THE MITRE CORPORATION

STIX™ 1.1.1
DATA MARKING SPECIFICATION (V1.1.1)

MAY 29, 2015

The Structured Threat Information eXpression (STIX^{TM}) framework defines eight core constructs and the relationships between them for the purposes of modeling cyber threat information and enabling cyber threat information analysis and sharing. This specification document defines the Data Marking data model, which provides an independent, flexible, structured capability for data marking expression.

Acknowledgements

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

Trademark Information

STIX, the STIX logo, and CybOX are trademarks of The MITRE Corporation. All other trademarks are the property of their respective owners.

Warnings

MITRE PROVIDES STIX "AS IS" AND MAKES NO WARRANTY, EXPRESS OR IMPLIED, AS TO THE ACCURACY, CAPABILITY, EFFICIENCY, MERCHANTABILITY, OR FUNCTIONING OF STIX. IN NO EVENT WILL MITRE BE LIABLE FOR ANY GENERAL, CONSEQUENTIAL, INDIRECT, INCIDENTAL, EXEMPLARY, OR SPECIAL DAMAGES, RELATED TO STIX OR ANY DERIVATIVE THEREOF, WHETHER SUCH CLAIM IS BASED ON WARRANTY, CONTRACT, OR TORT, EVEN IF MITRE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.[1]

Feedback

The STIX development team welcomes any feedback regarding this document. Please send any comments, questions, or suggestions to stix@mitre.org.[2]

---

[1] For detailed information see [TOU].
[2] For more information about the STIX Language, please visit [STIX].

Table of Contents

1 Introduction

The Structured Threat Information eXpression (STIX^{TM}) framework defines eight top-level component data models: Observable[3], Indicator, Incident, TTP, ExploitTarget, CourseOfAction, Campaign, and ThreatActor. In addition, it defines a data model that captures data marking information for STIX content. This document serves as the specification for the STIX Data Marking Version 1.1.1 data model.

Given the potentially sensitive nature of cyber threat information, a consistent requirement across many of the STIX component data models is the ability to represent markings of the data to specify things such as handling restrictions, terms of use, or copyright information. There currently exists no broad consensus standardized approach for such data markings; instead, there are various approaches within differing communities, driven by different motivations and usage contexts. Therefore, rather than adopting a single marking approach and expecting all STIX users to accept it, STIX takes a flexible and generic approach through the definition of the Data Marking data model.

In Section 1.1 we discuss STIX specification documents, and in Section 1.2 we give document conventions. In Section 2, we give background information necessary to fully understand the Data Marking data model, and we present the Data Marking data model specification details in Section 3. References are provided in the final section.

1.1   STIX Specification Documents

The STIX 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 STIX UML model.

The STIX specification overview document provides a comprehensive overview of the full set of STIX data models [STIX$_O$], which in addition to the eight top-level component data models mentioned in the Introduction, includes a core data model, a common data model, a cross-cutting data marking data model, various extension data models, and a set of default controlled vocabularies. [STIX$_O$] also summarizes the relationship of STIX to other languages, and outlines general STIX data model conventions.

Figure 1-1 illustrates the set of specification documents that are available. The color black is used to indicate the specification overview document, altered shading differentiates the overarching Core and Common data models from the supporting data models (default vocabularies, data marking, and extensions), and the color white indicates the component data models. The Observable component data model is shown as an oval shape to indicate

---

[3] The CybOX Observable data model is actually defined in the CybOX Language, not in STIX.

that it is defined as a CybOX specification (see [STIX$_O$] for details). This Data Marking specification document is highlighted in its associated color (see Section 1.2.4.3). For a list of all STIX documents and related information sources, please see [STIX$_O$].
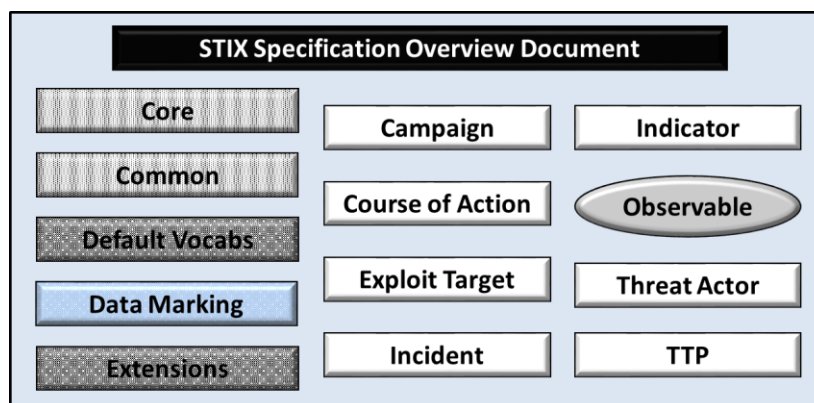
Figure 1-1. STIX Language v1.1.1 specification documents

All specification documents can be found on this STIX Website [STIX-SPECS].

1.2   Document Conventions

The following conventions are used in this document.

1.2.1   Key Words

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 *RFC 2119* [RFC2119].

1.2.2   Fonts

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

- Capitalization is used for STIX high level concepts, which are defined in the STIX Specification Overview [STIX$_O$].

  Examples: Indicator, Course of Action, Threat Actor

- The `Courier New` font is used for writing UML objects.

  Examples: `RelatedIndicatorsType`, `stixCommon:StatementType`

  Note that all high level concepts have a corresponding UML object. For example, the Course of Action high level concept is associated with a UML class named, `CourseOfActionType`.

- The *'italic'* font (with single quotes) is used for noting actual, explicit values for STIX Language properties. The *italic* font (without quotes) is used for noting example values.

Example: *'PackageIntentVocab-1.0,' high, medium, low*

1.2.3 UML Package References

Each STIX data model is captured in a different UML package (e.g., Core package, Campaign package, etc.) where the packages together compose the full STIX 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 STIX^{TM} 1.1.1 Specification Overview document [STIX$_O$] contains a list of the packages used by the Data Marking data model, along with the associated prefix notations, descriptions, examples.

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

1.2.4 UML Diagrams

This specification makes use of UML diagrams to visually depict relationships between STIX Language constructs. Note that the diagrams have been extracted directly from the full UML model for STIX; 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 STIX 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. 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.

1.2.4.1 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. For example, properties of a class that are identifiers, titles, and timestamps will be represented as attributes.

1.2.4.2 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.

1.2.4.3 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 Data Marking specification are illustrated via exemplars in Figure 1-2.

Figure 1-2. Data model color coding

1.2.5   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.

4

Package prefixes are provided for classes outside of the Data Marking 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.

In addition, properties that are part of a "choice" relationship (e.g., Prop1 OR Prop2 is used but not both) will be denoted by a unique letter subscript (e.g., API_Call$_A$, Code$_B$) and single logic expression in the Multiplicity column. For example, if there is a choice of property `API_Call`$_A$ and `Code`$_B$, the expression "A(1)|B(0..1)" will indicate that the `API_Call` property can be chosen with multiplicity 1 or the `Code` property can be chosen with multiplicity 0 or 1.

1.2.6   Property and Class Descriptions

Each class and property defined in STIX is described using the format, "The X property <u>verb</u> Y." For example, in the specification for the STIX Campaign, we write, "The `id` property <u>specifies</u> a globally unique identifier for the Campaign instance." 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 STIX.

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

Verb	STIX Definition
<u>captures</u>	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 `Source` property characterizes the source of the sighting information. Examples of details <u>captured</u> include identitifying characteristics, time-related attributes, and a list of the tools used to collect the information. The `Description` property <u>captures</u> a textual description of the Indicator.
<u>characterizes</u>	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.
	Example: The `Confidence` property <u>characterizes</u> the level of confidence in the accuracy of the overall content captured in the Incident. The `ActivityType` class <u>characterizes</u> basic information about an activity a defender might use in response to a Campaign.
<u>specifies</u>	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 `version` property <u>specifies</u> the version identifier of the STIX Campaign data model used to capture the information associated with the Campaign.

2 Background Information

In this section, we provide high level information about the Data Marking data model that is necessary to fully understand the Data Marking data model specification details given in Section 3. As explained in the introduction, the data marking construct is not conveyed as a separate entity in the STIX architecture diagram like the eight component data models; instead, the data marking construct exists as a cross-cutting structure across all of those constructs.

2.1   Marking Approach

There are two aspects to the STIX approach to data marking: (1) a controlled structure is used to specify the set of STIX content elements to which data markings apply, and (2) a marking structure is used to specify the particular data markings that are applied to the set of elements identified by the controlled structure.

This approach makes STIX data marking flexible in two ways. First, it permits the use of *any* data marking structure simply as a specialization of the Data Marking base class (the `MarkingStructureType` class; see Section 3.3). Second, data marking information is specified separately from the STIX content being marked: instead of embedding the marking information within an individual property, property locations are *referenced* from a higher level (the `Controlled_Structure` property of the `MarkingType` class; see Section 3.2). This makes data marking information space efficient and easier to update and refine, and it also enables any given STIX content to be marked with multiple marking schemes. Any level of information can be marked: individual properties, an entire STIX document, or anything in between.[4] For example, a copyright may be applied across a whole document while specific terms of use might apply only to certain properties of Indicator test mechanisms.

2.2   Using Markings

Before discussing about how markings are represented in STIX, it may be useful to understand how and where markings are used. The most common place to see data markings is in the `Handling` property of the STIX Header (`STIXHeaderType` class). Markings placed in this property are often used to apply markings globally, either to the entire STIX Package or to specific types of information regardless of where they appear in the STIX Package. For example, a copyright that applies to the entire STIX Package would be best placed in the `Handling` property of the STIX Header. Similarly, the indication that *all* Indicator Courses of Action are TLP:RED[5] would also be best placed in the STIX Header.

---

[4] STIX does not inherently provide for marking at every level; an appropriate document selection language defined outside of STIX must be used (see Section 3.2).

[5] The Traffic Light Protocol (TLP) was created by US-CERT; http://www.us-cert.gov/tlp/

However, the STIX Header is not the only place where data markings can be used. Individual STIX components (Indicators, Campaigns, etc.) all have their own `Handling` property, which if used restricts the marking applicability to just the properties within that component. This allows consumers to safely preserve markings within a component and move it between documents or into a datastore without worrying that the markings will change in meaning. Note that if the Handling property is placed directly in an individual component (e.g., `IndicatorType` class) rather than in a STIX Header, the `Handling` property name is still of type `marking:MarkingType` because the Data Marking data model provides a common structure, regardless of where data markings are used.

8

3 STIX Data Marking Data Model

The STIX Data Marking data model defines three classes used to capture data marking information for STIX content. Each of these classes is defined below.

3.1   MarkingType Class

The `MarkingType` class specifies a set of zero or more data marking specifications to be applied to the STIX content.

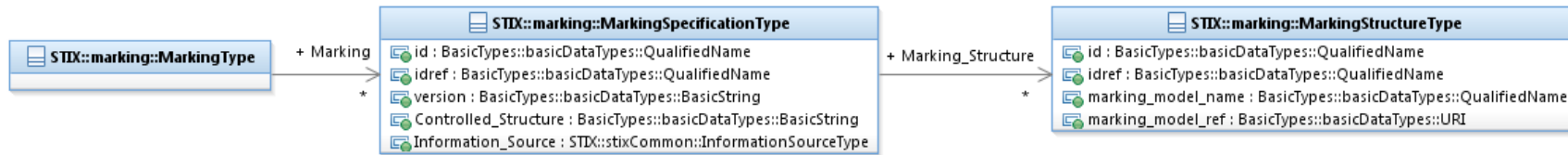The property table for the `MarkingType` class is shown in Table 3-1.

Table 3-1. Properties of the `MarkingType` class

Name	Type	Multiplicity	Description
Marking	MarkingSpecificationType	0..*	The `Marking` property characterizes a data marking specification that is applied to STIX content. Information captured includes the structured elements to which the data marking is to be applied, a set of marking structures, and source information.

3.2   MarkingSpecificationType Class

The `MarkingSpecificationType` class characterizes a data marking specification that is applied to the STIX content. Information captured includes the structured elements to which the data marking is to be applied, a set of marking structures, and source information.

The UML diagram of the `MarkingSpecificationType` class is shown in Figure 3-1.

9

Figure 3-1. UML diagram of the `MarkingSpecificationType` class

The property table of the `MarkingSpecificationType` class that corresponds to Figure 3-1 is given in Table 3-2.

Table 3-2. Properties of the `MarkingSpecificationType` class

Name	Type	Multiplicity	Description
id	`basicDataTypes:QualifiedName`	0..1	The `id` property specifies a globally unique identifier for a data marking specification instance.
idref	`basicDataTypes:QualifiedName`	0..1	The `idref` property specifies a reference to the identifier of a data marking specification instance specified elsewhere; the referenced data marking should be evaluated as if it were located where the data marking reference is defined. When the `idref` property is used, the `id` property MUST NOT also be specified and the other properties of the `MarkingSpecificationType` class SHOULD NOT hold any content.
version	`basicDataTypes:BasicString`	0..1	The `version` property specifies the version number of the STIX Data Marking data model used to capture the data marking associated with the STIX content.
Controlled_Structure	`basicDataTypes:BasicString`	0..1	The `Controlled_Structure` property specifies the full explicit set of STIX structured elements to which the marking is to be applied. The controlled structure MUST explicitly select *all* structured elements that the marking applies to; selecting a parent structured element may not imply that the marking also applies to its children. Specific syntax for how

|			the set of STIX structured elements will be specified is dependent on the particular syntactic implementation (XML, JSON, etc.) of the STIX language and MUST be explicitly specified in a separate binding specification for that syntactic implementation (e.g. a STIX XML Binding Specification). For example, a STIX XML Binding Specification could specify XPath 1.0[6] as an appropriate choice for the syntax of the `Controlled_Structure` property.
Marking_Structure	`MarkingStructureType`	0..*	The `Marking_Structure` property characterizes the marking information to be applied to a portion of STIX content as specified in the `Controlled_Structure` property. Its underlying class is intended to be extended to enable the expression of any structured or unstructured data marking mechanism.
Information_Source	`stixCommon: InformationSourceType`	0..1	The `Information_Source` property characterizes the source of the data marking specification information. Examples of details captured include identifying characteristics (e.g., who marked the data) and time-related attributes (e.g., when the data was marked).

3.3   MarkingStructureType Class

The `MarkingStructureType` class characterizes the marking information to be applied to STIX content. The class is simply a mechanism for leveraging externally defined marking systems, and it is intended to be extended to enable the expression of any structured or unstructured data marking mechanism.

As illustrated in Figure 3-2, STIX v1.1.1 defines default subclasses for three particular data marking formats: Simple, Traffic Light Protocol (TLP), and Terms of Use (qualified names are not shown in the figure due to space considerations). See [STIX_EXT] for details. Producers who want to use another marking system may simply define a new extension to the `MarkingStructureType` class.

---

[6] XPath 1.0 is a language for selecting portions of XML documents.

It is valid to mark a structured element with multiple markings from the same system or mark a structured element across multiple marking systems. If a structured element is marked multiple times using the same marking system, that system (not STIX) is responsible for specifying the semantic meaning of multiple markings, and if necessary, for specifying how conflicts should be resolved. If a structured element is marked across multiple marking systems, each system is considered individually applicable. If there are conflicting markings across marking systems the behavior is undefined; therefore, producers should make every effort to ensure documents are marked consistently and correctly among all marking systems. The data marking systems themselves should also define the interpretation of unmarked structured elements.
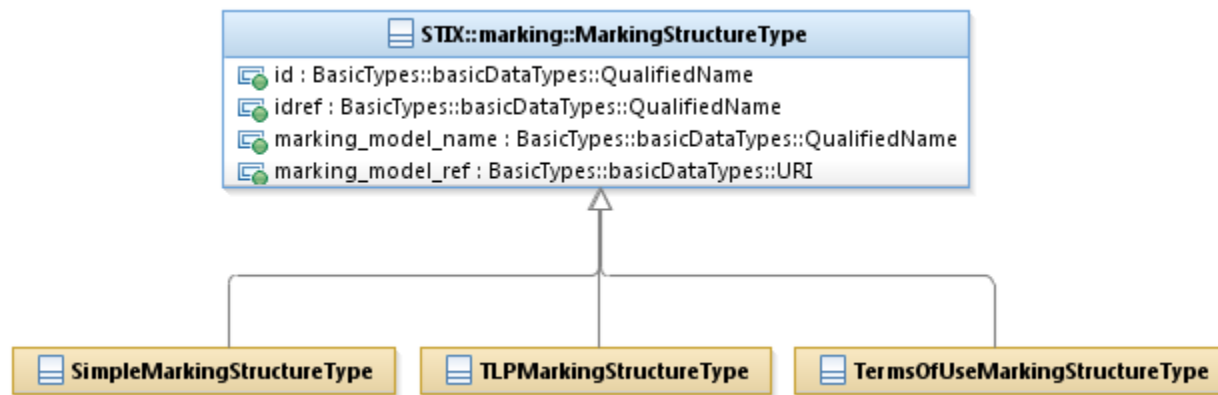
Figure 3-2. UML diagram of the `MarkingStructureType` class

As listed in Table 3-3, the three default subclasses and their descriptions are defined as possible extensions to the `MarkingStructureType` class. As stated above, additional markings can be used by defining a new subclass of the `MarkingStructureType` class. [STIX_{EXT}] gives further details of each extension shown.

Table 3-3. Default extensions of the `MarkingStructureType` class

Subclasses	Description
SimpleMarkingStructureType	The Simple marking structure allows users to make a text statement to mark the content. For example, copyright information can be communicated.
TLPMarkingStructureType	The Traffic Light Protocol (TLP) marking structure indicates how content may be shared. TLP statements are indicated through the use of a simple enumeration.
TermsOfUseMarkingStructureType	The Terms of Use marking structure allows users to make a text statement to specify the terms of use of the marked content. This marking is similar to the Simple marking structure, but it has stronger semantic meaning.

To reiterate, the `MarkingStructureType` class is simply a mechanism for leveraging externally defined marking systems. The data marking systems themselves define the semantics of what the markings mean, how multiple markings to the same structured element should be applied, and what to do if a structured element is unmarked. The `MarkingStructureType` class can be used to mark the data with anything. For example, data markings could be used to indicate that the STIX document is part of an exercise and is not actual production data.

The properties of the `MarkingStructureType` class are given in Table 3-4.

Table 3-4. Properties of the `MarkingStructureType` class

Name	Type	Multiplicity	Description
id	`basicDataType:QualifiedName`	0..1	The `id` property specifies a globally unique identifier for the marking structure instance.
idref	`basicDataType:QualifiedName`	0..1	The `idref` property specifies a reference to the identifier of a marking structure instance specified elsewhere; the referenced data marking should be evaluated as if it were located where the data marking reference is defined. When `idref` is specified, the `id` property MUST NOT also be specified, any other properties of the `MarkingStructureType` class

			SHOULD NOT hold any content, and the `MarkingStructureType` class SHOULD NOT be extended.
marking_model_name	`basicDataType:QualifiedName`	0..1	The `marking_model_name` property specifies the name of the marking model to be applied within the marking structure.
marking_model_ref	`basicDataType:URI`	0..1	The `marking_model_ref` property specifies a reference URI for the location of the authoriative descriptive source on the marking model to be applied within the marking structure.

References

References made in this document are listed below.

[CybOX$_{COR}$]   CybOX Core Specification (*not yet available*).

 [RFC2119]   RFC 2119 – Key words for use in RFCs to Indicate Requirement Levels
              http://www.ietf.org/rfc/rfc2119.txt

[STIX]   STIX Web Site
              https://stix.mitre.org

[STIX-SPECS] STIX^{TM} Project Github Site
              http://github.com/STIXProject/specifications

[STIX$_{EXT}$]   STIX^{TM} 1.1.1 Default Extensions Specification
              http://stix.mitre.org/about/documents/XXXX.pdf

[STIX$_O$]   STIX^{TM} 1.1.1 Specification Overview
              http://stix.mitre.org/about/documents/XXXX.pdf

[TOU]   Terms of Use
              http://stix.mitre.org/about/termsofuse.html