Detecting Windows Server Compromises with *Patchfinder 2*

Joanna Rutkowska
joanna@mailsnare.net

January 2004

Introduction

Patchfinder (PF) is sophisticated diagnostic utility designed to detected system libraries and kernel compromises. Its primary use is to check if the given machine has been attacked with some modern rootkits, i.e. programs which tries to hide attacker's activity on the hacked system by cheating operating system about the list of active processes, files on filesystem, running services, registry contents, etc…

New release (2.x) of PF is the first version, which is designed to be not only a proof-of-concept code for developers, but also to be useful tool for administrators. To make a proper use of the PF, every user should read this paper.

With this tool you should be able to detect even the newest versions of such rootkits like: *Hacker Defender*, *APX*, *Vaniquish*, *He4Hook*, and many more…

Due to its design PF is not able to detect rootkits which exploits DKOM (Direct Kernel Object Manipulation) technology, first introduced by James Butler in his paper [2] and implemented in *fu* rootkit [1]. However you can use *klister* [5] utility in order to detect such rootkits.

Patchfinder 2 design

New PF compromises of three components: kernel module, service program and an agent program to display results (see figure 0).

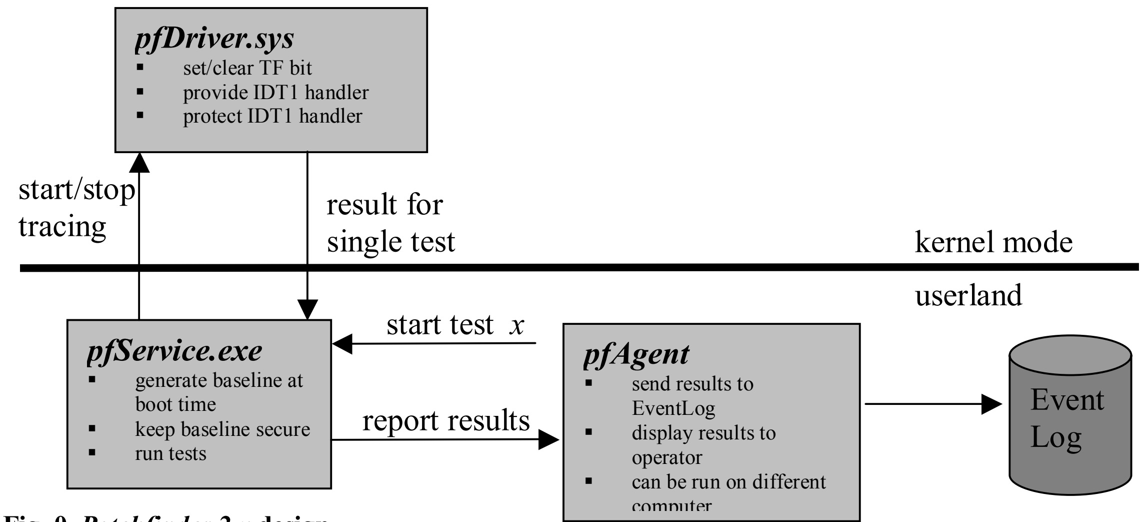


Fig. 0. *Patchfinder 2.x* design.

When the system boots, the `pfDriver.sys` is loaded into kernel, and also `pfService.exe` process is started. The service first goes to sleep for about 2 minutes, waiting for the rest of the system to complete initialization, and then the baseline generation phase starts. PF service runs all the tests and stores the results in its memory.

These results are called *baseline*, and all future tests will be compared against this baseline. Tests can be viewed as controlled execution of some system services by `pfService.exe` process. Execution path of each system service is traced. The tracing is supported by hardware (i.e. step mode) and kernel module. The testing mechanism and tests are described in more details in other papers about Patchfinder concept ([3] and [4]).

When the `pfService.exe` completes baseline generation it goes to ready state. This is about 3 minutes after the system boot. Now agent program can be used to instruct service process to run the tests again, and to compare the current results with the baseline. Every difference can be sign of one or more system service compromise. Because agent uses named pipe to communicate with the service process, it can be run on the remote computer. In current implementation it is required that the user who wants to run agent program belongs to *Authenticated-Users* group on the target computer (i.e. the one which runs the PF service).

Installation

1. Copy PF binary files to separate directory on your local disk, e.g.: `C:\pf2`.
2. From the command prompt:
   a. go to PF directory (`cd C:\pf2`)
   b. run `pfInstall.exe` with the PF directory as an argument:
      `pfInstall --install C:\pf2`
3. Reboot the machine.
4. When the system boots again, run *EventViewer* and look in the *System* log for any *pfService* messages. You should not see any errors, only diagnostic messages.
5. After few minutes, when the PF completes baseline generation, you can run `pfAgentConsole.exe` program to check your system.

Security Issues

First of all Patchfinder can be useful only when the system is *clear* during the boot time[1]. It means that when somebody has installed *Hacker Defender* rootkit for example, and then rebooted your system, PF will not detect anything. This should not be concerned as PF weakness however, since there are plenty of filesystem integrity checkers (like *Tripwire* [7] or *AIDE* [8]), which should be used in such circumstances. Please note however, that running those integrity checkers on the target system is not very useful, since most modern rootkits can cheat integrity checkers. This is why, when unexpected system reboot[2] has been discovered, administrator should remove the system disk form the server, put it into clean computer and run the filesystem integrity checker program from the clear system.

[1] i.e. not compromised by any rootkit
[2] Unexpected reboots should be detected in a secure environment.

You should then make sure that agent program is run just before all your *planned* shutdowns. You can add `pfAgentConsole.exe` program to be run as a shutdown script[3].

However it is strongly recommended that you run agent to check your system more often then just before reboot. The agent program is able to generate log entries about detected anomalies so you can use built-in windows Task Scheduler to run it periodically (e.g. once an hour). The logs are placed in the *System* log.

False Positives

The world is not perfect, and the operating systems are very complex unfortunately. It means that it is likely that you will notice some small differences appearing in the tests results. Peek positions will be slightly different, typically few to few dozens of instructions. The obvious question is how to determine if the difference let say, 50 instructions, is a false positive or a rootkit? Well, currently available rootkits seem to add much bigger difference, like few thousands of instructions, so few dozens of instructions is probably only a false positive…

However in security critical applications it would be desirable if the user has some utility which will allow her to make detailed system analysis, i.e. to show where exactly are the differences, on which addresses, to display the histograms graphically, and possibly even to allow a debugger, like *kd*, to attach to the kernel in order to analyze the system. Such utility, *pfAgentStudio*, is under development, will have a Graphical User Interface, and most of the features just described.

Patchfinder protection

There are some efforts, described in other papers ([3], [4]) concerning how to protect kernel driver against rootkit attacks. However, introduction of additional layer, i.e. service process, makes it necessary to discuss some other issues. The most important is how to protect the baseline information, which is stored in this process. The attacker could kill the process, start it again, and it will generate new baseline, this time with results for the compromised system. Such situation should be avoided.

Because we cannot guarantee that `pfService.exe` process will not be killed some time in the future by smart attacker, the special variable has been added to the kernel module, `pfsrv_started`, which is initialized to zero, when the kernel module is loaded (i.e. during system startup). When the service process starts it asks the module about this variable and if it is greater then zero service will exit, logging error information, which is a sign that the system is probably compromised. During the variable check it is automatically incremented by kernel module, which is why it should be not possible to run the service more then once.

Also the kernel module is protected so it cannot be unloaded. If you try to do it, using for example `w2k_load.exe` utility the system will crash. This is necessary for providing safety to PF service (see paragraph above) an also it is a result of hardware protection of IDT1 entry (see [4] for more details).

[3] `Group Policy\Local Policy\Computer Configuration\Windows Settings\Scripts\Shutdown`

References:

[1] fuzen_op, *fu_rootkit,* `http://rootkit.com`.

[2] James Butler et al., *HIDDEN PROCESSES: The Implication for Intrusion Detection*, Proceedings of the 2003 IEEE Workshop on Information Assurance United States Military Academy, West Point, NY, June, 2003.

[3] Joanna Rutkowska, *Detecting Windows Server Compromises*, HiverCon Security Conference, Dublin, November 2003, `http://www.hivercon.com/conf/archive/hc03/Rutkowska_Win32Rookit Detection_HC2003.ppt`

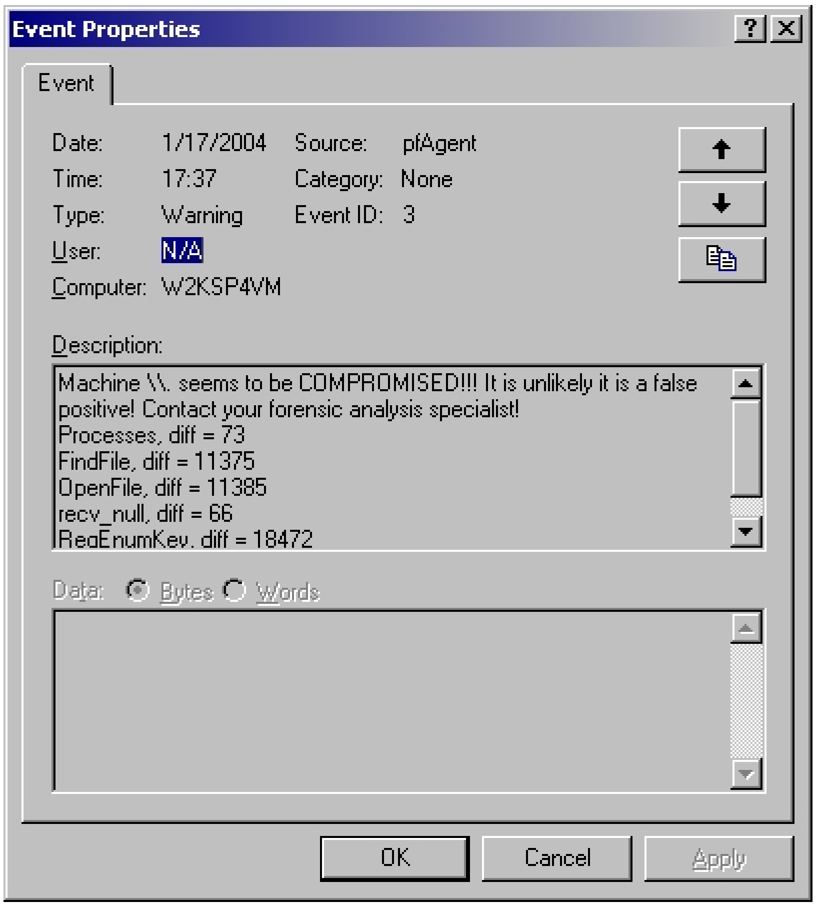
[4] Jan K. Rutkowski, *Advanced Windows 2000 Rootkits detection, Black Hat Briefings*, Las Vegas, July, 2003, `https://www.rootkit.com/vault/joanna/windows_rootkit_detection_joanna.pdf`

[5] Joanna Rutkowska, *klister tool*, `http://www.rootkit.com`.

[6] Joanna Rutkowska, *Patchfinder 2.x,* `http://www.rootkit.com`.

[7] Tripwire for Windows, `http://www.tripwire.com/`.

[8] AIDE, `http://www.cs.tut.fi/~rammer/aide.html` (requires *Cygwin* to run under Windows).



Date: 1/17/2004 Source: pfAgent
Time: 17:37 Category: None
Type: Warning Event ID: 3
User: N/A
Computer: W2KSP4VM

Description:

```
Machine \\. seems to be COMPROMISED!!! It is unlikely it is a false
positive! Contact your forensic analysis specialist!
Processes, diff = 73
FindFile, diff = 11375
OpenFile, diff = 11385
recv_null, diff = 66
RegEnumKey, diff = 18472
```

Data: [✓ Bytes] Words

Fig. 1. Example: Patchfinder detects popular Windows rootkit, *HackerDefender* v.1.0.0, on the local system.