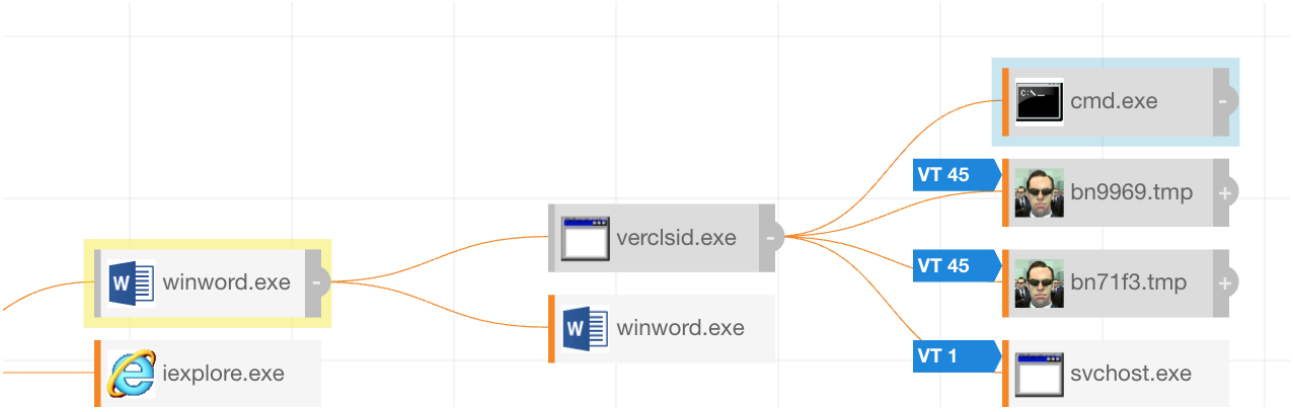
Phishing Attacks Using Verclsid.exe: Threat Detection

By Keshia LeVan, Michael Haag

Archived: 2026-04-05 18:37:15 UTC

Phishing is not exactly a new or groundbreaking attack method, but it's an ongoing problem (likely because it's effective and we all need e-mail). A wave of Hancitor malware spam campaigns recently hit many organizations. It's your typical pattern: a non-descriptively named Microsoft Word document sent with email subject lines like "USPS" or "eFax" using macros and heavily obfuscated VBScript to initiate the download of a malicious payload. What's interesting is not that an Office document is yet again being used to do something untoward…what's notable is the method being used. Normally you would expect to see something like PowerShell, WScript, or maybe just a plain old command prompt. Instead, a completely different process is launched: verclsid.exe.

What Is Verclsid.exe?



If you're like us, you probably never heard of verclsid.exe until malware started using it. According to Microsoft, this program was created to "validate shell extensions before they are instantiated by the Windows shell or Windows Explorer." We're not entirely sure what exactly this means, but we think the gist was to check COM extensions to see if they would crash explorer.exe.

It was released back in 2006 to mitigate a Remote Code Execution Vulnerability (MS06-015 – 908531) for XP and Server 2003 era systems. However, just to be clear, this isn't just running on XP systems; the process has persisted into modern operating systems. To be sure, we checked the file system and sure enough found verclsid.exe alive and well on our Windows 10 host. Though it's not explicitly stated why, there is a whole lot of backwards compatibility that Microsoft does under the surface. This can be both good and bad depending on what you're trying to do, so banning the process outright is probably not the best plan.

Regardless of its true purpose, what is clear is that verclsid.exe can do things like initiate network connections and download and write files to disk, as evidenced by the below activity, which was observed on a compromised host:

What to Do Next: Investigating Endpoint Data

So we know that verclsid.exe can be used nefariously, but that in and of itself is not terribly helpful. You could try to read the documentation on it, but you'd probably quickly discover that the Microsoft information on this particular process isn't incredibly voluminous. The question, then, is "What's next?" Maybe the real goal is not in knowing absolutely everything a process can do, but what does it normally do? We attempted to see if there were any outliers that only happen when it is being used to download and execute malware. Luckily for our investigative purposes, we had a fairly decent dataset of what abnormal behavior looked like across several customers that had been hit by this Hancitor variant, so we already probably had the outliers; we just needed to determine what they were and if they could be used to create a useful detection method.

In this case, having a good amount of diverse endpoint data can give us a lot of insight into a random process—in this case verclsid.exe. To start things off we looked at what verclsid.exe does most of the time, in a single environment where no malicious activity had been observed, with 193,000+ total executions of the process. The following table depicts the range of activity we found.[table id=1 /]

A couple things immediately jump out. First, this process never makes a network connection and it never launches any child processes. Filemod is a potential candidate, but since we're only looking at range data here and not frequency, we don't know right off the bat if it's very common for verclsid.exe to only write one filemod every time.

Let’s go back to our known bad example. We can see that it has a total of five network connections and three child processes. Likely, there’s something we can use here. A lot of what we do at Red Canary is to view things from the lens of “How do we detect this activity?” In this case, making a rule like “anytime verclsid.exe initiates a network connection, let me know” seems like a good start. Since it almost never launches network connections in normal day-to-day use, that rule will likely also have a low false positive rate (which is always nice).

A second option for detection is to look at what process launches verclsid.exe. If it’s almost never a Microsoft Office binary, that could be a good detection rule as well.

Detection is all well and good, but most folks would rather that the ‘badness’ never touched their systems in the

first place, and if the payload isn't delivered as part of the phishing e-mail, it has to be gotten from elsewhere (i.e., the web). So the next question is: can you stop verclsid.exe from making a network connection in the first place?

How to Prevent Verclsid.exe From Malicious Activity

When verclsid.exe is used for its intended purpose, it doesn't need to make a network connection (at least in our sample set), so why not put something in place that prevents this? That way, even if the initial drop on a system is successful, the subsequent download simply will not happen. Turns out, you can do this—and it doesn't require any high tech expensive tool. Simply go back to the Windows Firewall conveniently built in into any (or nearly any) modern version of Windows.

Add the following rule (and a second one for '%SystemRoot%\system32\verclsid.exe') to prevent this process from talking outbound.

```
netsh advfirewall firewall add rule name "Block Egress Verclsid" dir=out
program="%SystemRoot%\syswow64\verclsid.exe" enable=yes action=block profile=any
```

This can of course be implemented via GPO.

Key Takeaways for Defenders

This is just one example of a way that defenders can (at least partially) mitigate attacks that are very likely to get through perimeter defenses. After all, most organizations need (or prefer) Microsoft Office. (Mac users, you're not exactly safe here either; while vbscript and verclsid.exe are not an issue, you have osascript and python, so…)

The important takeaway is not necessarily to focus specifically on verclsid.exe, as implementing one specific rule will not fix everything. The key lessons are that (a) there are ways to identify when a process is doing something and alert on it, and; (b) sometimes you can use tools (sometimes even built-in OS ones) to mitigate the impact of attacks that do make it past the perimeter.

Source: https://redcanary.com/blog/verclsid-exe-threat-detection/