TTA Standard

정보통신단체표준(영문표준)

TTAx.xx-xx.xxxx/R1 제정일: 2018년 xx월 xx일

구조화된 위협 정보 표현 규격(STIX^{TM}) 버전 2.0 - 제5부: STIX 패터닝

Structured Threat Information eXpression(STIX^{TM}) Version 2.0 - Part5: STIX Patterning

한국정보통신기술협회
Telecommunications Technology Association

표준초안 검토 위원회 사이버보안 프로젝트그룹(PG503)

표준안 심의 위원회 정보보호 기술위원회(TC5)

	성명	소 속	직위	위원회 및 직위	표준번호
표준(과제) 제안	김종현	한국전자통신연구원	책임연구원	사이버보안 프로젝트 그룹 위원	미정
	박성민	한국인터넷진흥원	선임연구원	위원	미정
	김낙현	한국인터넷진흥원	선임연구원	위원	미정
	이철호	국가보안기술 연구소	책임연구원	위원	미정
표준 초안 작성자	김종현	한국전자통신연구원	책임연구원	사이버보안 프로젝트 그룹 위원	미정
	박성민	한국인터넷진흥원	선임연구원	위원	미정
	김낙현	한국인터넷진흥원	선임연구원	위원	미정
	이철호	국가보안기술 연구소	책임연구원	위원	미정
	염흥열	순천향대학교	교수	위원	미정
	김익균	한국전자통신연구원	책임연구원	위원	미정
사무국 담당	박수정	TTA	책임연구원	-	

본 표준 발간 이전에 접수된 지식재산권 확약서 정보는 본 표준의 '부록(지식재산권 확약서 정보)'에 명시하고 있으며, 이후 접수된 지식재산권 확약서는 TTA 웹사이트에서 확인할 수 있습니다.
본 표준과 관련하여 접수된 확약서 외의 지식재산권이 존재할 수 있습니다.

발행인 : 한국정보통신기술협회 회장
발행처 : 한국정보통신기술협회
13591, 경기도 성남시 분당구 분당로 47
Tel : 031-724-0114, Fax : 031-724-0109
발행일 : 2018.xx

서 문

1 표준의 목적

이 표준은 STIX 인디케이터에서 사용할 수 있는 STIX 패터닝 언어를 정의한다. STIX 패터닝 언어는 네트워크와 엔드포인트에서 악의적인 행위를 탐지하기 위한 표준이다.

2 주요 내용 요약

STIX 패터닝 언어는 네트워크와 엔드포인트의 악의적인 행위를 탐지하기 위한 방법을 정의하며, 관련된 용어의 설명과 네이밍 요구사항 (속성 이름 및 문자열 리터럴, 예약어), 문서 규칙(네이밍 규칙, 폰트 색상 및 스타일), 상수, STIX 패턴 및 패턴의 표현(관측 표현식 한정자, 관측 연산자, 연산자 우선순위, 비교 연산자, 문자열 비교, 바이너리 형식 비교, 네이티브 형식 비교), 객체 경로 구문에 관한 내용을 예시와 함께 다룬다. 또한 STIX 패터닝 표준을 준수하는 수준에 따라 세 단계(기본 적합성, 기본 적합성 및 관측 연산자, 완전 적합성)로 구분된 적합성의 평가 기준을 정의한다.

3 인용 표준과의 비교

3.1 인용 표준과의 관련성

이 표준은 인용 표준(STIX™ Version 2.0. Part 5: STIX Patterning)을 영문 그대로 완전 수용하는 표준이다.

3.2 인용 표준과 본 표준의 비교표

TTAE.xx-xx.xxxx	STIX™ Version 2.0. Part 5: Patterning	비고
1. 소개	1. Introduction	동일
2. 정의	2. Definitions	동일
3. STIX 패턴	3. STIX Patterns	동일
4. 패턴 표현식	4. Pattern Expressions	동일
5. 객체 경로 구문	5. Object Path Syntax	동일
6. 예시	6. Examples	동일
7. 적합성	7. Conformance	동일
부속서 A. 용어 사전	Appendix A. Glossary	동일
부속서 B. 감사의 글	Appendix B. Acknowledgements	동일
부속서 C. 개정이력	Appendix C. Revision History	동일

Preface

1 Purpose

This standard defines STIX patterning language to support STIX indicators. The STIX Patterning language is a standard for detecting malicious behavior on networks and endpoints.

2 Summary

The STIX patterning language defines methods for detecting malicious behavior on networks and endpoints, which includes descriptions of the terminology, naming requirements(property names and string literals, reserved names), document conventions(naming conventions, font colors and style), constants, STIX patterns, pattern expressions(observation expression qualifiers, observation operators, operator precedence, comparison operators, string comparison, binary type comparison, native format comparison), and object path syntax. It also defines three levels of conformance(basic conformance, basic conformance + observation operators, full conformance) based on the level of compliance with the STIX patterning standard.

3 Relationship to Reference Standards

3.1 The relationship of international standards

This standard is fully equivalent to STIX™ Version 2.0. Part 5: STIX Patterning.

3.2 Differences between International standards(recommendation) and this standard

TTAE.xx-xx.xxxx	STIX^{TM} Version 2.0. Part 5: Patterning	비고
1. Introduction	1. Introduction	Equals
2. Definitions	2. Definitions	Equals
3. STIX Patterns	3. STIX Patterns	Equals
4. Pattern Expressions	4. Pattern Expressions	Equals
5. Object Path Syntax	5. Object Path Syntax	Equals
6. Examples	6. Examples	Equals
7. Conformance	7. Conformance	Equals
Appendix A. Glossary	Appendix A. Glossary	Equals
Appendix B. Acknowledgements	Appendix B. Acknowledgements	Equals
Appendix C. Revision History	Appendix C. Revision History	Equals

목 차

STIX™Version2.0.Part5:STIX Patterning

Committee Specification 01

19 July 2017

Specification URIs

This version:

http://docs.oasis-open.org/cti/stix/v2.0/cs01/part5-stix-patterning/stix-v2.0-cs01-part5-stix-patterning.docx (Authoritative)
http://docs.oasis-open.org/cti/stix/v2.0/cs01/part5-stix-patterning/stix-v2.0-cs01-part5-stix-patterning.html
http://docs.oasis-open.org/cti/stix/v2.0/cs01/part5-stix-patterning/stix-v2.0-cs01-part5-stix-patterning.pdf

Previous version:

http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part5-stix-patterning/stix-v2.0-csprd02-part5-stix-patterning.docx (Authoritative)
http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part5-stix-patterning/stix-v2.0-csprd02-part5-stix-patterning.html
http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part5-stix-patterning/stix-v2.0-csprd02-part5-stix-patterning.pdf

Latest version:

http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part5-stix-patterning.docx (Authoritative)
http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part5-stix-patterning.html
http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part5-stix-patterning.pdf

Technical Committee:

OASIS Cyber Threat Intelligence(CTI) TC

Chair:

Richard Struse (Richard.Struse@HQ.DHS.GOV), DHSOfficeofCybersecurityandCommunications(CS&C)

Editors:

Trey Darley (trey@kingfisherops.com), KingfisherOperations,sprl
Ivan Kirillov (ikirillov@mitre.org), MITRECorporation

Additional artifacts:

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

- *STIX™ Version 2.0. Part 1: STIX Core Concepts.* http://docs.oasis-open.org/cti/stix/v2.0/cs01/part1-stix-core/stix-v2.0-cs01-part1-stix-core.html.
- *STIX™ Version 2.0. Part 2: STIX Objects.* http://docs.oasis-open.org/cti/stix/v2.0/cs01/part2-stix-objects/stix-v2.0-cs01-part2-stix-objects.html.
- *STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts.* http://docs.oasis-open.org/cti/stix/v2.0/cs01/part3-cyber-observable-core/stix-v2.0-cs01-part3-cyber-observable-core.html.
- *STIX™ Version 2.0. Part 4: Cyber Observable Objects.* http://docs.oasis-open.org/cti/stix/v2.0/cs01/part4-cyber-observable-objects/stix-v2.0-cs01-part4-cyber-observable-objects.html.
- (this document) *STIX™ Version 2.0. Part 5: STIX Patterning.* http://docs.oasis-open.org/cti/stix/v2.0/cs01/part5-stix-patterning/stix-v2.0-cs01-part5-stix-patterning.html.

Related work:

This specification replaces or supersedes:

- *STIX™ Version 1.2.1. Part 1: Overview.* Edited by Sean Barnum, Desiree Beck, Aharon Chernin, and Rich Piazza. Latest version: http://docs.oasis-open.org/cti/stix/v1.2.1/stix-v1.2.1-part1-overview.html.
- *CybOX™ Version 2.1.1. Part 01: Overview.* Edited by Trey Darley, Ivan Kirillov, Rich Piazza, and Desiree Beck. Latest version: http://docs.oasis-open.org/cti/cybox/v2.1.1/cybox-v2.1.1-part01-overview.html.

This specification is related to:

- TAXII™ Version 2.0. Edited by John Wunder, Mark Davidson, and Bret Jordan. Latest version: http://docs.oasis-open.org/cti/taxii/v2.0/taxii-v2.0.html.

Abstract:

Structured Threat Information Expression (STIX™)isalanguageforexpressingcyberthreatandobservableinformation.Thisdocument definesapatterninglanguagetoenablethedetectionofpossiblymaliciousactivityonnetworksandendpoints.

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 “SendAComment” button on the TC’s web page at https://www.oasis-open.org/committees/cti/.

This Committee Specification is provided under the Non-Assertion Mode of the OASISIPRPolicy, the mode chosen when the Technical Committee was established. 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).

Note that any machine-readable content (ComputerLanguageDefinitions) declared Normative for this Work Product is provided in separate plain text files. In the event of a discrepancy between any such plain text file and display content in the Work Product's prose narrative document(s), the content in the separate plain text file prevails.

Citation format:

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

[STIX-v2.0-Pt5-Patterning]

STIX™ Version 2.0. Part 5: STIX Patterning. Edited by Trey Darley and Ivan Kirillov. 19 July 2017. OASIS Committee Specification 01. http://docs.oasis-open.org/cti/stix/v2.0/cs01/part5-stix-patterning/stix-v2.0-cs01-part5-stix-patterning.html. Latest version: http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part5-stix-patterning.html.

Notices

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

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see https://www.oasis-open.org/policies-guidelines/trademark for above guidance.

STIX™, CYBOX™, AND TAXII™ (STANDARD OR STANDARDS) AND THEIR COMPONENT PARTS ARE PROVIDED "AS IS" WITHOUT ANY WARRANTY OF ANY KIND, EITHER EXPRESSED, IMPLIED, OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTY THAT THESE STANDARDS OR ANY OF THEIR COMPONENT PARTS WILL CONFORM TO SPECIFICATIONS, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR FREEDOM FROM INFRINGEMENT, ANY WARRANTY THAT THE STANDARDS OR THEIR COMPONENT PARTS WILL BE ERROR FREE, OR ANY WARRANTY THAT THE DOCUMENTATION, IF PROVIDED, WILL CONFORM

TO THE STANDARDS OR THEIR COMPONENT PARTS. IN NO EVENT SHALL THE UNITED STATES GOVERNMENT OR ITS CONTRACTORS OR SUBCONTRACTORS BE LIABLE FOR ANY DAMAGES, INCLUDING, BUT NOT LIMITED TO, DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES, ARISING OUT OF, RESULTING FROM, OR IN ANY WAY CONNECTED WITH THESE STANDARDS OR THEIR COMPONENT PARTS OR ANY PROVIDED DOCUMENTATION, WHETHER OR NOT BASED UPON WARRANTY, CONTRACT, TORT, OR OTHERWISE, WHETHER OR NOT INJURY WAS SUSTAINED BY PERSONS OR PROPERTY OR OTHERWISE, AND WHETHER OR NOT LOSS WAS SUSTAINED FROM, OR AROSE OUT OF THE RESULTS OF, OR USE OF, THE STANDARDS, THEIR COMPONENT PARTS, AND ANY PROVIDED DOCUMENTATION. THE UNITED STATES GOVERNMENT DISCLAIMS ALL WARRANTIES AND LIABILITIES REGARDING THE STANDARDS OR THEIR COMPONENT PARTS ATTRIBUTABLE TO ANY THIRD PARTY, IF PRESENT IN THE STANDARDS OR THEIR COMPONENT PARTS AND DISTRIBUTES IT OR THEM "AS IS."

Table of Contents

1 Introduction

In order to detect a large proportion of malicious behavior in the course of defending our networks, it is necessary to correlate telemetry from both host-based and network-based tools. Before undertaking work on STIX Patterning, as a technical subcommittee we made a thorough effort to evaluate whether there was already an existing patterning language that would support our use cases available as an open standard. In particular, we considered whether it would be possible to extend the syntax of Snort or Yara rather than create an entirely new language. This was eventually ruled out as unfeasible, both from a technical perspective as well as taking into consideration that from a licensing/IPR perspective, extending either of those languages under the auspices of OASIS would have been problematic.

Given that STIX Patterning exists to support STIX Indicators, consider what value Indicator-sharing provides: a mechanism for communicating how to find malicious code and/or threat actors active within a given network. Among the essential tools widely deployed by defenders are SIEMs (or similar data processing platforms capable of consuming, correlating, and interrogating large volumes of network and host-based telemetry.) These data processing platforms utilize proprietary query languages. As development began on STIX Patterning, one of the principal design goals was to create an abstraction layer capable of serializing these proprietary correlation rules so as to enhance the overall value proposition of indicator-sharing.

In order to enhance detection of possibly malicious activity on networks and endpoints, a standard language is needed to describe what to look for in a cyber environment. The STIX Patterning language allows matching against timestamped Cyber Observable data (such as STIX Observed Data Objects) collected by a threat intelligence platform or other similar system so that other analytical tools and systems can be configured to react and handle incidents that might arise.

This first language release is focused on supporting a common set of use cases and therefore allows for the expression of an initial set of patterns that producers and consumers of STIX can utilize. As more complex patterns are deemed necessary, the STIX patterning language will be extended in future releases to improve its effectiveness as an automated detection/remediation method.

1.0 IPR Policy

This Committee Specification is provided under the Non-Assertion Mode of the OASISIPRPolicy, the mode chosen when the Technical Committee was established. 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).

1.1 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].

All text is normative except for examples and any text marked non-normative.

1.2 Normative References

[Davis] M. Davis and K. Whistler, "UNICODE NORMALIZATION FORMS", Unicode® Standard Annex #15, February 2016. [Online] Available: http://unicode.org/reports/tr15/

[RFC2119] Bradner, S., “"Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, http://www.rfc-editor.org/info/rfc4648.

1.3 Non-Normative Reference

[Pattern Grammar] OASIS Cyber Threat Intelligence (CTI) TC, "STIX Pattern Grammar", OASIS. [Online]. Available: https://github.com/oasis-open/cti-stix2-json-schemas/tree/master/pattern_grammar

1.4 ANTLR Grammar

The latest ANTLR grammar for the patterning specification can be found on Github in the Pattern Grammar repository [Pattern Grammar].Notethatthisgrammarisnon-normativeandisintendedsolelyasanaidtoimplementers.

1.5 Naming Requirements

1.5.1 Property Names and String Literals

In the JSON serialization all property names and string literals **MUST** be exactly the same, including case, as the names listed in the property tables in this specification. For example, the SDO common property **created_by_ref** must result in the JSON key name "created_by_ref". Properties marked required in the property tables **MUST** be present in the JSON serialization.

1.5.2 Reserved Names

Reserved property names are marked with a type called RESERVED and a description text of “RESERVED FOR FUTURE USE”. Any property name that is marked as RESERVED **MUST NOT** be present in STIX content conforming to this version of the specification.

1.6 Document Conventions

1.6.1 Naming Conventions

All type names, property names, and literals are in lowercase, except when referencing canonical names defined in another standard (e.g., literal values from an IANA registry). Words in property names are separated with an underscore(_), while words in type names and string enumerations are separated with a hyphen-minus ('-' U+002d). All type names, property names, object names, and vocabulary terms are between three and 250 characters long.

1.6.2 Font Colors and Style

The following color, font and font style conventions are used in this document:

- The `Consolas` font is used for all type names, property names and literals.
  - type names are in red with a light red background - `hashes`
  - property names are in bold style - **`protocols`**
  - literals (values) are in blue with a blue background - `SHA-256`
- In an object's property table, if a common property is being redefined in some way, then the background is dark gray.
- All examples in this document are expressed in JSON. They are in `Consolas` 9-point font, with straight quotes, black text and a `light grey background`, and 2-space indentation.
- Parts of the example may be omitted for conciseness and clarity. These omitted parts are denoted with the ellipses (...).

2 Definitions

The terms defined below are used throughout this document.

Terms	Definitions	Example
whitespace	Any Unicode code point that has WSpace set as a property, for example, line feeds, carriage returns, tabs, and spaces.	n/a
Observation	Observations represent data about systems or networks that is observed at a particular point in time - for example, information about a file that existed, a process that was observed running, or network traffic that was transmitted between two IPs. In STIX, Observations are represented by Observed Data SDOs, with their **`first_observed`** timestamp defining the observation time.	n/a
Comparison Expression	Comparison Expressions are the basic components of Observation Expressions. They consist of an Object Path and a constant joined by a Comparison Operator (listed in section 4.2.1,ComparisonOperators).	`user-account:value = 'Peter'`
Comparison Operators	Comparison Operators are used within Comparison Expressions to compare an Object Path against a constant or set of constants.	`MATCHES`
Object Path	Object Paths define which properties of Cyber Observable Objects should be evaluated as part of a Comparison Expression. Cyber Observable Objects and their properties are defined in *STIX™Version 2.0. Part 4: Cyber Observable Objects*.	`ipv6-addr:value`

Observation Expression	Observation Expressions consist of one or more Comparison Expressions joined with Boolean Operators and surrounded by square brackets.
	An Observation Expression may consist of two Observation Expressions joined by an Observation Operator. This may be applied recursively to
		`[ipv4-addr:value = '203.0.113.1' OR ipv4-addr:value = '203.0.113.2']`
		or (with Observation Operator):

	compose multiple Observation Expressions into a single Observation Expression.
	Observation Expressions may optionally be followed by one or more Qualifiers further constraining the result set. Qualifiers may be applied to all of the Observation Expressions joined with Observation Operators; in this case, parentheses should be used to group the set of Observation Expressions, with the Qualifier following the closing parenthesis.
		`([ipv4-addr:value = '198.51.100.5'] FOLLOWEDBY [ipv4-addr:value = '198.51.100.10'])`
		or (with Observation Operator and Qualifier):
		`([ipv4-addr:value = '198.51.100.5' ] AND [ipv4-addr:value = '198.51.100.10']) WITHIN 300 SECONDS`

Boolean Operators	Boolean Operators are used to combine Comparison Expressions within an Observation Expression.	(Comparison Expressions) `user-account:value = 'Peter' OR user-account:value = 'Mary'`
Qualifier	Qualifiers provide a restriction on the Observations that are considered valid for matching the preceding Observation Expression.	`[file:name = 'foo.dll'] START '2016-06-01T00:00:00Z' STOP '2016-07-01T00:00:00Z'`
Observation Operators	Observation Operators are used to combine two Observation Expressions operating on two different Observed Data instances into a single pattern.	`[ipv4-addr:value = '198.51.100.5'] AND [ ipv4-addr:value = '198.51.100.10']`
Pattern Expression	A Pattern Expression represents a valid instance of a Cyber Observable pattern. The most basic Pattern Expression consists of a single Observation Expression containing a single Comparison Expression.	`[file:size = 25536]`

2.1 Constants

The data types enumerated below are supported as operands within Comparison Expressions. This table is included here as a handy reference for implementers.

Note that unlike Cyber Observable Objects (which are defined in terms of the MTI JSON serialization), STIX Patterns are Unicode strings, regardless of the underlying serialization, hence the data types defined in the table below in some cases differ from the definitions contained in *STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts*.

Each constant defined in Patterning has a limited set of Cyber Observable Data types that they are allowed to be compared against. In some cases, there are multiple Cyber Observable Data Types that could be compared against a STIX Patterning Constant; this is due to the fact that certain Cyber Observable Data Types are semantically indistinguishable because of their JSON serialization. The Cyber Observable Comparable Data Type(s) column in the table below defines these limitations.

STIX Patterning Constant	Cyber Observable Comparable Data Type(s)	Description
`boolean`	`boolean`	A constant of `boolean` type encodes truth or falsehood. Boolean truth is denoted by the literal `true` and falsehood by the literal `false`.

`binary`	`binary`
	`hex`
	`string`
		A constant of `binary` type is a base64 encoded array of octets (8-bit bytes) per [RFC4648].Thebase64string**MUST** be surrounded by apostrophes (''' U+0027) and prefixed by a 'b' (U+0062). Line feeds in the base64 encoded data **MUST** be supported and ignored, but are not required to be inserted.
		Example:
		`b'ABI='`

`hex`	`binary`
	`hex`
	`string`
		A constant of `hex` type encodes an array of octets (8-bit bytes) as hexadecimal. The string **MUST** consist of an even number of hexadecimal characters, which are the digits '0' through '9' and the letters 'a' through 'f'. The hex string **MUST** be surrounded by apostrophes (''' U+0027) and prefixed by an 'h' (U+0068).
		Example:
		`h'0012'`

`integer`	`integer` `float`	A constant of `integer` type encodes a signed decimal number in the usual fashion (e.g., 123). In the case of positive integers, the integer **MUST** be represented as-is, omitting the plus sign ('+' U+002b). Negative integers **MUST** be represented by prepending a hyphen-minus ('-' U+002d). When compared against a Cyber Observable `float`, the full value must be compared and must not be truncated. For example, the result of comparing a STIX Patterning constant integer value of 1 to a

		Cyber Observable `float` value of 1.5 is not equal. The valid range of values is defined in *STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts.*
`float`	`integer` `float`	A constant of `float` type encodes a floating point number in the usual fashion (e.g., 123.456). In the case of positive floating point number, the floating point number **MUST** be represented as-is, omitting the plus sign' ('+' U+002b). Negative floating point numbers **MUST** be represented by prepending a hyphen-minus ('-' U+002d). The valid range of values is defined in *STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts*.

`string`	`string`
	`binary`
	`hex`
		A constant of `string` type encodes a string as a list of Unicode code points surrounded by apostrophes ("'" U+0027).
		The escape character is the backslash ("\' U+005c). Only the apostrophe or the backslash may follow, and in that case, the respective character is used for the sequence.
		If a string only contains codepoints less than (U+0100), then the string **MAY** be converted to a binary type value (if needed for comparison). The mapping is code point U+0000 to 00 through U+00ff to ff.

`timestamp`	`timestamp`	A constant of `timestamp` type encodes a STIX timestamp (as specified in section 2.10 of *STIX™ Version 2.0 Part 1: STIX Core Concepts*) as a string. The timestamp string **MUST** be surrounded by apostrophes ("'" U+0027) and prefixed with a 't' (U+0074). Example: `t'2014-01-13T07:03:17Z'`

3 STIX Patterns

STIX Patterns are composed of multiple building blocks, ranging from simple key-value comparisons to more complex, context-sensitive expressions. The most fundamental building block is the Comparison Expression, which is a comparison between a single property of a Cyber Observable Object and a given constant using a Comparison Operator. As a simple example, one might use the following Comparison Expression (contained within an Observation Expression) to match against an IPv4 address:

```
[ipv4-addr:value = '198.51.100.1/32']
```

Moving up a level of complexity, the next building block of a STIX Pattern is the Observation Expression, which consists of one or more Comparison Expressions joined by Boolean Operators and bounded by square brackets. An Observation Expression refines which set of Cyber Observable data (i.e., as part of an Observation) will match the pattern, by selecting the set that has the Cyber Observable Objects specified by the Comparison Expressions. An Observation Expression consisting of a single Comparison Expression is the most basic valid STIX Pattern. Building upon the previous example, one might construct an Observation Expression to match against multiple IPv4 addresses and an IPv6 address:

```
[ipv4-addr:value = '198.51.100.1/32' OR ipv4-addr:value = '203.0.113.33/32' OR
ipv6-addr:value = '2001:0db8:dead:beef:dead:beef:dead:0001/128']
```

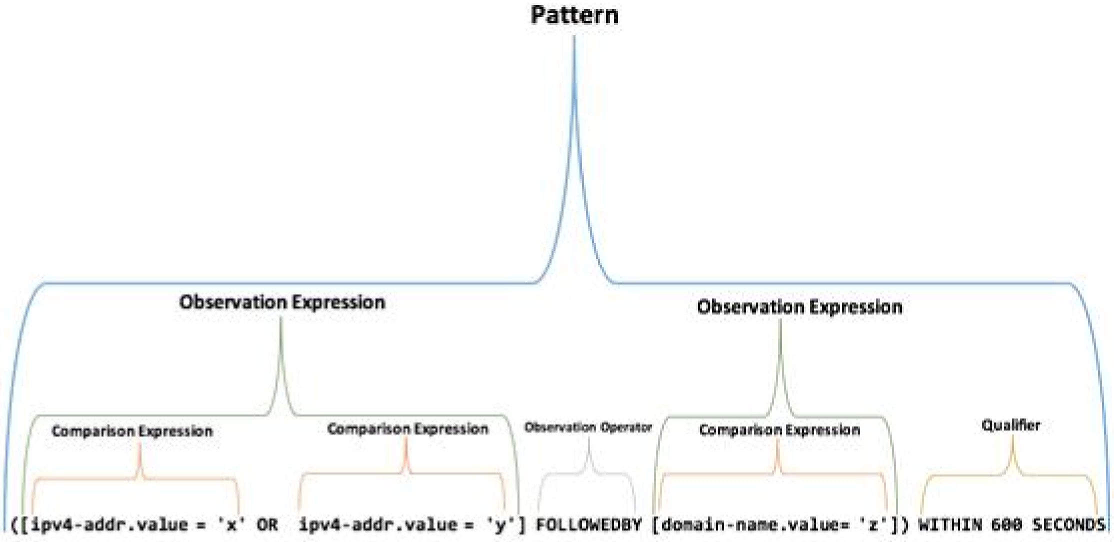
Observation Expressions may be followed by one or more Qualifiers, which allow for the expression of further restrictions on the set of data matching the pattern. Continuing with the above example, one might use a Qualifier to state that the IP addresses must be observed several times in repetition:

```
[ipv4-addr:value = '198.51.100.1/32' OR ipv4-addr:value = '203.0.113.33/32' OR
ipv6-addr:value = '2001:0db8:dead:beef:dead:beef:dead:0001/128'] REPEATS 5 TIMES
```

The final, highest level building block of STIX Patterning combines two or more Object Expressions via Observation Operators, yielding a STIX Pattern capable of matching across multiple STIX Observed Data SDOs. Building further upon our previous example, one might use an Observation Operator to specify that an observation of a particular domain name must follow the observation of the IP addresses (note the use of parentheses to encapsulate the two Observation Expressions), along with a different Qualifier to state that both the IP address and domain name must be observed within a specific time window:

```
([ipv4-addr:value = '198.51.100.1/32' OR ipv4-addr:value = '203.0.113.33/32' OR
ipv6-addr:value = '2001:0db8:dead:beef:dead:beef:dead:0001/128'] FOLLOWEDBY
[domain-name:value = 'example.com']) WITHIN 600 SECONDS
```

The diagram below depicts a truncated version of the various STIX Patterning components in the above example.


Pattern
Observation Expression
Observation Expression
Comparison Expression
Comparison Expression
Observation Operator
Comparison Expression
Qualifier
([ipv4-addr.value = 'x' OR ipv4-addr.value = 'y'] FOLLOWEDBY [domain-name.value= 'z']) WITHIN 600 SECONDS

4 Pattern Expressions

Pattern Expressions evaluate to true or false. They comprise one or more Observation Expressions joined by Observation Operators. Pattern Expressions are evaluated against a set of specific Observations. If one or more of those Observations match the Pattern Expression, then it evaluates to true. If no Observations match, the Pattern Expression evaluates to false.

Pattern Expressions **MUST** be encoded as Unicode strings.

Whitespace (i.e., Unicode code points where WSpace=Y) in the pattern string is used to delimit parts of the pattern, including keywords, constants, and field objects. Whitespace characters between operators, including line feeds and carriage returns, **MUST** be allowed. Multiple whitespace characters in a row **MUST** be treated as a single whitespace character.

An invalid pattern resulting from parsing error or invalid constants (e.g., an invalid hex or binary constant) **MUST NOT** match any Observations.

4.1 Observation Expressions

Observation Expressions comprise one or more Comparison Expressions, joined via Boolean Operators.

Observation Expressions **MUST** be delimited by square brackets left square bracket ('[' U+005b) and right square bracket (']' U+005d). One or more Observation Expression Qualifiers **MAY** be provided after the closing square bracket or closing parenthesis of an Observation Expression. Observation Expressions **MAY** be joined by Observation Operators.

Individual Observation Expressions (e.g., `[a = b]`) match against a single Observation, i.e., a single STIX Observed Data instance. In cases where matching against *multiple* Observations is required, two or more Observation Expressions may be combined via Observation Operators, indicating that the pattern must be evaluated against two or more distinct Observations.

When matching an Observation against an Observation Expression, all Comparison Expressions contained within the Observation Expression **MUST** match against the same Cyber Observable Object, including referenced objects. An Observation Expression **MAY** contain Comparison Expressions with Object Paths that are based on different object types, but such Comparison Expressions **MUST** be joined by OR. The Comparison Expressions of an Observation Expression that use AND **MUST** use the same base Object Path, e.g., `file:`.

For example, consider the following Pattern Expression:

```
[(type-a:property-j = 'W' AND type-a:property-k = 'X') OR (type-b:property-m = 'Y' AND 
type-b:property-n = 'Z')]
```

This expression can match an Observable with an object of either `type-a` or `type-b`, but both Comparison Expressions for that specific type must evaluate to true for the same object. Comparison Expressions that are intended to match a single object type can be joined by either AND or OR. For example:

```
[type-a:property-j = 'W' AND type-a:property-k = 'X' OR type-a:property-l = 'Z']
```

As AND has higher precedence than OR, the preceding example requires an Observation to have either both `property-j = 'W' and property-k = 'X' or just property-l = 'Z'`.

Observation Expressions, along with their Observation Operators and optional Qualifiers, **MAY** be surrounded with parenthesis to delineate which Observation Expressions the Qualifiers apply to. For example:

```
([ a ] AND [ b ] REPEATS 5 TIMES) WITHIN 5 MINUTES
```

The preceding example results in one *a* and 5 *b's* that all match in a 5 minute period. As another example:

```
([ a ] AND [ b ]) REPEATS 5 TIMES WITHIN 5 MINUTES
```

The preceding example results in 5 *a*'s and 5 *b's* (10 Observations) that all match in a 5 minute period.

4.1.1 Observation Expression Qualifiers

Each Observation Expression **MAY** have additional temporal or repetition restrictions using the respective `WITHIN`, `START`/`STOP`, and `REPEATS` keywords.

Qualifiers	Description
a `REPEATS` *x* `TIMES`	*a* **MUST** be an Observation Expression or a preceding Qualifier. *a* **MUST** match exactly *x* times, where each match is a different Observation. *x* **MUST** be a positive integer. This is purely a shorthand way of writing: "*a*" followed by "AND *a",* x-1 times. Example: `[ b ] FOLLOWEDBY [ c ]   REPEATS 5 TIMES` In this example, the `REPEATS` applies to *c*, and it does not apply to *b*. The results will be *b* plus 5 *c*'s where all 5 *c*'s were observed after the *b*. Note that there is only a single Qualifier in this example; more complex patterns may use more than one.
a `WITHIN` *x* `SECONDS`	*a* **MUST** be an Observation Expression or a preceding Qualifier. All Observations matched by *a* **MUST** occur, or have been observed, within the specified time window. *x* **MUST** be a positive floating point value. If there is a set of two or more Observations matched by *a*, the most recent Observation timestamp contained within that set **MUST NOT** be equal to or later than the delta of the earliest Observation timestamp within the set plus the specified time window.

	Example: `([file:hashes.'SHA-256' = '13987239847...'] AND [win-registry-key:key = 'hkey']) WITHIN 120 SECONDS` The above Pattern Expression looks for a file hash and a registry key that were observed within 120 seconds of each other. The parentheses are needed to apply the `WITHIN` Qualifier to both Observation Expressions.
a `START` *x* `STOP` *y*	*a* **MUST** be an Observation Expression or a preceding Qualifier. All Observations that match *a* **MUST** have an observation time >= *x* and < *y*. *x* and *y* **MUST** be a timestamp as defined in section 2.10 of *STIX™ Version 2.0. Part 1: STIX Core Concepts*.

4.1.2 Observation Operators

Two or more Observation Expressions **MAY** be combined using an Observation Operator in order to further constrain the set of Observations that match against the Pattern Expression.

Observation Operators	Description	Associativity
[*a*] `AND` [*b*]	*a* and *b* **MUST** both be Observation Expressions and **MUST** both evaluate to true on *different* Observations.	Left to right
[*a*] `OR` [*b*]	*a* and *b* **MUST** both be Observation Expressions and one of *a* or *b* **MUST** evaluate to true on *different* Observations.	Left to right
[*a*] `FOLLOWEDBY` [*b*]	*a* and *b* **MUST** both be Observation Expressions. Both *a* and *b* **MUST** both evaluate to true, where the observation timestamp associated with *b* is greater than or equal to the observation timestamp associated with *a* and **MUST** evaluate to true on *different* Observations.	Left to right

For example, consider the following Pattern Expression:

```
[ a = 'b' ] FOLLOWEDBY [ c = 'd' ] REPEATS 5 TIMES
```

The preceding expression says to match an Observation with *a* equal to 'b' that precedes 5 occurrences of Observations that have *c* equal to ‘d’, for a total of 6 Observations matched. This interpretation is due to qualifiers not being greedy, and is equivalent to `[ a = 'b' ] FOLLOWEDBY ( [ c = 'd' ] REPEATS 5 TIMES)`.

Alternatively, using parenthesis to group the initial portion, we get the following example:

```
([ a = 'b' ] FOLLOWEDBY [ c = 'd' ]) REPEATS 5 TIMES
```

The preceding expression will match 5 pairs of Observations where *a* equals 'b' followed by an Observation where *c* is equal to 'd', for a total of 10 Observations matched.

4.1.3 Operator Precedence

Operator associativity and precedence may be overridden by the use of parentheses. Unless otherwise specified, operator associativity (including for parentheses) is left-to-right. Precedence in the below table is from highest to lowest.

Operators	Associativity	Valid Scope
`()`	left to right	Observation Expression or Pattern Expression, Observation Expression and Qualifier
`AND`	left to right	Observation Expression, Pattern Expression
`OR`	left to right	Observation Expression, Pattern Expression
`FOLLOWEDBY` (Observation Operator)	left to right	Pattern Expression

4.2 Comparison Expression

Comparison Expressions are the most basic components of STIX Patterning, comprising an Object Path and a constant joined by a Comparison Operator. Each Comparison Expression is a singleton, and so they are evaluated from left to right.

A Boolean Operator joins two Comparison Expressions together. In the following table, *a* or *b* is either a Comparison Expression or a composite expression (which may be composed recursively) consisting of two or more Comparison Expressions joined with Boolean Operators and enclosed by parentheses.

Boolean Operator	Description	Associativity
a `AND` *b*	*a* and *b* **MUST** both be Comparison Expressions or a composite expression (which may be composed recursively) consisting of two or more Comparison Expressions joined with Boolean Operators and enclosed by parentheses. *a* and *b* **MUST** both evaluate to true on the same Observation.	Left to right

a `OR` *b*	*a* and *b* **MUST** both be Comparison Expressions or a composite expression (which may be composed recursively) consisting of two or more Comparison Expressions joined with Boolean Operators and enclosed by parentheses. Either *a* or *b* **MUST** evaluate to true.	Left to right

4.2.1 Comparison Operators

The table below describes the available Comparison Operators for use in Comparison Expressions; in the table, *a* **MUST** be an Object Path and *b* **MUST** be a constant. If the arguments to the Comparison Operators are of incompatible types (e.g., the Object Path is an integer and the constant is a string), the results are false; the sole exception is the `!=` operator in which case the result is true. Some STIX Patterning constants and Cyber Observable data types may be comparable in a Comparison Expression. For example, the `hex` and `binary` types both represent binary data, and their representative binary data is that which must be compared for equality. See section 2.1fortypecompatibilitybetweenSTIXPatterningandCyberObservabletypes.

A Comparison Operator **MAY** be preceded by the modifier `NOT`, in which case the resultant Comparison Expression is logically negated.

Comparison Operator	Description	Example
a `=` *b*	*a* and *b* **MUST** be equal (transitive), where *a* **MUST** be an Object Path and *b* **MUST** be a constant of the same data type as the Object property specified by *a*.	`file:name = 'foo.dll'`
a `!=` *b*	*a* and *b* **MUST NOT** be equal (transitive), where *a* **MUST** be an Object Path and *b* **MUST** be a constant of the same data type as the Object property specified by *a*.	`file:size != 4112`
a `>` *b*	*a* is numerically or lexically greater than *b*, where *a* **MUST** be an Object Path and *b* **MUST** be a constant of the same data type as the Object property specified by *a*.	`file:size > 256`
a `<` *b*	*a* is numerically or lexically less than *b*, where *a* **MUST** be an Object Path and *b* **MUST** be a constant of the same data type as the Object property specified by *a*.	`file:size < 1024`
a `<=` *b*	*a* is numerically or lexically less than or equal to *b*, where *a* **MUST** be an Object Path and *b* **MUST** be a constant of the same data type as the Object property specified by *a*.	`file:size <= 25145`

a `>=` *b*	*a* is numerically or lexically greater than or equal to *b*, where *a* **MUST** be an Object Path and *b* **MUST** be a constant of the same data type as the Object property specified by *a*.	`file:size >= 33312`
a `IN` (*x*,*y*,...)	*a* **MUST** be an Object Path and **MUST** evaluate to one of the values enumerated in the set of x,y,... (transitive). The set values in *b* **MUST** be constants of homogenous data type and **MUST** be valid data types for the Object Property specified by *a.* The return value is true if *a* is equal to one of the values in the list. If *a* is not equal to any of the items in the list, then the Comparison Expression evaluates to false.	`process:name IN ('proccy', 'proximus', 'badproc')`
a `LIKE` *b*	*a* **MUST** be an Object Path and **MUST** match the pattern specified in *b* where any '%' is 0 or more characters and '_' is any one character. This operator is based upon the SQL *LIKE* clause and makes use of the same wildcards. The string constant *b* **MUST** be NFC normalized [Davis]priortoevaluation.	`directory:path LIKE 'C:\\Windows\\%\\foo'`
a `MATCHES` *b*	*a* **MUST** be an Object Path and **MUST** be matched by the pattern specified in *b,* where *b* is a string constant containing a PCRE compliant regular expression. *a* **MUST** be NFC normalized [Davis]beforecomparisonifthepropertyisofstringtype. Regular expressions **MUST** be conformant to the syntax defined by the Perl-compatible Regular Expression (PCRE) library (http://www.pcre.org/original/doc/html/pcrepattern.html). The search function MUST be used. The DOTALL option MUST be specified. The standard beginning and end anchors may be used in the pattern to obtain match behavior. In the case that the property is binary (e.g., the property name ends in _bin or _hex), then the UNICODE flag MUST NOT be specified.	`directory:path MATCHES '^C:\\Windows\\w+$'`

Set Operator	Description	Example
a `ISSUBSET` *b*	When *a* is a set that is wholly contained by the set *b*, the Comparison Expression evaluates to true. *a* **MUST** be an Object Path referring to the **`value`** property of an Object of type `ipv4-addr` or `ipv6-addr`. *b* **MUST** be a valid `string` representation of the corresponding	`ipv4-addr:value ISSUBSET '198.51.100.0/24'`

	Object type (as defined in *STIX™ Version 2.0. Part 4: Cyber Observable Objects*). For example, if `ipv4-addr:value` was 198.51.100.0/27, `ISSUBSET '198.51.100.0/24'` would evaluate to true. In the case that both *a* and *b* evaluate to an identical single IP address or an identical IP subnet, the Comparison Expression evaluates to true.	
a `ISSUPERSET` *b*	When *a* is a set that wholly contains the set specified by *b*, the Comparison Expression evaluates to true. *a* **MUST** be an Object Path referring either an `ipv4-addr` or `ipv6-addr` Object. *b* **MUST** be a valid `string` representation of the corresponding Object type (as defined in *STIX™ Version 2.0. Part 4: Cyber Observable Objects*). For example, if `ipv4-addr:value` was 198.51.100.0/24, `ISSUPERSET '198.51.100.0/27'` would evaluate to true. In the case that both *a* and *b* evaluate to an identical single IP address or an identical IP subnet, the Comparison Expression evaluates to true.	`ipv4-addr:value ISSUPERSET '198.51.100.0/24'`

4.2.2 String Comparison

For simple string operators, i.e., "`=`", "`!=`", "`<`", "`>`", "`<=`" and "`>=`", as collation languages and methods are unspecifiable, a simple code point (binary) comparison **MUST** be used. If one string is longer than the other, but otherwise equal, the longer string is greater than, but not equal to, the shorter string. Unicode normalization **MUST NOT** be performed on the string. This means that combining marks [Davis]aresortedbytheircodepoint,nottheNFCnormalizedvalue.E.g.‘o'U+006f<‘oz'U+006fU+007a<‘ò'U+006fU+0300<‘z'U+007a<‘ò'U+00f2.AlthoughUnicoderecommendsnormalizingstringsforcomparisons,theuseofcombiningmarksmaybesignificant,andnormalizingbydefaultwouldremovethisinformation.

NFC normalization is, however, required for other Comparison Operators, e.g., `LIKE` and `MATCHES`.

4.2.3 Binary Type Comparison

When the value of two binary object properties are compared, they are compared as unsigned octets. That is, `00` is less than `ff`. If one value is longer than the other, but they are otherwise equal, the longer value is considered greater than, but not equal to, the shorter value.

4.2.4 Native Format Comparison

The Cyber Observable Object's value **MUST** be in its native format when doing the comparison. For example, Cyber Observable Object properties that use the `binary` type (defined in section 2.2 of *STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts*) must have their value decoded into its constituent bytes prior to comparison. This also means that Object Properties which use the `hex` type must be decoded into raw octets prior to being evaluated.

In cases where a binary Cyber Observable Object property (i.e., one ending with **_bin** or **_hex**) is evaluated against a string constant, the string constant **MUST** be converted into a binary constant when all of the constituent string code points are less than U+0100. If this conversion is not possible, the comparison **MUST** evaluate to false, unless the comparison operator is `!=`, in which case it **MUST** evaluate to true.

For example given the following object, where the **payload_bin** property is of `binary` type :

```
{
  "0":{
    "type": "artifact",
    "mime_type": "application/octet-stream",
    "payload_bin": "dGhpcyBpcyBhIHRlc3Q="
    }
}
```

The pattern "`artifact:payload_bin = 'dGhpcyBpcyBhIHRlc3Q='`" would evaluate to false, while the following patterns would all evaluate to true:

"`artifact:payload_bin = 'this is a test'`", "`artifact:payload_bin = b'dGhpcyBpcyBhIHRlc3Q='`",

and "`artifact:payload_bin = h'74686973206973206120746573740'`".

5 Object Path Syntax

Defined below is the syntax for addressing properties of Cyber Observable Objects within a STIX Pattern. The following notation is used throughout the definitions below:

Notation	Definition
`<object-type>`	The type of Cyber Observable Object to match against. This **MUST** be the value of the **type** field specified for a given Cyber Observable Object in an Observation.
`<property_name>`	The name of a Cyber Observable Object property to match against. This **MUST** be a valid property name as specified in the definition of the Cyber Observable Object type referenced by the `<object-type>` notation. If the `<property_name>` contains a hyphen-minus ('-' U+002d) or a full stop ('.' U+002e), the `<property_name>` **MUST** be enclosed in apostrophes (''' U+0027). Properties that are nested (i.e., are children of other properties in a Cyber Observable Object) **MUST** be specified using the syntax `<property_name>`.`<property_name>`, where the `<property_name>` preceding the '.' is the name of the parent property and the one following is the name of the child property. If the property name is a reference to another Cyber Observable Object, the referenced Object **MUST** be dereferenced, so that its properties function as if they are nested in the Object that it is referenced by. For example, if the **src_ref** property of the Network Traffic Object references an IPv4 Address Object, the value of this IPv4 address would be specified by **network-traffic**:**src_ref.value**.

5.1 Basic Object Properties

Any non-`dictionary` and non-`list` property that is directly specified on a Cyber Observable Object.

Syntax

`<object-type>`:`<property_name>`

Example

```
file:size
```

5.2 List Object Properties

Any property on a Cyber Observable Object that uses the `list` data type.

Syntax

`<object-type>`:`<property_name>[list_index]`.`<property_name>`

Where the first `property_name` **MUST** be the name of an Object property of type `list` and `[list_index]` **MUST** be one of the following:

- An integer in the range of 0..N-1, where N is the length of the list. If *list_index* is out of range, the result of any operation is false.
- The literal '`*`' indicates that if any of the items contained within a list matches against the Comparison Expression, the Comparison Expression evaluates to true.
-

Example

```
file:extensions.windows-pebinary.sections[*].entropy > 7.0
```

The above example will return true if any PE section has an entropy property whose value is greater than 7.0.

5.3 Dictionary Object Properties

Any property on a Cyber Observable Object that uses the `dictionary` data type.

Syntax

`<object-type>`:`<property_name>`.`<key_name>`

Where `<property_name>` **MUST** be the name of an Object property of type `dictionary` and `<key_name>` **MUST** be the name of key in the dictionary.

Examples

```
file:hashes.ssdeep
```

```
file:extensions.raster-image.image_height
```

5.4 Object Reference Properties

Any property on a Cyber Observable Object that uses the `object-ref` data type, either as a singleton or as a list (i.e., `list` of type `object-ref`).

Syntax

`<object-type>`:`<property_name>`.`<dereferenced_object_property>`

Where `<property_name>` **MUST** be the name of an Object property of type `object-ref` and `<dereferenced_object_property>` **MUST** be the name of a valid property of the dereferenced Object (i.e., the Object in an Observation that is referenced via `<property_name>`).

For cases where `<property_name>` is a `list` of type `object-ref`, the corresponding syntax applies:

`<object-type>:<property_name>[list_index].<dereferenced_object_property>`

Accordingly, the same semantics for list indices as defined in section 5.2applyinthiscase.

Examples

```
email-message:from_ref.value = 'mary@example.com'
```

```
directory:contains_refs[*].name = 'foobar.dll'
```

6 Examples

Note: the examples below are **NOT** JSON encoded. This means that some characters, like double quotes, are not escaped, though they will be when encoded in a JSON string.

Matching a File with a SHA-256 hash

```
[file:hashes.'SHA-256' =
'aec070645fe53ee3b3763059376134f058cc337247c978add178b6ccdfb0019f']
```

Matching an Email Message with a particular From Email Address and Attachment File Name Using a Regular Expression

```
[email-message:from_ref.value MATCHES '.+\\@example\\.com$' AND
email-message:body_multipart[*].body_raw_ref.name MATCHES '^Final Report.+\\.exe$']
```

Matching a File with a SHA-256 hash and a PDF MIME type

```
[file:hashes.'SHA-256' = 'aec070645fe53ee3b3763059376134f058cc337247c978add178b6ccdfb0019f'
AND file:mime_type = 'application/x-pdf']
```

Matching a File with SHA-256 or a MD5 hash (e.g., for the case of two different end point tools generating either an MD5 or a SHA-256), and a different File that has a different SHA-256 hash, against two different Observations

```
[file:hashes.'SHA-256' = 'bf07a7fbb825fc0aae7bf4a1177b2b31fcf8a3feeaf7092761e18c859ee52a9c'
OR file:hashes.MD5 = 'cead3f77f6cda6ec00f57d76c9a6879f']
```

```
 AND [file:hashes.'SHA-256' =
```

```
'aec070645fe53ee3b3763059376134f058cc337247c978add178b6ccdfb0019f']
```

Matching a File with a MD5 hash, followed by (temporally) a Registry Key Object that matches a value, within 5 minutes

```
([file:hashes.MD5 = '79054025255fb1a26e4bc422aef54eb4'] FOLLOWEDBY [win-registry-key:key =
'HKEY_LOCAL_MACHINE\\foo\\bar']) WITHIN 300 SECONDS
```

Matching three different, but specific Unix User Accounts

```
[user-account:account_type = 'unix' AND user-account:user_id = '1007' AND
user-account:account_login = 'Peter'] AND [user-account:account_type = 'unix' AND
user-account:user_id = '1008' AND user-account:account_login = 'Paul'] AND
[user-account:account_type = 'unix' AND user-account:user_id = '1009' AND
user-account:account_login = 'Mary']
```

Matching an Artifact Object PCAP payload header

```
[artifact:mime_type = 'application/vnd.tcpdump.pcap' AND artifact:payload_bin MATCHES
'\\xd4\\xc3\\xb2\\xa1\\x02\\x00\\x04\\x00']
```

Matching a File Object with a Windows file path

```
[file:name = 'foo.dll' AND file:parent_directory_ref.path = 'C:\\Windows\\System32']
```

Matching on a Windows PE File with high section entropy

```
[file:extensions.windows-pebinary-ext.sections[*].entropy > 7.0]
```

Matching on a mismatch between a File Object magic number and mime type

```
[file:mime_type = 'image/bmp' AND file:magic_number_hex = h'ffd8']
```

Matching on Network Traffic with a particular destination

```
[network-traffic:dst_ref.type = 'ipv4-addr' AND network-traffic:dst_ref.value = 
'203.0.113.33/32']
```

Matching on Malware Beaconing to a Domain Name

```
[network-traffic:dst_ref.type = 'domain-name' AND network-traffic:dst_ref.value = 
'example.com'] REPEATS 5 TIMES WITHIN 1800 SECONDS
```

Matching on a Domain Name with IPv4 Resolution

```
[domain-name:value = 'www.5z8.info' AND domain-name:resolves_to_refs[*].value = 
'198.51.100.1/32']
```

Matching on a URL

```
[url:value = 'http://example.com/foo' OR url:value = 'http://example.com/bar']
```

Matching on an X509 Certificate

```
[x509-certificate:issuer = 'CN=WEBMAIL' AND x509-certificate:serial_number = 
'4c:0b:1d:19:74:86:a7:66:b4:1a:bf:40:27:21:76:28']
```

Matching on a Windows Registry Key

```
[windows-registry-key:key = 'HKEY_CURRENT_USER\\Software\\CryptoLocker\\Files' OR 
windows-registry-key:key = 
'HKEY_CURRENT_USER\\Software\\Microsoft\\CurrentVersion\\Run\\CryptoLocker_0388']
```

Matching on a File with a set of properties

```
[(file:name = 'pdf.exe' OR file:size = '371712') AND file:created = t'2014-01-13T07:03:17Z']
```

Matching on an Email Message with specific Sender and Subject

```
[email-message:sender_ref.value = 'jdoe@example.com' AND email-message:subject = 'Conference 
Info']
```

Matching on a Custom USB Device

```
[x-usb-device:usbdrive.serial_number = '575833314133343231313937']
```

Matching on Two Processes Launched with a Specific Set of Command Line Arguments Within a Certain Time Window

```
[process:command_line MATCHES '^.+>-add GlobalSign.cer -c -s -r localMachine Root$']
FOLLOWEDBY [process:command_line MATCHES'^.+>-add GlobalSign.cer -c -s -r
localMachineTrustedPublisher$'] WITHIN 300 SECONDS
```

Matching on a Network Traffic IP that is part of a particular Subnet

```
[network-traffic:dst_ref.value ISSUBSET '2001:0db8:dead:beef:0000:0000:0000:0000/64']
```

Matching on several different combinations of Malware Artifacts. Note the following pattern requires that both a file and registry key exist, or that one of two processes exist.

```
([file:name = 'foo.dll'] AND [win-registry-key:key = 'HKEY_LOCAL_MACHINE\\foo\\bar']) OR
[process:name = 'fooproc' OR process:name = 'procfoo']
```

7 Conformance

Implementers of the STIX Patterning language are not required to support the full capabilities provided by the language. Rather, implementers are strongly encouraged to support as much of STIX Patterning as feasible, given the capabilities of their products, but only required to support the minimum conformance level (defined below) necessary for their particular use cases. For example, the vendor of a network intrusion detection system (NIDS) that looks for malicious network traffic may only need to implement the Comparison Operators and support basic Observation Expressions to explicitly match against network traffic and IP addresses.

While the STIX Patterning language specification is tightly coupled with the STIX Cyber Observable object data models, it is understood that in many (or even most) implementations STIX Patterns will be used as an abstraction layer for transcoding into other proprietary query formats. STIX Patterns may be evaluated directly against a corpus of STIX Observed Data instances but they may also, for example, be translated into some query syntax for a packet inspection device. In this second case, the STIX Patterns are in fact evaluated in the context of data passing on the wire, not in the form of STIX Cyber Observables.

The STIX Patterning language's Observation Operators allow for the creation of patterns that explicitly match across multiple Observations; however, the language purposefully does not specify anything about the source of the underlying data for each Observation. For example, depending on a particular patterning implementation, the data for a pattern that matches on network traffic could come from an endpoint or from a NIDS. It is incumbent upon implementers to ascertain the appropriate data sources (where applicable) for each Observation within a given pattern.

7.1 Pattern Producer

Software that creates STIX patterns is known as a "Pattern Producer". Such software **MUST** support the creation of patterns that conform to all normative statements and formatting rules in this document. Pattern Producers **MUST** specify their conformance in terms of the conformance levels defined in section 7.3.

7.2 Pattern Consumer

Software that consumes STIX patterns is known as a "Pattern Consumer". Such software MUST support the consumption of patterns that conform to all normative statements and formatting rules in this document. Pattern Consumers **MUST** specify their conformance in terms of the conformance levels defined in section 7.3.

7.3 Conformance Levels

7.3.1 Level 1: Basic Conformance

Software that conforms to the minimum required aspects of the patterning specification, is known as a “Level 1 STIX Patterning Implementation”.

Such software **MUST** support the following features by conforming to all normative statements and behaviors in the referenced sections:

- Single Observation Expressions (omitting Qualifiers), as described in section 4.1
- All Comparison Operators, as described in section 4.2.1

This level of conformance is intended primarily for software that is deployed at endpoints or network boundaries and which is architecturally unable to maintain state, as would be required in order to support Qualifiers such as `WITHIN`.

7.3.2 Level 2: Basic Conformance plus Observation Operators

Software that supports the minimum required aspects of the patterning specification but can operate on multiple Observations, is known as a “Level 2 STIX Patterning Implementation”.

Such software **MUST** support the following features by conforming to all normative statements and behaviors in the referenced sections:

- Single and Compound Observation Expressions (omitting Qualifiers) as described in section 4.1
- All Comparison Operators, as described in section 4.2.1
- The `AND` Observation Operator, as described in section 4.1.2
- The `OR` Observation Operator, as described in section 4.1.2

This level of conformance is intended primarily for software such as HIDS that can detect patterns across separate Observations but may not support temporal-based patterning.

7.3.3 Level 3: Full Conformance

Software that is fully conformant with **all** of the capabilities of the patterning specification is known as a "Level 3 STIX Patterning Implementation".

Such software **MUST** support the following features by conforming to all normative statements and behaviors in the referenced sections:

- Section 2.Definitions
- Section 3.STIXPatterns
- Section 4.PatternExpressions
- Section 5.ObjectPathSyntax

This level of conformance is intended primarily for software such as SIEMs that support temporal-based patterning and can also aggregate and detect patterns across multiple and disparate sources of Observations.

Appendix A. Glossary

CAPEC - Common Attack Pattern Enumeration and Classification

Consumer - Any entity that receives STIX content

CTI - Cyber Threat Intelligence

Embedded Relationship - A link (an "edge" in a graph) between one STIX Object and another represented as a property on one object containing the ID of another object

Entity - Anything that has a separately identifiable existence (e.g., organization, person, group, etc.)

IEP - FIRST (Forum of Incident Response and Security Teams) Information Exchange Policy

Instance - A single occurrence of a STIX object version

MTI - Mandatory To Implement

MVP - Minimally Viable Product

Object Creator - The entity that created or updated a STIX object (see section 3.3 of *STIX™ Version 2.0. Part 1: STIX Core Concepts*).

Object Representation - An instance of an object version that is serialized as STIX

Producer - Any entity that distributes STIX content, including object creators as well as those passing along existing content

SDO - STIX Domain Object (a "node" in a graph)

SRO - STIX Relationship Object (one mechanism to represent an "edge" in a graph)

STIX - Structured Threat Information Expression

STIX Content - STIX documents, including STIX Objects, STIX Objects grouped as bundles, etc.

STIX Object - A STIX Domain Object (SDO) or STIX Relationship Object (SRO)

STIX Relationship - A link (an "edge" in a graph) between two STIX Objects represented by either an SRO or an embedded relationship

TAXII - An application layer protocol for the communication of cyber threat information

TLP - Traffic Light Protocol

TTP - Tactic, technique, or procedure; behaviors and resources that attackers use to carry out their attacks

Appendix B. Acknowledgments

Cyber Observable Subcommittee Chairs:

Trey Darley, Kingfisher Operations, sprl

Ivan Kirillov, MITRE Corporation

STIX Subcommittee Chairs:

Sarah Kelley, Center for Internet Security (CIS)

John Wunder, MITRE Corporation

Special Thanks:

Substantial contributions to this specification from the following individuals are gratefully acknowledged:

Sarah Kelley, Center for Internet Security (CIS)

Terry MacDonald, Cosive

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

Richard Struse, DHS Office of Cybersecurity and Communications

Iain Brown, GDS

Jason Keirstead, IBM

Tim Casey, Intel

Trey Darley, Kingfisher Operations, sprl

Allan Thomson, LookingGlass Cyber

Greg Back, MITRE Corporation

Ivan Kirillov, MITRE Corporation

Jon Baker, MITRE Corporation

John Wunder, MITRE Corporation

Sean Barnum, MITRE Corporation

Richard Piazza, MITRE Corporation

Christian Hunt, New Context Services, Inc.

John-Mark Gurney, New Context Services, Inc.

Aharon Chernin, Perch

Dave Cridland, Surevine

Bret Jordan, Symantec Corp.

Participants:

The following individuals were members of the OASIS CTI Technical Committee during the creation of this specification and their contributions are gratefully acknowledged:

David Crawford, Aetna

Marcos Orallo, Airbus Group SAS

Roman Fiedler, AIT Austrian Institute of Technology

Florian Skopik, AIT Austrian Institute of Technology

Russell Spitler, AlienVault

Ryan Clough, Anomali

Nicholas Hayden, Anomali

Wei Huang, Anomali

Angela Nichols, Anomali

Hugh Njemanze, Anomali

Katie Pelusi, Anomali

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

Alexander Foley, Bank of America

Sounil Yu, Bank of America

Vicky Laurens, Bank of Montreal

Humphrey Christian, Bay Dynamics

Ryan Stolte, Bay Dynamics

Alexandre Dulaunoy, CIRCL

Andras Iklody, CIRCL

Rapha‘l Vinot, CIRCL

Sarah Kelley, CIS

Syam Appala, Cisco Systems

Ted Bedwell, Cisco Systems

David McGrew, Cisco Systems

Mark-David McLaughlin, Cisco Systems

Pavan Reddy, Cisco Systems

Omar Santos, Cisco Systems

Jyoti Verma, Cisco Systems

Doug DePeppe, Cyber Threat Intelligence Network, Inc. (CTIN)

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

Ben Othman, Cyber Threat Intelligence Network, Inc. (CTIN)

Jeff Odom, Dell

Sreejith Padmajadevi, Dell

Ravi Sharda, Dell

Will Urbanski, Dell

Sean Sobieraj, DHS Office of Cybersecurity and Communications (CS&C)

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

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

Jens Aabol, Difi-Agency for Public Management and eGovernment

Wouter Bolsterlee, EclecticIQ

Marko Dragoljevic, EclecticIQ

Oliver Gheorghe, EclecticIQ

Joep Gommers, EclecticIQ

Sergey Polzunov, EclecticIQ

Rutger Prins, EclecticIQ

Andrei S”rghi, EclecticIQ

Raymon van der Velde, EclecticIQ

Ben Sooter, Electric Power Research Institute (EPRI)

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

Phillip Boles, FireEye, Inc.

Prasad Gaikwad, FireEye, Inc.

Rajeev Jha, FireEye, Inc.

Anuj Kumar, FireEye, Inc.

Shyamal Pandya, FireEye, Inc.

Paul Patrick, FireEye, Inc.

Scott Shreve, FireEye, Inc.

Jon Warren, FireEye, Inc.

Remko Weterings, FireEye, Inc.

Gavin Chow, Fortinet Inc.

Steve Fossen, Fortinet Inc.

Kenichi Terashita, Fortinet Inc.

Ryusuke Masuoka, Fujitsu Limited

Daisuke Murabayashi, Fujitsu Limited

Derek Northrope, Fujitsu Limited

Jonathan Algar, GDS

Iain Brown, GDS

Adam Cooper, GDS

Mike McLellan, GDS

Tyrone Nembhard, GDS

Chris O'Brien, GDS

James Penman, GDS

Howard Staple, GDS

Chris Taylor, GDS

Laurie Thomson, GDS

Alastair Treharne, GDS

Julian White, GDS

Bethany Yates, GDS

Robert van Engelen, Genivia

Eric Burger, Georgetown University

Allison Miller, Google Inc.

Mark Risher, Google Inc.

Yoshihide Kawada, Hitachi, Ltd.

Jun Nakanishi, Hitachi, Ltd.

Kazuo Noguchi, Hitachi, Ltd.

Akihito Sawada, Hitachi, Ltd.

Yutaka Takami, Hitachi, Ltd.

Masato Terada, Hitachi, Ltd.

Peter Allor, IBM

Eldan Ben-Haim, IBM

Allen Hadden, IBM

Sandra Hernandez, IBM

Jason Keirstead, IBM

John Morris, IBM

Laura Rusu, IBM

Ron Williams, IBM

Paul Martini, iboss, Inc.

Jerome Athias, Individual

Peter Brown, Individual

Joerg Eschweiler, Individual

Stefan Hagen, Individual

Elysa Jones, Individual

Sanjiv Kalkar, Individual

Terry MacDonald, Individual

Alex Pinto, Individual

Tim Casey, Intel Corporation

Kent Landfield, Intel Corporation

Karin Marr, Johns Hopkins University Applied Physics Laboratory

Julie Modlin, Johns Hopkins University Applied Physics Laboratory

Mark Moss, Johns Hopkins University Applied Physics Laboratory

Mark Munoz, Johns Hopkins University Applied Physics Laboratory

Nathan Reller, Johns Hopkins University Applied Physics Laboratory

Pamela Smith, Johns Hopkins University Applied Physics Laboratory

David Laurance, JPMorgan Chase Bank, N.A.

Russell Culpepper, Kaiser Permanente

Beth Pumo, Kaiser Permanente

Michael Slavick, Kaiser Permanente

Trey Darley, Kingfisher Operations, sprl

Gus Creedon, Logistics Management Institute

Wesley Brown, LookingGlass

Jamison Day, LookingGlass

Kinshuk Pahare, LookingGlass

Allan Thomson, LookingGlass

Ian Truslove, LookingGlass

Chris Wood, LookingGlass

Greg Back, Mitre Corporation

Jonathan Baker, Mitre Corporation

Sean Barnum, Mitre Corporation

Desiree Beck, Mitre Corporation

Michael Chisholm, Mitre Corporation

Nicole Gong, Mitre Corporation

Ivan Kirillov, Mitre Corporation

Michael Kouremetis, Mitre Corporation

Chris Lenk, Mitre Corporation

Richard Piazza, Mitre Corporation

Larry Rodrigues, Mitre Corporation

Jon Salwen, Mitre Corporation

Charles Schmidt, Mitre Corporation

Alex Tweed, Mitre Corporation

Emmanuelle Vargas-Gonzalez, Mitre Corporation

John Wunder, Mitre Corporation

James Cabral, MTG Management Consultants, LLC.

Scott Algeier, National Council of ISACs (NCI)

Denise Anderson, National Council of ISACs (NCI)

Josh Poster, National Council of ISACs (NCI)

Mike Boyle, National Security Agency

Joe Brule, National Security Agency

Jessica Fitzgerald-McKay, National Security Agency

David Kemp, National Security Agency

Shaun McCullough, National Security Agency

John Anderson, NC4

Michael Butt, NC4

Mark Davidson, NC4

Daniel Dye, NC4

Angelo Mendonca, NC4

Michael Pepin, NC4

Natalie Suarez, NC4

Benjamin Yates, NC4

Daichi Hasumi, NEC Corporation

Takahiro Kakumaru, NEC Corporation

Lauri Korts-P_rn, 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.

Stephen Banghart, NIST

David Darnell, North American Energy Standards Board

Cory Casanave, Object Management Group

Aharon Chernin, Perch

Dave Eilken, Perch

Sourabh Satish, Phantom

Josh Larkins, PhishMe Inc.

John Tolbert, Queralt Inc.

Ted Julian, Resilient Systems, Inc..

Igor Baikalov, Securonix

Joseph Brand, Semper Fortis Solutions

Duncan Sparrell, sFractal Consulting LLC

Thomas Schreck, Siemens AG

Rob Roel, Southern California Edison

Dave Cridland, Surevine Ltd.

Bret Jordan, Symantec Corp.

Curtis Kostrosky, Symantec Corp.

Juha Haaga, Synopsys

Masood Nasir, TELUS

Greg Reaume, TELUS

Alan Steer, TELUS

Crystal Hayes, The Boeing Company

Wade Baker, ThreatConnect, Inc.

Cole Iliff, ThreatConnect, Inc.

Andrew Pendergast, ThreatConnect, Inc.

Ben Schmoker, ThreatConnect, Inc.

Jason Spies, ThreatConnect, Inc.

Ryan Trost, ThreatQuotient, Inc.

Patrick Coughlin, TruSTAR Technology

Chris Roblee, TruSTAR Technology

Mark Angel, U.S. Bank

Brian Fay, U.S. Bank

Joseph Frazier, U.S. Bank

Mark Heidrick, U.S. Bank

Mona Magathan, U.S. Bank

Yevgen Sautin, U.S. Bank

Richard Shok, U.S. Bank

James Bohling, US Department of Defense (DoD)

Eoghan Casey, US Department of Defense (DoD)

Gary Katz, US Department of Defense (DoD)

Jeffrey Mates, US Department of Defense (DoD)

Evette Maynard-Noel, US Department of Homeland Security

Robert Coderre, VeriSign

Kyle Maxwell, VeriSign

Eric Osterweil, VeriSign

Patrick Maroney, Wapack Labs LLC

Anthony Rutkowski, Yanna Technologies LLC

Appendix C. Revision History

Revision	Date	Editor	Changes Made
01	2017-01-20	Bret Jordan, John Wunder, Rich Piazza, Ivan Kirillov, Trey Darley	Initial Version
02	2017-04-24	Bret Jordan, John Wunder, Rich Piazza, Ivan Kirillov, Trey Darley	Changes made from first public review

부 록 Ⅰ-1

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

지식재산권 확약서 정보

Ⅰ-1.1 지식재산권 확약서(1)

- 해당 사항 없음

Ⅰ-1.2 지식재산권 확약서(2)

- 해당 사항 없음

부 록 Ⅰ-2

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

시험인증 관련 사항

Ⅰ-2.1 시험인증 대상 여부

- 해당 사항 없음

Ⅰ-2.2 시험표준 제정 현황

- 해당 사항 없음

부 록 Ⅰ-3

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

본 표준의 연계(family) 표준

Ⅰ-3.1 TTA, 구조화된 위협 정보 표현 규격(STIX^{TM}) 버전 2.0 - 제1부: STIX 핵심 개념

STIX의 핵심 개념을 정의하는 문서로 공통 데이터 형식, STIX 객체, 데이터 표시 등에 대한 설명을 제공

Ⅰ-3.2 TTA, 구조화된 위협 정보 표현 규격(STIX^{TM}) 버전 2.0 - 제2부: STIX 객체

STIX의 도메인 Objects 집합을 정의하는 문서로 Objects 의 구성요소와 구성요소에 대한 설명을 제공

Ⅰ-3.3 TTA, 구조화된 위협 정보 표현 규격(STIX^{TM}) 버전 2.0 - 제3부: STIX 사이버 관측 코어 개념

STIX의 사이버 관측 코어 개념을 정의하는 문서로 Observable을 구성하는 필드와 필드에 대한 설명을 제공

Ⅰ-3.4 TTA, 구조화된 위협 정보 표현 규격(STIX^{TM}) 버전 2.0 - 제4부: STIX 사이버 관측 객체

STIX의 사이버 관측 객체 집합을 정의하는 문서로 Observable Object의 구성요소와 구성요소에 대한 설명을 제공

부 록 Ⅰ-4

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

참고 문헌

[1] M. Davis and K. Whistler, "UNICODE NORMALIZATION FORMS", Unicode® Standard Annex #15, February 2016
http://unicode.org/reports/tr15/

[2] Bradner, S., “"Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997
http://www.ietf.org/rfc/rfc2119.txt.

[3] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006
http://www.rfc-editor.org/info/rfc4648

부 록 Ⅰ-5

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

영문표준 해설서

Ⅰ-5.1 개요

사이버 환경의 네트워크 및 엔드포인트에서 특정 위협을 탐지하고 대응하기 위해서는 해당 위협의 형태를 표현하기 위한 패터닝 언어가 필요하다. STIX 패터닝 언어는 위협 인텔리전스 플랫폼 또는 그와 유사한 시스템으로부터 수집한 STIX 관측 데이터와 연계하여 동작하도록 구성할 수 있다.

Ⅰ-5.2 정의

STIX 패터닝 언어를 구성하는 단위는 다음과 같다.

<표 Ⅰ-5.2-1> 용어 정의

용어	정의	예
whitespace(공백)	WSpace 집합을 속성으로 가진 유니코드 문자. 예: 줄 바꿈, 캐리지 리턴, 탭 및 공백.	해당 없음
Observation(관측)	관측은 존재하는 파일에 관한 정보, 실행 중인 것으로 관측된 프로세스 또는 두 IP 간에 송신된 네트워크 트래픽 등 특정 시점에 관측된 시스템 또는 네트워크에 관한 데이터를 표시한다. STIX에서 관측은 관측된 데이터 SDO에 의해 표시되며 해당 first_observed 타임스탬프를 통해 관측 시간을 정의한다.	해당 없음
Comparison Expression(비교 식)	비교 식은 관측 식의 기본 구성 요소이며, 경로 및 비교 연산자에 의해 결합된 상수로 구성된다(섹션 4.2.1,비교 연산자 참조).	user-account:value = 'Peter'
Comparison Operators(비교 연산자)	비교 연산자는 상수 또는 상수 집합을 기준으로 객체 경로를 비교하기 위해 비교 식 내에 사용된다.	MATCHES
Object Path (객체 경로)	객체 경로는 비교 식의 일부로 평가해야 하는 사이버 관측 가능 객체의 속성을 정의한다. 사이버 관측 가능 객체 및 속성은 STIX™Version 2.0. Part 4: Cyber Observable Objects를 참조한다.	ipv6-addr:value
Observation Expression(관측 식)	관측 식은 부울 연산자와 결합되고 대괄호로 둘러싸인 하나 이상의 비교 식으로 구성된다 관측 식이 관측 연산자로 결합된 관측 식 두 개로 구성될 수도 있다. 이러한 구성은 복수의 관측 식을 재귀적으로 단일 관측 식으로 조합하는 경우에 적용될 수 있다 관측 식 다음에 결과 집합을 추가로 포함하고 있는 하나 이상의 한정자가 올	[ipv4-addr:value = '203.0.113.1' OR ipv4-addr:value = '203.0.113.2'] 또는 (관측 연산자를 사용하여) ([ipv4-addr:value = '198.51.100.5'] FOLLOWEDBY

	수도 있다(선택 사항). 한정자는 관측 연산자와 결합된 모든 관측 식에 적용될 수 있으며, 이 경우 소괄호를 사용하여 관측 식의 집합을 그룹화하고 소괄호 닫기 기호 다음에 한정자가 온다.	[ipv4-addr:value = '198.51.100.10']) 또는 (관측 연산자와 한정자를 사용하여) ([ipv4-addr:value = '198.51.100.5'] AND [ipv4-addr:value = '198.51.100.10']) WITHIN 300 SECONDS
Boolean Operators(부울 연산자)	부울 연산자는 관측 식 내에서 비교 식을 결합하기 위해 사용된다.	(비교 식) user-account:value = 'Peter' OR user-account:value = 'Mary'
Qualifier(한정자)	한정자는 앞의 관측 식을 대조하는데 유효하다고 여겨지는 관측에 대한 제한사항을 제공한다.	[file:name = 'foo.dll'] START '2016-06-01T00:00:00Z' STOP '2016-07-01T00:00:00Z'
Observation Operators(관측 연산자)	관측 연산자는 서로 다른 두 관측 데이터 인스턴스에 작용하는 두 관측 식을 단일 패턴으로 조합하기 위해 사용된다.	[ipv4-addr:value = '198.51.100.5'] AND [ipv4-addr:value = '198.51.100.10']
Pattern Expression(패턴 식)	패턴 식은 사이버 관측 가능 패턴의 유효한 인스턴스를 표시한다. 가장 기본적인 패턴 식은 단일 비교 식을 포함하고 있는 단일 관측 식으로 구성된다.	[file:size = 25536]

Ⅰ-5.2.1 상수(Constants)

패턴화에 정의된 각 상수는 비교 기준이 될 수 있는 사이버 관측 가능 데이터 형식의 제한된 집합을 가진다. 때때로 STIX 패턴화 상수를 기준으로 비교할 수 있는 복수의 사이버 Observable 데이터 형식이 있을 수 있는데, 그 이유는 특정 사이버 관측 가능 데이터 형식이 해당 JSON 직렬화 때문에 의미상으로 구분되지 않기 때문이다. <표 Ⅰ-5.2.-2>의 유사한 사이버 관측 가능 데이터 형식 열에 이러한 제한사항을 정의하였다.

<표 Ⅰ-5.2-2> 데이터 형식

STIX 패턴화 상수	유사한 사이버 Observable 데이터 형식	설명
boolean	boolean	boolean 형식의 상수는 참 또는 거짓을 인코딩한다. 부울 참은 리터럴 true로 표시하며 거짓은 리터럴 false로 표시한다.
binary	binary hex string	binary 형식의 상수는 RFC4648]에 따른 8진수(8비트 바이트)의 base64 인코딩 배열이다. base64 문자열은 아포스트로피(''' U+0027)로 둘러싸고 앞에 'b'(U+0062)를 덧붙여야 한다. base64 인코딩 데이터의 줄바꿈을 반드시 지원하고 무시해야 하지만 꼭 삽입할 필요는 없다. 예: b'ABI='

hex	binary hex string	hex 형식의 상수는 8진수(8비트 바이트) 배열을 16진수로 인코딩한다. 문자열은 숫자 '0'~'9'와 문자 'a'~'f'인 짝수의 16진 문자로 구성해야 한다. 16진 문자열은 아포스트로피(''' U+0027)로 둘러싸고 앞에 'h'(U+0068)를 덧붙여야 한다. 예: h'0012'
integer	integer float	integer 형식의 상수는 부호 포함 십진수를 일상적인 방법으로 인코딩한다(예: 123). 양수의 경우 정수는 플러스 기호('+' U+002b)를 생략하고 있는 그대로 표시해야 한다. 음수는 앞에 하이픈('-' U+002d)을 추가하여 표시해야 한다. 사이버 관측 가능 float에 비교할 때 전체 값을 비교하고 자르지 않아야 한다. 예를 들어 STIX 패턴화 상수 정수 값 1을 사이버 관측 가능 float 값 1.5에 비교한 결과는 같지 않다. 유효한 값의 범위는 STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts.에 정의되어 있다.
float	integer float	float 형식의 상수는 부동 소수점 수를 일상적인 방법으로 인코딩한다(예: 123.456). 양수 부동 소수점 수의 경우 부동 소수점 수는 플러스 기호('+' U+002b)를 생략하고 있는 그대로 표시해야 한다. 음수 부동 소수점 수는 앞에 하이픈('-' U+002d)을 추가하여 표시해야 한다. 유효한 값의 범위는 STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts에 정의되어 있다.
string	string binary hex	string 형식의 상수는 문자열을 아포스트로피(''' U+0027)로 둘러싼 유니코드 코드 점의 목록으로 인코딩한다. 이스케이프 문자는 백슬래시('₩' U+005c)이다. 아포스트로피 또는 백슬래시만 뒤에 올 수 있으며 이 경우 해당 문자를 순서에 사용한다. 문자열이 (U+0100)보다 작은 코드 점만 포함하고 있는 경우 문자열을 이전 형식 값으로 변환할 수 있다(비교를 위해 필요). 매핑은 코드 점 U+0000 ~ 00 ~ U+00ff ~ ff이다.
timestamp	timestamp	timestamp 형식의 상수는 STIX 타임스탬프(STIX™ Version 2.0 Part 1: STIX Core Concepts의 섹션 2.10 참조)를 문자열로 인코딩한다. 타임스탬프 문자열은 아포스트로피(''' U+0027)로 둘러싸고 앞에 't'(U+0074)를 덧붙여야 한다. 예: t'2014-01-13T07:03:17Z'

Ⅰ-5.3 STIX 패턴

STIX 패턴은 단순 키 값 비교에서 더 복잡하고 문맥이 적용되는 식까지 여러가지 요소로 구성된다. 가장 기본적인 구성 요소는 비교 연산자를 사용하여 사이버 관측 가능 객체의 단일 속성과 지정된 상수를 비교하는 비교 식이다. 간단한 예로 다음과 같은 비교 식(관측 가능 식 내에 포함됨)을 사용하여 IPv4 주소에 대조할 수 있다.

예제

```
[ipv4-addr:value = '198.51.100.1/32']
```

STIX 패턴의 다음 구성 요소는 부울 연산자로 결합하고 대괄호로 둘러싼 하나 이상의 비교 식으로 구성되는 관측 식이 있다. 관측 식은 비교 식으로 지정한 사이버 관측 가능 객체를 가진 집합을 선택하여 패턴을 대조하는 사이버 관측 가능 데이터의 집합(즉, 관측의 일부)을 구체화한다. 단일 비교 식으로 구성된 관측 식은 가장 기본적인 유효한 STIX 패턴이다. 앞의 예를 좀 더 확장하면 여러 IPv4 주소와 IPv6 주소에 대조하는 관측 식을 구성할 수 있다.

예제

```
[ipv4-addr:value = '198.51.100.1/32' OR ipv4-addr:value = '203.0.113.33/32' OR
ipv6-addr:value = '2001:0db8:dead:beef:dead:beef:dead:0001/128']
```

관측 식 다음에는 하나 이상의 한정자가 올 수 있으며, 이 한정자는 패턴을 대조하는 데이터의 집합에 추가적으로 제한사항을 적용하는 식을 만들 수 있다. 또한, 한정자를 사용하여 IP 주소를 반복에서 여러 번 관측해야 한다는 것을 나타낼 수 있다.

예제

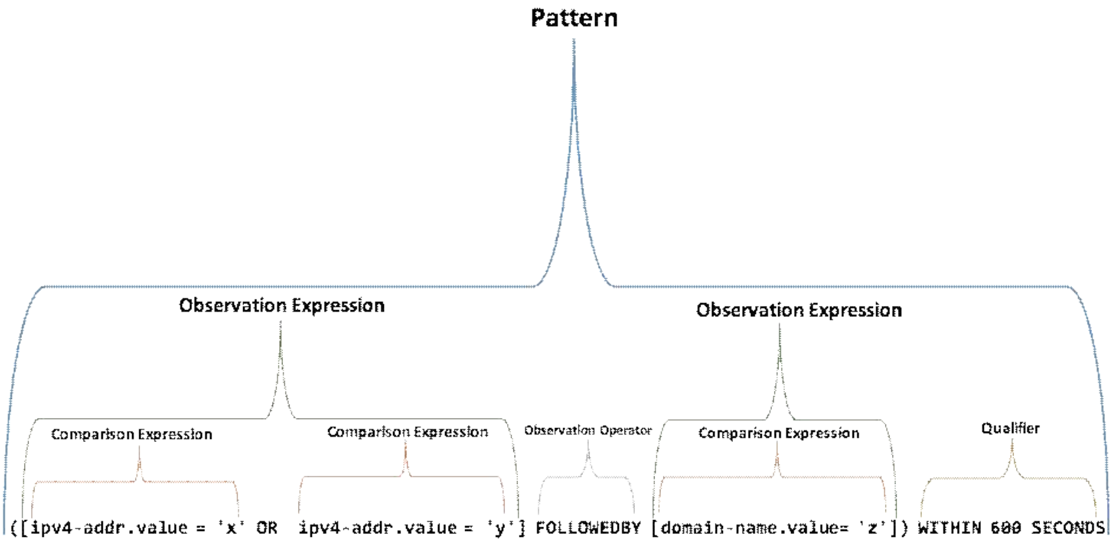
```
[ipv4-addr:value = '198.51.100.1/32' OR ipv4-addr:value = '203.0.113.33/32' OR
ipv6-addr:value = '2001:0db8:dead:beef:dead:beef:dead:0001/128'] REPEATS 5
TIMES
```

가장 높은 수준의 STIX 패턴화 구성 요소는 관측 연산자를 통해 둘 이상의 객체 식을 조합하여 여러 STIX 관측 대상 데이터 SDO에 대해 대조할 수 있는 STIX 패턴을 만든다. 앞의 예를 좀 더 확장하여 구성하면 관측 연산자를 사용하여 특정 도메인 이름의 관측에 이어서 IP 주소를 관측한다는 것을 지정하고, 다른 한정자를 사용하여 IP 주소와 도메인 이름을 둘 다 특정 시간 범위 내에서 관측해야 한다는 것도 함께 지정할 수 있다.

예제

```
([ipv4-addr:value = '198.51.100.1/32' OR ipv4-addr:value = '203.0.113.33/32' OR
ipv6-addr:value = '2001:0db8:dead:beef:dead:beef:dead:0001/128'] FOLLOWEDBY
[domain-name:value = 'example.com']) WITHIN 600 SECONDS
```

아래 개요도는 위의 예에 나오는 여러 STIX 패턴화 구성 요소의 발췌된 버전을 나타낸다.



(그림 I-5.3-1) STIX 패턴화 개요도

Ⅰ-5.4 패턴 식

패턴 식은 참(true) 또는 거짓(false)로 평가되며, 관측 연산자로 결합한 하나 이상의 관측 식으로 구성된다. 패턴 식은 특정 관측의 집합을 기준으로 평가된다. 그러한 관측 중 하나 이상이 패턴 식과 일치하면 true로 평가된다. 일치하는 관측이 없으면 패턴 식은 false로 평가된다.

패턴 식은 유니코드 문자열로 인코딩해야 한다.

패턴 문자열 내의 공백(즉, WSpace=Y인 경우 유니코드 점)은 키워드, 상수 및 필드 객체를 포함한 패턴의 부분을 구분하기 위해 사용된다. 연산자 사이의 공백 문자(줄 바꿈과 캐리지 리턴 포함)를 허용해야 한다. 연달아 있는 여러 개의 공백 문자는 한 개로 취급해야 한다.

구문 분석 오류 또는 유효하지 않은 상수를 발생시키는 유효하지 않은 패턴(예: 유효하지 않은 16진수 또는 이진 상수)을 관측과 대조해서는 안 된다.

Ⅰ-5.4.1 관측 식(Observation Expression)

관측 식은 부울 연산자를 통해 결합된 하나 이상의 비교 식으로 구성된다.

관측 식은 왼쪽 대괄호('[' U+005b) 및 오른쪽 대괄호(']' U+005d)로 구분해야 한다. 하나 이상의 관측 식 한정자를 닫는 대괄호 또는 관측 식의 닫는 소괄호 뒤에 제공할 수 있다. 관측 식을 관측 연산자로 결합할 수 있다.

개별 관측 식(예: [a = b])은 단일 관측, 즉 단일 STIX 관측 대상 데이터 인스턴스를 대조한다. 여러 관측에 대조할 필요가 있는 경우 관측 연산자를 통해 둘 이상의 관측 식을 조합하여 패턴을 서로 별개인 관측 두 개 이상을 기준으로 평가해야 한다는 것을 나타낼 수 있다.

관측 식에 관측을 대조하는 경우 관측 식 내에 포함된 모든 비교 식을 같은 사이버 관측 가능 객체(참조된 객체 포함)에 대조해야 한다. 관측 식이 서로 다른 객체 형식에 기초한 객체 경로를 가진 비교 식을 포함할 수 있지만 그러한 비교 식은 OR에 의해 결합해야 한다. AND를 사용하는 관측 식의 비교 식은 같은 기본 객체 경로를 사용해야 한다.

예제

```
[(type-a:property-j = 'W' AND type-a:property-k = 'X') OR (type-b:property-m =
'Y' AND type-b:property-n = 'Z')]
```

이 식은 관측 가능 객체를 type-a 또는 type-b의 객체와 대조하지만, 해당 특정 형식에

대한 두 비교 식 모두 같은 객체에 대해 true로 평가해야 한다. 단일 객체 형식을 대조하려는 비교 식을 AND 또는 OR에 의해 결합할 수 있다.

예제

```
[type-a:property-j = 'W' AND type-a:property-k = 'X' OR type-a:property-l = 'Z']
```

AND는 OR보다 우선 순위가 높으므로 앞의 예의 경우 관측이 `property-j = 'W'`와 `property-k = 'X'`를 둘 다 포함하거나 `property-l = 'Z'`만 포함해야 한다.

관측 식을 해당 관측 연산자 및 선택 사항 한정자와 함께 소괄호로 둘러싸서 한정자가 적용되는 관측 식을 구분할 수 있다.

예제

```
([ a ] AND [ b ] REPEATS 5 TIMES) WITHIN 5 MINUTES
```

앞의 예는 5분 기간에 모두가 일치하는 a와 5개의 b를 생성한다. 또 다른 예는 다음과 같다.

예제

```
([ a ] AND [ b ]) REPEATS 5 TIMES WITHIN 5 MINUTES
```

앞의 예는 5분 기간에 모두가 일치하는 5개의 *a*와 5개의 *b*(10개의 관측)를 생성한다.

Ⅰ-5.4.2 비교 식(Comparison Expression)

비교 식은 STIX 패턴화 중 가장 기본적인 구성 요소이며 객체 경로와 비교 연산자에 의해 결합된 상수를 비교한다. 각 비교 식은 단일 항목이므로 왼쪽에서 오른쪽 방향으로 평가한다.

부울 연산자는 두 개의 비교 식을 서로 결합한다. <표 Ⅰ-5.4-1>에서 a 또는 b는 비교 식 또는 부울 연산자로 결합되고 소괄호로 둘러싸인 둘 이상의 비교 식으로 구성된 복합 식(재귀적으로 구성될 수 있음)이다.

<표 I-5.4-1> 비교 식

부울 연산자	설명	연산 방향
a AND b	a 및 b는 비교 식 또는 부울 연산자로 결합되고 소괄호로 둘러싸인 둘 이상의 비교 식으로 구성된 복합 식(재귀적으로 구성될 수 있음)이어야 한다. a 및 b는 같은 관측에 대해 둘 다 true로 평가되어야 한다.	왼쪽에서 오른쪽으로
a OR b	a 및 b는 비교 식 또는 부울 연산자로 결합되고 소괄호로 둘러싸인 둘 이상의 비교 식으로 구성된 복합 식(재귀적으로 구성될 수 있음)이어야 한다. a 또는 b는 true로 평가되어야 한다.	왼쪽에서 오른쪽으로

Ⅰ-5.5 객체 경로 구문

I-5.5는 STIX 패턴 내에서 사이버 Observable 객체의 속성을 다루기 위한 구문을 설명한다. <표 I-5.5-1>와 같은 표기법은 STIX 패턴 전체에 걸쳐 사용된다.

<표 I-5.5-1> 객체 경로 구문

표기법	정의
<object-type>	대조하는 기준 사이버 Observable 객체의 형식. 이는 관측의 지정된 사이버 관측 가능 객체에 대해 지정된 type 필드의 값이어야 한다.
<property_name>	대조하는 기준 사이버 Observable 속성의 이름. 이는 <object-type> 표기법에서 참조한 사이버 관측 가능 객체 형식의 정의에 지정한 유효한 속성 이름이어야 한다. <property_name>이 하이픈('-' U+002d) 또는 마침표('.' U+002e)를 포함하는 경우 <property_name>을 아포스트로피(''' U+0027)로 둘러싸야 한다. 중첩된 속성(즉, 사이버 Observable 객체에서 다른 속성의 자식)은 구문 <property_name>.<property_name>을 사용하여 지정해야 하며, 여기서 ‘.' 앞의 <property_name>은 상위 속성의 이름이고 그 뒤의 것은 하위 속성의 이름이다. 속성 이름이 다른 사이버 Observable 객체에 대한 참조인 경우, 참조된 객체는 해당 속성이 마치 참조된 객체에 중첩된 것처럼 기능하도록 참조 해제해야 한다. 예를 들어 네트워크 객체 참조의 src_ref 속성이 IPv4 주소 객체이면 이 IPv4 주소의 값은 network-traffic:src_ref.value에 의해 지정할 수 있다.

Ⅰ-5.5.1 기본 객체 속성(Basic Object Properties)

사이버 Observable 객체에 직접 지정하는 dictionary 및 list 이외의 속성은 다음과 같이 표현한다.

구문

```
<object-type>:<property_name>
```

예

```
file:size
```

Ⅰ-5.5.2 목록 객체 속성(List Object Properties)

list 데이터 형식을 사용하는 사이버 Observable 객체에 대한 속성은 다음과 같이 표현한다.

구문

```
<object-type>:<property_name>[list_index].<property_name>
```

첫 번째 property_name은 list 형식의 객체 속성의 이름이어야 하며 [list_index]는 다음 중 하나이어야 한다.

- 0~N-1 범위의 정수, 여기서 N은 목록의 길이. *list_index*가 범위를 벗어난 경우 어떤 연산의 결과도 false이다.
- 리터럴 '*'는 목록에 포함된 항목을 비교 식에 대조하는 경우 비교 식이 true로 평가된다는 것을 나타낸다.

예제

```
file:extensions.windows-pebinary.sections[*].entropy > 7.0
```

위의 예는 PE 섹션이 7.0보다 큰 값의 엔트로피 속성을 가지면 true를 반환한다.

Ⅰ-5.5.3 사전 객체 속성(Dictionary Object Properties)

dictionary 데이터 형식을 사용하는 사이버 Observable 객체에 대한 속성은 다음과 같이 나타낸다.

구문

```
<object-type>:<property_name>.<key_name>
```

<property_name>은 형식 dictionary의 객체 속성의 이름이어야 하며 <key_name>은 사전의 키 이름이어야 한다.

예제

```
file:hashes.ssdeep
file:extensions.raster-image.image_height
```

Ⅰ-5.5.4 객체 참조 속성(Object Reference Properties)

object-ref 데이터 형식을 사용하는 사이버 관측 가능 객체에 대한 속성이며 단일 항목 또는 목록(즉, object-ref 형식의 list)이다.

구문

```
<object-type>:<property_name>.<dereferenced_object_property>
```

<property_name>은 object-ref 형식의 객체 속성의 이름이어야 하며 <dereferenced_object_property>는 참조 해제된 객체(즉, <property_name>을 통해 참조된 관측의 객체)의 유효한 속성의 이름이어야 한다.

<property_name>이 object-ref 형식의 list인 경우 해당 구문이 적용된다.

구문

```
<object-type>:<property_name>[list_index].<dereferenced_object_property>
```

따라서 이 경우 섹션 5.2에 정의된 목록 색인에 대해 같은 어의가 적용된다.

예제

```
email-message:from_ref.value = 'mary@example.com'
directory:contains_refs[*].name = 'foobar.dll'
```

Ⅰ-5.6 예제

참고: 아래 예제는 JSON 인코딩되지 않았다. 즉, 일부 문자(큰따옴표 등)는 JSON 문자열로 인코딩될 때에는 이스케이프되지만 이 경우는 이스케이프되지 않는다.

SHA-256 해시로 파일 대조

```
[file:hashes.'SHA-256' =
'aec070645fe53ee3b3763059376134f058cc337247c978add178b6ccdfb0019f']
```

정규식을 사용하여 특정 보낸 이메일 주소 및 첨부 파일 이름으로 이메일 메시지 대조

```
[email-message:from_ref.value MATCHES '.+₩₩@example₩₩.com$' AND
email-message:body_multipart[*].body_raw_ref.name MATCHES '^Final
Report.+₩₩.exe$']
```

SHA-256 해시와 PDF MIME 형식으로 파일 대조

```
[file:hashes.'SHA-256' =
'aec070645fe53ee3b3763059376134f058cc337247c978add178b6ccdfb0019f' AND
file:mime_type = 'application/x-pdf']
```

SHA-256 또는 MD5 해시로 파일 대조(예: MD5 또는 SHA-256을 생성하는 서로 다른 두 끝점 도구의 경우) 및 서로 다른 두 관측을 기준으로 서로 다른 SHA-256 해시를 가진 서로 다른 파일

```
[file:hashes.'SHA-256' = 
'bf07a7fbb825fc0aae7bf4a1177b2b31fcf8a3feeaf7092761e18c859ee52a9c' OR 
file:hashes.MD5 = 'cead3f77f6cda6ec00f57d76c9a6879f']

 AND [file:hashes.'SHA-256' = 

'aec070645fe53ee3b3763059376134f058cc337247c978add178b6ccdfb0019f']
```

MD5 해시에 이어서 (일시적으로) 5분 이내에 값을 대조하는 레지스트리 키 객체로 파일 대조

```
([file:hashes.MD5 = '79054025255fb1a26e4bc422aef54eb4'] FOLLOWEDBY 
[win-registry-key:key = 'HKEY_LOCAL_MACHINE₩₩foo₩₩bar']) WITHIN 300 
SECONDS
```

세 개의 서로 다르지만 구체적인 Unix 사용자 계정 대조

```
[user-account:account_type = 'unix' AND user-account:user_id = '1007' AND 
user-account:account_login = 'Peter'] AND [user-account:account_type = 'unix' 
AND user-account:user_id = '1008' AND user-account:account_login = 'Paul'] 
AND [user-account:account_type = 'unix' AND user-account:user_id = '1009' AND 
user-account:account_login = 'Mary']
```

아티팩트 객체 PCAP 페이로드 헤더 대조

```
[artifact:mime_type = 'application/vnd.tcpdump.pcap' AND artifact:payload_bin 
MATCHES '₩₩xd4₩₩xc3₩₩xb2₩₩xa1₩₩x02₩₩x00₩₩x04₩₩x00']
```

Windows 파일 경로로 파일 객체 대조

```
[file:name = 'foo.dll' AND file:parent_directory_ref.path = 
'C:₩₩Windows₩₩System32']
```

높은 섹션 엔트로피로 Windows PE 파일에 대해 대조

```
[file:extensions.windows-pebinary-ext.sections[*].entropy > 7.0]
```

파일 객체 매직 넘버와 MIME 형식 간의 불일치에 대해 대조

```
[file:mime_type = 'image/bmp' AND file:magic_number_hex = h'ffd8']
```

네트워크 트래픽에 대해 특정 대상으로 대조

```
[network-traffic:dst_ref.type = 'ipv4-addr' AND network-traffic:dst_ref.value =
'203.0.113.33/32']
```

도메인 이름에 대한 멀웨어 표지에 대해 대조

```
[network-traffic:dst_ref.type = 'domain-name' AND network-traffic:dst_ref.value =
'example.com'] REPEATS 5 TIMES WITHIN 1800 SECONDS
```

도메인 이름에 대해 IPv4 확인으로 대조

```
[domain-name:value = 'www.5z8.info' AND domain-name:resolves_to_refs[*].value
= '198.51.100.1/32']
```

URL에 대해 대조

```
[url:value = 'http://example.com/foo' OR url:value = 'http://example.com/bar']
```

X509 인증서에 대해 대조

```
[x509-certificate:issuer = 'CN=WEBMAIL' AND x509-certificate:serial_number =
'4c:0b:1d:19:74:86:a7:66:b4:1a:bf:40:27:21:76:28']
```

Windows 레지스트리 키에 대해 대조

```
[windows-registry-key:key =
'HKEY_CURRENT_USER₩₩Software₩₩CryptoLocker₩₩Files' OR
windows-registry-key:key =
'HKEY_CURRENT_USER₩₩Software₩₩Microsoft₩₩CurrentVersion₩₩Run₩₩Crypt
oLocker_0388']
```

파일에 대해 속성 집합으로 대조

```
[(file:name = 'pdf.exe' OR file:size = '371712') AND file:created =
t'2014-01-13T07:03:17Z']
```

이메일 메시지에 대해 특정 발신자와 주체로 대조

```
[email-message:sender_ref.value = 'jdoe@example.com' AND
email-message:subject = 'Conference Info']
```

사용자 지정 USB 장치에 대해 대조

```
[x-usb-device:usbdrive.serial_number = '575833314133343231313937']
```

특정 시간 범위 이내에 명령줄 인수의 특정 집합으로 시작한 두 프로세스에 대해 대조

```
[process:command_line MATCHES '^.+>-add GlobalSign.cer -c -s -r
localMachine Root$'] FOLLOWEDBY [process:command_line MATCHES'^.+>-add
GlobalSign.cer -c -s -r localMachineTrustedPublisher$'] WITHIN 300 SECONDS
```

특정 서브넷의 일부인 네트워크 트래픽 IP에 대한 대조

```
[network-traffic:dst_ref.value ISSUBSET
'2001:0db8:dead:beef:0000:0000:0000:0000/64']
```

멀웨어 아티팩트의 서로 다른 여러 조합에 대해 대조. 참고로 다음 패턴의 경우 파일과 레지스트리 키가 둘 다 존재하거나 두 프로세스 중 하나가 존재해야 한다.

```
([file:name = 'foo.dll'] AND [win-registry-key:key =
'HKEY_LOCAL_MACHINE₩₩foo₩₩bar']) OR [process:name = 'fooproc' OR
process:name = 'procfoo']
```

Ⅰ-5.7 적합성

STIX 패턴화 언어의 구현자가 언어에서 제공되는 전체 기능을 지원할 필요는 없다. 오히려 구현자는 자신의 제품 기능을 기준으로 타당한 만큼의 STIX 패턴화를 지원해야 하며, 특정 사용 사례에 필요한 최소한의 적합성 레벨만 지원하면 된다. 예를 들어 멀웨어 네트워크 트래픽을 검색하는 NIDS(네트워크 침입 감지 시스템, Network Intrusion Detection System)의 공급자가 네트워크 트래픽과 IP 주소를 명시적으로 대조하려면 비교 연산자를 구현하고 기본적인 관측 식만 지원하면 된다.

STIX 패턴화 언어 사양은 STIX 사이버 Observable 객체 데이터 모델과 결합되지만, 많은(또는 심지어 대부분의) 구현에서 STIX 패턴은 다른 독자적 쿼리 형식으로 코드 변환하는 추상 레이어로 사용된다. STIX 패턴은 STIX 관측 대상 데이터 인스턴스의 집합에 대해 직접 평가될 수 있지만, 예를 들어 패킷 검사 장치용 일부 쿼리 구문으로 변환될 수도 있다. 이 두 번째의 경우 STIX 패턴은 사실 STIX 사이버 관측 가능 객체 형식이 아닌 데이터의 컨텍스트에서 평가된다.

STIX 패턴화 언어의 관측 연산자를 사용하여 여러 관측에 걸쳐 명시적으로 대조하는 패턴을 만들 수 있지만, 이 언어는 각 관측에 대한 기반 데이터의 소스에 관하여 의도적으로 아무 것도 지정하지 않는다. 예를 들어 특정 패턴화 구현에 따라 네트워크 트래픽에 대해 대조하는 패턴의 데이터는 끝점 또는 NIDS에서 올 수 있다. 지정된 패턴 내의 각 관측에 대해 적합한 데이터 소스(적용 가능한 경우)를 확인하는 것은 구현자의 의무이다.

Ⅰ-5.7.1 패턴 생산자(Pattern Producer)

STIX 패턴을 생성하는 소프트웨어를 "패턴 생산자"라고 한다. 그러한 소프트웨어는 모든 표준 설명문과 이 문서의 형식 지정 규칙을 준수하는 패턴의 생성을 지원해야 한다. 패턴 생산자는 적합성 레벨에 의해 자신의 적합성을 지정해야 한다.

Ⅰ-5.7.2 패턴 소비자(Pattern Consumer)

STIX 패턴을 소비하는 소프트웨어를 "패턴 소비자"라고 한다. 그러한 소프트웨어는 모든 표준 설명문과 이 문서의 형식 지정 규칙을 준수하는 패턴의 소비를 지원해야 한다. 패턴 소비자는 적합성 레벨에 의해 자신의 적합성을 지정해야 한다.

Ⅰ-5.7.3 적합성 레벨(Conformance Levels)

Ⅰ-5.7.3.1 레벨 1: 기본 적합성(Level 1: Basic Conformance)

패턴화 사양의 최소 필요 특성을 준수하는 소프트웨어를 “레벨 1 STIX 패턴화 구현”이라 한다.

그러한 소프트웨어는 참조 대상 섹션의 모든 표준 명령문과 동작을 준수하여 다음과 같은 기능을 지원해야 한다.

- 섹션 4.1에서 설명한 단일 관측 식(한정자 생략)
- 섹션 4.2.1에서 설명한 모든 비교 연산자

적합성 레벨은 WITHIN 같은 한정자를 지원하기 위해 필요하므로 주로 엔드포인트 또는 네트워크 경계에 배포되고 아키텍처 상으로 상태를 유지할 수 없는 소프트웨어를 위한 것이다.

Ⅰ-5.7.3.2 레벨 2: 기본 적합성 + 관측 연산자(Level 2: Basic Conformance plus Observation Operators)

패턴화 사양의 최소 필요 특성을 지원하지만 여러 관측에 대해 작용할 수 있는 소프트웨어를 “레벨 2 STIX 패턴화 구현”이라 한다.

그러한 소프트웨어는 참조 대상 섹션의 모든 표준 명령문과 동작을 준수하여 다음과 같은 기능을 지원해야 한다.

- 섹션 4.1에서 설명한 단일 및 복합 관측 식(한정자 생략)
- 섹션 4.2.1에서 설명한 모든 비교 연산자
- 섹션 4.1.2에서 설명한 AND 관측 연산자
- 섹션 4.1.2에서 설명한 OR 관측 연산자

이 적합성 레벨은 주로 별도의 관측에 대해 패턴을 검색할 수 있지만 일시적 패턴화를

지원하지 않을 수 있는 HIDS 같은 소프트웨어를 위한 것이다.

Ⅰ-5.7.3.3 레벨 3: 완전한 적합성(Level 3: Full Conformance)

패턴화 사양의 모든 기능을 완전히 준수하는 소프트웨어를 "레벨 3 STIX 패턴화 구현"이라 한다.

그러한 소프트웨어는 참조 대상 섹션의 모든 표준 명령문과 동작을 준수하여 다음과 같은 기능을 지원해야 한다.

- 섹션 2. 정의
- 섹션 3. STIX 패턴
- 섹션 4. 패턴 식
- 섹션 5. 객체 경로 구문

이 레벨의 적합성은 주로 일시적 패턴화를 지원하지만 관측의 다양한 소스에 걸쳐 패턴을 집계하고 검색할 수도 있는 소프트웨어를 위한 것이다.

부 록 Ⅰ-6

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

표준의 이력

판수	채택일	표준번호	내용	담당 위원회
제1판	2018.xx.xx	제정 TTAx.xx-xx.xxxx		사이버보안 프로젝트 그룹(PG503), 정보보호 기술위원회(TC5)