Linux Shadow Password HOWTO

Michael H. Jackson, mhjack@tscnet.com. v1.3, 3 April 1996

번역: 조용일, 번역일: 1997년 2월 1일

이 문서는 Linux Passwd *Shadow Suite*를 어떻게 얻고, 설치하고, 초기화하는 방법을 설명하고 있다. 또한 user password를 필요로 하는 network daemon이나 software를 얻고, 재설치하는 것도 다루고 있다. 그런 software는 *Shadow Suite*의 일부가 아니지만, Shadow Suite를 지원하도록 재 compile할 필요가 있다. 글중에는 program에 shadow를 지원하는 programming example도 있다. 자주 묻는 질문들에 대한 답이 글 말미에 있다.

목 차

제 1 절 들어가면서.

이것은 Linux Shadow-Password-HOWTO이다. 이 글은 Linux system에서 shadow password가 왜 지원됐고, 어떤 식으로 지원하는지 기술하고 있다. *Shadow Suite*의 몇몇 기능을 어떻게 쓰는가에 대한 예제들도 포함하고 있다.

*Shadow Suite*를 설치하고, 많은 utility들을 사용할 때, 반드시 root로 login해야 한다. *Shadow Suite*를 설치할 때, system software에 변화를 주어야 할 것이다. 그리고, 지시하는 대로 program의 backup 복사본을 만들어 놓기를 강력히 권고한다. 또한, 시작하기 전에 안내서를 읽고 이해하기를 권한다.

1.1 이전 글과 다른 것들.

```
덧 붙여진 것들:
        shadow를 왜 설치하지 원하지 않는 이유에 대한 sub-section
        xdm을 update하는 것에 대한 sub-section
        작업에 Shadow Suite의 기능을 추가하는 방법에 대한 section
        자주 묻는 질문에 대한 section

수정 및 update된 것들:
        Sunsite의 html 참조를 정정
        Makefile에 -lshadow를 덧붙이도록 wu-ftp에 대한 section의 정정
        철자와 수다스러움을 정정
        ELF를 지원하도록 wu-ftpd에 대한 section 변경
        여러가지 login program의 보안 문제를 반영하도록 update
        Marek Michalkiewicz의 Linux Shadow Suite를 권하도록 update
```

1.2 이 문서의 최근 것은...

이 문서의 최근판은 anonymous FTP인

sunsite.unc.edu

/pub/Linux/docs/HOWTO/Shadow-Password-HOWTO

또는:

/pub/Linux/docs/HOWTO/other-formats/Shadow-Password-HOWTO{-html.tar,ps,dvi}.gz

에 있으며, 또는 *Linux Documentation Project Web Server* <http://sunsite.unc.edu/mdw/linux.html>를 통해서, *Shadow-Password-HOWTO* <http://sunsite.unc.edu/linux/HOWTO/Shadow-Password-HOWTO.html>, 또는 나(<mhjack@tscnet.com>)에게 직접 얻을 수 있다. 이 것은 newsgroup: comp.os.linux.answers에 항상 게시된다.

이 문서는 Shadow-YYDDMM package에 포함된다.

1.3 Feedback.

나(MichaelH.Jackson$<$mhjack@tscnet.com{$>$})에게 어떤 의견, 새로운 것, 제안을 보내주기 바란다. 내가 빨리 그러한 것들을 받을수록, 이 문서를 빨리 최신 정보를 담고, 잘못을 바로 잡을 수 있다. 어떤 문제가 있을 경우에는 나에게 직접 전해주기 바란다. 왜냐하면 내가 newsgroup에 매일 올라가지 않기 때문이다.

제 2 절 왜 passwd file을 숨겨야 하는가?

기본적으로, 대부분의 Linux 배포본들은 준비된 *Shadow Suite*를 포함하지 않는다. Slackware 2.3, Slackware 3.0, 다른 잘 알려진 배포본들이 그러하다. 이렇게 하는 이유중 하나는 원래의 *Shadow Suite*가 돈을 받고 재배포할 경우에 대한 저작권이 명확하지 않기 때문이다. Linux는 사용하기 편하게 포장(CD-ROM 배포처럼)한 뒤, 그에 대한 댓가로 돈을 받는 것을 허용하는 GNU의 저작권(Copyleft라고 불리우기도 한다)를 사용한다.

지금 *Shadow Suite*를 관리하는 MarekMichalkiewicz$<$marekm@i17linuxb.ists.pwr.wroc.pl{$>$}씨는 재배포를 허용하는 BSD의 저작권을 따르는 저자로부터 source code를 받았다. 따라서, 지금은 저작권 문제는 해결되어서, 이후에 나오는 배포본에는 password에 shadow가 기본으로 사용될 수 있을 것이다. 그때까지는 당신 스스로 설치해야 된다.

CD-ROM으로부터 배포본을 설치했다면, 비록 배포본이 *Shadow Suite*를 설치하지 않았더라도, CD-ROM에서 *Shadow Suite*를 원하는 몇몇 file들이 있을 것이다.

어쨌든, Shadow Suite 3.3.1, 3.3.1-2, shadow-mk는 login program과 suid root를 쓰는 program에 보안 허점이 있고, 더 이상 쓰지 말아야 한다.

모든 필요한 file들은 anonymous FTP나 WWW을 통해서 얻을 수 있다.

*Shadow Suite*를 깔지 않은 Linux system에서는, password를 포함한 사용자 정보는 /etc/passwd에 보관되어 있다. password는 *암호화되어서 (encrypted)* 저장된다. 만일 암호학의 전문가에게 묻는다면, 그는 password는 *encrypt*된 형식이라기 보다는 *encode*된 형식으로 되어 있다. 이유는 crypt(3)을 적용할 때, text는 null로 하고 password를 key로 사용하기 때문이라고 한다. 따라서 이 문서에서는 *encode*된이라는 말을 쓸 것이다. (역자주 : 사전에는 encode와 encrypt를 같은 뜻으로 사용하고 있읍니다. - 암호로 바꿔쓰다 - 라는 뜻입니다만, 암호학을 전공하신 분들에게는 뉘앙스가 다를 것같습니다. 이에 대한 보충 바랍니다.)

password를 encode하는 데 사용되는 algorithm은 기술적으로는 *단방향 hash function*과 같은 방법으로 간주되고 있다. 이 것은 순방향으로는 계산하기 편하게 되어 있지만 역방향은 연산이 매우 힘들게 되어 있다. 사용된 algorithm에 대한 자세한 설명은 section 2.4나 crypt(3) manual page에 있다.

사용자가 password를 선택하거나 할당받을 때, password는 *salt*(소금?)라고 불리는 무작위로 생성된 값과 같이 encode된다. 이것은 어떤 password든지 4096가지의 다른 방법으로 저장될 수 있다라는 얘기다. *salt* 값은 encode된 password와 같이 저장된다.

사용자가 login하고 password를 사용하면, *salt*는 encode되어 저장된 password에서 뽑혀져 나온다. 그다음 입력된 password와 *salt*가 같이 *encode*된다. 그리고, *encode되어* 저장된 password와 비교한다. 그 결과, 서로 같다면 사용자는 인증된다.

무질서하게 *encode된* password를 획득해서 원래의 password로 되돌리는 것은 계산상으로는 힘들다(그러나 불가능하지는 않다). 그러나, 적지 않은 사용자가 사용하는 system이라면, 적어도 몇몇 password는 일상단어로 이루어져 있다 (또는 간단한 변종이다).

system cracker들는 이런 것을 알고, 자주 쓰이는 password들과 단어의 사전과 가능한 4096가지 *salt* 값을 사용해서 encrypt을 행할 것이다. 그다음에 그들은 그들의 database에 있는 당신의 `/etc/passwd` file의 encode된 password와 비교할 것이다. 일단 하나라도 일치한다면 그들은 또다른 계정의 password를 가지게 되는 셈이다. 이는 *dictionary attack*(사전 공격?)이라 고 불리우고, system에 허가되지 않은 접속을 얻을 때 쓰는 가장 보편적인 방법중 하나이다.

생각해보라, 8문자된 password가 4096 * 13문자열로 encode된다. 그리고, 400,000개의 일반 단어, 이름, password, 약간의 변형들로 이루어진 사전은 4G Byte hard를 쉽게 채울 것이다. 공격자들은 이런 종류의 것이 필요하고, 맞는 지 검사해 볼 필요가 있다. 만일 10000 달러이하로 이런 4G byte짜리 hard를 가질 수 있다면, 대부분의 system cracker들에게는 충분하다.

또한, cracker가 당신의 `/etc/passwd` file을 이미 가지고 있다면, 그들은 `/etc/passwd` file에 포함되어 있는 `salt` 값만 가지고 사전을 encode하면 된다. 이 방법은 200 Megabyte의 공간과 486급 computer를 가지고 있는 보통 청소년이면 이용할 수 있다.

심지어 많은 공간없이, crack(1)과 같은 utility들은 최소한 충분히 많은 사용자를 확보하고 있는 system의 password를 2개정도는 깰 수 있다 (user가 자기 자신의 password를 고를 수 있는 system이라고 한다면).

`/etc/passwd` file은 user ID와 group ID와 같은 대부분의 system program에서 쓰는 정보를 가지고 있다. 게다가 `/etc/passwd` file은 "모두 읽기 가능"으로 남아 *있어야 한다*. `/etc/passwd` file을 아무도 보지 못하게 하면, 제일 먼저 `ls -l` 명령이 이제 user 이름대신 user ID를 출력하는 것을 보게 될 것이다!

*Shadow Suite*는 password를 다른 file(대개 `/etc/shadow`)에 위치시킴으로써 이 문제를 해결한다. `/etc/shadow` file은 어느 누구도 볼 수 없도록 되어 있다. *root*만이 `/etc/shadow`를 볼 수 있고, 쓸 수 있다. 어떤 program (xlock 같은)은 password를 바꿀 수 있는 권리를 원하지 않는다. password를 확인할 수 있으면 된다. 이런 program들은 *suid root*로 실행되거나, `/etc/shadow`를 읽기만 할 수 있는 *shadow*로 group을 바꾸어 주면 된다. 그러면 program은 *sgid shadow*로 실행시킬 수 있다.

password를 `/etc/shadow` file로 옮겨 줌으로써, *dictionary attack*를 하기 위해서 encode된 password들에 접근하는 공격자들은 효과적으로 방해할 수 있다.

추가적으로 *Shadow Suite*는 몇가지 괜찮은 기능을 더 가지고 있다:

- login 기본사항(`/etc/login.defs`)들이 준비된 configuration file
- user 계정 및 group을 추가, 수정, 삭제하는 utility들
- password의 유효기간 설정과 경과후 취소
- 계정 무효와 동결
- group password들의 shadow (선택사항)
- 2배 길이를 가지는 passwrd (16문자 password) (권하지 않음)
- user가 password를 고를 때, 적절한 통제
- 전화접속용 password
- 보조 인증 program (권하지 않음)

*Shadow Suite*를 설치하는 것은 좀 더 보안이 강화된 system으로 만들어 준다. 그러나, Linux system의 보안을 강화시켜주는 다른 많은 것들이 있고, 따라서 궁극적으로 다른 보안 도구나 관련된 사안을 다루는 Linux Security HOWTO series가 생길 것이다.

알려진 취약점을 포함한 Linux 보안 문제에 대한 정보를 얻으려면 *Linux Security home page* `<http://bach.cis.temple.edu/linux/linux-security/>`를 방문하기 바란다.

2.1 passwd file을 은폐하기를 주저합니까?

다음과 같은 환경들에서는, *Shadow Suite*가 좋은 대안이 될 수 *없다*:

- system이 사용자 계정을 가지고 있지 않다.
- 당신의 system이 LAN위에서 운영되고 있고, network상의 다른 기계에 사용자 이름과 password를 얻기 위해서 NIS(Network Information Services)를 사용한다. (이 것만으로 잘 운영되고 있고, - 그 이상은 이 문서의 범위를 넘는다 - 보안을 그다지 많이 강화시키기를 원하지 않는다.)
- 당신의 기계가 NFS(Network File System), NIS 또는 다른 방법을 통해 사용자를 확인하기 위한 terminal server로 사용되고 있다.
- 사용자를 확인하는 다른 software를 사용하고 있고, 쓸 수 있는 shadow version이 없다. 그리고, source code도 갖고 있지 않다.

2.2 /etc/passwd file의 형식

shadow의 세례를 받지 않은 `/etc/passwd` file은 다음과 같이 구성되어 있다.

```
username:passwd:UID:GID:full_name:directory:shell
```

각요소는:

`username`
  사용자 (login) 이름

`passwd`
  encode된 password

`UID`
  숫자로 된 user ID

`GID`
  숫자로 된 기본 group ID

`full_name`
  user의 실제 이름 - 실지로 이 field는 GECOS (General Electric Comprehensive Operating System: 일반 전자적 종합 운영 체계?) field라고 불리우며, 단지 실제 이름보다는 다른 정보를 가질 수 있다. Shadow 명령들과 manual page는 이 field를 comment로 다룬다.

`directory`
  사용자의 home directory (Full pathname)

`shell`
  사용자의 login shell (Full pathname)

예를 들면:

```
username:Npge08pfz4wuk:503:100:Full Name:/home/username:/bin/sh
```

`Np`는 salt이며, `ge08pfz4wuk`는 *encode된* password이다. encode된 salt/password는 `kbeMVnZM0oL7I`가 될 수도 있고, 둘은 같은 password를 가리킨다. 같은 password에 대해서 4096개의 다른 encoding이 존재할 수 있다. (예를 든 password는 'password'이며, 상당히 *나쁜* password이다).

shadow suite가 설치되면, `/etc/passwd` file은 다음처럼 바뀐다:

```
username:x:503:100:Full Name:/home/username:/bin/sh
```

두번째 field의 `x`는 아무 것도 아니다. (공간만 차지하고 있을 뿐이다.) `/etc/passwd` file의 형식은 전혀 바뀌지 않았다. 단지 *encode된* password를 포함하지 않을 뿐이다. 이는 `/etc/passwd` file을 읽기만 할 뿐 password를 검사하지 않은 program은 아무 이상없이 돌아간다는 것을 의미한다.

이제 password가 shadow file(대부분 `/etc/shadow` file)로 재배치된다.

2.3 shadow file의 형식

`/etc/shadow` file은 다음과 같은 정보를 갖고 있다:

```
username:passwd:last:may:must:warn:expire:disable:reserved
```

각 요소는:

`username`
: 사용자 이름

`passwd`
: encode된 password

`last`
: 최근의 password를 바꾼 날 (1970, 1, 1일부터 계산한 날수)

`may`
: password를 바꾼 다음, 또 바꾸기 위해 기다리는 날수 (다음 password로의 변경 유예기간)

`must`
: 다음 password로 바꿀어야 할 때까지의 기간 (현 password 유효기간)

`warn`
: password가 만료되기 전에 user에게 바꿀 것을 경고하는 기간

`expire`
: password가 만료된 뒤, user 계정 사용이 불가능하기까지 기간

`disable`
: 계정이 사용 불가능하게 된 날(1970, 1, 1일부터 계산한 날수)

`reserved`
: 남겨둠

전의 예제 경우 다음과 같다:

```
username:Npge08pfz4wuk:9479:0:10000::::
```

2.4 crypt(3)에 대해서.

crypt(3) manual 페이지에 의하면:

"*crypt*는 password를 encrypt하는 함수이다. 이는 *Data Encryption Standard* algorithm를 기반으로, (무엇보다) key를 찾는 기계적인 방법이 이용되기 힘들도록 약간의 변형이 가해져 있다.

key는 사용자가 입력한 password이다. [encode되는 string을 전부 NULL이다.]

*salt*은 [a-zA-Z0-9./]로 이루어진 집합으로부터 고른 두문자로 이루어진 문자열이다. 이 문자열은 4096개 경우중의 하나로 algorithm이 혼란스럽게 보이려는 목적으로 쓰인다.

key의 각 문자의 하위 7 bit을 취함으로써, 56-bit key가 주어진다. 이 56-bit key는 일정한 문자열을, 반복해서 encrypt하는 데 쓰인다. 결과는 13개 ASCII 문자열로, encrypt된 password를 가리킨다 (처음 두개 문자는 salt 그 자신이다). 결과값은 매번 호출될 때마다 다시 쓰이는 고정된 data를 가리킨다.

경고: key space는 2**56, 즉 7.2e16 가능한 값으로 이루어져 있다. key space를 샅샅이 뒤지는 것은 거대한 병렬 computer를 사용하면 **가능할 것이다**. `crack(1)`와 같은, 대부분의 사람들이 password로 삼는 key space의 특정 부분을 찾는 software가 있다. 따라서, 최소한 password를 선택할 때, 자주 쓰이는 단어나 이름은 피하기 바란다. `passwd` program을 사용하여, 찾기 쉬운 password를 선택하는 지 검사하기를 바란다.

DES algorithm, 그 자체는 가끔 `crypt(3)` interface를 사용하는 것이 다른 password 인증을 위한 어떤 것보다 더 나쁜 선택으로 만들어 버리는 경향이 있다. 보안 강화를 위해서 `crypt(3)`를 사용하려고 한다면, DES만 사용하지 마라: encryption에 대한 좋은 책과 널리 쓰이는 DES library들을 구하라."

(역자주 : 원문은 The DES algorithm itself has a few quirks which make the use of the `crypt(3)` interface a very poor choice for anything other than password authentication. If you are planning on using the `crypt(3)` interface for a cryptography project, don't do it: get a good book on encryption and one of the widely available DES libraries." 입니다. 그중에서 don't do it: get ...부분이 매우 애매합니다. it이 무얼 가리키는 건지 명확하지 않습니다. 일단, get ...을 권유하는 것으로 추측하고 번역을 했는 데...)

대부분 *Shadow Suite*들은 password의 길이를 16문자로 늘이는 code를 포함한다. `des`의 전문가들은 이를 권하지는 않는다. 왜냐하면 전반부를 encoding한 뒤, 긴 password의 후반부를 encoding하는 단순한 방법이기 때문이다. `crypt`의 방식대로라면, 긴 password를 사용하지 않는 것보다 *더 취약한* password를 만들 수 있다. 더우기, 사용자가 16문자나 되는 password를 기억하기 힘들다는 측면도 있다.

`crypt` 방법과 호환성을 지니면서, 긴 password를 지원하고 더 강화된 인증(특히, MD5 algorithm)을 할 수 있는 방법이 연구중이다.

encryption에 대한 책으로 다음을 권한다:

```
"Applied Cryptography: Protocols, Algorithms, and Source Code in C"
by Bruce Schneier <schneier@chinet.com>
ISBN: 0-471-59756-2
```

제 3 절 Shadow Suite 얻기.

3.1 Linux용 Shadow Suite의 역사

이 SECTION에서 소개하는 PACKAGE를 사용하지 말라. 문제점이 발견되었다

최초로 *Shadow Suite*를 만든 사람은 `John F. Haugh II`이다.

Linux system에서 사용되는 것으로는 다음과 같은 것들이 있다.

- `shadow-3.3.1`가 원본이다.

- shadow-3.3.1-2는 FlorianLaRoche$<$flla@stud.uni-sb.de{$>$}씨에 의해서 Linux에 맞게 고쳐졌고, 좀 더 나아진 것이 있다.
- shadow-mk는 Linux에 맞추어 구성되어 있다.

shadow-mk package는 shadow-3.3.1-2 patch가 적용된, John F. Haugh II씨에 의해 배포된 shadow-3.3.1 package를 포함하고 있다. 거기에 좀 더 설치가 쉽게 MohanKokal$<$magnus@texas.net{$>$}씨께서 조금 고치고, Joseph R.M. Zbiciak씨께서 /bin/login의 -f, -h 보안 구멍을 제거한 login1.c (login.secure)가 덧 붙여지고, 몇몇 다른 잡다한 patch가 적용되어 있다.

shadow.mk package는 현재 login program에 *보안상 허점*가 있어 조만간 대체될 것이다.

Shadow 3.3.1, 3.3.1-2, shadow-mk는 login program에 *보안상 허점*이 있다. 이 login bug는 login name의 길이를 검사하지 않는 것을 포함하고 있다. 이 것은 충돌 또는 더 나쁜 것을 유발시키는 buffer overflow를 발생시킨다. 이 buffer overflow가, 이 bug와 함께 shared library를 사용하는 system에서 어떤 사용자에게 *root* 권한을 준다는 소문이 있어 왔다. 나는 어떻게 이런 일이 가능한지 구체적으로 거론하지 않겠다. 그 이유는 이런 (bug가 있는) *Shadow Suite*를 설치해서 피해를 입을 수 있는 Linux system이 많고, *Shadow Suite*이 없는 ELF-이전 배포판에게도 위험하기 때문이다.

이 문제와 다른 Linux 보안관련 문제에 대해 더 자세히 알고 싶다면, *Linux Security home page (Shared Libraries and login Program Vulnerability)* <http://bach.cis.temple.edu/linux/linux-security/Linux-Security-FAQ/Linux-telnetd.html>를 참조하라.

3.2 어디서 Shadow Suite를 얻습니까?

권할만한 *Shadow Suite*은 아직 BETA testing중이다. 어쨌든 최근 version이 안전하며, 취약한 login program을 포함하지 않는다.

package는 다음과 같은 명명규칙을 갖는다:

```
shadow-YYMMDD.tar.gz
```

YYMMDD는 Suite가 발표된 날짜이다.

이 version은 Beta testing이 끝나면, 결국 *Version 3.3.3*이 될것이고, MarekMichalkiewicz$<$marekm@i17linuxb.ists.pwr.wroc.pl{$>$}에 의해서 유지보수 되고 있다. *shadow-current.tar.gz* <ftp://i17linuxb.ists.pwr.wroc.pl/pub/linux/shadow/shadow-current.tar.gz>에서 얻을 수 있다.

또한, 다음에 나오는 mirror site들에서 얻을 수 있다:

- ftp://ftp.icm.edu.pl/pub/Linux/shadow/shadow-current.tar.gz
- ftp://iguana.hut.fi/pub/linux/shadow/shadow-current.tar.gz
- ftp://ftp.cin.net/usr/ggallag/shadow/shadow-current.tar.gz
- ftp://ftp.netural.com/pub/linux/shadow/shadow-current.tar.gz

현재 나와있는 version을 사용하기 바란다.

shadow-960129보다 *이전에 나온* version을 쓰지 말기 바란다: 앞에서 논의한 login 보안 허점이 있다.

이 문서에서 *Shadow Suite*라고 말하는 것은 이 version을 가리킨다. 또한, 당신이 사용하고 있는 package라고 가정한다.

참고적으로, 설치 안내서를 작성하는 데, shadow-960129를 사용했다.

이전에 shadow-mk를 사용했다면, 이 version으로 upgrade를 하고, 이전에 compile했던 것을 다시 하기 바란다.

3.3 Shadow Suite에는 뭔가 있는 것같은 데...

*Shadow Suite*는 다음 program의 대체품을 가지고 있다:

`su, login, passwd, newgrp, chfn, chsh, id`

또한, 새로운 program들도 있다:

`chage, newusers, dpasswd, gpasswd, useradd, userdel, usermod, groupadd, groupdel, groupmod, groups, pwck, grpck, lastlog, pwconv, pwunconv`

덧붙여, library: `libshadow.a`가 사용자 password에 접근하는 program을 작성하거나 compile하기 위해 포함되어 있다.

또한, program들을 위한 manual page도 있다.

`/etc/login.defs`로 설치되는 login program의 설정 file도 있다.

제 4 절 program들 만들기.

4.1 압축풀기.

package를 받을 뒤 처음 할 일은 풀어 제치는 것이다. package는 gzip으로 앞축된 tar (tape archive) 형식으로 되어 있으므로, `/usr/src`로 옮긴 뒤:

```
tar -xzvf shadow-current.tar.gz
```

그러면, `/usr/src/shadown-YYMMDD`라는 directory에 풀릴 것이다.

4.2 config.h file을 가지고 설정합니다.

첫째, `Makefile`과 `config.h`를 복사한다:

```
cd /usr/src/shadow-YYMMDD
cp Makefile.linux Makefile
cp config.h.linux config.h
```

그리고 `config.h`를 보라. 이 file은 몇몇 설정 사항에 대한 정의를 담고 있다. 만일 *권고한* package를 가지고 있다면, 일단 group shadow 지원을 사용하지 않도록 하기를 권한다.

기본으로, shadow된 group password들을 사용할 수 있다. 이를 바꾸기 위해서 `config.h`의 `#define SHADOWGRP`를 `#undef SHADOWGRP`로 바꾼다. 나는 그들을 사용하지 않고 시작할 것을 원한다. 나중에 절실히 group password와 group 관리자를 원한다면, 다시 사용가능하도록 한 뒤 재compile하면 된다. 만일 사용가능으로 남겨둔다면, *반드시* `/etc/gshadow` file을 만들어야 한다.

긴 password 사용을 하는 것은 앞에서 얘기한대로 권하지 않는다.

`#undef AUTOSHADOW`라고 한 것을 바꾸지 *말라*.

`AUTOSHADOW` 선택사항은 shadow를 무시하는 program들이 계속 작동하도록 하려는 목적으로 준비된 것이었다. 이 이야기는 이론상으로는 괜찮지만, 제대로 기능하지 않는다. 이 option을 허용하고 root로써 program을 실행시키면, 그 놈은 `getpwnam()`를 root권한으로 부르고, 후에 `/etc/passwd` file에 수정된 내용을 다시 쓰게 된다 (*더이상 shadow되지 않은* 채로). 그런 program에는 chfn과 chsh가 있다. (`getpwnam()`를 호출하기 전에, 실제 uid와 유효 uid를 바꾼다해도 이를 회피할 수 없다. 왜냐하면 root도 chfn과 chsh를 사용할 것이기때문이다. (역자주: 모호하네요. system programming에 관한 내용같는 데... 아시는 분의 설명 바랍니다.))

libc를 만들 때도 같은 경우가 있다. SHADOW_COMPAT option이 그 것이다. 그 것은 쓰면 *안 된다!* /etc/passwd로부터 encode된 password를 얻기 시작한다는 것이 문제다.

지금 사용하고 있는 libc version이 4.6.27이전이라면, config.h와 Makefile을 고칠 것이 더 있다. config.h에서 바꿀 것은:

```
#define HAVE_BASENAME
```

을

```
#undef HAVE_BASENAME
```

으로. 그리고 Makefile에서는:

```
SOBJS = smain.o env.o entry.o susetup.o shell.o \
        sub.o mail.o motd.o sulog.o age.o tz.o hushed.o

SSRCS = smain.c env.c entry.c setup.c shell.c \
        pwent.c sub.c mail.c motd.c sulog.c shadow.c age.c pwpack.c rad64.c \
        tz.c hushed.c
```

을

```
SOBJS = smain.o env.o entry.o susetup.o shell.o \
        sub.o mail.o motd.o sulog.o age.o tz.o hushed.o basename.o

SSRCS = smain.c env.c entry.c setup.c shell.c \
        pwent.c sub.c mail.c motd.c sulog.c shadow.c age.c pwpack.c rad64.c \
        tz.c hushed.c basename.c
```

으로. 이 고침은 libc 4.6.27이나 그 이후에 포함된 basename.c에 있는 code를 덧붙인다.

4.3 원 program의 backup copy들을 만든다.

shadow suite가 대체시킬 program들을 추적해서 backup을 만드는 것도 좋은 생각이다. Slackware 3.0에는 다음과 같다:

- /bin/su
- /bin/login
- /usr/bin/passwd
- /usr/bin/newgrp
- /usr/bin/chfn
- /usr/bin/chsh
- /usr/bin/id

BETA package는 Makefile에 *backup을 만들* 목록이 있지만, 다른 배포판에서 다른 위치에 놓여 있을 수 있기에 설명으로 처리되어 있다.

또한 /etc/passwd file을 backup받기를 바란다. 그러나, 같은 directory에 만들 때, passwd 명령으로 덮어 쓰지 못하도록, 이름을 정할 때 조심해라.

4.4 make를 실행

거의 대부분의 설치과정에서 당신이 root 권한을 지닐 필요가 있다.

package를 compile하기 위해 make을 실행시킨다:

```
make all
```

다음과 같은 경고문가 나오는 경우가 있다: `rcsid defined but not used (rcsid가 정의되어 있지만 사용되지 않습니다)`. 괜찮다, 이 건 저자가 version control package를 사용하기에 나오는 것이다.

제 5 절 설치

5.1 일이 잘못돼어 갈 경우를 대비해서 boot disk를 준비합시다.

뭔가 잘못되어 간다면, boot disk를 준비해야 될 것이다. 설치시 boot/root disk를 사용했다면, 그 걸로 충분하다. 그렇지 않다면, *Bootdisk-HOWTO* `<http://sunsite.unc.edu/mdw/HOWTO/Bootdisk-HOWTO.html>`에 booting가능한 disk를 만드는 법이 적혀있으니 참조하라.

5.2 중복된 man page들을 제거하기

또한, 대체될 manual page들을 옮기길 바란다. 심지어 backup없이 Shadow Suite를 설치할 정도로 무모할지더라도, 여전히 옛 manual page들은 제거하기를 원할 것이다. 대개 옛 manual page가 압축되어 보관되어 있으므로, 새 것들은 이전 것에 덮어쓰지 못할 수 있다.

제거 또는 옮길 필요가 있는 manual page를 찾기 위해 `man -aW command`나 `locate command`를 사용할 수 있다. `make install`을 실행시키기 전에 그런 식으로 옛 page들을 찾는 것이 일반적으로 더 쉽다.

Slackware 3.0 배포판을 사용한다면, 제거해야 할 man page들은:

- /usr/man/man1/chfn.1.gz
- /usr/man/man1/chsh.1.gz
- /usr/man/man1/id.1.gz
- /usr/man/man1/login.1.gz
- /usr/man/man1/passwd.1.gz
- /usr/man/man1/su.1.gz
- /usr/man/man5/passwd.5.gz

또, `/var/man/cat[1-9]`의 subdirectory에 삭제해야 할 것과 같은 이름을 지닌 놈들이 있다.

5.3 make install을 실행

이제 준비가 끝났다: (root로서 이 일을 합시다)

```
make install
```

이 일은 새 것을 깔거나, 옛 것과 대체하며 file permission을 고친다. 또, man page도 설치한다.

그리고, `/usr/include/shadow`에 Shadow Suite에 있는 include file을 설치해준다.

BETA package를 쓴다면, 직접 `login.defs`를 `/etc`에 복사하고, *root*만이 이를 바꿀 수 있도록 해주어야 한다.

```
cp login.defs /etc
chmod 700 /etc/login.defs
```

이 file은 *login* program의 설정 file이다. 내용을 다시 보고, 당신의 system에 맞게 고치기 바란다. 이것은 root로 login할 수 있는 tty를 결정하고, 다른 보안 관련 setting을 결정한다(password 취소에 대한 기본값같은).

5.4 pwconv 실행

다음 일은 `pwconv`를 실행시키는 것이다. 반드시 *root*로서 이 일을 해야 될 뿐만 아니라, `/etc` directory에서 하면 금상첨화다:

```
cd /etc
/usr/sbin/pwconv
```

`pwconv`는 `/etc/passwd`와 그 안에서 몇몇 field을 가져와 다음 두 file을 만든다: `/etc/npasswd` 와 `/etc/nshadow`.

`pwunconv` program은 `/etc/passwd`와 `/etc/shadow`로부터 평범한 `/etc/passwd` file을 만들 경우에 대비해 주어진다.

5.5 npasswd와 nshadow의 이름을 바꾼다.

이제 `pwconv`를 실행시켜서 `/etc/npasswd`와 `/etc/nshadow`를 얻었다. 이 것들을 `/etc/passwd`와 `/etc/shadow`로 덮어쓸 필요가 있다. 우리는 원 `/etc/passwd`를 backup을 받기를 원하고, root만 읽을 수 있게 한다. 그리고 backup을 root의 home directory로 옮긴다:

```
cd /etc
cp passwd ~passwd
chmod 600 ~passwd
mv npasswd passwd
mv nshadow shadow
```

file의 소유와 permission에 관한 것을 정확하게 해라. *X-Windows*를 쓸 생각이라면, `xlock`와 `xdm` program은 `shadow` file을 읽을 수 있게 한다 (쓰는 것은 말고).

이 일을 가능하게 하는 방법은 두가지다. `xlock`에 suid root를 설정해 줄 수 있다(`xdm`가 root의 권한으로 실행될 수 있다). 또는 `shadow` file을 `shadow` group의 `root`가 소유한 것으로 만드는 것이다. 그러나 두 번째 제안을 하기 전에 shadow group(`/etc/group`를 보라)이 있는 지 확실히 해라. 현재 system의 어떤 사용자도 shadow group에 속해있으면 안된다.

```
chown root.root passwd
chown root.shadow shadow
chmod 0644 passwd
chmod 0640 shadow
```

이제 system을 shadow된 password file을 가지게 되었다. 다른 가상 terminal을 띄우고, login할 수 있는 지 점검하는 것이 *좋을 것이다*.

지금 해라!

안 돼면, 뭔가 잘못된거다! shadow되지 않은 상태로 돌아가기 위해서 다음처럼 한다:

```
cd /etc
cp ~passwd passwd
chmod 644 passwd
```

그리고 나서, 이전에 있던 장소로 모든 file을 되돌려 놓아야 할 것이다.

제 6 절 patch하거나 upgrade할 필요가 있는 다른 program들

password로 접근을 필요로 하는 대부분 program들의 대치품이 shadow suite에 포함되어 있다고 해도, 대부분 system에서 password 접근을 필요로 하는 다른 program들이 있다.

*Debian 배포판*을 쓰고 있다면 (또는 쓰고 있지 않더라도), ftp://ftp.debian.org/debian/stable/source/로부터 다시 만들어야 될 program들의 Debian source를 얻을 수 있다.

이 section의 나머지 부분은 `adduser`, `wu_ftpd`, `ftpd`, `pop3d`, `xlock`, `xdm`, `sudo`같은 program들이 shadow suite를 지원하도록 upgrade하는 법에 대해 다루고 있다.

shadow suite에 대한 지원을 어떻게 program에 넣는가하는 문제는 section ?? (C program에 Shadow를 지원하도록 덧붙이기)를 보라 (그리고나서 program이 shadow file을 접근할 수 있도록 SUID root나 SGID shadow로 실행해야 되지만)

6.1 Slackware adduser program

Slackware 배포판(다른 것들중에서도)은 `/sbin/adduser`라고 불리우는 사용자를 추가할 때 쓰는 대화식 program을 포함하고 있다. 이 program의 shadow version은 `ftp://sunsite.unc.edu/pub/Linux/system/Admin/accounts/adduser.shadow-1.4.tar.gz`에서 구할 수 있다.

나는 slackware의 `adduser`대신에 *Shadow Suite*에 있는 program들 (`useradd`, `usermod`, `userdel`)을 사용할 것을 권한다. 그들을 익히는 데 다소 시간이 걸리지만, 그만한 값어치를 한다. 왜냐하면 당신은 좀더 자세히 control할 수 있고, `/etc/passwd`와 `/etc/shadow`에 알맞은 file locking을 행해주기 때문이다 (`adduser`는 아니다).

좀 더 자세한 걸 알고 싶으면 7(Shadow Suite 사용하기)를 참조하도록.

하지만, 가지고 있다면 다음처럼 해라:

```
tar -xzvf adduser.shadow-1.4.tar.gz
cd adduser
make clean
make adduser
chmod 700 adduser
cp adduser /sbin
```

6.2 The wu_ftpd Server

대부분 Linux system은 `wu_ftpd` server를 쓰고 있다. 배포판으로부터 shadow를 설치하지 않았다면, 당신의 `wu_ftpd`는 shadow 지원을 하도록 compile되지 않았다. `wu_ftpd`는 *root* process로써 실행되는 `inetd/tcpd`로부터 시작된다. 아직도 낡은 `wu_ftpd` deamon을 쓰고 있다면, 그 놈은 *root* 계정을 위태롭게 하는 bug를 지니고 있기에 무조건 upgrade를 해야 된다 (*Linux security home page* `<http://bach.cis.temple.edu/linux/linux-security/Linux-Security-FAQ/Linux-wu.ftpd-2.4-Update.html>`를 참조).

다행히 source code를 가져와 shadow가 가능하도록 재compile하기만 하면 된다.

쓰고 있는 것이 ELF system이 아니라면, `wu_ftp` server는 sunsite의 *wu-ftp-2.4-fixed.tar.gz* `<ftp://sunsite.unc.edu/pub/Linux/system/Network/file-transfer/wu-ftpd-2.4-fixed.tar.gz>`를 쓰면 된다.

일단 가져와서 /usr/src에 놓은 뒤:

```
cd /usr/src
tar -xzvf wu-ftpd-2.4-fixed.tar.gz
cd wu-ftpd-2.4-fixed
cp ./src/config/config.lnx.shadow ./src/config/config.lnx
```

그런다음 ./src/makefiles/Makefile.lnx을 수정한다:

```
LIBES    = -lbsd -support
```

를:

```
LIBES    = -lbsd -support -lshadow
```

으로.

이제 script를 만들고 설치하기 위한 준비가 끝났다:

```
cd /usr/src/wu-ftpd-2.4-fixed
/usr/src/wu-ftp-2.4.fixed/build lnx
cp /usr/sbin/wu.ftpd /usr/sbin/wu.ftpd.old
cp ./bin/ftpd /usr/sbin/wu.ftpd
```

이는 Linux shadow 설정 file을 사용해서 compile하고 server를 설치한다.

내 Slackware 2.3 system에서는 build를 실행시키기 전에 다음과 같은 일을 해야 했다:

```
cd /usr/include/netinet
ln -s in_systm.h in_system.h
cd -
```

ELF system에서 이 package를 compile하는 데 몇가지 문제점들이 보고 되었지만, 다음 release의 Beta version에서는 잘 된다. 그것은 *wu-ftp-2.4.2-beta-10.tar.gz* <ftp://tscnet.com/pub/linux/network/ftp/wu-ftpd-2.4.2-beta-10.tar.gz>이다.

일단 가져와서 /usr/src에 놓은 뒤:

```
cd /usr/src
tar -xzvf wu-ftpd-2.4.2-beta-9.tar.gz
cd wu-ftpd-beta-9
cd ./src/config
```

그런 뒤 config.lnx를 수정한다:

```
#undef SHADOW.PASSWORD
```

을:

```
#define SHADOW.PASSWORD
```

으로. 그리고

```
cd ../Makefiles
```

Makefile.lnx를 수정한다:

```
LIBES = -lsupport -lbsd # -lshadow
```

를:

```
LIBES = -lsupport -lbsd -lshadow
```

으로. 마지막으로 만들고 설치:

```
cd ..
build lnx
cp /usr/sbin/wu.ftpd /usr/sbin/wu.ftpd.old
cp ./bin/ftpd /usr/sbin/wu.ftpd
```

당신의 wu.ftpd server가 실제로 어디 있는 지 확인하기 위해 `/etc/inetd.conf`를 점검해 보라. 어떤 배포판에서는 server deamon을 다른 장소에 두고, 특히 wu.ftpd는 다른 이름을 하고 있다는 말이 있다.

6.3 표준 ftpd

표준 `ftpd` server를 쓰고 있다면 `wu_ftpd` server를 쓰도록 권한다. 위에서 얘기한 bug외에는 일반적으로 더 안전한 것으로 알려져 있다.

표준을 계속 고수하거나 *NIS*을 지원할 필요가 있다면, Sunsite에 *ftpd-shadow-nis.tgz* `<ftp://sunsite.unc.edu/pub/Linux/system/Network/file-transfer/ftpd-shadow-nis.tgz>`가 있다.

6.4 pop3d (Post Office Protocol 3)

*POP3*가 필요하면, `pop3d` program을 재compile해야 한다. `pop3d`는 `root` 권한으로 `inetd/tcpd`에 의해 실행된다.

Sunsite에 두가지 version이 있다: *pop3d-1.00.4.linux.shadow.tar.gz* `<ftp://sunsite.unc.edu/pub/Linux/system/Mail/pop/pop3d-1.00.4.linux.shadow.tar.gz>` 와 *pop3d+shadow+elf.tar.gz* `<ftp://sunsite.unc.edu/pub/Linux/system/Mail/pop/pop3d+shadow+elf.tar.gz>`

둘 다 설치까지 수월하게 진행된다.

6.5 xlock

shadow suite를 설치하고, *X Windows System*과 upgrade하지 않고 `xlock`으로 screen에 lock을 건다면, `Ctrl-Atl-Fx`를 눌러 다른 *tty*로 login한 뒤 `xlock` process를 죽여 할 것이 틀림없다 (또는 `Ctrl-Alt-BS`로 X server를 죽이던가). 다행히 `xlock` program을 upgrade하는 것은 쉽다.

XFree86 3.x.x를 쓰고 있다면, 아마도 `xlockmore` (lock기능에 훌륭한 screen-saver까지 있는)를 쓰고 있을 것이다. 이 package는 *shadow*와 함께 재compile할 수 있도록 되어 있다. 낡은 `xlock`를 쓰고 있다면, 이 걸로 upgrade하도록 권한다.

`xlockmore-3.7.tgz`는 `<ftp://sunsite.unc.edu/pub/Linux/X11/xutils/screensavers/xlockmore-3.7.tgz>` 에 있다.

대개, 이게 구미에 딱 맞을 것이다.

`xlockmore-3.7.tgz`을 구한 다음, `/usr/src`에 푼다:

```
tar -xzvf xlockmore-3.7.tgz
```

`/usr/X11R6/lib/X11/config/linux.cf` file의 다음 line을 바꾸면 된다:

```
#define HasShadowPasswd    NO
```

를 다음처럼

```
#define HasShadowPasswd    YES
```

그리고 나서 실행 file을 만들자:

```
cd /usr/src/xlockmore
xmkmf
make depend
make
```

마지막으로 모두 제자리에, 바른 permission을 지니게 하면 끝이다:

```
cp xlock /usr/X11R6/bin/
cp XLock /var/X11R6/lib/app-defaults/
chown root.shadow /usr/X11R6/bin/xlock
chmod 2755 /usr/X11R6/bin/xlock
chown root.shadow /etc/shadow
chmod 640 /etc/shadow
```

이제 xlock은 잘 돌아갈 것이다.

6.6 xdm

`xdm`는 X-Windows상에서의 login screen을 보여준다. 어떤 system은 특정 level로 가도록 지정하면 `xdm`을 시동시킨다(`/etc/inittab`를 보도록).

*Shadow Suite*가 설치되면 `xdm`도 update될 필요가 있다. 이는 매우 쉽다.

`xdm.tar.gz`는 `<ftp://sunsite.unc.edu/pub/Linux/X11/xutils/xdm.tar.gz>`에 있다.

`xdm.tar.gz`를 구한 다음, `/usr/src`에서 푼다:

```
tar -xzvf xdm.tar.gz
```

`/usr/X11R6/lib/X11/config/linux.cf`에서 다음 line을 고친다:

```
#define HasShadowPasswd    NO
```

를 다음처럼

```
#define HasShadowPasswd    YES
```

그리고 나서 실행 file을 만들자:

```
cd /usr/src/xdm
xmkmf
make depend
make
```

모든 걸 제자리로...:

```
cp xdm /usr/X11R6/bin/
```

`xdm`은 *root* 권한으로 실행되기에 permission을 바꿀 필요는 없다.

6.7 sudo

sudo는 시스템 관리자가 사용자로 하여금 정상적으로 root 권한을 가지고 program들을 실행할 수 있게 하도록 허용해준다. 이 것은 drive를 mount하는 것과 같은 일을 사용자가 할 수 있도록 허용함으로써, system 관리자가 root 계정으로 접속할 필요를 억제할 수 있다는 면에서 간편하다.

sudo는 실행될 때 사용자 password를 확인하기 때문에 password를 읽을 필요가 있다. sudo는 이미 SUID root상태로 동작되기에 /etc/shadow file에 접근하는 데 문제는 없다.

shadow suite에 맞는 sudo는 <ftp://sunsite.unc.edu/pub/Linux/system/Admin/sudo-1.2-shadow.tgz>에 있다.

경고: sudo를 설치할 때, 기존의 /etc/sudoers는 기본 설정으로 대체된다. 그러므로 기본 설정이외의 것을 쓰고 있다면 backup을 하기 바란다 (또는, Makefile에서 기본 설정 file을 /etc로 복사하도록 지시하는 line을 제거하면 된다).

이 package는 이미 shadow를 쓸 수 있게 설정되어 있으므로, 재compile하기만 하면 된다 (/usr/src에 넣고):

```
cd /usr/src
tar -xzvf sudo-1.2-shadow.tgz
cd sudo-1.2-shadow
make all
make install
```

혹시 이 일을 해낸 사람이 있으면 내게 E-mail을 보내기 바란다. 그러면 나는 이 곳에 해결법을 포함시키겠다.

6.8 pppd (Point-to-Point Protocol Server)

pppd server는 여러가지 방식으로 인증할 수 있게 설정할 수 있다: *Password Authentication Protocol* (PAP)와 *Cryptographic Handshake Authentication Protocol* (CHAP). 대개 pppd server는 /etc/ppp/chap-secrets와/또는 /etc/ppp/pap-secrets에 있는 password를 읽는다. 이런 식으로 pppd를 쓴다면, pppd를 다시 설치할 필요가 없다. (역자주: ppp용 password를 따로 둔다는 뜻인 듯...)

pppd는 *login* parameter를 사용할 수 있다 (command line이건, option file이나 설정을 통해서건). *login* option이 주어지면, pppd는 *PAP*를 위해 /etc/passwd에 있는 username과 password를 사용할 것이다. 물론 이 경우에 shadow된 password file은 쓸모없다. pppd-1.2.1d에서 shadow를 지원하도록 code를 덧붙여야 된다.

다음 section에서 pppd-1.2.1d에 shadow를 지원하도록 하는 예를 보일 것이다 (pppd의 옛 version).

pppd-2.2.0는 이미 shadow가 지원된다.

제 7 절 Shadow Suite 사용하기

이 section은 system에 *Shadow Suite*을 깔고 나서 알고 싶은 몇몇 주제를 다룬다. 더 자세한 것은 각 명령의 manual page를 참조하기 바란다.

7.1 사용자계정 추가, 수정, 삭제

*Shadow Suite*는 사용자 계정을 관리하는, 다음과 같은 명령들을 추가했다. 이미 adduser program은 그전에 설치되어 있었을 것이다.