Firehand Ember by the intruder

Mexelite'97

Please send your comments to `devils_cave@hotmail.com`

I would like to thanks nIabI, JosephCo, SasbeJr, Sice_boy, Drlan and others I don't remenber now. Thanks guys!

Please excuse my english!

Firehand Ember is the image file manager for Windows u can get it at www.firehand.com. This proggie only allows 40 sessions and in the beginning a dialog box pops up telling u the number of the current session.

U can register this one but I'll use a different approach coz I have been seing a lot of newbies that are only able to crack proggies using 'getdlgitemtext' and 'getwindowtext' and I hope this tute will give them another perspective about cracking.

To crack this one u will need soft-ice, w32dasm (or other), an hex editor and Win-eXpose-Registry (this is a great tool, this baby will register every access to the registry. Get it at http://www.shetef.com/ .)

So let's cut the crap and start working!!!

So Ember is already installed and when u run it u can see the nag telling u `"Evaluation session XX of 40"`. After running out of evaluation sessions the nag will tell u that your evaluation period is over and u should register, but it will be fully functional (not sure). Even if u install it again it will remenber the session number, hmm… so the proggie is keeping the number of sessions somewhere, but where? First thing to do is to check if the uninstall routine didn't left any file behind. Check c:\windows for ember32.ini or something …. nothing. So let's check the registry to see if the installation left any key with the number of sessions. Run Win-eXpose-Registry and then run Ember, WOW a lot of registry access, but all we want is ember32 registry access so choose filters and check the box 'Report only one task registry ….' and clear all registry operations except **Query Key Value** and **Query Key Value Ex** coz this will be the operations that will get the the registry value of a key. Then check out all strange keys that appeard in WineXpose, hmm… what is this?

LOCAL \ SOFTWARE \ MICROSOFT \ CURRENTVERSION \ MSOFC \ SLC

and

LOCAL \ SOFTWARE \ MICROSOFT \ CURRENTVERSION \ MSOFC \ SC

check out this two keys.

Now use your registry editor (C:\WINDOWS\ Regedit.exe) open this keys and run Ember32 a couple of times.
Ahh LOCAL \ SOFTWARE \ MICROSOFT \ CURRENTVERSION \ MSOFC \ SC
the SC key as a value like 0xfffffffe and each time I run Ember32 the key decreases by one.

So now we know wich key is being used. Now quicky disassemble Ember32.exe with your favorite disassembler . Now look for RegQueryValue and RegQueryValueEx, it's the last one that is being used.

```
LONG RegQueryValueEx(
    HKEY  hKey,  // handle of key to query
    LPTSTR  lpszValueName,          // address of name of value to query
    LPDWORD  lpdwReserved,          // reserved
    LPDWORD  lpdwType,              // address of buffer for value type
    LPBYTE  lpbData,        // address of data buffer
    LPDWORD  lpcbData  // address of data buffer size
)
```

So load Ember into sice and bpx RegQueryValueExA, now run it! Each time sice pops up press F11 to get out of the call then check the 5th push to check the key name, after some calls u'll find our key name 'SC', now bpx just before the call and run the proggie again now check the 2nd push and u'll get the mem location where the value is going to be stored. Now just bpr mem location and press CTRL+D then sice pops up and u'll be inside the compare routine

```
:0040CAC3 8B442410      mov eax, [esp + 10]       ;eax=SC key value
:0040CAC7 F7D0          not eax                    ;now eax=number of runned
                                                   sesssions
:0040CAC9 83F801        cmp eax, 00000001         ;compare with 1
:0040CACC 7C17          jl 0040CAE5               ;jump if lower
:0040CACE 8B8F58020000  mov ecx, [edi+00000258]
:0040CAD4 398F5C020000  cmp [edi+0000025C], ecx
:0040CADA 7C13          jl 0040CAEF
:0040CADC 40            inc eax                   ;increment eax->session number
                                                   update
:0040CADD 898758020000  mov [edi+00000258], eax  ;store eax for later check
```

now let's check where mem location edi+00000258 is read , bpr mem location for reading
then CTRL+D and u'll be inside the following code:

```
:0040B3EE 8B9758020000  mov edx, [edi+00000258]  ;load edx number of current
                                                   session
:0040B3F4 83E002        and eax, 00000002         ;trash
:0040B3F7 31442434      xor [esp + 34], eax       ;trash
:0040B3FB 8BC1          mov eax, ecx              ;trash
:0040B3FD 33442434      xor eax, [esp + 34]       ;trash
:0040B401 83E004        and eax, 00000004         ;trash
:0040B404 31442434      xor [esp + 34], eax       ;trash
:0040B408 39975C020000  cmp [edi+0000025C], edx  ;cmp current session with
                                                   [edi+0000025C]= 28h =40decimal
:0040B40E 7D10          jge 0040B420              ;jump if greater
```

got it? Let's move on just press CTRL+D and sice pops again. Here's the code:

```
:0040B58D 8B875C020000         mov eax, [edi+0000025C] ;eax=28h=40d
:0040B593 8B8F58020000         mov ecx, [edi+00000258] ;ecx=current session number
:0040B599 C7878002000001000000 mov dword ptr [edi+00000280], 00000001
:0040B5A3 3BC1                 cmp eax, ecx            ;compare
:0040B5A5 7D0C                 jge 0040B5B3             ;jump if greater
```

hmm… how do we crack it? There are many ways. It seems to me that the just change the best way to crack this one is to change both jge to jmp this way no matter what session the program will always run the way u want. So change:

```
:0040B40E 7D10          jge 0040B420
to
:0040B40E EB10          jmp 0040B420
```
and

```
:0040B5A5 7D0C                 jge 0040B5B3
```
to
```
:0040B5A5 EB0C                 jmp 0040B5B3      ;
```

now just use your hex edito and patch it.

OK, the session counter is cracked now for the nag.

First thing to do is to search for strings used in the nag in your disassembled list let's search for evaluation coz in the nag appears that horribilis 'Evaluation session XX of 40'. If found a reference to this one right here:

```
* Possible StringData Ref from Data Obj ->"Evaluation session %ld of %ld"
                                  |
:0040CEAD 68B41E4300              push 00431EB4
:0040CEB2 50                      push eax

* Reference To: USER32.wsprintfA, Ord:0249h
                                  |
:0040CEB3 FF1590784300            Call dword ptr [00437890]
:0040CEB9 8D4C2414                lea ecx, [esp + 14]
:0040CEBD 8BB42498000000          mov esi, [esp + 00000098]
:0040CEC4 83C410                  add esp, 00000010
:0040CEC7 51                      push ecx
:0040CEC8 68FB030000              push 000003FB
:0040CECD 56                      push esi

* Reference To: USER32.SetDlgItemTextA, Ord:01DEh
                                  |
:0040CECE FF1598784300            Call dword ptr [00437898]
:0040CED4 6A00                    push 00000000
:0040CED6 56                      push esi
:0040CED7 E854080000              call 0040D730
:0040CEDC B801000000              mov eax, 00000001
:0040CEE1 5E                      pop esi
:0040CEE2 81C480000000            add esp, 00000080
:0040CEE8 C21000                  ret 0010
```

a `'ret 0010'` where r we going??? Ok, load ember to soft-ice again and bpx on 40CEE8 to find out from where this routine is being called. Now soft-ice pops up and we r inside the code, just press F8 and to see where the hell u are. Damn we r inside another call. hehehe fucking win95 it's just a bunch of calls inside other calls. So just keep pressing F10 to jump through the calls until u get back to the Ember32 code (just check the line that separates the code window from the command line).

When u get back to Ember32 (this is the name of the Ember32.exe module) u will be inside the following code:

```
:0040B5BF 6870CE4000              push 0040CE70
:0040B5C4 56                      push esi
:0040B5C5 6A7B                    push 0000007B
:0040B5C7 53                      push ebx

* Reference To: USER32.DialogBoxParamA, Ord:008Ah
                                  |
:0040B5C8 FF1564774300         Call dword ptr [00437764]
:0040B5CE C787800200000000000 mov dword ptr [edi+00000280],00000000;<
                                                               
                                                       HERE
```

WOW if it's `DialogBoxParamA`. Check this out:

int DialogBoxParam(

```
   HINSTANCE  hInstance,          // handle of application instance
   LPCTSTR  lpTemplateName,     // identifies dialog box template
   HWND  hWndParent,   // handle of owner window
   DLGPROC  lpDialogFunc,        // address of dialog box procedure
   LPARAM  dwInitParam             // initialization value
  )
```

Now how do we crack a nag? Well we'll just nop the call, right? Yeah, but don't forget that after the call the stack pointer has a different value and we don't want to cause a General Protection Fault (GPF) so the trick is to put in the stack pointer the same value that it would have after getting out of the `DialogBoxParamA`

So just load the proggie again into soft-ice and bpx just before the call 'bpx 40b5c8' when sice pops up just write down the ESP value (mine is 67F8A0) now press F10 to run the call and write the new ESP value (my ESP=67F8B4). Now let's quickly crack it:

instead of:

```
:0040B5C8 FF1564774300   Call dword ptr [00437764
```

which is 6 byte long we will have:

```
:0040B5C8 83C414         Add esp,14  ;this way we keep the final value of the stack
:0040B5CB 44             Inc esp     ;do nothing
:0040B5CC 4C             Dec esp     ;
:0040B5CE 90             Nop         ;
```

which is 6 byte long too.

Now get in your hex editor and crack it!!!
Hope u enjoyed!!!!!