บทที่ 6
หลักการทำงานย่อยของโปรแกรม

6.1 หลักการสร้างแพ็คเก็ต (Packet)

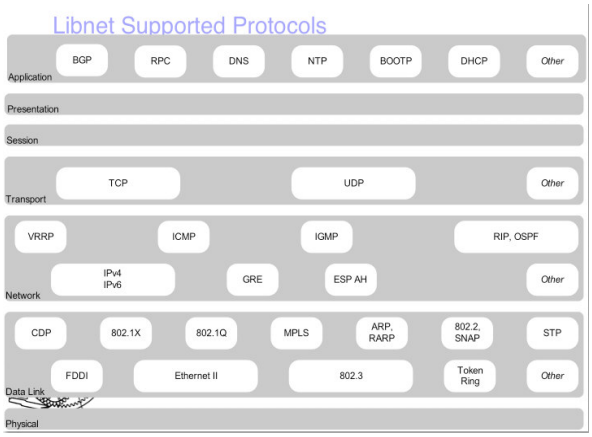
6.1.1 ความหมายและหลักการทำงานของลิปเน็ต (Libnet)

ลิปเน็ตเป็นไลบรารี่ (Library) ของภาษาซีที่เตรียมเอาไว้เพื่อใช้ในการเชื่อมต่อระดับสูง (high-level) เพื่อส่งผ่านแพ็กเก็ตจากตำแหน่งต้นทางปยังอีกฝั่งปลายทาง ซึ่งลิปเน็ตมีตัวอย่างเอกสารอธิบายด้วยไว้อย่างชัดเจนเกี่ยวกับเอพีไอ (API) ที่ใช้ได้อย่างง่ายดายและเรียกใช้ในโปรแกมได้อย่างเหมาะสมซึ่งลิปเน็ตถูกสร้างขึ้นนี้ประกอบด้วยเหตุผล 2 ประการ

- เพื่อสร้างการเชื่อมต่ออย่างง่ายสำหรับโปรแกรมเมอร์ที่ไม่สนใจความยุ่งยากของการทำเน็ตเวิร์กโปรแกรมมิ่ง(Network programming) ระดับต่ำ(low-level) และให้ความสนใจกับการเขียนโปรแกรมของพวกเขาและ การแก้ไขปัญหาต่างมากกว่า
- เพื่อลดความไม่พอใจของเน็ตเวิร์กโปรแกรมเมอร์หลายๆคน ที่ขาดประสบการณ์ที่เป็นมาตรฐาน

ลิปเน็ตเป็นเครื่องมือที่มีประโยชน์ในการเขียนแอฟพลิเคชั่น(Applications) ที่เกี่ยวข้องกับการรักษาความปลอดภัย, เครื่องมือและโมดูล(modules) ต่างๆ ซึ่งการสร้างโปรแกรมที่ทำงานไม่ประสงค์ดีหลายอันก็พัฒนาโปรแกรมจากการใช้ libnet และรวมถึงเครื่องมือที่เกี่ยวกับการรักษาความปลอดภัยหลายโปรแกรมโดยตัวโปรแกรมของลิปเน็ตมีประโยชน์พอสมควรคือสามารถสร้างและส่งแพ็กเก็ตลงไปบน เน็ตเวิร์กได้แต่อย่างไรก็ตามตัวลิปเน็ตก็ไม่สามารถทำในส่วนของการดักจับ(Capture packet) ได้ซึ่งสามารถหาไลบรารี่ลิปพีแคป(Libpcap) มาช่วยทำในส่วนของการดักจับแพ็กเก็ตได้ เนื้อหาในบทนี้ครอบคลุมถึง หัวข้อการจัดการกับไลบรารี่ให้เกิดประโยชน์

ปัจจุบันตัวไลบรารี่สิปเน็ตเป็นไลบรารี่ของภาษาซีที่ออกแบบมาให้เล็กและง่ายต่อการใช้ โดยจุดประสงค์หลักของตัวลิปเน็ตเองคือการสร้างแพ็คเก็ตปัจจุบันมีโปรดตคอลที่สนับสนุน ดังรูป



รูปที่ 6-1 แสดงโปรโตคอลที่ลิปเน็ตสนับสนุน

6.1.2 ขั้นตอนการดำเนินการ

เพื่อสร้างและส่งแพ็กเก็ตออกไป จะต้องมีคำสั่งพื้นฐานดังนี้

6.1.2.1 สร้างคอนเท็กซ์ (Context) ของตัวแพ็คเก็ต

เป็นการเลือกเน็ตเวิร์กการ์ด (Network card) และระดับการควบคุมแพ็คเก็ตตามที่ต้องการว่าต้องการควบคุมตั้งแต่ลิงค์เลเยอร์ (Link layer) หรือไอพีเลเยอร์ (IP Layer) ขึ้นไปโดยฟังก์ชันที่ใช้ คือ

```
libnet_t* libnet_init( int          // injection_type
                       char *       // device
                       char *       // err_buf   )
```

Injction_type เป็นตัวแปลที่บอกว่าต้องการควบแพ็คเก็ตในระดับไหน

- LIBNET_RAW4, LIBNET_RAW6 เป็นการสร้างแพ็คเก็ต ที่จะควบคุมตั้งแต่ไอพีเลเยอร์ขึ้นไป

- LIBNET_LINK, LIBNET_LINK6 เป็นการสร้างแพ็คเก็ต ที่จะควบคุมตั้งแต่ลิงค์เลเยอร์ขึ้นไป

ดีไวซ์(Device) เป็นการบอกว่าจะใช้เน็ตเวิร์กการ์ดอันไหนในการส่งแพ็คเก็ต โดยปกติค่าจะเป็น eth0 หรือไอพีแอดเดรส (IP address) ของเครื่องนั้น

Err_buf เป็นการแจ้งความผิดปกติ (Error) ถ้าเกิดการผิดพลาดระหว่างการสร้างจะต้องมีการเขียนข้อความแจ้งความผิดปกติ (Error Message) จะเขียนไว้ที่นี่

และตัวฟังก์ชัน Libnet_init ถ้าเกิดข้อผิดพลาดก็จะเขียนข้อความแจ้งความผิดปกติไว้ที่ err_buf และคืนค่านูล (NULL) ถ้าไม่มีการผิดพลาดจะคืนค่าพอย์เตอร์ (pointer) ที่ชี้ไปยังคอนเท็กซ์

6.1.2.2 สร้างแต่ละโพรโตคอลบล็อก (Protocol block)

ในการสร้างแต่ละแพ็คเก็ตต้องมีการสร้างโปรโตคอลของแต่ละชั้นในการสร้างแพ็คเก็ตเรียกว่าโพรโตคอลบล็อกเช่น ถ้าต้องการสร้างแพ็คเก็ตดีเอ็นเอสคิวรี่ (DNS query) ก็ต้องสร้างโพรโตคอลบล็อกของชั้นแอฟพลิเคชั่นคือ ดีเอนเอสและต้องการส่งแบบยูดีพี (UDP) ต้องสร้างโพรโตคอลบล็อกของยูดีพีเป็นต้น

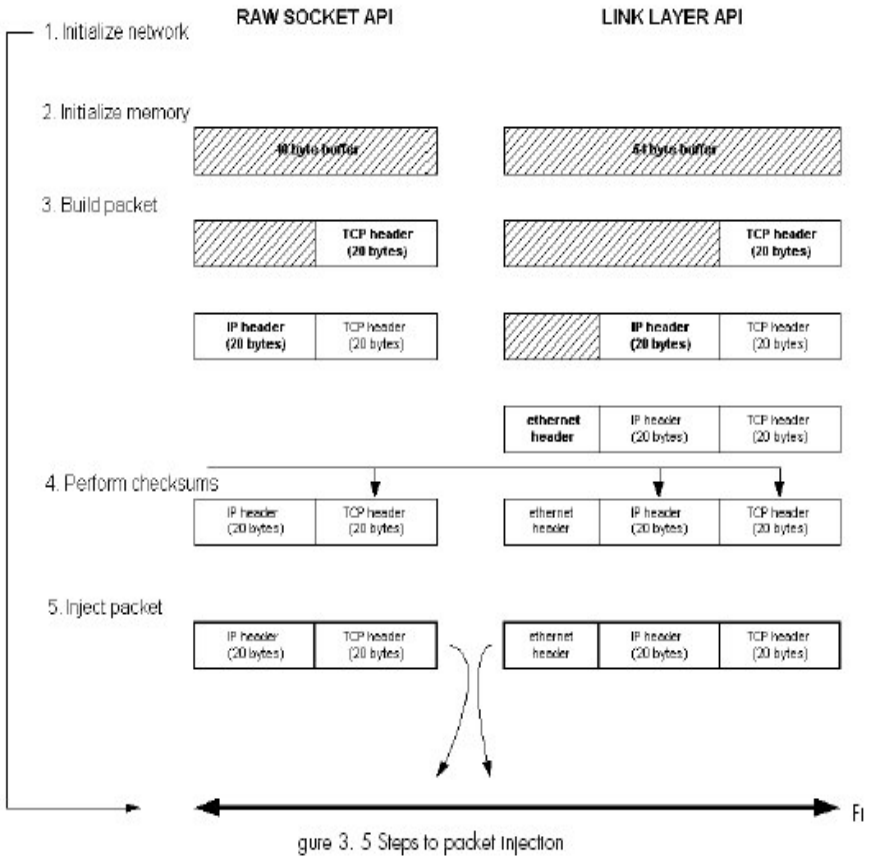
6.1.2.3 เป็นขั้นตอนในการส่งแพ็คเก็ต

ถ้าในการส่งต้องมีการทำเช็คซัม(Checksum) ซึ่งค่าเช็คซัมจะถูกคำนวณโดยอัตโนมัติเมื่อสร้างโพรโตคอลบล็อกสร็จ ในการส่งแพ็คเก็ตใช้ฟังก์ชัน

```
int libnet_write( libnet_t* )
```

6.1.2.4 เป็นขั้นตอนในการลบข้อมูลทั้งหมดที่สร้างไว้ออกจากหน่วยความจำ โดยการเรียกฟังก์ชัน

```
void libnet_destroy( libnet_t* )
```



รูปที่ 6-2 แสดงการขั้นตอนการสร้างและส่ง Packet ลงบนเครือข่าย

6.2 การทำซ้ำทราฟฟิกบนเครือข่าย

6.2.1 ความหมายของการทำซ้ำแพ็กเก็ตลงบนเครือข่าย (Replay packet)

หมายถึง การนำไฟล์ที่ดักจับแพ็คเก็ตมาได้จะเป็นไฟล์นามสกุล *.pcap จากระบบเน็ตเวิร์ก โดยวิธีการหรือโปรแกรมต่างๆ และนำมาแก้ไข หรือส่งแพ็คเก็ตออกไปตามไฟล์ที่ดักจับมาได้ ก่อนที่จะส่งออกไปลงบนเครือข่าย

6.2.2 องค์ประกอบของการทำซ้ำทราฟฟิกบนเครือข่าย

ในส่วนของการทำซ้ำทราฟฟิกบนเครือข่ายต้องมีไลบรารี่ที่เกี่ยวข้องคือลิปเน็ตและลิปพีแค็ปเพื่อทำให้การจัดการทำงานของการทำซ้ำทราฟฟิกบนเครือข่าย

6.2.3 การทำงานของการทำซ้ำทราฟฟิกบนเครือข่าย

การทำงานของการทำซ้ำทราฟฟิกบนเครือข่าย อาศัยไลบรารี่ลิปพีแค็ปในการอ่านข้อมูลจากจากไฟล์ที่ดักจับมาได้ และใช้ไลบรารี่ลิปเน็ตในการสร้างแพ็คเก็ตออกไปบนเครือข่าย

ในส่วนของการใช้**ไลบรารี่ลิปพีแค็ป**ฟังก์ชันหลักที่ใช้ คือ

1) ฟังก์ชันในการเปิดอ่านไฟล์ที่ดักจับมาได้

pcapnav_t* pcapnav_open_offline (const char *filename);

ฟังก์ชันนี้เป็นการเปิดไฟล์ด้วยชื่อของไฟล์ และจัดการเกี่ยวกับไฟล์ ส่งค่ากลับมาเป็นการจัดการเกี่ยวกับข้อมูล(Handle for trace) ถ้าสำเร็จ หรือนูลถ้าเกิดข้อผิดพลาด เมื่อคุณทำงานเกี่ยวกับไฟล์ที่เปิดออกมานี้เสร็จแล้วก็ควรปิดการใช้งานด้วยฟังก์ชัน pcapnav_close()

2) ฟังก์ชันในการอ่านแพ็คเก็ตในไฟล์ทีละแพ็คเก็ต

int pcap_next_ex(pcap_t *p, struct pcap_pkthdr **pkt_header,const u_char **pkt_data)

ฟังก์ชันนี้เป็นการอ่านแพ็คเก็ตต่อไป ค่าที่จะส่งกลับมาเป็น u_char pointer ของข้อมูลในแพ็คเก็ตนั้น ถ้าค่าที่ส่งกลับมาเป็น null แสดงว่าเกิดข้อผิดพลาดในการอ่านข้อมูลแพ็คเก็ตนั้นๆ

ในส่วนของการใช้**ไลบรารี่ลิปเน็ต**ฟังก์ชันหลักที่ใช้ คือ

1) ฟังก์ชันในการสร้าง

libnet_init(LIBNET_LINK_ADV, intf, ebuf)

ฟังก์ชันในการจองหน่วยความจำและเลือกชนิดในการสร้างแพ็คเก็ตว่าจะต้องการในระดับ เน็ตเวิร์กหรือระดับดาต้าลิงค์ที่เน็ตเวิร์กอันไหน ถ้าเกิดข้อผิดพลาดก็จะส่งข้อความเกี่ยวกับข้อผิดพลาดมาที่ ebuf

2) ฟังก์ชันในการส่งแพ็คเก็ตออกไปบนเครือข่าย

libnet_adv_write_link(l, pktdata, pkthdr.caplen)

ฟังก์ชันนการสร้างแพ็คเก็ตลงไปบนเครือข่ายโดยพารามิเตอร์(Parameter) ที่ส่งไปเป็นข้อมูลของแพ็คเก็ตและความยาวของแพ็คเก็ตที่จะสร้างออกไป ถ้าสามารถสร้างแพ็คเก็ตออกไปได้ฟังก์ชันนี้จะส่งค่ากลับมาเป็น 1 แต่ถ้าเกิดข้อผิดพลาดฟังก์ชันนี้จะส่งค่ากลับมาเป็น -1

6.2.4 ประโยชน์ของการทำซ้ำทราฟฟิกบนเครือข่าย

6.2.4.1 ในการตรวจสอบเครือข่าย (Network Debugging Tools) ในการตรวจสอบแอพพิเคชั่น หรือไฟล์วอล (firewall) และการทำการตรวจสอบเหมือนเดิมหลายๆครั้งเพื่อทำการเปรียบเทียบผลที่ได้ออกมา เพื่อวัดประสิทธภาพของเครือข่ายหรือวิเคราะห์ปัญหาที่เกิดขึ้น สามารถทำให้แก้ก่อนการเปิดให้บริการได้

6.2.4.2 เพื่อใช้ในการศึกษาการทำงานของโปรโตคอลในระดับชั้นต่างๆ ในระบบเครือข่าย ศึกษาลักษณะการโจมตีแบบต่างๆ ในระบบเครือข่าย

6.3 หลักการดักจับแพ็กเก็ต

6.3.1 ความหมายของแพ็คเก็ตแคปเจอร์ (Packet sniffer)

คำว่า สนิฟเฟอร์ (Sniffer) นั้นเครื่องหมายการค้าซึ่งจดทะเบียนไว้โดยบริษัท Network Associates Inc. ในสหรัฐฯเพื่อใช้ในผลิตภัณฑ์ของตนเองชื่อ Sniffer Network Analyzer ซึ่งเป็นโปรแกรมวิเคราะห์เน็ตเวิร์กโดยอาศัยการดักจับข้อมูลภายในเน็ตเวิร์กทั้งหมดมาวิเคราะห์การใช้งานระบบเน็ตเวิร์ก ต่อมาคำนี้ถูกนำมาใช้เรียกอุปกรณ์ประเภทนี้ที่นำมาในการดักอ่านข้อมูลกันอย่างแพร่หลาย จนเป็นที่เข้าใจว่าสนิฟเฟอร์เป็นเครื่องมือในการดักจับข้อมูลบนเน็ตเวิร์ก ซึ่งถ้าหากเรียกให้ถูกต้องอุปกรณ์ประเภทนี้ควรเรียกว่า Net wire Tapping Device

แพ็กเก็ตแคปเจอร์ คืออุปกรณ์ที่ทำหน้าที่ดักจับแพ็กเก็ตที่วิ่งอยู่บนเน็ตเวิร์ก โดยจะทำงานร่วมกับเน็ตเวิร์กอินเทอร์เฟซการ์ด (Network Interface card) ในโหมดโพรมิสคิวอัส (Promiscuous-mode)

โดยการทำงานร่วมกับโหมดนี้ การทำงานของเน็ตเวิร์กอินเทอร์เฟซการ์ดในโหมดนี้จะไม่มีการเปรียบเทียบแอดเดรสปลายทางของแพ็กเก็ตกับแอดเดรสของเน็ตเวิร์กอินเทอร์เฟซการ์ด นั่นคือทุกๆแพ็กเก็ตมาถึงยังเน็ตเวิร์กอินเทอร์เฟซการ์ดจะถูกจับแพ็กเก็ตขึ้นมาได้ทั้งหมดไม่เลือกว่าจะระบุแอดเดรสปลายทางของการ์ดเน็ตเวิร์กอินเทอร์เฟซอันไหน

แพ็กเก็ตแคปเจอร์จะถูกใช้ในการเฝ้าดูการจารจรในเครือข่ายให้มีประสิทธิภาพและแก้ไขปัญหาต่างๆที่เกิดจากการทำงานผิดพลาดของเครือข่ายและยังเป็นเครื่องมือในการศึกษาการทำงานของโพรโตคอลรวมถึงการดูแลความปลอดภัยของระบบ

6.3.2 องค์ประกอบของแพ็กเก็ตแคปเจอร์

6.3.2.1 ฮาร์ดแวร์ (Hardware) หมายถึงอุปกรณ์อิเล็กทรอนิคส์ที่สามารถดักอ่านสัญญาณจากเน็ตเวิร์กเข้ามาได้ และสามารถนำสัญญาณที่ให้ส่งต่อไปประมวลผลออกมาเป็นข้อมูลทางคอมพิวเตอร์ได้ หน้าที่หลักคือจัดการกับการรับข้อมูลในระดับฟิสิกคอล (Physical) อุปกรณ์นี้โดยทั่วไปคือเน็ตเวิร์กอะแดปเตอร์

6.3.2.2 ไดรเวอร์(Driver) เป็นโปรแกรมระดับล่างที่ควบคุมการดักจับข้อมูลของฮาร์ดแวร์ไปเก็บเป็นข้อมูลดิบรอการประมาวลผลในลำดับถัดไป

6.3.2.3 บัฟเฟอร์(Buffer) เป็นหน่วยความจำที่ใช้พักข้อมูลจากการดักจับมาได้ของ driver โดยจะทำงานการจัดเก็บเพียงชั่วคราว และหมุนเวียนข้อมูลใหม่เข้ามาเสมอ กลไกลการนำข้อมูลขากไดรเวอร์มาเก็บในบัฟเฟอร์ จะเป็นตัวบอกสมรรถนะของการดักข้อมูลได้ความเร็วสูงสุดเท่าใด หากกระบวนการนำข้อมูลไปเก็บเป็นไปอย่างล่าช้า ย่อมทำให้แพ็กเก็ตแคปเจอร์ไม่สามารถดักจับข้อมูลที่อยู่บนเน็ตเวิร์กได้ทันและต้องปล่อยข้อมูลนั้นทิ้งไป

6.3.2.4 ซอฟต์แวร์ (Software) เพื่อทำหน้าที่จัดการข้อมูลที่ได้เข้าโดยการประมวลผลตามวัตถุประสงค์ของการดักอ่านข้อมูลนั้น เนื่องจากข้อมูลดิบที่ดักได้นั้นจะเป็นข้อมูลในระดับต่ำคือ เน็ตเวิร์กอินเทอร์เฟซเลเยอร์ (Network Interface Layer) ซึ่งจะมีข้อมูลที่ยังไม่ผ่านการมัลติเพล็กซ์และจัดรูปแบบให้เข้าใจได้ ส่วนที่ได้จะเป็นข้อมูลเลขฐานสอง 0 กับ 1 จำนวนมหาศาลที่ต้องนำมาหาความหมายและอีกหน้าที่หนึ่งคือข้อมูลที่ดักได้นันเป็นการสื่อสารของทุกโฮสต์ที่อยู่ภายในเน็ตเวิร์กนั้นร่วมกันอยู่ อย่างไม่เป็นระเบียบและไม่มีการแยกแยะว่าเป็นการสื่อสารเรื่องอะไร ระหว่างโฮสต์ใดกับโฮสต์ใด การที่จะแปลความหมายของข้อมูลนี้ได้ก็จำเป็นต้องมีโปรแกรมสำหรับทำหน้าที่แปลความหมายของข้อมูลให้อยู่ในรูปแบบที่สามารถเข้าใจได้มากขึ้นนั่นคือการทำหน้าที่คล้ายกับดีมัลติเพล็กซ์ของโพรโตคอลปกติ แต่จะเป็นการดีมัลติเพล็กซ์ของข้อมูลทุกๆโฮสต์โดยไม่สนใจว่าเป็นข้อมูลของโฮสต์ใด

หลังจากข้อมูลผ่านการดีมัลติเพล็กซ์แล้วก็จะอยู่ในรูปที่สามารถเข้าใจได้ง่ายขึ้น แต่จะเข้าใจมากน้อยขึ้นอยู่กับความสามารถในการดีมัลติเพล็กซ์ของโปรแกรม หากสามารถดีมัลติพล็กซ์ได้จนถึงโพรโตคอลเลเยอร์ที่สูงรวมทั้งหากมีการแยกแยะหมวดหมู่ของการสื่อสารของแต่ละโฮสต์ก็จะเห็นความต่อเนื่องของการสื่อสารได้ดีขึ้น

ในส่วนของการจัดเก็บข้อมูลดิบที่ดักมาได้โปรแกรมแคปเจอร์ส่วนใหญ่จะทำการเก็บข้อมูลดิบที่ดักมาได้ไว้ในฐานข้อมูลเพื่อให้สามารถนำกลับมาวิเคราห์ในภายหลังได้ เพราะในการแคปเจอร์จะต้องคอยดักจับขัอมูลอยู่ตลอดเวลา หากทำการดีมัลติเพล็กซ์ทันทีที่ได้รับข้อมูลในแบบเรียลไทม์อาจทำให้ประ

สิทธิภาพของแคปเจอร์ลดต่ำลง และเนื่องจากการสื่อสารข้อมูลต้องอาศัยความต่อเนื่องของการเชื่อมโยงกันของหลายๆแพ็กเก็ต ประกอบการดีมัลติเพล็กซ์ส่วนใหญ่จะสามารถทำได้ก็ต่อเมื่อข้อมูลได้รับมาครบถ้วนสมบูรณ์ทั้งหมดแล้วเท่านั้น

6.3.3 การทำงานของแพ็กเก็ตแคปเจอร์

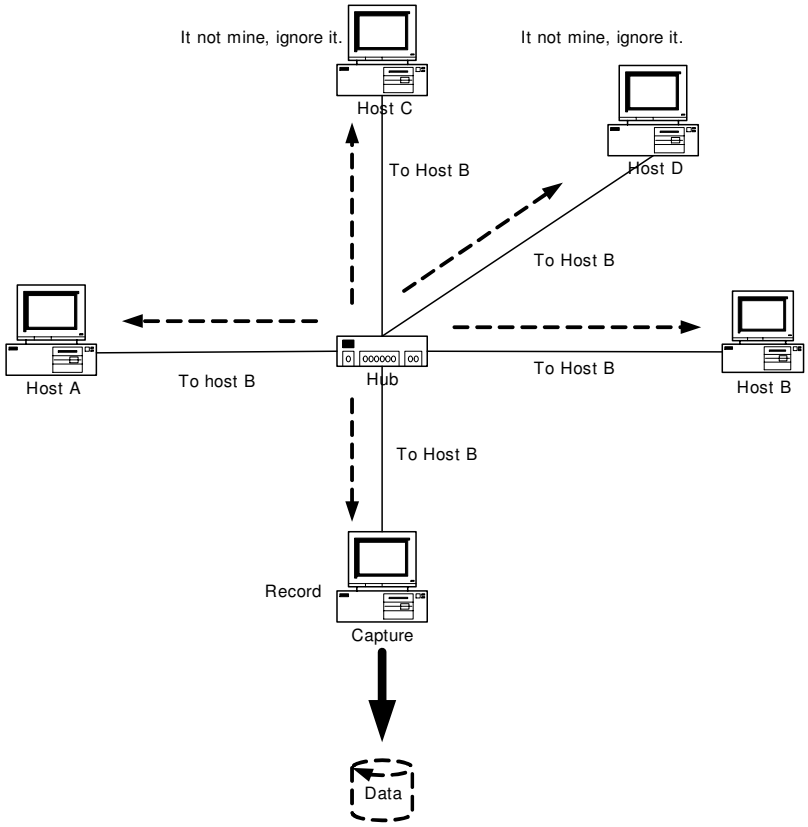
การทำงานของแพ็กเก็ตแคปเจอร์อาศัยหลักการของโพรโตคอลอีเทอร์เน็ตที่ใช้การกระจายของข้อมูลไปยังทุกโฮสต์ที่อยู่ในเน็ตเวิร์กและอาศัยโฮสต์แต่ละตัวทำหน้าที่จำแนกการสื่อสารของตนเอง นั่นหมายความว่าข้อมูลทุกแพ็กเก็ตที่ใช้สื่อสารกันได้ถูกส่งไปยังทุกโฮสต์ ซึ่งจะได้รับพร้อมกันแล้วเหมือนกัน เพียงแต่การที่จะสื่อสารกันได้อย่างถูกต้องนั้นโฮสต์แต่ละตัวจะต้องมีการบวนการที่สามารถรู้ได้ว่าข้อมูลแพ็กเก็ตใดเป็นของตนเอง และข้อมูลแพ็กเก็ตใดมิใช่ของตนเอง ทุกๆแพ็กเก็ตที่กระจายลงบนเน็ตเวิร์กนั้นจะมีหมายเลขระบุชัดเจตคือ แม็คแอดเดรส (MAC Address) หรือเรียกอีกอย่างหนึ่งว่าอีเทอร์เน็ต (Ethernet Address) ซึ่งจะเป็นสิ่งที่บอกว่าแพ็กเก็ตมาจากฮาร์ดแวร์ใดในเน็ตเวิร์ก ทำให้สามารถระบุได้ว่าส่งมาจากโฮสต์ใด

แม็คแอดเดรสจะเป็นหมายเลขเฉพาะฮาร์ดแวร์ทุกชนิดที่ใช้สื่อสารโดยโพรโตคอลอีเธอร์เน็ต และในทางทฤษฎีแล้วฮาร์ดแวร์ทุกชนิดจะไม่มีแม็คแอดเดรสซ้ำกัน โดยทั่วไปแม็คแอดเดรสจะกำหนดตายตัวอยู่ในรอม(Rom) ของฮาร์ดแวร์และไม่สามารถเปลี่ยนแปลงได้ด้วยซอฟต์แวร์

การใช้งานของฮาร์ดแวร์จะต้องควบคู่กับไดรเวอร์ของฮาร์ดแวร์นั้นๆโดยปกติแล้วไดรเวอร์จะถูกกำหนดให้ปฏิบัติตามโพรโตคอลอย่างเข้มงวดคือ

- ให้รับข้อมูลที่มีแม็คแอดเดรสเป็นของตนเองเท่านั้น (ห้ามอ่านข้อมูลของผู้อื่น)
- ให้ส่งข้อมูลโดยใช้แม็คแอดเดรสของตอนเองเท่านั้น (ห้ามปลอมเป็นผู้อื่น)

ดังนั้นหากใช้ไดรเวอร์ตามปกติที่มากับฮาร์ดแวร์แล้วเครื่องคอมพิวเตอร์ก็จะทำงานอยู่ในระเบียบเรียบร้อย และไม่สามารถวุ่นวายกับข้อมูลของผู้อื่นได้ แพ็กเก็ตแคปเจอร์จึงจำเป็นที่ต้องใช้โหมดการทำงานพิเศษที่อนุญาตให้รับข้อมูลที่แม็คแอดเดรสไม่ใช้ของตนได้โหมดนั้นเรียกว่า โพรมิสคิวอัสโหมด (Promiscuous Mode) ที่ทำให้ฮาร์ดแวร์สามารถรับข้อมูลดิบทั้งหมดบนเน็ตเวิร์กเข้ามาได้ โดยไม่สนใจว่าข้อมูลทั้งหมดเป็นของใคร ส่งให้ใคร



รูปที่ 6-3การทำงานของแพ็กเก็ตแคปเจอร์

จากรูปที่ 6-3 ในเน็ตเวิร์กมีโฮสต์อยู่ 4 ตัวต่อรวมกันเป็นเน็ตเวิร์กโดยใช้ฮับ (Hub) ร่วมกัน และมีโฮสต์ตัวหนึ่งรันโปรแกรมที่ทำหน้าที่เป็นโปรแกรมแคปเจอร์เพื่อดักจับข้อมูลเสียบต่อร่วมเข้ากับฮับนั้น จากรูปแสดงให้เห็นการกระจายของข้อมูลที่โฮสต์ A ต้องการส่งให้โฮสต์ B ข้อมูลได้กระจายไปยังทุกโฮสต์ที่ต่ออยู่บนฮับ รวมถึงแพ็กเก็ตแคปเจอร์ด้วย

โฮสต์ B เมื่อได้รับข้อมูลก็จะตรวจแม็คแอดเดรสถ้าพบว่าเป็นข้อมูลที่ถูกส่งมาหาตนเองก็ทำการอ่านและส่งต่อให้ระบบปฏิบัติการประมวลผล

โฮสต์ C และโฮสต์ D โฮสต์ทั้งสองซึ่งมีเน็ตเวิร์กอะแดปเตอร์ทำงานอยู่ในโหมดปกติ เมื่อได้ทำการตรวขสอบแล้วว่าไม่ใช่ข้อมูลของตนเอง ก็จะไม่สนใจข้อมูลนั้น

แพ็กเก็ตแคปเจอร์ สำหรับแพ็กเก็ตแคปเจอร์นั้นมีเน็ตเวิร์กแอแดปเตอร์ที่ทำงานอยู่ในโพรมิสคิวอัสโหมดนั่นคือไม่สนใจว่าข้อมูลที่เข้ามาเป็นของใคร แพ็กเก็ตแคปเจอร์จะรับข้อมูลทั้งหมดแล้วนำไปเก็บไว้ในบัฟเฟอร์และฐานข้อมูลเพื่อนำไปประมวลผลเพื่อหาข้อมูลที่มีประโยชน์ภายหลัง

จากเน็ตเวิร์กในภาพนั้นมีแพ็กเก็ตแคปเจอร์ถูกติดตั้งในตำแหน่งดังกล่าวก็จะทำให้ข้อมูลทั้งหมดที่โฮสต์ทั้ง 4 สื่อสารกันโดยผ่านฮับที่ใช้งานร่วมกัน ไม่ว่าข้อมูลใดจะปรากฏที่ฮับก็สามารถถูกดักอ่านได้โดยแพ็กเก็ตแคปเจอร์ทันที

6.3.4 ประโยชน์จากแพ็กเก็ตแคปเจอร์

6.3.4.1 วิเคราะห์ระบบเคือข่าย (Network Analyzer)

การที่แพ็กเก็ตแคปเจอร์สามารถดักจับข้อมูลทั้งหมดที่อยู่บนเน็ตเวิร์กได้ ทำให้เราสามารถนำข้อมูลดังกล่าวมาทำการประมวลผลวิเคราะห์คุณสมบัติต่างๆของเน็ตเวิร์กได้หลายแง่มุม ไม่ว่าการกระจายการใช้งานเน็ตเวิร์กของโพรโตคอลต่างๆเช่น เอชทีทีพี(HTTP), เอสเอ็มทีพี(SMTP), เอฟทีพี(FTP) เพื่อจะได้ทราบว่ามีการใช้เน็ตเวิร์กเพื่อการใด มากน้อยเพียงใด จะทำให้สามารถจัดการเน็ตเวิร์กได้อย่างเหมาะสมหรือในด้านการใช้งานเน็ตเวิร์กตามช่วงเวลาดังกล่าว หรือในด้านการใช้งานเน็ตเวิร์กตามช่วงเวลาต่างๆเวลาใดที่มีผู้ใช้มากเวลาใดที่มีผู้ใช้น้อย ผู้ใช้คนใดใช้แบนด์วิดธ์ไปในทางที่เป็นประโยชน์หรือไม่

6.3.4.2 เครื่องมือตรวจสอบเครือข่าย (Network Debugging Tools)

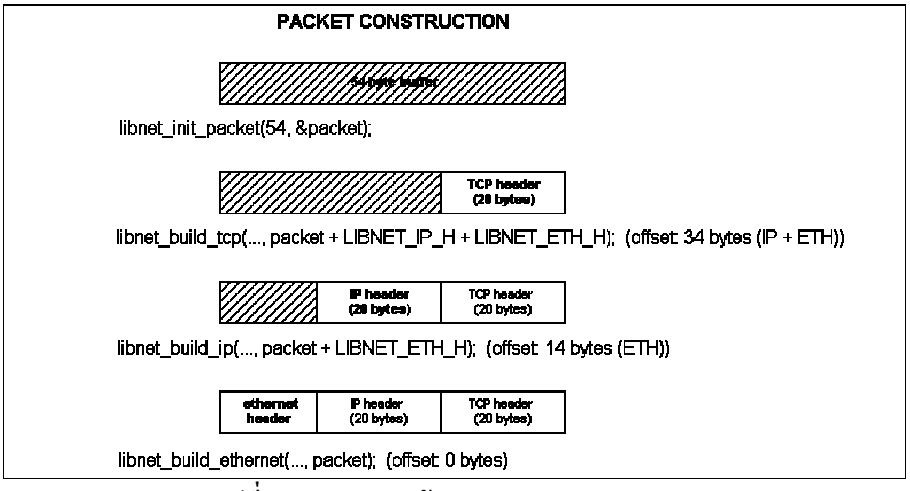
ในบางครั้งการหาสาเหตุความผิดปกติของแอพพลิเคชั่นต่างๆ จำเป็นที่ต้องสืบค้นลึกลงไปถึงเนื้อข้อมูลที่รับส่งกันจริงๆบนเน็ตเวิร์กเพื่อหาว่าเหตุใดการทำงานของแอพพลิเคชั่นจึงไม่สามาถทำงานได้ตามปกติ การได้เห็นข้อมูลที่ส่งกันจริงทำให้การวิเคราะห์และแก้ไขปัญหาสามาถดำเนินการอย่างรวดเร็วและตรงต่อหัญหาได้มากขึ้น โดยเฉพาะกรณีที่มีการใช้เครื่องมือในระดับเน็ตเวิร์กมาเกี่ยวข้องเช่น มีการส่งข้อมูลผ่านไฟร์วอลล์แล้วปัญหา หรือการทดสอบ ALC (Access Control List) ของเราเตอร์เป็นต้น

6.3.4.3 ตรวจจับแพ็กเก็ต (Packet Monitoring)

การศึกษาเทคนิคของโพรโตคอลในระดับเน็ตเวิร์กจำเป็นต้องเห็นข้อมูลที่สื่อสารกันอยู่จึงจะสามารถเข้าใจได้และเห็นภาพจริง แพ็กเก็ตมอนิเตอร์ริ่งจะเป็นเครื่องมือที่นำแพ็กเก็ตมาแสดงเห็นในรูปแบบต่างๆได้

6.3.4.4 ระบบตรวจจับผู้บุกรุก (Intrusion Detection System)

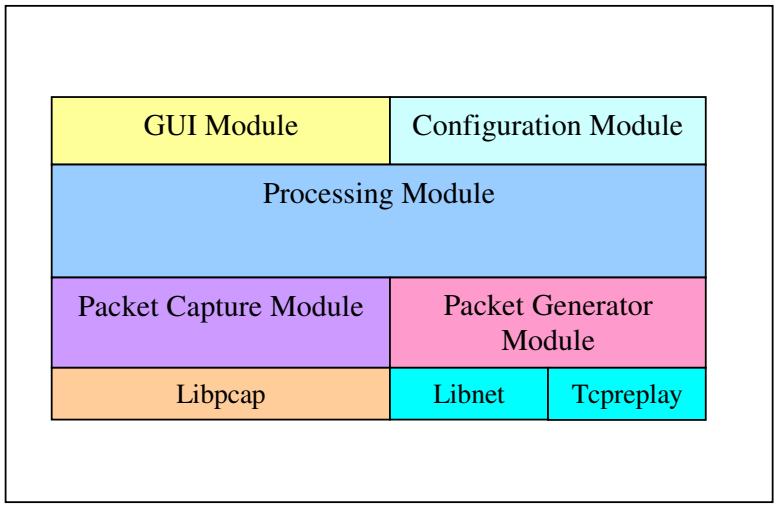
ระบบตรวจจับผู้บบุกรุกก็มีพื้นฐานมาจาการดักอ่านข้อมูลเพียงแต่เป็นการประยุกต์การวิเคราะห์รูปแบบการบุกรุกเข้ากับการดักอ่านข้อมูลบนเน็ตเวิร์กแบบเรียลไทม์ ทำให้สามารถทราบว่ามีกิจกรรมใดบนเน็ตเยิร์กที่มีแนวโน้มว่าจะเป็นการบุกรุกหรือทำอันตรายต่อผู้อื่น



รูปที่ 6-4 แสดงการสร้างแบบ LIBNET_LINK

6.4 การออกแบบโปรแกรม

<u>โครงสร้าง Software (Design)</u>



GUI Module : เป็นส่วนของโปรแกรมที่รับผิดชอบเกี่ยวกับ graphic user interface เพื่อสร้างความสะดวกในการติดต่อโดยตรงกับ ผู้ใช้

Configuration Module : เป็นส่วนของโปรแกรมที่รับผิดชอบการอ่านconfiguration file แล้วทำการสร้างแพ็กเก็ตตามลำดับใน configuration fileซึ่งผู้ใช้งานสามารถสร้าง configuration file ได้เอง

Packet Capture Module: เป็นส่วนของโปรแกรมที่รับผิดชอบการดักจับ packet ในเครือข่าย โดยใช้ไลบรารีชื่อ libpcap ช่วยในการดักจับข้อมูล

Packet Generator Module: เป็นส่วนของโปรแกรมที่รับผิดชอบการสร้าง packet อย่างที่ต้องการลงไปในระบบตามรูปแบบที่กำหนดจากConfiguration file สามารถสั่ง generate packet ได้ซ้ำๆ ตามต้องการ โดยใช้ไลบรารีชื่อ libnet และ Tcpreplay ช่วยในการสร้าง Packet Processing

Processing Module : เป็นส่วนของโปรแกรมที่ทำหน้าที่ประมวลผลลำดับของการทำงานที่สร้างจาก Configuration Module หรือจาก GUI แล้วสร้าง กระบวนการต่างๆ ตามลำดับนั้นโดยการสร้าง Packet จะส่งให้Packet Generator Module ทำงาน และในส่วนของการเก็บข้อมูลในเครือข่ายจะส่งให้ Packet Capture Module ทำงาน

GUI Module : เป็นส่วนของโปรแกรมที่รับผิดชอบเกี่ยวกับ graphic user interface เพื่อสร้างความสะดวกในการติดต่อโดยตรงกับ ผู้ใช้

Configuration Module : เป็นส่วนของโปรแกรมที่รับผิดชอบการอ่านconfiguration file แล้วทำการสร้างแพ็กเก็ตตามลำดับใน configuration fileซึ่งผู้ใช้งานสามารถสร้าง configuration file ได้เอง

Packet Capture Module:	เป็นส่วนของโปรแกรมที่รับผิดชอบการดักจับ packet ในเครือข่าย โดยใช้ไลบรารีชื่อ libpcap ช่วยในการดักจับข้อมูล
Packet Generator Module:	เป็นส่วนของโปรแกรมที่รับผิดชอบการสร้าง packet อย่างที่ต้องการลงไปในระบบตามรูปแบบที่กำหนดจากConfiguration file สามารถสั่ง generate packet ได้ซ้ำๆ ตามต้องการ โดยใช้ไลบรารีชื่อ libnet และ Tcpreplay ช่วยในการสร้าง Packet Processing
Processing Module :	เป็นส่วนของโปรแกรมที่ทำหน้าที่ประมวลผลลำดับของการทำงานที่สร้างจาก Configuration Module หรือจาก GUI แล้วสร้าง กระบวนการต่างๆ ตามลำดับนั้น โดยการสร้าง Packet จะส่งให้Packet Generator Module ทำงาน และในส่วนของการเก็บข้อมูลในเครือข่ายจะส่งให้ Packet Capture Module ทำงาน