

salfeld.de Exe Password - another bad security program

“Exe Password lets you protect any EXE file with its own password. It does this by storing the password directly in the EXE file. It is a simple matter to add a password to a file: Just select the desired file on the desktop, in the start menu or in Explorer, open the context menu by right-clicking, select Password protection, enter the password, and you’re done. From that time on, a password will be required to run the EXE file.” [quote](#)

costs: 19,90€ (via internet) / 27,90€ (CD)

free 30-day version download: [here](#)

First I created a small program and protected it with “Exe Password”.

Unpacked size: 14Kb

Packed size: 1,6Mb

not bad oO



Part one – extracting the protected file

After looking into the coded with an Hexeditor I found that the original executable isn't really craped. Its simple stored behind the “starter code”

```
00198BE0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198BF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198C00 53 4F 21 23 30 37 30 33 48 32 B6 54 65 6D 70 5F SO!#0703H2Temp_
00198C10 4B 6F 70 69 65 20 28 32 29 20 76 6F 6E 20 74 65 Kopie (2) von te
00198C20 73 74 2E 65 78 65 45 4F 21 23 30 37 30 33 48 32 st.exeEO!#0703H2
00198C30 53 4F 21 23 30 30 30 35 48 33 58 B6 2D 38 38 32 SO!#0005H3Xq-882
00198C40 31 34 38 32 38 38 45 4F 21 23 30 30 30 35 48 33 148288EO!#0005H3
00198C50 58 53 4F 21 23 33 30 30 30 31 31 42 B6 32 39 2E XSO!#300011Bq29.
00198C60 30 33 2E 32 30 30 37 20 31 32 3A 35 36 3A 34 36 03.2007 12:56:46
00198C70 45 4F 21 23 33 30 30 30 31 31 42 53 4F 21 23 30 EO!#300011BSO!#0
00198C80 58 31 33 35 47 4A B6 4A 45 4F 21 23 30 58 31 33 X135GJqJEO!#0X13
00198C90 35 47 4A 53 4F 21 23 41 41 41 53 31 30 31 32 33 5GJSO!#AAAS10123
00198CA0 47 35 B6 4D 5A 50 00 02 00 00 00 04 00 0F 00 FF G5qM2P.....ÿ
00198CB0 FF 00 00 B8 00 00 00 00 00 00 00 00 00 1A 00 00 y.....
00198CC0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198CD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198CE0 01 00 00 BA 10 00 0E 1F B4 09 CD 21 B8 01 4C CD ...°....'í!Li
00198CF0 21 90 90 54 68 69 73 20 70 72 6F 67 72 61 6D 20 !..This program
00198D00 6D 75 73 74 20 62 65 20 72 75 6E 20 75 6E 64 65 must be run unde
00198D10 72 20 57 69 6E 33 32 0D 0A 24 37 00 00 00 00 00 r Win32..$7.....
00198D20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198D30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198D40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198D50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198D60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198D70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198D80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198D90 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00198DA0 00 00 00 50 45 00 00 4C 01 08 00 19 5E 42 2A 00 ...PE..L...^B*.
00198DB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

You can simply extract the code if you know where it starts. The offsets isn't stored inside the executable! The decryption routine searches for string '!#AAAS10123G5' (startcode) and copies the code until the signature 'EO!#AAAS10123G5' (endcode). Then this code is copied to a new executbale which is started by the loader.

What happens if the executable which should be protected has the endcode-signature in their code? I created a small program and tested it myself.

```
program test;
uses
  windows;

const s = 'EO!#AAAS10123G5';

begin
  MessageBoxA(0,'Started',s,0);
end.
```

If you add a password for this program with “Exe Password” it can't be unprotected by it anymore. Everytime you enter the correct password, the unpacked program crashes, cause it isn't fully extracted, cause the decryption routine finds the wrong string.

So its possible thats the “Exe password” destroys the original file, and the file isnt really protected.

Part 2 – creating a new password

The password is also stored into the executable as a checksum:

```
SO!#0703H29[Temp_
Kopie (2) von te
st.exeEO!#0703H2
SO!#0005H3X9[-882
148288EC]!#0005H3 password
XSO!#300011B9[29.
03.2007 12:56:46
EO!#300011BSO!#0
X135GJ9[JEO!#OX13
5GJSO!#AAAS10123
G59[MZP.....ÿ
ÿ.....0....
.....
.....
...°.....'í!„Lí
!..This program
must be run unde
```

After debugging I got the code for creating the password:

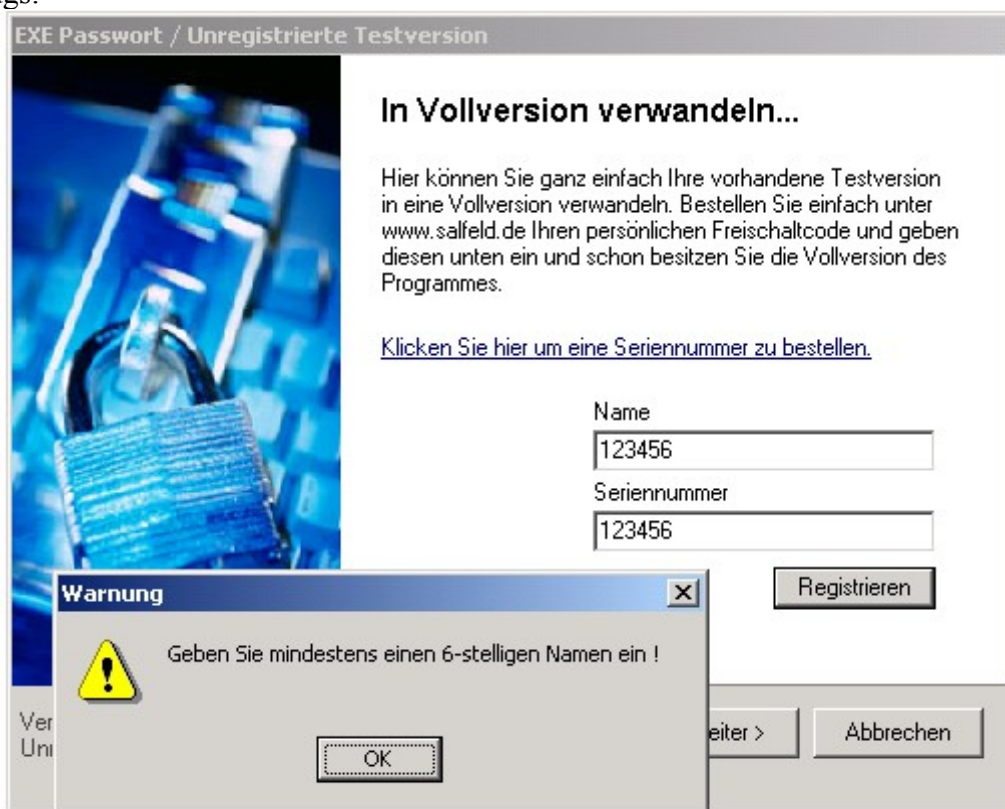
```
function Encode(s: string): Integer;
var
  a: Integer;
  b : Integer;
  i: Integer;
begin
  s := s+#0;
  a := 0;
  for i := 2 to length(s) do
  begin
    b := Ord(s[i])+a*$25;
    a := b;
  end;
  Result := a;
end;
```

As you can see the programmer did a mistake, he doesn't use the whole password (s). He uses the chars from 2..length + #0.

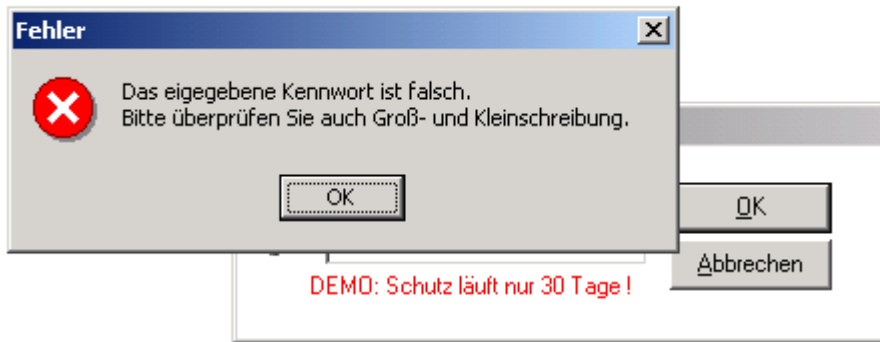
Its a little bit tricky to code an decode function, because of the multiplication with hex 25. Therefore you can get an integer overflow, and the result has lost some import information for creating a decode routine.

Part 3 – conclusion

- Many bugs:



Oops? Maybe we cant count?



Oops? Missing “n”.

- Nice Names for the controls (only 'noob' programmers don't change them):
Tform1, Tpanel, TpageControl, TtabSheet Tedit
- first character of password isn't used
- the protected executable isn't crypted
- it can destroy exe files, which have a special string in it
- password checksum is bypassable
- file too big

AND THIS PROGRAM FOR 19,90€? Sorry I can write a better one in some hours.

Here the full code for unpacking the protected file and creating a valid password:

```

program decodefly;
{$APPTYPE CONSOLE}

uses
  windows, tlHelp32, classes, sysutils, messages;

// the Password encode code
function Encode(s: string): Integer;
var
  a: Integer;
  b : Integer;
  i: Integer;
begin
  s := s+#0;
  a := 0;
  for i := 2 to length(s) do
  begin
    b := Ord(s[i])+a*$25;
    a := b;
  end;
  Result := a;
end;

```

```

var
  r: string =
    'abcdefghijklmnopqrstuvwxyz'+
    'ABCDEFGHIJKLMNOPQRSTUVWXYZ'+
    '01234567890'+
    '!"$%&/'()=?'+
    'ÖÄÜöüä'+
    '+*~#-_:;<>@€';

```

```

function GetPossible(a: int64; var b: int64): Boolean;
begin
  Result := False;
  while ((a mod $25) <> 0) and (a < $2500000000) do
    inc(a, $100000000);
  if (a < $2500000000) then
    Result := True;
  b := (a div $25);
end;

```

```

function decodebrute(a: int64; var s: string; tiefe: integer; maxdepth: integer): Boolean;
var
  i: integer;
  b: int64;
begin
  if (a = 0) then
    begin
      s := "";
      Result := True;
      Exit;
    end;
  if (a > $25) and (a <= $FF) and (Pos(Char(a), r) > 0) then
    begin
      s := Char(a);
      Result := True;
      Exit;
    end;
  if (tiefe > maxdepth) then
    begin
      Result := False;
      Exit;
    end;
  i := 1;
  Result := False;
  while (i <= length(r)) and (not Result) do
    begin
      b := a-ord(r[i]);
      if (b > 0) and (b mod $25 = 0) then
        if decodebrute(b div $25, s, tiefe+1, maxdepth) then
          begin
            s := s+char(r[i]);
            Result := True;
          end;
      inc(i);
    end;
end;

```

```
function decode(a: integer): String;
```

```
var
```

```
  s: String;
```

```
  b: int64;
```

```
begin
```

```
  if GetPossible(a, b) then
```

```
    if decodebrute(b, s, 0, 6) then
```

```
      Result := '?' + s;
```

```
  end;
```

```
function GetEdit: HWND;
```

```
var
```

```
  wnd: HWND;
```

```
begin
```

```
  Result := 0;
```

```
  wnd := FindWindow('TForm1', 'Bitte Kennwort eingeben');
```

```
  wnd := FindWindowEx(wnd, 0, 'TPanel', nil);
```

```
  wnd := FindWindowEx(wnd, 0, 'TPageControl', nil);
```

```
  wnd := FindWindowEx(wnd, 0, 'TTabSheet', nil);
```

```
  wnd := FindWindowEx(wnd, 0, 'TEdit', nil);
```

```
  if wnd <> 0 then
```

```
    Result := wnd;
```

```
  end;
```

```
function GetExecutableFromPID(dwProcessID: DWord): string; stdcall;
```

```
var FSnapshotHandle: THandle;
```

```
  FModuleEntry32 : TModuleEntry32;
```

```
begin
```

```
  Result := '';
```

```
  if (dwProcessID <> 0) then
```

```
    begin
```

```
      FSnapshotHandle := CreateToolhelp32Snapshot(TH32CS_SNAPMODULE, dwProcessID);
```

```
      if (FSnapshotHandle <> 0) then
```

```
        begin
```

```
          FModuleEntry32.dwSize := Sizeof(FModuleEntry32);
```

```
          Module32First(FSnapshotHandle, FModuleEntry32);
```

```
          Result := FModuleEntry32.szExePath;
```

```
          CloseHandle(FSnapshotHandle);
```

```
        end;
```

```
    end;
```

```
end;
```

```

function ExtractEncodedPassword(ExeName: String): integer;
var
  fm: TFileStream;
  f: string;
  i: integer;
  begx, endx: integer;
begin
  Result := 0;
  fm := TFileStream.Create(ExeName, fmOpenRead);
  SetLength(f, fm.Size);
  fm.Read(f[1],fm.size);
  fm.free;
  begx := 0;
  endx := 0;
  for i := 1 to length(f) do
  begin
    if Copy(f, i, length('!#0005H3X¶')) = '!#0005H3X¶' then
      begx := i+10;
    if Copy(f, i, length('EO!#0005H3XSO')) = 'EO!#0005H3XSO' then
      endx := i;
    end;
    if (begx <> 0) and (endx <> 0) and (endx > begx) then
      Result := StrToIntDef(Copy(f, begx, endx-begx),0)
  end;

function TrySetEditText(): Boolean;
var
  pid: DWord;
  Name: String;
  pwencoded: integer;
  Edit: Dword;
  pwdecoded: String;
begin
  Result := False;
  Edit := GetEdit;
  if (edit <> 0) then
  begin
    if GetWindowThreadProcessID(Edit,pid) <> 0 then
    begin
      Name := GetExecutableFromPID(pid);
      if (Name <> "") then
      begin
        pwencoded := ExtractEncodedPassword(Name);
        if (pwencoded <> 0) then
        begin
          pwdecoded := decode(pwencoded);
          if (pwdecoded <> "") then
          begin
            SendMessage(Edit,WM_SETTEXT,0,integer(PChar(pwdecoded)));
            Result := True;
          end;
        end;
      end;
    end;
  end;
  end;
end;
end;

```

```

function unpack(fl: String): Boolean;
var
  fm: TFileStream;
  InFile: String;
  i: integer;
begin
  Result := False;
  fm := TFileStream.Create(fl, fmOpenRead);
  setLength(InFile, fm.size);
  fm.Read(InFile[1], fm.Size);
  fm.free;

  for i := 1 to Length(InFile) do
  begin
  if (Copy(Infile, i, 3) = 'AAA') and (Copy(Infile, i+12, 3) = 'MZP') then
  begin
  fm := TFileStream.Create((fl+' extract.exe'), fmCreate);
  fm.Write(InFile[i+12], Length(InFile)-i-11);
  fm.free;
  Result := True;
  end;
  end;
end;

function ShowPw(fl: String): String;
var
  pwencoded: integer;
begin
  Result := "";
  pwencoded := ExtractEncodedPassword(fl);
  if (pwencoded <> 0) then
  Result := decode(pwencoded);
end;

```

```

procedure main;
var
  pw: String;
begin
  Writeln('-----');
  Writeln(' Salfeld.de Exe Depassword ');
  Writeln(' 2007 by uall ');
  Writeln('-----');
  Writeln("");
  Writeln('(1) -up "ExeName": unpacks the original executable');
  Writeln('(2) -pw "ExeName": extracts a password for the executable');
  Writeln('(3) no param: searches for started program and inserts password');
  Writeln("");
  if (paramcount < 2) then
  begin
    write('method (3) used: ');
    if TrySetEditText then
      writeln('successful') else
        writeln('failed');
    end else
    begin
      if (paramstr(1) = '-up') then
      begin
        write('method (1) used: ');
        if FileExists(paramstr(2)) and unpack(paramstr(2)) then
          writeln('successful') else
            writeln('failed');
        end else
        if (paramstr(1) = '-pw') then
        begin
          write('method (2) used: ');
          if FileExists(paramstr(2)) then
          begin
            pw := showpw(paramstr(2));
            if (pw <> "") then
              writeln('successful pw: '+pw) else
                writeln('failed');
            end else
              writeln('failed');
            end else
              writeln('unknown parameter used');
          end;
          ReadLn;
        end;

begin
  Main;
end.

```

[Download it here](#)