Windows XML Event Log (EVTX)

Analysis of EVTX

By Joachim Metz <joachim.metz@gmail.com>

Summary

The Windows XML EventLog (EVTX) format is used by Microsoft Windows to store system log information. This specification is based the work done by A. Schuster [SCHUSTER11] and on [MS-EVEN6]. It was complemented by other public available information and reverse engineering of the file format.

This document is intended as a working document for the Windows XML EventLog (EVXT) specification. Which should allow existing Open Source forensic tooling to be able to process this file type.

Special thanks to A. Schuster for his excellent work on the format and test files.

Document information

Author(s): Joachim Metz <joachim.metz@gmail.com>

Abstract: This document contains information about the Windows XML Event Viewer Log (EVTX) format.

Classification: Public

Keywords: Windows XML Event Viewer Log, EVTX

License

Version

Version	Author	Date	Comments
0.0.1	J.B. Metz	September 2011	Initial version.
0.0.2	J.B. Metz	March 2012 April 2012	Additional information.
0.0.3	J.B. Metz	May 2012	Additional information.
0.0.4	J.B. Metz	May 2012	Updates for Windows 8 Consumer Preview.
0.0.5	J.B. Metz	October 2012	Additional information regarding formatted messages.
0.0.6	J.B. Metz	December 2012	Additional information regarding formatted messages.
0.0.7	J.B. Metz	February 2013	Additional information regarding formatted messages.
0.0.8	J.B. Metz	February 2013	Additional information regarding chunk offset values seen in archived EVTX files with thanks to R. Rumble.
0.0.8	J.B. Metz	February 2013	Additional information regarding corruption scenarios.
0.0.9	J.B. Metz	March 2013	Additional information regarding corruption scenarios.
0.0.10	J.B. Metz	May 2013	Additional information regarding corruption scenarios.
0.0.11	J.B. Metz	July 2013	Additional information regarding XML escaping with thanks to G. Torres.
0.0.12	J.B. Metz	July 2013	Additional information regarding ProcessingErrorData.
0.0.13	J.B. Metz	July 2013	Additional information regarding dirty file with invalid number of chunks corruption scenario with thanks to G. Torres.

Table of Contents

1. Overview

The Windows XML EventLog (EVTX) format is used by Microsoft Windows, as of Windows Vista, to store system log information.

The EVTX format supersedes the Windows EventLog (EVT) format as used in Windows XP.

File consists of:

- file header
- chunks
- trailing empty values

Characteristics	Description
Byte order	little-endian
Date and time values	Filetime in UTC
Character string	ASCII strings are stored in extended ASCII with a codepage. Unicode strings are stored in UTF-16 little-endian without the byte order mark (BOM).

1.1. Test version

The following version of programs were used to test the information within this document:

- Windows Vista
- Windows 2008
- Windows 7
- Windows 8 (Consumer Preview)

1.2. Event Log files

The event logs files can normally be found in:

```
C:\Windows\System32\winevt\Logs\
```

Filename	Description
Application.evtx	Application events
DFS Replication.evtx	TODO
HardwareEvents.evtx	TODO
Internet Explorer.evtx	Internet Explorer events
Key Management Service.evtx	TODO
Media Center.evtx	TODO
Microsoft-Windows-Bits-Client %4Operational.evtx	TODO
Microsoft-Windows-CodeIntegrity %4Operational.evtx	TODO
Microsoft-Windows-	TODO

Filename	Description
CorruptedFileRecovery-Client %4Operational.evtx	
Microsoft-Windows-CorruptedFileRecovery-Server %4Operational.evtx	TODO
Microsoft-Windows-DateTimeControlPanel %4Operational.evtx	TODO
Microsoft-Windows-Diagnosis-DPS %4Operational.evtx	TODO
Microsoft-Windows-Diagnosis-PLA %4Operational.evtx	TODO
Microsoft-Windows-Diagnostics-Networking%4Operational.evtx	TODO
Microsoft-Windows-Diagnostics-Performance%4Operational.evtx	TODO
Microsoft-Windows-DiskDiagnostic %4Operational.evtx	TODO
Microsoft-Windows-DiskDiagnosticDataCollector %4Operational.evtx	TODO
Microsoft-Windows-DiskDiagnosticResolver %4Operational.evtx	TODO
Microsoft-Windows-DriverFrameworks-UserMode %4Operational.evtx	TODO
Microsoft-Windows-Forwarding %4Operational.evtx	TODO
Microsoft-Windows-GroupPolicy %4Operational.evtx	TODO
Microsoft-Windows-Help %4Operational.evtx	TODO
Microsoft-Windows-International %4Operational.evtx	TODO
Microsoft-Windows-Kernel-WDI %4Operational.evtx	TODO
Microsoft-Windows-Kernel-WHEA.evtx	TODO
Microsoft-Windows-LanguagePackSetup %4Operational.evtx	TODO

Filename	Description
Microsoft-Windows-MUI %4Operational.evtx	TODO
Microsoft-Windows-NetworkAccessProtection %4Operational.evtx	TODO
Microsoft-Windows-Program-Compatibility-Assistant %4Operational.evtx	TODO
Microsoft-Windows-ReadyBoost %4Operational.evtx	TODO
Microsoft-Windows-ReliabilityAnalysisComponent %4Metrics.evtx	TODO
Microsoft-Windows-ReliabilityAnalysisComponent %4Operational.evtx	TODO
Microsoft-Windows-Resource-Exhaustion-Detector %4Operational.evtx	TODO
Microsoft-Windows-Resource-Exhaustion-Resolver %4Operational.evtx	TODO
Microsoft-Windows-Resource-Leak-Diagnostic%4Operational.evtx	TODO
Microsoft-Windows-RestartManager%4Operational.evtx	TODO
Microsoft-Windows-TaskScheduler %4Operational.evtx	TODO
Microsoft-Windows-TerminalServices-RDPClient %4Operational.evtx	TODO
Microsoft-Windows-UAC %4Operational.evtx	TODO
Microsoft-Windows-UAC-FileVirtualization %4Operational.evtx	TODO
Microsoft-Windows-WindowsUpdateClient %4Operational.evtx	TODO
Microsoft-Windows-Winlogon %4Operational.evtx	TODO
Microsoft-Windows-Wired-AutoConfig%4Operational.evtx	TODO

Filename	Description
Microsoft-Windows-WLAN-AutoConfig%4Operational.evtx	TODO
ODiag.evtx	TODO
OSession.evtx	Office sessions events
Security.evtx	Security events
Setup.evtx	Setup events
System.evtx	System events

2. File header

The file header is 4096 bytes of size and consists of:

offset	size	value	description
0	8	“ElfFile\x00”	Signature
8	8		First chunk number
16	8		Last chunk number
24	8		Next record identifier
32	4	128	Header size
36	2	1	Minor version
38	2	3	Major version
40	2	4096	Header block size (or chunk data offset)
42	2		Number of chunks
44	76		Empty values
120	4		File flags See section: 2.1 File flags
124	4		Checksum CRC32 of the first 120 bytes of the file header
128	3968		Empty values

The CRC-32 is describe in RFC 1952 and uses an initial value of 0.

File size = (Number of chunks * 65536) + 4096 ?

2.1. File flags

Value	Identifier	Description
0x0001		Is dirty
0x0002		Is full

3. Chunk

The chunk is 65536 bytes of size and consists of:

- chunk header
- array of event records
- unused space

3.1. Chunk header

The chunk header is 512 bytes of size and consists of:

offset	size	value	description
0	8	“ElfChnk\x00”	Signature
8	8		First event record number
16	8		Last event record number
24	8		First event record identifier
32	8		Last event record identifier
40	4	128	Header size (or offset to pointer data)
44	4		Last event record data offset Offset to the data of the last event record. The offset is relative to the start of the chunk header.
48	4		Free space offset Offset to free space in the chunk. The offset is relative to the start of the chunk header.
52	4		Event records checksum CRC32 of the events records data
56	64		Empty values
120	4		Unknown (flags?)
124	4		Checksum CRC32 of the first 120 bytes and bytes 128 to 512 of the chunk.

The CRC-32 is describe in RFC 1952 with an uses an initial value of 0.

The free space offset is not the end of event records data offset, is sometimes point to the end of the chunk, where the chunk after the last event record was filled with 0-byte values. This behavior was seen in archived EVTX files.

offset	size	value	description
128	64 x 4 =256		Common string offset array The offsets are relative from the start of

offset	size	value	description
			the chunk
384	32 x 4 = 128		TemplatePtr Array of 32 x 32-bit values

The common string offset array contains the offsets of strings that are common in the event records stored in the chunk so that they only have to be stored once in the first event record and can be referenced from successive event records.

Identifier/Number of first and last event record in chunk

Data after header and before event record?

3.2. Event record

The event record is variable of size and consists of:

offset	size	value	description
0	4	“\x2a\x2a\x00\x00”	Signature
4	4		Size The size of the event record including the signature and the size
8	8		Event record identifier
16	8		Written date and time Contains a Filetime The date and time the event record was written (logged)
24	…		Event Contains binary XML See section: 4 Binary XML
...	4		Copy of size

4. Binary XML

4.1. Document structure

According [MS-EVEN6] the binary XML structure should consist of:

The document (BinXMLDocument) consists of:

- Prologue (BinXMLPI) (zero or one)
- Fragment (zero or more)
- Miscellaneous (BinXMLPI) (zero or one)
- End of file token

4.1.1. Fragment

The fragment (BinXMLFragment) consists of:

- fragment header
- an element or a template instance

TODO: is it valid for a fragment with more then one element?

4.1.2. Fragment header

The fragment header (BinXMLFragmentHeader) is 4 byte of size and consists of:

offset	size	value	description
0	1	0x0f	Fragment header token Should be: BinXmlFragmentHeaderToken See section: 4.2 Token types
1	1	0x01	Major version
2	1	0x01	Minor version
3	1	0x00	Flags

4.1.3. Element

An element (BinXMLElement) can either be 'empty' or a 'filled'.

BinXMLEmpyElement:
- element start
- close empty element token

Example of an 'empty' element in textual XML:

```
<Provider Name="Provider"/>
```

BinXMLFilledElement:
- element start
- close start element token
- content
- end element token

Example of a 'filled' element in textual XML:

```
<EventID>400</EventID>
```

4.1.4. Element start

The element start (BinXMLElementStart) is variable of size and consists of:

offset	size	value	description
0	1	0x01 0x41	Open start element tag token Should be: BinXmlTokenOpenStartElementTag See section: 4.2 Token types
1	2		Dependency identifier

offset	size	value	description
			-1 (0xffff) => not set
3	4		Data size The size of the data. This includes the size of the element name, attribute list, close element tag, content and end element tag, except for the first 7 bytes of the element start.
7	4		Element name offset The offset is relative from the start of the chunk See section: 4.1.7 Name
11	...		Attribute list See section: 4.1.5 Attribute list

A token type of 0x01 indicates that the element start tag contains no elements; a token type of 0x41 indicates that an attribute list can be expected in the element start tag.

Note that the element name can be stored before the attribute list.

The name offset is not used in the binary XML in the Windows Event Template resource.

4.1.5. Attribute list

The attribute (BinXmlAttributeList) is variable of size and consists of:

offset	size	value	description
0	4		Data size Does not include the 4 byte of the size.
4	...		Array of attributes See section: 4.1.6 Attribute

TODO: if attribute list is empty it is trailed by 2 bytes? Is this 32-bit alignment padding?

4.1.6. Attribute

The attribute (BinXmlAttribute) is variable of size and consists of:

offset	size	value	description
0	1	0x06 0x46	Attribute token Should be: BinXmlTokenAttribute See section: 4.2 Token types
1	4		Attribute name offset The offset is relative from the start of the chunk See section: 4.1.7 Name
5	...		Attribute data

A token type of 0x46 indicates that there is another attribute in the attribute list; a token type of 0x06 indicates that no more attributes exist.

Note that the attribute name can be stored before the attribute list.

The attribute data (BinXMLAttributeData) can be:
- value text
- substitution
- character entity reference
- entity reference

The name offset is not used in the binary XML in the Windows Event Template resource.

4.1.7. Name

The name (BinXmlName) is variable of size and consists of:

offset	size	value	description
0	4		Unknown
4	2		Name hash Which hash algorithm?
6	2		Number of characters
8	...		UTF-16 little-endian string with an end-of-string character

The unknown 4 bytes are not present in the binary XML in the Windows Event Template resource.

4.1.8. Content

The content (BinXMLContent) can be:
- an element
- content string data
- character entity reference
- entity reference
- CDATA section
- PI

4.1.9. Content string

The content string data (BinXMLContentStringData) can be:
- value text
- substitution

TODO: a content string containing an end-of-line character seems to be considered empty by Event Viewer

4.1.10. Value text

The value text (BinXmlValueText) is variable of size and consists of:

offset	size	value	description
0	1	0x05 0x45	Value token Should be: BinXmlTokenValue See section: 4.2 Token types
1	1	0x01	Value type Should be: StringType See section: 4.3 Value types
2	...		Value data See section: 4.1.20 Unicode text string

A token type of 0x45 indicates that more data can be expected to follow in the current content of the element or attribute; a token type of 0x05 indicates that no more such data follows.

A value text can be stored spanning multiple value tokens.

4.1.11. Substitution

The substitution (BinXmSubstitution) can be:

- normal substitution
- optional substitution

4.1.12. Normal substitution

The normal substitution (BinXmNormalSubstitution) is 4 byte of size and consists of:

offset	size	value	description
0	1	0x0d	Normal substitution token Should be: BinXmlTokenNormalSubstitution See section: 4.2 Token types
1	2		Substitution identifier Identifier of the value in the template instance data, where 0 represents the first value
3	1		Value type See section: 4.3 Value types

If the value type is an array type (0x80) the substitution is repeated for every element of the array. If the size of an array type is 0 then a single empty element should be created.

If the value type is Size (0x10) the corresponding substitution value should be a 32-bit hexadecimal integer (0x14) or 64-bit hexadecimal integer (0x15). The same applies to an array of Size (0x90) where the substitution value should be an array of 32-bit hexadecimal integer (0x94) or an array of 64-bit hexadecimal integer (0x95).

If the value type is the Binary XML type (0x21) the value data should be either a fragment or a template instance.

4.1.13. Optional substitution

The optional substitution (BinXmlOptionalSubstitution) is 4 byte of size and consists of:

offset	size	value	description
0	1	0x0e	Optional substitution token Should be: BinXmlTokenOptionallSubstitution See section: 4.2 Token types
1	2		Substitution identifier Identifier of the value in the template instance data, where 0 represents the first value
3	1		Value type See section: 4.3 Value types

If the value type of the corresponding template value is NULL (0x00) the element should be ignored and not created.

If the value type is an array type (0x80) the substitution is repeated for every element of the array. If the size of an array type is 0 then a single empty element should be created.

If the value type is Size (0x10) the corresponding substitution value should be a 32-bit hexadecimal integer (0x14) or 64-bit hexadecimal integer (0x15). The same applies to an array of Size (0x90) where the substitution value should be an array of 32-bit hexadecimal integer (0x94) or an array of 64-bit hexadecimal integer (0x95).

If the value type is the Binary XML type (0x21) the value data should be either a fragment or a template instance.

4.1.14. Character entity reference

The character entity reference (BinXmlCharacterEntityReference) is 3 byte of size and consists of:

offset	size	value	description
0	1	0x08 0x48	Character entity reference token Should be: BinXmlTokenCharRef See section: 4.2 Token types
1	2		Character entity value

A token type of 0x48 indicates that more data can be expected to follow in the current content of the element or attribute; a token type of 0x08 indicates that no more such data follows.

In the resulting XML the character entity is replaced e.g. “38” becomes “&”.

According to [MS-EVEN6] emit the characters '&' and '#' and the decimal string representation of the value. TODO create a test file.

4.1.15. Entity reference

The entity reference (BinXmlEntityReference) is 5 bytes of size and consists of:

offset	size	value	description
0	1	0x09 0x49	Entity reference token Should be: BinXmlTokenEntityRef See section: 4.2 Token types
1	4		Entity name offset The offset is relative from the start of the chunk See section: 4.1.7 Name

A token type of 0x49 indicates that more data can be expected to follow in the current content of the element or attribute; a token type of 0x09 indicates that no more such data follows.

In the resulting string the entity is replaced e.g. “amp” becomes & for a Unicode string and “&” for an XML string.

The name offset is not used in the binary XML in the Windows Event Template resource.

It currently is assumed that the following entity references are supported lt, gt, amp, quot and apos.

4.1.16. CDATA section

The entity reference (BinXmlEntityReference) is variable of size and consists of:

offset	size	value	description
0	1	0x07 0x47	CDATA section token Should be: BinXmlTokenCDATASection See section: 4.2 Token types
1	...		CDATA text See section: 4.1.20 Unicode text string

A token type of 0x47 indicates that more data can be expected to follow in the current content of the element or attribute; a token type of 0x07 indicates that no more such data follows.

4.1.17. Template instance

The template instance (BinXmlTemplateInstance) is variable of size and consists of:

offset	size	value	description
0	1	0x0c	Template instance token Should be: BinXmlTokenTemplateInstance See section: 4.2 Token types
1	...		Template definition
...	...		Template instance data

4.1.18. Template definition

The template definition (BinXmlTemplateDefinition) is variable of size and consists of:

offset	size	value	description
0	1	0x01	Unknown(Version? Or number of template defs?)
1	4		Unknown (Template identifier?)
5	4		Template definition data offset
Template definition data			
9	4		Unknown (Next template definition offset) 0 if not used
13	16		Template identifier Contains a GUID
29	4		Data size The size of the data. This includes the size of the fragment header, element and end of file token, except for the first 33 bytes of the template definition.
33	...		Fragment header
...	...		Element
...	1		End of file token Should be: BinXmlTokenEOF See section: 4.2 Token types

Note that the template definition data offset either point to the offset directly after this value or somewhere previously in the chunk. The template definition can therefore be stored non-continuous.

What does the %b0 in [MS-EVEN6] signify?

4.1.19. Template instance data

The template instance data (BinXmlTemplateInstanceData) is variable of size and consists of:

offset	size	value	description
0	4		Number of template values
4	...		Array of template value descriptors
			Array of template value data

The template value descriptor is 4 bytes of size and consists of:

offset	size	value	description
0	2		Value size
2	1		Value type
1	1	0x00	Unknown (Empty value)

4.1.20. Unicode text string

The Unicode text string is variable of size and consists of:

offset	size	value	description
2	2		Number of characters
4	...		UTF-16 little-endian string without an end-of-string character

4.1.21. PI

The PI consists of:

- PI target
- PI data

4.1.22. PI target

The PI target (BinXmlPITarget) is 5 bytes of size and consists of:

offset	size	value	description
0	1	0x0a	PI target reference token Should be: BinXmlTokenPITarget See section: 4.2 Token types
1	4		PI target name offset The offset is relative from the start of the chunk See section: 4.1.7 Name

The name offset is not used in the binary XML in the Windows Event Template resource.

4.1.23. PI data

The entity reference (BinXmlPIData) is variable of size and consists of:

offset	size	value	description
0	1	0x0b	PI data token Should be: BinXmlTokenCDATASection See section: 4.2 Token types
1	...		PI data text See section: 4.1.20 Unicode text string

4.2. Token types

Binary XML defines multiple token types.

Value	Identifier	Description
0x00	BinXmlTokenEOF	End of file

Value	Identifier	Description
0x01 0x41	BinXmlTokenOpenStartElementTag	Open start element tag Indicates the start of a start element, correlates to '<' in '<Event>'
0x02	BinXmlTokenCloseStartElementTag	Close start element tag Indicates the end of a start element, correlates to '>' in '<Event>'
0x03	BinXmlTokenCloseEmptyElementTag	Close empty element tag Indicates the end of a start element, correlates to '/>' in '<Event/>'
0x04	BinXmlTokenEndElementTag	Close end element tag Indicates the end of element, correlates to '</Event>'
0x05 0x45	BinXmlTokenValue	Value
0x06 0x46	BinXmlTokenAttribute	Attribute
0x07 0x47	BinXmlTokenCDATASection	CDATA section
0x08 0x48	BinXmlTokenCharRef	Character entity reference
0x09 0x49	BinXmlTokenEntityRef	Entity reference
0x0a	BinXmlTokenPITarget	Processing instructions (PI) target XML processing instructions
0x0b	BinXmlTokenPIData	Processing instructions (PI) data XML processing instructions
0x0c	BinXmlTokenTemplateInstance	Template instance
0x0d	BinXmlTokenNormalSubstitution	Normal substitution
0x0e	BinXmlTokenOptionalSubstitution	Optional substitution
0x0f	BinXmlFragmentHeaderToken	Fragment header token

Some of the token types can contain the has more data flag 0x40.

TODO bitmask of 0x1f ? is this defined in winevt.h ? If so what do the other flags signify?

4.3. Value types

Value	Identifier	Description
0x00	NullType	NULL or empty

Value	Identifier	Description
0x01	StringType	Unicode string Stored as UTF-16 little-endian without an end-of-string character
0x02	AnsiStringType	ASCII string Stored using a codepage without an end-of-string character
0x03	Int8Type	8-bit integer signed
0x04	UInt8Type	8-bit integer unsigned
0x05	Int16Type	16-bit integer signed
0x06	UInt16Type	16-bit integer unsigned
0x07	Int32Type	32-bit integer signed
0x08	UInt32Type	32-bit integer unsigned
0x09	Int64Type	64-bit integer signed
0x0a	UInt64Type	64-bit integer unsigned
0x0b	Real32Type	Floating point 32-bit (single precision)
0x0c	Real64Type	Floating point 64-bit (double precision)
0x0d	BoolType	Boolean An 32-bit integer that MUST be 0x00 or 0x01 (mapping to true or false, respectively).
0x0e	BinaryType	Binary data
0x0f	GuidType	GUID Stored in little-endian
0x10	SizeTType	Size type Either 32 or 64-bits. This value type should be pair up with a HexInt32Type or HexInt64Type
0x11	FileTimeType	Filetime (64-bit) Stored in little-endian
0x12	SysTimeType	System time (128-bit) Stored in little-endian
0x13	SidType	NT Security Identifier (SID) See [NTSID]
0x14	HexInt32Type	32-bit integer hexadecimal 32-bit (unsigned) integer that should be represented in hexadecimal notation
0x15	HexInt64Type	64-bit integer hexadecimal 64-bit (unsigned) integer that should be represented in hexadecimal notation
0x20	EvtHandle	
0x21	BinXmlType	Binary XML fragment

Value	Identifier	Description
0x23	EvtXml	

If the MSB of the value type (0x80) is use to indicate an array type. According to [MSDN] binary data and binary XML fragment types are not supported. For the string types the end-of-string character is used as a separator.

Value	Identifier	Description
0x81		Array of Unicode strings Individual strings are stored as UTF-16 little-endian with an end-of-string character
0x82		Array of ASCII strings Individual strings are stored as ASCII string using a codepage with an end-of-string character
0x83		Array of 8-bit integer signed Every 1 byte is an individual value
0x84		Array of 8-bit integer unsigned Every 1 byte is an individual value
0x85		Array of 16-bit integer signed Every 2 bytes are an individual value in little-endian
0x86		Array of 16-bit integer unsigned Every 2 bytes are an individual value in little-endian
0x87		Array of 32-bit integer signed Every 4 bytes are an individual value in little-endian
0x88		Array of 32-bit integer unsigned Every 4 bytes are an individual value in little-endian
0x89		Array of 64-bit integer signed Every 8 bytes are an individual value in little-endian
0x8a		Array of 64-bit integer unsigned Every 8 bytes are an individual value in little-endian
0x8b		Array of Floating point 32-bit (single precision) Every 4 bytes are an individual value in little-endian
0x8c		Array of Floating point 64-bit (double precision) Every 8 bytes are an individual value in little-endian
0x8d		Array of boolean Every 4 bytes are an individual value in little-endian

Value	Identifier	Description
0x8f		Array of GUID Every 16 bytes are an individual value in little-endian
0x90		Array of size type An individual value is either 32 or 64-bits. This value type should be pair up with an array of HexInt32Type or HexInt64Type
0x91		Array of Filetime Every 8 bytes are an individual value in little-endian
0x92		Array of system time Every 16 bytes are an individual value in little-endian
0x93		Array of NT Security Identifiers (SID)
0x94		Array of 32-bit integer hexadecimal Every 4 bytes are an individual value in little-endian
0x95		Array of 64-bit integer hexadecimal Every 8 bytes are an individual value in little-endian

4.3.1. String

If in a string the characters: <, >, &, " and ' are not escaped they must respectively be replaced by the following character entities: <, >, &, " and '. This does not apply to Character entity reference and Entity reference encoded strings.

Event Viewer will not escape the character entities in the XML view, but will when exported as XML. Event Viewer seems to apply the XML character entity escaping inside element values for &, < and > but not for ' and ".

4.3.2. Systemtime

The systemtime is 16 bytes of size and consists of:

offset	size	value	description
0	2		Year
2	2		Month
4	2		Day of week
6	2		Day of month
8	2		Hours
10	2		Minutes

offset	size	value	description
12	2		Seconds
14	2		Milliseconds

4.3.3. Floating point

Floating point values are represented as the following stings.

Value	Identifier	Description
-1.#INF		Negative infinity/overflow
1.#INF		Positive infinity/overflow
-1.#IND		Indeterminate
[-]?0		Positive or negative zero
[-]?[0-9]+		Any positive or negative value that can be represented as an integer
[-]?[0-9]+.[0-9]{6}		Any positive or negative value that can be represented in 6 fractional digits
[-]?[0-9]+.[0-9]{6}e-[0-9]{3}		Any positive or negative value that could not be represented in 6 fractional digits

TODO validate the highlighted ones; 32-bit fractional of 6, 64-bit fractional of 14

5. Event

5.1. Event identifier

The event identifier is 4 bytes of size and consist of:

offset	size	value	description
0.0	16 bits		Code
2.0	12 bits		Facility
3.4	1 bit		Reserved
3.5	1 bit		Customer flags 0 => System code 1 => Customer code
3.6	2 bits		Severity 00 => Success 01 => Informational 10 => Warning 11 => Error

5.2. Level

Value	Identifier	Description
0x00000000		Identifies an event that should always be logged (win:LogAlways) Shown as “Information” in Event Viewer
0x00000001	WINEVENT_LEVEL_CRITICAL	Identifies an abnormal exit or termination event (win:Critical)
0x00000002	WINEVENT_LEVEL_ERROR	Identifies a severe error event (win:Error)
0x00000003	WINEVENT_LEVEL_WARNING	Identifies a warning event such as an allocation failure (win:Warning)
0x00000004	WINEVENT_LEVEL_INFO	Identifies a non-error event such as an entry or exit event (win:Informational)
0x00000005	WINEVENT_LEVEL_VERBOSE	Identifies a detailed trace event (win:Verbose)
0x00000006		Reserved (win:ReservedLevel6)
0x00000007		Reserved (win:ReservedLevel7)
0x00000008		Reserved (win:ReservedLevel8)
0x00000009		Reserved (win:ReservedLevel9)
0x0000000a		Reserved (win:ReservedLevel10)
0x0000000b		Reserved (win:ReservedLevel11)
0x0000000c		Reserved (win:ReservedLevel12)
0x0000000d		Reserved (win:ReservedLevel13)
0x0000000e		Reserved (win:ReservedLevel14)
0x0000000f		Reserved (win:ReservedLevel15)

5.3. Keywords

Value	Identifier	Description
0x0000000000000000		win:AnyKeyword

Value	Identifier	Description
0x0000000000010000		Shell
0x0000000000020000		Properties
0x0000000000040000		FileClassStoreAndIconCache
0x0000000000080000		Controls
0x0000000000100000		APICalls
0x0000000000200000		InternetExplorer
0x0000000000400000		ShutdownUX
0x0000000000800000		CopyEngine
0x0000000001000000		Tasks
0x0000000002000000		WDI
0x0000000004000000		StartupPerf
0x0000000008000000		StructuredQuery
0x0001000000000000		win:Reserved
0x0002000000000000		win:WDIContext
0x0004000000000000		win:WDIDiag
0x0008000000000000		win:SQM
0x0010000000000000		win:AuditFailure
0x0020000000000000		win:AuditSuccess
0x0040000000000000		win:CorrelationHint
0x0080000000000000		Classic win:EventlogClassic
0x0100000000000000		win:ReservedKeyword56
0x0200000000000000		win:ReservedKeyword57
0x0400000000000000		win:ReservedKeyword58
0x0800000000000000		win:ReservedKeyword59
0x1000000000000000		win:ReservedKeyword60
0x2000000000000000		win:ReservedKeyword61
0x4000000000000000		win:ReservedKeyword62
0x8000000000000000		win:ReservedKeyword63 Microsoft-Windows-Shell-Core/Diagnostic

5.4. Externally stored values

Some of the data that Event Viewer shows is stored outside the event log files.

On Windows XP (and earlier) the first step to determine the location of these values is find the corresponding “event log type sub key” in the Windows registry under:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Event Log\
```

Every event log type has its own sub key, e.g.:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Event Log\System
```

Common event log types are:

- Application
- Security
- System

NOTE: the event log type is also stored in the “Channel” event XML element.

The event log type sub key has a “event source sub key” for every source name, e.g for the source name “Workstation”:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\EventLog\System\Workstation
```

Note that the source name is case insensitive; so “Workstation” and “workstation” are considered equivalent.

The source name is stored as an attribute of the “Provider” element within the Event XML, e.g.

```
<Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
  <System>
    <Provider Name="Microsoft-Windows-Search"
              Guid="{CA4E628D-8567-4896-AB6B-835B221F373F}"
              EventSourceName="Windows Search Service"/>
```

The “EventSourceName” attribute contains the source name. If there is no “EventSourceName” attribute the “Name” attribute is used.

As of Windows Vista the event log type sub key contains the value “ProviderGuid” which should contain the same GUID as indicated in the Event XML:

```
{CA4E628D-8567-4896-AB6B-835B221F373F}
```

The corresponding provider settings can be found in the event message provider registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\Publishers\{ca4e628d-8567-4896-ab6b-835b221f373f}
```

On a Windows Vista (or later) system “wevtutil” can be used to determine more about the provider. E.g.

```
wevtutil gp Microsoft-Windows-Search
```

5.4.1. Message strings

The event message strings are stored in event message files.

The event message provider registry key has a value named “EventMessageFile” which contains a

path specification of the event message file, e.g.

```
%SystemRoot%\System32\netmsg.dll
```

Note that the value can contain multiple filenames separated by a semi colon (;) character and that the name of the event message files is case insensitive.

On Windows XP (and earlier) the event source sub key had a value named “EventMessageFile” which contains the same path. As of Windows Vista this value is not always present and using the value “MessageFileName“ in the event message provider registry key seems to be the preferred method. However it is possible that the event message provider registry key is not present and the event source sub key is needs to be used instead.

Here “%SystemRoot%” is case insensitive and needs to be expanded to the Windows directory which is depended on the Windows version:

Value	**Version**
\WINNT35	Windows NT 3.5x
\WINNT	Windows NT 3.1, Windows NT 4.0 and Windows 2000 (NT 5.0)
\WINDOWS	Windows XP (NT 5.1) and later

The actual value of %SystemRoot% can be found in the Registry value:

```
Key:      HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\
Value:    SystemRoot
```

Other placeholders that found to be used are:

```
%WinDir%
```

The actual value of e.g. %WinDir% can be found in the Registry value:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session
Manager\Environment\windir
```

Event message files are PE/COFF executables that contains a resource (“.rsrc ”) section. Event message files can have various extensions, e.g. “.exe”, “.dll”, “.dll.mui”, “.sys”.

There different types of event message files:

- Message-table resource
- Multilingual User Interface (MUI) resource

Note that event message files can have any combination of these resources. The rules of preference seems to be:

- use message-table resource if present, before checking MUI resource

5.4.1.1. Event resource file

The event message provider registry key has a value named “ResourceFileName”. It is currently assumed that this Registry value contains a path specification of the event resource file, e.g.

```
%SystemRoot%\System32\tquery.dll
```

The event resource file should contain a Windows Event Template (WEVT_TEMPLATE) resource. The MUI resource should also contain a main name type “WEVT_TEMPLATE”.

The information stored in this resource is used to:

- determine the message string identifier
- determine the string identifiers of channels, keywords, levels, opcodes and tasks
- parse Event XML “UserData”

For more detailed information see: [LIBEXE].

5.4.1.2. Message string identifier

On Windows XP (and earlier) the message string identifier was a direct mapping of the event identifier as of Windows Vista this is no longer the case. There seem to be multiple methods how the event identifier is mapped to the message string identifier, namely:

- Using the event identifier qualifiers
- Using the Windows Event Template resource

5.4.1.2.1. Using the event identifier qualifiers

If the EventID element in Event XML has the Qualifiers attribute set, e.g.:

```
<EventID Qualifiers="16384">7036</EventID>
```

Then the message string identifier can be determined as following:

```
16384 in hexadecimal is 0x4000
 7036 in hexadecimal is 0x1b7c
```

```
message string identifier = ( 0x4000 << 16 ) | 0x1b7c = 0x40001b7c
```

5.4.1.2.2. Using the Windows Event Template (WEVT_TEMPLATE) resource

If an event resource file has been specified and if the Provider element in the Event XML has the GUID attribute set, e.g.:

```
<Provider Name="Microsoft-Windows-UAC"
          Guid="{E7558269-3FA5-46ED-9F4D-3C6E282DDE55}"/>
<EventID>1</EventID>
```

This GUID can be used to find a corresponding provider in the Windows Event Template (WEVT_TEMPLATE) resource. This resource should contain an event definition with the same identifier as the EventID in the Event XML, e.g. in case of the example 1. The event definition will also contain a reference the the message identifier, e.g. in case of the example 0xb9000001.

5.4.1.3. Message-table resource event message files

In a message-table resource event message file the event message strings are stored in the message-table resource of the event message file.

The resource section of a message-table resource event message file contains a message-table resource which contains the event message strings. E.g. on Windows Vista

```
C:\Windows\Microsoft.NET\Framework\v2.0.50727\EventLogMessages.dll
```

The event message strings have identifiers similar to the event identifiers. E.g. if the event identifier is 0 and the message string identifier 0, the corresponding event message string would be:

```
%1
```

The placeholder values %1 represent the first string in the event.

The event strings are stored as “Data” elements in the “EventData” element within the Event XML, e.g.

```
  <EventData>
    <Data>Service has been successfully shut down.</Data>
  </EventData>
```

For a more comprehensive description of how to extract the event strings from the Event XML see section: 5.4.1.5 Event data. Sometimes the message string can have more placeholder than the event data contains strings, it seems in such a case the placeholders are not replaced and shown as %# in the resulting string.

Note that the event message strings are language specific. An event message file can therefore contain event message strings for multiple languages.

5.4.1.4. Multilingual User Interface (MUI) event message files

The resource section of a Multilingual User Interface (MUI) event message file contains Multilingual User Interface (MUI) resource. E.g. on Windows Vista

```
C:\Windows\System32\services.exe
```

The MUI event message files do not have to contain a message-table resource but forward to a a language specific message-table resource event message file, e.g. “en-US”:

```
C:\Windows\System32\en-US\services.exe.mui
```

It is this file that contains the language specific event message-table resource.

The event message strings have identifiers similar to the event identifiers. E.g. if the event identifier in XML is:

```
<EventID Qualifiers="16384">7036</EventID>
```

This would correspond to the event message string identifier:

```
16384 in hexadecimal is 0x4000
 7036 in hexadecimal is 0x1b7c
```

```
event message string identifier  = ( 0x4000 << 16 ) | 0x1b7c = 0x40001b7c
```

The corresponding event message string would be:

```
The %1 service entered the %2 state.
```

The placeholder values %1 and %2 represent the first and second string in the event.

The event strings are stored as “Data” elements in the “EventData” element within the Event XML, e.g.

```
  <EventData>
    <Data Name="param1">Volume Shadow Copy</Data>
    <Data Name="param2">stopped</Data>
  </EventData>
```

5.4.1.5. Event data

As previously mentioned the event strings (and binary data) are stored as “Data” elements in the “EventData” element within the Event XML. Another way to store the event data is in a “UserData” element. The information in this section is partially deduced on the behavior of the “General”, “Details Friendly View” and “Details XML View” of Event Viewer.

Let's start out with the following variant of event data.

```
  <EventData>
    <Data>SessionEnv</Data>
    <Binary>D9060000</Binary>
  </EventData>
```

In this case “EventData” in the “Details Friendly View” contains both the value of the “Data” and the “Binary” tag. The value of the Binary tag is additionally interpreted as “Binary Data”, which is base16 encoded.

```
  SessionEnv
  D9060000
```

If the Data has a corresponding “Name” attribute the “EventData” in the “Details Friendly View” shows the value of the “Name” attribute followed by the value of the “Data” tag, e.g.

```
  <EventData>
    <Data Name="param1">86400</Data>
    <Data Name="param2">SuppressDuplicateDuration</Data>
    <Data Name="param3">Software\Microsoft\EventSystem\EventLog</Data>
  </EventData>
```

```
  param1 86400
  param2 SuppressDuplicateDuration
  param3 Software\Microsoft\EventSystem\EventLog
```

The data of an empty “Data” is not ignored but not directly visible int the “Details Friendly View”. In case of the following example on the value of the “Name” attribute would be shown.

```
  <EventData>
    <Data Name="ExtraInfo"/>
  </EventData>
```

```
  ExtraInfo
```

ProcessingErrorData is a variation of EventData:

```
  <ProcessingErrorData>
    <ErrorCode>15005</ErrorCode>
    <DataItemName>Value</DataItemName>
    <EventPayload>804110C3E253BF01</EventPayload>
  </ProcessingErrorData>
```

```
    ErrorCode 15005
    DataItemName Value
    EventPayload 804110C3E253BF01
```

In some events the data is not stored in a “EventData” tag within the Event XML but in a “UserData” tag, e.g.

```
  <UserData>
    <EventXML xmlns:auto-ns2="..." xmlns="LoadPerf">
      <param1>WmiApRpl</param1>
      <param2>WmiApRpl</param2>
      <binaryDataSize>4</binaryDataSize>
      <binaryData>44415441</binaryData>
    </EventXML>
  </UserData>
```

In this case the “EventData” in the “Details Friendly View” will show the data as:

```
EventXML
  param1 WmiApRpl
  param2 WmiApRpl
  binaryDataSize 4
  binaryData 44415441
```

The binary data is not interpreted as the binary data seen with the “EventData” tag.

Here “WmiApRpl” is the first string and “44415441” the fourth.

Event strings can also be stored as attribute values.

```
  <UserData>
    <EventProcessingFailure xmlns="http://manifests.microsoft.com/...">
      <Error Code="15007"/>
      <EventID>4616</EventID>
      <PublisherID>Microsoft-Windows-Security-Auditing</PublisherID>
    </EventProcessingFailure>
  </UserData>
```

```
EventProcessingFailure
  Error
     [Code] 15007
  EventID 4616
  PublisherID Microsoft-Windows-Security-Auditing
```

The corresponding message string is:

```
The event logging service encountered an error while processing an incoming
event published from %3.
```

Which indicates the attribute value should be considered the first event string.

Some event records have a corresponding template definition in the WEVT_TEMPLATE data.

An example of an event record with a corresponding template definition is:

```
  <EventData Name="EVENT_HIVE_LEAK">
    <Data Name="Detail">1 user registry handles leaked from ...</Data>
  </EventData>
```

```
<EventData Name="EVENT_HIVE_LEAK">
  <Data Name="Detail">Detail</Data>
</EventData>
```

Note that not all event records have corresponding WEVT_TEMPLATE data or template definition. Sometimes the template definition does not entirely match the event record e.g. the following example where the template definition contains Name="%1" but not the event record.

```
  <EventData>
    <Data>http://www.download.windowsupdate.com/...</Data>
    <Data>The data is invalid.</Data>
  </EventData>
```

```
<EventData>
  <Data Name="%1">%1</Data>
  <Data Name="%2">%2</Data>
</EventData>
```

This however might be a special case of the “EventData”.

5.4.1.6. Parsing event data

In the initial phases of the libevtx project several attempts have been made to uniformly parse the event data.

Firstly the naive approach. This approach considers the element values of the sub elements of the “EventData” or “UserData” elements as event string. Alas this approach fails to handle event strings that are defined as element attributes values mainly seen in “UserData” elements, e.g.

```
  <UserData>
    <EventProcessingFailure xmlns="http://manifests.microsoft.com/...">
      <Error Code="15007"/>
      <EventID>4616</EventID>
      <PublisherID>Microsoft-Windows-Security-Auditing</PublisherID>
    </EventProcessingFailure>
  </UserData>
```

The next approach was to use the template definitions, if available, to parse the “EventData” and “UserData” elements. This approach seemed to solve the issue with the event strings defined as attribute values. Alas not every template definition seem to match the event record data, at least for some of the “EventData” elements, e.g.

```
  <EventData>
    <Data>http://www.download.windowsupdate.com/...</Data>
    <Data>The data is invalid.</Data>
  </EventData>
```

```
<EventData>
  <Data Name="%1">%1</Data>
  <Data Name="%2">%2</Data>
</EventData>
```

However using the template definitions to parse the event data proved an interesting insight that the the binary XML substitution tokens of the template definition match those of the event record. Which is the technique used as of version 20130208.

5.4.2. Category

TODO: CategoryMessageFile

6. Recovery

1. Scan the chunk free space for event records and make sure the size and copy of size match.
2. Ignore any record with an identifier that already exists. Often the free space contains former versions of existing event records.
3.

How useful are former versions of event records for correcting corrupted event records?

6.1. Detecting corrupted records

Comparing the size and copy of size is a quick way to detect corrupted records but sometimes the sizes match while the record is not recoverable. The detection of corrupted records can be improved by looking at:

- the Binary XML data.

TODO what about the identifier is it signed?

According [MS-EVEN6] the binary XML structure should consist of:

The document (BinXMLDocument) consists of:

- Prologue (BinXMLPI) (zero or one)
- Fragment (zero or more)
- Miscellaneous (BinXMLPI) (zero or one)
- End of file token

This translates to the Binary XML data should start with either:

- 0x0a; the data size must be 5 or more bytes (for EVTX)
- 0x0f 0x01 0x01 0x00; the data size must be 4 or more bytes
- 0x00; which means there is no Binary XML data

7. Corruption scenarios

7.1. String value oddities

This has been seen in PI data and CDATA section structures.

```
libevtx_binary_xml_document_read_pi_data: type                  : 0x0b
libevtx_binary_xml_document_read_pi_data: number of characters  : 18
libevtx_binary_xml_document_read_pi_data: value data:
00000000: 4d 00 79 00 50 00 69 00  44 00 61 00 74 00 61 00   M.y.P.i. D.a.t.a.
00000010: 3d 00 22 00 76 00 61 00  6c 00 75 00 65 00 22 00   =.".v.a. l.u.e.".
00000020: 01 ff ff 0f 05 ff ff 0f                            ....
```

EventViewer seems to interpret 05 ff ff 0f as part of the string? But 18 x 2 seems to be the correct data size.

```
<?MyPiTarget MyPiData="value" ! <U+0FFF> ! <U+05FF>?>
```

Even 01 ff ff 0f part of the string looks like valid BinXML.

```
libevtx_binary_xml_document_read_cdata_section: type                  : 0x07
libevtx_binary_xml_document_read_cdata_section: number of characters  : 110
libevtx_binary_xml_document_read_cdata_section: value data:
00000000: 0d 00 0a 00 66 00 75 00  6e 00 63 00 74 00 69 00   ....f.u. n.c.t.i.
00000010: 6f 00 6e 00 20 00 6d 00  61 00 74 00 63 00 68 00   o.n. .m. a.t.c.h.
00000020: 77 00 6f 00 28 00 61 00  2c 00 62 00 29 00 0d 00   w.o.(.a. ,.b.)...
00000030: 0a 00 7b 00 0d 00 0a 00  69 00 66 00 20 00 28 00   ..{..... i.f. .(.
00000040: 61 00 20 00 3c 00 20 00  62 00 20 00 26 00 26 00   a. .<. . b. .&.&.
00000050: 20 00 61 00 20 00 3c 00  20 00 30 00 29 00 20 00    .a. .<.  .0.). .
00000060: 74 00 68 00 65 00 6e 00  0d 00 0a 00 20 00 20 00   t.h.e.n. .... . .
00000070: 7b 00 0d 00 0a 00 20 00  20 00 72 00 65 00 74 00   {..... .  .r.e.t.
00000080: 75 00 72 00 6e 00 20 00  31 00 3b 00 0d 00 0a 00   u.r.n. . 1.;.....
00000090: 20 00 20 00 7d 00 0d 00  0a 00 65 00 6c 00 73 00    . .}... ..e.l.s.
000000a0: 65 00 0d 00 0a 00 20 00  20 00 7b 00 0d 00 0a 00   e..... .  .{.....
000000b0: 20 00 20 00 72 00 65 00  74 00 75 00 72 00 6e 00    . .r.e. t.u.r.n.
000000c0: 20 00 30 00 3b 00 0d 00  0a 00 20 00 20 00 7d 00    .0.;... .. . .}.
000000d0: 0d 00 0a 00 7d 00 0d 00  0a 00 04 04 04 04         ....}... ....
```

```
      <![CDATA[
function matchwo(a,b)
{
if (a < b && a < 0) then
  {
  return 1;
  }
else
  {
  return 0;
  }
}
€]]>
```

EventViewer shows the last line as:

```
€€]]>
```

Even the 04 04 part of the string looks like valid BinXML.

7.2. Corrupted file header with correct checksum

For some reason in EVTX file the file header was written with incorrect data although the checksum checks out. As you can see the first chunk number: 206 exceeds last chunk number: 205.

```
signature                       : ElfFile\x00
first chunk number              : 206
last chunk number               : 205
next record identifier          : 123510
header size                     : 128
minor version                   : 1
major version                   : 3
header block size               : 4096
number of chunks                : 1024
flags                           : 0x00000000
checksum                        : 0x7fc747e2
```

TODO check the number of chunks in the file and if the event ids are in sequential order. At first glance it seems to be this way.

7.3. Dirty file with invalid number of chunks

In the dirty file with invalid offset values scenarios the file header indicates the incorrect number of chunks in the file; in this case less than the actual number of chunks.

```
signature                       : ElfFile\x00
first chunk number              : 0
last chunk number               : 35
next record identifier          : 150158
header size                     : 128
minor version                   : 1
major version                   : 3
header block size               : 4096
number of chunks                : 36
flags                           : 0x00000001
checksum                        : 0x98053517
```

Event Viewer seems to “correct” files that are dirty and where the number of chunks in the file header is less than the actual number of chunks.

The approach implemented in libevtx 20130713 to deal with these files is to keep scanning for chunks after the last chunk indicated by the file header. The records in these chunks are not marked as recovered records.

7.4. Corrupt event record

Corruption of an event record can occur in multiple ways, the following variant have been seen:

- In the middle of a chunk there is suddenly a large block of 0-byte values directly after an event record.
- In the middle of a chunk there is an event record that is corrupt e.g. the size of the event

record does not match the copy of size.

The approach is to start scanning for recoverable event records in the remainder of the chunk. Any event records found are considered recovered.

7.5. Corrupted chunk

Corruption of an chunk can occur in multiple ways, the following variant have been seen:

- In the middle of a chunk there is suddenly a large block of 0-byte values directly after an event record. These 0-byte values continue across the next (expected) chunk header.

The approach is to start scanning for recoverable event records until a correct chunk header is found or the end of file is reached. Any event records found are considered recovered.

8. Notes

8.1. Normal behavior

Lets consider a “normal” Application.evtx file.

EventViewer shows 20568 events.

Using “Save All Events As ...” as an XML file from EventViewer shows 4168 events.

Wevtutil get-log-info shows 20568 events.

```
wevtutil qli /lf:true file.evtx
```

TODO behavior of oldestRecordNumber

Wevtutil query-events shows 20568 events.

```
wevtutil qe /lf:true file.evtx > file.xml
```

```
cat file.xml | grep EventRecordID | wc -l
```

This file has the following header.

```
signature                       : ElfFile\x00
first chunk number              : 0
last chunk number               : 181
next record identifier          : 20569
header size                     : 128
minor version                   : 1
major version                   : 3
header block size               : 4096
number of chunks                : 182
file flags                      : 0x00000000
checksum                        : 0x9d4c00e2
```

In the file the event records are in order, meaning that the first chunk contains the event record with the lowest event record number.

```
signature                                       : ElfChnk\x00
first event record number                       : 1
last event record number                        : 117
first event record identifier                   : 1
last event record identifier                    : 117
header size                                     : 128
last event record offset                        : 0x0000e380
free space offset                               : 0x0000f3b0
event records checksum                          : 0x731087d8
```

The number of event records in the chunk should be:

```
last event record number - first event record number + 1
```

Successive chunks contain successive event record numbers.

```
signature                                       : ElfChnk\x00
first event record number                       : 118
last event record number                        : 232
first event record identifier                   : 118
last event record identifier                    : 232
header size                                     : 128
last event record offset                        : 0x0000fcc8
free space offset                               : 0x0000ff30
event records checksum                          : 0x7fa7a9df
```

TODO determine if gaps in event record identifiers is normal behavior?

8.2. Corruption scenario: event record mismatch between size and copy of size

Lets consider a dirty Security.evtx file.

EventViewer shows 4001 events.

Using “Save All Events As ...” as an XML file from EventViewer shows 1180 events.

Wevtutil get-log-info shows 4001 events.

```
wevtutil qli /lf:true file.evtx
```

The “oldestRecordNumber” is 1 and does not match the data in the file.

Wevtutil query-events shows 4001 events.

```
wevtutil qe /lf:true file.evtx > file.xml
```

```
cat file.xml | grep EventRecordID | wc -l
```

Looking at the file in more detail the following chunk seems to be corrupt.

```
signature                                       : ElfChnk\x00
first event record number                       : 72431823
last event record number                        : 72431919
first event record identifier                   : 72433834
last event record identifier                    : 72433930
```

```
header size                                             : 128
last event record offset                                : 0x0000fd18
free space offset                                       : 0x0000ffb0
event records checksum                                  : 0x6df0577c
checksum                                                : 0x5ff97a22
```

```
mismatch in chunk: 14 event records CRC-32 checksum (0x6df0577c != 0xd97de631)
```

In the middle of this chunk the size of the event record does not match the copy of size.

```
signature                               : \x2a\x2a\x00\x00
size                                    : 664
identifier                              : 72433924
written time                            : Feb 20, 2013 20:50:20.671208000 UTC
size copy                               : 1694526976
```

Judging by the data structures the size points in the middle of the binary XML.

In this case scanning for event record signatures in the remainder of the chunk yields 6 results:

- 1x corrupt event record (72433924)
- 5x recoverable event records (73882240 - 73882244)

The discontinuation in event record numbers suggest that the file was copied while event record 72433924 was being written.

By continuing scanning for event records in total 21045 event records were found with the first event number of 72432422.

8.3. Corruption scenario: cross chunk 0-byte values

Lets consider a dirty Security.evtx file.

EventViewer shows 102019 events.

Using “Save All Events As ...” as an XML file from EventViewer shows 68269 events.

Wevtutil get-log-info shows 102019 events.

```
wevtutil qli file.evtx /lf:true
```

The “oldestRecordNumber” is 20496.

Wevtutil query-events shows 19660 events.

```
wevtutil qe file.evtx /lf:true > file.xml
```

```
Failed to read events. The event log file is corrupted.
```

```
cat file.xml | grep EventRecordID | wc -l
```

Recall that in the previous corruption scenario wevtutil did not report it but in this case it does.

```
signature                                               : ElfChnk\x00
first event record number                               : 40163
last event record number                                : 40261
first event record identifier                           : 41158
last event record identifier                            : 41256
header size                                             : 128
last event record offset                                : 0x0000fba8
free space offset                                       : 0x0000fe18
event records checksum                                  : 0x9981f715
checksum                                                : 0x4931f4a2
```

```
mismatch in chunk: 402 event records CRC-32 checksum (0x9981f715 !=
0x31aa1bb0).
```

```
signature                               : \x2a\x2a\x00\x00
size                                    : 624
identifier                              : 41173
written time                            : Mar 15, 2012 11:03:23.546212500 UTC
size copy                               : 0
```

```
chunk header data:
00000000: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
...
00000070: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

By continuing scanning for event records in total 98927 event records and 1043 recoverable event records were found.

Appendix A. References

[CHAPPEL08]
Title: The Shell Core Provider
Author(s): G. Chappel
Date: December 29, 2008
URL: http://www.geoffchappell.com/notes/windows/shell/events/core.htm

[LIBEXE]
Title: MZ, PE-COFF executable file format (EXE)
Author(s): J.B. Metz
Date: October 2011
URL: http://code.google.com/p/libexe/downloads/detail?name=Executable%20%28EXE%29%20file%20format.pdf

[MS-EVEN6]
Title: EventLog Remoting Protocol Version 6.0 Specification
URL: http://msdn.microsoft.com/en-us/library/cc231282(v=prot.10).aspx

[MSDN]
Title: BinXml
URL: http://msdn.microsoft.com/en-us/library/cc231334(v=prot.10).aspx
URL: http://msdn.microsoft.com/en-us/library/cc231337(v=prot.10).aspx
URL: http://msdn.microsoft.com/en-us/library/cc231339(v=prot.10).aspx
URL: http://msdn.microsoft.com/en-us/library/aa382793%28v=VS.85%29.aspx
URL: http://msdn.microsoft.com/en-us/library/cc238875(v=prot.10).aspx

[NTSID]
Tile: NT security descriptor definitions
URL: https://downloads.sourceforge.net/project/libpff/documentation/MAPI%20definitions/NT%20security%20descriptor.pdf

[SCHUSTER07]
Title: Introducing the Microsoft Vista Event Log File Format.
Author(s): A. Schuster
Date: 2007
URL: http://www.dfrws.org/2007/proceedings/p65-schuster_pres.pdf

[SCHUSTER10]
Tittle: Linking Event Messages and Resource DLLs
Author(s): A. Schuster
Date: October 5, 2010
URL: http://computer.forensikblog.de/en/2010/10/linking-event-messages-and-resource-dlls.html

[SCHUSTER11]
Title: Microsoft Windows Event Logging - Dokumentation der Binärformate
Author(s): A. Schuster
Version: 148
Date: February 6, 2011

[W3C]

Title: Extensible Markup Language (XML) 1.0 (Fifth Edition)
Date: November 26, 2008
URL: http://www.w3.org/TR/REC-xml/

Appendix B. GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE
The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS
This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those

of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the

conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in

the Addendum below.

- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS
You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS
A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION
Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION
You may not copy, modify, sublicense, or distribute the Document except as expressly provided

under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.