Uppgift i Internetworking för Chip Maguire.
Problemställning:

Uppgift: Design and evaluation of a TCP proxy which provides secure tunneling to another TCP proxy.

Problemet

Ett par av proxies ska sättas mellan en HTTP server respektive en HTTP klient för att skapa en säker tunnel mellan dem för överföringar. Den som lyssnar på trafiken skall antingen befinna säg på server sidans nätverk eller på nätverket där klienten befinner sig. Ett antagande görs att HTTP servern och datorn som proxyn ligger på är kopplade till samma nätverk och HTTP klienten och den närliggande proxyn ligger på samma nätverk.

Inledning

När nu Internet tenderar att bli allt större och större och man vet att informationen som skickas inte alltid skickas i säkra kanaler sk tunnlar kan det ses som som en nödvändighet att använda sig av sk proxies. Det finns två olika proxies den klassiska och den transparanta (RFC 1919)

Klassisk

- All kommunikation mellan klienten och omvärlden sker genom denna proxy. Klienten skickar information om vilka tjänster man vill komma åt på utsidan av nätet.
-
- Nätet skyddas av att proxyn maskerar IP-adresserna inne i nätet med sin egen.

Transparant

- Ligger som ett tyst filter mellan klientens privata nätverk och omvärlden.
- Klienten är inte medventen om proxyns existens.
- Trafik påväg ut ur nätverket fångas upp av proxyn och det egna nätets IP-adresser döljs genom att proxyn byter ut dessa med sin egen utan att klienten blir varse om detta.

I denna uppgift behandlas pga vissa skäl som visas i texten endast den transparanta.

Tunneln

En tunnel är en säker koppling mellan två noder som förbinder två eller flera nätverk, om nätverken har tunnlar mellan sig så upplever man nätverken som ihopkopplade , transporten i den säkra öveföringen i tunneln sker med antingen authentikering eller kryptering.

Uppgiften hos TCP-tunneln är att den ska kunna hantera HTTP-trafik mellan en HTTP-klient och en HTTP-server som ligger på skilda subnät i texten kallade subnätA och subnätB.

Kopplingen mellan näten ska vara säker. Om man vill föra över en HTML-sida från en server på ett subnätB till en webläsare ligger på subnätA. Klienten öppnar en end-to-end TCP anslutning mellan klient och server på de olika näten. För en utomstående betraktare så kommer det att se ut som om klienten och servern bara skickar trafik mellan sig och inte genom den krypterade tunneln. För att avlyssna trafiken är man tvungen att vara ansluten till något av subnäten. Om data skickas över en icke säker tunnel istället för i vårt fall så finns risken att privat information hamnar i fel händer då det skickas oskyddat över Internet och en avlyssning är möjlig. Att ha en kraftfull kryptering är en bra början då man använder sig av säkra tunnlar. Dock krävs en hel del extra bandbredd för att kryptera, enkapsulera och komprimera datat som skall överföras. Det beror också på vilken typ av data som skall överföras samt vilka finesser som skall finnas.

Routern konfigureras utifrån givna förutssätningar och för stöd av användandet av TCP-proxyn.
Trafiken leds in i tunneln genom att routern omvandlar datagrammen åt dessa TCP-proxies som därefter initierar en TCP-uppkoppling mellan dessa. . Det går till på detta sättet : Klienten börjar med att skicka iväg en HTTP request till servern, en förfrågan, och servern kommer därpå när den tagit emot förfrågan att svara med en HTTP response, ett svar. Om man lägger till en tunnel mellan server och klient så kommer inte det överförda innehållet i HTTP meddelandena att påvärkas/ändras utan alla meddelandena kommer att passera tunneln precis som om det var en direkt anslutning. Det som är intressant är hur själva trafiken i tunneln upprättas.

När TCP-uppkopplingen är klar börjar TCP-proxyn på subnätA att kryptera och överföra trafiken till subnätB:s proxy. Via TCP-uppkopplingen och TCP-proxyn på subnätB dekrypteras trafiken och vidarebefordrar den till mottagaren. Likaså kan TCP-proxyn på subnätB skicka trafik via samma tunnel tillbaka till subnätA.

HTTP använder sig av TCP för sin överföring av HTTP-request men i fallet med proxyn placerad innanför routern spelar detta ingen roll. All trafik som routas mellan de båda subnäten A&B kan slussas in i tunneln så länge de består av IP-trafik.
Alla TCP-paket är inkapslade IP-paket och för att få reda på vilken MAC-adress klienten skall sända sin trafik till börjar klienten med att broadcasta en ARP-request på nätverket som innehåller serverB:s IP-adress, sedan följer TCP-uppkoppling med trevägs handskakning:

Client -> Server - Trevägs handskakning

1. SYN segmentet specifierar vilket portnummer på servern som klienten vill koppla sig mot samt vilken ISN som clienten vill starta med.

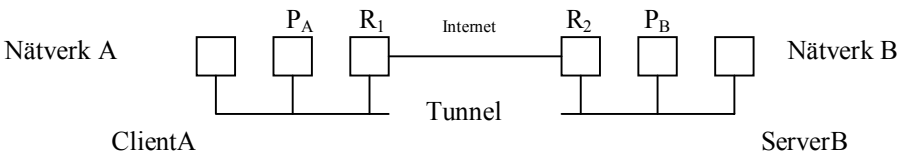
2. Servern svarar med ett SYN-segment som innehåller serverns ISN. Servern skickar även en ACK för servens SYN och ISN+1

3. Klienten skickar (som bekräftelse) iväg ett TCP segment, innehållande ACK, tillbaka till servern och bekräftar serverns SYN och ISN+1.

Nu är TCP-uppkopplingen klar att användas och client A kan börja överföringen av sin HTTP-request.
KlientA skickar med HTTP-request till TCP-uppkopplingen som delar den i TCP-paket av lämplig storlek samt paketerar de i IP-paket storlek och skickar över dem till serverB När hela HTTP-requesten mottagits av serverB påbörjar den överföringen av sin HTTP-response.

En transparant proxy är en proxy som inte 'ses' av någon. Med andra ord att användaren ser ingen skillnad på att tala direkt med en proxyserver jämfört med en riktigt server. Vad beträffar servern så kommunicerar den med en användare som använder en host på vilken proxyservern används. Servern vet inte att användaren befinner sig på annan plats. Användarens klientprogram pratar med proxyn istället för den riktiga servern ute på nätet. Proxyservern behandlar förfrågningar som kommer från klienten och bestämmer sig för vilka den ska skicka vidare och vilka som ska ignoreras.

Jag väljer att lösa uppgiften genom att göra en TCP-uppkoppling parallellt mellan proxy och en router.
Dvs proxyn är placerad inne på subnätet innanför routern.Alla trafik som går igenom routern måste då ej processas igen utan bara den som skall til ldet andra subnätet.
Se bild nedan.



Då ClientA vill upprätta en förbindelse med ServerB så måste man veta vilken MAC-adress datorn med IP-adress IP.B har. Detta görs genom att man skickar en ARP-request (förfrågan) till nätvärket för att ta reda på MAC-adressen till ServerB med IP adressen IP.B. ROUTER1 svarar med sin MAC adress och ClientA skickar paketet till ROUTER1.

Nu när det finns en uppkoppling mellan dator A och dator B så skall vi börja med att studera vad som händer i de båda nätverken och vad som händer i underliggande lager på TCP/IP stacken
Vi tittar på problemet genom att studera överföringen från en webclient på dator A till en webserver på dator B. Uppkopplingen får därför följande utseende:

ClientA-> R_1
$Router_1$ ⟶ $Proxy_A$
$Proxy_A$ ⟶ $Router_1$
$Router_1$ ⟶ $Router_2$
$Router_2$ ⟶ $Proxy_B$
$Proxy_B$ ⟶ ServerB

Sedan är det bara för dator B att skicka tillbaka svaret och det sker på samma sätt som ovan fast åt andra hållet. Mellan $Proxy_A$ och $Proxy_B$ sköts trafiken av en TCP uppkoppling som fungerar som en tunnel. Det finns på proxydatorerna en applikation som körs hela tiden och som har till uppgift att ligga och lyssna på den inkommande IP-kön och att hantera IP-paket som är adresserade till och kommer in till proxyn.

Routerns uppgift

Då $Router_1$ tar emot paketet kommer en jämförelse mellna B:s IP-adress och routingtabellen för $Router_1$. Routern går igenom sin tabell varje gång den skall skicka ett paket till den IP adress som angetts som destination för IP-paketet.

U- routern är aktiv/uppe
G- routern är till en gateway (router). Om flaggan inte är satt är destinationen direkt inkopplad mot routerns interface.
H- routen är till en host dvs destinationen är en komplett host-adress. Om routen inte är satt är routen till ett nätverk, och destinationen är en nätverksadress.

Routingtabell

Dest	**Gateway**	**Flags**	**Interface**
$Proxy_B$	$Router_2$	UGH	serial0
$Proxy_A$	$Proxy_A$	UH	eth0
$Nätv_B$	$Proxy_A$	U	eth0
$Nätv_A$	$Router_1$	UG	eth0
Default	$Router_2$	UG	serial0

Routingtabellen hos RouterA innehåller host-entries till både proxyA och proxyB, och routingtabellen hos routerB är i princip bara spegelvänd så när proxyA och proxyB vill utföra trevägshandskakningen, så utförs routingen av IP-paketen mellan dessa väldigt enkelt med ARP-request osv. $Proxy_A$ skickar sålunda sitt SYN-segment till proxyB genom $router_A$, som i sin routingtabell vidarebefordrar IP-paketet till routerB, som skickar det till $proxyn_B$. $Proxy_B$ och proxyA fullbordar sedan trevägshandskakningen enligt ovan.

U- routern är aktiv/uppe
G- routen är till en gateway (router). Om flaggan inte är satt är destinationen direkt inkopplad mot routerns interface.
H- routen är till en host dvs destinationen är en komplett host-adress. Om routen inte är satt är routen till ett nätverk, och destinationen är en nätverksadress.

Jämförelsen genomförs enligt följande schema.

1. En direkt jämförelse med alla rader i routingtabellen dvs mot alla hostdatorer som routern känner till.
2. B:s nätverks-ID jämförs med alla rader i routingtabellen som inte har H-flaggan satt, dvs alla nätverk som routern känner till.
3. Routern kommer att söka efter en default route som den kan skicka paketet till då det inte kan skickas någon annantans.

Här gäller att paketen vidarebefordras enligt rad 3.
Paketet kommer att skickas ut på ett ethernet till $Proxy_A$. Då skickas ett ICMP redirect till dator A för att tala om att i fortsättnigen skall den veta att alla paket som skall skickas till dator B skall skickas direkt till $Proxy_A$. ICMP (Internet Control Message Protocol) redirect skickas därför att man vill tala om att det finns en väg som är bättre nämligen den direkt till $Proxy_A$.

En applikation som körs på $Proxy_A$ har direkt åtkomst till den ingående IP-kön. Vi har ett paket vars adress finns i nätverk2 ,paketet skall krypteras och detta görs efter att applikationen har tagit ur paketet ur IP-kön. Hela paketet

IP huvud +TCP huvud +TCP data = PAKPT

kommer att krypteras. Det krypterade paketet benämns PAKPT. En ny TCP uppkoppling upprättas mellan slumpvis valda portar på ProxyA och ProxyB. Nu genomförs ännu en trevägshandskakning mellan ProxyA och ProxyB och efter denna lilla manöver så skickas PAKPT över till Router2. PAKPT kommer naturligtvis att innehålla initieringen av uppkopplingen mellan dator A och B. Med andra ord så kommer alltså PAKPT att innehålla ett TCP-huvud med SYN flaggan satt.

Router2 kommer att ta emot och ta hand om PAKPT den kommer att kolla i sin routingtabell och skicka PAKPT vidare till ProxyB som tar emot paketet och kollar att det stämmer med sin egen IP-adress → nu skickas paketet till applikationen som har hand om krypteringen. PAKPT avkrypteras nu direkt och sedan så sätts det ner i kön där utgående paket finns. Nu kommer ProxyB att skicka en ACK till ProxyA som en bekräftelse på att paketet har anlänt. Nu är det dags för paketet att skickas ut på nätverket där webserven B finns och den tar emot paketet och jämför det med sin egen IP-adress. Om det är så att adresserna stämmer så kommer IP-adressen att tas bort från dator A:s TCP paket och paketet kommer att komma fram till dator B:s TCP nivå

Nu befinner vi oss på TCP nivån och vad händer nu här? Paketet kommer att på TCP-nivån att tas om hand av rätt applikation men eftersom dator A börjar initieringen av TCP-uppkoplingen (se ovan) finns det ännu ingen uppkoppling utan det kommer istället ett svar från dator B med ett TCP-huvud där SYN och ACK flaggorna ar satta. Paketet fortsätter ner till IP-nivån. På denna nivå så kommer A:s IP-adress att läggas till framför TCP-huvudet. Nästa steg är att paketet skickas till länknivån och där kommer MAC adressen till Router2 att läggas till paketet. Detta eftersom dator B ännu inte känner till att paketen som ska till nätverkA kan skickas direkt till proxy2 Nu har paketet följaktligen följande utseende:

MAC(Router2) +IP-A + TCP huvud + paket.

$Router_2$ kommer att kolla i sin rountingtabell efter nätverkA och se att paketet skall till $Proxy_B$. För att detta skall genomföras skickas ett ICMP-direct till dator B för att meddela att paketet kan skickas direkt till $Proxy_B$ istället. Mellan $Proxy_A$ och $Proxy_B$ finns det redan en uppkoppling. Mellan $proxy_B$ och $proxy_A$ finns redan en TCP-uppkoppling och då kommmer $proxy_B$ att använda denna befintliga uppkoppling när den märker att paketet är avsett för nätverkA Detta gör att paketet som är avsett för nätverkA kan direkt tas ur IP-kön och krypteras. Paketet avsätts för att skickas till $Proxy_A$ med dess IP-adress.Läggs i TCP-sändbuffern för utgående trafik till proxyA. Nu tar $Proxy_A$ hand om paketet och kollar att det har rätt adress och att uppkopplingen ligger på rätt portnummer. Sedan skickas paketet till applikationen där paketet krypteras av och det sätts i IP-kön för utgående paket och sedan adresseras det till dator A.

I dator A så tas IP-huvudet bort från paketet och det skickas upp till TCP-nivån. På TCP nivån sätts ACK flaggan som en bekräftelse på att SYN-segmentet från dator B har anlänt. HTTP anropet läggs i utgående buffert hos dator A. TCP huvudet och TCP-data skickas till IP-nivån.
På IP-nivån läggs IP-adressen till dator B på och paketet befinner säg nu på länknivå nu läggs även MAC -adressen till $Proxy_A$ på paketet för ett ICMP direct har ju tidigare skickats från $Router_1$ till dator A. Dator B kommer att på samma sätt svara med ett HTTP respons.

Fördelar

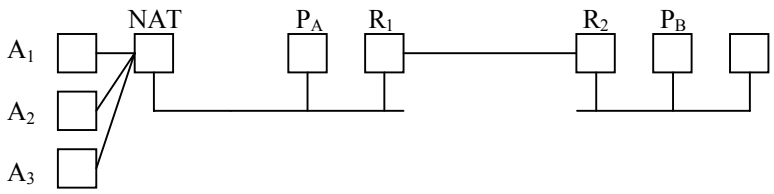
- Enkel installation. Ingen speciell router behövs och så är proxyn bara en vanlig dator.
- Det är inte nödvändigt för känslig trafik att lämna det säkra nätet innanför routern.
- Uppkopplingen kan använda sig av en slumpmässigt vald TCP-port som inte avslöjar någonting om vilken typ av data som överförs.
- Tillgänglig information är endast proxyarnas IP-adresser ,all annan data krypteras innan det överförs.
- Alla trafik mellan nätet sköts av en och samma TCP-uppkoppling.
- All IP-trafik klarar tunneln av att hantera.
- Svårt att knäcka krypteringen, eftersom man inte riktigt känner till den form av data som är krypterat.
- Lösningen är inte begränsad till att verka mellan två subnät.
- TCP kopplingen behöver ej avslutas när data tillfälligt är överförd. Vilket gör att uppkopplingstiden kan minimeras

Nackdelar

- Trafiken som färdas mellan subnäten kommer att belasta dem flera gånger då den färdas genom proxyn. En flaskhals i sammanhang rörande nätverk där nätverksresurserna är knappa eller där det finns mycket utgående trafik.
- Samtliga headers och hela paketet krypteras utan förminskning och en fragmentering blir därmed nödvändig sedan ursprungspaketet redan var av maxstorlek. Parterna kommer dock inte att märka fragmenteringen pga att datat kapslas in tidigt i överföringen.
- Då proxyn krypterar ett helt paket med samtliga headers utan att förminska det blir en fragmentering nödvändig eftersom ursprungspaketet redan var av maxstorlek. De inblandade kommunicerande parterna kommer dock inte att märka något av fragmenteringen pga att datat kapslas in relativt tidigt i överföringen.
- Cache-minnet i proxyn för buffring av de olika överföringarna bör vara stort.
- Eftersom proxyns IP-adress är synlig från utsidan kan detta komma att medföra att den utsätts för attacker.

NAT

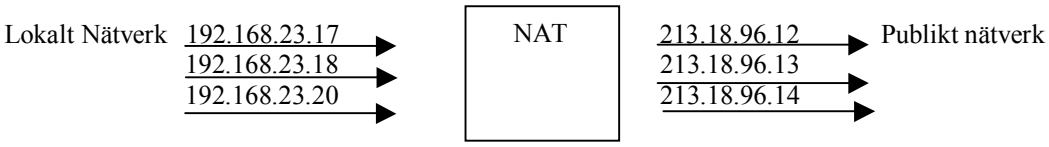
Modellen med klienten bakom en NAT



NAT, Network Address Translation hjälper till att skydda en mängd privata IP adresser så det enda användaren ser utifrån är NAT:en globalt publika adress. Iden är sedan att då ett paket går genom NAT-routern så kommer den att

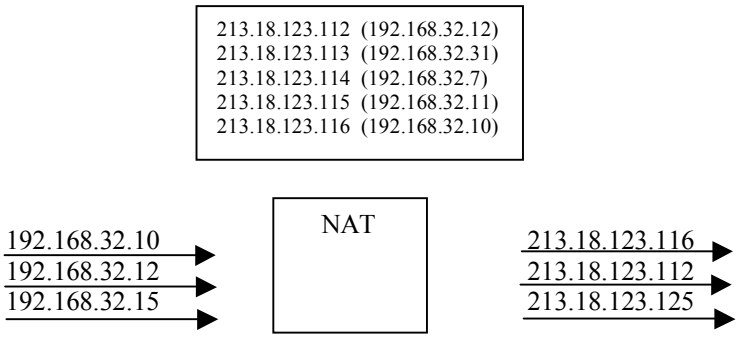
tilldelas en ny adress som är routerns publika adress. Denna adress ersätter clientens adress som source adress och då paketet kommer fram till sin destination så verkar det som om det skickades från routern. Då ett paket kommer tillbaka matchar routern adressen mot den riktiga ursprungliga adressen och paketet skickas vidare. Det finns 2 huvudtyper av NAT:

Statisk NAT: En mappning av oregistrerade IP adresser till registrerade IP adresser.



Vid statisk NAT så gäller att adressen 192.168.23.17 kommer alltid att översättas till 213.18.96.12

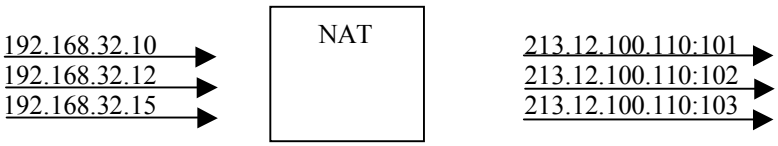
Dynamisk NAT: Mappar en oregistrerad adress till en registrerad från en pool av registrerade adresser.



Datorn med IP adressen 192.168.32.10 översätts till den första lediga adressen mellan 213.18.123.100 och 213.18.123.150.

Förutom vid statisk NAT så behöver vi vid dynamisk NAT lagra och överblicka information om klienterna som använder systemet, så tex om en host har varit avstängd en längre tid eller slutat sända paket en nedkoppling skett eller att adressen är borttagen från NAT:ens pool, en timeout har skett, så att den är redo för återanvändning.

Finns även en form av dynamisk NAT som tittar på portarna istället, här kallad PAT, Port address translation. Då översätts de oregistrerade adresserna till samma nya publika adress men tilldelas olika portnummer.



Vid användandet av en NAT-device, Network Address Translation, kommer de privata adresserna på det lokala nätverket att skyddas utåt genom att alla paket som skickas genom NAT:en kommer att tilldelelas en ny IP-adress hämtad från en pool av adresser i NAT:en. Om NAT-tabellen är stor kanske en genomletning av en matchande adress kan komma att ta lång tid. Då ett nytt paket anländer till NAT:en sker en mappning och paketet skickas vidare till den privata adress som den publika adressen utifrån NAT:en motsvarar. Förutom en modifikation av IP adressen så ska NAT även modifiera IP checksum.

Det finns 5 saker som unikt identifierar en uppkoppling då NAT är med i bilden. Protocol,source IP och port,Destinations IP och port. Vi vet också att NAT routern måste lagra en hel del information om uppkopplingen och hur den ska översätta adresserna. Det finns alltså ett samband mellan dem och en bradvägg eftersom den (NAT:en) också måste kolla vilka paket som ska transporteras till ‘andra’ sidan. De måste lagra tillståndet hos paketet och detta ger mer overhead.

Då det i vårt problem existerar en tunnel mellan proxyarna och kryptering sker i desamma kommer inte NAT:en att ställa till några problem med tanke på krypteringen. Om man däremot låter en applikation i klienten sköta krypteringen och låter ett paket med krypterad source adressen komma till NAT:en kommer detta att droppas då där inte sker någon avkryptering och NAT:en inte kan tolka den krypterade source adressen och inte heller adressen var paketet ska någonstans och följden blir att paketet även här droppas. Vid ändring av IP adresserna så måste även NAT modifiera IP checksum och TCP checksum. NAT ska även kolla ICPM meddelandena och ändra fältet där ursprungs IP adressen finns.

Om NAT droppar eller skickar vidare ett ICMP freagment beror på en hel del saker som tex i vilken ordning NAT:en tar emot fragmenten och i vilket tillstång som translation table befonner sig i. Under speciella förhållanden så översätter NAT:en fragmenten olika och gör det omöjligt för destinationens maskin att sätta ihop paketet igen.Inte någon applikation som bär på IP-adressen inne i en applikation kommer att fungera med NAT om inte NAT känner till dessa och genomför nödvändiga översättningar.

ICMP meddelanden

Bara fragment 0 i datagrammet innehåller ICMP headern. För att bestämma om flera flera fragment är delar av ett paket så måste NAT kolla på ett sk IP ident value, som finns i IP headern i alla fragment av det ursprungliga IP-datagrammet. Om flera fragment har samma IP ident value som fragment 0, som skapade översättingen och matchningen så kommer NAT att behandla dessa fragment lika. Om ett annat fragment än det första kommer först så skapar NAT:en en enkel översättning så länge som det finns en ledig adress i poolen. Om NAT tar emot ett nytt paket som har ett annat IP ident value och som inte matchar någon existerande översättning, eller om det inte finns några lediga adresser i tabellen att tillgå och att det inte finns något IP ident value att skapa en ny matchning med så droppas helt enkelt paketet och följden blir att alla paket som skickades inte kommer fram och därmed genereras inget svar.

NAT är inte alltid den transparanta process som den borde vara. Allt skulle vara bra om IP var det enda protokollet som bär på IP-adress informationen men så är inte fallet. Det finns några protokoll som sänder IP som del av datat och om detta översätts av en NAT-router så innebär det problem. Alltså ett problem för mottagaren eftersom den inte kan nå hosten som IP-datat som skickats skulle till. Det enda sättet att lösa detta är att titta på speciella protokoll som inkluderar IP-informationen och detta resulterar i ökad overhead och komplicerar förfarandet vid översättningen.
I fallet med proxyuppkopplingen gäller detta ICMP meddelandena som skickas. ICMP inkluderar,beroende på typen av meddelande som skickas, en del av headern från orgoinalpaketet, (8 bitar) som genererade ICMP meddelandet och hela IP headern för det paketet. Om paketet översätts kommer denna header att innehålla NAT-IP istället för den lokala IP adressen till hosten som får ICMP meddelandet. Beroende på hur denna extra header information används så kan det komma att skapa ett problem.

Problem med NAT

Det är ofta önskvärt att ta ett beslut om ett paket ska översättas som inte bara beror på IP adressen utan även fokuserar på TCP adressen, nämligen vilket port nummer. Problemet är det att så snart ett paket är fragmenterat så kan NAT routern inte känna igen vilken port det tillhör undantaget det första fregmentet som tillhör TCP headern. På grund av detta måste fragmentets tillståndsinformationen lagras. Första fragmentets TCP port lagras för att sedan kunna tala om vart de efterföljande fragmenten skall skickas. Detta är dock kanske inte tillräckligt eftersom IP inte garanterar att paketen anländeri rätt ordning. Dock borde det inte ske ofta att paketen anläder i fel ordning om inte deras väg över Internet hindras vid tex avbrott etc. Då paketen tar samma väg över de fåtal routrar de passerar ska inte några komplikationer uppkomma. För tex om det tredje fragmentet i ett fragmenterat paket kanske passerar NAT:en först innan fragmentet som bär på port informationen. I det här fallet är det enda som går att göra att hålla kvar fragmenten till det första anländer så vi vet om en översättingen av paketet måste göras eller inte.

Så länge som den privata hosten har en utgående uppkoppling så kan den nås av inkommande paket sända till den publika adressen. Å andra sidan om en attack kommer bakifrån från det privata närverket så är det svårt att ta reda på den exakta användaren.