Abysssec Research

1) Advisory information

Title	: Excel RTD Memory Corruption
Version	: Excel 2002 sp3
Analysis	: http://www.abysssec.com
Vendor	: http://www.microsoft.com
Impact	: Critical
Contact	: shahin [at] abysssec.com , info [at] abysssec.com
Twitter	: @abysssec
CVE	: CVE-2010-1246

2) Vulnerable version

Microsoft Office Excel 2002 Service Pack 3

3) Vulnerability information

Class

1- Stack overflow

Impact

The vulnerability is caused by a stack overflow error when processing malformed RTD (recType 0x813) records, which could be exploited by attackers to execute arbitrary code by tricking a user into opening a specially crafted Excel document.

Remotely Exploitable

Yes

Locally Exploitable

Yes

4) Vulnerabilities detail

This vulnerability is a stack overflow exists in the processing of RTD record fields. RTD record is a FRT type that introduced in Excel xp. For each RTD, there is a RealTimeData record in Workbook. Every RealTimeData contains subject title, RTD data and an array of RTDE structures which explain collection of related cells. This record can be continued by some CONTINUEFRT record.

Here are the fields of this record:

Offset	Name	Size	Contents
4	rt	2	Record type; this matches the BIFF rt in the first two bytes of the record; =0813h
6	grbitFrt	2	FRT flags; must be zero
8	ichSamePrefix	4	Number of leading characters in common with the previous Topic string (implicitly understood, and not to be repeated in this record); basically the length of any common prefix between the Topic of this record and the Topic of the previous REALTIMEDATA record. Zero if there is no prefix in common, or if this is the first REALTIMEDATA record.
12	cchTopic	4	Count of characters in the Topic string, not including the implicit prefix if

			ichSamePrefix is greater than zero.
16	rgchTopic	var	Topic string, not including the implicit prefix, if any. May be encoded as a compressed or uncompressed Unicode string. (See section titled _Unicode Strings in Biff8`for more information about these encodings.)
var	RTDOPER	var	RTDOPER contains variant type and data of RTD data (similar to but not identical to OPER structure used elsewhere)
var	rgRTDE	var	Variable-length array of RTDE structures, describing the set of cells associated with the RTD topic. Each RTDE contains row, column, and sheet tab index. Length of array determined by record size of this record and any CONTINUEFRT records.

For the purpose of creating RealTimeData record in excel file we can us RTD function. RTD function retrieves data from a COM server at real time. This function use COM technology for this purpose.

Here is the syntax of using this function:

```
RTD(RealTimeServerProgID,ServerName,Topic1,[Topic2], ...)
```

The first argument is a string that specify ID of installed program on local RTD server.

The second argument is name of RTD server as string. If it is a local server it can be "".

The third argument and the next arguments are strings that specify data that should be retrieved. This function can retrieve up to 28 subjects.

For example:

```
RTD("MyRTDServerProdID","MyServer","RaceNum","RunnerID","StatType")
```

To use this function in excel file, choose a cell and add the function after '=' operator.

```
=RTD("MyRTDServerProdID","MyServer","RaceNum","RunnerID","StatType")
```

Now if you save this excel file with xls extension RealTimedata record will be generated.

Our examinations show that sub_3041A187 function is responsible for processing RealTimeData(RTD) record. This function takes two arguments, an address of the content of the record and length of the record. In the body of this function, sub_3041A0B1 function is called multiple times which copies some values to a buffer in a known location. The first argument is length of bytes to be copied and second is a pointer to buffer.

In one of the calls to sub_3041A0B1 function, value of RTDOPER field from RTD record is stored in a buffer as 4byte.

```
3041A3AB    PUSH 4
3041A3AD    LEA EAX,DWORD PTR SS:[EBP-10]
3041A3B0    POP ESI
3041A3B1    PUSH ESI
3041A3B2    PUSH EAX
3041A3B3    CALL EXCEL.3041A0B1
3041A3B8    MOV EAX,DWORD PTR SS:[EBP-10]
3041A3BB    MOV EDI,7F0
3041A3C0    DEC EAX
3041A3C1    JE SHORT EXCEL.3041A41E
3041A3C3    DEC EAX
3041A3C4    JE SHORT EXCEL.3041A3E6
3041A3C6    DEC EAX
3041A3C7    DEC EAX
3041A3C8    JE SHORT EXCEL.3041A3E3
3041A3CA    SUB EAX,0C
3041A3CD    JE SHORT EXCEL.3041A3E3
3041A3CF    SUB EAX,EDI
3041A3D1    JE SHORT EXCEL.3041A3E3
3041A3D3    SUB EAX,800
3041A3D8    JE SHORT EXCEL.3041A3E6
3041A3DA    AND DWORD PTR SS:[EBP-18],0
3041A3DE    MOV DWORD PTR SS:[EBP-10],ESI   →   ESI = 4
```

As above code demonstrate, after reading 4bytes it is compared with 1, 2, 4, 16, 2048, 4096 and if not equal it will be set to 4. Now if value of these 4bytes is not equal to the above constant values the execution flow reach to a loop. The loop first checks length of remaining bytes of RTD record which is not read and in case of greater than zero, sub_3041A0B1 function will be called three times with 2 as its second argument. It means three 2bytes value is read.

```
3041A4CA    CALL EXCEL.3041A14D
3041A4CF    TEST EAX,EAX
3041A4D1    JNZ EXCEL.3041A568
3041A4D7    CALL EXCEL.304C5077
3041A4DC    MOV ESI,EAX
3041A4DE    PUSH 2
3041A4E0    LEA EAX,DWORD PTR DS:[ESI+14]
3041A4E3    PUSH EAX
3041A4E4    CALL EXCEL.3041A0B1
3041A4E9    LEA EAX,DWORD PTR SS:[EBP-3C]
3041A4EC    PUSH 2
3041A4EE    PUSH EAX
3041A4EF    CALL EXCEL.3041A0B1
3041A4F4    MOV AX,WORD PTR DS:[ESI+16]
```

```
3041A4F8        PUSH 2
3041A4FA        MOV ECX,EAX
3041A4FC        XOR ECX,DWORD PTR SS:[EBP-3C]
3041A4FF        AND CX,7FFF
3041A504        XOR ECX,EAX
3041A506        LEA EAX,DWORD PTR SS:[EBP+E]
3041A509        PUSH EAX
3041A50A        MOV WORD PTR DS:[ESI+16],CX
3041A50E        CALL EXCEL.3041A0B1
. . .
3041A550        JMP EXCEL.3041A4CA
3041A555        MOV EAX,DWORD PTR DS:[307DAB54]
3041A55A        MOV DWORD PTR DS:[ESI+8],EAX
3041A55D        MOV DWORD PTR DS:[307DAB54],ESI
3041A563        JMP EXCEL.3041A4CA
```

The above code show this loop. The EXCEL.3041A14D function control length of bytes that is not read from RTD record. If this function returns zero, means length of bytes that is not read is greater than zero. Next step sub_3041A0B1 function is called three times. The point is bytes of record that are not read from RTD record is only checked at the beginning of the function. If two bytes are remained after first sub_3041A0B1 call, there would be no unread bytes. In this case the next function call can be problematic.

```
3041A0B1        MOV EAX,DWORD PTR DS:[307E6194]
3041A0B6        PUSH EBX
3041A0B7        PUSH ESI
3041A0B8        MOV ESI,DWORD PTR SS:[ESP+10]
3041A0BC        CMP ESI,EAX
3041A0BE        JG SHORT EXCEL.3041A0E1
3041A0C0        PUSH ESI
3041A0C1        PUSH DWORD PTR DS:[307E6198]
3041A0C7        PUSH DWORD PTR SS:[ESP+14]
3041A0CB        CALL EXCEL.30002A8A
3041A0D0        ADD DWORD PTR DS:[307E6198],ESI
3041A0D6        ADD ESP,0C
3041A0D9        SUB DWORD PTR DS:[307E6194],ESI
3041A0DF        JMP SHORT EXCEL.3041A148
3041A0E1        MOV EBX,DWORD PTR SS:[ESP+C]
3041A0E5        PUSH EAX
3041A0E6        PUSH DWORD PTR DS:[307E6198]
3041A0EC        PUSH EBX
3041A0ED        CALL EXCEL.30002A8A
3041A0F2        MOV EAX,DWORD PTR DS:[307E6194]
3041A0F7        AND DWORD PTR DS:[307E6194],0
3041A0FE        ADD DWORD PTR DS:[307E6198],EAX
3041A104        SUB ESI,EAX
3041A106        ADD ESP,0C
3041A109        ADD EBX,EAX
3041A10B        TEST ESI,ESI
3041A10D        JLE SHORT EXCEL.3041A148
3041A10F        PUSH EDI
3041A110          CALL EXCEL.307B302B
3041A115        PUSH DWORD PTR DS:[307E6194]
3041A11B        PUSH ESI
3041A11C        CALL EXCEL.30001A9B
3041A121        MOV EDI,EAX
3041A123        PUSH EDI
3041A124        PUSH DWORD PTR DS:[307E6198]
3041A12A        PUSH EBX
3041A12B          CALL EXCEL.30002A8A
3041A130        ADD DWORD PTR DS:[307E6198],EDI
3041A136        SUB DWORD PTR DS:[307E6194],EDI
3041A13C        SUB ESI,EDI
3041A13E        ADD ESP,0C
3041A141        ADD EBX,EDI
3041A143        TEST ESI,ESI
3041A145        JG SHORT EXCEL.3041A110
3041A147        POP EDI
3041A148        POP ESI
3041A149        POP EBX
3041A14A        RETN 8
```

In the first line of this function, length of unread bytes that is stored at address 307E6194 is compared with 2 (second argument of this function). If content of address 307E6194 is equal to zero, the execution flow transfers to address 3041A0E1. Then sub_30002A8A function is called which is responsible for copying some certain values to a buffer. This function takes three arguments. First argument is a pointer to some buffer. Second argument is a pointer to bytes of record and third argument is length of bytes which should be read. Our third argument to sub_30002A8A function at current state is zero, so nothing will be copied.

Then sub_307B302B function is called. By calling this function value of next record and its length will be retrieved. Also length of the next record is stored at address 307E6194 and then 12 (c) is substituted from contents of 307E6194.

```
307B3038        CALL EXCEL.300ACD15
307B303D        JMP SHORT EXCEL.307B3044
307B303F        CALL EXCEL.306EEDB4
307B3044        CMP DWORD PTR DS:[307E2404],0
307B304B        MOV ESI,EAX
307B304D        JNZ SHORT EXCEL.307B3056
307B304F        CALL EXCEL.300ACD15
307B3054        JMP SHORT EXCEL.307B305B
307B3056        CALL EXCEL.306EEDA4
307B305B        MOV DWORD PTR DS:[307E6194],EAX
307B3060        MOV EAX,2020
307B3065        SUB DWORD PTR DS:[307E6198],EAX
...
307B308F        ADD DWORD PTR DS:[307E6198],0C
307B3096        ADD ESP,0C
307B3099        SUB DWORD PTR DS:[307E6194],0C
```

Now if length of next record is less than 12, Results of the last line of above code is a negative number and will be stored at address 307E6194.

Now we follow sub_3041A0B1 function. At next steps sub_30001A9B function is called and two arguments have passed to it. First argument is 2, and second argument is the contents of address 307E6194 which contain a negative number. Here is the implementation of this function:

```
30001A9B        MOV EAX,DWORD PTR SS:[ESP+4]
30001A9F        MOV ECX,DWORD PTR SS:[ESP+8]
30001AA3        CMP EAX,ECX
30001AA5        JL SHORT EXCEL.30001AA9
30001AA7        MOV EAX,ECX
30001AA9        RETN 8
```

This function compares two numbers and return smaller one. Problem here is considering these numbers as signed value. Because 2 is greater than this signed negative value, the negative value is returned.

Then sub_30002A8A will be called. The point here is third argument of this function is the returned valued of sub_30001A9B function which is a negative or very big number. As discussed earlier this argument specify length of bytes that should be copied and because of this big number, a buffer overflow occurs.

To create a proof of concept excel file we should only change the first two bytes RTDOPER field of RealTimeData record to some value such as EE.