

**Lindsay Von Tish  
lmv9443@nyu.edu**

**Challenge Write Up**

**Week Zero**

**01/31/2024**

Table of Contents

[Challenge Details 3](#_Toc157618192)

[Are You Alive 3](#_Toc157618193)

[Doors of Durin 4](#_Toc157618194)

[Mathwhiz 6](#_Toc157618195)

[Appendix A: Student Information 8](#_Toc157618196)

[Appendix B: Tools 8](#_Toc157618197)

[Appendix C: Mathwhiz Code Solution 8](#_Toc157618198)

# Challenge Details

## Are You Alive

#### **Overview**

|  |  |  |
| --- | --- | --- |
| Are You Alive | | |
| **1 Point** | Flag Value | flag{youve\_been\_lied\_to\_about\_how\_your\_computer\_works} |
| Exploit Type | N/A |

#### **Details**

The challenge begins with a prompt asking the user to download a text file titled flag.txt, as shown in the following figure:

A screenshot of a computer

Description automatically generated  
**Download Prompt**

The text file contains the challenge flag, as shown below:

|  |
| --- |
| ┌──(kali㉿kali)-[~/Downloads]  └─$ ls -latr  total 12  -rw-r--r-- 1 kali kali 54 Jan 31 14:09 flag.txt  ...omitted for brevity...  ┌──(kali㉿kali)-[~/Downloads]  └─$ cat flag.txt  flag{youve\_been\_lied\_to\_about\_how\_your\_computer\_works} |

**File Contents**

#### **Steps to Reproduce**

1. Download file
2. View file contents  
   Ex:  
   cat flag.txt

## Doors of Durin

#### **Overview**

|  |  |  |
| --- | --- | --- |
| Doors of Durin | | |
| **100 Points** | Flag Value | flag{the\_dwarves\_dug\_too\_deep} |
| Exploit Type | Nerd |

#### **Details**

The challenge begins with a prompt directing students to connect to offsec-chalbroker.osiris.cyber.nyu.edu at port 1235.

A screenshot of a computer

Description automatically generated  
**Challenge Prompt**

Connecting to the service reveals a text-based adventure game playing through a Lord of the Rings scene. The game prompts the user to answer the riddle, "Speak friend and enter." The game text is shown in the following figure:

|  |
| --- |
| ─$ nc offsec-chalbroker.osiris.cyber.nyu.edu 1235  You and your party of a Wizard, a Dwarf, and Elf, 2 Men, and 3 other Hobbits stand around the Doors of Durin, the entrance to the Dwarven Mines of Moria.  A door blocks your way into the Mines, the only remaining path you have to get to the forest Lothlórien, where the Lady Galadriel is sure to offer you sanctuary from the dark forces pursuing you.  The Wizard looks at the Doors, and reads:  "Ennyn Durin Aran Moria. Pedo mellon a Minno. Im Narvi hain echant. Celebrimbor o Eregion tethant. I thiw hin."  You ask, "What does it mean?"  "Oh, it is a simple riddle," says the Wizard.  "The Doors of Durin, Lord of Moria. Speak friend and enter. I Narvi made them. Celebrimbor of Hollin drew these signs."  You think for a moment. "Speak friend and enter." What could it mean?  Suddenly, the answer comes to you!  You shout: |

**Game Text**

The Tolkien elvish word for friend is "Mellon," and after a few spelling errors, the user successfully input the password and gained access to the challenge flag, as shown below:

|  |
| --- |
| Suddenly, the answer comes to you!  You shout: mellon  The Doors open! As you delve into the Mines, you hear a whisper on the wind:  The flag is: flag{the\_dwarves\_dug\_too\_deep} |

**Password Correct**

An attacker would not need to use an exploit to attack this system. Any attacker with working knowledge of Lord of the Rings would be able to guess the password and obtain the challenge flag.

#### **Steps to Reproduce**

1. Spend your childhood in Alaska with no TV and only books for company.
2. Connect to the system with the following command:  
   nc offsec-chalbroker.osiris.cyber.nyu.edu 1235
3. Input the password

## Mathwhiz

#### **Overview**

|  |  |  |
| --- | --- | --- |
| Mathwhiz | | |
| **200 Points** | Flag Value | flag{you\_sure\_are\_a\_math\_genius} |
| Exploit Type | Programming |

#### **Details**

This challenge begins with a prompt similar to the one provided with the Doors of Durin challenge. The URL is the same as the Doors of Durin challenge, but the port number is different. After connecting to the service running on port 1236, the student faces a math challenge. To complete the challenge, the user must successfully answer 100 math questions in a row. The questions increase in difficulty by switching between decimal numbers, text values, and numbers encoded in hexadecimal and binary. The connection closes if the user inputs the wrong answer or an answer that is not a number.

To solve this challenge, the user's code must account for all the different formats in which the math problems are presented. The following method is a part of a program that successfully defeated the challenge.

|  |
| --- |
| def main():  # Start remote connection  URL = "offsec-chalbroker.osiris.cyber.nyu.edu"  PORT = 1236  conn = start(URL, PORT)  log = open("MathWhiz.txt", "a")  # Get the greeting, which is 183 char/183 bytes long  g = conn.recvn(183)  log.write("Greeting: " + str(g) + "\n")  # Get the math problem next  n = QandA(conn, log)  log.close() |

**Main Method**

After connecting to the remote service and receiving a greeting, the program enters the QandA method, shown below, which takes in new math problems, passes them to the solver method, and then responds with the correct answer.

|  |
| --- |
| def QandA(conn, log):  n = 0  while n < 100:  log.write("Question " + str(n) + "\n")  b = conn.recvline()  ans = doSomeMath(b, log)  #conn.pack(ans)  conn.sendline(ans)  b = conn.recvline()  log.write(str(b) + "\n")  n+=1  b = conn.recvall()  print((str(b) + "\n"))  log.write(str(b) + "\n") |

**Q and A Method**

The solver method, depicted in the following figure, translates a math problem into a format that the Python eval function can accept. The solver checks each number to see if it is written in text, in which case it calls a translator method to translate the written number into an integer. Because the eval function accepts encoded input, the only condition that the method must check for is text.

|  |
| --- |
| def doSomeMath(byteString, log):  # Translate into readable problem  byteString = byteString[:-4]  problem = str(byteString, encoding='utf-8')  log.write(problem)  mathList = problem.split()  i = 0  while i < len(mathList):  # Check to see if the number is made of text or not  if re.search("^\D+$", mathList[i]):  mathList[i] = textToIntStr(mathList[i])  i += 2  mathString = " ".join(mathList)  # Perform calculation and return answer  str\_ans = str(eval(mathString))  byte\_ans = str\_ans.encode()  log.write(" = " + str\_ans+ "\n")  return byte\_ans |

**Solver Method**

After checking and translating each number, the solver calculates the answer using the eval function.

The text numbers are formatted as [Digit]-[Digit]-[Digit]. The translator method takes the text number and iterates through each digit to translate it to the decimal format before returning a numerical string. The method is shown below:

|  |
| --- |
| def textToIntStr(t):  tStr = t.split("-")  a = tStr  n = 0  for num in tStr:  a[n] = str(getValue(num))  n+=1  ans = "".join(a)  return ans |

**Solver Method**

The complete code is available in [Appendix C](#_Appendix_C:_Mathwhiz).

# Appendix A: Student Information

|  |  |
| --- | --- |
| Lindsay Von Tish | |
| Email | lmv9443@nyu.edu |

# Appendix B: Tools

|  |  |
| --- | --- |
| Name | URL |
| PwnTools | <https://github.com/Gallopsled/pwntools> |
| Net Cat | <https://netcat.sourceforge.net/> |

# Appendix C: Mathwhiz Code Solution

|  |
| --- |
| from pwn import \*  import re  # A function to start the remote connection  # Input: URL string, Port int  # Output: Connection  def start(U, P):  io = remote(U, P)  return io  # Gets the integer value of a text number  # Input: Text string number  # Output: Integer  def getValue(t):  if t == "ONE":  return 1  elif t == "TWO":  return 2  elif t == "THREE":  return 3  elif t == "FOUR":  return 4  elif t == "FIVE":  return 5  elif t == "SIX":  return 6  elif t == "SEVEN":  return 7  elif t == "EIGHT":  return 8  elif t == "NINE":  return 9  elif t == "ZERO":  return 0  # A function to translate a text string into a string of integers  # Input: Text string number  # Output: Itegers in string form  def textToIntStr(t):  tStr = t.split("-")  a = tStr  n = 0  for num in tStr:  a[n] = str(getValue(num))  n+=1  ans = "".join(a)  return ans  # A function to translate the response into a math problem and return the answer  # Input: Byte string representing math problem and the log file  # Output: Byte string representing the answer  def doSomeMath(byteString, log):  # Translate into readable problem  byteString = byteString[:-4]  problem = str(byteString, encoding='utf-8')  log.write(problem)  mathList = problem.split()  i = 0  while i < len(mathList):  # Check to see if the number is made of text or not  if re.search("^\D+$", mathList[i]):  mathList[i] = textToIntStr(mathList[i])  i += 2  mathString = " ".join(mathList)  # Perform calculation and return answer  str\_ans = str(eval(mathString))  byte\_ans = str\_ans.encode()  log.write(" = " + str\_ans+ "\n")  return byte\_ans  # A function to recieve questions and send answers  # Input: Connection  # Output: Number of problems completed  def QandA(conn, log):  n = 0  while n < 100:  log.write("Question " + str(n) + "\n")  b = conn.recvline()  ans = doSomeMath(b, log)  #conn.pack(ans)  conn.sendline(ans)  b = conn.recvline()  log.write(str(b) + "\n")  n+=1  b = conn.recvall()  print((str(b) + "\n"))  log.write(str(b) + "\n")  def main():  # Start remote connection  URL = "offsec-chalbroker.osiris.cyber.nyu.edu"  PORT = 1236  conn = start(URL, PORT)  log = open("MathWhiz.txt", "a")  # Get the greeting, which is 183 char/183 bytes long  g = conn.recvn(183)  log.write("Greeting: " + str(g) + "\n")  # Get the math problem next  n = QandA(conn, log)  log.close()  if \_\_name\_\_=="\_\_main\_\_":  main() |