SELinux si grsecurity

1. Introducere

Securitatea sistemelor de operare este problema prinicpala pentru toti utilizatorii de calculatoare si programatori. Dezvoltatorii de sisteme de operare si-au exprimat ingrijorarea in diferite moduri, rezultand numeroase imbunatatiri aduse sistemelor de operare actuale. In acesta tema vom analiza SELinux si grsecurity, doua imbunatatiri aduse kernului linux.

SELinux aduce un mecanism imbunatatit pe tipuri de domenii si control al accesului bazat pe roluri, in timp ce grsecurity aduce “accesul prin liste de control” (Access Control List – ACLs).

S-a observat ca SELinux este mai puternic in implementarea MAC(Mandatory Access Control), iar grsecurity este mai simplu de utilizat si ofera si alte caracteristici exclusive, cum ar fi protectia spatiului de adrese si limitarea resurselor. In plus cele doua variante folosesc abordari diferite pentru atingerea aceluiasi scop, fara diferente prea mari in imbunatatirea performantei.

Se va face o comparatie privind structura, implementarea, functionalitatea, flexibilitatea, utilizabilitatea si performanta intre cele doua unelte de securitate SELinux si grsecurity.

MAC poate fi definit ca:

“Un mecanism de sistem controleaza accesul la un obiect si un user nu poate modifica acest access, controlul este facut prin delegari(mandatory access control), ocazional mai este numit si control bazat pe reguli.[1]”

SELinux ofera suport pentru configurarea flexibila a politicilor de securitate cu intentia de a trece peste limitarile aduse de mecanismul MAC. Pe de alta parte grsecurity implementeaza mecanismul MAC folosind ACLs. ACL reprezinta un set de perechi asociate cu un obiect. Fiecare pereche contine un subiect si un set de drepturi. Subiectii pot accesa doar obiectele asociate lor folosind oricare din drepturile acestora. In grsecurity prin folosirea listelor ACLs se ajunge la controlul accesului bazat pe procese. Grsecurity include un utilitar numit **gradm** pentru crearea si managementul listelor ACLs. Tot grsecurity include mai multe mecanisme pentru lucrul cu ACLs, cum ar fi protectia expunerilor la supraincarcare, protectia sistemului de fisiere, de audit, si optiuni de permutare.

2. Securitatea folosind SELinux

CALOTA BENIAMIN NATANAEL

2.1 **Flask**

Mecanismele traditionale MAC au fost restranse la politici de securitate multi-nivel(multi-level security policy), care isi bazeaza deciziide pe clasificarea obiectelor si pe claritatea subiectului.

“Aceasta abordare traditionala este mult prea limitata pentru a indeplini cele mai multe

dintre cerintele de securitate. Ea ofera suport slab pentru date si integritatea aplicatiilor, separarea sarcinilor, si minimul de privilegii de securitate. Totodata are nevoie de subiecti de incredere care sa actioneze din afara modelului de control al accesului. Nu reuseste sa mentina controlul strans in relatia dintre subiect si codul pe care acesta il executa. Acest lucru limiteaza abilitatea sistemului de a oferi protectie bazata pe functii si pe increderea in codul executat, `pentru a controla permisiunile cerute in executie si a minimiza impactul executarii codurilor malitioase."

Datorita acestor limitari a fost creata Arhitectura Flask, o arhitectura bazata pe arhitectura MAC. Flask a for creata pentru a aduce suport flexibil in politicile de securitate, dand posibilitatea administratorilor de a folosi modelul politicilor MAC in diversele lor cerinte de securitate. Tot aceasta arhitectura face posibila folosirea a multor modele de tip MAC, datorita separarii politicilor logice de securitate de cele ale de executie.

In Flask fiecare proces si obiect are propriul context de securitate care stipuleaza toate atributele de securitate ale procesului sau obiectului. SELinux foloseste indentificatori de secirtate, simpli intregi, pentru a reprezenta fiecare context de securitate. Cand apare o problema de securitate, codul in executie trimite o serie de identidicatori de securitate(SIDs), identificatorul subiectului, si identificatorul obiectului, si se ia o decizie in functie de acesti identificatori de securitate. Este important de notat ca aceste contexte de securitate au propria lor identitate implementata separat fata de traditionalul Linux user IDs(UIDs).

Flask pune la dispozitie un ACCESS VECTOR CACHE(AVC) in care se stocheaza deciziile de securitate ale serverului pentru a fi folosite ulterior. Un manager de obiecte comunoca cu serverul de securitate pentru a verifica si modifica permisiunile. AVC-ul permite managerului de obiecte sa comunice in acest fel mult mai repid, din moment ce deciziile sunt deja stocate.

Flask integreaza etichete de securitate, este flexivil in etichetarea si accesul la decizii, si suporta schimbarea politicilor de securitate. Contine si o componenta pentru management-ul proceselor si aduce un mecanism pentru controlul intregului sistem de fisiere, precum si pentru fisiere individuale si directoare.

2.2 SELinux - aplicatie de tip Flask

SELinux este o simpla aplicatie a arhitecturii Flask folosita de sistemele Linux. Mecanismele de control al politicilor de securitate din SELienux sunt vazute ca verificari ale permisiunilor care au fost introduse la un anumit moment dealungul kernelului. Aproximativ 140 de permisiuni, grupate in 28 de clase de obiecte, sunt definite si pot fi controlate de aproape orice sistem. Verificare permisiunilor de securitate se face prin comunicarea cu serverul de securitate pentru a accesa SID-urile si pentru a determina accesul fiecarei instante a obiectelor din sistem.

In referinta [4] gasim un exemplu de server de securitate implementat pentru SELinux. In acest exemplu se foloseste o combinatie intre "Identity-based Access Control(IBAC)" – control al accesului bazat pe identitate- , "Role-based Access Control(RBAC)" – Controlul accesului bazat pe rol -, si "Type Enforcement(TE)" - tipul de executie. Un context de securitate in SELinux are trei atribute: o identitate, un rol si un tip. Fiecare proces din sistem are propria identitate. Aceasta identitate difera fata de cea standard din Unix. Identiatea userilor din SELinux detemina rolul si domeniul pe care userul sau procesul il poate folosi. Astfel un user are un set de roluri de care se poate folosi in orice moment.

Rolul determina domeniul care poate fi folosit. Politicile de securiate arata rolul de care

se poate folosi fiecare identitate. Fiecare obiect din sistem este de un anumit tip, care determina cine poate avea acces la obiect. Fiecare proces ruleaza intr-un domeniu. Domeniul determina nivelul de acces pe care il are procesul. In SELinux tipurile si domeniile sunt sinonime. Conceptul de domeniu si cel de tip sunt asemenatoare, singura diferenta este ca tipurile se aplica obiectelor(ex. fisiere, socket-uri), iar domeniile se aplica proceselor. In SELinux, RBAC permite userilor sa sa intre intr-un anumit domeniu folosind reguli din setul propriu. SELinux contine cateva mecanisme de securitate inluzand cele pentru controlul proceselor, controlul fisierelor si controlul socket-urilor.

2.3 Configurarea politicilor de securitate

Folosind SELinux putem crea un model combinat din modelurile TE si RBAC. Astfel putem defini politici de securitate pentru procese si obiecte la un nivel foarte jos folosind modelul TE, iar folosind RBAC putem pastra un nivel ridicat de abstractizare.

Instructiuniile din TE reprezinta declaratii de atribute, declaratii de tipuri, reguli de tranzitie al tipurilor, reguli de acces la vectori sau afirmatii. Toate acestea descriu atributele si tipurile de roluri care pot fi create

Instructiunile RBAC reprezinta declaratii de roluri, definiri de ierarhii, sau permisiuni de acces ale rolurilor. Codul pentru TE si RBAC este:

```
te_rbac -> te_rbac_statement |
te_rbac
te_rbac_statement
te_rbac_statement -> te_statement |
rbac_statement
te_statement -> attrib_decl |
        type_decl |
        type_transition_rule |
        type_change_rule |
        te_av_rule |
        te_assertion
rbac_statement -> role_decl |
            role_dominance |
            role_allow_rule
```

[5]

Un cod mai specializat se poate obtine folosind limbajul SELinux, un limbaj mai robust si mai complet.

Acesta include instructiuni pentru declararea atributelor, tipurilor, rolurilor, regulilor de tranzitie a rolurilor, etc. Mai jos sunt prezentate cateva exemple de instructiuni in care se vede modul cum TE si RBAC interactioneaza.

```
user root roles {staff_r sysadm_r}
```

In acest exemplu se acorda userului root rolul de a putea opera ca *staff_r* si *sysadm_r* .

```
allow system_r sysadm_r;
```

Aceasta regula permite tranzitia de la rolul *system_r* la rolul *sysadm_r*.

```
type lib_t file_type, sysadmfile;
```

Aceasta regula defineste un nou tip, *lib_t*, pentru sistemul de fisiere. Doar administratorul poate modifica aceste fisiere, astfel acest tip este definit folosind modificatorul *sysadmfile.*

```
/sbin/insmod.*
system_u: object_r:insmod_exec_t;

allow sysadm_t insmod_exec_t:file x_file_perms;
allow sysadm_t insmod_t:process transition;
allow insmod_t insmod_exec_t:process {entrypoint execute};
allow insmod_t sysadm_t:fd inherit_fd_perms;
allow insmod_t self:capability sys_module;
allow insmod_t sysadm_t:process sigchld;
```

[5]

2.4 Concluzii

SELinux este o aplicatie din arhitectura FLASK. Ea reprezinta un mecanism de tip MAC ce incorporeaza IBAC, RBAC si TE. Sintaxele folosite sunt destul de greoaie si neobisnuite, dar permit o flexibilitate scazuta in configurarea politicilor de securitate.

3. Securitatea folosind grsecurity

SELARU CATALIN

Grsecurity este o multime de “patch-uri” de aproximativ 300K construite in incercarea de a imbunatatii securitatea linux. Potrivit creatorului aceasta multime a fost contruita pentru a atinge patru scopuri. Prima, sa ofere libertatea in posibilitatea configurarii. Al doilea, sa ofere protectie impotriva tuturor breselor de securitate din spatiul de adrese. Al treilea, grsecurity include un nivel ridicat la listele de control de acces ale sistemului. Iar al patrulea, sa opereze pe mai multe arhitecturi de procesoare si sisteme de operare.

3.1 Listele de control al accesului in grsecurity

Controlul accesului prin delegare este implementat in grsecurity folosind Liste ale controlului de acces(ACLs)[6].

Listele ACLs din grsecurity sunt construite din subiecte(procese) si obiecte(fisiere, resure, liste IP). Sctructura ACL defineste restrictiile asupra subiectelor si obiectelor. Mostenirea este implementata pentru a reduce necesarul de configuratie asemanator in binare diferite.

Listele ACL au o structura generala dupa cum urmeaza:

```
<path of subject process><optional subject modes>{
```

```
                        <file object><optional object modes>
                        [+|-]<capability>
                        <resource name><soft limit><hard limit>
                        connect{
                             <ip>/<netmask>:<low port>-<high port>
<type><proto>
                                  }
                        bind{
                             <ip>/<netmask>:<low port>-<high
port><type><proto>
                             }
     }
```

Acest mod de implementare creaza o forma de delegare bazata pe procese de a controla accesul. Este posibila restrictionarea a unui proces, prin setarea a ceea ce poate face si a ce nu poate face. In plus, poate fi restrictionat accesul la un obiect oricarui user, chiar si userului *root*. Aceste restrictii nu pot si modificate de userii normali. Sistemul ofera acces al controlului bazat pe roluri.

3.1.1 Listele de control al accesului IP

Aceste liste permit administratorilor sa controlez multe lucruri, cu ar fi IP-urile si porturile prin care procesele pot comunica cu serverele, ce IP si pe ce porturi se pot realiza conexiuni de la distanta, ce tipuri de socketuri poate folosi procesul, ce protocoale de socket sunt permise a fi folosite.

Formatul unei astfel de liste este:

```
connect{
      <ip>/<netmask>:<low port>-<high port> <type><proto>
}
bind{
      <ip>/<netmask>:<low port>-<high port><type><proto>
}
```

Un exemplu de lista IP ACL valida este:

```
connect{
      192.168.1.2/24   stream dgram tcp udp
      134.55.22.12/24:80    stream tcp
      }
bind{
      192.168.1.2/24:1024-65535 any_type any_proto
}
```

3.1.2 Gradm tool

Grsecurity include un utilitar foarte puternic numit gradm. Acesta este folosit pentru cofigurarea listelor ACL. In special gradm modifica listele ACL, comanda politici sigure, optimizeaza listele, si ofera un mod de invatare pentru o mai buna construire a listelor ACL.

Modul de invatare in grsecurity este bazat pe procese. Poate fi folosit de un singur proces in timp ce restul sistemului sa ramana protejat. Poaet fi folosit pentru a creea liste de control al

accesului optimizate pentru noi procese intru mediu particular. Modul de invatare suporta folosirea fisierelor, capabilitatilor, resurselor si a socket-urilor.

3.2 Mecanisme de securitate aditionale

Grsecurity include multe mecanisme de securitate in afara listelor de control al accesului. Include PaX, optiuni de salvare a istoricului, protectia executabilelor, protectia retelelor si altele.

3.2.1 Protectia sistemului de fisiere

In linux sistemul de fisiere /proc este un pseud-sistem de fisiere folosit ca o interfata pentru structurile de date ale kernelului. De aceea cea mai mare parte a acestui sistem de fisiere permite accesul doar pentru citire, iar unele fisiere permite schimbare unor variable din kernel[2]. Uneori acesta este tinta atacurile, de acceea grsecurity vine cu mecanismul numit “Proc restrictions”.

Cand este folosita protectia sistemului de fisiere, avem acces la mai mult configurari ale restrictiilor. Prima, restrictia doar pentru user, asigura ca doar userul sa detina informatii despre procesul pe care acesta le executa. Restrictii aditionale sunt disponibile pentru a ascunde informatii legate de procesor si alte componente hardware. In plus “Linkin restrictions” ofera posibilitatea userului de a folosi legaturi simbolice si imposibilitatea creeri hrad-link-urilor catre fisierele care nu ii apartin. O ultima configurare, numita “FIFO restrictions”, blocheaza userii sa scrue in directoarele FIFO daca nu fac parte din grupul care detine aceste directoare[3].

3.2.2 Protectia executabilelor

Acest tip de protectia este folosit de grsecurity, de vreme ce cele mai multe atacuri se folosesc de rularea proceselor. Daca “Enforce RLIMIT_NPROC” este activat, resursele folosite de proces in timpul apelari functiei execve() sunt limitate. “Emesg restrictions” previn non-userii in a folosi dmesg pentru a vedea istoricul bufferului. Alte optiuni includ PID-uri generate random si “Trusted path execution”.

3.2.4 Protectia spatiului de adrese PaX

Multe vulnerabilitati ale sistemului linux, cum ar fi supraincarcarea bufferului, profita de modul in care sistemul linux utilizeaza memoria. Proiectul PaX incearca sa previna si contine mecanisme impotriva vulnerabilitatilor care dau posibilitatea unui atacator de a ataca spatiul de adrese al sarcinilor.

Noile coduri executabile pot fi introduse in spatiul de adrese al sarcinilor in doua moduri. Primul, prin creeare mapari executabilului. A doua, prin modificarea unei mapari deja existente. PaX incearca sa foloseasca a doua metoda, lasand prima metoda in responsabilitatea mecanismelor de control al accesului.

Pentru a combate problema modificarii mapari de scriere/executie, PaX vine cu o noua

cateogie de posibilitati numita NOEXEC. Filosofia NOEXEC spune ca daca o anumita data din spatiu de adrese al sarcinilor nu are nevoie sa fie executabila, atunci sa nu fie. Aceste pagini vor fi marcate ca non-executabile. Mai mult, daca o aplicatie nu are nevoie sa genereze cod in timpul executiei, atunci nu ar trebui sa o faca. PaX ar trebui sa aiba posibilitatea de a preveni tranzitiile tipului paginilor de executie la cele de scriere.

Implementarea PaX in linux este impartita in doua seturi importante de trasaturi, NOEXEC(PAGEEXEC si SEGMEXEC) si MPROTECT care protejeaza restrictiile paginilor. PAGEEXEC implementeaza paginilie non-executabile folosind logica de paginare IA-32 a procesorului. SEGMEXEC, pe de alta parte, implementeaza paginile non-executabile folosind logica de segmentare IA-32 a procesorului. MPROTECT incearca sa previna introducerea unor noi coduri executabile in spatiul adreselor de sarcini.

3.3 Concluzii

Grsecurity este un pachet de patch-uri care incearca sa imbunateasca securitatea linux in mai multe feluri. In primul rand ofera sistemul listelor ACL. In al doilea rand mentine un istoric al sistemului pentru detectia atacurilor. PaX este folosit pentru a preveni atacurile prin protejarea spatiului de adrese. Aceasta combinatie de utilitare detecteaza, previn si impiedica atacurile.

4. Referinte

[1] M. Bishop – *Computer Security Art and Science*
Addison-Wesley, 2002.

[2] D. Bovet and M. Cesati – *Understanding the Linux Kernel.* O'Reilly & Associates, 2002

[3] Gentoo Linux grsecurity Guide
http://www.gentoo.org/proh/en/hardened/grsecurity.xml

[4] P. Loscocco and S. Smalley – *Meeting Critical Security Objectives and Security-Enhanced Linux,* In *Proceedings of the 2001 Ottawa Linux Symposium,* July 2001

[5] S. Smalley – *Configuring the SELinux Policy.* Tehnical report 02-007, NSA and NAI Labs, February 2002

[6] Brad Spengler – *Grsecurity ACL Documentation* V1.5, April 1, 2003