TAXII™ Version 2.0.

Working Draft 01 - RC1

20 January 2017

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:
Bret Jordan (bret_jordan@symantec.com), Symantec Corp.
Mark Davidson (Mark.Davidson@nc4.com), NC4

Related work:
This specification replaces or supersedes:
• TAXII™ Version 1.1.1. Part 1: Overview. Edited by Mark Davidson, Charles Schmidt, and Bret Jordan. Latest version:

This specification is related to:
  http://docs.oasis-open.org/cti/stix/v2.0/csd01/part1-stix-core/stix-v2.0-csd01-part1-stix-core.html.
  http://docs.oasis-open.org/cti/stix/v2.0/csd01/part2-stix-objects/stix-v2.0-csd01-part2-stix-objects.html.
• STIX™ Version 2.0. Part 4: Cyber Observable Objects. Edited by Ivan Kirillov and Trey Darley. Latest version:
  http://docs.oasis-open.org/cti/stix/v2.0/csd01/part4-cyber-observable-objects/stix-v2.0-csd01-part4-cyber-observable-objects.html.
• STIX™ Version 2.0. Part 5: STIX Patterning. Edited by Ivan Kirillov and Trey Darley. Latest version:
  http://docs.oasis-open.org/cti/stix/v2.0/csd01/part5-stix-patterning/stix-v2.0-csd01-part5-stix-patterning.html.

Abstract:
TAXII™ is an application layer protocol for the communication of cyber threat information in a simple and scalable manner. This specification defines the TAXII RESTful API and its resources along with the requirements for TAXII Client and Server implementations.
Status:
This Working Draft (WD) has been produced by one or more TC Members; it has not yet been voted on by the TC or approved as a Committee Draft (Committee Specification Draft or a Committee Note Draft). The OASIS document Approvals begin officially with a TC vote to approve a WD as a Committee Draft. A TC may approve a Working Draft, revise it, and re-approve it any number of times as a Committee Draft.

URI patterns:
Initial publication URI:
http://docs.oasis-open.org/cti/taxii/v2.0/csd01/taxii-v2.0-csd01.docx
Permanent "Latest version" URI:
http://docs.oasis-open.org/cti/taxii/v2.0/taxii-v2.0.docx
(Managed by OASIS TC Administration; please don’t modify.)
Table of Contents

1 Introduction 5
  1.1 Terminology 5
  1.2 Normative References 5
  1.3 Non-Normative References 5
  1.4 Overview 6
    1.4.1 Channels Overview 6
    1.4.2 Collections Overview 6
    1.4.3 Transport 7
    1.4.4 Network Discovery 7
    1.4.5 Server Discovery 7
    1.4.6 TAXII API Root 8
    1.4.7 Content Negotiation 8
    1.4.8 Authentication 8
    1.4.9 Property Names 9
  1.5 Document Conventions 9
    1.5.1 Naming Conventions 9
    1.5.2 Font Colors and Style 9

2 Common Data Types 10

3 TAXII™ API 11
  3.1 URL Endpoints 11
  3.2 URL Parameters 11
  3.3 HTTP Status Codes 12
  3.4 GET <discovery> 14
    3.4.1 Requirements 14
  3.5 GET <api-root> 15
    3.5.1 Requirements 15
  3.6 GET /collections 16
    3.6.1 Requirements 16
  3.7 GET /collections/<name> 18
    3.7.1 Requirements 18
  3.8 GET /collections/<name>/manifest 19
    3.8.1 Requirements 19
  3.9 GET /collections/<name>/objects 21
    3.9.1 Requirements 21
  3.10 POST /collections/<name>/objects 22
    3.10.1 Requirements 22
  3.11 GET /collections/<name>/objects/<object-id> 23
    3.11.1 Requirements 24
3.12 GET /object-search 25
   3.12.1 Requirements 25
3.13 GET /status/<status-id> 26
   3.13.1 Requirements 26

4 TAXII™ Resources 29
   4.1 API Root Resource 29
   4.2 Discovery Resource 30
   4.3 Collection Resource 31
   4.4 Error Response 32
   4.5 Manifest Resource 32
   4.6 Object Resource 33
   4.7 Status Resource 34

5 Customizing TAXII Resources 37
   5.1 Custom Properties 37
      5.1.1 Requirements 37

6 Conformance 39
   6.1 TAXII Servers 39
   6.2 TAXII Clients 40
   6.3 Mandatory Features 40
      6.3.1 TODO 40
   6.4 Optional Features 40
      6.4.1 Certificate Authentication 40

Appendix A. Acknowledgments 42
Appendix B - Security Considerations 48
   11.1 Best Practices 48

Appendix C. Revision History 49
1 Introduction

TAXII is an application layer protocol for the communication of cyber threat information in a simple and scalable manner. This specification defines the TAXII RESTful API and its resources along with the requirements for TAXII Client and Server implementations.

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].

API Root - An instance of the TAXII API that is often used to align to trust groups.
API Root URL - The "root" URL for any particular instance of the TAXII API.
Channel - A publish-subscribe communications method where messages are exchanged.
CTI - Cyber Threat Intelligence
Collection - A logical group of CTI objects.
STIX - Structured Threat Information Expression
STIX Content - STIX documents, including STIX Objects, grouped as STIX Bundles.
STIX Object - A STIX Domain Object (SDO) or STIX Relationship Object (SRO)
TAXII - An application layer protocol for the communication of cyber threat information.
TAXII Client - A software package that connects to a TAXII server and supports the exchange of CTI.
TAXII Server - A software package that supports the exchange of CTI.

1.2 Normative References


1.3 Non-Normative References

TODO
1.4 Overview

This specification defines two communication methods - Channels and Collections - for transmitting CTI and is designed around a RESTful architecture.

1.4.1 Channels Overview

A TAXII Channel is maintained by a TAXII Server and enables TAXII Clients to exchange information with other TAXII Clients in an asynchronous, event-based manner. TAXII Clients may send and/or receive information to/from a channel. A TAXII Server may host multiple Channels per API Root and Channels are used to exchange information in a publish–subscribe manner.

Figure 1.2 below illustrates how Channel communications are used when a single TAXII Client sends a message to the TAXII Server, and that TAXII Server then distributes the message to all authorized TAXII Clients that are connected to the channel.

1.4.2 Collections Overview

A Collection is used by TAXII clients to send information to the TAXII server or request information from the TAXII server. A TAXII Server may host multiple Collections per API Root, and Collections are used to exchange information in a request–response manner.

Figure 1.3 below illustrates how Collection based communications are used when a single TAXII Client makes a request to a TAXII Server and the TAXII Server fulfills that request with information available to the TAXII Server (nominally from a database).
1.4.3 Transport
The TAXII protocol defined in this specification uses HTTPS as the default transport for all communications.

1.4.4 Network Discovery
This specification defines a DNS SRV record [TODO RFC 2782] that can be used to allow clients to auto-discover the server that the TAXII server is running on. This can be used to advertise TAXII Servers within a network (e.g., so that TAXII-enabled security infrastructure can locate an organization's internal TAXII Server) as well as to the general internet.

Requirements
- Organizations MAY implement a DNS SRV record in their DNS server to advertise the location of their TAXII Server.
  - If implemented, the service name for this version of TAXII MUST be “taxii”.

Examples
The following example is for a DNS SRV record advertising a TAXII server for the domain “example.com” located at taxi-hub-1.example.com:443:

```
_taxii._tcp.example.com. 86400 IN SRV 0 5 443 taxi-hub-1.example.com
```

1.4.5 Server Discovery
This specification defines a Discovery API that clients can use to discover the capabilities that the TAXII Server offers as well as meta-information about the TAXII Server (e.g., contact information). This specification uses the notation `<discovery>` to refer to the Discovery API URL.

Requirements
- A TAXII deployment SHOULD implement the Discovery API
- A TAXII deployment MAY advertise TAXII services through the Discovery API that are not hosted on that server.
### 1.4.6 TAXII API Root

Each instance of the TAXII API is located at an API Root. This API Root is the "root" URL of that particular instance of the TAXII API. Hosting multiple API Roots allows an implementer to mimic trust groups or groups of interest on a single TAXII Server.

#### Requirements
- A TAXII deployment **MAY** host more than one API Root on a single TAXII Server.
- A TAXII deployment **MUST** use a unique URL for every API Root.
- A TAXII deployment **MAY** have different authentication requirements for every API Root.

#### Examples
- `https://www.example.com:12345/api-group1`
- `https://www.example.com/api-group2`
- `https://taxii.someotherexample.com`

### 1.4.7 Content Negotiation

This specification uses HTTP content negotiation (TODO REF). It also defines the following media type for TAXII and STIX:

<table>
<thead>
<tr>
<th>Media Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>application/vnd.oasis.taxii+json</code></td>
<td>Any version of TAXII in JSON</td>
</tr>
<tr>
<td><code>application/vnd.oasis.taxii+json; version=2.0</code></td>
<td>TAXII version 2.0 in JSON</td>
</tr>
<tr>
<td><code>application/vnd.oasis.stix+json</code></td>
<td>Any version of STIX in JSON</td>
</tr>
<tr>
<td><code>application/vnd.oasis.stix+json; version=2.0</code></td>
<td>STIX version 2.0 in JSON</td>
</tr>
</tbody>
</table>

### 1.4.8 Authentication

This specification defines HTTP Basic as the mandatory to implement authentication system.

#### Requirements
- A TAXII deployment **MAY** disable HTTP Basic authentication.
- A TAXII deployment **MAY** silently ignore unauthorized requests.
- A TAXII deployment **SHOULD** filter the records returned instead of refusing to fulfill the request if a TAXII Client is unauthorized to access one or more objects or resources in a returned list.
- A TAXII deployment **MAY** deny access to any URLs in the TAXII API or any HTTP Methods defined in section 3.1 (todo).
For example, a server might choose to do this for Authentication or Authorization reasons.

### 1.4.9 Property Names

- 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 `discovery` resource has a property called `api_roots` and it must result in the JSON key name "api_roots".
- Properties marked required in the property tables **MUST** be present in the JSON serialization.

### 1.5 Document Conventions

#### 1.5.1 Naming Conventions

All type names, property names and literals are in lowercase. 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.

#### 1.5.2 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.
  - resource and type names are in red with a light red background – `collection`
  - property names are in bold style – `description`
  - literals (values) are in green with a green background – `complete`
- All examples in this document are expressed in Consolas 9-point font, with straight quotes and have a 2-space indentation.
  - Parts of the example may be omitted for conciseness and clarity. These omitted parts are denoted with the ellipses (...).
# 2 Common Data Types

This section defines the names and permitted values of common types used throughout this specification. These types are referenced by the “Type” column in other sections. This table does not, however, define the meaning of any fields using these types. These types may be further restricted elsewhere in the document.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>A boolean is a value of either true or false. Properties with this type MUST have a literal (unquoted) value of true or false.</td>
</tr>
<tr>
<td>integer</td>
<td>The integer data type represents a whole number. Unless otherwise specified, all integers MUST be capable of being represented as a signed 64-bit value. Additional restrictions MAY be placed on the type as described where it is used.</td>
</tr>
<tr>
<td>list</td>
<td>The list type defines an ordered sequence of one or more values. The phrasing “list of type &lt;type&gt;” is used to indicate that all values within the list MUST conform to a specific type. For instance, list of type string means that all values of the list must be of the string type. This definition does not specify the maximum or minimum number of allowed values in a list, however specific TAXII resource properties may define more restrictive upper and/or lower bounds for the length of the list. If a list property is required but no data is available, then an empty list MUST be returned.</td>
</tr>
<tr>
<td>string</td>
<td>The string data type represents a finite-length string of valid characters from the Unicode coded character set [ISO10646] that are encoded in UTF-8. Unicode incorporates ASCII [RFC0020] and the characters of many other international character sets.</td>
</tr>
<tr>
<td>timestamp</td>
<td>The timestamp type defines how timestamps are represented in TAXII and is represented in serialization as a string.</td>
</tr>
<tr>
<td>● The timestamp field MUST be a valid RFC 3339-formatted timestamp [RFC3339] using the format YYYY-MM-DDTHH:mm:ss[.s+]Z where the “s+” represents 1 or more sub-second values. The brackets denote that sub-second precision is optional, and that if no digits are provided, the decimal place MUST NOT be present.</td>
<td></td>
</tr>
<tr>
<td>● The timestamp MUST be represented in the UTC timezone and MUST use the “Z” designation to indicate this.</td>
<td></td>
</tr>
</tbody>
</table>
3 TAXII™ API

This section defines the TAXII API and all URL endpoints defined by this specification.

3.1 URL Endpoints

The following tables specifies the HTTP Methods and Resources that are defined for each URL.

<table>
<thead>
<tr>
<th>Resource URL</th>
<th>Methods</th>
<th>Resource Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;discovery&gt;</td>
<td>GET</td>
<td>discovery</td>
</tr>
<tr>
<td>&lt;api-root&gt;</td>
<td>GET</td>
<td>api</td>
</tr>
<tr>
<td>Collections</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;api-root&gt;/status/&lt;status-id&gt;</td>
<td>GET</td>
<td>status</td>
</tr>
<tr>
<td>&lt;api-root&gt;/collections</td>
<td>GET</td>
<td>list of type collection</td>
</tr>
<tr>
<td>&lt;api-root&gt;/collections/&lt;name&gt;</td>
<td>GET</td>
<td>collection</td>
</tr>
<tr>
<td>&lt;api-root&gt;/collections/&lt;name&gt;/manifest</td>
<td>GET</td>
<td>list of type manifest</td>
</tr>
<tr>
<td>&lt;api-root&gt;/collections/&lt;name&gt;/objects</td>
<td>GET, POST</td>
<td>object*</td>
</tr>
<tr>
<td>&lt;api-root&gt;/collections/&lt;name&gt;/objects/&lt;object-id&gt;</td>
<td>GET</td>
<td>object*</td>
</tr>
<tr>
<td>&lt;api-root&gt;/object-search</td>
<td>GET</td>
<td>object*</td>
</tr>
<tr>
<td>Channels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;TBD in a future Working Draft&gt;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* The actual format of objects is dependent on HTTP Content negotiation, as discussed in section [TODO REF]

3.2 URL Parameters

This section defines URL parameters and their meaning. The URL parameters defined in this section are used in the query portion of a URL. Each URL section defines which URL parameters are used.

<table>
<thead>
<tr>
<th>URL Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The identifier of the object that you are requesting. This is often a STIX ID. One or more identifiers MAY be specified in a single URL ID Parameter, comma separated.</td>
</tr>
</tbody>
</table>
Example
?
id=1234,123,12334

type
The object type that you want to filter on. For a media type of
application/vnd.oasis.stix+json; version=2.0

TAXII Servers MUST support all STIX Domain Object (SDO) and STIX
Relationship Object (SRO) types for versions of STIX that it supports. For
instance, a TAXII Server that supports only STIX 2.0 MUST support all STIX 2.0
SDO and SRO types.

Requests with a type value for STIX Objects defined by MUST NOT result in a
client error due to an invalid type.

TAXII Servers MAY support other values.

version
The version of the STIX object that is being requested.
- last tells the server to give you the highest (or latest) version it knows
  about.
- first tell the server to give you the lowest (or earliest) version it knows
  about.
- all tells the server to give you all versions it knows about.
- <datetime> tells the server to give you the object whose modified time
  matches exactly the provided datetime.
  - For example: "2016-01-01T01:01:01Z" tells the server to give
    you the exact STIX object with a modified time of
    "2016-01-01T01:01:01Z".
- If the version parameter is not present in the request, it MUST default to
  last.

added_after
Filters the result set to only include items added to the Channel or Collection
after the specified datetime. The value of this parameter is a timestamp.

Importantly, the added_after parameter is not in any way related to dates or
times in a STIX object.

3.3 HTTP Status Codes

This section lists commonly used HTTP status codes as a reference for implementers. This specification
does not modify the usage or meaning of HTTP status codes, and implementations are not restricted to
using HTTP status codes listed in this section.

<table>
<thead>
<tr>
<th>HTTP Code</th>
<th>Text Value</th>
<th>Notes (If any)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP 200</td>
<td>OK</td>
<td></td>
</tr>
<tr>
<td>HTTP Code</td>
<td>Status</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HTTP 202</td>
<td>Accepted</td>
<td>The request was accepted but has not yet been processed. This is used when a group of Objects or Messages are POSTed, and the server will process them asynchronously.</td>
</tr>
<tr>
<td>HTTP 400</td>
<td>Bad Request</td>
<td></td>
</tr>
<tr>
<td>HTTP 401</td>
<td>Unauthorized</td>
<td></td>
</tr>
<tr>
<td>HTTP 403</td>
<td>Forbidden</td>
<td></td>
</tr>
<tr>
<td>HTTP 404</td>
<td>Not Found</td>
<td></td>
</tr>
<tr>
<td>HTTP 405</td>
<td>Method Not Allowed</td>
<td>Each section defines requirements for certain HTTP Methods (e.g., GET, POST); each of these methods is said to be a supported method for the URL. HTTP requests that use a supported method MUST NOT result in an HTTP response with a status of 405 (Method Not Allowed). Other methods MAY result in an HTTP response with a status of 405. For example, the &lt;api-root&gt; defines requirements for the GET method, but not other methods. In this case, GET requests cannot result in an HTTP response with a status of 405; but POST requests may.</td>
</tr>
<tr>
<td>HTTP 406</td>
<td>Not Acceptable</td>
<td>For HTTP responses that contain a message body, the format of the message body is negotiated using the HTTP Accept header. Formats specified in the HTTP Accept header that the server is capable of providing are said to be acceptable. Formats specified in the HTTP Accept header that the server is not capable of providing are said to be unacceptable. If all options listed in the HTTP request's Accept header are unacceptable, the HTTP response must have a status code of HTTP 406 (Not Acceptable). Each section defines which response formats must be acceptable.</td>
</tr>
<tr>
<td>HTTP 410</td>
<td>Gone</td>
<td></td>
</tr>
<tr>
<td>HTTP 415</td>
<td>Unsupported Media Type</td>
<td>For HTTP requests that contain a message body, the format of the message body is identified using the HTTP Content-Type header. For Content-Types that the server does not support, the HTTP response must have a status code of HTTP 415 (Unsupported Media Type). Each section defines which Content-Types must be supported. Additional Content-Types, beyond those listed, MAY be supported.</td>
</tr>
<tr>
<td>HTTP 429</td>
<td>Too Many Requests</td>
<td></td>
</tr>
</tbody>
</table>
3.4 GET <discovery>

This URL allows TAXII Clients to discover information about a TAXII Server and any API Roots that the TAXII Client has access to.

3.4.1 Requirements

- Requests with an Accept header that contains `application/vnd.oasis.taxii+json` MUST NOT result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of `application/vnd.oasis.taxii+json` MUST contain a JSON `discovery` object.

Examples

**Conformant Discovery Request Properties**

<table>
<thead>
<tr>
<th>Request Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Line</td>
</tr>
<tr>
<td>URL Variable(s)</td>
</tr>
<tr>
<td>URL Parameters</td>
</tr>
<tr>
<td>Accept Header</td>
</tr>
<tr>
<td>Content-Type Header</td>
</tr>
<tr>
<td>Message Body</td>
</tr>
</tbody>
</table>

**GET Request**

```
GET /taxii HTTP/1.1
Accept: application/vnd.oasis.taxii+json; version=2.0
```

**Conformant Discovery Response Properties**

<table>
<thead>
<tr>
<th>Response Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Line</td>
</tr>
<tr>
<td>Content-Type Header</td>
</tr>
<tr>
<td>Message Body</td>
</tr>
</tbody>
</table>

**GET Response**
HTTP/1.1 200 OK
Content-Type: application/vnd.oasis.taxii+json; version=2.0
{
  "display_name": "Some TAXII Server",
  "description": "This TAXII server contains a listing of...",
  "contact": "string containing contact information",
  "default": "https://example.com/api2",
  "api_roots": [
    "https://example.com/api1",
    "https://example.com/api2",
    "https://companyfoo.com/trustgroup1"
  ]
}

3.5 GET <api-root>

This URL allows TAXII Clients to discover the available channels and collections at this specific API Root.

3.5.1 Requirements

- Requests with an Accept header that contains `application/vnd.oasis.taxii+json` MUST NOT result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of `application/vnd.oasis.taxii+json` MUST contain a JSON `api-root` object.

Examples

Conformant API Root Request Properties

<table>
<thead>
<tr>
<th>Request Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Line</td>
<td>GET &lt;api-root&gt;</td>
</tr>
<tr>
<td>URL Variable(s)</td>
<td>&lt;api-root&gt; - the base URL of the API Root containing the collection</td>
</tr>
<tr>
<td>URL Parameters</td>
<td>n/a</td>
</tr>
<tr>
<td>Accept Header</td>
<td>application/vnd.oasis.taxii+json; version=2.0</td>
</tr>
<tr>
<td>Content-Type Header</td>
<td>n/a</td>
</tr>
<tr>
<td>Message Body</td>
<td>n/a</td>
</tr>
</tbody>
</table>

GET Request

```
GET /some-api-base HTTP/1.1
Accept: application/vnd.oasis.taxii+json; version=2.0
```
**GET Response**

```
GET Response
HTTP/1.1 200 OK
Content-Type: application/vnd.oasis.taxii+json; version=2.0
{
   "display_name": "Malware Research Group",
   "description": "A trust group setup for malware researchers",
   "versions": ["taxii-2.0"],
   "channels": [],
   "collections": [
      {
         "url": "https://example.com/api-1/collections/high-value-indicators",
         "display_name": "High Value Indicator Collection",
         "description": "This data collection is for collecting high value IOCs",
         "can_read": true,
         "can_write": false,
         "media_types": [
            "application/vnd.oasis.stix+json; version=2.0"
         ],
         "objects_count": 923
      },
      {
         "url": "https://example.com/tg1/collections/24-hour-indicators",
         "display_name": "Indicators from the past 24-hours",
         "description": "This data collection is for collecting current IOCs",
         "can_read": true,
         "can_write": false,
         "media_types": [
            "application/vnd.oasis.stix+json; version=2.0"
         ],
         "objects_count": 7
      }
   ],
   "max_content_length": 9765625
}
```

### 3.6 GET /collections

This URL allows TAXII Clients to get a list of Collection resources that are available within an API Root.

#### 3.6.1 Requirements

- Requests with an Accept header that contains `application/vnd.oasis.taxii+json` **MUST NOT** result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of `application/vnd.oasis.taxii+json` **MUST** contain a JSON list, where each item in the list is a **collection**.
  - If there are zero **collection** to return, the result **MUST** be an empty list.
  - If there is one **collection** to return, the result **MUST** be a list with one item.
Examples

Conformant Collections Request Properties

<table>
<thead>
<tr>
<th>Request Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Line</td>
<td>GET &lt;api-root&gt;/collections</td>
</tr>
<tr>
<td>URL Variable(s)</td>
<td>&lt;api-root&gt; - the base URL of the API Root containing the collection</td>
</tr>
<tr>
<td>URL Parameters</td>
<td>n/a</td>
</tr>
<tr>
<td>Accept Header</td>
<td>application/vnd.oasis.taxii+json; version=2.0</td>
</tr>
<tr>
<td>Content-Type Header</td>
<td>n/a</td>
</tr>
<tr>
<td>Message Body</td>
<td>n/a</td>
</tr>
</tbody>
</table>

GET Request

GET https://example.com/api-1/collections HTTP/1.1
Accept: application/vnd.oasis.taxii+json; version=2.0

Conformant Collections Response Properties

<table>
<thead>
<tr>
<th>Response Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Line</td>
<td>HTTP/1.1 200 OK</td>
</tr>
<tr>
<td>Content-Type Header</td>
<td>application/vnd.oasis.taxii+json; version=2.0</td>
</tr>
<tr>
<td>Message Body</td>
<td>list of type collection</td>
</tr>
</tbody>
</table>

GET Response

HTTP/1.1 200 OK
Content-Type: application/vnd.oasis.taxii+json; version=2.0
[
  {
    "url": "https://example.com/api-1/collections/high-value-indicators",
    "display_name": "High Value Indicator Collection",
    "description": "This data collection is for collecting high value IOCs",
    "can_read": true,
    "can_write": false,
    "media_types": [
      "application/vnd.oasis.stix+json; version=2.0"
    ],
    "objects_count": 923
  },
  {
    "url": "https://example.com/tg1/collections/24-hour-indicators",
    "display_name": "Indicators from the past 24-hours",
    "can_read": true,
    "can_write": false,
    "media_types": [
      "application/vnd.oasis.stix+json; version=2.0"
    ]
  }
]
3.7 GET /collections/<name>

This URL allows TAXII Clients to get details about this specific Collection.

3.7.1 Requirements

- Requests with an Accept header that contains `application/vnd.oasis.taxii+json` MUST NOT result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of `application/vnd.oasis.taxii+json` MUST contain a JSON `collection` object.
  - If no `collection` is returned, the result MUST be an HTTP 404 (Not Found).

Examples

Conformant Collection Request Properties

<table>
<thead>
<tr>
<th>Request Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Request Line</strong></td>
<td>GET &lt;api-root&gt;/collections/&lt;name&gt;</td>
</tr>
<tr>
<td><strong>URL Variable(s)</strong></td>
<td>&lt;api-root&gt; - the base URL of the API Root containing the collection, &lt;name&gt; - the name of the collection being requested</td>
</tr>
<tr>
<td><strong>URL Parameters</strong></td>
<td>n/a</td>
</tr>
<tr>
<td><strong>Accept Header</strong></td>
<td>application/vnd.oasis.taxii+json; version=2.0</td>
</tr>
<tr>
<td><strong>Content-Type Header</strong></td>
<td>n/a</td>
</tr>
<tr>
<td><strong>Message Body</strong></td>
<td>n/a</td>
</tr>
</tbody>
</table>

GET Request

```
GET https://example.com/api-1/collections/high-value-indicators HTTP/1.1
Accept: application/vnd.oasis.taxii+json; version=2.0
```

Conformant Collection Response Properties

<table>
<thead>
<tr>
<th>Response Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Status Line</strong></td>
<td>HTTP/1.1 200 OK</td>
</tr>
</tbody>
</table>
### 3.8 GET /collections/<name>/manifest

This URL allows TAXII Clients to get a Manifest of Objects in this collection. This URL supports URL parameters to filter the results.

#### 3.8.1 Requirements

- Requests with an Accept header that contains `application/vnd.oasis.taxii+json` MUST NOT result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of `application/vnd.oasis.taxii+json` MUST contain a JSON list, where each item in the list is a `manifest`.
  - If there are zero `manifest` to return, the result MUST be an empty list.
  - If there is one `manifest` to return, the result MUST be a list with one item.
- The `id`, `type`, `version`, and `added_after` URL parameters MUST be supported at this URL endpoint.

### Examples

**Conformant Collection Manifest Request Properties**

<table>
<thead>
<tr>
<th>Request Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>id</strong></td>
</tr>
<tr>
<td><strong>type</strong></td>
</tr>
<tr>
<td><strong>version</strong></td>
</tr>
<tr>
<td><strong>added_after</strong></td>
</tr>
</tbody>
</table>
Request Line
GET <api-root>/collections/<name>/manifest

URL Variable(s)
@api-root - the base URL of the API Root containing the collection
@name - the name of the collection being requested

URL Parameters
id
| type
| version
| added_after

Accept Header
application/vnd.oasis.taxii+json; version=2.0

Content-Type Header
n/a

Message Body
n/a

GET Request
GET https://example.com/api-1/collections/high-value-indicators/manifest HTTP/1.1
Accept: application/vnd.oasis.taxii+json; version=2.0

Collection Manifest Response Properties

Response Properties

Status Line
HTTP/1.1 200 OK

Content-Type Header
application/vnd.oasis.taxii+json; version=2.0

Message Body
list of type manifest

GET Response
HTTP/1.1 200 OK
Content-Type: application/vnd.oasis.taxii+json; version=2.0
[
  {
    "url": "https://example.com/api-1/collections/high-value/objects/indicator--c410...",
    "date_added": "2016-11-01T03:04:05Z",
    "last_modified": "2016-11-03T12:30:59Z",
    "versions": [1,2,4,6,9],
    "media_types": ["application/vnd.oasis.stix+json; version=2.0"]
  },
  {
    "url": "https://example.com/api-1/collections/high-value-indicators/objects/
indicator--c410e480...-85307c121112",
    "date_added": "2016-11-01T10:29:05Z",
    "last_modified": "2016-11-01T10:29:05Z",
    "versions": [4],
    "media_types": ["application/vnd.oasis.stix+json; version=2.0"]
  }
]
3.9 GET /collections/<name>/objects

This URL allows TAXII Clients to get multiple Objects from this collection. This URL supports URL parameters to filter the results.

3.9.1 Requirements

- Requests with an Accept header that contains `application/vnd.oasis.stix+json` MUST NOT result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of `application/vnd.oasis.stix+json` MUST contain an `object` resource.
  - If there are no `object` to return, the result MUST be an HTTP 404 (Not Found).
- The `id`, `type`, `version`, and `added_after` URL parameters MUST be supported at this URL endpoint.

Examples

Conformant Collection Objects Request Properties

<table>
<thead>
<tr>
<th>Request Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Line</td>
<td>GET &lt;api-root&gt;/collections/&lt;name&gt;/objects</td>
</tr>
<tr>
<td>URL Variable(s)</td>
<td>&lt;api-root&gt; - the base URL of the API Root containing the collection &lt;name&gt; - the name of the collection being requested</td>
</tr>
<tr>
<td>URL Parameters</td>
<td>id, type, version, added_after</td>
</tr>
<tr>
<td>Accept Header</td>
<td>application/vnd.oasis.stix+json; version=2.0</td>
</tr>
<tr>
<td>Content-Type Header</td>
<td>n/a</td>
</tr>
<tr>
<td>Message Body</td>
<td>n/a</td>
</tr>
</tbody>
</table>

GET Request

```
GET https://example.com/api-1/collections/high-value-indicators/objects HTTP/1.1
Accept: application/vnd.oasis.stix+json; version=2.0
```

Conformant Collection Objects Response Properties

<table>
<thead>
<tr>
<th>Response Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Line</td>
<td>HTTP/1.1 200 OK</td>
</tr>
<tr>
<td>Content-Type Header</td>
<td>application/vnd.oasis.stix+json; version=2.0</td>
</tr>
</tbody>
</table>
GET Response

HTTP/1.1 200 OK
Content-Type: application/vnd.oasis.stix+json; version=2.0
{
  "type": "bundle",
  ...
  "indicators": [
    {
      "type": "indicator",
      ...
    }
  ]
}

3.10 POST /collections/<name>/objects

This URL allows authorized TAXII Clients to add Objects to this collection.

3.10.1 Requirements

- Requests with an Accept header that contains application/vnd.oasis.taxii+json MUST NOT result in an HTTP 406 (Not Acceptable) response.
- Requests with a Content-Type header that contains application/vnd.oasis.stix+json MUST NOT result in an HTTP 415 (Unsupported Media Type) response.
- HTTP 202 Responses with a Content-Type of application/vnd.oasis.taxii+json MUST contain an status resource.
  - The client SHOULD periodically poll the URL contained in the status url property to retrieve the most up-to-date status, until such a time that the status property returns a value of complete.
  - TAXII Servers SHOULD NOT delete status messages for at least 24 hours.

Examples

Conformant Collection Objects Request Properties

<table>
<thead>
<tr>
<th>Request Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Line</td>
</tr>
<tr>
<td>URL Variable(s)</td>
</tr>
<tr>
<td>URL Parameters</td>
</tr>
<tr>
<td>Accept Header</td>
</tr>
</tbody>
</table>
### POST Request

```
POST https://example.com/api-1/collections/high-value-indicators/objects HTTP/1.1
Accept: application/vnd.oasis.taxii+json
Content-Type: application/vnd.oasis.stix+json; version=2.0
{
  "type": "bundle",
  ...
  "objects": [
    {
      "type": "indicator",
      "id": "indicator--c410e480-e42b-47d1-9476-85307c12bcbf",
      ...
    }
  ]
}
```

### Conformant Collection Objects Response Properties

<table>
<thead>
<tr>
<th>Response Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Status Line</strong></td>
</tr>
<tr>
<td><strong>Content-Type Header</strong></td>
</tr>
<tr>
<td><strong>Message Body</strong></td>
</tr>
</tbody>
</table>

### POST Response

```
HTTP/1.1 202 Accepted
Content-Type: application/vnd.oasis.taxii+json; version=2.0
{
  "url": "https://example.com/api-1/status/1234",
  "status": "pending",
  "request_url": "https://example.com/api-1/collections/coll1/objects",
  "request_timestamp": "2016-11-02T12:34:34.12345Z",
  "total_items": 1,
  "success_count": 1,
  "success_items": [
    {
      "id": "indicator--c410e480-e42b-47d1-9476-85307c12bcbf",
      "url": "https://example.com/api-1/collections/coll1/objects/indicator--c410e480..."
    }
  ]
}
```

### 3.11 GET /collections/<name>/objects/<object-id>

This URL allows TAXII Clients to get a specific Object from this collection. This URL only supports the version URL parameters to filter the results.
3.11.1 Requirements

- Requests with an Accept header that contains `application/vnd.oasis.stix+json` MUST NOT result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of `application/vnd.oasis.stix+json` MUST contain an object resource.
  - If no object is returned, the result MUST be an HTTP 404 (Not Found).
- The version URL parameter MUST be supported at this URL endpoint.

Examples

Conformant Collection Objects ID Request Properties

<table>
<thead>
<tr>
<th>Request Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Request Line</strong></td>
</tr>
<tr>
<td><strong>URL Variable(s)</strong></td>
</tr>
<tr>
<td><strong>URL Parameters</strong></td>
</tr>
<tr>
<td><strong>Accept Header</strong></td>
</tr>
<tr>
<td><strong>Content-Type Header</strong></td>
</tr>
<tr>
<td><strong>Message Body</strong></td>
</tr>
</tbody>
</table>

GET Request

```
GET https://example.com/api-1/collections/high-value-indicators/object/indicator--252c7c11-daf2-42bd-b43b-be6sedca9f61 HTTP/1.1
Accept: application/vnd.oasis.stix+json; version=2.0
```

Conformant Collection Objects ID Response Properties

<table>
<thead>
<tr>
<th>Response Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Status Line</strong></td>
</tr>
<tr>
<td><strong>Content-Type Header</strong></td>
</tr>
<tr>
<td><strong>Message Body</strong></td>
</tr>
</tbody>
</table>

*The actual format of objects is dependent on HTTP Content negotiation, as discussed in Section [TODO REF]*

GET Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.oasis.stix+json; version=2.0
{
  "type": "bundle",
```
3.12 GET /object-search

This URL allows TAXII Clients to retrieve multiple Objects from any collection in this API Root. This URL supports URL parameters to filter the results.

3.12.1 Requirements

- Requests with an Accept header that contains application/vnd.oasis.stix+json MUST NOT result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of application/vnd.oasis.stix+json MUST contain an object resource.
  - If no object is returned, the result MUST be an HTTP 404 (Not Found).
- The id, type, version, and added_after URL parameters MUST be supported at this URL endpoint.

Examples

Conformant Object Search Request Properties

<table>
<thead>
<tr>
<th>Request Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Line</td>
<td>GET &lt;api-root&gt;/object-search</td>
</tr>
<tr>
<td>URL Variable(s)</td>
<td>&lt;api-root&gt; - the base URL of the API Root</td>
</tr>
<tr>
<td>URL Parameters</td>
<td>id, type, version, added_after</td>
</tr>
<tr>
<td>Accept Header</td>
<td>application/vnd.oasis.stix+json; version=2.0</td>
</tr>
<tr>
<td>Content-Type Header</td>
<td>n/a</td>
</tr>
<tr>
<td>Message Body</td>
<td>n/a</td>
</tr>
</tbody>
</table>

GET Request

GET https://example.com/api-1/object-search?type=indicator HTTP/1.1
Accept: application/vnd.oasis.stix+json; version=2.0
Conformant Object Search Response Properties

<table>
<thead>
<tr>
<th>Response Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Status Line</strong></td>
<td>HTTP/1.1 200 OK</td>
</tr>
<tr>
<td><strong>Content-Type Header</strong></td>
<td>application/vnd.oasis.stix+json; version=2.0</td>
</tr>
<tr>
<td><strong>Message Body</strong></td>
<td>object*</td>
</tr>
</tbody>
</table>

* The actual format of objects is dependent on HTTP Content negotiation, as discussed in Section [TODO REF]

GET Response

```
HTTP/1.1 200 OK
Content-Type: application/vnd.oasis.stix+json; version=2.0
{
  "type": "bundle",
  "indicators": [
    {
      "type": "indicator",
      "indicators": []
    }
  ]
}
```

3.13 GET /status/<status-id>

This URL allows TAXII Clients to get a Status of a previous POST request (e.g., a POST to create objects in a Collection) that is being processed or has been recently processed by the server. This is used to monitor the status of requests that have resulted in an HTTP 202 (Accepted) response.

3.13.1 Requirements

- Requests with an Accept header that contains application/vnd.oasis.taxii+json MUST NOT result in an HTTP 406 (Not Acceptable) response.
- HTTP 200 Responses with a Content-Type of application/vnd.oasis.taxii+json MUST contain an status resource..
  - If no status is returned, the result MUST be an HTTP 404 (Not Found).

Examples

Conformant Status Request Properties

<table>
<thead>
<tr>
<th>Request Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Request Line</strong></td>
<td>GET &lt;api-root&gt;/status/&lt;status-id&gt;</td>
</tr>
<tr>
<td><strong>URL Variable(s)</strong></td>
<td>&lt;api-root&gt; - the base URL of the API Root &lt;status-id&gt; - the ID of the status message being requested</td>
</tr>
</tbody>
</table>
GET Request

GET /some-api-root/status/123456 HTTP/1.1
Accept: application/vnd.oasis.taxii+json; version=2.0

Conformant Status Response Properties

<table>
<thead>
<tr>
<th>Response Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Line</td>
</tr>
<tr>
<td>Content-Type Header</td>
</tr>
<tr>
<td>Message Body</td>
</tr>
</tbody>
</table>

Common Errors

<table>
<thead>
<tr>
<th>Common Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Code</td>
</tr>
<tr>
<td>HTTP 404</td>
</tr>
</tbody>
</table>

GET Response

HTTP/1.1 200 OK
Content-Type: application/vnd.oasis.taxii+json; version=2.0
{
    "url": "https://example.com/api-1/status/1234",
    "status": "pending",
    "request_url": "https://example.com/api-1/collections/coll1/objects",
    "request_timestamp": "2016-11-02T12:34:34.12345Z",
    "total_items": 4,
    "success_count": 1,
    "success_items": [
        {
            "id": "indicator--c410e480-e42b-47d1-9476-85307c12bcbf",
            "url": "https://example.com/api-1/collections/coll1/objects/indicator--
c410e480-e42b-47d1-9476-85307c12bcbf"
        }
    ],
    "failure_count": 1,
    "failure_items": [
        {
            "id": "malware--664fa29d-bf65-4f28-a667-bdb76f29ec98",
            "message": "Unable to process object"
        }
    ]
}
{ }
"pending_count": 2,
"pending_items": [
  "indicator--252c7c11-daf2-42bd-843b-be65edca9f61",
  "relationship--045585ad-a22f-4333-af31-bfd503a683b5"
]
}
4 TAXII™ Resources

This section defines the TAXII resources and their representations.

4.1 API Root Resource

Resource Name: api-root

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display_name (required)</td>
<td>string</td>
<td>A human readable text/plain name used to identify this API instance.</td>
</tr>
<tr>
<td>description (optional)</td>
<td>string</td>
<td>A human readable text/plain description for this API Root.</td>
</tr>
<tr>
<td>versions (required)</td>
<td>list of type string</td>
<td>Lists the versions of TAXII that this API Root is compatible with.</td>
</tr>
<tr>
<td></td>
<td>taxii-2.0</td>
<td>taxii-2.0 MUST be included in this list to indicate conformance with this specification.</td>
</tr>
<tr>
<td>channels (required)</td>
<td>list of type channel</td>
<td>&lt;TODO&gt;</td>
</tr>
<tr>
<td>collections (required)</td>
<td>list of type collection</td>
<td>&lt;TODO&gt;</td>
</tr>
<tr>
<td>max_content_length (required)</td>
<td>integer</td>
<td>The maximum value of the request body &quot;Content-Length&quot; in octets (8-bit bytes) that the server can support. This applies to requests and responses and is determined by the server. Requests or responses with total body length values smaller than this value MUST NOT result in an HTTP 413 (Request Entity Too Large) response. Absence of this value means the server is choosing to not provide this information. This property is needed to help make sure a client does not have to keep guessing at how much data it can send. For example, if a server only supports payloads up to 10MB and a client wants to send 50MB of data, without this, the client would have to keep guessing as to what the server will support.</td>
</tr>
</tbody>
</table>

Examples

```
{
  "display_name": "Malware Research Group",
  "description": "A trust group setup for malware researchers",
  "versions": ["taxii-2.0"],
  "channels": []
}
```
4.2 Discovery Resource

Resource Name: discovery

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display_name (require)</td>
<td>string</td>
<td>A human readable text/plain name used to identify this server.</td>
</tr>
<tr>
<td>description (optional)</td>
<td>string</td>
<td>A human readable text/plain description for this server.</td>
</tr>
<tr>
<td>contact (optional)</td>
<td>string</td>
<td>The human readable text/plain contact information for this server and or the administrator of this server.</td>
</tr>
<tr>
<td>default (optional)</td>
<td>string</td>
<td>The default API Root that a TAXII Client MAY use. Absence of this field indicates that there is no default API Root.</td>
</tr>
<tr>
<td>api_roots (required)</td>
<td>list of type string</td>
<td>A list of URLs that identify API Roots that are hosted on this server or that this server knows about. This list MAY be filtered on a per-client basis.</td>
</tr>
</tbody>
</table>

Examples

```json
{
  "display_name": "Some TAXII Server",
  "description": "This TAXII server contains a listing of...",
} ```
4.3 Collection Resource

Resource Name: collection

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url (optional)</td>
<td>string</td>
<td>The full URL of this collection. This property <strong>MUST</strong> be present in GET responses.</td>
</tr>
<tr>
<td>display_name (required)</td>
<td>string</td>
<td>A human readable text/plain name used to identify this Collection.</td>
</tr>
<tr>
<td>description (optional)</td>
<td>string</td>
<td>A human readable text/plain description for this Collection.</td>
</tr>
<tr>
<td>can_read (optional)</td>
<td>boolean</td>
<td>Indicates if the requester can read (i.e., GET) objects from this Collection. Absence of this field is equivalent to a value of <strong>false</strong>.</td>
</tr>
<tr>
<td>can_write (optional)</td>
<td>boolean</td>
<td>Indicates if the requester can write (i.e., POST) objects to this Collection. Absence of this field is equivalent to a value of <strong>false</strong>.</td>
</tr>
<tr>
<td>media_types (optional)</td>
<td>list of type</td>
<td>A list of supported media types for Objects in this collection. Absence of this field is equivalent to a value of <strong>application/vnd.oasis.stix+json</strong>.</td>
</tr>
<tr>
<td>objects_count (optional)</td>
<td>integer</td>
<td>The current number of objects contained in this Collection. This field <strong>SHOULD</strong> be populated by the server. A server might choose to omit this field for clients that are not able to read from the collection.</td>
</tr>
</tbody>
</table>

Examples

```
{
    "url": "https://example.com/api-1/collections/high-value-indicators",
    "display_name": "High Value Indicator Collection",
    "description": "This data collection is for collecting high value IOCs",
    "can_read": true,
    "can_write": false,
    "media_types": [
        "application/vnd.oasis.stix+json; version=2.0"
    ],
    "objects_count": 923
}
```
4.4 Error Response

Resource Name: error

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error_id (optional)</td>
<td>string</td>
<td>An identifier for this particular error. A TAXII Server might choose to assign each error occurrence it's own identifier in order to facilitate debugging.</td>
</tr>
<tr>
<td>display_name (required)</td>
<td>string</td>
<td>A human readable text/plain name describing this error.</td>
</tr>
<tr>
<td>description (optional)</td>
<td>string</td>
<td>A human readable text/plain description that gives details about the error or problem that was encountered by the application.</td>
</tr>
<tr>
<td>error_code (optional)</td>
<td>string</td>
<td>The error code for this error. A TAXII Server might choose to assign a common error code to all errors of the same type.</td>
</tr>
<tr>
<td>http_status (optional)</td>
<td>string</td>
<td>The HTTP status code applicable to this error.</td>
</tr>
<tr>
<td>external_details (optional)</td>
<td>string</td>
<td>A URL that points to additional details. Absence of this field indicates that there are no additional details.</td>
</tr>
<tr>
<td>details (optional)</td>
<td>object</td>
<td>The details objects provide a location for additional application specific details.</td>
</tr>
</tbody>
</table>

Examples

```json
{
  "error_id": "1234",
  "display_name": "Error condition XYZ",
  "description": "This error is caused when the application tries to access data...",
  "error_code": "581234",
  "http_status": "409",
  "external_details": "http://example.com/ticketnumber1/errorid-1234",
  "details": {
    "somekey1": "somevalue",
    "somekey2": "some other value"
  }
}
```

4.5 Manifest Resource

Resource Name: manifest

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>url (required)</td>
<td>string</td>
<td>The full URL of this object</td>
</tr>
<tr>
<td>date_added (optional)</td>
<td>timestamp</td>
<td>The date this object was add to the server.</td>
</tr>
<tr>
<td>last_modified (optional)</td>
<td>timestamp</td>
<td>The date this object was last updated or the last modified date of the most current version. For example, if version 2 was added after version 3, this date would contain the last modified date from version 3 not version 2 as version 3 is the most current version.</td>
</tr>
<tr>
<td>versions (optional)</td>
<td>list of type string</td>
<td>A list of available STIX versions. This field is only meaningful for objects that are available in a STIX format.</td>
</tr>
<tr>
<td>media_types (optional)</td>
<td>list of type string</td>
<td>The media types that this object can be requested in.</td>
</tr>
</tbody>
</table>

Examples

```
{
  "url": "https://example.com/api-1/collections/high-value/objects/indicator--c410...",
  "date_added": "2016-11-01T03:04:05Z",
  "last_modified": "2016-11-03T12:30:59Z",
  "versions": [1,2,4,6,9],
  "media_types": ["application/vnd.oasis.stix+json; version=2.0"]
}
```

4.6 Object Resource

Resource Name: object

This resource type is negotiated based on the media type. If the media type in the Accept or Content-Type header contains application/vnd.oasis.stix+json; version=2.0 then this resource type is a STIX Bundle version 2.0 as defined in the STIX specification located here [TODO add reference].

Examples

```
{
  "url": "https://example.com/api-1/status/1234",
  "status": "pending",
  "request_url": "https://example.com/api-1/collections/coll1/objects",
  "request_timestamp": "2016-11-02T12:34:34.12345Z",
  "total_items": 1,
  "success_count": 1,
  "success_items": [
    {
      "id": "indicator--c410e480-e42b-47d1-9476-85307c12acb6",
      "url": "https://example.com/api-1/collections/coll1/objects/indicator--c410e480...
    }
  ]
}
```
### 4.7 Status Resource

**Resource Name:** status

This resource is returned when an HTTP 202 (Accepted) response is given to a POST request. This resource conveys the status of a bulk creation.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url (required)</td>
<td>string</td>
<td>The absolute full URL of this Status resource.</td>
</tr>
<tr>
<td>status (required)</td>
<td>string</td>
<td>The overall status of a previous POST request where an HTTP 202 (Accept) was returned. The value of this property MUST be one of <strong>complete</strong> or <strong>pending</strong>. A value of <strong>complete</strong> indicates that this resource will not be updated further and <strong>MAY</strong> be removed in the future. A status of <strong>pending</strong> indicates that this resource <strong>MAY</strong> update in the future.</td>
</tr>
<tr>
<td>request_url (optional)</td>
<td>string</td>
<td>The URL of the request that this status resource is monitoring.</td>
</tr>
<tr>
<td>request_timestamp (optional)</td>
<td>timestamp</td>
<td>The datetime of the request that this status resource is monitoring.</td>
</tr>
<tr>
<td>total_items (required)</td>
<td>integer</td>
<td>The total number of items that were in the request. For a STIX Bundle this would be the number of objects in the Bundle.</td>
</tr>
<tr>
<td>success_count (optional)</td>
<td>integer</td>
<td>The number of items that were successfully created. Absence of this field is equivalent to a value of &quot;0&quot; (zero).</td>
</tr>
<tr>
<td>success_items (optional)</td>
<td>list of type success-item</td>
<td>A list of items that were successfully processed.</td>
</tr>
<tr>
<td>failure_count (optional)</td>
<td>integer</td>
<td>The number of items that failed to be created. Absence of this field is equivalent to a value of &quot;0&quot; (zero).</td>
</tr>
<tr>
<td>failure_items (optional)</td>
<td>list of type failure-item</td>
<td>A list of items that were not successfully processed.</td>
</tr>
<tr>
<td>pending_count (optional)</td>
<td>integer</td>
<td>The number of items that have yet to be processed.</td>
</tr>
<tr>
<td>pending_items (optional)</td>
<td>list of type string</td>
<td>A list of identifiers for items that have yet to be processed.</td>
</tr>
</tbody>
</table>
Type Name: **success-item**

This type contains a success item by ID and location.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (required)</td>
<td>string</td>
<td>The identifier of the item that was created. For types that have an identifier, that identifier should be used here.</td>
</tr>
<tr>
<td>url (required)</td>
<td>string</td>
<td>The URL location of the created item.</td>
</tr>
</tbody>
</table>

Type Name: **failure-item**

This type contains a failure item by ID and location.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (required)</td>
<td>string</td>
<td>The identifier of the item that was created. For types that have an identifier, that identifier should be used here.</td>
</tr>
<tr>
<td>message (optional)</td>
<td>string</td>
<td>A message indicating why the item failed to be created.</td>
</tr>
</tbody>
</table>

Examples

```json
{
  "url": "https://example.com/api-1/status/1234",
  "status": "pending",
  "request_url": "https://example.com/api-1/collections/coll1/objects",
  "request_timestamp": "2016-11-02T12:34:34.12345Z",
  "total_items": 4,
  "success_count": 1,
  "success_items": [
    {
      "id": "indicator--c410e480-e42b-47d1-9476-85307c12bcbf",
      "url": "https://example.com/api-1/collections/coll1/objects/indicator--c410e480-e42b-47d1-9476-85307c12bcbf"
    }
  ],
  "failure_count": 1,
  "failure_items": [
    {
      "id": "malware--664fa29d-bf65-4f28-a667-bdb76f29ec98",
      "message": "Unable to process object"
    }
  ],
  "pending_count": 2,
  "pending_items": [
    "indicator--252c7c11-daf2-42bd-843b-be65edca9f61",
    "relationship--045585ad-a22f-4333-af33-bfd503a683b5"
  ]
}
```
5 Customizing TAXII Resources

[TODO add section description]

5.1 Custom Properties

It is understood that there will be cases where certain information exchanges can be improved by adding properties that are not specified nor reserved in this document; these properties are called Custom Properties. This section provides guidance and requirements for how TAXII Servers can use Custom Properties and how TAXII Clients should interpret them in order to extend TAXII in an interoperable manner.

5.1.1 Requirements

- A TAXII resource MAY have any number of Custom Properties.
- Custom Property names MUST be in ASCII and are limited to characters a-z (lowercase ASCII) and underscore (_).
- Custom Property names SHOULD start with "x_" followed by a source unique identifier (like a domain name), an underscore and then the name. For example: x_examplecom_customfield.
- Custom Property names SHOULD be no longer than 30 ASCII characters in length.
- Custom Property names MUST have a minimum length of 3 ASCII characters.
- Custom Property names MUST be no longer than 256 ASCII characters in length.
- Custom Property names that are not prefixed with "x_" may be used in a future version of the specification for a different meaning. If compatibility with future versions of this specification is required, the "x_" prefix MUST be used.
- Custom Property names SHOULD be unique when produced by the same source and SHOULD use a consistent namespace prefix (e.g., a domain name).
- Custom Properties SHOULD only be used when there is no existing properties defined by the TAXII specification that fulfills that need.

A TAXII Client that receives a TAXII message with one or more Custom Properties it does not understand MAY refuse to process the message further, or silently ignore non-understood properties and continue processing the message.

The reporting and logging of errors originating from the processing of Custom Properties depends heavily on the TAXII Server and Client implementations and is therefore not covered in this specification.

Non-Normative: TAXII Servers that produce messages that contain Custom Properties should be aware of the variability of TAXII Client behavior depending on whether or not the TAXII Client understands the Custom Properties present in a TAXII message. Rules for processing Custom Properties should be well defined and accessible to any TAXII Client that would be reasonably expected to parse them.

Examples

```json
{
  ...
  "x_acmeinc_scoring": {
    "impact": "high",
  }
}```
"probability": "low",
},
...
]
6 Conformance

6.1 TAXII Servers

A "TAXII 2.0 Server" is any software that conforms to the following normative requirements:

1. It MUST accept connections using HTTPS [TODO RFC7230].
2. It MUST accept connections using TLS version 1.2 [TODO RFC5246] and SHOULD accept connections using TLS version 1.3 [TODO REF] or higher.
3. It SHOULD NOT accept any TLS 1.2 connections that use any of the cipher suites that are listed in the cipher suite black list in Appendix A of RFC 7540 (TODO REF).
4. It MUST support at least one API Root.
5. It MUST support at least Collections or Channels.
   a. For Collections it MUST support the entire TAXII API including the Discovery API as defined in section (todo)
   b. For Channels it MUST support the entire TAXII API including the Discovery API as defined in section (todo)
   c. It MAY implement other URLs and/or methods.
6. It MUST define the URL of the Discovery API to be /taxii and it MUST be located at the root of the server, e.g., https://someserver.com/taxii
7. It MUST accept the HTTP Methods defined for each URL as specified in section 3.1 (todo).
   a. It MAY support other HTTP Methods at each URL.
8. It MUST honor the version parameter found in the Accept: header during content negotiation.
   a. It SHOULD return a HTTP 406 (Not Acceptable) error if the server does not support the requested version.
   b. It SHOULD respond with the highest version of TAXII that the server supports if the version parameter is omitted.
9. It MUST NOT return an HTTP 406 (Not Acceptable) when presented with a valid media type in either the Accept: or Content-Type: header.
10. It MUST include the version parameter in the Content-Type: header with the appropriate TAXII version number when responding to Accept: headers of application/vnd.oasis.taxii+json and application/vnd.oasis.stix+json.
    a. For example: Content-Type: application/vnd.oasis.taxii+json; version=2.0
11. It MUST return the status resource when an HTTP 202 (Accepted) response is given to a POST request.
12. It SHOULD include the error message in the response payload of any HTTP error response to give additional application specific details about the error.
13. It MUST be able to create content encoded as JSON.
14. It MUST include all required properties in the created content.
15. All properties MUST conform to the data type and normative requirements for that property.
16. It MUST support all features listed in Section 7.2, Mandatory Features.
17. It MAY support any features listed in Section 7.3, Optional Features. Software supporting an optional feature MUST comply with the normative requirements of that feature.
6.2 TAXII Clients

A "TAXII 2.0 Client" is any software that consumes CTI content and conforms to the following normative requirements:

1. It **MUST** be able to communicate over HTTPS.
2. It **MUST** be able to use TLS 1.2 and **SHOULD** be able to use TLS version 1.3 [TODO REF] or higher.
3. It **SHOULD NOT** use TLS 1.2 with any of the cipher suites that are listed in the cipher suite blacklist in Appendix A of RFC 7540 (TODO REF).
4. It **SHOULD** be capable of looking up and using the TAXII SRV record from DNS.
5. It **MUST** include an **Accept:** header in all HTTP requests.
   a. It **SHOULD** include the version parameter in the **Accept:** header wherever a TAXII or STIX media type is used.
   b. It **MUST** include an appropriate media range in the **Accept:** header for STIX and TAXII content. For example;
      i. A media range of `application/vnd.oasis.taxii+json` in the accept header indicates that any version of TAXII is acceptable in the response.
      ii. A media range of `application/vnd.oasis.taxii+json; version=2.0` in the accept header indicates that ONLY TAXII 2.0 is acceptable in the response.
      iii. A media range of `application/vnd.oasis.stix+json` in the accept header indicates that any version of STIX is acceptable in the response.
      iv. A media range of `application/vnd.oasis.stix+json; version=2.0` in the accept header indicates that ONLY STIX 2.0 is acceptable in the response.
6. It **MUST** support parsing all required properties for the content that it supports.
7. It **MUST** support all features listed in section 7.2, Mandatory Features.
8. It **MAY** support any features listed in section 7.3, Optional Features. Software supporting an optional feature **MUST** comply with the normative requirements of that feature.

6.3 Mandatory Features

6.3.1 TODO

A TAXII 2.0 Server or TAXII 2.0 Client **MUST** support X by following the normative requirements listed in Section X.

6.4 Optional Features

6.4.1 Certificate Authentication

A TAXII 2.0 Server **MAY** support certificate authentication. Software claiming to support certificate authentication **MUST** follow the normative requirements listed in this section.

- **TAXII Servers MUST** implement PKIX X.509 certificates and certificate revocation lists [TODO RFC5280 and TODO RFC6818].
- **TAXII Servers MUST** support authenticating certificates using PKIX [TODO RFC6125].
• The default strategy for authenticating certificates **SHOULD** be PKIX as defined in RFC 5280, RFC 6818, RFC 6125 et al.

• Taxii Servers and Clients **MAY** support other certification verification policies such as:
  ○ Certificate Pinning: A single or limited set of either hard-coded or physically distributed pinned certificate authorities or end-entity certificates.
  ○ DANE: DNS-based Authentication of Named Entities [TODO RFC 7671]
  ○ Note that Self-Signed Certificates (like other certificates which cannot be verified by PKIX) **MAY** be supported via Certificate Pinning and/or DANE as noted above for limited, closed user group applications.
Appendix A. Acknowledgments

TAXII Subcommittee Chairs:
   Bret Jordan, Symantec Corp.
   Mark Davidson, NC4

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

   Terry MacDonald, Cosive
   Jane Ginn, Cyber Threat Intelligence Network, Inc. (CTIN)
   Richard Struse, DHS Office of Cybersecurity and Communications
   Iain Brown, GDS
   Eric Burger, Georgetown University
   Jason Keirstead, IBM
   Allan Thomson, LookingGlass Cyber
   Mark Davidson, NC4
   John-Mark Gurney, New Context Services, Inc.
   Bret Jordan, Symantec Corp.

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

   David Crawford, Aetna
   Marcos Orallo, Airbus Group SAS
   Sébastien Rummelhardt, Airbus Group SAS
   Roman Fiedler AIT Austrian Institute of Technology
   Giuseppe Settanni, AIT Austrian Institute of Technology
   Florian Skopik, AIT Austrian Institute of Technology
   Ryan Clough, Anomali
   Wei Huang, Anomali
   Hugh Njemanze, Anomali
   Katie Pelusi, Anomali
   Aaron Shelmire, Anomali
   Jason Trost, Anomali
   Dean Thompson, Australia and New Zealand Banking Group (ANZ Bank)
   Alexander Foley, Bank of America
   Tony Pham, Bank of America
   Sounil Yu, Bank of America
   Humphrey Christian, Bay Dynamics
   Ryan Stolte, Bay Dynamics
   Owen Johnson, Blue Coat Systems, Inc.
   Aubrey Merchant, Blue Coat Systems, Inc.
   Sarah Kelley, Center for Internet Security (CIS)
   Cory Kennedy, CenturyLink
Appendix B - Security Considerations

This appendix should talk about security issues with DNS SRV Records. For TAXII Server the reference identifier expected in certificates SHALL be the original name, as per RFC 6125. The concern has to do with a DNS query to example.com and then getting back a result of taxii-hub-1.example.com and the client then making a followon request. Dave Cridland can give more details here. The problem we are trying to address is that if you do a DNS lookup for the domain example.com and it returns a SRV record for taxi.example.com, there is no way to verify that record without DNSSEC. 1) remove 2) just call out that you SHOULD use DNSSEC 3) talk about the end server needing a cert for all of the names aka example.com + foo.example.com

11.1 Best Practices

- TAXII Clients and TAXII Servers should implement and use NTP to maintain system clocks.
## Appendix C. Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Editor</th>
<th>Changes Made</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>2017-01-20</td>
<td>Bret Jordan, Mark Davidson</td>
<td>Initial Version</td>
</tr>
</tbody>
</table>