2 패스워드 관리 및 패스워드 crack

실험 영역	사용자 관리 및 패스워드 크래킹의 이해
실험 제목	패스워드 관리 및 crack
실험 요약	추측하기 쉬운 사용자 패스워드를 설정하고 /etc/shadow 파일을 참조하여 패스워드를 크랙한다.

2.1 이론

2.1.1 사용자 계정 관리 관련 파일

가. 패스워드 파일(/etc/passwd)

시스템상의 사용자들에 대한 정보를 저장하기 위해 /etc/passwd 파일이 사용된다.

```
root:x:0:1:Super-User:/root:/bin/csh
daemon:x:1:1::/:
bin:x:2:2::/usr/bin:
sys:x:3:3::/:
adm:x:4:4:Admin:/var/adm:
lp:x:71:8:Line Printer Admin:/usr/spool/lp:
uucp:x:5:5:uucp Admin:/usr/lib/uucp:
nuucp:x:9:9:uucp Admin:/var/spool/uucppublic:/usr/lib/uucp/uucico
listen:x:37:4:Network Admin:/usr/net/nls:
nobody:x:60001:60001:Nobody:/:
noaccess:x:60002:60002:No Access User:/:
nobody4:x:65534:65534:SunOS 4.x Nobody:/:
ymir:x:100:10::/export/home/ymir:/bin/csh
castle:x:101:10::/export/home/castle:/bin/csh
user001:x:102:100::/export/home/user001:/bin/csh
user002:x:103:100::/export/home/user002:/bin/csh
user003:x:104:100::/export/home/user003:/bin/csh
user004:x:105:100::/export/home/user004:/bin/csh
user005:x:106:100::/export/home/user005:/bin/csh
user006:x:107:100::/export/home/user006:/bin/csh
user007:x:108:100::/export/home/user007:/bin/csh
user008:x:109:100::/export/home/user008:/bin/csh
user009:x:110:100::/export/home/user009:/bin/csh
```

패스워드 파일의 필드는 아래와 같이 구성되어 있다.

o Login Name: 1- 8 자의 고유 로그인명

o Password: 위치지정을 위해 "x"문자가 사용되며 암호화된 패스워드는/etc/shadow 화일에 저장

o User ID: 약 60,000 사이의 고유 식별 번호

o Group ID: 로그인시 사용자에게 배정되는 그룹 번호

o Comments: 사용자 로그인에 대한 요약설명 (최대 256 자 지정)

o Home Directory: 사용자가 로그인한후 위하는 디렉토리

o Shell: 사용자가 로그인할때 자동적으로 실행될 실행화일의 완전한경로명

나. 쉐도우 파일(/etc/shadow)

보안관리를 위해 암호화된 패스워드를 /etc/shadow 파일에 저장하며 이 파일은 수퍼유저만이 읽을수 있다.

```
root:a1qcEaL5wwoeA:6445::::::
daemon:NP:6445::::::
bin:NP:6445::::::
sys:NP:6445::::::
adm:NP:6445::::::
lp:NP:6445::::::
uucp:NP:6445::::::
nuucp:NP:6445::::::
listen:*LK*:::::::
nobody:NP:6445::::::
noaccess:NP:6445::::::
nobody4:NP:6445::::::
ymir:oC96c3TdKKP/s:12276::::::
castle:GoQi.Kg9iaTA.:12276::::::
user001:54EK0RcI0fVbw:12284::::::
user002:31wMgQ6eZHDp.:::::::
user003:NWFwUnSAVaH7c:::::::
user004:rYxzE4zI9u6Uw:::::::
user005:VJgO2Y2nCG1.I:::::::
user006:dRbsvCnpGf9I6:::::::
user007:Luzle35vS2MRU:::::::
user008:18Sftn7Gmg5vM:::::::
user009:x9y5oR6DTDmoI:::::::
```

쉐도우 파일의 필드는 아래와 같이 구성되어 있다.

o Login Name: /etc/passwd 화일의 사용자명과 같아야 한다.

o Encrypted: 사용자 패스워드를 13 문자의 암호문자로 표시 Password

o Last Changed: 1970 년 1 월 1 일부터 패스워드가 수정된 날자까지의 일수

o Minimum: 패스워드가 변경되기전 최소한 경과해야할 일수

o Maximum: 패스워드 변경전에 최대한 사용할수 있는 일수

o Warn: 패스워드 사용 만기일전에 사용자에게 주의 메시지 가 제공되는 일수

o Inactive: 사용자명이 몇일간 사용되지 않은 경우 로그인 사용 금지를 지정하는 일수

o Expire: 로그인을 더이상 사용할수 없는 일자. mm/dd/yy

o Not Used: 현재 사용되지 않음

다. 그룹 파일

사용자 계정은 반드시 한 개 이상의 그룹에 포함되어 있어야 한다. 그룹의 용도는 파일이나 디렉토리에 대한 그룹 퍼미션 설정에 사용된다. 사용자 등록시에 특별히 그룹을 지정하지 않으면 디폴트로 "other" 그룹에 등록되며 /etc/group 파일에 update 되지는 않는다. 대신 /etc/passwd 파일에서 자신의 그룹에 대한 정보를 보관하고 있다.

```
root::0:root
other::1:
bin::2:root,bin,daemon
sys::3:root,bin,sys,adm
adm::4:root,adm,daemon
uucp::5:root,uucp
mail::6:root
tty::7:root,tty,adm
lp::8:root,lp,adm
nuucp::9:root,nuucp
staff::10:
daemon::12:root,daemon
sysadmin::14:
nobody::60001:
noaccess::60002:
nogroup::65534:
student::100:
```

그룹 파일의 필드는 아래와 같이 구성되어 있다.

o Group Name: 13 문자까지 그룹이름 지정

o Encrypted Password: 사용하지 않음

o Group ID: 그룹 고유 식별 번호

o User List: 그룹상의 모든 사용자명

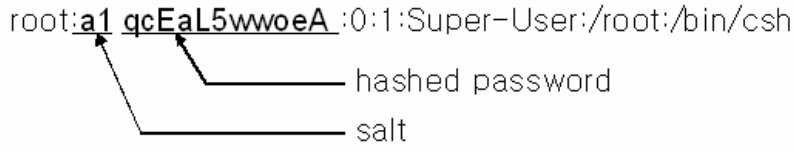
2.1.2 패스워드 생성 및 크래킹 원리

몇몇 운영체제에서 패스워드 파일을 사용자가 이용할 수 없도록 되어 있으나 일반적으로 UNIX 에서는 패스워드 파일을 이용할 수 있도록 설계되었다. 또한 유추하기 쉬운 패스워드의 사용, 불필요한 계정, 그리고 패스워드가 없는 계정 등은 패스워드 관련 취약성으로 알려져있다. 해커들은 일반사용자의 패스워드 파일 이용성과 패스워드 관련 취약성을 이용하여 패스워드를 크래킹하는 반면 보안 취약성을 보완하기 위하여 대부분의 UNIX 에서는 암호화된 패스워드를 일반 사용자가 읽을 수 없도록 숨기기 시작하였다. 예를 들면 SVR4 시스템에서는 /etc/shadow 파일을 사용한다. /etc/passwd 파일에는 일반적인 정보만 가지고 있으며 /etc/shadow 파일에는 암호화된 패스워드를 가지고 있어 일반사용자가 읽지 못하도록 한다.

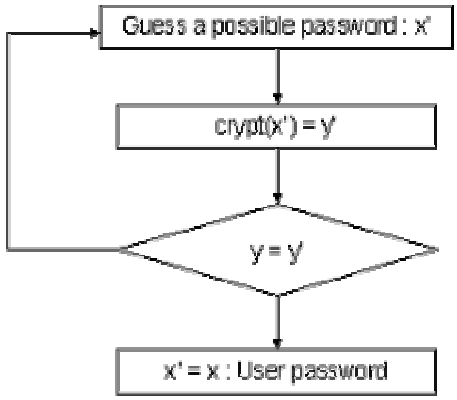
일반적으로 패스워드는 해쉬 형식으로 저장된다. 이 해쉬 함수는 일방향 함수이며 DES(Data Encryption Standard)의 변형된 버전에 기초한다. 일반적으로 패스워드는 "/etc/passwd"에 저장된다. 해쉬를 위한 라이브러리 함수로는 crypt 가 사용자에게 제공된다. 패스워드를 해쉬하는 방법은 다음과 같다.

o “ salt"로 사용하기 위한 랜덤한 12 비트를 획득
  - 주로 시스템 time 을 사용
o 0 비트의 메세지를 암호화하기 위한 키로써 패스워드를 사용
  - DES(Data Encryption Standard)를 변형한 crypt 함수 사용
o 암호화의 결과에 "salt"를 앞에 붙인다.
o 프린트할 수 있는 형식으로 변형

여기서 "salt"는 두개의 다른 시스템에서 동일한 패스워드를 가질 경우 slat 를 사용하므로써 다르게 암호화되어 다른 사람들이 같은 패스워드임을 인식하지 못하도록 한다. 이렇게 생성된 패스워드는 패스워드 파일에 저장되며 형식은 아래 그림과 같다.

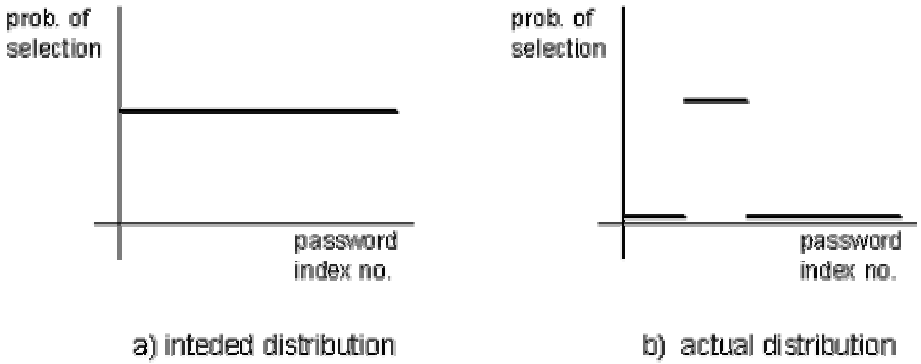


패스워드는 일방향 해쉬함쉬 함수로 생성되었으므로 역으로 계산할 수 없다. 그러나 주어진 해쉬 y 로 crypt(x)=y 의 x 를 발견하기 위하여 사소한 실수나 부주의하게 설정된 패스워드를 사전의 단어로써 공격할 수 있다. 사전의 단어를 이용한 패스워드 공격은 아래 그림과 같다.



x : User password(text)
y : encrypted password in the password file

이러한 패스워드 공격을 위하여 해쉬된 패스워드와 crypt() 함수로의 접근이 가능해야하며 오랜 시간동안 패스워드 추측 알고리즘을 수행할 수 있어야 한다. 일반적으로 표준 UNIX 해쉬 알고리즘의 수행시간은 cray2 의 경우 0.1 초, Sun3/50 의 경우 0.4 초로 상당한 시간을 요구한다. 하지만 일반적인 패스워드의 구성은 아래 그림과 같은 분포를 가짐으로써 취약성을 가진다.



1979 년 Morris 와 Thompson 은 한 사이트에서 패스워드의 83%를 찾아내었으며 1989 년 Klein 은 여러 사이트에서 15,000 개의 패스워드를 찾아냈다. Klein 에 의하여 크래킹된 결과는 아래 표와 같다.

type of password	number		
directory words	8%	length	number
common names	4%	1	0.03%
user/account name	3%	2	0.03%
phrases, patterns	2%	3	0.48%
male names	1%	4	1.36%
female names	1%	5	2.30%
uncommon names	1%	6	8.41%
machine names	1%	7	5.89%
place names	1%	8	5.65%
King James Bible	1%		

2.1.3 패스워드 보안

사용자의 취약한 패스워드 선택에 대한 대안으로 다음과 같은 방법을 사용할 수 있다.

- o 컴퓨터로 랜덤 패스워드 생성
- o 패턴에 따라 컴퓨터로 패스워드를 생성
- o 컴퓨터로 생성된 패스워드 중 하나를 사용자가 선택
- o 사용자가 선택한 패스워드가 안전한지 검증

랜덤한 문자로부터 패스워드를 생성하는 방법은 매우 추측하기 어렵다. UNIX 에서 5-8 글자의 패스워드일 경우, 7,300,000,000,000,000 개의 가능한 패스워드가 있다. 초당 10,000 개의 패스워드를 계산할 경우 패스워드를 알아내려면 평균 11,500 년이 소요된다.) 하지만 이러한 패스워드는 기억하기 힘들다. 또한 패스워드 생성 프로그램이 충분히 랜덤하지 않을 경우 패스워드는 쉽게 찾을 수 있다.

일반적으로 패스워드는 어디에 기록하여 두어서는 안된다. 하지만 다른 사람과 공유하는 20-30 개의 계정을 가지고 있고 모두가 랜덤 패스워드일 경우에 패스워드를 적어둘 필요가 있다. 이럴 경우 패스워드를 적어 놓을 때 몇가지 비밀알고리즘을 적용하는 것이 좋다. 예를들면, "9"가 세번째 문자로 삽입되고 마지막에 더하질 경우 실제 패스워드가 "Hel9Lot9"이면 "Hel9LoT"로 적어둔다.이러한 알고리즘과 패스워드는 자주 바꾸어주는 것이 좋으며 결코 알고리즘을 적어두어서는 안된다.

쉽게 추측할 수 없는 패스워드를 사용자가 선택하도록 하기 위하여 패스워드 선택 프로그램을 사용할 수 있다. 이 경우 패스워드 선택 프로그램은 다음과 같은 기능을 수행하여야 한다.

- o 패스워드를 선택하는 프로그램은 너무 쉽게 추측할 수 있는 패스워드를 허락하지 않는다.
- o 사용자 또는 사이트에 기초하여 패스워드를 구성할 수 있다.
- o 패턴을 검사할 수 있다.

패스워드 선택 프로그램 중 하나가 npasswd 이다. 이것은 Clyde Hoover 로부터 구할 수 있으며 구할 수 있는 곳은 ftp://ftp.cc.utexas.edu/pub/npasswd 이며 다음과 같은 사항을 점검한다.

- o 사전단어
- o case
- o 최소, 최대 길이
- o 프린터할 수 있는 문자들
- o 허용되지 않는 문자

이러한 프로그램 선택 프로그램 중 환경설정이 잘된 것 중의 하나가 passwd+이다. 이것은 Matt Bishop 의 ftp://nob.cs.ucdavis.edu/ pub/security/passwd* .tar 에서 구할 수 있다. passwd+는 다음과 같은 패스워드를 점검한다.

- o 사전단어
- o 로그인 또는 실제 이름
- o case, type 의 혼합
- o 패스워드의 길이
- o 프린트할 수 있거나 허용되지 않는 문자
- o 시스템 관리자에 의하여 정의되는 패턴
- o 사용자마다의 환경설정을 허용

일반적으로 보안에 취약하게 설정된 패스워드는 다음과 같다.

- o 계정명에 기초한 패스워드
- o 사용자명에 기초한 패스워드

- 사전에 있는 패스워드
- 일부 또는 모두 대문자로 쓰여진 사전 단어와 일치하는 패스워드
- 제어문자가 되는 임의의 문자로 사전 단어와 일치하는 패스워드
- 다음과 같은 변화를 가지면서 사전 단어와 일치하는 패스워드

  a→2 또는 4, e→3, h→4, i→1, l→1, o→0, s→$, z→5
- 사전단어의 연결이나 어형변화와 일치되는 패스워드
- 키보드로부터의 패턴과 일치하는 패스워드
- 6 자보다 짧은 패스워드
- 숫자로만 구성된 패스워드
- 숫자와 문자, 문자와 구두점과 같이 혼합된 경우가 아닌 패스워드
- 등록번호표같이 보이지 않도록 한 패스워드
- 두문자어로된 패스워드(DPMA, IFIPTC11, ACM, IEEE, MULTICS 등)
- 과거에 사용되지 않은 패스워드
- 연결단어로된 패스워드
- 예상되는 단어나 숫자나 마침표 또는 스페이스가 따르는 단어로된 패스워드
- 모든 모음이 삭제된 단어로된 패스워드
- 스페이스가 삭제된 단어로된 패스워드

반면 보안성이 높게 잘 설정된 패스워드는 다음과 같은 것들이다.

- 문자와 숫자가 아닌 그자에 의하여 분리된 두 단어의 결합

  예) cat,bear%, mac2#beav
- 워드로 출력되지 않는 글자를 삽입

  예) helo<TAB>gleep

이러한 좋은 패스워드를 설정하는데 필요한 몇가지 규칙은 다음과 같다.

- 패스워드 중간에 <shift>나 제어문자 또는 meta 문자를 사용
- 각 패스워드에 항상 적어도 하나의 문자 숫자식이 아닌 글자가 있어야 한다. 강제로 적어도 하나의 제어문자를 삽입하는 것도 좋다.
- 약간 예외적인면으로 사용되는 단어의 첫문자

예) B/itfotn(William Blake 의 시 "Tyger! Tyger!"의 두번째 라인의 첫문자)

마지막으로 설정된 패스워드에 대하여 Crack 같은 패스워드 크래킹 프로그램을 이용하여 패스워드를 점검하는 것도 좋은 방법이다.

2.1.4 패스워드 크래킹 도구

가. 간단한 패스워드 크래킹 프로그램

여기에서는 간단한 패스워드 크래킹 프로그램을 작성하고자 한다. Sun Solaris 의 경우 패스워드 /etc/shadow 파일에 있다.

먼저 /etc/shadow 파일에서 사용자 명을 읽고 getspnam() 함수를 사용하여 사용자에 대한 쉐도우 파일 정보를 구조체로 얻을 수 있다.

```
#include <shadow.h>
struct spwd *sp;

sp = getspnam(username);
```

shadow 파일 구조체는 shadow.h 에 정의되어 있으며 다음과 같다.

```
struct spwd {
        char *sp_namp;  /* user name */
        char *sp_pwdp;  /* user password */
        int sp_lstchg;  /* password lastchanged date */
        int sp_min;     /* minimum number of days between password changes */
        int sp_max;     /* number of days password is valid */
        int sp_warn;    /* number of days to warn user to change passwd */
        int sp_inact;   /* number of days the login may be inactive */
        int sp_expire;  /* date when the login is no longer valid */
        unsigned int sp_flag;   /* currently not being used */
};
```

패스워드 크래킹을 하기 위하여 사용되는 패스워드 사전은 영문사전, 국어사전, 사용자 ID 를 기반으로 유추할 수 있는 패스워드 등으로 구분된다. 인터넷 등에서 구할 수 있는 영문사전이나 국문사전 중에는 약 5 만개 이상의 단어를 가지고 있는 것도 있다. 본 실습에서는 사용되는 영문사전과 국문사전은 5 만개로 구성된 사전에서 각각 200 개의 단어를 추출한 것이다. 영어사전의 경우 ‘ab’로 시작되는 200 개, 국어사전의 경우 ‘가(rk)’로 시작되는 200 개의 단어로 구성되어 있다.

사용자 ID 를 기반으로 유추할 수 있는 단어는 사용자 ID 와 사용자 ID 앞뒤로 1, 2, 3, 4 를 붙인 패스워드로 구성되어 있으며 프로그램 실행과 동시에 자동 생성되도록 awk 문으로 구성되어 있다.

```
if [ -f /bin/awk ]; then
        AWK=/bin/awk
elif [ -f /usr/bin/awk ]; then
        AWK=/usr/bin/awk
else
        AWK=awk
fi
passwd=/etc/passwd
HOSTNAME=hostname

$AWK -F: '{print $1}' $passwd > idword
$AWK -F: '{print $1"1"}' $passwd >> idword
$AWK -F: '{print $1"2"}' $passwd >> idword
$AWK -F: '{print $1"3"}' $passwd >> idword
$AWK -F: '{print $1"4"}' $passwd >> idword
$AWK -F: '{print "1"$1}' $passwd >> idword
$AWK -F: '{print "2"$1}' $passwd >> idword
$AWK -F: '{print "3"$1}' $passwd >> idword
$AWK -F: '{print "4"$1}' $passwd >> idword
```

패스워드 크래킹은 매우 단순하다. 사전에 있는 단어를 crypt()함수를 통하여 암호화시키고 쉐도우 파일에 있는 암호화된 패스워드와 비교하여 일치하는지는 보면된다.

```
if(!strcmp(crypt(word, sp->sp_pwdp), sp->sp_pwdp)
      printf("User Password = %s\n", word);
```

여기에서 crypt() 함수에 sp->sp_pwdp 를 넣는 이유는 salt 값을 획득하기 위함이다.

본 실습에서는 /TainingToolkit/Crack/SimpleCrack.tar 를 사용자 홈디렉토리로 복사 후 아래와 같이 컴파일하여 실행하면 된다.

① tar 를 푼다.

```
# tar xvf SimpleCrack.tar
```

② SimpleCrack.c 를 캄파일한다.

```
# gcc -o SimpleCrack SimpleCrack.c
```

③ SimpleCrack 을 수행한다.

- 사용자 user001 에 대한 패스워드 크래킹

```
# SimpleCrack user001
```

o 모든 사용자에 대한 패스워드 크래킹

```
# SimpleCrack all
```

나. Crack

Crack 은 시스템에 설정되어 있는 패스워드 파일(/etc/passwd)에 대하여 다음과 같은 점검을 하기위해 사용하는 도구입니다.

- o 패스워드 알아내기
- o 쉬운패스워드 점검하기
- o 취약한 패스워드 점검하기

이 도구는 주로 해커들이 특정시스템의 사용자패스워드를 알아내기 위해 사용되는 것입니다.

도구의 작동 원리는 앞에서 언급한 간단한 패스워드 크래킹 프로그램의 원리와 동일하나 실행시간을 빠르게 하기 위하여 기존의 표준 crypt()를 사용하지 않고 fcrypt()함수 및 ufc-crypt()함수를 이용하여 수행시간이 매우 빠르다. 또한 디스크 점유 공간을 줄이기 위하여 사전을 압축하여 저장하고 크랙이 실행될 때 압축을 풀어서 수행하고, 크랙이 수행중 점유 메모리를 줄이기 위하여 패스워드 파일과 사전을 참조하여 실제 메모리에는 단축된 사전을 사용하여 수행한다.

Crack5.0.tar.Z 를 얻을 수 있는 사이트는 국내에도 여러 곳이 있고 대표적인 사이트는 다음과 같다.

ftp://ftp.cerias.purdue.edu/pub/tools/unix/pwdutils/crack/

본 실습에서는 /TainingToolkit/Crack/crack5.0.tar 에 있는 파일을 사용자 홈디렉토리로 복사 후 실행하면 된다.

① 압축을 푼다.

```
# compress -d crack5.0.tar.Z
or
# gzip -d crack5.0.tar.Z
```

② tar 를 푼다.

```
# tar xvf crack5.0.a.tar
```

③ 필요한 환경을 설정한다.(본 실험에서 사용되는 Crack 프로그램은 Solaris 에 맞게 환경이 설정되어 있음)

o Crack 파일에서 컴파일러를 설정한다.

- src/libdes/Makefile
- src/libdes/Makefile.uni

o cc 또는 gcc 컴파일러에 따라 Makefile 을 조정한다.

④ 컴파일한다.

```
# Crack   makeonly
```

⑤ 패스워드 사전을 만든다.

```
# Crack   makedict
```

⑥ Solaris 의 경우 /etc/shadow 파일을 사용하므로 크래킹할 수 있는 파일로 만들어야 한다. 이 기능을 자동적으로 수행해 주는 것이 script 디렉토리의 shadmrg.sv 명령어이다.

```
# ./scripts/shadmrg.sv > ./shadow
```

⑦ 패스워드 크래킹을 시도한다.

```
# Crack ./shadow
```

⑨ 패스워드 크래킹 보고서를 본다.

```
# Reporter
```

2.2 실험

본 실험에서는 SimpleCrack 프로그램을 이용하여 Crack 의 원리를 이해하고 좀 더 복잡한 공개도구인 Crack5.0 을 실행해본다. 패스워드를 바꿔가며 안전한 패스워드의 조건을 알아본다. 아래 디렉토리에서 실험에 사용되는 프로그램을 사용자 홈디렉토리로 복사하고 su 명령어를 이용하여 root 로 권한 이동 후 프로그램을 실행시켜야 한다.

- /root/Question/Q02/SimpleCrack.tar
- /root/Question/Q02/Crack5.0.a.tar

가. 먼저 SimpleCrack 프로그램을 컴파일하시오. 사용자 환경에서 추측하기 쉬운 패스워드(engword200, korword200, idword 에 있는 단어)를 패스워드로 설정하고 su 를 이용하여 root 로 권한이동 후 SimpleCrack 을 수행하시오.

나. 사용자 환경에서 추측하기 어려운 패스워드로 자신의 패스워드 설정하고 su 를 이용하여 root 로 권한이동 후 SimpleCrack 을 이용하여 다시 크래킹을 시도하시오.