CybOX Version 2.1.1 Part 89: Win Task Object

Working Draft 01

28 September 2015

Technical Committee:

[OASIS Cyber Threat Intelligence (CTI) TC](https://www.oasis-open.org/committees/cti)

Chair:

Richard Struse ([Richard.Struse@HQ.DHS.GOV](mailto:Richard.Struse@HQ.DHS.GOV)), [DHS Office of Cybersecurity and Communications (CS&C)](http://www.dhs.gov/office-cybersecurity-and-communications)

Editors:

Desiree Beck ([dbeck@mitre.org](mailto:ikirillov@mitre.org)), [MITRE Corporation](http://www.mitre.org/)

Trey Darley ([trey@soltra.com](mailto:trey@soltra.com)), [Soltra](http://www.soltra.com/)

Ivan Kirillov ([ikirillov@mitre.org](mailto:ikirillov@mitre.org)), [MITRE Corporation](http://www.mitre.org/)

Rich Piazza ([rpiazza@mitre.org](mailto:ikirillov@mitre.org)), [MITRE Corporation](http://www.mitre.org/)

Additional artifacts:

This prose specification is one component of a Work Product which consists of:

*CybOX Version 2.1.1 Part 1: Overview*. [URI]

*CybOX Version 2.1.1 Part 2: Common*. [URI]

*CybOX Version 2.1.1 Part 3: Core*. [URI]

*CybOX Version 2.1.1 Part 4: Default Extensions*. [URI]

*CybOX Version 2.1.1 Part 5: Default Vocabularies*. [URI]

*CybOX Version 2.1.1 Part 6: UML Model*. [URI]

*CybOX Version 2.1.1 Part 7: API Object*. [URI]

*CybOX Version 2.1.1 Part 8: ARP Cache Object*. [URI]

*CybOX Version 2.1.1 Part 9: AS Object*. [URI]

*CybOX Version 2.1.1 Part 10: Account Object*. [URI]

*CybOX Version 2.1.1 Part 11: Address Object*. [URI]

*CybOX Version 2.1.1 Part 12: Archive File Object*. [URI]

*CybOX Version 2.1.1 Part 13: Artifact Object*. [URI]

*CybOX Version 2.1.1 Part 14: Code Object*. [URI]

*CybOX Version 2.1.1 Part 15: Custom Object*. [URI]

*CybOX Version 2.1.1 Part 16: DNS Cache Object*. [URI]

*CybOX Version 2.1.1 Part 17: DNS Query Object*. [URI]

*CybOX Version 2.1.1 Part 18: DNS Record Object*. [URI]

*CybOX Version 2.1.1 Part 19: Device Object*. [URI]

*CybOX Version 2.1.1 Part 20: Disk Object*. [URI]

*CybOX Version 2.1.1 Part 21: Disk Partition Object*. [URI]

*CybOX Version 2.1.1 Part 22: Domain Name Object*. [URI]

*CybOX Version 2.1.1 Part 23: Email Message Object*. [URI]

*CybOX Version 2.1.1 Part 24: File Object*. [URI]

*CybOX Version 2.1.1 Part 25: GUI Dialogbox Object*. [URI]

*CybOX Version 2.1.1 Part 26: GUI Object*. [URI]

*CybOX Version 2.1.1 Part 27: GUI Window Object*. [URI]

*CybOX Version 2.1.1 Part 28: HTTP Session Object*. [URI]

*CybOX Version 2.1.1 Part 29: Hostname Session Object*. [URI]

*CybOX Version 2.1.1 Part 30: Image File Object*. [URI]

*CybOX Version 2.1.1 Part 31: Library File Object*. [URI]

*CybOX Version 2.1.1 Part 32: Link Object*. [URI]

*CybOX Version 2.1.1 Part 33: Linux Package Object*. [URI]

*CybOX Version 2.1.1 Part 34: Memory Object*. [URI]

*CybOX Version 2.1.1 Part 35: Mutex Object*. [URI]

*CybOX Version 2.1.1 Part 36: Network Connection Object*. [URI]

*CybOX Version 2.1.1 Part 37: Network Flow Object*. [URI]

*CybOX Version 2.1.1 Part 38: Network Packet Object*. [URI]

*CybOX Version 2.1.1 Part 39: Network Route Entry Object*. [URI]

*CybOX Version 2.1.1 Part 40: Network Route Object*. [URI]

*CybOX Version 2.1.1 Part 41: Network Socket Object*. [URI]

*CybOX Version 2.1.1 Part 42: Network Subnet Object*. [URI]

*CybOX Version 2.1.1 Part 43: PDF File Object*. [URI]

*CybOX Version 2.1.1 Part 44: Pipe Object*. [URI]

*CybOX Version 2.1.1 Part 45: Port Object*. [URI]

*CybOX Version 2.1.1 Part 46: Process Object*. [URI]

*CybOX Version 2.1.1 Part 47: Product Object*. [URI]

*CybOX Version 2.1.1 Part 48: SMS Message Object*. [URI]

*CybOX Version 2.1.1 Part 49: Semaphore Object*. [URI]

*CybOX Version 2.1.1 Part 50: Socket Address Object*. [URI]

*CybOX Version 2.1.1 Part 51: System Object*. [URI]

*CybOX Version 2.1.1 Part 52: URI Object*. [URI]

*CybOX Version 2.1.1 Part 53: URL History Object*. [URI]

*CybOX Version 2.1.1 Part 54: Unix File Object*. [URI]

*CybOX Version 2.1.1 Part 55: Unix Network Route Entry Object*. [URI]

*CybOX Version 2.1.1 Part 56: Unix Pipe Object*. [URI]

*CybOX Version 2.1.1 Part 57: Unix Process Object*. [URI]

*CybOX Version 2.1.1 Part 58: Unix User Account Object*. [URI]

*CybOX Version 2.1.1 Part 59: Unix Volume Object*. [URI]

*CybOX Version 2.1.1 Part 60: Unix Account Object*. [URI]

*CybOX Version 2.1.1 Part 61: User Session Object*. [URI]

*CybOX Version 2.1.1 Part 62: Volume Object*. [URI]

*CybOX Version 2.1.1 Part 63: Whois Object*. [URI]

*CybOX Version 2.1.1 Part 64: Win Computer Account Object*. [URI]

*CybOX Version 2.1.1 Part 65: Win Critical Section Object*. [URI]

*CybOX Version 2.1.1 Part 66: Win Driver Object*. [URI]

*CybOX Version 2.1.1 Part 67: Win Event Log Object*. [URI]

*CybOX Version 2.1.1 Part 68: Win Event Object*. [URI]

*CybOX Version 2.1.1 Part 69: Win Executable File Object*. [URI]

*CybOX Version 2.1.1 Part 70: Win File Object*. [URI]

*CybOX Version 2.1.1 Part 71: Win Filemapping Object*. [URI]

*CybOX Version 2.1.1 Part 72: Win Handle Object*. [URI]

*CybOX Version 2.1.1 Part 73: Win Hook Object*. [URI]

*CybOX Version 2.1.1 Part 74: Win Kernel Hook Object*. [URI]

*CybOX Version 2.1.1 Part 75: Win Kernel Object*. [URI]

*CybOX Version 2.1.1 Part 76: Win Mailslot Object*. [URI]

*CybOX Version 2.1.1 Part 77: Win Memory Page Region Object*. [URI]

*CybOX Version 2.1.1 Part 78: Win Mutex Object*. [URI]

*CybOX Version 2.1.1 Part 79: Win Network Route Entry Object*. [URI]

*CybOX Version 2.1.1 Part 80: Win Network Share Object*. [URI]

*CybOX Version 2.1.1 Part 81: Win Pipe Object*. [URI]

*CybOX Version 2.1.1 Part 82: Win Prefetch Object*. [URI]

*CybOX Version 2.1.1 Part 83: Win Process Object*. [URI]

*CybOX Version 2.1.1 Part 84: Win Registry Key Object*. [URI]

*CybOX Version 2.1.1 Part 85: Win Semaphore Object*. [URI]

*CybOX Version 2.1.1 Part 86: Win Service Object*. [URI]

*CybOX Version 2.1.1 Part 87: Win System Object*. [URI]

*CybOX Version 2.1.1 Part 88: Win System Restore Object*. [URI]

*CybOX Version 2.1.1 Part 89: Win Task Object*. [URI]

*CybOX Version 2.1.1 Part 90: Win Thread Object*. [URI]

*CybOX Version 2.1.1 Part 91: Win User Account Object*. [URI]

*CybOX Version 2.1.1 Part 92: Win Volume Object*. [URI]

*CybOX Version 2.1.1 Part 93: Win Waitable Timer Object*. [URI]

*CybOX Version 2.1.1 Part 94: X509 Certificate Object*. [URI]

Related work:

This specification is related to:

*STIX Version 1.2.1 (placeholder)*

Abstract:

The Cyber Observable Expression (CybOX) is a standardized language for encoding and communicating high-fidelity information about cyber observables, whether dynamic events or stateful measures that are observable in the operational cyber domain. By specifying a common structured schematic mechanism for these cyber observables, the intent is to enable the potential for detailed automatable sharing, mapping, detection and analysis heuristics. This specification document defines the Win Task Object data model, which is one of the Object data models for CybOX content.

Status:

This [Working Draft](https://www.oasis-open.org/policies-guidelines/tc-process#dWorkingDraft) (WD) has been produced by one or more TC Members; it has not yet been voted on by the TC or [approved](https://www.oasis-open.org/policies-guidelines/tc-process#committeeDraft) as a Committee Draft (Committee Specification Draft or a Committee Note Draft). The OASIS document [Approval Process](https://www.oasis-open.org/policies-guidelines/tc-process#standApprovProcess) begins officially with a TC vote to approve a WD as a Committee Draft. A TC may approve a Working Draft, revise it, and re-approve it any number of times as a Committee Draft.

URI patterns:

Initial publication URI:  
http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part1-overview/stix-v1.2.1-csd01-part1-overview.docx

Permanent “Latest version” URI:  
http://docs.oasis-open.org/cti/stix/v1.2.1/stix-v1.2.1-part1-overview.docx

(Managed by OASIS TC Administration; please don’t modify.)

Copyright © OASIS Open 2015. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](https://www.oasis-open.org/policies-guidelines/ipr) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

# Introduction

The Cyber Observable Expression (CybOX) provides a common structure for representing cyber observables across and among the operational areas of enterprise cyber security. CybOX improves the consistency, efficiency, and interoperability of deployed tools and processes, and it increases overall situational awareness by enabling the potential for detailed automatable sharing, mapping, detection, and analysis heuristics.

This document serves as the specification for the CybOX Win Task Object Version 2.1.1 data model, which is one of ninety-four Object data models for CybOX content.

In Section **1.1** we discuss additional specification documents, in Section **1.2** we provide document conventions in, and in Section **1.3** we provide terminology. References are given in Sections **1.4** and **1.5**. In Section **2**, we give background information necessary to fully understand the Win Task Object data model. We present the Win Task Object data model specification details in Section **3** and conformance information in Section **4**.

## CybOX Specification Documents

The CybOX specification consists of a formal UML model and a set of textual specification documents that explain the UML model. Specification documents have been written for each of the key individual data models that compose the full CybOX UML model.

CybOX has a modular design comprising two fundamental data models and a collection of Object data models. The fundamental data models – CybOX Core and CybOX Common – provide essential CybOX structure and functionality. The CybOX Objects, defined in individual data models, are precise characterizations of particular types of observable cyber entities (e.g., HTTP session, Windows registry key, DNS query).

Use of the CybOX Core and Common data models is required; however, use of the CybOX Object data models is purely optional: users select and use only those Objects and corresponding data models that are needed. Importing the entire CybOX suite of data models is not necessary.

The [*CybOX Version 2.1.1 Part 1: Overview*](#AdditionalArtifacts) document provides a comprehensive overview of the full set of CybOX data models, which in addition to the Core, Common, and numerous Object data models, includes a set of default controlled vocabularies. [*CybOX Version 2.1.1 Part 1: Overview*](#AdditionalArtifacts) also summarizes the relationship of CybOX to other languages, and outlines general CybOX data model conventions.

## Document Conventions

The following conventions are used in this document.

### Fonts

The following font and font style conventions are used in the document:

* Capitalization is used for CybOX high level concepts, which are defined in [*CybOX Version 2.1.1 Part 1: Overview*](#AdditionalArtifacts).
  + Examples: Action, Object, Event, Property
* The Courier New font is used for writing UML objects.
  + Examples: ActionType, cyboxCommon:BaseObjectPropertyType
  + Note that all high level concepts have a corresponding UML object. For example, the Action high level concept is associated with a UML class named, ActionType.
* The ‘*italic’* font (withsingle quotes) is used for noting actual, explicit values for CybOX Language properties. The *italic* font (without quotes) is used for noting example values.
  + - Example:  *‘HashNameVocab-1.0,’ high, medium, low*

### UML Package References

Each CybOX data model is captured in a different UML package (e.g., Core package) where the packages together compose the full CybOX UML model. To refer to a particular class of a specific package, we use the format package\_prefix:class, where package\_prefix corresponds to the appropriate UML package. The [*CybOX Version 2.1.1 Part 1: Overview*](#AdditionalArtifacts) document contains the full list of CybOX packages, along with the associated prefix notations, descriptions, and examples.

Note that in this specification document, we do not explicitly specify the package prefix for any classes that originate from the Win Task Object data model.

### UML Diagrams

This specification makes use of UML diagrams to visually depict relationships between CybOX Language constructs. Note that the diagrams have been extracted directly from the full UML model for CybOX; they have not been constructed purely for inclusion in the specification documents.  Typically, diagrams are included for the primary class of a data model, and for any other class where the visualization of its relationships between other classes would be useful.  This implies that there will be very few diagrams for classes whose only properties are either a data type or a class from the CybOX Common data model.  Other diagrams that are included correspond to classes that specialize a superclass and abstract or generalized classes that are extended by one or more subclasses.

In UML diagrams, classes are often presented with their attributes elided, to avoid clutter. The fully described class can usually be found in a related diagram. A class presented with an empty section at the bottom of the icon indicates that there are no attributes other than those that are visualized using associations.

#### Class Properties

Generally, a class property can be shown in a UML diagram as either an attribute or an association (i.e., the distinction between attributes and associations is somewhat subjective). In order to make the size of UML diagrams in the specifications manageable, we have chosen to capture most properties as attributes and to capture only higher level properties as associations, especially in the main top-level component diagrams. In particular, we will always capture properties of UML data types as attributes. For example, properties of a class that are identifiers, titles, and timestamps will be represented as attributes.

#### Diagram Icons and Arrow Types

Diagram icons are used in a UML diagram to indicate whether a shape is a class, enumeration, or a data type, and decorative icons are used to indicate whether an element is an attribute of a class or an enumeration literal. In addition, two different arrow styles indicate either a directed association relationship (regular arrowhead) or a generalization relationship (triangle-shaped arrowhead). The icons and arrow styles we use are shown and described in **Table 1‑1**

Table 1-1 needs to be c & p here

#### Color Coding

The shapes of the UML diagrams are color coded to indicate the data model associated with a class.  The colors used in the Win Task Object specification are illustrated via exemplars in **Figure 1‑1**.

[need diagram]

Figure 1‑1. Data model color coding

### Property Table Notation

Throughout Section 3, tables are used to describe the properties of each data model class. Each property table consists of a column of names to identify the property, a type column to reflect the datatype of the property, a multiplicity column to reflect the allowed number of occurrences of the property, and a description column that describes the property. Package prefixes are provided for classes outside of the Win Task Object data model (see Section 1.2.3).

Note that if a class is a specialization of a superclass, only the properties that constitute the specialization are shown in the property table (i.e., properties of the superclass will not be shown). However, details of the superclass may be shown in the UML diagram.

### Property and Class Descriptions

Each class and property defined in CybOX is described using the format, “The X property verbY.” For example, in the specification for the CybOX Core data model, we write, “The id property specifies a globally unique identifier for the Action.” In fact, the verb “specifies” could have been replaced by any number of alternatives: “defines,” “describes,” “contains,” “references,” etc.

However, we thought that using a wide variety of verb phrases might confuse a reader of a specification document because the meaning of each verb could be interpreted slightly differently. On the other hand, we didn’t want to use a single, generic verb, such as “describes,” because although the different verb choices may or may not be meaningful from an implementation standpoint, a distinction could be useful to those interested in the modeling aspect of CybOX.

Consequently, we have chosen to use the three verbs, defined as follows, in class and property descriptions:

|  |  |
| --- | --- |
| **Verb** | **CybOX Definition** |
| captures | Used to record and preserve information without implying anything about the structure of a class or property. Often used for properties that encompass general content. This is the least precise of the three verbs. |
|  | *Examples*:  The Observable\_Source property characterizes the source of the Observable information. Examples of details captured include identitifying characteristics, time-related attributes, and a list of the tools used to collect the information.  The Description property captures a textual description of the Action. |
| characterizes | Describes the distinctive nature or features of a class or property. Often used to describe classes and properties that themselves comprise one or more other properties. |
|  | *Examples*:  The Action property characterizes a cyber observable Action.  The Obfuscation\_Technique property characterizes a technique an attacker could potentially leverage to obfuscate the Observable. |
| specifies | Used to clearly and precisely identify particular instances or values associated with a property. Often used for properties that are defined by a controlled vocabulary or enumeration; typically used for properties that take on only a single value. |
|  | *Example*:  The cybox\_major\_version property specifies the major version of the CybOX language used for the set of Observables. |

## Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

## Normative References

[RFC2119] Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels”, BCP 14, RFC 2119, March 1997. <http://www.ietf.org/rfc/rfc2119.txt>.

## Non-Normative References

# Background Information

In this section, we provide high level information about the Win Task Object data model that is necessary to fully understand the specification details given in Section 3.

## Cyber Observables

A cyber observable is a dynamic event or a stateful property that occurs, or may occur, in the operational cyber domain. Examples of stateful properties include the value of a registry key, the MD5 hash of a file, and an IP address. Examples of events include the deletion of a file, the receipt of an HTTP GET request, and the creation of a remote thread.

A cyber observable is different than a cyber indicator. A cyber observable is a statement of fact, capturing what was observed or could be observed in the cyber operational domain. Cyber indicators are cyber observable patterns, such as a registry key value associated with a known bad actor or a spoofed email address used on a particular date.

## Objects

Objects in CybOX are individual data models for characterizing a particular cyber entity, such as a Windows registry key, or an Email Message. Accordingly, each release of the CybOX language includes a particular set of Objects that are part of the release. The data model for each of these Objects is defined by its own specification that describes the context-specific classes and properties that compose the Object.

# Data Model

## WindowsTaskObjectType Class

The WindowsTaskObjectType class is intended to characterize Windows task scheduler tasks. See Also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381311(v=vs.85).aspx.

The property table of the WindowsTaskObjectType class is given in ???.

Table 3‑1. Properties of the WindowsTaskObjectType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Status** | WinTaskObj:TaskStatusType | 0..1 | The Status property specifies the current status of the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381263(v=vs.85).aspx. |
| **Priority** | WinTaskObj:TaskPriorityType | 0..1 | The Priority property specifies the priority of the scheduled task. This can either be a free-form string or one the values in the TaskPriorityEnum enumeration. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381876(v=vs.85).aspx. |
| **Name** | cyboxCommon:StringObjectPropertyType | 0..1 | The Name property specifies the image name for the task. |
| **Application\_Name** | cyboxCommon:StringObjectPropertyType | 0..1 | The Application\_Name property The Application\_Name specifies the application name associated with the task. |
| **Parameters** | cyboxCommon:StringObjectPropertyType | 0..1 | The Parameters property specifies the command line parameters used to launch the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381875(v=vs.85).aspx. |
| **Flags** | WinTaskObj:TaskFlagType | 0..1 | The Flags property specifies any flags that modify the behavior of the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381248(v=vs.85).aspx. |
| **Account\_Name** | cyboxCommon:StringObjectPropertyType | 0..1 | The Account\_Name property specifies the name of the account used to run the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381228(v=vs.85).aspx. |
| **Account\_Run\_Level** | cyboxCommon:StringObjectPropertyType | 0..1 | The Account\_Run\_Level property specifies the permission level of the account that the task will be run at. |
| **Account\_Logon\_Type** | cyboxCommon:StringObjectPropertyType | 0..1 | The Account\_Logon\_Type property specifies the security logon method required to run the tasks associated with the account. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa383013(v=vs.85).aspx. |
| **Creator** | cyboxCommon:StringObjectPropertyType | 0..1 | The Creator property specifies the name of the creator of the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381235(v=vs.85).aspx. |
| **Creation\_Date** | cyboxCommon:DateTimeObjectPropertyType | 0..1 | The Creation\_Date property specifies the date and time that the task was registered. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa382623(v=vs.85).aspx. |
| **Most\_Recent\_Run\_Time** | cyboxCommon:DateTimeObjectPropertyType | 0..1 | The Most\_Recent\_Run\_Time property specifies the most recent run date/time of this scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381254(v=vs.85).aspx. |
| **Exit\_Code** | cyboxCommon:LongObjectPropertyType | 0..1 | The Exit\_Code property specifies the last exit code of the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381245(v=vs.85).aspx. |
| **Max\_Run\_Time** | cyboxCommon:UnsignedLongObjectPropertyType | 0..1 | The Max\_Run\_Time property specifies the maximum run time of the scheduled task before terminating, in milliseconds. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381874(v=vs.85).aspx. |
| **Next\_Run\_Time** | cyboxCommon:DateTimeObjectPropertyType | 0..1 | The Next\_Run\_Time property specifies the next run date/time of the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381257(v=vs.85).aspx. |
| **Action\_List** | WinTaskObj:TaskActionListType | 0..1 | The Action\_List property specifies a list of actions to be performed by the scheduled task. |
| **Trigger\_List** | WinTaskObj:TriggerListType | 0..1 | The Trigger\_List property specifies a set of triggers used by the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa383264(v=vs.85).aspx. |
| **Comment** | cyboxCommon:StringObjectPropertyType | 0..1 | The Comment property specifies a comment for the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381232(v=vs.85).aspx. |
| **Working\_Directory** | cyboxCommon:StringObjectPropertyType | 0..1 | The Working\_Directory property specifies the working directory for the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381878(v=vs.85).aspx. |
| **Work\_Item\_Data** | cyboxCommon:Base64BinaryObjectPropertyType | 0..1 | The Work\_Item\_Data property specifies application defined data associated with the scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381271(v=vs.85).aspx. |

## TriggerListType Class

The TriggerListType class specifies a set of triggers associated with the scheduled task.

The property table of the TriggerListType class is given in ???.

Table 3‑2. Properties of the TriggerListType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Trigger** | WinTaskObj:TriggerType | 1..\* | The Trigger property A trigger associated with this scheduled task. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381264(v=vs.85).aspx. |

## TriggerType Class

The TriggerType class characterizes task triggers. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa383868(v=vs.85).aspx.

The property table of the TriggerType class is given in ???.

Table 3‑3. Properties of the TriggerType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **enabled** | basicDataTypes:Boolean | 0..1 | The enabled property specifies whether the trigger is enabled. |
| **Trigger\_Begin** | cyboxCommon:DateTimeObjectPropertyType | 0..1 | The Trigger\_Begin property The Trigger\_Begin\_Element specifies the date/time that the trigger is activated. |
| **Trigger\_Delay** | cyboxCommon:DurationObjectPropertyType | 0..1 | The Trigger\_Delay property specifies the delay that takes place between when the task is registered and when the task is started. |
| **Trigger\_End** | cyboxCommon:DateTimeObjectPropertyType | 0..1 | The Trigger\_End property specifies the date/time that the trigger is deactivated. |
| **Trigger\_Frequency** | WinTaskObj:TaskTriggerFrequencyType | 0..1 | The Trigger\_Frequency property specifies the frequency at which the trigger repeats. |
| **Trigger\_Max\_Run\_Time** | cyboxCommon:DurationObjectPropertyType | 0..1 | The Trigger\_Max\_Run\_Time property The maximum amount of time that the task launched by the trigger is allowed to run. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa383868(v=vs.85).aspx. |
| **Trigger\_Session\_Change\_Type** | cyboxCommon:StringObjectPropertyType | 0..1 | The Trigger\_Session\_Change\_Type property specifies the type of Terminal Server session change that would trigger a task launch. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381298(v=vs.85).aspx. |
| **Trigger\_Type** | WinTaskObj:TaskTriggerType | 0..1 | The Trigger\_Type property The Trigger\_Type specifies the type of the task trigger. |

## TaskActionListType Class

The TaskActionListType class specifies a list of task actions.

The property table of the TaskActionListType class is given in ???.

Table 3‑4. Properties of the TaskActionListType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Action** | WinTaskObj:TaskActionType | 1..\* | The Action property The work items performed by a task are called actions. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa383549(v=vs.85).aspx. |

## TaskActionType Class

The TaskActionType class characterizes scheduled task actions.

The property table of the TaskActionType class is given in ???.

Table 3‑5. Properties of the TaskActionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Action\_Type** | WinTaskObj:TaskActionTypeType | 0..1 | The Action\_Type property specifies the type of the action. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380596(v=vs.85).aspx. |
| **Action\_ID** | cyboxCommon:StringObjectPropertyType | 0..1 | The Action\_ID property specifies the user-defined identifier for the action. This identifier is used by the Task Scheduler for logging purposes. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380590(v=vs.85).aspx. |
| **IEmailAction** | EmailMessageObj:EmailMessageObjectType | 0..1 | The IEmail\_Action property specifies an action that sends an e-mail, which in this context refers to actual email message sent. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380693(v=vs.85).aspx. |
| **IComHandlerAction** | WinTaskObj:IComHandlerActionType | 0..1 | The IComHandlerAction property specifies an action that fires a handler. |
| **IExecAction** | WinTaskObj:IExecActionType | 0..1 | The IExecAction property specifies an action that executes a command-line operation. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380715(v=vs.85).aspx. |
| **IShowMessageAction** | WinTaskObj:IShowMessageActionType | 0..1 | The IShowMessageAction property specifies an action that shows a message box when a task is activated. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381302(v=vs.85).aspx. |

## TaskActionTypeType Class

The TaskActionTypeType characterizes the specific classs of task actions.

## IComHandlerActionType Class

The IComHandlerActionType class characterizes IComHandler actions.

The property table of the IComHandlerActionType class is given in ???.

Table 3‑6. Properties of the IComHandlerActionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **COM\_Data** | cyboxCommon:StringObjectPropertyType | 0..1 | The COM\_Data property specifies the data associated with the COM handler. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380613(v=vs.85).aspx. |
| **COM\_Class\_ID** | cyboxCommon:StringObjectPropertyType | 0..1 | The COM\_Class\_ID property specifies the ID of the COM action. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380613(v=vs.85).aspx. |

## IExecActionType Class

The IExecActionType class characterizes IExec actions.

The property table of the IExecActionType class is given in ???.

Table 3‑7. Properties of the IExecActionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Exec\_Arguments** | cyboxCommon:StringObjectPropertyType | 0..1 | The Exec\_Arguments property specifies the arguments associated with the command-line operation launched by the action. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380715(v=vs.85).aspx. |
| **Exec\_Program\_Path** | cyboxCommon:StringObjectPropertyType | 0..1 | The Exec\_Program\_Path property specifies the path to the executable file launched by the action. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380715(v=vs.85).aspx. |
| **Exec\_Working\_Directory** | cyboxCommon:StringObjectPropertyType | 0..1 | The Exec\_Working\_Directory property specifies the directory that contains either the executable file or the files that are used by the executable file launched by the action. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa380715(v=vs.85).aspx. |
| **Exec\_Program\_Hashes** | cyboxCommon:HashListType | 0..1 | The Exec\_Program\_Hashes property The Exec\_Program\_Element specifies the hashes of the executable file launched by the action. |

## IShowMessageActionType Class

The IShowMessageActionType class characterizes IShowMessage actions.

The property table of the IShowMessageActionType class is given in ???.

Table 3‑8. Properties of the IShowMessageActionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Show\_Message\_Body** | cyboxCommon:StringObjectPropertyType | 0..1 | The Show\_Message\_Body property specifies the message text that is displayed in the body of the message box by the action. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381302(v=vs.85).aspx. |
| **Show\_Message\_Title** | cyboxCommon:StringObjectPropertyType | 0..1 | The Show\_Message\_Title property specifies the title of the message box shown by the action. See also: http://msdn.microsoft.com/en-us/library/windows/desktop/aa381302(v=vs.85).aspx. |

## TaskFlagType Class

The TaskFlagType class specifies Windows Task flag types via a union of the TaskFlagEnum type and the atomic xs:string type. Its base type is the CybOX Core BaseObjectPropertyType, for permitting complex (i.e. regular-expression based) specifications.

## TaskPriorityType Class

The TaskPriorityType class specifies Windows Task priority types via a union of the TaskPriorityEnum type and the atomic xs:string type. Its base type is the CybOX Core BaseObjectPropertyType, for permitting complex (i.e. regular-expression based) specifications.

## TaskTriggerFrequencyType Class

The TaskTriggerFrequencyType class specifies Windows Task trigger frequency types via a union of the TriggerFrequencyEnum type and the atomic xs:string type. Its base type is the CybOX Core BaseObjectPropertyType, for permitting complex (i.e. regular-expression based) specifications.

## TaskTriggerType Class

The TaskTriggerType class specifies Windows Task trigger types via a union of the TriggerTypeEnum enumeration and the atomic xs:string type. Its base type is the CybOX Core BaseObjectPropertyType, for permitting complex (i.e. regular-expression based) specifications.

## TaskStatusType Class

The TaskStatusType class specifies Windows Task states via a union of the TaskStatusEnum type and the atomic xs:string type. Its base type is the CybOX Core BaseObjectPropertyType, for permitting complex (i.e. regular-expression based) specifications.

## TaskActionTypeEnum Enumeration

The literals of the TaskActionTypeEnum enumeration are given in ???.

Table 3‑9. Literals of the TaskActionTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **TASK\_ACTION\_EXEC** | This action performs a command-line operation. For example, the action could run a script, launch an executable, or, if the name of a document is provided, find its associated application and launch the application with the document. |
| **TASK\_ACTION\_COM\_HANDLER** | This action fires a handler. |
| **TASK\_ACTION\_SEND\_EMAIL** | This action sends an e-mail. |
| **TASK\_ACTION\_SHOW\_MESSAGE** | This action shows a message box. |

## TaskPriorityEnum Enumeration

The literals of the TaskPriorityEnum enumeration are given in ???.

Table 3‑10. Literals of the TaskPriorityEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **HIGH\_PRIORITY\_CLASS** | A priority class of high (1). |
| **NORMAL\_PRIORITY\_CLASS** | A priority class of normal (4-6). |
| **IDLE\_PRIORITY\_CLASS** | A priority class of idle (9-10). |
| **REALTIME\_PRIORITY\_CLASS** | A priority class of realtime (0). |
| **ABOVE\_NORMAL\_PRIORITY\_CLASS** | A priority class of above normal (2-3). |
| **BELOW\_NORMAL\_PRIORITY\_CLASS** | A priority class of below normal (7-8). |

## TriggerFrequencyEnum Enumeration

The literals of the TriggerFrequencyEnum enumeration are given in ???.

Table 3‑11. Literals of the TriggerFrequencyEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **TASK\_TIME\_TRIGGER\_ONCE** | Trigger is set to run the task a single time. |
| **TASK\_EVENT\_TRIGGER\_ON\_IDLE** | Trigger is set to run the task if the system remains idle for the amount of time specified by the idle wait time of the task. |
| **TASK\_EVENT\_TRIGGER\_AT\_SYSTEMSTART** | Trigger is set to run the task at system startup. |
| **TASK\_EVENT\_TRIGGER\_AT\_LOGON** | Trigger is set to run the task when a user logs on. |
| **TASK\_TIME\_TRIGGER\_DAILY** | Trigger is set to run the task on a daily interval. |
| **TASK\_TIME\_TRIGGER\_WEEKLY** | Trigger is set to run the work item on specific days of a specific week of a specific month. |
| **TASK\_TIME\_TRIGGER\_MONTHLYDATE** | Trigger is set to run the task on a specific day(s) of the month. |
| **TASK\_TIME\_TRIGGER\_MONTHLYDOW** | Trigger is set to run the task on specific days, weeks, and months. |

## TriggerTypeEnum Enumeration

The literals of the TriggerTypeEnum enumeration are given in ???.

Table 3‑12. Literals of the TriggerTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **TASK\_TRIGGER\_EVENT** | Triggers the task when a specific system event occurs. |
| **TASK\_TRIGGER\_TIME** | Triggers the task at a specific date and time. |
| **TASK\_TRIGGER\_IDLE** | Triggers the task when the computer enters an idle state. |
| **TASK\_TRIGGER\_REGISTRATION** | Triggers the task when the task is registered or updated. |
| **TASK\_TRIGGER\_BOOT** | Triggers the task when the system is booted. |
| **TASK\_TRIGGER\_LOGON** | Triggers the task when a user logs on. |
| **TASK\_TRIGGER\_SESSION\_STATE\_CHANGE** | Triggers the task when a Terminal Server session changes state. |

## TaskStatusEnum Enumeration

The literals of the TaskStatusEnum enumeration are given in ???.

Table 3‑13. Literals of the TaskStatusEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **SCHED\_S\_TASK\_READY** | The task is ready to run at its next scheduled time. |
| **SCHED\_S\_TASK\_RUNNING** | The task is currently running. |
| **SCHED\_S\_TASK\_NOT\_SCHEDULED** | One or more of the properties that are needed to run this task on a schedule have not been set. |
| **SCHED\_E\_SERVICE\_NOT\_RUNNING** | The Task Scheduler service is not running. |
| **SCHED\_E\_UNSUPPORTED\_ACCOUNT\_OPTION** | The task has been configured with an unsupported combination of account settings and run time options. |
| **SCHED\_E\_UNKNOWN\_OBJECT\_VERSION** | The task object version is either unsupported or invalid. |
| **SCHED\_E\_NO\_SECURITY\_SERVICES** | Task Scheduler security services are available only on Windows NT. |
| **SCHED\_E\_ACCOUNT\_DBASE\_CORRUPT** | Corruption was detected in the Task Scheduler security database; the database has been reset. |
| **SCHED\_E\_ACCOUNT\_NAME\_NOT\_FOUND** | Unable to establish existence of the account specified. |
| **SCHED\_E\_ACCOUNT\_INFORMATION\_NOT\_SET** | No account information could be found in the Task Scheduler security database for the task indicated. |
| **SCHED\_E\_INVALID\_TASK** | The object is either an invalid task object or is not a task object. |
| **SCHED\_E\_CANNOT\_OPEN\_TASK** | The task object could not be opened. |
| **SCHED\_E\_SERVICE\_NOT\_INSTALLED** | The Task Scheduler service is not installed on this computer. |
| **SCHED\_E\_TASK\_NOT\_RUNNING** | There is no running instance of the task. |
| **SCHED\_E\_TASK\_NOT\_READY** | One or more of the properties required to run this task have not been set. |
| **SCHED\_E\_TRIGGER\_NOT\_FOUND** | A task's trigger is not found. |
| **SCHED\_S\_EVENT\_TRIGGER** | Event triggers do not have set run times. |
| **SCHED\_S\_TASK\_NO\_VALID\_TRIGGERS** | Either the task has no triggers or the existing triggers are disabled or not set. |
| **SCHED\_S\_TASK\_TERMINATED** | The last run of the task was terminated by the user. |
| **SCHED\_S\_TASK\_NO\_MORE\_RUNS** | There are no more runs scheduled for this task. |
| **SCHED\_S\_TASK\_HAS\_NOT\_RUN** | The task has not been run. This value is returned whenever the task has not been run, even if the task is ready to be run at the next scheduled time or the task is a recurring task. |
| **SCHED\_S\_TASK\_DISABLED** | The task will not run at the scheduled times because it has been disabled. |
| **TASK\_STATE\_UNKNOWN** | The state of the task is unknown. |
| **TASK\_STATE\_QUEUED** | Instances of the task are queued. |

# Conformance

Implementations have discretion over which parts (components, properties, extensions, controlled vocabularies, etc.) of CybOX they implement (e.g., Observable/Object).

[1] Conformant implementations must conform to all normative structural specifications of the UML model or additional normative statements within this document that apply to the portions of CybOX they implement (e.g., implementers of the entire Observable class must conform to all normative structural specifications of the UML model regarding the Observable class or additional normative statements contained in the document that describes the Observable class).

[2] Conformant implementations are free to ignore normative structural specifications of the UML model or additional normative statements within this document that do not apply to the portions of CybOX they implement (e.g., non-implementers of any particular properties of the Observable class are free to ignore all normative structural specifications of the UML model regarding those properties of the Observable class or additional normative statements contained in the document that describes the Observable class).

The conformance section of this document is intentionally broad and attempts to reiterate what already exists in this document.

Acknowledgments

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Participants:

Dean Thompson, Australia and New Zealand Banking Group (ANZ Bank)

Bret Jordan, Blue Coat Systems, Inc.

Adnan Baykal, Center for Internet Security (CIS)

Liron Schiff, Comilion (mobile) Ltd.

Jane Ginn, Cyber Threat Intelligence Network, Inc. (CTIN)

Richard Struse, DHS Office of Cybersecurity and Communications (CS&C)

Ryusuke Masuoka, Fujitsu Limited

Eric Burger, Georgetown University

Jason Keirstead, IBM

Paul Martini, iboss, Inc.

Jerome Athias, Individual

Sanjiv Kalkar, Individual

Terry MacDonald, Individual

Alex Pinto, Individual

Patrick Maroney, Integrated Networking Technologies, Inc.

Wouter Bolsterlee, Intelworks BV

Joep Gommers, Intelworks BV

Sergey Polzunov, Intelworks BV

Rutger Prins, Intelworks BV

Andrei Sîrghi, Intelworks BV

Jonathan Baker, MITRE Corporation

Sean Barnum, MITRE Corporation

Mark Davidson, MITRE Corporation

Ivan Kirillov, MITRE Corporation

John Wunder, MITRE Corporation

Mike Boyle, National Security Agency

Jessica Fitzgerald-McKay, National Security Agency

Takahiro Kakumaru, NEC Corporation

John-Mark Gurney, New Context Services, Inc.

Christian Hunt, New Context Services, Inc.

Andrew Storms, New Context Services, Inc.

Igor Baikalov, Securonix

Bernd Grobauer, Siemens AG

John Anderson, Soltra

Trey Darley, Soltra

Paul Dion, Soltra

Brandon Hanes, Soltra

Ali Khan, Soltra

The authors would also like to thank the larger CybOX Community for its input and help in reviewing this document.

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Editor** | **Changes Made** |
| wd01 | 28 August 2015 | Desiree Beck Trey Darley Ivan Kirillov Rich Piazza | Initial transfer to OASIS template |