CVE-2013-0640 - Further Investigation into an Adobe PDF Zero-day Malware Attack

Our colleagues from FireEye recently discovered a zero-day malware attack which made use of an exploit for Adobe as described in and article titled "Adobe Investigating Reports of Reader Zero-Day Exploit". In addition to this, Symantec Security Response published some interesting details of the inner workings of this attack in their article New Adobe PDF Zero-day Unleashes Trojan.Swaylib.

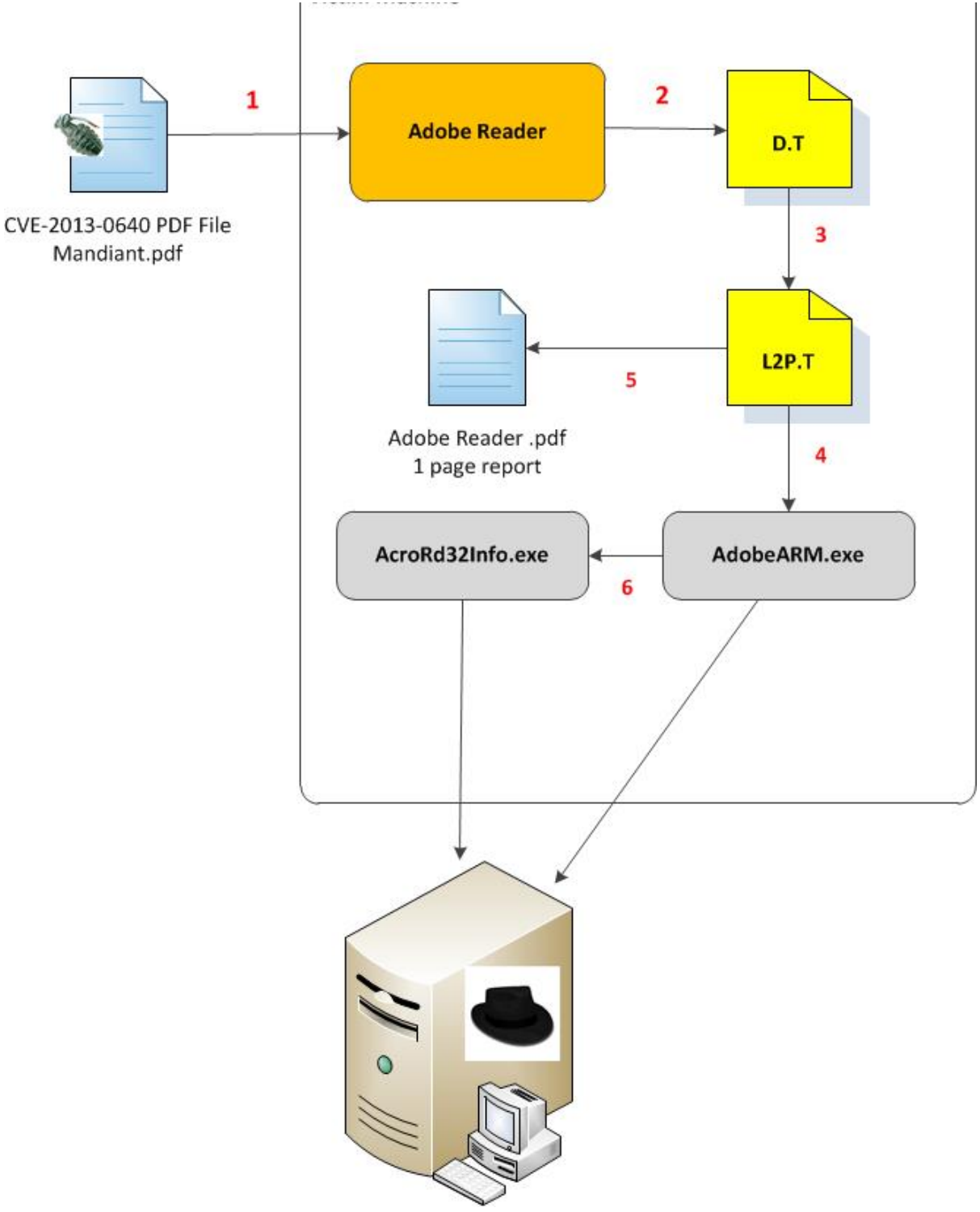
We have done additional research using a malicious file titled Mandiant.pdf (2A42BF17393C3CAAA663A6D1DADE9C93) and found additional details or what is possibly a newer variation on this attack. With our research we not only confirm the prior findings that several files were being dropped, but also have observed even more malicious files being dropped in the overall attack than have been reported. This is a sophisticated attack and we are sure there will be more details to come.

This is what the attack looks per our investigation:

recent blog posts

Facebook-spread Adobe Update Malware Dissected and Source Code Revealed (0)

A week ago security researcher Dancho Danchev published an excellent post - Fake Adobe Flash Player Serving Campaign Utilizes Google [...]



1 MONTH AGO

Scripting Bot Malware: No Need to Learn C to Launch a Cyber Attack (0)

T wo weeks ago we came across a piece of malware that turned out to be a full-blown bot—one that is capable of taking full control over a […]

2 MONTHS AGO

Vinsula Execution Engine Analysis of Venomous Snake Zero-Day Malware – CopyHook.131019.A (1)

Malware authors frequently seek code-execution methods that not only evade detection by AV software but also cover tracks and remove […]

4 MONTHS AGO

Catching a Headless Horseman (or analysis of Trojan.Downloader.1301007.C-Jottix) (0)

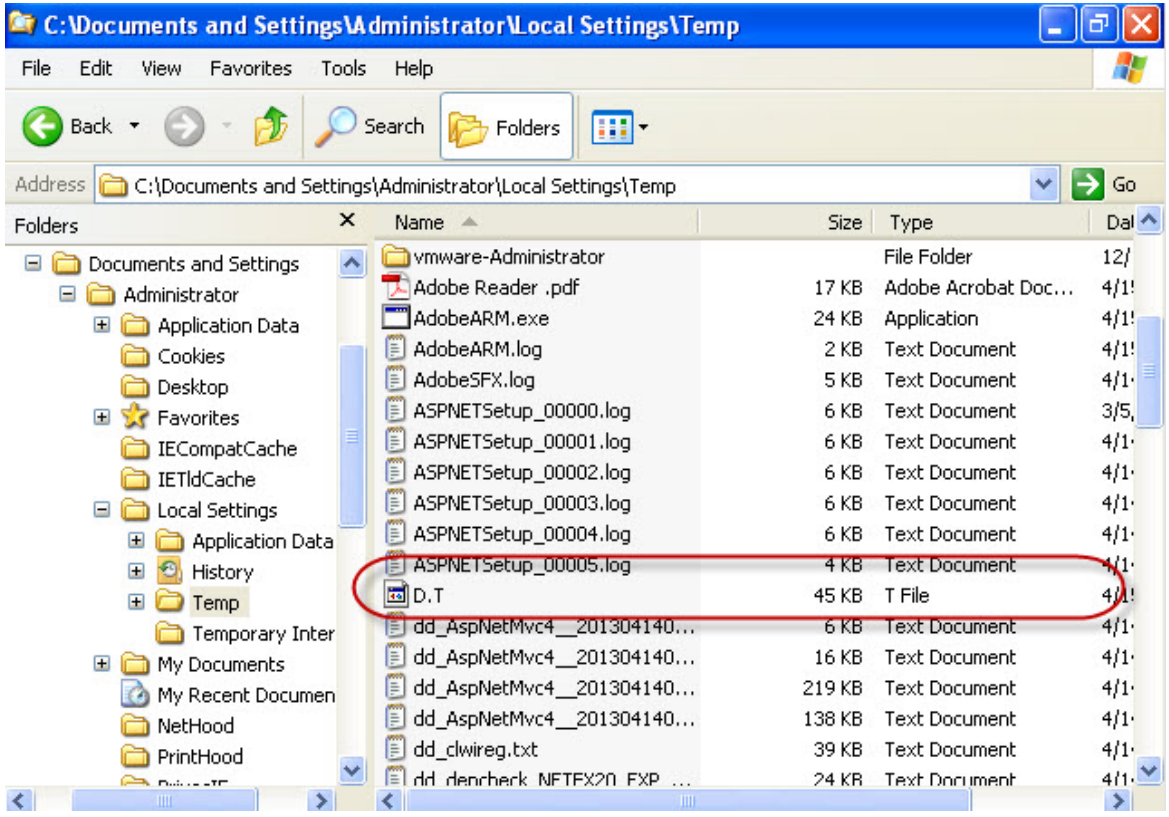
Defeating security controls in antivirus and anti-malware systems is a common goal among malware authors. There are many sophisticated […]

4 MONTHS AGO

Hunting Down FTP Password Stealer Malware with Vinsula Execution Engine (0)

Malware authors are getting increasingly creative in their attempts to bypass security controls and gain access to

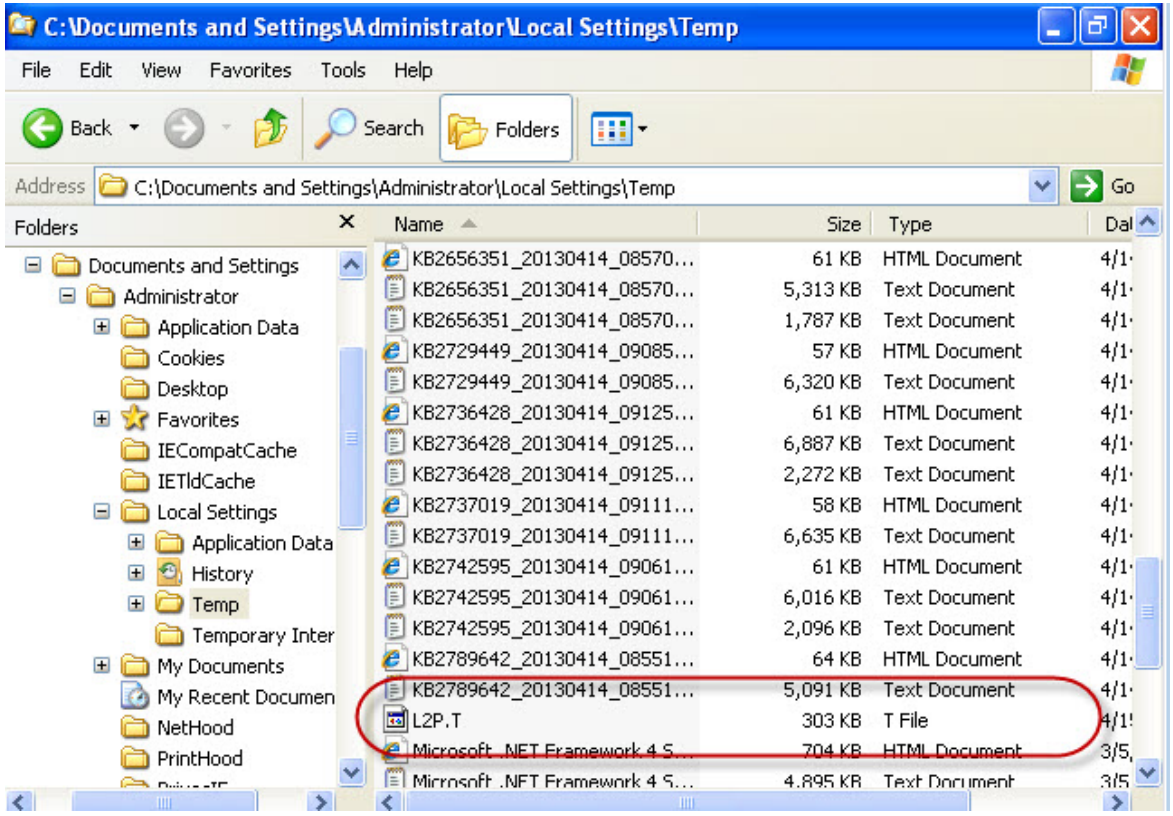
Adobe Reader reads the malicious PDF file which triggers the exploit and allows the malicious code to drop a DLL named D.T with size 45KB. The DLL is being being dropped in folder C:\Documents and Settings\MyUserName\Local Settings\Temp. As a part of the attack this DLL is then loaded by Adobe Reader.



Next the malicious code in D.T DLL creates two threads.

The first thread displays a message box that appears to come from Adobe Reader. The actual messages reads “Adobe Reader cannot display the file because it appears to be incomplete (for example, it was sent as an email attachment and its size exceeded the sender’s data limit). Adobe Reader will now switch to compatibility view.”

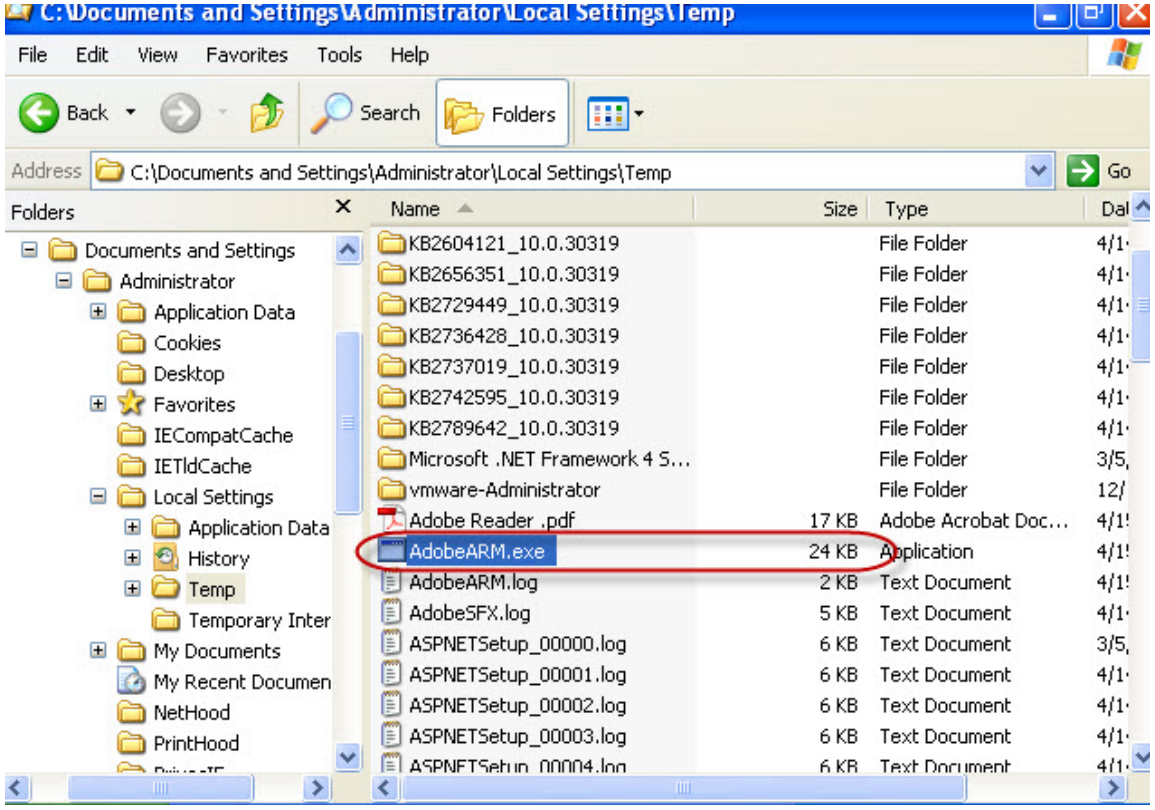
The second thread that is created by the code in D.T DLL downloads a buffer with what seems to be encrypted content, decrypts the content using the password “!H2bYm.Sw@”, and then drops an additional DLL titled L2P.t. This new DLL is stored in the same folder (C:\Documents and Settings\MyUserName\Local Settings\Temp).



Although it is not easy to decipher, we observed that D.T DLL is also trying to load clbcatq.dll using LoadLibrary(“clbcatq”). Interestingly, clbcatq.dll is not known to provide any interface for external components and has been reported as banking malware, which provides an indication of suspicious activity (see ThreatExpert’s awareness of the file “clbcatq.dll” for additional details). It is quite possible that D.T is checking whether this malware is already on the machine and if so taking advantage of that fact.

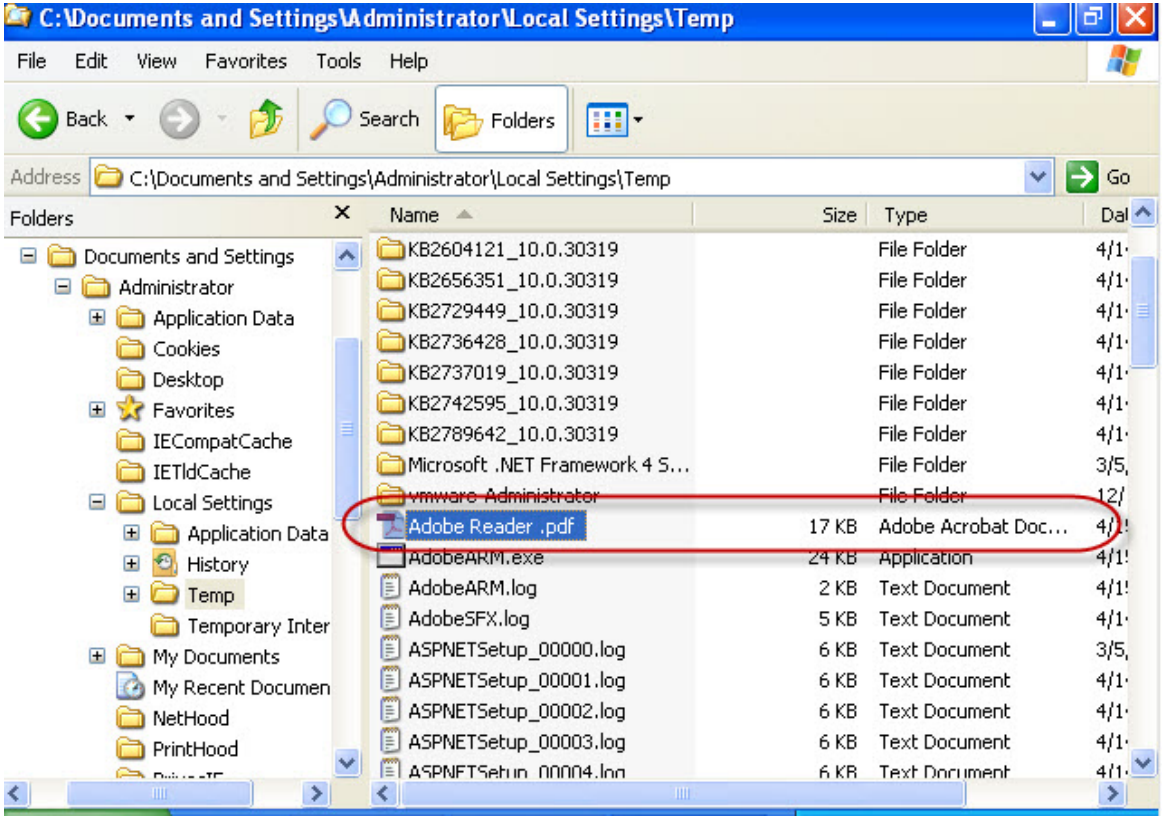
An interesting piece of code in D.T DLL shows that the code there is performing some of its business logic if the two members of _STARTUPINFO cbReserved2 and lpReserved2 are both not NULL. According to MSDN these two members are used by the C Run-time libeary and cbReserved2 must be zero and lpReserved2 must be NULL, respectively.

The entry point of the second dropped DLL L2P.T has some interesting code as well. This code executes when D.T DLL loads L2P.T DLL. A new executable AdobeARM.exe is dropped in the same Temp folder. The name of the executable is the same as one of the Adobe files, but the goal of the dropped malicious file is completely different.



What also is taking place within the entry point of L2P.T DLL is what may mislead the user that the machine has not been compromised. An instance of the malicious AdobeARM.exe is

launched using WinExec API. The code then sleeps for 5 seconds. After that a PDF file with name “Adobe Reader.pdf” is launched using ShellExecute(“open”). This PDF file is a single page from a recently released Mandiant APT report. At the end of the entry point, the code forces the running instance of infected Adobe Reader to exit using ExitProcess(0) API.



All of the above is executing in the context of an Adobe Reader process.

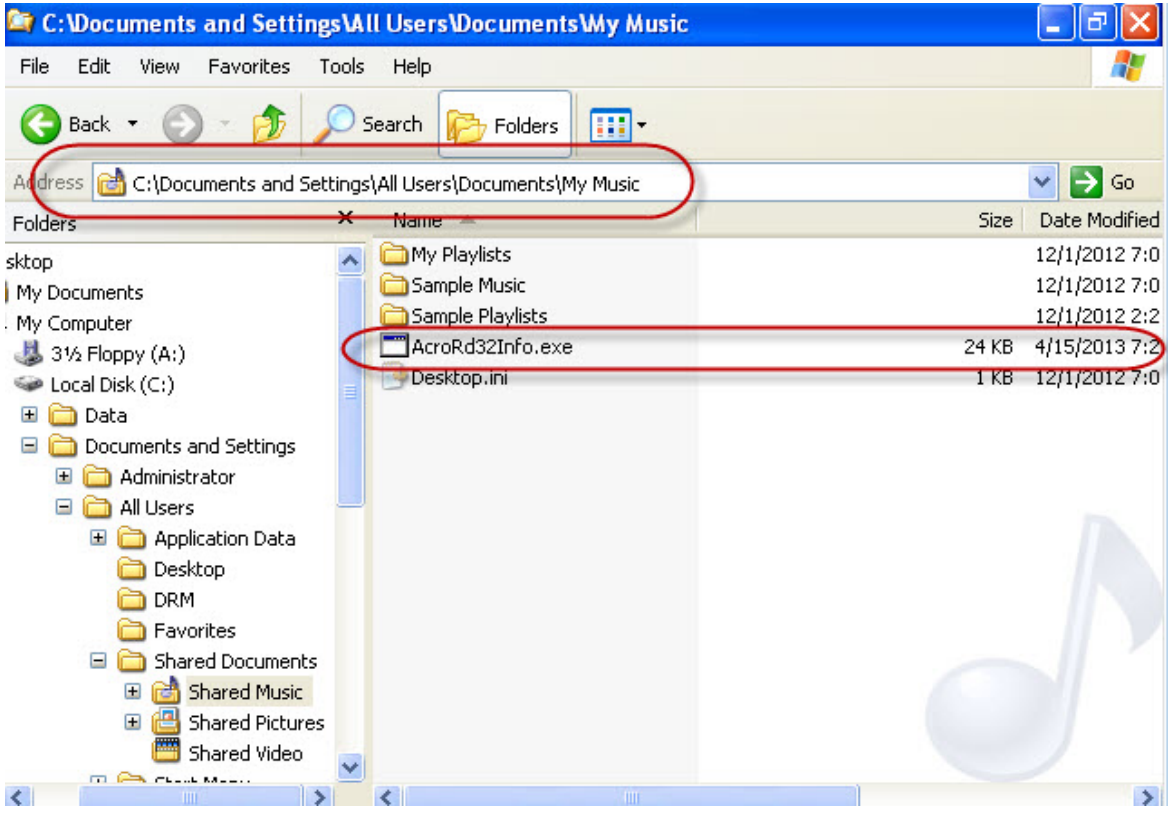
Once running the malicious process AdobeARM.exe constructs three strings on the fly that contain following URLs (to prevent users from accidentally clicking on the URLs, I’ve put a capital Z in front of each URL).

Zwww.google.co.jp.www.nifty
Zhttp://expires.ddn.dynssl.co.jp

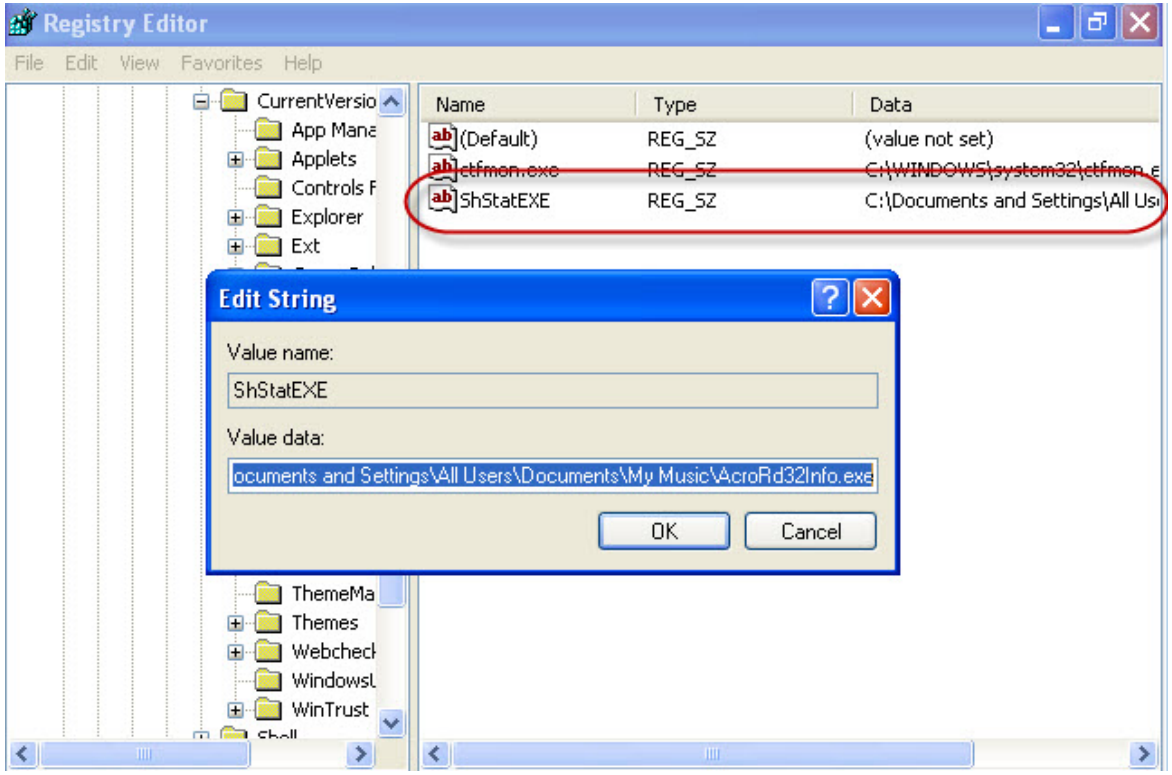
Zhttp://www.sh.ounkaku.jp

The way the code is written for creating the strings for these URLs is quite interesting. To prevent automatic string extraction by AV and AntiMalware software, the code is concatenating groups of 1 to 4 characters using strcat C function. In an additional attempt to make the analysis process more difficult, the malware authors have made extensive use of the Sleep API throughout the entirety of the AdobeARM.exe code, including at the very beginning where the URL strings are concatenated.

Next the malicious file AdobeARM.exe uses SHGetSpecialFolderPath, passing CSIDL_COMMON_MUSIC flag to retrieve the location of the music folder. A typical path is C:\Documents and Settings\All Users\Documents\My Music. AdobeARM.exe then copies itself to the music folder under the name AcroRd32Info.exe.



AdobeARM.exe then updates the registry by adding following value “C:\Documents and Settings\All Users\Documents\My Music\AcroRd32Info.exe” to the startup folder “HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run”. This will force Windows to start the malicious AcroRd32Info.exe each time the user logs on to the system.



AdobeARM.exe and AcroRd32Info.exe use WinInet library to communicate with the CnC server(s).

I’m sure there is much more than what we have found so far—we will keep investigating this attack and continue to post our findings.

Credit for the malware sample files to Mila Parkour: http://contagiodump.blogspot.com.au

Share this:

This entry was posted by Ivo Ivanov on April 17, 2013 at 6:47 am, and is filed under Analysis, Technical Research. Follow any responses to this post through RSS 2.0.You can skip to the end and leave a response. Pinging is currently not allowed.

0 Comments Vínsula, Inc.

Sort by Best

Start the discussion…

Be the first to comment.

Subscribe Add Disqus to your site