Vodic za pocetnike za pravljenje zastitnog zida koristeci pf

Ovo uputstvo je napisano za one osobe koje nemaju mnogo iskustva sa konfigurisanjem zastitnih zidova. Shvatite da zastitni zid koji cemo konfigurisati nije podesan za svaku upotrebu. Samo pokusavam da obuhvatim nekoliko osnovnih stvari, koje sam savladao koristeci poznatiju (i detaljniju) dokumentaciju koja je dole navedena.
Nadam se da vas ovo uputstvo nece samo podstaci, nego i dati vam dovoljno da savladate koriscenje pf-a tako da cete onda moci da predjete na ona naprednija uputstva i usavrsite vase znanje o zastitnim zidovima.
pf paket filter je razvijen za OpenBSD ali je sada ukljucen i u FreeBSD, gde sam ga ja i koristio. Kako pokrenuti pf prilikom podizanja sistema i slicno je vec obradjeno u razlicitim dokumentima, medjutim na kratko cu objasniti korake za FreeBSD.
Prvo dodajte sledece u /etc/rc.conf

```
pf_enable="YES"
pf_rules="/etc/pf.conf"
pflog_enable="YES"
pflog_logfile="/var/log/pflog"
```

Ovo ce startovati pf filter pri podizanju sistema. Obradicemo startovanje pf-a nakon sto napisemo nekoliko pravila.

Nekoliko upozorenja

Budite pazljivi kada eksperimentisete sa pravilima na udaljenim masinama. Mnogi su zakljucali udaljene masine tako da vise nisu imali pristup. Jedno pametno resenje je da dodate cronjob za iskljucenje pf-a tako da iako napravite gresku veoma brzo se mozete vratiti na masinu. Ne radite ovo sada, sacekajte da napravimo nekoliko pravila ali prvo da napravimo par priprema. Kao sto znate, treba izbegavati da se bilo sta radi ulogovan kao root. Kao prvo, napravite brzu izmenu u /usr/local/etc/sudoers (na FreeBSD-u, moze biti drugacije na drugim operativnim sistemima)

```
visudo -f /usr/local/etc/sudoers
```

Ako je vase korisnicko ime john, onda dodajte liniju (ispod bilo koje druge linije dajuci vam manje privilegija – sudo obradjuje sudoers fajl odozgo nadole)

```
john    ALL= NOPASSWD: /sbin/pfctl
```

Sada editujte crontab.

```
crontab -e
```

Onda u crontab-u dodajte

```
*/10 * * * * /usr/local/bin/sudo /sbin/pfctl -d
```

Ne volim NOPASSWD opciju, zato sto je laske napraviti ozbiljne greske-medjutim kada testiram gomilu pf pravila onda je uglavnom stavljam zato sto konstantno koristim pfctl. Kada zavrsim, i lista pravila funkcionise, onda uklanjam NOPASSWD.

Ok,sada kada smo osigurali da se mozemo vratiti na masinu u slucaju da se nesto desi, kreirajmo sada /etc/pf.conf fajl. U FreeBSD-u, taj fajl vec postoji. Ono sto ja obicno radim je da zapocnem u mom home direktorijumu, kreiram pravila, i onda ih testiram, ucitavajucih ih sa sudo. Onda ako sve radi onako kako sam ja zeleo, backup-ujem osnovni pf.conf fajl i prekopiram ga.

```
mv /etc/pf.conf /etc/pf.conf.bak
cp pf.conf /etc
```

(Posto jos uvek nismo kreirali fajl nemojte jos nista raditi) ☺.

Jednostavan skup pravila

U vasem home direktorijumu, otvorite vas omiljeni tekst editor i kreirajte fajl pf.conf da bismo poceli sa kreiranjem vasih pravila. Pocecemo sa nekoliko makroa. Makroi su slicni varijablama u programiranju. Kratak primer

```
tcp_pass = "{ 80 22 }"
```

Kreirali smo makro. Kasnije kada kreiramo pravila, da nebismo kreirali jedno pravilo za port 80 i drugo za port 22 mozemo jednostavno uraditi nesto kao

```
pass out proto tcp to port $tcp_pass
```

(Morate imati znak $ ispred imena makroa)

Onda, ako odlucimo da dodamo port 123, Network Time Protocol, da ne bismo pisali novo pravilo jednostavno dodajemo 123 nasem makrou. Ako smo zaboravili da dodamo pravilo za email dodacemo sva tri, pop3 da primi email, smtp da posalje i ntp da bi se konektovao na servere.

```
tcp_pass= "{ 80 22 25 110 123 }"
```

pf filter ce citati pravila koja mi kreiramo odozgo nadole. Pocnimo sa pravilom da blokiramo sve

```
block all
```

Ovo ce prilicno osigurati kompjuter ali onda nece moci nista da uradi. Zato, da bi se omogucio pristup web stranama dozvolite Network Time Protocol-u da se konektuje na servere i da se poveze ssh-om na udaljen kompjuter, mozemo iskoristiti makro koji smo definisali. Nas pf.conf ce izgledati kao sledece.

```
tcp_pass = "{ 80 22 25 110 123 }"
block all
pass out on fxp0 proto tcp to any port $tcp_pass keep state
```

Pogledajmo sta smo uradili do sada. Prvo, defnisali smo makro. Koristili smo brojeve portova. Mozemo stavljati i zareze ako zelimo ili ime protokola ako je definisan u /etc/services. Tako da bi makro mogao da izgleda kao

```
tcp_pass = "{ 80 ssh, ntp smtp 110}"
```

i da jos uvek radi. Medjutim treba teziti ujednacenoscu.

Mozete koristiti ime samo ako je port definisan u /etc/services. Na primer, ako odlucite da namestite ssh da prihvata konekcije samo na portu 1222, onda moramo da taj port dodamo tom makrou. NE ssh. Port 1222 je definisan u /etc/services kao nerv, SNI R&D network, tako da ako proverite vas skup pravila koristeci pfctl pokazace da imate pravilo koje dozvoljava ulaz do nerv. Da bi se izbegla konfuzija, ako zelite da koristite nestandardan port, koristite nesto sto nije izlistovano u /etc/services.

Mozete odrediti opseg portova koristeci dve tacke. Na primer, ako zelim da dodam Sambu, koja koristi portove 137, 138 i 139, mogu da dodam 137:139.

Sledece, blokiramo sve kao osnovno pravilo. Sada moramo dodati pravila da bi dopustili aplikacijama prolaz kroz zastitni zid. Setite se, pf obradjuje pravila odozgo nadole.

Dakle odobrili smo web saobracaj, ssh konekcije, slanje mailova, primanje mailova preko pop3 i mozemo da kontaktiramo vremenske servere.

Da pocnemo sa akcijom, prolazak. Propustamo, ne blokiramo. Red u sintaksi je vazan. Ovde dajem samo osnovne opcije, opet, ovaj clanak se moze posmatrati kao priprema za naprednije tutorijale.

pass out znaci da dozvoljavamo da stvari izlaze, ne da ulaze. Sledeci deo, on fxp0 se odnosi na interface—u ovom slucaju mreznu karticu koja je jezikom FreeBSD-a nazvana fxp0 (Intelova kartica)

to any znaci da dozvoljavamo da ode bilo gde. Sledece, odredicemo portove o kojima pricamo—ovde cemo kloristiti nas makro, tcp_pass, sto znaci da dozvoljavamo portove navedene gore.

keep state moze biti vazno. To znaci da kada se jednom osnuje konekcija, pf ce motriti na nju, tako da odgovor od, na primer web stranice, ne mora da prolazi kroz pravila ponovo nego ce jednostavno otvoriti konekciju. Isto vazi i za pop3. Kada koristite pop3 da kontaktirate pop3 server (hrrm, ocito), i server salje odgovor. Posto imamo keep state, odgovor servera moze direktno proci kada je konekcija uspostavljena.

Postoji mnogo opcija. Na primer, ako kreiramo pravilo pass on fxp0 umesto pass out on fxp0, onda ce saobracaj biti dozvoljen u oba pravca, ulazni i dolazni.

Ako pogledate /etc/services, videcete da neke stvari, kao ipp, port 631 koje se koriste u CUPS, koriste i tcp i udp. Da bi se nosili sa ovakvim stvarima mozemo ubaciti jos jedan makro

```
udp_pass = "{ 631 }"
```

Mozemo primetiti da slanje emailova ne funkcionise pravilno. Proverom /etc/services mozemo videti da smtp moze koristiti udp.

```
grep smtp /etc/services
smtp            25/tcp    mail        #Simple Mail Transfer
smtp            25/udp    mail         #Simple Mail Transfer
smtps           465/tcp    #smtp protocol over TLS/SSL (was ssmtp)
smtps           465/udp    #smtp protocol over TLS/SSL (was ssmtp)
```

Ipak, sa nasim novim makroom ovo je jednostavno—jednostavno ga dodajemo udp_pass-u

```
udp_pass = "{ 110 631 }"
```

Sada, dodajmo udp_pass makro nasem skupu pravila

```
tcp_pass = "{ 80 22 25 110 123 }"
udp_pass = "{ 110 631 }"
block all
pass out on fxp0 proto tcp to any port $tcp_pass keep state
pass out on fxp0 proto udp to any port $udp_pass keep state
```

Medjutim, nalazimo da CUPS ne radi. Brzi grep ipp u /etc/services pokazuje da on koristi i tcp. Tako da dodajemo 631 nasoj listi portova u tcp_pass makrou. Sada imamo

```
tcp_pass = "{ 80 22 25 110 123 631 }"
udp_pass = "{ 110 631 }"
block all
pass out on fxp0 proto tcp to any port $tcp_pass keep state
pass out on fxp0 proto udp to any port $udp_pass keep state
```

Ovo vazi za bilo koji port koji ste zaboravili. Na primer, ovaj skup pravila ne dozvoljava DNS, port 53. Uradimo grep domain in /etc/services, i vidimo da koristi tcp kao i udp tako da dodajemo53 u oba makroa

```
tcp_pass = "{ 80 22 25 53 110 123 631 }"
udp_pass = "{ 53 110 631 }"
```

Koriscenje reci quick

Paketi koji ulaze i izlaze se proveravaju skupom svih pravila pre nego sto se propuste ili blokiraju.. Ponekad zelite da ubrzate stvari i da brzo blokirate ili propustite paket.

U takvim slucajevima mozete koristiti quick. Ako se paket podudara sa necim u toj liniji on vise ne ide kroz skup pravila nego propusta paket. Na primer, recimo da imate web server u vasoj LAN mrezi iza zastitnog zida, i sigurni ste da svi zahtevi ka portu 80 dolaze od vase unutrasnje mreze tako da mozete brzo da ih propustite (u stvarnosti ovo nebi bilo tako ali ovo je za primer koriscenja quick-a)

```
pass in quick on fxp0 proto tcp to any port 80 keep state
```

Stavite ovo iznad ostalih pravila, odmah nakon definisanja makroa. Sada, zahtevi koji dolaze prema LAN webserveru ce biti odmah propusteni (u ovom slucaju je bilo pass in) bez provere paketa ostalim pravilima.

Tabele

Tabele su korisne i brze. One se koriste da grupisu adrese. Na primer,ako zelite da dozvolite sve ulazne konekcije sa lokalne mreze, 192.168.8.0 and 192.168.9.0

```
table <local> { 192.168.8.0/24, 192.168.9.0/24 }
```

Ubacite ovu liniju iznad vasih pravila. Sada, da bi odobrili sve od ove dve mreze (iskoristicemo quick takodje) dodajte pravilo. S obzirom da korstimo quick stavicemo ovo pravilo blize vrhu. Kada bi bilo pri dnu onda se gubi svrha quick-a zato sto bi paketi vec bili pregledani od strane svakog pravila iznad ovog.

```
pass in quick from <local> to any keep state
```

Nekada cete napraviti greske u kucanju tako da je uvek dobro da proverite vase tabele.

```
pfctl -t local -T show
```

ce vam pokazati sadrzaj vase tabele. Primeticete da ako editujete vas pf.conf, dodajuci tabelu, i onda jednostavno pfctl -d i -e, da ugasite ili pokrenete pf, mozda se pravila u tabelama nece primeniti. Nacin da se ovo uradi je

```
pfctl -t local -Tl -f /etc/pf.conf
```

Oznaka -t je za ime tabele, u ovom slucaju, local. Oznaka -T se koristi da pruzi razne komande. U ovom slucaju, mi koristimo l za ucitavanje. To je malo slovo L, nije broj. Oznaka -f je za fajl, u ovom slucaju, /etc/pf.conf gde smo dodali tabelu.

Tabele se mogu dodavati i brisati u toku rada, pogledajte link na kraju ovog clanka za vise detalja.

Pfctl komanda

Vec smo pomenuli pfctl komadu ranije. Komanda radi razlicite stvari—kao sto mozete da pogodite na osnovu imena, ona kontrolise pf. Vec sam napomenuo pravljenje sudoers fajla tako da mozete da testirate vas skup pravila, kao i da koristite cron job da bi ste ponovo imali pristup kompjuteru ako ga zakljucate. Ona se koristi da ukljuci, iskljuci, ponovo pokrene i izbrise skupove pravila kao i da vam ocita status. Napomenucu samo nekoliko glavnih opcija ovde, za vise pogledajte man stranice. Kao sto znate cronjob koji smo kreirali ce jednostavno disable-ovati paket filter. To je jednostavna -d oznaka

```
sudo pfctl -d
```

Da biste pokrenuli paket filter, pretpostavljajuci da je vas username john i da ste ga kreirali u vasem home direktorijumu kao john

```
sudo pfctl -ef pf.conf
```

Ovo -f oznacava fajl. Recimo da smo ga proverili i da nam se nije svideo tako da hocemo da se vratimo na default pravila, u /etc/

```
sudo pfctl -Rf /etc/pf.conf
```

Opcije su razne za koriscenje komande pfctl. Komanda pfctl -s info vam daje brzi pregled sta se dogadja. Komanda pfctl -vs rules pokazuje vasa pravila i sta se desava, na primer

```
pass out proto tcp from any to any port = http keep state
  [ Evaluations: 96        Packets: 906       Bytes: 496407      States: 0     ]
pass out proto tcp from any to any port = pop3 keep state
  [ Evaluations: 96        Packets: 514       Bytes: 71260       States: 0     ]
```

Man stranica vam daje komplentu listu.

Sada je vreme da ovo testiramo. Rano je jutro i ne razmisljam bas dobro. Nadam se da sam dodao sebe u sudoers da bih mogao da pokrenem pfctl bez lozinke a i da sam napravio cronjob tako da, ako napravim gresku mogu brzo da zaustavim pf. (VEOMA vazno ako ovo testirate na kompjuteru na kome nemate fizicki pristup). Sada pokrecem pravila koja sam sacuvao kao pf.conf fajl koji sam sacuvao u mom home direktorijumu

```
sudo pfctl -ef pf.conf
```

Saznajem da sam zabranio svaki pristup kompjuteru. Ups. Zaboravio sam da dozvolim ulaz za ssh konekciju i konekcija koju sam koristio je upravo blokirana.

Sacekacu nekoliko minuta dok crontab disable-uje pf i ulogovacu se ponovo na udaljenio racunar. Ovog puta dodajem pravilo

```
pass in proto tcp to port 22 keep state
```

Sada se cini da su ova pravila u redu. Sada kopiram ta pravila u /etc/pf.conf

```
sudo cp pf.conf /etc/
sudo pftcl -Rf /etc/pf.conf
sudo pftcl -s rules
```

Izlistavajuci pravila vidim da ona rade ono sto sam ja hteo da rade.
Sada, naravno, uklanjam cronjob. Ako cesto eksperimentisete sa svojim pravilima i ponovo vam zatreba samo otkucajte crontab –e i ispred linije stavite znak #.

Nekoliko mogucnosti I zavrsetaka

Koriscenje ftp-a sa paket filterom moze biti problematicno. Za detaljno objasnjenje resenja pogledajte detaljnije tutorijale koje sam naveo na kraju ovog clanka. Ukratko, morate konekcije preusmeriti na proxy.

Jedini trik za koji mislim da nije pomenut u detaljnijim tutirijalima je da mora da bude izand vasih filter pravila. U suprotnom, dobicete gresku da pravila moraju biti u odredjenom redu, normalizaciji, itd.

Postoji i scrub. Opet, iako je ovo bolje objasnjeno u tutoijalima navedenim dole, on ustvari normalizuje fragmentirane pakete. Najcesce se korsiti

```
scrub in all
```

Ovo pravilo mora da ide iznad pravila za filtriranje (i iznad rdr da bi mogli da preusmerite ftp ako ga koristite) To je ustvari ta “normalizacija” koja se javlja u error poruci. Na vrhu skupa pravila definisete makroe, tabele i slicno. Onda dolazi scrub pravilo, onda rdr pravilo, onda vasa pravila za filtriranje (grupa koja u nasem primeru pocinje sa block all). scrub vas moze zastititi od pojedinih vrsta napada i dobro je da ga stavite.

U FreeBSD-u morate jos jednom da dodate pf u vas kernel. Proverite /usr/src/sys/conf/NOTES za vise opcija ali ja imam samo ove

```
device      pf
device      pflog
device      pfsync
```

Imam i

```
options     ALTQ
```

u mom kernelul, iako ne koristim queuing. Zato sto svaki put kada ponovo ucitate vasa pf pravila javlja se upozorenje da ALTQ nije omogucen u kernelu.

Ako koristite modul, pretpostavlja se da imate bpf, INET i INET6 u kernelu. (Vidite stranicu u handbook-u o pf-u.)

Ovaj tutorijal je predvidjen da bude samo kao uvod. Ipak, ako ste razumeli ova jednostavna pravila onda ste spremni da predjete na slozenije tutorijale. Dva koja ja najvise koristim su OpenBSD PF Prirucnik i Tutorial Peter N.M. Hansteen-a

url:http://home.nyc.rr.com/computertaijutsu/pf.html