CybOX 3.0 Specification - Pre-Draft

CybOX Host Objects – Version 0.1

Specifications Cover Page

1. Objects

A CybOX object represents an instance of information observed on a host or network; for example, this could include the properties of a PDF file observed on an endpoint or a network connection between two IP addresses as observed by a firewall.

CybOX defines a set of **object data models,** such as the *File Object*, for the normalized capture of observed data as supporting evidence in STIX or a similar, higher-level language.

Defined CybOX **object data models** provide a base set of **properties** which are applicable across a broad spectrum of use cases relevant to the particular data model as well as (in certain cases) a set of object **extensions** targeting more specific use cases.

2. File Object

Type Name: `file-object`

The File Object represents the properties of a file. A File Object **MUST** contain at least one of **hashes** OR **file_name** OR **file_name_hex**.

2.1. Properties

Common Properties		
type, description, extended_properties		
File Object Specific Properties		
hashes, size, file_name, file_name_enc, file_name_hex, magic_number, mime_type, created, modified, accessed, parent_director_ref, is_encrypted, encryption_algorithm, decryption_key, contains_ref, file_content_ref		
Property Name	**Type**	**Description**

type (required)	`string`	The value of this field **MUST** be `file-object`.
extended_properties (optional)	`dictionary`	The File Object defines the following extensions. In addition to these, producers **MAY** create their own. `file-metadata-mismatch-extension`, `ntfs-file-extension`, `raster-image-file-extension`, `pdf-file-extension`, `archive-file-extension`, `windows-pebinary-file-extension` Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.
hashes (optional)	`hashes-type`	Specifies a dictionary of hashes for the file.
size (optional)	`integer`	Specifies the size of the file, in bytes, as a non-negative integer.
file_name (optional)	`string`	Specifies the name of the file.
file_name_enc (optional)	`string`	Specifies the observed encoding for the name of the file. This value **MUST** be specified using the corresponding name from the 2013-12-20 revision of the IANA character set registry. If the value from the the Preferred MIME Name column for a character set is defined, this value **MUST** be used; if it is not defined, then the value from the Name column in the registry **MUST** be used instead. This field allows for the capture of the original text encoding for the file name, which may be forensically relevant; for example, a file on an NTFS volume whose name was created using the windows-1251 encoding, commonly used for languages based on Cyrillic script

file_name_hex (optional)	`hex`	Specifies the name of the file as a hex-encoded string. This field **MUST NOT** be specified in conjunction with the **file_name** field; only one of **file_name** OR **file_name_hex** may be used.
magic_number (optional)	`hex`	Specifies the hexadecimal constant ("magic number") associated with a specific file format that corresponds to the file, if applicable.
mime_type (optional)	`string`	Specifies the MIME type name specified for the file, e.g., "application/msword". Whenever feasible, this value **SHOULD** be one of the values defined in the Template column in the IANA media type registry, located at [IANA] (http://www.iana.org/assignments/media-types/media-types.xhtml). Maintaining a comprehensive universal catalog of all extant file types is obviously not possible. When specifying a mime_type not included in the IANA registry, implementers should use their best judgement so as to facilitate interoperability.
created (optional)	`timestamp`	Specifies the date/time the file was created.
modified (optional)	`timestamp`	Specifies the date/time the file was last written to/modified.
accessed (optional)	`timestamp`	Specifies the date/time the file was last accessed.
parent_directory_ref (optional)	`object-ref`	Specifies the parent directory of the file, as a reference to a Directory Object. The object referenced in this property **MUST** be of type `directory-object`.
is_encrypted (optional)	`boolean`	Specifies whether the file is encrypted. The default value is `false`.
encryption_algorithm (optional)	`open-vocab`	Specifies the name of the encryption algorithm used to encrypt the file. This is an open vocabulary and values **SHOULD** come

		from the `encryption-algorithm-ov` vocabulary. This field **MUST NOT** be used if **is_encrypted** is `false` or not included.
decryption_key (optional)	`string`	Specifies the decryption key used to decrypt the archive file. This field **MUST NOT** be used if **is_encrypted** is `false` or not included.
contains_refs (optional)	`list` of type `object-ref`	Specifies a list of references to other CybOX Objects contained within the file, such as another file that is appended to the end of the file, or an IP address that is contained somewhere in the the file. (This is intended for use cases other than those targeted by the Archive extension.)
file_content_ref (optional)	`object-ref`	Specifies the content of the file, represented as an Artifact Object. The object referenced in this property **MUST** be of type `artifact-object`.

Examples

Basic file with file system properties without observed encoding

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes":{
        "md5":"4472ea40dc71e5bb701574ea215a81a1"
      },
      "size":25536,
      "file_name":"foo.dll"
      }
    }
  }
}
```

Basic file with file system properties with observed encoding

```
{
```

```
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes":{
        "md5":"66e2ea40dc71d5ba701574ea215a81f1"
      },
      "file_name":"quêry.dll",
      "file_name_enc":"windows-1252"
  }
}
```

In this example, the file name would have originally appeared using the bytes 71 75 **ea** 72 79 2e 64 6c 6c. Representing it in UTF-8, as required for JSON, would use the bytes 71 75 **c3 aa** 72 79 2e 64 6c 6c.

Basic file with parent directory

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"directory-object",
      "path":"C:\\Windows\\System32"
    },
    "1":{
      "type":"file-object",
      "hashes":{
        "md5":"A2FD2B3F4D5A1BD5E7D283299E01DCE9"
      },
      "parent_directory_ref":"0",
      "file_name":"qwerty.dll"
  }
}
```

2.2. NTFS File Extension

Type Name: `ntfs-file-extension`

The NTFS File extension specifies a default extension for capturing properties specific to the storage of the file on the NTFS file system. The key for this extension when used in the **`extended_properties`** dictionary MUST be *ntfs*.

2.2.1. Properties

Property Name	Type	Description
sid (optional)	`string`	Specifies the security ID (SID) value assigned to the file.
alternate_data_streams (optional)	`list` of type `alternate-data-stream-type`	Specifies a list of NTFS alternate data streams that exist for the file.

2.2.2. Alternate Data Stream Type

Type Name: `alternate-data-stream-type`

The Alternate Data Stream type represents an NTFS alternate data stream.

2.2.2.1. Properties

Property Name	Type	Description
name (required)	`string`	Specifies the name of the alternate data stream.
hashes (optional)	`hashes-type`	Specifies a dictionary of hashes for the data contained in the alternate data stream.
size (optional)	`integer`	Specifies the size of the alternate data stream, in bytes, as a non-negative integer.

2.2.3. Example

File with a single alternate data stream

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes-type":{
```

```
        "md5":"B4D33B0C7306351B9ED96578465C5579"
      },
      "extended_properties":{
        "ntfs":{
          "alternate_data_streams":[
            {
              "type":"alternate-data-stream",
              "name":"second.stream",
              "size":25536
            }
          ]
        }
      }
    }
  }
}
```

2.3. Raster Image File Extension

Type Name: `raster-image-file-extension`

The Raster Image File extension specifies a default extension for capturing properties specific to image files. The key for this extension when used in the **`extended_properties`** dictionary **MUST** be *raster-image*.

2.3.1. Properties

Property Name	Type	Description
`image_height` (optional)	`integer`	Specifies the height of the image in the image file, in pixels.
`image_width` (optional)	`integer`	Specifies the width of the image in the image file, in pixels.
`bits_per_pixel` (optional)	`integer`	Specifies the sum of bits used for each color channel in the image in the image file, and thus the total number of pixels used for expressing the color depth of the image.
`image_compression_algorithm` (optional)	`string`	Specifies the name of the compression algorithm used to compress the image in the image file, if applicable.
`exif_tags` (optional)	`dictionary`	Specifies the set of EXIF tags found in

		the image file, as a dictionary. Each key/value pair in the dictionary represents the name/value of a single EXIF tag. Accordingly, each dictionary key **MUST** be a lowercase string version of the EXIF tag name, e.g., “imagewidth”. Each dictionary value **MUST** be either an integer (for int* EXIF datatypes) or a string (for all other EXIF datatypes).

Example

Simple Image File w/ EXIF Data

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes-type":{
        "md5":"B4D33B0C7306351B9ED96578465C5579"
      },
      "extended_properties":{
        "raster-image":{
          "image_is_compressed":true,
          "exif_tags":{
            "make":"Nikon",
            "model":"D7000",
            "xresolution":4928,
            "yresolution":3264
          }
        }
      }
    }
  }
}
```

2.4. PDF File Extension

Type Name: `pdf-file-extension`

The PDF File extension specifies a default extension for capturing properties specific to PDF files. The key for this extension when used in the **`extended_properties`** dictionary **MUST** be *pdf*.

2.4.1. Properties

Property Name	Type	Description
`version` (optional)	`string`	Specifies the decimal version number of the string from the PDF header that specifies the version of the PDF specification to which the PDF file conforms. E.g., “1.4”.
`is_optimized` (optional)	`boolean`	Specifies whether the PDF file has been optimized.
`document_information_dictionary` (optional)	`dictionary`	Specifies details of the PDF document information dictionary (DID), which includes properties like the document creation data and producer, as a dictionary. Each key in the dictionary **MUST** be a lowercase version of the corresponding entry in the document information dictionary, e.g., “title”. The corresponding value for the key **MUST** be the value specified for the document information dictionary entry, as a `string`.
`pdfid0` (optional)	`string`	Specifies the first file identifier found for the PDF file.
`pdfid1` (optional)	`string`	Specifies the second file identifier found for the PDF file.

Example
Basic PDF file

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes":{
        "md5":"66e2ea40dc71d5ba701574ea215a81f1"
      },
      "extended_properties":{
        "pdf":{
          "version":"1.7",
```

```
          "pdfid0":"DFCE52BD827ECF765649852119D",
          "pdfid1":"57A1E0F9ED2AE523E313C"
        }
      }
    }
  }
}
```

2.5. Archive File Extension

Type Name: `archive-file-extension`

The Archive File extension specifies a default extension for capturing properties specific to archive files. The key for this extension when used in the **`extended_properties`** dictionary **MUST** be *archive*.

2.5.1. Properties

Property Name	Type	Description
`file_refs` (required)	`list` of type `object-ref`	Specifies the files contained in the archive, as a reference to one or more other File Objects. The objects referenced in this list **MUST** be of type `file-object`.
`version` (optional)	`string`	Specifies the version of the archive type used in the archive file.
`comment` (optional)	`string`	Specifies a comment included as part of the archive file.

2.5.2. Example

Basic unencrypted ZIP Archive

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes":{
```

```
        "md5":"66e2ea40dc71d5ba701574ea215a81f1"
      }
    },
    "1":{
      "type":"file-object",
      "hashes":{
        "md5":"22A0FB8F3879FB569F8A3FF65850A82E"
      }
    },
    "2":{
      "type":"file-object",
      "hashes":{
        "md5":"8D98A25E9D0662B1F4CA3BF22D6F53E9"
      }
    },
    "3":{
      "type":"file-object",
      "hashes":{
        "md5":"B365B9A80A06906FC9B400C06C33FF43"
      },
      "mime_type":"application/zip",
      "extended_properties":{
        "archive":{
          "file_refs":[
            "0",
            "1",
            "2"
          ],
          "version":"5.0"
        }
      }
    }
  }
}
```

2.6. Windows™ PE Binary File Extension

Type Name: `windows-pebinary-file-extension`

The Windows PE Binary File extension specifies a default extension for capturing properties specific to Windows portable executable (PE) files. The key for this extension when used in the **`extended_properties`** dictionary **MUST** be *windows-pebinary*.

2.6.1. Properties

Property Name	Type	Description
pe_type (required)	`open-vocab`	Specifies the type of the PE binary. This is an open vocabulary and values **SHOULD** come from the `windows-pebinary-type-ov` vocabulary.
signed_with_ref (optional)	`object-ref`	Specifies the certificate used to sign the PE binary, as a reference to an X509 Certificate Object. The object referenced in this property **MUST** be of type `x509-certificate-object`.
imphash (optional)	`string`	Specifies the special import hash, or 'imphash', calculated for the PE Binary based on its imported libraries and functions. For more information on the imphash algorithm, see the original article by Mandiant/FireEye: https://www.fireeye.com/blog/threat-research/2014/01/tracking-malware-import-hashing.html
file_header (optional)	`windows-pe-file-header-type`	Specifies the PE file header (sometimes referred to as the COFF header) of the PE binary.
optional_header (optional)	`windows-pe-optional-header-type`	Specifies the PE optional header of the PE binary.
sections (optional)	`list` of type `windows-pe-section-type`	Specifies metadata about the sections in the PE file.

2.6.2. Windows PE Binary Type Vocabulary

Type Name: `windows-pebinary-type-ov`

An open vocabulary of Windows PE binary types.

Value	Description
`exe`	Specifies that the PE binary is an executable image (i.e., not an OBJ or DLL).
`dll`	Specifies that the PE binary is a dynamically linked library (DLL).
`sys`	Specifies that the PE binary is a device driver (SYS).

2.6.3. PE File Header Type

Type Name: `windows-pe-file-header-type`

The PE File Header type represents the properties of the PE file header (sometimes referred to as the COFF header.)

2.6.3.1. Properties

Property Name	Type	Description
`machine` (required)	`hex`	Specifies the type of target machine.
`number_of_sections` (optional)	`integer`	Specifies the number of sections in the PE binary, as a non-negative integer.
`time_date_stamp` (optional)	`timestamp`	Specifies the time when the PE binary was created. The timestamp value **MUST BE** precise to the second.
`pointer_to_symbol_table` (optional)	`hex`	Specifies the file offset of the COFF symbol table.
`number_of_symbols` (optional)	`integer`	Specifies the number of entries in the symbol table of the PE binary, as a non-negative integer.
`size_of_optional_h`	`integer`	Specifies the size of the optional header of the PE

eader (optional)		binary.
characteristics (optional)	`hex`	Specifies the flags that indicate the file's characteristics.
hashes (optional)	`hashes-type`	Specifies any hashes that were computed for the file header.

2.6.4. PE Optional Header

Type Name: `windows-pe-optional-header-type`

The Windows PE Optional Header type represents the properties of the PE optional header.

2.6.4.1. Properties

Property Name	Type	Description
magic (optional)	`hex`	Specifies the unsigned integer that indicates the type of the PE binary.
major_linker_version (optional)	`integer`	Specifies the linker major version number.
minor_linker_version (optional)	`integer`	Specifies the linker minor version number.
size_of_code (optional)	`integer`	Specifies the size of the code (text) section. If there are multiple such sections, this refers to the sum of the sizes of each section.
size_of_initialized_data (optional)	`integer`	Specifies the size of the initialized data section. If there are multiple such sections, this refers to the sum of the sizes of each section.
size_of_uninitialized_data (optional)	`integer`	Specifies the size of the uninitialized data section. If there are multiple such sections, this refers to the sum of the sizes of each section.
address_of_entry_point (optional)	`hex`	Specifies the address of the entry point relative to the image base when the executable is loaded into memory.
base_of_code (optional)	`hex`	Specifies the address that is relative to the image base of the beginning-of-code section

		when it is loaded into memory.
base_of_data (optional)	`hex`	Specifies the address that is relative to the image base of the beginning-of-data section when it is loaded into memory.
image_base (optional)	`hex`	Specifies the preferred address of the first byte of the image when loaded into memory.
section_alignment (optional)	`integer`	Specifies the alignment (in bytes) of PE sections when they are loaded into memory.
file_alignment (optional)	`integer`	Specifies the factor (in bytes) that is used to align the raw data of sections in the image file.
major_os_version (optional)	`integer`	Specifies the major version number of the required operating system.
minor_os_version (optional)	`integer`	Specifies the minor version number of the required operating system.
major_image_version (optional)	`integer`	Specifies the major version number of the image.
minor_image_version (optional)	`integer`	Specifies the minor version number of the image.
major_subsystem_version (optional)	`integer`	Specifies the major version number of the subsystem.
minor_subsystem_version (optional)	`integer`	Specifies the minor version number of the subsystem.
win32_version_value (optional)	`hex`	Specifies the reserved win32 version value,.
size_of_image (optional)	`integer`	Specifies the size, in bytes, of the image, including all headers, as the image is loaded in memory.
size_of_headers (optional)	`integer`	Specifies the combined size of the MS-DOS, PE header, and section headers, rounded up a multiple of the value specified in the file_alignment header.
checksum (optional)	`hex`	Specifies the checksum of the PE binary.
subsystem (optional)	`hex`	Specifies the subsystem (e.g., GUI, device driver, etc.) that is required to run this image.

dll_characteristics (optional)	`hex`	Specifies the flags that characterize the PE binary.
size_of_stack_reserve (optional)	`integer`	Specifies the size of the stack to reserve.
size_of_stack_commit (optional)	`integer`	Specifies the size of the stack to commit.
size_of_heap_reserve (optional)	`integer`	Specifies the size of the local heap space to reserve.
size_of_heap_commit (optional)	`integer`	Specifies the size of the local heap space to commit.
loader_flags (optional)	`hex`	Specifies the reserved loader flags.
number_of_rva_and_sizes (optional)	`integer`	Specifies the number of data-directory entries in the remainder of the optional header.
hashes (optional)	`hashes-type`	Specifies any hashes that were computed for the optional header.

2.6.5. 2.8.4.Windows PE Section Type

Type Name: `windows-pe-section-type`

The PE Section type specifies metadata about a PE file section.

2.6.5.1. Properties

Property Name	Type	Description
name (required)	`string`	Specifies the name of the section.
size (optional)	`integer`	Specifies the size of the section, in bytes.
entropy (optional)	`float`	Specifies the calculated entropy for the section, as calculated using the Shannon algorithm (https://en.wiktionary.org/wiki/Shannon_entropy). The size of each input character is defined as a byte, resulting in a possible range

		of 0 through 8.
hashes (optional)	`hashes-type`	Specifies any hashes computed over the section.

2.6.6. Examples

Typical EXE File


```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes":{
        "md5":"1C19FC56AEF2048C1CD3A5E67B099350"
      },
      "extended_properties":{
        "windows-pebinary":{
          "pe_type":"exe",
          "file_header":{
            "machine":"014c",
            "number_of_sections":4,
            "time_date_stamp":"2016-01-22T12:31:12",
            "pointer_to_symbol_table":"74726144",
            "number_of_symbols":4542568,
            "size_of_optional_header":224,
            "characteristics":"818f"
          },
          "optional_header":{
            "magic":"010b",
            "major_linker_version":2,
            "minor_linker_version":25,
            "size_of_code":512,
            "size_of_initialized_data":283648,
            "size_of_unitialized_data":0,
            "address_of_entrypoint":"2000",
            "base_of_code":"1000",
            "base_of_data":"2000",
            "image_base":"de0000",
            "section_alignment":4096,
            "file_alignment":4096,
            "major_operating_system_version":1,
            "minor_operating_system_version":0,
            "major_image_version":0,
            "minor_image_version":0,
            "major_subsystem_version":4,
```

```
            "minor_subsystem_version":0,
            "win32_version_value":"00",
            "size_of_image":299008,
            "size_of_headers":4096,
            "checksum":"00",
            "subsystem":"03",
            "dll_characteristics":"00",
            "size_of_stack_reserve":"100000",
            "size_of_stack_commit":8192,
            "size_of_heap_reserve":"100000",
            "size_of_heap_commit":4096,
            "loader_flags":"abdbffde",
            "number_of_rva_and_sizes":"dfffddde"
          },
          "sections":[
            {
              "name":"CODE",
              "entropy":0.061089
            },
            {
              "name":"DATA",
              "entropy":7.980693
            },
            {
              "name":"NicolasB",
              "entropy":0.607433
            },
            {
              "name":".idata",
              "entropy":0.607433
            }
          ]
        }
      }
    }
  }
}
```


3. Directory Object

Type Name: `directory-object`

The Directory Object represents the properties of a file system directory.

3.1. Properties

The base Directory Object type that defines the set of properties common to a directory.

Common Properties		
type, description, extended_properties		
File Object Specific Properties		
path, path_enc, path_hex, created, modified, accessed, contains_refs		
Property Name	**Type**	**Description**
type (required)	`string`	The value of this field **MUST** be `directory-object`.
path (required)	`string`	Specifies the path to the directory on the file system.
path_enc (optional)	`string`	Specifies the observed encoding for the path. The value **MUST** be specified if the path is stored in a non-Unicode encoding. This value **MUST** be specified using the corresponding name from the 2013-12-20 revision of the IANA character set registry. If the preferred MIME name for a character set is defined, this value **MUST** be used; if it is not defined, then the Name value from the registry **MUST** be used instead.
path_hex (optional)	`hex`	Specifies the directory path as a hexadecimal string. This field **MUST NOT** be specified in conjunction with the **path** field; only one of **path** OR **path_b64** may be used.
created (optional)	`timestamp`	Specifies the date/time the directory was created.
modified (optional)	`timestamp`	Specifies the date/time the directory was last written to/modified.
accessed (optional)	`timestamp`	Specifies the date/time the directory was last accessed.
contains_refs (optional)	`list` of type `object-ref`	Specifies a list of references to

		other CybOX File and/or Directory Objects contained within the directory.

Example

Basic directory

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type": "directory-object",
      "path":"C:\\Windows\\System32"
      }
  }
}
```

4. Windows Registry Key Object

Type Name: `windows-registry-key-object`

The Registry Key Object represents the properties of a Windows registry key.

4.1. Properties

Common Properties		
`type, description, extended_properties`		
File Object Specific Properties		
`key, values, modified, creator_ref, number_of_subkeys`		
Property Name	**Type**	**Description**
type (required)	`string`	The value of this field **MUST** be `windows-registry-key-object` .
key (required)	`string`	Specifies the full registry key, as a case-preserved string, including the hive. The hive **MUST** be fully expanded and not truncated; e.g.,

		HKEY_LOCAL_MACHINE must be used instead of HKLM.
values (optional)	`list` of type `windows-registry-value-type`	Specifies the values found under the registry key.
modified (optional)	`timestamp`	Specifies the last date/time that the registry key was modified.
creator_ref (optional)	`object-ref`	Specifies a reference to a user account, represented as a User Account Object, that created the registry key. The object referenced in this field **MUST** be of type `user-account-object`.
number_of_subkeys (optional)	`integer`	Specifies the number of subkeys contained under the registry key.

4.3. Windows Registry Value Type

Type Name: `windows-registry-value-type`

4.3.1. Properties

Property Name	Type	Description
name (required)	`string`	Specifies the name of the registry value, as a lowercase string. For specifying the default value in a registry key, an empty string **MUST** be used.
data (optional)	`string`	Specifies the data contained in the registry value.
data_type (optional)	`controlled-vocab`	Specifies the registry (REG_*) datatype used in the registry

		value. This is a controlled vocabulary and values **MUST** come from the `windows-registry-data-type-cv` vocabulary.

4.4. Registry Datatype Vocabulary

Type Name: `windows-registry-data-type-cv`

A controlled vocabulary of Windows registry data types.

Vocabulary Value	Description
`reg_none`	No defined value type.
`reg_sz`	A null-terminated string. This will be either a Unicode or an ANSI string, depending on whether you use the Unicode or ANSI functions.
`reg_expand_sz`	A null-terminated string that contains unexpanded references to environment variables (for example, "%PATH%"). It will be a Unicode or ANSI string depending on whether you use the Unicode or ANSI functions.
`reg_binary`	Binary data in any form.
`reg_dword`	A 32-bit number.
`reg_dword_big_endian`	A 32-bit number in big-endian format.
`reg_link`	A null-terminated Unicode string that contains the target path of a symbolic link.
`reg_multi_sz`	A sequence of null-terminated strings, terminated by an empty string (\0).
`reg_resource_list`	A series of nested lists designed to store a resource list used by a hardware device driver or one of the physical devices it controls. This data is detected and written into the ResourceMap tree by the system and is displayed in Registry Editor in hexadecimal format as a Binary Value.
`reg_full_resource_descrip`	A series of nested lists designed to store a resource list

`tor`	used by a physical hardware device. This data is detected and written into the HardwareDescription tree by the system and is displayed in Registry Editor in hexadecimal format as a Binary Value.
`reg_resource_requirements_list`	Device driver list of hardware resource requirements in Resource Map tree.
`reg_qword`	A 64-bit number.
`reg_invalid_type`	Specifies an invalid key.

Example
Simple registry key

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type": "windows-registry-key-object",
      "key":"hkey_local_machine\\system\\foo\\bar"
      }
  }
}
```

5. Mutex Object

Type Name: `mutex-object`

The Mutex Object represents the properties of a mutual exclusion object.

5.1. Properties

Common Properties		
`type, description, extended_properties`		
File Object Specific Properties		
`name`		
Property Name	**Type**	**Description**

type (required)	`string`	The value of this field **MUST** be `mutex-object`.
name (required)	`string`	Specifies the name of the mutex object.

Example

Malware mutex

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type": "mutex-object",
      "name":"__CLEANSWEEP__"
      }
  }
}
```

6. X509 Certificate Object

Type Name: `x509-certificate-object`

The X509 Certificate Object represents the properties of an X.509 certificate, as defined by ITU recommendation X.509.

6.1. Properties

Common Properties		
type, description, extended_properties		
File Object Specific Properties		
is_self_signed, hashes, version, serial_number, signature_algorithm, issuer, validity_not_before, validity_not_after, subject, subject_public_key_modulus, subject_public_key_exponent, x509_v3_extensions		
Property Name	**Type**	**Description**
type (required)	`string`	The value of this field **MUST** be `x509-certificate-object`.

is_self_signed (optional)	`boolean`	Specifies whether the certificate is self-signed, i.e., whether it is signed by the same entity whose identity it certifies.
hashes (optional)	`hashes-type`	Specifies any hashes that were calculated for the entire contents of the certificate.
version (optional)	`string`	Specifies the version of the encoded certificate.
serial_number (optional)	`string`	Specifies the unique identifier for the certificate, as issued by a specific Certificate Authority.
signature_algorithm (optional)	`string`	Specifies the name of the algorithm used to sign the certificate.
issuer (optional)	`string`	Specifies the name of the Certificate Authority that issued the certificate.
validity_not_before (optional)	`timestamp`	Specifies the date on which the certificate validity period begins.
validity_not_after (optional)	`timestamp`	Specifies the date on which the certificate validity period ends.
subject (optional)	`string`	Specifies the name of the entity associated with the public key stored in the subject public key field of the certificate.
subject_public_key_algorithm (optional)	`string`	Specifies the name of the algorithm with which to encrypt data being sent to the subject.
subject_public_key_modulus (optional)	`string`	Specifies the modulus portion of the subject's public RSA key.
subject_public_key_exponent (optional)	`integer`	Specifies the exponent portion of the subject's public RSA key, as an integer.
x509_v3_extensions (optional)	`x509-v3-extensions-type`	Specifies any standard X.509 v3 extensions that may be used in the certificate.

6.3. X509 v3 Extensions Type

Type Name: `x509-v3-extensions-type`

6.3.1. Properties

Property Name	Type	Description
`basic_constraints` (optional)	`string`	Specifies a multi-valued extension which indicates whether a certificate is a CA certificate. The first (mandatory) name is CA followed by TRUE or FALSE. If CA is TRUE then an optional pathlen name followed by an non-negative value can be included. Also equivalent to the object ID (OID) value of 2.5.29.19.
`name_constraints` (optional)	`string`	Specifies a namespace within which all subject names in subsequent certificates in a certification path MUST be located. Also equivalent to the object ID (OID) value of 2.5.29.30.
`policy_constraints` (optional)	`string`	Specifies any constraints on path validation for certificates issued to CAs. Also equivalent to the object ID (OID) value of 2.5.29.36.
`key_usage` (optional)	`string`	Specifies a multi-valued extension consisting of a list of names of the permitted key usages. Also equivalent to the object ID (OID) value of 2.5.29.15.
`extended_key_usage` (optional)	`string`	Specifies a list of usages indicating purposes for which the certificate public key can be used for. Also equivalent to the object ID (OID) value of 2.5.29.37.

subject_key_identifier (optional)	`string`	Specifies the identifier that provides a means of identifying certificates that contain a particular public key. Also equivalent to the object ID (OID) value of 2.5.29.14.
authority_key_identifier (optional)	`string`	Specifies the identifier that provides a means of identifying the public key corresponding to the private key used to sign a certificate. Also equivalent to the object ID (OID) value of 2.5.29.35.
subject_alternative_name (optional)	`string`	Specifies the additional identities to be bound to the subject of the certificate. Also equivalent to the object ID (OID) value of 2.5.29.17.
issuer_alternative_name (optional)	`string`	Specifies the additional identities to be bound to the issuer of the certificate. Also equivalent to the object ID (OID) value of 2.5.29.18.
subject_directory_attributes (optional)	`string`	Specifies the identification attributes (e.g., nationality) of the subject. Also equivalent to the object ID (OID) value of 2.5.29.9.
crl_distribution_points (optional)	`string`	Specifies how CRL information is obtained. Also equivalent to the object ID (OID) value of 2.5.29.31.
inhibit_any_policy (optional)	`string`	Specifies the number of additional certificates that may appear in the path before anyPolicy is no longer permitted. Also equivalent to the object ID (OID) value of 2.5.29.54.
private_key_usage_period_not_before (optional)	`timestamp`	Specifies the date on which the validity period begins for the private key, if it is different from the validity period of the certificate.
private_key_usage_period_not_after (optional)	`timestamp`	Specifies the date on which the validity period ends for the private key, if it is different from the validity period of the certificate.

certificate_policies (optional)	`string`	Specifies a sequence of one or more policy information terms, each of which consists of an object identifier (OID) and optional qualifiers. Also equivalent to the object ID (OID) value of 2.5.29.32.
policy_mappings (optional)	`string`	Specifies one or more pairs of OIDs; each pair includes an issuerDomainPolicy and a subjectDomainPolicy. The pairing indicates whether the issuing CA considers its issuerDomainPolicy equivalent to the subject CA's subjectDomainPolicy. Also equivalent to the object ID (OID) value of 2.5.29.33.

Example

Simple x.509 certificate

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type": "x509-certificate-object",
      "issuer":"C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc, OU=Certification
Services Division, CN=Thawte Server CA/emailAddress=server-certs@thawte.com",
      "subject":"C=US, ST=Maryland, L=Pasadena, O=Brent Baccala, OU=FreeSoft,
CN=www.freesoft.org/emailAddress=baccala@freesoft.org",
      "validity_not_before":"2016-03-12T12:00:00Z",
      "validity_not_after":"2016-08-21T12:00:00Z"
      }
  }
}
```

7. Software Object

Type Name: `software-object`

The Software Object represents high-level properties associated with software, including software products.

7.1. Properties

Property Name	Type	Description
type (required)	`string`	The value of this field **MUST** be `software-object`.
name (required)	`string`	Specifies the name of the software.
language (optional)	`string`	Specifies the language of the software. The value of this field **MUST** be an ISO 639-2 language code.
vendor (optional)	`string`	Specifies the name of the vendor of the software.
version (optional)	`string`	Specifies the version of the software.
swid (optional)	`string`	Specifies the SWID identifier for the software product.
extended_properties (optional)	`dictionary`	Specifies any extended properties of the object, as a dictionary. Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.

Example

Typical Software Instance

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"software-object",
      "name":"word",
      "update":"sp3",
      "version":"2002",
      "vendor":"microsoft"
    }
  }
}
```

8. Artifact Object

Type Name: `artifact-object`

The Artifact Object permits capturing an array of bytes (8-bits), as a hexadecimal-encoded string, or linking to a file-like payload. The size of the hexadecimal-encoded data captured in the **payload** field **MUST** be less than or equal to **10MB**.

It is incumbent on object creators to ensure that the URL is accessible for downstream consumers. If a URL is provided, then the **hashes** field **MUST** contain the hash of the URL contents. The hash **SHOULD** be either sha-256 or sha-512.

8.1. Properties

Property Name	Type	Description
`type` (required)	`string`	The value of this field **MUST** be `artifact-object`.
`mime_type` (optional)	`string`	The value of this field **MUST** be a valid MIME type as specified in the IANA Media Types registry.
`description` (optional)	`string`	The value of this field is a free-text description of the artifact and/or its contents. It is primarily intended for characterizing files where there is no valid MIME type for their contents; for example, in the case of shellcode, an a.out, or an ELF, or other binary data format.
`payload` (optional)	`hex`	Specifies artifact data as a hexadecimal string. This field **MUST NOT** be present if **url** is provided.
`url` (optional)	`string`	The value of this field **MUST** be a valid URL that resolves to the

		unencoded content. This field **MUST NOT** be present if **payload** is provided.
hashes (optional)	`hashes-type`	Specifies one or more hashes of the contents of the **url** or the payload.
extended_properties (optional)	`dictionary`	Specifies any extended properties of the object, as a dictionary. Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.

Example

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type": "artifact-object",
      "mime_type": "image/jpeg",
      "payload": "VBORw0KGgoAAAANSUhEUgAAADI== ..."
      }
  }
}
```

9. Process Object

Type Name: `process-object`

The Process Object represents common properties of an instance of a computer program as executed on an operating system.

9.1. Properties

Property Name	Type	Description

type (required)	`string`	The value of this field **MUST** be `process-object`.
is_hidden (optional)	`boolean`	Specifies whether the process is hidden.
pid (optional)	`integer`	Specifies the Process ID, or PID, of the process.
name (optional)	`string`	Specifies the name of the process.
creation_time (optional)	`timestamp`	Specifies the date/time at which the process was created.
current_working_directory (optional)	`file-path-type`	Specifies the current working directory of the process.
arguments (optional)	`list` of type `string`	Specifies the list of arguments used in executing the process. Each argument should be captured separately as a string.
environment_variables (optional)	`dictionary`	Specifies the list of environment variables associated with the process as a dictionary. Each key in the dictionary **MUST** be a case preserved version of the name of the environment variable, and each corresponding value **MUST** be the environment variable value as a string.

opened_network_connection_refs (optional)	`list` of type `object-ref`	Specifies the list of network connections opened by the process, as a reference to one or more Network Connection Objects. The objects referenced in this list **MUST** be of type `network-connection-object`.
creator_user_ref (optional)	`object-ref`	Specifies the name of the user that created the process, as a reference to a User Account Object. The object referenced in this field **MUST** be of type `user-account-object`.
binary_ref (optional)	`object-ref`	Specifies the executable binary that was executed as the process, as a reference to a File Object. The object referenced in this field **MUST** be of type `file-object`.
parent_ref (optional)	`object-ref`	Specifies the other process that spawned (i.e. is the parent of) this one, as represented by a Process Object. The object referenced in this field **MUST** be of type `process-object`.

child_refs (optional)	list of type object-ref	Specifies the other processes that were spawned by (i.e. children of) this process, as a reference to one or more other Process Objects. The objects referenced in this list **MUST** be of type process-object.
extended_properties (optional)	dictionary	Specifies any extended properties of the object, as a dictionary. Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.

Example

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes-type":{
        "md5":"B4D33B0C7306351B9ED96578465C5579"
      },
      "1":{
        "type":"process-object",
        "pid":1221,
        "name":"gedit-bin",
        "creation_time":"2016-01-20T14:11:25.55Z",
        "arguments":[
          "--new-window"
        ],
        "binary_ref":"0"
      }
    }
  }
}
```

9.2. Windows Process Extension

Type Name: `windows-process-extension`

The Windows Process extension specifies a default extension for capturing properties specific to Windows processes. The key for this extension when used in the **`extended_properties`** dictionary MUST be *windows-process*.

9.2.1. Properties

Property Name	Type	Description
`aslr_enabled` (optional)	`boolean`	Specifies whether Address Space Layout Randomization (ASLR) is enabled for the process.
`dep_enabled` (optional)	`boolean`	Specifies whether Data Execution Prevention (DEP) is enabled for the process.
`priority` (optional)	`string`	Specifies the current priority class of the process in Windows. This value **SHOULD** be a string that ends in "_CLASS".
`owner_sid` (optional)	`string`	Specifies the Security ID (SID) value of the owner of the process.
`window_title` (optional)	`string`	Specifies the title of the main window of the process.
`startup_info` (optional)	`dictionary`	Specifies the STARTUP_INFO struct used by the process, as a dictionary. Each name/value pair in the struct **MUST** be represented as a key/value pair in the dictionary. For example., given a name of 'lpDesktop' the corresponding key would be 'lpdesktop'.

9.2.2. Example

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"process-object",
      "pid":314,
      "name":"foobar.exe",
      "extended_properties":{
        "windows-process":{
          "aslr_enabled":true,
          "dep_enabled":true,
          "priority":"HIGH_PRIORITY_CLASS",
          "owner_sid":"S-1-5-21-186985262-1144665072-74031268-1309"
        }
      }
    }
  }
}
```

9.3. Windows Service Extension

Type Name: `windows-service-extension`

The Windows Service extension specifies a default extension for capturing properties specific to Windows services. The key for this extension when used in the **`extended_properties`** dictionary **MUST** be *windows-service*.

9.3.1. Properties

Property Name	Type	Description
`service_name` (required)	`string`	Specifies the name of the service.
`descriptions` (optional)	`list` of type `string`	Specifies the descriptions defined for the service.
`display_name` (optional)	`string`	Specifies the displayed name of the service in Windows GUI controls.

group_name (optional)	`string`	Specifies the name of the load ordering group of which the service is a member.
start_command_line (optional)	`string`	Specifies the full command line used to start the service.
start_type (optional)	`controlled-vocab`	Specifies the start options defined for the service. This is a controlled vocabulary and values **MUST** come from the `windows-service-start-type-cv` vocabulary.
service_dll_refs (optional)	`list` of type `object-ref`	Specifies the DLLs loaded by the service, as a reference to one or more File Objects. The objects referenced in this field **MUST** be of type `file-object`.
service_type (optional)	`controlled-vocab`	Specifies the type of the service. This is a controlled vocabulary and values **MUST** come from the `windows-service-type-cv` vocabulary.
service_status (optional)	`controlled-vocab`	Specifies the current status of the service. This is a controlled vocabulary and values **MUST** come from the `windows-service-status-cv` vocabulary.

9.3.2. Windows Service Start Type Vocabulary

Type Name: `windows-service-start-type-cv`

A controlled vocabulary of Windows service start types.

Vocabulary Value	Description
`service_auto_start`	A service started automatically by the service control manager during system startup.
`service_boot_start`	A device driver started by the system loader. This value is valid only for driver services.
`service_demand_start`	A service started by the service control manager when a process calls the StartService function.
`service_disabled`	A service that cannot be started. Attempts to start the service result in the error code ERROR_SERVICE_DISABLED.
`service_system_alert`	A device driver started by the IoInitSystem function. This value is valid only for driver services.

9.3.3. Windows Service Type Vocabulary

Type Name: `windows-service-type-cv`

A controlled vocabulary of Windows service start types.

Vocabulary Value	Description
`service_kernel_driver`	The service is a device driver.
`service_file_system_driver`	The service is a file system driver.
`service_win32_own_process`	The service runs in its own process.
`service_win32_share_process`	The service shares a process with other services.

9.3.4. Window Service Status Vocabulary

Type Name: `windows-service-status-cv`

A controlled vocabulary of Windows service statuses.

Value	Description
`service_continue_pending`	The service continue is pending.
`service_pause_pending`	The service pause is pending.

`service_paused`	The service is paused.
`service_running`	The service is running.
`service_start_pending`	The service is starting.
`service_stop_pending`	The service is stopping.
`service_stopped`	The service is not running.

Example


```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type":"file-object",
      "hashes":{
        "md5":"B4D33B0C7306351B9ED96578465C5579"
      },
      "is_directory":false,
      "file_name":"sirvizio.exe",
      "file_path":"C:\\Windows\\System32"
    },
    "1":{
      "type":"process-object",
      "pid":2217,
      "name":"sirvizio",
      "binary_ref":"0",
      "extended_properties":{
        "windows-service":{
          "display_name":"Sirvizio",
          "start_command_line":"C:\\Windows\\System32\\sirvizio.exe /s",
          "start_type":"service_auto_start",
          "service_type":"service_win32_own_process",
          "service_status":"service_running"
        }
      }
    }
  }
}
```


10. User Account Object

Type Name: `user-account-object`

The User Account Object represents an instance of any type of user account, including but not limited to operating system, device, messaging service, and social media platform accounts.

10.1. Properties

Property Name	Type	Description
type (required)	`string`	The value of this field **MUST** be `user-account-object`.
user_id (required)	`string`	Specifies the identifier of the account. The format of the identifier depends on the system the user account is maintained in, and may be a numeric ID, a GUID, an account name, an email address, etc. The **user_id** field should be populated with whatever field is the unique identifier for the system the account is a member of; as an example, on UNIX systems it would be populated with the UID, and on Windows systems it would be populated with the account SID.
account_login (optional)	`string`	Specifies the account login string, used in cases where the user_id field specifies something other than what a user would type when they login. For example, in the case of a Unix account with user_id 0, the account_login might be “root”. Similarly, in the case of a Windows account, users normally do not login with the object SID associated with their account, but might login, for example, as “Administrator”.
account_type (optional)	`open-vocab`	Specifies the type of the account. This is an open vocabulary and values **SHOULD** come from the `account-type-ov` vocabulary.
display_name (optional)	`string`	Specifies the display name of the account, to be shown in user interfaces, if applicable. On Unix, this is equivalent to the GECOS field.
is_service_account (optional)	`boolean`	Indicates that the account is associated with a network service or system process (daemon), not a specific individual.

is_privileged (optional)	`boolean`	Specifies that the account has elevated privileges (i.e., in the case of root on Unix or the Windows Administrator account).
can_escalate_privs (optional)	`boolean`	Specifies that the account has the ability to escalate privileges (i.e., in the case of sudo on Unix or a Windows Domain Admin account)
is_disabled (optional)	`boolean`	Specifies that the account is disabled.
extended_properties (optional)	`dictionary`	Specifies any extended properties of the object, as a dictionary. Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.

10.3. Account Type Vocabulary

Type Name: `account-type-ov`

An open vocabulary of User Account types.

Vocabulary Value	Description
`unix`	A POSIX account.
`windows_local`	A Windows local account.
`windows_domain`	A Windows domain account.
`ldap`	An LDAP account.
`tacacs`	A TACACS account.
`radius`	A RADIUS account.
`nis`	An NIS account
`openid`	An OpenID account.

10.4. Example

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type": "user-account-object",
      "user_id": "1001",
      "user_login": "bwayne",
      "account_type": "unix",
      "display_name": "Bruce Wayne",
      "is_service_account": false,
      "is_privileged": false,
      "can_escalate_privs": true
    }
  }
}
```

10.5. UNIX Account Extension

Type Name: `unix-account-extension`

The UNIX account extension specifies a default extension for capturing the additional information for an account on a UNIX system. The key for this extension when used in the **`extended_properties`** dictionary MUST be *unix*.

10.5.1. Properties

Property Name	Type	Description
`gid` (optional)	`number`	Specifies the primary group ID of the account.
`groups` (optional)	`list` of type `string`	Specifies a list of names of groups that the account is a member of.
`home_dir` (optional)	`file-path-type`	Specifies the home directory of the account.
`shell` (optional)	`string`	Specifies the account's command shell.

10.5.2. Example

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type": "user-account-object",
      "user_id": "1001",
      "user_login": "bwayne",
      "account_type": "unix",
      "display_name": "Bruce Wayne",
      "is_service_account": false,
      "is_privileged": false,
      "can_escalate_privs": true,
      "extended_properties":{
        "unix": {
          "gid": 1001,
          "groups": ["wheel"],
          "home_dir": "/home/bwayne",
          "shell": "/bin/bash"
          }
      }
    }
  }
}
```

10.6. Windows Account Extension

Type Name: `windows-account-extension`

The Windows account extension specifies a default extension for capturing the additional information for a user account on a Microsoft Windows(tm) system. The key for this extension when used in the **`extended_properties`** dictionary MUST be *windows*.

10.6.1. Properties

Property Name	Type	Description
`groups` (optional)	`list` of type `string`	Specifies a list of names of local system groups that the account is a member of.

10.7. Windows Active Directory Account Extension

Type Name: `windows-ad-account-extension`

The Windows Active Directory Account extension is for capturing the additional information for a Windows Active Directory account. The key for this extension when used in the **`extended_properties`** dictionary MUST be *windows-ad*.

10.7.1. Properties

Property Name	Type	Description
`object_guid` (required)	`string`	Specifies the GUID of the Active Directory account.
`groups` (optional)	`list` of type `string`	Specifies a list of names of Active Directory groups that the account is a member of.

10.8. Account Authentication Extension

Type Name: `account-auth-extension`

The account authentication extension specifies a default extension for capturing the authentication information related to an account. The key for this extension when used in the **`extended_properties`** dictionary MUST be *auth*.

10.8.1. Properties

Property Name	Type	Description
`account_created` (optional)	`timestamp`	Specifies when the account was created.
`account_expires` (optional)	`timestamp`	Specifies the expiration date of the account.
`password_last_changed` (optional)	`timestamp`	Specifies when the account password was last changed.
`account_first_login` (optional)	`timestamp`	Specifies when the account was first accessed.

account_last_login (optional)	`timestamp`	Specifies when the account was last accessed.

10.8.2. Example

```
{
  "type":"cybox-container",
  "spec_version":"3.0",
  "objects":{
    "0":{
      "type": "user-account-object",
      "user_id": "1001",
      "user_login": "bwayne",
      "account_type": "unix",
      "display_name": "Bruce Wayne",
      "is_service_account": false,
      "is_privileged": false,
      "can_escalate_privs": true,
      "extended_properties":{
        "auth": {
          "account_created": "2016-01-20T12:31:12Z",
          "password_last_changed": "2016-01-20T14:27:43Z",
          "account_first_login": "2016-01-20T14:26:07Z",
          "account_last_login": "2016-07-22T16:08:28Z"
          }
        }
    }
  }
}
```

11. Custom Object

Type Name: `custom-object`

The Custom Object is not a first class CybOX object. It provides a template for those wishing to create their own CybOX objects and provides normative text to maximize the likelihood of interoperability. Custom CybOX objects defined in conformance with the CybOX Custom Object specification may be freely used but the CybOX Custom Object itself **MUST NOT** be used as-is. It is a template.

11.1. Properties

Property Name	Type	Description

type (required)	`string`	Indicates the type of the Custom CybOX Object. **MUST** be a lowercase string that represents the name of the object. If the custom object name contains multiple words, they **SHOULD** be specified with additional hyphens. The **type** name of a Custom CybOX Object **MUST NOT** collide with the name of a CybOX Object already defined in the CybOX specification. The **type** name of a Custom CybOX Object **SHOULD** be prefixed with `x_`.
custom_object_field (required)	`custom object field type`	A Custom CybOX Object **MUST** contain one or more fields. Custom CybOX Object field names **MUST** conform to the key naming restrictions stipulated by the CybOX `dictionary` primitive in the CybOX Core specification. Custom CybOX Object field values **MUST** be a valid CybOX primitive, type, or a homogenous list of types.