištenje vanjske biblioteke za pristup mrežnih resursima, ali nedostatak je u tom slučaju smanjena propusnost komunikacijskog kanala. Iz tog razloga preporuča se korištenje paketa u podrazumijevanom načinu rada, bez korištenja posebnih biblioteka, a tek u slučaju neispravnog rada u tom režimu, treba koristiti pomoćnu biblioteku.

2.11. Preuzimanje i korištenje paketa

Izvorni kod programskog paketa `ptunnel` korisnik može preuzeti sa sljedeće web adrese: http://www.cs.uit.no/~daniels/PingTunnel/PingTunnel-0.61.tar.gz. Tekuća inačica nosi oznaku 0.61. Inačica za Microsoft Windows operacijske sustave dostupna je na adresi http://www.cs.uit.no/~daniels/PingTunnel/PingTunnel-Windows.zip. Instalacija na Debian Linux operacijskom sustavu obuhvaća dodavanje sljedećih linija u datoteku `/etc/apt/sources.list`

```
deb http://www.cti.ecp.fr/~beauxir5/debian binary/
deb-src http://www.cti.ecp.fr/~beauxir5/debian source/
```

Zatim slijedi pokretanje naredbe:

```
apt-get update
```

nakon koje slijedi:

```
apt-get install ptunnel
```

Korištenje paketa na klijentu određeno je sljedećom sintaksom:

```
bash$ ./ptunnel -p <proxy address> -lp <listen port> -da <destination address> -dp <destination port> [-c <network device>] [-v <verbosity>] [-f <logfile>] [-u] [-x password]
```

dok je na poslužitelju određeno izrazom

```
bash$ ./ptunnel [-c <network device>] [-v <verbosity>] [-f <logfile>] [-u] [-x password]
```

Prekidač `-p` koristi se za postavljanje adrese računala na kojemu je pokrenut posrednički poslužitelj. Jednostavan test za ispravnost adrese je slanje `ping` paketa, ukoliko korisnik dobije odgovor, uvjeti za tuneliranje su ispunjeni.
Prekidači `-lp`, `-da` i `-dp` postavljaju lokalni priključak, odredišnu adresu i odredišni priključak. Za tuneliranje SSH konekcija od klijenta preko posredničkog poslužitelja do željenog odredišta, koristi se sljedeća naredba:

```
sudo ./ptunnel -p posrednick.posluzitelj.com -lp 8000 -da odredisni.posluzitelj.com -dp 22

##Uspostavljanje SSH veze do odredisni.posluzitelj.com izvodi se na sljedeći način:##

ssh -p 8000 localhost
```

Kako bi tuneliranje ispravno radilo, potrebno je pokrenuti posrednički poslužitelj na odgovarajućem računalu što se čini sljedećom naredbom:

```
sudo ./ptunnel
```

Ukoliko posrednički poslužitelj ne radi ispravno, potrebno je omogućiti praćenje paketa na mrežnom uređaju. Omogućavanje ovog načina rada naznačuje se prekidačem `-c` i određivanjem mrežnog sučelja tijekom zadavanja naredbe za pokretanje poslužitelja. Isti postupak koristi se i na klijentskom računalu.
Za ograničavanje korištenja poslužitelja moguće je postaviti zaporku korištenjem prekidača `-x`. Zaporka se ne šalje u izvornom obliču (eng. *plain text*), ali treba imati na umu da je moguće do nje doći korištenjem alata `top` ili `ps` koji ispisuju zadanu naredbu.
Prekidač `-u` pokušat će pokrenuti posrednički poslužitelj u načinu rada koji ne zahtijeva *root* ovlasti, a prekidačem `-v` postavlja se količina izlaznih informacija iz `ptunnel` programa.

3. Tuneliranje korištenjem HTTP protokola

Povećanjem količine prometa i broja ljudi prisutnih na Internetu pojedine usluge na mreži sve se više onemogućavaju iz sigurnosnih razloga. Ipak, jedna od najčešće korištenih usluga je ona na priključku broj 80. Radi se o vrlo poznatom HTTP servisu, koga se u pravilu rijetko isključuje iz očitih razloga.
Tuneliranje nije novitet u svijetu komunikacija. Najpoznatije izvedbe ove tehnologije predstavljaju IPSec i SSH tuneli. Međutim, mala je vjerojatnost da će zlonamjeran korisnik izabrati IPSec ili SSH kao način za izvođenje neovlaštenog pristupa željenom računalu. On to neće učiniti jer većina ozbiljnijih računalnih mreža posjeduje i sustave za detekciju i prevenciju neovlaštenog pristupa (eng. *IDS - Intrusion Detection Systems, IPS - Intrusion Prevention Systems*). Zadaća takvih mehanizama je uočavanje svih potencijalno sumnjivih aktivnosti i obavještavanje administratora o prijetnjama. Unatoč snazi i vrlo inteligentnim metodama prepoznavanja sumnjivog ponašanja ugrađenim u navedene sustave, i oni imaju svoja ograničenja. HTTP tuneliranje jedno je od njih.
Ovo tuneliranje utemeljeno je na principu ugrađivanja i slanja cjelokupnog prometa, odnosno svih korisnih podataka unutar HTTP zaglavlja. Odredišni poslužitelj zatim analizira primljena zaglavlja, ekstrahira podatke i usmjerava pakete željenim protokolom do njihovog konačnog odredišta. I UDP i TCP podaci mogu biti ugrađeni u zaglavlja. To je tako jer tunel "vidi" pakete isključivo kao sirove podatke, bez analize njihova sadržaja.
Postoji više paketa koji implementiraju HTTP tuneliranje, a jedan od najranijih je *GNU HTTPtunnel*. Uz njega, postoji i komercijalno rješenje s nazivom *HTTP Tunnel*. Prvi paket dostupan je na adresi http://www.nocrew.org/software/httptunnel.html, a drugi na http://www.http-tunnel.com/html/. Daljnji opisi dani u nastavku temelje se na besplatnom rješenju otvorenog koda budući je ono prihvatljivo svim postojećim korisnicima.

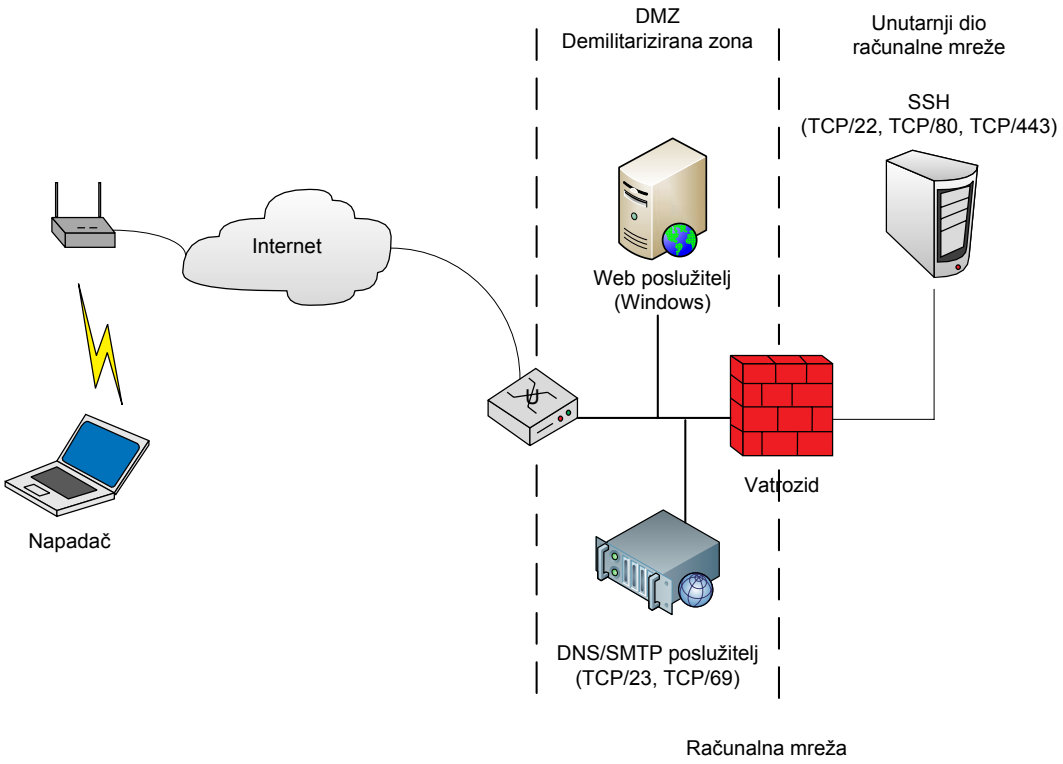
3.1. Opis HTTP tuneliranja

HTTP tuneliranje može se iskoristiti za pristup mrežnim priključcima i računalima kojima je inače nemoguće pristupiti s javnog dijela mreže. Računalo napadača prikazano je na lijevoj strani slike Slika *3* i ono koristi Unix-temeljen operacijski sustav, pri čemu je desni dio prikaza rezerviran za shemu demilitarizirane zone i unutrašnjosti privatne mreže na kojemu je također računalo s Unix-temeljenim operacijskim sustavom. Pri tome je usmjerivač postavljen na slijedeći način:

```
inbound:
  permit tcp any host WWW port 80
  permit tcp any host WWW port 443
  permit tcp any host DNS/SMTP port 25
  permit udp any host DNS/SMTP port 53

outbound:
  permit ip any any
```

Opisana pravila omogućuju preusmjeravanja paketa iz vanjske mreže jedino na TCP priključke 80 i 443 računala s nazivom WWW (web poslužitelj na slici) i Windows operacijskim sustavom, te 25 i 53 na računalo s nazivom DNS/SMTP i Unix-temeljenim operacijskim sustavom (označen kao DNS/SMTP poslužitelj na slici). Za izlazni promet nema ograničenja.



Slika 3. Shematski prikaz mreže pogodne za HTTP tuneliranje

Vatrozid ima postavljena sigurnosna pravila koja omogućuju uspostavu veza prema unutarnjoj zoni mreže, i to omogućavanjem pristupa priključcima 22, 80 i 443 računalu s nazivom DNS/SMTP, uz korištenje sigurnosnog protokola SSH. Izlaz iz mreže je dopušten svima:

```
inbound:
  permit ip host DNS/SMTP host SSH eq 22
  permit ip host DNS/SMTP host SSH eq 80
  permit ip host DNS/SMTP host SSH eq 443

outbound:
  permit ip any any
```

Iako je dani primjer iznimno jednostavan, dovoljan je za potpunu prezentaciju i opis načina rada tuneliranja. Zlonamjeran korisnik usmjeren je na iskorištavanje web poslužitelja kroz koga će se kanalizirati željeni promet prema konačnom odredištu – u jednostavnom primjeru tu je riječ o DNS/SMTP poslužitelju, a u proširenom primjeru to je računalo s unutarnjeg dijela mreže. Prvi korak u postavljanju odgovarajućih okolnosti za tuneliranje je pristup naredbenom retku računala korištenog za posredovanje tijekom tuneliranja (web poslužitelj na slici). Način na koji se može ostvariti pristup naredbenom retku je proizvoljan, a napadač se najčešće odlučuje za iskorištavanje poznatih propusta u ovisnosti o inačici sustava pokrenutog na ciljanom poslužitelju. Jednom kada napadač stekne potpuno pravo zadavanja proizvoljnih naredbi iz naredbenog retka, sljedeći korak je postavljanje izvršne inačice HTTP posredničkog poslužitelja na cilj. U danom primjeru riječ je o računalu s operacijskim sustavom Microsoft Windows. Budući da je riječ o datoteci s nazivom `hts.exe` naredba za pokretanje poslužitelja za HTTP tuneliranje glasi:

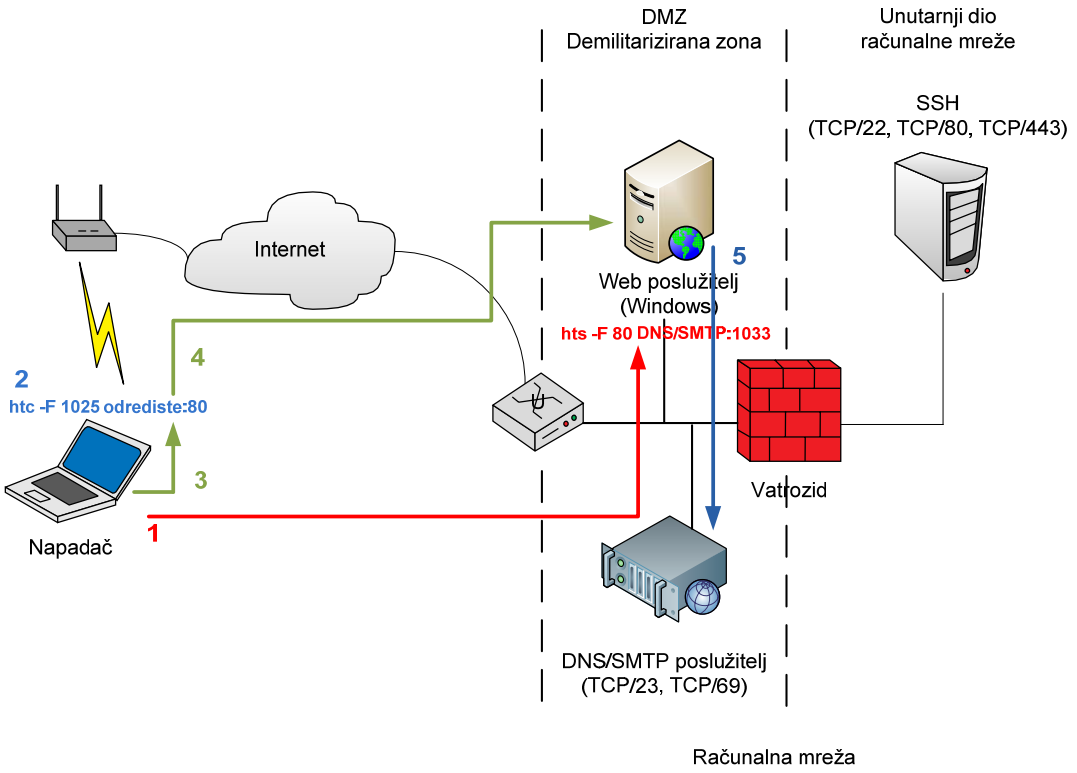
```
hts.exe -F (SRC PORT) (TARGET):(DST PORT)
```

Oznaka `(SRC PORT)` predstavlja priključak na kome posrednički poslužitelj za HTTP tuneliranje očekuje uspostavljanje veze. Oznaka `(TARGET)` je IP adresa odredišnog računala, dok je `(DST PORT)` oznaka priključka na odredišnom računalu kojemu se usmjerava promet. Uspostavom tunela

klijent šalje podatke posredničkom poslužitelju na priključak označen sa `(SRC PORT)`. Posrednički poslužitelj ih zatim raspakira i prosljeđuje odredišnom poslužitelju s adresom `(TARGET)` i to na priključak `(DST PORT)`. Odredišno računalo može biti i računalo na kojemu je pokrenut poslužitelj HTTP tuneliranja. Nakon što je `hts` poslužitelj postavljen u ispravno stanje rada, korisnik pokreće klijent aplikaciju na svom računalu te promet koji želi prenijeti do odredišnog računala usmjerava na računalo na kojemu je posrednički poslužitelj i to na priključak određen prethodnim korakom. Oblik naredbe koju je potrebno pokrenuti na klijentskoj strani glasi:

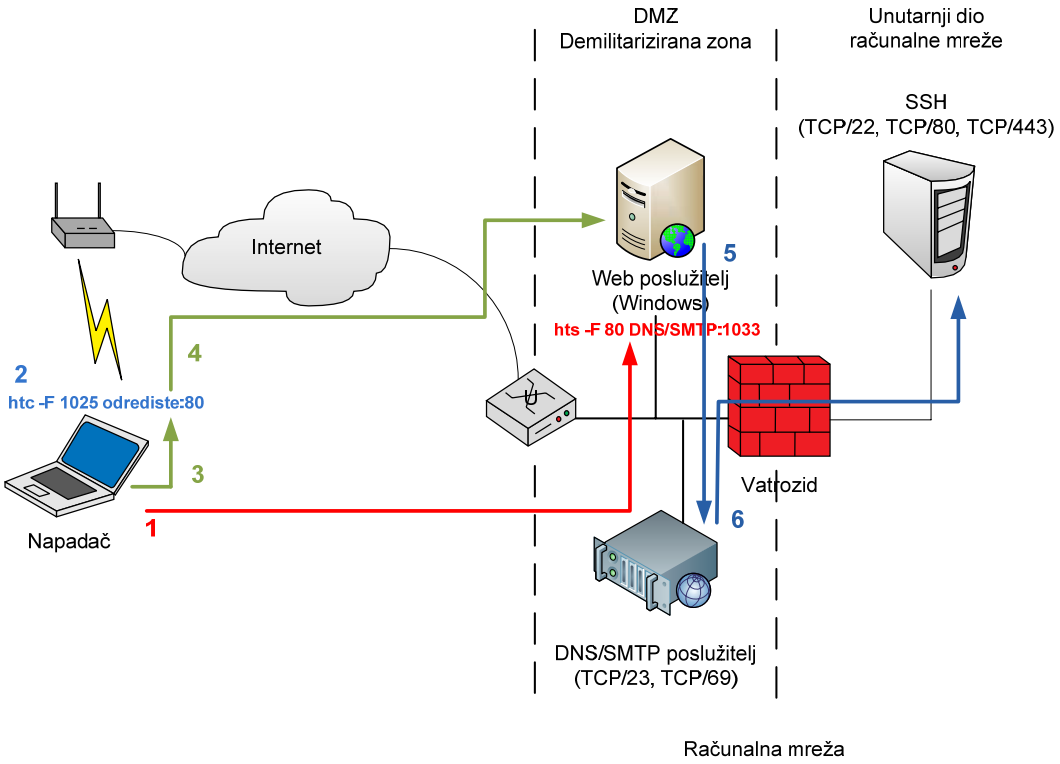
```
htc -F (SRC PORT) (TARGET):(DST PORT)
```

Zadatak klijentske aplikacije jest očekivanje uspostave veze na `(SRC PORT)` priključku i usmjeravanje prometa na `(DST PORT)` računala određenog adresom `(TARGET)`.
Imajući na umu prethodan primjer odnosa napadača i ciljane mreže, treba promotriti sliku 4. Ona prikazuje postavljanje poslužitelja za HTTP tuneliranje (prvi korak) na web poslužitelj odredišne mreže. Sigurnosna arhitektura takve mreže zahtijeva postavljanje web poslužitelja u tzv. demilitariziranu zonu. Riječ je o posebnom dijelu mreže kojemu je omogućen javni pristup, ali radi sigurnosti taj dio mreže odvojen je od ostatka računalne mreže kojemu nije omogućen pristup s Interneta. Ta dva dijela mreže komuniciraju obično preko vatrozida. U drugom koraku napadač na svom računalu pokreće klijentsku aplikaciju HTTP tuneliranja – program `htc`. Proces `hts`, pokrenut u prvom koraku, na web poslužitelju očekuje promet na priključku broj 80. To je priključak rezerviran za uspostavu veze namijenjene prenošenju web, odnosno HTTP podataka. Važnost ovog procesa nije samo u čekanju uspostave veze na spomenutom priključku, nego i u mogućnosti preusmjeravanja prometa. U danom primjeru preusmjerava se na `telnet` uslugu (TCP priključak 23) računala zaduženog za ostvarivanje DNS i SMTP usluga. Napadaču u trećem koraku preostaje spajanje na priključak 1025 na vlastitom sustavu. Klijent `htc` tada maskira promet u HTTP zaglavlja i takve mrežne pakete usmjerava prema web poslužitelju na priključak 80 (označeno na slici brojem 4). Poslužitelj zatim prihvaća takve mrežne pakete, uklanja HTTP zaglavlja i prosljeđuje čitav promet DNS/SMTP poslužitelju na priključak 23, što je označeno brojem 5. Konačna svrha čitavog postupka je ostvarivanje pristupa prema prethodno spomenutom poslužitelju, kojemu pristup s Interneta inače nije omogućen. U slučaju poznavanja korisničkih podataka nekog korisnika sustava ili pogađanjem korisničkog imena i zaporke, napadač dobiva pristup ciljanom računalu.



Slika 4. Prikaz redoslijeda radnji kod tuneliranja

Alternativna mogućnost jest usmjeriti `hts` poslužitelj na `finger` priključak, tj. TCP priključak 79, SMTP/DNS poslužitelja i na taj način doći do informacija o korisnicima sustava. Ovaj postupak koristan je za dobivanje podataka o korisničkim računima – korisničkog imena u prvom redu. Sljedeći korak obuhvaća iskorištavanje nekog poznatog propusta kao što je napad uzrokovanjem pojave prepisivanja spremnika (eng. *buffer overflow*) u svrhu neovlaštenog pristupa udaljenom sustavu. Drugi način je tzv. *brute force* metoda iscrpljivanja svih mogućih kombinacija zaporki dok se ne pogodi ispravna. Pristup poslužitelju do koga je inače nemoguće doći s javne mreže sada je ostvaren. Nakon stečenog neovlaštenog pristupa naredbenom retku ovog računala, postavljanjem odgovarajućih alata napadač može ostvariti pristup prema zaštićenom dijelu računalne mreže. Primjerice, kontinuiranim nadziranjem uspostavljenih veza SMTP/DNS poslužitelja i nekog od računa s unutarnje strane mreže, zlonamjeran korisnik dolazi do podataka o pokrenutim uslugama s druge strane vatrozida. Nešto manje neprimjetna metoda je uobičajeno skeniranje priključaka (eng. *port scanning*). Pronalaskom SSH poslužitelja na sigurnoj strani mreže moguće je iskoristiti SMTP/DNS računalo za pokušaj spajanja na poslužitelj korištenjem ranije prikupljenih korisničkih podataka ili pogađanjem nepoznatih korisničkih imena i zaporki. Opisani pristup prikazan je na slijedećoj slici.



Slika 5. Pristup zaštićenom dijelu mreže

3.2. HTTP tuneliranje i testiranje sigurnosti mreže

Jedna od korisnih primjena procesa HTTP tuneliranja jest provjeravanje mrežne sigurnosti. Većina računalnih mreža postavljena je tako da ograničava ulazni promet na točkama spajanja s javnim mrežama, ali jednako tako omogućava promet prema van. Postupkom HTTP tuneliranja tijekom testiranja neprobojnosti, moguće je obaviti iscrpnije testiranje i time poboljšati kvalitetu procjene sigurnosti. Nadalje, HTTP tuneliranje korisno je i za ocjenu kvalitete djelovanja sustava za otkrivanje nedozvoljenog pristupa mreži (eng. *IDS – Intrusion Detection System*).

4. Zaključak

Postupak tuneliranja odličan je primjer općenitosti protokola korištenih na Internetu. Iz samog načina rada uočava se kako je moguće koristiti proizvoljan protokol za prijenos paketa stvorenih prema bilo kojem drugom proizvoljnom protokolu. Temelj svega je postupak ugrađivanja jednog paketa u drugi (eng. *encapsulation*) i činjenica da su Internet protokoli stvarani tako da ne ovise o vrsti podataka koju prenose. Upravo to je ujedno i prednost i nedostatak. Neovisnost podataka otežava uređajima za otkrivanje neovlaštenog pristupa i sličnima uočavanje potencijalno opasnih korisnika. S druge strane, takav pristup dovoljno je općenit tako da dozvoljava postupke poput tuneliranja. Oni su korisni u osiguravanju podataka tijekom komunikacije na mreži, ali i u slučaju kada je preko mreže koja ne koristi određeni protokol potrebno prenijeti podatke stvorene upravo tim protokolom kojeg mreža ne podržava.

Tuneliranje korištenjem ICMP i HTTP protokola, kako je pokazano, zahtijeva korištenje programskih rješenja koja implementiraju posredničke poslužitelje – pozadinske aplikacije koje su u mogućnosti otpakirati podatke ugrađene u podatkovni dio prijenosnog protokola, ali i klijentske programe – aplikacije koje su u stanju pripremiti odgovarajuće oblikovane podatke. Postavljanje poslužitelja na posredničko računalo nije jednostavan zadatak i zahtijeva odgovarajuću vještinu koja uključuje i sposobnost iskorištavanja propusta i preuzimanja ovlasti nad udaljenim računalima. Dakako, prethodno rečeno vrijedi samo za zlonamjerne aktivnosti. Budući da se radi o netrivijalnom zadatku, tuneliranje u zlonamjerne svrhe nije moguće jednostavno iskoristiti iako sam pristup ima dosta prednosti, kako je pokazano u dokumentu. Što se tiče korištenja tuneliranja u legitimne svrhe, postojeći komercijalni i besplatni alati toliko su pojednostavljeni da se korištenje svodi na pokretanje poslužitelja i klijenta te postavljanje njihovih nekoliko parametara na ispravne vrijednosti.

Daljnja poboljšavanja tuneliranja kreću se u smjeru razvoja kvalitetnijih posredničkih poslužitelja koji dozvoljavaju velike brzine prijenosa neovisne o korištenom protokolu i kodiranje ugrađenih podataka tako da ih vatrozidi i druge slične aplikacije ne mogu jednostavno analizirati i prepoznati. Budućnost postupaka tuneliranja, a posebice onih korištenjem ICMP i HTTP protokolâ, ponajviše ovisi o zlonamjernim korisnicima koji od toga imaju najveću korist pa je za očekivati daljnje napretke i usavršavanja protokola i pratećih aplikacija.

5. Reference

[1] Internet Control Message Protocol, http://en.wikipedia.org/wiki/Internet_Control_Message_Protocol, siječanj 2008.

[2] Tunneling protocol, http://en.wikipedia.org/wiki/Tunneling_protocol, siječanj 2008.

[3] Hypertext Transfer Protocol, http://en.wikipedia.org/wiki/HTTP, siječanj 2008.

[4] HTTP Tunnel, http://www.nocrew.org/software/httptunnel.html, listopad 2006.

[5] Ido Dubrawsky: Data Driven Attacks Using HTTP Tunneling, http://www.securityfocus.com/infocus/1793, kolovoz 2004.

[6] HTTP Tunnels, http://www.windowsecurity.com/articles/HTTP-Tunnels.html, siječanj 2006.

[7] HTTP Tunnel, http://en.wikipedia.org/wiki/HTTP_tunnel_(software), siječanj 2008.

[8] HTTP Tunnel, http://www.http-tunnel.com/html/solutions/http_tunnel/client.asp, 2006.

[9] Internet Control Message Protocol, http://en.wikipedia.org/wiki/Internet_Control_Message_Protocol, siječanj 2008.

[10] RFC Internet Control Message Protocol, http://www.faqs.org/rfcs/rfc792.html, rujan 1981.

[11] Hypertext Transfer Protocol, http://www.faqs.org/rfcs/rfc2616.html, lipanj 1999.