漏洞解析之 5：缓冲区越界读取

1 概述

应用程序从缓冲区读取数据，如果缓冲区的偏移位于其所分配的范围之外时，将发生“缓冲区越界读取”。

2 后果

- 系统崩溃
- 敏感信息泄露

4 示例

- 示例 1

```
void CWE126_Buffer_Overread__char_alloca_loop_01_bad()
{
    char * data;
    char * dataBadBuffer = (char *)ALLOCA(50*sizeof(char));
    char * dataGoodBuffer = (char *)ALLOCA(100*sizeof(char));
    memset(dataBadBuffer, 'A', 50-1); /* fill with 'A's */
    dataBadBuffer[50-1] = '\0'; /* null terminate */
    memset(dataGoodBuffer, 'A', 100-1); /* fill with 'A's */
    dataGoodBuffer[100-1] = '\0'; /* null terminate */
    /* FLAW: Set data pointer to a small buffer */
    data = dataBadBuffer;
    {
        size_t i, destLen;
        char dest[100];
        memset(dest, 'C', 100-1);
        dest[100-1] = '\0'; /* null terminate */
        destLen = strlen(dest);
        /* POTENTIAL FLAW: using length of the dest where data
          * could be smaller than dest causing buffer overread */
        for (i = 0; i < destLen; i++)
        {
            dest[i] = data[i];
        }
        dest[100-1] = '\0';
        printLine(dest);
    }
}
```

- 示例 2

```
void CWE126_Buffer_Overread__malloc_char_loop_01_bad()
{
```

```
        char * data;
        data = NULL;
        /* FLAW: Use a small buffer */
        data = (char *)malloc(50*sizeof(char));
        if (data == NULL) {exit(-1);}
        memset(data, 'A', 50-1); /* fill with 'A's */
        data[50-1] = '\0'; /* null terminate */
        {
            size_t i, destLen;
            char dest[100];
            memset(dest, 'C', 100-1);
            dest[100-1] = '\0'; /* null terminate */
            destLen = strlen(dest);
            /* POTENTIAL FLAW: using length of the dest where data
              * could be smaller than dest causing buffer overread */
            for (i = 0; i < destLen; i++)
            {
                dest[i] = data[i];
            }
            dest[100-1] = '\0';
            printLine(dest);
            free(data);
        }
    }
```

- 示例 3

```
    void CWE127_Buffer_Underread__char_alloca_cpy_01_bad()
    {
        char * data;
        char * dataBuffer = (char *)ALLOCA(100*sizeof(char));
        memset(dataBuffer, 'A', 100-1);
        dataBuffer[100-1] = '\0';
        /* FLAW: Set data pointer to before the allocated memory buffer */
        data = dataBuffer - 8;
        {
            char dest[100*2];
            memset(dest, 'C', 100*2-1); /* fill with 'C's */
            dest[100*2-1] = '\0'; /* null terminate */
            /* POTENTIAL FLAW: Possibly copy from a memory location located
    before the source buffer */
            strcpy(dest, data);
            printLine(dest);
        }
    }
```

5 应对

- 在从缓冲区读取数据前要对地址偏移进行必要的检查

6 相关漏洞

发生此漏洞后，后继可能会发生“使用越界指针偏移”、“访问未初始化指针”、“使用失效指针”、“缓冲区溢出”等漏洞