Stateful firewall

Termen de predare: Duminica 7 iunie, ora 24.00 (2009-06-07)

Cuprins

Enunt

Sa se scrie un modul de kernel care sa implementeze un "stateful firewall" pentru pachete IP. Rolul unui firewall este de a controla pe baza unei configuratii ce pachete pot intra sau iesi. Cerintele de implementare a firewall-ului sunt:

- vor fi lasate sa iasa toate pachetele;
- vor fi lasate sa intre pachetele ce reprezinta raspunsuri la pachetele ce au iesit (pachete ce fac parte dintr-o conversatie initiata din interior);
- vor fi lasate sa intre si pachete ce nu reprezinta raspunsuri la pachetele ce au iesit pe baza unei tabele de reguli, reguli ce sunt specificate de utilizator (din user-space); regulile vor specifica adresa sursa, adresa destinatie, portul sursa, portul destinatie pentru pachete ce vor fi lasate sa intre.

Filtrarea se va face doar pentru pachete TCP si UDP. Alte tipuri de pachete vor fi lasate sa treaca, atat inauntru cat si in afara. Filtrarea se va face pentru pachete generate local sau destinate statiei.

Formatul regulilor este dat de urmatoarea structura:

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

Valorile numerice sunt exprimate in network byte-order.

Pentru ca un pachet sa faca "match" pe o regula si sa fie lasat sa intre, trebuie indeplinite urmatoarele conditii:

- adresa IP sursa sa faca parte din reteaua definita de adresa IP `ip_src` si masca de retea `ip_src_mask`;
- adresa IP destinatie sa faca parte din reteaua definita de adresa IP `ip_dst` si masca de retea `ip_dst_mask`;
- portul sursa sa faca parte din intervalul `port_src[0] - port_src[1]`;
- portul destinatie sa faca parte din intervalul `port_dst[0] - port_dst[1]`.

Driverul trebuie sa poata fi comandat din user-space cu urmatoarele operatii speciale (comenzi ioctl):

```
#define FW_ADD_RULE             1
#define FW_ENABLE               2
#define FW_DISABLE              3
#define FW_LIST                 4
```

unde:

- `FW_LIST` se va folosit pentru copierea regulilor actuale din kernel in userspace (atat statice cat si dinamice);
- `FW_ADD_RULE` se va folosi pentru a adauga o noua regula din user-space; operatia primeste ca parametru o regula (un `fwr_t`);
- `FW_ENABLE` va porni firewall-ul;
- `FW_DISABLE` va opri firewall-ul; nu se vor sterge regulile.

Precizari Generale

- daca sa implementeaza cerinta 3, cerinta 2 se poate implementa usor, prin inspectarea traficului de iesire si adaugarea unor reguli dinamice in tabela de reguli, odata ce este detectata o conversatie initiata din interior; in cazul in care se detecteaza un pachet ce initiaza o conexiune din interior, fie el **(ip sursa, ip destinatie, port sursa, port destinatie) = (a, b, c, d)** se va insera in tabela de reguli un pachet **(b, a, d, c)**;
- regulile dinamice vor trebuie sterse din tabela de reguli o data ce conversatia ia sfarsit;
- pentru detectarea unei conversatii TCP initiate din interior este suficient sa se inspecteze pachetele ce au flagul `SYN` setat si flagul `ACK` nesetat; (pentru detalii studiati aceasta diagrama);
- pentru detectarea sfarsitului unei conversatii TCP se poate folosi flagul `FIN`; datorita secventei de inchidere a unei conexiuni TCP, nu se poate scoate regula dinamica din tabela de reguli imediat ce se detecteaza un pachet `FIN`; pentru detalii studiati aceasta diagrama; o implementare simpla se poate face prin stergerea regulii dupa un timeout (maxim 100ms);
- pentru detectarea unei conversatii UDP initiate din interior trebuie analizate toate pachetele UDP ce pleaca; se considera ca o conversatie UDP ia sfarsit daca timp de 100ms nu sunt generate pachete ce fac parte din conversatie; din aceasta cauza trebuie asociat un timer pentru fiecare regula dinamica ce se refera la pachete UDP; vor fi inspectate pachete UDP care ies dar si intra si se va:
    - ♦ arma/rearma timer-ul asociat regulii pentru pachetele UDP ce ies;
    - ♦ rearma timer-ul asociat regulii pentru pachetele UDP ce intra.

Precizari Linux

- `FW_LIST` va primi ca argument o zona de memorie alocate de user in care se vor pune regulile actuale. Utilizatorul va pune in primul intreg din aceasta zona numarul de reguli pentru care a alocat zona. Daca numarul de reguli e mai mare decat cel specificat de utilizator, driverul trebuie sa puna in primul intreg din zona pasata de utilizator numarul de reguli si sa intoarca `-ENOSPC`. Altfel, driverul va copia regulile in userspace si va intoarce numarul de reguli copiate.
- driverul va fi vizibil in user-space ca un device driver de tip caracter, cu majorul 42;
- structura si operatiile speciale descrise le puteti gasi si in acest header.

Precizari Windows

- pentru `FW_LIST` driver-ul primeste in buffer-ul de intrare un numar de reguli care indica zona alocata. Daca numarul de reguli e mai mare decat cel dat, driver-ul trebuie sa puna in buffer-ul de iesire numarul de reguli si sa intoarca `STATUS_SUCCESS` (ideea e ca daca nu intoarce succes nu se copiaza in buffer-ul de iesire ce pune el). Daca zona alocata e suficient de mare driver-ul trebuie sa puna in buffer-ul de iesire mai intai numarul de reguli si apoi lista lor. Trebuie pus si numarul de reguli pentru ca utilitarul sa aiba cum sa isi dea seama daca apelul a reusit si a fost intoarsa si lista de reguli;
- driverul va fi accesat din user-space cu "\\.\ipnecklace";
- structura si operatiile speciale descrise le puteti gasi si in acest header;
- din cauza constrangerilor de testare, in cazul temei pe Windows va trebui sa **nu** includeti reguli de firewall pentru pachete UDP (care intra sau ies) care au ca adresa IP sursa adresa de loopback (adica situatiile in care `ip_src == 0x0100007f`); pachetele UDP care au aceasta adresa IP sursa vor fi lasate sa treaca fara a fi adaugate reguli pentru ele.

Testare

Pentru simplificarea procesului de corectare al temelor, dar si pentru a reduce greselile temelor trimise, corectarea temelor se va face automat cu ajutorul unor teste publice (linux, windows).

Testele presupun ca numele modulului de kernel este `ipdriver`.

Testarea temei trebuie facuta pe doua sisteme. Fie pe doua masini virtuale fie pe o masina virtuala si sistemul gazda. Mai multe detalii despre cum se realizeaza testarea gasiti in fiserul README din cadrul arhivei de test.

Depunctari

- 0.3: sincronizare incompleta
- 0.3: memory leaks
- 0.2: lipsa readme
- 0.2: cod neingrijit
- 0.1: utilizare "nefireasca" a api-ului
- 0.0: observatii

Intrebari

Pentru intrebari legate de tema puteti consulta arhivelele listei de discutii sau puteti trimite un e-mail (trebuie sa fiti inregistrati).