บทที่ 6

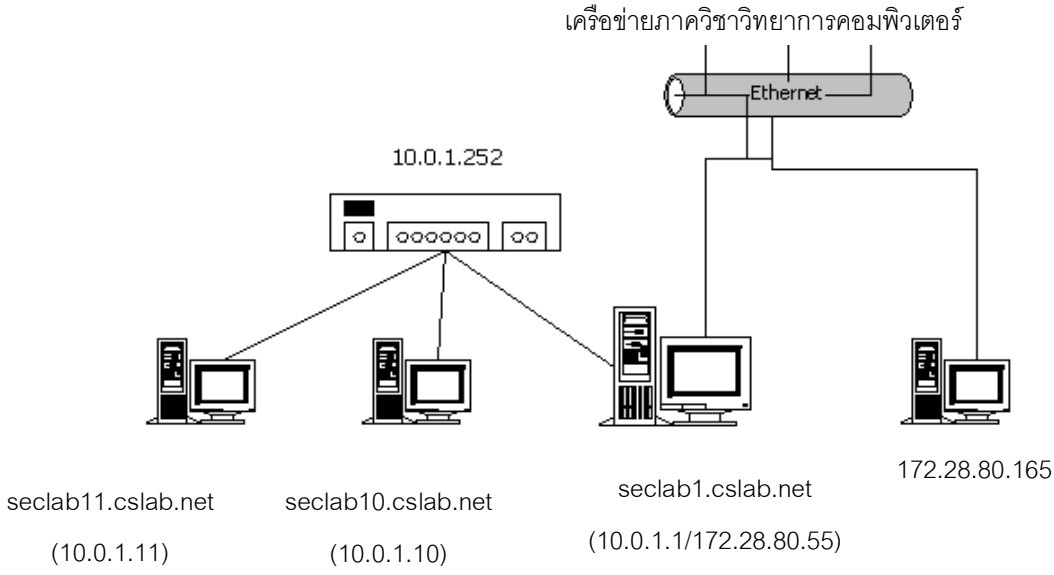
การทดสอบระบบตรวจจับการบุกรุก

6.1 บทนำ

สำหรับบทนี้จะกล่าวถึงผลที่ได้จากการศึกษาโดยจะแบ่งได้เป็น 2 ส่วนคือ โดยส่วนแรกเป็นผลการทดสอบระบบตรวจจับการบุกรุก โดยอธิบายถึงสภาพแวดล้อมและเครือข่ายที่ใช้ในการทดสอบ วิธีการทดสอบ ผลการทดสอบการบุกรุกและประสิทธิภาพของระบบที่พัฒนาขึ้น ส่วนที่สองเป็นผลการศึกษาเพื่อวิเคราะห์เพื่อหาจุดสมดุลในการกำหนดว่าควรจะมีการบันทึกเหตุการณ์หรือกิจกรรมใดบ้างของระบบ ให้เหมาะสมกับทรัพยากรที่มีอยู่ในระบบและเป็นประโยชน์สำหรับการวิเคราะห์เพื่อตรวจจับการบุกรุก

6.2 สภาพแวดล้อมและเครือข่ายที่ใช้ทดสอบระบบตรวจจับการบุกรุก

การบุกรุกระบบและการทดสอบระบบตรวจจับการบุกรุกที่พัฒนาบนเครือข่ายที่จัดทำขึ้นประกอบด้วยเครื่องคอมพิวเตอร์ทั้งหมดจำนวน 4 เครื่อง และเครือข่ายทดสอบนี้จะเชื่อมต่อกับเครือข่ายของภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ ซึ่งแสดงได้ดังภาพประกอบที่ 6.1



ภาพประกอบที่ 6.1 เครือข่ายที่สร้างขึ้นเพื่อทดสอบการทำงานของระบบที่พัฒนา

จากภาพประกอบที่ 6.1 รายละเอียดของเครื่องคอมพิวเตอร์มีดังนี้

1. เครื่อง seclab1.cslab.net ทำหน้าที่เป็น Gateway ในการเชื่อมต่อระหว่างเครือข่ายที่จัดทำขึ้นกับเครือข่ายของภาควิชาวิทยาการคอมพิวเตอร์ ใช้ระบบปฏิบัติการ Linux RedHat 7.2
2. เครื่อง seclab10.cslab.net เป็นเครื่องเป้าหมายของการบุกรุก ใช้ระบบปฏิบัติการเป็น Linux RedHat 7.2 โดยมีการติดตั้งโปรแกรม Sendmail 8.11.6-3 เพื่อทำหน้าที่เป็นเมล์เซิร์ฟเวอร์ และระบบตรวจจับการบุกรุก
3. เครื่อง seclab11.cslab.net เป็นเครื่องที่ใช้สำหรับการบุกรุก ใช้ระบบปฏิบัติการเป็น Linux RedHat 7.2 และติดตั้งโปรแกรม Sendmail 8.11.6-3 ทำหน้าที่เป็นเมล์เซิร์ฟเวอร์
4. เครื่อง 172.28.80.165 เป็นเครื่องที่ใช้สำหรับการบุกรุก ใช้ระบบปฏิบัติการ Windows 98

6.3 การทดสอบระบบตรวจจับการบุกรุก

สำหรับการทดสอบส่วนนี้จะเป็นการทดสอบระบบตรวจจับการบุกรุกโดยเป็นการทดสอบว่ากฎแต่ที่ละข้อที่กำหนดขึ้นมาในบทที่ 5 สามารถใช้ตรวจจับเหตุการณ์ต่างๆ ได้ตามที่ต้องการหรือไม่ โดยในการโจมตีจะใช้โปรแกรมจากอินเทอร์เน็ตทำการบุกรุกตามเงื่อนไขเหตุการณ์ของกฎแต่ละข้อ ซึ่งเหตุการณ์และวิธีการทดสอบเป็นดังนี้

6.3.1 การทดสอบการตรวจจับไวรัส

กฎที่ใช้สำหรับการตรวจจับไวรัสเป็นดังนี้

```
Virus (\S+) found in file::Found virus $1::Correlation-*** be careful
***::email=ssupacho::0::3::300
```

กฎข้อนี้กำหนดว่าหากมีเหตุการณ์พบไวรัสชนิดเดียวกันจำนวน 3 ครั้งในระยะเวลา 300 วินาทีก็ให้แจ้งเตือนโดยการส่งเมล์ไปยังบัญชีผู้ใช้ชื่อ ssupacho การทดสอบทำโดยการส่งเมล์จากบัญชีผู้ใช้บนเครื่อง seclab1.cslab.net พร้อมแนบไฟล์ที่มีโปรแกรมไวรัสไปด้วย ซึ่งไวรัสที่แนบไป คือ

- Blaster
- Bagle

ผลการทดสอบพบว่าระบบที่พัฒนาขึ้นสามารถตรวจจับไวรัสได้และส่งเมล์แจ้งไปยังบัญชีผู้ใช้ชื่อ ssupacho ตามที่ได้กำหนดไว้ในกฎของการตรวจจับ ซึ่งตัวอย่างเมล์แจ้งเตือนเป็นดังนี้

```
Date: Sun, 19 Jun 2005 14:08:54 +0700
From: laid@seclab10.cslab.net
To: ssupacho@seclab10.cslab.net
Subject: *** WARNING *** Found virus 'W32/Blaster-A'


Event
======
 Found virus 'W32/Blaster-A'

Log Messages(s)
===============
 Jun 19 02:15:54 seclab10 MailScanner[1248]: >>> Virus 'W32/Blaster-A' found in
file
./j5IJEYS06026/Win32.Msblast.Worm.zip/msblast.exe
 Jun 19 02:16:09 seclab10 MailScanner[1248]: >>> Virus 'W32/Blaster-A' found in
file
./j5IJEYS06026/Win32.Msblast.Worm.zip/msblast.exe
 Jun 19 02:17:48 seclab10 MailScanner[1423]: >>> Virus 'W32/Blaster-A' found in
file
./j5IJGbS06177/Win32.Msblast.Worm.zip/msblast.exe

Number of event occured 3 time(s)

Solution
========
Correlation-*** be careful  ***
```

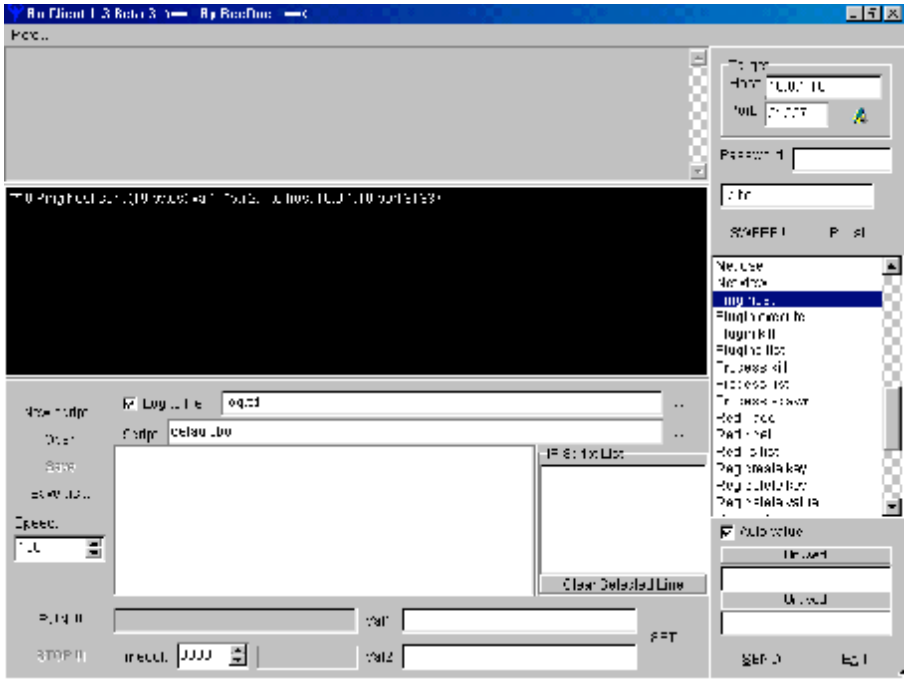
จากการทดสอบพบว่าระบบสามารถตรวจจับไวรัสซึ่งอยู่ในไฟล์ต่าง ๆ ที่แนบมาเมล์ได้ ทั้งนี้โดยทำงานร่วมกับโปรแกรม MailScanner และ Sophos Anti-virus

6.3.2 การทดสอบการตรวจจับโปรแกรมม้าโทรจัน

กฎที่ใช้สำหรับการตรวจจับโปรแกรมม้าโทรจัน ชื่อ Back Orifice เป็นดังนี้

```
Back Orifice Traffic detected::Found trojan - Back Orifice::Correlation-*** plase
check trojan horse program in your system ***::email=ssupacho::0::0::0
```

กฎข้อนี้กำหนดว่าหากมีเหตุการณ์ที่พบว่ามีข้อมูลแพ็กเก็ตของโปรแกรมม้าโทรจันชื่อ Back Orifice ติดต่อเข้ามา ก็ให้แจ้งเตือนโดยการส่งเมล์ไปยังบัญชีผู้ใช้ชื่อ ssupacho การทดสอบใช้โปรแกรมชื่อ BoClient ซึ่งเป็นโปรแกรมที่ใช้สำหรับการติดต่อกับโปรแกรมม้าโทรจันที่อยู่บนเครื่องเป้าหมาย ในที่นี้คือเครื่อง seclab10.cslab.net ซึ่งมีหมายเลข IP เป็น 10.0.1.10 โดยที่โปรแกรม BoClient ทำงานอยู่บนเครื่องที่มีหมายเลข IP เป็น 172.28.80.165 จอภาพของโปรแกรม BoClient แสดงได้ดังภาพประกอบที่ 6.2



ภาพประกอบที่ 6.2 จอภาพการทำงานของโปรแกรม BoClient

ผลการทดสอบพบว่าระบบสามารถที่พัฒนาขึ้นสามารถตรวจจับโทรจันนี้ได้และส่งเมล์ไปยังบัญชีผู้ใช้ชื่อ ssupacho ตามที่ได้กำหนดไว้ในกฎของการตรวจจับ ดังนี้

```
Date: Sun, 19 Jun 2005 14:08:54 +0700
From: laid@seclab10.cslab.net
To: ssupacho@seclab10.cslab.net
Subject: *** WARNING *** Found trojan - Back Orifice

Event
======
 Found trojan - Back Orifice

Log Messages(s)
===============
 Jun 19 14:08:53 seclab10 snort: [105:1:1] (spo_bo) Back Orifice Traffic detected
{UDP}
172.28.80.165:1136 -> 10.0.1.10:31337

Number of event occured 1 time(s)

Solution
========
 Correlation-*** plase check trojan horse program in your system ***
```

6.3.3 การทดสอบการตรวจจับเหตุการณ์ Ping Flood

กฎที่ใช้สำหรับการตรวจจับเหตุการณ์ Ping Flood เป็นดังนี้

```
Large ICMP Packet::Found Ping Flood attack::Correlation-***  be careful
***::email=ssupacho::600::20::60
```

กฎข้อนี้กำหนดว่าหากมีเหตุการณ์ที่พบว่ามีข้อมูลแพ็กเก็ต ICMP Echo Request ซึ่งเป็นแพ็กเก็ตแบบเดียวกับที่ได้จากคำสั่ง Ping แต่เป็นแพ็กเก็ตที่มีขนาดใหญ่กว่าปกติจำนวน 20 เหตุการณ์ในช่วงเวลา 60 วินาที ก็ให้เมล์แจ้งเตือนไปยัง ssupacho และจะหน่วงเวลาในการแจ้งเตือนไว้ 600 วินาที จึงจะแจ้งเตือนไปใหม่หากยังตรวจพบเหตุการณ์นี้อยู่ ในการทดสอบทำโดยใช้โปรแกรม ping โดยระบุขนาดแพ็กเก็ตเป็น 1024 bytes โดยทำการโจมตีจากเครื่องหมายเลข IP 10.0.1.1 ไปยังเครื่องเป้าหมายคือ 10.0.1.10 ดังนี้

```
[ssupacho@seclab1 ssupacho]$ ping -s 1024 10.0.1.10
PING 10.0.1.10 (10.0.1.10) from 10.0.1.1 : 1024(1052) bytes of data.
Warning: time of day goes back, taking countermeasures.
1032 bytes from 10.0.1.10: icmp_seq=0 ttl=255 time=1.548 msec
1032 bytes from 10.0.1.10: icmp_seq=1 ttl=255 time=586 usec
1032 bytes from 10.0.1.10: icmp_seq=2 ttl=255 time=610 usec
1032 bytes from 10.0.1.10: icmp_seq=3 ttl=255 time=632 usec
1032 bytes from 10.0.1.10: icmp_seq=4 ttl=255 time=581 usec
1032 bytes from 10.0.1.10: icmp_seq=5 ttl=255 time=580 usec
…
1032 bytes from 10.0.1.10: icmp_seq=6 ttl=255 time=574 usec
1032 bytes from 10.0.1.10: icmp_seq=7 ttl=255 time=588 usec

--- 10.0.1.10 ping statistics ---
159 packets transmitted, 159 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.481/0.550/0.755/0.040 ms
```

ผลการทดสอบพบว่าระบบที่พัฒนาขึ้นสามารถตรวจจับเหตุการณ์ Ping Flood ได้และส่งเมล์ไปยังบัญชีผู้ใช้ชื่อ ssupacho ตามที่ได้กำหนดไว้ในกฎของการตรวจจับ

```
Date: Sun, 19 Jun 2005 23:46:49 +0700
From: laid@seclab10.cslab.net
To: ssupacho@seclab10.cslab.net
Subject: *** WARNING *** Found Ping Flood attack


Event
======
 Found Ping Flood attack

Log Messages(s)
```

```
===============
 Jun 19 23:46:39 seclab10 snort: [1:499:4] ICMP Large ICMP Packet
[Classification: Potentially
Bad Traffic] [Priority: 2]: {ICMP} 10.0.1.1 -> 10.0.1.10
 Jun 19 23:46:39 seclab10 snort: [1:499:4] ICMP Large ICMP Packet
[Classification: Potentially
Bad Traffic] [Priority: 2]: {ICMP} 10.0.1.10 -> 10.0.1.1
 Jun 19 23:46:40 seclab10 snort: [1:499:4] ICMP Large ICMP Packet
[Classification: Potentially
Bad Traffic] [Priority: 2]: {ICMP} 10.0.1.1 -> 10.0.1.10
…
Jun 19 23:46:48 seclab10 snort: [1:499:4] ICMP Large ICMP Packet
[Classification: Potentially
Bad Traffic] [Priority: 2]: {ICMP} 10.0.1.10 -> 10.0.1.1

Number of event occured 20 time(s)

Solution
========
 Correlation-***  Be careful ***
```

6.3.4 การทดสอบการตรวจจับเหตุการณ์ SYN Flood

กฎที่ใช้สำหรับการตรวจจับเหตุการณ์ Ping Flood เป็นดังนี้

SYN Flood detected::possible SYN Flood attack::Correlation-*** SYN Flooding ***::email=ssupacho

กฎข้อนี้กำหนดว่าตรวจพบเหตุการณ์ SYN Flood ให้แจ้งเตือนโดยการส่งเมล์ไปยังบัญชีผู้ใช้ชื่อ ssupacho การทดสอบทำโดยใช้โปรแกรมชื่อ synflood ที่เขียนขึ้นด้วยภาษาซี ทำการโจมตีจากเครื่องหมายเลข IP 10.0.1.1 ไปยังเครื่อง 10.0.1.10 โดยโจมตีผ่านพอร์ตหมายเลข 80 และจำนวนที่ขอการเชื่อมต่อเป็น 1000 ครั้ง ซึ่งวิธีการเรียกใช้โปรแกรม synflood เป็นดังนี้

```
[ssupacho@seclab1 ssupacho]# ./synflood 10.0.1.1 10.0.1.10 80 1000
synflooding 10.0.1.10 from 10.0.1.1 port 80 1000 times
spoofing 129.62.13.81
spoofing 129.62.13.191
spoofing 129.62.13.26
…
spoofing 129.62.13.27
spoofing 129.62.13.90
```

ผลการทดสอบพบว่าระบบสามารถที่พัฒนาขึ้นสามารถตรวจจับเหตุการณ์ SYN Flood ได้และส่งเมล์ไปยังบัญชีผู้ใช้ชื่อ ssupacho ตามที่ได้กำหนดไว้ในกฎของการตรวจจับ

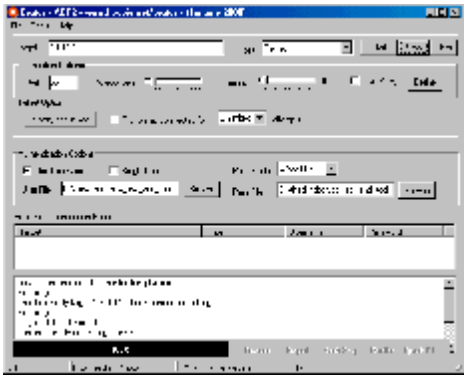
```
Date: Sun, 19 Jun 2005 18:29:54 +0700
From: laid@seclab10.cslab.net
To: ssupacho@seclab10.cslab.net
Subject: *** WARNING *** possible SYN flooding

Event
======
 possible SYN flooding

Log Messages(s)
===============
 Jun 19 18:29:53 seclab10 kernel: possible SYN flooding on port 80. Sending
cookies.

Number of event occured 1 time(s)

Solution
========
 Correlation-*** SYN FLOODING ***
```

6.3.5 การทดสอบการตรวจจับเหตุการณ์ Brute-force attack

กฎที่ใช้สำหรับการตรวจจับเหตุการณ์ Brute-force attack ซึ่งเป็นการพยายามเดารหัสผ่านของผู้ใช้ โดยการพยายามใช้ทุกค่าที่เป็นไปได้ สามารถเขียนได้ดังนี้

```
authentication failure.* rhost=(\S+)  user=(\S+)::User $2 login failure more
than 10 times from $1::*** Password Brute-force attack
***::email=ssupacho,root::0::10::60
```

กฎข้อนี้กำหนดให้ต้องมีจำนวนเหตุการณ์เกิดขึ้น 10 ครั้งในระยะเวลา 60 วินาที เท่านั้นจึงจะถือว่ามีการโจมตีเกิดขึ้นแล้วทำการแจ้งเตือนโดยการส่งเมล์ไปยัง ssupacho ในการทดสอบใช้โปรแกรมชื่อ Brutus ทำการโจมตีผ่านทางพอร์ตหมายเลข 23 โดยโจมตีจากเครื่องหมายเลข IP 172.28.80.165 ไปยังเครื่อง 10.0.1.10 จอภาพของโปรแกรม Brutus แสดงได้ภาพประกอบที่ 6.3



ภาพประกอบที่ 6.3 จอภาพการทำงานของโปรแกรม Brutus

ผลการทดสอบพบว่าระบบที่พัฒนาขึ้นสามารถตรวจจับการโจมตีด้วยวิธีนี้ได้ และมีเมล์แจ้งเตือนการเกิดเหตุการณ์ดังกล่าว

```
Date: Mon, 20 Jun 2005 01:07:56 +0700
From: laid@seclab10.cslab.net
To: ssupacho@seclab10.cslab.net
Subject: *** WARNING *** User root login failure more than 10 times from
172.28.80.165


Event
======
 User root login failure more than 10 times from 172.28.80.165

Log Messages(s)
===============
 Jun 20 01:07:48 seclab10 login(pam_unix)[14139]: authentication failure;
logname= uid=0 euid=0
tty=pts/6 ruser= rhost=172.28.80.165  user=root
 Jun 20 01:07:48 seclab10 login(pam_unix)[14140]: authentication failure;
logname= uid=0 euid=0
tty=pts/3 ruser= rhost=172.28.80.165  user=root
 Jun 20 01:07:48 seclab10 login(pam_unix)[14141]: authentication failure;
logname= uid=0 euid=0
tty=pts/7 ruser= rhost=172.28.80.165  user=root
 Jun 20 01:07:48 seclab10 login(pam_unix)[14142]: authentication failure;
logname= uid=0 euid=0
tty=pts/8 ruser= rhost=172.28.80.165  user=root
 Jun 20 01:07:48 seclab10 login(pam_unix)[14143]: authentication failure;
logname= uid=0 euid=0
tty=pts/9 ruser= rhost=172.28.80.165  user=root
 Jun 20 01:07:48 seclab10 login(pam_unix)[14145]: authentication failure;
logname= uid=0 euid=0
tty=pts/10 ruser= rhost=172.28.80.165  user=root
….
Jun 20 01:07:55 seclab10 login(pam_unix)[14177]: authentication failure;
logname= uid=0 euid=0
tty=pts/5 ruser= rhost=172.28.80.165  user=root

Number of event occured 10 time(s)

Solution
========
 ** Password Brute-force attack **
```

6.3.6 การทดสอบการตรวจจับเหตุการณ์ sniffer

กฎที่ใช้สำหรับการตรวจจับเหตุการณ์การดักจับข้อมูลในเครือข่ายของโปรแกรมจำพวก sniffer ซึ่งเขียนได้เป็นดังนี้

```
device (\S+) entered promiscuous mode::Device $1 entered promiscuous
mode::*** device $1 entered promiscuous mode, check sniffer program in
your computer***::email=ssupacho
```

กฎข้อนี้กำหนดว่าเมื่อเน็ตเวิร์กอะแดปเตอร์ทำงานในโพรมิสคูอัสโหมด ก็แสดงว่าอาจจะมีโปรแกรมจำพวก sniffer ทำงานอยู่ในระบบ สำหรับการทดสอบใช้โปรแกรมจากอินเทอร์เน็ตชื่อ sniifer

```
[root@seclab10 root]# ./sniffer
socket: 3
Received 46 bytes from 172.28.80.165 to 10.0.1.10
Protocol was 6
TCP packet had source port 1034 and dest port 22
TCP data field had:
dC
Received 46 bytes from 172.28.80.165 to 10.0.1.10
Protocol was 6
TCP packet had source port 1034 and dest port 22
```

เมื่อโปรแกรมนี้ทำงานก็จะกำหนดให้เน็ตเวิร์กอะแดปเตอร์ทำงานในโพรมิสคูอัสโหมดและผลการทดสอบพบว่าสามารถตรวจจับเหตุการณ์นี้ได้และได้มีเมล์แจ้งเตือนไปยังผู้ใช้ชื่อ ssupacho ตามที่ได้ระบุไว้ในกฎ

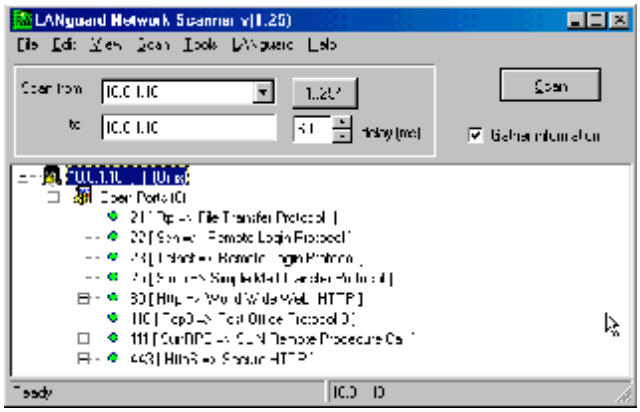
```
Date: Mon, 20 Jun 2005 02:05:53 +0700
From: laid@seclab10.cslab.net
To: ssupacho@seclab10.cslab.net
Subject: *** WARNING *** Device lo entered promiscuous mode

Event
======
 Device lo entered promiscuous mode

Log Messages(s)
===============
 Jun 20 02:05:53 seclab10 kernel: device lo entered promiscuous mode

Number of event occured 1 time(s)

Solution
========
 *** device lo entered promiscuous mode, check sniffer program in  your
computer***
```

6.3.7 การทดสอบการตรวจจับเหตุการณ์ Port Scanning

กฎที่ใช้สำหรับการตรวจจับเหตุการณ์ Port Scanning เป็นดังนี้

```
NOQUEUE: \[(.*?)\] did not issue .*? during connection to MTA::Client quit before
communicating, may be scanning service from $1::Correlation-*** be careful
***::email=ssupacho

\[([0-9\.]+)]: (VRFY|vrfy) (\S+) \[rejected\]::rejected VRFY, may be scanning
service from $1::Correlation-*** be careful  ***::email=ssupacho
```

ในการทดสอบใช้โปรแกรมชื่อ GFI LANguard Network Security Scanner โดยติดตั้งโปรแกรมนี้ไว้ในเครื่อง 172.28.80.165 แล้วทำการสแกนไปยังเครื่องเป้าหมายคือ 10.0.1.10 ซึ่งจอภาพการทำงานของโปรแกรมนี้แสดงได้ดังภาพประกอบ 6.4



ภาพประกอบที่ 6.4 จอภาพการทำงานของโปรแกรม GFI LANguard Network Security Scanner

ผลการทดสอบระบบว่าสามารถตรวจจับการโจมตีด้วยวิธีนี้ได้และได้มีเมล์แจ้งเตือนไปยังผู้ใช้ชื่อ ssupacho ตามที่ได้ระบุไว้ในกฎทั้งสองข้อ

```
Date: Mon, 20 Jun 2005 11:08:20 +0700
From: laid@seclab10.cslab.net
To: ssupacho@seclab10.cslab.net
Subject: *** WARNING *** Client quit before communicating,
    may be scanning service from 172.28.80.165

Event
======
 Client quit before communicating, may be scanning service from 172.28.80.165

Log Messages(s)
===============
```

```
 Jun 20 11:08:19 seclab10 sendmail[27871]: NOQUEUE: [172.28.80.165] did not
issue
MAIL/EXPN/VRFY/ETRN during connection to MTA
Number of event occured 1 time(s)


Solution
========
 Correlation-*** be careful  ***


Date: Mon, 20 Jun 2005 11:09:27 +0700
From: laid@seclab10.cslab.net
To: ssupacho@seclab10.cslab.net
Subject: *** WARNING *** Client quit before communicating,
    may be scanning service from 172.28.80.165


Event
======
 Client quit before communicating, may be scanning service from 172.28.80.165

Log Messages(s)
===============
 Jun 20 11:09:26 seclab10 sendmail[27885]: NOQUEUE: [172.28.80.165] did not
issue
MAIL/EXPN/VRFY/ETRN during connection to MTA

Number of event occured 1 time(s)

Solution
========
 Correlation-*** be careful  ***
```

6.3.8 การทดสอบการตรวจจับเหตุการณ์การบุกรุกโดยอาศัยช่องโหว่ของโปรแกรม

สำหรับทดสอบระบบเพื่อตรวจจับการบุกรุกโดยอาศัยช่องโหว่ของโปรแกรมต่าง ๆ ที่ทำงานอยู่บนเครื่องนั้นไม่สามารถทำได้ เพราะว่าจากการเก็บร่องรอยการบุกรุกในลักษณะในบทที่ 5 พบว่าไม่มีร่องรอยใด ๆ เกิดขึ้น ดังนั้นจึงไม่มีข้อมูลที่จะนำมาใช้ในการเขียนกฎสำหรับตรวจจับการบุกรุก

6.4 ผลการทดสอบระบบตรวจจับการบุกรุก

จากการทดสอบระบบตรวจจับการบุกรุกที่พัฒนาขึ้นสามารถสรุปผลการทดสอบได้ดังตารางที่ 6.1 ดังนี้

ตารางที่ 6.1 แสดงผลการทดสอบระบบตรวจจับการบุกรุก

เหตุการณ์การบุกรุก	ผลการตรวจจับ		กลไก/เครื่องมือที่ติดตั้งเพิ่มเติม	หมายเหตุ
	ได้	ไม่ได้		
ไวรัส	✓		MailScanner และ Sophos Anti-virus	
ม้าโทรจัน Back Orifice	✓		Snort	
Ping Flood	✓		Snort	
SYN Flood	✓		TCP SYN cookies	
Brute-force attack	✓		-	
Sniffer	✓		-	
Port Scanning	✓		-	
ช่องโหว่ของโปรแกรม	N/A	N/A	-	ไม่ปรากฏร่องรอยการบุกรุก

6.5 การทดสอบประสิทธิภาพของระบบ

การทดสอบประสิทธิภาพของระบบตรวจจับการบุกรุกที่พัฒนาขึ้นนี้จะพิจารณาถึงการใช้ทรัพยากรของระบบคอมพิวเตอร์เมื่อระบบตรวจจับการบุกรุกทำงาน โดยใช้คำสั่ง top พบว่าเมื่อระบบที่พัฒนาประกอบด้วย 2 โปรเซส คือ laid (laid เป็นชื่อของโปรแกรมที่พัฒนาขึ้น) และ tail เมื่อใช้คำสั่ง top ตรวจดูการใช้ทรัพยากร สามารถแสดงได้ดังตารางที่ 6.2 – 6.5

ตารางที่ 6.2 ตารางการใช้ทรัพยากรของระบบตรวจจับโดยใช้คำสั่ง top ในกรณีที่ไม่มีทั้งกฎการกรองและกฎการตรวจจับ

PID	USER	PRI	NI	Process SIZE (KB)	RSS	SHARE	STAT	%CPU	%MEM	TIME	COMMAND
18207	root	9	0	2372	2372	1292	S	0.0	1.8	0:00	laid
18210	root	9	0	540	540	472	S	0.0	0.4	0:00	tail

ตารางที่ 6.3 ตารางการใช้ทรัพยากรของระบบตรวจจับโดยใช้คำสั่ง top เมื่อมีกฎการกรองจำนวน 25 ข้อ และกฎการตรวจจับจำนวน 15 ข้อ

PID	USER	PRI	NI	Process SIZE (KB)	RSS	SHARE	STAT	%CPU	%MEM	TIME	COM MAND
29356	root	14	0	2432	2432	1332	S	0.0	1.9	0:00	laid
29359	root	14	0	540	540	472	S	0.0	0.4	0:00	tail

ตารางที่ 6.4 ตารางการใช้ทรัพยากรของระบบตรวจจับโดยใช้คำสั่ง top เมื่อมีกฎการกรองจำนวน 50 ข้อ และกฎการตรวจจับจำนวน 30 ข้อ

PID	USER	PRI	NI	Process SIZE (KB)	RSS	SHARE	STAT	%CPU	%MEM	TIME	COM MAND
17946	root	9	0	2448	2448	1332	S	0.0	1.9	0:00	laid
17947	root	9	0	540	540	472	S	0.0	0.4	0:00	tail

ตารางที่ 6.5 ตารางการใช้ทรัพยากรของระบบตรวจจับโดยใช้คำสั่ง top เมื่อมีกฎการกรองจำนวน 50 ข้อ และกฎการตรวจจับจำนวน 30 ข้อ เมื่อเวลาผ่านไประยะเวลาหนึ่ง

PID	USER	PRI	NI	Process SIZE (KB)	RSS	SHARE	STAT	%CPU	%MEM	TIME	COM MAND
17946	root	9	0	2524	2524	1352	S	0.0	1.9	0:00	laid
17947	root	9	0	540	540	472	S	0.0	0.4	0:00	tail

ผลที่ได้จากตารางที่ 6.2 จะเห็นว่าขนาดของโปรเซส laid ในหน่วยความจำมีขนาดเริ่มต้นเป็น 2372 KB ถือว่ามีขนาดที่เล็กและจากตารางที่6.2-6.4 เมื่อมีจำนวนกฎที่เพิ่มมากขึ้นขนาดของโปรแกรมก็เพิ่มมากขึ้นด้วยตามจำนวนกฎที่เพิ่มขึ้น และจากตารางที่6.4-6.5 เมื่อระยะเวลาผ่านไประยะหนึ่งในกรณีที่มีกฎการกรองจำนวน 50 ข้อ และกฎการตรวจจับจำนวน 30 ข้อขนาดของโปรเซสเพิ่มจาก 2448KB เป็น 2524KB ขนาดที่เพิ่มขึ้นเท่ากับ 76 KB คิดเป็น 3.10 เปอร์เซ็นต์

6.6 ผลการศึกษาเพื่อวิเคราะห์หาระดับของการบันทึกเหตุการณ์ของ syslog

สำหรับการศึกษาเพื่อศึกษาวิเคราะห์และนำเสนอระดับของการบันทึกเหตุการณ์ของ syslog เพื่อให้ได้ข้อมูลที่มีประโยชน์กับการวิเคราะห์เพื่อตรวจจับการบุกรุกสูงสุดและใช้เนื้อที่ในการเก็บเหมาะสมที่สุด ตามการออกแบบที่ได้กล่าวไว้ในหัวข้อที่ 4.4 โดยจากการศึกษาข้อมูลของเครื่อง ratree.psu.ac.th พบว่าเครื่องดังกล่าวใช้ระบบปฏิบัติการ SunOS 5.6 Generic_105181-14 sun4u sparc และมีบริการต่าง ๆ ดังตารางที่ 6.6

ตารางที่ 6.6 แสดงข้อมูลบริการต่าง ๆและโปรแกรมที่ทำงานบนเครื่อง ratree.psu.ac.th

บริการ	โปรแกรมที่ใช้
Web server	Apache 1.3.27
Mail server	Sendmail 8.11.6
FTP server	FTP server (SunOS 5.6)
telnet	UNIX(r) System V Release 4.0
Ssh	SSH-1.99-OpenSSH_2.9p2
Imap	IMAP4rev1 v12.254
pop3	POP3 v7.61 server

จากการเก็บรวบรวมข้อมูลล็อกต่าง ๆที่เกิดขึ้นทั้งหมดของเครื่อง ratree.psu.ac.th หรือเรียกโดยย่อว่า ratree โดยได้เก็บรวบรวมข้อมูลในระหว่างวันที่ 1 ถึง 31 สิงหาคม 2546 และนำข้อมูลเหล่านั้นมาวิเคราะห์ว่าเป็นข้อมูลล็อกที่มาจากโปรแกรมใดบ้าง เป็นล็อกที่เกิดขึ้นในเหตุการณ์ระดับใด การใช้พื้นที่ในการจัดเก็บทั้งหมดและแต่ละวันเป็นเท่าใด โดยแสดงข้อมูลในรูปแบบของตารางและกราฟเปรียบเทียบให้เห็นถึงสัดส่วนการใช้พื้นที่ของล็อกเหล่านั้น ดังต่อไปนี้

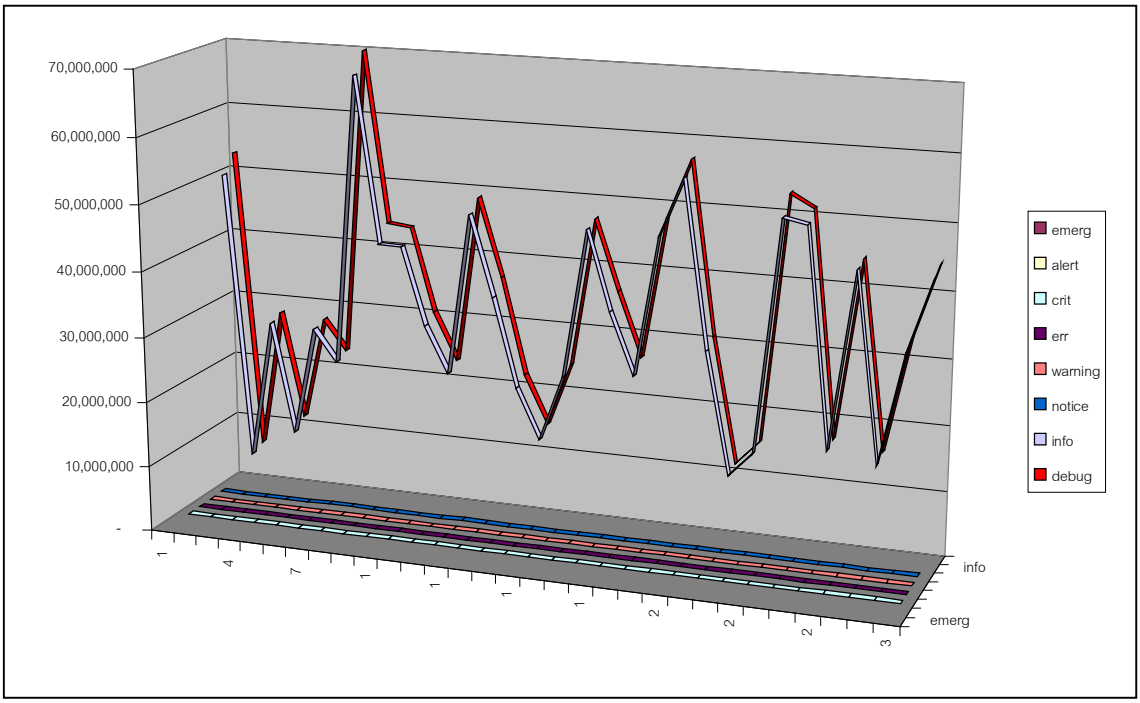
6.6.1 การใช้พื้นที่สำหรับเก็บข้อมูลล็อกในแต่ละระดับ

จากการเก็บข้อมูลล็อกในแต่ละวันที่โดยจำแนกตามระดับแสดงได้ดังตารางที่ 6.7

ตารางที่ 6.7 การใช้พื้นที่ในเก็บล็อกของแต่ละวันที่โดยจำแนกตามระดับความสำคัญเหตุการณ์

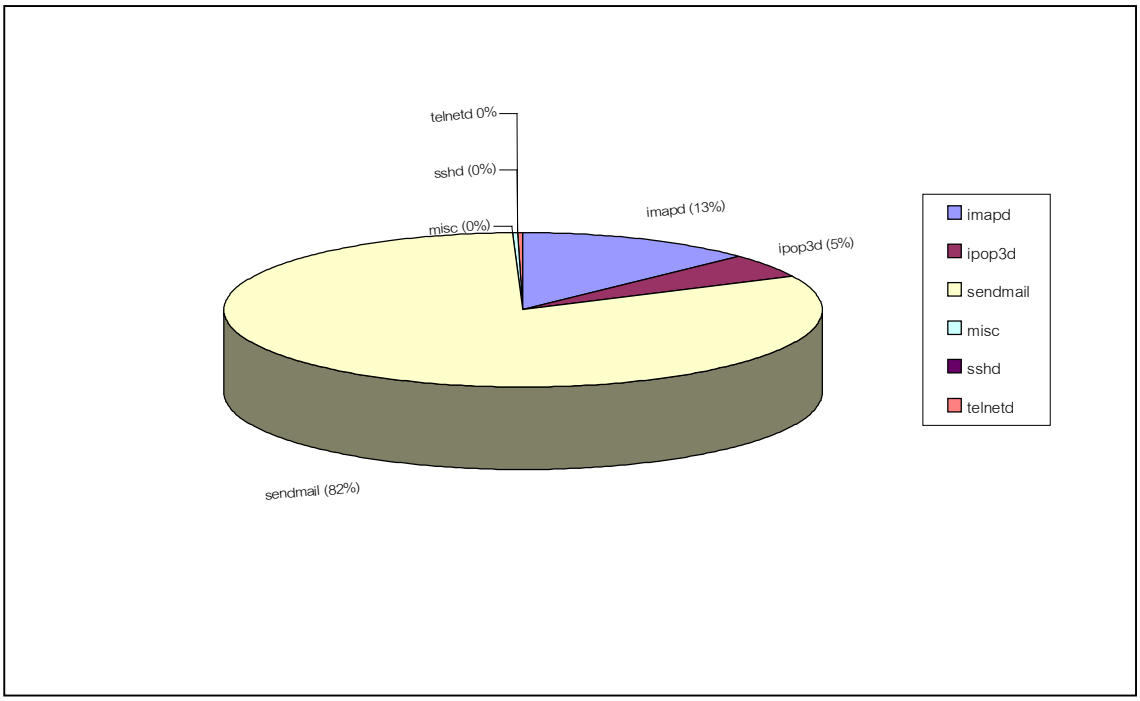
Level	Emergency	alert	Critical	Error	Warning	notice	Information	Debug
1			3,209	3,694	15,606	178,625	49,907,095	52,644,995
2			1,207	9,476	16,463	81,035	5,558,226	6,410,944
3			461	1,444	5,514	61,223	27,060,872	27,802,762
4			3,428	3,666	8,876	92,112	9,957,877	11,494,974
5			3,017	3,215	7,245	82,155	26,822,111	27,425,625
6			6,375	18,741	25,987	211,424	21,946,793	22,842,103
7			10,367	10,734	35,813	261,909	66,652,611	69,772,119
8		253	10,433	11,157	23,158	143,152	41,112,450	43,614,664
9			9,275	10,218	19,855	139,855	41,032,656	43,164,191
10			3,039	3,651	7,021	87,625	29,021,553	30,328,596
11			2,087	2,598	6,522	69,781	21,982,244	23,037,772
12			10,791	12,079	24,874	179,223	46,825,844	48,646,151
13			7,850	10,393	24,223	139,115	34,433,794	36,655,973
14			2,983	3,347	9,549	87,396	20,791,431	21,885,743
15			4,635	5,612	11,279	72,652	13,101,768	14,419,720
16			1,348	1,717	6,811	77,808	23,229,565	24,133,135
17			827	2,048	8,476	71,620	46,083,452	46,829,719
18			2,492	2,492	10,957	101,758	33,934,826	36,174,740
19			3,999	4,488	11,319	111,492	24,464,498	26,352,289
20		204	11,783	12,387	29,147	157,163	46,026,761	47,901,320
21			3,443	9,200	24,412	138,442	54,833,981	56,876,416
22			1,408	2,006	8,371	164,075	29,345,403	30,586,888
23			5,513	6,602	12,634	70,652	10,696,802	11,402,274
24			762	1,007	3,565	185,424	14,519,442	15,224,658
25			7,909	9,116	23,380	161,006	50,126,853	52,911,573
26			6,566	6,809	14,095	135,829	49,414,787	51,011,371
27		119	2,577	2,577	7,233	43,007	16,202,033	16,842,084
28			5,021	6,581	13,722	132,994	43,526,891	44,220,371
29			2,897	3,102	8,223	65,214	14,766,262	15,660,648
30			4,572	5,782	11,238	142,661	31,988,488	33,113,126
31			4,326	5,641	10,344	132,188	43,225,181	44,279,275
รวม	0	576	144,600	191,580	445,912	3778,615	988,592,550	1,033,666,219
เฉลี่ยต่อวัน	0	192	4,665	6,180	14,384	121,891	31,890,082	33,344,072

* **หน่วยเป็น** byte



ภาพประกอบ 6.5 พื้นที่ในเก็บล็อกของแต่ละวันที่โดยจำแนกตามระดับความสำคัญเหตุการณ์

จากตาราง 6.7 และภาพประกอบ 6.5 จะเห็นว่าพื้นที่ที่ใช้ในการเก็บล็อกตั้งแต่ระดับ emergency, alert, critical, error, warning และ notice มีแนวโน้มเพิ่มขึ้นอย่างต่อเนื่องและจะเพิ่มขึ้นเป็นอย่างมากสำหรับระดับ info โดยในระดับ info และ debug จะใช้พื้นที่ในการเก็บไม่แตกต่างกันมากนัก ซึ่งค่าเฉลี่ยของการใช้พื้นที่ในแต่ละวันประมาณ 33 MB และเมื่อพิจารณาถึงสัดส่วนการใช้พื้นที่ของแต่ละโปรแกรมในระดับ debug ซึ่งเป็นระดับที่เก็บข้อมูลล็อกทั้งหมด สามารถแสดงได้ดังภาพประกอบ 6.6



ภาพประกอบ 6.6 สัดส่วนการใช้พื้นที่ในการเก็บล็อกของแต่ละโปรแกรม

จากภาพประกอบ 6.6 จะเห็นว่าสัดส่วนการใช้พื้นที่ของแหล่งกำเนิด mail ซึ่งประกอบด้วยโปรแกรม sendmail imapd และ ipopd รวมกันแล้วเกือบ 100 เปอร์เซ็นต์ ซึ่งแสดงให้เห็นเครื่องดังกล่าวมีกิจกรรมเกือบทั้งหมดที่เกี่ยวกับการให้บริการอีเมล์

6.6.2 การใช้พื้นที่เก็บข้อมูลล็อกในปัจจุบัน

เมื่อพิจารณาการเก็บข้อมูลล็อกบนเครื่อง ratree.psu.ac.th โดยพิจารณาจากค่าในไฟล์กำหนดค่าการทำงาน ของ syslog ซึ่งอยู่ที่ /etc/syslog.conf มีการกำหนดค่าการทำงานดังนี้

```
#ident  "@(#)syslog.conf        1.4     96/10/11 SMI"  /* SunOS 5.0 */
# Copyright (c) 1991-1993, by Sun Microsystems, Inc.
#
# syslog configuration file.
#
# This file is processed by m4 so be careful to quote (`') names
# that match m4 reserved words.  Also, within ifdef's, arguments
# containing commas must be quoted.
#
```

```
*.err;kern.notice;auth.notice                   /dev/console
*.err;kern.debug;daemon.notice;mail.crit        /var/adm/messages
*.info                                          /var/adm/messages

*.alert;kern.err;daemon.err                     operator
*.alert                                         root

*.emerg                                         *
# if a non-loghost machine chooses to have authentication messages
# sent to the loghost machine, un-comment out the following line:
auth.notice             ifdef(`LOGHOST', /var/log/authlog, @loghost)
mail.debug              ifdef(`LOGHOST', /var/log/syslog, @loghost)
local2.debug            /var/log/sudo
local2.err              /var/log/poppassd-log
#
# non-loghost machines will use the following lines to cause "user"
# log messages to be logged locally.
ifdef(`LOGHOST', ,
user.err                                        /dev/console
user.err                                        /var/adm/messages
user.alert                                      `root, operator'
user.emerg                                      *
)
```

จากข้อมูลในไฟล์ดังกล่าวพบว่ามีการบันทึกข้อมูลล็อกลงในล็อกไฟล์ 4 ไฟล์คือ

1. **ล็อกไฟล์** /var/adm/messages **เก็บข้อมูลดังนี้**

```
*.err;kern.debug;daemon.notice;mail.crit        /var/adm/messages
*.info                                          /var/adm/messages
user.err                                        /var/adm/messages
```

2. **ล็อกไฟล์** /var/log/syslog **เก็บข้อมูลดังนี้**

```
mail.debug                                      ifdef(`LOGHOST', /var/log/syslog, loghost)
```

3. ล็อกไฟล์ /var/log/sudo เก็บข้อมูลดังนี้

   local2.debug */var/log/sudo*

4. ล็อกไฟล์ /var/log/poppassd-log เก็บข้อมูลดังนี้

   local2.err */var/log/poppassd-log*

จากการกำหนดค่าการทำงานของ syslog ดังกล่าวและพิจารณาจากสภาพการใช้งานของเครื่องคอมพิวเตอร์ดังกล่าวสามารถสรุปได้ดังนี้

1. ข้อมูลล็อกในระดับข้อมูลทั่วไป (info) จากทุกแหล่ง (facility) จะบันทึกลงในไฟล์ /etc/adm/messages
2. ข้อมูลล็อกในระดับที่ใช้ตรวจสอบการทำงานของโปรแกรม(debug) จาก facility ชื่อ mail จะบันทึกลงในไฟล์ /etc/log/syslog
3. ล็อกไฟล์ /var/log/sudo และ /var/log/poppassd-log ไม่มีการบันทึกข้อมูลล็อกเพิ่มเติมตั้งแต่วันที่ 2 เมษายน 2544 และ 16 พฤษภาคม 2544 ตามลำดับ

เมื่อพิจารณาพื้นที่รวม โดยนำพื้นที่ของ /etc/log/syslog รวมกับพื้นที่ของ /etc/adm/messages ในแต่ละวันของการกำหนดค่า syslog แสดงได้ดังตาราง 6.8

ตาราง 6.8 การใช้พื้นที่ในการเก็บล็อกในแต่ละวันของเครื่อง ratree

วันที่	รวมพื้นที่ที่ใช้ในแต่ละวัน (byte)
01	102,426,605
02	11,872,390
03	54,779,557
04	21,374,292
05	53,077,997
06	44,752,176
07	136,321,189
08	84,659,080
09	83,941,916
10	57,889,933
11	43,277,028
12	95,295,561
13	71,002,730
14	42,631,202
15	27,478,939
16	47,301,927
17	92,858,368

วันที่	รวมพื้นที่ที่ใช้ในแต่ละวัน (byte)
18	70,049,467
19	50,753,607
20	93,853,082
21	111,638,450
22	59,876,180
23	22,041,497
24	29,687,504
25	102,952,883
26	100,361,646
27	33,002,524
28	87,334,127
29	29,349,645
30	64,735,306
31	85,410,843
รวม	2,011,987,651
เฉลี่ยต่อวัน	64,902,827

จากตาราง 6.8 จะเห็นว่าพื้นที่ที่จะต้องใช้ในการเก็บล็อกแต่ละวันโดยเฉลี่ยคือ 64 MB และเมื่อเปรียบเทียบการใช้พื้นที่กับกรณีที่กำหนดให้ syslog เก็บล็อกทั้งหมดตามตาราง 6.4 ที่พื้นที่เฉลี่ยต่อวันอยู่ที่ 33 MB นั่นคือพื้นที่ที่ใช้เก็บล็อกในปัจจุบันจะมากกว่าการเก็บล็อกแบบทั้งหมดเกือบ 2 เท่า ทั้งนี้เนื่องจากการกำหนดค่าของ syslog.conf ในบรรทัด

**.info /var/adm/messages*

ซึ่งเป็นการระบุว่าให้เก็บล็อกของทุกแหล่งกำเนิดในระดับ info ไปบันทึกลงไฟล์ /var/adm/messages ซึ่งรวมถึงแหล่งกำเนิดจาก mail ด้วย ทำให้มีการเก็บล็อกที่ซ้ำซ้อนกับการกำหนดในบรรทัดถัดมาคือ

mail.debug ifdef(`LOGHOST', /var/log/syslog, @loghost)

ซึ่งจะระบุว่าให้เก็บล็อกของแหล่งกำเนิด mail ทั้งหมด ให้บันทึกลงไฟล์ /var/log/syslog นั่นคือข้อมูลล็อกของ mail ในระดับ info จะบันทึกลงในไฟล์ /var/adm/messages และขณะเดียวกันข้อมูลล็อกของ mail ในระดับ debug ก็จะบันทึกลงในไฟล์ /var/log/syslog ด้วย ดังนั้นจึงทำให้พื้นที่รวมที่ใช้เก็บล็อกเพิ่มขึ้นเกือบ 2 เท่าเมื่อเปรียบเทียบกับการเก็บข้อมูลล็อกทั้งหมด
ดังนั้นจึงสรุปได้ว่าในการกำหนดค่าในไฟล์ syslog.conf ที่ใช้อยู่ปัจจุบันบนเครื่อง ratree ทำให้มีการเก็บข้อมูลล็อกที่มาจาก mail ซ้ำซ้อนกัน ซึ่งทำให้เสียพื้นที่ในการเก็บเพิ่มเป็น 2 เท่า ดังนั้นในการกำหนดค่าดังกล่าวควรแก้ไขของเก็บข้อมูลล็อกลงไฟล์ /var/adm/messages จากเดิม

**.info /var/adm/messages*

เปลี่ยนเป็น

**.info;mail.none /var/adm/messages*

ซึ่งตารางแสดงปริมาณข้อมูลล็อกก่อนและหลังการเปลี่ยนแปลงการกำหนดค่าในไฟล์ syslog.conf แสดงได้ดังตาราง 6.9

ตารางที่ 6.9 แสดงปริมาณการใช้พื้นที่ก่อนและหลังเปลี่ยนแปลงค่าในไฟล์ syslog.conf

วันที่	ค่าเดิม	ค่าใหม่
1	102,426,605	49,907,095
2	11,872,390	5,558,226
3	54,779,557	27,060,872
4	21,374,292	9,957,877
5	53,077,997	26,822,111
6	44,752,176	21,946,793
7	136,321,189	66,652,611
8	84,659,080	41,112,450
9	83,941,916	41,032,656
10	57,889,933	29,021,553
11	43,277,028	21,982,244
12	95,295,561	46,825,844
13	71,002,730	34,433,794
14	42,631,202	20,791,431
15	27,478,939	13,101,768
16	47,301,927	23,229,565
17	92,858,368	46,083,452
18	70,049,467	33,934,826
19	50,753,607	24,464,498
20	93,853,082	46,026,761
21	111,638,450	54,833,981
22	59,876,180	29,345,403
23	22,041,497	10,696,802
24	29,687,504	14,519,442
25	102,952,883	50,126,853
26	100,361,646	49,414,787
27	33,002,524	16,202,033
28	87,334,127	43,526,891
29	29,349,645	14,766,262
30	64,735,306	31,988,488
31	85,410,843	43,225,181
รวม	2,011,987,651	988,592,550
เฉลี่ยต่อวัน	64,902,827	31,890,082

* หน่วยเป็น byte

สำหรับการพิจารณาว่าควรจะเก็บข้อมูลล็อกในระดับที่จะเหมาะสม โดยพิจารณาถึงประเด็นการนำข้อมูลล็อกนั้นมาใช้งานสำหรับการตรวจจับการบุกรุกได้ ซึ่งผลจากการทดสอบการบุกรุก เพื่อใช้ในการสร้างกฎสำหรับการตรวจจับการบุกรุกในหัวข้อที่ 5.8.1 สามารถแสดง facility และ priorityของข้อมูลล็อกที่เกิดร่องรอยที่สามารถนำใช้ในการตรวจจับบุกรุกได้ ดังตารางที่ 6.10

ตารางที่ 6.10 แสดงวิธีการโจมตีและระดับ facility และ priority ที่เกิดร่องรอยการบุกรุก

วิธีการโจมตี	facility	Priority
ไวรัส	mail	Information
เวิร์ม/โทรจัน	authorization	Alert
Ping Flood	authorization	Alert
Syn Flood	kernel	Information
Brute-force attack	authorization	Information
Sniffer	kernel	Information
Surveillance/Probe	mail	Information
Local exploit for sendmail 8.11.6	-	-
Linux kernel ptrace/kmod local root exploit	-	-

จากข้อมูลในตารางที่ 6.10 สามารถสรุปได้ว่าควรเก็บข้อมูลล็อกที่มีที่มาจากแหล่ง mail และ kernel ในระดับ information ส่วนข้อมูลล็อกจาก authorization นั้นหากต้องการให้สามารถตรวจจับการโจมตีด้วยวิธีการ Brute-force attack ก็ควรเก็บข้อมูลล็อกในระดับ info เช่นกัน หรืออาจจะเก็บข้อมูลในระดับ alert ก็ได้เช่นกัน ดังนั้นจึงพอจะสรุปโดยภาพรวมได้ว่าระดับของการเก็บข้อมูลล็อกที่เหมาะสมสำหรับการนำข้อมูลไปใช้สำหรับการวิเคราะห์เพื่อตรวจจับการบุกรุกควรเป็นระดับ information

6.7 สรุป

ผลการทดสอบระบบที่พัฒนาขึ้นพบว่าระบบที่พัฒนาสามารถตรวจจับการโจมตีในลักษณะ Active attack ได้ดีกว่า Passive attack ทั้งนี้เนื่องจากการโจมตีในลักษณะดังกล่าวส่วนใหญ่ทำให้เกิดล็อกที่บันทึกการเกิดเหตุการณ์ ซึ่งสามารถนำเอาข้อมูลล็อกเหล่านั้นมาเขียนเป็นกฎเพื่อใช้ในการตรวจจับในภายหลังได้ แต่ในบางเหตุการณ์เช่นการบุกรุกโดยอาศัยช่องโหว่ของโปรแกรม ซึ่งพบว่าไม่มีร่องรอยใดๆ เกิดขึ้น จึงทำให้ระบบไม่สามารถตรวจจับการบุกรุกได้ โดยการตรวจจับอาจจะใช้วิธีการอื่นในการตรวจจับ เช่น State Transition Analysis Technique เป็นต้น ส่วนการศึกษาหาจุดสมดุลในการบันทึกข้อมูลล็อก สรุปได้ว่าในการบันทึกข้อมูลล็อกของเหตุการณ์ต่างๆ ที่เกิดขึ้นบนเครื่องคอมพิวเตอร์ที่ใช้ระบบปฏิบัติการยูนิกซ์ ควรกำหนดในระดับ information ทั้งนี้โดยจากการพิจารณาถึงการนำข้อมูลล็อกไปใช้ในการตรวจจับบุกรุก นอกจากนี้ข้อมูลล็อกที่ได้จากระดับดังกล่าวจะมีรายละเอียดของเหตุการณ์ต่างๆ รวมถึงข้อผิดพลาดที่เกิดขึ้นในระบบที่ชัดเจนกว่า และในการวิเคราะห์ข้อมูลหากไม่ต้องการข้อมูลส่วนใดก็สามารถแยกข้อมูลเหล่านั้นออกไปก่อนได้ แต่หากถ้าผู้ดูแลระบบกำหนดให้บันทึกเฉพาะข้อมูลที่สำคัญหรือที่สนใจเท่านั้น แล้วในภายหลังเกิดมีความต้องการข้อมูลอื่นๆ นอกเหนือจากที่กำหนดไว้ก็จะไม่สามารถทำได้ ในบทถัดไปจะกล่าวถึงบทสรุปและข้อเสนอแนะ