STIX 2.0 Specification - Pre-Draft

Top Level Objects - Version 0.2

Document Table of Contents

- 1. STIX Top Level Object (TLO)
  - 1.1. TLO Common Properties
  - 1.2. IDs and References
  - 1.3. Object Creator
  - 1.4. Versioning
    - 1.4.1. Revising an Object Series
    - 1.4.2. Revoking an Object Series
    - 1.4.3. New Object or Revision?
    - 1.4.4. Versioning Timestamps
    - 1.4.5. Examples
  - 1.5. Data Markings
    - 1.5.1. Object-Level Markings
    - 1.5.2. Granular Markings
    - 1.5.3. Precedence Rules
    - 1.5.4. Interoperability
    - 1.5.5. Resolving References
    - 1.5.6. Examples
  - 1.6. Common Relationships
  - 1.7. Reserved Properties
- 2. Non-Cyber Top Level Objects
  - 2.1. STIX Package
    - 2.1.1. Properties
    - 2.1.2. Relationships
  - 2.2. Report
    - 2.2.1. Properties
    - 2.2.2. Relationships
    - 2.2.3. 2.2.3.Examples
  - 2.3. Relationship
    - 2.3.1. Properties
  - 2.4. Marking Definition
    - 2.4.1. Examples

Document Development Status

TODO - This section should be removed from the document prior to completion. It is included here to help visually track where things are at in the process.

Each physical documents contains a table that defines 4 levels of development for each TLO and CTI concept. The first level is called Conce . Content coming in to one of the documents starts as a Concept. Once the community starts to work on it it will move to Developn . During this phase, the group will flesh out the design and come up with normative text. As the group comes to general consensus the TLO will move to a Revi phase. During this phase the community can comment and offer suggestions on the normative text and design. After a period of time of no comments or feedback, the TLO will move to its final stage of Draft.

Object / Concept	Status	MVP
`TLO Common Properties`	Review	Yes
`IDs and References`	Review	Yes
`Object Creator`	Review	Yes
`Versioning`	Review	Yes
`Data Marking`	Review	Yes
`Common Relationships`	Review	Yes
Non-Cyber TLOs		
`package`	Developmen	Yes
`report`	Developmen	Yes
`relationship`	Developmen	Yes
`marking-definition`	Review	Yes

1. STIX Top Level Object (TLO)

1.1. TLO Common Properties

	Status: Review **MVP**: Yes

This section defines properties and behaviors common to all STIX Top Level Objects (TLOs).

Property Name	Type	Description
`type` (required)	`string`	The **`type`** field identifies the type of the STIX TLO The value of the **`type`** field **MUST** be one of the types defined by a STIX TLO in this section (i.e., `indicator`).
`id` (required)	`identifier`	Globally identifies this top-level object series.
`created_by_ref` (optional)	`identifier`	The ID of the Identity Object that describes who created this object (`identity--`).
`revision` (required)	`number`	**`revision`** indicates the revision number of this object. This field **MUST** be present in all STIX objects. This field's value **MUST** be greater than or equal to 1 and less than or equal to 999,999,999. Higher revision numbers indicate later versions of the object. Object creators **MUST** increase the revision number (**SHOULD** increment it by exactly 1) when creating a new version of a top-level object.
`revoked` (optional)	`boolean`	**`revoked`** indicates whether this object series has been revoked. If this field is present, the timestamp indicates the timestamp that the objec series was revoked. Revoking a top-level object terminates the complete object series (all version of the object); future revisions of that object are not permitted. When revoking a top-level object, the revision number **MUST** be incremented as above.

`revision_comment` (optional)	`string`	A description of the change that was made by this specific version.
`created_time` (required)	`timestamp`	The time that the object series with this ID was created.
`modified_time` (required)	`timestamp`	The time that this particular version of the object (ID and revision) was created. In other words, the time at which this object series was modified.
`object_marking_refs` (optional)	`array` of type `identifier`	The list of `data-marking` objects to be applied to this object.
`granular_markings` (optional)	`array` of type `granular-marking`	The set of granular markings applied to this object.
`confidence` (reserved)	`RESERVED`	RESERVED FOR FUTURE USE

1.2. IDs and References

	Status: Review **MVP**: Yes

The **`id`** field uniquely identifies a top-level object series. It **MUST** conform to the `identifier` type.

The STIX language makes use of globally unique identifiers as defined by the `identifier` type. The type is also used to define fields that are *references* to other constructs (such as the **`created_by_ref`** field in all TLOs). *Resolving* a reference is the process of identifying and obtaining the actual object referred to by the reference field. References resolve to an object when the value of the reference field (e.g. **`source_ref`**) is an exact match with the **`id`** field of another object. References **MAY** refer to objects that the consumer may not currently have access to.

1.3. Object Creator

	Status: Review **MVP**: Yes

The **`created_by_ref`** field captures the `identifier` of the Identity of the object creator. The object creator is the entity (e.g. system, organization, tool) that generates the `identifier` field for a given object. Entities (proxies) that re-publish a top-level object from another entity, maintaining the original `identifier`, are not considered the object creator. Entities (brokers) that accept objects and republish them with a new identifier are considered the object creator of the new objects.

1.4. Versioning

	Status: Review **MVP**: Yes

This section defines how STIX TLOs are versioned using the **`created_time`**, **`modified_time`**, **`revision`**, **`revision_comment`**, and **`revoked`** fields.

Open Questions:

1. Need to define what a material change is, since there is a MUST with it, we need to make sure the definition is solid. - **We will define non-normative text**
2. Do we need the ability to revoke a specific version of the object? **- NO you can not revoke a single version of the object.**
3. Do we need to better define "entity" in object creator? Is it an organization, a system, something else? The intent was to leave it open to any, is that OK? **- Up to the provider we may add some non-normative text**
4. Is there a better name for the revision field?
5. Do we need any timestamp fields? Which ones? **- ADDED**
6. What are the versioning-specific relationships? Do we need derived-by, suggested-update, etc.? **- Yes, but this is not MVP**

1.4.1. Revising an Object Series

STIX objects, uniquely identified by **`id`** and **`revision`** values, are considered to be immutable (this means that the contents (fields and values) of the object **MUST NOT** be changed). Versioning provides the mechanism to change STIX "object series".

STIX objects with the same **`id`** value and different **`revision`** values are said to be of the same “object series”. Objects with different **`id`** values are said to be of different “object series”. Within a top-level object series, lower **`revision`** values indicate “earlier” revisions, and higher **`revision`** values indicate later revisions.

The **`revision_comment`** field **MAY** be used by the object creator to indicate a description of the change that was made.

1.4.2. Revoking an Object Series

If the `revoked` field is present in a top-level object, that object is said to have been "revoked". If the `revoked` field is not present, the object is said to be "active". Revocation terminates a top-level object series. Once a top-level object has been revoked, additional updates to the object series are not permitted. object creators **MUST NOT** publish new revisions to a top-level object series once that object series has been terminated.

Revoking an object creates a revision of it, therefore the `revision` field **MUST** be increased and the `revision_comment` field may be used to provide a reason for the revokation.

Relationships may have sources or targets to object series' that have been revoked. The consumer may choose to handle those relationships however it wishes.

1.4.3. New Object or Revision?

Eventually, an implementation will encounter a case where a decision must be made regarding whether a change is a new object series or a revision to an existing object series. This is generally considered a data quality problem and therefore this specification does not provide any normative text. However, to assist implementers and promote consistency across implementations, some rules of thumb are provided.

Anytime a change indicates a material change to the meaning of the object (say different malware, different actor) a new object id should be used. The determination of whether a change is a material change is at the object creator's discretion. The creator should also think about references to the object when deciding if a change is material. If the change would invalidate the references to the object, then the change is material, and a new object ID should be used.

1.4.4. Versioning Timestamps

The `created_time` field indicates when a particular object series was first created. Note that, for the purposes of the STIX specifications, creation of an object series within a particular tool or system is irrelevant: this time field simply indicates when the object creator wishes to denote that the STIX representation of the object series was created.

The `modified_time` field indicates when a particular object (a specific revision of an object series) was first created. Note that, for the purposes of the STIX specifications, creation of an object within a particular tool or system is irrelevant: this time field simply indicates when the object creator wishes to denote that the STIX representation of the object was created.

`created_time` **MUST** be either the same or earlier than `created_time` for a given object.

1.4.5. Examples

Example Revision

Given: One object creator has decided that for their content, a change to the list of IP addresses in network indicators does not constitute a material change.

That object creator would consider each change of IP address to be an update to that indicator, and when changing the indicator would update the **`revision`** but not issue a new **`id`**.

Step #	STIX Object	Object Creator Action

1	`{`
	` "type": "example",`
	` "id": "example-1",`
	` "revision": 1,`
	` "title": "attention",`
	` "description": "this is the description"`
	`}`
		Original object created.

2		Object creator changes the title.

3	`{`
	` "type": "example",`
	` "id": "example-1",`
	` "revision": 2,`
	` "title": "Attention!",`
	` "description": "this is the description"`
	`}`
		Object creator increases current object **revision** by 1.

Example of Derived Object

Given: A different object creator has decided that for their content, a change to the list of IP addresses in network indicators does constitute a material change.

That object creator would then issue two objects for each update:

- A revised initial indicator, with the same **`id`**, updated **`revision`**, and the **`revoked`** flag set.
- A new indicator with a new **`id`**

Step #	STIX Object	Object Creator Action

1	`{`
	` "type": "example",`
		Original object created (via new id and set **revision** to *1*).

	`"id": "example-1",`
	`"revision": 1,`
	`"title": "attention",`
	`"description": "this is the description"`
	`}`	

2		Object creator changes the title.

3	`{`
	`"type": "example",`
	`"id": "example-2",`
	`"revision": 1,`
	`"title": "Attention!",`
	`"description": "this is the description"`
	`}`
		Object creator creates a new object (via new **id** and set **revision** to *1*).

Example Recipient Workflow
This section describes an example workflow where a recipient receives multiple updates to a particular object series. The STIX Objects have been truncated for brevity.

Step #	STIX Object	Recipient Action

1	`{`
	`"type": "example",`
	`"id": "example-1",`
	`"revision": 1`
	`}`
		Recipient stores example object because this is the first time the recipient has seen the object.

2	`{`
	`"type": "example",`
	`"id": "example-1",`
	`"revision": 4`
	`}`
		Recipient updates example object because the received revision number is higher than the object that is currently stored.

3	`{`
	`"type": "example",`
	`"id": "example-1",`
	`"revision": 3`
	`}`
		Recipient ignores this object because the recipient already has a newer version of the object.
		Note: recipient might choose to store meta-information about received objects, including revisions that were received out-of-order.

4	`{`
	`"type": "example",`
	`"id": "example-1",`
	`"revision": 12,`
	`"revoked": "2016-03-11T07:23:00Z"`
		Recipient deletes example object, but keeps some metadata regarding the object.

	`}`	
5	`{ "type": "example", "id": "example-1", "revision": 11 }`	Recipient ignores this object because the recipient already has a newer version of the object (the revoked version).

Example Object Creator Workflow

This section describes an example workflow where a object creator publishes multiple updates to a particular object series. The STIX Objects have been truncated for brevity. This scenario assumes a human using a STIX implementation.

Step #	User Action	STIX Object
1	User clicks a create button in the UI, creates a top-level object, then clicks save. This action causes information t be stored in the product's database.	n/a – STIX is not involved in this scenario. (tools *could* choose to create and track STIX revisions for internal changes, bu it is not required by the specification)
2	The user clicks the "share" button, delivering the latest cutting-edge threa intel to multiple social media platforms using STIX.	`{ "type": "example", "id": "example-2", "revision": 1 }`
3	The user performs additional analysis within the STIX implementation, performing multiple modifications and saving their work multiple times.	n/a – STIX is not involved in this scenario. (tools *could* choose to create and track STIX revisions for internal changes, bu it is not required by the specification)
4	The user, happy with the status of thei work, decides to provide an update to the previously published object.	`{ "type": "example", "id": "example-2", "revision": 2 }`
5	The user receives lots of negative feedback regarding the quality of their work and decides to retract the object by pressing the "revoke" button.	`{ "type": "example", "id": "example-2", "revision": 3,`

		`"revoked": "2016-03-11T07:23:00Z"`
		`}`

Example: Broker
A broker is an entity that accepts STIX objects from entities wants to become the object creator (to maintain and update the object), and republishes them to recipients.

It is expected that that there will be CTI brokers, such as an ISAC, that aggregate, validate and maintain a feed of STIX objects. In these cases, the object(s) may originally come from a member of the ISAC, but the ISAC will want to maintain the object(s) and to update the object(s) based upon feedback from other members or research they themselves did, without involving the member that produced it. In these cases, it is expected that the ISAC will take the original STIX object, and reissue the object w/ a new `identifier`. The ISAC is the object creator of this new object, and will be able update the object. If the ISAC simply republishes the original object from the member, any changes made to the object would have to be issued by the same member that published it. This would impact the ability of the ISAC to get up to date information to it's members.

Step #	STIX Object	Broker Action

1	`{`
	`  "type": "example",`
	`  "id": "example-1",`
	`  "revision":1`
	`}`
		Broker receives this object

2	`{`
	`  "type": "example",`
	`  "id": "example-B1",`
	`  "revision": 1`
	`}`
		Broker sends the object, as the object creator, with a new **`id`** and **`revision`** values.

Example: Proxy
In contrast to a broker, a proxy simply passes some or all STIX objects to recipients without modification. An example is an entity that accepts content from a number of object creators and distributes them to all consumers, unmodified.

Note that if a proxy filters the set of objects that might change the picture of the intelligence, and is in effect "updating" the content. But, as they are not updating any of the individual objects, for the purposes of object versioning only they aren't an object updater.

Step #	STIX Object	Proxy Action

1	`{ "type": "example", "id": "example-1", "revision": 1 }`	Proxy receives this object
2	`{ "type": "example", "id": "example-1", "revision": 1 }`	Proxy sends the object, as received.

1.5. Data Markings

	Status: Review **MVP**: Yes

Data markings provide the ability for producers to convey to consumers how they may use and share the marked data that they receive.

The CTI Common specification defines two types of data markings:

- **Object** data markings define how markings are applied to entire packages and top-level objects
- **Granular** data markings define how markings are applied to one or more individual data points within a package or top-level object

1.5.1. Object-Level Markings

Object-level markings are contained in the **`object_marking_refs`** field, which is an array of ID references (of type `identifier`) that resolve to objects of type `marking-definition`. This field **MUST ONLY** be used on packages and top-level objects: when used at the *package* level, the markings referenced in the list apply to the package itself and to each of the top-level objects in the package. When used on a *top-level object*, the markings apply only to that marked object.

1.5.2. Granular Markings

Granular markings are contained in the **`granular_markings`** field, which is an array of `granular-marking` objects. Each of those objects contains a list of content selectors to select what is marked and a list of marking references that refer to the markings to be applied. The consumer must evaluate each `granular-marking` in the list individually. For each marking, the list of **`content_selectors`** must be evaluated to determine the set of content that the markings are applied against. The list of `marking-definition` objects referenced in the **`marking_refs`** field are then applied to the selected content.

1.5.3. Precedence Rules

Markings of the same type may override other markings of that type. Processors **MUST** honor the following order of precedence (items appearing earlier in the list are overridden by those appearing later) when encountering more than one marking application with the same type:

1. Object markings applied at the package level. Within the **`marking_refs`** field, markings appearing later override markings appearing earlier. These are overridden by,
2. Object markings applied at the object level. Within the **`marking_refs`** field, markings appearing later override markings appearing earlier. These are overridden by,
3. Granular markings applied at the package level. Within the **`granular_markings`** and **`content_selectors`** fields, markings appearing later override markings type appearing earlier. These are overridden by,
4. Granular markings applied at the object level. Within the **`granular_markings`** and **`content_selectors`** fields on the object, markings appearing later override markings appearing earlier.

Markings of different types *never* override each other.

1.5.4. Interoperability

Producers **MAY** create any combination of object-level and granular data markings. Producers **MUST** ensure that all markings they create comply with the functional and data marking requirements defined in this document.

A consumer **MAY** support:
- No data markings (known as a **Level 0 Marking Consumer**)
- **Object-level** data markings (known as a **Level 1 Marking Consumer**)
- Both **object-level** and **granular** data markings (known as a **Level 2 Marking Consumer**)

A **Level 0 Markings Processor** is not able to process data markings. A **Level 1 Markings Processor** is only able to process object-level data markings. A **Level 2 Markings Processor** is able to process both object-level markings and granular markings.

Level 0:
- A Level 0 processor who receives a package that contains either the **`object_marking_refs`** field or the **`granular_markings`** field at the package level **MUST** reject the entire package.

- A Level 0 processor who receives a package that **DOES NOT** contain the **`object_marking_refs`** field or the **`granular_markings`** field at the package level **MAY** accept the package.
- A Level 0 processor processing an object that contains the **`object_marking_refs`** field or the **`granular_markings`** field **MUST** reject the object. It **MAY** continue to process the rest of the document.

Level 1:

- A Level 1 processor who receives a package that contains the **`granular_markings`** field at the package level **MUST** reject the document.
- A Level 1 processor who receives a document that DOES NOT contain the **`granular_markings`** field at the package level **MAY** accept the document.
- A Level 1 processor processing a document that contains an object with the **`granular_markings`** field **MUST** reject the object. It **MAY** continue to process the rest of the document.
- A Level 1 processor **MUST** process all object-level markings applied in the **`object_marking_refs`** field per the mechanisms outlined in this specification.

Level 2:

- **2.1**: A Level 2 processor **MUST** process all object-level markings applied in the **`object_marking_refs`** field and granular markings applied in the **`granular_markings`** field per the mechanisms outlined in this specification.

1.5.5. Resolving References

- Level 1 and Level 2 processors that are unable to resolve a reference to a marking definition **MUST** reject content marked by that definition.

1.5.6. Examples

```
{
  "type": "indicator",
  "id": "indicator--089a6ecb-cc15-43cc-9494-767639779235",
  "object_marking_refs": ["marking-definition--a82a2ecb-2c1a-42cn-9494-772156779431"],
  "granular_markings": [
    {
      "content_selectors": ["$.description"],
      "marking_refs": ["marking-definition--089a6ecb-cc15-43cc-9494-767639779123"]
  ],
  "description": ["Some description"]
}

{
  "type": "indicator",
  "id": "indicator--089a6ecb-cc15-43cc-9494-767639779235",
  "object_marking_refs": ["marking-definition--089a6ecb-cc15-43cc-9494-767639779123"],
}
```

1.6. Common Relationships

	Status: Review **MVP**: Yes

The following descriptions of relationships are defined for all TLOs.

Kind of Relationship	Source	Target	Description
`duplicate-of`	*`<Object>`*	*`<Object of same type>`*	Allows recording that two different Objects of the same type are duplicates of each other.
`other`	*`<Object>`*	`*`	The catch-all generic relationship type that allows description of relationships between an Object and any other Domain Object. It is essentially a fallback to allow a relationship to be defined when none of the others fit.

1.7. Reserved Properties

This section defines property names that are reserved for future use in revisions of this document. The property names defined in this section **MUST NOT** be used for the name of any Extension Field.

These properties are:
`confidence`
`severity`

Open Questions:
- How should this be defined? What's the scale?
- Should this be required or optional (may be different in different locations)?
- Is it a controlled vocab with _ext fields or just an enum

2. Non-Cyber Top Level Objects

2.1. STIX Package

Type Name: package	**Status:** Developmen **MVP**: Yes

A collection of TLOs used specifically for transport.
< to do, please add descriptions >

2.1.1. Properties

STIX TLO Common Properties		
type, id, created_by_ref, created_time, revision, modified_time, revoked, revision_comment, confidence, object_markings_refs, granular_markings		
Property Name	**Type**	**Description**
type (required)	string	Indicates that this object is a STIX Package. The value of this field **MUST** be package
spec_version (required)	string	The release version of the language specification used to represent this construct
attack_patterns (optional)	array of type attack-pattern	Specifies a set of one or more Attack_Pattern TTPs.
campaigns (optional)	array of type campaign	Specifies a set of one or more Campaigns.
configurations (optional)	array of type configuration	Specifies a set of one or more Configuration exploit targets.
courses_of_action (optional)	array of type course-of-action	Specifies a set of one or more Courses of Action that could be taken in regard to one of more cyber threats.

exploits (optional)	`array` of type `exploit`	Specifies a set of one or more Exploit TTPs.
identities (optional)	`array` of type `identity`	Specifies a set of one or more identities of individuals or organizations.
incidents (optional)	`array` of type `incident`	Specifies a set of one or more cyber threat Incidents.
indicators (optional)	`array` of type `indicator`	Specifies a set of one or more cyber threat Indicators.
kill_chains (optional)	`array` of type `kill-chain`	Specifies a set of one or more Kill Chains.
kill_chain_phases (optional)	`array` of type `kill-chain-phase`	Specifies a set of one or more Kill Chain Phases.
malicious_ infrastructures (optional)	`array` of type `malicious-infrastructure`	Specifies a set of one or more Infrastructure TTPs.
malicious_tools (optional)	`array` of type `malicious-tool`	Specifies a set of one or more Malicious Tool TTPs.
malware (optional)	`array` of type `malware`	Specifies a set of one or more Malware TTPs.
marking_definitions (optional)	`array` of type `marking-definition`	Specifies a set of one or more Marking Definitions.
observations (optional)	`array` of type `observation`	Specifies a set of one or more cyber observations.
personas (optional)	`array` of type `persona`	Specifies a set of one or more Personas.
external-references (optional)	`array` of type `external-reference`	Specifies a set of one or more references to a non-STIX object
relationships (optional)	`array` of type `relationship`	Specifies a set of one or more relationships between top-level objects (TLOs).
reports (optional)	`array` of type `report`	Specifies a set of one or more reports.
sightings (optional)	`array` of type	Specifies a set of one or more

	sighting	sightings.
threat_actors (optional)	array of type threat-actor	Specifies a set of one or more Threat Actors.
tools (optional)	array of type tool	<add text>
victim_targetings (optional)	array of type victim-targeting	Specifies a set of one or more Victim Targeting TTPs.
vulnerabilities (optional)	array of type vulnerability	Specifies a set of one or more Vulnerability exploit targets.
weaknesses (optional)	array of type weakness	Specifies a set of one or more Weakness exploit targets.

2.1.2. Relationships

NONE

2.2. Report

Type Name: report	**Status:** Developmen **MVP**: Yes

A grouping of asserted related TLOs for purposes of reporting. For example, a threat report by an intel provider discussing the techniques used by a threat actor would be represented with this TLO.

2.2.1. Properties

STIX TLO Common Properties		
type, id, created_by_ref, created_time, revision, modified_time, revoked, revision_comment, confidence, object_markings_refs, granular_markings		
Property Name	**Type**	**Description**
type (required)	string	(Overrides cti-core) The value of this field **MUST** be report
title (required)	string	A human readable title
description (optional)	string	A human readable description

labels (required)	array of type report-label-cv	Specifies the intended purposes or uses of this Report.
labels_ext (optional)	array of type vocab-ext	Specifies alternate intended purposes or uses of this Report.
report_contains_refs (required)	array of type identifier	Specifies the objects that are in this Report.

2.2.2. Relationships

There are no uninherited default relationships defined between the Report Object and other objects.

Inherited From	Inherited Kinds of Relationships
stix-core	duplicate-of, other

2.2.3. 2.2.3.Examples

```
// Just a report, where the consumer may or may not already have access to the TLOs
{
  "type": "report",
  "id": "report--84e4d88f-44ea-4bcd-bbf3-b2c1c320bcb3",
  "created_time": "2015-12-21T19:59:11Z",
  "created_by_ref": "identity--a463ffb3-1bd9-4d94-b02d-74e4f1658283",
  "title": "The Black Vine Cyberespionage Group",
  "descriptions": ["A simple report with an indicator and campaign"],
  "labels": ["Threat Report"],
  "report_contains_refs": [
    "indicator--26ffb872-1dd9-446e-b6f5-d58527e5b5d2",
    "campaign--83422c77-904c-4dc1-aff5-5c38f3a2c55c"
  ]
}


// A full package with a report and the TLOs / Relationships that are part of the report
{
  "type": "package",
  "id": "package--44af6c39-c09b-49c5-9de2-394224b04982",

  "identities": [
    {
      "type": "identity",
      "id": "identity--a463ffb3-1bd9-4d94-b02d-74e4f1658283",
      "name": "Symantec",
    }
  ],

   "reports": [
    {
      "type": "report",
      "id": "report--84e4d88f-44ea-4bcd-bbf3-b2c1c320bcbd",
      "created_time": "2015-12-21T19:59:11Z",
      "created_by_ref": "identity--a463ffb3-1bd9-4d94-b02d-74e4f1658283",
      "title": "The Black Vine Cyberespionage Group",
      "descriptions": ["A simple report with an indicator and campaign"],
      "labels": ["Threat Report"],
      "report_contains_refs": [
        "indicator--26ffb872-1dd9-446e-b6f5-d58527e5b5d2",
        "campaign--83422c77-904c-4dc1-aff5-5c38f3a2c55c"
      ]
    }
  ],

  "indicators": [
    {
      "type": "indicator",
      "id": "indicator--26ffb872-1dd9-446e-b6f5-d58527e5b5d2",
      "created_time": "2015-12-21T19:59:17Z",
      "title": "Some indicator",
      "indicator_types": ["IP Watchlist"],
      "created_by_ref": "identity--a463ffb3-1bd9-4d94-b02d-74e4f1658283"
    }
  ],

  "campaigns": [
    {
      "type": "campaign",
      "id": "campaign--83422c77-904c-4dc1-aff5-5c38f3a2c55c",
      "created_time": "2015-12-21T19:59:17Z",
      "title": "Some Campaign",
      "created_by_ref": "identity--a463ffb3-1bd9-4d94-b02d-74e4f1658283"
    }
  ],

  “Relationships”: [
    {
       "id": "relationship--f82356ae-fe6c-437c-9c24-6b64314ae68a",
       "type": "relationship",
       "created_at": "2015-12-21T19:59:17.000000+00:00",
       "from": "indicator--26ffb872-1dd9-446e-b6f5-d58527e5b5d2",
       "to": "campaign--26ffb872-1dd9-446e-b6f5-d58527e5b5d2",
       "kind_of_": "Related Campaign",
        "created_by_ref": "identity--a463ffb3-1bd9-4d94-b02d-74e4f1658283"
    },
  ]
}
```

2.3. Relationship

Type Name: `relationship`	**Status:** Developmen **MVP**: Yes

The Relationship object is used to link together other top-level objects, such as Indicator, Observation, Threat Actor and the like. If other top-level objects are considered “nodes” in the graph, the relationship object represents “edges”.

Open Questions:

1. Need to discuss the values of the kind_of_relationship vocab

2.3.1. Properties

STIX TLO Common Properties		
`type, id, created_by_ref, created_time, revision, modified_time, revoked, revision_comment, confidence, object_markings_refs, granular_markings`		
Property Name	**Type**	**Description**
`type` (required)	`string`	(Overrides `cti-core`) The value of this field **MUST** be `relationship`
`title` (optional)	`string`	A human readable title
`description` (optional)	`string`	A human readable description
`kind_of_relationship` (required)	`string`	The descriptor for this relationship.
`source_ref` (required)	`string`	The ID of the source (from) object.
`target_ref` (required)	`string`	The ID of the target (to) object.
`is_directional` (required)	`boolean`	Indicates whether the relationship is directional. For values in the standard vocabulary, this attribute may be

		hardcoded to a particular value.

2.4. Marking Definition

Type Name: marking-definition	**Status:** Review **MVP**: Yes

An abstract type that represents how something is marked. Individual types of marking approaches (e.g. TLP) extend marking-definition to define their markings. Marking definitions are applied by the fields on stix-core.

STIX TLO Common Properties		
type, id, created_by_ref, created_time, revision, modified_time, revoked, revision_comment, confidence, object_markings_refs, granular_markings		
Property Name	**Type**	**Description**
type (required)	string	(Overrides cti-core) The value of this field **MUST** be marking-definition
definition_type (required)	string	
definition (required)	object	The type of marking this represents.
(other fields)	Various	Used to represent the marking itself. This contains other fields as needed to represent the marking data.

2.4.1. Examples

```
{
  "type": "marking-definition",
  "id": "marking-definition--089a6ecb-cc15-43cc-9494-767639779123",
  "created_time": "2016-02-19T09:11:01Z",
  "definition_type": "tlp",
  "definition": {
    "tlp": "GREEN"
  }
}
```

```
{
  "type": "marking-definition",
  "id": "marking-definition--089a6ecb-cc15-43cc-9494-767639779124",
  "created_time": "2016-02-19T09:11:01Z",
  "defintion_type": "isa",
  "definition": {
    "classification": "UNCLASSIFIED",
    "caveats": []
  }
}
```