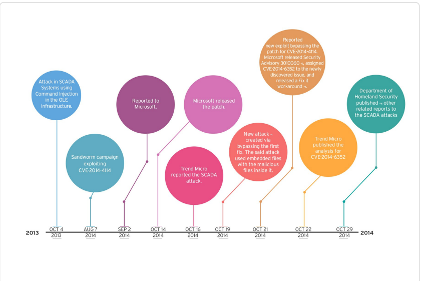
Timeline of Sandworm Attacks

Posted on: November 10, 2014 at 1:12 pm **Posted in:** Exploits
Author: William Gamazo Sanchez (Vulnerability Research)

The Sandworm vulnerability, also known as CVE-2014-4114, is an interesting vulnerability for two reasons. For one, it is related to the timing of the vulnerability life cycle. In this blog post, we will tackle vulnerability analysis, and user awareness on what actions to take when they are under attack. Note that all dates and times discussed here are based on publicly available information and in the internal metadata of the sample files. Here's a timeline:


Attack in SCADA Systems using Command Injection in the OLE infrastructure.
Sandworm campaign exploiting CVE-2014-4114
Reported to Microsoft.
Microsoft released the patch.
Trend Micro reported the SCADA attack.
New attack *1 created via bypassing the first fix. The said attack used embedded files with the malicious files inside it.
Reported new exploit bypassing the patch for CVE-2014-4114. Microsoft released Security Advisory 3010060 *2, assigned CVE-2014-6352 to the newly discovered issue, and released a Fix it workaround *3.
Trend Micro published the analysis for CVE-2014-6352
Department of Homeland Security published *4 other related reports to the SCADA attacks
2013
OCT 4 2013
AUG 7 2014
SEP 2 2014
OCT 14 2014
OCT 16 2014
OCT 19 2014
OCT 21 2014
OCT 22 2014
OCT 29 2014
2014

Click image to enlarge

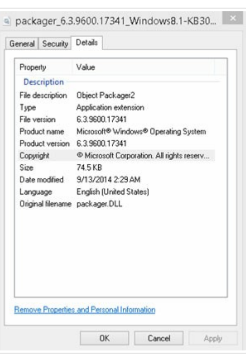
*1: New CVE-2014-4114 Attacks Seen One Week After Fix
*2: https://technet.microsoft.com/library/security/3010060
*3: https://support.microsoft.com/kb/3010060
*4: https://ics-cert.us-cert.gov/alerts/ICS-ALERT-14-281-01A

CVE-2014-4114 is also related to the OLE design by itself. We can classify it as a Command Injection in the OLE infrastructure. This area is sufficiently complex and its hard to evaluate the scope of the attack surface; this caused the release of an incomplete fix and the release of CVE-2014-6352. This is because an attacker can control two external variables to invoke different paths inside the affected component *package.dll*. The variables are: OLE Verbs and Embedded File Type.

Vulnerability time cycle

Looking at the timelines is always helpful to understand and correlate major events. Sandworm became known to the public when iSIGHT released a blog entry on October 14 discussing the vulnerability and how it was being used in targeted attacks. It was fixed on the same day as part of the scheduled Patch Tuesday release, in MS14-060. A week later, on October 21, it was disclosed that under certain circumstances the patch could be bypassed, resulting in Microsoft Security Advisory 3010060 and published workarounds.

What was in the patches? We found that they contained a new version of the file *packager.dll*. The following image shows the Windows properties of the file:

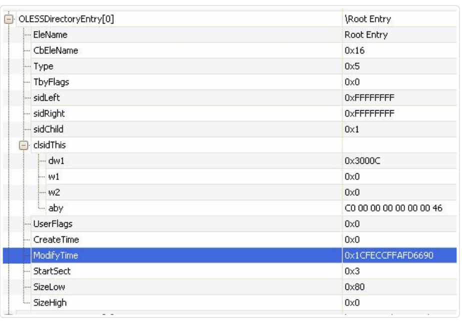

packager_6.3.9600.17341_Windows8.1-KB30...
General
Security
Details
Remove Properties and Personal Information
OK
Cancel
Apply

Property	Value
Description	
File description	Object Packager2
Type	Application extension
File version	6.3.9600.17341
Product name	Microsoft® Windows® Operating System
Product version	6.3.9600.17341
Copyright	© Microsoft Corporation. All rights reserv...
Size	74.5 KB
Date modified	9/13/2014 2:29 AM
Language	English (United States)
Original filename	packager.DLL

Figure 1. Package.dll updated version (6.3.9600.17341) Windows file properties

This file was created on September 13 – which is reasonable, since iSIGHT first spotted this attack on September 3. Other security vendors indicate they reported this flaw to Microsoft on September 2.

The email campaign of Sandworm (or BlackEnergy) that targeted this vulnerability took place from August 13 onwards, as reported in various articles. These emails used a PPSX attachment with two embedded files. These embedded files contain an internal property informing the modification and created time. The following image shows this property:



```
⊟ OLESSDirectoryEntry[0]          \Root Entry
   ├ EleName                      Root Entry
   ├ CbEleName                    0x16
   ├ Type                         0x5
   ├ TbyFlags                     0x0
   ├ sidLeft                      0xFFFFFFFF
   ├ sidRight                     0xFFFFFFFF
   ├ sidChild                     0x1
   ⊟ clsidThis
   │  ├ dw1                       0x3000C
   │  ├ w1                        0x0
   │  ├ w2                        0x0
   │  └ aby                       C0 00 00 00 00 00 00 46
   ├ UserFlags                    0x0
   ├ CreateTime                   0x0
   ├ ModifyTime                   0x1CFECCFFAFD6690
   ├ StartSect                    0x3
   ├ SizeLow                      0x80
   └ SizeHigh                     0x0
```

Figure 2. OLE Compound tree structure. Here we can see the ModifyTime is highlighted.

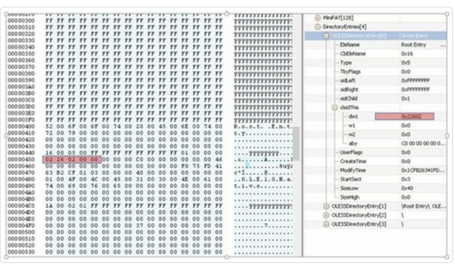
A known file (SHA256 hash: 70b8d220469c8071029795d32ea91829f683e3fbbaa8b978a31a0974daee8aaf) used in this campaign is detected by Trend Micro as TROJ_MDLOAD.PGTY. The embedded files *oleObject1* and *OleObject2* have the modified date/time of 8/7/2014 1:15:59 PM. Following the timeline until here, this would seem like a valid and logical date. On October 16, 2014, Trend Micro reported that the same type of attack is being used to exploit SCADA systems. The said attack employed the same technique – Command Injection in the OLE infrastructure – and used the same file origin. In this case two OLE files were used: *devlist.cim* and *config.bak*. Both files were created on 10/4/2013.

There are several samples in VirusTotal related to this campaign. Some of these samples are directly related to the attacks, while others are simple modification to the attacks done by analysts. Extracting the attack IPs from all the samples we can get the following list:

- \\10[.]0[.]0[.]34\public\slide1.gif
- \\10[.]0[.]0[.]34\public\slide1.inf
- \\10[.]0[.]0[.]27\share\xxx.inf
- \\10[.]0[.]0[.]27\share\xxx.gif
- \\10[.]80[.]65[.]87\impct\losslides.gif
- \\216[.]66[.]74[.]22\/root/smb4k/teamths\ths.inf
- \\216[.]66[.]74[.]22\/root/smb4k/teamths\ths.gif
- \\210[.]209[.]86[.]152\p\z\slides.inf
- \\210[.]209[.]86[.]152\p\z\slides.gif
- \\185[.]29[.]8[.]212\share\sliiides.inf
- \\185[.]29[.]8[.]212\share\sliiides.exe
- \\121[.]166[.]55[.]120\file\lint.inf
- \\121[.]166[.]55[.]120\file\head.gif
- \\121[.]166[.]55[.]12\file\head.gif
- \\192[.]168[.]10[.]10\shared\msf\XrHI.inf
- \\192[.]168[.]10[.]10\shared\msf\XrHI.inf
- \\192[.]168[.]10[.]10\shared\msf\TBSZ.gif
- \\192[.]168[.]1[.]122\Support\xxx.gif
- \\192[.]168[.]1[.]11\share\xxx.inf
- \\192[.]168[.]1[.]11\share\xxx.gif
- \\192[.]168[.]187[.]147\xpl\calc.gif
- \\192[.]168[.]15[.]4\rdb\blah.gif
- \\192[.]168[.]58[.]95\rdb\test.gif
- \\192[.]168[.]58[.]95\rdb\test.inf
- \\192[.]157[.]198[.]1\public\word.gif
- \\118[.]99[.]13[.]236\docs\partyhis.gif
- \\37[.]59[.]5[.]18\11\test.gif
- \\109[.]163[.]233[.]151\public\aaaa.gif
- \\109[.]163[.]233[.]151\public\aaaa.inf
- \\94[.]185[.]85[.]122\public\slide1.inf (This is from the sample mentioned before)
- \\94[.]185[.]85[.]122\public\slide1.gif (This is from the sample mentioned before)
- \\94[.]185[.]85[.]122\public\default.txt (This is the sample attacking SCADA Systems)

First patch and second attack

In this blog post we analyzed how the attacker can control the OLE Verb to execute the file once the PPSX is run. However, another interesting part of the attack is how the attacker control the file type to bypass the Mark on the Web (MOTW) and avoid the alert message in Windows showing the file as untrusted. The user can control the file type using the CLSID in the OLE compound document. The said property is under /Root Entry of the embedded object. The following image shows one example. In this case, the embedded type is 0x22602.



```
00000300  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000310  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000320  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000330  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000340  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000350  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000360  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000370  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000380  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000390  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
000003A0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
000003B0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
000003C0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
000003D0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
000003E0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
000003F0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00000400  52 00 6F 00 6F 00 74 00 20 00 45 00 6E 00 74 00   R.o.o.t. .E.n.t.
00000410  72 00 79 00 00 00 00 00 00 00 00 00 00 00 00 00   r.y.............
00000420  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000430  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000440  16 00 05 00 FF FF FF FF FF FF FF FF 01 00 00 00   ....????????....
00000450  02 26 02 00 00 00 00 00 C0 00 00 00 00 00 00 46   .&......À......F
00000460  00 00 00 00 00 00 00 00 00 00 00 00 F0 75 FD 41   ............ðuýA
00000470  63 B2 CF 01 03 00 00 00 40 00 00 00 00 00 00 00   c²Ï.....@.......
00000480  01 00 4F 00 4C 00 45 00 31 00 30 00 4E 00 61 00   ..O.L.E.1.0.N.a.
00000490  74 00 69 00 76 00 65 00 00 00 00 00 00 00 00 00   t.i.v.e.........
000004A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000004B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000004C0  1A 00 02 01 FF FF FF FF FF FF FF FF FF FF FF FF   ....????????????
000004D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000004E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000004F0  00 00 00 00 00 00 00 00 37 00 00 00 00 00 00 00   ........7.......
00000500  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000510  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000520  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000530  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
```

```
⊕ MiniFAT[128]
⊟ DirectoryEntries[4]
   ⊟ OLESSDirectoryEntry[0]       \Root Entry
      EleName                     Root Entry
      CbEleName                   0x16
      Type                        0x5
      TbyFlags                    0x0
      sidLeft                     0xFFFFFFFF
      sidRight                    0xFFFFFFFF
      sidChild                    0x1
      ⊟ clsidThis
         dw1                      0x22602
         w1                       0x0
         w2                       0x0
         aby                      C0 00 00 00 00 0...
      UserFlags                   0x0
      CreateTime                  0x0
      ModifyTime                  0x1CFB26341FD...
      StartSect                   0x3
      SizeLow                     0x40
      SizeHigh                    0x0
   ⊕ OLESSDirectoryEntry[1]       \Root Entry\ OLE...
   ⊕ OLESSDirectoryEntry[2]       \
   ⊕ OLESSDirectoryEntry[3]       \
```

Figure 3. OLE RootEntry property CLSID. The first value is the embedded type(0x22602).

When *package.dll* is processing embedded files, the actual operation or extraction of the file depends on the file type. There are several type of files. The following image shows a big picture on how this works.

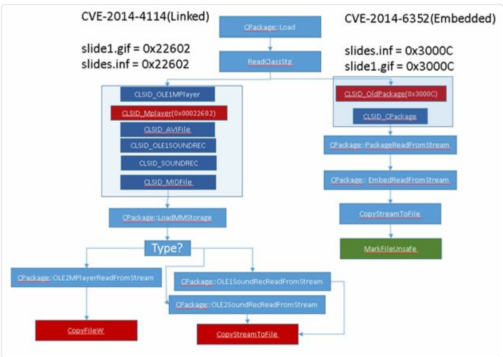


Figure 4. Call paths inside package.dll.

The attack for CVE-2014-4114 used *0x22602* as file type. This allows the attacker to bypass the MOTW protection. The OLE infrastructure will call *CPackage::Load* for each embedded file included in the PPSX file. This method calls *ReadClassStg* to get the embedded file type, which is *0x22602* in both cases. This type is MPlayer. Next, *CPackage::Load* will call *LoadMMSStorage*. The method *LoadMMSStorage* calls *OLE2MPlayerReadFromStream* or *OLE1SoundReadFromStream* depending on the OLE file type returned by *ReadClassStg,* which is *MPlayer* in this case.

The problem is that methods call to *CopyFileW* or *CopyStreamToFile* both will result in creating the temporary file without MOTW. This is because the first patch from Microsoft changed the "XXReadFromStream" methods to call *MarkFileUnsave*. After the first patch the protection looks like the following screenshot:

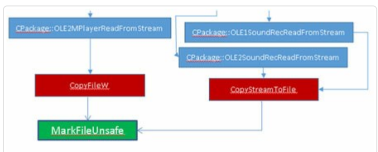
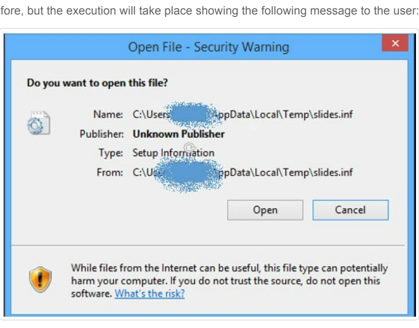

CPackage::OLE2MPlayerReadFromStream
CPackage::OLE1SoundRecReadFromStream
CPackage::OLE2SoundRecReadFromStream
CopyFileW
CopyStreamToFile
MarkFileUnsafe

Figure 5. Protection using MOTW after patch.

Note that the automatic execution using specific OLE Verb was not patched. The patch only added MOTW protection for these methods.

For the attack related to CVE-2014-6352, the protection MOTW is not bypassed, as seen in the image before, but the execution will take place showing the following message to the user:



Open File - Security Warning

Do you want to open this file?

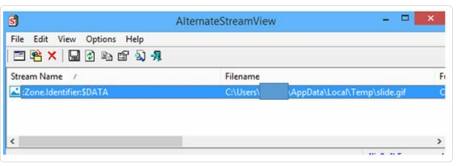
Name: C:\Users[redacted]ppData\Local\Temp\slides.inf
Publisher: **Unknown Publisher**
Type: Setup Information
From: C:\U[redacted]ppData\Local\Temp\slides.inf

Open | Cancel

While files from the Internet can be useful, this file type can potentially harm your computer. If you do not trust the source, do not open this software. What's the risk?

Figure 6. . Pop-up message alerting the user when the file is protected with MOTW.

The MOTW protection will create one NTFS stream to the created file that Windows will use to check to launch the warning message. The created NTFS stream is seen in the following image:



Stream Name	Filename
:Zone.Identifier:$DATA	C:\Users\[redacted]\AppData\Local\Temp\slide.gif

Figure 7. NTFS stream of a file with MOTW activated.

Conclusion

The attack technique for Command Injection in the OLE Infrastructure has been around since at least October 2013. If the attack happens in a system where the patch MS14-060 has been applied, the user will see the warning message shown in Figure 6.

Trend Micro secures users from this threat via detecting the exploit and malware payload via the Smart Protection Network. Trend Micro Deep Security and Office Scan with the Intrusion Defense Firewall (IDF) plugin protect user systems from threats that may leverage this vulnerability via the following DPI rules:

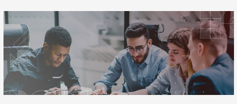
- 1006290 – Microsoft Windows OLE Remote Code Execution Vulnerability (CVE-2014-4114)
- 1006291 Microsoft Windows OLE Remote Code Execution Vulnerability (CVE-2014-4114) – 1

Users are strongly advised to patch their systems once Microsoft releases their security update for this. In addition, it is recommended for users and employees not to open PowerPoint files from unknown sources as this may possibly lead to a series of malware infection.

With additional insights from Pawan Kinger

Tags: sandworm timeline

Security Predictions for 2020



Cybersecurity in 2020 will be viewed through many lenses — from differing attacker motivations and cybercriminal arsenal to technological developments and global threat intelligence — only so defenders can keep up with the broad range of threats.
Read our security predictions for 2020.

Business Process Compromise

Attackers are starting to invest in long-term operations that target specific processes enterprises rely on. They scout for vulnerable practices, susceptible systems and operational loopholes that they can leverage or abuse. To learn more, read our Security 101: Business Process Compromise.

Recent Posts

- OpenSMTPD Vulnerability (CVE-2020-8794) Can Lead to Root Privilege Escalation and Remote Code Execution
- Operation Overtrap Targets Japanese Online Banking Users Via Bottle Exploit Kit and Brand-New Cinobi Banking Trojan
- March Patch Tuesday: LNK, Microsoft Word Vulnerabilities Get Fixes, SMBv3 Patch Follows
- Busting Ghostcat: An Analysis of the Apache Tomcat Vulnerability (CVE-2020-1938 and CNVD-2020-10487)
- Dissecting Geost: Exposing the Anatomy of the Android Trojan Targeting Russian Banks

Popular Posts

- LokiBot Impersonates Popular Game Launcher and Drops Compiled C# Code File
- Misconfigured Container Abused to Deliver Cryptocurrency-mining Malware
- Dissecting Geost: Exposing the Anatomy of the Android Trojan Targeting Russian Banks
- Outlaw Updates Kit to Kill Older Miner Versions, Targets More Systems
- February Patch Tuesday: Fixes for Critical LNK, RDP, Trident Vulnerabilities

Stay Updated

Email Subscription
Your email here
Subscribe

HOME AND HOME OFFICE | FOR BUSINESS | SECURITY INTELLIGENCE | ABOUT TREND MICRO

Asia Pacific Region (APAC): Australia / New Zealand, 中国, 日本, 대한민국, 台灣 Latin America Region (LAR): Brasil, México North America Region (NABU): United States, Canada
Europe, Middle East, & Africa Region (EMEA): France, Deutschland / Österreich / Schweiz, Italia, Россия, España, United Kingdom / Ireland

Privacy Statement Legal Policies