Matt Nelson Follow
Red Teamer | Security Researcher | Enjoys abusing features | Tweets are my own |
http://github.com/enigma0x3
Aug 3, 2017 · 3 min read

# WSH Injection: A Case Study

At BSides Nashville 2017, Casey Smith (@SubTee) and I gave a talk titled Windows Operating System Archaeology. At this talk, we released a handful of offensive techniques that utilized the Component Object Model (COM) in Windows. One such technique described was abusing attacker controlled input passed to calls to GetObject(), which I will be discussing here.

Some environments use whitelisting to prevent unsigned Windows Scripting Host (WSH) files from running, especially with the rise of malicious .js or .vbs files. However, by “injecting” our malicious code into a Microsoft signed WSH script, we can bypass such a restriction.

Before diving into the different scripts that can be used for injection, it’s important to understand some of the mechanics behind why this works. When abusing injection, we are taking advantage of attacker controlled input passed to GetObject() and then combining that with the “script:” or “scriptlet:” COM monikers.

## GetObject()

This method allows you to access an already instantiated COM object. If there isn’t an instance of the object already (if invoked without a moniker), this call will fail. For example, accessing Microsoft Excel’s COM object via GetObject() would look like this:

```
Set obj = GetObject( , "Excel.Application")
```

For the above to work, an instance of Excel has to be running. You can read more about GetObject() here.

## COM Monikers

While GetObject() is interesting by itself, it only allows us to access an instance of an already instantiated COM object. To get around this, we can implement a COM moniker to facilitate our payload execution. If you aren’t familiar with COM monikers, you can read more about them here. There are various COM monikers on Windows that allow you to instantiate objects in various ways. From an offensive standpoint, you can use these monikers to execute malicious code. That is a topic for another blog post :-).

For this post, we will focus on the "script:" and "scriptlet:" monikers. These particular monikers interface with scrobj.dll and help facilitate execution of COM scriptlets, which will be the payload. This was discovered by Casey Smith (@SubTee) and discussed at DerbyCon 2016 as well as blogged about here.

An example COM scriptlet will look like this:

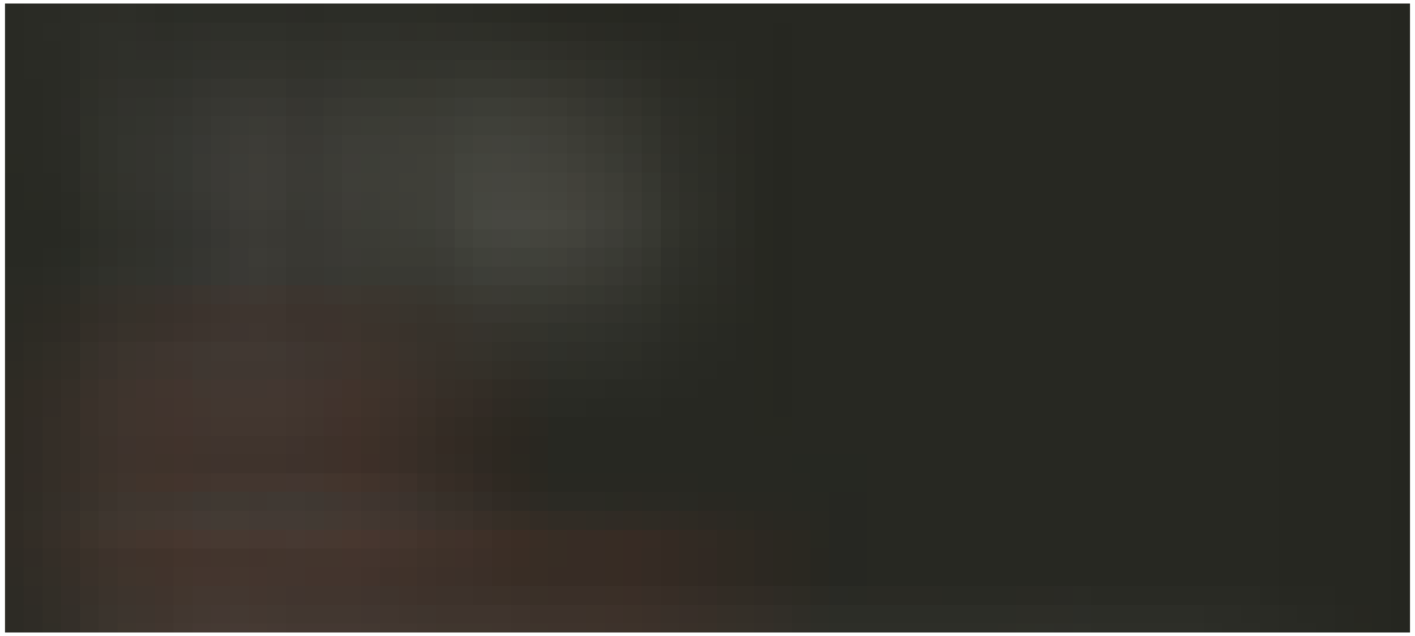
```
<?XML version="1.0"?>
var r = new ActiveXObject("WScript.Shell").Run("calc.exe");
]]>
</scriptlet>
```

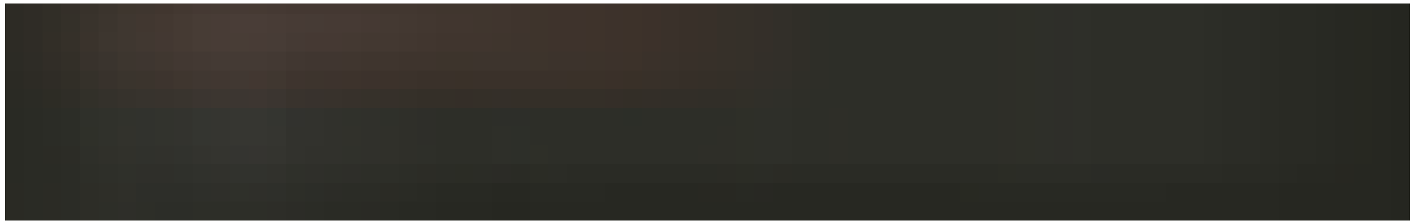
You can also use James Forshaw's (@tiraniddo) tool DotNetToJScript to extend the JScript/VBScript in the COM Scriptlet, allowing for Win32 API access and even Shellcode execution. When you combine one of these two monikers and various calls to GetObject(), a lot of fun is had.

Now that the very brief COM background is over, time to look at an example ☺

**PubPrn.vbs**

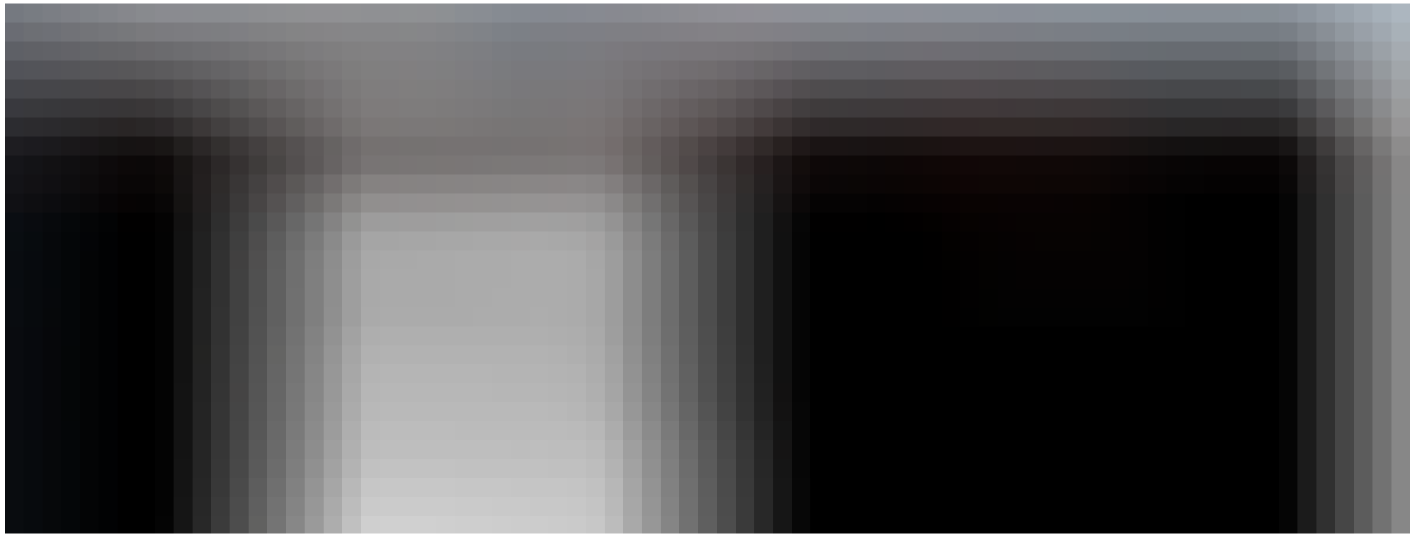
On Windows 7+, there is a Microsoft Signed WSH script called “PubPrn.vbs,” which resides in “C:\Windows\System32\Printing_Admin_Scripts\en-US”. When looking at this particular script, it becomes apparent that it is taking input provided by the user (via command line arguments) and passing an argument to “GetObject()”.

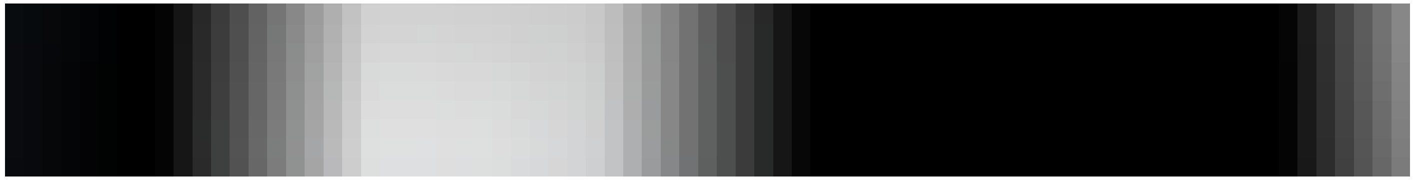




This means that we can run this script and pass it the two arguments it expects. The first argument can be anything and the second argument is the payload via the script: moniker.

Note: If you provide a value that isn't a network address for the first argument (since it expects a ServerName), you can add the "/b" switch to cscript.exe when calling to suppress any additional error messages.





Since VBScript relies on COM to perform actions, it is used heavily in numerous Microsoft signed scripts. While this is just one example, there are bound to be others that can be exploited in a similar fashion. I encourage you to go hunting ☺

. . .

*Originally published at enigma0x3.net on August 3, 2017.*

Script Vbscript Red Team Research

**One clap, two clap, three clap, forty?**

By clapping more or less, you can signal to us which stories really stand out.

13

**Matt Nelson** Follow

**Posts By SpecterOps Team** Follow

Red Teamer | Security Researcher | Enjoys abusing features | Tweets are my own | http://github.com/enigma0x3

## Members

Posts from SpecterOps team members on various topics relating information security

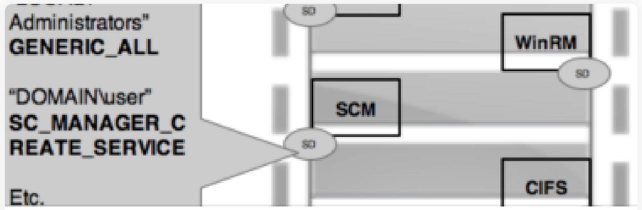
More from Posts By SpecterOps Team Members

### A Push Toward Transparency

Information security is a young field, which is still rapidly evolving compared to…

David McGuire
6 min read

32

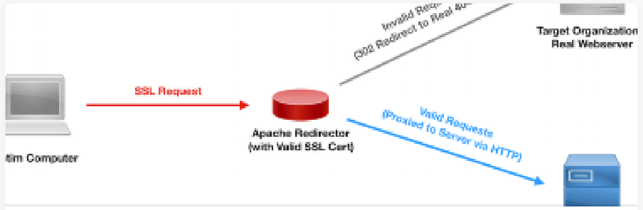


More from Posts By SpecterOps Team Members

### Remote Hash Extraction On Demand Via Host Security…

Will
13 min read

60



Target Organization Real Webserver
SSL Request
Apache Redirector (with Valid SSL Cert)
Valid Requests (Proxied to Server via HTTP)
tim Computer

More from Posts By SpecterOps Team Members

### HTTPS Payload and C2 Redirectors

Jeff Dimmock
10 min read

40

## Responses

Write a response…