STIX™ Version 2.0. Part 4: Cyber Observable Objects

Committee Specification Draft 02 / Public Review Draft 02

03 May 2017

Specification URIs

This version:

http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part4-cyber-observable-objects/stix-v2.0-csprd02-part4-cyber-observable-objects.docx (Authoritative)
http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part4-cyber-observable-objects/stix-v2.0-csprd02-part4-cyber-observable-objects.html
http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part4-cyber-observable-objects/stix-v2.0-csprd02-part4-cyber-observable-objects.pdf

Previous version:

http://docs.oasis-open.org/cti/stix/v2.0/csprd01/part4-cyber-observable-objects/stix-v2.0-csprd01-part4-cyber-observable-objects.docx (Authoritative)
http://docs.oasis-open.org/cti/stix/v2.0/csprd01/part4-cyber-observable-objects/stix-v2.0-csprd01-part4-cyber-observable-objects.html
http://docs.oasis-open.org/cti/stix/v2.0/csprd01/part4-cyber-observable-objects/stix-v2.0-csprd01-part4-cyber-observable-objects.pdf

Latest version:

http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part4-cyber-observable-objects.docx (Authoritative)
http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part4-cyber-observable-objects.html
http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part4-cyber-observable-objects.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:

Trey Darley (trey@kingfisherops.com), Kingfisher Operations, sprl
Ivan Kirillov (ikirillov@mitre.org), MITRE Corporation

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/csprd02/part1-stix-core/stix-v2.0-csprd02-part1-stix-core.html.
- *STIX™ Version 2.0. Part 2: STIX Objects.* http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part2-stix-objects/stix-v2.0-csprd02-part2-stix-objects.html.
- *STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts.* http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part3-cyber-observable-core/stix-v2.0-csprd02-part3-cyber-observable-core.html.

- (this document) *STIX™ Version 2.0. Part 4: Cyber Observable Objects.* http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part4-cyber-observable-objects/stix-v2.0-csprd02-part4-cyber-observable-objects.html.
- *STIX™ Version 2.0. Part 5: STIX Patterning.* http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part5-stix-patterning/stix-v2.0-csprd02-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™) is a language for expressing cyber threat and observable information. This document defines a set of cyber observable objects that can be used in STIX and elsewhere.

Citation format:

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

[STIX-v2.0-Pt4-Cyb-Objects]

STIX™ Version 2.0. Part 4: Cyber Observable Objects. Edited by Trey Darley and Ivan Kirillov. 03 May 2017. OASIS Committee Specification Draft 02 / Public Review Draft 02. http://docs.oasis-open.org/cti/stix/v2.0/csprd02/part4-cyber-observable-objects/stix-v2.0-csprd02-part4-cyber-observable-objects.html. Latest version: http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part4-cyber-observable-objects.html.

Notices

Table of Contents

1 Introduction

The STIX 2.0 specification defines structured representations for observable objects and their properties in the cyber domain. These can be used to describe data in many different functional domains, including but not limited to:

- Malware characterization
- Intrusion detection
- Incident response & management
- Digital forensics

STIX Cyber Observables document the facts concerning **what** happened on a network or host, but not necessarily the who or when, and never the why. For example, information about a file that existed, a process that was observed running, or that network traffic occurred between two IPs can all be captured as Cyber Observable data.

STIX Cyber Observables are used by various STIX Domain Objects (SDOs) to provide additional context to the data that they characterize. The Observed Data SDO, for example, indicates that the raw data was observed at a particular time and by a particular entity.

The Cyber Observable Objects chosen for inclusion in STIX 2.0 represent a minimally viable product (MVP) that fulfills basic consumer and producer requirements. Objects and properties not included in STIX 2.0, but deemed necessary by the community, will be included in future releases.

This document (*STIX™ Version 2.0. Part 4: Cyber Observable Objects*) contains the definitions for the various Cyber Observable Objects.

1.0.1 IPR Policy

This Committee Specification Public Review Draft is provided under the Non-Assertion Mode of the OASIS IPR Policy, 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

[Character Sets]	N. Freed and M. Dürst, "Character Sets", IANA, December 2013, [Online]. Available: http://www.iana.org/assignments/character-sets/character-sets.xhtml
[IPFIX]	IANA, "IP Flow Information Export (IPFIX) Entities", December 2016, [Online]. Available: http://www.iana.org/assignments/ipfix/ipfix.xhtml

[ISO639-2] "ISO 639-2:1998 Codes for the representation of names of languages -- Part 2: Alpha-3 code", 1998. [Online]. Available: http://www.iso.org/iso/catalogue_detail?csnumber=4767

[Media Types] N. Freed, M. Kucherawy, M. Baker and B. Hoehrmann, "Media Types", IANA, December 2016. [Online]. Available: http://www.iana.org/assignments/media-types/media-types.xhtml

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, http://www.rfc-editor.org/info/rfc1034.

[RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, DOI 10.17487/RFC2047, November 1996, http://www.rfc-editor.org/info/rfc2047.

[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.rfc-editor.org/info/rfc2119.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, http://www.rfc-editor.org/info/rfc3986.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, http://www.rfc-editor.org/info/rfc5322.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, http://www.rfc-editor.org/info/rfc5890.

[Port Numbers] J.Touch, A. Mankin, E. Kohler, et. al., "Service Name and Transport Protocol Port Number Registry", IANA, January 2017. [Online]. Available: http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

[NVD] Official Common Platform Enumeration (CPE) Dictionary, National Vulnerability Database [Online]. Available: https://nvd.nist.gov/cpe.cfm

[X.509] X.509 : Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks, ITU, October 2016. [Online]. Available: https://www.itu.int/rec/T-REC-X.509/

1.3 Naming Requirements

1.3.1 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.

1.3.2 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.

1.4 Document Conventions

1.4.1 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 dash (-). All type names, property names, object names, and vocabulary terms are between three and 250 characters long.

1.4.2 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 Defined Object Data Models

2.1 Artifact Object

Type Name: `artifact`

The Artifact Object permits capturing an array of bytes (8-bits), as a base64-encoded string, or linking to a file-like payload. The size of the base64-encoded data captured in the **`payload_bin`** property **MUST** be less than or equal to 10MB.

One of **`payload_bin`** or **`url`** **MUST** be provided. It is incumbent on object creators to ensure that the URL is accessible for downstream consumers. If a URL is provided, then the **`hashes`** property **MUST** contain the hash of the URL contents.

2.1.1 Properties

Common Properties		
`type, extensions`		
Artifact Object Specific Properties		
`mime_type, payload_bin, url, hashes`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `artifact`.
`mime_type` (optional)	`string`	The value of this property **MUST** be a valid MIME type as specified in the IANA Media Types registry [Media Types].
`payload_bin` (optional)	`binary`	Specifies the binary data contained in the artifact as a base64-encoded string. This property **MUST NOT** be present if url is provided.
`url` (optional)	`string`	The value of this property **MUST** be a valid URL that resolves to the unencoded content. This property **MUST NOT** be present if **`payload_bin`** is provided.
`hashes` (optional)	`hashes`	Specifies a dictionary of hashes for the contents of the **url** or the **`payload_bin`**. This **MUST** be provided when the **url** property is present.

Examples

Basic Image Artifact

```
{
  "0": {
    "type": "artifact",
    "mime_type": "image/jpeg",
    "payload_bin": "VBORw0KGgoAAAANSUhEUgAAADI== ..."
    }
}
```

2.2 AS Object

Type Name: `autonomous-system`

The AS object represents the properties of an Autonomous System (AS).

2.2.1 Properties

Common Properties		
`type, extensions`		
AS Object Specific Properties		
`number, name, rir`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `autonomous-system`.
`number` (required)	`integer`	Specifies the number assigned to the AS. Such assignments are typically performed by a Regional Internet Registry (RIR).
`name` (optional)	`string`	Specifies the name of the AS.
`rir` (optional)	`string`	Specifies the name of the Regional Internet Registry (RIR) that assigned the number to the AS.

Examples

Basic AS Object

```
{
  "0": {
    "type": "autonomous-system",
    "number": 15139,
    "name": "Slime Industries",
    "rir": "ARIN"
  }
}
```

2.3 Directory Object

Type Name: `directory`

The Directory Object represents the properties common to a file system directory.

2.3.1 Properties

Common Properties		
`type, extensions`		
File Object Specific Properties		
`path, path_enc, created, modified, accessed, contains_refs`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `directory`.
`path` (required)	`string`	Specifies the path, as originally observed, to the directory on the file system.
`path_enc` (optional)	`string`	Specifies the observed encoding for the path. The value **MUST** be specified if the path is stored in a non-Unicode encoding. This value **MUST** be specified using the corresponding name from the 2013-12-20 revision of the IANA character set registry [Character Sets]. If the preferred MIME name for a character set is defined, this value **MUST** be used; if it is not defined, then the Name value from the registry **MUST** be used instead.
`created` (optional)	`timestamp`	Specifies the date/time the directory was created.
`modified` (optional)	`timestamp`	Specifies the date/time the directory was last written to/modified.
`accessed` (optional)	`timestamp`	Specifies the date/time the directory was last accessed.
`contains_refs` (optional)	`list` of type `object-ref`	Specifies a list of references to other File and/or Directory Objects contained within the directory. The objects referenced in this list **MUST** be of type `file` or `directory`.

Examples

Basic directory

```
{
  "0": {
    "type": "directory",
    "path": "C:\\Windows\\System32"
    }
}
```

2.4 Domain Name Object

Type Name: `domain-name`

The Domain Name represents the properties of a network domain name.

2.4.1 Properties

Common Properties		
`type, extensions`		
Domain Name Object Specific Properties		
`value, resolves_to_refs`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `domain-name`.
`value` (required)	`string`	Specifies the value of the domain name. The value of this property **MUST** conform to [RFC1034], and each domain and sub-domain contained within the domain name **MUST** conform to [RFC5890].
`resolves_to_refs` (optional)	`list` of type `object-ref`	Specifies a list of references to one or more IP addresses or domain names that the domain name resolves to. The objects referenced in this list **MUST** be of type `ipv4-addr` or `ipv6-addr` or `domain-name` (for cases such as CNAME records).

Examples

Basic FQDN

```
{
  "0": {
    "type": "domain-name",
    "value": "example.com",
    "resolves_to_refs": [
      "1"
    ]
  },
  "1": {
    "type": "ipv4-addr",
    "value": "198.51.100.2"
  }
}
```

2.5 Email Address Object

Type Name: `email-addr`

The Email Address Object represents a single email address.

2.5.1 Properties

Common Properties		
`type, extensions`		
Email Address Object Specific Properties		
`value, display_name, belongs_to_ref`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `email-addr`.
`value` (required)	`string`	Specifies a single email address. This **MUST NOT** include the display name. This property corresponds to the *addr-spec* construction in section 3.4 of [RFC5322], for example, `jane.smith@example.com`.
`display_name` (optional)	`string`	Specifies a single email display name, i.e., the name that is displayed to the human user of a mail application. This property corresponds to the *display-name* construction in section 3.4 of [RFC5322], for example, `Jane Smith`.
`belongs_to_ref` (optional)	`object-ref`	Specifies the user account that the email address belongs to, as a reference to a User Account Object.

		The object referenced in this property **MUST** be of type `user-account`.

Examples

Basic Email Address

```
{
  "0": {
    "type": "email-addr",
    "value": "john@example.com",
    "display_name": "John Doe"
  }
}
```

2.6 Email Message Object

Type Name: `email-message`

The Email Message Object represents an instance of an email message, corresponding to the internet message format described in [RFC5322] and related RFCs.

Header field values that have been encoded as described in section 2 of [RFC2047] **MUST** be decoded before inclusion in Email Message Object properties. For example, `this is some text` **MUST** be used instead of `=?iso-8859-1?q?this=20is=20some=20text?=`. Any characters in the encoded value which cannot be decoded into Unicode **SHOULD** be replaced with the 'REPLACEMENT CHARACTER' (U+FFFD). If it is necessary to capture the header value as observed, this can be achieved by referencing an Artifact Object through the **`raw_email_ref`** property.

2.6.1 Properties

Common Properties		
`type, extensions`		
Email Message Object Specific Properties		
`is_multipart, date, content_type, from_ref, sender_ref, to_refs, cc_refs, bcc_refs, subject, received_lines, additional_header_fields, body, body_multipart, raw_email_ref`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `email-message`.

`is_multipart` (required)	`boolean`	Indicates whether the email body contains multiple MIME parts.
`date` (optional)	`timestamp`	Specifies the date/time that the email message was sent.
`content_type` (optional)	`string`	Specifies the value of the "Content-Type" header of the email message.
`from_ref` (optional)	`object-ref`	Specifies the value of the "From:" header of the email message. The "From:" field specifies the author(s) of the message, that is, the mailbox(es) of the person(s) or system(s) responsible for the writing of the message. The object referenced in this property **MUST** be of type `email-address`.
`sender_ref` (optional)	`object-ref`	Specifies the value of the "Sender" field of the email message. The "Sender:" field specifies the mailbox of the agent responsible for the actual transmission of the message. The object referenced in this property **MUST** be of type `email-address`.
`to_refs` (optional)	`list` of type `object-ref`	Specifies the mailboxes that are "To:" recipients of the email message. The objects referenced in this list **MUST** be of type `email-address`.
`cc_refs` (optional)	`list` of type `object-ref`	Specifies the mailboxes that are "CC:" recipients of the email message. The objects referenced in this list **MUST** be of type `email-address`.
`bcc_refs` (optional)	`list` of type `object-ref`	Specifies the mailboxes that are "BCC:" recipients of the email message. As per [RFC5322], this list may be empty, which should not be treated the same as the key being absent. The objects referenced in this list **MUST** be of type `email-address`.

subject (optional)	string	Specifies the subject of the email message.
received_lines (optional)	list of type string	Specifies one or more "Received" header fields that may be included in the email headers. List values **MUST** appear in the same order as present in the email message.
additional_header_fields (optional)	dictionary	Specifies any other header fields (except for **date**, **received_lines**, **content_type**, **from_ref**, **sender_ref**, **to_refs**, **cc_refs**, **bcc_refs**, and **subject**) found in the email message, as a dictionary. Each key/value pair in the dictionary represents the name/value of a single header field or names/values of a header field that occurs more than once. Each dictionary key **SHOULD** be a case-preserved version of the header field name. For cases where a header field occurs exactly once, the corresponding value for the dictionary key **MUST** be a string. For cases where a header field occurs more than once, the corresponding value for the dictionary key **MUST** be a list of type string, where each string in the list represents a single value of the header field.
body (optional)	string	Specifies a string containing the email body. This property **MAY** only be used if **is_multipart** is false.
body_multipart (optional)	list of type mime-part-type	Specifies a list of the MIME parts that make up the email body. This property **MAY** only be used if **is_multipart** is true.
raw_email_ref (optional)	object-ref	Specifies the raw binary contents of the email message, including both the headers and body, as a reference to an Artifact Object. The object referenced in this property **MUST** be of type artifact.

2.6.2 Email MIME Component Type

Type Name: mime-part-type

Specifies one component of a multi-part email body.

There is no property to capture the value of the “Content-Transfer-Encoding” header field, since the body **MUST** be decoded before being represented in the **body** property.

One of **body** OR **body_raw_ref** **MUST** be included.

2.6.2.1 2.6.2.1 Properties

Property Name	Type	Description
body (optional)	string	Specifies the contents of the MIME part if the **content_type** is not provided or starts with text/ (e.g., in the case of plain text or HTML email). For inclusion in this property, the contents **MUST** be decoded to Unicode. Note that the charset provided in **content_type** is for informational usage and not for decoding of this property.
body_raw_ref (optional)	object-ref	Specifies the contents of non-textual MIME parts, that is those whose **content_type** does not start with text/, as a reference to an Artifact Object or File Object. The object referenced in this property **MUST** be of type artifact or file. For use cases where conveying the actual data contained in the MIME part is of primary importance, artifact **SHOULD** be used. Otherwise, for use cases where conveying metadata about the file-like properties of the MIME part is of primary importance, file **SHOULD** be used.
content_type (optional)	string	Specifies the value of the “Content-Type” header field of the MIME part. Any additional “Content-Type” header field parameters such as charset **SHOULD** be included in this property. Example: `text/html; charset=UTF-8`
content_disposition (optional)	string	Specifies the value of the “Content-Disposition” header field of the MIME part.

Examples

Simple Email Message

```
{
  "0": {
    "type": "email-addr",
    "value": "jdoe@example.com",
    "display_name": "John Doe"
  },
  "1": {
    "type": "email-addr",
    "value": "mary@example.com",
    "display_name": "Mary Smith"
  },
  "2": {
    "type": "email-message",
    "from_ref": "0",
    "to_refs": ["1"],
    "is_multipart": false,
    "date": "1997-11-21T15:55:06.000Z",
    "subject": "Saying Hello"
  }
}
```

Simple Email Message with Additional Header Properties

```
{
  "0": {
    "type": "email-addr",
    "value": "joe@example.com",
    "display_name": "Joe Smith"
  },
  "1": {
    "type": "email-addr",
    "value": "bob@example.com",
    "display_name": "Bob Smith"
  },
  "2": {
    "type": "email-message",
    "from_ref": "0",
    "to_refs": [
      "1"
    ],
    "is_multipart": false,
    "date": "2004-04-19T12:22:23.000Z",
    "subject": "Did you see this?",
    "additional_header_fields": {
      "Reply-To": [
        "steve@example.com",
        "jane@example.com"
      ]
    }
  }
}
```

Complex MIME Email Message

```
{
  "0": {
    "type": "email-message",
    "is_multipart": true,
    "received_lines": [
      "from mail.example.com ([198.51.100.3]) by smtp.gmail.com with ESMTPSA id
q23sm23309939wme.17.2016.07.19.07.20.32 (version=TLS1_2 cipher=ECDHE-RSA-AES128-GCM-SHA256
bits=128/128); Tue, 19 Jul 2016 07:20:40 -0700 (PDT)"
    ],
    "content_type": "multipart/mixed",
    "date": "2016-06-19T14:20:40.000Z",
    "from_ref": "1",
    "to_refs": [
      "2"
    ],
    "cc_refs": [
      "3"
    ],
    "subject": "Check out this picture of a cat!",
    "additional_header_fields": {
      "Content-Disposition": "inline",
      "X-Mailer": "Mutt/1.5.23",
      "X-Originating-IP": "198.51.100.3"
    },
    "body_multipart": [
      {
        "content_type": "text/plain; charset=utf-8",
        "content_disposition": "inline",
        "body": "Cats are funny!"
      },
      {
        "content_type": "image/png",
        "content_disposition": "attachment; filename=\"tabby.png\"",
        "body_raw_ref": "4"
      },
      {
        "content_type": "application/zip",
        "content_disposition": "attachment; filename=\"tabby_pics.zip\"",
        "body_raw_ref": "5"
      }
    ]
  },
  "1": {
    "type": "email-addr",
    "value": "jdoe@example.com",
    "display_name": "John Doe"
  },
  "2": {
    "type": "email-addr",
    "value": "bob@example.com",
    "display_name": "Bob Smith"
  },
  "3": {
    "type": "email-addr",
    "value": "mary@example.com",
    "display_name": "Mary Jones"
  },
  "4":{
    "type": "artifact",
    "mime_type": "image/jpeg",
    "payload_bin": "VBORw0KGgoAAAANSUhEUgAAADI== ...",
    "hashes": {
      "SHA-256": "effb46bba03f6c8aea5c653f9cf984f170dcdd3bbbe2ff6843c3e5da0e698766"
    }
  },
  "5": {
    "type": "file",
    "name": "tabby_pics.zip",
    "magic_number_hex": "504B0304",
    "hashes": {
      "SHA-256": "fe90a7e910cb3a4739bed9180e807e93fa70c90f25a8915476f5e4bfbac681db"
    }
  }
}
```

2.7 File Object

Type Name: `file`

The File Object represents the properties of a file. A File Object **MUST** contain at least one of **`hashes`** or **`name`**.

2.7.1 Properties

Common Properties
`type, extensions`

File Object Specific Properties		
hashes, size, name, name_enc, magic_number_hex, mime_type, created, modified, accessed, parent_directory_ref, is_encrypted, encryption_algorithm, decryption_key, contains_refs, content_ref		

Property Name	Type	Description
type (required)	string	The value of this property **MUST** be file.
extensions (optional)	dictionary	The File Object defines the following extensions. In addition to these, producers **MAY** create their own. ntfs-ext, raster-image-ext, pdf-ext, archive-ext, windows-pebinary-ext Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.
hashes (optional)	hashes	Specifies a dictionary of hashes for the file.
size (optional)	integer	Specifies the size of the file, in bytes. The value of this property **MUST NOT** be negative.
name (optional)	string	Specifies the name of the file.
name_enc (optional)	string	Specifies the observed encoding for the name of the file. This value **MUST** be specified using the corresponding name from the 2013-12-20 revision of the IANA character set registry [Character Sets]. If the value from the Preferred MIME Name column for a character set is defined, this value **MUST** be used; if it is not defined, then the value from the Name column in the registry **MUST** be used instead. This property allows for the capture of the original text encoding for the file name, which may be forensically relevant; for example, a file on an NTFS volume whose name was created using the windows-1251 encoding, commonly used for languages based on Cyrillic script.
magic_number_hex (optional)	hex	Specifies the hexadecimal constant ("magic number") associated with a specific file format that corresponds to the file, if applicable.

mime_type (optional)	string	Specifies the MIME type name specified for the file, e.g., application/msword. Whenever feasible, this value **SHOULD** be one of the values defined in the Template column in the IANA media type registry [Media Types]. Maintaining a comprehensive universal catalog of all extant file types is obviously not possible. When specifying a MIME Type not included in the IANA registry, implementers should use their best judgement so as to facilitate interoperability.
created (optional)	timestamp	Specifies the date/time the file was created.
modified (optional)	timestamp	Specifies the date/time the file was last written to/modified.
accessed (optional)	timestamp	Specifies the date/time the file was last accessed.
parent_directory_ref (optional)	object-ref	Specifies the parent directory of the file, as a reference to a Directory Object. The object referenced in this property **MUST** be of type directory.
is_encrypted (optional)	boolean	Specifies whether the file is encrypted.
encryption_algorithm (optional)	open-vocab	Specifies the name of the encryption algorithm used to encrypt the file. This is an open vocabulary and values **SHOULD** come from the encryption-algo-ov vocabulary. This property **MUST NOT** be used if **is_encrypted** is false or not included.
decryption_key (optional)	string	Specifies the decryption key used to decrypt the file. This property **MUST NOT** be used if **is_encrypted** is false or not included.
contains_refs (optional)	list of type object-ref	Specifies a list of references to other Observable Objects contained within the file, such as another file that is appended to the end of the file, or an IP address that is contained somewhere in the the file. This is intended for use cases other than those targeted by

		the Archive extension.
`content_ref` (optional)	`object-ref`	Specifies the content of the file, represented as an Artifact Object. The object referenced in this property **MUST** be of type `artifact`.

Examples

Basic file with file system properties without observed encoding

```
{
  "0": {
    "type": "file",
    "hashes": {
      "SHA-256": "fe90a7e910cb3a4739bed9180e807e93fa70c90f25a8915476f5e4bfbac681db"
    },
    "size": 25536,
    "name": "foo.dll"
  }
}
```

Basic file with file system properties with observed encoding

```
{
  "0": {
    "type": "file",
    "hashes": {
      "SHA-256": "841a8921140aba50671ebb0770fecc4ee308c4952cfeff8de154ab14eeef4649"
    },
    "name": "quêry.dll",
    "name_enc": "windows-1252"
  }
}
```

In this example, the file name would have originally appeared using the bytes 71 75 **ea** 72 79 2e 64 6c 6c. Representing it in UTF-8, as required for JSON, would use the bytes 71 75 **c3 aa** 72 79 2e 64 6c 6c.

Basic file with parent directory

```
{
  "0": {
    "type": "directory",
    "path": "C:\\Windows\\System32"
  },
  "1": {
    "type": "file",
    "hashes": {
      "SHA-256": "ceafbfd424be2ca4a5f0402cae090dda2fb0526cf521b60b60077c0f622b285a"
    },
    "parent_directory_ref": "0",
    "name": "qwerty.dll"
  }
}
```

2.7.2 Archive File Extension

Type Name: `archive-ext`

The Archive File extension specifies a default extension for capturing properties specific to archive files. The key for this extension when used in the **`extensions`** dictionary **MUST** be `archive-ext`.

2.7.2.1 Properties

Property Name	Type	Description
`contains_refs` (required)	`list` of type `object-ref`	Specifies the files contained in the archive, as a reference to one or more other File Objects. The objects referenced in this list **MUST** be of type `file`.
`version` (optional)	`string`	Specifies the version of the archive type used in the archive file.
`comment` (optional)	`string`	Specifies a comment included as part of the archive file.

Examples

Basic unencrypted ZIP Archive

```
{
  "0": {
    "type": "file",
    "hashes": {
      "SHA-256": "ceafbfd424be2ca4a5f0402cae090dda2fb0526cf521b60b60077c0f622b285a"
    }
  },
  "1": {
    "type": "file",
    "hashes": {
      "SHA-256": "19c549ec2628b989382f6b280cbd7bb836a0b461332c0fe53511ce7d584b89d3"
    }
  },
  "2": {
    "type": "file",
    "hashes": {
      "SHA-256": "0969de02ecf8a5f003e3f6d063d848c8a193aada092623f8ce408c15bcb5f038"
    }
  },
  "3": {
    "type": "file",
    "name": "foo.zip",
    "hashes": {
      "SHA-256": "35a01331e9ad96f751278b891b6ea09699806faedfa237d40513d92ad1b7100f"
    },
    "mime_type": "application/zip",
    "extensions": {
      "archive-ext": {
        "contains_refs": [
          "0",
          "1",
          "2"
        ],
        "version": "5.0"
      }
    }
  }
}
```

2.7.3 NTFS File Extension

Type Name: `ntfs-ext`

The NTFS file extension specifies a default extension for capturing properties specific to the storage of the file on the NTFS file system. The key for this extension when used in the **`extensions`** dictionary **MUST** be `ntfs-ext`. An object using the NTFS File Extension **MUST** contain at least one property from this extension.

2.7.3.1 Properties

Property Name	Type	Description
`sid` (optional)	`string`	Specifies the security ID (SID) value assigned to the file.
`alternate_data_streams` (optional)	`list` of type `alternate-data-stream`	Specifies a list of NTFS alternate data streams that exist for the file.

2.7.3.2 Alternate Data Stream Type

Type Name: `alternate-data-stream`

The Alternate Data Stream type represents an NTFS alternate data stream.

2.7.3.2.1 Properties

Property Name	Type	Description
name (required)	string	Specifies the name of the alternate data stream.
hashes (optional)	hashes	Specifies a dictionary of hashes for the data contained in the alternate data stream.
size (optional)	integer	Specifies the size of the alternate data stream, in bytes. The value of this property **MUST NOT** be negative.

Examples

NTFS File with a single alternate data stream

```
{
  "0": {
    "type": "file",
    "hashes": {
      "SHA-256": "35a01331e9ad96f751278b891b6ea09699806faedfa237d40513d92ad1b7100f"
    },
    "extensions": {
      "ntfs-ext": {
        "alternate_data_streams": [
          {
            "name": "second.stream",
            "size": 25536
          }
        ]
      }
    }
  }
}
```

2.7.4 PDF File Extension

Type Name: pdf-ext

The PDF file extension specifies a default extension for capturing properties specific to PDF files. The key for this extension when used in the **extensions** dictionary **MUST** be pdf-ext. An object using the PDF File Extension **MUST** contain at least one property from this extension.

2.7.4.1 Properties

Property Name	Type	Description
version (optional)	string	Specifies the decimal version number of the string from the PDF header that specifies the version of the PDF specification to which the PDF file conforms. E.g., `1.4`.
is_optimized (optional)	boolean	Specifies whether the PDF file has been optimized.
document_info_dict (optional)	dictionary	Specifies details of the PDF document information dictionary (DID), which includes properties like the document creation data and producer, as a dictionary. Each key in the dictionary **SHOULD** be a case-preserved version of the corresponding entry in the document information dictionary without the prepended forward slash, e.g., `Title`. The corresponding value for the key **MUST** be the value specified for the document information dictionary entry, as a string.
pdfid0 (optional)	string	Specifies the first file identifier found for the PDF file.
pdfid1 (optional)	string	Specifies the second file identifier found for the PDF file.

Examples

Basic PDF file

```
{
  "0": {
    "type": "file",
    "hashes": {
      "SHA-256": "35a01331e9ad96f751278b891b6ea09699806faedfa237d40513d92ad1b7100f"
    },
    "extensions": {
      "pdf-ext": {
        "version": "1.7",
        "document_info_dict": {
          "Title": "Sample document",
          "Author": "Adobe Systems Incorporated",
          "Creator": "Adobe FrameMaker 5.5.3 for Power Macintosh",
          "Producer": "Acrobat Distiller 3.01 for Power Macintosh",
          "CreationDate": "20070412090123-02"
        },
        "pdfid0": "DFCE52BD827ECF765649852119D",
        "pdfid1": "57A1E0F9ED2AE523E313C"
      }
    }
  }
}
```

2.7.5 Raster Image File Extension

Type Name: `raster-image-ext`

The Raster Image file extension specifies a default extension for capturing properties specific to raster image files. The key for this extension when used in the **`extensions`** dictionary **MUST** be `raster-image-ext`. An object using the Raster Image File Extension **MUST** contain at least one property from this extension.

2.7.5.1 Properties

Property Name	Type	Description
`image_height` (optional)	`integer`	Specifies the height of the image in the image file, in pixels.
`image_width` (optional)	`integer`	Specifies the width of the image in the image file, in pixels.
`bits_per_pixel` (optional)	`integer`	Specifies the sum of bits used for each color channel in the image file, and thus the total number of pixels used for expressing the color depth of the image.
`image_compression_algorithm` (optional)	`string`	Specifies the name of the compression algorithm used to compress the image in the image file, if applicable.
`exif_tags` (optional)	`dictionary`	Specifies the set of EXIF tags found in the image file, as a dictionary. Each key/value pair in the dictionary represents the name/value of a single EXIF tag. Accordingly, each dictionary key **MUST** be a case-preserved version of the EXIF tag name, e.g., `XResolution`. Each dictionary value **MUST** be either an `integer` (for int* EXIF datatypes) or a `string` (for all other EXIF datatypes).

Examples

Simple Image File with EXIF Data

```
{
  "0": {
    "type": "file",
    "name": "picture.jpg",
    "hashes": {
      "SHA-256": "35a01331e9ad96f751278b891b6ea09699806faedfa237d40513d92ad1b7100f"
    },
    "extensions": {
      "raster-image-ext": {
        "exif_tags": {
          "Make": "Nikon",
          "Model": "D7000",
          "XResolution": 4928,
          "YResolution": 3264
        }
      }
    }
  }
}
```

2.7.6 Windows™ PE Binary File Extension

Type Name: `windows-pebinary-ext`

The Windows™ PE Binary File extension specifies a default extension for capturing properties specific to Windows portable executable (PE) files. The key for this extension when used in the **`extensions`** dictionary **MUST** be `windows-pebinary-ext`.

2.7.6.1 Properties

Property Name	Type	Description
`pe_type` (required)	`open-vocab`	Specifies the type of the PE binary. This is an open vocabulary and values **SHOULD** come from the `windows-pebinary-type-ov` vocabulary.
`imphash` (optional)	`string`	Specifies the special import hash, or 'imphash', calculated for the PE Binary based on its imported libraries and functions. For more information on the imphash algorithm, see the original article by Mandiant/FireEye: https://www.fireeye.com/blog/threat-research/2014/01/tracking-malware-import-hashing.html.
`machine_hex` (optional)	`hex`	Specifies the type of target machine.
`number_of_sections` (optional)	`integer`	Specifies the number of sections in the PE binary, as a non-negative integer.
`time_date_stamp` (optional)	`timestamp`	Specifies the time when the PE binary was created. The timestamp value

		MUST be precise to the second.
pointer_to_symbol_table_hex (optional)	hex	Specifies the file offset of the COFF symbol table.
number_of_symbols (optional)	integer	Specifies the number of entries in the symbol table of the PE binary, as a non-negative integer.
size_of_optional_header (optional)	integer	Specifies the size of the optional header of the PE binary. The value of this property **MUST NOT** be negative.
characteristics_hex (optional)	hex	Specifies the flags that indicate the file's characteristics.
file_header_hashes (optional)	hashes	Specifies any hashes that were computed for the file header.
optional_header (optional)	windows-pe-optional-header-type	Specifies the PE optional header of the PE binary.
sections (optional)	list of type windows-pe-section	Specifies metadata about the sections in the PE file.

2.7.6.2 Windows™ PE Binary Vocabulary

Vocabulary Name: windows-pebinary-type-ov

An open vocabulary of Windows PE binary types.

Value	Description
exe	Specifies that the PE binary is an executable image (i.e., not an OBJ or DLL).
dll	Specifies that the PE binary is a dynamically linked library (DLL).
sys	Specifies that the PE binary is a device driver (SYS).

2.7.6.3 PE Optional Header Type

Type Name: windows-pe-optional-header-type

The Windows PE Optional Header type represents the properties of the PE optional header.

2.7.6.3.1 Properties

Property Name	Type	Description
magic_hex (optional)	hex	Specifies the hex value that indicates the type of the PE binary.
major_linker_version (optional)	integer	Specifies the linker major version number.
minor_linker_version (optional)	integer	Specifies the linker minor version number.
size_of_code (optional)	integer	Specifies the size of the code (text) section. If there are multiple such sections, this refers to the sum of the sizes of each section. The value of this property **MUST NOT** be negative.
size_of_initialized_data (optional)	integer	Specifies the size of the initialized data section. If there are multiple such sections, this refers to the sum of the sizes of each section. The value of this property **MUST NOT** be negative.
size_of_uninitialized_data (optional)	integer	Specifies the size of the uninitialized data section. If there are multiple such sections, this refers to the sum of the sizes of each section. The value of this property **MUST NOT** be negative.
address_of_entry_point (optional)	integer	Specifies the address of the entry point relative to the image base when the executable is loaded into memory.
base_of_code (optional)	integer	Specifies the address that is relative to the image base of the beginning-of-code section when it is loaded into memory.
base_of_data (optional)	integer	Specifies the address that is relative to the image base of the beginning-of-data section when it is loaded into memory.
image_base (optional)	integer	Specifies the preferred address of the first byte of the image when loaded into memory.
section_alignment (optional)	integer	Specifies the alignment (in bytes) of PE sections when they are loaded into memory.

file_alignment (optional)	integer	Specifies the factor (in bytes) that is used to align the raw data of sections in the image file.
major_os_version (optional)	integer	Specifies the major version number of the required operating system.
minor_os_version (optional)	integer	Specifies the minor version number of the required operating system.
major_image_version (optional)	integer	Specifies the major version number of the image.
minor_image_version (optional)	integer	Specifies the minor version number of the image.
major_subsystem_version (optional)	integer	Specifies the major version number of the subsystem.
minor_subsystem_version (optional)	integer	Specifies the minor version number of the subsystem.
win32_version_value_hex (optional)	hex	Specifies the reserved win32 version value.
size_of_image (optional)	integer	Specifies the size of the image in bytes, including all headers, as the image is loaded in memory. The value of this property **MUST NOT** be negative.
size_of_headers (optional)	integer	Specifies the combined size of the MS-DOS, PE header, and section headers, rounded up to a multiple of the value specified in the file_alignment header. The value of this property **MUST NOT** be negative.
checksum_hex (optional)	hex	Specifies the checksum of the PE binary.
subsystem_hex (optional)	hex	Specifies the subsystem (e.g., GUI, device driver, etc.) that is required to run this image.
dll_characteristics_hex (optional)	hex	Specifies the flags that characterize the PE binary.
size_of_stack_reserve (optional)	integer	Specifies the size of the stack to reserve, in bytes. The value of this property **MUST NOT** be negative.
size_of_stack_commit	integer	Specifies the size of the stack to commit, in bytes.

(optional)		The value of this property **MUST NOT** be negative.
`size_of_heap_reserve` (optional)	`integer`	Specifies the size of the local heap space to reserve, in bytes.. The value of this property **MUST NOT** be negative.
`size_of_heap_commit` (optional)	`integer`	Specifies the size of the local heap space to commit, in bytes. The value of this property **MUST NOT** be negative.
`loader_flags_hex` (optional)	`hex`	Specifies the reserved loader flags.
`number_of_rva_and_sizes` (optional)	`integer`	Specifies the number of data-directory entries in the remainder of the optional header.
`hashes` (optional)	`hashes`	Specifies any hashes that were computed for the optional header.

2.7.6.4 Windows™ PE Section Type

Type Name: `windows-pe-section`

The Windows PE Section type specifies metadata about a PE file section.

2.7.6.4.1 Properties

Property Name	Type	Description
`name` (required)	`string`	Specifies the name of the section.
`size` (optional)	`integer`	Specifies the size of the section, in bytes. The value of this property **MUST NOT** be negative.
`entropy` (optional)	`float`	Specifies the calculated entropy for the section, as calculated using the Shannon algorithm (https://en.wiktionary.org/wiki/Shannon_entropy). The size of each input character is defined as a byte, resulting in a possible range of 0 through 8.
`hashes` (optional)	`hashes`	Specifies any hashes computed over the section.

Examples

Typical EXE File

```
{
  "0": {
    "type": "file",
    "hashes": {
      "SHA-256": "35a01331e9ad96f751278b891b6ea09699806faedfa237d40513d92ad1b7100f"
    },
    "extensions": {
      "windows-pebinary-ext": {
        "pe_type": "exe",
        "machine_hex": "014c",
        "number_of_sections": 4,
        "time_date_stamp": "2016-01-22T12:31:12Z",
        "pointer_to_symbol_table_hex": "74726144",
        "number_of_symbols": 4542568,
        "size_of_optional_header": 224,
        "characteristics_hex": "818f"
        "optional_header": {
          "magic_hex": "010b",
          "major_linker_version": 2,
          "minor_linker_version": 25,
          "size_of_code": 512,
          "size_of_initialized_data": 283648,
          "size_of_uninitialized_data": 0,
          "address_of_entry_point": 4096,
          "base_of_code": 4096,
          "base_of_data": 8192,
          "image_base": 14548992,
          "section_alignment": 4096,
          "file_alignment": 4096,
          "major_os_system_version": 1,
          "minor_os_system_version": 0,
          "major_image_version": 0,
          "minor_image_version": 0,
          "major_subsystem_version": 4,
          "minor_subsystem_version": 0,
          "win32_version_value_hex": "00",
          "size_of_image": 299008,
          "size_of_headers": 4096,
          "checksum_hex": "00",
          "subsystem_hex": "03",
          "dll_characteristics_hex": "00",
          "size_of_stack_reserve": 100000,
          "size_of_stack_commit": 8192,
          "size_of_heap_reserve": 100000,
          "size_of_heap_commit": 4096,
          "loader_flags_hex": "abdbffde",
          "number_of_rva_and_sizes": 3758087646
    },
    "sections": [
     {
      "name": "CODE",
      "entropy": 0.061089
     },
     {
      "name": "DATA",
      "entropy": 7.980693
     },
     {
      "name": "NicolasB",
      "entropy": 0.607433
     },
     {
      "name": ".idata",
      "entropy": 0.607433
     }
    ]
   }
  }
 }
}
```

2.8 IPv4 Address Object

Type Name: `ipv4-addr`

The IPv4 Address Object represents one or more IPv4 addresses expressed using CIDR notation.

2.8.1 Properties

Common Properties		
`type, extensions`		
IPv4 Address Object Specific Properties		
`value, resolves_to_refs, belongs_to_refs`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `ipv4-addr`.
`value` (required)	`string`	Specifies one or more IPv4 addresses expressed using CIDR

		notation. If a given IPv4 Address Object represents a single IPv4 address, the CIDR /32 suffix **MAY** be omitted. Example: `10.2.4.5/24`
`resolves_to_refs` (optional)	`list` of type `object-ref`	Specifies a list of references to one or more Layer 2 Media Access Control (MAC) addresses that the IPv4 address resolves to. The objects referenced in this list **MUST** be of type `mac-addr`.
`belongs_to_refs` (optional)	`list` of type `object-ref`	Specifies a list of reference to one or more autonomous systems (AS) that the IPv4 address belongs to. The objects referenced in this list **MUST** be of type `autonomous-system`.

Examples

IPv4 Single Address

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "198.51.100.3"
  }
}
```

IPv4 CIDR Block

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "198.51.100.0/24"
  }
}
```

2.9 IPv6 Address Object

Type Name: `ipv6-addr`

The IPv6 Address Object represents one or more IPv6 addresses expressed using CIDR notation.

2.9.1 Properties

Common Properties		
type, extensions		
IPv6 Address Object Specific Properties		
value, resolves_to_refs, belongs_to_refs		
Property Name	**Type**	**Description**
type (required)	string	The value of this property **MUST** be ipv6-addr.
value (required)	string	Specifies one or more IPv6 addresses expressed using CIDR notation. If a given IPv6 Address Object represents a single IPv6 address, the CIDR /128 suffix **MAY** be omitted.
resolves_to_refs (optional)	list of type object-ref	Specifies a list of references to one or more Layer 2 Media Access Control (MAC) addresses that the IPv6 address resolves to. The objects referenced in this list **MUST** be of type mac-addr.
belongs_to_refs (optional)	list of type object-ref	Specifies a list of reference to one or more autonomous systems (AS) that the IPv6 address belongs to. The objects referenced in this list **MUST** be of type autonomous-system.

Examples

IPv6 Single Address

```
{
  "0": {
    "type": "ipv6-addr",
    "value": "2001:0db8:85a3:0000:0000:8a2e:0370:7334"
  }
}
```

IPv6 CIDR block

```
{
  "0": {
    "type": "ipv6-addr",
    "value": "2001:0db8::/96"
  }
}
```

2.10 MAC Address Object

Type Name: `mac-addr`

The MAC Address Object represents a single Media Access Control (MAC) address.

2.10.1 Properties

Common Properties		
`type, extensions`		
MAC Address Object Specific Properties		
`value`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `mac-addr`.
`value` (required)	`string`	Specifies a single MAC address. The MAC address value **MUST** be represented as a single colon-delimited, lowercase MAC-48 address, which **MUST** include leading zeros for each octet. Example: `00:00:ab:cd:ef:01`

Examples

Typical MAC address

```
{
  "0": {
    "type": "mac-addr",
    "value": "d2:fb:49:24:37:18"
  }
}
```

2.11 Mutex Object

Type Name: `mutex`

The Mutex Object represents the properties of a mutual exclusion (mutex) object.

2.11.1 Properties

Common Properties		
`type, extensions`		
File Object Specific Properties		
`name`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `mutex`.
`name` (required)	`string`	Specifies the name of the mutex object.

Examples

Malware mutex

```
{
  "0": {
    "type": "mutex",
    "name": "__CLEANSWEEP__"
  }
}
```

2.12 Network Traffic

Type Name: `network-traffic`

The Network Traffic Object represents arbitrary network traffic that originates from a source and is addressed to a destination. The network traffic **MAY** or **MAY NOT** constitute a valid unicast, multicast, or broadcast network connection. This **MAY** also include traffic that is not established, such as a SYN flood.

To allow for use cases where a source or destination address may be sensitive and not suitable for sharing, such as addresses that are internal to an organization's network, the source and destination properties (**`src_ref`** and **`dst_ref`**, respectively) are defined as optional in the properties table below. However, a Network Traffic Object **MUST** contain the **`protocols`** property and at least one of the **`src_ref`** or **`dst_ref`** properties and **SHOULD** contain the **`src_port`** and **`dst_port`** properties.

2.12.1 Properties

Common Properties

type, extensions		
Network Traffic Specific Properties		
start, end, is_active, src_ref, dst_ref, src_port, dst_port, protocols, src_byte_count, dst_byte_count, src_packets, dst_packets, ipfix, src_payload_ref, dst_payload_ref, encapsulates_refs, encapsulated_by_ref		
Property Name	**Type**	**Description**
type (required)	string	The value of this property **MUST** be network-traffic.
extensions (optional)	dictionary	The Network Traffic Object defines the following extensions. In addition to these, producers **MAY** create their own. http-ext, tcp-ext, icmp-ext, socket-ext Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.
start (optional)	timestamp	Specifies the date/time the network traffic was initiated, if known.
end (optional)	timestamp	Specifies the date/time the network traffic ended, if known. If the **is_active** property is true, then the **end** property **MUST NOT** be included.
is_active (optional)	boolean	Indicates whether the network traffic is still ongoing.
src_ref (optional)	object-ref	Specifies the source of the network traffic, as a reference to one or more Observable Objects. The objects referenced in this list **MUST** be of type ipv4-addr or ipv6-addr or mac-addr or domain-name (for cases where the IP address for a domain name is unknown).
dst_ref (optional)	object-ref	Specifies the destination of the network traffic, as a reference to one or more Observable Objects. The objects referenced in this list **MUST** be of type ipv4-

		`addr` or `ipv6-addr` or `mac-addr` or `domain-name` (for cases where the IP address for a domain name is unknown).
`src_port` (optional)	`integer`	Specifies the source port used in the network traffic, as an integer. The port value **MUST** be in the range of 0 - 65535.
`dst_port` (optional)	`integer`	Specifies the destination port used in the network traffic, as an integer. The port value **MUST** be in the range of 0 - 65535.

`protocols` (required)	`list` of type `string`	Specifies the protocols observed in the network traffic, along with their corresponding state.
		
		Protocols **MUST** be listed in low to high order, from outer to inner in terms of packet encapsulation. That is, the protocols in the outer level of the packet, such as IP, **MUST** be listed first.
		
		The protocol names **SHOULD** come from the service names defined in the Service Name column of the IANA Service Name and Port Number Registry [Port Numbers]. In cases where there is variance in the name of a network protocol not included in the IANA Registry, content producers should exercise their best judgement, and it is recommended that lowercase names be used for consistency with the IANA registry.
		
		Examples:
		`ipv4, tcp, http`
		`ipv4, udp`
		`ipv6, tcp, http`
		`ipv6, tcp, ssl, https`

`src_byte_count` (optional)	`integer`	Specifies the number of bytes sent from the source to the destination.
`dst_byte_count` (optional)	`integer`	Specifies the number of bytes sent from the destination to the source.
`src_packets` (optional)	`integer`	Specifies the number of packets sent from the source to the destination.
`dst_packets` (optional)	`integer`	Specifies the number of packets sent destination to the source.
`ipfix` (optional)	`dictionary`	Specifies any IP Flow Information Export [IPFIX] data for the traffic, as a dictionary. Each key/value pair in the dictionary represents the name/value of a single IPFIX

		element. Accordingly, each dictionary key **SHOULD** be a case-preserved version of the IPFIX element name, e.g., `octetDeltaCount`. Each dictionary value **MUST** be either an `integer` or a `string`, as well as a valid IPFIX property.
`src_payload_ref` (optional)	`object-ref`	Specifies the bytes sent from the source to the destination. The object referenced in this property **MUST** be of type `artifact`.
`dst_payload_ref` (optional)	`object-ref`	Specifies the bytes sent from the destination to the source. The object referenced in this property **MUST** be of type `artifact`.
`encapsulates_refs` (optional)	`list` of type `object-ref`	Links to other `network-traffic` objects encapsulated by this `network-traffic` object. The objects referenced in this property **MUST** be of type `network-traffic`.
`encapsulated_by_ref` (optional)	`object-ref`	Links to another `network-traffic` object which encapsulates this object. The object referenced in this property **MUST** be of type `network-traffic`.

Examples

Basic TCP Network Traffic

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "198.51.100.2"
  },
  "1":{
    "type": "ipv4-addr",
    "value": "198.51.100.3"
  },
  "2": {
    "type": "network-traffic",
    "src_ref": "0",
    "dst_ref": "1",
    "protocols": [
      "tcp"
    ]
  }
}
```

Basic HTTP Network Traffic

```
{
  "0": {
    "type": "domain-name",
    "value": "example.com"
  },
  "1": {
    "type": "network-traffic",
    "dst_ref": "0",
    "protocols": [
      "ipv4",
      "tcp",
      "http"
    ]
  }
}
```

Network Traffic with Netflow Data

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "203.0.113.1"
  },
  "1": {
    "type": "ipv4-addr",
    "value": "203.0.113.5"
  },
  "2": {
    "type": "network-traffic",
    "src_ref": "0",
    "dst_ref": "1",
    "protocols": [
      "ipv4",
      "tcp"
    ],
    "src_byte_count": 147600,
    "src_packets": 100,
    "ipfix": {
      "minimumIpTotalLength": 32,
      "maximumIpTotalLength": 2556
    }
  }
}
```

Basic Tunneled Network Traffic

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "198.51.100.2"
  },
  "1": {
    "type": "ipv4-addr",
    "value": "203.0.113.1"
  },
  "2": {
    "type": "ipv4-addr",
    "value": "203.0.113.2"
  },
  "3": {
    "type": "network-traffic",
    "src_ref": "0",
    "dst_ref": "1",
    "src_port": 2487,
    "dst_port": 1723,
    "protocols": [
      "ipv4",
      "pptp"
    ],
    "src_byte_count": 35779,
    "dst_byte_count": 935750,
    "encapsulates_refs": [
      "4"
    ]
  },
  "4": {
    "type": "network-traffic",
    "src_ref": "0",
    "dst_ref": "2",
    "src_port": 24678,
    "dst_port": 80,
    "protocols": [
      "ipv4",
      "tcp",
      "http"
    ],
    "src_packets": 14356,
    "dst_packets": 14356,
    "encapsulated_by_ref": "3"
  }
}
```

Web traffic tunneled over DNS

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "203.0.113.1"
  },
  "1": {
    "type": "ipv4-addr",
    "value": "198.51.100.34"
  },
  "2": {
    "type": "ipv4-addr",
    "value": "198.51.100.54"
  },
  "3": {
    "type": "network-traffic",
    "src_ref": "0",
    "dst_ref": "1",
    "src_port": 2487,
    "dst_port": 53,
    "protocols": [
      "ipv4",
      "udp",
      "dns"
    ],
    "src_byte_count": 35779,
    "dst_byte_count": 935750,
    "encapsulates_refs": [
      "4"
    ]
  },
  "4": {
    "type": "network-traffic",
    "src_ref": "0",
    "dst_ref": "2",
    "src_port": 24678,
    "dst_port": 443,
    "protocols": [
      "ipv4",
      "tcp",
      "ssl",
      "http"
    ],
    "src_packets": 14356,
    "dst_packets": 14356,
    "encapsulated_by_ref": "3"
  }
}
```

2.12.2 HTTP Request Extension

Type Name: `http-request-ext`

The HTTP request extension specifies a default extension for capturing network traffic properties specific to HTTP requests. The key for this extension when used in the **`extensions`** dictionary **MUST** be `http-request-ext`.

2.12.2.1 2.12.2.1 Properties

Property Name	Type	Description
`request_method` (required)	`string`	Specifies the HTTP method portion of the HTTP request line, as a lowercase string.
`request_value` (required)	`string`	Specifies the value (typically a resource path) portion of the HTTP request line.
`request_version` (optional)	`string`	Specifies the HTTP version portion of the HTTP request line, as a lowercase string.
`request_header` (optional)	`dictionary`	Specifies all of the HTTP header fields that may be found in the HTTP client request, as a dictionary. Each key in the dictionary **MUST** be the name of the header field and **SHOULD** preserve case, e.g., `User-Agent`. The corresponding value for each dictionary key **MUST** be a `string`.
`message_body_length` (optional)	`integer`	Specifies the length of the HTTP message body, if included, in bytes.
`message_body_data_ref` (optional)	`object-ref`	Specifies the data contained in the HTTP message body, if included. The object referenced in this property **MUST** be of type `artifact`.

Examples

Basic HTTP Request

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "198.51.100.53"
  },
  "1": {
    "type": "network-traffic",
    "dst_ref": "0",
    "protocols": [
      "tcp",
      "http"
    ],
    "extensions": {
      "http-request-ext": {
        "request_method": "get",
        "request_value": "/download.html",
        "request_version": "http/1.1",
        "request_header": {
          "Accept-Encoding": "gzip,deflate",
          "User-Agent": "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.6)
Gecko/20040113",
          "Host": "www.example.com"
        }
      }
    }
  }
}
```

2.12.3 ICMP Extension

Type Name: `icmp-ext`

The ICMP extension specifies a default extension for capturing network traffic properties specific to ICMP. The key for this extension when used in the **`extensions`** dictionary **MUST** be `icmp-ext`.

2.12.3.1 Properties

Property Name	Type	Description
`icmp_type_hex` (required)	`hex`	Specifies the ICMP type byte.
`icmp_code_hex` (required)	`hex`	Specifies the ICMP code byte.

Examples

Basic ICMP Traffic

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "198.51.100.9"
  },
  "1": {
    "type": "ipv4-addr",
    "value": "203.0.113.5"
  },
  "2": {
    "type": "network-traffic",
    "src_ref": "0",
    "dst_ref": "1",
    "protocols": [
      "icmp"
    ],
    "extensions": {
      "icmp-ext": {
        "icmp_type_hex": "08",
        "icmp_code_hex": "00"
      }
    }
  }
}
```

2.12.4 Network Socket Extension

Type Name: `socket-ext`

The Network Socket extension specifies a default extension for capturing network traffic properties associated with network sockets. The key for this extension when used in the extensions dictionary MUST be `socket-ext`.

2.12.4.1 Properties

Property Name	Type	Description
address_family (required)	`network-socket-address-family-enum`	Specifies the address family (AF_*) that the socket is configured for.
is_blocking (optional)	`boolean`	Specifies whether the socket is in blocking mode.
is_listening (optional)	`boolean`	Specifies whether the socket is in listening mode.

protocol_family (optional)	network-socket-protocol-family-enum	Specifies the protocol family (PF_*) that the socket is configured for.
options (optional)	dictionary	Specifies any options (SO_*) that may be used by the socket, as a dictionary. Each key in the dictionary SHOULD be a case-preserved version of the option name, e.g., SO_ACCEPTCONN. Each key value in the dictionary MUST be the value for the corresponding options key.
socket_type (optional)	network-socket-type-enum	Specifies the type of the socket.
socket_descriptor (optional)	integer	Specifies the socket file descriptor value associated with the socket, as a non-negative integer.
socket_handle (optional)	integer	Specifies the handle or inode value associated with the socket.

2.12.4.2 Network Socket Address Family Enumeration

Enumeration Name: network-socket-address-family-enum

An enumeration of network socket address family types.

Vocabulary Value	Description
AF_UNSPEC	Specifies an unspecified address family.
AF_INET	Specifies the IPv4 address family.
AF_IPX	Specifies the IPX (Novell Internet Protocol) address family.
AF_APPLETALK	Specifies the APPLETALK DDP address family.
AF_NETBIOS	Specifies the NETBIOS address family.
AF_INET6	Specifies the IPv6 address family.
AF_IRDA	Specifies IRDA sockets.
AF_BTH	Specifies BTH sockets.

2.12.4.3 Network Socket Protocol Family Enumeration

Enumeration Name: `network-socket-protocol-family-enum`

An enumeration of network socket protocol family types.

Vocabulary Value	Description
`PF_INET`	Specifies the IP protocol family.
`PF_AX25`	Specifies the amateur radio AX.25 family.
`PF_IPX`	Specifies the Novell Internet Protocol family.
`PF_INET6`	Specifies the IP version 6 family.
`PF_APPLETALK`	Specifies the Appletalk DDP protocol family.
`PF_NETROM`	Specifies the Amateur radio NetROM protocol family.
`PF_BRIDGE`	Specifies the Multiprotocol bridge protocol family.
`PF_ATMPVC`	Specifies the ATM PVCs protocol family.
`PF_X25`	Specifies the protocol family reserved for the X.25 project.
`PF_ROSE`	Specifies the PF_KEY key management API family.
`PF_DECNET`	Specifies the protocol family reserved for the DECnet project.
`PF_NETBEUI`	Specifies the protocol family reserved for the 802.2LLC project.
`PF_SECURITY`	Specifies the Security callback pseudo AF protocol family.
`PF_KEY`	Specifies the PF_KEY key management API protocol family.
`PF_NETLINK`	Specifies the netlink routing API family.
`PF_ROUTE`	Specifies the PF_ROUTE routing API family.
`PF_PACKET`	Specifies the packet family.

PF_ASH	Specifies the Ash family.
PF_ECONET	Specifies the Acorn Econet family.
PF_ATMSVC	Specifies the ATM SVCs protocol family.
PF_SNA	Specifies the Linux SNA Project protocol family.
PF_IRDA	Specifies IRDA sockets.
PF_PPPOX	Specifies PPPoX sockets.
PF_WANPIPE	Specifies Wanpipe API sockets.
PF_BLUETOOTH	Specifies Bluetooth sockets.

2.12.4.4 Network Socket Type Enumeration

Enumerations Name: `network-socket-type-enum`

An enumeration of network socket types.

Vocabulary Value	Description
SOCK_STREAM	Specifies a pipe-like socket which operates over a connection with a particular remote socket, and transmits data reliably as a stream of bytes.
SOCK_DGRAM	Specifies a socket in which individually-addressed packets are sent (datagram).
SOCK_RAW	Specifies raw sockets which allow new IP protocols to be implemented in user space. A raw socket receives or sends the raw datagram not including link level headers.
SOCK_RDM	Specifies a socket indicating a reliably-delivered message.
SOCK_SEQPACKET	Specifies a datagram congestion control protocol socket.

Examples

Basic Stream Socket

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "198.51.100.2"
  },
  "1": {
    "type": "network-traffic",
    "src_ref": "0",
    "src_port": 223,
    "protocols": [
      "ip",
      "tcp"
    ],
    "extensions": {
      "socket-ext": {
        "is_listening": true,
        "address_family": "AF_INET",
        "protocol_family": "PF_INET",
        "socket_type": "SOCK_STREAM"
      }
    }
  }
}
```

2.12.5 TCP Extension

Type Name: `tcp-ext`

The TCP extension specifies a default extension for capturing network traffic properties specific to TCP. The key for this extension when used in the **`extensions`** dictionary **MUST** be `tcp-ext`. An object using the TCP Extension **MUST** contain at least one property from this extension.

2.12.5.1 Properties

Property Name	Type	Description
`src_flags_hex` (optional)	`hex`	Specifies the source TCP flags, as the union of all TCP flags observed between the start of the traffic (as defined by the **start** property) and the end of the traffic (as defined by the **end** property). If the start and end times of the traffic are not specified, this property **SHOULD** be interpreted as the union of all TCP flags observed over the entirety of the network traffic being reported upon.
`dst_flags_hex` (optional)	`hex`	Specifies the destination TCP flags, as the union of all TCP flags observed between the start of the traffic (as defined by the **start** property) and the end of the traffic (as defined by the **end** property). If the start and end times of the traffic are not specified, this property

		SHOULD be interpreted as the union of all TCP flags observed over the entirety of the network traffic being reported upon.

Examples

Basic TCP Traffic

```
{
  "0": {
    "type": "ipv4-addr",
    "value": "198.51.100.5"
  },
  "1": {
    "type": "ipv4-addr",
    "value": "198.51.100.6"
  },
  "2": {
    "type": "network-traffic",
    "src_ref": "0",
    "dst_ref": "1",
    "src_port": 3372,
    "dst_port": 80,
    "protocols": [
      "tcp"
    ],
    "extensions": {
      "tcp-ext": {
        "src_flags_hex": "00000002"
      }
    }
  }
}
```

2.13 Process Object

Type Name: `process`

The Process Object represents common properties of an instance of a computer program as executed on an operating system. A Process Object MUST contain at least one property (other than **`type`**) from this object (or one of its extensions).

2.13.1 Properties

Common Properties
`type, extensions`

Process Object Specific Properties		
`is_hidden, pid, name, created, cwd, arguments, command_line, environment_variables, opened_connection_refs, creator_user_ref, binary_ref, parent_ref, child_refs`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `process`.
`extensions` (optional)	`dictionary`	The Process Object defines the following extensions. In addition to these, producers **MAY** create their own. `windows-process-ext`, `windows-service-ext` Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.
`is_hidden` (optional)	`boolean`	Specifies whether the process is hidden.
`pid` (optional)	`integer`	Specifies the Process ID, or PID, of the process.
`name` (optional)	`string`	Specifies the name of the process.
`created` (optional)	`timestamp`	Specifies the date/time at which the process was created.
`cwd` (optional)	`string`	Specifies the current working directory of the process.
`arguments` (optional)	`list` of type `string`	Specifies the list of arguments used in executing the process. Each argument **MUST** be captured separately as a string.
`command_line` (optional)	`string`	Specifies the full command line used in executing the process, including the process name (depending on the operating system).
`environment_variables` (optional)	`dictionary`	Specifies the list of environment variables associated with the process as a dictionary. Each key in the dictionary **MUST** be a case preserved version of the name of the environment variable, and each

		corresponding value **MUST** be the environment variable value as a string.
`opened_connection_refs` (optional)	`list` of type `object-ref`	Specifies the list of network connections opened by the process, as a reference to one or more Network Traffic Objects. The objects referenced in this list **MUST** be of type `network-traffic`.
`creator_user_ref` (optional)	`object-ref`	Specifies the user that created the process, as a reference to a User Account Object. The object referenced in this property **MUST** be of type `user-account`.
`binary_ref` (optional)	`object-ref`	Specifies the executable binary that was executed as the process, as a reference to a File Object. The object referenced in this property **MUST** be of type `file`.
`parent_ref` (optional)	`object-ref`	Specifies the other process that spawned (i.e. is the parent of) this one, as reference to a Process Object. The object referenced in this property **MUST** be of type `process`.
`child_refs` (optional)	`list` of type `object-ref`	Specifies the other processes that were spawned by (i.e. children of) this process, as a reference to one or more other Process Objects. The objects referenced in this list **MUST** be of type `process`.

Examples

Basic Process

```
{
  "0": {
    "type": "file",
    "hashes": {
      "SHA-256": "35a01331e9ad96f751278b891b6ea09699806faedfa237d40513d92ad1b7100fSHA"
    },
  },
  "1": {
    "type": "process",
    "pid": 1221,
    "name": "gedit-bin",
    "created": "2016-01-20T14:11:25.55Z",
    "arguments" :[
      "--new-window"
    ],
    "binary_ref": "0"
  }
}
```

2.13.2 Windows™ Process Extension

Type Name: `windows-process-ext`

The Windows Process extension specifies a default extension for capturing properties specific to Windows processes. The key for this extension when used in the **`extensions`** dictionary **MUST** be `windows-process-ext`. An object using the Windows Process Extension **MUST** contain at least one property from this extension.

2.13.2.1 Properties

Property Name	Type	Description
`aslr_enabled` (optional)	`boolean`	Specifies whether Address Space Layout Randomization (ASLR) is enabled for the process.
`dep_enabled` (optional)	`boolean`	Specifies whether Data Execution Prevention (DEP) is enabled for the process.
`priority` (optional)	`string`	Specifies the current priority class of the process in Windows. This value **SHOULD** be a string that ends in `_CLASS`.
`owner_sid` (optional)	`string`	Specifies the Security ID (SID) value of the owner of the process.
`window_title` (optional)	`string`	Specifies the title of the main window of the process.
`startup_info` (optional)	`dictionary`	Specifies the STARTUP_INFO struct used by the process, as a dictionary. Each name/value pair in the struct **MUST** be represented as a key/value pair in the dictionary, where each key **MUST** be a case-preserved version of the original name. For example, given a name of "lpDesktop" the corresponding key would be `lpDesktop`.

Examples

Basic Windows Process

```
{
  "0": {
    "type": "process",
    "pid": 314,
    "name": "foobar.exe",
    "extensions": {
      "windows-process-ext": {
        "aslr_enabled": true,
        "dep_enabled": true,
        "priority": "HIGH_PRIORITY_CLASS",
        "owner_sid": "S-1-5-21-186985262-1144665072-74031268-1309"
      }
    }
  }
}
```

2.13.3 Windows™ Service Extension

Type Name: `windows-service-ext`

The Windows Service extension specifies a default extension for capturing properties specific to Windows services. The key for this extension when used in the **`extensions`** dictionary **MUST** be `windows-service-ext`.

2.13.3.1 Properties

Property Name	Type	Description
`service_name` (required)	`string`	Specifies the name of the service.
`descriptions` (optional)	`list` of type `string`	Specifies the descriptions defined for the service.
`display_name` (optional)	`string`	Specifies the displayed name of the service in Windows GUI controls.
`group_name` (optional)	`string`	Specifies the name of the load ordering group of which the service is a member.
`start_type` (optional)	`windows-service-start-type-enum`	Specifies the start options defined for the service.
`service_dll_refs` (optional)	`list` of type `object-ref`	Specifies the DLLs loaded by the service, as a reference to one or more File Objects.

		The objects referenced in this property **MUST** be of type `file`.
`service_type` (optional)	`windows-service-type-enum`	Specifies the type of the service.
`service_status` (optional)	`windows-service-status-enum`	Specifies the current status of the service.

2.13.3.2 Windows™ Service Start Type Enumeration

Enumeration Name: `windows-service-start-type-enum`

An enumeration of Windows service start types.

Vocabulary Value	**Description**
`SERVICE_AUTO_START`	A service started automatically by the service control manager during system startup.
`SERVICE_BOOT_START`	A device driver started by the system loader. This value is valid only for driver services.
`SERVICE_DEMAND_START`	A service started by the service control manager when a process calls the StartService function.
`SERVICE_DISABLED`	A service that cannot be started. Attempts to start the service result in the error code ERROR_SERVICE_DISABLED.
`SERVICE_SYSTEM_ALERT`	A device driver started by the IoInitSystem function. This value is valid only for driver services.

2.13.3.3 Windows™ Service Type Enumeration

Enumeration Name: `windows-service-type-enum`

An enumeration of Windows service types.

Vocabulary Value	**Description**
`SERVICE_KERNEL_DRIVER`	The service is a device driver.

`SERVICE_FILE_SYSTEM_DRIVER`	The service is a file system driver.
`SERVICE_WIN32_OWN_PROCESS`	The service runs in its own process.
`SERVICE_WIN32_SHARE_PROCESS`	The service shares a process with other services.

2.13.3.4 Windows™ Service Status Enumeration

Enumeration Name: `windows-service-status-enum`

An enumeration of Windows service statuses.

Value	Description
`SERVICE_CONTINUE_PENDING`	The service continue is pending.
`SERVICE_PAUSE_PENDING`	The service pause is pending.
`SERVICE_PAUSED`	The service is paused.
`SERVICE_RUNNING`	The service is running.
`SERVICE_START_PENDING`	The service is starting.
`SERVICE_STOP_PENDING`	The service is stopping.
`SERVICE_STOPPED`	The service is not running.

Examples

Basic Windows Service

```
{
  "0": {
    "type": "file",
    "hashes": {
      "SHA-256": "35a01331e9ad96f751278b891b6ea09699806faedfa237d40513d92ad1b7100f"
    },
    "name": "sirvizio.exe"
  },
  "1":{
    "type": "process",
    "pid": 2217,
    "name": "sirvizio",
    "command_line": "C:\\Windows\\System32\\sirvizio.exe /s",
    "binary_ref": "0",
    "extensions": {
      "windows-service-ext": {
        "service_name": "sirvizio",
        "display_name": "Sirvizio",
        "start_type": "SERVICE_AUTO_START",
        "service_type": "SERVICE_WIN32_OWN_PROCESS",
        "service_status": "SERVICE_RUNNING"
      }
    }
  }
}
```

2.14 Software Object

Type Name: `software`

The Software Object represents high-level properties associated with software, including software products.

2.14.1 Properties

Common Properties		
`type, extensions`		
Software Object Specific Properties		
`name, cpe, languages, vendor, version`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `software`.
`name` (required)	`string`	Specifies the name of the software.
`cpe` (optional)	`string`	Specifies the Common Platform Enumeration (CPE) entry for the software, if available. The value for this property **MUST** be a CPE v2.3 entry from the official NVD CPE Dictionary [NVD]. While the CPE dictionary does not contain entries for *all* software, whenever it *does* contain an identifier for a given instance of software, this property **SHOULD** be present.

languages (optional)	list of type string	Specifies the languages supported by the software. The value of each list member **MUST** be an ISO 639-2 language code [ISO639-2].
vendor (optional)	string	Specifies the name of the vendor of the software.
version (optional)	string	Specifies the version of the software.

Examples

Typical Software Instance

```
{
  "0": {
    "type": "software",
    "name": "Word",
    "cpe": "cpe:2.3:a:microsoft:word:2000:*:*:*:*:*:*:*",
    "version": "2002",
    "vendor": "Microsoft"
  }
}
```

2.15 URL Object

Type Name: url

The URL Object represents the properties of a uniform resource locator (URL).

2.15.1 Properties

Common Properties		
type, extensions		
URL Object Specific Properties		
value		
Property Name	**Type**	**Description**
type (required)	string	The value of this property **MUST** be url.
value (required)	string	Specifies the value of the URL. The value of this property **MUST** conform to [RFC3986], more specifically section 1.1.3 with reference to the definition for "Uniform Resource Locator".

Examples

Typical URL

```
{
  "0": {
    "type": "url",
    "value": "https://example.com/research/index.html"
  }
}
```

2.16 User Account Object

Type Name: `user-account`

The User Account Object represents an instance of any type of user account, including but not limited to operating system, device, messaging service, and social media platform accounts.

2.16.1 Properties

Common Properties		
`type`, `extensions`		
User Account Object Specific Properties		
`user_id`, `account_login`, `account_type`, `display_name`, `is_service_account`, `is_privileged`, `can_escalate_privs`, `is_disabled`, `account_created`, `account_expires`, `password_last_changed`, `account_first_login`, `account_last_login`		
Property Name	**Type**	**Description**
`type` (required)	`string`	The value of this property **MUST** be `user-account`.
`extensions` (optional)	`dictionary`	The User Account Object defines the following extensions. In addition to these, producers **MAY** create their own. `unix-account-ext` Dictionary keys **MUST** identify the extension type by name. The corresponding dictionary values **MUST** contain the contents of the extension instance.

user_id (required)	string	Specifies the identifier of the account. The format of the identifier depends on the system the user account is maintained in, and may be a numeric ID, a GUID, an account name, an email address, etc. The **user_id** property should be populated with whatever field is the unique identifier for the system the account is a member of. For example, for UNIX systems it would be populated with the UID.
account_login (optional)	string	Specifies the account login string, used in cases where the **user_id** property specifies something other than what a user would type when they login. For example, in the case of a Unix account with user_id 0, the account_login might be “root”.
account_type (optional)	open-vocab	Specifies the type of the account. This is an open vocabulary and values **SHOULD** come from the account-type-ov vocabulary.
display_name (optional)	string	Specifies the display name of the account, to be shown in user interfaces, if applicable. On Unix, this is equivalent to the GECOS field.
is_service_account (optional)	boolean	Indicates that the account is associated with a network service or system process (daemon), not a specific individual.
is_privileged (optional)	boolean	Specifies that the account has elevated privileges (i.e., in the case of root on Unix or the Windows Administrator account).
can_escalate_privs (optional)	boolean	Specifies that the account has the ability to escalate privileges (i.e., in the case of sudo on Unix or a Windows Domain Admin account)
is_disabled (optional)	boolean	Specifies if the account is disabled.
account_created (optional)	timestamp	Specifies when the account was created.
account_expires (optional)	timestamp	Specifies the expiration date of the account.
password_last_changed (optional)	timestamp	Specifies when the account password was last changed.

`account_first_login` (optional)	`timestamp`	Specifies when the account was first accessed.
`account_last_login` (optional)	`timestamp`	Specifies when the account was last accessed.

2.16.2 Account Type Vocabulary

Vocabulary Name: `account-type-ov`

An open vocabulary of User Account types.

Vocabulary Value	Description
`unix`	Specifies a POSIX account.
`windows-local`	Specifies a Windows local account.
`windows-domain`	Specifies a Windows domain account.
`ldap`	Specifies an LDAP account.
`tacacs`	Specifies a TACACS account.
`radius`	Specifies a RADIUS account.
`nis`	Specifies a NIS account
`openid`	Specifies an OpenID account.
`facebook`	Specifies a Facebook account.
`skype`	Specifies a Skype account.
`twitter`	Specifies a Twitter account.

Examples

Basic Unix Account

```
{
  "0": {
    "type": "user-account",
    "user_id": "1001",
    "account_login": "jdoe",
    "account_type": "unix",
    "display_name": "John Doe",
    "is_service_account": false,
    "is_privileged": false,
    "can_escalate_privs": true,
    "account_created": "2016-01-20T12:31:12Z",
    "password_last_changed": "2016-01-20T14:27:43Z",
    "account_first_login": "2016-01-20T14:26:07Z",
    "account_last_login": "2016-07-22T16:08:28Z"
  }
}
```

Basic Twitter Account

```
{
  "0": {
    "type": "user-account",
    "user_id": "thegrugq_ebooks",
    "account_login": "thegrugq_ebooks",
    "account_type": "twitter",
    "display_name": "the grugq"
  }
}
```

2.16.3 UNIX™ Account Extension

Type Name: `unix-account-ext`

The UNIX account extension specifies a default extension for capturing the additional information for an account on a UNIX system. The key for this extension when used in the **`extensions`** dictionary **MUST** be `unix-account-ext`. An object using the UNIX Account Extension **MUST** contain at least one property from this extension.

2.16.3.1 Properties

Property Name	Type	Description
`gid` (optional)	`number`	Specifies the primary group ID of the account.
`groups` (optional)	`list` of type `string`	Specifies a list of names of groups that the account is a member of.
`home_dir` (optional)	`string`	Specifies the home directory of the account.

shell (optional)	string	Specifies the account's command shell.

Examples

Basic UNIX Account

```
{
  "0": {
    "type": "user-account",
    "user_id": "1001",
    "account_login": "jdoe",
    "account_type": "unix",
    "display_name": "John Doe",
    "is_service_account": false,
    "is_privileged": false,
    "can_escalate_privs": true,
    "extensions": {
      "unix-account-ext": {
        "gid": 1001,
        "groups": ["wheel"],
        "home_dir": "/home/jdoe",
        "shell": "/bin/bash"
      }
    }
  }
}
```

2.17 Windows™ Registry Key Object

Type Name: windows-registry-key

The Registry Key Object represents the properties of a Windows registry key.

2.17.1 Properties

Common Properties		
type, extensions		
File Object Specific Properties		
key, values, modified, creator_user_ref, number_of_subkeys		
Property Name	**Type**	**Description**

type (required)	`string`	The value of this property **MUST** be `windows-registry-key`.
key (required)	`string`	Specifies the full registry key including the hive. The value of the key, including the hive portion, **SHOULD** be case-preserved. The hive portion of the key **MUST** be fully expanded and not truncated; e.g., HKEY_LOCAL_MACHINE must be used instead of HKLM.
values (optional)	`list` of type `windows-registry-value-type`	Specifies the values found under the registry key.
modified (optional)	`timestamp`	Specifies the last date/time that the registry key was modified.
creator_user_ref (optional)	`object-ref`	Specifies a reference to the user account (represented as a User Account Object) that created the registry key. The object referenced in this property **MUST** be of type `user-account`.
number_of_subkeys (optional)	`integer`	Specifies the number of subkeys contained under the registry key.

2.17.2 Windows™ Registry Value Type

Type Name: `windows-registry-value-type`

The Windows Registry Value type captures the properties of a Windows Registry Key Value.

2.17.2.1 Properties

Property Name	Type	Description
`name` (required)	`string`	Specifies the name of the registry value. For specifying the default value in a registry key, an empty string **MUST** be used.
`data` (optional)	`string`	Specifies the data contained in the registry value.
`data_type` (optional)	`windows-registry-datatype-enum`	Specifies the registry (REG_*) data type used in the registry value.

2.17.3 Windows™ Registry Datatype Enumeration

Enumeration Name: `windows-registry-datatype-enum`

An enumeration of Windows registry data types.

Vocabulary Value	Description
`REG_NONE`	No defined value type.
`REG_SZ`	A null-terminated string. This will be either a Unicode or an ANSI string, depending on whether you use the Unicode or ANSI functions.
`REG_EXPAND_SZ`	A null-terminated string that contains unexpanded references to environment variables (for example, "%PATH%"). It will be a Unicode or ANSI string depending on whether you use the Unicode or ANSI functions.
`REG_BINARY`	Binary data in any form.
`REG_DWORD`	A 32-bit number.
`REG_DWORD_BIG_ENDIAN`	A 32-bit number in big-endian format.
`REG_LINK`	A null-terminated Unicode string that contains the target path of a symbolic link.
`REG_MULTI_SZ`	A sequence of null-terminated strings, terminated by an empty string (\0).
`REG_RESOURCE_LIST`	A series of nested lists designed to store a resource list used by a hardware device driver or one of the physical devices it controls. This data is detected and written into the ResourceMap tree by the system and is displayed in Registry Editor in hexadecimal format as a Binary Value.
`REG_FULL_RESOURCE_DESCRIPTION`	A series of nested lists designed to store a resource list used by a physical hardware device. This data is detected and written into the HardwareDescription tree by the system and is displayed in Registry Editor in hexadecimal format as a Binary Value.
`REG_RESOURCE_REQUIREMENTS_LIST`	Device driver list of hardware resource requirements in Resource Map tree.

`REG_QWORD`	A 64-bit number.
`REG_INVALID_TYPE`	Specifies an invalid key.

Examples

Simple registry key

```
{
  "0": {
    "type": "windows-registry-key",
    "key": "HKEY_LOCAL_MACHINE\\System\\Foo\\Bar"
    }
}
```

Registry key with values

```
{
  "0": {
    "type": "windows-registry-key",
    "key": "hkey_local_machine\\system\\bar\\foo",
    "values": [
      {
        "name": "Foo",
        "data": "qwerty",
        "data_type": "REG_SZ"
      },
      {
        "name": "Bar",
        "data": "42",
        "data_type": "REG_DWORD"
      }
    ]
  }
}
```

2.18 X.509 Certificate Object

Type Name: `x509-certificate`

The X.509 Certificate Object represents the properties of an X.509 certificate, as defined by ITU recommendation X.509 [X.509]. An X.509 Certificate Object **MUST** contain at least one property (other than **type)** from this object or one of its extensions.

2.18.1 Properties

Common Properties

type, extensions
File Object Specific Properties
is_self_signed, hashes, version, serial_number, signature_algorithm, issuer, validity_not_before, validity_not_after, subject, subject_public_key_algorithm, subject_public_key_modulus, subject_public_key_exponent, x509_v3_extensions

Property Name	Type	Description
type (required)	string	The value of this property **MUST** be x509-certificate.
is_self_signed (optional)	boolean	Specifies whether the certificate is self-signed, i.e., whether it is signed by the same entity whose identity it certifies.
hashes (optional)	hashes	Specifies any hashes that were calculated for the entire contents of the certificate.
version (optional)	string	Specifies the version of the encoded certificate.
serial_number (optional)	string	Specifies the unique identifier for the certificate, as issued by a specific Certificate Authority.
signature_algorithm (optional)	string	Specifies the name of the algorithm used to sign the certificate.
issuer (optional)	string	Specifies the name of the Certificate Authority that issued the certificate.
validity_not_before (optional)	timestamp	Specifies the date on which the certificate validity period begins.
validity_not_after (optional)	timestamp	Specifies the date on which the certificate validity period ends.
subject (optional)	string	Specifies the name of the entity associated with the public key stored in the subject public key field of the certificate.
subject_public_key_algorithm (optional)	string	Specifies the name of the algorithm with which to encrypt data being sent to the subject.

`subject_public_key_modulus` (optional)	`string`	Specifies the modulus portion of the subject’s public RSA key.
`subject_public_key_exponent` (optional)	`integer`	Specifies the exponent portion of the subject’s public RSA key, as an integer.
`x509_v3_extensions` (optional)	`x509-v3-extensions-type`	Specifies any standard X.509 v3 extensions that may be used in the certificate.

2.18.2 X.509 v3 Extensions Type

Type Name: `x509-v3-extensions-type`

The X.509 v3 Extensions type captures properties associated with X.509 v3 extensions, which serve as a mechanism for specifying additional information such as alternative subject names. An object using the X.509 v3 Extensions type **MUST** contain at least one property from this type.

Note that the X.509 v3 Extensions type is not a STIX Cyber Observables extension, it is a type that describes X.509 extensions.

2.18.2.1 Properties

Property Name	Type	Description
`basic_constraints` (optional)	`string`	Specifies a multi-valued extension which indicates whether a certificate is a CA certificate. The first (mandatory) name is CA followed by *TRUE* or *FALSE*. If CA is *TRUE* then an optional pathlen name followed by an non-negative value can be included. Also equivalent to the object ID (OID) value of 2.5.29.19.
`name_constraints` (optional)	`string`	Specifies a namespace within which all subject names in subsequent certificates in a certification path **MUST** be located. Also equivalent to the object ID (OID) value of 2.5.29.30.
`policy_constraints` (optional)	`string`	Specifies any constraints on path validation for certificates issued to CAs. Also equivalent to the object ID (OID) value of 2.5.29.36.
`key_usage` (optional)	`string`	Specifies a multi-valued extension consisting of a list of names of the permitted key usages. Also equivalent to the object ID (OID) value of 2.5.29.15.

`extended_key_usage` (optional)	`string`	Specifies a list of usages indicating purposes for which the certificate public key can be used for. Also equivalent to the object ID (OID) value of 2.5.29.37.
`subject_key_identifier` (optional)	`string`	Specifies the identifier that provides a means of identifying certificates that contain a particular public key. Also equivalent to the object ID (OID) value of 2.5.29.14.
`authority_key_identifier` (optional)	`string`	Specifies the identifier that provides a means of identifying the public key corresponding to the private key used to sign a certificate. Also equivalent to the object ID (OID) value of 2.5.29.35.
`subject_alternative_name` (optional)	`string`	Specifies the additional identities to be bound to the subject of the certificate. Also equivalent to the object ID (OID) value of 2.5.29.17.
`issuer_alternative_name` (optional)	`string`	Specifies the additional identities to be bound to the issuer of the certificate. Also equivalent to the object ID (OID) value of 2.5.29.18.
`subject_directory_attributes` (optional)	`string`	Specifies the identification attributes (e.g., nationality) of the subject. Also equivalent to the object ID (OID) value of 2.5.29.9.
`crl_distribution_points` (optional)	`string`	Specifies how CRL information is obtained. Also equivalent to the object ID (OID) value of 2.5.29.31.
`inhibit_any_policy` (optional)	`string`	Specifies the number of additional certificates that may appear in the path before anyPolicy is no longer permitted. Also equivalent to the object ID (OID) value of 2.5.29.54.
`private_key_usage_period_not_before` (optional)	`timestamp`	Specifies the date on which the validity period begins for the private key, if it is different from the validity period of the certificate.
`private_key_usage_period_not_after` (optional)	`timestamp`	Specifies the date on which the validity period ends for the private key, if it is different from the validity period of the certificate.
`certificate_policies` (optional)	`string`	Specifies a sequence of one or more policy information terms, each of which consists of an object identifier (OID) and optional qualifiers. Also equivalent to the object ID (OID) value of

		2.5.29.32.
policy_mappings (optional)	string	Specifies one or more pairs of OIDs; each pair includes an issuerDomainPolicy and a subjectDomainPolicy. The pairing indicates whether the issuing CA considers its issuerDomainPolicy equivalent to the subject CA's subjectDomainPolicy. Also equivalent to the object ID (OID) value of 2.5.29.33.

Examples

Basic X.509 certificate

```
{
  "0": {
    "type": "x509-certificate",
    "issuer": "C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc, OU=Certification
Services Division, CN=Thawte Server CA/emailAddress=server-certs@thawte.com",
    "validity_not_before": "2016-03-12T12:00:00Z",
    "validity_not_after": "2016-08-21T12:00:00Z",
    "subject": "C=US, ST=Maryland, L=Pasadena, O=Brent Baccala, OU=FreeSoft,
CN=www.freesoft.org/emailAddress=baccala@freesoft.org"
    }
}
```

3 Conformance

3.1 Defined Object Producers

A "Defined Object Producer" that creates an Object from section 2 (Defined Object Data Models) is a “Producer” of that Object. Defined Object Producers **MUST** conform to all normative requirements in the section for that Object along with all of the general requirements pertaining to Objects as defined in section 3 of *STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts*.

For example, a "Defined Object Producer" that can produce File Object is a "File Object Producer". That producer has to conform to all normative requirements in Cyber Observable Objects section 2.7, File Object.

3.2 Defined Object Consumers

A "Defined Object Consumer" that receives an Object from section 2 (Defined Object Data Models) is a "Consumer" of that Object. Defined Object Consumers **MUST** conform to all normative requirements in the section for that Object along with all of the general requirements pertaining to Objects as defined in section 3 of *STIX™ Version 2.0. Part 3: Cyber Observable Core Concepts*.

For example, an "Object Consumer" that can receive Network Traffic Objects is a "Network Traffic Object Consumer". That consumer has to conform to all normative requirements in Cyber Observable Objects Section 2.12, Network Traffic Object.

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

The contributions of the OASIS Cyber Threat Intelligence (CTI) Technical Committee members, enumerated in STIX™ Version 2.0. Part 1: STIX Core Concepts, are gratefully acknowledged.

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