Tema 5 - Stateful firewall

Termen de predare: **joi, 19 Mai 2011, ora 23:00**

Scopul temei

Implemenarea unui firewall de tip stateful pentru controlul traficului pachetelor IP, folosind API-ul de hook-uri pus la dispoziţie de kernel.

Obiective didactice

- Familiarizarea cu API-urile pentru implementarea unui filtru de pachete / firewall.
- Întelegerea modului de funcţionare al firewall-urilor de tip stateful.
- Deprinderea cu modul de lucru cu pachetele IP la nivelul nucleului.
- Obţinerea de deprinderi avansate de lucru cu acţiuni amânabile, în special timere.

Enunţ

Să se scrie un modul de kernel care să implementeze un "stateful firewall" pentru pachete IP. Rolul unui firewall este de a controla, pe baza unei configuraţii, ce pachete pot intra sau ieşi. Filtrarea se va face doar pentru pachete TCP şi UDP. Alte tipuri de pachete vor fi lăsate să treacă, atât în interior cât şi în exterior. Filtrarea se va face şi pentru pachete generate local şi pentru pachete destinate staţiei.

Spre deosebire de un firewall bazat doar pe liste de acces cu reguli statice (stateless), un firewall stateful, pe lângă regulile statice, trebuie să creeze şi să folosească reguli dinamice, în funcţie de starea unei conexiuni. Regulile dinamice sunt create pentru traficul de răspuns (trafic ce intră în sistem) atunci când există trafic permis să iasă din sistem şi necesită un răspuns.

Cerinţele de implementare ale firewall-ului sunt:

1. vor fi lăsate să iasă toate pachetele
2. vor fi lăsate să intre pachetele ce reprezintă răspunsuri la pachetele ce au iesit (pachete ce fac parte dintr-o conversaţie iniţiată din interior)
3. vor fi lăsate să intre şi pachete ce nu reprezintă răspunsuri la pachetele ce au iesit, pe baza unor reguli specificate de utilizator (din user-space); regulile vor specifica adresa sursă, adresa destinaţie, portul sursă, portul destinaţie pentru pachete ce vor fi lăsate să intre
4. orice alt trafic TCP sau UDP ce intră este respins

Reguli

Formatul regulilor este dat de următoarea structură:

```
struct fwr {
        unsigned int ip_src, ip_dst, ip_src_mask, ip_dst_mask;
        unsigned short port_src[2], port_dst[2];
};
```

Valorile numerice sunt exprimate în network byte-order.

Pentru ca un pachet să facă "match" pe o regulă şi să fie lăsat să intre, trebuie indeplinite următoarele condiţii:

- adresa IP sursă să facă parte din reţeaua definită de adresa IP `ip_src` şi masca de reţea `ip_src_mask`;
- adresa IP destinaţie să facă parte din reţeaua definită de adresa IP `ip_dst` şi masca de reţea `ip_dst_mask`;
- portul sursă să facă parte din intervalul închis `port_src[0]` - `port_src[1]`;
- portul destinatie să faca parte din intervalul închis `port_dst[0]` - `port_dst[1]`.

Interfaţă

Driverul trebuie să poata fi comandat din user-space prin următoarele operaţii speciale (comenzi ioctl):

- `FW_ADD_RULE` - adăugă o nouă regulă din user-space; operaţia primeşte ca parametru o structură `fwr`;
- `FW_ENABLE` - activează firewall-ul;
- `FW_DISABLE` - dezactivează firewall-ul, dar nu şterge regulile statice; ştergerea sau păstrarea regulilor dinamice nu este relevantă pentru testare, dar nu ar trebui să se creeze reguli dinamice noi cât timp firewall-ul este dezactivat;
- `FW_LIST` - copiază lista regulilor active din kernel în userspace (atât cele *statice*, cât şi cele *dinamice*).

Observaţii legate de reguli şi ioctl-ul `FW_LIST`:

- La inserarea modulului, lista de reguli este goală.
- Regulile *statice* sunt adăugate de utilizator prin ioctl-ul `FW_ADD_RULE` şi vor rămâne pâna la descărcarea modulului.
- Regulile *dinamice* trebuie adăugate de driver atunci când se detectează un pachet care iniţiază o conversaţie şi trebuie şterse după ce conversaţii a luat sfârşit. Detalii despre detecţia conversaţiilor în secţiunea următoare.
- Nu contează cum este implementată intern lista de reguli şi nici ordinea în care sunt returnate regulile, atâta timp cât sunt prezente în listă toate regulile care ar trebui sa fie, şi numai acelea.
- O implementare care foloseşte vectori de dimensiune fixă pentru reguli este considerată neadecvată.

Conversaţii şi reguli dinamice

- Dacă se implementează cerinţa 3, cerinţa 2 se poate implementa uşor, prin inspectarea traficului de ieşire şi adăugarea unor reguli dinamice în tabela de reguli, odată ce este detectată o conversaţie iniţiată din interior. În cazul în care se detecteaza un pachet ce iniţiază o conexiune din interior, fie el **(ip sursă, ip destinaţie, port sursă, port destinaţie) = (a, b, c, d)** se va insera în tabela de reguli un pachet **(b, a, d, c)**.
- Regulile dinamice trebuie sterse din tabela de reguli odată ce conversaţia ia sfârşit.
- Pentru detectarea unei conversaţii TCP iniţiate din interior este suficient să se inspecteze pachetele ce au flagul `SYN` setat şi flagul `ACK` nesetat (pentru detalii studiaţi diagrama TCP).
- Pentru detectarea sfârşitului unei conversaţii TCP se poate testa prezenţa flagului `FIN`. Totuşi, datorită secvenţei de închidere a unei conexiuni TCP, nu se poate scoate regula dinamică din tabela de reguli imediat ce se detectează un pachet `FIN` (vedeţi din nou diagrama TPC). O implementare simplă poate şterge regula după un timeout (de exemplu, 300ms).
- Pentru detectarea unei conversaţii UDP iniţiate din interior trebuie analizate toate pachetele UDP care pleacă. Se consideră că o conversaţie UDP ia sfârşit dacă timp de **300ms** nu sunt generate pachete ce fac parte din conversaţie. Din această cauză trebuie asociat un timer pentru fiecare regulă dinamică ce se referă la pachete UDP. Vor fi inspectate pachete UDP care ies şi care intră şi se va:
  - arma/rearma timer-ul asociat regulii pentru pachetele UDP care ies;
  - rearma timer-ul asociat regulii pentru pachetele UDP care intră.

Precizări Linux

- `FW_ADD_RULE` va primi ca argument o variabilă de tip `struct fwr` în buffer-ul de intrare al ioctl-ului.
- `FW_LIST` va primi ca argument o zonă de memorie alocată de user în care se vor pune regulile actuale. Utilizatorul va pune un întreg pe 32 biţi la începutul zonei de memorie, reprezentând numărul de reguli pentru care a alocat zona. Dacă numărul de reguli active este mai mare decât cel specificat de utilizator, driverul trebuie să pună în primul întreg din zona pasată de utilizator numărul de reguli şi să întoarcă `-ENOSPC`. Altfel, driverul va copia regulile în user-space şi va întoarce numărul de reguli copiate.
- Driverul va fi vizibil în user-space ca un device driver de tip caracter, cu majorul 42.
- Structura unei reguli şi operaţiile ioctl descrise sunt definite în ipfirewall.h; header-ul poate fi folosit ca atare sau poate fi modificat, atâta timp cât se păstrează interfaţa ioctl descrisă în enunţ.

Precizări Windows

- `FW_ADD_RULE` va primi ca argument adresa unei variabile de tip `struct fwr`.
- `FW_LIST` va primi în buffer-ul de intrare un întreg pe 32 de biţi ce reprezintă numărul de reguli pentru care a fost alocat buffer-ul. Dacă numărul de reguli active este mai mare decât cel specificat, driver-ul trebuie să pună în buffer-ul de ieşire numărul de reguli şi să întoarcă `STATUS_SUCCESS` (dacă nu întoarce succes nu se copiază în buffer-ul de ieşire ce s-a scris din kernel). Dacă zona alocată este suficient de mare, driver-ul trebuie să pună în buffer-ul de ieşire mai intâi numărul de reguli şi apoi lista lor. Trebuie pus şi numărul de reguli pentru ca iniţiatorul să poată determina dacă apelul a reuşit şi a fost întoarsă lista de reguli, sau dacă trebuie să folosească un buffer mai mare.
- Driverul va fi accesat din user-space prin numele: `\\.\ipfirewall`
- Structura unei reguli şi operaţiile ioctl descrise sunt definite în ipfirewall.h; header-ul poate fi folosit ca atare sau poate fi

modificat, atâta timp cât se păstrează interfaţa ioctl descrisă în enunţ.

Testare

Pentru simplificarea procesului de corectare al temelor, dar şi pentru a reduce greşelile temelor trimise, corectarea temelor se va face automat cu ajutorul unor teste publice.

Trimiterea temei se face prin interfaţa VMchecker, iar tot acolo puteţi vedea indicaţii despre modul de trimitere al temei.

Testele presupun că numele modulului de kernel este `ipfirewall`.

Testarea temei trebuie facută pe două sisteme: fie pe două maşini virtuale, fie pe o maşină virtuală şi sistemul gazdă. Mai multe detalii despre cum se realizează testarea gasiţi in fiserul README.checker din cadrul arhivei de test.

Important! La testarea temei, orice conexiuni TCP iniţiate de pe maşina virtuală, sau pachete UDP trimise de aceasta, pot duce la picarea unor teste din cauza regulilor suplimentare care apar. În cazul în care testerul detectează că numărul de reguli nu corespunde, va declara testul respectiv picat şi va afişa întreagă listă de reguli returnată de driver.

Astfel, dacă trimiteţi tema şi VMchecker-ul detectează erori de acest gen, dar testele merg fără probleme pe sistemul vostru, retrimiteţi tema pentru o reevaluare.

Depunctari

În corectarea temei se va ţine cont de următoarele criterii de depunctare:

- -1.0 implementare incorectă a cerinţelor temei
- -1.0 implementare neadecvată a listei de reguli
- -0.1 utilizarea "nefirească" a API-ului
- -0.1 pentru fiecare test picat

De asemenea, consultaţi sfaturile şi depunctările generale.

Întrebări

Pentru întrebari legate de temă puteţi consulta pagina FAQ, arhivelele listei de discuţii sau puteţi trimite un e-mail pe listă (trebuie să fiţi abonaţi).

Înainte să puneţi o întrebare verificaţi că:

1. Aţi citit bine enunţul temei;
2. Întrebarea nu este deja prezentată pe pagina de FAQ;
3. Nu se poate găsi răspunsul în arhivele listei de discuţii.