

Guidance for investigating attacks using CVE-2022-21894: The BlackLotus campaign | Microsoft Security Blog

Microsoft Incident Response : 13-17 minutes : 4/11/2023

This guide provides steps that organizations can take to assess whether users have been targeted or compromised by threat actors exploiting [CVE-2022-21894](#) via a Unified Extensible Firmware Interface (UEFI) bootkit called BlackLotus. UEFI bootkits are particularly dangerous as they run at computer startup, prior to the operating system loading, and therefore can interfere with or deactivate various operating system (OS) security mechanisms such as BitLocker, hypervisor-protected code integrity (HVCI), and Microsoft Defender Antivirus. Though this could impede investigations and threat hunting efforts, several artifacts can still be leveraged to identify affected devices. This document covers:

- Techniques to determine if devices in an organization are infected
- Recovery and prevention strategies to protect your environment

It is critical to note that a threat actor's use of this bootkit is primarily a persistence and defense evasion mechanism. It is not a first-stage payload or an initial access vector and can only be deployed to a device to which a threat actor has already gained either privileged access or physical access. The malware uses CVE-2022-21894 (also known as [Baton Drop](#)) to bypass Windows Secure Boot and subsequently deploy malicious files to the EFI System Partition (ESP) that are launched by the UEFI firmware. This allows the bootkit to:

- Achieve persistence by enrolling the threat actor's Machine Owner Key (MOK)
- Turn off HVCI to allow deployment of a malicious kernel driver
- Leverage the kernel driver to deploy the user-mode HTTP downloader for command and control (C2)
- Turn off BitLocker to avoid tamper protection strategies on Windows
- Turn off Microsoft Defender Antivirus to avoid further detection

For a comprehensive analysis of the BlackLotus installation process and follow-on actions, read [this blog by ESET](#).

Detection opportunities

Microsoft Incident Response (previously known as Microsoft Detection and Response Team – DART), through forensic analysis of devices infected with BlackLotus, has identified multiple opportunities for detection along several steps in its installation and execution processes. The artifacts analyzed include:

- Recently written bootloader files
- Staging directory artifacts created
- Registry key modified
- Windows Event logs entries generated
- Network behavior
- Boot Configuration log entries generated

As threat hunters begin examining environments, it is crucial to adopt a comprehensive hunting strategy across these artifacts to down-filter false positives and surface true positives. Many of these artifacts, when observed in isolation, are low fidelity. Observing them in tandem with others, however, increases their significance in determining if a device has been infected.

Recently created and locked bootloader files

BlackLotus writes malicious bootloader files to the EFI system partition (ESP) and subsequently locks them to protect them from deletion or tampering. If recently modified and locked files are identified in the ESP on a device, especially those matching known BlackLotus bootloader filenames, these should be considered highly suspect and the devices should be removed from the network to be examined for further evidence of BlackLotus or follow-on activity.

To determine if such files exist in the ESP, threat hunters can mount the boot partition (with the *mountvol* command-line utility, for example) to examine the creation dates of the files within. Files with mismatched creation times, as well as those with names matching those protected by the BlackLotus kernel driver, should be considered suspicious (Figure 1). The *LastModified* timestamps of the files in the ESP should be compared to each other; the timestamps and filenames can also be compared against those in the OS partition under *C:\Windows\Boot\EFI*.

The files protected by the driver include, as originally listed by ESET:

- *ESP:\EFI\Microsoft\Boot\winload.efi*
- *ESP:\EFI\Microsoft\Boot\bootmgfw.efi*

• ESP:\EFI\Microsoft\Boot\grubx64.efi

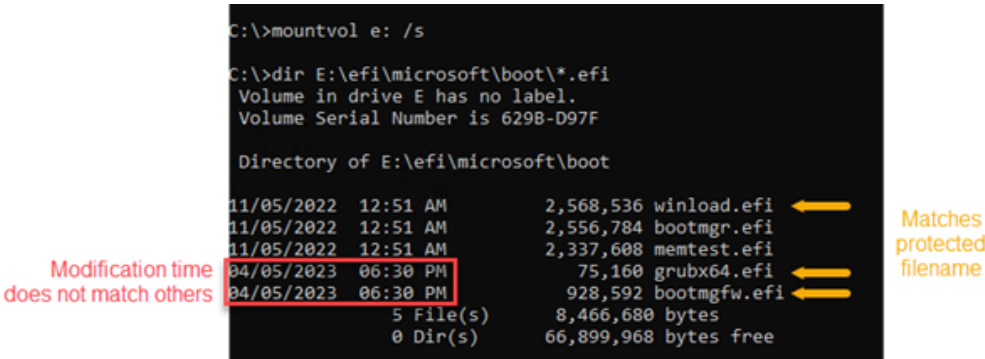


Figure 1: Evidence of recent modification dates and matching filenames of BlackLotus-associated files on a BlackLotus infected device.

To further confirm if any files with matching filenames or mismatched modification times are suspect, threat hunters can leverage the local *CertUtil* command-line utility to attempt to calculate the hash of a suspected bootloader file in the ESP. In Figure 1, winload.efi does NOT have a mismatched modified time, yet matches the filename protected by the BlackLotus kernel driver.

Since these protected bootloader files are locked by BlackLotus, any attempt to access these files generates an *ERROR_SHARING_VIOLATION* error with the message “The process cannot access the file because it is being used by another process”. Figure 2 depicts this error being generated when attempting to hash winload.efi on the infected device from Figure 1, further confirming that it is BlackLotus-related in this scenario.

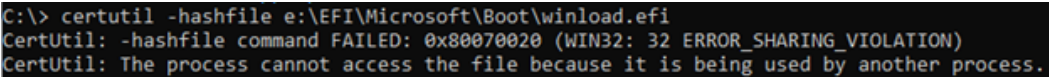


Figure 2: CertUtil reporting an ERROR_SHARING_VIOLATION message upon attempting to hash winload.efi in the ESP of a BlackLotus infected device.

If the malware is active, *CertUtil* reports the sharing violation error as in Figure 2; if not, *CertUtil* reports the hash of the file. Files in the ESP that return this error should be considered highly suspicious, especially those matching the protected filenames listed above.

BlackLotus staging directory presence

During the installation process, BlackLotus creates a custom directory under *ESP:/system32/*. Though the files within are deleted following successful installation, the directory itself is not deleted. Additionally, forensic analysis of the ESP may reveal the historical presence of the files previously contained in this directory (Figure 3).

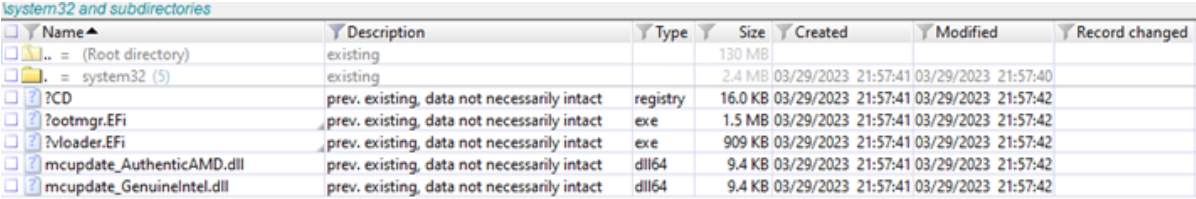


Figure 3: Evidence of deleted files in ESP:/system32/ associated with BlackLotus, in a custom staging directory still present post-installation.

Registry modification

To turn off HVCI, the installer modifies the registry key *HKLM:\SYSTEM\CurrentControlSet\Control\DeviceGuard\Scenarios\HypervisorEnforcedCodeIntegrity* by setting the value *Enabled* to “0” – but only if the key already exists. Threat hunters should examine their environment for this registry key modification (Figure 4).

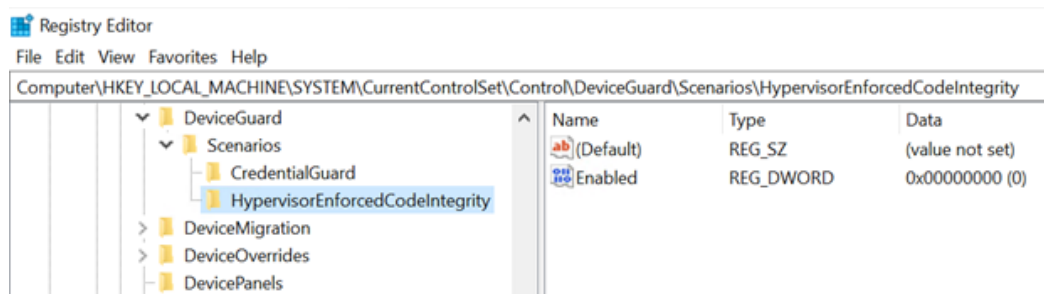


Figure 4: RegEdit depiction of the modified registry key to disable HVCI

Event logs entries

BlackLotus disables Microsoft Defender Antivirus as a defense evasion method by patching its drivers and stripping the main process's privileges.

This behavior may produce entries in the *Microsoft-Windows-Windows Defender/Operational* log in Windows Event Logs. Relevant log entries will indicate that *Antimalware security intelligence has stopped functioning for an unknown reason* (see Figure 5).

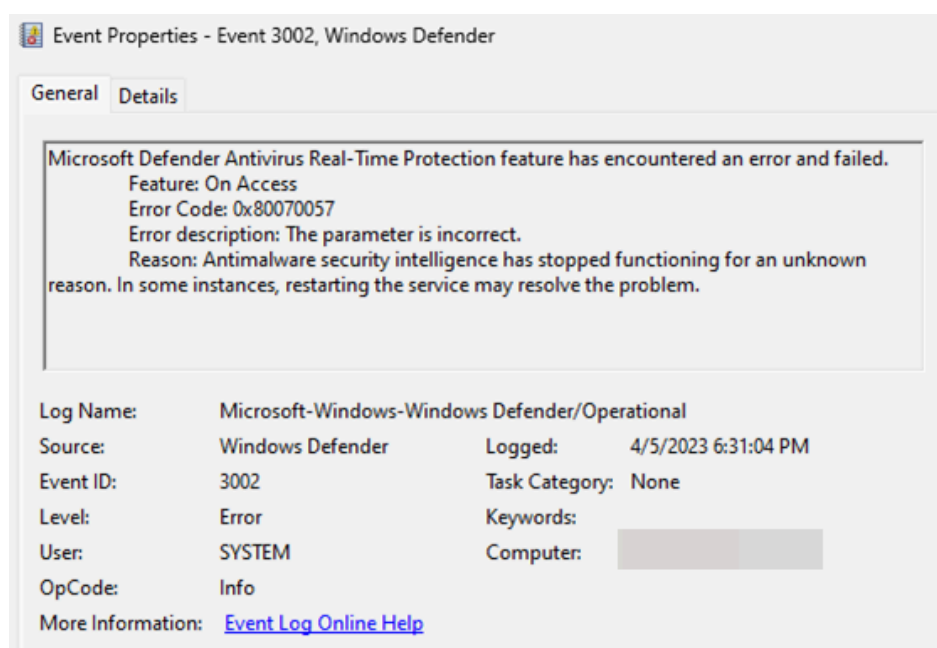


Figure 5: Event Log indicating Microsoft Defender Antivirus real-time protection has stopped functioning

The disabling of Microsoft Defender Antivirus may also result in the service stopping unexpectedly, producing an Event ID 7023 in the System event log (with *Service Control Manager* as the Provider Name). Relevant log entries will name the *Microsoft Defender Antivirus Service* as the affected service (Figure 6).

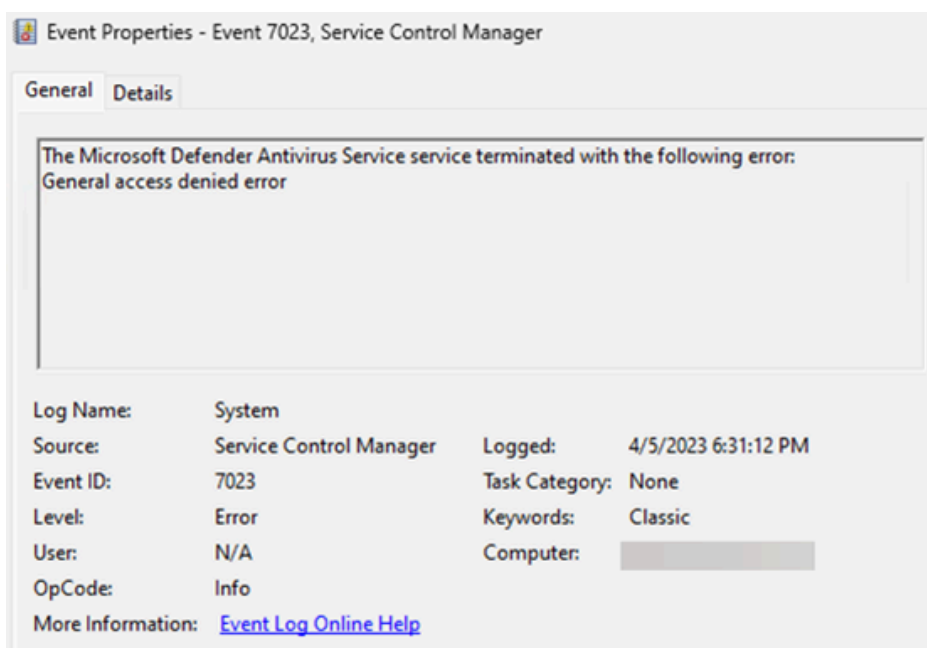


Figure 6: System event log entry indicating the Microsoft Defender Antivirus service has been terminated with an error

Network logging

Outbound network connections from *winlogon.exe*, particularly to port 80, should be considered highly suspicious. This is the result of the injected HTTP downloader function of BlackLotus connecting to the C2 server or performing network configuration discovery. Microsoft Incident Response observed this connection with Sysmon monitoring on an infected device. Figure 7 depicts *winlogon.exe* attempting to communicate to the *api.ipify.org* service to determine the public IP address of the compromised device.

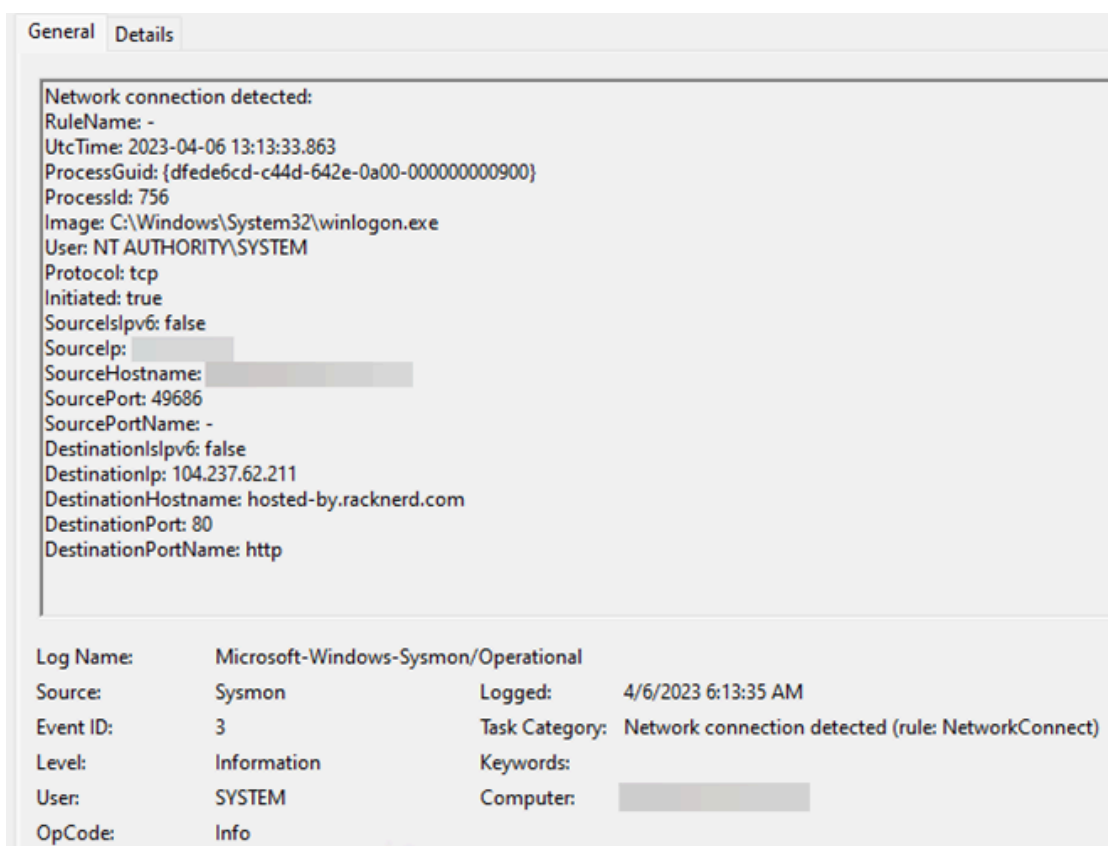


Figure 7: Sysmon event entry indicating *winlogon.exe* attempting to communicate outbound on port 80

This entry was captured with a simple modification to the [SwiftOnSecurity Sysmon configuration](#) (see Figure 8).


```

<RuleGroup name="" groupRelation="or">
  <NetworkConnect onmatch="include">
    <!--Suspicious sources for network-connecting binaries-->
    <Image name="Usermode" condition="begin with">C:\Users</Image> <!--Tools downloaded by users ca
    <Image name="Caution" condition="begin with">C:\Recycle</Image> <!--Nothing should operate from
    <Image condition="begin with">C:\ProgramData</Image> <!--Normally, network communications shoul
    <Image condition="begin with">C:\Windows\Temp</Image> <!--Suspicious anything would communicate
    <Image name="Caution" condition="begin with">\</Image> <!--Devices and VSC shouldn't be executi
    -->
    <Image name="Caution" condition="begin with">C:\perflogs</Image> <!-- Credit @blu3_team [ https
    <Image name="Caution" condition="begin with">C:\intel</Image> <!-- Credit @blu3_team [ https://
    <Image name="Caution" condition="begin with">C:\Windows\fonts</Image> <!-- Credit @blu3_team [
    <Image name="Caution" condition="begin with">C:\Windows\system32\config</Image> <!-- Credit @bl
    <!--Suspicious Windows tools-->
    <Image condition="image">at.exe</Image> <!--Windows: Remote task scheduling, removed in Win10 |
    <Image condition="image">certutil.exe</Image> <!--Windows: Certificate tool can contact outboun
    <Image condition="image">cmd.exe</Image> <!--Windows: Remote command prompt-->
    <Image condition="image">cmstp.exe</Image> <!--Windows: Connection manager profiles can launch
    @subTee -->
    <Image condition="image">winlogon.exe</Image>
    <Image condition="image">cscrip.exe</Image> <!--WindowsScriptingHost: | Credit @Cyb3rOps [ htt
    <Image condition="image">driverquery.exe</Image> <!--Windows: Remote recognisance of system con
    <Image condition="image">dsquery.exe</Image> <!--Microsoft: Query Active Directory -->
    <Image condition="image">hh.exe</Image> <!--Windows: HTML Help Executable, opens CHM files -->
    <Image condition="image">infDefaultInstall.exe</Image> <!--Microsoft: [ https://github.com/hunt
    <Image condition="image">java.exe</Image> <!--Java: Monitor usage of vulnerable application and
    <Image condition="image">javaw.exe</Image> <!--Java: Monitor usage of vulnerable application an
    <Image condition="image">javaws.exe</Image> <!--Java: Monitor usage of vulnerable application a
    <Image condition="image">mmc.exe</Image> <!--Windows: -->
  </NetworkConnect>
</RuleGroup>

```

Figure 8: Modified Sysmon configuration to detect *winlogon.exe* network connection behavior

Analysis of *netstat* output on an affected device may also reveal *winlogon.exe* maintaining a network connection on port 80. Given the configuration capabilities of the implant, the connection may be intermittent.

Boot configuration log analysis

Trusted Computing Group (TCG) logs, also known as *MeasuredBoot* logs, are Windows Boot Configuration Logs that contain information about the [Windows OS boot process](#). To retrieve these logs, the device must be running at least Windows 8 and have the Trusted Platform Module (TPM) enabled.

From [How Windows uses the Trusted Platform Module](#): “Windows 8 introduced Measured Boot as a way for the operating system to record the chain of measurements of software components and configuration information in the TPM through the initialization of the Windows operating system.” “For software, Measured Boot records measurements of the Windows kernel, Early-Launch Anti-Malware drivers, and boot drivers in the TPM.”

The BlackLotus bootkit has boot drivers that are loaded in the boot cycle. *MeasuredBoot* logs list the BlackLotus components as *EV_EFI_Boot_Services_Application*.

These logs are in the *C:\Windows\Logs\MeasuredBoot* directory, which contains multiple files with the extension *.log* – one for each reboot of the system. These logs can be compared to one another to identify deltas in components added or removed from each boot.

In the case of BlackLotus installation, two components are added when BlackLotus becomes active on a system: the *grubx64.efi* driver and *winload.efi* driver (see Figure 9).

Figure 9: BlackLotus components visible in MeasuredBoot logs after parsing to XML

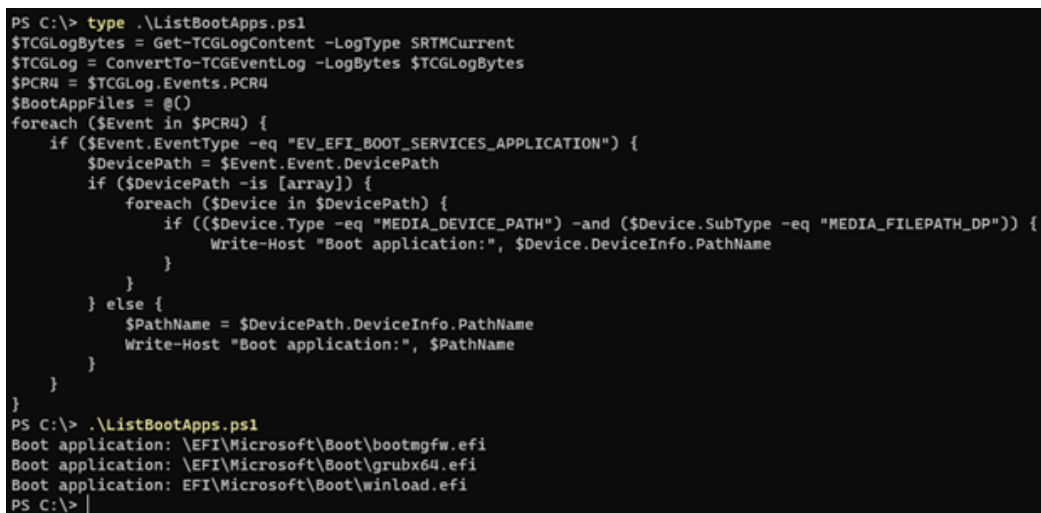
On live machines, the active *MeasuredBoot* log files are stored in system memory and can be read either via the TPM base services API or via the PowerShell shown below. Alternatively, they can be acquired from a forensic image. The log files must then be decoded and converted to XML/JSON. A sample script to extract and parse these logs is presented here, based on [GitHub – mattifestation/TCGLogTools: A set of tools to retrieve and parse TCG measured boot logs](#).

```

$TCGLogBytes = Get-TCGLogContent -LogType SRTMCurrent
$TCGLog = ConvertTo-TCGEventLog -LogBytes $TCGLogBytes
$PCR4 = $TCGLog.Events.PCR4
foreach ($Event in $PCR4) {
    if ($Event.EventType -eq "EV_EFI_BOOT_SERVICES_APPLICATION") {
        $DevicePath = $Event.Event.DevicePath
        if ($DevicePath -is [array]) {
            foreach ($Device in $DevicePath) {
                if (($Device.Type -eq "MEDIA_DEVICE_PATH") -and ($Device.SubType -eq
"MEDIA_FILEPATH_DP")) {
                    Write-Host "Boot application:", $Device.DeviceInfo.PathName
                }
            }
        } else {
            $PathName = $DevicePath.DeviceInfo.PathName
            Write-Host "Boot application:", $PathName
        }
    }
}
}

```

Example output of this script from an infected device can be seen in Figure 10.



```

PS C:\> type .\ListBootApps.ps1
$TCGLogBytes = Get-TCGLogContent -LogType SRTMCurrent
$TCGLog = ConvertTo-TCGEventLog -LogBytes $TCGLogBytes
$PCR4 = $TCGLog.Events.PCR4
$BootAppFiles = @()
foreach ($Event in $PCR4) {
    if ($Event.EventType -eq "EV_EFI_BOOT_SERVICES_APPLICATION") {
        $DevicePath = $Event.Event.DevicePath
        if ($DevicePath -is [array]) {
            foreach ($Device in $DevicePath) {
                if (($Device.Type -eq "MEDIA_DEVICE_PATH") -and ($Device.SubType -eq "MEDIA_FILEPATH_DP")) {
                    Write-Host "Boot application:", $Device.DeviceInfo.PathName
                }
            }
        } else {
            $PathName = $DevicePath.DeviceInfo.PathName
            Write-Host "Boot application:", $PathName
        }
    }
}
PS C:\> .\ListBootApps.ps1
Boot application: \EFI\Microsoft\Boot\bootmgfw.efi
Boot application: \EFI\Microsoft\Boot\grubx64.efi
Boot application: EFI\Microsoft\Boot\winload.efi
PS C:\>

```

Figure 10: Example output of the TCG parsing script to enumerate boot components

Detection details

Microsoft Defender Antivirus detects threat components as the following malware (note that these signatures trigger on hashes of known BlackLotus samples):

- [Trojan:Win32/BlackLotus](#)
- [Trojan:Win64/BlackLotus](#)

Microsoft Defender for Endpoint alerts on known BlackLotus activity and/or post-exploitation activity. The following alert title can indicate threat activity on your network:

- Possible vulnerable EFI bootloader

Network protection in Microsoft Defender for Endpoint blocks connections to known indicators associated with BlackLotus C2 servers.

Recovery and prevention guidance

If a device is determined to have been infected with BlackLotus, the device should be removed from the network and reformatted (both the OS partition and EFI partition) or restored from a known clean backup that includes the EFI partition.

To prevent infection via BlackLotus or other variants abusing CVE-2022-21894, organizations should:

- Practice the principle of least privilege and maintain credential hygiene. Avoid the use of domain-wide, admin-level service accounts. Restricting local administrative privileges can help limit installation of remote access trojans (RATs) and other unwanted applications.

This is key to preventing threat actors looking to deploy BlackLotus, which requires either remote administrative privileges on a target machine or physical access to the device. Organizations should implement defense-in-depth strategies to minimize the risk of threat actors gaining access and an administrative foothold in the environment. This can include detection and/or prevention at multiple stages prior to deployment of BlackLotus:

- - A threat actor gaining initial access via phishing, perimeter device compromise, or other vectors
 - A threat actor compromising user or service account credentials on the network
 - A threat actor moving laterally through the network using unusual or unauthorized accounts, abusing remote access software, or other mechanisms
 - A threat actor escalating and gaining domain or local administrative privileges
 - A threat actor creating malicious files on disk, including the BlackLotus installers or EFI files
- Customers should keep antimalware products up to date. Customers utilizing automatic updates for Microsoft Defender Antivirus do not need to take additional action. Enterprise customers managing updates should select the detection build **383.1029.0** or newer and deploy it across their environments.
- [Remove the Microsoft 3rd Party UEFI CA from your system's UEFI Secure boot configuration if this is not required for your system to boot](#). Performing this step blocks BlackLotus from working but **does not** eliminate the vulnerability.

References

- [BlackLotus UEFI bootkit: Myth confirmed](#) (ESET)
- [Malware dev claims to sell BlackLotus new Windows UEFI bootkit](#) (BleepingComputer)
- [Secure Boot Security Feature Bypass Vulnerability](#)