Icon

Description automatically generated

Practical Malware Analysis & Triage

Malware Analysis Report

RAT.Unknown.exe.malz

PMAT Class Final

Nov 2024

# Table of Contents

[Table of Contents 2](#_Toc183337435)

[Executive Summary 3](#_Toc183337436)

[High-Level Technical Summary 4](#_Toc183337437)

[Malware Composition 5](#_Toc183337438)

[RAT.Unknown.exe 5](#_Toc183337439)

[mscordll.exe: 5](#_Toc183337440)

[Basic Static Analysis 6](#_Toc183337441)

[VirusTotal / Signature 6](#_Toc183337442)

[String Analysis 6](#_Toc183337443)

[Structure of File 6](#_Toc183337444)

[Basic Dynamic Analysis 6](#_Toc183337445)

[Advanced Static Analysis 12](#_Toc183337446)

[Advanced Dynamic Analysis 13](#_Toc183337447)

[The NO SOUP Kill Switch 13](#_Toc183337448)

[Bind Server 13](#_Toc183337449)

[Indicators of Compromise 15](#_Toc183337450)

[Network Indicators 15](#_Toc183337451)

[Host-based Indicators 15](#_Toc183337452)

[Rules & Signatures 16](#_Toc183337453)

# Executive Summary

|  |  |
| --- | --- |
| MD5 hash | 689ff2c6f94e31abba1ddebf68be810e |
| SHA1 hash | 69b8ecf6b7cde185daed76d66100b6a31fd1a668 |
| SHA256 hash | 248d491f89a10ec3289ec4ca448b19384464329c442bac395f680c4f3a345c8c |

RAT\_Unknown is a bind shell that allows remote commands to be executed via a TCP connection to port 5555 and returns the command result text in base64 encoding.

At logout/reboot the original malware process will not persist. So, to maintain persistence, during initial detonation it reaches out to download a resource called “msdcorelib.exe” and writes it into the current user’s startup folder (names it to “mscordll.exe” in the file system).

If the attempt to connect to the resource server via HTTP fails a message box is displayed, “NO SOUP FOR YOU”, and the process instead exits without establishing the bind shell listener.

# High-Level Technical Summary

RAT.Unknown.exe is the initial stage that tries to download a second item “msdcorelib.exe” which is stored, renamed to “mscordll.exe”. The initial executable also establishes a bind shell for remote command execution on port 5555.

User runs RAT.Unknown.exe

No

No

Yes

Yes

Exit

Return output in base64

Listen on port 5555 for command

Write mscordll.exe to startup folder

Response

HTTP GET hxxp://serv1.ec2-102-95-13-2-ubuntu.local/msdcorelib.exe

Response

NO SOUP FOR YOU

Message

HTTP GET hxxp://serv1.ec2-102-95-13-2-ubuntu.local

# Malware Composition

RAT.Unknown.exe consists of the following components:

|  |  |
| --- | --- |
| File Name | SHA256 Hash |
| RAT.Unknown.exe | 248d491f89a10ec3289ec4ca448b19384464329c442bac395f680c4f3a345c8c |
| mscordll.exe | Not captured – would be download from server at hxxp://serv1.ec2-102-95-13-2-ubuntu.local |

## RAT.Unknown.exe

The initial executable. Establishes an initial bind shell listening on 5555, but also downloads the second stage for persistence and adds it to the current user’s startup folder.

mscordll.exe:

Presumably the persistence mechanism. Downloaded as msdcorelib.exe but saved as mscordll.exe. Not analyzed as we don’t have the real server.

# Basic Static Analysis

## VirusTotal / Signature

At the time of writing, this malware’s signature was reported as malicious by 45/73 vendors on VirusTotal.

### String Analysis

The following suspicious/significant strings were detected:

@[+] what command can I run for you

@[+] online

@NO SOUP FOR YOU

@\mscordll.exe

@Nim httpclient/1.0.6

@AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup

@hxxp://serv1.ec2-102-95-13-2-ubuntu.local

Of particular interest is the URI (defanged by changing “tt” to “xx” in the URL):

hxxp://serv1.ec2-102-95-13-2-ubuntu.local

Multiple strings with “nim” detected.

## Structure of File

The EXE is a 64-bit windows PE file (first two bytes of the file contain the “MZ” signature).

Some of the suspicious imports include: GetCurrentProcess | GetCurrentProcessId | GetCurrentThreadId | VirtualAlloc | VirtualProtect

# Basic Dynamic Analysis

Run normally with no inetsim on the analysis network, no files are written.

In this scenario an error box is displayed:

A screenshot of a computer error

Description automatically generated

DNS Query to the URL identified in the string analysis:

A screenshot of a computer

Description automatically generated

Followed by an outgoing HTTP/TCP connection to port 80:A screenshot of a computer

Description automatically generated

Correlating information from procmon during connection.

A screen shot of a computer screen

Description automatically generated

Wireshark capture of initial HTTP GET attempt:

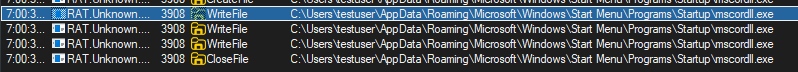
A screenshot of a computer

Description automatically generated

If a successful HTTP get is made to the URI, a second HTTP GET is made for resource “msdcorelib.exe”:A screenshot of a computer

Description automatically generated

This is then written to the filesystem – see procmon output:



Resulting in a file being written to the Start Menu\Programs\Startup directory:

A screenshot of a computer

Description automatically generated

Verified the written file was obtained from the GET – the inetsim stub is the file we found written:

A screenshot of a computer error

Description automatically generated

After successful deployment, the process remains listening on port 5555:

A screenshot of a computer

Description automatically generated

A client connecting to port 5555 is presented with a base64 response:

A screen shot of a computer

Description automatically generated

Which decodes to:

[+] what command can I run for you

Submitting text that matches an executable found on the infected box causes this executable to run and the resulting output returned (in base64). If the command is not found:

VGhlIHN5c3RlbSBjYW5ub3QgZmluZCB0aGUgZmlsZSBzcGVjaWZpZWQuDQpBZGRpdGlvbmFsIGluZm86ICJSZXF1ZXN0ZWQgY29tbWFuZCBub3QgZm91bmQ6IFwnaWRcJy4gT1MgZXJyb3I6Ig==

Which decodes to:

The system cannot find the file specified.

Additional info: "Requested command not found: \'id\'. OS error:"

At login, mscordll.exe is run because it has been added to the Startup folder. Verified the inetsim stub was added there and runs at login.

# Advanced Static Analysis

Analysis in Cutter confirms development in Nim with sections:

A group of black text

Description automatically generated

The NimMainModule calls two main routines to perform the download, write it out, and start a server. Addresses of these modules are below for use in setting breakpoints in advanced dynamic analysis.

A screenshot of a computer program

Description automatically generated

* downloadToStartup\_\_YnywBc1swkyMbNJ9b4UuShA is located at relative address: 0x004144a6
* startServer\_\_YnywBc1swkyMbNJ9b4UuShA\_2 is located at relative address: 0x004146d1

# Advanced Dynamic Analysis

## The NO SOUP Kill Switch

Downloading the payload “msdcorelib.dll” happens in this part of the disassembled code:

A screen shot of a computer

Description automatically generated

Instruction at relative offset 0x00414CD8 calls out to the code that attempts the HTTP connections. If we want to analyze the bind shell functionality without allowing the download, we can fill the 5 bytes starting at 0x00414CD8 with NOOP instructions (0x90 bytes).

Reaching instruction at 0x00414CDD offset will call the section of code (symbol in cutter names this “startServer\_\_”) and will set up listening on port 5555 for commands even though the payload was not downloaded:

A screenshot of a computer

Description automatically generated

## Bind Server

Areas of the disassembled code related to the command server are:

* Main entry port for starting the server: 0x004146d1
* Receiving a line from the port: 0x0040deae
* Encoding text before sending to the connected control program: 0x0040e780

The port number to listen on is hard-coded here with the mov instruction at relative offset 0x00414723 – it stores the port number (0x15B3 = 5555 dec) into the EDX before calling the code that establishes the socket.

A screen shot of a computer

Description automatically generated

# Indicators of Compromise

(See Analysis sections for screenshots)

## Network Indicators

* HTTP GET request to hxxp://serv1.ec2-102-95-13-2-ubuntu.local
* HTTP GET request to hxxp://serv1.ec2-102-95-13-2-ubuntu.local/msdcorelib.exe

## Host-based Indicators

* Execution of malware with no response to HTTP requests results in “NO SOUP FOR YOU” message box
* Execution of malware with inet simulation/response to HTTP requests results in a new file, mscordll.exe, written to the users Startup folder, “AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup”
* With successful detonation the original process “RAT.Unknown.exe” will continue to run and will maintain an open port 5555 listening for TCP connections.

# Rules & Signatures

A yara file that will detect RAT.Unknown.exe:

rule RAT\_Unknown\_Sample {

    meta:

        last\_updated = "20224-11-24"

        author = "PMAT"

        description = "Rule for PMAT example \"RAT.Unknown.exe\""

    strings:

        $pe\_magic\_bytes = { 4D 5A }

        $no\_soup\_string = "NO SOUP FOR YOU"

        $payload\_server\_name = "serv1.ec2-102-95-13-2-ubuntu.local"

    condition:

        $pe\_magic\_bytes at 0 and  // must be a PE

        $no\_soup\_string and       // contains the "NO SOUP" message

        $payload\_server\_name      // payload download server

}