Daniel Beck

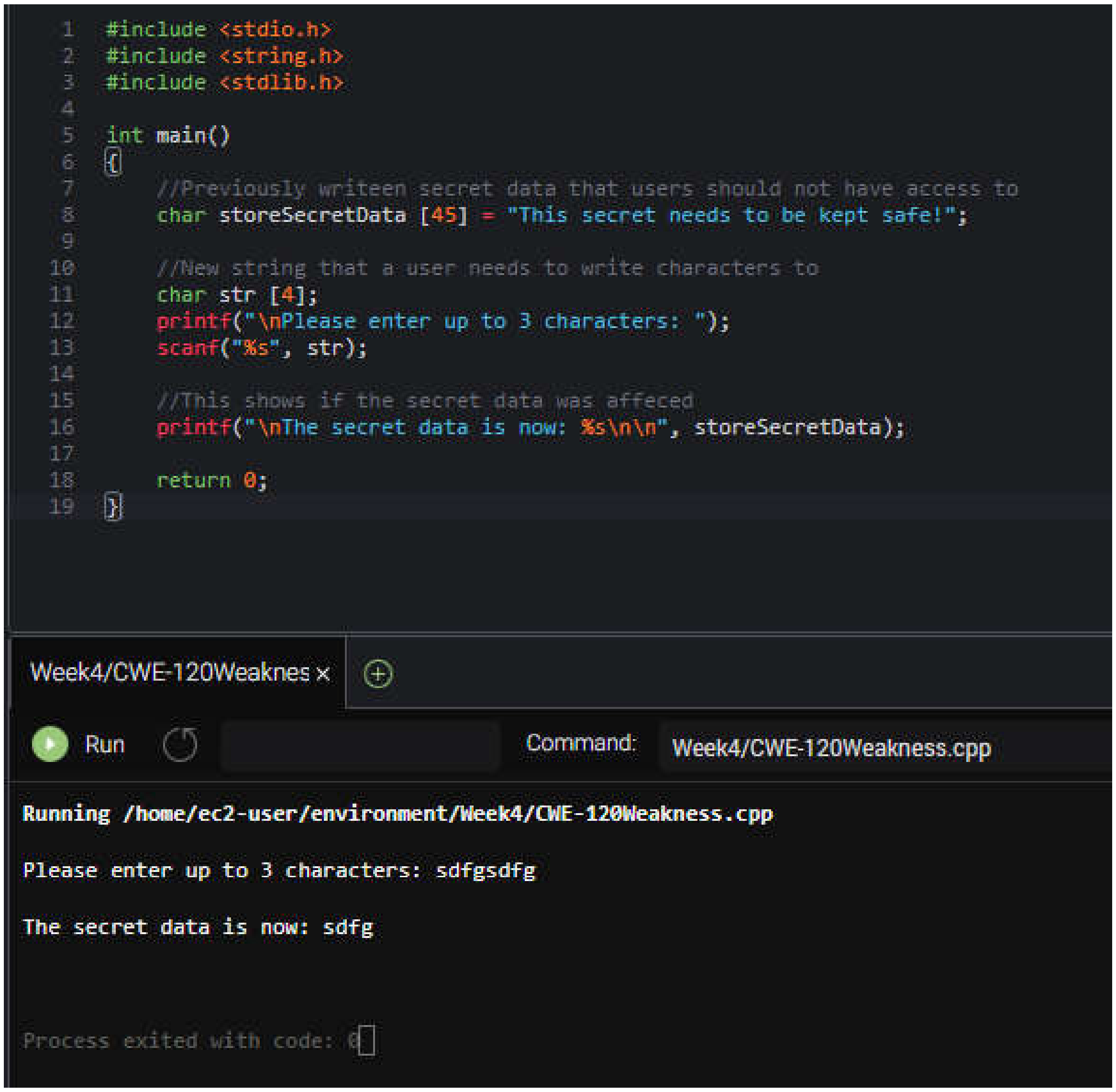
SDEV-325

September 15, 2020

1. The first software vulnerability that was addressed was CWE-120: Buffer Copy without Checking Size of Input ('Classic Buffer Overflow'). "A buffer overflow condition exists when a program attempts to put more data in a buffer than it can hold, or when a program attempts to put data in a memory area outside of the boundaries of a buffer. The simplest type of error, and the most common cause of buffer overflows, is the "classic" case in which the program copies the buffer without restricting how much is copied. Other variants exist, but the existence of a classic overflow strongly suggests that the programmer is not considering even the most basic of security protections (*Common Weakness Enumeration,* 2020)." An example of this would be if a user was prompted to enter three characters but entered 4 or more. If the program does not catch that, the buffer may leak that data and overwrite something that the user is not supposed to have access to. 1a shows the result of the weakness occurring and overwriting the secret data. 1b shows the result of the weakness being mitigated.

2. The second software vulnerability that was addressed was CWE-676: Use of Potentially Dangerous Function. When a function that is potentially dangerous gets implemented, the danger increases as it may be implemented in multiple areas of that program or in multiple programs. This make it especially important to catch the vulnerability early. 2a shows an example of a function that allows an error to occur when too many digits are entered by the user. The error never gets caught so a vulnerability like this may cause the program to crash. 2b shows the mitigation of the vulnerability by making sure that the user can only enter 8 characters for their password.

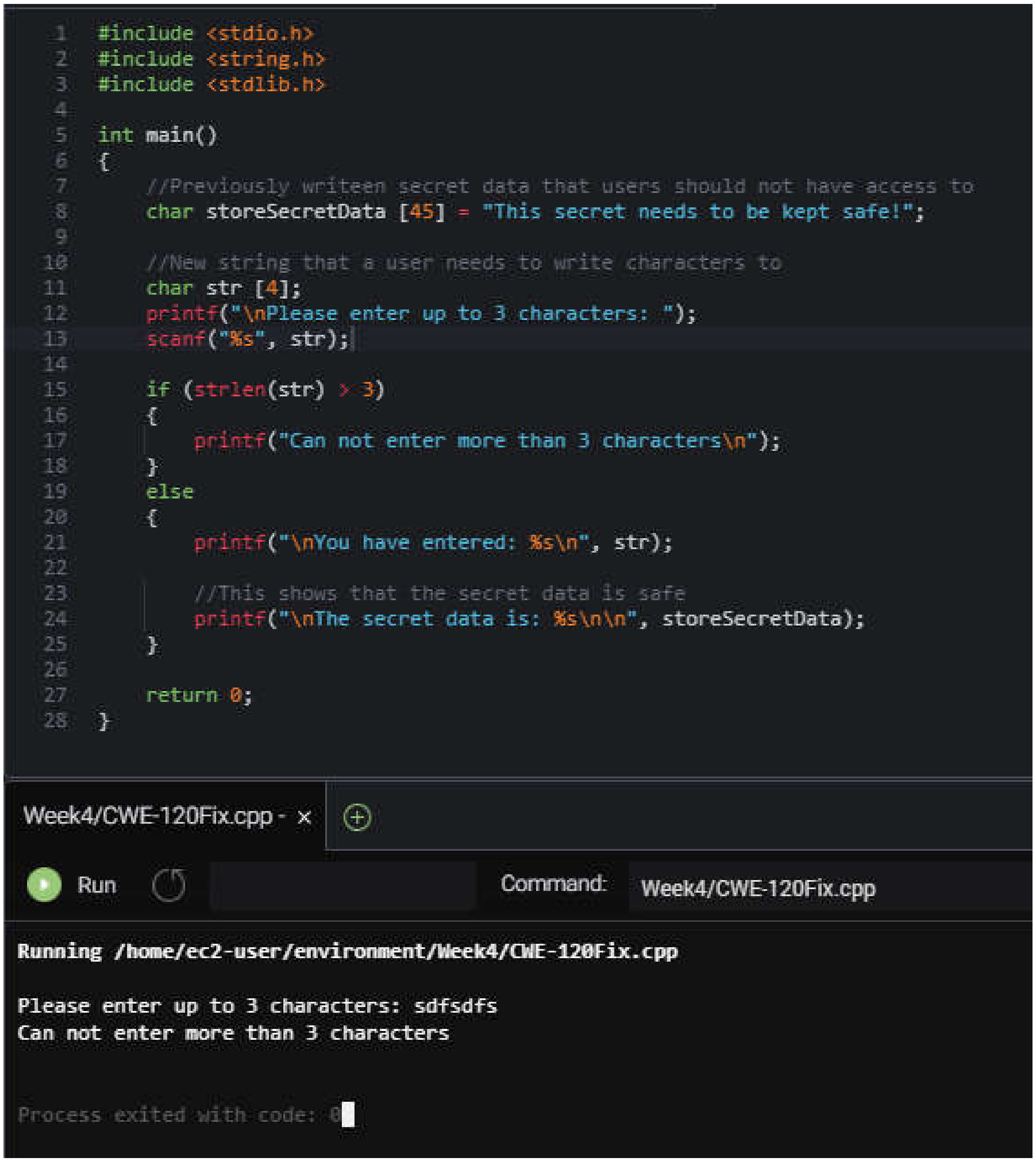
1a.



```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main()
{
    //Previously writeen secret data that users should not have access to
    char storeSecretData [45] = "This secret needs to be kept safe!";

    //New string that a user needs to write characters to
    char str [4];
    printf("\nPlease enter up to 3 characters: ");
    scanf("%s", str);

    //This shows if the secret data was affeced
    printf("\nThe secret data is now: %s\n\n", storeSecretData);

    return 0;
}
```

```
Running /home/ec2-user/environment/Week4/CWE-120Weakness.cpp

Please enter up to 3 characters: sdfgsdfg

The secret data is now: sdfg


Process exited with code: 0
```

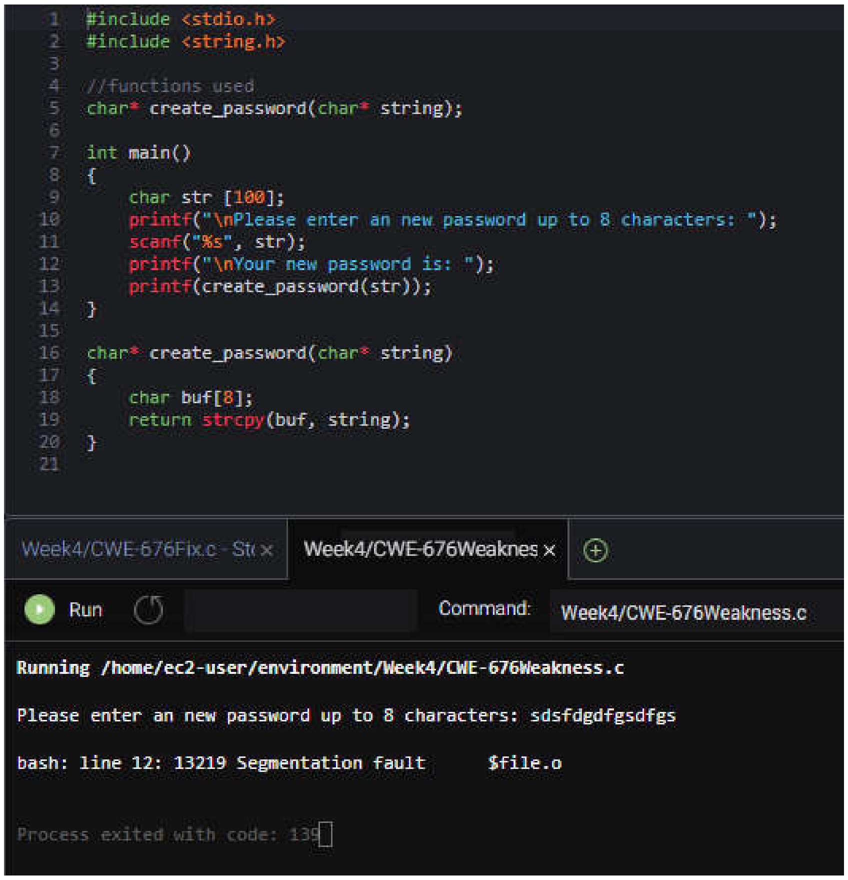
1b.



```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main()
{
    //Previously writeen secret data that users should not have access to
    char storeSecretData [45] = "This secret needs to be kept safe!";

    //New string that a user needs to write characters to
    char str [4];
    printf("\nPlease enter up to 3 characters: ");
    scanf("%s", str);

    if (strlen(str) > 3)
    {
        printf("Can not enter more than 3 characters\n");
    }
    else
    {
        printf("\nYou have entered: %s\n", str);

        //This shows that the secret data is safe
        printf("\nThe secret data is: %s\n\n", storeSecretData);
    }

    return 0;
}
```

```
Running /home/ec2-user/environment/Week4/CWE-120Fix.cpp

Please enter up to 3 characters: sdfsdfs
Can not enter more than 3 characters


Process exited with code: 0
```

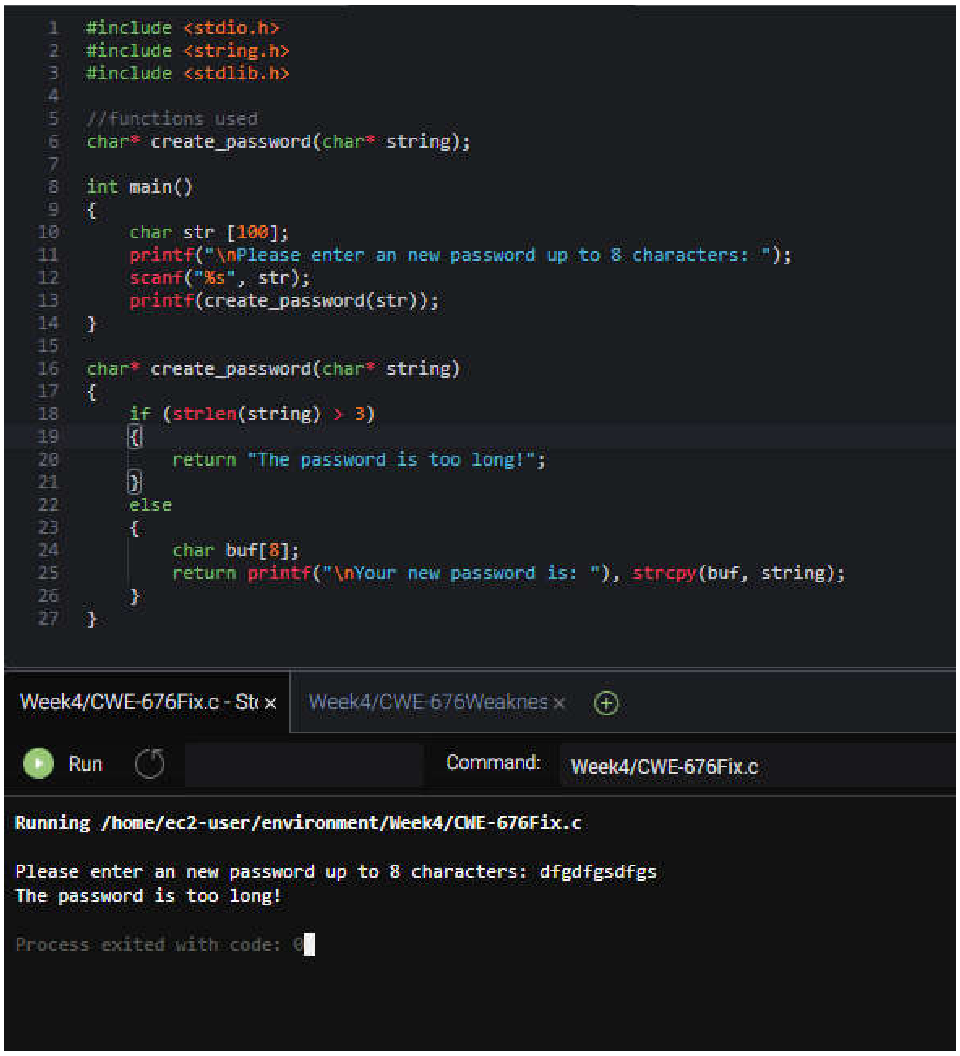
2a.



```c
#include <stdio.h>
#include <string.h>

//functions used
char* create_password(char* string);

int main()
{
    char str [100];
    printf("\nPlease enter an new password up to 8 characters: ");
    scanf("%s", str);
    printf("\nYour new password is: ");
    printf(create_password(str));
}

char* create_password(char* string)
{
    char buf[8];
    return strcpy(buf, string);
}
```

```
Running /home/ec2-user/environment/Week4/CWE-676Weakness.c

Please enter an new password up to 8 characters: sdsfdgdfgsdfgs

bash: line 12: 13219 Segmentation fault      $file.o


Process exited with code: 139
```

2b.



```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

//functions used
char* create_password(char* string);

int main()
{
    char str [100];
    printf("\nPlease enter an new password up to 8 characters: ");
    scanf("%s", str);
    printf(create_password(str));
}

char* create_password(char* string)
{
    if (strlen(string) > 3)
    {
        return "The password is too long!";
    }
    else
    {
        char buf[8];
        return printf("\nYour new password is: "), strcpy(buf, string);
    }
}
```

```
Running /home/ec2-user/environment/Week4/CWE-676Fix.c

Please enter an new password up to 8 characters: dfgdfgsdfgs
The password is too long!

Process exited with code: 0
```

References:

Common Weakness Enumeration. (2020, August 20). Retrieved September 14, 2020, from https://cwe.mitre.org/data/definitions/