System & Network Engineering

Forensic examination of Windows Live Messenger 2009 Extensible Storage Engine

IDS course project report, May 2009, Version 1.0

Wouter van Dongen: <wouter.vandongen@os3.nl>

Willem Toorop: <willem.toorop@os3.nl>

Joeri Blokhuis: <joeri.blokhuis@os3.nl>

Table of Contents

1 Introduction

Windows Live Messenger (WLM) - formally known as MSN Messenger and still referred to as simply MSN - is the most common used instant messaging client. In 2007 Wouter S. van Dongen researched Windows Live Messenger 8.0 and described what traces are left behind after the use of WLM 8.0 (Dongen, 2007). The current version of WLM (at time of writing) is version 2009 (build 14.0), which is the successor of WLM 8.0. This article focuses on the new contact storage mechanism of WLM 2009. WLM 2009 creates an Extensible Storage Engine database to store contact information.

Extensible Storage Engine (ESE), also known as JET Blue, is an Indexed Sequential Access Method (ISAM) data storage technology from Microsoft. ESE is notably a core of Microsoft Exchange Server and Active Directory. Its purpose is to allow applications to store and retrieve data via indexed and sequential access. Windows Mail and Desktop Search in the Windows Vista operating system also make use of ESE to store indexes and property information respectively (Wikipedia.org, 2009).

This article explains what traces are left behind in WLM ESE database and provides general insight into how the WLM ESE database works. The Python scripts that were created for analyzing and dumping the database are available at: https://www.os3.nl/2008-2009/students/wouter_van_dongen/WLM2009. These scripts can be used for further research into WLM or other ESE databases.

In the next chapter the used research method is described. The following chapter describes characteristics of the WLM Extensible Storage Engine. The subsequent chapter 4 describes the structure of the ESE database and the information that can be found within. Conclusions are given in chapter 5.

2 Methodology

The ESE database used by WLM 2009 is equal to the ESE database used by Microsoft Exchange 2007. In order to obtain general information about the ESE database such as the pagesize, table names, file header and page headers the Exchange Server Database Utilities tool (eseutil.exe) was used which was copied from a Mirosoft Exchange 2007 server. Windows Vista and XP are both shipped with similar tool called 'esentutl.exe' (stored in <drive>\Windows\System32\), however this tool provides less options compared to 'eseutil.exe'.

Furthermore WinHex (Available from: http://www.x-ways.net) was used to examine the files.

A python script was written using the Extensible Storage Engine API (references available from: http://msdn.microsoft.com/en-us/library/ms683072(VS.85).aspx) to open the databases and perform queries on them. Insight in the schema of the databases was gained by another python script revealing the possible relations within. The Python scripts were extended to allow easy monitoring of the differences made to the database during a WLM session.

Finally, we investigated the real time usage of the databases by WLM by spying on the ESE API with Deviare (available from: http://www.nektra.com/products/deviare/index.php) in combination with another Python script.

3 WLM Extensible Storage Engine

This section describes the ESE database as used within WLM. Research is done by indentifying what files are relating to the WLM database and are further researched on content and their values.

3.1 Files

3.1.1 Shared Computer Option

As with previous versions of Windows Live Messenger, WLM 2009 caches the contact list by default but still provides the possibility to disable caching. This can be done by selecting 'This is a shared computer so don't save my contact list or what's new information on it' under the security tab in the WLM options screen. The behaviour of this option has not changed since WLM 8.0. As described by van Dongen (2007): " *In the registry under the 'HKEY_CURRENT_USER\Software\Microsoft\MSNMessenger\PerPassportSettings\<MSN_Passport_ID>\Disable Cache' can be verified if caching is enabled. This registry key is only created if this option is enabled. If this key has the value '01', caching is disabled. When subsequently the option is disabled the value is set to 00. Because of this the conclusion can be made that if the value of the key is 00 the user has used the 'shared computer' option in the past and if the key does not exist the user might not have used this option or deleted the key.*

However, in order to enable the option 'shared computer' under the security tab in the options screen, the user will first need to login with the default settings. Because of this, contacts are first saved and while logging out – after enabling the shared computer option – removed. Due to this it could be possible to recover contacts from the free space and slack space or Windows swap file of the hard disk with the use of the known structure of the files.".

3.1.2 Location

On Windows Vista or Windows 7 the WLM ESE database files reside in the directory *'<system drive>:\Users\<user>\AppData\Local\Microsoft\Windows Live Contacts\<GUID>\DBStore*'. On Windows XP the files are located in '*<system drive>:\Documents and Settings\<user>\Local Settings\Application Data\Microsoft\Windows Live Contacts\<GUID>\DBStore*'. The 'Windows Live Contact' directory holds two directories per WLM account which are named by the Global Unique Identifier (GUID) algorithm. One of these GUID-named directories we will call the main ESE directory as the database files in this directory are primarily used by WLM. The other GUID-named directory holds the shadow ESE database which shows far less usage activity by WLM.

In the registry key '*HKEY_CURRENT_USER\Software\Microsoft\Windows Live Contacts\Database*' can be found to what WLM accounts (see key name) to what GUID-named directory belongs (see data of the key), and which directory is used as shadow directory, see Figure 1. Although the key names refer to the directory *'<system drive>:\Users\<user>\Contacts'*, no database or any other WLM file activity was monitored in this directory.

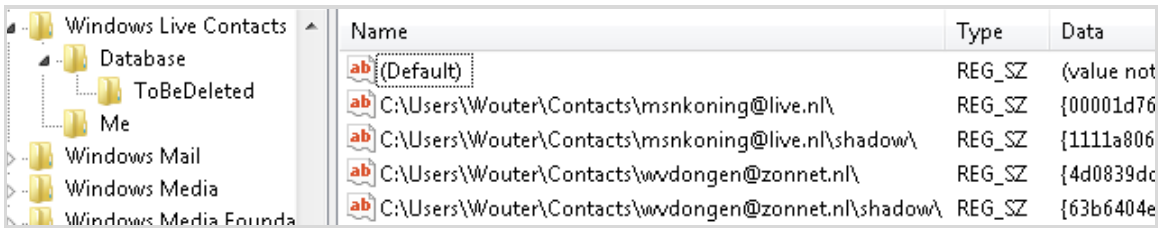


Figure 1. Example of the registry key '*HKEY_CURRENT_USER\Software\Microsoft\Windows Live Contacts\Database*' showing to WLM accounts and their GUIDs.

In case the user has used an older version of WLM prior to WLM 2009, the 'Windows Live Contacts' directory might also contain old contact files stored in separate directories named after the corresponding WLM account. These old contact files (members.stg, .WindowsLiveContact, .LiveContact, .MeContact) are not removed after uninstalling or upgrading the older version of WLM. Note that these files are not used by WLM 2009. However, these older contact files might be of forensic interest. More information on old contact files can be found in 'Forensic artefacts left by Windows Live Messenger 8.0'.

As with older versions of WLM, the WLM ESE directories and all underlying files are not removed by uninstalling Windows Live Messenger.

3.1.3 Structure and file overview

The tree below is an example of the WLM ESE directory structure and shows the kind of files that can be found. The main and shadow ESE directories have the same directory structure and files.

```
{ed0e046e-7457-42ee-a3bb-a2bfbd9725f6}
|-- DBStore
|   |-- Backup
|   |   `-- new
|   |       |-- contacts.edb
|   |       |-- contacts.pat
|   |       `-- edb00006.log
|   |-- LogFiles
|   |   |-- edb.chk
|   |   |-- edb.log
|   |   |-- edb00006.log
|   |   |-- edbres00001.jrs
|   |   `-- edbres00002.jrs
|   |-- Logs
|   |   |-- edb.chk
|   |   |-- edb.log
|   |   |-- edbres00001.jrs
|   |   `-- edbres00002.jrs
|   |-- contacts.edb
|   |-- contacts.pat
|   |-- dbstore.ini
|   |-- edb.chk
|   `-- tempedb.edb
`-- Logs
    |-- edb.chk
    |-- edb.log
    |-- edbres00001.jrs
    |-- edbres00002.jrs
    `-- edbtmp.log
```

Of all files listed in the tree the .EDB files are the actual databases. The file '*DBStore**Contacts.edb*' is the main database file in the GUID-directory and holds the most information. The directory '*DBStore**Backup**new*' contains a back-up version of the main contacts.edb. It is not clear when the backup is exactly updated. Process Monitor does not capture any on a specific activity such as signing in/out, changing profile information. However the modification date of the backup ESE shows the file is updated at least once a day when the user is logged into WLM. Tempedb.edb is a temporary workspace for processing transactions that is created upon signing into WLM and removed when signing out.

Although EDB files are the actual databases, the other files could also contain interesting information as these are used to keep the database consistent. Due to the limited available time for this project the contents of these file are not extensively examined. Table 1 will give a brief explanation about these files.

Table 1. Description of the ESE related files.

File	Description
Checkpoints (.chk)	Files with the extension .chk are referred to as checkpoint files. A CHK file is always 8-KB in size and is used as a pointer to a location in the log files. Data stored before a checkpoint is committed to the database, while data beyond the checkpoint is not.
Log Files (.log)	Log files are important to the database as all transactions are first written into logs, before being committed. Log files in WLM are always 4096-KB in size. When exceeding this size, the log file is renamed and a new edb.log is created.
JRS File(.jrs)	Files with the extension .jrs are reserved log files. These files are created in case the hard disk is about to run out or has ran out of space. By reserving files information should get stored for the time being
Patch files (.pat)	Patch files are generated when making backups of the database and record details when a page split occurs. A page split occasionally occur when data cannot fit into a single page.
tempedb.edb	This is a temporary workspace for processing transactions. tempedb.edb contains temporary information that is deleted when the WLM account has signed out.

When examining Windows Live Messenger, CHK and LOG files are considered to be a point of interest. The CHK file stores a checkpoint which points to a location in a LOG file. Any data from this point is not yet committed to the database, thereby it could contain valuable information.

3.2 Contacts.edb file characteristic

3.2.1 File size

The used database page size is 8192-bytes. The page size is important in order to be able to read the database. The minimum size of contacts.edb is 2064-KB, this is exactly 2-MB (2048-KB) for the database itself, including two pages (16-KB) containing the file header. Contacts.edb is extended by 2-MB when the available space is not sufficient. As an indication of the size of contacts.edb, a WLM account with 110 contacts has contacts.edb file of 6160-KB.

3.2.2 File header in detail

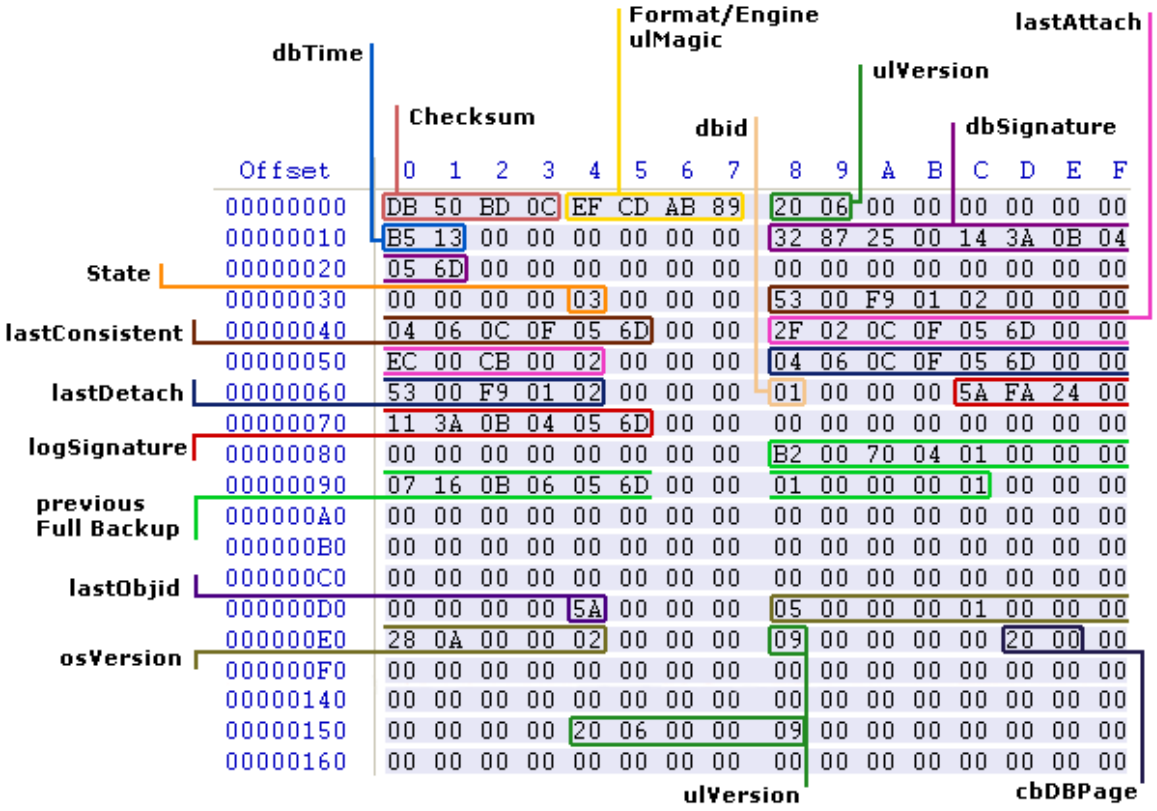
In order to distinguish the different ESE databases (main, shadow, back-up and tempedb) and reveal possible interesting information of the file header was examined. As mentioned in the methodology, the exchange tool 'ESEUTIL' can uncover more information about the database. The header of the database can be dumped with the '/mh' option. Dumping the file header of the real version of contact.edb with 'ESEUTIL' will output the information as shown in Table 2.

Table 2. File header information obtained by using ESEUTIL. The red values were empty outputted.

Entry	Description
Filetype	Database or Streaming file
Checksum	4-byte checksum of the database header
ulMagic	ESE identifier, hardcoded since Exchange 5.5 (Jim McBee, 2006)
ulVersion	ESE version
dbTime	Database change counter. Counter increments on every change made.
dbSignature	Creation time of the database and a unique random value identifying the database.
cbDBPage	Database page size (current 8192-bytes, previous version 4096-bytes)
State	Indicates state of database(consistent: “clean shutdown”) or non-consistent: “dirty shutdown”)
Log Required	Specifies a range of log files that are required to bring the database to a consistent state.
Log Committed	Specifies which log files are committed to the database
Streaming File	Indicates if the database has a streaming file (yes/no)
Shadowed	Indicates if the database has a shadow (yes/no)
Repair Count/Date	Amount of times the ESEUTIL has been executed with the option /P.
lastConsistent	Date and time of last clean shutdown
lastAttach	Date and time when database was mounted
lastDetach	Date and time when database was un-mounted
dbID	A unique sequential number provided to each database
logSignature	Indicates creation time and a random number to identify the transactions.
Previous Full Backup	Timestamp of last full backup
Previous Incremental Backup	Timestamp of backup which only save the changes made since the previous full backup of log files
Previous Copy Backup	Indicates date and time when a full backup was made without alternating header information
Previous Differential Backup	Indicates date and time when changes were made since the previous full backup, log files are not purged.
Current Shadow copy Backup	Indicates backup date and time of the shadow file
lastObjid	Current amount of B+ Trees in the database
osVersion	Version of Operating System

Figure 1 illustrates were these bytes are located in the file.

Header information is stored in Little-endian format. An ESE database file always starts with a 4-byte checksum of the file header, followed by an ESE identifier of 4-bytes (ulMagic) which is hardcoded and also used by other ESE database besides WLM. Next, the version number of the used ESE database follows which is always 0x20 06. The rest of all the header information is variable. The figure shows that ESEUTIL identified all of the header information. When examining more ESE databases it should be possible to determine the highlighted entries listed in Table 2.



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00000000	DB	50	BD	0C	EF	CD	AB	89	20	06	00	00	00	00	00	00
00000010	B5	13	00	00	00	00	00	00	32	87	25	00	14	3A	0B	04
00000020	05	6D	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	03	00	00	00	53	00	F9	01	02	00	00	00
00000040	04	06	0C	0F	05	6D	00	00	2F	02	0C	0F	05	6D	00	00
00000050	EC	00	CB	00	02	00	00	00	04	06	0C	0F	05	6D	00	00
00000060	53	00	F9	01	02	00	00	00	01	00	00	00	5A	FA	24	00
00000070	11	3A	0B	04	05	6D	00	00	00	00	00	00	00	00	00	00
00000080	00	00	00	00	00	00	00	00	B2	00	70	04	01	00	00	00
00000090	07	16	0B	06	05	6D	00	00	01	00	00	00	01	00	00	00
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000D0	00	00	00	00	5A	00	00	00	05	00	00	00	01	00	00	00
000000E0	28	0A	00	00	02	00	00	00	09	00	00	00	00	20	00	00
000000F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000140	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000150	00	00	00	00	20	06	00	00	09	00	00	00	00	00	00	00
00000160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure 1: Example of the file header of the contacts.edb file and the meaning of the bytes.

The header of contacts.edb contains several timestamps, such as creation time (dbSignature), last consistent, last attached and last detached. The six bytes containing the timestamp can be interpreted as follows: the first byte represents the seconds, the second byte the minute, the third byte the hour, the fourth byte the day, the fifth byte the month and the sixth byte the year. The year is stored in years since 1900. Figure 2 is an example of a timestamp extracted from the header.

Figure 2. Hexadecimal file header timestamp

As mentioned before there are four different ESE databases found that are used by WLM. Table 3 shows differences in the file header that are found in shadow, backup and tempedb compared to the main database. These differences could be useful for restoring a database.

Table 3. Header differences compared to the main ESE database.

Database	Difference with main database
Shadow	The timestamp written in DbSignature is always higher than the timestamp in the main database. The dbTime (amount of changes) is significant lower than the main database as far more activity takes place in the main database.
Backup of main/shadow	A backup will never contain a timestamp like full previous backup or previous differential backup. These timestamps can only be found in the main and shadow database.
Tempedb of main/shadow	Contains less header information compared to the other databases. In particular it misses a log signature (offset 6C), because no log files are created for this file.

3.2.3 Page header

Each page starts off with a 40-byte page header (Microsoft, 2009) which can be dumped with ESEUTIL. A page dump contains the following entries:

Table 4. Page header information obtained by using ESEUTIL.

Entry	Description
checksum	8-byte checksum of a page
dbTimeDirtied	Indicates dbTime
pgnoPrev	Indicates the page number of its neighbour on the left
pgnoNext	Indicates the page number of its neighbour on the right
objidFDP	Father of DataPage indicates the root
cbFree	Number of bytes available on the page
cbUncommitedFree	Indicates how many bytes are allocated for roll-back
ibMicFree	Indicates the offset for the next available byte at the top of the page
itagMicFree	Indicates the next tag to be free
fFlags	-

Figure 3 represents a hexadecimal view of a page header, containing the entries mentioned in table 4.

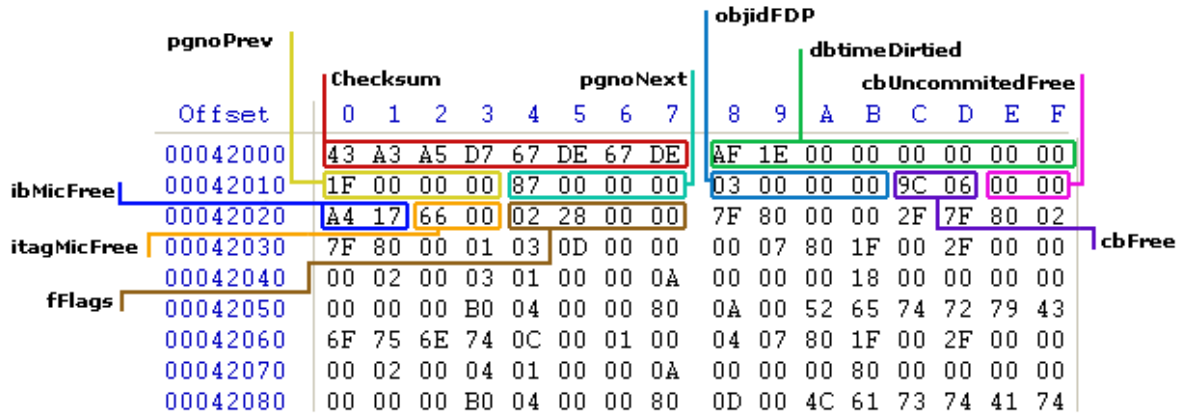


Figure 3. Example of the page header of the contacts.edb file and the meaning of the bytes.

Each page starts off with a checksum. Currently 8-byte checksums are used in Windows Vista/7 and 4-byte checksums in Windows XP. Windows XP is using the previous checksum of 4-bytes, which are followed by 4-bytes indicating the page number. The 8-byte checksum is called an Error Correcting Checksum (ECC) which has the advantage to correct single-bit errors on a page.

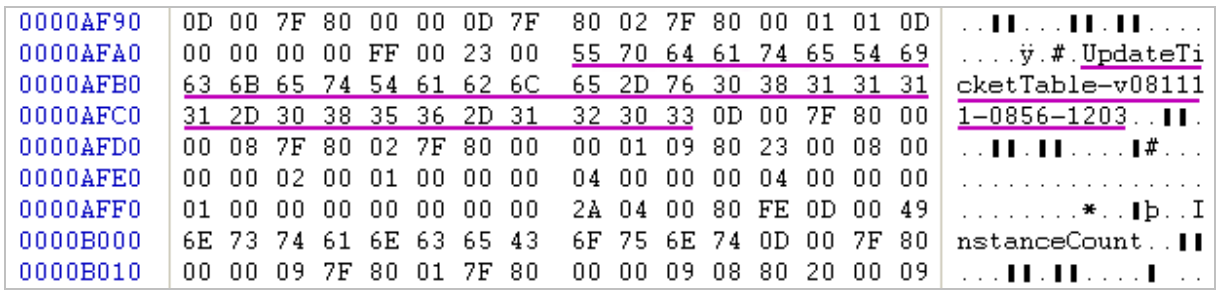
DbTimeDirtied indicates how many times a page was changed (data written to a page). When cycling through each page it should be possible to determine 'hot' pages. These are pages with the highest dbTime and therefore could contain potential valuable information.

As each page uses its own checksum to verify the integrity of that page, it is not possible to make any alternations without corrupting the page. Therefore it would be highly unlikely that there can be tampered with an offline database, as each page can be checked with the ESEUTIL tool.

3.3 Restoring

The Contacts.edb file is characterized by the following hex values which indicate the start of the file (header): <four variable bytes> EF CD AB 89 20 06 00 00 00 00 00 00. This pattern is repeated on the second page (offset 8192). Unfortunately contacts.edb has no specific end signature, but its size is limited to a factor of 2-MB whereby the size can be estimated and the file restored from the free space and slack space of the hard drive.

It is assumable that not only WLM ESE databases reside on a computer, because, for example, Windows Mail and Desktop search are also using the same technology. This could lead to recovering non WLM databases, when carving only on the hardcoded ESE identifier. To distinguish WLM databases from other ESE databases it is possible to search for the string: "UpdateTicketTable-v081111-0856-1203". "UpdateTicketTable-v081111-0856-1203"is believed to be the first unique table of a WLM ESE database. However due to limited time of this project this has not been verified.



```
0000AF90  0D 00 7F 80 00 00 0D 7F  80 02 7F 80 00 01 01 0D  ..▌▌...▌▌.▌▌....
0000AFA0  00 00 00 00 FF 00 23 00  55 70 64 61 74 65 54 69  ....ÿ.#.UpdateTi
0000AFB0  63 6B 65 74 54 61 62 6C  65 2D 76 30 38 31 31 31  cketTable-v08111
0000AFC0  31 2D 30 38 35 36 2D 31  32 30 33 0D 00 7F 80 00  1-0856-1203..▌▌.
0000AFD0  00 08 7F 80 02 7F 80 00  00 01 09 80 23 00 08 00  ..▌▌.▌▌....▌#...
0000AFE0  00 00 02 00 01 00 00 00  04 00 00 00 04 00 00 00  ................
0000AFF0  01 00 00 00 00 00 00 00  2A 04 00 80 FE 0D 00 49  ........*..▌þ..I
0000B000  6E 73 74 61 6E 63 65 43  6F 75 6E 74 0D 00 7F 80  nstanceCount..▌▌
0000B010  00 00 09 7F 80 01 7F 80  00 00 09 08 80 20 00 09  ...▌▌.▌▌....▌ ..
```

Figure 4. Assumed first unique value in WLM ESE database.

4 Analyzing the database

In this chapter the methods used to gain insight in the main and shadow databases are explored. First we will look at the information that can be gained by inspecting the two databases through the ESE API. We will describe its scheme and what tables and columns are actually used in the files we inspected. A method of finding the relations between the tables by looking at which values are used in which columns and records in which tables will be given. Then we will try to gain more insight in the usage of the databases by WLM by inspecting them before and after a WLM session and looking at the altered values. Finally we will present a method of inspecting the usage by spying on the usage of certain functions in `esent.dll` and logging each database transaction done through it.

4.1 General overview

4.1.1 Accessing the databases

Sysinternals "File Monitor" showed that the process accessing the `contact.edb` databases is `wlcomm.exe`, the Windows Live Communications Platform. The process is started when signing into WLM. By using Sysinternals "Process Explorer" it was found that `wlcomm.exe` is started by the "DCOM Server Process Launcher" service and not by `msnmsgr.exe`, the process that provides the user interface of WLM.

ESE databases can be accessed through functions in the `esent.dll` shared library. The ESE reference describes the C API for the functions in this DLL and the types of data they operate on. All our programs for analyzing the databases and its usage are written in the Python programming language. We accessed the functions and provided the appropriate data types with the Python ctypes module.

It was not possible to open any of the databases when an account was signed in. Therefore, it can conclude that WLM opens and locks the database. When the account is signed-off, the lock on the shadow database is released by `wlcomm.exe`. When no WLM activity was performed while signed in, the main database is also released and `wlcomm.exe` is terminated. Then it is possible to open both databases. However if there was WLM activity during the signed-in period, some of the accounts that were tested left the `wlcomm.exe` process open and it kept the main `contacts.edb` locked. WLM has to be closed completely to terminate it or it has to be killed manually (by using the task manager or another process manager).

4.1.2 Tables and columns

Using the ESE API it was possible to analyze the scheme and content of the databases.

The main and the shadow contact.edb use the same schema; having the same amount of tables, with the same names each having the same number of columns with the same names, of the same type and with the same column ids. Excluding the system tables (MSysObjects, MSysObjectsShadow and MSysDefrag1), the `contacts.edb` databases contain 29 tables. All table names end with the text '-v081111-0856-1203', it not clear what this means. This text could be a version number.

By looking at the column names some interesting properties can already be observed. These properties were also found and confirmed by running queries on the schema-data read by the Python script through the ESE API.

1. All tables except changeTable, streamTable, ContactIdTable and UpdateTicketTable have a column named RowId and a row named UpdateId.
2. Seventeen tables have nine columns in common. Fifteen of those tables are shown on the next page. The column names in red indicate the columns that they have in common.

Certificate-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Value	LongText
ValueType	Text
ThumbPrint	LongText
ThumbPrintType	Text

Date-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Value	DateTime

ContactID-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Value	Text

EmailAddress-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Address	LongText
Type	Text

IMAddress-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowId	Text
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Value	LongText
Protocol	Text

Name-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowId	Text
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
FormattedName	LongText
Phonetic	LongText
Prefix	Text
Title	Text
GivenName	Text
FamilyName	Text
MiddleName	Text
Generation	Text
Suffix	Text
NickName	LongText

Person-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
FormattedName	LongText
PersonID	LongText

PhoneNumber-v081111-0856-1203

RowId	Binary
vUpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Number	Text
Alternate	Text

Photo-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowId	Text
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Value	LongText
ValueType	Text
Url	LongText

PhysicalAddress-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
AddressLabel	LongText
Street	LongText
Locality	Text
Region	Text
PostalCode	Text
Country	Text
POBox	Text
ExtendedAddress	LongText

Position-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Organization	Text
Company	Text
Department	Text
Office	Text
JobTitle	Text
Profession	LongText
Role	Text

PresenceData-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowId	Text
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
OnlineStatus	Text
PersonalStatusMessage	LongText
ListeningTo	LongText
Private_ObjectStore-UserTileLocation	LongText
Private_ObjectStore-DynamicDisplayPicture-Location	LongText

SyncItem-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowId	Text
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
ServiceId	Text
ServerId	Text
UniqueId	Text
ChangeNumber	Text
RetryCount	Text
LastAttempted	Text
NextTryTime	Text
ReplicationErrorCode	Text
ReplicationErrorField	Text
IsReplicationError	Text
IsDeleted	Text

Url-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Value	LongText

WindowsLiveID-v081111-0856-1203

RowId	Binary
UpdateId	Binary
ContactRowId	Text
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
CID	Text
IsValueHidden	Text
Value	LongText

The two tables that are also in this group are CircleSpecific and GroupSpecific. These tables also have many columns in common with two other tables named CircleView and GroupView. They are shown

next to each other in Table 5 to indicate the columns in common. Besides this the table Level1Properties is shown, which has six of those initial nine columns in common. It does not have the NoneNumber, ElementId and Labels columns.

Table 5. Tables CircleSpecific, GroupSpecific, CircleView and Groupview

CircleSpecific-v081111-0856-1203	
RowId	Binary
UpdateId	Binary
ContactRowIdText	
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
Domain	Text
HostedDomain	Text
CircleType	Text
ProfileLastUpdated	Text
CreateDate	Text
IsPresenceEnabled	Text
MembershipAccess	Text
RequestMembershipOption	Text
CircleNotes	LongText
IsFamily	Text
IsFavorite	Text
SharedFolderInfo	Text
Roles	Text
Status	Text
IsNotMobileVisible	Text
DeltaMembershipTS	Text
LastDeltaMembershipTS	Text
VersionMembership	Text
BuildMembership	Text
LastFullMembershipTime	Text
HiddenContactId	Text
MembershipContactId	Text
CircleTicketVersionRequired	Text
NeedToSyncMembershipTS	Text
LastChangedValueDuringLast-	
Sync	Text
InviteMessage	LongText
InviterCID	LongText
InviterEmail	LongText
InviterName	LongText
JoinedDate	Text
IsBlocked	Text
StatusMessage	LongText
StatusState	Text
UBXInfo	LongText
UserTileLocation	LongText
UserTileFilePath	LongText
DesiredSignatureSound	LongText
DDPLocation	LongText
WallPaperColor	Text
WallPaperLocation	LongText
Scene	LongText
ColorScheme	Text
Online	Text
Members	Text

CircleView-v081111-0856-1203	
UniqueId	Binary
RowId	Binary
UpdateId	Binary
Name	Text
Domain	Text
HostedDomain	Text
CircleType	Text
ProfileLastUpdated	Text
CreateDate	Text
IsPresenceEnabled	Text
MembershipAccess	Text
RequestMembershipOption	Text
CircleNotes	LongText
Comments	LongText
IsFamily	Text
IsFavorite	Text
NickName	LongText
SharedFolderInfo	Text
Roles	Text
Status	Text
ServiceId	Text
ServerId	Text
ChangeNumber	Text
RetryCount	Text
LastAttempted	Text
NextTryTime	Text
ReplicationErrorCode	Text
ReplicationErrorField	Text
IsReplicationError	Text
IsDeleted	Text
IsNotMobileVisible	Text
DeltaMembershipTS	Text
LastDeltaMembershipTS	Text
VersionMembership	Text
BuildMembership	Text
LastFullMembershipTime	Text
HiddenContactId	Text
MembershipContactId	Text
CircleTicketVersionRequired	Text
NeedToSyncMembershipTS	Text
LastChangedValueDuringLast-	
Sync	Text
InviteMessage	LongText
InviterCID	LongText
InviterEmail	LongText
InviterName	LongText
JoinedDate	Text
IsBlocked	Text
StatusMessage	LongText
StatusState	Text
UBXInfo	LongText
UserTileLocation	LongText
UserTileFilePath	LongText
DesiredSignatureSound	LongText
DDPLocation	LongText
WallPaperColor	Text
WallPaperLocation	LongText
Scene	LongText
ColorScheme	Text
Online	Text
Members	Text

GroupSpecific-v081111-0856-1203	
RowId	Binary
UpdateId	Binary
ContactRowId	Text
NodeNumber	Text
PropertyVersion	Text
ModificationDate	Text
ElementId	Text
Labels	LongText
SimpleContactTarget	Text
IsFavorite	Text
IsPrivate	Text
IsNotMobileVisible	Text
DisplayInMessenger	Text
HasDisplayAnnotation	Text

GroupView-v081111-0856-1203	
UniqueId	Binary
RowId	Binary
UpdateId	Binary
Name	Text
ServiceId	Text
ServerId	Text
ChangeNumber	Text
RetryCount	Text
LastAttempted	Text
NextTryTime	Text
ReplicationErrorCode	Text
ReplicationErrorField	Text
IsReplicationError	Text
IsDeleted	Text
IsFavorite	Text
IsPrivate	Text
IsNotMobileVisible	Text
DisplayInMessenger	Text
HasDisplayAnnotation	Text

Level1Properties-v081111-0856-1203	
RowId	Binary
UpdateId	Binary
ContactRowIdText	
PropertyVersion	Text
ModificationDate	Text
SimpleContactTarget	Text
Notes	LongText
Mailer	Text
ProgId	Text
Gender	Text
CreationDate	Text

3. Two tables that are also very similar are SimpleContact and SimpleContactWrite. All SimpleContactWrite columns are present in the table SimpleContact as well. SimpleContact has an additional 56 columns. Their similarities are illustrated in Table 6. CircularContactView has all but two of its columns in common with SimpleContact.

Table 6. Tables SimpleContact, SimpleContactWrite, CircleContactView, ChangeTable, StreamTable, UpdateTicketTable.

SimpleContact-v081111-0856-1203	
UniqueId	Binary
RowId	Binary
UpdateId	Binary
ContactType	Text
StatusState	Text
Flags	Text
Name	LongText
CalculatedBuddyIdentifier	LongText
CalculatedPhoneNumber	Text
CalculatedIsBuddy	Text
RoamLiveProperties	Text
SuggestUpsellCount	Text
SuggestVideoCount	Text
SuggestVoiceCount	Text
SuggestUpsellTimer	Text
SuggestVideoTimer	Text
SuggestVoiceTimer	Text
CurrentMedia	LongText
$StatusMessage	LongText
$UserTileLocation	LongText
WallPaperColor	LongText
WallPaperLocation	LongText
CID	Text
DDPLocation	LongText
SendInviteMailText	LongText
ServiceId	Text
ServerId	LongText
ChangeNumber	Text
IsDeleted	Text
RetryCount	Text
LastAttempted	Text
NextTryTime	Text
IsReplicationError	Text
ReplicationErrorCode	Text
ReplicationErrorField	Text
IsAutoUpdateDisabled	Text
IsNotMobileVisible	Text
IsMobileEnabled	Text
HasSpace	Text
IsPrivate	Text
IsPassportNameHidden	Text
SendInviteMail	LongText
MsgrErrorCode	Text
IsMobileBuddy	Text
IsBusinessMobileBuddy	Text
IsLKPNeeded	Text
IsFavorite	Text
IsFQYNeeded	Text
IsSmtp	Text
FederatedEmailCategory	Text
IsFederatedBuddy	Text
FederatedEmailExternal	LongText
FederatedAddress	LongText
Messenger4Category	Text
IsMessenger4Buddy	Text
Messenger4External	LongText
Messenger4	LongText
OtherEmail	LongText
OtherIMEmail	LongText
PassportCategory	Text
IsBuddy	Text
MsnAddress	LongText
HomeEmail	LongText
UnknownEmailCategory	Text
IsUnresolvedBuddy	Text
UnresolvedEmail	LongText
WorkEmail	LongText
WvidCategory	Text
IsWvidBuddy	Text
WvidExternal	LongText
Wvid	LongText
BusinessMobilePhone	Text
CompanyName	Text
DefaultEmailType	Text
FirstName	Text
FullName	LongText
HomeFax	Text
HomePhone	Text
LastName	Text
MobilePhone	Text
$MobilePhoneCountryCode	Text
$FriendlyName	LongText
NickName	LongText
PublicDisplayName	LongText
OtherPhone	Text
Pager	Text

SimpleContactWrite-v081111-0856-1203	
UniqueId	Binary
RowId	Binary
UpdateId	Binary
ContactType	Text
Flags	Text
Name	LongText
RoamLiveProperties	Text
SuggestUpsellCount	Text
SuggestVideoCount	Text
SuggestVoiceCount	Text
SuggestUpsellTimer	Text
SuggestVideoTimer	Text
SuggestVoiceTimer	Text
WallPaperColor	LongText
WallPaperLocation	LongText
SendInviteMailText	LongText
ServiceId	Text
IsDeleted	Text
RetryCount	Text
LastAttempted	Text
NextTryTime	Text
IsReplicationError	Text
ReplicationErrorCode	Text
ReplicationErrorField	Text
IsAutoUpdateDisabled	Text
IsNotMobileVisible	Text
IsMobileEnabled	Text
HasSpace	Text
IsPrivate	Text
SendInviteMail	LongText
MsgrErrorCode	Text
IsMobileBuddy	Text
IsBusinessMobileBuddy	Text
IsLKPNeeded	Text
IsFavorite	Text
IsFQYNeeded	Text
IsSmtp	Text
FederatedEmailCategory	Text
IsFederatedBuddy	Text
FederatedEmailExternal	LongText
FederatedAddress	LongText
Messenger4Category	Text
IsMessenger4Buddy	Text
Messenger4External	LongText
Messenger4	LongText
PassportCategory	Text
IsBuddy	Text
UnknownEmailCategory	Text
IsUnresolvedBuddy	Text
UnresolvedEmail	LongText
WvidCategory	Text
IsWvidBuddy	Text
WvidExternal	LongText
Wvid	LongText
DefaultEmailType	Text
FullName	LongText
$MobilePhoneCountryCode	Text
$FriendlyName	LongText
PublicDisplayName	LongText

SimpleContact-v081111-0856-1203	
QuickName	Text
WorkFax	Text
WorkPhone	Text
DefaultPhoneType	Text
Anniversary	DateTime
Birthdate	DateTime
AssistantName	LongText
HomeCity	Text
Comments	LongText
HomeCountry	Text
DefaultLocationType	Text
EmailFormat	LongText
FileAs	LongText
Gender	Text
JobTitle	LongText
HomeLatitude	Text
HomeLongitude	Text
ManagerName	LongText
MiddleName	Text
Profession	LongText
SpouseName	LongText
HomeState	Text
HomeStreet	LongText
Suffix	Text
TimeZone	Text
Title	Text
$UserTileFilePath	LongText
HomeWebSite	LongText
WorkCity	Text
WorkCountry	Text
WorkDepartment	Text
WorkLatitude	Text
WorkLongitude	Text
WorkOffice	Text
WorkState	Text
WorkStreet	LongText
WorkWebSite	LongText
WorkZipCode	Text
XboxLiveTag	LongText
HomeZipCode	Text
DesiredSignatureSound	LongText
$Scene	LongText
$ColorScheme	Text
FavoriteOrder	Text
WL_DomainId	Text
WL_DomainTag	LongText
WL_UserTileURL	LongText
WL_ProfileURL	LongText
WL_DisplayName	LongText
WL_InviterCIDText	
WL_InviterEmail	LongText
WL_InviterName	LongText
WL_InviterMessage	LongText
WL_RelationshipType	Text
WL_RelationshipState	Text
WL_RelationshipStateDate	Text
WL_RelationshipRole	Text
WL_ExtendedData	LongText
WL_NDRCount	Text
WL_CreateDate	Text
WL_LastChanged	Text

SimpleContactWrite-v081111-0856-1203	
QuickName	Text
DefaultPhoneType	Text
DefaultLocationType	Text
EmailFormat	LongText
FileAs	LongText
HomeLatitude	Text
HomeLongitude	Text
TimeZone	Text
WorkLatitude	Text
WorkLongitude	Text
XboxLiveTag	LongText
DesiredSignatureSound	LongText
$Scene	LongText
$ColorScheme	Text
FavoriteOrder	Text
WL_DomainId	Text
WL_DomainTag	LongText
WL_UserTileURL	LongText
WL_ProfileURL	LongText
WL_DisplayName	LongText
WL_InviterCIDText	
WL_InviterEmail	LongText
WL_InviterName	LongText
WL_InviterMessage	LongText
WL_RelationshipType	Text
WL_RelationshipState	Text
WL_RelationshipStateDate	Text
WL_RelationshipRole	Text
WL_ExtendedData	LongText
WL_NDRCount	Text
WL_CreateDate	Text
WL_LastChanged	Text

changeTable-v081111-0856-1203	
changeNumber	Long
changeType	Long
changeRowId	Text
changeViewId	Binary

streamTable-v081111-0856-1203	
streamName	Binary
streamData	LongBinary

UpdateTicketTable-v081111-0856-1203	
InstanceCount	Long

CircleContactView-v081111-0856-1203		MsnAddress	LongText	WL_UserTileURL	LongText	WL_LastChanged	Text
UniqueId	Binary	IsPassportNameHidden	Text	WL_ProfileURL	LongText	FriendlyName	LongText
RowId	Binary	PublicDisplayName	LongText	WL_DisplayName	LongText	StatusMessage	LongText
UpdateId	Binary	Birthdate	DateTime	WL_InviterCIDText		UserTileLocation	LongText
NamespaceRowId	***Text***	ServiceId	Text	WL_InviterEmail	LongText	UserTileFilePath	LongText
ServerId	LongText	RetryCount	Text	WL_InviterName	LongText	DesiredSignatureSound	LongText
MembershipContactId	***Text***	LastAttempted	Text	WL_InviterMessage	LongText	DDPLocation	LongText
ChangeNumber	Text	NextTryTime	Text	WL_RelationshipType	Text	WallPaperColor	LongText
FirstName	Text	ReplicationErrorCode	Text	WL_RelationshipState	Text	WallPaperLocation	LongText
MiddleName	Text	ReplicationErrorField	Text	WL_RelationshipStateDate	Text	Scene	LongText
LastName	Text	IsReplicationError	Text	WL_RelationshipRole	Text	ColorScheme	Text
Suffix	Text	IsDeleted	Text	WL_ExtendedData	LongText		
Title	Text	WL_DomainId	Text	WL_NDRCount	Text		
CID	Text	WL_DomainTag	LongText	WL_CreateDate	Text		

4. From the remaining tables three have a very small number of columns and their names suggest a more operational purpose as opposed to storage. These tables are: changeTable, streamTable and UpdateTicketTable that are shown above.
5. There is another group of six tables that have eleven columns in common that are not found in any other table. Five of those tables have already been shown; These tables are CircleContactView, CircleView GroupView, SimpleContact and SyncItem. The one remaining table in this group is Member and its schema is shown in Table 7. The eleven columns are: ChangeNumber, IsDeleted, IsReplicationError, LastAttempted, NextTryTime, ReplicationErrorCode, ReplicationErrorField, RetryCount, ServerId, ServiceId and UniqueId and are type faced bold and italic. SimpleContactWriter has all those columns as well, except ChangeNumber and ServerId.

Table 7. Table Member.

Member-v081111-0856-1203		***IsDeleted***	***Text***	Role	Text	NetworkName	LongText
RowId	Binary	***ReplicationErrorCode***	***Text***	IsPassportNameHidden	Text	Keywords	LongText
UniqueId	***Binary***	***ReplicationErrorField***	***Text***	LocationId	Text	Notes	LongText
UpdateId	Binary	ServiceRef	Text	LocationIsPassportNameHidden	Text	InvitationType	LongText
ServiceId	***Text***	ServiceType	Text	LocationCID	Text	NetworkIconUrl	LongText
ServerId	***LongText***	ServiceForeignId	Text	LocationPassport	LongText	InviteMessage	LongText
ChangeNumber	***Text***	ServiceDisplayName	LongText	TargetRole	Text	PartnerEmail	LongText
Type	Text	ServiceLastChanged	Text	TargetRoleRecursion	Text	TargetPartnerAppId	LongText
Identifier	LongText	MemberId	Text	TargetRoleSeparation	Text	TargetPartnerScope	LongText
Status	Text	DisplayName	LongText	TargetServiceId	Text	JoinedDate	LongText
RetryCount	***Text***	Email	LongText	TargetServiceType	Text		
LastAttempted	***Text***	Passport	LongText	TargetServiceForeignId	Text		
NextTryTime	***Text***	Phone	Text	TargetServiceDisplayName	LongText		
IsReplicationError	***Text***	CID	Text	Category	LongText		

The two remaining tables, ContactId and Group, do not have a striking large number of uniquely shared columns in common with other tables. The column names PersonId, FormattedName, Phone and Email in table Group suggest that they refer to records in tables Person, Name, PhoneNumber and EmailAddress respectively, but we need to inspect the actual data to be able to confirm that.

Table 8. Tables ContactIdTable and Group.

ContactIdTable-v081111-0856-1203	
ContactId	Binary
CT_Contact	Bit
CT_Group	Bit
!CT_Member	Bit
!CT_Circle	Bit
!CT_Business	Bit
!CT_LMC_Me	Bit
!CT_LMC_AddressBook	Bit
!CT_LMC_Other	Bit

Group-v081111-0856-1203	
RowId	Binary
!MemberPath	Binary
GroupId	Binary
MemberId	Binary
UpdateId	Binary
!PersonId	LongText
!FormattedName	LongText
!Phone	Text
!Email	LongText
!PreferredLabel	LongText

4.1.3 Database schema

To gain insight in the database schema the key columns that uniquely identify a record and may be referenced by other tables are identified. For this purpose we have made an inventory of all the unique values in the databases and enumerated the tables, records and columns that contain those values.

RowIds are unique throughout the database. In some of our tested databases, the SimpleContactWrite table contained the same RowIds as SimpleContact and the same amount of records and basically mirrored the SimpleContact table RowIds; but this was not the case with all of our tested `contact.edb` files.

```
u'f0737bfc-0012-8703-11de-3e07fa101496'
  doncgambino@hotmail.com
    ContactIdTable-v081111-0856-1203 ContactId:Binary  55
    PresenceData-v081111-0856-1203   ContactRowId:Text 14
    WindowsLiveID-v081111-0856-1203  ContactRowId:Text 46
    Group-v081111-0856-1203          MemberId:Binary   7
    SyncItem-v081111-0856-1203       ContactRowId:Text 49
    SimpleContact-v081111-0856-1203  RowId:Binary      46

u'c64e1a0e-0010-8712-11de-4b618fcad4a0'
  doncgambino@hotmail.com
    ContactIdTable-v081111-0856-1203 ContactId:Binary  11
    GroupSpecific-v081111-0856-1203  ContactRowId:Text 1
    GroupSpecific-v081111-0856-1203  ElementId:Text    1
    Group-v081111-0856-1203          GroupId:Binary    5 6 7
    Name-v081111-0856-1203           ContactRowId:Text 2
    Name-v081111-0856-1203           ElementId:Text    2
    SyncItem-v081111-0856-1203       ContactRowId:Text 5
    SyncItem-v081111-0856-1203       ElementId:Text    5
    GroupView-v081111-0856-1203      RowId:Binary      1
```

An example of a database relations output. Record 7 of table Group connects record 46 of SimpleContact and record 1 of GroupView

Although all RowId columns are of the same type (binary) the data they contain are not formatted in the same manor. The tables SimpleContact, GroupView, CircleContactView and CircleView contain GUIDs formatted as UTF-16. Using the inventory of unique values it was possible to establish that with the exception of CircleContactView those are the tables that are referenced to, by other tables. Most references are by columns with the name ContactRowId and ElementId. They are of type Text and in the UTF-16 format.

Some of the columns that ended with the text 'Id' were of a binary type, but did not contain GUIDs in the UTF-16 format. Instead they contained 16 bytes. When those bytes are converted into GUIDs, three more columns could be found: ContactId of table ContactIdTable, and GroupId and MemberId of table Group. There is also a column named MemberId in table Member, but it is of a different type (Text) and doesn't reference to any RowId; they contain integer values formatted as UTF-16.

Converting the 16 byte binary values to GUIDs, care should be taken with the endianness of the values in the GUID. The first three values in a GUID are little-endian and the last two big, see Figure 5.

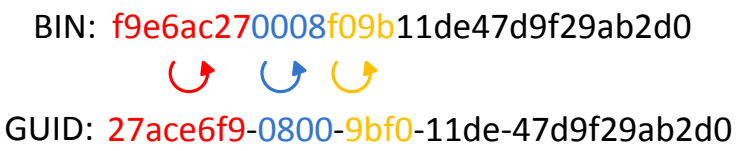


Figure 5. Binary conversion to GUID.

Evaluating the references the following database schema can be constructed:

1. CircleView, GroupView and Simplecontact appear to be the main tables to which all other tables are referring. They are all referred to by the other tables with the value of their RowId column; except CircleContactView which refers to SimpleContact by with and by the value of the CID column.
2. For each record in CircleView, GroupView and SimpleContact there is a record in ContactIdTable referring to it with its ContactId column. Their relation to the main table suggests that ContactIdTable contain values of a type that all the main tables have in common. The only column that contained data

in the inspected data, apart from ContactId, was CT_Group. CT_Group always contained the value True for every record in all the inspected databases.

3. Some of the records in the main tables are referred to by SyncItem. Sometimes just with the ContactRowId value and sometimes also with the ElementId. The ElementId refers to a record in the main tables only when ContactRowId is referring to it as well. When the ContactRowId is referring and ElementId not, ElementId does contain a GUID, but it does not refer to any other record in the database. If a SyncItem is referring to a main table record, sometimes a record from table Name is referring to it as well. When it does, it does so with its ContactRowId column and sometimes with its ElementId column as well just as SyncItem. If a SyncItem refers to a record with its ElementId and a Name table is referring to, it does not mean the Name table is referring with its ElementId as well. It is not clear what the meaning of these tables are. They suggest an operational function (Synchronizing data).

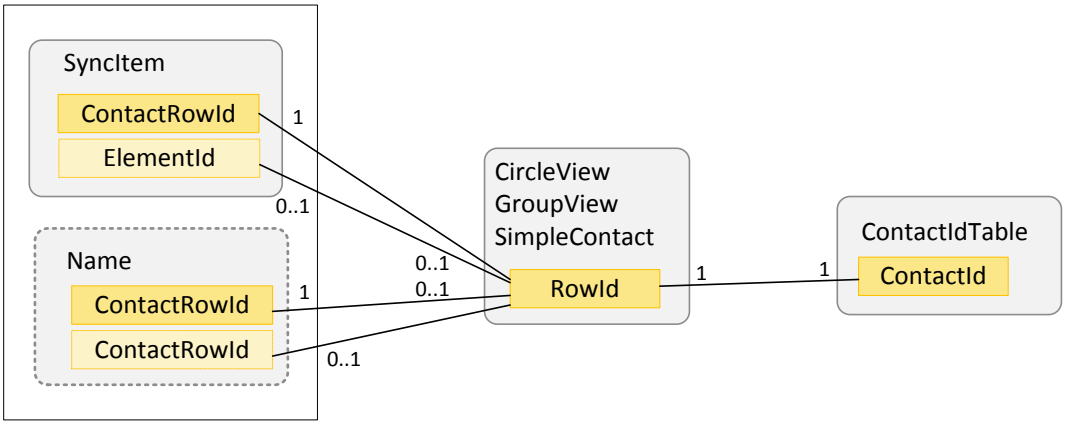


Figure 6. Relation between the tables SyncItem, Name ContactIdTable and the main tables.

4. SimpleContact reflects the contacts in the contact list shown in WLM. Those contacts can have properties that are optional. Sometimes more than one such property can be given for a contact. The property tables that were in the investigated databases are: EmailAddress, IMAddress, PhoneNumber, Photo, PresenceData and WindowsLiveId. PresenceData is probably used to store availability information, but that is never available when WLM is not running (and the databases can be opened). Although not found, it is likely that the tables Certificate, Date, PhysicalAddress, Position and URL perform a similar role as optional property for a contact.

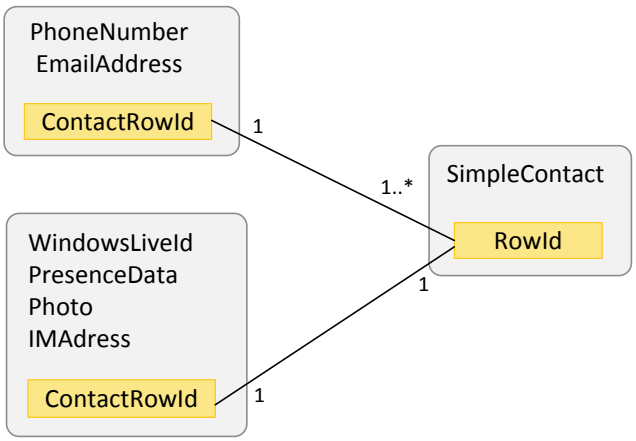


Figure 7. SimpleContact properties.

5. From the records found in table GroupView it shows that they represent the contact categories in WLM. They have a one-on-one relationship with table GroupSpecific with its ContactRowId and ElementId columns. The purpose that could be determined is that it stores the ModificationDate of a category.
   More than one Group table record can refer to a GroupView record with the value of its GroupId column. Group also refers to exactly one SimpleContact with its MemberId column. The Group table its purpose seems to be coupling one or more SimpleContact records to one GroupView record.

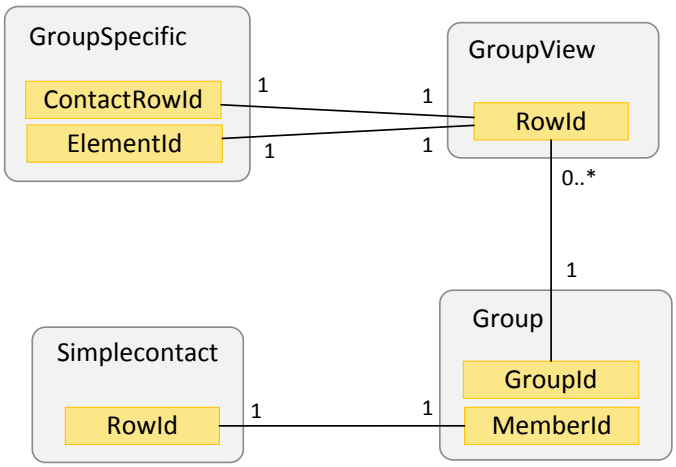


Figure 8. WLM categories.

6. The records in the CircleView table represent groups in WLM. Each group in WLM has a single CircleView record. There is an one-on-one relationship with table CircleSpecific and with table Level1Properties. CircleSpecific adds extra properties to a GroupView record, but Level1Properties seems to store a CircleViews modification date (just like GroupSpecific with GroupViews).
   Many CircleContactViews can refer to a single CircleView. The table CircleContactView records refer to a single SimpleContact record with their CID column. The purpose of CircleContactView thus is enlisting all SimpleContact records in a specific CircleView (WLM group).

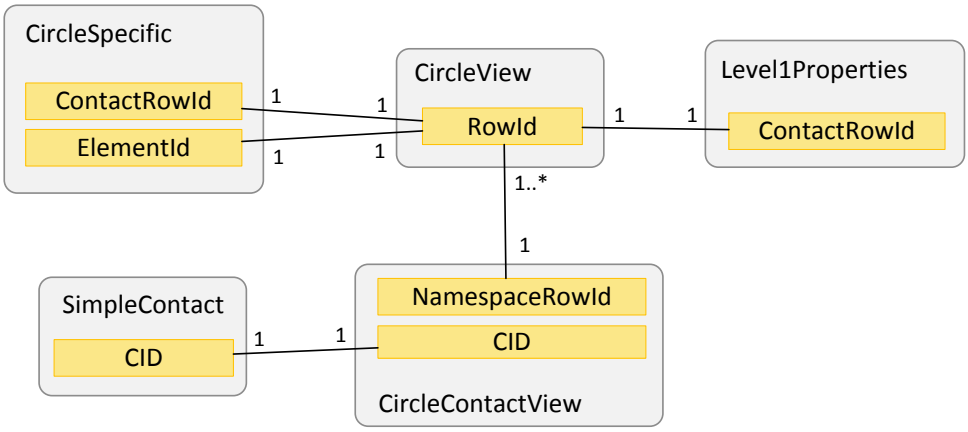


Figure 9. WLM groups.

4.1.4 Further research

Many properties of the database have not yet been investigated.

1. All relations in the main database are also present in the shadow database. Though from a first glance it is clear that the shadow database does not contain all the values that the main database has. It would be interesting to investigate what the differences are.
2. The streamData column in table streamData contains XML. From a first glance it is clear that many values stored within the XML structure also appear in the databases themselves. The program we wrote to find relationship in the database should be extended to process the values in the XML structure and refer to it as well.
3. Some of the XML blocks contain information about the WLM user itself (including screen names, display pictures, timestamps and version numbers). It is evident that this information is interesting to examine.

4.2 What data is changed

It is clear that much insight in the working of the databases could be gained by examining the differences between the database content before and after a WLM session. We have built two extra scripts to do this. One script shows the difference in the content of the database, the other the difference in the relations found by enumerating unique values and were they were found.

4.2.1 The database difference scripts

One of the things that immediately showed was the increase of the counter of the UpdateTicketTable by one, each time WLM was started and closed. Besides this, all altered records that are from tables with an UpdateId column, the UpdateId columns changed.

The last four bytes of an UpdateId seem to reflect the value of the InstanceCount column. The first four seem to contain a value that is unique for the main table the altered table relates to. Tables relating to GroupViews (and a GroupView itself) have the same unique initial four bytes and tables relating to SimpleContact each have the same four bytes in their UpdateIds which is differen from that relating to GroupViews.

The same holds for tables relating to CircleViews.

```
UpdateTicketTable-v081111-0856-1203 row 1/1 changed from:
          InstanceCount :Long = 161
UpdateTicketTable-v081111-0856-1203 row 1/1 changed to:
          InstanceCount :Long = 162

SimpleContact-v081111-0856-1203 row 11/12 changed from:
          UpdateId :Binary = b1040000a1000000
SimpleContact-v081111-0856-1203 row 11/12 changed to:
          UpdateId      :Binary   = 7f020000a2000000
          StatusMessage :LongText = u'hallo willem'

PresenceData-v081111-0856-1203 row 1/11 changed from:
          UpdateId         :Binary = b1040000a1000000
          PropertyVersion  :Text   = u'336'
          ModificationDate :Text   = u'2009-05-25T08:05:12Z'
PresenceData-v081111-0856-1203 row 1/11 changed to:
          UpdateId         :Binary = 7f020000a2000000
          PropertyVersion  :Text   = u'338'
          ModificationDate :Text   = u'2009-05-25T09:23:59Z'

GroupSpecific-v081111-0856-1203 row 1/2 changed from:
          UpdateId         :Binary = 58020000a1000000
          ModificationDate :Text   = u'2009-05-25T07:32:54Z'
GroupSpecific-v081111-0856-1203 row 1/2 changed to:
          UpdateId         :Binary = 54020000a2000000
          ModificationDate :Text   = u'2009-05-25T09:23:01Z'

GroupView-v081111-0856-1203 row 1/2 changed from:
          UpdateId     :Binary = 58020000a1000000
          ChangeNumber :Text   = u'2009-05-25T00:32:52.847-07:00'
GroupView-v081111-0856-1203 row 1/2 changed to:
          UpdateId     :Binary = 54020000a2000000
          ChangeNumber :Text   = u'2009-05-25T02:23:00.953-07:00'
```

An example of a database difference output.

Although insightful, the procedure to dump a database - start WLM, sign in, perform an action or let one of the contacts do something, close WLM, do a database dump again and compare the differences - is tedious. Therefore we decided to take a different approach and try to log the transactions with the ESE databases in real time.

4.2.2 Spying on `esent.dll` usage

There is no option available to log transactions to an ESE database. An ESE database is stored in a file with the .edb extension and is simply opened and used through the ESE API. The only method to monitor the real-time usage of an ESE database is thus by spying on the usage of the ESE API.

There are many approaches for spying a DLL. We have used the Deviare API from Nektra Advanced Computing (available from: http://www.nektra.com/products/deviare/index.php). Deviare provides COM interfaces to intercept the entry and exit point of a DLL function call. It is possible to alter the arguments given to a function and the value returned, but this was not required as we were merely investigating the modifications done to the database by ESE API usage by WLM.

The number and types of the arguments and the type of the return value can not be determined by Deviare or any other DLL spying method. Deviare provides a database in which the arguments and types can be defined for a certain function. The most important aspect of this definition is the size of a type in bytes, as this defines what the offset in the call-stack should be (see Figure 10).

There are only two functions in the ESE API that actually modify or add data: JetSetColumn and JetSetColumns. JetSetColomns has an array of structures as an argument. Each structure has values that reflect the arguments that would have been given on separate calls to JetSetColumn.

Which database and which tables are updated are specified with their IDs as arguments. To be able to identify which databases and tables are referred to by these Ids, the functions JetOpenDatabase and JetOpenTable had

to be spied upon as well. To be sure that all database Ids and table Ids are captured, the script has to be started before signed in with WLM.

The column for which the value is added or changed is also given as an Id. The Id of the column is not determined from an ESE API function call. It can by read from the MSysObjects system table, and stays throughout the database's lifetime. They have to be read from the database before WLM is started

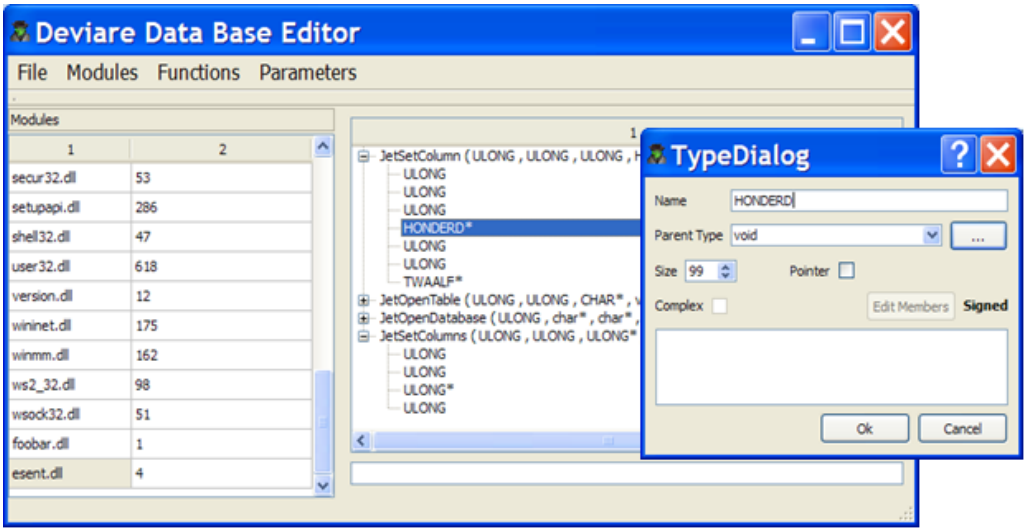


Figure 10. Specifying argument types in Deviare.

Table 9. An example of a log of the esent.dll spy script. A contact changes it state to 'online'.

```
**************************************************************************
****  timestamp: 2009-05-27 09:57:38.384999                            ****
****  database : willem@myahk.nl                                       ****
****  table    : PresenceData-v081111-0856-1203                        ****
****  column   : UpdateId   type: Binary                               ****
****  data     : 3E 01 00 00 A5 00 00 00                           >   ****
****  returnval: 0                                                     ****
**************************************************************************

**************************************************************************
****  timestamp: 2009-05-27 09:57:38.404999                            ****
****  database : willem@myahk.nl                                       ****
****  table    : PresenceData-v081111-0856-1203                        ****
****  column   : ContactRowId   type: Text                             ****
****  data     : u'27ace6f9-0800-9bc7-11de-37ccde4e81fd'               ****
****  ---------------------------------------------------------------  ****
****  column   : ElementId   type: Text                                ****
****  data     : u'4874ffcf-02ba-4eb2-a09e-c64a81e6e0ce'               ****
****  ---------------------------------------------------------------  ****
****  column   : Labels   type: LongText                               ****
****  data     : u'Preferred'                                          ****
****  ---------------------------------------------------------------  ****
****  column   : ModificationDate   type: Text                         ****
****  data     : u'2009-05-27T07:57:36Z'                               ****
****  ---------------------------------------------------------------  ****
****  column   : NodeNumber   type: Text                               ****
****  data     : u'1'                                                  ****
****  ---------------------------------------------------------------  ****
****  column   : OnlineStatus   type: Text                             ****
****  data     : u'1'                                                  ****
****  ---------------------------------------------------------------  ****
****  column   : PropertyVersion   type: Text                          ****
****  data     : u'361'                                                ****
****  ---------------------------------------------------------------  ****
****  returnval: 0                                                     ****
**************************************************************************

**************************************************************************
****  timestamp: 2009-05-27 09:57:38.585000                            ****
****  database : willem@myahk.nl                                       ****
****  table    : SimpleContact-v081111-0856-1203                       ****
****  column   : UpdateId   type: Binary                               ****
****  data     : 3E 01 00 00 A5 00 00 00                           >   ****
****  returnval: 0                                                     ****
**************************************************************************

**************************************************************************
****  timestamp: 2009-05-27 09:57:38.595000                            ****
****  database : willem@myahk.nl                                       ****
****  table    : SimpleContact-v081111-0856-1203                       ****
****  column   : StatusState   type: Text                              ****
****  data     : u'1'                                                  ****
****  ---------------------------------------------------------------  ****
****  returnval: 0                                                     ****
**************************************************************************
```

4.2.3 Behavior found

WLM 8.0 stored contact information encrypted (van Dongen, 2007); remarkably no encryption is used in WLM 2009 anymore. Spying on ESE API usage by WLM the following interesting forensic information was found which is presented in the table below. Columns listed the tables have been verified to be correct several times. Please note that by far not all columns are examined and documented.

Tabel: SimpelContact		stores contact information and contact
Behaviour	**Result**	
1 Row per contact		
Delete contact and remove from Windows Live Hotmail contactlist	Complete row with contact information is removed	
Field	**Value**	**Meaning**
General		
RowId	GUID	This is an unique value which is used to link data from other tables to the contact stored in this row. Other tables use the field 'ContactRowId' to do this.
CalculatedBuddyIdentifier	E-mailadress	This field is empty if the contact has been deleted, see the field MsnAdress. If not this field holds the e-mailadress of an contact.
CalculatedIsBuddy	0,1 or 5	0 - the contact has been deleted. 1 - The contact is is visible in the contact list. 5 - The contact has been added but has not yet been accepted, see field SendInviteMailText.
Friendlyname	Screenname	Contacts's screenname
MsnAddress	E-mailadress	contact's WLM account: e-mailadress
PublicDisplayName	Public (First) Name	First name from profile, visible to everybody
SendInviteMailText	Invite message	Message send by the user to the contact that the user upon adding a new contact.
StatusMessage	Quickmessage	Quickmessage that is display behind the Screenname
UserTitleLocation	msnobj	The displaypicture is stored in an 'msnobj' tag. The 'SHA1D' field holds the filename of the displaypicture which is stored under this name in '*<disk>:\Users\<user>\AppData\Local\Microsoft\Messenger\<WLM_account>\ObjectStore\UserTile'.* The Creator field holds the correspondig WLM account. ' Type = 3' signifies as displaypicture.
UserTitlePath	Displaypicture	Path to local cached contact dislaypicture
WallPaperLocation	Wallpaper image	A wallpaper image is set by the WLM user and is not visible to the contact. The wallpaper image is stored in an 'msnobj' tag. The 'SHA1D' field holds the filename of the wallpaper image. 'Type=5' signifies as static background image which is stored under SHA1D name in '<disk>:\Users\<user>\AppData\Local\Microsoft\Messenger\<WLM_account>\ObjectStore\Backgrounds'. 'Type=10' signifies a dynamic background which is stored under the SHA1D name in '<disk>:\Users\<user>\AppData\Local\Microsoft\Messenger\<WLM_account>\ObjectStore\DynamicBackgrounds'. The Creator field holds the correspondig WLM account.

Scene	msnobj	The scene image is stored in an 'msnobj' tag. The 'SHA1D' field holds the filename of the scene image which is stored under this name in '<disk>:\Users\<user>\AppData\Local\Microsoft\Messenger\<WLM_account>\ObjectStore\Scenes'. The Creator field holds the correspondig WLM account. ' Type = 16' signifies as scene.
Comments	Note	User added note to a contact
OnlineStatus	Online status	1 = offline, 2 = available, 10 = busy, 34 = away
CID	Integer	Causality identifier
WL_ProfileUrl	URL	The format of the URL to an users Windows-Live profile page is formated as: http://cid-<CID in hexadecimal format>.profile.live.com/
		The columns below are categorized as displayed in the Windows Live Profile of the contact. Most columns have a self-explanatory name
Contact info		
FirstName		
MiddleName		
LastName		
NickName		
MobilePhoneCountryCode		First two letters of the selected country
CalculatedPhoneNumber	Phone number	Personal Mobile phone number from profile
Personal Info		
HomeStreet		
HomeZipCode		
HomeCity		
HomeCountry		
HomeState		
HomeEmail		
HomePhone		
OtherPhone		
Anniversary	Double floatingpoint number	Fractional days since 1-1-1900
Birthdate		
HomeWebSite		
SpouseName		
Work info		
JobTitle		
Profession		
CompanyName		
WorkStreet		
WorkZipCode		
WorkCity		

WorkCountry		
WorkEmail		
WorkPhone		
WorkFax		
Pager	Work page number	

Tabel: Member		
Field	**Value**	**Meaning**
Type	passport	Contains the role which shows if a contact is blocked
Identifier	E-mail address	E-misaddress of WLM account
Passport	E-mail address	E-misaddress of WLM account
Role	Contact role	Allow= see your contacts online presence, Blocked = contacts don't receive online presence of you, Reverse= let contacts see your online presence or Pending=awaiting role state
Displayname	Screen name	Contact's screen name
InviteMessage	Message	Message send by the requesting contact, to be added

5 Conclusion

Just as its predecessor, WLM 2009 leaves traces behind of its users. Changing to the ESE database technology has lead to a different way of storing contact information. Information that was encrypted in WLM 8.0 is now stored in clear text. WLM 2009 offers more contact/profile options compared to WLM 8.0. It seems all of the contact information is stored in the ESE database.

By using and creating our own tools it was possible to dump the WLM ESE database and reveal information about the contents of the database. This gave insight of what tables and rows are used and how tables are related to each other. The methods and scripts presented might be valuable for other ESE related examinations.

Although valuable information is documented, it would be useful to explore and document more columns. The methods and tools are available for use.

The information provided is this article can be used to create a tool that is able to reconstruct the contact list from a WLM ESE database.

6 Bibliography

Dongen, W. S. (2007). Forensic artefacts left by Windows Live Messenger 8.0 . *Digital Investigation* , 73-87.

Wikipedia.org (May, 2009). Extensible Storage Engine. Available From: <http://en.wikipedia.org/wiki/Extensible_Storage_Engine>.

Microsoft (May, 2009). Extensible Storage Architecture <http://technet.microsoft.com/en-us/library/aa998171(EXCHG.65).aspx>.

Jim McBee (2006). Microsoft Exchange Server 2003 Advanced Administration: In the Field Results, ISBN-13: 978-0470038512, page 131