# บทที่ 4

## การพัฒนาส่วนขอรับบริการ

ในบทนี้กล่าวถึง โปรแกรมที่ทำงานบนระบบยูนิกซ์โดยโปรแกรมเหล่านี้ขอใช้บริการจากส่วนให้บริการรหัสผ่านแบบใช้ครั้งเดียวที่ทำงานบนระบบปฏิบัติการดอส

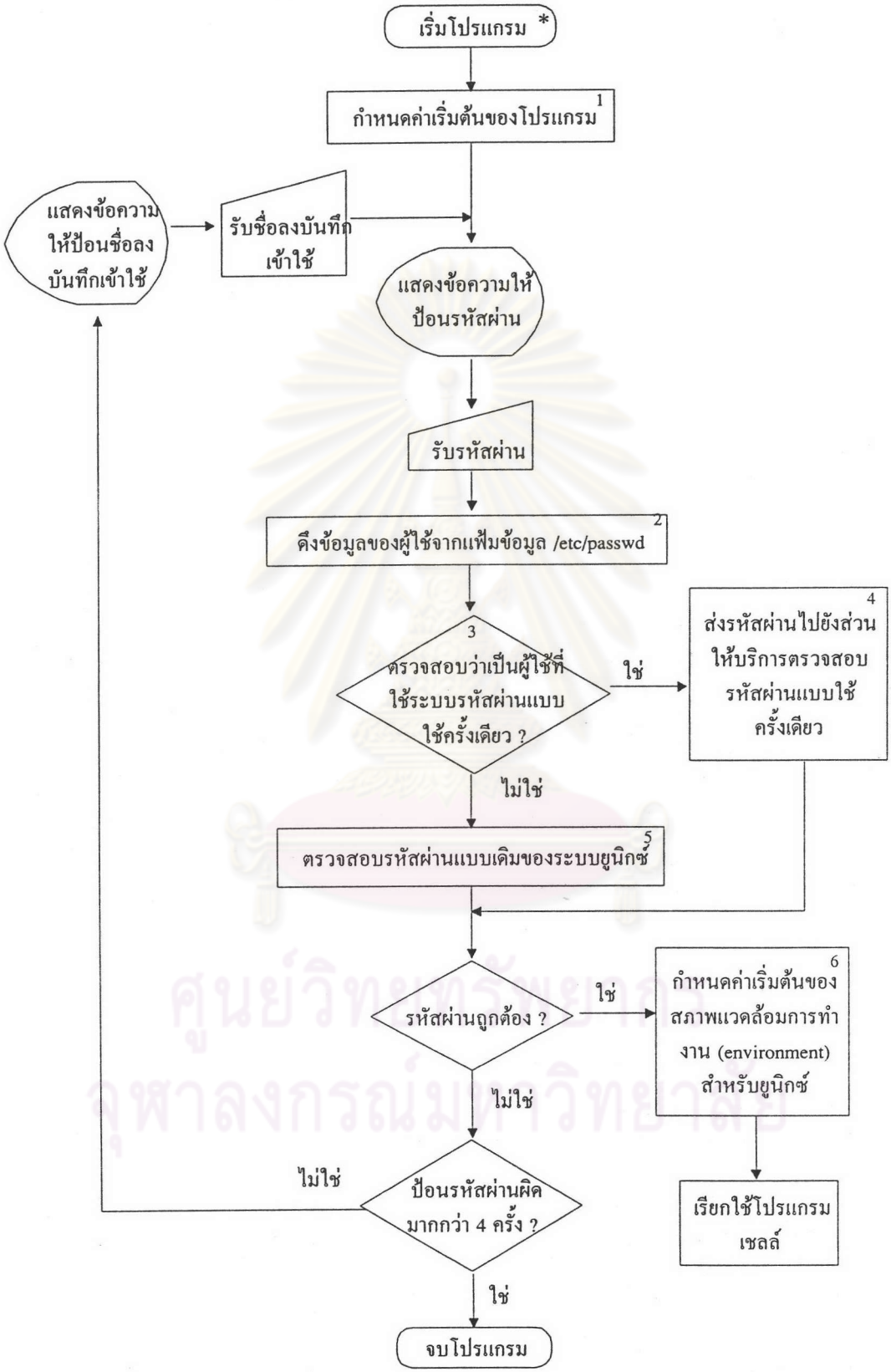
### หลักการพัฒนาโปรแกรมล็อกอิน

เนื่องจากการขอเข้าใช้ระบบต้องกระทำผ่านโปรแกรมล็อกอิน ดังนั้นจึงต้องปรับปรุงโปรแกรมล็อกอินเพื่อให้สามารถทำงานในลักษณะเดิมและสนับสนุนการตรวจสอบรหัสผ่านแบบใช้ครั้งเดียว ถ้าเป็นการตรวจสอบรหัสผ่านแบบเดิมจะยังคงการทำงานอย่างเดิม คือ ตรวจสอบรหัสผ่านจากแฟ้มข้อมูล /etc/passwd แต่ถ้าเป็นการตรวจสอบรหัสผ่านแบบใช้ครั้งเดียวต้องส่งรหัสผ่านไปยังโปรแกรมให้บริการตรวจสอบรหัสผ่าน

นอกจากการส่งรหัสผ่านเพื่อไปทำการตรวจสอบยังมีการส่งคำสั่งพิเศษอื่นไปยังโปรแกรมให้บริการตรวจสอบรหัสผ่านเพื่อทำการแก้ไขรหัสผ่านของผู้ใช้เมื่อประสบปัญหาในการขอเข้าใช้ระบบเป็นการทำให้ระบบสามารถดำเนินงานต่อไปอย่างถูกต้อง การส่งคำสั่งพิเศษอื่นจะกระทำผ่านทางโปรแกรมล็อกอินและรูปแบบของคำสั่งพิเศษได้อธิบายไว้ในหัวข้อรูปแบบการขอเข้าใช้ระบบของโปรแกรมล็อกอิน ในภาคผนวก ข.

### ขั้นตอนการทำงานของโปรแกรมล็อกอิน

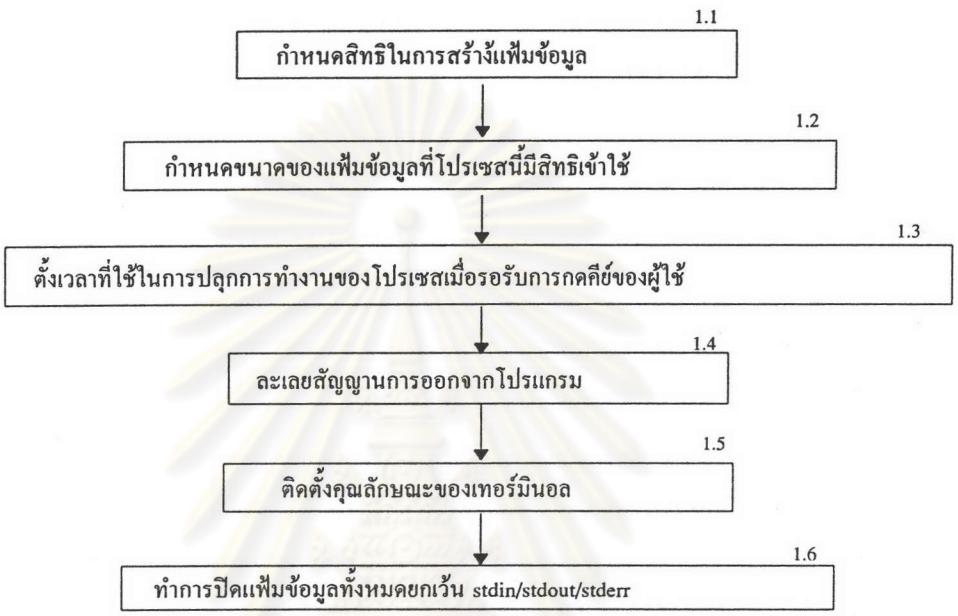
ในรูปที่ 4.1 แสดงขั้นตอนการทำงานของโปรแกรมล็อกอินพร้อมอธิบายชุดคำสั่งที่เกี่ยวข้องในแต่ละขั้นตอนของการทำงาน




* รับช่วงการทำงานจากโปรแกรม getty ซึ่งได้แสดงข้อความให้ป้อนชื่อลงบันทึกเข้าใช้และรับชื่อลงบันทึกเข้าใช้ไปแล้ว

รูป 4.1 แสดงขั้นตอนการทำงานของโปรแกรมล็อกอินในระบบให้บริการรหัสผ่านแบบใช้ครั้งเดียว

1. กำหนดค่าเริ่มต้นให้กับโปรแกรม เป็นการกำหนดสภาพแวดล้อมให้กับโปรแกรมล็อกอินเพื่อให้สามารถตรวจสอบรหัสผ่านของผู้ใช้ได้ เพราะในขณะที่เข้าใช้โปรแกรมล็อกอินยังไม่มีสภาพแวดล้อมในการทำงาน โดยแบ่งขั้นตอนการกำหนดค่าเริ่มต้นได้ดังรูป 4.2




รูป 4.2 แสดงขั้นตอนกำหนดค่าเริ่มต้นให้โปรแกรม

1.1 กำหนดสิทธิในการเข้าใช้แฟ้มข้อมูล (file mode creation mask) เนื่องจากหากมีการสร้างแฟ้มข้อมูลขึ้นใหม่ ระบบจะต้องกำหนดสิทธิในการเข้าใช้แฟ้มข้อมูลนั้นโดยอัตโนมัติ จึงต้องทำการกำหนดสิทธิโดยใช้ซิสเต็มคอล umask() ซึ่งมีรูปแบบดังนี้

```
#include <sys/types.h>
#include <stat.h>
int umask(int cmask);
```

โดยที่ cmask หมายถึง สิทธิในการเข้าใช้แฟ้มข้อมูลอยู่ในรูปเลขฐาน 8 เช่น กำหนดให้ cmask มีค่าเท่ากับ 664 หมายถึง แฟ้มข้อมูลที่ถูกสร้างขึ้นสามารถถูกอ่านและเขียนโดยเจ้าของและผู้ใช้ที่อยู่ในกลุ่มเดียวกับเจ้าของ ส่วนผู้ใช้คนอื่นสามารถอ่านได้เพียงอย่างเดียว

1.2 กำหนดขนาดแฟ้มข้อมูลที่โปรเซสมีสิทธิเข้าใช้โดยใช้ซิสเต็มคอล ulimit()

```
#include <ulimit.h>
long ulimit(int command, long limit);
```

โดยอาร์กิวเมนต์ตัวแรก คือ ชนิดของคำสั่ง เช่น 1 หมายถึง ค้นหาขนาดของแฟ้มข้อมูลที่โปรเซสนี้สามารถใช้ได้ และ 2 หมายถึง กำหนดขนาดของแฟ้มข้อมูลที่โปรเซสนี้มีสิทธิเข้าใช้ ส่วนอาร์กิวเมนต์ตัวที่ 2 คือ ขนาดของแฟ้มข้อมูล

1.3 ตั้งเวลาที่ใช้ในการปลุกการทำงานของโปรเซสเมื่อรอรับการกดแป้นพิมพ์จากผู้ใช้โดยใช้ซิสเต็มคอล alarm() และ signal() ซึ่งมีรูปแบบดังนี้

```
#include <signal.h>
#include <sys/unistd.h>
void alarm(int timeout);
void signal( SIGALRM, void (*func) (int) );
```

โดยที่ timeout เป็นระยะเวลาที่โปรเซสจะรอเมื่อไม่มีการทำงานใด ๆ เกิดขึ้น ถ้าหากเกินระยะเวลาดังกล่าวจะเกิดสัญญาน SIGALRM และใช้ซิสเต็มคอล signal() กำหนดฟังก์ชั่นที่จะทำงานเมื่อเกิดสัญญาน SIGALRM ขึ้น

1.4 ละเลยสัญญานการออกจากโปรแกรมโดยใช้ซิสเต็มคอล signal() ทำให้ผู้ใช้ไม่สามารถหยุดหรือขัดจังหวะการทำงานของโปรแกรมล็อกอินกลางคัน ซึ่งมีผลต่อความมั่นคงของระบบ

```
#include <signal.h>
void signal (SIGNAL, SIG_IGN);
```

โดยที่อาร์กิวเมนต์ตัวแรกคือ ชื่อของสัญญานที่ต้องการละเลยได้แก่ SIGINT และ SIGQUIT ส่วน SIG_IGN เป็นส่วนที่ระบุให้ละเลยสัญญานดังกล่าว

1.5 ทำการติดตั้งคุณลักษณะของเทอร์มินอล

```
#include  <sgtty.h>
void gtty(int fd, struct sgttyb *buf);
void stty(int fd, struct sgttyb *buf);
```

โดยที่อาร์กิวเมนต์ตัวแรกคือ fd เป็นตัวบอกแฟ้มในที่นี้เป็นการติดตั้ง stdin จึงใช้ 0 และอาร์กิวเมนต์ตัวที่สองคือ buf แสดงถึงคุณลักษณะเทอร์มินอล โดยที่ซิสเต็มคอล gtty() ทำการเรียกคุณลักษณะของเทอร์มินอลขึ้นมา ส่วนซิสเต็มคอล stty() ทำการติดตั้งคุณลักษณะของเทอร์มินอลลงไป และคุณลักษณะของเทอร์มินอลแสดงได้โดยใช้โครงสร้างต่อไปนี้

```
struct sgttyb  {
        char    sg_ispeed;      /* input speed   */
        char    sg_ospeed;      /* output speed */
        char    sg_erase;       /* erase character */
        char    sg_kill;        /* kill character */
        char    sg_flags;       /* mode flags   */
};
```

1.6 ปิดตัวบอกแฟ้มข้อมูลทั้งหมดยกเว้น stdin, stdout, stderr โดยใช้ซิสเต็มคอล close() ซึ่งมีรูปแบบดังนี้

```
#include  <unistd.h>
int close(filedes);     /*  โดยที่ fieldes เป็นตัวบอกแฟ้มข้อมูล  */
```

2. ดึงข้อมูลของผู้ใช้จากแฟ้มข้อมูล /etc/passwd โดยใช้ซิสเต็มคอล getpwnam()

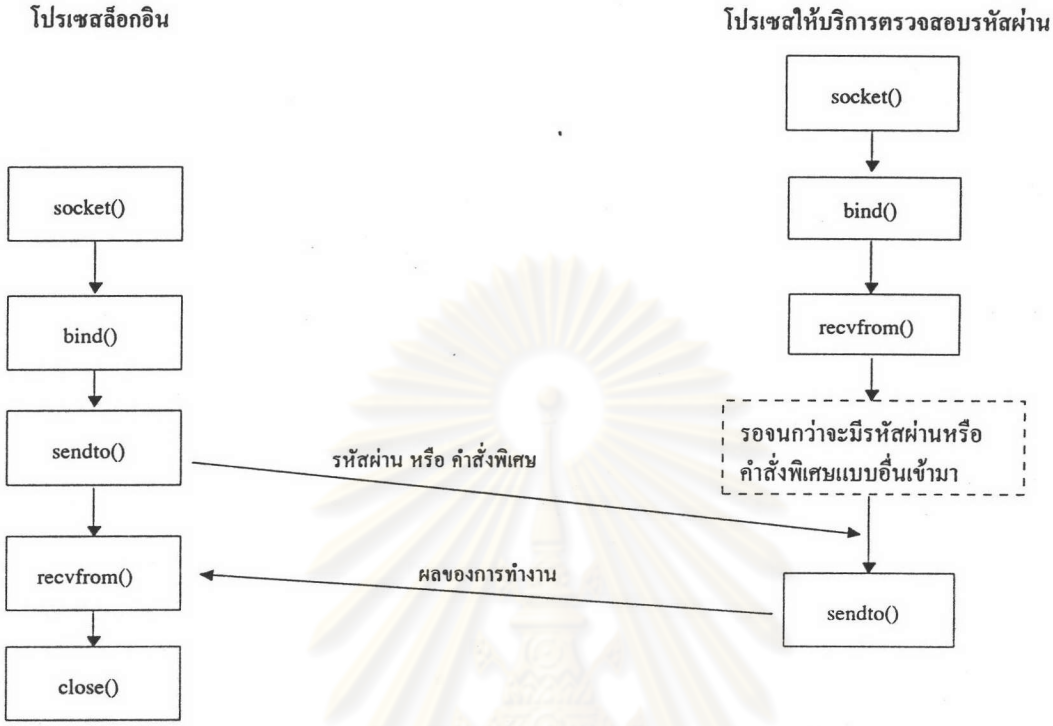
```
#include <pwd.h>
struct passwd *getpwnam(char *name);
```

โดยที่ name คือ ชื่อลงบันทึกเข้าใช้ของผู้ใช้ที่ต้องการค้นหาข้อมูล และเมื่อเรียกใช้ซิสเต็มคอลนี้ค่าที่ได้กลับมา คือ ข้อมูลของผู้ใช้ที่มีโครงสร้างดังนี้

```
struct passwd  {
        char    *pw_name;       /* login-name */
        char    *pw_passwd;     /* encrypted-password */
        int     pw_uid;         /* user-ID    */
        int     pw_gid;         /* group-ID   */
        char    *pw_comment;    /* not used   */
        char    *pw_gecos;      /* miscellany */
        char    *pw_dir;        /* login-directory */
        char    *pw_shell;      /* shell */
};
```

3. ตรวจสอบว่าเป็นผู้ใช้ที่ใช้ระบบรหัสผ่านแบบใช้ครั้งเดียว โดยตรวจสอบจากข้อมูลในฟิลด์ pw_gecos ซึ่งจะพบชื่อและนามสกุลของผู้ใช้ ในกรณีที่เป็นผู้ใช้ที่ใช้ระบบรหัสผ่านแบบใช้ครั้งเดียวจะต้องปรากฏคำว่า one-time password รวมอยู่ในฟิลด์ pw_gecos ด้วย

4. การส่งรหัสผ่านไปยังส่วนให้บริการตรวจสอบรหัสผ่านแบบใช้ครั้งเดียว เนื่องจากมีการติดต่อสื่อสารระหว่างโปรเซสในเครือข่าย จึงต้องมีการสร้างซอกเกต (socket) เพื่อใช้ในการติดต่อระหว่างกัน ในการพัฒนาโปรแกรมล็อกอินด้วยภาษาซีได้ใช้ เอพีไอ (application programming interface, API) แบบ Berkeley socket interface ซึ่งเป็นเอพีไอสำหรับการสื่อสารในเครือข่าย (communication API) ที่ใช้กันมากในระบบยูนิกซ์ โดยใช้รูปแบบการติดต่อสื่อสารจากหนังสือ UNIX NETWORK PROGRAMMING และในรูปที่ 4.3 แสดงขั้นตอนการติดต่อสื่อสารผ่านระบบซอกเกตพร้อมคำอธิบาย




รูป 4.3 แสดงการใช้ชุดคำสั่งซอกเกตสำหรับการสื่อสารแบบคอนเนคชั่นเลส

ในการติดต่อสื่อสารผ่านซอกเกตมีการเรียกใช้ซิสเต็มคอลต่าง ๆ ดังต่อไปนี้

4.1 ซอกเกตแอดเดรส (socket address) ซิสเต็มคอลในระบบเครือข่ายทั่วไปต้องการตัวชี้ไปยังโครงสร้างของซอกเกตแอดเดรสเพื่อเป็นอาร์กิวเมนต์ โดยโครงสร้างนี้ได้ถูกกำหนดไว้ใน <sys/socket.h> ดังนี้

```
struct sockaddr {
    u_short   sa_family;   /* address family: AF_xxx */
    char      sa_data[14];  /* up to 14 bytes of protocol-specific address */
};
```

สำหรับเนื้อที่ 14 ไบต์ที่เป็นเนื้อที่สำหรับ protocol-specific address นั้นจะถูกแปลตามชนิดของแอดเดรส ซึ่งสำหรับอินเตอร์เน็ตแฟมิลี่ (internet family) ได้กำหนดโครงสร้างไว้ใน <netinet/in.h> ดังนี้

```
struct in_addr {
    u_long    s_addr;   /*  32-bit netid/hostid network byte ordered  */
};

struct sockaddr_in {
    short      sin_family;          /* AF_INET */
    u_short    sin_port;            /* 16-bit port number */
    struct  in_addr  sin_addr;      /* 32-bit netid/hostid */
    char       sin_zero[8];         /* unused */
};
```

4.2 ซอกเกตซิสเต็มคอลพื้นฐาน ที่จำเป็นในการพัฒนาโปรแกรม

4.2.1 ซิสเต็มคอล socket เป็นสิ่งแรกที่โปรเซสที่ทำงานรับส่งข้อมูลผ่านระบบเครือข่ายจะต้องเรียกใช้ มีรูปแบบการใช้งานดังนี้

```
#include <sys/types.h>
#include <sys/socket.h>
int socket(int family, int type int protocal)
```

โพรโตคอลที่ใช้ในงานวิจัยชิ้นนี้เป็นแบบยูดีพี ค่าของ family จึงเป็นแบบ AF_INET และค่าของ type เป็น SOCK_DGRAM สำหรับอินเตอร์เน็ตโพรโตคอลแบบ datagram socket เมื่อเรียกใช้ซิสเต็มคอลนี้จะได้ค่าจำนวนเต็มที่เป็นตัวบอกซอกเกต sockfd หรือ socket descriptor เพื่อใช้ในการอ้างถึงซอกเกตนี้ต่อไป

4.2.2 ซิสเต็มคอล bind ใช้กำหนดชื่อให้กับซอกเกต มีรูปแบบการใช้งานดังนี้

```
#include <sys/types.h>
#include <sys/socket.h>
int bind(int sockfd, struct sockaddr *myaddr, int addrlen)
```

อาร์กิวเมนต์แรกคือ ตัวบอกซอกเกต อาร์กิวเมนต์ตัวที่สองคือ ตัวชี้ไปยัง protocol-specific address และอาร์กิวเมนต์ตัวที่สามคือ ขนาดของโครงสร้างข้อมูลแอดเดรส ซิสเต็มคอล bind นี้ใช้ทำให้แน่ใจว่าในระบบได้กำหนดแอดเดรสที่ไม่ซ้ำกันกับโปรเซสอื่น

4.2.3 ซิสเต็มคอล sendto ใช้ในการส่งกลุ่มข้อมูล มีรูปแบบการใช้งานดังนี้

```
#include <sys/types.h>
int sendto(int sockfd,  char *buff,  int nbytes,  int flag,
        struct sockaddr *to,  int addrlen)
```

โดยที่ sockfd คือ ค่าของตัวบอกซอกเกตที่ได้จากการเรียกใช้ซิสเต็มคอล socket อาร์กิวเมนต์ตัวที่ 2 และ 3 คือข้อมูลและขนาดของข้อมูลที่ต้องการส่ง และ flag จะถูกติดตั้งเป็น 0 ส่วนอาร์กิวเมนต์ที่ 5 และ 6 คือตัวชี้ไปยัง protocol-specific address และ ขนาดของโครงสร้างข้อมูลแอดเดรส

4.2.4 ซิสเต็มคอล recvfrom ใช้ในการรับกลุ่มข้อมูล มีรูปแบบการใช้งานดังนี้

```
#include <sys/types.h>
int  recvfrom(int sockfd,  char *buff,  int nbytes,  int flag,
        struct sockaddr *to,  int addrlen)
```

โดยที่ sockfd คือ ค่าของตัวบอกซอกเกต อาร์กิวเมนต์ตัวที่ 2 และ 3 คือ ข้อมูลและขนาดของข้อมูลที่รับเข้ามา และ flag จะถูกติดตั้งเป็น 0 ส่วนอาร์กิวเมนต์ที่ 5 และ 6 คือ ตัวชี้ไปยัง protocol-specific address และ ขนาดของโครงสร้างข้อมูลแอดเดรส

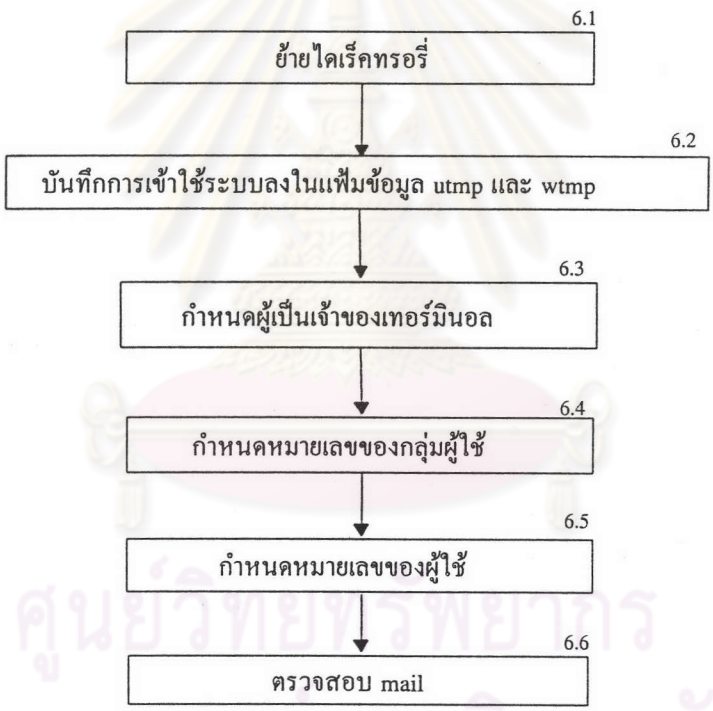
4.2.5 ซิสเต็มคอล close ใช้ปิดการใช้งานซอกเกต ในการติดต่อสื่อสารแบบคอนเนคชั่นเลสต้องมีการปิดการใช้งานซอกเกต เพื่อให้โปรเซสอื่นสามารถใช้ซิสเต็มคอล bind กำหนดแอดเดรสส่วนนี้ไปใช้ได้

5. การตรวจสอบว่าเป็นรหัสผ่านที่ถูกต้อง แบ่งการตรวจสอบออกเป็น 2 ประเภทคือ

5.1 ตรวจสอบโดยใช้ระบบรหัสผ่านแบบเดิม นำรหัสผ่านที่ผู้ใช้ป้อนมาทำการเข้ารหัสโดยใช้ซิสเต็มคอล crypt() แล้วนำผลที่ได้ไปเปรียบเทียบกับข้อมูลในฟิลด์ pw_passwd

5.2 ตรวจสอบโดยใช้ระบบรหัสผ่านแบบใช้ครั้งเดียว โดยดูจากผลการตรวจสอบรหัสผ่านที่ส่งมาจากส่วนให้บริการตรวจสอบรหัสผ่านแบบใช้ครั้งเดียว

6. กำหนดค่าเริ่มต้นของสภาพแวดล้อมการทำงานให้แก่ผู้ใช้ โดยแบ่งขั้นตอนการทำงานได้ดังรูปที่ 4.4




รูป 4.4 แสดงขั้นตอนการกำหนดค่าเริ่มต้นของสภาพแวดล้อมสำหรับผู้ใช้

6.1 การย้ายไดเร็คทรอรี่ไปยังไดเร็คทรอรี่ที่ต้องการโดยใช้ซิสเต็มคอล chdir()

```
int chdir(char *pathname);
```

โดยที่ pathname เป็นไดเร็คทรอรี่ที่ต้องการย้ายไป เมื่อเข้าใช้ระบบยูนิกซ์จะ

ย้ายไปอยู่ในไดเร็คทรอรี่บ้าน (home directory) ของผู้ใช้ ซึ่งได้กำหนดไว้ในฟิลด์ pw_dir ของไฟล์ /etc/passwd

6.2 บันทึกการเข้าใช้ระบบ ในระบบยูนิกซ์จะบันทึกการเข้าใช้ระบบของผู้ใช้ไว้ในแฟ้มข้อมูล utmp และบันทึกทั้งการเข้าใช้และออกจากระบบของผู้ใช้ไว้ในแฟ้มข้อมูล wtmp ซึ่งประกอบด้วย เทอร์มินอลที่ใช้ ชื่อของผู้ใช้ ชื่อของเครื่องแม่ข่ายที่ผู้ใช้ใช้ติดต่อเข้ามากรณีที่เป็นเน็ตเวิร์คล็อกอิน และเวลาเข้าใช้ระบบ

6.3 กำหนดผู้เป็นเจ้าของเทอร์มินอล โดยใช้ซิสเต็มคอล chown() ที่มีรูปแบบดังนี้

```
#include <sys/types.h>
int chown(const char *path, int owner, int group);
```

อาร์กิวเมนต์แรกเป็นอุปกรณ์ที่ต้องการเปลี่ยนแปลงผู้เป็นเจ้าของ อาร์กิวเมนต์ตัวที่สองระบุถึงเจ้าของโดยให้มีค่าเท่ากับหมายเลขของผู้ใช้ในฟิลด์ pw_uid และอาร์กิวเมนต์ตัวที่สามระบุถึงกลุ่มที่เป็นเจ้าของโดยให้มีค่าเท่ากับหมายเลขของกลุ่มผู้ใช้ในฟิลด์ pw_gid

6.4 กำหนดหมายเลขของกลุ่มผู้ใช้ โดยใช้ซิสเต็มคอล setgid() ที่มีรูปแบบดังนี้

```
#include <sys/types.h>
int setgid(int group);
```

โดยที่ group คือกลุ่มของผู้ใช้โดยให้มีค่าเท่ากับกลุ่มผู้ใช้ในฟิลด์ pw_gid

6.5 กำหนดหมายเลขของผู้ใช้ โดยใช้ซิสเต็มคอล setuid() ที่มีรูปแบบดังนี้

```
#include <unistd.h>
int setuid(int uid);
```

โดยที่ uid คือ หมายเลขของผู้ใช้มีค่าเท่ากับหมายเลขของผู้ใช้ในฟิลด์ pw_uid

6.6 ตรวจสอบ mail ของผู้ใช้ โดยใช้ซิสเต็มคอล stat() ที่มีรูปแบบดังนี้

```
#include <sys/types.h>
#include <stat.h>
int stat(char *pathname, struct stat *buf);
```

โดยที่อาร์กิวเมนต์ตัวแรกระบุถึงชื่อแฟ้มข้อมูลที่บรรจุ mail ของผู้ใช้ เมื่อเรียกใช้ซิสเต็มคอลนี้สิ่งที่ได้กลับมาคือ ข้อมูลที่บรรจุอยู่ในอาร์กิวเมนต์ตัวที่สองที่มีโครงสร้างดังนี้

```
struct stat {
   ushort  st_mode;     /* file type and file access permission */
   ino_t   st_ino;      /* i-node number      */
   dev_t   st_dev;      /* ID of device contain directory for this file */
   short   st_nlink;    /* number of link     */
   ushort  st_uid;      /* user ID            */
   ushort  st_gid;      /* group ID           */
   dev_t   st_rdev;     /* ID of device       */
   off_t   st_size;     /* file size in byte  */
   time_t  st_atime;    /* time of last file access          */
   time_t  st_mtime;    /* time of last file modification  */
   time_t  st_ctime;    /* time of last file status change */
   };
```

ในบทนี้ได้นำเสนอการพัฒนาโปรแกรมในส่วนขอรับบริการซึ่งได้แก่โปรแกรมล็อกอิน และบทต่อไปจะกล่าวถึงการพัฒนาโปรแกรมในส่วนให้บริการ