STIX 1.0 Release Notes

4/8/2013

Summary

This document contains release notes for STIX 1.0. Compared to STIX 1.0 Draft 2, STIX 1.0 contains significant modifications and a number of new concepts. These release notes describe the modifications and new concepts at a high level, and only address a subset of detailed schema changes. The core concepts of STIX are unchanged, but their schematic expressions may have changed. This document describes those changes.

New Concepts

This section contains concepts introduced in STIX 1.0.

Controlled Vocabularies

The Controlled Vocabulary mechanism provides the capability to specify a set of valid values for a particular term concept (e.g. Incident_Status) that can be instantiated in particular fields. Each field that is deemed relevant for controlled vocabulary use (including a good portion of fields that had preexisting enumerations defined) has been set to use this capability. Content authors may, at the time of content creation, specify the appropriate controlled vocabulary to be applied to that field using the xsi:type attribute to reference a controlled vocabulary type extension of the base 'stixCommon:ControlledVocabularyStringType'. A set of default vocabularies are made available (in the stix_default_vocabularies.xsd file) with STIX (including preexisting enumerations and some new ones as well) and are the suggested use but content authors are free to define and use their own custom vocabularies if desired or even to intentionally not specify a vocabulary and simply enter a custom value in the field.

To be more specific, where Controlled Vocabularies are used in STIX, they have a type of 'stixCommon:ControlledVocabularyStringType'. XML elements that are of this type must use the xsi:type mechanism (discussed below) to specify which controlled vocabulary is being used or leave xsi:type unspecified to enable free use of custom values.

In order to use a Controlled Vocabulary from the default Controlled Vocabulary schema file, an XML instance document must contain the following:

1. A namespace declaration of the XML Schema Instance namespace (e.g., xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance")
2. A namespace declaration of the Controlled Vocabulary namespace (e.g., xmlns:stixVocabs="http://stix.mitre.org/default_vocabularies-1")

3. In an element of type ControlledVocabularyStringType, the xsi:type attribute must be present, and it must be set to a type whose base type is ControlledVocabularyStringType (e.g., xsi:type="stixVocabs:PackageIntentVocab-1.0" for the Package Intent Controlled Vocabulary).
4. [Optional] An xsi:schemaLocation element specifying the location of the default Controlled Vocabulary schema file. This is useful for validation in XML editors (e.g., xsi:schemaLocation="http://stix.mitre.org/default_vocabularies-1 stix_default_vocabularies.xsd")

A snippet demonstrating use of the Package Intent Controlled Vocabulary (in typical usage, the namespaces and schemaLocation defined here would occur once within the file header rather than at the field level):

```
<stix:Package_Intent
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:stixVocabs="http://stix.mitre.org/default_vocabularies-1"
   xsi:schemaLocation="http://stix.mitre.org/default_vocabularies-1 stix_default_vocabularies.xsd"
   xsi:type="stixVocabs:PackageIntentVocab-1.0">Indicators - Watchlist</stix:Package_Intent>
```

A number of new vocabularies were added including Package_Intent and Indicator Type as well as a set of vocabularies for characterizing ThreatActors, TTPs and Incidents based on practical input from iSight Partners.

Extension using xsi:type

While previous revisions of STIX used the xsi:type construct for specialization, it is used much more frequently in this revision of STIX for specialization and for decoupling. Understanding of the xsi:type construct is required to use STIX effectively.

These release notes do not discuss the xsi:type construct in detail, as it is defined in the XML Schema specification (http://www.w3.org/TR/xmlschema-1/#xsi_type) and discussed at length in a variety of places (An internet search of 'using xsi:type' will reveal these discussions). Users of STIX are encouraged to become familiar with the xsi:type mechanism.

Cross-cutting Changes

Files

This is a summary of file changes in STIX 1.0:

1. File names in STIX 1.0 no longer contain version information. Version information can be found in two locations: the complete version is found in the version attribute on the root element of the XML Schema, and the major version is found in the namespace of a schema.

2. File names are now in all lowercase (with the exception of Data_Marking.xsd) and have underscores rather than hyphens, where applicable.

Schema Namespaces

Schema namespaces previously did not indicate version information. Namespaces now have a suffix that indicates the major version of the schema (e.g., schemas with a major version of 1 end in '-1').

Schema Decoupling

In previous revisions of STIX, stix.xsd imported schema files for each of the STIX core constructs (Indicator, Incident, et al) directly in a tightly-coupled fashion. This resulted in unnecessary dependencies and a larger than necessary footprint for users that only used a portion of the core constructs. All core constructs (except Observables) have been decoupled through the use of extension and the xsi:type mechanism, allowing users to use only the portion of STIX that is relevant for them.

In addition, all externally-defined schemas that are leveraged in STIX (MAEC, CAPEC, CIQ, CVRF, etc.) are now decoupled in a similar fashion. There are now base types for each type of content appropriate for extension (attack patterns, malware characterization, structured identity characterization, vulnerability characterization, etc.) and default extension of these base types utilizing the preferred externally-defined schema appropriate for that type of content. It is strongly suggested that these default extensions be leveraged but it is possible for content authors to define and use their own custom extensions in other formats if desired.

Test Mechanism Extensions

The STIX Indicator component contains a Test_Mechanism extension point for conveying indicator pattern content using non-standard, external formats (Snort, OpenIOC, OVAL, YARA, etc.) utilized by common tooling. STIX 1.0 provides default extensions of this extension point for expressing Snort rules, OpenIOC indicators, OVAL checks, YARA rules, and a generic mechanism for expressing any other format desired without a design-time understanding of format. Users are free to define other desired extensions for Test-Mechanism to provide design-time structure and implementation-time validation capabilities.

Data Marking Extensions

The STIX Data_Marking schema provides a flexible mechanism for specifying data markings utilizing various marking models implemented as extensions of the MarkingStructureType. STIX 1.0 provides two default marking model extensions: US-CERT's Traffic Light Protocol (TLP) and a Simple_Marking structure that enables specification of freeform string Statements.

In addition, the guidance for applying data markings was revised for clarity.

Versioning Policy

STIX 1.0 now has a defined versioning policy that describes how core, common, the default vocabularies, components, and extensions will be updated. This will allow developers, users, authors, and others to understand the extent and nature of changes to STIX schemas and what that means to compatibility for existing content, consumers, and producers. The full details of the STIX versioning policy can be found on the STIX website (http://stix.mitre.org/).

Schema Documentation

Annotations were added to each schema file, documenting the version, creation date, description, and terms of use.

CybOX

STIX 1.0 now uses CybOX 2.0. Previous revisions of STIX used CybOX 1.0.

Notable Schema Changes

1. Added a 'title' field to all top-level constructs.
2. StructuredTextType was added directly to STIX and removed the custom structures previously defined and replaced with a simpler and more flexible structure.
3. Normalization and refinement of a common Relationship structure for use in characterizing relationships between STIX components and other relevant constructs.
4. Definition of a new Statement type for broad use in fields that require some sort of value assertion (potentially with an associated controlled vocabulary), source of the assertion, time of the assertion, description of the assertion and confidence in the assertion. This improves consistency of such structures across STIX.
5. Added Handling constructs to each STIX component
6. Added the ability to specify a simple name for Identity fields if a structured identity representation is not needed.
7. Fleshed out ThreatActor component with other meaningful information like Type of actor, Motivation, Intended_Effects, and Planning_And_Operational_Support.
8. Fleshed out VictimTargeting construct with ability to specify Targeted_Sytems and Targeted_Information in addition to Identity
9. Fixed a bug in Kill Chains that prevented the complete usage of Kill Chains.