บทที่ 4

การพัฒนาโปรแกรม

ในบทนี้กล่าวถึง การพัฒนากระบวนงานที่เกี่ยวข้องกับการพิสูจน์ตัวจริงของโปรแกรมให้บริการถ่ายโอนแฟ้มโดยกระบวนงานเหล่านี้จะทำหน้าที่ตรวจสอบผู้ใช้ว่ามีสิทธิในการขอเข้ารับบริการถ่ายโอนแฟ้มหรือไม่ รวมทั้งยังแยกแยะคุณสมบัติของผู้ใช้ที่เข้ามาขอรับบริการด้วยว่าเป็นอย่างไร โดยกระบวนงานหลักๆ ได้แก่

- ฟังก์ชัน sgetpwnam
- กระบวนงาน PASS
- ฟังก์ชัน request_otp_password

4.1 ฟังก์ชัน sgetpwnam

ปรับปรุงฟังก์ชัน sgetpwnam ตามขั้นตอนในรูปที่ 3.5

4.1.1 ดึงข้อมูลผู้ใช้จากระบบยูนิกซ์

4.1.1.1 ดึงข้อมูลทั่วไปของผู้ใช้ โดยในระบบยูนิกซ์แต่ละระบบจะมีแฟ้มที่เก็บข้อมูลทั่วไปของผู้ใช้คือ แฟ้ม /etc/passwd สามารถดึงข้อมูลโดยใช้ซิสเต็มคอล getpwnam()

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

โดยที่ name คือ ชื่อลงบันทึกเข้าของผู้ใช้ โดยค่าที่คืนกลับมาจะมีโครงสร้างเป็น passwd ซึ่งได้กล่าวถึงโครงสร้างนี้แล้วในบทที่ 3 หากระบบรักษาความปลอดภัยของระบบยูนิกซ์เป็นแบบพิเศษ จะไม่สามารถดึงข้อมูลรหัสผ่านได้ ดังตารางที่ 4.1

Unix/Version	Security	รหัสผ่านที่ได้รับการเข้ารหัสแล้วในเขตข้อมูล pw_passwd
AIX 4.2	AIX Standard	x
Digital Unix 3.2	Digital Enhanced C2	o
HP-UX 10.20	Standard	x
HP-UX 10.20	HP Trusted System	o
Linux 2.0.27	Standard	x
Linux 2.0.27	Shadow	o
SunOS 5.3/5.5.1	Shadow	o

x คือ สามารถดึงข้อมูลได้

o คือ ไม่สามารถดึงข้อมูลได้

ตารางที่ 4.1 แสดงการดึง pw_gecos โดยใช้ซิสเต็มคอล getpwnam()

4.1.1.2 ดึงข้อมูลผู้ใช้อย่างละเอียดโดยใช้คำสั่งเฉพาะของแต่ละระบบรักษาความปลอดภัยของระบบยูนิกซ์ ดังตารางที่ 4.2

Unix/Version	Security	Subroutine
Digital Unix 3.2	Digital Enhanced C2	set_auth_parameters() getprpwnam() getprdfnam()
HP-UX 10.20	HP Trusted System	getprpwnam() getprdfnam()
Linux 2.0.27	Shadow	getspnam()
SunOS 5.3/5.5.1	Shadow	getspnam()

ตารางที่ 4.2 แสดงการดึงข้อมูลรหัสผ่านและข้อมูลอื่นโดยใช้คำสั่งของแต่ละระบบ

คำสั่งและรูปแบบการเรียกใช้	โครงสร้างข้อมูล

#include <sys/types.h>
#include <sys/security.h>
/* for Digital */
#include <hpsecurity.h>
/* for HP-UX */
#include <prot.h>
 	
struct pr_passwd
*getprpwnam (char *name);
	Struct pr_field {
	/* Identity: */
	char fd_name[AUTH_MAX_UNAME_SIZE];
	uid_t fd_uid; /* uid associated with name above */
	char fd_encrypt[AUTH_MAX_CIPHERTEXT_LENGTH];
	/* Encrypted password */
	...
	/* Password maintenance parameters: */
	time_t fd_min; /* minimum time between password changes */
	time_t fd_expire; /* expiration time duration in secs */
	time_t fd_lifetime; /* account death duration in seconds */
	time_t fd_schange; /* last successful change in secs past 1/1/70 */
	time_t fd_uchange; /* last unsuccessful change */
	...
	/* Login parameters: */
	short fd_nlogins; /* consecutive unsuccessful logins */
	char fd_tod[AUTH_TOD_SIZE]; /* times when user may login */
	short fd_max_tries; /* maximum unsuc login tries allowed */
	char fd_lock; /* Unconditionally lock account? */
	time_t fd_expdate; /* time at which to auto-retire the account for Digital */
	time_t fd_acct_expire; /* time at which to auto-retire the account for HP-UX */
	};
	struct pr_flag {
	unsigned int
	/* Identity: */
	fg_name:1, /* Is fd_name set? */
	fg_uid:1, /* Is fd_uid set? */
	fg_encrypt:1, /* Is fd_encrypt set? */
	...
	/* Password maintenance parameters: */
	fg_min:1, /* Is fd_min set? */
	fg_expire:1, /* Is fd_expire set? */
	fg_lifetime:1, /* Is fd_lifetime set? */
	fg_schange:1, /* Is fd_schange set? */
	fg_uchange:1, /* Is fd_fchange set? */
	...
	/* Login parameters: */
	fg_nlogins:1, /* Is fd_nlogins set? */
	fg_max_tries:1, /* Is fd_max_tries set? */
	fg_lock:1, /* Is fd_lock set? */
	fg_tod:1, /* Is fd_tod set? */
	fg_expdate:1, /* Is fd_expdate set? For Digital Unix */
	fg_acct_expire:1, /* Is fd_acct_expire set? For HP-UX */
	};
	struct pr_passwd {
	struct pr_field ufld; /* user specific fields */
	struct pr_flag uflg; /* user specific flags */
	struct pr_field sfld; /* system wide fields */
	struct pr_flag sflg; /* system wide flags */
	};

ตารางที่ 4.3 แสดงรูปแบบการเรียกใช้คำสั่ง getprpwnam() และโครงสร้างข้อมูล

คำสั่งและรูปแบบการเรียกใช้	โครงสร้างข้อมูล
#include <sys/types.h> #include<sys/security.h> /* for Digital */ #include <hpsecurity.h> /* for HP-UX */ #include <prot.h> **struct pr_default *getprdfnam(char *name);**	struct pr_default { char dd_name[20]; char dg_name; struct pr_field prd; struct pr_flag prg; struct t_field tcd; struct t_flag tcg; struct dev_field devd; struct dev_flag devg; struct system_default_fields sfld; struct system_default_flags sflg; } ;

ตารางที่ 4.4 แสดงรูปแบบการเรียกใช้คำสั่ง getprdfnam() และโครงสร้างข้อมูล

คำสั่งและรูปแบบการเรียกใช้	โครงสร้างข้อมูล

#include <shadow.h>
 	
struct spwd *getspnam (char *name);
	struct spwd {
	char *sp_namp; /* user login name */
	char *sp_pwdp; /* encrypted password */
	long sp_lstchg; /* last password change */
	int sp_min; /* days until change allowed. */
	int sp_max; /* days before change required */
	int sp_warn; /* days warning for expiration */
	int sp_inact; /* days before account inactive */
	int sp_expire; /* date when account expires */
	int sp_flag; /* reserved for future use */
	}

ตารางที่ 4.5 แสดงรูปแบบการเรียกใช้คำสั่ง getspnam() และโครงสร้างข้อมูล

4.1.2 ตรวจสอบคุณสมบัติของผู้ใช้ โดยอ้างอิงข้อกำหนดของคำสั่งและโครงสร้างข้อมูลจากตารางที่ 4.2 – 4.5 กำหนดให้

df คือ ค่าโดยปริยายที่ได้จากการเรียกคำสั่ง getprdfnam()

pr คือ ข้อมูลของผู้ใช้ ได้จากการเรียกคำสั่ง getprpwnam()

sp คือ ข้อมูลของผู้ใช้ ได้จากการเรียกคำสั่ง getspnam()

Disable Account		
Unix/Version	**Security**	**Operation**
AIX 4.2	AIX Standard	*If ErrorNo. Of loginrestriction() = EPERM then "Error"*
Digital Unix 3.2 HP-UX 10.20	Digital Enhanced C2 HP Trusted System	*If pr->uflg.fg_lock then* *begin If pr->ufld.fd_lock then* *"Account Disable"* *end* *else* *begin If df->prg.fg_lock then* *If df->prd.fd_lock then* *"Account Disable"* *end*
Linux 2.0.27	Shadow	*If sp->sp_pwdp has "!" then "Account Disable"*
SunOS 5.3/5.5.1	Shadow	*If sp->sp_pwdp has "*LK*" then "Account Disable"*

ตารางที่ 4.6 แสดงการพัฒนาโปรแกรมเพื่อตรวจสอบคุณสมบัติการปิดทาง(disable account)

Error Login Retries		
Unix/Version	**Security**	**Operation**
AIX 4.2	AIX Standard	*If ErrorNo. Of loginrestriction() = EPERM then "Error"*
Digital Unix 3.2 HP-UX 10.20	Digital Enhanced C2 HP Trusted System	*If pr->uflg.fg_max_tries then* *begin If (pr->ufld.fd_nlogins >=* *pr->ufld.fd_max_tries) then* *"Login retries error"* *end* *else* *begin If df->prg.fg_max_tries then* *If (pr->ufld.fd_nlogins >=* *df->prd.fd_max_tries) then* *"Login retries error"* *end*

ตารางที่ 4.7 แสดงการพัฒนาโปรแกรมเพื่อตรวจสอบจำนวนครั้งในการบันทึกเข้าผิดพลาด

Password Expired		
Unix/Version	**Security**	**Operation**
AIX 4.2	AIX Standard	*If passwdexpired() = 1 then "Password Expired"*
Digital Unix 3.2 HP-UX 10.20	Digital Enhanced C2 HP Trusted System	*If pr->uflg.fg_expire then* *begin If (pr->ufld.fd_expire <> 0) and* *((pr->ufld.fd_schange +* *pr->ufld.fd_expire) < now) then* *"Password Expired"* *end* *else* *begin If df->prg.fg_expire then* *If (df->prd.fd_expire <> 0) and* *((pr->ufld.fd_schange +* *df->prd.fd_expire) < now) then* *"Password Expired"* *end*
Linux 2.0.27 SunOS 5.3/5.5.1	Shadow	*If (sp->sp_max > 0) and (sp->sp_lstchg > 0) and* *((sp->sp_lstchg + sp->sp_max) < today) then* *"Password Expired"*

ตารางที่ 4.8 แสดงการพัฒนาโปรแกรมตรวจสอบอายุการใช้งานรหัสผ่าน

Password Dead		
Unix/Version	**Security**	**Operation**
AIX 4.2	AIX Standard	*If passwdexpired() = 2 then "Password Dead"*
Digital Unix 3.2 HP-UX 10.20	Digital Enhanced C2 HP Trusted System	*If pr->ufig.fg_lifetime then* *begin If (pr->ufld.fd_lifetime <> 0) and* *((pr->ufld.fd_schange +* *pr->ufld.fd_lifetime) < now) then* *"Password Dead"* *end* *else* *begin If df->prg.fg_lifetime then* *If (df->prd.fd_lifetime <> 0) and* *((pr->ufld.fd_schange +* *df->prd.fd_lifetime) < now) then* *"Password Dead"* *end*
Linux 2.0.27 SunOS 5.3/5.5.1	Shadow	*If (sp->sp_max > 0) and (sp->sp_lstchg > 0)* *and (sp->sp_inact >0) and* *((sp->sp_lstchg + sp->sp_max + sp->sp_inact)* *< today) then* *"Password Dead"*

ตารางที่ 4.9 แสดงการพัฒนาโปรแกรมตรวจสอบการปิดทางเพราะรหัสผ่านหมดอายุ

Account Expired		
Unix/Version	**Security**	**Operation**
AIX 4.2	AIX Standard	*If ErrorNo of loginrestrictions()=ESTALE then* *"Account Expired"*
Digital Unix 3.2	Digital Enhanced C2	*If pr->uflg.fg_expdate then* *begin If (pr->ufld.fd_expdate <> 0) and* *(pr->ufld.fd_expdate < now) then* *"Account Expired"* *end* *else* *begin If df->prg.fg_expdate then* *If (df->prd.fd_expdate <> 0) and* *(df->prd.fd_expdate < now) then* *"Account Expired"* *end*
HP-UX 10.20	HP Trusted System	*If pr->uflg.fg_acct_expire then* *begin If (pr->ufld.fd_acct_expire <> 0) and* *(pr->ufld.fd_acct_expire < now) then* *"Account Expired"* *end* *else* *begin If df->prg.fg_acct_expire then* *If (df->prd.fd_acct_expire <> 0) and* *(df->prd.fd_acct_expire < now) then* *"Account Expired"* *end*
Linux 2.0.27 SunOS 5.3/5.5.1	Shadow	*If (sp->sp_expire > 0) and* *(sp->sp_expire < today) then* *"Account Expired"*

ตารางที่ 4.10 แสดงการพัฒนาโปรแกรมเพื่อตรวจสอบการหมดอายุของบัญชีผู้ใช้

Login Time		
Unix/Version	**Security**	**Operation**
AIX 4.2	AIX Standard	*If ErrorNo. Of loginrestrictions() = EACCS then "Error"*
Digital Unix 3.2 HP-UX 10.20	Digital Enhanced C2 HP Trusted System	*If pr->uflg.fg_tod then* *begin if now is not in pr->ufld.fd_tod then* *"Can not login this time"* *end* *else* *begin if df->prg.fg_tod then* *If now is not in df->prd.fd_tod then* *"Can not login this time"* *end*

ตารางที่ 4.11 แสดงการพัฒนาโปรแกรมเพื่อตรวจสอบเวลาที่ให้บันทึกเข้าได้

คำสั่ง loginrestrictions มีโครงสร้างและการเรียกใช้ดังนี้

```
#include <login.h>
int loginrestrictions (char *name, int mode, char *tty, char **msg);
```

หากค่าที่คืนจากคำสั่ง loginrestrictions() เป็น 0 แปลว่าผู้ใช้สามารถบันทึกเข้าได้ แต่หากเป็น –1 หมายความว่า ไม่สามารถบันทึกเข้าได้ โดยที่ตัวแปรบอกความผิดพลาด (errno) จะบอกความผิดพลาดที่เกิดขึ้นเช่น

ENOENT คือ ไม่มีผู้ใช้ชื่อนี้
ESTALE คือ บัญชีผู้ใช้หมดอายุ
EPERM คือ บัญชีถูกปิดทาง หรือ บันทึกเข้าผิดเกินกำหนด
EACCS คือ ไม่สามารถบันทึกเข้าได้ขณะนี้ ฯลฯ

คำสั่ง passwdexpired() มีโครงสร้างและการเรียกใช้ดังนี้

```
int passwdexpired(char *name, char **msg);
```

หากค่าที่คืนจากคำสั่ง passwdexpired() คือ

1 คือ รหัสผ่านหมดอายุ ผู้ใช้ควรจะเปลี่ยนรหัสผ่าน

2 คือ รหัสผ่านหมดอายุ ต้องให้ผู้บริหารระบบเปิดทางให้ (enable)

-1 คือ เกิดความผิดพลาดในการทำงาน

4.2 กระบวนงาน PASS

ปรับปรุงกระบวนงาน PASS ตามขั้นตอนในรูปที่ 3.7 ของบทที่ 3

4.2.1 หาค่าเปรียบเทียบรหัสผ่าน โดยนำรหัสผ่านที่ผู้ใช้ป้อนมาเข้ารหัสโดยใช้ซิสเต็มคอล crypt()

```
#include <crypt.h>
#include <unistd.h>
char *crypt( const char *key, const char *salt);
```

หากใช้ระบบรักษาความปลอดภัย Digital Enhanced C2 หรือ HP Trusted System ใช้ซิสเต็มคอล bigcrypt()

```
#include <hpsecurity.h> OR #include <sys/security.h>
#include <prot.h>
char *bigcrypt(char *key, char *salt);
```

4.2.2 ใช้ระบบรหัสผ่านแบบใช้ครั้งเดียวหรือไม่ ทราบได้โดยการตั้งค่าไว้ก่อนการแปลโปรแกรม ดังนั้นหากไม่ต้องการให้มีการใช้รหัสผ่านแบบใช้ครั้งเดียว แต่ต้องการให้มีการตรวจสอบคุณสมบัติเพิ่มเติมในการบันทึกเข้า สามารถปิดทางของการแปลไม่ให้ใช้รหัสผ่านแบบใช้ครั้งเดียว

4.2.3 ตรวจสอบว่าผู้ใช้เป็นระบบรหัสผ่านแบบใช้ครั้งเดียวหรือไม่ โดยการตรวจที่เขตข้อมูล pw_gecos หากปรากฏข้อความ one-time password หมายความว่าเป็นผู้ใช้ระบบรหัสผ่านแบบใช้ครั้งเดียว

4.2.4 ส่งรหัสผ่านไปยังโปรแกรมให้บริการตรวจสอบรหัสผ่าน โดยใช้ฟังก์ชัน request_otp_passsword

4.2.5 ตรวจสอบรหัสผ่านบนระบบยูนิกซ์ โดยใช้ค่าที่ได้จากข้อ 4.2.1 มาเปรียบเทียบกับค่าที่ดึงมาได้จากกระบวนงาน sgetpwnam

4.2.6 กำหนดค่าเริ่มต้นของสภาพแวดล้อมการทำงานให้กับผู้ใช้ ได้แก่

- กำหนดหมายเลขกลุ่มผู้ใช้ (setgid())
- บันทึกการเข้าใช้ระบบใน wtmp
- กำหนดหมายเลขผู้ใช้ (setuid())
- ย้ายสารบบ (chdir())
- กำหนดสิทธิในการสร้างแฟ้มข้อมูล (umask())

4.3 ฟังก์ชัน request_otp_password

พัฒนาฟังก์ชัน request_otp_password ตามขั้นตอนในรูปที่ 3.8 ของบทที่ 3

4.3.1 อ่านค่าต่างๆในแฟ้ม /etc/otpconfig เพื่อกำหนดค่าโครงแบบให้กับฟังก์ชัน request_otp_password

4.3.2 ตรวจสอบค่า otphost ในแฟ้ม /etc/otpconfig หากไม่พบจะคืนค่า โครงแบบไม่สมบูรณ์ ส่วนค่าอื่นๆ หากไม่พบ จะใช้ค่าโดยปริยายที่ได้กำหนดไว้แล้วในโปรแกรม

4.3.3 กำหนดค่าเริ่มต้นในการติดต่อสื่อสาร โดยใช้เอพีไอ (Application Programming Interface, API) แบบ Berkley socket interface

4.3.3.1 ซิสเต็มคอล socket จะต้องมีการบ่งชี้ว่าจะใช้การติดต่อสื่อสารแบบใด เช่น อินเทอร์เน็ต ทีซีพี หรือ อินเทอร์เน็ต ยูดีพี เป็นต้น

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

ค่าของ family เป็นตัวบ่งชี้โพรโทคอลที่ใช้ โดยในซิสเต็มคอล socket จะอนุญาตให้ใช้ โพรโทคอลได้ 4 ประเภท ได้แก่

AF_UNIX	หมายความว่า ยูนิกซ์โพรโทคอล
AF_INET	หมายความว่า อินเทอร์เน็ตโพรโทคอล
AF_NS	หมายความว่า Xerox NS โพรโทคอล
AF_IMPLINK	หมายความว่า IMP ลิงก์เลเยอร์

ค่าของ type เป็นตัวบ่งชี้ชนิดของ socket

SOCK_STREAM	หมายความว่า stream socket
SOCK_DGRAM	หมายความว่า datagram socket
SOCK_RAW	หมายความว่า raw socket
SOCK_SEQPACKET	หมายความว่า sequenced packet Socket

ค่าของ protocol เป็นตัวบ่งชี้ชนิดของโพรโทคอลในระดับสูงกว่า โดยในตารางที่ 4.1 แสดงการใช้งานร่วมกันของ family type protocol

Family	*type*	*protocol*	*Actual protocol*
AF_INET	***SOCK_DGRAM***	***IPPROTO_UDP***	***UDP***
AF_INET	SOCK_STREAM	IPPORTO_TCP	TCP
AF_INET	SOCK_RAW	IPPROTO_ICMP	ICMP
AF_INET	SOCK_RAW	IPPROTO_RAW	(raw)
AF_NS	SOCK_STREAM	NSPROTO_SPP	SPP
AF_NS	SOCK_SEQPACKET	NSPROTO_SPP	SPP
AF_NS	SOCK_RAW	NSPROTO_ERROR	Error Protocol
AF_NS	SOCK_RAW	NSPROTO_RAW	(raw)

ตารางที่ 4.12 แสดงการใช้งานร่วมกันของ family type และ protocol

ในการติดต่อสื่อสารกับส่วนให้บริการตรวจสอบรหัสผ่านแบบใช้ครั้งเดียวจะต้องใช้โพรโทคอล ยูดีพี ดังนั้นค่าของ family type protocol จึงเป็น AF_INET SOCK_DGRAM และ IPPROTO_UDP ตามลำดับ

4.3.3.2 ซิสเต็มคอล bind ใช้ในการกำหนดชื่อให้กับ socket

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

4.3.4 ส่งและรับข้อมูล สามารถทำได้โดยใช้ซิสเต็มคอล sendto และ recvfrom

4.3.4.1 ซิสเต็มคอล sendto ใช้ในการส่งกลุ่มข้อมูล

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

4.3.4.2 ซิสเต็มคอล recvfrom ใช้ในการรับกลุ่มข้อมูล

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

4.3.4.3 ซิสเต็มคอล alarm ใช้ในการขัดจังหวะหาก ซิสเต็มคอล recvfrom ยังไม่ได้รับข้อมูลจาก ส่วนให้บริการตรวจสอบรหัสผ่านแบบใช้ครั้งเดียว

```
unsigned int alarm(unsigned int sec);
```

ในบทนี้ได้กล่าวถึงการพัฒนาโปรแกรม ในบทต่อไปจะกล่าวถึงการทดสอบโปรแกรมที่ได้พัฒนาแล้ว