Threat Intelligence

New Targeted Attack in the Middle East by APT34, a Suspected Iranian Threat Group, Using CVE-2017-11882 Exploit

December 7, 2017

Mandiant

Written by: Manish Sardiwal, Vincent Cannon, Nalani Fraser, Yogesh Londhe, Nick Richard, Jacqueline O'Leary

Less than a week after Microsoft issued a patch for CVE-2017-11882 on Nov. 14, 2017, FireEye observed an attacker using an exploit for the Microsoft Office vulnerability to target a government organization in the Middle East. We assess this activity was carried out by a suspected Iranian cyber espionage threat group, whom we refer to as APT34, using a custom PowerShell backdoor to achieve its objectives.

We believe APT34 is involved in a long-term cyber espionage operation largely focused on reconnaissance efforts to benefit Iranian nation-state interests and has been operational since at least 2014. This threat group has conducted broad targeting across a variety of industries, including financial, government, energy, chemical, and telecommunications, and has largely focused its operations within the Middle East. We assess that APT34 works on behalf of the Iranian government based on infrastructure details that contain references to Iran, use of Iranian

infrastructure, and targeting that aligns with nation-state interests.

APT34 uses a mix of public and non-public tools, often conducting spear phishing operations using compromised accounts, sometimes coupled with social engineering tactics. In May 2016, we published a blog detailing a spear phishing campaign targeting banks in the Middle East region that used macro-enabled attachments to distribute POWBAT malware. We now attribute that campaign to APT34. In July 2017, we observed APT34 targeting a Middle East organization using a PowerShell-based backdoor that we call POWRUNER and a downloader with domain generation algorithm functionality that we call BONDUPDATER, based on strings within the malware. The backdoor was delivered via a malicious .rtf file that exploited CVE-2017-0199.

In this latest campaign, APT34 leveraged the recent Microsoft Office vulnerability CVE-2017-11882 to deploy POWRUNER and BONDUPDATER.

The full report on APT34 is available to our MySIGHT customer community. APT34 loosely aligns with public reporting related to the group "OilRig". As individual organizations may track adversaries using varied data sets, it is possible that our classifications of activity may not wholly align.

CVE-2017-11882: Microsoft Office Stack Memory Corruption Vulnerability

CVE-2017-11882 affects several versions of Microsoft Office and, when exploited, allows a remote user to run arbitrary code in the context of the current user as a result of improperly handling objects in memory. The vulnerability was patched by Microsoft on Nov. 14, 2017. A full proof of concept (POC) was publicly released a week later by the reporter of the vulnerability.

The vulnerability exists in the old Equation Editor (EQNEDT32.EXE), a component of Microsoft Office that is used to insert and evaluate mathematical formulas. The Equation Editor is embedded in Office documents using object linking and embedding (OLE) technology. It is created as a separate process instead of child process of Office applications. If a crafted formula is passed to the Equation Editor, it does not check the data length properly while copying the data, which results in stack memory corruption. As the EQNEDT32.exe is compiled using an older compiler and does not support address space layout randomization (ASLR), a technique

that guards against the exploitation of memory corruption vulnerabilities, the attacker can easily alter the flow of program execution.

Analysis

APT34 sent a malicious .rtf file (MD5: a0e6933f4e0497269620f44a083b2ed4) as an attachment in a malicious spear phishing email sent to the victim organization. The malicious file exploits CVE-2017-11882, which corrupts the memory on the stack and then proceeds to push the malicious data to the stack. The malware then overwrites the function address with the address of an existing instruction from EQNEDT32.EXE. The overwritten instruction (displayed in Figure 1) is used to call the "WinExec" function from kernel32.dll, as depicted in the instruction at 00430c12, which calls the "WinExec" function.

Figure 1: Disassembly of overwritten function address

After exploitation, the 'WinExec' function is successfully called to create a child process, "mshta.exe", in the context of current logged on user. The process "mshta.exe" downloads a malicious script from hxxp://mumbai-m[.]site/b.txt and executes it, as seen in Figure 2.

Figure 2: Attacker data copied to corrupt stack buffer

Execution Workflow

The malicious script goes through a series of steps to successfully execute and ultimately establish a connection to the command and control (C2) server. The full sequence of events starting with the exploit document is illustrated in Figure 3.

Figure 3: CVE-2017-11882 and POWRUNER attack sequence

1. The malicious .rtf file exploits CVE-2017-11882.
2. The malware overwrites the function address with an existing

instruction from EQNEDT32.EXE.

3. The malware creates a child process, "mshta.exe," which downloads a file from: hxxp://mumbai-m[.]site/b.txt.
4. b.txt contains a PowerShell command to download a dropper from: hxxp://dns-update[.]club/v.txt. The PowerShell command also renames the downloaded file from v.txt to v.vbs and executes the script.
5. The v.vbs script drops four components (hUpdateCheckers.base, dUpdateCheckers.base, cUpdateCheckers.bat, and GoogleUpdateschecker.vbs) to the directory: C:\ProgramData\Windows\Microsoft\java\
6. v.vbs uses CertUtil.exe, a legitimate Microsoft command-line program installed as part of Certificate Services, to decode the base64-encoded files hUpdateCheckers.base and dUpdateCheckers.base, and drop hUpdateCheckers.ps1 and dUpdateCheckers.ps1 to the staging directory.
7. cUpdateCheckers.bat is launched and creates a scheduled task for GoogleUpdateschecker.vbs persistence.
8. GoogleUpdateschecker.vbs is executed after sleeping for five seconds.
9. cUpdateCheckers.bat and *.base are deleted from the staging directory.

Figure 4 contains an excerpt of the v.vbs script pertaining to the Execution Workflow section.

Figure 4: Execution Workflow Section of v.vbs

After successful execution of the steps mentioned in the Execution Workflow section, the Task Scheduler will launch GoogleUpdateschecker.vbs every minute, which in turn executes the dUpdateCheckers.ps1 and hUpdateCheckers.ps1 scripts. These PowerShell scripts are final stage payloads – they include a downloader with domain generation algorithm (DGA) functionality and the backdoor component, which connect to the C2 server to receive commands and perform additional malicious activities.

hUpdateCheckers.ps1 (POWRUNER)

The backdoor component, POWRUNER, is a PowerShell script that sends and receives commands to and from the C2 server.

POWRUNER is executed every minute by the Task Scheduler. Figure 5 contains an excerpt of the POWRUNER backdoor.

Figure 5: POWRUNER PowerShell script hUpdateCheckers.ps1

POWRUNER begins by sending a random GET request to the C2 server and waits for a response. The server will respond with either "not_now" or a random 11-digit number. If the response is a random number, POWRUNER will send another random GET request to the server and store the response in a string. POWRUNER will then check the last digit of the stored random number response, interpret the value as a command, and perform an action based on that command. The command values and the associated actions are described in Table 1.

Command	Description	Action
0	Server response string contains batch commands	Execute batch commands and send results back to server
1	Server response string is a file path	Check for file path and upload (PUT) the file to server
2	Server response string is a file path	Check for file path and download (GET) the file

Table 1: POWRUNER commands

After successfully executing the command, POWRUNER sends the results back to the C2 server and stops execution.

The C2 server can also send a PowerShell command to capture and store a screenshot of a victim's system. POWRUNER will send the captured screenshot image file to the C2 server if the "fileupload" command is issued. Figure 6 shows the PowerShell "Get-Screenshot" function sent by the C2 server.

Figure 6: PowerShell Screenshot Functionality

dUpdateCheckers.ps1 (BONDUPDATER)

One of the recent advancements by APT34 is the use of DGA to generate subdomains. The BONDUPDATER script, which was named based on the hard-coded string "B007", uses a custom DGA algorithm to generate subdomains for communication with the C2 server.

DGA Implementation

Figure 7 provides a breakdown of how an example domain (4563419213000006B0C8B2CE9C9B007.mumbai-m[.]site) is generated using BONDUPDATER's custom DGA.

Figure 7: Breakdown of subdomain created by BONDUPDATER

1. This is a randomly generated number created using the following expression: $rnd = -join (Get-Random -InputObject (10..99) -Count (%{ Get-Random -InputObject (1..6)}));
2. This value is either 0 or 1. It is initially set to 0. If the first resolved domain IP address starts with 24.125.X.X, then it is set to 1.
3. Initially set to 000, then incremented by 3 after every DNS request
4. First 12 characters of system UUID.
5. "B007" hardcoded string.
6. Hardcoded domain "mumbai-m[.]site"

BONDUPDATER will attempt to resolve the resulting DGA domain and will take the following actions based on the IP address resolution:

1. Create a temporary file in %temp% location
    - The file created will have the last two octets of the resolved IP addresses as its filename.
2. BONDUPDATER will evaluate the last character of the file name and perform the corresponding action found in Table 2.

Character	Description

0	File contains batch commands, it executes the batch commands
1	Rename the temporary file as .ps1 extension
2	Rename the temporary file as .vbs extension

Table 2: BONDUPDATER Actions

Figure 8 is a screenshot of BONDUPDATER's DGA implementation.

Figure 8: Domain Generation Algorithm

Some examples of the generated subdomains observed at time of execution include:

143610035BAF04425847B007.mumbai-m[.]site

835710065BAF04425847B007.mumbai-m[.]site

376110095BAF04425847B007.mumbai-m[.]site

Network Communication

Figure 9 shows example network communications between a POWRUNER backdoor client and server.

Figure 9: Example Network Communication

In the example, the POWRUNER client sends a random GET request to the C2 server and the C2 server sends the random number (99999999990) as a response. As the response is a random number that ends with '0', POWRUNER sends another random GET request to receive an additional command string. The C2 server sends back Base64 encoded response.

If the server had sent the string "not_now" as response, as shown in Figure 10, POWRUNER would have ceased any further requests and terminated its execution.

Figure 10: Example "not now" server response

Batch Commands

POWRUNER may also receive batch commands from the C2 server to collect host information from the system. This may include information about the currently logged in user, the hostname, network configuration data, active connections, process information, local and domain administrator accounts, an enumeration of user directories, and other data. An example batch command is provided in Figure 11.

Figure 11: Batch commands sent by POWRUNER C2 server

Additional Use of POWRUNER / BONDUPDATER

APT34 has used POWRUNER and BONDUPDATER to target Middle East organizations as early as July 2017. In July 2017, a FireEye Web MPS appliance detected and blocked a request to retrieve and install an APT34 POWRUNER / BONDUPDATER downloader file. During the same month, FireEye observed APT34 target a separate Middle East organization using a malicious .rtf file (MD5: 63D66D99E46FB93676A4F475A65566D8) that exploited CVE-2017-0199. This file issued a GET request to download a malicious file from:

hxxp://94.23.172.164/dupdatechecker.doc.

As shown in Figure 12, the script within the dupatechecker.doc file attempts to download another file named dupatechecker.exe from the same server. The file also contains a comment by the malware author that appears to be an apparent taunt to security researchers.

Figure 12: Contents of dupdatechecker.doc script

The dupatechecker.exe file (MD5: C9F16F0BE8C77F0170B9B6CE876ED7FB) drops both

BONDUPDATER and POWRUNER. These files connect to proxychecker[.]pro for C2.

Outlook and Implications

Recent activity by APT34 demonstrates that they are capable group with potential access to their own development resources. During the past few months, APT34 has been able to quickly incorporate exploits for at least two publicly vulnerabilities (CVE-2017-0199 and CVE-2017-11882) to target organizations in the Middle East. We assess that APT34's efforts to continuously update their malware, including the incorporation of DGA for C2, demonstrate the group's commitment to pursing strategies to deter detection. We expect APT34 will continue to evolve their malware and tactics as they continue to pursue access to entities in the Middle East region.

IOCs

Filename / Domain / IP Address	**MD5 Hash or Description**
CVE-2017-11882 exploit document	A0E6933F4E0497269620F44A08
b.txt	9267D057C065EA7448ACA1511C6
v.txt/v.vbs	B2D13A336A3EB7BD27612BE7D4E
dUpdateCheckers.base	4A7290A279E6F2329EDD0615178
hUpdateCheckers.base	841CE6475F271F86D0B5188E4F8E
cUpdateCheckers.bat	52CA9A7424B3CC34099AD21862
dUpdateCheckers.ps1	BBDE33F5709CB1452AB941C08A
hUpdateCheckers.ps1	247B2A9FCBA6E9EC29ED818948
GoogleUpdateschecker.vbs	C87B0B711F60132235D7440ADDC

hxxp://mumbai-m[.]site	POWRUNER C2
hxxp://dns-update[.]club	Malware Staging Server
CVE-2017-0199 exploit document	63D66D99E46FB93676A4F475A6
94.23.172.164:80	Malware Staging Server
dupdatechecker.doc	D85818E82A6E64CA185EDFDDBA
dupdatechecker.exe	C9F16F0BE8C77F0170B9B6CE87
proxycheker[.]pro	C2
46.105.221.247	Has resolved mumbai-m[.]site & hpserver[.]online
148.251.55.110	Has resolved mumbai-m[.]site and update[.]club
185.15.247.147	Has resolved dns-update[.]club
145.239.33.100	Has resolved dns-update[.]club
82.102.14.219	Has resolved ns2.dns-update[.]clu hpserver[.]online & anyportals[.]c
v7-hpserver.online.hta	E6AC6F18256C4DDE5BF06A9191
dUpdateCheckers.base	3C63BFF9EC0A340E0727E56834
hUpdateCheckers.base	EEB0FF0D8841C2EBE643FE328B
cUpdateCheckers.bat	FB464C365B94B03826E67EABE4
dUpdateCheckers.ps1	635ED85BFCAAB7208A8B5C730
hUpdateCheckers.ps1	13B338C47C52DE3ED0B68E1CB7
googleupdateschecker.vbs	DBFEA6154D4F9D7209C1875B2D

hpserver[.]online	C2
v7-anyportals.hta	EAF3448808481FB1FDBB675BC5E
dUpdateCheckers.base	42449DD79EA7D2B5B6482B6F0D
hUpdateCheckers.base	A3FCB4D23C3153DD42AC124B112
dUpdateCheckers.ps1	EE1C482C41738AAA5964730DCB/
hUpdateCheckers.ps1	E516C3A3247AF2F2323291A67008
anyportals[.]com	C2

Posted in Threat Intelligence—Security & Identity

Related articles

Threat Intelligence

UNC4393 Goes Gently into the SILENTNIGHT

By Mandiant • 21-minute read

Threat Intelligence

APT45: North Korea's Digital Military Machine

By Mandiant • 5-minute read

Threat Intelligence

Whose Voice Is It Anyway? AI-Powered Voice Spoofing for Next-Gen Vishing Attacks

By Mandiant • 8-minute read

Threat Intelligence

APT41 Has Arisen From the DUST

By Mandiant • 28-minute read

Follow us