URLZone Zones in on Japan

fireeye.com/blog/threat-research/2016/01/urlzone_zones_inon.html

Recently we've seen an interesting trend from several crimeware families that were mainly active in the European region, and have now expanded their activity to Japan. Rovnix is one such family, as recently reported by IBM X-Force.

At the same time, we've seen another spam campaign break out in Japan. The malware attempted to deliver another old banking trojan named URLZone (aka Shiotob/Bebloh), which was initially discovered in 2009. URLZone is known to be very active in the European region, especially Spain and Germany. Now we have noticed that the spam group is focusing on Japan.

This blog describes a URLZone spam campaign targeting Japan in December 2015. We discuss its new persistence and evasion techniques, as well as its well-known password stealing method and command and control (CnC) communication.

Spam Campaign

On Dec. 16, 2015, and Dec. 21, 2015, we saw an extensive amount of URLZone spam emails being delivered to Japanese email users. Figure 1 represents the spikes of URLZone spam activity we detected during that time.

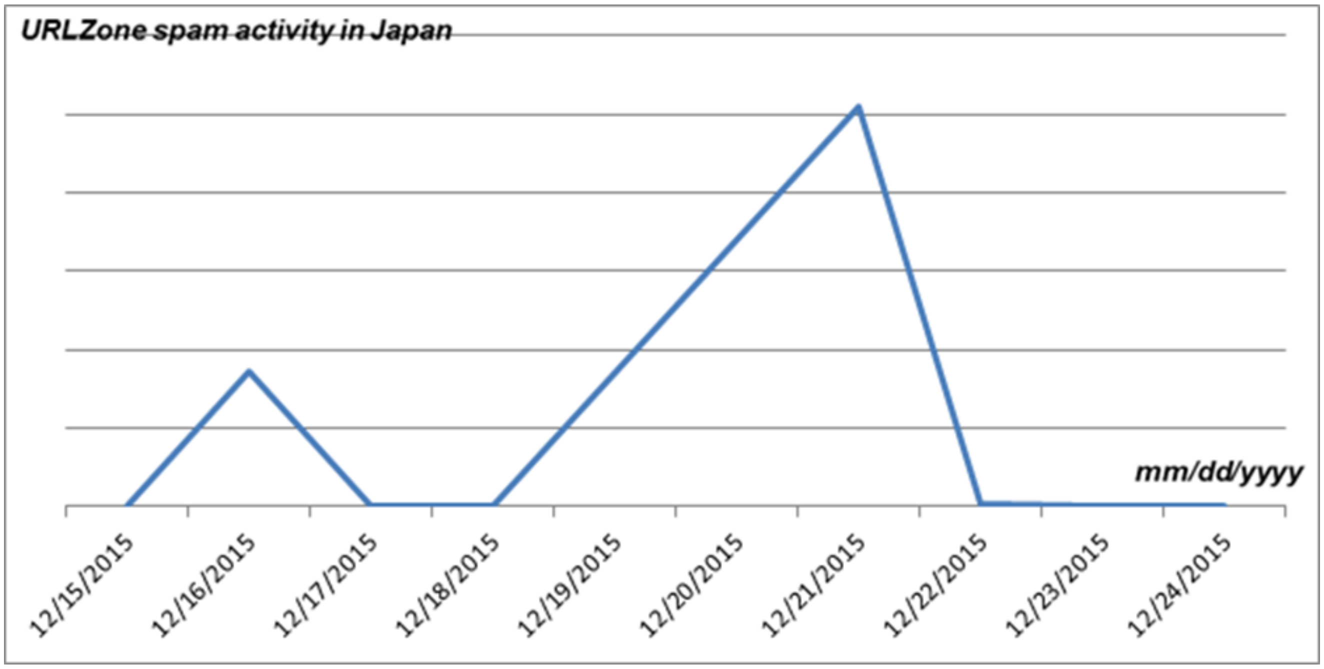
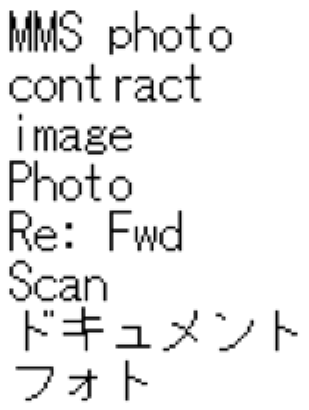


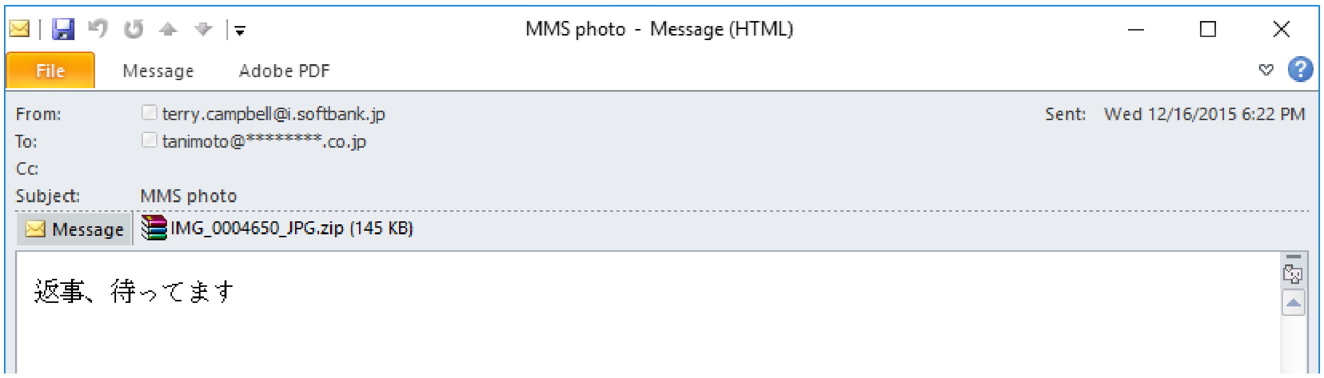
Figure 1. Timeline of URLZone activities

URLZone spam campaigns usually take place by targeting a specific region. The spam emails are crafted with the target region’s language, and are often sent using email account domains belonging to the target region. This increases the chance of recipients opening the malicious attachment, especially in non-English speaking regions, as the recipients are more familiar in exchanging emails in their native language.

The email subject and content were simple and generic. The subjects were written in English and Japanese with short Japanese sentences for the body, as shown in Figure 2.



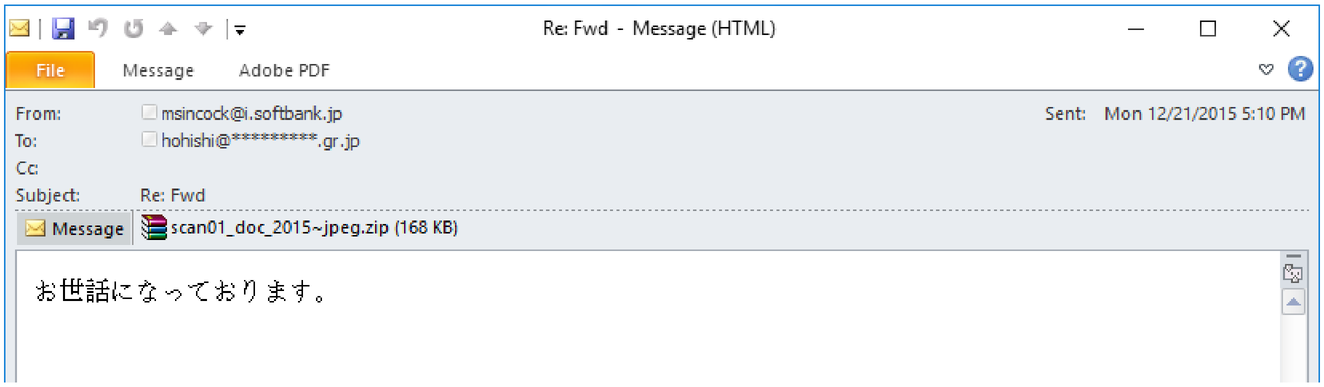
MMS photo
contract
image
Photo
Re: Fwd
Scan
ドキュメント
フォト



MMS photo - Message (HTML)

From: terry.campbell@i.softbank.jp
Sent: Wed 12/16/2015 6:22 PM
To: tanimoto@*********.co.jp
Cc:
Subject: MMS photo
Message IMG_0004650_JPG.zip (145 KB)

返事、待ってます



Re: Fwd - Message (HTML)

From: msincock@i.softbank.jp
Sent: Mon 12/21/2015 5:10 PM
To: hohishi@**********.gr.jp
Cc:
Subject: Re: Fwd
Message scan01_doc_2015~jpeg.zip (168 KB)

お世話になっております。

Figure 2. URLZone spam email samples

Most of the spam emails were sent from freely available web email accounts in Japan. The majority of the emails used the domains softbank.jp and yahoo.co.jp, which are the largest mobile carrier and web portal service in Japan respectively.

An attached ZIP archive containing URLZone binaries was given two extension names in order to disguise it as a DOC or JPG file and trick recipients into opening the malicious attachment. For example “scan01_doc_2015~jpeg.zip” extracts “scan01_doc_2015~jpeg.jpeg.exe”.

Malware Analysis

URLZone is a banking trojan. It downloads a configuration file that contains information on targeted financial institutions, and uses web injection techniques to steal a user’s banking credentials. While the basic characteristics of URLZone samples in the campaign in Japan remained the same as the previous analysis done by Arbor Networks, several new features were added to the latest URLZone sample.

Initial Infection Stage

The malware uses process hollowing (also known as process replacement) to mask its execution. The malware tries to hollow explorer.exe or iexplorer.exe with a **“_section”** as added command-line parameter to identify this process as spawned by the malware.

The process it hollows is initially started as suspended. It then modifies or writes its malicious code to the entry point of the hollowed process. Once the necessary code is written, it will resume the suspended process, thus executing its malicious payload.

Next, the malware does one of the following:

1. Continues to run the malicious routine in this hollowed process if the hollowed process is 64-bit or it has a window (this is for hollowed iexplore.exe).

2. Continues the malicious routine by injecting itself into the running explorer.exe on the system.

Stolen Information

System Survey

To identify the victim system, the malware acquires the following details:

- Computer name
- OS major/minor version, as well as install date
- Hollowed process name version and time stamp
- IP address
- Keyboard layout

This is sent out in a beacon POST to the CnC server as described in Arbor’s report.

Email Addresses

URLZone steals the email addresses stored in the Windows Address Book (WAB). It does so by querying both wab32.dll and WAB file name in the registry. It then uses this library to parse through the WAB file and save the information to a randomly generated value in a randomly generated key /SOFTWARE/<random>/<random_value>.

Web/FTP/Email Information and Credentials

The malware steals web and FTP information by injecting malicious code into commonly used programs for connectivity. It injects a specific malicious routine on each target program that hooks a certain library used to send or receive network traffic. A continuous thread is running that constantly checks for the presence of these applications, and concurrently injects a certain hooking function dependent of the process name. The hooking process is described in depth here.

- iexplore.exe – WinInet hooking

- explorer.exe – WinInet hooking
- myie.exe – WinInet hooking
- firefox.exe. – WinInet hooking
- ftpte.exe – WinSock hooking
- coreftp.exe - WinSock hooking
- filezilla.exe - WinSock hooking
- TOTALCMD.EXE - WinSock hooking
- cftp.exe - WinSock hooking
- FTPVoyager.exe - WinSock hooking
- SmartFTP.exe - WinSock hooking
- WinSCP.exe - WinSock hooking
- chrome.exe – WinInet hooking
- opera.exe – WinInet hooking

For FTP/Email applications (that do WinSock hooking), it hooks 3 APIs from ws2_32.dll:

- ws2_32_send
- ws2_32_connect
- ws2_32_close

It monitors FTP/MAIL transactions through ws2_32_connect by checking for the string “FTP” and “MAIL” on the connect parameters. The FTP/MAIL server address and connection handle is stolen from this hook. The ws2_32_send hook captures the authentication request by checking the strings “USER” and “PASS” to steal the user’s credentials.

For Web applications that use WinInet, it consists of API hooks used for monitoring HTTP/S sessions using the hooks shown in the appendix.

These hooks look for strings as specified in the malware’s configuration file, and these strings target the data from financial institutions. If a match is found, the malware sends out the information to its CnC server.

Command and Control

URLZone uses a Domain Generation Algorithm (DGA), as stated in other reports. The initial CnC URL starts of as a hard-coded encrypted string within the malware body. If the hard-coded URL doesn't work, the malware then uses DGA to find the right one.

The malware checks for Internet connectivity by first connecting to google.com. It then proceeds to check, through SSLv3 handshake, whether the generated URL responds to its certificate. It continuously does this until a valid URL is found.

The DGA takes in the previous URL as a seed to generate other domains.

Persistence Mechanism

Unlike many other banking trojans, URLZone uses a clever persistence mechanism and clears its registry configuration only upon logoff, reboot, and shutdown.

It does this using a Window Procedure that monitors Window Messages for WM_QUERYENDSESSION

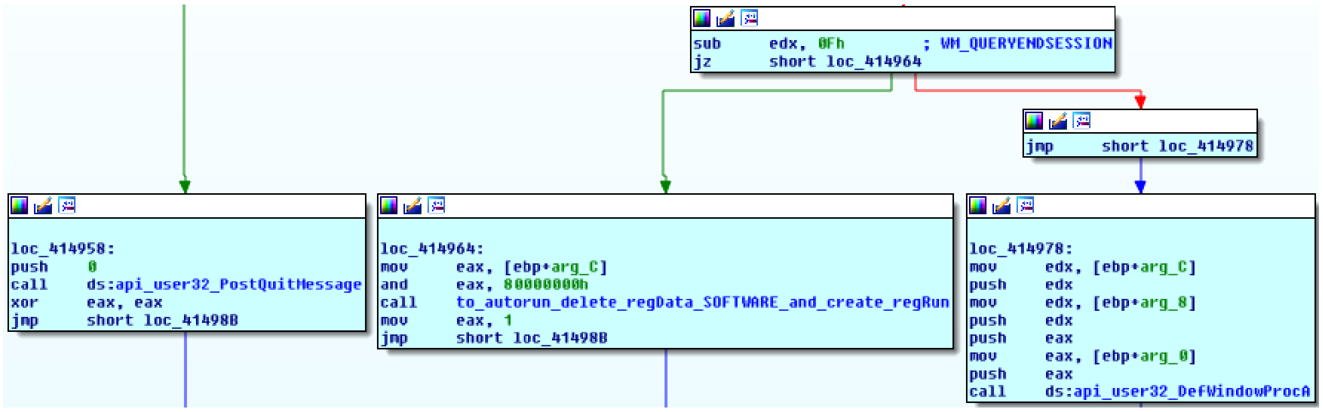
Figure 3 shows a subroutine of the Windows Procedure.

The persistence mechanism and registry clearing is done using the following method:

1. Malware creates a copy of itself unto %ProgramFiles% for Windows XP and %AppData% for Windows Vista and up with a randomly generated filename from a given list of strings. We'll call this %dropfilepath%.

2. It registers a Window Procedure to monitor the window messages.

3. The Window Procedure checks the Windows Messages coming in the system and waits for a Window Message WM_QUERYENDSESSION to execute its routine, as describe below in Figure 3.



```
sub     edx, 0Fh            ; WM_QUERYENDSESSION
jz      short loc_414964
```

```
jmp     short loc_414978
```

```
loc_414958:
push    0
call    ds:api_user32_PostQuitMessage
xor     eax, eax
jmp     short loc_41498B
```

```
loc_414964:
mov     eax, [ebp+arg_C]
and     eax, 80000000h
call    to_autorun_delete_regData_SOFTWARE_and_create_regRun
mov     eax, 1
jmp     short loc_41498B
```

```
loc_414978:
mov     edx, [ebp+arg_C]
push    edx
mov     edx, [ebp+arg_8]
push    edx
push    eax
mov     eax, [ebp+arg_0]
push    eax
call    ds:api_user32_DefWindowProcA
```

Figure 3. Monitoring system shutdowns

If the Window Procedure catches a WM_QUERYENDSESSION Window message it performs the following actions:

a. Delete the random registry key HKLM/SOFTWARE/<random>, which contains the stolen email addresses and interprocess configuration of injected routines.

b. Create Startup registry on Software\Microsoft\Windows\CurrentVersion\Run and write its corresponding registry value in either one of these methods:

i. Generate shortcut file (LNK file) pointing to %dropfilepath% and append "**-autorun**" to generated lnk file.
ii. If %dropfilepath% doesn't exist, continue to write a registry value with a **–autorun** parameter.

· Generate a random 20 character string with file extension .txt, and concatenate it with the folder of %dropfilepath% and name this as %txtfilepath%. Write a copy of the malware binary to %txtfilepath%, which is found on the running malware's process heap or memory. Call MoveFileExA with the DELAY_UNTIL_REBOOT flag to copy the contents from %txtfilepath% to %dropfilepath%. Specifying the DELAY_UNTIL_REBOOT flag, the binary delays this file's move operation until the system reboots. This is likely done to prevent security software from being suspicious. Figure 4 shows the corresponding MoveFileExA API call.

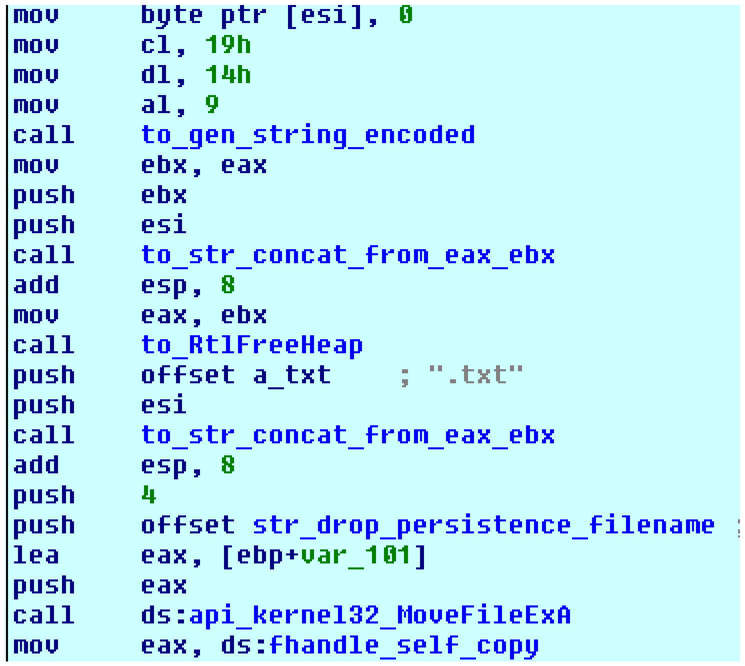

```
mov     byte ptr [esi], 0
mov     cl, 19h
mov     dl, 14h
mov     al, 9
call    to_gen_string_encoded
mov     ebx, eax
push    ebx
push    esi
call    to_str_concat_from_eax_ebx
add     esp, 8
mov     eax, ebx
call    to_RtlFreeHeap
push    offset a_txt    ; ".txt"
push    esi
call    to_str_concat_from_eax_ebx
add     esp, 8
push    4
push    offset str_drop_persistence_filename ;
lea     eax, [ebp+var_101]
push    eax
call    ds:api_kernel32_MoveFileExA
mov     eax, ds:fhandle_self_copy
```

Figure 4. Delayed MoveFileExA API call to prevent antivirus detection

Random Filename Generation

URLZone uses an interesting algorithm to generate random filenames. Unlike most banking trojans, which generate random looking strings for dropped filenames, URLZone uses an array of strings to generate the filename of the dropped file. To add entropy to the random string generation algorithm, it uses a subroutine that creates a random byte using the RDTSC instruction combined with other arithmetic operations.

The array of strings used to generate the filename is as follows:

char *filenames[] = ["win", "video", "def", "mem", "dns", "user", "logon", "hlp", "mixer", "pack", "mon", "srv", "exec", "play"]

The random string generation algorithm can be invoked in two ways:

1. rand(len_min, len_max, upper_offset_limit) -> to construct a random string from a given string.
2. rand(upper_offset_limit) -> to get a random string from a given array.

len_min and len_max are the minimum and maximum lengths of the string to be returned. The upper_offset_limit is the upper limit of the offset, which is generated randomly.

The above algorithms are used to generate the filename as follows:

1. Get a string randomly from the above array.
2. Construct a random string of length between 1 and 2 from the string: "qwertyuiopasdfghjklzxcvbnm123945678"
3. Calculate a flag randomly and check its value. If the value of the flag is 0, then it proceeds to concatenate the strings generated in step 1 and 2. If the value of flag is 1, then it gets one more string randomly from the array similar to step 1.

By putting all this together, we have the following two ways filenames can be generated based on the value of the random flag:

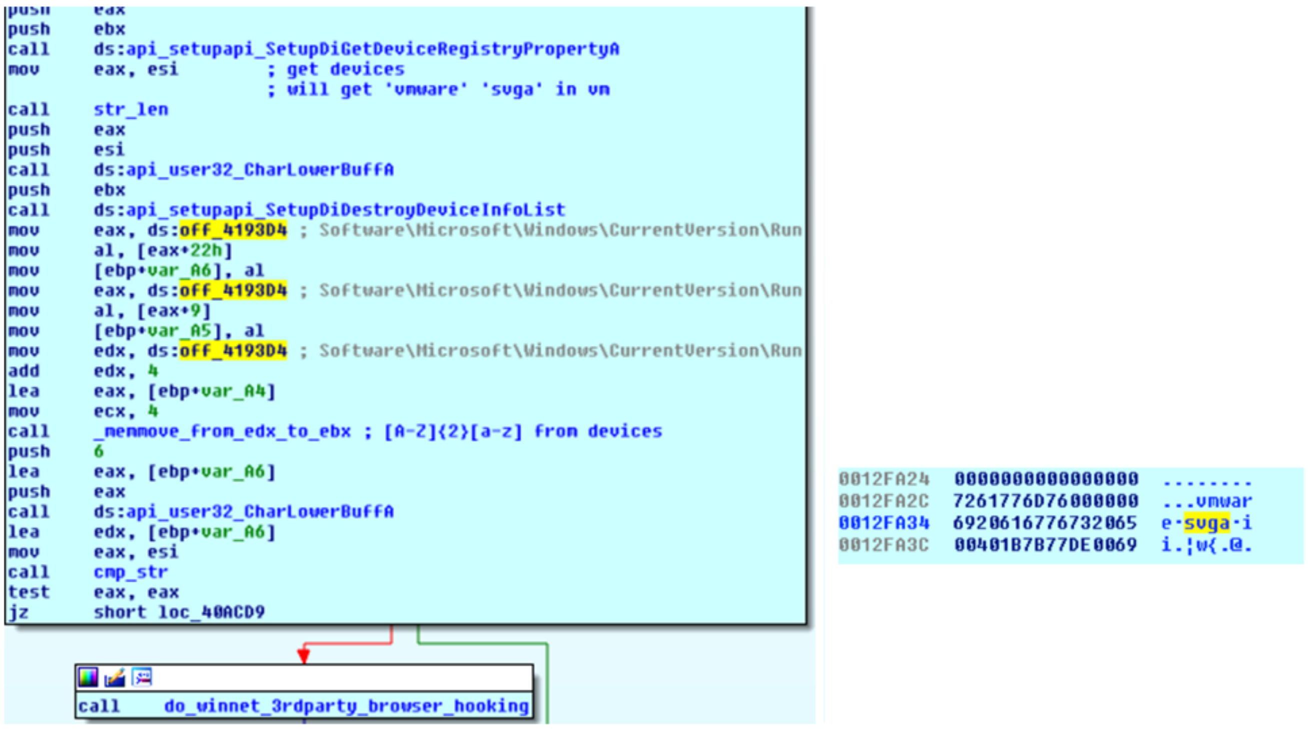
1. If flag is 0, then concatenate strings in steps 1 and 2.
2. If flag is 1, then concatenate strings in steps 1, 2, and 3.

The reason for generating the filenames this way may be to evade heuristics of security products, which alert on dropped executable files with randomly generated names. The filename generated using the above algorithm looks human readable and less suspicious.

Evasion Technique

URLZone attempts to detect the use of VMware using the following method:

1. Resolve SetupDi APIs by pre-calculated string hash from setupapi.dll
2. Retrieve the device information using those APIs
3. Check if the device names acquired contain the string “vm”



```
push    eax
push    ebx
call    ds:api_setupapi_SetupDiGetDeviceRegistryPropertyA
mov     eax, esi         ; get devices
                         ; will get 'vmware' 'svga' in vm
call    str_len
push    eax
push    esi
call    ds:api_user32_CharLowerBuffA
push    ebx
call    ds:api_setupapi_SetupDiDestroyDeviceInfoList
mov     eax, ds:off_4193D4 ; Software\Microsoft\Windows\CurrentVersion\Run
mov     al, [eax+22h]
mov     [ebp+var_A6], al
mov     eax, ds:off_4193D4 ; Software\Microsoft\Windows\CurrentVersion\Run
mov     al, [eax+9]
mov     [ebp+var_A5], al
mov     edx, ds:off_4193D4 ; Software\Microsoft\Windows\CurrentVersion\Run
add     edx, 4
lea     eax, [ebp+var_A4]
mov     ecx, 4
call    _memmove_from_edx_to_ebx ; [A-Z]{2}[a-z] from devices
push    6
lea     eax, [ebp+var_A6]
push    eax
call    ds:api_user32_CharLowerBuffA
lea     edx, [ebp+var_A6]
mov     eax, esi
call    cmp_str
test    eax, eax
jz      short loc_40ACD9

call    do_winnet_3rdparty_browser_hooking
```

```
0012FA24  0000000000000000  ........
0012FA2C  7261776D76000000  ...vmwar
0012FA34  6920616776732065  e·svga·i
0012FA3C  00401B7B77DE0069  i.¦w{.@.
```

Figure 5. VMware sandbox detection

As shown in Figure 5, the malware partially collects two characters from “Software\Microsoft\Windows\CurrentVersion\Run” string to build up “vm” for comparison. SetupDi API enumerates all the names of the devices installed in the system, such as “vmware”, “svga ii”.

As soon as one of the device names starts with the string “vm” it will jump to a hooking function and soon terminates the thread, thus not allowing the malware to continue further with is routine and CnC callback.

An Ongoing Campaign

On Jan. 19, 2016, and Jan. 20, 2016, we observed another round of URLZone spam targeting Japan. The basic TTPs are unchanged, but the scale is larger than the spam campaign we observed in December 2015.

Conclusion

Although URLZone has been around for a while and primarily targets countries in Europe, we still see it active and now shifting to Japan. It is likely that URLZone will further expand its activity in Japan with improved localization and techniques. Email users should be cautious about viewing emails coming from unknown senders.

Appendix

URLZone sample hashes

- 15896a44319d18f8486561b078146c30a0ce1cd7e6038f6d614324a39dfc6c28
- 884fccbbfa5a5b96d2e308856b996ee20d9656d04505fb3cdf926270f5d11c28

Hooked APIs

- WinInet.HttpEndRequestA
- WinInet.HttpEndRequestW
- WinInet.HttpOpenRequestA
- WinInet.HttpOpenRequestW
- WinInet.HttpQueryInfoA
- WinInet.HttpQueryInfoW
- WinInet.HttpSendRequestA
- WinInet.HttpSendRequestExW
- WinInet.HttpSendRequestW
- WinInet.InternetCloseHandle
- WinInet.InternetQueryDataAvailable
- WinInet.InternetReadFile
- WinInet.InternetReadFileExA
- WinInet.InternetReadFileExW
- WinInet.InternetWriteFile
- nspr.PR_Read
- nspr.PR_Write
- nspr.PR_Close
- ws2_32.send
- ws2_32.connect
- ws2_32.closesocket