Membuat Distro Linux Untuk Security

Anom Sulistyo
Politeknik Elektronika Negeri Surabaya
Kampus ITS Keputih Sukolilo Surabaya 60111, Indonesia
E-mail:anomali77@mailworks.org

Abstrak

Linux termasuk sistem operasi yang stabil dan pada perkembanganya banyak perusahaan yang menggunakan linux sebagai server, namun bug kernel juga terus berkembang dan mengancam privasi dan data perusahaan tersebut yang bergantung pada sistem yang ada. Karena itu diperlukan peningkatan pada keamanan sistem komputer. Pondasi dari keamanan sistem komputer ini adalah peningkatan keamanan pada level kernel.

Grsecurity adalah salah satu dari sekian proyek sekuriti kernel yang menawarkan Akses kontrol, free konfigurasi, bufferoverflow proteksi serta kompatibel pada segala jenis arsitektur komputer[1][2].

Pada penelitian ini penulis akan menggubah suatu variant linux untuk meningkatkan pertahanan sistem komputer.

Proses nya adalah dengan membangun kernel yang disertai patch kernel sekuriti kemudian menerapkannya pada suatu variant linux.

Pada penelitian ini akan dihasilkan sebuah distro linux yang dapat berjalan secara live cd tanpa harus di-install. Distro linux tersebut memiliki akses kontrol sistem yang berupa RBAC(Role Based Access Control) dan dapat mencegah serangan eksploitasi yang memanfaatkan bug pada kernel.

Kata kunci – Grsecurity , Bug kernel, RBAC

1. PENDAHULUAN

Bagi penguna Linux, Linux memang sudah termasuk salah satu sistem operasi yang cukup stabil, tapi terkadang setiap software maupun sistem operasi pasti mempunyai lubang security. Salah satu bug yang berbahaya bagi penguna linux adalah bug kernel. Begitu berbahayanya sampai bug kernel masih merupakan topik yang cukup hangat untuk dibicarakan. Privasi perusahaan terancam hanya karena beberapa bug pada kernel yang dapat membuat kita menyesal, belum lagi data perusahaan kita yang sangat bergantung kepada sistem yang ada.

Salah satu contoh bug linux yang ditemukan akhir tahun lalu adalah Bug Null pointer dereference. Berdasarkan advisories yang dirilis oleh Tavis Ormandy dan Julien Tinnes (Google Security Team) kepada public. Bug tersebut merupakan NULL pointer dereference yang disebabkan oleh "incorrect proto_ops initializations" pada kernel linux (CVE-2009-2692). Berdasarkan summary CVE:

"The Linux kernel 2.6.0 through 2.6.30.4, and 2.4.4 through 2.4.37.4, does not initialize all function pointers for socket operations in proto_ops structures, which allows local users to trigger a NULL pointer dereference and gain privileges by using mmap to map page zero, placing arbitrary kode on this page, and then invoking an unavailable operation, as demonstrated by the sendpage operation (sock_sendpage function) on a PF_PPPOX socket. "

Pada advisories mereka yang dipublish oleh beberapa mailing-list security diberikan juga bagaimana metode untuk men-trigger bug tersebut.

Pada dasarnya bug atau lubang sekuriti tersebut adalah hal yang umum bagi programmer dimana penyebabnya adalah kekurangan proses checking dalam suatu aplikasi.Programmer yang baik biasanya akan melakukan beragam test case terhadap aplikasi miliknya , hal in dilakukan untuk melihat sejauh mana aplikasi tersebut dapat menghandle berbagai situasi yang akan ditemukan pada saat digunakan oleh seorang user. Namun biasanya disebabkan oleh aplikasi yang sangat kompleks, ataupun dikarenakan berbagai macam alasan lainnya bug jenis ini tidak dapat dihindari sekalipun oleh seorang programmer tingkat tinggi.

Berangkat dari permasalahan tersebut maka peneliti akan mengembangkan suatu variant linux untuk keamanan sistem komputer pada level kernel dengan menggunakan distro slax sebagai basis nya

Prosesnya adalah dengan membangun kernel yang disertai sekuriti patch kemudian menerapakannya pada distro melalui remastering.

Alasan peneliti menggunakan distro Slax :

1. Slax merupakan distro yang diturunkan dari slackware linux yakni distro linux tertua di dunia. Distro ini sangat sering dipakai sebagai
server dan terkenal dengan stabilitas sistemnya.

2. Slax merupakan distro yang meniru slackware dan BSD system dengan kemudahan konfigurasinya, ini akan membuat pengguna dengan mudah mengkonfigurasi sistemnya dengan melakukan pengeditan file konfigurasi.

3. Slax merupakan salah satu distro tercepat, tidak menghabiskan memory yang besar.

4. Slax merupakan distro yang berukuran kecil akan tetapi memilki software aplikasi standar yang dapat dipakai oleh pengguna secara langsung. Hal ini dikarenakan slax menggunakan sistem kompresi yang akan di load pada saat sistem hendak dijalankan.

5. Slax menjunjung tinggi open source dengan menyertakan source pada direktori source di dalam sistemnya

1.2. Klasifikasi keamanan

Klasifikasi keamanan berdasarkan elemen sistem adalah sebagai berikut :

1. Keamanan sistem jaringan

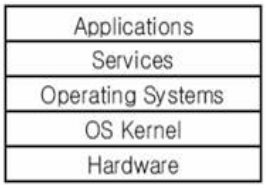
Fokus kepada saluran (media) pembawa informasi

2. Keamanan Aplikasi

Fokus kepada aplikasinya sendiri, termasuk di dalamnya adalah database

3. Keamanan sistem computer

Fokus kepada keamanan dari komputer (endsystem), termasuk Operating system (OS).



gambar 1 lapisan pada sistem komputer

Berdasarkan pada lapisan sistem komputer untuk menunjang keamanan yang lebih kompleks maka perlu dilakukan pengamanan pada lapisan pengontrol segala aktivitas dan proses yaitu level kernel.

Berikut adalah beberapa proyek keamanan kernel :

1. Openwall patch yang melindungi dari serangan stack, buffer overflow, /tmp dan beberapa serangan.

2. LIDS (Linux Intrusion Detection) , pemanfaatan capability bit lebih tinggi lagi, sehingga bisa diatur model kepemilikan, lebih ditail (misal walau root ketika sistem beroperasi tetap tak bisa menghapus log atau meload module, dlsb)

3. RSBAC (Rule Set Base Access Control) untuk menambah kontrol akses pada sistem linux. Dengan cara ini seorang pengguna dapat diatur akses-nya sesuai dengan "role" pada organisasi.

4. Medusa yang menerapkan model virtual space untuk mengakses object dengan menggunakan matrix access. Medusa ini memungkinkan sistem memiliki kebijakan akses yang lebih luwes (RSBAC, MAC, atau lainnya)

5.SElinux (FLASKModel) merupakan suatu sistem Linux yg dikembangkan oleh NSA sehingga model akses, model akses sistem call menjadi lebih aman dan tercatat.

6.Grsecurity , merupakan sistem ACL yang dapat membatasi akses ke berkas, kapabilitas, sumber daya komputasi dan atau socket ke semua pengguna termasuk root. Fitur lain dalam grsecurity ini termasuk Openwall ,RBAC, dan PAX *address space protection.*

1.3 Grsecurity

Grsecurity merupakan salah satu alternatif sekuriti yang ada dari sekian banyak yang melakukan pengamanan terhadap sistem kita dari segi lokal. Sering bug yang ada pada kernel dapat di-exploit begitu saja dengan kode-kode program yang beredar bebas di Internet. menurut brad splenger, perancang grsecurity, patch ini memiliki 4 tujuan. Pertama, grsecurity menawarkan pengoperasian tanpa perlu konfigurasi. Kedua, memberikan perlindungan terhadap semua jenis *address space modification bugs.*Berikutnya, grsecurity meliputi banyak Acces Control List system dan Auditing System. Terakhir, dapat dioperasikan pada beragam arsitektur komputer dan sistem operasi.

1.3.1. Access Control List

Access Control List adalah sekumpulan peraturan yang membatasi program program dan user di dalam sebuah sistem. Acces Control List digunakan dengan tujuan membatasi pengunaan files, resources, capability, dan sockets oleh semua penguna termasuk user/root. Akses kontrol yang digunakan pada grsecurity saat ini adalah RBAC (Role Based Access Control),dimana hak akses kontrol berdasarkan dari role dari setiap individu yang merupakan bagian dari sebuah organisasi. RBAC dapat menggambarkan struktur suatu organisasi. Untuk lebih jelasnya akan dijelaskan penggunaan RBAC secara lebih mendetail. Struktur policy dari RBAC adalah seperti berikut:

```
role <role1> <rolemode>
<role attributes>
subject / <subject mode>
<subject attributes>
    / <object mode>
    <extra objects>
    <capability rules>
    <IP ACLs>
    <resource restrictions>
subject <extra subject> <subject mode>
<subject attributes>
    / <object mode>
    <extra objects>
    ...
role <role2> <rolemode>
...
```

Gambar 2.2 struktur policy dari RBAC system

Policy dibuat berdasarkan role, subjek, dan objek. Role adalah abstraksi dari tradisional user dan group yang ada di linux distribusi. Subjek meliputi berbagai proses atau direktori , dan objek adalah meliputi file, kapabilitas, sumber daya (resource), PAX flags ,dan IP ACL. Adapun lokasi dari policy utama terdapat pada file /etc/grsec/policy.

1.3.2. Roles

Di dalam RBAC sistem ,role adalah sebuah kontainer yang di dalam nya berisi sekumpulan subjek yang digunakan untuk skenario tertentu.Klasifikasi role adalah sebagai berikut:

1. User roles

User roles adalah role yang secara otomatis diaplikasikan ketika suatu proses dieksekusi oleh seorang user dari suatu UID ataupun juga proses yang mengalami perubahan terhadap UID tersebut.

```
role user1 u
```

2. Group Roles

Sama seperti user roles, group roles berdasar pada GID. Group roles hanya diterapkan ketika sebuah user roles tidak cocok dengan proses uid.

```
role group1 g
```

3. Default Roles

Jika pada suatu kondisi tidak ditemukan user roles ataupun group roles terhadap suatu proses yang berjalan maka default role akan digunakan. Default role merupakan role yang dibuat dengan kondisi hampir tidak mempunyai akses ke sistem. Role ini dikonfigurasi lebih lanjut pada saat Full learning mode berjalan

```
role default
```

4. Special Roles

Special role digunakan untuk memberikan ekstra privilege kepada normal user ataupun akun. Sebagai contoh penggunaan special roles adalah memberikan ijin kepada "admin" role agar dapat melakukan konfigurasi ataupun menjalankan servis. Special role dapat juga diberikan pada user biasa untuk mengamankan akun mereka.Mekanisme special role terbagi menjadi dua yaitu dengan otentikasi dan tanpa otentikasi Special roles itu sendiri tidak akan melakukan apa apa hingga suatu non special roles(user,group,default) melakukan transisi menjadi special roles ini. Untuk melakukan transisi ke special role digunakan perintah gradm –a <rolename> bentuk dari special roles :

```
role specialauth s

role specialnoauth sN

role specialpamauth sP
```

1.3.3. Socket Policies

RBAC system mendukung policy terhadap lokal IP dan port apa saja yang dapat dijalankan pada lokal komputer, demikian juga dengan remote komputer dan port mana saja yang dapat digunakan untuk berkomunikasi dalam suatu jaringan. Hal ini diimplementasikan pada *bind* dan *connect* rules. Sintak dari rule ini adalah

```
  connect <IP/host>/<netmask>:<port/portrange>
<socket type 1> ... <socket type n> <proto 1> ...
<proto n>
  bind <IP/host>/<netmask>:<port/portrange>
<socket type 1> ... <socket type n> <proto 1> ...
<proto n>

  or:

  connect disabled
  bind disabled
```

Berikut diberikan contoh penggunaan rules :

```
subject /usr/bin/ssh o
...
connect 192.168.0.0/24:22 stream tcp
connect ourdnsserver.com:53 dgram udp
```

pada contoh ini , ssh diijinkan untuk melakukan koneksi ke server ssh manapun yang terdapat pada jaringan Class C 192.168.0.X dan juga diperbolehkan untuk melakukan DNS lookup pada server yang telah ditentukan.

```
subject /usr/bin/nc o
  ...
  bind 0.0.0.0/0:1024-65535 stream tcp
  connect 22.22.22.22:5190 stream tcp
```

berikut nya , netcat diperbolehkan untuk menggunakan port 1024 hingga 65535 pada tiap lokal interface untuk koneksi TCP..

.

2 . METODOLOGI PENELITIAN

2.2 Alat Penelitian

Sebelum melakukan pembuatan distro linux, sistem harus memiliki beberapa kriteria pokok agar pelaksanaanya tidak terganggu.

Pembuatan distro ini dijalankan pada sistem dengan spesifikasi :

1. komputer dengan spesifikasi :
   - Processor Pentium IV 3.0.GHZ
   - RAM 256
   - CD RW
   - Keyboard mouse standart
2. Perangkat lunak yang dibutuhkan
   - CD SLAX 6.0.9 dengan kernel 2.6.27.8
   - Kernel source 2.6.27.10
   - Modul AUFS,SQUASHFS
   - Modul LZMA,SQLZMA
   - Grsecurity patch
   - Tools sekuriti seperti nmap, snort,rkhunter dan lain sebagainya.

2.2. Metode Pemaketan Distro

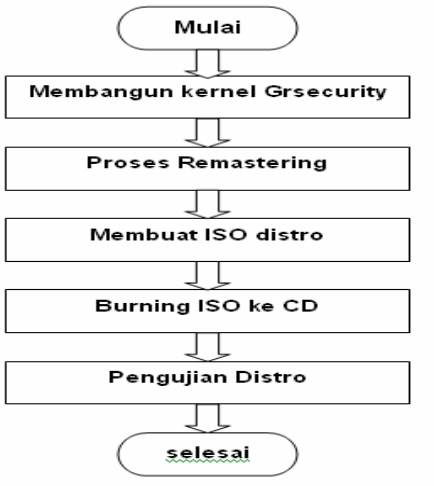
Ada beberapa metode yang dapat digunakan dalam memaketkan distro. Pertama, *Linux from Scratch* atau membangun dari nol. Metode ini cocok untuk mendalami lay-out dari Linux namun membutuhkan banyak *effort* untuk mendistribusikannya kembali. Sehingga metode ini lebih cocok digunakan secara pribadi. Kedua, membangun distro dengna cara memodifikasi distro besar yang sudah mapan.

Penelitian ini akan menggunakan metode kedua karena membutuhkan *effort* yang lebih sedikit untuk distribusi ulang hasil modifikasinya. Selain itu, perawatan paket aplikasi akan menjadi lebih efisien

karena dilakukan oleh distro yang menjadi basis. Sehingga penegmbang dapat berfokus pada sisi yang membedakan.

3. PERANCANGAN DAN PEMBUATAN DISTRO

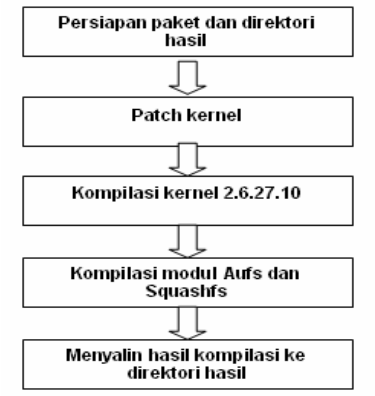
Proses pembuatan distro ini meliputi pembangunan kernel grsecurity dan kemudian menerapkan kernel tersebut pada distro slax melalui proses remastering.



gambar 2 proses pembuatan distro

3.1 Membangun kernel grsecurity

Pada tahap ini kita membangun kernel baru yaitu linux kernel 2.6.27.10 yg nantinya akan di patch dengan Grsecurity. Linux Slax menggunakan AUFS dan SQUASHFS sebagai pendukung live sistemnya oleh karena itu agar kernel baru ini dapat berjalan di live system maka kita harus melakukan patch Aufs dan Squashfs pada kernel baru ini, serta membuat modules nya. Tahapan tahapan dalam proses membangun kernel ini dapat digambarkan pada diagram berikut



gambar 3 membangun kernel grsecurity

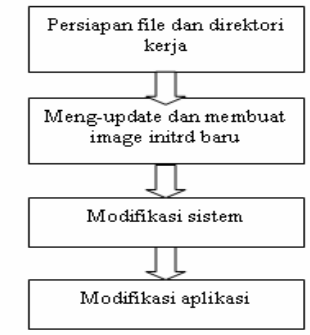
1. Proses membangun kernel ini akan kita lakukan pada direktori /usr/src karena itu kita harus melakukan ekstraksi paket paket yg dibutuhkan pada direktori tersebut dan juga diperlukan pembuatan satu direktori untuk penyimpanan file hasil kompilasi kernel.
2. Patch yg diterapkan pada kernel antara lain Grsecurity ,Aufs, Squashfs, dan Lzma.
3. Kompilasi kernel dilakukan dengan menerapkan file config dari /proc/config ke dalam *source kernel* dan pastikan squashfs tidak diaktifkan pada konfigurasi kernel. Adapun sekuriti level yang diterapkan pada distro nirvana ini adalah level high,hal ini dapat dilakukan pada menu konfigurasi kernel yaitu dengan cara masuk ke ***security options->Grsecurity->Security level*** dan kemudian pilih high. Dengan level high ini fitur yang diaktifkan adalah sebagai berikut :

High Security Level mempunyai konfigurasi dari Low dan Medium security level dan:

a. Additional /proc restrictions
b. Chmod restrictions in chroot
c. No signals, ptrace, or viewing processes outside of chroot
d. Capability restriction in chroot
e. Deny fchdir out of chroot
f. Priority restrictions in chroot

3.2 Proses Remastering

Proses remasteing distro dapat digambarkan pada diagram di bawah ini



Gambar 4 proses remastering

1.Proses remastering ini dimulai dengan menyiapkan direktori kerja yang meliputi remaster, ekstrak , initrd .
2. Pada tahap kedua kita akan melakukan pembaharuan *modules* pada initramdisk yang telah di *mount* pada direktori /tmp/initrd, dimana pada initramdisk ini proses pembentukan *live system* dimulai. Adapun *source* dari *modules* adalah /tmp/linux-26.27.10 yang merupakan direktori hasil kompilasi kernel yang baru.
3. Untuk memodifikasi sistem dilakukan dengan mengubah file ./slax/base/001-core.lzm yang terdapat pada direktori /tmp/remaster.Tidak seperti versi 5 yang menggunakan ekstensi *.mo untuk modulnya, Slax versi 6 menggunakan ekstensi *.lzm yang dibuat dengan metode kompresi LZMA. Modifikasi ini meliputi penggantian hostname dan meng-update modul sistem.
4. Aplikasi yang ditambahkan pada distro nirvana ini meliputi *security tool* dan *grsecurity administration*,

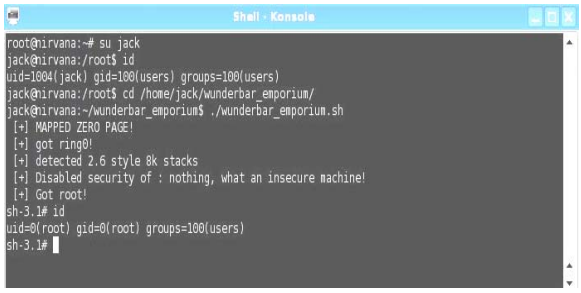
aplikasi tersebut nantinya akan di load sebagai modul pada saat proses *live cd* berjalan.

4. PENGUJIAN

Distro yang dibangun ini merupakan turunan dari distro slax 6.0.9 dan patch kernel yang diaplikasikan adalah grsecurity. Pengujian distro ini meliputi pengujian ekploit dan uji RBAC sistem.

4.1 Tahap Pengujian Eksploit

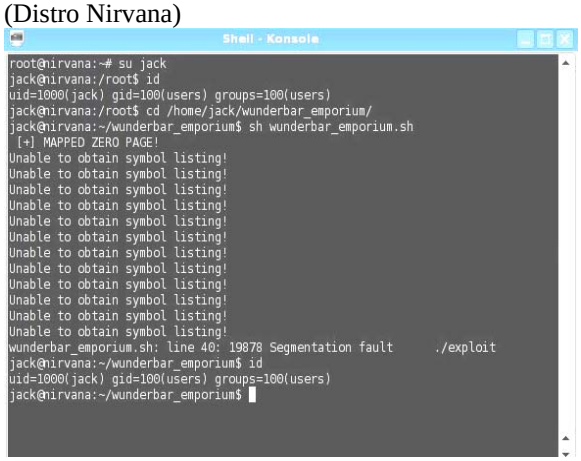
1. Menjalankan eksploit dengan user ‘jack‘ yg diasumsikan sebagai attacker yg ingin memperoleh akses root



Gambar 5 eksploit pada slax

Eksploit di atas dijalankan pada kernel yang belum di patch dengan Grsecurity (Distro Slax).

2. Menjalankan eksploit pada distro yang dibangun (Distro Nirvana)



Gambar 6 eksploit pada nirvana

4.1.1. Analisa Pengujian Eksploit

1. Eksploitasi berjalan sukses pada distro slax dikarenakan pada linux slax terdapat bug kernel Null Pointer dereference , hal ini dilakukan dengan menulisi zero page memory kemudian men-trigger bug kernel untuk mengeksekusi shell code di zero page memory

2. Eksploitasi mengalami kegagalan pada distro nirvana , dimana distro ini dilengkapi dengan grsecurity. Proses eksploit gagal setelah melakukan mapping zero page, penyebab kegagalan ini adalah eksekusi pada zero page tidak diijinkan sehingga program mengembalikan nilai error atau segmentation default.

No	Tahap	Tujuan	Hasil	
			SLax	Nirvana
1	Trigger bug kernel	Mendapatkan Ring 0 dan mengeksekusi shell code	Ok	Gagal
2	Merubah uid proses menjadi 0	Mendapatkan akses root	Ok	Gagal
3	Eksekusi execl	Membangkitkan shell /bin/sh	Ok	Gagal

Tabel 1 hasil pengujian eksploit

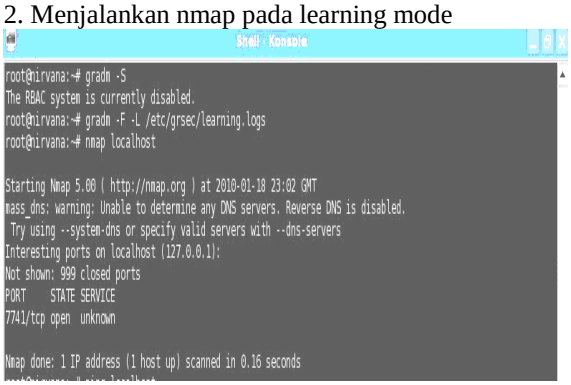
4.2 Tahap pengujian RBAC sistem

Pengujian ini menggunakan aplikasi nmap yang mana akan dijalankan pada Role default dan learning mode.

1. Menjalankan nmap pada Role default

Gambar 7 nmap pada role default

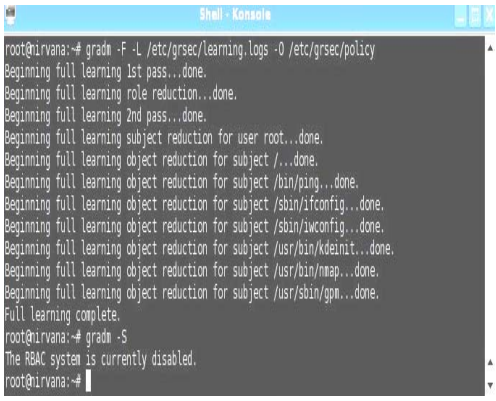
2. Menjalankan nmap pada learning mode



Gambar 8 nmap pada learning mode

Setelah melakukan learning mode maka kita akan mencoba kembali menjalankan nmap pada role default nantinya.

3. Men-generate policy yang baru



Gambar 9 generate policy baru

4. policy untuk subject /ust/bin/nmap sebagai berikut

#cat /etc/grsec/policy

```
subject /usr/bin/nmap o {
/                               h
/dev                            h
/dev/tty                        r
/dev/urandom                    r
/etc                            r
/etc/grsec                      h
/etc/ssh                        h
/etc/shadow                     h
/etc/shadow-                    h
/etc/gshadow                    h
/etc/gshadow-                   h
/etc/ppp/chap-secrets           h
/etc/ppp/pap-secrets            h
/etc/samba/smbpasswd            h
/lib                            rx
/proc                           h
/proc/meminfo                   r
/usr                            h
/usr/bin                        h
/usr/bin/nmap                   x
/usr/lib                        h
/usr/lib/libgcc_s.so.1          rx
/usr/lib/libpcre.so.0.0.1       rx
/usr/lib/libstdc++.so.6.0.9     rx
/usr/local                      h
/usr/local/share                r
/root
        -CAP_ALL
        +CAP_NET_RAW
        bind 0.0.0.0/32:0 dgram ip
        connect 0.0.0.0/32:0 raw_sock
raw_proto
}
```

4.2.1 Analisa pengujian RBAC sistem

1. Nmap tidak dapat menggunakan socket pada role default dikarenakan tidak ada policy untuk melakukan hal tersebut.
2. Learning mode merekam aktivitas nmap dan hasil dari policy untuk nmap adalah :kapabilitas CAP_NET_RAW yang mengijinkan akses pada socket.

4.3 Hasil pengujian Distro

Berdasarkan pengujian di atas didapatkan data seperti pada tabel 1.

No	Deskripsi Pengujian	Skenario	Hasil yang diharapkan	Hasil
1	Menguji proses dan mode booting	Menjalankan semua mode booting pada komputer	Proses booting sesuai dengan mode nya	Ok
2	Menguji eksploitasi sistem	Menjalankan skrip eksploit dari userland	Proses eksploitasi dapat digagalkan	Ok
3	Menguji aplikasi proteksi oleh PAX	Menjalankan perintah nmap	Dengan konversi header nmap berjalan tanpa *error*	Ok
4	Menguji RBAC sistem pada role default	Menjalankan nmap pada role default	Dengan learning mode nmap berjalan tanpa error	Ok

Tabel 2. hasil pengujian distro

5. KESIMPULAN

Dari hasil uji coba penelitian ini dapat diperoleh beberapa kesimpulan antara lain:

- Distro nirvana dapat berjalan secara live cd tanpa harus diinstall ke hardisk.
- Distro nirvana dapat menggagalkan ekploitasi sistem.
- Distro nirvana memiliki akses kontrol sistem yang berdasar pada role.
- Pada distro nirvana disertakan beberapa tools sekuriti untuk pertahanan sistem.

6. DAFTAR PUSTAKA

[1].Michael Fox, John Giordano, Lori Stotler,Arun Thomas SELinux and grsecurity: A Side-by-Side Comparison of Mandatory Access Control and Access Control List Implementations, April 3,2004

[2] Brad Spengler. Detection, Prevention, and Containment: A Study of grsecurity. http://grsecurity.urc.bl.ac.yu/grsecurity-slide.ppt.

[3] Brad Spengler. Grsecurity ACL Documentation V1.5, April 1, 2003. http://www.grsecurity.org/gracldoc.pdf.

[4] Slax. (2008). *Kumpulan Modul Slax*.[Online]. Tersedia: http://slax.org/modules.php. [5 Oktober 2008].

[5] Slax. (2008). *Dokumentasi Slax*.[Online]. Tersedia: http://slax.org/documentation.php. [5 Oktober 2008]

[6].http://xorl.wordpress.com/2009/08/18/cve-2009-2692-linux-kernel-proto_ops-null-pointer-dereference/

[7].http://www.grsecurity.net/~spender/wunderbar_emporium2.tgz