TCP hijacking

Vladimír Kotal <vlada@devnull.cz>

30. července 2004

Zadání 1 *TCP hijacking:*
Popište podrobně metodu útoku unesením TCP spojení. Uveďte slabiny na kterých je útok založen a příklady realizace.

1 Úvod

TCP protokol jako jedna z hlavních součástí suity protokolů TCP/IP slouží pro přenos informací na Internetu již více než 20 let. TCP tvoří spolu s protokolem UDP 4. vrstvu modelu ISO/OSI. Dnes se protokol TCP používá např. k přenosu dat pro aplikační protokol HTTP, ale také pro udržování BGP spojení mezi páteřními směrovači (routery), které slouží k výměně směrovacích informací.

TCP tedy představuje přenosovou (neboli transportní) platformu pro protokoly vyšších vrstev ISO/OSI modelu. Některé z těchto protokolů mohou mít vyšší nároky na bezpečnost, což jim TCP nemusí vždy zaručit. V tomto článku uvedu přehled útoků na protokol TCP a rozeberu jeden z útoků - útok *TCP hijacking* ("unesení TCP spojení"). Dále nastíním některé metody ochrany proti podobným útokům.

2 Úvod do protokolu TCP

TCP je stavový transportní protokol, který byl navržen pro spolehlivý přenos dat po Internetu. TCP garantuje integritu a sekvenční doručování dat mezi dvěma entitami poté, co bylo ustaveno TCP spojení. TCP velmi dobře škáluje - pracuje uspokojivě i na linkách s vysokou latencí nebo vysokou ztrátovostí paketů.

TCP spojení lze rozdělit do tří fází - ustavení (synchronizace), přenos dat, uzavření spojení.

2.1 Formát TCP hlavičky

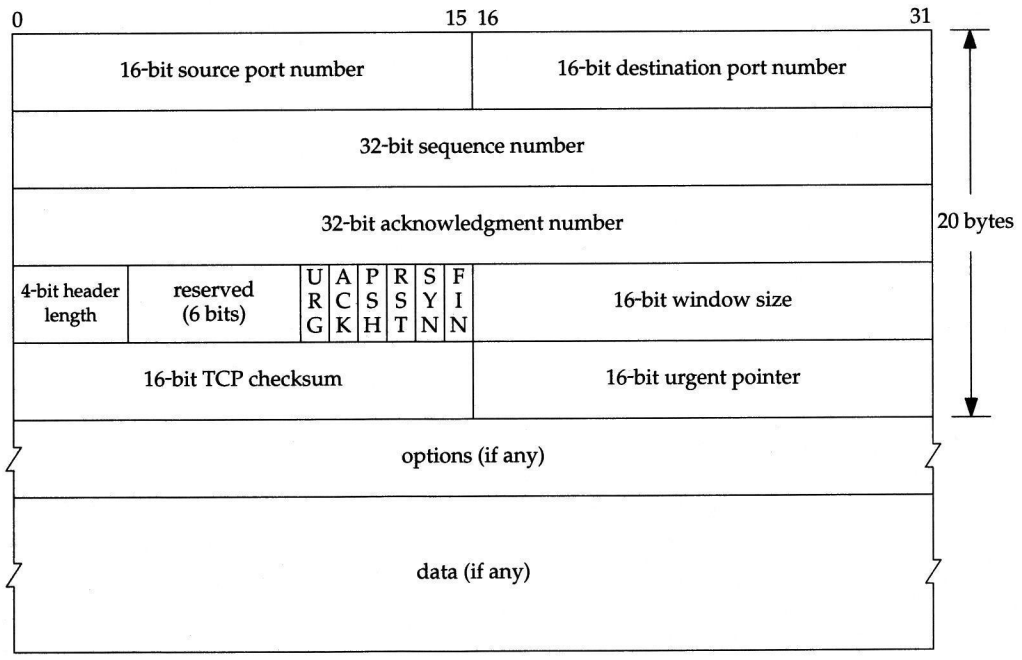
Hlavička TCP paketu dle [4] je na obr. 1.

TCP hlavička následuje v paketu typicky bezprostředně za IP hlavičkou[1].

Pro naše účely jsou důležité zejména následující položky:

- *Flags* (`URG`, `ACK`, ...) - pole s příznaky nese informaci potřebnou pro detekci stavu druhé strany. Lze je kombinovat, ale pouze některé kombinace jsou dle RFC (viz. [4]) povolené. Paket obsahující flag `ACK` budeme nazývat `ACK` paket.
- *Sequence number* - 32-bitové číslo sloužící pro zaručení sekvenčního doručování dat. Každý TCP segment poslaný po síti obsahuje sekvenční číslo prvního bajtu dat v tomto segmentu.

[1] výjimkou je např. použití protokolu IPsec



Obrázek 1: TCP hlavička. Pokud není délka pole 'Options' násobkem osmi, je doplněna. Převzato z [5].

- *Acknowledgment number* - 32-bitové číslo sloužící pro zajištění spolehlivosti doručení. Úspěšně přijatý segment je odesilateli potvrzen posláním `ACK` paketu, který má jako *Acknowledgment number* sekvenční číslo posledního bajtu přijatého segmentu.
- *Window* - udává velikost "okna" - tj. maximálně kolik dat může druhá strana poslat

2.2 Model TCP protokolu

TCP protokol je možné modelovat jako konečný statový automat (viz. obr. 2), ve kterém se vyskytují stavy klienta i serveru.

Celkem má stavový automat 11 stavů, pro naše účely[2] ho můžeme zjednodušit na automat se sedmi stavy: `CLOSED`, `LISTEN`, `SYN_RCVD`, `SYN_SENT`, `ESTABLISHED`, `active close`, `passive close`. Procesy uzavření spojení jsme každý uzavřeli do jednoho "podautomatu" a provedli substituci.

2.3 3-way handshake

TCP je stavový protokol. Předtím, než může začít výměna dat mezi oběma stranami, je potřeba synchronizovat stavové automaty na obou stranách. Jednak je třeba, aby se obě strany dostaly konzistentně do stavu `ESTABLISHED` (tento přechod musí být odolný vůči výpadkům paketů po cestě), dále tato synchronizační fáze slouží k vzájemné výměně počátečních sekvenčních čísel (ISN, Initial Sequence Number) a počátečních velikostí oken TCP spojení. Toto zajišťuje přechod nazvaný *3-way handshake*.

[2] zajímá nás zejména ustavené TCP spojení, pro mechanismus ukončení nám postačí vedět, že souvisí s příznakem `RST`

Na počátku je server ve stavu `LISTEN`, klient ve stavu `CLOSED`. *3-way handshake* probíhá tak, že klient zvolí počáteční sekvenční číslo (ISN)[3] a pošle serveru TCP paket s příznakem `SYN`, čímž přejde do stavu `SYN_SENT`. Server po obdržení tohoto paketu přejde do stavu `SYN_RCVD` a odpoví TCP paketem s příznaky `SYN/ACK`. Klient obdrží tento paket a odpoví TCP paketem s příznakem `ACK` a oba přejdou do stavu `ESTABLISHED`.

2.4 Implementace TCP

V současnosti se implementace TCP nachází v široké řadě různých zařízení, od počítačů přes průmyslové systémy až po různá tzv. *embedded* zařízení, která slouží jenom úzkému oboru činností.

V minulosti byl protokol TCP co se návrhu týče, minimálně upravován. Jeho implementace se nicméně celkem mezi jednotlivými operačními systémy liší. Odlišnosti spočívají zejména v

1. generování čísel ISN
2. chování při přijetí out-of-window TCP paketů s různými volbami
3. chování při přijetí TCP paketů s různými volbami

Nedostatky při generování čísel ISN umožňují tzv. *blind-spoofing* útoky (viz. dále), chování při přijetí out-of-window TCP paketů může umožnit TCP reset útoky nebo TCP hijacking (viz. dále). (podle voleb v TCP paketu)

Různé chování operačních systémů při přijetí TCP paketů s různými volbami je podkladem pro vzdálenou detekci OS.

3 Slabiny protokolu TCP

Protokol TCP byl navržen pro prostředí ARPANETu, kdy ještě nebylo zřejmé, jak nehostinné místo může být Internet. Z toho vyplývají jeho slabiny, které mohou být násobeny nesprávnou implementací.

3.1 Predikovatelná čísla ISN

Generování čísel ISN (Initial Sequence Number) pro každou instanci TCP spojení vyžaduje, aby tato čísla byla co nejméně predikovatelná. Toto není ani tak chyba návrhu (i když v příslušném RFC (viz. [4]) to mohlo být lépe zdůrazněno), jako spíše implementací TCP.

Pokud totiž daný operační systém implementuje generátor čísel ISN tak, že jeho výsledky nejsou dostatečně náhodně rozptýleny po stavovém prostoru, je možné provést snadno tzv. *blind-spoofing* útok (viz. dále).

Z přehledné studie, o tom, jak si stojí jednotlivé operační systémy (viz. [1]) vyplývá, že implementace generátorů ISN čísel se postupně zlepšují.

3.2 Chybějící autentizace

Jednotlivé segmenty TCP protokolu nejsou mezi jednotlivými stranami nijak autentizovány, takže pokud je možné podvrhovat pakety, je zároveň možné podvrhovat data v rámci TCP spojení. Kdyby toto nebylo možné, útoky typu TCP hijacking by nebyly uskutečnitelné.

Tento nedostatek lze řešit na vyšších vrstvách ISO/OSI modelu nebo pomocí technologie IPsec (viz. sekce "Metody obrany").

[3] pro každé nové spojení pokud možno náhodné

4 Útoky na TCP

Předtím, než podrobně rozebereme TCP hijacking útok, uvedeme několik nejznámějších útoků na protokol TCP pro lepší kontext.

4.1 Spoofing

Spoofing neboli podvrhování paketů je jednoduchá technika, která zkonstruuje paket s danou zdrojovou nebo cílovou adresu v IP hlavičce, zdrojový a cílový port v TCP hlavičce a dalšími poli.

Pokud jsou čísla ISN predikovatelná, může být proveden tzv. *blind spoofing* útok, kdy útočník nemusí odposlouchávat spojení, protože může odhadnout (s jistou pravděpodobností), jaké jsou aktuální hodnoty sekvenčních čísel obou stran.

Spoofing sám o sobě nepředstavuje typ útoku, je to jen komponenta využívaná při TCP hijacking útoku a dalších útocích.

4.2 TCP reset

TCP reset útok využívá nedostatečně striktní implementace TCP protokolu, kde `RST` paket poslaný v rámci okna může resetovat celé TCP spojení, není nutné, aby sekvenční číslo sědělo přesně (viz. [9]). Protože jsou dnes okna pro TCP spojení nezřídka velká, není toto riziko tak malé, jak se původně myslelo. Protože touto chybou byla ohrožena stabilita páteře Internetu[4], muselo být upgradováno v krátkém čas množství směrovačů. Opět, spoofing je komponentou pro tento útok.

4.3 SYN flooding

SYN flooding je útok proti implementaci TCP protokolu využívající spoofingu. Na cíl jsou posílány TCP `SYN` pakety s podvrženou zdrojovou adresou, cíl alokuje paměť pro nové spojení a odpoví `SYN/ACK` paketem. Pokud posílá útočník pakety dostatečně rychle, může tak dojít k vyčerpání systémových prostředků oběti.

4.4 Hijacking

Hijacking útoky obecně jsou všechny takové útoky, kdy útočník může za určitých podmínek ovlivnit stav cíle pomocí manipulace s protokoly, které využívá jeho cíl ke komunikaci. Tyto útoky se mohou odehrávat na různých vrstvách ISO/OSI modelu, známé jsou např. hijacking na 2. vrstvě (*ARP spoofing*, *ARP relaying*) a na 3. a 4. vrstvě ISO/OSI modelu. (*TCP hijacking*, *ICMP redirection*)

Útok *TCP hijacking* ("unesení TCP spojení") má za cíl ovládnutí existujícího client-server nebo peer-to-peer spojení na 4. vrstvě ISO/OSI modelu pomocí vložení TCP paketů.

Tento útok bude podrobně popsán v následujících sekcích.

5 Podstata TCP hijacking útoku

TCP hijacking útok využívá v podstatě jen jedné slabiny TCP protokolu: jednotlivé segmenty TCP spojení nejsou autentizovány. K "úspěšnému" provedení útoku (tj. takovému, kdy ani jedna strana pokud možno nepozná, že byl útok proveden) jsou nicméně třeba další předpoklady:

[4] TCP spojení se používají pro výměnu směrovacích informací protokolu BGP

- možnost tzv. *duální desynchronizace* umožňující tzv. *ACK storm*
- možnost úspěšné predikce čísel ISN (*Initial Sequence Number*) nebo pozice pro tzv. *Man in the Middle attack*[5]

5.1 Duální desynchronizace a ACK storm

Podle [2] si můžeme přenos dat v rámci ustaveného TCP spojení (tj. po proběhnutí *3-way handshake*) představovat jako klasickou počítačovou hru PONG, kde paket "na cestě" představuje letící míček a posunující se okna TCP spojení představují pálky odrážející míček. Pokud míček spadne na pálku (tedy TCP segment patří do okna), odpoví tato strana paketem s nastaveným příznakem ACK. Pokud je navíc tento TCP segment poslední v rámci daného okna, okno se posune o svou velikost dopředu. Pokud míček dopadne mimo pálku (tzv. *out-of-window* segment), vyšle tato strana paket s příznakem ACK, který potvrzuje poslední paket v rámci okna. To umožňuje detekovat situaci, kdy se nějaký paket po cestě ztratil.

Toto chování je zneužitelné. V případě, kdy se podaří dosáhnout stavu, kdy si obě strany pošlou out-of-window segment, bude se toto chování opakovat až do chvíle, kdy se ztratí jeden paket tohoto typu. To může nastat záhy, protože tyto pakety mohou zahltit linku.

V případě TCP hijacking útoku je žádoucí dosáhnout posunutí oken obou stran (odtud *dual desynchronization*). Ve chvíli, kdy mají obě strany uměle posunutá okna, stačí poslat podvržené out-of-windows pakety na obě strany a tyto se začnou opakovat. Pokud se jeden z nich ztratí (např. v důsledku ucpání linky), výměna těchto paketů se na chvíli zastaví. Opět se ale rozeběhne, jakmile se jedna ze stran rozhodne poslat nějká data. Out-of-window pakety se pak budou opakovat tak dlouho, dokud jedna ze stran nepřeruší spojení pomocí paketu s příznakem RST. Tomuto chování se říká *ACK storm*.

Chování při ACK storm může být použito jako krytí pro TCP hijacking útok.

5.2 Vkládání TCP segmentů

Nejjednodušší varianta TCP hijacking útoku (tzv. *simplex hijacking*) provede pouhé vložení paketu s podvrženými daty. K tomu je nejprve nutné dané spojení sledovat, aby mohl být zkonstruován se správnými *Seq* a *Ack* čísly. V důsledku poslání podvrženého paketu může a nemusí nastat ACK storm[6].

6 Realizace útoku

Pro realizaci TCP hijacking útoku je nejprve nutné vybrat spojení, na které má být útok proveden. Poté je možné vložit jeden TCP segment (tzv. *simplex hijack*). Tím může dojít k ACK storm. Před vložením TCP segmentu je vhodné provést duální desynchronizaci stran účastnících se sledovaného spojení. Lze provést i komplikovanější hijack útok tak, že neprovedeme duální desynchronizaci a po vložení segmentu počkáme, jestli se objeví ACK storm a pokud ne, můžeme vložit další segment, resetovat spojení nebo jej synchronizovat.

Pokud dojde k ACK storm, je další posílání podvržených paketů téměř nemožné, protože ACK pakety zahltí dostupnou šířku pásma.

6.1 Nástroje

Nejznámějšími nástroji pro implementaci TCP hijacking útoku jsou Juggernaut [3] a Hunt [6].

[5] útočník má možnost odposlouchávat a ovlivňovat síťový provoz obou stran spojení

[6] Např. v dokumentaci k [6] je uvedeno, že ACK storm nenastává u linuxových jader řady 2.0

6.1.1 Juggernaut

Juggernaut je klasický nástroj uveřejněný v době, kdy byly hijacking útoky už nějakou dobu "populární".

Tento nástroj umí provést základní hijacking útok vložením TCP segmentu do telnet spojení. Při *simplex hijacking* útoku neprovede nejprve desynchronizaci spojení, takže výsledný ACK storm nemusí nastat. Toto sice může způsobit ACK storm na mnoha klientech, ale není to opravdová duální desynchronizace. Navíc příkaz v podvrženém paketu je vypsán telnet serverem, takže jej pravý klient vidí.

Autoři článku [2] modifikovali Juggernaut tak, že nejdříve provedl duální desynchronizaci pomocí zaslání podvržených NOP (No operation) paketů telnet protokolu (viz. [8]) na obě strany spojení. To posune okna obou stran bez toho, že by se změnil stav aplikací. Vložením podvržených dat do ustaveného spojení pak vyvolá ACK storm, takže pravý klient nevidí, že byla do spojení vložena data.

Autor tohoto nástroje pracuje v současné době pro firmu Cisco systems[7].

6.1.2 Hunt

Hunt je moderní nástroj pro manipulaci se síťovým provozem od 2. vrstvy ISO/OSI výše.

Autor nástroje Hunt pracoval ve firmě zabývající se mj. vývojem profesionálních průmyslových přístrojů pro analýzu různých protokolů. Proto není divu, že Hunt obsahuje komfortní prostředí pro práci se síťovým provozem.

Hunt je program určený pro manipulaci s TCP spojením. Může jej sledovat, narušovat a resetovat. Umí filtrovat spojení, detekovat spojení, které je už ustaveno[8], aktivní hijacking útok s detekcí ACK storm, hijacking útok s ARP spoofingem, resetovat spojení.
Je napsán modulárně, obsahuje démony[9] pro sledování MAC adres, sniffovacího démona pro zaznamenávání síťového provozu se schopností hledat v provozu určitý řetězec znaků atd.

Co se týče implementace TCP hijacking útoku, je Hunt pravděpodobně nejvyspělejším volně dostupným nástrojem. Umí *simple hijack* útok s detekcí ACK storm i TCP hijacking útok se synchronizací obou stran.

7 Metody obrany

Proti hijacking útokům všeobecně se lze bránit zavedením autentizace jednotlivých TCP segmentů, např. pomocí technologie IPsec. Riziko snižují i tzv. *best current practices* neboli doporučení pro konfiguraci a správu síťových zařízení. Mezi tyto doporučení patří například ochrana proti spoofing útokům zavedením *access listů*.

7.1 Přidání stavu

Jedním z řešením je úprava stavového automatu TCP protokolu (viz. obr. 2) tak, aby obsahoval stav, který pomůže detekovat duální desynchronizaci a provede případnou resynchronizaci obou stran. Duální desynchronizaci lze detekovat tak, že po příjmu *out-of-window* segmentu se pošle `ACK` paket bez dat a automat přejde ze stavu `ESTABLISHED` do tohoto nového stavu. Pokud nyní přijde další out-of-window `ACK` paket bez dat, víme, že došlo k duální desynchronizaci.

Tento přístup (viz. [2]) ovšem není lehké zavést do praxe. Prostředky k resynchronizaci (3-way handshake) jsou k dispozici, musí být ovšem zachována zpětná kompatibilita s implementacemi TCP stacku, které neumí

[7] což svědčí o tom, že hry s protokoly mohou být užitečné
[8] tedy nejenom spojení, u kterých zaznamená 3-way handshake
[9] vlastně vlákna, protože Hunt je aplikace napsaná pomocí implementace POSIX threads

duální desynchronizaci detekovat. Pokud jedna ze stran neobsahuje stack, který umí detekovat duální desynchronizaci, musí být spojení zrušeno[10].

V praxi se tento přístup ukázal jako navyhovující, protože nelze garantovat, že přijatý ACK paket bez dat byl vyvolán určitým vyslaným ACK paketem bez dat. V článku [2] byl podán důkaz, že může dojít k tomu, kdy spojení mezi dvěma stranami (oběma schopnými detekce duální desynchronizace) může být zresetováno díky zpoždění při doručení paketu. Řešení této chyby by spočívalo v použití pole 'Options' (viz. obr. 1) pro rozpoznání ACK paketu náležejícího k vyslanému ACK paketu, to by s sebou ale neslo problémy v podobě sníženého výkonu[11], navíc některé implementace nemusí přijmout TCP paket s polem 'options', které nezná.

Další možností by bylo zjistit možnosti svého protějšku při sestavování spojení. To je cesta, kterou využívá koncept SACK (Selective ACKnowledgments, viz. [7]).

7.2 Antispoof pravidla

Současné implementace firewallů v různých operačních systémech dovolují přidat pravidla, která zaručují jistý stupěň obrany proti podvrhování paketů. Máme-li např. router s dvěma síťovými rozhraními, kdy jedno z nich má přiřazeny routovatelné adresy a druhé je zapojeno do vnitřní sítě s privátními adresami, můžeme pomocí pravidel firewallu zakázat příjem paketů se zdrojovou adresou patřící do vnitřní sítě přicházejících přes vnější rozhraní.

Následujících několik příkladů uvádí pravidla pro obranu proti podvrhování adres na několika implementacích firewallů.

Tato pravidla pomohou v případě, že útočník podvrhuje adresy patřící do vnitřní sítě a přitom do ní nemá přístup[12]. Pokud by se útočník nacházel ve vnitřní síti a neoprávněně by používal adresy z této sítě, tato ochrana tomu nezabrání. Antispoof pravidla tedy spíše slouží jako jakási forma prevence.

- PF

  PF je populární implementace firewallu dostupná na OS OpenBSD, NetBSD a FreeBSD. K jeho vlastnostem patří zejména přehlednost a snadnost konfigurace (oproti implementacím firewallů v OS Linux), integrace s ALTQ mechanismy, škálovatelnost, integrovaná podpora pro fail-over mechanismy a mnoho promyšlených detailů. Jedním z těchto ne nevýznamných detailů je i klíčové slovo `antispoof`, které zakáže příchozí provoz se zdrojovými adresami z neroutovatelných bloků (dle IANA).

  Máme-li vnější (s adresou a přístupem do sítě s routovatelnými adresami) síťové rozhraní `fxp0`, pak ochranu proti podvržení provedeme jednoduše pomocí

  ```
  ExtIF="fxp0"

  antispoof for $ExtIF
  ```

  Pokud se za nějakým dalším rozhhraním nachází síť s neveřejnými adresami, je tímto provedena základní ochrana. Pokud se naopak za dalším síťovým rozhraním nachází síť s routovatelnými adresami, budeme muset přidat ještě další pravidla zakazující použití těchto adres jako zdrojových v příchozím provozu via vnější síťové rozhraní a jiných adres než z tohoto bloku pro odchozí provoz via toto síťové rozhraní.

[10] namísto resynchronizace v případě, kdy obě strany detekci umí

[11] routery zpracovávají zpravidla pakety s nestanndardními 'options' v procesoru namísto v hardware

[12] tím se myslí jak odesílání paketů z Internetu do vnitřní sítě se zdrojovou adresou z vnitřní sítě, tak odesílání paketů z vnitřní sítě se zdrojovou adresou nepatřící do této sítě

- IPF

  IPF je další populární implementací firewallu, kterou lze najít v NetBSD a FreeBSD [13] , IRIXu (narozdíl od BSD systému je nutné ho ručně přidat) a dalších UNIXových systémech.

  IPF nemá makrojazyk, je tedy nutné vyjmenovat všechny adresy/adresní bloky, jichž se mají *antispoof* pravidla týkat.

```
# fxp0 - external interface

# don't allow anyone to spoof non-routeable addresses
block in log quick on fxp0 from 10.0.0.0/8 to any
...
```

- Cisco extended ACL

  Směrovače Cisco dnes představují jednu z nejpoužívanějších přenosových platforem pro páteře a tzv. hraniční směrovače (border routers) ISP (Internet Service Provider). Právě tato zařízení by měla poskytovat ochranu proti podvrhování paketů. Na každém rozhraní, za kterým je připojena síť přijímající nebo odesílající pakety "zvějšku", by měla mít v access listu alespoň pravidla pro ochranu proti podvržení paketů. Tím se myslí jak ochrana proti paketům přicházející z tohoto rozhraní (ingress směr), tak paketů vycházejících z této sítě (outgress směr).

  Mějme hypotetickou síť `195.113.1.0/24`, která je součástí virtuální sítě. Ochranu proti podvržení adres provedeme na routeru (zde layer-3 switchi), do kterého jsou zařízení s adresami z této sítě připojena, pomocí následujícího kusu konfigurace tohoto routeru:

```
interface Vlan150
  description Vlan with routeable addresses
  ip address 195.113.1.1 255.255.255.0
  ! extended ACLs for antispoof protection
  ip access-group 151 in
  ip access-group 150 out
!
! ingress filter for Vlan 150
access-list 150 remark ANTISPOOF-FOR-INGRESS-TRAFFIC
access-list 150 deny ip 195.113.1.0 0.0.0.255 any log
access-list 150 permit ip any any
!
! outgress filter for Vlan 150
access-list 151 remark ANTISPOOF-FOR-OUTGRESS-TRAFFIC
access-list 151 deny ip 195.113.1.0 0.0.0.255 any log
access-list 151 permit ip any any
```

7.3 Autentizace na vyšších vrstvách ISO/OSI

Použitím autentizace (např. pomocí protokolu SSL) lze dosáhnout docela dobrého stupně ochrany. Pakety, které nejsou autentizovány, jsou sice zpracovány operačním systémem a tudíž může dojít k ACK storm, ale nejsou dále předány aplikaci.

[13] z OpenBSD byl IPF odstraněn po neshodách způsobených licencí IPF

7.4 IPsec

IPsec je ochrana na nižších vrstvách ISO/OSI - může zapouzdřovat celý TCP paket, a tím chránit proti celé řadě útoků vedených proti TCP. IPsec nicméně není příliš rozšířen, masověji se využívá zejména pro ochranu korporátních VPN sítí, zejména kvůli náročnosti na výkon směrovačů a konfiguraci, která je spíše statického charakteru.

8 Závěr

TCP hijacking útoky stále představují bezpečnostní riziko. Toto riziko je nutné v procesu realizace bezpečnostní politiky zvážit a podniknout příslušné kroky pro jeho omezení resp. eliminaci.

Reference

[1] M.Zalewski. *Strange attractors and tcp/ip sequence number analysis*, `http://lcamtuf.coredump.cx/newtcp/`, 2001

[2] David Anderson, Brian Teague. *ACK Storms and TCP Hijacking*

[3] route. *Juggernaut*, Phrack 50, article 6

[4] J. Postel. *RFC 793: Transmission control protocol*, Sept. 1981

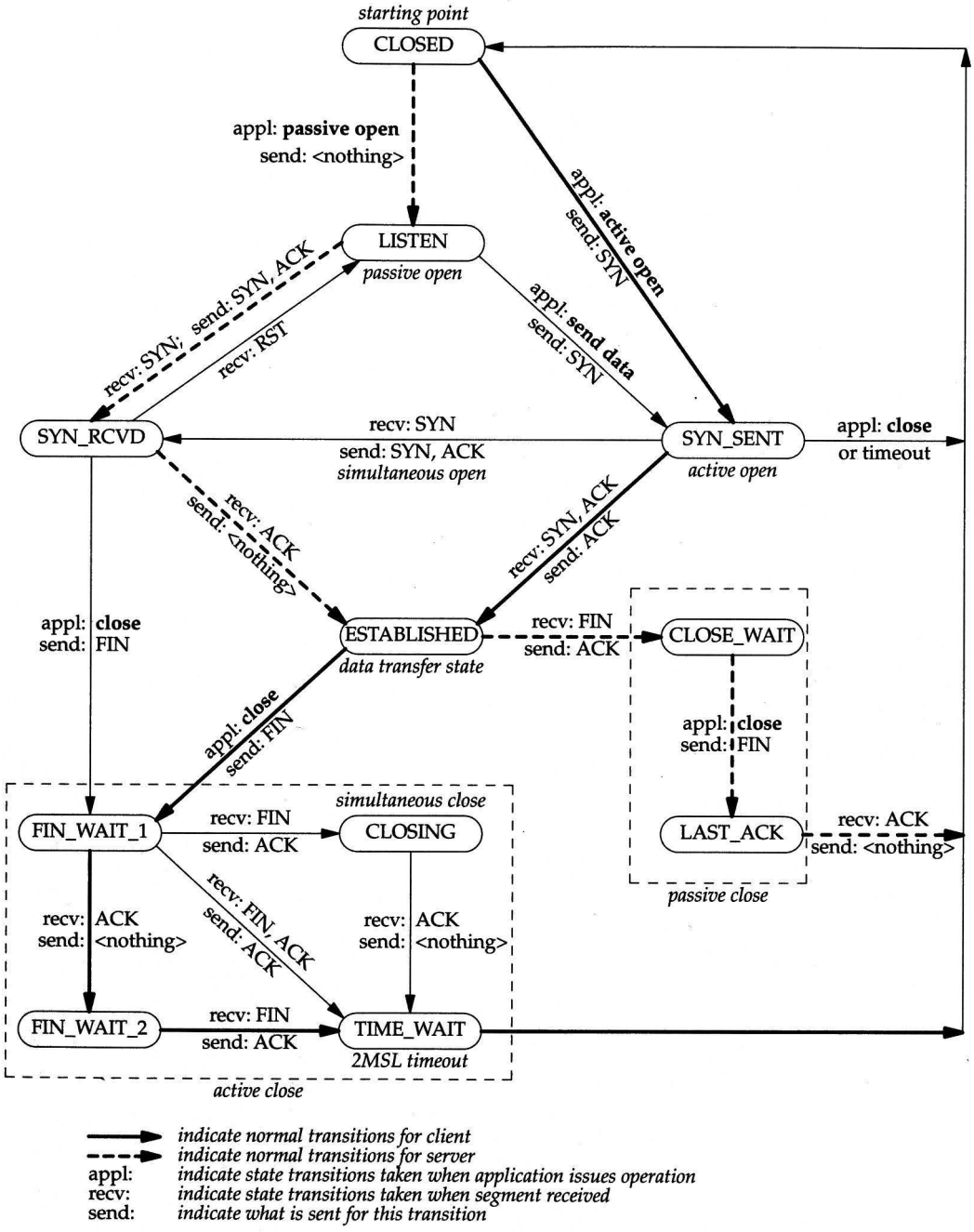
[5] W. R. Stevens. *TCP/IP Illustrated*, volume 1. Addison-Wesley, Jan. 1994

[6] Pavel Krauz. *Hunt*, `http://packetstormsecurity.nl/sniffers/hunt/`

[7] Mathis, M., Mahdavi, J., Floyd, S., and Romanow, A., *TCP Selective Acknowledgement Options*. RFC 2018, April 1996.

[8] J. Postel, J. Reynolds. *Telnet protocol specification*. RFC 854, May 1983.

[9] Cisco Security Advisory. *TCP Vulnerabilities in Multiple IOS-Based Cisco Products*, April 2004.



Obrázek 2: Stavový automat TCP. Čárkované šipky znázorňují přechody stavů pro klienta, tučné čáry přechody stavů serveru. Čárkovaný rámeček vlevo dole znázorňuje aktivní uzavření spojení (active close), rámeček vpravo dole pasivní uzavření spojení (passive close). Převzato z [5].