

Brute Force Attacks Write-up [LetsDefend]

Brute Force Attacks

Our web server has been compromised, and it's up to you to investigate the breach. Dive into the system, analyze logs, dissect network traffic, and uncover clues to identify the attacker and determine the extent of the damage. Are you up for the challenge?

File Location: /root/Desktop/ChallengeFile/BruteForce.7z

File Password: infected

Start Investigation

Q1: What is the IP address of the server targeted by the attacker's brute-force attack?

The BruteForce.7z file was successfully extracted using the 7z tool. After extraction, the contents were found to include two main files:

```
root@ip-172-31-12-223:~/Desktop/ChallengeFile# 7z x BruteForce.7z
[...]
Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
zip Version 16.02 (locale=C.UTF-8,Utf16=on,HugeFiles=on,64 bits,2 CPUs Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz (406F1),ASM,AES-NI)
Scanning the drive for archives:
1 file, 1984128 bytes (1938 KiB)

Extracting archive: BruteForce.7z
--
Path = BruteForce.7z
Type = 7z
Physical Size = 1984128
Headers Size = 240
Method = LZMA2:23 7zAES
Solid = +
Blocks = 1

Enter password (will not be echoed):
Everything is Ok

Folders: 1
Files: 2
Size:      7613481
          1984128
```

- BruteForce.pcap
- Auth.log

To identify the target IP address, I navigated to the **Endpoints** statistics in Wireshark. I noticed several IP addresses listed; however, two specific IPs caught my attention due to their unusually high packet counts, indicating significant network activity.

Wireshark - Endpoints - BruteForce.pcap												
Ethernet · 7	IPv4 · 8	IPv6 · 4	TCP · 431	UDP · 11								
Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes	Country	City	Latitude	Longitude	AS Number	AS Organization
51.116.96.181	23,199	4 MB	12,643	3 MB	10,556	1 MB						
91.189.91.96	10	856 bytes	5	479 bytes	5	377 bytes						
162.213.33.48	8	578 bytes	1	60 bytes	7	518 bytes						
192.168.190.1	3	413 bytes	3	413 bytes	0	0 bytes						
192.168.190.2	8	1kB	4	983 bytes	4	378 bytes						
192.168.190.137	23,225	4 MB	10,572	1 MB	12,653	3 MB						
192.168.190.255	1	243 bytes	0	0 bytes	1	243 bytes						
224.0.0.251	2	170 bytes	0	0 bytes	2	170 bytes						

- Ip address: 51.116.96.181
- Ip address: 192.168.190.137

BruteForce.pcap											
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help											
Apply a display filter ... <Ctrl-/>											
No.	Time	Source	Destination	Protocol	Length	Info					
1 0.000000		192.168.190.137	51.116.96.181	TCP	74	58386 - 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1388683853 TSecr...					
2 0.002519		192.168.190.137	51.116.96.181	TCP	74	58310 - 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1388683856 TSecr...					
3 0.008929		192.168.190.137	51.116.96.181	TCP	74	58316 - 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1388683862 TSecr...					
4 0.011968		192.168.190.137	51.116.96.181	TCP	74	58324 - 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1388683865 TSecr...					
5 0.065377		51.116.96.181	192.168.190.137	TCP	68	3389 - 58386 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460					

By analyzing the TCP 3-way handshake, I identified the server IP **51.116.96.181** as it responds with [SYN, ACK] to the repeated [SYN] requests initiated by the attacker.

Q2: Which directory was targeted by the attacker's brute-force attempt?

Wireshark - Export · HTTP object list

Text Filter: Content Type: All Content-Types

Packet	Hostname	Content Type	Size	Filename
20947	51.116.96.181	application/x-www-form-urlencoded	31 bytes	index.php
20949	51.116.96.181	text/html	472 bytes	index.php
20959	51.116.96.181	application/x-www-form-urlencoded	31 bytes	index.php
20962	51.116.96.181	text/html	472 bytes	index.php
20972	51.116.96.181	application/x-www-form-urlencoded	29 bytes	index.php
20975	51.116.96.181	text/html	472 bytes	index.php
20985	51.116.96.181	application/x-www-form-urlencoded	31 bytes	index.php
20988	51.116.96.181	text/html	472 bytes	index.php
20998	51.116.96.181	application/x-www-form-urlencoded	32 bytes	index.php
21001	51.116.96.181	text/html	472 bytes	index.php
21011	51.116.96.181	application/x-www-form-urlencoded	34 bytes	index.php
21014	51.116.96.181	text/html	472 bytes	index.php
21024	51.116.96.181	application/x-www-form-urlencoded	39 bytes	index.php
21027	51.116.96.181	text/html	472 bytes	index.php
21037	51.116.96.181	application/x-www-form-urlencoded	30 bytes	index.php
21040	51.116.96.181	text/html	472 bytes	index.php
21050	51.116.96.181	application/x-www-form-urlencoded	29 bytes	index.php
21053	51.116.96.181	text/html	472 bytes	index.php
21064	51.116.96.181	application/x-www-form-urlencoded	29 bytes	index.php
21066	51.116.96.181	text/html	472 bytes	index.php
21076	51.116.96.181	application/x-www-form-urlencoded	31 bytes	index.php

By exporting HTTP objects, I identified that the attacker targeted the **index.php** directory for the brute-force attempt.

Q3: Identify the correct username and password combination used for login.

After analyzing the index.php packets and seeing multiple 'incorrect' login responses, I filtered for the successful attempt to identify the correct credentials.

Wireshark - Follow HTTP Stream (tcp.stream eq 252) - BruteForce.pcap

tcp.stream eq 252

```

POST /index.php HTTP/1.1
Host: 51.116.96.181
User-Agent: python-requests/2.31.0
Accept-Encoding: gzip, deflate
Accept: /*
Connection: keep-alive
Content-Length: 31
Content-Type: application/x-www-form-urlencoded
username=t3m&password=TestTestHTTP/1.1 200 OK
Date: Sun, 25 Feb 2024 12:39:09 GMT
Server: Apache/2.4.52 (Ubuntu)
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 256
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
<!DOCTYPE html>
<html>
<head>
<title>Fake Login Page</title>
</head>
<body>
<h2>Login</h2>
<p style="color: red;">Incorrect</p>
<form method="post">
<label for="username">username:</label><br>
<input type="text" id="username" name="username"><br>

```

client pkt[3], 1 server pkt[3], 1 return.

Entire conversation (978 bytes) Show data as ASCII Stream 252

The screenshot shows an HTTP session in Wireshark. A specific packet is selected, revealing its content as an HTML response. The response body contains a login form with fields for 'username' and 'password'. The 'username' field is populated with 'web-hacker' and the 'password' field with 'admin12345'. The packet details pane shows the full URL as 'POST /index.php HTTP/1.1'.

After identifying the packet containing the 'correct' login response, I analyzed the HTTP stream to extract the valid username and password.

- Username: web-hacker

This screenshot shows the 'Follow HTTP Stream' view in Wireshark. It displays the raw POST request sent by the client. The 'username' and 'password' fields are clearly visible in the payload as 'username=web-hacker&password=admin12345'.

- Password: admin12345

Q4: How many user accounts did the attacker attempt to compromise via RDP brute-force?

This screenshot shows a list of RDP negotiation requests in Wireshark. The 'Frame contains "Cookie"' filter is applied, highlighting multiple requests from different sources. The 'Info' column in the details pane shows the 'Cookie' header for each request, indicating that 10 unique users sent negotiation requests.

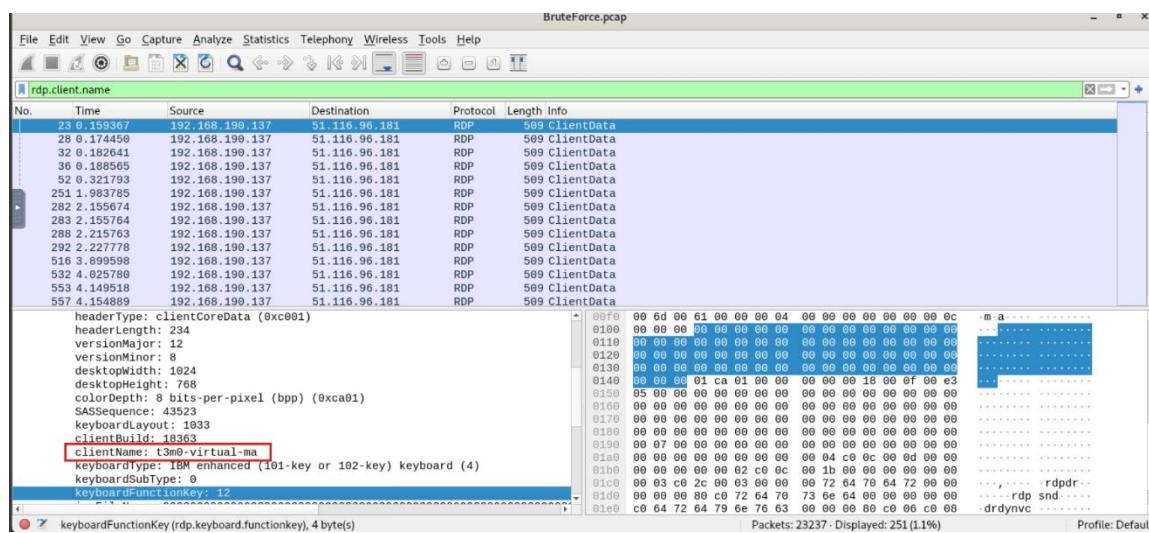
By using frame contains "Cookie", we can determine 10 unique users that sent negotiation request which are

- t3m0
- Mosalah
- Messi
- web-hacker
- Kareem
- Mostafa
- mmox
- Mohamed
- Ali
- Mohsen

But there are only 7 that got bruteforced

Q5: What is the “clientName” of the attacker's machine?

To find the attacker's machine name, I applied the rdp.client.name filter in Wireshark, which revealed the client name as: t3m0-virtual-ma.



Q6: When did the user last successfully log in via SSH, and who was it?

```
root@ip-172-31-0-35:~/Desktop/ChallengeFile/BruteForce# grep -i "accepted" auth.log
Feb 24 10:14:08 chall sshd[1039]: Accepted password for mmax from 196.136.60.15 port 32976 ssh2
Feb 24 10:38:38 chall sshd[2618]: Accepted password for mmax from 196.136.60.15 port 32932 ssh2
Feb 24 10:43:28 chall sshd[2755]: Accepted password for mmax from 196.136.60.15 port 32933 ssh2
Feb 24 10:44:04 chall sshd[2825]: Accepted password for mmax from 196.136.60.15 port 32903 ssh2
Feb 24 13:57:41 chall sshd[5367]: Accepted password for mmax from 41.38.160.33 port 55871 ssh2
Feb 24 13:57:42 chall sshd[5400]: Accepted password for mmax from 41.38.160.33 port 55868 ssh2
Feb 24 14:29:30 chall sshd[6012]: Accepted password for mmax from 41.38.160.33 port 25846 ssh2
Feb 24 14:48:57 chall sshd[6484]: Accepted password for mmax from 41.38.160.33 port 26242 ssh2
Feb 24 14:50:43 chall sshd[6592]: Accepted password for mmax from 41.38.160.33 port 26268 ssh2
Feb 24 14:59:37 chall sshd[6751]: Accepted password for mmax from 41.38.160.33 port 26502 ssh2
Feb 24 17:30:28 chall sshd[8504]: Accepted password for mmax from 41.38.160.33 port 28460 ssh2
Feb 24 18:01:50 chall sshd[9424]: Accepted password for mmax from 196.136.60.15 port 12175 ssh2
Feb 24 20:56:05 chall sshd[3579]: Accepted password for mmax from 41.38.160.33 port 52007 ssh2
Feb 24 21:35:45 chall sshd[5330]: Accepted password for mmax from 41.38.160.33 port 52665 ssh2
Feb 24 21:40:05 chall sshd[5466]: Accepted password for mmax from 41.38.160.33 port 52730 ssh2
Feb 24 22:29:15 chall sshd[7236]: Accepted password for mmax from 41.38.160.33 port 53803 ssh2
Feb 24 23:04:49 chall sshd[7851]: Accepted password for mmax from 41.38.160.33 port 54597 ssh2
Feb 24 23:06:41 chall sshd[7943]: Accepted password for mmax from 41.38.160.33 port 54634 ssh2
Feb 25 10:45:33 chall sshd[14130]: Accepted password for mmax from 41.38.160.33 port 52860 ssh2
Feb 25 10:46:22 chall sshd[14266]: Accepted password for mmax from 41.38.160.33 port 52890 ssh2
Feb 25 11:08:45 chall sshd[14449]: Accepted password for mmax from 41.38.160.33 port 53735 ssh2
Feb 25 11:11:27 chall sshd[15010]: Accepted password for mmax from 41.38.160.33 port 53787 ssh2
Feb 25 11:21:48 chall sshd[15709]: Accepted password for mmax from 41.38.160.33 port 54057 ssh2
Feb 25 11:21:57 chall sshd[15765]: Accepted password for mmax from 41.38.160.33 port 54058 ssh2
Feb 25 11:34:45 chall sshd[16532]: Accepted password for mmax from 41.38.160.33 port 54294 ssh2
Feb 25 11:39:22 chall sshd[18857]: Accepted password for mmax from 41.38.160.33 port 54357 ssh2
Feb 25 11:43:54 chall sshd[981]: Accepted password for mmax from 41.38.160.33 port 54464 ssh2
root@ip-172-31-0-35:~/Desktop/ChallengeFile/BruteForce#
```

The user 'mmax' last successfully logged in via SSH on Feb 25 at 11:43:54, as indicated by the final 'Accepted password' entry in the log file.

Q7: How many unsuccessful SSH connection attempts were made by the attacker?

```
root@ip-172-31-0-35:~/Desktop/ChallengeFile/BruteForce# grep -i "failed password" auth.log | wc -l
7480
root@ip-172-31-0-35:~/Desktop/ChallengeFile/BruteForce#
```

By running the command grep -i "failed password" auth.log | wc -l, I determined that the attacker made 7,480 unsuccessful SSH login attempts.

Q8: What technique is used to gain access?

Answer: T1110

Brute Force

Sub-techniques (4)

Adversaries may use brute force techniques to gain access to accounts when passwords are unknown or when password hashes are obtained.^[1] Without knowledge of the password for an account or set of accounts, an adversary may systematically guess the password using a repetitive or iterative mechanism.^[2] Brute forcing passwords can take place via interaction with a service that will check the validity of those credentials or offline against previously acquired credential data, such as password hashes.

Brute forcing credentials may take place at various points during a breach. For example, adversaries may attempt to brute force access to [Valid Accounts](#) within a victim environment leveraging knowledge gathered from other post-compromise behaviors such as [OS Credential Dumping](#), [Account Discovery](#), or [Password Policy Discovery](#). Adversaries may also combine brute forcing activity with behaviors such as [External Remote Services](#) as part of Initial Access.

ID: T1110

Sub-techniques: [T1110.001](#), [T1110.002](#), [T1110.003](#), [T1110.004](#)

① Tactic: Credential Access

① Platforms: Azure AD, Containers, Google Workspace, IaaS, Linux, Network, Office 365, SaaS, Windows, macOS

Contributors: Alfredo Oliveira, Trend Micro; David Fiser, @anu4is, Trend Micro; Ed Williams, Trustwave, SpiderLabs; Magno Logan, @magnologan, Trend Micro; Mohamed Kmal; Yossi Weizman, Azure Defender Research Team

Version: 2.5

Created: 31 May 2017

Last Modified: 29 January 2024

Summary

In this challenge, I analyzed network traffic using Wireshark to identify the threat actor's IP address and uncover brute-force attacks targeting the web server and RDP. I also performed a forensic review of the authentication logs to calculate the number of failed login attempts and pinpoint the exact moment the attacker gained a foothold on the server.