How to make key generators?

Introduction
-------------
I take no responsibility of the usage of this information.
This tutorial, is for educational knowledge ONLY.

Hi there, in this tutorial, I intend to teach you how to make a pretty
simple keygen, of a program called W3Filer 32 V1.1.3.
W3Filer is a pretty good web downloader...
I guess some of you might know the program.

I'll assume you know:
A. How to use debugger (in this case, SoftIce).
B. How to crack, generally (finding protection routines, patching them, etc...).
C. How to use Disassembler (This knowledge can help).
D. Assembly.
E. How to code in Turbo Pascal (tm).

Tools you'll need:
A. SoftIce 3.00/01 or newer.
B. WD32Asm. (Not a must).
C. The program W3Filer V1.13 (if not provided in this package), can be found in
   www.windows95.com I believe.
D. Turbo Pascal (ANY version).

Well, enough blah blah, let's go cracking...
Run W3Filer 32.
A nag screen pops, and, demands registration (Hmm, this sux ;-)) Now,
We notice this program has some kind of serial number (Mine is 873977046),
Let's keep the serial in mind, I bet we'll meet it again while we're on
the debugger.

Well, now, let's put your name and a dummy reg code...
set a BP on GetDlgItemTextA, and, press OK.
We pop inside GetDlgItemTextA, Let's find the registration routine...
I'll save you the work, the registration routine is this:

```
:00404DB2 8D95A8FAFFFF            lea edx, dword ptr [ebp+FFFFFAA8]
:00404DB8 52                      push edx          ---> Your user name here.
:00404DB9 E80B550000              call 0040A2C9     ---> Registration routine.
:00404DBE 83C408                  add esp, 00000008 ---> Dunno exactly what is it.
:00404DC1 85C0                    test eax, eax     ---> Boolean identifier, 0 if
:00404DC3 7D17                    jge 00404DDC      ---> registration failed, 1 if
                                       OK.
```

Well, Let's enter the CALL 40A2C9, and see what's inside it:
(Read my comments in the code).

* Referenced by a CALL at Addresses:
  |:00404DB9 , :00407F76
  |
  :00404A2C9 55                      push ebp
  :0040A2CA 8BEC                    mov ebp, esp
  :0040A2CC 81C4B0FEEFFF            add esp, FFFFE0B
  :0040A2D2 53                      push ebx
  :0040A2D3 56                      push esi
:(040A2D4) 57    push edi
:(040A2D5) 8B5508 mov edx, dword ptr [ebp+08]
:(040A2D8) 8DB500FFFFFF lea esi, dword ptr [ebp+FFFFFF00]
:(040A2DE) 33C0 xor eax, eax
:(040A2E0) EB16 jmp 0040A2F8

* Referenced by a (U)nconditional or (C)onditional Jump at Address:
  |:0040A2FB(C)
  |
  :040A2E2 0FBE0A movsx ecx, byte ptr [edx] ----> Here Starts the
interesting part.
  :040A2E5 83F920 cmp ecx, 00000020 ----> ECX is the the current
char in the user name, Hmm, 20h=' '...
  :040A2E8 740D je 0040A2F7 ----> Let's see,
  :040A2EA 8A0A mov cl, byte ptr [edx] ----> Generally, all this loop
does, is copying
  the user name from
  [EDX], to [ESI], WITHOUT the spaces!
  (Keep this in mind! ).

  :040A2EC 880C06 mov byte ptr [esi+eax], cl
  :040A2EF 42 inc edx
  :040A2F0 40 inc eax
  :040A2F1 C6040600 mov byte ptr [esi+eax], 00
  :040A2F5 EB01 jmp 0040A2F8

* Referenced by a (U)nconditional or (C)onditional Jump at Address:
  |:0040A2E8(C)
  |
  :040A2F7 42 inc edx

* Referenced by a (U)nconditional or (C)onditional Jump at Addresses:
  |:0040A2E0(U), :0040A2F5(U)
  |
  :040A2F8 803A00 cmp byte ptr [edx], 00
  :040A2FB 75E5 jne 0040A2E2 ----------------> This is the loop , we got
what it does,
  Let's continue tracing
  the code...

  :040A2FD 56 push esi  ------- > The user name is pushed, in order
to
  Upcase it's chars.

* Reference To: USER32.CharUpperA, Ord:0000h
  |
  :040A2FE E80F3300000 Call User!CharUpper --- > After this, our name is in
upper case.
  :040A303 56 push esi  ------- > Our name in upper case here.

* Reference To: cw3220mt._strlen, Ord:0000h
  |
  :040A304 E86F3000000 Call 0040D378 --- > This is the length of our name.
  :040A309 59 pop ecx
  :040A30A 8BC8 mov ecx, eax  ---> ECX=Length.
  :040A30C 83F904 cmp ecx, 00000004  ---> Length>=4 (MUST).
  :040A30F 7D05 jge 0040A316  ---> Let's go to this address...
  :040A311 83C8FF or eax, FFFFFFFF
  :040A314 EB67 jmp 0040A37D

* Referenced by a (U)nconditional or (C)onditional Jump at Address:
One thing before we continue, EDX = 00000000h as we enter to the next instructions.

* Referenced by a (U)nconditional or (C)onditional Jump at Address: |:0040A333(C) |
| :0040A331E 0FBE1C06 movsx ebx, byte ptr [esi+eax] ----> EBX <--- char in user name, offset EAX. | :0040A322 C1E303 shl ebx, 03 ----> Hmm, it shr's the char by 03h... (Remember that). |
| :0040A325 0FBE3C06 movsx edi, byte ptr [esi+eax] ----> Now EDI <--- Char in user name, offset EAX. | :0040A329 0FAFF8 imul edi, eax ----> It multiplies the char by the offset in user name! (Remember that). |
| :0040A32C 03DF add ebx, edi ----> Adds the result to EBX (That was Shelled (Ding Dong =)). | :0040A32E 03D3 add edx, ebx ----> EDX=EDX+EBX!!! - This is the CORE of this registration routine!!! |
| :0040A330 40 inc eax ----> Increase EAX by one (next char). | :0040A333 7FE9 jg 0040A331E ----> If ECX<EAX then, we leave the loop. |

* Possible StringData Ref from Data Obj ->"%lx"
Making the actual Keygen
~~~~~~~~~~~~~~~~~~~~~~~~

Now, after I've explained how does the program calculate the registration code, you can either write your own keymaker, without looking at my code, or look at my code (in Turbo Pascal - sorry for all you C lovers ;-) Next time).

That's it, here's the source of my keygen:

-------------------- Cut here ---------------------------------------------

Program W3FilerKeygen;
var
Key, SerialNum, EB, ED, digit: Longint;
I, x: Byte;
Name, KeyHex: String;

begin
  Writeln(' W3Filer32 V1.1.3 Keymaker');
  writeln('Cracked by ^pain^ ''97 / Rebels!');
  Write('Your Name:');      { Read the name }
  readln(Name);
  Write('Serial Number:');
  readln(SerialNum);        {Yes, we need the serial number for the calculation!}
  Key:=0;
  x:=0;
  For I:=1 to length(Name) do
    begin
      Name[I]:=upcase(Name[I]);
      If Name[I]<> ' ' then begin
        eb:=ord(Name[I]) shl 3;  {EB = Name[I] Shl 03h}
        Ed:=ord(Name[I]);        {ED = Name[I]}
        ed:=ed*(x);              {ED=ED*Offset}
        inc(x);
        eb:=eb+ed;               {Add ED to EB}
        Key:=Key+EB;             {Add EB to KEY}
      end;
    end;
  Key:=Key+(SerialNum shr 3);  { Add SerialNum shr 03h to Key}

  { From here, this is just HEX2STRING --> I`m quite sure it's
    Self explaintory, else - go and learn number bases again! ;-)
  KeyHex:='';
  repeat
    digit:=Key mod 16;
    key:=key div 16;
    If digit<10 then KeyHex:=Chr(Digit+ord('0'))+KeyHex;
    If digit>10 then KeyHex:=Chr(Digit-10+ord('a'))+KeyHex;
    until key=0;
  writeln('Your Key:',KeyHex);
  writeln('                 Enjoy!');
end.

-------------------------------- Cut here -------------------------------------

This tutorial was written by ^pain^ / [mEXELiTE '97], Hope you enjoyed
reading it, I`m always trying to improve my writing skills =).

Hmm, I'd like to greet the following: (No special order)

Blast Soft, Teraphy, J0b, Qapla, +ORC, Fravia, Charley, GhostRdr, Odin, kOUGER
Niabi, Acpizer, Klagosong, Mystic Ripter, rANDOM, riddLER (Come back man!
we NEED ya), yoshi, JosephCo, Leddy, Krazy_N, Vizion, Gunnar_, Volcanic,
Fant0m, Caruso, |PSA|, razzi, ThePharao, |KAIRN| + Everyone in #cracking & in
#cracking4newbies, And ofcourse - everyone else I forgot. ;)

------------- Signing off - ^pain^ --------------------------------------