STIX[TM] Version 1.2.1. Part 5: TTP

Committee Specification Draft 01

06 November 2015

Specification URIs

This version:

http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part5-ttp/stix-v1.2.1-csd01-part5-ttp.docx (Authoritative)
http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part5-ttp/stix-v1.2.1-csd01-part5-ttp.html
http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part5-ttp/stix-v1.2.1-csd01-part5-ttp.pdf

Previous version:

N/A

Latest version:

http://docs.oasis-open.org/cti/stix/v1.2.1/stix-v1.2.1-part5-ttp.docx (Authoritative)
http://docs.oasis-open.org/cti/stix/v1.2.1/stix-v1.2.1-part5-ttp.html
http://docs.oasis-open.org/cti/stix/v1.2.1/stix-v1.2.1-part5-ttp.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:

Sean Barnum (sbarnum@mitre.org), MITRE Corporation
Desiree Beck (dbeck@mitre.org), MITRE Corporation
Aharon Chernin (achernin@soltra.com), Soltra
Rich Piazza (rpiazza@mitre.org), MITRE Corporation

Additional artifacts:

This prose specification is one component of a Work Product that also includes:

- *STIX Version 1.2.1. Part 1: Overview*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part1-overview/stix-v1.2.1-csd01-part1-overview.html
- *STIX Version 1.2.1. Part 2: Common*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part2-common/stix-v1.2.1-csd01-part2-common.html
- *STIX Version 1.2.1. Part 3: Core*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part3-core/stix-v1.2.1-csd01-part3-core.html
- *STIX Version 1.2.1. Part 4: Indicator*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part4-indicator/stix-v1.2.1-csd01-part4-indicator.html
- *STIX Version 1.2.1. Part 5: TTP* (this document). http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part5-ttp/stix-v1.2.1-csd01-part5-ttp.html
- *STIX Version 1.2.1. Part 6: Incident*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part6-incident/stix-v1.2.1-csd01-part6-incident.html
- *STIX Version 1.2.1. Part 7: Threat Actor*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part7-threat-actor/stix-v1.2.1-csd01-part7-threat-actor.html
- *STIX Version 1.2.1. Part 8: Campaign*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part8-campaign/stix-v1.2.1-csd01-part8-campaign.html

- *STIX Version 1.2.1. Part 9: Course of Action*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part9-coa/stix-v1.2.1-csd01-part9-coa.html
- *STIX Version 1.2.1. Part 10: Exploit Target*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part10-exploit-target/stix-v1.2.1-csd01-part10-exploit-target.html
- *STIX Version 1.2.1. Part 11: Report*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part11-report/stix-v1.2.1-csd01-part11-report.html
- *STIX Version 1.2.1. Part 12: Default Extensions*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part12-extensions/stix-v1.2.1-csd01-part12-extensions.html
- *STIX Version 1.2.1. Part 13: Data Marking*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part13-data-marking/stix-v1.2.1-csd01-part13-data-marking.html
- *STIX Version 1.2.1. Part 14: Vocabularies*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part14-vocabularies/stix-v1.2.1-csd01-part14-vocabularies.html
- *STIX Version 1.2.1. Part 15: UML Model*. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part15-uml-model/stix-v1.2.1-csd01-part15-uml-model.html
- UML Model Serialization: http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/uml-model/

Related work:

This specification replaces or supersedes:

- *STIX™ 1.2 TTP Specification (v1.2)* https://github.com/STIXProject/specifications/blob/version1.2/documents/pdf%20versions/STIX_TTP_Draft.pdf

This specification is related to:

- *CybOX[TM] Version 2.1.1.* Work in progress. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cti-cybox
- *CybOX[TM] 2.1.* https://cyboxproject.github.io/

Abstract:

The Structured Threat Information Expression (STIX) framework defines nine 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 Tactics, Techniques, and Procedures (TTP) construct, which captures the behavior or modus operandi of cyber adversaries.

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:

[STIX-v1.2.1-TTP]

STIX[TM] Version 1.2.1. Part 5: TTP. Edited by Sean Barnum, Desiree Beck, Aharon Chernin, and Rich Piazza. 06 November 2015. OASIS Committee Specification Draft 01. http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part5-ttp/stix-v1.2.1-csd01-part5-ttp.html. Latest version: http://docs.oasis-open.org/cti/stix/v1.2.1/stix-v1.2.1-part5-ttp.html.

Notices

Table of Contents

1 Introduction 6
1.1 STIX[TM] Specification Documents 6
1.2 Document Conventions 7
1.2.1 Fonts 7
1.2.2 UML Package References 7
1.2.3 UML Diagrams 7
1.2.4 Property Table Notation 9
1.2.5 Property and Class Descriptions 9
1.3 Terminology 10
1.4 Normative References 10
2 Background 11
2.1 TTP-Related Component Data Models 11
3 STIX[TM] TTP Data Model 12
3.1 TTPVersionType Enumeration 15
3.2 BehaviorType Class 15
3.2.1 AttackPatternsType Class 17
3.2.2 MalwareType Class 19
3.2.3 ExploitsType Class 21
3.3 ResourceType Class 23
3.3.1 ToolsType Class 24
3.3.2 InfrastructureType Class 24
3.3.3 PersonasType Class 26
3.4 VictimTargetingType Class 26
3.5 ExploitTargetsType Class 28
3.6 RelatedTTPsType Class 29
4 Conformance 31
Appendix A. Acknowledgments 32
Appendix B. Revision History 34

1 Introduction

[All text is normative unless otherwise labeled]

The Structured Threat Information Expression (STIX[TM]) framework defines nine top-level component data models: Observable[1], Indicator, Incident, TTP, ExploitTarget, CourseOfAction, Campaign, ThreatActor, and Report. This document serves as the specification for the STIX Tactics, Techniques, and Procedures (TTP) data model.

As defined within the STIX language, a TTP construct characterizes adversarial mode of operations (often referred to as the adversary's "Tactics, Techniques, and Procedures"), such as the victims targeted, the attack patterns and malware used, and the resources (infrastructure, tools, and personas) leveraged. Because the TTP construct describes adversary behavior, which is a central objective of STIX, it is one of the most commonly used and expressive constructs.

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 Course of Action data model. We present the TTP data model specification details in Section **3** and conformance information in Section **4**.

1.1 STIX[TM] 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 Version 1.2.1 Part 1: Overview* document provides a comprehensive overview of the full set of STIX data models, which in addition to the nine 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 Version 1.2.1 Part 1: Overview* 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 (vocabularies, data marking, and default extensions), and the color white indicates the component data models. The solid grey color denotes the overall STIX Language UML model. This TTP specification document is highlighted in its associated color (see Section **1.2.3.3**). For a list of all STIX documents and related information sources, please see *STIX Version 1.2.1 Part 1: Overview*.

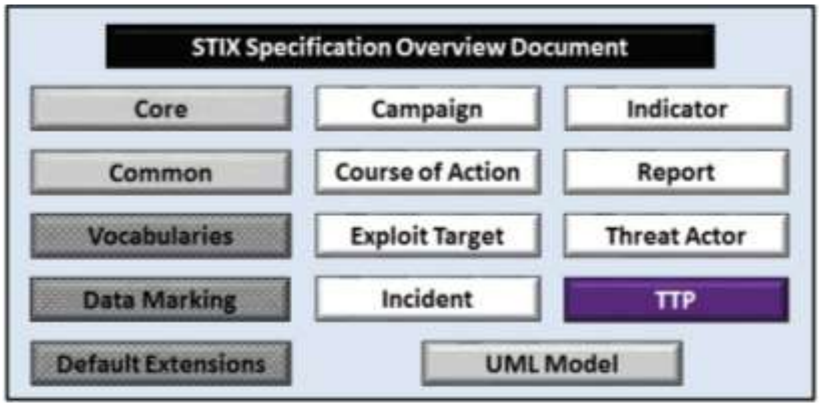


Figure 1-1. STIX[TM] Language v1.2.1 specification documents

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 STIX high level concepts, which are defined in *STIX Version 1.2.1 Part 1: Overview*.

  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*

1.2.2 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. *STIX Version 1.2.1 Part 1: Overview* contains a list of the packages used by the TTP 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 TTP data model.

1.2.3 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. 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. For example, properties of a class that are identifiers, titles, and timestamps will be represented 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 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.
‹E›	This diagram icon indicates an enumeration.
‹D›	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.

1.2.3.3 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 TTP specification are illustrated in **Figure 1-2**.

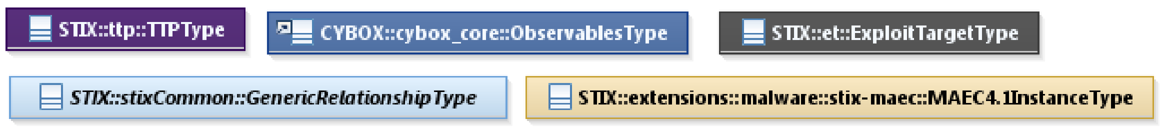

STIX::ttp::TTPType
CYBOX::cybox_core::ObservablesType
STIX::et::ExploitTargetType
STIX::stixCommon::GenericRelationshipType
STIX::extensions::malware::stix-maec::MAEC4.1InstanceType

Figure 1-2. Data model color coding

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 TTP 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.

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.

1.2.5 Property and Class Descriptions

Each class and property defined in STIX is described using the format, "The X property verb Y." For example, in the specification for the STIX Indicator, we write, "The `id` property specifies a globally unique identifier for the kill chain 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
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 `Source` property characterizes the source of the sighting 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 Indicator.
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 `Confidence` property characterizes the level of confidence in the accuracy of the overall content captured in the Incident. The `ActivityType` class characterizes basic information about an activity a defender might use in response to a Campaign.
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 `version` property specifies the version identifier of the STIX Campaign data model used to capture the information associated with the Campaign.

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

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

2.1 TTP-Related Component Data Models

As will be explicitly detailed in Section **3**, a STIX TTP leverages the Exploit Target data model (as indicated by the outward-oriented arrow). **Figure 2-1** illustrates the relationship between the TTP and the other core constructs. As stated in Section **1.1**, each of these components is defined in a separate specification document.

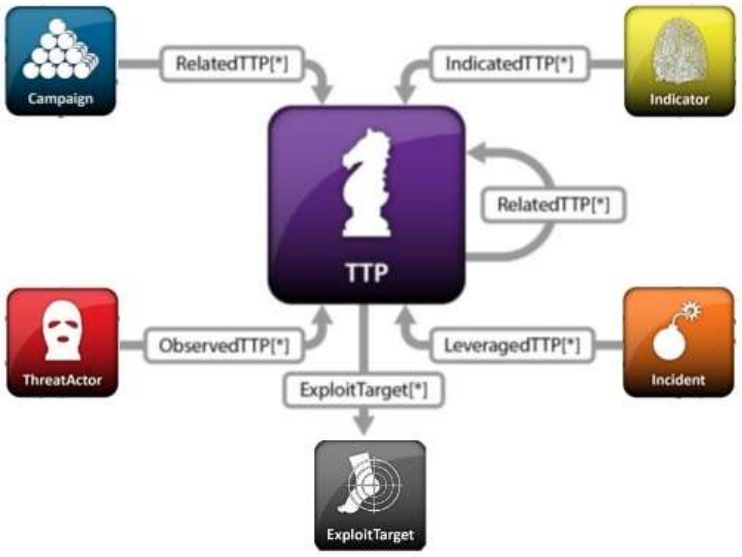


Figure 2-1. High level view of the Campaign data model

In this section, we give a high level summary of the relationship between the TTP data model and the Exploit Target data model to which a TTP may refer. We also make note of the fact that the TTP data model can be self-referential. Other relationships are defined in the specification of the component from which they originate.

- **TTP**

  The TTP data model is self-referential, enabling one TTP to reference other TTPs that are asserted to be related. Self-referential relationships between TTPs may indicate general associativity or can be used to indicate relationships between different versions of the same TTP.

- **Exploit Target**

  A STIX Exploit Target conveys information about a vulnerability, weakness, or misconfiguration in software, systems, networks, or configurations that may be targeted for exploitation by an adversary. Please see *STIX Version 1.2.1 Part 10: Exploit Target* for details.

  The TTP data model references the Exploit Target data model in order to identify possible targets for exploitation by the TTP.

3 STIX[TM] TTP Data Model

The primary class of the STIX TTP package is the `TTPType` class, which characterizes adversarial mode of operations (often referred to as the adversary's "Tactics, Techniques, and Procedures"). The `TTPType` class captures information that includes the victims targeted, the attack patterns and malware used, and the resources (infrastructure, tools, and personas) leveraged. Similar to the primary classes of all the component data models in STIX, the `TTPType` class extends a base class defined in the STIX Common data model; more specifically, it extends the `TTPBaseType` base class, which provides the essential identifier (`id`) and identifier reference (`idref`) properties.

The relationship between the `TTPType` class and the `TTPBaseType` base class, as well as the properties of the `TTPType` class, are illustrated in the UML diagram given in **Figure 3-1**.

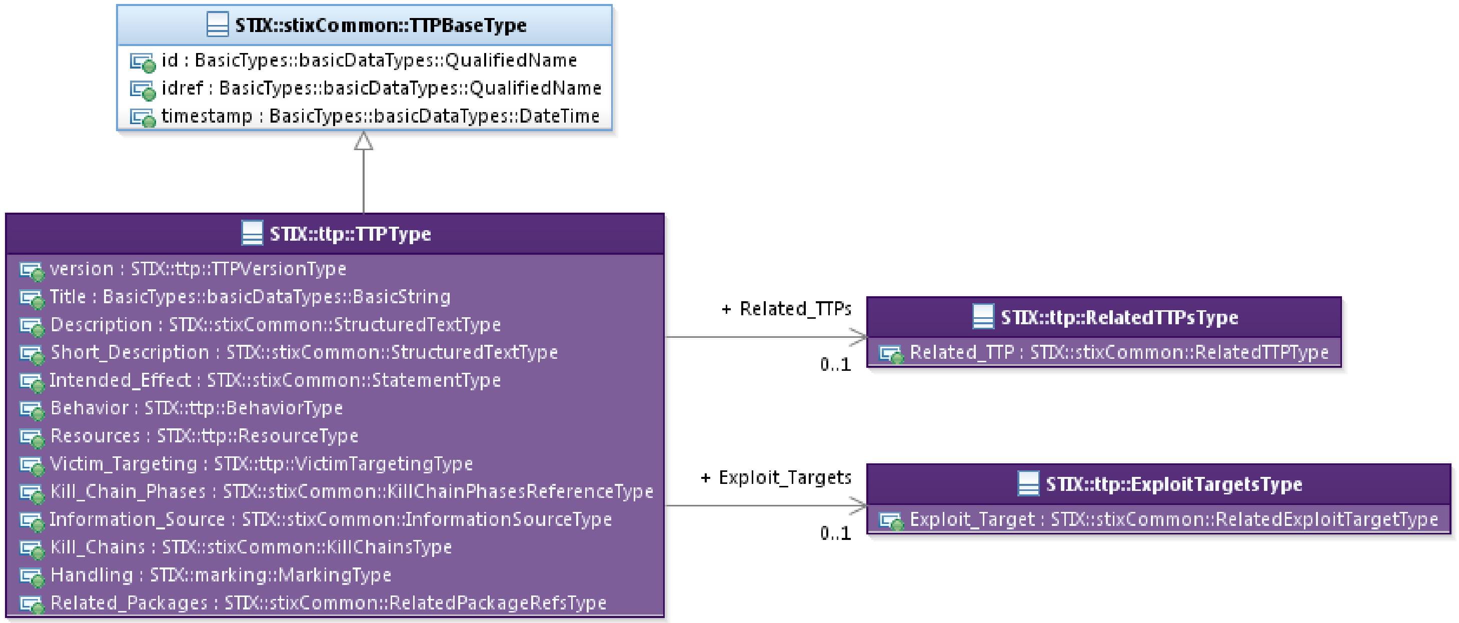

STIX::stixCommon::TTPBaseType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
timestamp : BasicTypes::basicDataTypes::DateTime
STIX::ttp::TTPType
version : STIX::ttp::TTPVersionType
Title : BasicTypes::basicDataTypes::BasicString
Description : STIX::stixCommon::StructuredTextType
Short_Description : STIX::stixCommon::StructuredTextType
Intended_Effect : STIX::stixCommon::StatementType
Behavior : STIX::ttp::BehaviorType
Resources : STIX::ttp::ResourceType
Victim_Targeting : STIX::ttp::VictimTargetingType
Kill_Chain_Phases : STIX::stixCommon::KillChainPhasesReferenceType
Information_Source : STIX::stixCommon::InformationSourceType
Kill_Chains : STIX::stixCommon::KillChainsType
Handling : STIX::marking::MarkingType
Related_Packages : STIX::stixCommon::RelatedPackageRefsType
+ Related_TTPs
0..1
STIX::ttp::RelatedTTPsType
Related_TTP : STIX::stixCommon::RelatedTTPType
+ Exploit_Targets
0..1
STIX::ttp::ExploitTargetsType
Exploit_Target : STIX::stixCommon::RelatedExploitTargetType

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

The property table, which includes property descriptions and corresonds to the UML Lmodel above, is given in **Table 3-1**.

All classes defined in the TTP data model are described in detail in Sections 3.1 through Section 3.6. Details are not provided for classes defined in non-TTP data models; instead, the reader is refered to the corresponding data model specification as indicated by the package prefix specified in the Type column of the table.

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

Name	Type	Multiplicity	Description
version	`TTPVersionType`	0..1	The `version` property specifies the version identifier of the STIX TTP data model for STIX v1.2.1 used to capture the information associated with the TTP.
Title	`basicDataTypes:BasicString`	0..1	The `Title` property captures a title for the TTP and reflects what the content producer thinks the TTP as a whole should be called. The `Title` property is typically used by humans to reference a particular TTP; however, it is not suggested for correlation.
Description	`stixCommon: StructuredTextType`	0..*	The `Description` property captures a textual description of the TTP. Any length is permitted. Optional formatting is supported via the `structuring_format` property of the `StructuredTextType` class.
Short_Description	`stixCommon: StructuredTextType`	0..*	The `Short_Description` property captures a short textual description of the TTP. This property is secondary and should only be used if the `Description` property is already populated and another, shorter description is available.
Intended_Effect	`stixCommon: StatementType`	0..*	The `Intended_Effect` property characterizes the suspected intended effect of the TTP, which includes a `Value` property that specifies the type of the effect. Examples of potential types include *theft*, *disruption*, and *unauthorized access* (these specific values are only provided to help explain the `Value` 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 `Value` property

			is '*IntendedEffectVocab-1.0*' (which is different than the default vocabulary provided for the `StatementType` class).
Behavior	`BehaviorType`	0..1	The `Behavior` property characterizes forms of adversarial behavior by capturing the attack patterns, malware, and/or exploits that the adversary may leverage.
Resources	`ResourceType`	0..1	The `Resources` property characterizes adversarial resources by capturing the tools, infrastructure, or personas that the adversary may leverage.
Victim_Targeting	`VictimTargetingType`	0..1	The `Victim_Targeting` property characterizes the sort of victims that an adversary may target including details of identity, systems, and/or information types targeted.
Exploit_Targets	`ExploitTargetsType`	0..1	The `Exploit_Targets` property specifies a set of one or more Exploit Targets potentially targeted by the TTP.
Related_TTPs	`RelatedTTPsType`	0..1	The `Related_TTPs` property specifies a set of one or more other TTPs related to this TTP.
Kill_Chain_Phases	`stixCommon: KillChainPhasesReferenceType`	0..1	A cyber kill chain is a phase-based model to describe the stages of an attack, and a cyber kill chain phase is an individual phase within a kill chain definition. The `Kill_Chain_Phases` property specifies a set of one or more kill chain phases (from one or more kill chains defined elsewhere) for which the TTP is asserted to be representative. The kill chain property is further defined in the STIX Common specification document.
Information_Source	`stixCommon: InformationSourceType`	0..1	The `Information_Source` property characterizes the source of the TTP information. Examples of details captured include identitifying characteristics, time-related attributes, and a list of the tools used to collect the information.
Kill_Chains	`stixCommon:KillChainsType`	0..1	A cyber kill chain is a phase-based model to describe the stages of an attack. The `Kill_Chains` property specifies a set of one or more specific kill chain definitions. The kill chain property is further defined in the STIX Common specification document.
Handling	`marking:MarkingType`	0..1	The `Handling` property specifies data handling markings for the properties of this TTP. The marking scope is limited to the TTP and the content is contains. Note that data handling markings

			can also be specified at a higher level.
Related_Packages	stixCommon: RelatedPackagesRefsType	0..1	The `Related_Packages` property specifies a set of one or more Packages for which the TTP may be relevant.

3.1 TTPVersionType Enumeration

The `TTPVersionType` enumeration is an inventory of all versions of the TTP data model for STIX Version 1.2.1. The enumeration literals are given in **Table 3-2**.

Table 3-2. Literals of the `TTPVersionType` enumeration

Enumeration Literal	Description
stix-1.2.1	TTP data model for STIX v1.2.1

3.2 BehaviorType Class

The `BehaviorType` class characterizes adversarial behavior by capturing details of cyber attack patterns, malware or exploits that the adversary may leverage.

The UML diagram corresponding to the `BehaviorType` class is shown in **Figure 3-2**.

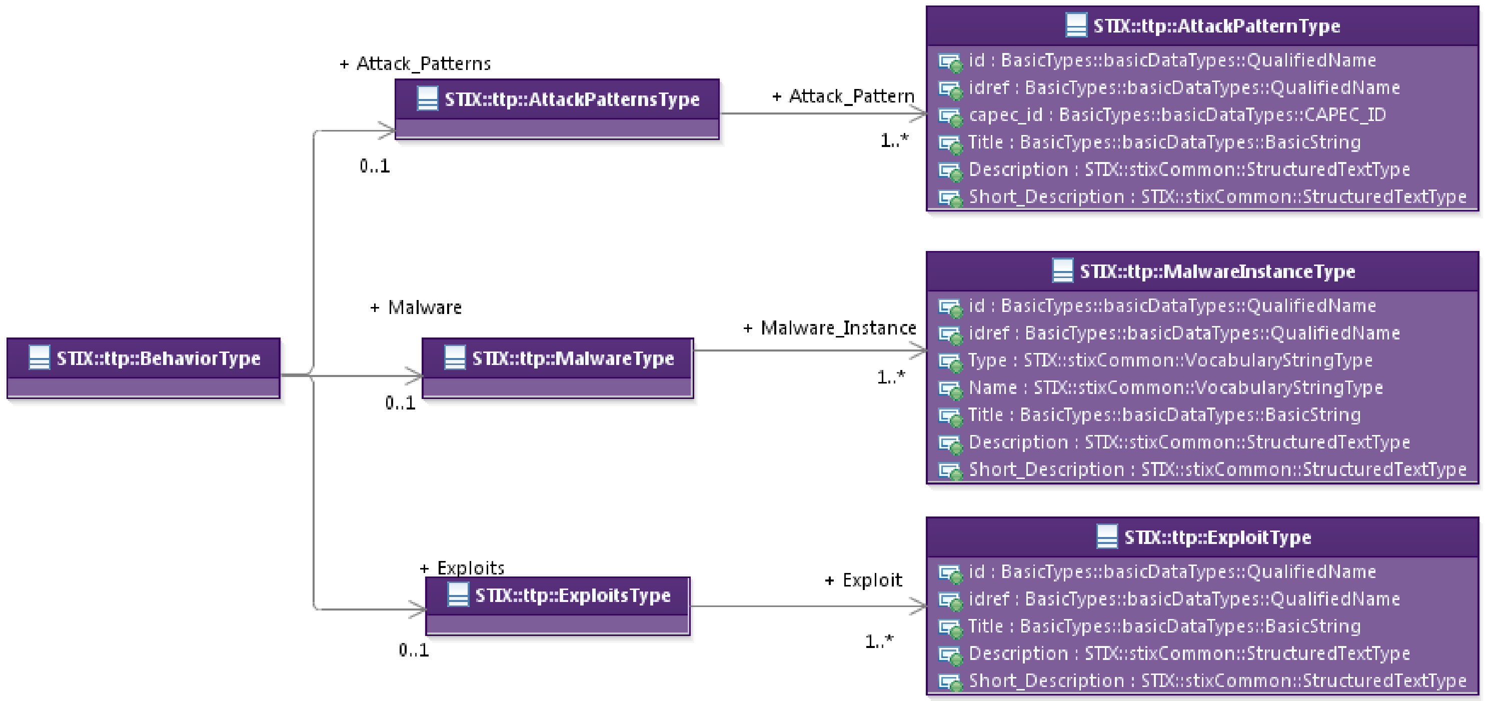

STIX::ttp::BehaviorType
+ Attack_Patterns
0..1
STIX::ttp::AttackPatternsType
+ Attack_Pattern
1..*
STIX::ttp::AttackPatternType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
capec_id : BasicTypes::basicDataTypes::CAPEC_ID
Title : BasicTypes::basicDataTypes::BasicString
Description : STIX::stixCommon::StructuredTextType
Short_Description : STIX::stixCommon::StructuredTextType
+ Malware
0..1
STIX::ttp::MalwareType
+ Malware_Instance
1..*
STIX::ttp::MalwareInstanceType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
Type : STIX::stixCommon::VocabularyStringType
Name : STIX::stixCommon::VocabularyStringType
Title : BasicTypes::basicDataTypes::BasicString
Description : STIX::stixCommon::StructuredTextType
Short_Description : STIX::stixCommon::StructuredTextType
+ Exploits
0..1
STIX::ttp::ExploitsType
+ Exploit
1..*
STIX::ttp::ExploitType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
Title : BasicTypes::basicDataTypes::BasicString
Description : STIX::stixCommon::StructuredTextType
Short_Description : STIX::stixCommon::StructuredTextType

Figure 3-2. UML diagram of the `BehaviorType` class

The property table given in **Table 3-3** corresponds to the UML diagram shown in **Figure 3-2**, and the associated classes defining the property types are discussed in Sections **3.2.1** through **3.2.3**.

Table 3-3. Properties of the `BehaviorType` class

Name	Type	Multiplicity	Description
Attack_Patterns	`AttackPatternsType`	0..1	The `Attack_Patterns` property specifies a set of one or more attack patterns that an adversary may leverage.
Malware	`MalwareType`	0..1	The `Malware` property specifies a set of one or more instances of malware that an adversary may leverage.

Exploits	`ExploitsType`	0..1	The `Exploits` property specifies a set of one or more exploits that an adversary may leverage.

3.2.1 AttackPatternsType Class

The `AttackPatternsType` class specifies a set of one or more attack patterns that an adversary may leverage.

The property of the `AttackPatternsType` class is shown in **Table 3-4**.

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

Name	**Type**	**Multiplicity**	**Description**
Attack_Pattern	`AttackPatternType`	1..*	The `Attack_Pattern` property specifies a single Attack Pattern that an adversary may leverage.

3.2.1.1 AttackPatternType Class

The `AttackPatternType` class characterizes an individual attack pattern[2] through the capture of information such as a textual description and a Common Attack Pattern Enumeration and Classification (CAPEC) reference. The `AttackPatternType` class is intended to be extended as appropriate to enable the structured description of an attack pattern. STIX v1.2.1 defines a default extension to the `AttackPatternType` class to leverage the Common Attack Pattern Enumeration and Classification (CAPEC) data model.

The UML diagram corresponding to the `SuggestedCOAsType` class is shown in **Figure 3-3**. Please see *STIX Version 1.2.1 Part 12: Default Extensions* for extension-related details.

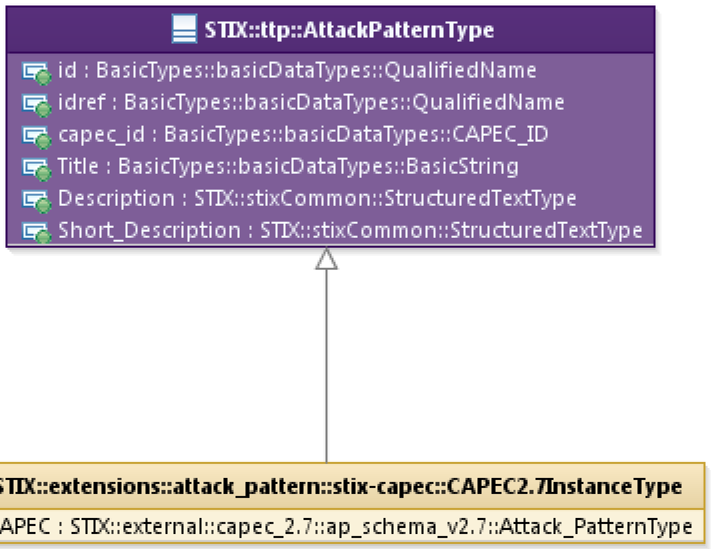

STIX::ttp::AttackPatternType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
capec_id : BasicTypes::basicDataTypes::CAPEC_ID
Title : BasicTypes::basicDataTypes::BasicString
Description : STIX::stixCommon::StructuredTextType
Short_Description : STIX::stixCommon::StructuredTextType
STIX::extensions::attack_pattern::stix-capec::CAPEC2.7InstanceType
CAPEC : STIX::external::capec_2.7::ap_schema_v2.7::Attack_PatternType

Figure 3-3. UML diagram of the `AttackPatternType` class

The property table given in **Table 3-5** corresponds to the UML diagram given in **Figure 3-3**.

Table 3-5. Properties of the `AttackPatternType` class

Name	Type	Multiplicity	Description
id	`basicDataTypes:QualifiedName`	0..1	The `id` property specifies a globally unique identifier for the attack pattern.
idref	`basicDataTypes:QualifiedName`	0..1	The `idref` property specifies an identifier reference to an attack pattern specified elsewhere. When the `idref` property is used, the `id` property MUST NOT also be specified and the other properties of the `AttackPatternType` class SHOULD NOT hold any content.
capec_id	`basicDataTypes:CAPEC_ID`	0..1	The `capec_id` property specifies a particular attack pattern (via identifier) in the Common Attack Pattern

			Enumeration and Classification (CAPEC) registry.
Title	basicDataTypes:BasicString	0..1	The Title property captures a title for the attack pattern and reflects what the content producer thinks the attack pattern as a whole should be called. The Title property is typically used by humans to reference a particular attack pattern; however, it is not suggested for correlation.
Description	stixCommon:StructuredTextType	0..*	The Description property captures a textual description of the attack pattern. Any length is permitted. Optional formatting is supported via the structuring_format property of the StructuredTextType class.
Short_Description	stixCommon:StructuredTextType	0..*	The Short_Description property captures a short textual description of the attack pattern. This property is secondary and should only be used if the Description property is already populated and another, shorter description is available.

3.2.2 MalwareType Class

The MalwareType class characterizes a set of one or more malware instances that an adversary may leverage.

The property of the MalwareType class is shown in **Table 3-6**.

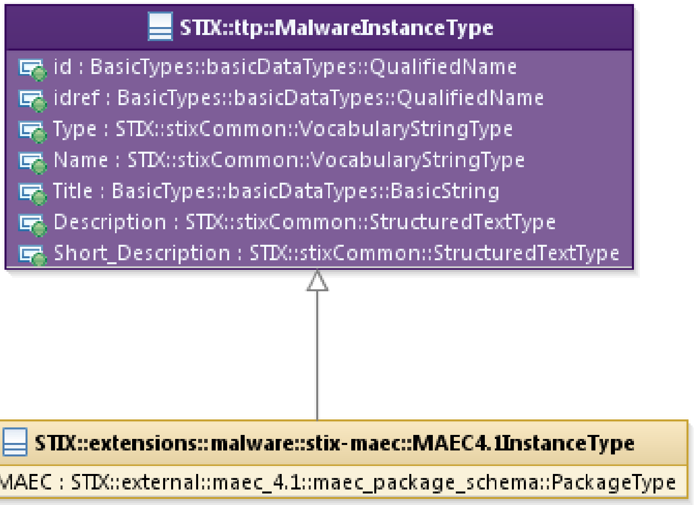
Table 3-6. Properties of the MalwareType class

Name	Type	Multiplicity	Description
Malware_Instance	MalwareInstanceType	1..*	The Malware_Instance property characterizes a single malware instance that an adversary may leverage.

1.1.1.1 MalwareInstanceType Class

The MalwareInstanceType class characterizes a malware instance through the capture of basic information such as the type, name, and description of the malware. A malware instance may characterize anything from a specific malware sample to an entire family. The MalwareInstanceType class is intended to be extended as appropriate to enable the structured description of a malware instance. STIX v1.2.1 defines a default extension to the MalwareInstanceType class to leverage the Malware Attribute Enumeration and Classification (MAEC) data model.

The UML diagram corresponding to the `MalwareInstanceType` class is shown in **Figure 3-4**. Please see *STIX Version 1.2.1 Part 12: Default Extensions* for extension-related details.



STIX::ttp::MalwareInstanceType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
Type : STIX::stixCommon::VocabularyStringType
Name : STIX::stixCommon::VocabularyStringType
Title : BasicTypes::basicDataTypes::BasicString
Description : STIX::stixCommon::StructuredTextType
Short_Description : STIX::stixCommon::StructuredTextType
STIX::extensions::malware::stix-maec::MAEC4.1InstanceType
MAEC : STIX::external::maec_4.1::maec_package_schema::PackageType

Figure 3-4. UML diagram of the `MalwareInstanceType` class

The property table given in **Table 3-7** corresponds to the UML diagram given in **Figure 3-4**.

Table 3-7. Properties of the `MalwareInstanceType` class

Name	**Type**	**Multiplicity**	**Description**
id	`basicDataTypes:QualifiedName`	0..1	The `id` property specifies a globally unique identifier for the malware instance.
idref	`basicDataTypes:QualifiedName`	0..1	The `idref` property specifies an identifier reference to a malware instance specified elsewhere. When the `idref` property is used, the `id` property MUST NOT also be specified and the other properties of the `MalwareInstanceType` class SHOULD NOT hold any content.
Type	`stixCommon:VocabularyStringType`	0..*	The `Type` property specifies the type of the malware instance being characterized. Examples of potential types include *bot, exploit kit*, and *ransomware* (these specific values are only provided to help explain the property: they are neither recommended types 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 '*MalwareTypeVocab-1.0.*'
Name	`stixCommon: VocabularyStringType`	0..*	The `Name` property is used to specify a single name or alias that identifies the malware instance. The content creator may choose any arbitrary name or may constrain the set of possible names by referencing an externally-defined vocabulary or leveraging a formally defined vocabulary extending from the `stixCommon:ControlledVocabularyStringType` class. No default vocabulary class for use in the property has been defined for STIX 1.2.
Title	`basicDataTypes: BasicString`	0..1	The `Title` property captures a title for the malware instance and reflects what the content producer thinks the malware instance as a whole should be called. The `Title` property is typically used by humans to reference a particular malware instance; however, it is not suggested for correlation.
Description	`stixCommon: StructuredTextType`	0..*	The `Description` property captures a textual description of the malware instance. Any length is permitted. Optional formatting is supported via the `structuring_format` property of the `StructuredTextType` class.
Short_Description	`stixCommon: StructuredTextType`	0..*	The `Short_Description` property captures a short textual description of the malware instance. This property is secondary and should only be used if the `Description` property is already populated and another, shorter description is available.

3.2.3 ExploitsType Class

The `ExploitsType` class specifies a set of one or more exploits that an adversary may leverage.

The property of the `ExploitsType` class is shown in **Table 3-8**.

Table 3-8. Properties of the `ExploitsType` class

Name	Type	Multiplicity	Description
Exploit	`ExploitType`	1..*	The `Exploit` property specifies a single exploit that an adversary may leverage.

3.2.3.1 ExploitType Class

The `ExploitType` class characterizes an individual exploit instance through the capture of basic information such as the title and description of the exploit. The `ExploitType` class is intended to be extended to enable the structured description of an exploit instance. However, no extension is provided by STIX v 1.2; producers wanting to represent structured exploit instance information are encouraged to develop such an extension.

The properties of the `ExploitType` class are shown in **Table 3-9**.

Table 3-9. Properties of the `ExploitType` class

Name	Type	Multiplicity	Description
id	`basicDataTypes:BasicString`	0..1	The `id` property specifies a globally unique identifier for the exploit instance.
idref	`basicDataTypes:BasicString`	0..1	The `idref` property specifies an identifier reference to an exploit instance specified elsewhere. When the `idref` property is used, the `id` property MUST NOT also be specified and the other properties of the `ExploitType` class SHOULD NOT hold any content.
Title	`basicDataTypes:BasicString`	0..1	The `Title` property captures a title for the exploit instance and reflects what the content producer thinks the exploit instance as a whole should be called. The `Title` property is typically used by humans to reference a particular exploit instance; however, it is not suggested for correlation.
Description	`stixCommon:StructuredTextType`	0..*	The `Description` property captures a textual description of the exploit instance. Any length is permitted. Optional formatting is supported via the `structuring_format` property of the `StructuredTextType` class.
Short_Description	`stixCommon:StructuredTextType`	0..*	The `Short_Description` property captures a short textual description of the exploit instance. This property is secondary and should only be used if the `Description` property is already populated and another, shorter description is available.

3.3 ResourceType Class

The `ResourceType` class characterizes resources the adversary may leverage.

The UML diagram corresponding to the `ResourceType` class is shown in **Figure 3-5**.

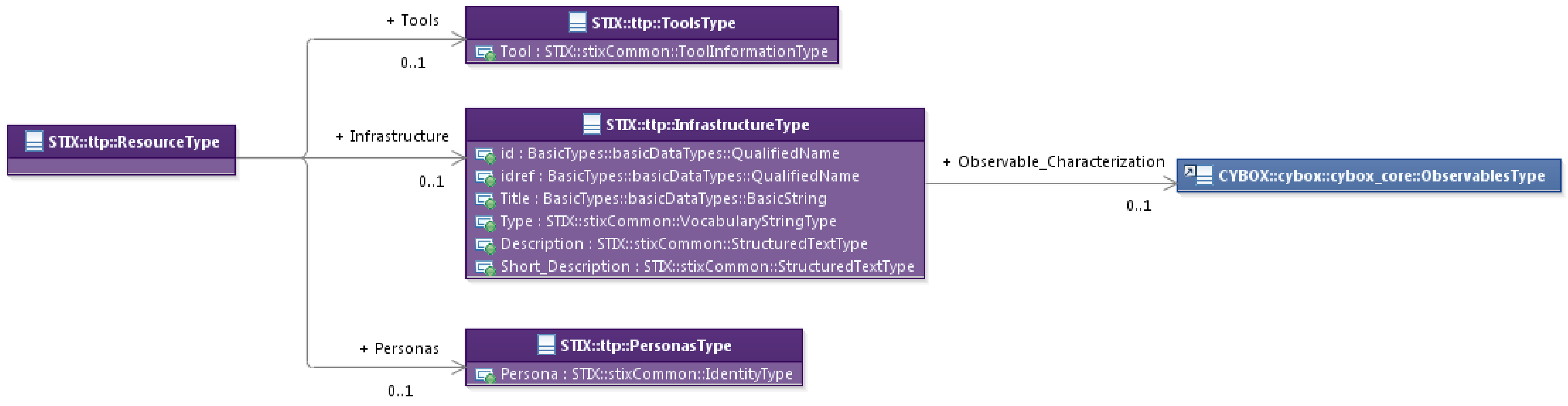

STIX::ttp::ResourceType
+ Tools
0..1
STIX::ttp::ToolsType
Tool : STIX::stixCommon::ToolInformationType
+ Infrastructure
0..1
STIX::ttp::InfrastructureType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
Title : BasicTypes::basicDataTypes::BasicString
Type : STIX::stixCommon::VocabularyStringType
Description : STIX::stixCommon::StructuredTextType
Short_Description : STIX::stixCommon::StructuredTextType
+ Observable_Characterization
0..1
CYBOX::cybox::cybox_core::ObservablesType
+ Personas
0..1
STIX::ttp::PersonasType
Persona : STIX::stixCommon::IdentityType

Figure 3-5. UML diagram of the `ResourceType` class

The property table given in **Table 3-10** corresponds to the UML diagram given in **Figure 3-5**.

Table 3-10. Properties of the `ResourceType` class

Name	**Type**	**Multiplicity**	**Description**
Tools	`ToolsType`	0..1	The `Tools` property specifies a set of one or more tools that an adversary may leverage.
Infrastructure	`InfrastructureType`	0..1	The `Infrastructure` property characterizes infrastructure that an adversary may leverage.
Personas	`PersonasType`	0..1	The `Personas` property specifies a set of one or more personas that an adversary may leverage. Different personas are often used as a method of masquerade.

3.3.1 ToolsType Class

The ToolsType class specifies a set of one or more tools that an adversary may leverage. Tools specified may cover a wide range of types (DDOS tools, exploit kits, packers, communications tools, etc.). While ToolsType may be appropriate for characterizing the use of a particular malware as an attack tool including details of specific version or configuration, it is not appropriate for characterizing the structure or behavior of malware which is more appropriately characterized using MalwareInstanceType.

The property of the ToolsType class is shown in **Table 3-11**.

Table 3-11. Properties of the ToolsType class

Name	Type	Multiplicity	Description
Tool	stixCommon: ToolInformationType	1..*	The Tool property characterizes a single adversarial tool. Note that the STIX Common ToolInformationType class includes a Type property that specifies the type of the tool. Examples of potential tool types include *pentester*, *port scanner*, and *password cracker* (these specific values are only provided to help explain the Type property: they are neither recommended types 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 Type property is '*AttackerToolTypeVocab-1.0.*'

3.3.2 InfrastructureType Class

The InfrastructureType class characterizes adversarial infrastructure that an adversary may leverage.

Properties of the InfrastructureType class are shown in **Table 3-12**.

Table 3-12. Properties of the `InfrastructureType` class

Name	Type	Multiplicity	Description
id	`basicDataTypes:BasicString`	0..1	The `id` property specifies a globally unique identifier for the infrastructure.
idref	`basicDataTypes:BasicString`	0..1	The `idref` property specifies an identifier reference to an infrastructure specified elsewhere. When the `idref` property is used, the `id` property MUST NOT also be specified and the other properties of the `InfrastructureType` class SHOULD NOT hold any content.
Title	`basicDataTypes:BasicString`	0..1	The `Title` property captures a title for the infrastructure and reflects what the content producer thinks the infrastructure as a whole should be called. The `Title` property is typically used by humans to reference a particular infrastructure; however, it is not suggested for correlation.
Type	`stixCommon:VocabularyStringType`	0..*	The `Type` property specifies the type of infrastructure being characterized. Examples of potential types include *anonymization*, *domain registration*, and *hosting* (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 '*AttackerInfrastructureTypeVocab-1.0.*'
Description	`stixCommon:StructuredTextType`	0..*	The `Description` property captures a textual description of the infrastructure. Any length is permitted. Optional formatting is supported via the `structuring_format` property of the `StructuredTextType` class.
Short_Description	`stixCommon:StructuredTextType`	0..*	The `Short_Description` property captures a short textual description of the infrastructure. This property is secondary and should only be used if the `Description` property is

			already populated and another, shorter description is available.
Observable_Characterization	`cybox:ObservablesType`	0..1	The `Observable_Characterization` property characterizes the adversarial infrastructure through specification of a structured cyber Observables pattern.

3.3.3 PersonasType Class

The `PersonasType` class specifies a set of one or more personas that an adversary may leverage.

The property of the `PersonasType` class is shown in **Table 3-13**.

Table 3-13. Properties of the `PersonasType` class

Name	Type	Multiplicity	Description
Persona	`stixCommon:IdentityType`	1..*	The `Persona` property characterizes a persona identity potentially used in malicious activity. Personas are typically used to masquerade as another party. For situations calling for more than a simple name, the underlying class may be extended using a more complete structure such as the `CIQIdentity3.0InstanceType` subclass as defined in *STIX Version 1.2.1 Part 12: Default Extensions*.

3.4 VictimTargetingType Class

The `VictimTargetingType` class characterizes victim targeting information by capturing information about the people, organizations, systems and/or data potentially targeted by the adversary.

The UML diagram corresponding to the `VictimTargetingType` class is shown in **Figure 3-6**.

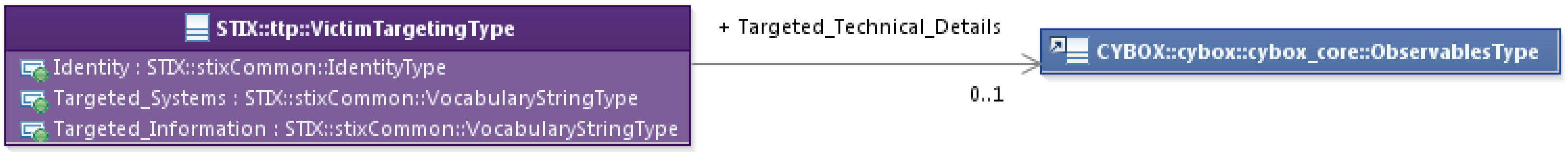

STIX::ttp::VictimTargetingType
Identity : STIX::stixCommon::IdentityType
Targeted_Systems : STIX::stixCommon::VocabularyStringType
Targeted_Information : STIX::stixCommon::VocabularyStringType
+ Targeted_Technical_Details
0..1
CYBOX::cybox::cybox_core::ObservablesType

Figure 3-6. UML diagram of the `VictimTargetingType` class

The property table given in **Table 3-14** corresponds to the UML diagram given in **Figure 3-6**.

Table 3-14. Properties of the `VictimTargetingType` class

Name	Type	Multiplicity	Description
Identity	`stixCommon:` `IdentityType`	0..1	The `Identity` property characterizes traits common to the people or organizations that are targeted. For situations calling for more than a simple name, the underlying class may be extended using a more complete structure such as the `CIQIdentity3.0InstanceType` subclass as defined in *STIX Version 1.2.1 Part 12: Default Extensions*.
Targeted_Systems	`stixCommon:` `VocabularyStringType`	0..*	The `Targeted_Systems` property specifies a type of system that may be targeted by the adversary. Examples of potential types include *web layer, third-party services,* and *user workstations* (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 '*SystemTypeVocab-1.0*.'
Targeted_Information	`stixCommon:` `VocabularyStringType`	0..*	The `Targeted_Information` property specifies a type of information that may be targeted by the adversary. Examples of potential types include *customer PII*, *mobile phone contacts*, and *authentication cookies* (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 '*InformationTypeVocab-1.0.*'
Targeted_Technical_Details	`cybox:ObservablesType`	0..1	The `Targeted_Technical_Details` property characterizes details of specific technologies targeted by the adversary. It is implemented through specification of a structured cyber Observables pattern using the CybOX `ObservablesType` class.

3.5 ExploitTargetsType Class

The `ExploitTargetsType` class specifies a set of one or more Exploit Targets potentially targeted by the TTP. It extends the `GenericRelationShipListType` superclass defined in the STIX Common data model, which specifies the scope (whether the elements of the set are related individually or as a group).

The UML diagram corresponding to the `ExploitTargetsType` class is shown in **Figure 3-7**.

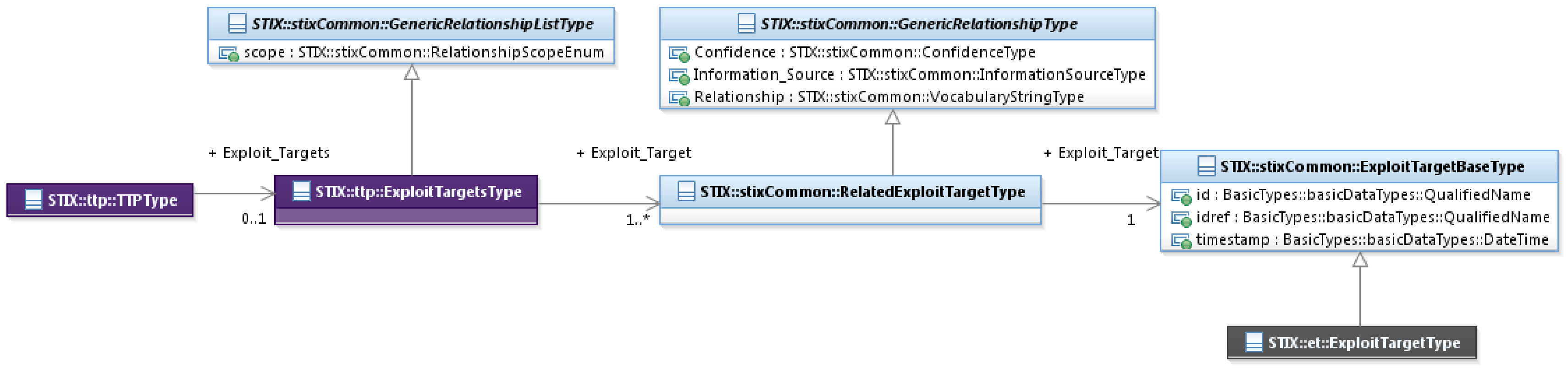

STIX::stixCommon::GenericRelationshipListType
scope : STIX::stixCommon::RelationshipScopeEnum
STIX::stixCommon::GenericRelationshipType
Confidence : STIX::stixCommon::ConfidenceType
Information_Source : STIX::stixCommon::InformationSourceType
Relationship : STIX::stixCommon::VocabularyStringType
+ Exploit_Targets
+ Exploit_Target
+ Exploit_Target
STIX::ttp::TTPType
0..1
STIX::ttp::ExploitTargetsType
1..*
STIX::stixCommon::RelatedExploitTargetType
1
STIX::stixCommon::ExploitTargetBaseType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
timestamp : BasicTypes::basicDataTypes::DateTime
STIX::et::ExploitTargetType

Figure 3-7. UML diagram of the `ExploitTargetsType` class

The property table given in **Table 3-15** corresponds to the UML diagram shown in **Figure 3-7**.

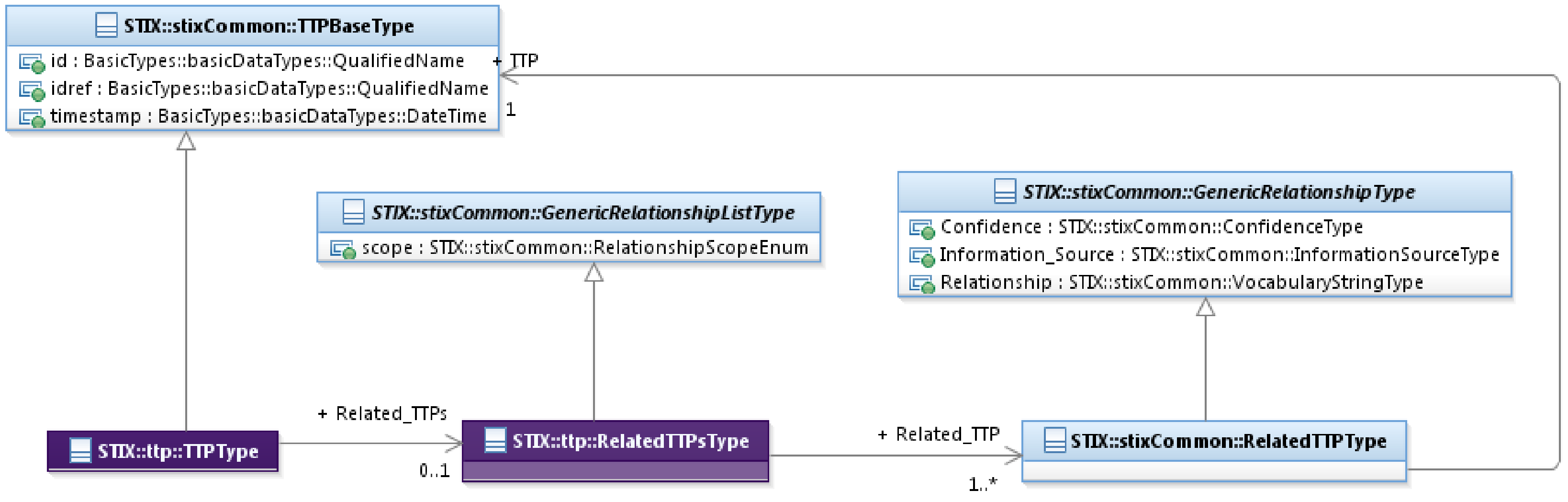
Table 3-15. Properties of the `ExploitTargetsType` class

Name	Type	Multiplicity	Description
Exploit_Target	`stixCommon: RelatedExploitTargetType`	1..*	The `Exploit_Target` property specifies an Exploit Target potentially targetd by the TTP and characterizes the relationship between the Exploit Target and the TTP by capturing information such as the level of confidence that the Exploit Target and the TTP are related, the source of the relationship information, and the type of relationship.

3.6 RelatedTTPsType Class

The `RelatedTTPsType` class specifies a set of one or more other TTPs asserted to be related to this TTP and therefore is a self-referential relationship. It extends the `GenericRelationshipListType` superclass defined in the STIX Common data model, which specifies the scope (whether the elements of the set are related individually or as a group).

The UML diagram corresponding to the `RelatedTTPsType` class is shown in **Figure 3-8**.



STIX::stixCommon::TTPBaseType
id : BasicTypes::basicDataTypes::QualifiedName
idref : BasicTypes::basicDataTypes::QualifiedName
timestamp : BasicTypes::basicDataTypes::DateTime
+ TTP
1
STIX::stixCommon::GenericRelationshipListType
scope : STIX::stixCommon::RelationshipScopeEnum
STIX::stixCommon::GenericRelationshipType
Confidence : STIX::stixCommon::ConfidenceType
Information_Source : STIX::stixCommon::InformationSourceType
Relationship : STIX::stixCommon::VocabularyStringType
+ Related_TTPs
0..1
STIX::ttp::TTPType
STIX::ttp::RelatedTTPsType
+ Related_TTP
1..*
STIX::stixCommon::RelatedTTPType

Figure 3-8. UML diagram of the `RelatedTTPsType` class

The property table given in **Table 3-16** corresponds to the UML diagram shown in **Figure 3-8**.

Table 3-16. Properties of the `RelatedTTPsType` class

Name	Type	Multiplicity	Description
Related_TTP	`stixCommon:RelatedTTPType`	1..*	The `Related_TTP` property specifies another TTP associated with this TTP and characterizes the relationship between the TTPs by capturing information such as the level of confidence that the TTPs are related, the source of the relationship information, and type of the relationship. A relationship between TTPs may represent assertions of general associativity or different versions of the same TTP.

4 Conformance

Implementations have discretion over which parts (components, properties, extensions, controlled vocabularies, etc.) of STIX they implement (e.g., Indicator/Suggested_COAs).

[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 STIX they implement (e.g., Implementers of the entire TTP component must conform to all normative structural specifications of the UML model or additional normative statements within this document regarding the TTP component).

[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 STIX they implement (e.g., Non-implementers of any particular properties of the TTP component are free to ignore all normative structural specifications of the UML model or additional normative statements within this document regarding those properties of the TTP component).

The conformance section of this document is intentionally broad and attempts to reiterate what already exists in this document. The STIX 1.2 Specifications, which this specification is based on, did not have a conformance section. Instead, the STIX 1.2 Specifications relied on normative statements and the non-mandatory implementation of STIX profiles. STIX 1.2.1 represents a minimal change from STIX 1.2, and in that spirit no requirements have been added, modified, or removed by this section.

Appendix A. 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)
Jyoti Verma, Cisco Systems
Liron Schiff, Comilion (mobile) Ltd.
Jane Ginn, Cyber Threat Intelligence Network, Inc. (CTIN)
Richard Struse, DHS Office of Cybersecurity and Communications (CS&C)
Marlon Taylor, DHS Office of Cybersecurity and Communications (CS&C)
David Eilken, Financial Services Information Sharing and Analysis Center (FS-ISAC)
Sarah Brown, Fox-IT
Ryusuke Masuoka, Fujitsu Limited
Eric Burger, Georgetown University
Jason Keirstead, IBM
Paul Martini, iboss, Inc.
Jerome Athias, 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
Raymon van der Velde, Intelworks BV
Jonathan Baker, MITRE Corporation
Sean Barnum, MITRE Corporation
Desiree Beck, MITRE Corporation
Mark Davidson, MITRE Corporation
Ivan Kirillov, MITRE Corporation
Jon Salwen, 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.
Daniel Riedel, New Context Services, Inc.
Andrew Storms, New Context Services, Inc.
John Tolbert, Queralt, Inc.
Igor Baikalov, Securonix
Bernd Grobauer, Siemens AG
Jonathan Bush, Soltra
Aharon Chernin, Soltra
Trey Darley, Soltra
Paul Dion, Soltra
Ali Khan, Soltra
Natalie Suarez, Soltra
Cedric LeRoux, Splunk Inc.
Brian Luger, Splunk Inc.

Crystal Hayes, The Boeing Company
Brad Butts, U.S. Bank
Mona Magathan, U.S. Bank
Adam Cooper, United Kingdom Cabinet Office
Mike McLellan, United Kingdom Cabinet Office
Chris O'Brien, United Kingdom Cabinet Office
Julian White, United Kingdom Cabinet Office
Anthony Rutkowski, Yaana Technologies, LLC

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

Appendix B. Revision History

Revision	Date	Editor	Changes Made
wd01	21 August 2015	Sean Barnum Desiree Beck Aharon Chernin Rich Piazza	Initial transfer to OASIS template

Notes

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

[2] Attack Patterns are descriptions of common elements, approaches and techniques used in attacks against vulnerable cyber-enabled capabilities.