CybOX™ Version 2.1.1. Part 10: Account Object

Committee Specification Draft 01 / Public Review Draft 01

20 June 2016

Specification URIs

This version:

http://docs.oasis-open.org/cti/cybox/v2.1.1/csprd01/part10-account/cybox-v2.1.1-csprd01-part10-account.docx (Authoritative)
http://docs.oasis-open.org/cti/cybox/v2.1.1/csprd01/part10-account/cybox-v2.1.1-csprd01-part10-account.html
http://docs.oasis-open.org/cti/cybox/v2.1.1/csprd01/part10-account/cybox-v2.1.1-csprd01-part10-account.pdf

Previous version:

N/A

Latest version:

http://docs.oasis-open.org/cti/cybox/v2.1.1/part10-account/cybox-v2.1.1-part10-account.docx (Authoritative)
http://docs.oasis-open.org/cti/cybox/v2.1.1/part10-account/cybox-v2.1.1-part10-account.html
http://docs.oasis-open.org/cti/cybox/v2.1.1/part10-account/cybox-v2.1.1-part10-account.pdf

Technical Committee:

OASIS Cyber Threat Intelligence (CTI) TC

Chair:

Richard Struse (Richard.Struse@HQ.DHS.GOV), DHS Office of Cybersecurity and Communications (CS&C)

Editors:

Desiree Beck (dbeck@mitre.org), MITRE Corporation
Trey Darley (trey@kingfisherops.com), Individual member
Ivan Kirillov (ikirillov@mitre.org), MITRE Corporation
Rich Piazza (rpiazza@mitre.org), MITRE Corporation

Additional artifacts:

This prose specification is one component of a Work Product whose components are listed in http://docs.oasis-open.org/cti/cybox/v2.1.1/csprd01/cybox-v2.1.1-csprd01-additional-artifacts.html.

Related work:

This specification is related to:

- *STIX™ Version 1.2.1*. Edited by Sean Barnum, Desiree Beck, Aharon Chernin, and Rich Piazza. 05 May 2016. OASIS Committee Specification 01. http://docs.oasis-open.org/cti/stix/v1.2.1/cs01/part1-overview/stix-v1.2.1-cs01-part1-overview.html.

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 Account Object data model, which is one of the Object data models for CybOX content.

Status:

This document was last revised or approved by the OASIS Cyber Threat Intelligence (CTI) TC on the above date. The level of approval is also listed above. Check the “Latest version” location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cti#technical.

TC members should send comments on this specification to the TC’s email list. Others should send comments to the TC’s public comment list, after subscribing to it by following the instructions at the “Send A Comment” button on the TC’s web page at https://www.oasis-open.org/committees/cti/.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the TC’s web page (https://www.oasis-open.org/committees/cti/ipr.php).

Citation format:

When referencing this specification the following citation format should be used:

[CybOX-v2.1.1-account]

CybOX™ Version 2.1.1. Part 10: Account Object. Edited by Desiree Beck, Trey Darley, Ivan Kirillov, and Rich Piazza. 20 June 2016. OASIS Committee Specification Draft 01 / Public Review Draft 01. http://docs.oasis-open.org/cti/cybox/v2.1.1/csprd01/part10-account/cybox-v2.1.1-csprd01-part10-account.html. Latest version: http://docs.oasis-open.org/cti/cybox/v2.1.1/part10-account/cybox-v2.1.1-part10-account.html.

Notices

Table of Contents

1 Introduction

[All text is normative unless otherwise labeled.]

The Cyber Observable Expression (CybOX™) language 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 Account Object Version 2.1.1 data model, which is one of eighty-eight CybOX Object data models.

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

1.1 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 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* 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 various extension data models and a vocabularies data model, which contains a set of default controlled vocabularies. *CybOX™ Version 2.1.1 Part 1: Overview* also summarizes the relationship of CybOX to other languages, and outlines general CybOX data model conventions.

1.2 Document Conventions

The following conventions are used in this document.

1.2.1 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*.

  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 (with single 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*

1.2.2 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 package_prefix for the Account data model is `AccountObj`. Note that in this specification document, we do not explicitly specify the package prefix for any classes that originate from the Account Object data model.

1.2.3 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.

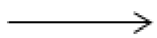
1.2.3.1 Class Properties

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.

1.2.3.2 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. UML diagram icons

Icon	Description
	This diagram icon indicates a class. If the name is in italics, it is an abstract class.
	This diagram icon indicates an enumeration.
	This diagram icon indicates a data type.
	This decorator icon indicates an attribute of a class. The green circle means its visibility is public. If the circle is red or yellow, it means its visibility is private or protected.
	This decorator icon indicates an enumeration literal.
	This arrow type indicates a directed association relationship.
	This arrow type indicates a generalization relationship.

1.2.4 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 Account Object data model (see Section 1.2.2).

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.

1.2.5 Property and Class Descriptions

Each class and property defined in CybOX is described using the format, “The X property verb Y.” 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 preferred 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 identifying 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.

1.3 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]**.

1.4 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.

2 Background Information

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

2.1 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.

2.2 Objects

Cyber observable objects (Files, IP Addresses, etc) in CybOX are characterized with a combination of two levels of data models.

The first level is the Object data model which specifies a base set of properties universal to all types of Objects and enables them to integrate with the overall cyber observable framework specified in the CybOX Core data model.

The second level are the object property models which specify the properties of a particular type of Object via individual data models each focused on 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.

Any specific instance of an Object is represented utilizing the particular object properties data model within the general Object data model.

3 Data Model

3.1 AccountObjectType Class

The AccountObjectType class is intended to characterize generic accounts. The UML diagram corresponding to the AccountObjectType class is shown in **Figure 3-1**.

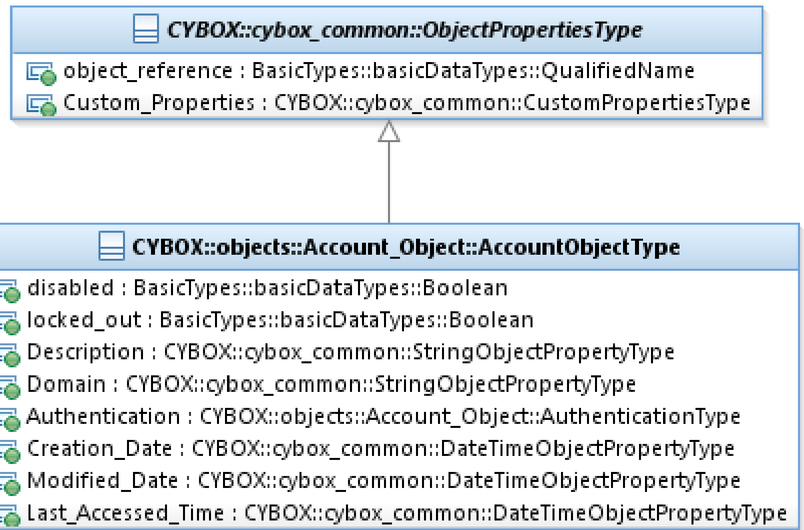


Figure 3-1. UML diagram of the AccountObjectType class

The property table of the AccountObjectType class is given in **Table 3-1**.

Table 3-1. Properties of the AccountObjectType class

Name	Type	Multiplicity	Description
disabled	basicDataTypes:Boolean	0..1	The disabled property specifies whether or not the account is disabled.

locked_out	basicDataTypes:Boolean	0..1	The locked_out property specifies whether or not the account is locked out.
Description	cyboxCommon: StringObjectPropertyType	0..1	The Description property captures a technical description of the Account. Any length is permitted. Optional formatting is supported via the structuring_format property of the StructuredTextType class.
Domain	cyboxCommon: StringObjectPropertyType	0..1	The Domain property is used for specifying the domain to which the account belongs.
Authentication	AuthenticationType	0..*	The Authentication property specifies authentication information associated with this account.
Creation_Date	cyboxCommon: DateTimeObjectPropertyType	0..1	The Creation_Date property specifies the date and time that the account was created.
Modified_Date	cyboxCommon: DateTimeObjectPropertyType	0..1	The Modified_Date property specifies the date and time that the account was last modified.
Last_Accessed_Time	cyboxCommon: DateTimeObjectPropertyType	0..1	The Last_Accessed_Time property specifies the date and time that the account was last accessed.

3.2 AuthenticationType Class

The AuthenticationType class specifies authentication information for an account.

The property table of the AuthenticationType class is given in **Table 3-2**.

Table 3-2. Properties of the AuthenticationType class

Name	Type	Multiplicity	Description
Authentication_Type	cyboxCommon: VocabularyStringType	0..1	The Authentication_Type property specifies the type of authentication required by this Account. Examples of potential values are *no authentication*, *password* and *biometric* (these specific values are only provided to help explain the property: they are neither recommended values nor necessarily part of any existing vocabulary). The content creator may choose any arbitrary value or may constrain the set of possible values by referencing an externally-defined vocabulary or leveraging a formally defined vocabulary extending from the stixCommon:ControlledVocabularyStringType class. The CybOX default vocabulary class for use in the property is AutenticationTypeVocab-1.0.
Authentication_Data	cyboxCommon: StringObjectPropertyType	0..1	The Authentication_Data property specifies the data used for the authentication type specified by the Authentication_Type field. For example, if Authentication_Type is set to *password*, this would be the actual password value.
Authentication_Token_ Protection_Mechanism	cyboxCommon: VocabularyStringType	0..1	The Authentication_Token_Protection_Mechanism property specifies the type of authentication required by this Account. Examples of potential values are *plain text*, *salted SHA-1 hash*, and *DES* (these specific values are only provided to help explain the property: they are neither recommended values nor necessarily part of any existing vocabulary). The content creator may choose any arbitrary value or may constrain the set of possible values by referencing an externally-defined vocabulary or leveraging a formally defined vocabulary extending from the stixCommon:ControlledVocabularyStringType class. The STIX default vocabulary class for use in the property is AuthenticationTokenProtectionMechanismTypeVocab-1.0.
Structured_	StructuredAuthentication	0..1	The Structured_Authentication_Mechanism property describes the authentication mechanism information in a

Authentication_ Mechanism	MechanismType		structured language defined outside of CybOX.

3.3 StructuredAuthenticationMechanismType Class

The StructuredAuthenticationMechanismType class characterizes the description of an authentication mechanism, such as biometrics-based authentication. The StructuredAuthenticationMechanismType class is an abstract class and is intended to be extended via a subclass to enable the expression of any structured authentication. No extension is provided by CybOX to support this, however, those wishing to represent structured authentication mechanism information may develop such an extension.
The property table of the StructuredAuthenticationMechanismType class is given in **Table 3-3**.

Table 3-3. Properties of the StructuredAuthenticationMechanismType class

Name	Type	Multiplicity	Description
Description	cyboxCommon: StructuredTextType	0..1	The Description property captures a technical description of the StructuredAuthenticationMechanism. Any length is permitted. Optional formatting is supported via the structuring_format property of the StructuredTextType class.

3.4 AuthenticationTypeVocab-1.0 Enumeration

The AuthenticationTypeVocab class is the default CybOX vocabulary for authentication class, used in the AuthenticationType/Authentication_Type found in the Account Object package.

The literals of the AuthenticationTypeVocab-1.0 enumeration are given in **Table 3-4**.

Table 3-4. Literals of the AuthenticationTypeEnum-1.0 enumeration

Enumeration Literal	Description

No Authentication	The *No Authentication* value specifies that there is no authentication mechanism in place.
Password	The *Password* value specifies password based authentication.
Cryptographic Key	The *Cryptographic Key* value specifies cryptographic key based authentication.
Biometrics	The Biometrics value specifies biometrics based authentication. Examples include fingerprint or retina readers.
Hardware Token	The Hardware Token value specifies authentication requiring physical or hardware tokens. Examples include smart cards, bluetooth tokens, and USB tokens.
Software Token	The *Software Token* value specifies an authentication device stored in software form.
Multifactor	The *Multifactor* authentication value specifies multifactor authentication.

3.5 AuthenticationTokenProtectionMechanismTypeVocab-1.0 Enumeration

The `AuthenticationTokenProtectionMechanismTypeVocab` enumeration is the default CybOX vocabulary for authentication token protection mechanisms, used in the `AuthenticationType`/`Authentication_Token_Protection_Mechanism` found in the Account Object schema.

The literals of the `AuthenticationTokenProtectionMechanismTypeEnum-1.0` enumeration are given in **Table 3-5**.

Table 3-5. Literals of the `AuthenticationTokenProtectionMechanismTypeEnum-1.0` enumeration

Enumeration Literal	Description

Plaintext	The authentication tokens are stored in plaintext.
Salted GOST Hash	The authentication tokens have been salted and hashed with the GOST hash algorithm.
Unsalted GOST Hash	The authentication tokens have been hashed with the GOST hash algorithm, without salting.
Salted HAVAL Hash	The authentication tokens have been salted and hashed with the HAVAL hash algorithm.
Unsalted HAVAL Hash	The authentication tokens have been hashed with the HAVAL hash algorithm, without salting.
Salted MD2 Hash	The authentication tokens have been salted and hashed with the MD2 hash algorithm.
Unsalted MD2 Hash	The authentication tokens have been hashed with the MD2 hash algorithm, without salting.
Salted MD4 Hash	The authentication tokens have been salted and hashed with the MD4 hash algorithm.
Unsalted MD4 Hash	The authentication tokens have been hashed with the MD4 hash algorithm, without salting.
Salted MD5 Hash	The authentication tokens have been salted and hashed with the MD5 hash algorithm.
Unsalted MD5 Hash	The authentication tokens have been hashed with the MD5 hash algorithm, without salting.

Salted PANAMA Hash	The authentication tokens have been salted and hashed with the PANAMA hash algorithm.
Unsalted PANAMA Hash	The authentication tokens have been hashed with the PANAMA hash algorithm, without salting.
Salted RadioGatun Hash	The authentication tokens have been salted and hashed with the RadioGatun hash algorithm.
Unsalted RadioGatun Hash	The authentication tokens have been hashed with the RadioGatun hash algorithm, without salting.
Salted RIPEMD Hash	The authentication tokens have been salted and hashed with the RIPEMD hash algorithm.
Unsalted RIPEMD Hash	The authentication tokens have been hashed with the RIPEMD hash algorithm, without salting.
Salted RIPEMD-128/256 Hash	The authentication tokens have been salted and hashed with the RIPEMD-128/256 hash algorithm.
Unsalted RIPEMD-128/256 Hash	The authentication tokens have been hashed with the RIPEMD-128/256 hash algorithm, without salting.
Salted RIPEMD-160 Hash	The authentication tokens have been salted and hashed with the RIPEMD-160 hash algorithm.
Unsalted RIPEMD-160 Hash	The authentication tokens have been hashed with the RIPEMD-160 hash algorithm, without salting.
Salted RIPEMD-320 Hash	The authentication tokens have been salted and hashed with the RIPEMD-320 hash algorithm.

Unsalted RIPEMD-320 Hash	The authentication tokens have been hashed with the RIPEMD-320 hash algorithm, without salting.
Salted SHA-0 Hash	The authentication tokens have been salted and hashed with the SHA-0 hash algorithm.
Unsalted SHA-0 Hash	The authentication tokens have been hashed with the SHA-0 hash algorithm, without salting.
Salted SHA-1 Hash	The authentication tokens have been salted and hashed with the SHA-1 hash algorithm.
Unsalted SHA-1 Hash	The authentication tokens have been hashed with the SHA-1 hash algorithm, without salting.
Salted SHA-256/224 Hash	The authentication tokens have been salted and hashed with the SHA-256/224 hash algorithm.
Unsalted SHA-256/224 Hash	The authentication tokens have been hashed with the SHA-256/224 hash algorithm, without salting.
Salted SHA-512/384 Hash	The authentication tokens have been salted and hashed with the SHA-512/384 hash algorithm.
Unsalted SHA-512/384 Hash	The authentication tokens have been hashed with the SHA-512/384 hash algorithm, without salting.
Salted SHA-3 Hash	The authentication tokens have been salted and hashed with the SHA-3 hash algorithm.
Unsalted SHA-3 Hash	The authentication tokens have been hashed with the SHA-3 hash algorithm, without salting.

Salted SHA-3-224 Hash	The authentication tokens have been salted and hashed with the SHA-3-224 hash algorithm.
Unsalted SHA-3-224 Hash	The authentication tokens have been hashed with the SHA-3-224 hash algorithm, without salting.
Salted SHA-3-256 Hash	The authentication tokens have been salted and hashed with the SHA-3-256 hash algorithm.
Unsalted SHA-3-256 Hash	The authentication tokens have been hashed with the SHA-3-256 hash algorithm, without salting.
Salted SHA-3-384 Hash	The authentication tokens have been salted and hashed with the SHA-3-384 hash algorithm.
Unsalted SHA-3-384 Hash	The authentication tokens have been hashed with the SHA-3-384 hash algorithm, without salting.
Salted SHA-3-512 Hash	The authentication tokens have been salted and hashed with the SHA-3-512 hash algorithm.
Unsalted SHA-3-512 Hash	The authentication tokens have been hashed with the SHA-3-512 hash algorithm, without salting.
Salted Tiger(2)-192/160/128 Hash	The authentication tokens have been salted and hashed with the Tiger (2)-192/160/128 hash algorithm.
Unsalted Tiger(2)-192/160/128 Hash	The authentication tokens have been hashed with the Tiger (2)-192/160/128 hash algorithm, without salting.
Salted WHIRLPOOL Hash	The authentication tokens have been salted and hashed with the WHIRLPOOL hash algorithm.

Unsalted WHIRLPOOL Hash	The authentication tokens have been hashed with the WHIRLPOOL hash algorithm, without salting.
Salted Skein-256 Hash	The authentication tokens have been salted and hashed with the Skein-256 hash algorithm.
Unsalted Skein-256 Hash	The authentication tokens have been hashed with the Skein-256 hash algorithm, without salting.
Salted Skein-512 Hash	The authentication tokens have been salted and hashed with the Skein-512 hash algorithm.
Unsalted Skein-512 Hash	The authentication tokens have been hashed with the Skein-512 hash algorithm, without salting.
Salted Skein-1024 Hash	The authentication tokens have been salted and hashed with the Skein-1024 hash algorithm.
Unsalted Skein-1024 Hash	The authentication tokens have been hashed with the Skein-1024 hash algorithm, without salting.
Salted Snefru-128 Hash	The authentication tokens have been salted and hashed with the Snefru-128 hash algorithm.
Unsalted Snefru-128 Hash	The authentication tokens have been hashed with the Snefru-128 hash algorithm, without salting.
Salted Snefru-256 Hash	The authentication tokens have been salted and hashed with the Snefru-256 hash algorithm.
Unsalted Snefru-256 Hash	The authentication tokens have been hashed with the Snefru-256 hash algorithm, without salting.

Iterative Hash	The authentication tokens have been hashed using an iterative hashing algorithm.
AES	The authentication tokens have been encrypted with the AES algorithm.
Blowfish	The authentication tokens have been encrypted with the Blowfish algorithm.
DES	The authentication tokens have been encrypted with the DES algorithm.
IDEA	The authentication tokens have been encrypted with the IDEA algorithm.
RC4	The authentication tokens have been encrypted with the RC4 algorithm.
TEA	The authentication tokens have been encrypted with the Tiny Encryption Algorithm (TEA).

4 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.

Appendix A. Acknowledgments

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

Aetna
David Crawford

AIT Austrian Institute of Technology
Roman Fiedler
Florian Skopik

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

Blue Coat Systems, Inc.
Owen Johnson
Bret Jordan

Century Link
Cory Kennedy

CIRCL
Alexandre Dulaunoy
Andras Iklody
Raphaël Vinot

Citrix Systems
Joey Peloquin

Dell
Will Urbanski
Jeff Williams

DTCC
Dan Brown
Gordon Hundley
Chris Koutras

EMC
Robert Griffin
Jeff Odom
Ravi Sharda

Financial Services Information Sharing and Analysis Center (FS-ISAC)
David Eilken
Chris Ricard

Fortinet Inc.
Gavin Chow

Airbus Group SAS
Joerg Eschweiler
Marcos Orallo

Anomali
Ryan Clough
Wei Huang
Hugh Njemanze
Katie Pelusi
Aaron Shelmire
Jason Trost

Bank of America
Alexander Foley

Center for Internet Security (CIS)
Sarah Kelley

Check Point Software Technologies
Ron Davidson

Cisco Systems
Syam Appala
Ted Bedwell
David McGrew
Pavan Reddy
Omar Santos
Jyoti Verma

Cyber Threat Intelligence Network, Inc. (CTIN)
Doug DePeppe
Jane Ginn
Ben Othman

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

EclecticIQ
Marko Dragoljevic
Joep Gommers
Sergey Polzunov

Kenichi Terashita

Fujitsu Limited

Neil Edwards

Frederick Hirsch

Ryusuke Masuoka

Daisuke Murabayashi

Google Inc.

Mark Risher

Hitachi, Ltd.

Kazuo Noguchi

Akihito Sawada

Masato Terada

iboss, Inc.

Paul Martini

Individual

Jerome Athias

Peter Brown

Elysa Jones

Sanjiv Kalkar

Bar Lockwood

Terry MacDonald

Alex Pinto

Intel Corporation

Tim Casey

Kent Landfield

JPMorgan Chase Bank, N.A.

Terrence Driscoll

David Laurance

LookingGlass

Allan Thomson

Lee Vorthman

Mitre Corporation

Greg Back

Jonathan Baker

Sean Barnum

Desiree Beck

Nicole Gong

Jasen Jacobsen

Ivan Kirillov

Richard Piazza

Jon Salwen

Rutger Prins

Andrei Sîrghi

Raymon van der Velde

eSentire, Inc.

Jacob Gajek

FireEye, Inc.

Phillip Boles

Pavan Gorakav

Anuj Kumar

Shyamal Pandya

Paul Patrick

Scott Shreve

Fox-IT

Sarah Brown

Georgetown University

Eric Burger

Hewlett Packard Enterprise (HPE)

Tomas Sander

IBM

Peter Allor

Eldan Ben-Haim

Sandra Hernandez

Jason Keirstead

John Morris

Laura Rusu

Ron Williams

IID

Chris Richardson

Integrated Networking Technologies, Inc.

Patrick Maroney

Johns Hopkins University Applied Physics Laboratory

Karin Marr

Julie Modlin

Mark Moss

Pamela Smith

Kaiser Permanente

Russell Culpepper

Beth Pumo

Lumeta Corporation

Brandon Hoffman

MTG Management Consultants, LLC.

Charles Schmidt
Emmanuelle Vargas-Gonzalez
John Wunder

National Council of ISACs (NCI)
Scott Algeier
Denise Anderson
Josh Poster

NEC Corporation
Takahiro Kakumaru

North American Energy Standards Board
David Darnell

Object Management Group
Cory Casanave

Palo Alto Networks
Vishaal Hariprasad

Queralt, Inc.
John Tolbert

Resilient Systems, Inc.
Ted Julian

Securonix
Igor Baikalov

Siemens AG
Bernd Grobauer

Soltra
John Anderson
Aishwarya Asok Kumar
Peter Ayasse
Jeff Beekman
Michael Butt
Cynthia Camacho
Aharon Chernin
Mark Clancy
Brady Cotton
Trey Darley
Mark Davidson
Paul Dion
Daniel Dye
Robert Hutto
Raymond Keckler
Ali Khan
Chris Kiehl
James Cabral

National Security Agency
Mike Boyle
Jessica Fitzgerald-McKay

New Context Services, Inc.
John-Mark Gurney
Christian Hunt
James Moler
Daniel Riedel
Andrew Storms

OASIS
James Bryce Clark
Robin Cover
Chet Ensign

Open Identity Exchange
Don Thibeau

PhishMe Inc.
Josh Larkins

Raytheon Company-SAS
Daniel Wyschogrod

Retail Cyber Intelligence Sharing Center (R-CISC)
Brian Engle

Semper Fortis Solutions
Joseph Brand

Splunk Inc.
Cedric LeRoux
Brian Luger
Kathy Wang

TELUS
Greg Reaume
Alan Steer

Threat Intelligence Pty Ltd
Tyron Miller
Andrew van der Stock

ThreatConnect, Inc.
Wade Baker
Cole Iliff
Andrew Pendergast
Ben Schmoker
Jason Spies

TruSTAR Technology

Clayton Long
Michael Pepin
Natalie Suarez
David Waters
Benjamin Yates

Symantec Corp.
Curtis Kostrosky

The Boeing Company
Crystal Hayes

ThreatQuotient, Inc.
Ryan Trost

U.S. Bank
Mark Angel
Brad Butts
Brian Fay
Mona Magathan
Yevgen Sautin

US Department of Defense (DoD)
James Bohling
Eoghan Casey
Gary Katz
Jeffrey Mates

VeriSign
Robert Coderre
Kyle Maxwell
Eric Osterweil

Chris Roblee

United Kingdom Cabinet Office
Iain Brown
Adam Cooper
Mike McLellan
Chris O'Brien
James Penman
Howard Staple
Chris Taylor
Laurie Thomson
Alastair Treharne
Julian White
Bethany Yates

US Department of Homeland Security
Evette Maynard-Noel
Justin Stekervetz

ViaSat, Inc.
Lee Chieffalo
Wilson Figueroa
Andrew May

Yaana Technologies, LLC
Anthony Rutkowski

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

Appendix B. Revision History

Revision	Date	Editor	Changes Made
wd01	15 December 2015	Desiree Beck Trey Darley Ivan Kirillov Rich Piazza	Initial transfer to OASIS template

