

Forum: [Hack Forums](#)

Thread: [\[TUTORIAL\] How To Get Root Access to Any Server](#)

PSEUCHO - January 19, 2017, 8:07 am

How To Get Root Access to Any Server

Hello Hf! Today I will show you a simple way to get root access to almost any Linux server (at least it worked with every server I tried it with)

Step 1: Download

We will be using the so called 'Dirty cow exploit'

The github link for the exploit: <https://github.com/gbonacini/CVE-2016-5195>

To install it type Code git clone <https://github.com/gbonacini/CVE-2016-5195> in the terminal

Step 2: Setup

Once you are done downloading it, navigate into the folder and you will see these files:

Code changelog CONTRIBUTING.md dcow dcow.cpp makefile README.md version
run Code make clean and when it is done runCode make

Step 3: Using the exploit

To run the exploit simply use: Code ./dcow

It will then change the password to dirtyCowFun

Done, you should be able to login as root now!

For any questions regarding this, pm me!

As always, feel free to leave an honest feedback

PSEUCHO ?

B?ker - January 19, 2017, 8:11 am

Seems pretty dope curious how well it works

Papa Cuda - January 19, 2017, 8:21 am

There's actually a way to mitigate this attack on all affected servers. Unfortunately the number of affected servers is in the 10's, if not 100's of thousands.

To mitigate the dirty cow exploit you can use the following system tap script:

The SystemTap script is relatively small and efficient, broken into 4 distinct sections as follows:

```
Code probe kernel.function("mem_write").call ? {  
    $count = 0  
}
```

```

probe syscall.ptrace { // includes compat ptrace as well
    $request = 0xfff
}

probe begin {
    printk(0, "CVE-2016-5195 mitigation loaded")
}

```

```

probe end {
    printk(0, "CVE-2016-5195 mitigation unloaded")
}

```

First, the script places a probe at the beginning of the kernel function “mem_write” when called and not loaded inline:

```

Code probe kernel.function("mem_write").call ? {
    $count = 0
}

```

Next, the script places a probe at the ptrace syscalls that disables them (this bit can impair antivirus software and potentially other kinds of software such as debuggers):

```

Code probe syscall.ptrace { // includes compat ptrace as well
    $request = 0xfff
}

```

Finally, the “probe begin” and “probe end” code blocks tell systemtap to add the supplied text to the kernel log buffer via the printk function. This creates an audit trail by registering in the system logs exactly when the mitigation is loaded and unloaded.

This solution works in all affected RHEL versions: 5, 6, and 7.

If you want me to remove this just let me know.

PSEUCHO - January 19, 2017, 11:20 am

(Jan 19, 2017 09:21 AM)cudacore Wrote: [CITE](#) There's actually a way to mitigate this attack on all affected servers. Unfortunately the number of affected servers is in the 10's, if not 100's of thousands.

To mitigate the dirty cow exploit you can use the following system tap script:

The SystemTap script is relatively small and efficient, broken into 4 distinct sections as follows:

```

Code probe kernel.function("mem_write").call ? {
    $count = 0
}

probe syscall.ptrace { // includes compat ptrace as well
    $request = 0xfff
}

probe begin {
    printk(0, "CVE-2016-5195 mitigation loaded")
}

```

```
probe end {  
    printk(0, "CVE-2016-5195 mitigation unloaded")  
}
```

First, the script places a probe at the beginning of the kernel function "mem_write" when called and not loaded inline:

```
Code probe kernel.function("mem_write").call ? {  
    $count = 0
```

}Next, the script places a probe at the ptrace syscalls that disables them (this bit can impair antivirus software and potentially other kinds of software such as debuggers):

```
Code probe syscall.ptrace { // includes compat ptrace as well  
    $request = 0xfff  
}
```

Finally, the "probe begin" and "probe end" code blocks tell systemtap to add the supplied text to the kernel log buffer via the printk function. This creates an audit trail by registering in the system logs exactly when the mitigation is loaded and unloaded.

This solution works in all affected RHEL versions: 5, 6, and 7.

If you want me to remove this just let me know.

Thanks a lot for your contribution, why would I want you to remove this? ?

Freddy Kru?ger - January 19, 2017, 11:21 am

Quite a well known exploit honestly, thanks for posting anyway.

Papa Cuda - January 19, 2017, 11:24 am

(Jan 19, 2017 12:20 PM)PSEUCHO Wrote: [\[CITE\]](#) Thanks a lot for your contribution, why would I want you to remove this? ?

Well as much as it is good to know how to USE the dirty cow exploit it is also important to know how to protect against it :)

DeathByHR - January 20, 2017, 5:50 am

Quick question i amuse this is for kail Linux yes? second question is this could work on RPD server's as well am i correct?

Freddy Kru?ger - January 20, 2017, 12:39 pm

(Jan 20, 2017 06:50 AM)DeathByHR Wrote: [\[CITE\]](#) Quick question i amuse this is for kail Linux yes? second question is this could work on RPD server's as well am i correct?

You don't need Kali for this, any distro will do. It will work on RPD servers running Linux and haven't protected against the exploit. I think.

Wizmo - January 20, 2017, 12:43 pm

will try this later. Anyone tried this yet?

Freddy Kru?ger - January 20, 2017, 1:13 pm

(Jan 20, 2017 01:43 PM)Wizmo Wrote: [\[CITE\]](#) will try this later. Anyone tried this yet?

Most big names in the linux world like Debian, Ubuntu and RHL have already patched this.