

Daniel Beck

SDEV-325

August 31, 2020

1. The first software vulnerability that was addressed was CWE-787: Out-of-bounds Write in Java language. “Typically, this can result in corruption of data, a crash, or code execution. The software may modify an index or perform pointer arithmetic that references a memory location that is outside of the boundaries of the buffer. A subsequent write operation then produces undefined or unexpected results (*Common Weakness Enumeration*, 2020).” An example of this would be having an array that can only hold a certain amount of data, but the user tries to enter more information than the software is written to handle. 1a shows the error in a java program when the user tries to enter more than three numbers into an array. 1b shows the error being mitigated using a try/catch that catches the `ArrayIndexOutOfBoundsException`, prompting the user that they can not add more than three items.

2. The second software vulnerability that was addressed was CWE-20: Improper Input Validation in Python language. “Input validation is a frequently-used technique for checking potentially dangerous inputs in order to ensure that the inputs are safe processing within the code, or when communicating with other components. When software does not validate input properly, an attacker is able to craft the input in a form that is not expected by the rest of the application. This will lead to parts of the system receiving unintended input, which may result in altered control flow, arbitrary control of a resource, or arbitrary code execution (*Common Weakness Enumeration*, 2020).” An example of this would be at a register where the employee may accidentally type in a negative number and credit money onto the customers credit card. 2a shows the vulnerability being exposed that would allow a negative number to

be entered into the system. 2b. show the vulnerability being mitigated by adding a try/except that raises a type error when a negative number is entered.

1a.

```
1  import java.util.Scanner;
2  import java.util.StringTokenizer;
3
4  class CWE787Weakness
5  {
6      public static void main(String[] args)
7      {
8          // Create a Scanner object
9          Scanner scan = new Scanner(System.in);
10         System.out.println("Enter 3 numbers for an array seperated by a space");
11
12         // Read user input
13         String makeArray = scan.nextLine();
14
15         //tokenize the string
16         StringTokenizer st = new StringTokenizer(makeArray);
17         int count = st.countTokens();
18         int counter = 1;
19
20         //write the array
21         int[] ar = new int[3];
22         for(int i = 0; i < count; i++)
23         {
24             ar[i] = Integer.parseInt(st.nextToken());
25             System.out.println("Array" + counter + ": " + ar[i]);
26             counter++;
27         }
28
29         //close scanner
30         scan.close();
31     } //end main
32 } //end class
```

Week2/CWE787Weaknes: x



Run



Command:

Week2/CWE787Weakness.java

Building CWE787Weakness.java and running CWE787Weakness

Enter 3 numbers for an array seperated by a space

54 57 96 3

Array1: 54

Array2: 57

Array3: 96

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 3
at CWE787Weakness.main(CWE787Weakness.java:24)

Process exited with code: 1

1b.

```
1 import java.util.Scanner;
2 import java.util.StringTokenizer;
3
4 class CWE787Fix
5 {
6     public static void main(String[] args)
7     {
8         // Create a Scanner object
9         Scanner scan = new Scanner(System.in);
10        System.out.println("Enter 3 numbers for an array seperated by a space");
11
12        // Read user input
13        String makeArray = scan.nextLine();
14
15        //tokenize the string
16        try
17        {
18            StringTokenizer st = new StringTokenizer(makeArray);
19            int count = st.countTokens();
20            int counter = 1;
21
22            //write the array
23
24            int[] ar = new int[3];
25            for(int i = 0; i < count; i++)
26            {
27                ar[i] = Integer.parseInt(st.nextToken());
28                System.out.println("Array" + counter + ": " + ar[i]);
29                counter++;
30            }
31        }
32        catch(ArrayIndexOutOfBoundsException e)
33        {
34            System.out.println("Can only have three elements in this array");
35        }
36
37        //close scanner
38        scan.close();
39    } //end main
40 } //end class
```

Week2/CWE787Fix.java · !x


Run Command: Week2/CWE787Fix.java



Enter 3 numbers for an array seperated by a space
65 7 5 32 65
Array1: 65
Array2: 7
Array3: 5
Can only have three elements in this array

Process exited with code: 0

2a.

```
1  def itemOneTotal(q):
2      itemOnePrice = 15.00
3      total = float(q) * float(itemOnePrice)
4      return total
5
6  def itemTwoTotal(q):
7      itemTwoPrice = 12.00
8      total = float(q) * float(itemTwoPrice)
9      return total
10
11  qauntity = input("How many item ones were purchased? ")
12  qauntity2 = input("How many item twos were purchased? ")
13
14  print("Customer will be charged $", itemOneTotal(qauntity) + itemTwoTotal(qauntity2))
```

Week2/CWE-20:\ Imprope x 


 Run  Command: Week2/CWE-20:\ Improper\ Input\ Validation\ Wea



```
How many item ones were purchased? -9
How many item twos were purchased? 2
Customer will be charged $ -111.0

Process exited with code: 0
```

2b.

```
1  def itemOneTotal(q):
2      itemOnePrice = 15.00
3      total = float(q) * float(itemOnePrice)
4      return total
5
6  def itemTwoTotal(q):
7      itemTwoPrice = 12.00
8      total = float(q) * float(itemTwoPrice)
9      return total
10
11  try:
12      q1 = input("How many item ones were purchased? ")
13      q2 = input("How many item twos were purchased? ")
14
15      if float(q1) < 0 or float(q2) < 0:
16          raise TypeError
17      else:
18          print("Customer will be charged $", itemOneTotal(q1) + itemTwoTotal(q2))
19  except TypeError:
20      print("Can not enter a negative number!")
21
```

Week2/CWE-20:\ Imprope x 

 Run  Command: Week2/CWE-20:\ Improper\ Input\ Validation\ I

How many item ones were purchased? -9
How many item twos were purchased? 3
Can not enter a negative number!

Process exited with code: 0

References:

Common Weakness Enumeration. (2020, August 20). Retrieved September 02, 2020, from <https://cwe.mitre.org/data/definitions/>