

MILE Working Group  
Internet-Draft  
Intended status: Informational  
Expires: February 16, 2014

J. Field  
Pivotal  
August 15, 2013

Resource-Oriented Lightweight Indicator Exchange  
draft-field-mile-rolie-02.txt

Abstract

This document defines a resource-oriented approach to cyber security information sharing. Using this approach, a CSIRT or other stakeholder may share and exchange representations of cyber security incidents, indicators, and other related information as Web-addressable resources. The transport protocol binding is specified as HTTP(S) with a MIME media type of Atom+XML. An appropriate set of link relation types specific to cyber security information sharing is defined. The resource representations leverage the existing IODEF [RFC5070] and RID [RFC6545] specifications as appropriate. Coexistence with deployments that conform to existing specifications including RID [RFC6545] and Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS [RFC6546] is supported via appropriate use of HTTP status codes.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 16, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	3
3. Background and Motivation . . . . .	4
3.1. Message-oriented versus Resource-oriented Architecture . . . . .	5
3.1.1. Message-oriented Architecture . . . . .	5
3.1.2. Resource-Oriented Architecture . . . . .	5
3.2. Authentication of Users . . . . .	7
3.3. Authorization Policy Enforcement . . . . .	7
3.3.1. Enforcement at Destination System . . . . .	7
3.3.2. Enforcement at Source System . . . . .	8
4. RESTful Usage Model . . . . .	9
4.1. Dynamic Service Discovery versus Static URL Template . . . . .	10
4.2. Non-Normative Examples . . . . .	11
4.2.1. Service Discovery . . . . .	11
4.2.2. Feed Retrieval . . . . .	14
4.2.3. Entry Retrieval . . . . .	16
4.2.4. Use of Link Relations . . . . .	19
5. Requirements for RESTful (Atom+xml) Binding . . . . .	29
5.1. Transport Layer Security . . . . .	29
5.2. User Authentication . . . . .	29
5.3. User Authorization . . . . .	30
5.4. Content Model . . . . .	30
5.5. HTTP methods . . . . .	31
5.6. Service Discovery . . . . .	31
5.6.1. Workspaces . . . . .	32
5.6.2. Collections . . . . .	32
5.6.3. Service Document Security . . . . .	32
5.7. Category Mapping . . . . .	32
5.7.1. Collection Category . . . . .	32
5.7.2. Entry Category . . . . .	33
5.8. Entry ID . . . . .	33
5.9. Entry Content . . . . .	33
5.10. Link Relations . . . . .	33
5.10.1. Additional Link Relation Requirements . . . . .	35
5.11. Member Entry Forward Security . . . . .	36
5.12. Date Mapping . . . . .	36

5.13. Search . . . . .	37
5.14. / (forward slash) Resource URL . . . . .	37
6. Security Considerations . . . . .	38
7. IANA Considerations . . . . .	40
8. ToDo and Open Issues . . . . .	40
9. Acknowledgements . . . . .	41
10. References . . . . .	41
10.1. Normative References . . . . .	41
10.2. Informative References . . . . .	42
Appendix A. Change Tracking . . . . .	43
Appendix B. Resource Authorization Model . . . . .	43
B.1. Example XACML Profile . . . . .	44
Author's Address . . . . .	44

## 1. Introduction

This document defines a resource-oriented approach to cyber security information sharing that follows the REST (Architectural Styles and the Design of Network-based Software Architectures) architectural style. The resource representations leverage the existing IODEF [RFC5070] and RID [RFC6545] specifications as appropriate. The transport protocol binding is specified as HTTP(S) with a media type of Atom+XML. An appropriate set of link relation types specific to cyber security information sharing is defined. Using this approach, a CSIRT or other stakeholder may exchange cyber security incident and/or indicator information as Web-addressable resources.

The goal of this specification is to define a loosely-coupled, agile approach to cyber security situational awareness. This approach has architectural advantages for some use case scenarios, such as when a CSIRT or other stakeholder is required to share cyber security information broadly (e.g., at internet scale), or when an information sharing consortium requires support for asymmetric interactions amongst their stakeholders.

Coexistence with deployments that conform to existing specifications including RID [RFC6545] and Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS [RFC6546] is supported via appropriate use of HTTP status codes.

## 2. 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]. Definitions for some of the common computer security-related terminology used in this document can be found in Section 2 of [RFC5070].

### 3. Background and Motivation

It is well known that Internet security threats are evolving ever more rapidly, and are becoming ever more sophisticated than before. The threat actors are frequently distributed and are not constrained to operating within a fixed, closed consortium. The technical skills needed to perform effective analysis of a security incident, or to even recognize an indicator of compromise are already specialized and relatively scarce. As threats continue to evolve, even an established network of CSIRT may find that it does not always have all of the skills and knowledge required to immediately identify and respond to every new incident. Effective identification of and response to a sophisticated, multi-stage attack frequently depends upon cooperation and collaboration, not only amongst the defending CSIRTs, but also amongst other stakeholders, including, potentially, individual end users.

Existing approaches to cyber security information sharing are based upon message exchange patterns that are point-to-point, and event-driven. Sometimes, information that may be useful to, and sharable with multiple peers is only made available to peers after they have specifically requested it. Unfortunately, a sharing peer may not know, a priori, what information to request from another peer. Sending unsolicited RID reports does provide a mechanism for alerting, however these reports are again sent point-to-point, and must be reviewed for relevance and then prioritized for action by the recipient. Thus, distribution of some relevant incident and indicator information may exhibit significant latency.

In order to appropriately combat the evolving threats, the defending CSIRTs should be enabled to operate in a more agile manner, sharing selected cyber security information proactively, if and as appropriate.

For example, a CSIRT analyst would benefit by having the ability to search a comprehensive collection of indicators that has been published by a government agency, or by another member of a sharing consortium. The representation of each indicator may include links to the related resources, enabling an appropriately authenticated and authorized analyst to freely navigate the information space of indicators, incidents, and other cyber security domain concepts, as needed. In general, a more Web-centric sharing approach will enable a more dynamic and agile collaboration amongst a broader, and varying constituency.

The following sections discuss additional specific technical issues that motivate the development of an alternative approach.

### 3.1. Message-oriented versus Resource-oriented Architecture

The existing approaches to cyber security information sharing are based upon message-oriented interactions. The following paragraphs explore some of the architectural constraints associated with message-oriented interactions and consider the relative merits of an alternative model based on a Resource-oriented architecture for use in some use case scenarios.

#### 3.1.1. Message-oriented Architecture

In general, message-based integration architectures may be based upon either an RPC-style or a document-style binding. The message types defined by RID represent an example of an RPC-style request. This approach imposes implied requirements for conversational state management on both of the communicating RID endpoint(s). Experience has shown that this state management frequently becomes the limiting factor with respect to the runtime scalability of an RPC-style architecture.

In addition, the practical scalability of a peer-to-peer message-based approach will be limited by the administrative procedures required to manage  $O(N^2)$  trust relationships and at least  $O(N)$  policy groups.

As long as the number of CSIRTs participating in an information sharing consortium is limited to a relatively smaller number of nodes (i.e.,  $O(2^N)$ , where  $N < 5$ ), these scalability constraints may not represent a critical concern. However, when there is a requirement to support a significantly larger number of participating peers, a different architectural approach will be required. One alternative to the message-based approach that has demonstrated scalability is the REST [REST] architectural style.

#### 3.1.2. Resource-Oriented Architecture

Applying the REST architectural style to the problem domain of cyber security information sharing would take the approach of exposing incidents, indicators, and any other relevant types as simple Web-addressable resources. By using this approach, a CSIRT or other organization can more quickly and easily share relevant incident and indicator information with a much larger and potentially more diverse constituency. A client may leverage virtually any available HTTP user agent in order to make requests of the service provider. This improved ease of use could enable more rapid adoption and broader participation, thereby improving security for everyone.

A key interoperability aspect of any RESTful Web service will be the choices regarding the available resource representations. For example, clients may request that a given resource representation be returned as either XML or JSON. In order to enable back-compatibility and interoperability with existing CSIRT implementations, IODEF [RFC5070] is specified for this transport binding as a mandatory to implement (MTI) data representation for incident and indicator resources. In addition to the REQUIRED representation, an implementation MAY support additional representations if and as needed such as IODEF extensions, the RID schema, or other schemas. For example, an implementation may choose to provide support for returning a JSON representation of an incident resource.

Finally, an important principle of the REST architectural style is the use of hypertext links as the embodiment of application state (HATEOAS). Rather than the server maintaining conversational state for each client context, the server will instead include a suitable set of hyperlinks in the resource representation that is returned to the client. In this way, the server remains stateless with respect to a series of client requests. The included hyperlinks provide the client with a specific set of permitted state transitions. Using these links the client may perform an operation, such as updating or deleting the resource representation. The client may also be provided with hypertext links that can be used to navigate to any related resource. For example, the resource representation for an incident object may contain links to the related indicator resource(s).

This document specifies the use of Atom Syndication Format [RFC4287] and Atom Publishing Protocol [RFC5023] as the mechanism for representing the required hypertext links.

#### 3.1.2.1. A Resource-Oriented Use Case: "Mashup"

In this section we consider a non-normative example use case scenario for creating a cyber security "mashup".

Any CSIRT can enable any authenticated and authorized client that is a member of the sharing community to quickly and easily navigate through any of the cyber security information that that provider is willing to share. An authenticated and authorized analyst may then make HTTP(S) requests to collect incident and indicator information known at one CSIRT with threat actor data being made available from another CSIRT. The resulting correlations may yield new insights that enable a more timely and effective defensive response. Of course, this report may, in turn, be made available to others as a new Web-addressable resource, reachable via another URL. By

employing the RESTful Web service approach the effectiveness of the collaboration amongst a consortium of CSIRTs and their stakeholders can be greatly improved.

### 3.2. Authentication of Users

In the store-and-forward, message-based model for information sharing client authentication is provided via a Public Key Infrastructure (PKI) -based trust and mutually authenticated TLS between the messaging system endpoints. There is no provision to support authentication of a client by another means. As a result, participation in the sharing community is limited to those organizations that have sufficient resources and capabilities to manage a PKI.

A CSIRT may apply XML Security to the content of a message, however the contact information provided within the message body represents a self-asserted identity, and there is no guarantee that the contact information will be recognized by the peer. As a result, the audit trail and the granularity of any authorization policies is limited to the identity of the peer CSIRT organization.

A CSIRT implementing this specification MUST implement server-authenticated TLS. The CSIRT may choose to authenticate its client users via any suitable authentication scheme that can be implemented via HTTP(S). A participating CSIRT MAY choose to support more than one authentication method. Support for use of a Federated Identity approach is RECOMMENDED. Establishing a specific end user identity prior to processing a request is RECOMMENDED. Doing so will enable the source system to maintain a more complete audit trail of exactly what cyber security incident and indicator information has been shared, when, and with whom.

### 3.3. Authorization Policy Enforcement

A key aspect of any cyber security information sharing arrangement is assigning the responsibility for authorization policy enforcement. The authorization policy must be enforced either at the destination system, or the source system, or both. The following sections discuss these alternatives in greater detail.

#### 3.3.1. Enforcement at Destination System

The store-and-forward, message-based approach to cyber security information sharing requires that the origin system delegate authorization policy enforcement to the destination system. The origin system may leverage XML Encryption and DigitalSignature to protect the message content. In addition, the origin system assigns

a number of policy-related attribute values, including a "restriction" attribute, before the message is sent. These labels indicate the sender's expectation for confidentiality enforcement and appropriate handling at the destination. Section 9.1 of RFC6545 provides specific guidance to implementers on use of the XML security standards in order to achieve the required levels of security for the exchange of incident information.

Once the message has been received at the destination system, the XML encryption and digital signature protections on the message will be processed, and based upon the pre-established PKI-based trust relationships, the message content is validated and decrypted. Typical implementations will then pass the cleartext data to an internal Incident Handling System (IHS) for further review and/or action by a human operator or analyst. Regardless of where in the deployment architecture the XML message-level security is being handled, eventually the message content will be made available as cleartext for handling by human systems analysts and other operational staff.

The authorization policy enforcement of the message contents must then be provided by the destination IHS. It is the responsibility of the destination system to honor the intent of the policy restriction labels assigned by the origin system. Ideally, these policy labels would serve as part of a distributed Mandatory Access Control scheme. However, in practice a typical IHS will employ a Discretionary Access Control (DAC) model rather than a MAC model and so the policy related attributes are defined to represent handling "hints" and provide no guarantee of enforcement at the destination.

As a result, ensuring that the destination system or counterparty will in fact correctly enforce the intended authorization policies becomes a key issue when entering into any information sharing agreements. The origin CSIRT must accept a non-zero risk of information leakage, and therefore must rely upon legal recourse as a compensating control. Establishing such legal sharing agreements can be a slow and difficult process, as it assumes a high level of trust in the peer, with respect to both intent and also technical capabilities.

### 3.3.2. Enforcement at Source System

In this model, the required authorization policy enforcements are implemented entirely within the source system. Enforcing the required authorization policy controls at the source system eliminates the risk of subsequent information leakage at the destination system due to inadequate or incomplete implementation of the expected controls. The destination system is not expected to



perform any additional authorization enforcements. Authorization enforcement at the source system may be based on, e.g. Role-based Access Controls applied in the context of an established user identity. The source system may use any appropriate authentication mechanism in order to determine the user identity of the requestor, including, e.g. federated identity. An analyst or operator at a CSIRT may request specific information on a given incident or indicator from a peer CSIRT, and the source system will return a suitable representation of that resource based upon the specific role of the requestor. A different authenticated user (perhaps from the same destination CSIRT) may receive a different representation of the same resource, based upon the source system applying suitable Role-based Access Control policy enforcements for the second user identity.

Consistent with HTTP [RFC2616] a user's request MAY be denied with a resulting HTTP status code value of 4xx such as 401 Unauthorized, 403 Forbidden, or 404 Not Found, or 405 Method Not Allowed, if and as appropriate.

#### 4. RESTful Usage Model

This section describes the basic use of Atom Syndication Format [RFC4287] and Atom Publishing Protocol [RFC5023] as a RESTful transport binding and dynamic discovery protocol, respectively, for cyber security information sharing.

As described in Atom Publishing Protocol [RFC5023], an Atom Service Document is an XML-based document format that allows a client to dynamically discover the collections provided by a publisher.

As described in Atom Syndication Format [RFC4287], Atom is an XML-based document format that describes lists of related information items known as collections, or "feeds". Each feed document contains a collection of zero or more related information items called "member entries" or "entries".

When applied to the problem domain of cyber security information sharing, an Atom feed may be used to represent any meaningful collection of information resources such as a set of incidents, or indicators. Each entry in a feed could then represent an individual incident, or indicator, or some other resource, as appropriate. Additional feeds could be used to represent other meaningful and useful collections of cyber security resources. A feed may be categorized, and any feed may contain information from zero or more categories. The naming scheme and the semantic meaning of the terms used to identify an Atom category are application-defined.

#### 4.1. Dynamic Service Discovery versus Static URL Template

In order to specify a protocol for cyber security information sharing using the REST architectural style it is necessary to define the set of resources to be modeled, and how these resources are related. Based on this interface contract, clients will then interact with the REST service by navigating the modeled entities, and their relationships. The interface contract between the client and the server may either be statically bound or dynamically bound.

In the statically bound case, the clients have a priori knowledge of the resources that are supported. In the REST architectural style this static interface contract takes the form of a URL template. This approach is not appropriate for the cyber security information sharing domain for at least two reasons.

First, there is no standard for a cyber security domain model. While information security practitioners can generally agree on some of the basic concepts that are important to modeling the cyber security domain -- such as "indicator," "incident," or "attacker," -- there is no single domain model that can be referenced as the basis for specifying a standardized RESTful URI Template. Second, the use of static URL templates creates a tighter coupling between the client implementation and the server implementation. Security threats on the internet are evolving ever more rapidly, and it will never be possible to establish a statically defined resource model and URL Template. Even if there were an initial agreement on an appropriate URL template, it would eventually need to change. If and when a CSIRT finds that it needs to change the URL template, then any existing deployed clients would need to be upgraded.

Thus, rather than attempting to define a fixed set of resources via a URI Template, this document has instead specified an approach based on dynamic discovery of resources via an Atom Publishing Protocol Service Document. By using this approach, it is possible to standardize the RESTful usage model, without needing to standardize on the definitions of specific, strongly-typed resources. A client can dynamically discover what resources are provided by a given CSIRT, and then navigate that domain model accordingly. A specific server implementation may still embody a particular URL template, however the client does not need a priori knowledge of the format of the links, and the URL itself is effectively opaque to the client. Clients are not bound to any particular server's interface.

The following paragraphs provide a number of non-normative examples to illustrate the use of Atom Publishing Protocol for basic cyber security information sharing service discovery, as well as the use of Atom Syndication Format as a mechanism to publish cyber security information feeds.

Normative requirements are defined below, in Section 5.

## 4.2. Non-Normative Examples

### 4.2.1. Service Discovery

This section provides a non-normative example of a client doing service discovery.

An Atom service document enables a client to dynamically discover what feeds a particular publisher makes available. Thus, a CSIRT may use an Atom service document to enable clients of the CSIRT to determine what specific cyber security information the CSIRT makes available to the community. The service document could be made available at any well known location, such as via a link from the CSIRT's home page. One common technique is to include a link in the <HEAD> section of the organization's home page, as shown below:

Example of bootstrapping Service Document discovery:

```
<link rel="introspection" type="application/atomsvc+xml" title="Atom P  
ublishing Protocol Service Document" href="/csirt/svcdoc.xml" />
```

A client may then format an HTTP GET request to retrieve the service document:

```
GET /csirt/svcdoc.xml  
Host: www.example.org  
Accept: application/atomsvc+xml
```

Notice the use of the HTTP Accept: request header, indicating the MIME type for Atom service discovery. The response to this GET request will be an XML document that contains information on the specific feed collections that are provided by the CSIRT.

Example HTTP GET response:

```
HTTP/1.1 200 OK
Date: Fri, 24 Aug 2012 17:09:11 GMT
Content-Length: 570
Content-Type: application/atomsvc+xml;charset="utf-8"

<?xml version="1.0" encoding="UTF-8"?>
<service xmlns="http://www.w3.org/2007/app"
  xmlns:atom="http://www.w3.org/2005/Atom">
  <workspace xml:lang="en-US" xmlns:xml="http://www.w3.org/XML/1998/n
amespace">
    <atom:title type="text">Incidents</atom:title>
    <collection href="http://example.org/csirt/incidents">
      <atom:title type="text">Incidents Feed</atom:title>
      <accept>application/atom+xml; type=entry</accept>
    </collection>
  </workspace>
</service>
```

This simple Service Document example shows that this CSIRT provides one workspace, named "Incidents." Within that workspace, the CSIRT makes one feed collection available. When attempting to GET or POST entries to that feed collection, the client must indicate a content type of application/atom+xml.

A CSIRT may also offer a number of different feeds, each containing different types of cyber security information. In the following example, the feeds have been categorized. This categorization will help the clients to decide which feeds will meet their needs.

HTTP/1.1 200 OK  
 Date: Fri, 24 Aug 2012 17:10:11 GMT  
 Content-Length: 1912  
 Content-Type: application/atomsvc+xml; charset="utf-8"

```
<?xml version="1.0" encoding='utf-8'?>
  <service xmlns="http://www.w3.org/2007/app"
    xmlns:atom="http://www.w3.org/2005/Atom">
    <workspace>
      <atom:title>Cyber Security Information Sharing</atom:title>
      <collection href="http://example.org/csirt/public/indicators" >
        <atom:title>Public Indicators</atom:title>
        <categories fixed="yes">
          <atom:category scheme="http://example.org/csirt/restriction
" term="public" />
          <atom:category scheme="http://example.org/csirt/purpose" te
rm="reporting" />
        </categories>
        <accept>application/atom+xml; type=entry</accept>
      </collection>
      <collection href="http://example.org/csirt/public/incidents" >
        <atom:title>Public Incidents</atom:title>
        <categories fixed="yes">
          <atom:category scheme="http://example.org/csirt/restriction
" term="public" />
          <atom:category scheme="http://example.org/csirt/purpose" te
rm="reporting" />
        </categories>
        <accept>application/atom+xml; type=entry</accept>
      </collection>
    </workspace>
    <workspace>
      <atom:title>Private Consortium Sharing</atom:title>
      <collection href="http://example.org/csirt/private/incidents" >
        <atom:title>Incidents</atom:title>
        <accept>application/atom+xml; type=entry</accept>
        <categories fixed="yes">
          <atom:category scheme="http://example.org/csirt/purpose" te
rm="traceback, mitigation, reporting" />
          <atom:category scheme="http://example.org/csirt/restriction
" term="private, need-to-know" />
        </categories>
      </collection>
    </workspace>
  </service>
```

In this example, the CSIRT is providing a total of three feed collections, organized into two different workspaces. The first workspace contains two feeds, consisting of publicly available indicators and publicly available incidents, respectively. The second workspace provides one additional feed, for use by a sharing consortium. The feed contains incident information containing entries related to three purposes: traceback, mitigation, and

reporting. The entries in this feed are categorized with a restriction of either "Need-to-Know" or "private". An appropriately authenticated and authorized client may then proceed to make GET requests for one or more of these feeds. The publicly provided incident information may be accessible with or without authentication. However, users accessing the feed targeted to the private sharing consortium would be expected to authenticate, and appropriate authorization policies would subsequently be enforced by the feed provider.

#### 4.2.2. Feed Retrieval

This section provides a non-normative example of a client retrieving an incident feed.

Having discovered the available cyber security information sharing feeds, an authenticated and authorized client who is a member of the private sharing consortium may be interested in receiving the feed of known incidents. The client may retrieve this feed by performing an HTTP GET operation on the indicated URL.

Example HTTP GET request for a Feed:

```
GET /csirt/private/incidents
Host: www.example.org
Accept: application/atom+xml
```

The corresponding HTTP response would be an XML document containing the incidents feed:

Example HTTP GET response for a Feed:

```

HTTP/1.1 200 OK
Date: Fri, 24 Aug 2012 17:20:11 GMT
Content-Length: 2882
Content-Type: application/atom+xml;type=feed;charset="utf-8"

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/Atom file:/C:/schemas/at
om.xsd
                        urn:ietf:params:xml:ns:iodef-1.0 file:/C:/schem
as/iodef-1.0.xsd"
      xml:lang="en-US">
  <generator version="1.0" xml:lang="en-US">emc-csirt-iodef-feed-serv
ice</generator>
  <id xml:lang="en-US">http://www.example.org/csirt/private/incidents
</id>
  <title type="text" xml:lang="en-US">Atom formatted representation o
f a feed of IODEF documents</title>
  <updated xml:lang="en-US">2012-05-04T18:13:51.0Z</updated>
  <author>
    <email>csirt@example.org</email>
    <name>EMC CSIRT</name>
  </author>

  <!-- By convention there is usually a self link for the feed -->
  <link href="http://www.example.org/csirt/private/incidents" rel="se
lf"/>

  <entry>
    <id>http://www.example.org/csirt/private/incidents/123456</id>
    <title>Sample Incident</title>
    <link href="http://www.example.org/csirt/private/incidents/1234
56" rel="self"/>
    <!-- by convention -->
    <link href="http://www.example.org/csirt/private/incidents/1234
56" rel="alternate"/>
    <!-- required by Atom spec -->
    <published>2012-08-04T18:13:51.0Z</published>
    <updated>2012-08-05T18:13:51.0Z</updated>
    <!-- The category is based upon IODEF purpose and restriction a
ttributes -->
    <category term="traceback" scheme="purpose" label="trace back"
/>
    <category term="need-to-know" scheme="restriction" label="need
to know" />
    <summary>A short description of this incident, extracted from t
he IODEF Incident class, <description> element. </summary>
  </entry>

  <entry>
    <!-- ...another entry... -->
  </entry>

</feed>

```

This feed document has two atom entries, one of which has been elided. The completed entry illustrates an Atom <entry> element that provides a summary of essential details about one particular





incident. Based upon this summary information and the provided category information, a client may choose to do an HTTP GET operation to retrieve the full details of the incident. This example provides a RESTful alternative to the RID investigation request message, as described in sections 6.1 and 7.2 of RFC6545.

#### 4.2.3. Entry Retrieval

This section provides a non-normative example of a client retrieving an incident as an Atom entry.

Having retrieved the feed of interest, the client may then decide based on the description and/or category information that one of the entries in the feed is of further interest. The client may retrieve this incident Entry by performing an HTTP GET operation on the indicated URL.

Example HTTP GET request for an Entry:

```
GET /csirt/private/incidents/123456
Host: www.example.org
Accept: application/atom+xml
```

The corresponding HTTP response would be an XML document containing the incident:

Example HTTP GET response for an Entry:

```
HTTP/1.1 200 OK
Date: Fri, 24 Aug 2012 17:30:11 GMT
Content-Length: 4965
Content-Type: application/atom+xml;type=entry;charset="utf-8"

<?xml version="1.0" encoding="UTF-8"?>
<entry>
  <id>http://www.example.org/csirt/private/incidents/123456</id>
  <title>Sample Incident</title>
  <link href="http://www.example.org/csirt/private/incidents/123456" rel="self"/>
    <!-- by convention -->
  <link href="http://www.example.org/csirt/private/incidents/123456" rel="alternate"/>
    <!-- required by Atom spec -->
  <published>2012-08-04T18:13:51.0Z</published>
  <updated>2012-08-05T18:13:51.0Z</updated>
  <!-- The category is based upon IODEF purpose and restriction attributes -->
  <category term="traceback" scheme="purpose" label="trace back" />
```

```

        <category term="need-to-know" scheme="restriction" label="need to know" />
        <summary>A short description of this incident, extracted from the IODEF Incident class, <description> element. </summary>

        <!-- Refer to section 5.9 for the list of supported (cyber information-specific) link relationships -->
        <!-- Typical operations that can be performed on this IODEF message include edit -->
        <link href="http://www.example.org/csirt/private/incidents/123456" rel="edit"/>

        <!-- the next and previous are just sequential access, may not map to anything related to this IODEF Incident ID -->
        <link href="http://www.example.org/csirt/private/incidents/123457" rel="next"/>
        <link href="http://www.example.org/csirt/private/incidents/123455" rel="previous"/>

        <!-- navigate up to the full collection. Might also be rel="collection" as per IANA registry -->
        <link href="http://www.example.org/csirt/private/incidents" rel="up"/>
    >

    <content type="application/xml">
        <iodef:IODEF-Document lang="en" xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0">
            <iodef:Incident purpose="traceback" restriction="need-to-know">

                <!-- Note that the ID is assigned using a namespace that is our base URL, so that it can also be leveraged as an Atom link -->
                <iodef:IncidentID name="http://www.example.org/csirt/private/incidents">123456</iodef:IncidentID>

                <iodef:DetectTime>2004-02-02T22:49:24+00:00</iodef:DetectTime>
                <iodef:StartTime>2004-02-02T22:19:24+00:00</iodef:StartTime>
                <iodef:ReportTime>2004-02-02T23:20:24+00:00</iodef:ReportTime>
                <iodef:Description>
                    Host involved in DoS attack
                </iodef:Description>
                <iodef:Assessment>
                    <iodef:Impact completion="failed" severity="low" type="dos"/>
                </iodef:Assessment>
                <iodef:Contact role="creator" type="organization">
                    <iodef:ContactName>Constituency-contact for 192.0.2.35</iodef:ContactName>
                    <iodef:Email>Constituency-contact@192.0.2.35</iodef:Email>
                </iodef:Contact>
                <iodef:EventData>
                    <iodef:Flow>
                        <iodef:System category="source">
                            <iodef:Node>
                                <iodef:Address category="ipv4-addr">192.0.2.35</iodef:Address>
                            </iodef:Node>
                            <iodef:Service ip_protocol="6">
                                <iodef:Port>38765</iodef:Port>
                            </iodef:Service>
                        </iodef:System>
                        <iodef:System category="target">
                            <iodef:Node>

```

Field

Expires February 16, 2014

[Page 17]

```

        <iodef:Address category="ipv4-addr">192.0.2.67
      </iodef:Address>
    </iodef:Node>
    <iodef:Service ip_protocol="6">
      <iodef:Port>80</iodef:Port>
    </iodef:Service>
  </iodef:System>
</iodef:Flow>
<iodef:Expectation action="rate-limit-host" severity="high">
  <iodef:Description>
    Rate-limit traffic close to source
  </iodef:Description>
</iodef:Expectation>
<iodef:Record>
  <iodef:RecordData>
    <iodef:Description>
      The IPv4 packet included was used in the described atta
ck
    </iodef:Description>
    <iodef:RecordItem dtype="ipv4-packet">450000522ad9
      0000ff06c41fc0a801020a010102976d0050103e020810d9
      4a1350021000ad6700005468616e6b20796f7520666f7220
      6361726566756c6c792072656164696e6720746869732052
      46432e0a
    </iodef:RecordItem>
  </iodef:RecordData>
</iodef:Record>
</iodef:EventData>
</iodef:Incident>
</iodef:IODEF-Document>
</content>
</entry>

```

As can be seen in the example response, above, an IODEF document is contained within the Atom <content> element. The client may now process the IODEF document as needed.

Note also that, as described previously, the content of the Atom <category> element is application-defined. In the present context, the Atom categories have been assigned based on a mapping of the <restriction> and <purpose> attributes, as defined in the IODEF schema. In addition, the IODEF <incidentID> element has been judiciously chosen so that the associated name attribute, as well as the corresponding incidentID value, can be concatenated in order to easily create the corresponding <id> element for the Atom entry. These and other mappings are normatively defined in Section 5, below.

Finally, it should be noted that in order to optimize the client experience, and avoid an additional round trip, a feed provider may choose to include the entry content inline, as part of the feed document. That is, an Atom <entry> element within a Feed document may contain an Atom <content> element as a child. In this case, the client will receive the full content of the entries within the feed. The decision of whether to include the entry content inline or to include it as a link is a design choice left to the feed provider (e.g. based upon local environmental factors such as the number of entries contained in a feed, the available network bandwidth, the available server compute cycles, the expected client usage patterns, etc.).

#### 4.2.4. Use of Link Relations

As noted previously, a key benefit of using the RESTful architectural style is the ability to enable the client to navigate to related resources through the use of hypermedia links. In the Atom Syndication Format, the type of the related resource identified in a <link> element is indicated via the "rel" attribute, where the value of this attribute identifies the kind of related resource available at the corresponding "href" attribute. Thus, in lieu of a well-known URI template the URI itself is effectively opaque to the client, and therefore the client must understand the semantic meaning of the "rel" attribute in order to successfully navigate. Broad interoperability may be based upon a sharing consortium defining a well-known set of Atom Link Relation types. These Link Relation types may either be registered with IANA, or held in a private registry.

Individual CSIRTs may always define their own link relation types in order to support specific use cases, however support for a core set of well-known link relation types is encouraged as this will maximize interoperability.

In addition, it may be beneficial to define use case profiles that correspond to specific groupings of supported link relationship types. In this way, a CSIRT may unambiguously specify the classes of use cases for which a client can expect to find support.

The following sections provide NON-NORMATIVE examples of link relation usage. Four distinct cyber security information sharing use case scenarios are described. In each use case, the unique benefits of adopting a resource-oriented approach to information sharing are illustrated. It is important to note that these use cases are intended to be a small representative set and is by no means meant to be an exhaustive list. The intent is to illustrate how the use of link relationship types will enable this resource-oriented approach

to cyber security information sharing to successfully support the complete range of existing use cases, and also to motivate an initial list of well-defined link relationship types.

#### 4.2.4.1. Use Case: Incident Sharing

This section provides a non-normative example of an incident sharing use case.

In this use case, a member CSIRT shares incident information with another member CSIRT in the same consortium. The client CSIRT retrieves a feed of incidents, and is able to identify one particular entry of interest. The client then does an HTTP GET on that entry, and the representation of that resource contains link relationships for both the associated "indicators" and the incident "history", and so on. The client CSIRT recognizes that some of the indicator and history may be relevant within her local environment, and can respond proactively.

Example HTTP GET response for an incident entry:

```

<?xml version="1.0" encoding="UTF-8"?>
<entry>
  <id>http://www.example.org/csirt/private/incidents/123456</id>
  <title>Sample Incident</title>
  <link href="http://www.example.org/csirt/private/incidents/123456" rel="self"/>
    <!-- by convention -->
  <link href="http://www.example.org/csirt/private/incidents/123456" rel="alternate"/>
    <!-- required by Atom spec -->
  <published>2012-08-04T18:13:51.0Z</published>
  <updated>2012-08-05T18:13:51.0Z</updated>

  <link href="http://www.example.org/csirt/private/incidents/123456" rel="edit"/>

  <!-- The links to indicators related to this incident, and the history of this incident, and so on.... -->
  <link href="http://www.example.org/csirt/private/incidents/123456/relationships/indicators" rel="indicators"/>
  <link href="http://www.example.org/csirt/private/incidents/1234456/relationships/history" rel="history"/>
  <link href="http://www.example.org/csirt/private/incidents/1234456/relationships/campaign" rel="campaign"/>

  <!-- navigate up to the full collection. Might also be rel="collection" as per IANA registry -->
  <link href="http://www.example.org/csirt/private/incidents" rel="up"/>
>

  <content type="application/xml">
    <iodef:IODEF-Document lang="en" xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0">
      <iodef:Incident purpose="traceback" restriction="need-to-know">
        <iodef:IncidentID name="http://www.example.org/csirt/private/incidents">123456</iodef:IncidentID>
        <!-- ...additional incident data.... -->
      </iodef:Incident>
    </iodef:IODEF-Document>
  </content>
</entry>

```

As can be seen in the example response, the Atom <link> elements enable the client to navigate to the related indicator resources, and/or the history entries associated with this incident.

#### 4.2.4.2. Use Case: Collaborative Investigation

This section provides a non-normative example of a collaborative investigation use case.

In this use case, two member CSIRTs that belong to a closed sharing consortium are collaborating on an incident investigation. The initiating CSIRT performs an HTTP GET to retrieve the service document of the peer CSIRT, and determines the collection name to be used for creating a new investigation request. The initiating CSIRT then POSTs a new incident entry to the appropriate collection URL. The target CSIRT acknowledges the request by responding with an HTTP status code 201 Created.





Example HTTP GET response for the service document:

```

HTTP/1.1 200 OK
Date: Fri, 24 Aug 2012 17:09:11 GMT
Content-Length: 934
Content-Type: application/atomsvc+xml;charset="utf-8"

<?xml version="1.0" encoding="UTF-8"?>
<service xmlns="http://www.w3.org/2007/app"
  xmlns:atom="http://www.w3.org/2005/Atom">
  <workspace xml:lang="en-US" xmlns:xml="http://www.w3.org/XML/1998/namespace">
    <atom:title type="text">RID Use Case Requests</atom:title>
    <collection href="http://www.example.org/csirt/RID/Investigation
Requests">
      <atom:title type="text">Investigation Requests</atom:title>
      <accept>application/atom+xml; type=entry</accept> <!-- perhaps
s we should have a more specific media type -->
    </collection>
    <collection href="http://www.example.org/csirt/RID/TraceRequests
">
      <atom:title type="text">Trace Requests</atom:title>
      <accept>application/atom+xml; type=entry</accept>
    </collection>
    <!-- ...and so on.... -->
  </workspace>
</service>

```

As can be seen in the example response, the Atom <collection> elements enable the client to determine the appropriate collection URL to request an investigation or a trace.

The client CSIRT then POSTs a new entry to the appropriate feed collection. Note that the <content> element of the new entry may contain a RID message of type "InvestigationRequest" if desired, however this would NOT be required. The entry content itself need only be an IODEF document, with the choice of the target collection resource URL indicating the callers intent. A CSIRT would be free to use any URI template to accept investigationRequests.

```

POST /csirt/RID/InvestigationRequests HTTP/1.1
Host: www.example.org
Content-Type: application/atom+xml;type=entry
Content-Length: 852

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <title>New Investigation Request</title>
  <id>http://www.example2.org/csirt/private/incidents/123456</id>  <!-- id an
d updated not guranteed to be preserved -->
  <updated>2012-08-12T11:08:22Z</updated>                                <!-- may wa
nt to profile that behavior in this document -->
  <author><name>Name of peer CSIRT</name></author>
  <content type="application/xml">
    <iodef:IODEF-Document lang="en" xmlns:iodef="urn:ietf:params:xml:ns:iodef
-1.0">
      <iodef:Incident purpose="traceback" restriction="need-to-know">
        <iodef:IncidentID name="http://www.example2.org/csirt/private/incidents
">123</iodef:IncidentID>
        <!-- ...additional incident data.... -->
      </iodef:Incident>
    </iodef:IODEF-Document>
  </content>
</entry>

```

The receiving CSIRT acknowledges the request with HTTP return code 201 Created.

```

HTTP/1.1 201 Created
Date: Fri, 24 Aug 2012 19:17:11 GMT
Content-Length: 906
Content-Type: application/atom+xml;type=entry
Location: http://www.example.org/csirt/RID/InvestigationRequests/823
ETag: "8a9h9he4qphqh"

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <title>New Investigation Request</title>
  <id>http://www.example.org/csirt/RID/InvestigationRequests/823</id>  <!-- i
d and updated not guranteed to be preserved -->
  <updated>2012-08-12T11:08:30Z</updated>                                <!-- m
ay want to profile that behavior in this document -->
  <published>2012-08-12T11:08:30Z</published>
  <author><name>Name of peer CSIRT</name></author>
  <content type="application/xml">
    <iodef:IODEF-Document lang="en" xmlns:iodef="urn:ietf:params:xml:ns:iodef
-1.0">
      <iodef:Incident purpose="traceback" restriction="need-to-know">
        <iodef:IncidentID name="http://www.example.org/csirt/private/incidents"
>123</iodef:IncidentID>
        <!-- ...additional incident data.... -->
      </iodef:Incident>
    </iodef:IODEF-Document>
  </content>
</entry>

```

```
</content>
</entry>
```

Consistent with HTTP/1.1 RFC, the location header indicates the URL of the newly created InvestigationRequest. If for some reason the request were not authorized, the client would receive an HTTP status code 403 Unauthorized. In this case the HTTP response body may contain additional details, if an as appropriate.

#### 4.2.4.3. Use Case: Search (Query)

This section provides a non-normative example of a search use case.

The following example provides a RESTful alternative to the RID Query message, as described in sections 6.5 and 7.4 of RFC6545. Note that in the RESTful approach described herein there is no requirement to define a query language specific to RID queries. Instead, CSIRTs may provide support for search operations via existing search facilities, and advertise these capabilities via an appropriate URL template. Clients dynamically retrieve the search description document, and invoke specific searches via an instantiated URL template.

An HTTP response body may include a link relationship of type "search." This link provides a reference to an OpenSearch description document.

Example HTTP response that includes a "search" link:

```
HTTP/1.1 200 OK
Date: Fri, 24 Aug 2012 17:20:11 GMT
Content-Length: nnnn
Content-Type: application/atom+xml;type=feed;charset="utf-8"

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/Atom file:/C:/schemas/at
om.xsd
                        urn:ietf:params:xml:ns:iodef-1.0 file:/C:/schem
as/iodef-1.0.xsd"
      xml:lang="en-US">
  <link href="http://www.example.org/opensearchdescription.xml" rel="
search"
        type="application/opensearchdescription+xml"
        title="CSIRT search facility" />

  <!-- ...other links... -->

  <entry>
    <!-- ...zero or more entries... -->
  </entry>

</feed>
```

The OpenSearch Description document contains the information needed by a client to request a search. An example of an Open Search description document is shown below:

Example HTTP response that includes a "search" link:

```

<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1
.1/">
  <ShortName>CSIRT search example</ShortName>
  <Description>Cyber security information sharing consortium se
arch interface</Description>
  <Tags>example csirt indicator search</Tags>
  <Contact>admin@example.org</Contact>
  <!-- ...optionally, other elements, as per OpenSearch specifi
cation... -->
  <Url type="application/opensearchdescription+xml" rel="self"
template="http://www.example.com/csirt/opensearchdescription.xml"/>
  <Url type="application/atom+xml" rel="results" template="http
://www.example.org/csirt?q={searchTerms}&format=Atom+xml"/>
  <LongName>www.example.org CSIRT search</LongName>
  <Query role="example" searchTerms="incident" />
  <Language>en-us</Language>
  <OutputEncoding>UTF-8</OutputEncoding>
  <InputEncoding>UTF-8</InputEncoding>
</OpenSearchDescription>

```

The OpenSearch Description document shown above contains two <Url> elements that contain parameterized URL templates. These templates provide a representation of how the client should make search requests. The exact format of the query string, including the parameterization is specified by the feed provider.

This OpenSearch Description Document also contains an example of a <Query> element. Each <Query> element describes a specific search request that can be made by the client. Note that the parameters of the <Query> element correspond to the URL template parameters. In this way, a provider may fully describe the search interface available to the clients. Section 5.12, below, provides specific NORMATIVE requirements for the use of Open Search.

#### 4.2.4.4. Use Case: Cyber Data Repository

This section provides a non-normative example of a cyber security data repository use case.

In this use case a client accesses a persistent repository of cyber security data via a RESTful usage model. Retrieving a feed collection is analogous to an SQL SELECT statement producing a result set. Retrieving an individual Atom Entry is analogous to a SQL SELECT statement based upon a primary key producing a unique record. The cyber security data contained in the repository may include different data types, including indicators, incidents, becnmarks, or any other related resources. In this use case, the repository is queried via HTTP GET, and the results that are returned to the client may optionally contain URL references to other cyber security

resources that are known to be related. These related resources may also be persisted locally, or they may exist at another (remote) cyber data repository.

Example HTTP GET request to a persistent repository for any resources representing Distributed Denial of Service (DDOS) attacks:

```
GET /csirt/repository/ddos
Host: www.example.org
Accept: application/atom+xml
```

The corresponding HTTP response would be an XML document containing the DDOS feed.

Example HTTP GET response for a DDOS feed:

```

HTTP/1.1 200 OK
Date: Fri, 24 Aug 2012 17:20:11 GMT
Content-Length: nnnn
Content-Type: application/atom+xml;type=feed; charset="utf-8"

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/Atom file:/C:/schemas/at
om.xsd
                        urn:ietf:params:xml:ns:iodef-1.0 file:/C:/schem
as/iodef-1.0.xsd"
      xml:lang="en-US">

  <generator version="1.0" xml:lang="en-US">emc-csirt-iodef-feed-serv
ice</generator>
  <id xml:lang="en-US">http://www.example.org/csirt/repository/ddos</
id>
  <title type="text" xml:lang="en-US">Atom formatted representation o
f a feed of known ddos resources.</title>
  <updated xml:lang="en-US">2012-05-04T18:13:51.0Z</updated>
  <author>
    <email>csirt@example.org</email>
    <name>EMC CSIRT</name>
  </author>

  <!-- By convention there is usually a self link for the feed -->
  <link href="http://www.example.org/csirt/repository/ddos" rel="self
"/>

  <entry>
    <id>http://www.example.org/csirt/repository/ddos/123456</id>
    <title>Sample DDOS Incident</title>
    <link href="http://www.example.org/csirt/repository/ddos/123456
" rel="self"/>      <!-- by convention -->
    <link href="http://www.example.org/csirt/repository/ddos/123456
" rel="alternate"/>  <!-- required by Atom spec -->
    <link href="http://www.example.org/csirt/repository/ddos/987654
" rel="related"/>    <!-- link to a related DDOS resource in this repository
-->
    <link href="http://www.cyber-agency.gov/repository/indicators/1
a2b3c" rel="related"/> <!-- link to a related DDOS resource in another reposito
ry -->
    <published>2012-08-04T18:13:51.0Z</published>
    <updated>2012-08-05T18:13:51.0Z</updated>
    <!-- The category is based upon IODEF purpose and restriction a
ttributes -->
    <category term="traceback" scheme="purpose" label="trace back"
/>
    <category term="need-to-know" scheme="restriction" label="need
to know" />
    <category term="ddos" scheme="ttp" label="tactics, techniques,
and procedures"/>
    <summary>A short description of this DDOS attack, extracted fro
m the IODEF Incident class, <description> element. </summary>
  </entry>

  <entry>
    <!-- ...another entry... -->
  </entry>

</feed>

```

Field

Expires February 16, 2014

[Page 28]



This feed document has two atom entries, one of which has been elided. The completed entry illustrates an Atom <entry> element that provides a summary of essential details about one particular DDOS incident. Based upon this summary information and the provided category information, a client may choose to do an HTTP GET operation to retrieve the full details of the DDOS incident. This example shows how a persistent repository may provide links to additional resources, both local and remote.

Note that the provider of a persistent repository is not obligated to follow any particular URL template scheme. The repository available at the hypothetical provider "www.example.com" uses a different URL pattern than the hypothetical repository available at "www.cyber-agency.gov". When a client de-references a link to resource that is located in a remote repository the client may be challenged for authentication credentials acceptable to that provider. If the two repository providers choose to support a federated identity scheme or some other form of single-sign-on technology, then the user experience can be improved for interactive clients (e.g., a human user at a browser). However, this is not required and is an implementation choice that is out of scope for this specification.

## 5. Requirements for RESTful (Atom+xml) Binding

This section provides the NORMATIVE requirements for using Atom format and Atom Pub as a RESTful binding for cyber security information sharing.

### 5.1. Transport Layer Security

Servers implementing this specification MUST support server-authenticated TLS.

Servers MAY support mutually authenticated TLS.

### 5.2. User Authentication

Servers MUST require user authentication.

Servers MAY support more than one client authentication method.

Servers participating in an information sharing consortium and supporting interactive user logins by members of the consortium SHOULD support client authentication via a federated identity scheme as per SAML 2.0.

Servers MAY support client authenticated TLS.

### 5.3. User Authorization

This document does not mandate the use of any specific user authorization mechanisms. However, service implementers SHOULD provide appropriate authorization checking for all resource accesses, including individual Atom Entries, Atom Feeds, and Atom Service Documents.

Authorization for a resource MAY be adjudicated based on the value(s) of the associated Atom <category> element(s).

When the content model for the Atom <content> element of an Atom Entry contains an <IODEF-Document>, then authorization MUST be adjudicated based upon the Atom <category> element(s), whose values have been mapped as per Section 5.7.

Any use of the <category> element(s) as an input to an authorization policy decision MUST include both the "scheme" and "term" attributes contained therein. As described in Section 5.7 below, the namespace of the "term" attribute is scoped by the associated "scheme" attribute.

### 5.4. Content Model

Member entry resources providing a representation of an incident resource (e.g., as specified in the link relation type) MUST use the IODEF schema as the content model for the Atom Entry <content> element.

Member Entry resources providing a representation of an indicator resource (e.g., as specified in the link relation type) MUST use the IODEF schema as the content model for the Atom Entry <content> element.

The resource representation MAY include an appropriate indicator schema type within the <AdditionalData> element of the IODEF Incident class. Supported indicator schema types SHALL be registered via an IANA table (todo: IANA registration/review).

Member Entry resources providing a representation of a RID report resource (e.g., as specified in the link relation type) MUST use the RID schema as the content model for the Atom Entry <content> element.

Member Entry resources providing representation of other types, SHOULD use the IODEF schema as the content model for the Atom Entry <content> element.

If the member entry content model is not IODEF, then the <content> element of the Atom entry MUST contain an appropriate XML namespace declaration.

### 5.5. HTTP methods

The following table defines the HTTP [RFC2616] uniform interface methods supported by this specification:

HTTP method	Description
GET	Returns a representation of an individual member entry resource, or a feed collection.
PUT	Replaces the current representation of the specified member entry resource with the representation provided in the HTTP request body.
POST	Creates a new instance of a member entry resource. The representation of the new resource is provided in the HTTP request body.
DELETE	Removes the indicated member entry resource, or feed collection.
HEAD	Returns metadata about the member entry resource, or feed collection, contained in HTTP response headers.
PATCH	Support TBD.

Table 1: Uniform Interface for Resource-Oriented Lightweight Indicator Exchange

Clients MUST be capable of recognizing and prepared to process any standard HTTP status code, as defined in [RFC2616]

### 5.6. Service Discovery

This specification requires that a CSIRT MUST publish an Atom Service Document that describes the set of cyber security information sharing feeds that are provided.

The service document SHOULD be discoverable via the CSIRT organization's Web home page or another well-known public resource.

#### 5.6.1. Workspaces

The service document MAY include multiple workspaces. Any CSIRT providing both public feeds and private consortium feeds MUST place these different classes of feeds into different workspaces, and provide appropriate descriptions and naming conventions to indicate the intended audience of each workspace.

#### 5.6.2. Collections

A CSIRT MAY provide any number of collections within a given Workspace. It is RECOMMENDED that each collection appear in only a single Workspace. It is RECOMMENDED that at least one collection be provided that accepts new incident reports from users. At least one collection MUST provide a feed of incident information for which the content model for the entries uses the IODEF schema. The title of this collection SHOULD be "Incidents".

#### 5.6.3. Service Document Security

Access to the service document MUST be protected via server-authenticated TLS and a server-side certificate.

When deploying a service document for use by a closed consortium, the service document MAY also be digitally signed and/or encrypted, using XML DigSig and/or XML Encryption, respectively.

#### 5.7. Category Mapping

This section defines normative requirements for mapping IODEF metadata to corresponding Atom category elements. (todo: decide between IANA registration of scheme, or use a full URI).

##### 5.7.1. Collection Category

An Atom collection MAY hold entries from one or more categories. The collection category set MUST contain at least the union of all the member entry categories. A collection MAY have additional category metadata that are unique to the collection, and not applicable to any individual member entry. A collection containing IODEF incident content MUST contain at least two <category> elements. One category MUST be specified with the value of the "scheme" attribute as "restriction". One category MUST be specified with the value of the "scheme" attribute as "purpose". The value of the "fixed" attribute for both of these category elements MUST be "yes". When the category scheme="restriction", the allowable values for the "term" attribute are constrained as per section 3.2 of IODEF, e.g. public, need-to-know, private, default. When the category scheme="purpose", the

allowable values for the "term" attribute are constrained as per section 3.2 of IODEF, e.g. traceback, mitigation, reporting, other.

#### 5.7.2. Entry Category

An Atom entry containing IODEF content MUST contain at least two <category> elements. One category MUST be specified with the value of the "scheme" attribute as "restriction". One category MUST be specified with the value of the "scheme" attribute as "purpose". When the category scheme="restriction", the value of the "term" attribute must be exactly one of ( public, need-to-know, private, default). When the category scheme="purpose", the value of the "term" attribute must be exactly one of (traceback, mitigation, reporting, other). When the purpose is "other"....

Any member entry MAY have any number of additional categories.

#### 5.8. Entry ID

The ID element for an Atom entry SHOULD be established via the concatenation of the value of the name attribute from the IODEF <IncidentID> element and the corresponding value of the <IncidentID> element. This requirement ensures a simple and direct one-to-one relationship between an IODEF incident ID and a corresponding Feed entry ID and avoids the need for any system to maintain a persistent store of these identity mappings.

(todo: Note that this implies a constraint on the IODEF document that is more restrictive than the current IODEF schema. IODEF section 3.3 requires only that the name be a STRING type. Here we are stating that name must be an IRI. Possible request to update IODEF to constrain, or to support a new element or attribute).

#### 5.9. Entry Content

The <content> element of an Atom <entry> SHOULD include an IODEF document. The <entry> element SHOULD include an appropriate XML namespace declaration for the IODEF schema. If the content model of the <entry> element does not follow the IODEF schema, then the <entry> element MUST include an appropriate XML namespace declaration.

A client MAY ignore content that is not using the IODEF schema.

#### 5.10. Link Relations

In addition to the standard Link Relations defined by the Atom specification, this specification defines the following additional

Link Relation terms, which are introduced specifically in support of the Resource-Oriented Lightweight Indicator Exchange protocol.

Name	Description	Conformance
service	Provides a link to an atom service document associated with the collection feed.	MUST
search	Provides a link to an associated Open Search document that describes a URL template for search queries.	MUST
history	Provides a link to a collection of zero or more historical entries that are associated with the resource.	MUST
incidents	Provides a link to a collection of zero or more instances of actual cyber security event(s) that are associated with the resource.	MUST
indicators	Provides a link to a collection of zero or more instances of cyber security indicators that are associated with the resource.	MUST
evidence	Provides a link to a collection of zero or more resources that provides some proof of attribution for an incident. The evidence may or may not have any identified chain of custody.	SHOULD
campaign	Provides a link to a collection of zero or more resources that provides a representation of the associated cyber attack campaign.	SHOULD
attacker	Provides a link to a collection of zero or more	SHOULD

	resources that provides a representation of the attacker.	
vector	Provides a link to a collection of zero or more resources that provides a representation of the method used by the attacker.	SHOULD
assessments	Provides a link to a collection of zero or more resources that represent the results of executing a benchmark.	SHOULD
reports	Provides a link to a collection of zero or more resources that represent RID reports.	SHOULD
traceRequests	Provides a link to a collection of zero or more resources that represent RID traceRequests.	SHOULD
investigationRequests	Provides a link to a collection of zero or more resources that represent RID investigationRequests.	SHOULD
swid	Provides a link to a collection of zero or more resources that represent related Software ID tags.	SHOULD

Table 2: Link Relations for Resource-Oriented Lightweight Indicator Exchange

Unless specifically registered with IANA these short names MUST be fully qualified via concatenation with a base-uri. An appropriate base-uri could be established via agreement amongst the members of an information sharing consortium. For example, the rel="indicators" relationship would become rel="http://www.example.org/csirt/incidents/relationships/indicators."

#### 5.10.1. Additional Link Relation Requirements

An IODEF document that is carried in an Atom Entry SHOULD NOT contain a <relatedActivity> element. Instead, the related activity SHOULD be available via a link rel=related.

An IODEF document that is carried in an Atom Entry SHOULD NOT contain a <history> element. Instead, the related history SHOULD be available via a link rel="history" (todo: or a fully qualified link rel name). The associated href MAY leverage OpenSearch to specify the required query.

An Atom Entry MAY include additional link relationships not specified here. If a client encounters a link relationship of an unknown type the client MUST ignore the offending link and continue processing the remaining resource representation as if the offending link element did not appear.

The link relationship "swid" may be used for a collection of zero or more software identification tags that are related to the current indicator. The representation of the resources available via this link relationship MAY follow an appropriate standard, such as ISO/IEC 19770-2:2009 Information technology -- Software asset management -- Part 2: Software identification tag.

#### 5.11. Member Entry Forward Security

As described in Authorization Policy Enforcement (Authorization Policy Enforcement) a RESTful model for cyber security information sharing requires that all of the required security enforcement for feeds and entries MUST be enforced at the source system, at the point the representation of the given resource(s) is created. A CSIRT provider SHALL NOT return any feed content or member entry content for which the client identity has not been specifically authenticated, authorized, and audited.

Sharing communities that have a requirement for forward message security (such that client systems are required to participate in providing message level security and/or distributed authorization policy enforcement), MUST use the RID schema as the content model for the member entry <content> element.

#### 5.12. Date Mapping

The Atom feed <updated> element MUST be populated with the current time at the instant the feed representation was generated. The Atom entry <published> element MUST be populated with the same time value as the <reportTime> element from the IODEF document.



### 5.13. Search

Implementers MUST support OpenSearch 1.1 [opensearch] as the mechanism for describing how clients may form search requests.

Implementers MUST provide a link with a relationship type of "search". This link SHALL return an Open Search Description Document as defined in OpenSearch 1.1.

Implementers MUST support an OpenSearch 1.1 compliant search URL template that enables a search query via Atom Category, including the scheme attribute and terms attribute as search parameters.

Implementers SHOULD support search based upon the IODEF AlternativeID class as a search parameter.

Implementers SHOULD support search based upon the four timestamp elements of the IODEF Incident class: <startTime>, <EndTime>, <DetectTime>, and <ReportTime>.

Implementers MAY support additional search capabilities based upon any of the remaining elements of the IODEF Incident class, including the <Description> element.

Collections that support use of the RID schema as a content model in the Atom member entry <content> element (e.g. in a report resource representation reachable via the "report" link relationship) MUST support search operations that include the RID MessageType as a search parameter, in addition to the aforementioned IODEF schema elements, as contained within the <ReportSchema> element.

Implementers MUST fully qualify all OpenSearch URL template parameter names using the defined IODEF or RID XML namespaces, as appropriate.

### 5.14. / (forward slash) Resource URL

The "/" resource MAY be provided for compatibility with existing deployments that are using Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS [RFC6546]. Consistent with RFC6546 errata, a client requesting a GET on "/" MUST receive an HTTP status code 405 Method Not Allowed. An implementation MAY provide full support for RFC6546 such that a POST to "/" containing a recognized RID message type just works. Alternatively, a client requesting a POST to "/" MAY receive an HTTP status code 307 Temporary Redirect. In this case, the location header in the HTTP response will provide the URL of the appropriate RID endpoint, and the client may repeat the POST method at the indicated location. This resource could also leverage the new draft by reschke that

proposes HTTP status code 308 (cf: draft-reschke-http-status-308-07.txt).

## 6. Security Considerations

This document defines a resource-oriented approach to lightweight indicator exchange using HTTP, TLS, Atom Syndicate Format, and Atom Publishing Protocol. As such, implementers must understand the security considerations described in those specifications.

In addition, there are a number of additional security considerations that are unique to this specification.

As described above in the section Authentication of Users (Section 3.2), the approach described herein is based upon all policy enforcements being implemented at the point when a resource representation is created. As such, CSIRTS sharing cyber security information using this specification must take care to authenticate their HTTP clients using a suitably strong user authentication mechanism. Sharing communities that are exchanging information on well-known indicators and incidents for purposes of public education may choose to rely upon, e.g. HTTP Authentication, or similar. However, sharing communities that are engaged in sensitive collaborative analysis and/or operational response for indicators and incidents targeting high value information systems should adopt a suitably stronger user authentication solution, such as TLS client certificates, or a risk-based or multi-factor approach. In general, trust in the sharing consortium will depend upon the members maintaining adequate user authentication mechanisms.

Collaborating consortiums may benefit from the adoption of a federated identity solution, such as those based upon SAML-core [SAML-core] and SAML-bind [SAML-bind] and SAML-prof [SAML-prof] for Web-based authentication and cross-organizational single sign-on. Dependency on a trusted third party identity provider implies that appropriate care must be exercised to sufficiently secure the Identity provider. Any attacks on the federated identity system would present a risk to the CISRT, as a relying party. Potential mitigations include deployment of a federation-aware identity provider that is under the control of the information sharing consortium, with suitably stringent technical and management controls.

As discussed above in the section Authorization Policy Enforcement (Section 3.3), authorization of resource representations is the responsibility of the source system, i.e. based on the authenticated user identity associated with an HTTP(S) request. The required authorization policies that are to be enforced must therefore be

managed by the security administrators of the source system. Various authorization architectures would be suitable for this purpose, such as RBAC [1] and/or ABAC, as embodied in XACML [XACML]. In particular, implementers adopting XACML may benefit from the capability to represent their authorization policies in a standardized, interoperable format.

Additional security requirements such as enforcing message-level security at the destination system could supplement the security enforcements performed at the source system, however these destination-provided policy enforcements are out of scope for this specification. Implementers requiring this capability should consider leveraging, e.g. the <RIDPolicy> element in the RID schema. Refer to RFC6545 section 9 for more information.

When security policies relevant to the source system are to be enforced at both the source and destination systems, implementers must take care to avoid unintended interactions of the separately enforced policies. Potential risks will include unintended denial of service and/or unintended information leakage. These problems may be mitigated by avoiding any dependence upon enforcements performed at the destination system. When distributed enforcement is unavoidable, the usage of a standard language (e.g. XACML) for the expression of authorization policies will enable the source and destination systems to better coordinate and align their respective policy expressions.

Adoption of the information sharing approach described in this document will enable users to more easily perform correlations across separate, and potentially unrelated, cyber security information providers. A client may succeed in assembling a data set that would not have been permitted within the context of the authorization policies of either provider when considered individually. Thus, providers may face a risk of an attacker obtaining an access that constitutes an undetected separation of duties (SOD) violation. It is important to note that this risk is not unique to this specification, and a similar potential for abuse exists with any other cyber security information sharing protocol. However, the wide availability of tools for HTTP clients and Atom feed handling implies that the resources and technical skills required for a successful exploit may be less than it was previously. This risk can be best mitigated through appropriate vetting of the client at account provisioning time. In addition, any increase in the risk of this type of abuse should be offset by the corresponding increase in effectiveness that this specification affords to the defenders.

While it is a goal of this specification to enable more agile cyber security information sharing across a broader and varying constituency, there is nothing in this specification that necessarily

requires this type of deployment. A cyber security information sharing consortium may chose to adopt this specification while continuing to operate as a gated community with strictly limited membership.

## 7. IANA Considerations

If the values of the newly defined link relations are not fully qualified URIs then we need to register these link types with IANA (e.g. rel="history") It is possible to adjust this document so that it has no actions for IANA.

## 8. ToDo and Open Issues

The following is the "todo" and open issues list:

1. Need to make a decision on whether new IANA link registrations are required, or whether fully qualified (private) link types are sufficient.
2. Should we require Atom categories that correspond to IODEF Expectation class and/or IODEF Impact class?
3. Should we include specific requirements for Archive and Paging? Perhaps just reference RFC 5005?
4. We need more requirements input on use cases involving RID schema in the Atom member entry content model for link rel=report.
5. An Atom service document will have categories, but this is still coarse-grained, and not visible at the transport protocol level. Should we include a MIME media type parameter to support negotiation and better document the content model schema contained in a collection, i.e.:

Accept: application/atom+xml;type=entry;content=iodef

Accept: application/atom+xml;type=entry;content=rid

Accept: application/atom+xml;type=entry;content=iodef+openioc

6. If so, I think these parameters may require media type registration as per RFC4288?
7. Further work is needed to investigate the use of a link relationship for SWID tags, as related resources.

8. It has been suggested that a RESTful binding approach similar to ROLIE may be relevant to certain related use cases being considered in SACM. This requires further discussion to explore the requirements in more detail.

## 9. Acknowledgements

The author gratefully acknowledges the valuable contributions of Tom Maguire, Kathleen Moriarty, and Vijayanand Bharadwaj. These individuals provided detailed review comments on earlier drafts, and many suggestions that have helped to improve this document.

## 10. References

### 10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.
- [RFC4287] Nottingham, M., Ed. and R. Sayre, Ed., "The Atom Syndication Format", RFC 4287, December 2005.
- [RFC5023] Gregorio, J. and B. de hOra, "The Atom Publishing Protocol", RFC 5023, October 2007.
- [RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, December 2007.
- [RFC6545] Moriarty, K., "Real-time Inter-network Defense (RID)", RFC 6545, April 2012.
- [opensearch] Clinton, D., "OpenSearch 1.1 draft 5 specification", 2011, <<http://www.opensearch.org/Specifications/OpenSearch/1.1>>.
- [SAML-core] Cantor, S., Kemp, J., Philpott, R., and E. Mahler, "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0 ", OASIS Standard , March 2005, <<http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>>.
- [SAML-prof]

Hughes, J., Cantor, S., Hodges, J., Hirsch, F., Mishra, P., Philpott, R., and E. Mahler, "Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard, March 2005, <<http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf>>.

[SAML-bind]

Cantor, S., Hirsch, F., Kemp, J., Philpott, R., and E. Mahler, "Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard, March 2005, <<http://docs.oasis-open.org/security/saml/v2.0/saml-bindings-2.0-os.pdf>>.

## 10.2. Informative References

[XMLencrypt]

Imaura, T., Dillaway, B., and E. Simon, "XML Encryption Syntax and Processing", W3C Recommendation, December 2002, <<http://www.w3.org/TR/xmlenc-core/>>.

[XMLsig]

Bartel, M., Boyer, J., Fox, B., LaMaccia, B., and E. Simon, "XML-Signature Syntax and Processing", W3C Recommendation Second Edition, June 2008, <<http://www.w3.org/TR/xmldsig-core/>>.

[XACML]

Rissanen, E., "eXtensible Access Control Markup Language (XACML) Version 3.0", August 2010, <<http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf>>.

[REST]

Fielding, R., "Architectural Styles and the Design of Network-based Software Architectures", 2000, <<http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>>.

[SWID]

Surnyn, W., "ISO/IEC 19770-2:2009 Information technology - Software asset management - Part 2: Software identification tag", 2009, <[http://www.iso.org/iso/home/store/catalogue\\_ics/catalogue\\_detail\\_ics.htm%3Fics1%3D35%26ics2%3D080%26ics3%3D%26csnumber%3D53670](http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm%3Fics1%3D35%26ics2%3D080%26ics3%3D%26csnumber%3D53670)>.

[RFC2396]

Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[RFC2822]

Resnick, P., "Internet Message Format", RFC 2822, April 2001.

[RFC3339]

Klyne, G., Ed. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

- [RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC6546] Trammell, B., "Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS", RFC 6546, April 2012.

#### Appendix A. Change Tracking

Changes since -01 version, Feb 15, 2013:

- o Updated author's contact information.
- o Added a new link relationship definition to Table 2, for Software Identification tag (SWID) as another potential related resource.
- o Included a non-normative reference to the related ISO/IEC standard in this section, Additional Link Relation Requirements. See: Section 5.10.1
- o Corrected a small number of spelling errors and/or typos throughout.

#### Appendix B. Resource Authorization Model

As described in Section 3.3.2 above, ROLIE assumes that all authorization policy enforcement is provided at the source server. The implementation details of the authorization scheme chosen by a ROLIE-compliant provider are out of scope for this specification. Implementers are free to choose any suitable authorization mechanism that is capable of fulfilling the policy enforcement requirements relevant to their consortium and/or organization.

It is well known that one of the major barriers to information sharing is ensuring acceptable use of the information shared. In the case of ROLIE, one way to lower that barrier may be to develop a XACML profile. Use of XACML would allow a ROLIE-compliant provider to express their information sharing authorization policies in a standards-compliant, and machine-readable format.

This improved interoperability may, in turn, enable more agile interactions in the cyber security sharing community. For example, a peer CSIRT, or another interested stakeholder such as an auditor,

would be able to review and compare CSIRT sharing policies using appropriate tooling.

The XACML 3.0 standard is based upon the notion that authorization policies are defined in terms of predicate logic expressions written against the attributes associated with one or more of the following four entities:

- o SUBJECT
- o ACTION
- o RESOURCE
- o ENVIRONMENT

Thus, a suitable approach to a XACML 3.0 profile for ROLIE authorization policies could begin by using the 3-tuple of [SUBJECT, ACTION, RESOURCE] where:

- o SUBJECT is the suitably authenticated identity of the requestor.
- o ACTION is the associated HTTP method, GET, PUT, POST, DELETE, HEAD, (PATCH).
- o RESOURCE is an XPath expression that uniquely identifies the instance or type of the ROLIE resource being requested.

Implementers who have a need may also choose to evaluate based upon the additional ENVIRONMENT factors, such as current threat level, and so on. One could also write policy to consider the CVSS score associated with the resource, or the lifecycle phase of the resource (vulnerability unverified, confirmed, patch available, etc.), and so on.

Having these policies expressed in a standards-compliant and machine-readable format could improve the agility and effectiveness of a cyber security information sharing group or consortium, and enable better cyber defenses.

#### B.1. Example XACML Profile

Work-in-Progress. If this approach finds support in the community then this section (or a new draft, as a separate document) could provide a more complete XACML 3.0 compliant example.

Author's Address



John P. Field  
Pivotal  
841 Broadway  
8th Floor  
New York, New York  
USA

Phone: 973-635-0228  
Email: [jfield@gopivotal.com](mailto:jfield@gopivotal.com)

MILE Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: August 16, 2013

T. Takahashi  
NICT  
K. Landfield  
McAfee  
T. Millar  
USCERT  
Y. Kadobayashi  
NAIST  
Feb 12, 2013

IODEF-extension to support structured cybersecurity information  
draft-ietf-mile-sci-06.txt

#### Abstract

This document extends the Incident Object Description Exchange Format (IODEF) defined in RFC 5070 [RFC5070] to exchange enriched cybersecurity information among cybersecurity entities and facilitate their operations. It provides the capability of embedding structured information, such as identifier- and XML-based information.

#### Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 16, 2013.

#### Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	3
3. Applicability . . . . .	3
4. Extension Definition . . . . .	4
4.1. IANA Table for Structured Cybersecurity Information . . . . .	4
4.2. Extended Data Type: XMLDATA . . . . .	5
4.3. Extended Classes . . . . .	5
4.3.1. AttackPattern . . . . .	6
4.3.2. Platform . . . . .	8
4.3.3. Vulnerability . . . . .	9
4.3.4. Scoring . . . . .	10
4.3.5. Weakness . . . . .	11
4.3.6. EventReport . . . . .	12
4.3.7. Verifcation . . . . .	13
4.3.8. Remediation . . . . .	14
5. Mandatory to Implement features . . . . .	15
6. Security Considerations . . . . .	16
6.1. Transport-Specific Concerns . . . . .	16
7. IANA Considerations . . . . .	16
8. Acknowledgment . . . . .	18
9. Appendix I: XML Schema Definition for Extension . . . . .	18
10. Appendix II: Candidate Specifications for the IANA Table . . . . .	23
11. Appendix III: An XML Example . . . . .	27
12. References . . . . .	29
12.1. Normative References . . . . .	29
12.2. Informative References . . . . .	30
Authors' Addresses . . . . .	31

## 1. Introduction

The number of cyber attacks is growing day by day, and incident information needs to be reported, exchanged, and shared among organizations in order to cope with the situation. IODEF is one of the tools enabling such exchange, and is already in use.

To efficiently run cybersecurity operations, these exchanged information needs to be machine-readable. IODEF provides a structured means to describe the information, but it needs to embed various non-structured such information in order to convey detailed information. Further structure within IODEF increases IODEF documents' machine-readability and thus facilitates streamlining cybersecurity operations.

On the other hand, there exist various other activities facilitating detailed and structured description of cybersecurity information, as listed in Section 10. Since such structured description facilitates cybersecurity operations, it would be beneficial to embed and convey these information inside IODEF document.

To enable that, this document extends the IODEF to embed and convey various structured cybersecurity information, with which cybersecurity operations can be facilitated. Since IODEF defines a flexible and extensible format and supports a granular level of specificity, this document defines an extension to IODEF instead of defining a new report format. For clarity, and to eliminate duplication, only the additional structures necessary for describing the exchange of such structured information are provided.

## 2. Terminology

The terminology used in this document follows the one defined in RFC 5070 [RFC5070].

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 RFC 2119 [RFC2119].

## 3. Applicability

To maintain cybersecurity, organization needs to exchange cybersecurity information, which includes the following information: attack pattern, platform information, vulnerability and weakness, countermeasure instruction, computer event log, and the severity.

IODEF provides a scheme to describe and exchange such information among interested parties. However, it does not define the detailed format to describe such information.

On the other hand, there already exist structured and detailed formats for describing those information and facilitating such exchange. Major of them are listed in Section 10. By embedding them into the IODEF document, the document can convey more detailed contents to the receivers, and the document can be easily reused.

These structured cybersecurity information facilitates cybersecurity operation at the receiver side. Since the information is machine-readable, the data can be processed by computers. That expedites the automation of cybersecurity operations

For instance, an organization wishing to report a security incident wants to describe what vulnerability was exploited. Then the sender can simply use IODEF, where an XML [XML1.0]-based attack pattern record that follows the syntax and vocabulary defined by an industry specification is embedded instead of describing everything in free format text. Receiver can identify the needed details of the attack pattern by looking up some of the XML tags defined by the specification. Receiver can accumulate the attack pattern record in its database and could distribute it to the interested parties if needed, without needing human interventions.

Another example is that, when an administrator wishes to check the configuration of host computers in his organization, he may send a query to host computers, which may automatically generate XML-based software configuration information upon receiving the query by running a software and may embed that to an IODEF document, which is then sent back to the administrator.

#### 4. Extension Definition

This draft extends IODEF to embed structured cybersecurity information by introducing new classes, with which these information can be embedded inside IODEF document as element contents of AdditionalData and RecordItem classes.

##### 4.1. IANA Table for Structured Cybersecurity Information

This extension embeds structured cybersecurity information defined by the other specifications. The list of supported specifications is managed by IANA, and this draft defines the needed field for the list's entry.

Each entry has namespace [XMLNames], specification name, version, reference URI, and applicable classes for each specification. Arbitrary URIs that may help readers to understand the specification could be embedded inside the Reference URI field, but it is recommended that standard/informational URI describing the specification is prepared and is embedded here.

The initial IANA table has only one entry, as below.

```
Namespace:          http://xml/metadataSharing.xsd
Specification Name:  Malware Metadata Exchange Format
Version:            1.2
Reference URI:       http://standards.ieee.org/develop/
                    indconn/icsg/mmdef.html
Applicable Classes: AttackPattern
```

The table is to be managed by IANA using the Expert Review [RFC5226] and Specification Required [RFC5226] allocation policies as further specified in Section 7.

The SpecID attributes of extended classes (Section 4.3) must allow the values of the specifications' namespace fields, but otherwise, implementations are not required to support all specifications of the IANA table and may choose which specifications to support, though the specification listed in the initial table needs to be minimally supported, as described in Section 5. In case an implementation received a data it does not support, it may expand its functionality by looking up the IANA table or notify the sender of its inability to parse the data by using any means defined outside the scope of this specification.

#### 4.2. Extended Data Type: XMLDATA

This extension inherits all of the data types defined in the IODEF model. One data type is added: XMLDATA. An embedded XML data is represented by the XMLDATA data type. This type is defined as the extension to the iodef:ExtensionType [RFC5070], whose dtype attribute is set to "xml."

#### 4.3. Extended Classes

The IODEF Incident element [RFC5070] is summarized below. It is expressed in Unified Modeling Language (UML) syntax as used in the IODEF specification. The UML representation is for illustrative purposes only; elements are specified in XML as defined in Appendix

A.

Incident	
ENUM purpose	<>-----[IncidentID]
STRING ext-purpose	<>--{0..1}-[AlternativeID]
ENUM lang	<>--{0..1}-[RelatedActivity]
ENUM restriction	<>--{0..1}-[DetectTime]
	<>--{0..1}-[StartTime]
	<>--{0..1}-[EndTime]
	<>-----[ReportTime]
	<>--{0..*}-[Description]
	<>--{1..*}-[Assessment]
	<>--{0..*}-[Method]
	<>--[AdditionalData]
	<>--[AttackPattern]
	<>--[Vulnerability]
	<>--[Weakness]
	<>--{1..*}-[Contact]
	<>--{0..*}-[EventData]
	<>--[Flow]
	<>--[System]
	<>--[AdditionalData]
	<>--[Platform]
	<>--[Expectation]
	<>--[Record]
	<>--[RecordData]
	<>--[RecordItem]
	<>--[EventReport]
	<>--{0..1}-[History]
	<>--{0..*}-[AdditionalData]
	<>--[Verification]
	<>--[Remediation]

Figure 1: Incident class

This extension defines the following seven elements.

#### 4.3.1. AttackPattern

An AttackPattern consists of an extension to the Incident.Method.AdditionalData element with a dtype of "xml". The extension describes attack patterns of incidents or events.

It is recommended that Method class SHOULD contain one or more of the extension elements whenever available.

An AttackPattern class is structured as follows.

```

+-----+
| AttackPattern |
+-----+
| ENUM SpecID   | <>--(0..*)-[ RawData ]
| STRING ext-SpecID | <>--(0..*)-[ Reference ]
| STRING AttackPatternID | <>--(0..*)-[ Platform ]
+-----+

```

Figure 2: AttackPattern class

This class has the following attributes.

**SpecID:** REQUIRED. ENUM. A specification's identifier that specifies the format of a structured cybersecurity information. The value should be chosen from the namespaces [XMLNames] listed in the IANA table (Section 4.1) or "private". The value "private" is prepared for conveying RawData based on a format that is not listed in the table. This is usually used for conveying data formatted according to an organization's private schema. When the value "private" is used, ext-SpecID element MUST be used.

**ext-SpecID:** OPTIONAL. STRING. A specification's identifier that specifies the format of a structured cybersecurity information. When this element is used, the value of SpecID element must be "private."

**AttackPatternID:** OPTIONAL. STRING. An identifier of an attack pattern to be reported. This attribute SHOULD be used whenever such identifier is available. Both RawData and Reference elements MUST NOT be used when this attribute is used, while either of them MUST be used if this attribute is omitted.

The AttackPattern class is composed of the following aggregate classes.

**RawData:** Zero or more. XMLDATA. A complete document that is formatted according to the specification and its version identified by the SpecID/ext-SpecID. When this element is used, writers/senders MUST ensure that the namespace specified by SpecID/ext-SpecID and the one used in the RawData element are consistent; if not, the namespace identified by SpecID SHOULD be



preferred, and the inconsistency SHOULD be logged so a human can correct the problem.

**Reference:** Zero or more of iodef:Reference [RFC5070]. This element allows an IODEF document to include a link to a structured information instead of directly embedding it into a RawData element.

**Platform:** Zero or more. An identifier of software platform involved in the specific attack pattern, which is elaborated in Section 4.3.2.

#### 4.3.2. Platform

A Platform identifies a software platform. It is recommended that AttackPattern, Vulnerability, Weakness, and System classes contain this elements whenever available.

A Platform element is structured as follows.

```

+-----+
| Platform |
+-----+
| ENUM SpecID | <>--(0..*)-[ RawData ]
| STRING ext-SpecID | <>--(0..*)-[ Reference ]
| STRING PlatformID |
+-----+

```

Figure 3: Platform class

This class has the following attributes.

**SpecID:** REQUIRED. ENUM. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**ext-SpecID:** OPTIONAL. STRING. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**PlatformID:** OPTIONAL. STRING. An identifier of a platform to be reported. This attribute SHOULD be used whenever such identifier is available. Both RawData and Reference elements MUST NOT be used when this attribute is used, while either of them MUST be used if this attribute is omitted.

This class is composed of the following aggregate classes.

**RawData:** Zero or more. XMLDATA. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

**Reference:** Zero or more of iodef:Reference [RFC5070]. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

#### 4.3.3. Vulnerability

A Vulnerability consists of an extension to the Incident.Method.AdditionalData element with a dtype of "xml". The extension describes the (candidate) vulnerabilities of incidents or events.

It is recommended that Method class SHOULD contain one or more of the extension elements whenever available.

A Vulnerability element is structured as follows.

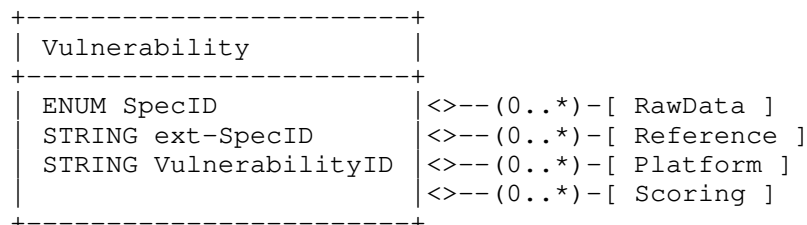


Figure 4: Vulnerability class

This class has the following attributes.

**SpecID:** REQUIRED. ENUM. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**ext-SpecID:** OPTIONAL. STRING. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**VulnerabilityID:** OPTIONAL. STRING. An identifier of a vulnerability to be reported. This attribute SHOULD be used whenever such identifier is available. Both RawData and Reference elements MUST NOT be used when this attribute is used, while either of them MUST be used if this attribute is omitted.

This class is composed of the following aggregate classes.

**RawData:** Zero or more. XMLDATA. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

**Reference:** Zero or more of iodef:Reference [RFC5070]. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

**Platform:** Zero or more. An identifier of software platform affected by the vulnerability, which is elaborated in Section 4.3.2.

**Scoring:** Zero or more. An indicator of the severity of the vulnerability, such as CVSS and CCSS scores, which is elaborated in Section 4.3.4. Some of the structured information may include scores within it. In this case, the Scoring element SHOULD NOT be used since the RawData element contains the scores. If a reader/receiver detects scores in both RawData and Scoring elements and their inconsistency, it SHOULD prefer the scores derived from the RawData element, and SHOULD log the inconsistency so a human can correct the problem.

#### 4.3.4. Scoring

A Scoring class describes the scores of the severity in terms of security. It is recommended that Vulnerability and Weakness classes contain the elements whenever available.

A Scoring class is structured as follows.

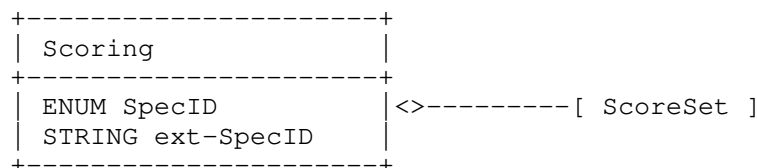


Figure 5: Scoring class

This class has two attributes.

**SpecID:** REQUIRED. ENUM. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**ext-SpecID:** OPTIONAL. STRING. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

This class is composed of an aggregate class.

**ScoreSet:** One. XMLDATA. A complete document that is formatted according to the specification and its version identified by the SpecID/ext-SpecID. This element includes a set of score information. When this element is used, writers/senders MUST ensure that the namespace specified by SpecID/ext-SpecID and the one used in the RawData element are consistent; if not, the namespace identified by SpecID SHOULD be preferred, and the inconsistency SHOULD be logged so a human can correct the problem.

Writers/senders MUST ensure the specification name and version identified by the SpecID are consistent with the contents of the Score; if a reader/receiver detects an inconsistency, it SHOULD prefer the specification name and version derived from the content, and SHOULD log the inconsistency so a human can correct the problem.

#### 4.3.5. Weakness

A Weakness consists of an extension to the Incident.Method.AdditionalData element with a dtype of "xml". The extension describes the weakness types of incidents or events.

It is recommended that Method class SHOULD contain one or more of the extension elements whenever available.

A Weakness element is structured as follows.

+-----+	
Weakness	
+-----+	
ENUM SpecID	<>--(0..*)-[ RawData ]
STRING ext-SpecID	<>--(0..*)-[ Reference ]
STRING WeaknessID	<>--(0..*)-[ Platform ]
	<>--(0..*)-[ Scoring ]
+-----+	

Figure 6: Weakness class

This class has the following attributes.

**SpecID:** REQUIRED. ENUM. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**ext-SpecID:** OPTIONAL. STRING. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**WeaknessID:** OPTIONAL. STRING. An identifier of a weakness to be reported. This attribute SHOULD be used whenever such identifier is available/ Both RawData and Reference elements MUST NOT be used when this attribute is used, while either of them MUST be used if this attribute is omitted.

This class is composed of the following aggregate classes.

**RawData:** Zero or more. XMLDATA. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

**Reference:** Zero or more of iodef:Reference [RFC5070]. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

**Platform:** Zero or more. An identifier of software platform affected by the weakness, which is elaborated in Section 4.3.2.

**Scoring:** Zero or more. An indicator of the severity of the weakness, such as CWSS score, which is elaborated in Section 4.3.4.

#### 4.3.6. EventReport

An EventReport consists of an extension to the Incident.EventData.Record.RecordData.RecordItem element with a dtype of "xml". The extension embeds structured event reports.

It is recommended that RecordItem class SHOULD contain one or more of the extension elements whenever available.

An EventReport element is structured as follows.

+-----+	
EventReport	
+-----+	
ENUM SpecID	<>--(0..*)-[ RawData ]
STRING ext-SpecID	<>--(0..*)-[ Reference ]
STRING EventID	
+-----+	

Figure 7: EventReport class

This class has the following attributes.

SpecID: REQUIRED. ENUM. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

ext-SpecID: OPTIONAL. STRING. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

EventID: OPTIONAL. STRING. An identifier of an event to be reported. This attribute SHOULD be used whenever such identifier is available. Both RawData and Reference elements MUST NOT be used when this attribute is used, while either of them MUST be used if this attribute is omitted.

This class is composed of three aggregate classes.

RawData: Zero or more. XMLDATA. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

Reference: Zero or more of iodef:Reference [RFC5070]. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

This class MUST contain at least one of RawData or Reference elements. Writers/senders MUST ensure the specification name and version identified by the SpecID are consistent with the contents of the RawData; if a reader/receiver detects an inconsistency, it SHOULD prefer the specification name and version derived from the content, and SHOULD log the inconsistency so a human can correct the problem.

#### 4.3.7. Verification

A Verification consists of an extension to the Incident.AdditionalData element with a dtype of "xml". The extension elements describes incident on vefifying incidents.

A Verification class is structured as follows.

```
+-----+
| Verification |
+-----+
| ENUM SpecID   | <>--(0..*)-[ RawData ]
| STRING ext-SpecID | <>--(0..*)-[ Reference ]
| STRING VerificationID |
+-----+
```

Figure 8: Verification class

This class has the following attributes.

SpecID: REQUIRED. ENUM. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

ext-SpecID: OPTIONAL. STRING. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

VerificationID: OPTIONAL. STRING. An identifier of an check item to be reported. This attribute SHOULD be used whenever such identifier is available. Both RawData and Reference elements MUST NOT be used when this attribute is used, while either of them MUST be used if this attribute is omitted.

This class is composed of two aggregate classes.

RawData: Zero or more. XMLDATA. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

Reference: Zero or more of iodef:Reference [RFC5070]. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

This class MUST contain at least either of RawData and Reference elements. Writers/senders MUST ensure the specification name and version identified by the SpecID are consistent with the contents of the RawData; if a reader/receiver detects an inconsistency, it SHOULD prefer the specification name and version derived from the content, and SHOULD log the inconsistency so a human can correct the problem.

#### 4.3.8. Remediation

A Remediation consists of an extension to the Incident.AdditionalData element with a dtype of "xml". The extension elements describes incident remediation information including instructions.

It is recommended that Incident class SHOULD contain one or more of this extension elements whenever available.

A Remediation class is structured as follows.

+-----+   Remediation   +-----+	
ENUM SpecID	<>--(0..*)-[ RawData ]
STRING ext-SpecID	<>--(0..*)-[ Reference ]
String RemediationID	
+-----+	

Figure 9: Remediation class

This class has the following attributes.

**SpecID:** REQUIRED. ENUM. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**ext-SpecID:** OPTIONAL. STRING. The meaning of this attribute is the same as that of the AttackPattern class (Section 4.3.1).

**RemediationID:** OPTIONAL. STRING. An identifier of a remediation information to be reported. This attribute SHOULD be used whenever such identifier is available. Both RawData and Reference elements MUST NOT be used when this attribute is used, while either of them MUST be used if this attribute is omitted.

This class is composed of two aggregate classes.

**RawData:** Zero or more. XMLDATA. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

**Reference:** Zero or more of iodef:Reference [RFC5070]. The meaning of this element is the same as that of the AttackPattern class (Section 4.3.1).

This class MUST contain at least either of RawData and Reference elements. Writers/senders MUST ensure the specification name and version identified by the SpecID are consistent with the contents of the RawData; if a reader/receiver detects an inconsistency, it SHOULD prefer the specification name and version derived from the content, and SHOULD log the inconsistency so a human can correct the problem.

## 5. Mandatory to Implement features

The implementation of this draft MUST be capable of sending and receiving the XML conforming to the specification listed in the initial IANA table described in Section 4.1 without error.



The receiver MUST be capable of validating received XML documents that are embeddeed inside that against their schemata. Note that the receiver can look up the namespace in the IANA table to understand what specifications the embedded XML documents follows.

## 6. Security Considerations

This document specifies a format for encoding a particular class of security incidents appropriate for exchange across organizations. As merely a data representation, it does not directly introduce security issues. However, it is guaranteed that parties exchanging instances of this specification will have certain concerns. For this reason, the underlying message format and transport protocol used MUST ensure the appropriate degree of confidentiality, integrity, and authenticity for the specific environment.

Organizations that exchange data using this document are URGED to develop operating procedures that document the following areas of concern.

### 6.1. Transport-Specific Concerns

The underlying messaging format and protocol used to exchange instances of the IODEF MUST provide appropriate guarantees of confidentiality, integrity, and authenticity. The use of a standardized security protocol is encouraged. The Real-time Inter-network Defense (RID) protocol [RFC6045] and its associated transport binding [RFC6046] provide such security.

The critical security concerns are that these structured information may be falsified or they may become corrupt during transit. In areas where transmission security or secrecy is questionable, the application of a digital signature and/or message encryption on each report will counteract both of these concerns. We expect that each exchanging organization will determine the need, and mechanism, for transport protection.

## 7. IANA Considerations

This document uses URNs to describe XML namespaces and XML schemata [XMLschemaPart1] [XMLschemaPart2] conforming to a registry mechanism described in [RFC3688].

Registration request for the IODEF structured cybersecurity information extension namespace:

URI: urn:ietf:params:xml:ns:iodef-sci-1.0

Registrant Contact: Refer here to the authors' addresses section of the document.

XML: None

Registration request for the IODEF structured cybersecurity information extension XML schema:

URI: urn:ietf:params:xml:schema:iodef-sci-1.0

Registrant Contact: Refer here to the authors' addresses section of the document.

XML: Refer here to the XML Schema in the appendix of the document.

This memo creates the following registry for IANA to manage:

Name of the registry: "IODEF Structured Cyber Security Information Specifications"

Namespace details: A registry entry for a Structured Cyber Security Information Specification (SCI specification) consists of:

Namespace: A URI [RFC3986] that is the XML namespace name used by the registered SCI specification.

Specification Name: A string containing the spelled-out name of the SCI specification in human-readable form.

Reference URI: A list of one or more of the URIs [RFC3986] from which the registered specification can be obtained. The registered specification MUST be readily and publicly available from that URI.

Applicable Classes: A list of one or more of the Extended Classes specified in Section 4.3 of this document. The registered SCI specification MUST only be used with the Extended Classes in the registry entry.

Information that must be provided to assign a new value: The above list of information.

Fields to record in the registry: Namespace/Specification Name/Version/Applicable Classes.

Initial registry contents: none

Allocation Policy: Expert Review [RFC5226] and Specification Required [RFC5226].

The Designated Expert is expected to consult with the mile (Managed Incident Lightweight Exchange) working group or its successor if any such WG exists (e.g., via email to the working group's mailing list). The Designated Expert is expected to retrieve the SCI specification from the provided URI in order to check the public availability of the specification and verify the correctness of the URI. An important responsibility of the Designated Expert is to ensure that the registered Applicable Classes are appropriate for the registered SCI specification.

## 8. Acknowledgment

We would like to acknowledge Mr. David Black from EMC, who kindly provided generous support, especially on the IANA registry issues. We also would like to thank Jon Baker from MITRE, Paul Cichonski from NIST, Panos Kampanakis from CISCO, Robert Martin from MITRE, Kathleen Moriarty from EMC, Lagadec Philippe from NATO, Shuhei Yamaguchi from NICT, Anthony Rutkowski from Yaana Technology, Brian Trammel from CERT/NetSA, and David Waltermire from NIST for their sincere discussion and feedback on this document.

## 9. Appendix I: XML Schema Definition for Extension

The XML Schema describing the elements defined in the Extension Definition section is given here. Each of the examples in Section 11 should be verified to validate against this schema by automated tools.

```
<?xml version="1.0" encoding="UTF-8"?>

<xsd:schema targetNamespace="urn:ietf:params:xml:ns:iodef-sci-1.0"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:iodef-sci="urn:ietf:params:xml:ns:iodef-sci-1.0"
  elementFormDefault="qualified" attributeFormDefault="unqualified">

  <xsd:import namespace="urn:ietf:params:xml:ns:iodef-1.0"
    schemaLocation="iodef_schema.xsd"/>

  <!--
```

```

    schemaLocation="urn:ietf:params:xml:schema:iodef-1.0"/>
-->

<!--=====
== XMLDATA ==
=====-->

<xsd:complexType name="XMLDATA">
  <xsd:complexContent>
    <xsd:restriction base="iodef:ExtensionType">
      <xsd:sequence>
        <xsd:any namespace="##any" processContents="lax" minOccurs="0"
          maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="dtype" type="iodef:dtype-type" use="required" fixed="x
ml"/>
      <xsd:attribute name="ext-dtype" type="xsd:string" use="optional"/>
      <xsd:attribute name="meaning" type="xsd:string"/>
      <xsd:attribute name="formatid" type="xsd:string"/>
      <xsd:attribute name="restriction" type="iodef:restriction-type"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<!--=====
== Scoring Class ==
=====-->

  <xsd:element name="Scoring">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="ScoreSet" type="iodef-sci:XMLDATA"
          minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="SpecID" type="xsd:string" use="required"/>
      <xsd:attribute name="ext-SpecID" type="xsd:string"
        use="optional"/>
    </xsd:complexType>
  </xsd:element>

<!--=====
== AttackPattern Class ==
=====-->

  <xsd:element name="AttackPattern">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:choice>

```

```

        <xsd:element name="RawData" type="iodef-sci:XMLDATA"
          minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="iodef:Reference" minOccurs="0"
          maxOccurs="unbounded"/>
      </xsd:choice>
      <xsd:element ref="iodef-sci:Platform" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="SpecID" type="xsd:string" use="required"/>
    <xsd:attribute name="ext-SpecID" type="xsd:string"
      use="optional"/>
    <xsd:attribute name="AttackPatternID" type="xsd:string"
      use="optional"/>
  </xsd:complexType>
</xsd:element>

<!--=====
== Vulnerability Class                                     ==
=====-->

<xsd:element name="Vulnerability">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:choice>
        <xsd:element name="RawData" type="iodef-sci:XMLDATA"
          minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="iodef:Reference" minOccurs="0"
          maxOccurs="unbounded"/>
      </xsd:choice>
      <xsd:element ref="iodef-sci:Platform" minOccurs="0"
        maxOccurs="unbounded"/>
      <xsd:element ref="iodef-sci:Scoring" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="SpecID" type="xsd:string" use="required"/>
    <xsd:attribute name="ext-SpecID" type="xsd:string"
      use="optional"/>
    <xsd:attribute name="VulnerabilityID" type="xsd:string"
      use="optional"/>
  </xsd:complexType>
</xsd:element>

<!--=====
== Weakness Class                                         ==
=====-->

<xsd:element name="Weakness">
  <xsd:complexType>

```

```

    <xsd:sequence>
      <xsd:choice>
        <xsd:element name="RawData" type="iodef-sci:XMLDATA"
          minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="iodef:Reference" minOccurs="0"
          maxOccurs="unbounded"/>
      </xsd:choice>
      <xsd:element ref="iodef-sci:Platform" minOccurs="0"
        maxOccurs="unbounded"/>
      <xsd:element ref="iodef-sci:Scoring" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="SpecID" type="xsd:string" use="required"/>
    <xsd:attribute name="ext-SpecID" type="xsd:string"
      use="optional"/>
    <xsd:attribute name="WeaknessID" type="xsd:string"
      use="optional"/>
  </xsd:complexType>
</xsd:element>

<!--=====
== Platform Class                                     ==
=====-->

<xsd:element name="Platform">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:choice>
        <xsd:element name="RawData" type="iodef-sci:XMLDATA"
          minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="iodef:Reference" minOccurs="0"
          maxOccurs="unbounded"/>
      </xsd:choice>
    </xsd:sequence>
    <xsd:attribute name="SpecID" type="xsd:string" use="required"/>
    <xsd:attribute name="ext-SpecID" type="xsd:string"
      use="optional"/>
    <xsd:attribute name="PlatformID" type="xsd:string"
      use="optional"/>
  </xsd:complexType>
</xsd:element>

<!--=====
== EventReport Class                                   ==
=====-->

<xsd:element name="EventReport">
  <xsd:complexType>

```

```

    <xsd:sequence>
      <xsd:choice>
        <xsd:element name="RawData" type="iodef-sci:XMLDATA"/>
        <xsd:element ref="iodef:Reference"/>
      </xsd:choice>
    </xsd:sequence>
    <xsd:attribute name="SpecID" type="xsd:string" use="required"/>
    <xsd:attribute name="ext-SpecID" type="xsd:string"
      use="optional"/>
    <xsd:attribute name="EventID" type="xsd:string"
      use="optional"/>
  </xsd:complexType>
</xsd:element>

<!--=====
== Verification Class                                     ==
=====-->

<xsd:element name="Verification">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:choice>
        <xsd:element name="RawData" type="iodef-sci:XMLDATA"/>
        <xsd:element ref="iodef:Reference"/>
      </xsd:choice>
    </xsd:sequence>
    <xsd:attribute name="SpecID" type="xsd:string" use="required"/>
    <xsd:attribute name="ext-SpecID" type="xsd:string"
      use="optional"/>
    <xsd:attribute name="VerificationID" type="xsd:string"
      use="optional"/>
  </xsd:complexType>
</xsd:element>

<!--=====
== Remediation Class                                     ==
=====-->

<xsd:element name="Remediation">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:choice>
        <xsd:element name="RawData" type="iodef-sci:XMLDATA"/>
        <xsd:element ref="iodef:Reference"/>
      </xsd:choice>
    </xsd:sequence>
    <xsd:attribute name="SpecID" type="xsd:string" use="required"/>
    <xsd:attribute name="ext-SpecID" type="xsd:string"

```

```
        use="optional"/>
        <xsd:attribute name="RemediationID" type="xsd:string"
        use="optional"/>
    </xsd:complexType>
</xsd:element>

</xsd:schema>
```

## 10. Appendix II: Candidate Specifications for the IANA Table

This draft defined the structure of the IANA table in Section 4.1. Though the management of the table is up to IANA, this appendix provides candidate entries. Note that OVAL and CVE are registered trademarks, and CAPEC, CCE, CEE, CPE, CWE, CWSS, MAEC, and OCIL are trademarks, of The MITRE Corporation.

### 1. CAPEC 1.6

Namespace: <http://capec.mitre.org/observables>  
Specification Name: Common Attack Pattern Enumeration and Classification  
Version: 1.6  
Reference URI: <http://capec.mitre.org/>  
Applicable Classes: AttackPattern

### 2. CCE 5.0

Namespace: <http://cce.mitre.org>  
Specification Name: Common Configuration Enumeration  
Version: 5.0  
Reference URI: <http://cce.mitre.org/>  
Applicable Classes: Verification

### 3. CCSS 1.0

Namespace: N/A  
Specification Name: Common Configuration Scoring System  
Version: 1.0  
Reference URI: <http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7502>  
Applicable Classes: Scoring



## 4. CEE 1.0 alpha

Namespace: <http://cee.mitre.org>  
Specification Name: Common Event Expression  
Version: 1.0 alpha  
Reference URI: <http://cee.mitre.org/>  
Applicable Classes: EventReport

## 5. CPE 2.3 Language

Namespace: <http://cpe.mitre.org/language/2.0>  
Specification Name: Common Platform Enumeration Reference  
Version: 2.3  
Reference URI: <http://scap.nist.gov/specifications/cpe/>,  
[http://csrc.nist.gov/publications/PubsNISTIRs.html](http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7695)  
[#NIST-IR-7695](http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7695)  
Applicable Classes: Platform

## 6. CPE 2.3 Dictionary

Namespace: <http://cpe.mitre.org/dictionary/2.0>  
Specification Name: Common Platform Enumeration Dictionary  
Version: 2.3  
Reference URI: <http://scap.nist.gov/specifications/cpe/>,  
[http://csrc.nist.gov/publications/PubsNISTIRs.html](http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7697)  
[#NIST-IR-7697](http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7697)  
Applicable Classes: Platform

## 7. CVE 1.0

Namespace: <http://cve.mitre.org/cve/downloads/1.0>  
Specification Name: Common Vulnerability and Exposures  
Version: 1.0  
Reference URI: <http://cve.mitre.org/>  
Applicable Classes: Vulnerability

## 8. CVRF 1.0

Namespace: <http://www.icas.org/CVRF/schema/cvrf/1.0>  
Specification Name: Common Vulnerability Reporting Format  
Version: 1.0  
Reference URI: <http://www.icas.org/cvrf>  
Applicable Classes: Vulnerability

#### 9. CVSS 2.0

Namespace: <http://scap.nist.gov/schema/cvss-v2/1.0>  
Specification Name: Common Vulnerability Scoring System  
Version: 2  
Reference URI: <http://www.first.org/cvss>  
Applicable Classes: Scoring

#### 10. CWE 5.0

Namespace: N/A  
Specification Name: Common Weakness Enumeration  
Version: 5.1  
Reference URI: <http://cwe.mitre.org/>  
Applicable Classes: Weakness

#### 11. CWSS 0.8

Namespace: N/A  
Specification Name: Common Weakness Scoring System  
Version: 0.8  
Reference URI: <http://cwe.mitre.org/cwss/>  
Applicable Classes: Scoring

#### 12. MAEC 2.0

Namespace: <http://maec.mitre.org/XMLSchema/maec-core-2>  
Specification Name: Malware Attribute Enumeration and Characterization  
Version: 2.0  
Reference URI: <http://maec.mitre.org/>  
Applicable Classes: EventReport, AttackPattern

## 13. OCIL 2.0

Namespace: <http://scap.nist.gov/schema/ocil/2.0>  
Specification Name: Open Checklist Interactive Language  
Version: 2.0  
Reference URI: <http://scap.nist.gov/specifications/ocil/>,  
[http://csrc.nist.gov/publications/PubsNISTIRs.html](http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7692)  
#NIST-IR-7692  
Applicable Classes: Verification

## 14. OVAL 5.10.1 Definitions

Namespace: <http://oval.mitre.org/XMLSchema/oval-definitions-5>  
Specification Name: Open Vulnerability and Assessment Language  
Version: 5.10.1  
Reference URI: <http://oval.mitre.org/>  
Applicable Classes: Verification

## 15. OVAL 5.10.1 Results

Namespace: <http://oval.mitre.org/XMLSchema/oval-results-5>  
Specification Name: Open Vulnerability and Assessment Language  
Version: 5.10.1  
Reference URI: <http://oval.mitre.org/>  
Applicable Classes: Verification

## 16. OVAL 5.10.1 Common

Namespace: <http://oval.mitre.org/XMLSchema/oval-common-5>  
Specification Name: Open Vulnerability and Assessment Language  
Version: 5.10.1  
Reference URI: <http://oval.mitre.org/>  
Applicable Classes: Verification

## 17. XCCDF 1.2

Namespace: <http://checklists.nist.gov/xccdf/1.2>  
Specification Name: Extensible Configuration Checklist Description Format  
Version: 1.2  
Reference URI: <http://scap.nist.gov/specifications/xccdf/>,  
[http://csrc.nist.gov/publications/PubsNISTIRs.html](http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7275-r4)  
#NIST-IR-7275-r4  
Applicable Classes: Verification

## 11. Appendix III: An XML Example

This section provides an example of an incident encoded in the IODEF. This does not necessarily represent the only way to encode a particular incident. This example reports an attack to a CSIRT and is extended from the example described in [RFC5070]. It uses identifiers whose dictionary follows CVE 1.0 schema, and it embeds XML following CEE 0.6.

```
<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="1.00" lang="en"
  xmlns="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:iodef-sci="urn:ietf:params:xml:ns:iodef-sci-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Incident purpose="reporting">
    <IncidentID name="csirt.example.com">189493</IncidentID>
    <ReportTime>2001-09-13T23:19:24+00:00</ReportTime>
    <Description>Incident report in company xx</Description>
    <Assessment>
      <Impact completion="failed" type="admin"/>
    </Assessment>
    <Method>
      <Description>An identifier of a vulnerability is embedded</Description>
      <AdditionalData dtype="xml">
        <iodef-sci:Vulnerability
          SpecID="http://cve.mitre.org/cve/downloads/1.0"
          VulnerabilityID="CVE-2010-0483"/>
      </AdditionalData>
    </Method>
    <Contact role="creator" type="organization">
      <ContactName>Example.com CSIRT</ContactName>
      <RegistryHandle registry="arin">example-com</RegistryHandle>
      <Email>contact@csirt.example.com</Email>
    </Contact>
    <EventData>
      <Flow>
```

```
<System category="source">
  <Node>
    <Address category="ipv4-addr">192.0.2.200</Address>
    <Counter type="event">57</Counter>
  </Node>
</System>
<System category="target">
  <Node>
    <Address category="ipv4-net">192.0.2.16/28</Address>
  </Node>
  <Service ip_protocol="6">
    <Port>80</Port>
  </Service>
  <AdditionalData dtype="xml">
    <iodef-sci:Platform SpecID="http://cpe.mitre.org/dictionary/2.0"
      PlatformID="cpe://microsoft:windows:xp:pro:sp2"/>
  </AdditionalData>
</System>
</Flow>
<Expectation action="block-host" />
<Expectation action="other"/>
<!-- <RecordItem> has an excerpt from a log -->
<Record>
  <RecordData>
    <DateTime>2001-09-13T18:11:21+02:00</DateTime>
    <Description>a Web-server event record</Description>
    <RecordItem dtype="xml">
      <iodef-sci:EventReport SpecID="http://cee.mitre.org">
        <iodef-sci:RawData dtype="xml">
          <cee:cee xmlns="http://cee.mitre.org/1.0/profile/"
            xmlns:cee="http://cee.mitre.org/1.0/">
            <cee:event>
              <host>system.example.com</host>
              <pname>auth</pname>
              <time>2011-12-20T12:38:05.123456-05:00</time>
              <appname>application</appname>
              <pid>123</pid>
              <sev>10</sev>
              <action>login</action>
              <domain>app</domain>
              <object>account</object>
              <service>web</service>
              <status>success</status>
            </cee:event>
          </cee:cee>
        </iodef-sci:RawData>
      </iodef-sci:EventReport>
    </RecordItem>
```

```
</RecordData>
</Record>
</EventData>
<History>
  <!-- Contact was previously made with the source network owner -->
  <HistoryItem action="contact-source-site">
    <DateTime>2001-09-14T08:19:01+00:00</DateTime>
    <Description>Notification sent to
      constituency-contact@192.0.2.200</Description>
  </HistoryItem>
</History>
</Incident>
</IODEF-Document>
```

## 12. References

### 12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.
- [RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, December 2007.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC6045] Moriarty, K., "Real-time Inter-network Defense (RID)", RFC 6045, November 2010.
- [RFC6046] Moriarty, K. and B. Trammell, "Transport of Real-time Inter-network Defense (RID) Messages", RFC 6046, November 2010.
- [MMDEF] IEEE ICSG Malware Metadata Exchange Format Working Group, "Malware Metadata Exchange Format".
- [XML1.0] Bray, T., Maler, E., Paoli, J., Sperberg-McQueen, C., and F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", W3C Recommendation, November 2008.

## [XMLSchemaPart1]

Thompson, H., Beech, D., Maloney, M., and N. Mendelsohn,  
"XML Schema Part 1: Structures Second Edition",  
W3C Recommendation, October 2004.

## [XMLSchemaPart2]

Biron, P. and A. Malhotra, "XML Schema Part 2: Datatypes  
Second Edition", W3C Recommendation, October 2004.

## [XMLNames]

Bray, T., Hollander, D., Layman, A., Tobin, R., and H.  
Thomson, "Namespaces in XML (Third Edition)",  
W3C Recommendation, December 2009.

## 12.2. Informative References

- [RFC3339] Klyne, G., Ed. and C. Newman, "Date and Time on the  
Internet: Timestamps", RFC 3339, July 2002.
- [RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC  
Text on Security Considerations", BCP 72, RFC 3552,  
July 2003.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,  
January 2004.
- [RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322,  
October 2008.
- [RFC6116] Bradner, S., Conroy, L., and K. Fujiwara, "The E.164 to  
Uniform Resource Identifiers (URI) Dynamic Delegation  
Discovery System (DDDS) Application (ENUM)", RFC 6116,  
March 2011.
- [CAPEC] The MITRE Corporation, "Common Attack Pattern Enumeration  
and Classification (CAPEC)".
- [CCE] The MITRE Corporation, "Common Configuration Enumeration  
(CCE)".
- [CCSS] Scarfone, K. and P. Mell, "The Common Configuration  
Scoring System (CCSS)", NIST Interagency Report 7502,  
December 2010.
- [CEE] The MITRE Corporation, "Common Event Expression (CEE)".
- [CPE] National Institute of Standards and Technology, "Common  
Platform Enumeration", June 2011.

- [CVE] The MITRE Corporation, "Common Vulnerability and Exposures (CVE)".
- [CVRF] ICASI, "Common Vulnerability Reporting Framework (CVRF)".
- [CVSS] Peter Mell, Karen Scarfone, and Sasha Romanosky, "The Common Vulnerability Scoring System (CVSS) and Its Applicability to Federal Agency Systems".
- [CWE] The MITRE Corporation, "Common Weakness Enumeration (CWE)".
- [CWSS] The MITRE Corporation, "Common Weakness Scoring System (CWSS)".
- [OCIL] David Waltermire and Karen Scarfone and Maria Casipe, "The Open Checklist Interactive Language (OCIL) Version 2.0", April 2011.
- [OVAL] The MITRE Corporation, "Open Vulnerability and Assessment Language (OVAL)".
- [SCAP] Waltermire, D., Quinn, S., Scarfone, K., and A. Halbardier, "The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.2", NIST Special Publication 800-126 Revision 2, September 2011.
- [XCCDF] David Waltermire and Charles Schmidt and Karen Scarfone and Neal Ziring, "Specification for the Extensible Configuration Checklist Description Format (XCCDF) version 1.2 (DRAFT)", July 2011.

#### Authors' Addresses

Takeshi Takahashi  
National Institute of Information and Communications Technology  
4-2-1 Nukui-Kitamachi Koganei  
184-8795 Tokyo  
Japan

Phone: +80 423 27 5862  
Email: takeshi\_takahashi@nict.go.jp



Kent Landfield  
McAfee, Inc  
5000 Headquarters Drive  
Plano, TX 75024  
USA

Email: Kent\_Landfield@McAfee.com

Thomas Millar  
US Department of Homeland Security, NPPD/CS&C/NCSD/US-CERT  
245 Murray Lane SW, Building 410, MS #732  
Washington, DC 20598  
USA

Phone: +1 888 282 0870  
Email: thomas.millar@us-cert.gov

Youki Kadobayashi  
Nara Institute of Science and Technology  
8916-5 Takayama, Ikoma  
630-0192 Nara  
Japan

Email: youki-k@is.aist-nara.ac.jp



INTERNET-DRAFT  
Intended Status: Standards Track  
Expires: June 17, 2013

Adam W. Montville  
(Tripwire, Inc.)  
December 14, 2012

IODEF Enumeration Reference Format  
draft-montville-mile-enum-reference-format-02

Abstract

The Incident Object Description Exchange Format [IODEF] provides a Reference class used to reference external entities (such as enumeration identifiers). However, the method of external entity identification has been left unstructured. This document describes a method to provide structure for referencing external entities for the [IODEF] Reference class.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at  
<http://www.ietf.org/shadow.html>

Copyright and License Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal

Provisions Relating to IETF Documents  
(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1	Introduction . . . . .	3
1.1	Terminology . . . . .	3
2.	Referencing External Enumerations . . . . .	3
3	Security Considerations . . . . .	6
4	IANA Considerations . . . . .	6
5	References . . . . .	7
5.1	Normative References . . . . .	7
5.2	Informative References . . . . .	7
	Authors' Addresses . . . . .	7

## 1 Introduction

There is an identified need to specify a format to include relevant enumeration values in an IODEF document. It is anticipated that this requirement will exist in other standardization efforts within several IETF Working Groups, but the scope of this document pertains solely to [IODEF].

### 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 RFC 2119 [RFC2119].

## 2. Referencing External Enumerations

The need is to place enumeration identifiers and their references in [IODEF]'s Reference class. There are several ways to accomplish this goal, but one that seems the most appropriate at this point is to require a specific format for the ReferenceName string of the [IODEF] Reference class, such that an IANA table can be used to catalog a variety of reference types.

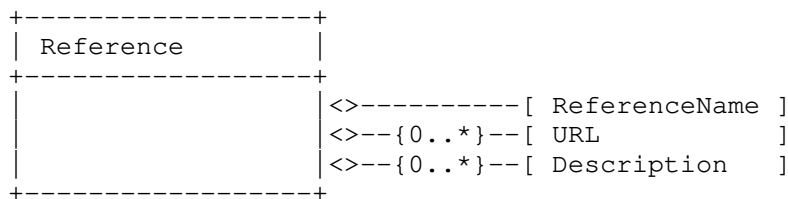


FIGURE 1: [IODEF] Reference Class

Per [IODEF] the ReferenceName is of type ML\_STRING. This becomes problematic when specific references, especially enumerations such as CEE, CVE, CCE, and so on, are referenced - how is an implementer to know which type of reference this is, and thus how to parse it? One solution, presented here, is to require that ReferenceName follow a particular format.

### 2.1 Reference Name Format

The format of the ReferenceName MUST follow the form of

id\_type:version:id

Where id\_type is an IANA-registered type having the form

<Abbreviation>

And where version is an IANA-registered type having the form

<Version>

And where id is the actual enumeration identifier string.

The IANA Considerations section of this document provides details for <Abbreviation> and <Version>. This format allows the <Version> to be associated with the id rather than the id\_type. By requiring that a specific type and version be associated with the identifier, an implementer can look up the type in an IANA table to understand exactly what the identifier in ReferenceName is and how s/he may expect that identifier to be structured.

## 2.2 Reference Example

The operation of this method can be described using a fictitious example. Assume a Reference class as described in the Section 2 introduction and an enumeration of formatted strings used to identify Concept X. Then, the string format of Concept X Identifiers would be registered with IANA (see Section 4), such that implementations of the Reference class understand how to handle the formatted string.

```
<Reference>
  <ReferenceName>CXI:1.0:CXI-1234-XYZ</ReferenceName>
  <URL>http://cxi.example.com</URL>
  <Description>Foo</Description>
</Reference>
```

Information in the IANA table (see Section 4) would include:

```
Full Name: Concept X Identifier
Abbreviation: CXI
Version: 1.0
Specification URI: http://cxi.example.com/spec_url
```

## 2.3 Reference Method Applicability

While the scope of this document pertains to [IODEF], it should be readily apparent that any standard needing to reference an enumeration identified by a specially formatted string can use this method of providing structure after the standard has been published. In effect, this method provides a standardized interface for enumerations, thus allowing a loose coupling between

a given standard and the enumeration identifiers it needs to reference now and in the future.

### 3 Security Considerations

None.

### 4 IANA Considerations

This document specifies an identifier format for the [IODEF] ReferenceName string of the Reference class.

Registration request for the IODEF Enumeration Reference Format:

Name of the Registry: "Enumeration Reference Type Identifiers"

The registry is intended to enable enumeration value additions to attributes of a given reference class of an IETF standard, for example, the Reference class of the IODEF schema. Note that certain name requests should not be permitted as either Full Name or Abbreviation entries for the requested IANA table. For example, the following list should not be permitted: foo, bar, example.com. It is anticipated that the Expert Review process will flag any additional undesired Full Name or Abbreviation issues.

Fields to record in the registry:

Full Name: The full name of the enumeration as a string

Abbreviation: The abbreviation of the enumeration as a string, e.g. a short initialization is encouraged

Version: The version of the enumeration as a string, e.g. dot-separated numbers are a good idea

Specification URI: A list of one or more URIs [RFC3986] from which the registered specification can be obtained. The registered specification MUST be readily and publicly available from that URI.

Initial registry contents: None.

Allocation Policy: Expert Review [RFC5226] and Specification Required [RFC5226]

The Designated Expert is expected to consult with the MILE (Managed Incident Lightweight Exchange) working group or its successor if any such WG exists (e.g., via email to the working group's mailing list). The Designated Expert is expected to review the request and validate



the appropriateness of the enumeration for the attribute. If a specification is associated with the request, it MUST be reviewed by the Designated Expert.

## 5 References

### 5.1 Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [IODEF] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, December 2007.
- [3986] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

### 5.2 Informative References

None.

## Authors' Addresses

Adam W. Montville  
Tripwire, Inc.  
101 SW Main Street  
Suite 1500  
Portland, OR 97204

EMail: [amontville@tripwire.com](mailto:amontville@tripwire.com)

Extended Incident Handling Working  
Group  
Internet-Draft  
Obsoletes: 5070 (if approved)  
Intended status: Informational  
Expires: August 21, 2013

R. Danyliw  
CERT  
P. Stoecker  
RSA  
February 17, 2013

The Incident Object Description Exchange Format  
draft-stoecker-danyliw-rfc5070bis-00

Abstract

The Incident Object Description Exchange Format (IODEF) defines a data representation that provides a framework for sharing information commonly exchanged by Computer Security Incident Response Teams (CSIRTs) about computer security incidents. This document describes the information model for the IODEF and provides an associated data model specified with XML Schema.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 21, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	4
1.1. Changes from 5070 . . . . .	5
1.2. Terminology . . . . .	6
1.3. Notations . . . . .	6
1.4. About the IODEF Data Model . . . . .	6
1.5. About the IODEF Implementation . . . . .	7
2. IODEF Data Types . . . . .	7
2.1. Integers . . . . .	7
2.2. Real Numbers . . . . .	7
2.3. Characters and Strings . . . . .	8
2.4. Multilingual Strings . . . . .	8
2.5. Bytes . . . . .	8
2.6. Hexadecimal Bytes . . . . .	8
2.7. Enumerated Types . . . . .	8
2.8. Date-Time Strings . . . . .	9
2.9. Timezone String . . . . .	9
2.10. Port Lists . . . . .	9
2.11. Postal Address . . . . .	9
2.12. Person or Organization . . . . .	9
2.13. Telephone and Fax Numbers . . . . .	10
2.14. Email String . . . . .	10
2.15. Uniform Resource Locator strings . . . . .	10
3. The IODEF Data Model . . . . .	10
3.1. IODEF-Document Class . . . . .	10
3.2. Incident Class . . . . .	11
3.3. IncidentID Class . . . . .	14
3.4. AlternativeID Class . . . . .	15
3.5. RelatedActivity Class . . . . .	16
3.6. AdditionalData Class . . . . .	16
3.7. Contact Class . . . . .	19
3.7.1. RegistryHandle Class . . . . .	21
3.7.2. PostalAddress Class . . . . .	22
3.7.3. Email Class . . . . .	23
3.7.4. Telephone and Fax Classes . . . . .	23
3.8. Time Classes . . . . .	24
3.8.1. StartTime . . . . .	24
3.8.2. EndTime . . . . .	24
3.8.3. DetectTime . . . . .	24
3.8.4. ReportTime . . . . .	25
3.8.5. DateTime . . . . .	25
3.9. Method Class . . . . .	25
3.9.1. Reference Class . . . . .	26

3.10. Assessment Class . . . . .	27
3.10.1. Impact Class . . . . .	28
3.10.2. TimeImpact Class . . . . .	30
3.10.3. MonetaryImpact Class . . . . .	32
3.10.4. Confidence Class . . . . .	33
3.11. History Class . . . . .	34
3.11.1. HistoryItem Class . . . . .	34
3.12. EventData Class . . . . .	36
3.12.1. Relating the Incident and EventData Classes . . . . .	38
3.12.2. Cardinality of EventData . . . . .	38
3.13. Expectation Class . . . . .	39
3.14. Flow Class . . . . .	41
3.15. System Class . . . . .	42
3.16. Node Class . . . . .	44
3.16.1. Counter Class . . . . .	45
3.16.2. Address Class . . . . .	47
3.16.3. NodeRole Class . . . . .	48
3.17. Service Class . . . . .	50
3.17.1. Application Class . . . . .	51
3.18. OperatingSystem Class . . . . .	53
3.19. Record Class . . . . .	53
3.19.1. RecordData Class . . . . .	53
3.19.2. RecordPattern Class . . . . .	55
3.19.3. RecordItem Class . . . . .	56
4. Processing Considerations . . . . .	56
4.1. Encoding . . . . .	57
4.2. IODEF Namespace . . . . .	57
4.3. Validation . . . . .	57
5. Extending the IODEF . . . . .	58
5.1. Extending the Enumerated Values of Attributes . . . . .	59
5.2. Extending Classes . . . . .	59
6. Internationalization Issues . . . . .	61
7. Examples . . . . .	62
7.1. Worm . . . . .	62
7.2. Reconnaissance . . . . .	63
7.3. Bot-Net Reporting . . . . .	65
7.4. Watch List . . . . .	67
8. The IODEF Schema . . . . .	68
9. Security Considerations . . . . .	94
10. IANA Considerations . . . . .	95
11. Acknowledgments . . . . .	96
12. References . . . . .	96
12.1. Normative References . . . . .	96
12.2. Informative References . . . . .	97

## 1. Introduction

Organizations require help from other parties to mitigate malicious activity targeting their network and to gain insight into potential threats. This coordination might entail working with an ISP to filter attack traffic, contacting a remote site to take down a bot-network, or sharing watch-lists of known malicious IP addresses in a consortium.

The Incident Object Description Exchange Format (IODEF) is a format for representing computer security information commonly exchanged between Computer Security Incident Response Teams (CSIRTs). It provides an XML representation for conveying incident information across administrative domains between parties that have an operational responsibility of remediation or a watch-and-warning over a defined constituency. The data model encodes information about hosts, networks, and the services running on these systems; attack methodology and associated forensic evidence; impact of the activity; and limited approaches for documenting workflow.

The overriding purpose of the IODEF is to enhance the operational capabilities of CSIRTs. Community adoption of the IODEF provides an improved ability to resolve incidents and convey situational awareness by simplifying collaboration and data sharing. This structured format provided by the IODEF allows for:

- o increased automation in processing of incident data, since the resources of security analysts to parse free-form textual documents will be reduced;
- o decreased effort in normalizing similar data (even when highly structured) from different sources; and
- o a common format on which to build interoperable tools for incident handling and subsequent analysis, specifically when data comes from multiple constituencies.

Coordinating with other CSIRTs is not strictly a technical problem. There are numerous procedural, trust, and legal considerations that might prevent an organization from sharing information. The IODEF does not attempt to address them. However, operational implementations of the IODEF will need to consider this broader context.

Sections 3 and 8 specify the IODEF data model with text and an XML schema. The types used by the data model are covered in Section 2. Processing considerations, the handling of extensions, and internationalization issues related to the data model are covered in

Sections 4, 5, and 6, respectively. Examples are listed in Section 7. Section 1 provides the background for the IODEF, and Section 9 documents the security considerations.

#### 1.1. Changes from 5070

- o This document contains changes with respect to its predecessor RFC5070
- o All of the Errata that has been submitted at RFC5070 Errata has been implemented with the exception of #17 which instead of changing the UML, the schema was changed.
- o Addition of xmlns:ds and import of same namespace. This is to use the digital signature hash inclusion of a file by referencing the existing standard as was done in RFC5901, RFC3275 is the reference, see RFC5901 section 5.9.5.2">
- o New indicator uid and set id values in the schema. The purpose of the proposed changes is to include commonly shared indicators in the base IODEF schema. This class will contain indicators from the list below that are not represented elsewhere in the schema. IODEF extensions or embedded schemas via the SCI classes will be required to include additional data types. A table could be maintained through IANA to extend or change this class in between IODEF revisions.
- o RFC5901 provides a method to include an entire email, the following included indicators are ones commonly used when you do not need the entire email
- o The following are in the Service class: Email Address, Email Subject, and X-Mailer
- o The following are in the Record class: File Name, File Hash (5.9.5.2 - using ds:reference), and WindowsRegistryKey (using method from RFC5901
- o The following are now in the Node class as a proposed location: URL
- o HTTPUserAgent is included as a SoftwareType - HTTP User Agent String
- o The following are already represented elsewhere in the schema (Node): IP address, Network CIDR / ASN, Host Name, and Domain Name (additional options for RFC5901 were not included in this revision - can include point-in-time dig info)

## 1.2. 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 [6].

Definitions for some of the common computer security-related terminology used in this document can be found in Section 2 of [16].

## 1.3. Notations

The normative IODEF data model is specified with the text in Section 3 and the XML schema in Section 8. To help in the understanding of the data elements, Section 3 also depicts the underlying information model using Unified Modeling Language (UML). This abstract presentation of the IODEF is not normative.

For clarity in this document, the term "XML document" will be used when referring generically to any instance of an XML document. The term "IODEF document" will be used to refer to specific elements and attributes of the IODEF schema. The terms "class" and "element" will be used interchangeably to reference either the corresponding data element in the information or data models, respectively.

## 1.4. About the IODEF Data Model

The IODEF data model is a data representation that provides a framework for sharing information commonly exchanged by CSIRTs about computer security incidents. A number of considerations were made in the design of the data model.

- o The data model serves as a transport format. Therefore, its specific representation is not the optimal representation for on-disk storage, long-term archiving, or in-memory processing.
- o As there is no precise widely agreed upon definition for an incident, the data model does not attempt to dictate one through its implementation. Rather, a broad understanding is assumed in the IODEF that is flexible enough to encompass most operators.
- o Describing an incident for all definitions would require an extremely complex data model. Therefore, the IODEF only intends to be a framework to convey commonly exchanged incident information. It ensures that there are ample mechanisms for extensibility to support organization-specific information, and techniques to reference information kept outside of the explicit data model.

- o The domain of security analysis is not fully standardized and must rely on free-form textual descriptions. The IODEF attempts to strike a balance between supporting this free-form content, while still allowing automated processing of incident information.
- o The IODEF is only one of several security relevant data representations being standardized. Attempts were made to ensure they were complimentary. The data model of the Intrusion Detection Message Exchange Format [17] influenced the design of the IODEF.

Further discussion of the desirable properties for the IODEF can be found in the Requirements for the Format for Incident Information Exchange (FINE) [16].

### 1.5. About the IODEF Implementation

The IODEF implementation is specified as an Extensible Markup Language (XML) [1] Schema [2] in Section 8.

Implementing the IODEF in XML provides numerous advantages. Its extensibility makes it ideal for specifying a data encoding framework that supports various character encodings. Likewise, the abundance of related technologies (e.g., XSL, XPath, XML-Signature) makes for simplified manipulation. However, XML is fundamentally a text representation, which makes it inherently inefficient when binary data must be embedded or large volumes of data must be exchanged.

## 2. IODEF Data Types

The various data elements of the IODEF data model are typed. This section discusses these data types. When possible, native Schema data types were adopted, but for more complicated formats, regular expressions (see Appendix F of [3]) or external standards were used.

### 2.1. Integers

An integer is represented by the INTEGER data type. Integer data MUST be encoded in Base 10.

The INTEGER data type is implemented as an "xs:integer" [3] in the schema.

### 2.2. Real Numbers

Real (floating-point) attributes are represented by the REAL data type. Real data MUST be encoded in Base 10.



The REAL data type is implemented as an "xs:float" [3] in the schema.

### 2.3. Characters and Strings

A single character is represented by the CHARACTER data type. A character string is represented by the STRING data type. Special characters must be encoded using entity references. See Section 4.1.

The CHARACTER and STRING data types are implemented as an "xs:string" [3] in the schema.

### 2.4. Multilingual Strings

STRING data that represents multi-character attributes in a language different than the default encoding of the document is of the ML\_STRING data type.

The ML\_STRING data type is implemented as an "iodef:MLStringType" in the schema.

### 2.5. Bytes

A binary octet is represented by the BYTE data type. A sequence of binary octets is represented by the BYTE[] data type. These octets are encoded using base64.

The BYTE data type is implemented as an "xs:base64Binary" [3] in the schema.

### 2.6. Hexadecimal Bytes

A binary octet is represented by the HEXBIN (and HEXBIN[]) data type. This octet is encoded as a character tuple consisting of two hexadecimal digits.

The HEXBIN data type is implemented as an "xs:hexBinary" [3] in the schema.

### 2.7. Enumerated Types

Enumerated types are represented by the ENUM data type, and consist of an ordered list of acceptable values. Each value has a representative keyword. Within the IODEF schema, the enumerated type keywords are used as attribute values.

The ENUM data type is implemented as a series of "xs:NMTOKEN" in the schema.

## 2.8. Date-Time Strings

Date-time strings are represented by the DATETIME data type. Each date-time string identifies a particular instant in time; ranges are not supported.

Date-time strings are formatted according to a subset of ISO 8601: 2000 [13] documented in RFC 3339 [12].

The DATETIME data type is implemented as an "xs:dateTime" [3] in the schema.

## 2.9. Timezone String

A timezone offset from UTC is represented by the TIMEZONE data type. It is formatted according to the following regular expression: "Z|[\+|-](0[0-9]|1[0-4]):[0-5][0-9]".

The TIMEZONE data type is implemented as an "xs:string" with a regular expression constraint in the schema. This regular expression is identical to the timezone representation implemented in an "xs:dateTime".

## 2.10. Port Lists

A list of network ports are represented by the PORTLIST data type. A PORTLIST consists of a comma-separated list of numbers and ranges (N-M means ports N through M, inclusive). It is formatted according to the following regular expression: "\d+(\-\d+)?(,\d+(\-\d+)?)\*". For example, "2,5-15,30,32,40-50,55-60".

The PORTLIST data type is implemented as an "xs:string" with a regular expression constraint in the schema.

## 2.11. Postal Address

A postal address is represented by the POSTAL data type. This data type is an ML\_STRING whose format is documented in Section 2.23 of RFC 4519 [10]. It defines a postal address as a free-form multi-line string separated by the "\$" character.

The POSTAL data type is implemented as an "xs:string" in the schema.

## 2.12. Person or Organization

The name of an individual or organization is represented by the NAME data type. This data type is an ML\_STRING whose format is documented in Section 2.3 of RFC 4519 [10].

The NAME data type is implemented as an "xs:string" in the schema.

### 2.13. Telephone and Fax Numbers

A telephone or fax number is represented by the PHONE data type. The format of the PHONE data type is documented in Section 2.35 of RFC 4519 [10].

The PHONE data type is implemented as an "xs:string" in the schema.

### 2.14. Email String

An email address is represented by the EMAIL data type. The format of the EMAIL data type is documented in Section 3.4.1 RFC 2822 [11]

The EMAIL data type is implemented as an "xs:string" in the schema.

### 2.15. Uniform Resource Locator strings

A uniform resource locator (URL) is represented by the URL data type. The format of the URL data type is documented in RFC 2396 [8].

The URL data type is implemented as an "xs:anyURI" in the schema.

## 3. The IODEF Data Model

In this section, the individual components of the IODEF data model will be discussed in detail. For each class, the semantics will be described and the relationship with other classes will be depicted with UML. When necessary, specific comments will be made about corresponding definition in the schema in Section 8

### 3.1. IODEF-Document Class

The IODEF-Document class is the top level class in the IODEF data model. All IODEF documents are an instance of this class.

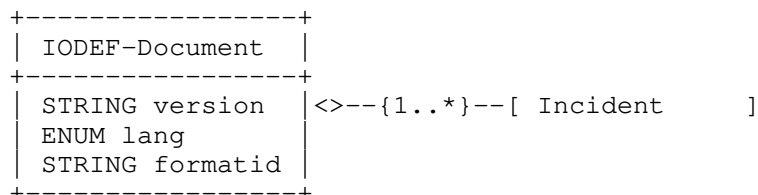


Figure 1: IODEF-Document Class

The aggregate class that constitute IODEF-Document is:

#### Incident

One or more. The information related to a single incident.

The IODEF-Document class has three attributes:

#### version

Required. STRING. The IODEF specification version number to which this IODEF document conforms. The value of this attribute MUST be "1.00"

#### lang

Required. ENUM. A valid language code per RFC 4646 [7] constrained by the definition of "xs:language". The interpretation of this code is described in Section 6.

#### formatid

Optional. STRING. A free-form string to convey processing instructions to the recipient of the document. Its semantics must be negotiated out-of-band.

### 3.2. Incident Class

Every incident is represented by an instance of the Incident class. This class provides a standardized representation for commonly exchanged incident data.

Incident	
ENUM purpose	<>-----[ IncidentID ]
STRING ext-purpose	<>--{0..1}--[ AlternativeID ]
ENUM lang	<>--{0..1}--[ RelatedActivity ]
ENUM restriction	<>--{0..1}--[ DetectTime ]
	<>--{0..1}--[ StartTime ]
	<>--{0..1}--[ EndTime ]
	<>-----[ ReportTime ]
	<>--{0..*}--[ Description ]
	<>--{1..*}--[ Assessment ]
	<>--{0..*}--[ Method ]
	<>--{1..*}--[ Contact ]
	<>--{0..*}--[ EventData ]
	<>--{0..1}--[ History ]
	<>--{0..*}--[ AdditionalData ]

Figure 2: The Incident Class

The aggregate classes that constitute Incident are:

IncidentID

One. An incident tracking number assigned to this incident by the CSIRT that generated the IODEF document.

AlternativeID

Zero or one. The incident tracking numbers used by other CSIRTs to refer to the incident described in the document.

RelatedActivity

Zero or one. The incident tracking numbers of related incidents.

DetectTime

Zero or one. The time the incident was first detected.

StartTime

Zero or one. The time the incident started.

EndTime

Zero or one. The time the incident ended.

ReportTime

One. The time the incident was reported.

Description

Zero or more. ML\_STRING. A free-form textual description of the incident.

Assessment

One or more. A characterization of the impact of the incident.

Method

Zero or more. The techniques used by the intruder in the incident.

Contact

One or more. Contact information for the parties involved in the incident.

EventData

Zero or more. Description of the events comprising the incident.

#### History

Zero or one. A log of significant events or actions that occurred during the course of handling the incident.

#### AdditionalData

Zero or more. Mechanism by which to extend the data model.

The Incident class has five attributes:

#### purpose

Required. ENUM. The purpose attribute represents the reason why the IODEF document was created. It is closely related to the Expectation class (Section 3.13). This attribute is defined as an enumerated list:

1. traceback. The document was sent for trace-back purposes.
2. mitigation. The document was sent to request aid in mitigating the described activity.
3. reporting. The document was sent to comply with reporting requirements.
4. other. The document was sent for purposes specified in the Expectation class.
5. ext-value. An escape value used to extend this attribute. See Section 5.1.

#### ext-purpose

Optional. STRING. A means by which to extend the purpose attribute. See Section 5.1.

#### lang

Optional. ENUM. A valid language code per RFC 4646 [7] constrained by the definition of "xs:language". The interpretation of this code is described in Section 6.

#### restriction

Optional. ENUM. This attribute indicates the disclosure guidelines to which the sender expects the recipient to adhere for the information represented in this class and its children. This guideline provides no security since there are no specified technical means to ensure that the recipient of the document handles the information as the sender requested.

The value of this attribute is logically inherited by the children of this class. That is to say, the disclosure rules applied to this class, also apply to its children.

It is possible to set a granular disclosure policy, since all of the high-level classes (i.e., children of the Incident class) have a restriction attribute. Therefore, a child can override the guidelines of a parent class, be it to restrict or relax the disclosure rules (e.g., a child has a weaker policy than an ancestor; or an ancestor has a weak policy, and the children selectively apply more rigid controls). The implicit value of the restriction attribute for a class that did not specify one can be found in the closest ancestor that did specify a value.

This attribute is defined as an enumerated value with a default value of "private". Note that the default value of the restriction attribute is only defined in the context of the Incident class. In other classes where this attribute is used, no default is specified.

1. public. There are no restrictions placed in the information.
2. need-to-know. The information may be shared with other parties that are involved in the incident as determined by the recipient of this document (e.g., multiple victim sites can be informed of each other).
3. private. The information may not be shared.
4. default. The information can be shared according to an information disclosure policy pre-arranged by the communicating parties.
5. Optional. STRING. The indicator set ID is used to group related indicators.

### 3.3. IncidentID Class

The IncidentID class represents an incident tracking number that is unique in the context of the CSIRT and identifies the activity characterized in an IODEF Document. This identifier would serve as an index into the CSIRT incident handling system. The combination of the name attribute and the string in the element content MUST be a globally unique identifier describing the activity. Documents generated by a given CSIRT MUST NOT reuse the same value unless they are referencing the same incident.

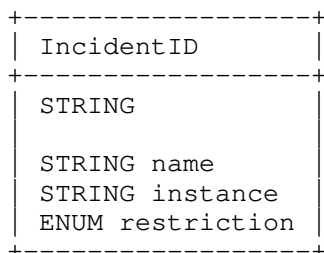


Figure 3: The IncidentID Class

The IncidentID class has three attributes:

name

Required. STRING. An identifier describing the CSIRT that created the document. In order to have a globally unique CSIRT name, the fully qualified domain name associated with the CSIRT MUST be used.

instance

Optional. STRING. An identifier referencing a subset of the named incident.

restriction

Optional. ENUM. This attribute has been defined in Section 3.2. The default value is "public".

### 3.4. AlternativeID Class

The AlternativeID class lists the incident tracking numbers used by CSIRTs, other than the one generating the document, to refer to the identical activity described the IODEF document. A tracking number listed as an AlternativeID references the same incident detected by another CSIRT. The incident tracking numbers of the CSIRT that generated the IODEF document should never be considered an AlternativeID.

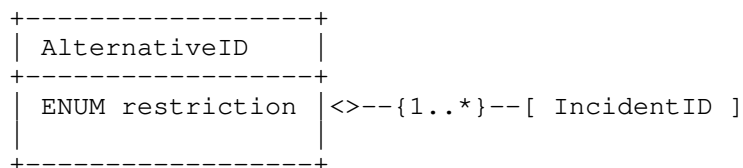


Figure 4: The AlternativeID Class



The aggregate class that constitutes AlternativeID is:

IncidentID

One or more. The incident tracking number of another CSIRT.

The AlternativeID class has one attribute:

restriction

Optional. ENUM. This attribute has been defined in Section 3.2.

### 3.5. RelatedActivity Class

The RelatedActivity class lists either incident tracking numbers of incidents or URLs (not both) that refer to activity related to the one described in the IODEF document. These references may be to local incident tracking numbers or to those of other CSIRTs.

The specifics of how a CSIRT comes to believe that two incidents are related are considered out of scope.

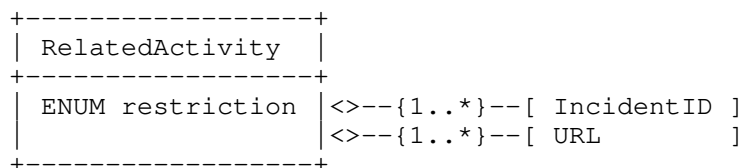


Figure 5: RelatedActivity Class

The aggregate classes that constitutes RelatedActivity are:

IncidentID

One or more. The incident tracking number of a related incident.

URL

One or more. URL. A URL to activity related to this incident.

The RelatedActivity class has one attribute:

restriction

Optional. ENUM. This attribute has been defined in Section 3.2.

### 3.6. AdditionalData Class

The AdditionalData class serves as an extension mechanism for information not otherwise represented in the data model. For relatively simple information, atomic data types (e.g., integers,

strings) are provided with a mechanism to annotate their meaning. The class can also be used to extend the data model (and the associated Schema) to support proprietary extensions by encapsulating entire XML documents conforming to another Schema (e.g., IDMEF). A detailed discussion for extending the data model and the schema can be found in Section 5.

Unlike XML, which is self-describing, atomic data must be documented to convey its meaning. This information is described in the 'meaning' attribute. Since these description are outside the scope of the specification, some additional coordination may be required to ensure that a recipient of a document using the AdditionalData classes can make sense of the custom extensions.

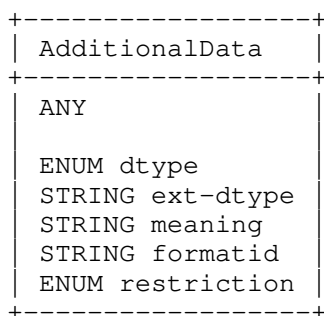


Figure 6: The AdditionalData Class

The AdditionalData class has five attributes:

#### dtype

Required. ENUM. The data type of the element content. The permitted values for this attribute are shown below. The default value is "string".

1. boolean. The element content is of type BOOLEAN.
2. byte. The element content is of type BYTE.
3. character. The element content is of type CHARACTER.
4. date-time. The element content is of type DATETIME.
5. integer. The element content is of type INTEGER.
6. portlist. The element content is of type PORTLIST.

7. real. The element content is of type REAL.
8. string. The element content is of type STRING.
9. file. The element content is a base64 encoded binary file encoded as a BYTE[] type.
10. frame. The element content is a layer-2 frame encoded as a HEXBIN type.
11. packet. The element content is a layer-3 packet encoded as a HEXBIN type.
12. ipv4-packet. The element content is an IPv4 packet encoded as a HEXBIN type.
13. ipv6-packet. The element content is an IPv6 packet encoded as a HEXBIN type.
14. path. The element content is a file-system path encoded as a STRING type.
15. url. The element content is of type URL.
16. csv. The element content is a common separated value (CSV) list per Section 2 of [20] encoded as a STRING type.
17. winreg. The element content is a Windows registry key encoded as a STRING type.
18. xml. The element content is XML (see Section 5).
19. ext-value. An escape value used to extend this attribute. See Section 5.1.

ext-dtype

Optional. STRING. A means by which to extend the dtype attribute. See Section 5.1.

meaning

Optional. STRING. A free-form description of the element content.

formatid

Optional. STRING. An identifier referencing the format and semantics of the element content.

restriction

Optional. ENUM. This attribute has been defined in Section 3.2.

### 3.7. Contact Class

The Contact class describes contact information for organizations and personnel involved in the incident. This class allows for the naming of the involved party, specifying contact information for them, and identifying their role in the incident.

People and organizations are treated interchangeably as contacts; one can be associated with the other using the recursive definition of the class (the Contact class is aggregated into the Contact class). The 'type' attribute disambiguates the type of contact information being provided.

The inheriting definition of Contact provides a way to relate information without requiring the explicit use of identifiers in the classes or duplication of data. A complete point of contact is derived by a particular traversal from the root Contact class to the leaf Contact class. As such, multiple points of contact might be specified in a single instance of a Contact class. Each child Contact class logically inherits contact information from its ancestors.

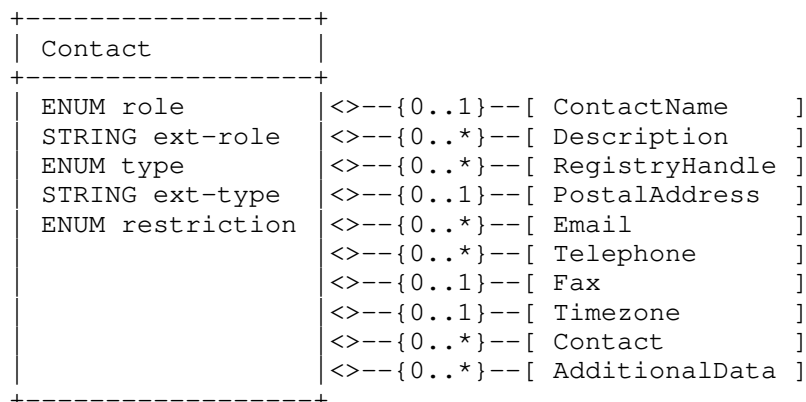


Figure 7: The Contact Class

The aggregate classes that constitute the Contact class are:

**ContactName**

Zero or one. ML\_STRING. The name of the contact. The contact may either be an organization or a person. The type attribute disambiguates the semantics.

**Description**

Zero or many. ML\_STRING. A free-form description of this contact. In the case of a person, this is often the organizational title of the individual.

**RegistryHandle**

Zero or many. A handle name into the registry of the contact.

**PostalAddress**

Zero or one. The postal address of the contact.

**Email**

Zero or many. The email address of the contact.

**Telephone**

Zero or many. The telephone number of the contact.

**Fax**

Zero or one. The facsimile telephone number of the contact.

**Timezone**

Zero or one. TIMEZONE. The timezone in which the contact resides formatted according to Section 2.9.

**Contact**

Zero or many. A Contact instance contained within another Contact instance inherits the values of the parent(s). This recursive definition can be used to group common data pertaining to multiple points of contact and is especially useful when listing multiple contacts at the same organization.

**AdditionalData**

Zero or many. A mechanism by which to extend the data model.

At least one of the aggregate classes MUST be present in an instance of the Contact class. This is not enforced in the IODEF schema as there is no simple way to accomplish it.

The Contact class has five attributes:

**role**

Required. ENUM. Indicates the role the contact fulfills. This attribute is defined as an enumerated list:

1. creator. The entity that generate the document.
2. admin. An administrative contact for a host or network.
3. tech. A technical contact for a host or network.
4. irt. The CSIRT involved in handling the incident.
5. cc. An entity that is to be kept informed about the handling of the incident.
6. ext-value. An escape value used to extend this attribute. See Section 5.1.

**ext-role**

Optional. STRING. A means by which to extend the role attribute. See Section 5.1.

**type**

Required. ENUM. Indicates the type of contact being described. This attribute is defined as an enumerated list:

1. person. The information for this contact references an individual.
2. organization. The information for this contact references an organization.
3. ext-value. An escape value used to extend this attribute. See Section 5.1.

**ext-type**

Optional. STRING. A means by which to extend the type attribute. See Section 5.1.

**restriction**

Optional. ENUM. This attribute is defined in Section 3.2.

### 3.7.1. RegistryHandle Class

The RegistryHandle class represents a handle into an Internet registry or community-specific database. The handle is specified in the element content and the type attribute specifies the database.

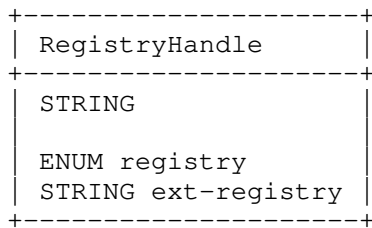


Figure 8: The RegistryHandle Class

The RegistryHandle class has two attributes:

`registry`

Required. ENUM. The database to which the handle belongs. The possible values are:

1. `internic`. Internet Network Information Center
2. `apnic`. Asia Pacific Network Information Center
3. `arin`. American Registry for Internet Numbers
4. `lacnic`. Latin-American and Caribbean IP Address Registry
5. `ripe`. Reseaux IP Europeens
6. `afrinic`. African Internet Numbers Registry
7. `local`. A database local to the CSIRT
8. `ext-value`. An escape value used to extend this attribute. See Section 5.1.

`ext-registry`

Optional. STRING. A means by which to extend the registry attribute. See Section 5.1.

### 3.7.2. PostalAddress Class

The PostalAddress class specifies a postal address formatted according to the POSTAL data type (Section 2.11).

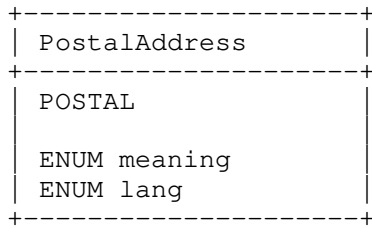


Figure 9: The PostalAddress Class

The `PostalAddress` class has two attributes:

`meaning`

Optional. ENUM. A free-form description of the element content.

`lang`

Optional. ENUM. A valid language code per RFC 4646 [7] constrained by the definition of `"xs:language"`. The interpretation of this code is described in Section 6.

### 3.7.3. Email Class

The `Email` class specifies an email address formatted according to `EMAIL` data type (Section 2.14).

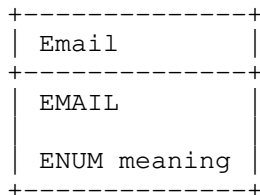


Figure 10: The Email Class

The `Email` class has one attribute:

`meaning`

Optional. ENUM. A free-form description of the element content.

### 3.7.4. Telephone and Fax Classes

The `Telephone` and `Fax` classes specify a voice or fax telephone number respectively, and are formatted according to `PHONE` data type (Section 2.13).



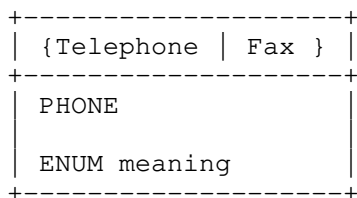


Figure 11: The Telephone and Fax Classes

The Telephone class has one attribute:

meaning

Optional. ENUM. A free-form description of the element content (e.g., hours of coverage for a given number).

### 3.8. Time Classes

The data model uses five different classes to represent a timestamp. Their definition is identical, but each has a distinct name to convey a difference in semantics.

The element content of each class is a timestamp formatted according to the DATETIME data type (see Section 2.8).



Figure 12: The Time Classes

#### 3.8.1. StartTime

The StartTime class represents the time the incident began.

#### 3.8.2. EndTime

The EndTime class represents the time the incident ended.

#### 3.8.3. DetectTime

The DetectTime class represents the time the first activity of the incident was detected.

#### 3.8.4. ReportTime

The ReportTime class represents the time the incident was reported. This timestamp SHOULD coincide to the time at which the IODEF document is generated.

#### 3.8.5. DateTime

The DateTime class is a generic representation of a timestamp. Its semantics should be inferred from the parent class in which it is aggregated.

#### 3.9. Method Class

The Method class describes the methodology used by the intruder to perpetrate the events of the incident. This class consists of a list of references describing the attack method and a free form description of the technique.

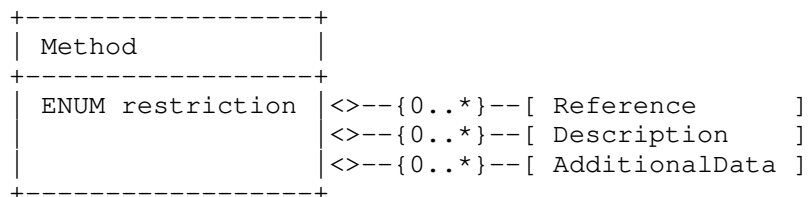


Figure 13: The Method Class

The Method class is composed of three aggregate classes.

##### Reference

Zero or many. A reference to a vulnerability, malware sample, advisory, or analysis of an attack technique.

##### Description

Zero or many. ML\_STRING. A free-form text description of the methodology used by the intruder.

##### AdditionalData

Zero or many. A mechanism by which to extend the data model.

Either an instance of the Reference or Description class MUST be present.

The Method class has one attribute:

restriction

Optional. ENUM. This attribute is defined in Section 3.2.

### 3.9.1. Reference Class

The Reference class is a reference to a vulnerability, IDS alert, malware sample, advisory, or attack technique. A reference consists of a name, a URL to this reference, and an optional description.

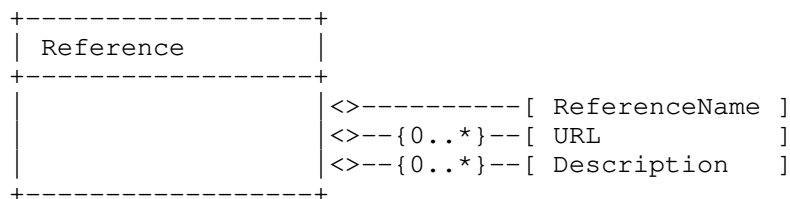


Figure 14: The Reference Class

The aggregate classes that constitute Reference:

ReferenceName

One. ML\_STRING. Name of the reference.

URL

Zero or many. URL. A URL associated with the reference.

Description

Zero or many. ML\_STRING. A free-form text description of this reference.

The Reference class has 4 attributes.

indicator-uid

Optional. STRING. A unique identifier for an Indicator.

indicator-set-id

Optional. STRING. The indicator set ID is used to group related indicators.

attacktype

Optional. ENUM. A unique identifier for an Indicator.

ext-attacktype

Optional. STRING. A mechanism by which to extend the Attack Type.

### 3.10. Assessment Class

The Assessment class describes the technical and non-technical repercussions of the incident on the CSIRT's constituency.

This class was derived from the IDMEF[17].

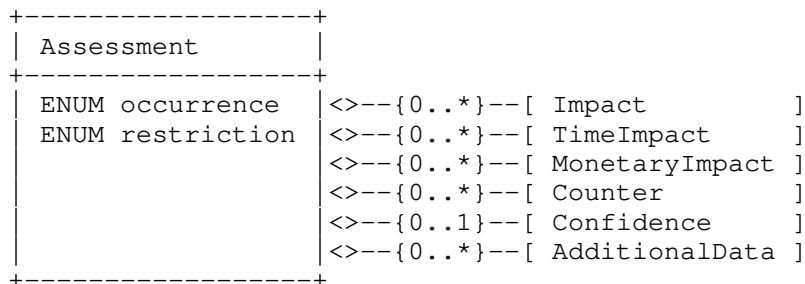


Figure 15: Assessment Class

The aggregate classes that constitute Assessment are:

#### Impact

Zero or many. Technical impact of the incident on a network.

#### TimeImpact

Zero or many. Impact of the activity measured with respect to time.

#### MonetaryImpact

Zero or many. Impact of the activity measured with respect to financial loss.

#### Counter

Zero or more. A counter with which to summarize the magnitude of the activity.

#### Confidence

Zero or one. An estimate of confidence in the assessment.

#### AdditionalData

Zero or many. A mechanism by which to extend the data model.

A least one instance of the possible three impact classes (i.e., Impact, TimeImpact, or MonetaryImpact) MUST be present.

The Assessment class has four attributes:

**occurrence**

Optional. ENUM. Specifies whether the assessment is describing actual or potential outcomes.

1. actual. This assessment describes activity that has occurred.
2. potential. This assessment describes potential activity that might occur.

**restriction**

Optional. ENUM. This attribute is defined in Section 3.2.

**indicator-uid**

Optional. STRING. A unique identifier for an Indicator.

**indicator-set-id**

Optional. STRING. The indicator set ID is used to group related indicators.

**3.10.1. Impact Class**

The Impact class allows for categorizing and describing the technical impact of the incident on the network of an organization.

This class is based on the IDMEF [17].

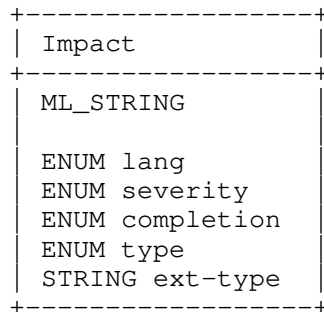


Figure 16: Impact Class

The element content will be a free-form textual description of the impact.

The Impact class has five attributes:

**lang**

Optional. ENUM. A valid language code per RFC 4646 [7] constrained by the definition of "xs:language". The interpretation of this code is described in Section 6.

**severity**

Optional. ENUM. An estimate of the relative severity of the activity. The permitted values are shown below. There is no default value.

1. low. Low severity
2. medium. Medium severity
3. high. High severity

**completion**

Optional. ENUM. An indication whether the described activity was successful. The permitted values are shown below. There is no default value.

1. failed. The attempted activity was not successful.
2. succeeded. The attempted activity succeeded.

**type**

Required. ENUM. Classifies the malicious activity into incident categories. The permitted values are shown below. The default value is "other".

1. admin. Administrative privileges were attempted.
2. dos. A denial of service was attempted.
3. file. An action that impacts the integrity of a file or database was attempted.
4. info-leak. An attempt was made to exfiltrate information.
5. misconfiguration. An attempt was made to exploit a mis-configuration in a system.
6. policy. Activity violating site's policy was attempted.
7. recon. Reconnaissance activity was attempted.
8. social-engineering. A social engineering attack was attempted.

9. user. User privileges were attempted.
10. unknown. The classification of this activity is unknown.
11. ext-value. An escape value used to extend this attribute.  
See Section 5.1.

#### ext-type

Optional. STRING. A means by which to extend the type attribute.  
See Section 5.1.

### 3.10.2. TimeImpact Class

The TimeImpact class describes the impact of the incident on an organization as a function of time. It provides a way to convey down time and recovery time.

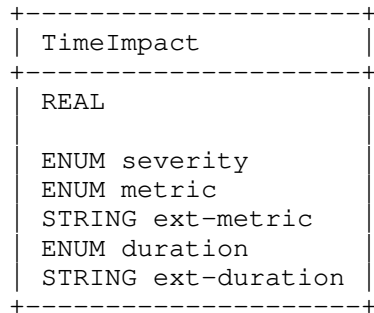


Figure 17: TimeImpact Class

The element content is a positive, floating point (REAL) number specifying a unit of time. The duration and metric attributes will imply the semantics of the element content.

The TimeImpact class has five attributes:

#### severity

Optional. ENUM. An estimate of the relative severity of the activity. The permitted values are shown below. There is no default value.

1. low. Low severity
2. medium. Medium severity

3. high. High severity

#### metric

Required. ENUM. Defines the metric in which the time is expressed. The permitted values are shown below. There is no default value.

1. labor. Total staff-time to recovery from the activity (e.g., 2 employees working 4 hours each would be 8 hours).
2. elapsed. Elapsed time from the beginning of the recovery to its completion (i.e., wall-clock time).
3. downtime. Duration of time for which some provided service(s) was not available.
4. ext-value. An escape value used to extend this attribute. See Section 5.1.

#### ext-metric

Optional. STRING. A means by which to extend the metric attribute. See Section 5.1.

#### duration

Optional. ENUM. Defines a unit of time, that when combined with the metric attribute, fully describes a metric of impact that will be conveyed in the element content. The permitted values are shown below. The default value is "hour".

1. second. The unit of the element content is seconds.
2. minute. The unit of the element content is minutes.
3. hour. The unit of the element content is hours.
4. day. The unit of the element content is days.
5. month. The unit of the element content is months.
6. quarter. The unit of the element content is quarters.
7. year. The unit of the element content is years.
8. ext-value. An escape value used to extend this attribute. See Section 5.1.



**ext-duration**

Optional. STRING. A means by which to extend the duration attribute. See Section 5.1.

**3.10.3. MonetaryImpact Class**

The MonetaryImpact class describes the financial impact of the activity on an organization. For example, this impact may consider losses due to the cost of the investigation or recovery, diminished productivity of the staff, or a tarnished reputation that will affect future opportunities.

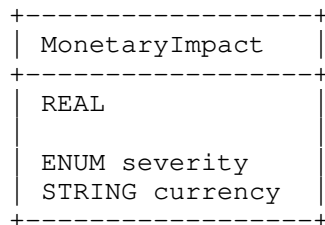


Figure 18: MonetaryImpact Class

The element content is a positive, floating point number (REAL) specifying a unit of currency described in the currency attribute.

The MonetaryImpact class has two attributes:

**severity**

Optional. ENUM. An estimate of the relative severity of the activity. The permitted values are shown below. There is no default value.

1. low. Low severity
2. medium. Medium severity
3. high. High severity

**currency**

Optional. STRING. Defines the currency in which the monetary impact is expressed. The permitted values are defined in ISO 4217:2001, Codes for the representation of currencies and funds [14]. There is no default value.

#### 3.10.4. Confidence Class

The Confidence class represents a best estimate of the validity and accuracy of the described impact (see Section 3.10) of the incident activity. This estimate can be expressed as a category or a numeric calculation.

This class is based upon the IDMEF [17]).

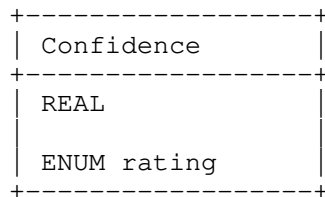


Figure 19: Confidence Class

The element content expresses a numerical assessment in the confidence of the data when the value of the rating attribute is "numeric". Otherwise, this element should be empty.

The Confidence class has one attribute.

##### rating

Required. ENUM. A rating of the analytical validity of the specified Assessment. The permitted values are shown below. There is no default value.

1. low. Low confidence in the validity.
2. medium. Medium confidence in the validity.
3. high. High confidence in the validity.
4. numeric. The element content contains a number that conveys the confidence of the data. The semantics of this number outside the scope of this specification.
5. unknown. The confidence rating value is not known.

### 3.11. History Class

The History class is a log of the significant events or actions performed by the involved parties during the course of handling the incident.

The level of detail maintained in this log is left up to the discretion of those handling the incident.

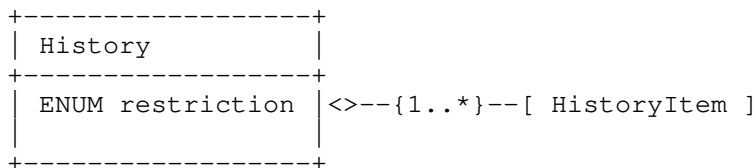


Figure 20: The History Class

The class that constitutes History is:

#### HistoryItem

One or many. Entry in the history log of significant events or actions performed by the involved parties.

The History class has one attribute:

#### restriction

Optional. ENUM. This attribute is defined in Section 3.2. The default value is "default".

#### 3.11.1. HistoryItem Class

The HistoryItem class is an entry in the History (Section 3.11) log that documents a particular action or event that occurred in the course of handling the incident. The details of the entry are a free-form description, but each can be categorized with the type attribute.

```

+-----+
| HistoryItem |
+-----+
| ENUM restriction | <>-----[ DateTime      ]
| ENUM action      | <>--{0..1}--[ IncidentId    ]
| STRING ext-action | <>--{0..1}--[ Contact        ]
|                  | <>--{0..*}--[ Description    ]
|                  | <>--{0..*}--[ AdditionalData ]
+-----+

```

Figure 21: HistoryItem Class

The aggregate classes that constitute HistoryItem are:

#### DateTime

One. Timestamp of this entry in the history log (e.g., when the action described in the Description was taken).

#### IncidentID

Zero or One. In a history log created by multiple parties, the IncidentID provides a mechanism to specify which CSIRT created a particular entry and references this organization's incident tracking number. When a single organization is maintaining the log, this class can be ignored.

#### Contact

Zero or One. Provides contact information for the person that performed the action documented in this class.

#### Description

Zero or many. ML\_STRING. A free-form textual description of the action or event.

#### AdditionalData

Zero or many. A mechanism by which to extend the data model.

The HistoryItem class has five attributes:

#### restriction

Optional. ENUM. This attribute has been defined in Section 3.2.

#### action

Required. ENUM. Classifies a performed action or occurrence documented in this history log entry. As activity will likely have been instigated either through a previously conveyed expectation or internal investigation, this attribute is identical to the category attribute of the Expectation class. The difference is only one of tense. When an action is in this class,

it has been completed. See Section 3.13.

#### ext-action

Optional. STRING. A means by which to extend the action attribute. See Section 5.1.

#### indicator-uid

Optional. STRING. A unique identifier for an Indicator.

#### indicator-set-id

Optional. STRING. The indicator set ID is used to group related indicators.

### 3.12. EventData Class

The EventData class describes a particular event of the incident for a given set of hosts or networks. This description includes the systems from which the activity originated and those targeted, an assessment of the techniques used by the intruder, the impact of the activity on the organization, and any forensic evidence discovered.

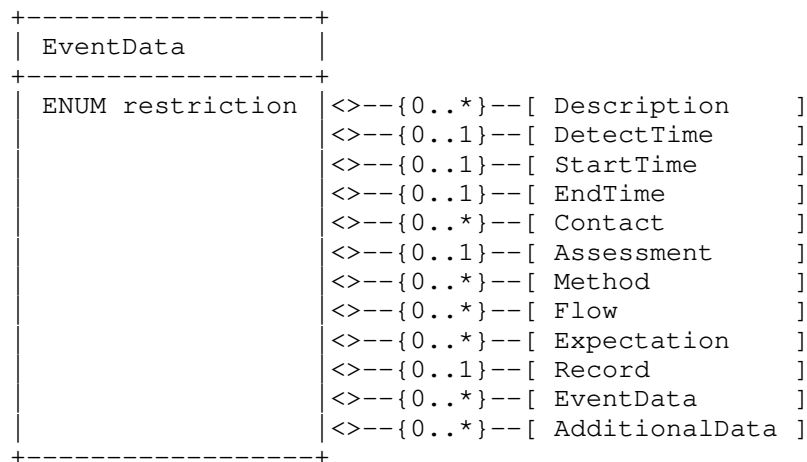


Figure 22: The EventData Class

The aggregate classes that constitute EventData are:

#### Description

Zero or more. ML\_STRING. A free-form textual description of the event.

**DetectTime**

Zero or one. The time the event was detected.

**StartTime**

Zero or one. The time the event started.

**EndTime**

Zero or one. The time the event ended.

**Contact**

Zero or more. Contact information for the parties involved in the event.

**Assessment**

Zero or one. The impact of the event on the target and the actions taken.

**Method**

Zero or more. The technique used by the intruder in the event.

**Flow**

Zero or more. A description of the systems or networks involved.

**Expectation**

Zero or more. The expected action to be performed by the recipient for the described event.

**Record**

Zero or one. Supportive data (e.g., log files) that provides additional information about the event.

**EventData**

Zero or more. EventData instances contained within another EventData instance inherit the values of the parent(s); this recursive definition can be used to group common data pertaining to multiple events. When EventData elements are defined recursively, only the leaf instances (those EventData instances not containing other EventData instances) represent actual events.

**AdditionalData**

Zero or more. An extension mechanism for data not explicitly represented in the data model.

At least one of the aggregate classes **MUST** be present in an instance of the EventData class. This is not enforced in the IODEF schema as there is no simple way to accomplish it.

The EventData class has two attributes:

**restriction**

Optional. ENUM. This attribute is defined in Section 3.2. The default value is "default".

**indicator-set-id**

Optional. STRING. The indicator set ID is used to group related indicators.

**3.12.1. Relating the Incident and EventData Classes**

There is substantial overlap in the Incident and EventData classes. Nevertheless, the semantics of these classes are quite different. The Incident class provides summary information about the entire incident, while the EventData class provides information about the individual events comprising the incident. In the most common case, the EventData class will provide more specific information for the general description provided in the Incident class. However, it may also be possible that the overall summarized information about the incident conflicts with some individual information in an EventData class when there is a substantial composition of various events in the incident. In such a case, the interpretation of the more specific EventData MUST supersede the more generic information provided in IncidentData.

**3.12.2. Cardinality of EventData**

The EventData class can be thought of as a container for the properties of an event in an incident. These properties include: the hosts involved, impact of the incident activity on the hosts, forensic logs, etc. With an instance of the EventData class, hosts (i.e., System class) are grouped around these common properties.

The recursive definition (or instance property inheritance) of the EventData class (the EventData class is aggregated into the EventData class) provides a way to related information without requiring the explicit use of unique attribute identifiers in the classes or duplicating information. Instead, the relative depth (nesting) of a class is used to group (relate) information.

For example, an EventData class might be used to describe two machines involved in an incident. This description can be achieved using multiple instances of the Flow class. It happens that there is a common technical contact (i.e., Contact class) for these two machines, but the impact (i.e., Assessment class) on them is different. A depiction of the representation for this situation can be found in Figure 23.

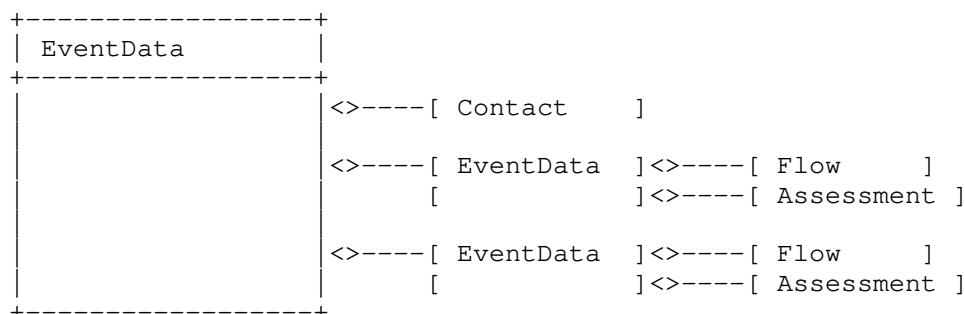


Figure 23: Recursion in the EventData Class

### 3.13. Expectation Class

The Expectation class conveys to the recipient of the IODEF document the actions the sender is requesting. The scope of the requested action is limited to purview of the EventData class in which this class is aggregated.

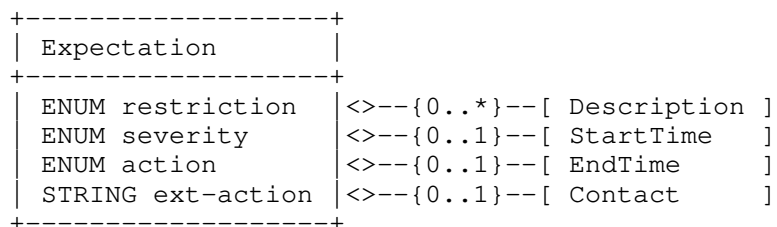


Figure 24: The Expectation Class

The aggregate classes that constitute Expectation are:

#### Description

Zero or many. ML\_STRING. A free-form description of the desired action(s).

#### StartTime

Zero or one. The time at which the action should be performed. A timestamp that is earlier than the ReportTime specified in the Incident class denotes that the expectation should be fulfilled as soon as possible. The absence of this element leaves the execution of the expectation to the discretion of the recipient.



**EndTime**

Zero or one. The time by which the action should be completed. If the action is not carried out by this time, it should no longer be performed.

**Contact**

Zero or one. The expected actor for the action.

The Expectations class has six attributes:

**restriction**

Optional. ENUM. This attribute is defined in Section 3.2. The default value is "default".

**severity**

Optional. ENUM. Indicates the desired priority of the action. This attribute is an enumerated list with no default value, and the semantics of these relative measures are context dependant.

1. low. Low priority
2. medium. Medium priority
3. high. High priority

**action**

Optional. ENUM. Classifies the type of action requested. This attribute is an enumerated list with a default value of "other".

1. nothing. No action is requested. Do nothing with the information.
2. contact-source-site. Contact the site(s) identified as the source of the activity.
3. contact-target-site. Contact the site(s) identified as the target of the activity.
4. contact-sender. Contact the originator of the document.
5. investigate. Investigate the systems(s) listed in the event.
6. block-host. Block traffic from the machine(s) listed as sources the event.
7. block-network. Block traffic from the network(s) lists as sources in the event.

8. `block-port`. Block the port listed as sources in the event.
9. `rate-limit-host`. Rate-limit the traffic from the machine(s) listed as sources in the event.
10. `rate-limit-network`. Rate-limit the traffic from the network(s) lists as sources in the event.
11. `rate-limit-port`. Rate-limit the port(s) listed as sources in the event.
12. `remediate-other`. Remediate the activity in a way other than by rate limiting or blocking.
13. `status-triage`. Conveys receipts and the triaging of an incident.
14. `status-new-info`. Conveys that new information was received for this incident.
15. `other`. Perform some custom action described in the Description class.
16. `ext-value`. An escape value used to extend this attribute. See Section 5.1.

`ext-action`

Optional. `STRING`. A means by which to extend the action attribute. See Section 5.1.

`indicator-uid`

Optional. `STRING`. A unique identifier for an Indicator.

`indicator-set-id`

Optional. `STRING`. The indicator set ID is used to group related indicators.

## 3.14. Flow Class

The Flow class groups related the source and target hosts.

```
+-----+
| Flow   |
+-----+
|        | <--{1..*}--[ System  ]
+-----+
```

Figure 25: The Flow Class

The aggregate class that constitutes Flow is:

#### System

One or More. A host or network involved in an event.

The Flow System class has no attributes.

### 3.15. System Class

The System class describes a system or network involved in an event. The systems or networks represented by this class are categorized according to the role they played in the incident through the category attribute. The value of this category attribute dictates the semantics of the aggregated classes in the System class. If the category attribute has a value of "source", then the aggregated classes denote the machine and service from which the activity is originating. With a category attribute value of "target" or "intermediary", then the machine or service is the one targeted in the activity. A value of "sensor" dictates that this System was part of an instrumentation to monitor the network.

+-----+	
System	
+-----+	
ENUM restriction	<>-----[ Node ]
ENUM category	<>--{0..*}--[ Service ]
STRING ext-category	<>--{0..*}--[ OperatingSystem ]
STRING interface	<>--{0..*}--[ Counter ]
ENUM spoofed	<>--{0..*}--[ Description ]
	<>--{0..*}--[ AdditionalData ]
+-----+	

Figure 26: The System Class

The aggregate classes that constitute System are:

#### Node

One. A host or network involved in the incident.

#### Service

Zero or more. A network service running on the system.

#### OperatingSystem

Zero or more. The operating system running on the system.

**Counter**

Zero or more. A counter with which to summarize properties of this host or network.

**Description**

Zero or more. ML\_STRING. A free-form text description of the System.

**AdditionalData**

Zero or many. A mechanism by which to extend the data model.

The System class has six attributes:

**restriction**

Optional. ENUM. This attribute is defined in Section 3.2.

**category**

Optional. ENUM. Classifies the role the host or network played in the incident. The possible values are:

1. source. The System was the source of the event.
2. target. The System was the target of the event.
3. watchlist-source. The source of the event was on a watchlist.
4. watchlist-target. The target of the event was on a watchlist.
5. intermediate. The System was an intermediary in the event.
6. sensor. The System was a sensor monitoring the event.
7. infrastructure. The System was an infrastructure node of IODEF document exchange.
8. ext-value. An escape value used to extend this attribute. See Section 5.1.

**ext-category**

Optional. STRING. A means by which to extend the category attribute. See Section 5.1.

**indicator-set-id**

Optional. STRING. The indicator set ID is used to group related indicators.

**interface**

Optional. STRING. Specifies the interface on which the event(s) on this System originated. If the Node class specifies a network rather than a host, this attribute has no meaning.

**spoofed**

Optional. ENUM. An indication of confidence in whether this System was the true target or attacking host. The permitted values for this attribute are shown below. The default value is "unknown".

1. unknown. The accuracy of the category attribute value is unknown.
2. yes. The category attribute value is probably incorrect. In the case of a source, the System is likely a decoy; with a target, the System was likely not the intended victim.
3. no. The category attribute value is believed to be correct.

**3.16. Node Class**

The Node class names a system (e.g., PC, router) or network.

This class was derived from the IDMEF [17].

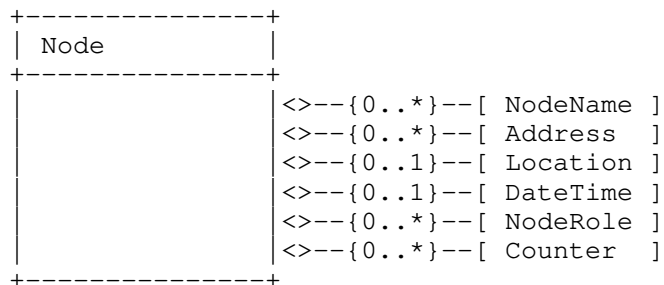


Figure 27: The Node Class

The aggregate classes that constitute Node are:

**NodeName**

Zero or more. ML\_STRING. The name of the Node (e.g., fully qualified domain name). This information MUST be provided if no Address information is given.

**Address**

Zero or more. The hardware, network, or application address of the Node. If a NodeName is not provided, at least one Address MUST be specified.

**Location**

Zero or one. ML\_STRING. A free-form description of the physical location of the equipment.

**DateTime**

Zero or one. A timestamp of when the resolution between the name and address was performed. This information SHOULD be provided if both an Address and NodeName are specified.

**NodeRole**

Zero or more. The intended purpose of the Node.

**Counter**

Zero or more. A counter with which to summarize properties of this host or network.

**3.16.1. Counter Class**

The Counter class summarize multiple occurrences of some event, or conveys counts or rates on various features (e.g., packets, sessions, events).

The value of the counter is the element content with its units represented in the type attribute. A rate for a given feature can be expressed by setting the duration attribute. The complete semantics are entirely context dependant based on the class in which the Counter is aggregated.

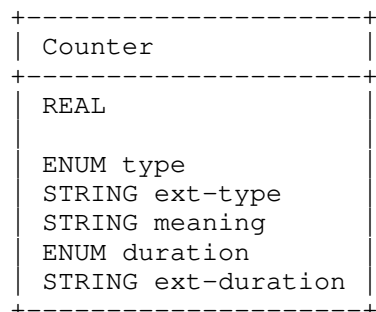


Figure 28: The Counter Class

The Counter class has three attribute:

type

Required. ENUM. Specifies the units of the element content.

1. byte. Count of bytes.
2. packet. Count of packets.
3. flow. Count of flow (e.g., NetFlow records).
4. session. Count of sessions.
5. alert. Count of notifications generated by another system (e.g., IDS or SIM).
6. message. Count of messages (e.g., mail messages).
7. event. Count of events.
8. host. Count of hosts.
9. site. Count of site.
10. organization. Count of organizations.
11. ext-value. An escape value used to extend this attribute. See Section 5.1.

ext-type

Optional. STRING. A means by which to extend the type attribute. See Section 5.1.

duration

Optional. ENUM. If present, the Counter class represents a rate rather than a count over the entire event. In that case, this attribute specifies the denominator of the rate (where the type attribute specified the nominator). The possible values of this attribute are defined in Section 3.10.2

ext-duration

Optional. STRING. A means by which to extend the duration attribute. See Section 5.1.

### 3.16.2. Address Class

The Address class represents a hardware (layer-2), network (layer-3), or application (layer-7) address.

This class was derived from the IDMEF [17].

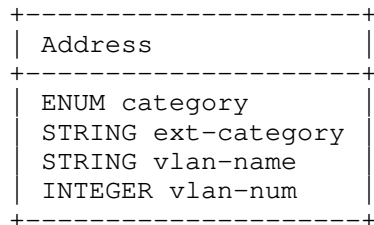


Figure 29: The Address Class

The Address class has five attributes:

#### category

Optional. ENUM. The type of address represented. The permitted values for this attribute are shown below. The default value is "ipv4-addr".

1. asn. Autonomous System Number
2. atm. Asynchronous Transfer Mode (ATM) address
3. e-mail. Electronic mail address (RFC 822)
4. ipv4-addr. IPv4 host address in dotted-decimal notation (a.b.c.d)
5. ipv4-net. IPv4 network address in dotted-decimal notation, slash, significant bits (a.b.c.d/nn)
6. ipv4-net-mask. IPv4 network address in dotted-decimal notation, slash, network mask in dotted-decimal notation (a.b.c.d/w.x.y.z)
7. ipv6-addr. IPv6 host address
8. ipv6-net. IPv6 network address, slash, significant bits
9. ipv6-net-mask. IPv6 network address, slash, network mask



- 10. `mac`. Media Access Control (MAC) address
- 11. `site-uri`. A URL or URI for a site.
- 12. `ext-value`. An escape value used to extend this attribute.  
See Section 5.1.

`ext-category`

Optional. `STRING`. A means by which to extend the category attribute. See Section 5.1.

`vlan-name`

Optional. `STRING`. The name of the Virtual LAN to which the address belongs.

`vlan-num`

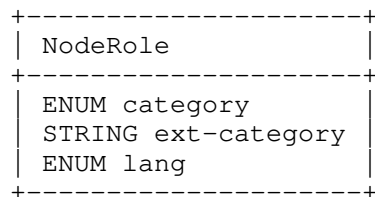
Optional. `STRING`. The number of the Virtual LAN to which the address belongs.

`indicator-uid`

Optional. `STRING`. A unique identifier for an Indicator.

3.16.3. `NodeRole` Class

The `NodeRole` class describes the intended function performed by a particular host.

Figure 30: The `NodeRole` Class

The `NodeRole` class has three attributes:

`category`

Required. `ENUM`. Functionality provided by a node.

- 1. `client`. Client computer
- 2. `server-internal`. Server with internal services

3. server-public. Server with public services
4. www. WWW server
5. mail. Mail server
6. messaging. Messaging server (e.g., NNTP, IRC, IM)
7. streaming. Streaming-media server
8. voice. Voice server (e.g., SIP, H.323)
9. file. File server (e.g., SMB, CVS, AFS)
10. ftp. FTP server
11. p2p. Peer-to-peer node
12. name. Name server (e.g., DNS, WINS)
13. directory. Directory server (e.g., LDAP, finger, whois)
14. credential. Credential server (e.g., domain controller, Kerberos)
15. print. Print server
16. application. Application server
17. database. Database server
18. infra. Infrastructure server (e.g., router, firewall, DHCP)
19. log. Logserver (e.g., syslog)
20. ext-value. An escape value used to extend this attribute.  
See Section 5.1.

ext-category

Optional. STRING. A means by which to extend the category attribute. See Section 5.1.

lang

Optional. ENUM. A valid language code per RFC 4646 [7] constrained by the definition of "xs:language". The interpretation of this code is described in Section 6.

### 3.17. Service Class

The Service class describes a network service of a host or network. The service is identified by specific port or list of ports, along with the application listening on that port.

When Service occurs as an aggregate class of a System that is a source, then this service is the one from which activity of interest is originating. Conversely, when Service occurs as an aggregate class of a System that is a target, then that service is the one to which activity of interest is directed.

This class was derived from the IDMEF [17].

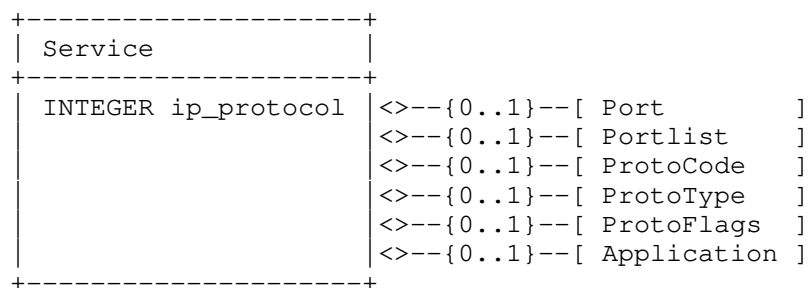


Figure 31: The Service Class

The aggregate classes that constitute Service are:

#### Port

Zero or one. INTEGER. A port number.

#### Portlist

Zero or one. PORTLIST. A list of port numbers formatted according to Section 2.10.

#### ProtoCode

Zero or one. INTEGER. A layer-4 protocol-specific code field (e.g., ICMP code field).

#### ProtoType

Zero or one. INTEGER. A layer-4 protocol specific type field (e.g., ICMP type field).

**ProtoFlags**

Zero or one. INTEGER. A layer-4 protocol specific flag field (e.g., TCP flag field).

**Application**

Zero or one. The application bound to the specified Port or Portlist.

Either a Port or Portlist class MUST be specified for a given instance of a Service class.

For a given source, System@type="source", a corresponding target, System@type="target", maybe defined, or vice versa. When a Portlist class is defined in the Service class of both the source and target in a given instance of the Flow class, there MUST be symmetry in the enumeration of the ports. Thus, if n-ports are listed for a source, n-ports should be listed for the target. Likewise, the ports should be listed in an identical sequence such that the n-th port in the source corresponds to the n-th port of the target. This symmetry in listing and sequencing of ports applies whether there are 1-to-1, 1-to-many, or many-to-many sources-to-targets. In the 1-to-many or many-to-many, the exact order in which the System classes are enumerated in the Flow class is significant.

The Service class has three attributes:

**ip\_protocol**

Required. INTEGER. The IANA protocol number.

**indicator-uid**

Optional. STRING. A unique identifier for an Indicator.

**indicator-set-id**

Optional. STRING. The indicator set ID is used to group related indicators.

### 3.17.1. Application Class

The Application class describes an application running on a System providing a Service.

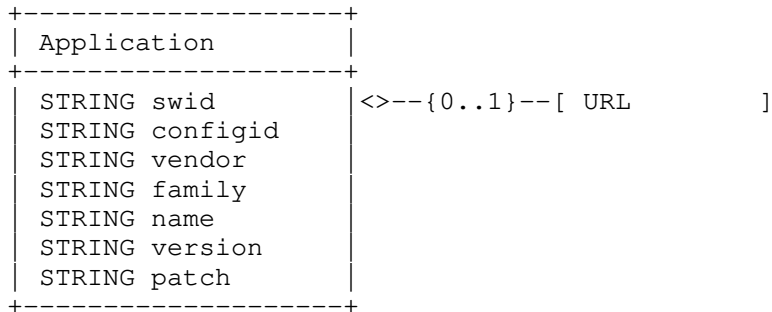


Figure 32: The Application Class

The aggregate class that constitute Application is:

URL

Zero or one. URL. A URL describing the application.

The Application class has seven attributes:

swid

Optional. STRING. An identifier that can be used to reference this software, where the default value is "0".

configid

Optional. STRING. An identifier that can be used to reference a particular configuration of this software, where the default value is "0".

vendor

Optional. STRING. Vendor name of the software.

family

Optional. STRING. Family of the software.

name

Optional. STRING. Name of the software.

version

Optional. STRING. Version of the software.

patch

Optional. STRING. Patch or service pack level of the software.

### 3.18. OperatingSystem Class

The OperatingSystem class describes the operating system running on a System. The definition is identical to the Application class (Section 3.17.1).

### 3.19. Record Class

The Record class is a container class for log and audit data that provides supportive information about the incident. The source of this data will often be the output of monitoring tools. These logs should substantiate the activity described in the document.

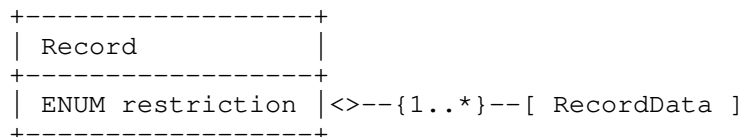


Figure 33: Record Class

The aggregate class that constitutes Record is:

#### RecordData

One or more. Log or audit data generated by a particular type of sensor. Separate instances of the RecordData class SHOULD be used for each sensor type.

The Record class has one attribute:

#### restriction

Optional. ENUM. This attribute has been defined in Section 3.2.

#### 3.19.1. RecordData Class

The RecordData class groups log or audit data from a given sensor (e.g., IDS, firewall log) and provides a way to annotate the output.

```

+-----+
| RecordData |
+-----+
| ENUM restriction | <>--{0..1}--[ DateTime ]
|                  | <>--{0..*}--[ Description ]
|                  | <>--{0..1}--[ Application ]
|                  | <>--{0..*}--[ RecordPattern ]
|                  | <>--{1..*}--[ RecordItem ]
|                  | <>--{0..1}--[ FileName ]
]
|                  | <>--{0..*}--[ WindowsRegistryKeysModified ]
]
|                  | <>--{0..*}--[ AdditionalData ]
+-----+

```

Figure 34: The RecordData Class

The aggregate classes that constitutes RecordData is:

#### DateTime

Zero or one. Timestamp of the RecordItem data.

#### Description

Zero or more. ML\_STRING. Free-form textual description of the provided RecordItem data. At minimum, this description should convey the significance of the provided RecordItem data.

#### Application

Zero or one. Information about the sensor used to generate the RecordItem data.

#### RecordPattern

Zero or more. A search string to precisely find the relevant data in a RecordItem.

#### RecordItem

One or more. Log, audit, or forensic data.

#### FileName

Zero or one. ML\_STRING. The file name and hash of a file indicator.

#### WindowsRegistryKeysModified

Zero or more. The registry keys that were modified that are indicator(s).

#### AdditionalData

Zero or more. An extension mechanism for data not explicitly represented in the data model.

The RecordData class has three attribute:

restriction

Optional. ENUM. This attribute has been defined in Section 3.2.

indicator-uid

Optional. STRING. A unique identifier for an Indicator.

indicator-set-id

Optional. STRING. The indicator set ID is used to group related indicators.

### 3.19.2. RecordPattern Class

The RecordPattern class describes where in the content of the RecordItem relevant information can be found. It provides a way to reference subsets of information, identified by a pattern, in a large log file, audit trail, or forensic data.

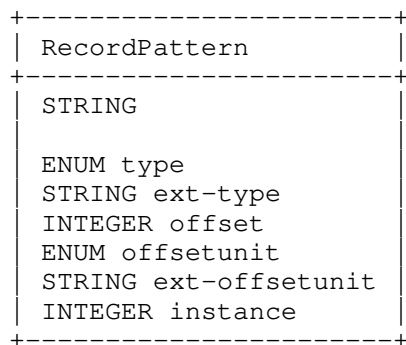


Figure 35: The RecordPattern Class

The specific pattern to search with in the RecordItem is defined in the body of the element. It is further annotated by four attributes:

type

Required. ENUM. Describes the type of pattern being specified in the element content. The default is "regex".

1. regex. regular expression, per Appendix F of [3].
2. binary. Binhex encoded binary pattern, per the HEXBIN data type.



3. `xpath`. XML Path (XPath) [5]

4. `ext-value`. An escape value used to extend this attribute.  
See Section 5.1.

`ext-type`

Optional. `STRING`. A means by which to extend the type attribute.  
See Section 5.1.

`offset`

Optional. `INTEGER`. Amount of units (determined by the `offsetunit` attribute) to seek into the `RecordItem` data before matching the pattern.

`offsetunit`

Optional. `ENUM`. Describes the units of the offset attribute.  
The default is "line".

1. `line`. Offset is a count of lines.

2. `byte`. Offset is a count of bytes.

3. `ext-value`. An escape value used to extend this attribute.  
See Section 5.1.

`ext-offsetunit`

Optional. `STRING`. A means by which to extend the `offsetunit` attribute. See Section 5.1.

`instance`

Optional. `INTEGER`. Number of types to apply the specified pattern.

### 3.19.3. `RecordItem` Class

The `RecordItem` class provides a way to incorporate relevant logs, audit trails, or forensic data to support the conclusions made during the course of analyzing the incident. The class supports both the direct encapsulation of the data, as well as, provides primitives to reference data stored elsewhere.

This class is identical to `AdditionalData` class (Section 3.6).

## 4. Processing Considerations

This section defines additional requirements on creating and parsing IODEF documents.

#### 4.1. Encoding

Every IODEF document MUST begin with an XML declaration, and MUST specify the XML version used. If UTF-8 encoding is not used, the character encoding MUST also be explicitly specified. The IODEF conforms to all XML data encoding conventions and constraints.

The XML declaration with no character encoding will read as follows:

```
<?xml version="1.0" ?>
```

When a character encoding is specified, the XML declaration will read like the following:

```
<?xml version="1.0" encoding="charset" ?>
```

Where "charset" is the name of the character encoding as registered with the Internet Assigned Numbers Authority (IANA), see [9].

The following characters have special meaning in XML and MUST be escaped with their entity reference equivalent: "&", "<", ">", "\" (double quotation mark), and "'" (apostrophe). These entity references are "&amp;", "&lt;", "&gt;", "&quot;", and "&apos;" respectively.

#### 4.2. IODEF Namespace

The IODEF schema declares a namespace of "urn:ietf:params:xml:ns:iodef-1.0" and registers it per [4]. Each IODEF document SHOULD include a valid reference to the IODEF schema using the "xsi:schemaLocation" attribute. An example of such a declaration would look as follows:

```
<IODEF-Document  
  version="1.00" lang="en-US"  
  xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0"  
  xsi:schemaLocation="urn:ietf:params:xml:ns:iodef-1.0"
```

#### 4.3. Validation

The IODEF documents MUST be well-formed XML and SHOULD be validated against the schema described in Section 8. However, mere conformance to the schema is not sufficient for a semantically valid IODEF document. There is additional specification in the text of Section 3 that cannot be readily encoded in the schema and it must also be considered by an IODEF parser. The following is a list of discrepancies in what is more strictly specified in the normative text (Section 3), but not enforced in the IODEF schema:

- o The elements or attributes that are defined as POSTAL, NAME, PHONE, and EMAIL data-types are implemented as "xs:string", but more rigid formatting requirements are specified in the text.
- o The IODEF-Document@lang and MLStringType@lang attributes are declared as an "xs:language" that constrains values with a regular expression. However, the value of this attribute still needs to be validated against the list of possible enumerated values is defined in [7].
- o The MonetaryImpact@currency attribute is declared as an "xs:string", but the list of valid values as defined in [14].
- o All of the aggregated classes Contact and EventData are optional in the schema, but at least one of these aggregated classes MUST be present.
- o There are multiple conventions that can be used to categorize a system using the NodeRole class or to specify software with the Application and OperatingSystem classes. IODEF parsers MUST accept incident reports that do not use these fields in accordance with local conventions.
- o The Confidence@rating attribute determines whether the element content of Confidence should be empty.
- o The Address@type attribute determines the format of the element content.
- o The attributes AdditionalData@dtype and RecordItem@dtype derived from iodef:ExtensionType determine the semantics and formatting of the element content.
- o Symmetry in the enumerated ports of a Portlist class is required between sources and targets. See Section 3.17.

## 5. Extending the IODEF

In order to support the changing activity of CSIRTS, the IODEF data model will need to evolve along with them. This section discusses how new data elements that have no current representation in the data model can be incorporated into the IODEF. These techniques are designed so that adding new data will not require a change to the IODEF schema. With proven value, well documented extensions can be incorporated into future versions of the specification. However, this approach also supports private extensions relevant only to a closed consortium.

### 5.1. Extending the Enumerated Values of Attributes

The data model supports a means by which to add new enumerated values to an attribute. For each attribute that supports this extension technique, there is a corresponding attribute in the same element whose name is identical, less a prefix of "ext-". This special attribute is referred to as the extension attribute, and the attribute being extended is referred to as an extensible attribute. For example, an extensible attribute named "foo" will have a corresponding extension attribute named "ext-foo". An element may have many extensible, and therefore many extension, attributes.

In addition to a corresponding extension attribute, each extensible attribute has "ext-value" as one its possible values. This particular value serves as an escape sequence and has no valid meaning.

In order to add a new enumerated value to an extensible attribute, the value of this attribute **MUST** be set to "ext-value", and the new desired value **MUST** be set in the corresponding extension attribute. For example, an extended instance of the type attribute of the Impact class would look as follows:

```
<Impact type="ext-value" ext-type="new-attack-type">
```

A given extension attribute **MUST NOT** be set unless the corresponding extensible attribute has been set to "ext-value".

### 5.2. Extending Classes

The classes of the data model can be extended only through the use of the AdditionalData and RecordItem classes. These container classes, collectively referred to as the extensible classes, are implemented with the iodef:ExtensionType data type in the schema. They provide the ability to have new atomic or XML-encoded data elements in all of the top-level classes of the Incident class and a few of the more complicated subordinate classes. As there are multiple instances of the extensible classes in the data model, there is discretion on where to add a new data element. It is **RECOMMENDED** that the extension be placed in the most closely related class to the new information.

Extensions using the atomic data types (i.e., all values of the dtype attributes other than "xml") **MUST**:

1. Set the element content of extensible class to the desired value, and

2. Set the dtype attribute to correspond to the data type of the element content.

The following guidelines exist for extensions using XML:

1. The element content of the extensible class MUST be set to the desired value and the dtype attribute MUST be set to "xml".
2. The extension schema MUST declare a separate namespace. It is RECOMMENDED that these extensions have the prefix "iodef-".
3. It is RECOMMENDED that extension schemas follow the naming convention of the IODEF data model. The names of all elements are capitalized. For composed names, a capital letter is used for each word. Attribute names are lower case.
4. When a parser encounters an IODEF document with an extension it does not understand, this extension MUST be ignored (and not processed), but the remainder of the document MUST be processed. Parsers will be able to identify these extensions for which they have no processing logic through the namespace declaration. Parsers that encounter an unrecognized element in a namespace that they do support SHOULD reject the document as a syntax error.
5. Implementations SHOULD NOT download schemas at runtime due to the security implications, and extensions MUST NOT be required to provide a resolvable location of their schema.

The following schema and XML document excerpt provide a template for an extension schema and its use in the IODEF document.

This example schema defines a namespace of "iodef-extension1" and a single element named "newdata".

```
<xs:schema
  targetNamespace="iodef-extension1.xsd"
  xmlns:iodef-extension1="iodef-extension1.xsd"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  attributeFormDefault="unqualified"
  elementFormDefault="qualified">
  <xs:import
    namespace="urn:ietf:params:xml:ns:iodef-1.0"
    schemaLocation=" urn:ietf:params:xml:ns:iodef-1.0"/>

    <xs:element name="newdata" type="xs:string" />
</xs:schema>
```

The following XML excerpt demonstrates the use of the above schema as an extension to the IODEF.

```
<IODEF-Document
  version="1.00" lang="en-US"
  xmlns="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:iodef-extension1="iodef-extension1.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="iodef-extension1.xsd">
  <Incident purpose="reporting">
  ...
  <AdditionalData dtype="xml" meaning="xml">
    <iodef-extension1:newdata>
      Field that could not be represented elsewhere
    </iodef-extension1:newdata>
  </AdditionalData>
</IODEF-Document>
```

## 6. Internationalization Issues

Internationalization and localization is of specific concern to the IODEF, since it is only through collaboration, often across language barriers, that certain incidents be resolved. The IODEF supports this goal by depending on XML constructs, and through explicit design choices in the data model.

Since IODEF is implemented as an XML Schema, it implicitly supports all the different character encodings, such as UTF-8 and UTF-16, possible with XML. Additionally, each IODEF document MUST specify the language in which their contents are encoded. The language can be specified with the attribute "xml:lang" (per Section 2.12 of [1]) in the top-level element (i.e., IODEF-Document@lang) and letting all other elements inherit that definition. All IODEF classes with a free-form text definition (i.e., all those defined of type iodef:MLStringType) can also specify a language different from the rest of the document. The valid language codes for the "xml:lang" attribute are described in RFC 4646 [7].

The data model supports multiple translations of free-form text. In the places where free-text is used for descriptive purposes, the given class always has a one-to-many cardinality to its parent (e.g., Description class). The intent is to allow the identical text to be encoded in different instances of the same class, but each being in a different language. This approach allows an IODEF document author to send recipients speaking different languages an identical document. The IODEF parser SHOULD extract the appropriate language relevant to the recipient.

While the intent of the data model is to provide internationalization and localization, the intent is not to do so at the detriment of interoperability. While the IODEF does support different languages, the data model also relies heavily on standardized enumerated attributes that can crudely approximate the contents of the document. With this approach, a CSIRT should be able to make some sense of an IODEF document it receives even if the text based data elements are written in a language unfamiliar to the analyst.

## 7. Examples

This section provides examples of an incident encoded in the IODEF. These examples do not necessarily represent the only way to encode a particular incident.

### 7.1. Worm

An example of a CSIRT reporting an instance of the Code Red worm.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- This example demonstrates a report for a very
      old worm (Code Red) -->
<IODEF version="1.00" lang="en"
  xmlns="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:iodef-1.0">
  <Incident purpose="reporting">
    <IncidentID name="csirt.example.com">189493</IncidentID>
    <ReportTime>2001-09-13T23:19:24+00:00</ReportTime>
    <Description>Host sending out Code Red probes</Description>
    <!-- An administrative privilege was attempted, but failed -->
    <Assessment>
      <Impact completion="failed" type="admin"/>
    </Assessment>
    <Contact role="creator" type="organization">
      <ContactName>Example.com CSIRT</ContactName>
      <RegistryHandle registry="arin">example-com</RegistryHandle>
      <Email>contact@csirt.example.com</Email>
    </Contact>
    <EventData>
      <Flow>
        <System category="source">
          <Node>
            <Address category="ipv4-addr">192.0.2.200</Address>
            <Counter type="event">57</Counter>
          </Node>
```

```

    </System>
    <System category="target">
      <Node>
        <Address category="ipv4-net">192.0.2.16/28</Address>
      </Node>
      <Service ip_protocol="6">
        <Port>80</Port>
      </Service>
    </System>
  </Flow>
  <Expectation action="block-host" />
  <!-- <RecordItem> has an excerpt from a log -->
  <Record>
    <RecordData>
      <DateTime>2001-09-13T18:11:21+02:00</DateTime>
      <Description>Web-server logs</Description>
      <RecordItem dtype="string">
        192.0.2.1 - - [13/Sep/2001:18:11:21 +0200] "GET /default.ida?
        XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
        XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
        XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
        XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
      </RecordItem>
      <!-- Additional logs -->
      <RecordItem dtype="url">
        http://mylogs.example.com/logs/httpd_access</RecordItem>
    </RecordData>
  </Record>
</EventData>
<History>
  <!-- Contact was previously made with the source network owner -->
  <HistoryItem action="contact-source-site">
    <DateTime>2001-09-14T08:19:01+00:00</DateTime>
    <Description>Notification sent to
      constituency-contact@192.0.2.200</Description>
  </HistoryItem>
</History>
</Incident>
</IODEF-Document>

```

## 7.2. Reconnaissance

An example of a CSIRT reporting a scanning activity.

```
<?xml version="1.0" encoding="UTF-8" ?>
```



```
<!-- This example describes reconnaissance activity: one-to-one and
one-to-many scanning -->
<IODEF-Document version="1.00" lang="en"
  xmlns="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:schema:iodef-1.0">
  <Incident purpose="reporting">
    <IncidentID name="csirt.example.com">59334</IncidentID>
    <ReportTime>2006-08-02T05:54:02-05:00</ReportTime>
    <Assessment>
      <Impact type="recon" completion="succeeded" />
    </Assessment>
    <Method>
      <!-- Reference to the scanning tool "nmap" -->
      <Reference>
        <ReferenceName>nmap</ReferenceName>
        <URL>http://nmap.toolsite.example.com</URL>
      </Reference>
    </Method>
    <!-- Organizational contact and that for staff in that
    organization -->
    <Contact role="creator" type="organization">
      <ContactName>CSIRT for example.com</ContactName>
      <Email>contact@csirt.example.com</Email>
      <Telephone>+1 412 555 12345</Telephone>
      <!-- Since this <Contact> is nested, Joe Smith is part of the
      CSIRT for example.com -->
      <Contact role="tech" type="person" restriction="need-to-know">
        <ContactName>Joe Smith</ContactName>
        <Email>smith@csirt.example.com</Email>
      </Contact>
    </Contact>
    <EventData>
      <!-- Scanning activity as follows:
      192.0.2.1:60524 >> 192.0.2.3:137
      192.0.2.1:60526 >> 192.0.2.3:138
      192.0.2.1:60527 >> 192.0.2.3:139
      192.0.2.1:60531 >> 192.0.2.3:445
      -->
    <Flow>
      <System category="source">
        <Node>
          <Address category="ipv4-addr">192.0.2.200</Address>
        </Node>
        <Service ip_protocol="6">
          <Portlist>60524,60526,60527,60531</Portlist>
        </Service>
      </System>
```

```

    <System category="target">
      <Node>
        <Address category="ipv4-addr">192.0.2.201</Address>
      </Node>
      <Service ip_protocol="6">
        <Portlist>137-139,445</Portlist>
      </Service>
    </System>
  </Flow>
  <!-- Scanning activity as follows:
        192.0.2.2 >> 192.0.2.3/28:445 -->
  <Flow>
    <System category="source">
      <Node>
        <Address category="ipv4-addr">192.0.2.240</Address>
      </Node>
    </System>
    <System category="target">
      <Node>
        <Address category="ipv4-net">192.0.2.64/28</Address>
      </Node>
      <Service ip_protocol="6">
        <Port>445</Port>
      </Service>
    </System>
  </Flow>
</EventData>
</Incident>
</IODEF-Document>

```

### 7.3. Bot-Net Reporting

An example of a CSIRT reporting a bot-network.

```

<?xml version="1.0" encoding="UTF-8" ?>
<!-- This example describes a compromise and subsequent installation
      of bots -->
<IODEF-Document version="1.00" lang="en"
  xmlns="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:schema:iodef-1.0">
  <Incident purpose="mitigation">
    <IncidentID name="csirt.example.com">908711</IncidentID>
    <ReportTime>2006-06-08T05:44:53-05:00</ReportTime>
    <Description>Large bot-net</Description>
  </Incident>
</IODEF-Document>

```

```
<Assessment>
  <Impact type="dos" severity="high" completion="succeeded" />
</Assessment>
<Method>
  <!-- References a given piece of malware, "GT Bot" -->
  <Reference>
    <ReferenceName>GT Bot</ReferenceName>
  </Reference>
  <!-- References the vulnerability used to compromise the
        machines -->
  <Reference>
    <ReferenceName>CA-2003-22</ReferenceName>
    <URL>http://www.cert.org/advisories/CA-2003-22.html</URL>
    <Description>Root compromise via this IE vulnerability to
                  install the GT Bot</Description>
  </Reference>
</Method>
<!-- A member of the CSIRT that is coordinating this
      incident -->
<Contact type="person" role="irt">
  <ContactName>Joe Smith</ContactName>
  <Email>jsmith@csirt.example.com</Email>
</Contact>
<EventData>
  <Description>These hosts are compromised and acting as bots
                communicating with irc.example.com.</Description>
  <Flow>
    <!-- bot running on 192.0.2.1 and sending DoS traffic at
          10,000 bytes/second -->
    <System category="source">
      <Node>
        <Address category="ipv4-addr">192.0.2.1</Address>
      </Node>
      <Counter type="byte" duration="second">10000</Counter>
      <Description>bot</Description>
    </System>
    <!-- a second bot on 192.0.2.3 -->
    <System category="source">
      <Node>
        <Address category="ipv4-addr">192.0.2.3</Address>
      </Node>
      <Counter type="byte" duration="second">250000</Counter>
      <Description>bot</Description>
    </System>
    <!-- Command-and-control IRC server for these bots-->
    <System category="intermediate">
      <Node>
        <NodeName>irc.example.com</NodeName>
```

```

        <Address category="ipv4-addr">192.0.2.20</Address>
        <DateTime>2006-06-08T01:01:03-05:00</DateTime>
    </Node>
    <Description>IRC server on #give-me-cmd channel</Description>
</System>
</Flow>
<!-- Request to take these machines offline -->
<Expectation action="investigate">
    <Description>Confirm the source and take machines off-line and
        remediate</Description>
</Expectation>
</EventData>
</Incident>
</IODEF-Document>

```

#### 7.4. Watch List

An example of a CSIRT conveying a watch-list.

```

<?xml version="1.0" encoding="UTF-8" ?>
<!-- This example demonstrates a trivial IP watch-list -->
<!-- @formatid is set to "watch-list-043" to demonstrate how additional
    semantics about this document could be conveyed assuming both
    parties understood it-->
<IODEF-Document version="1.00" lang="en" formatid="watch-list-043"
    xmlns="urn:ietf:params:xml:ns:iodef-1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:iodef-1.0">
    <Incident purpose="reporting" restriction="private">
        <IncidentID name="csirt.example.com">908711</IncidentID>
        <ReportTime>2006-08-01T00:00:00-05:00</ReportTime>
        <Description>Watch-list of known bad IPs or networks</Description>
        <Assessment>
            <Impact type="admin" completion="succeeded" />
            <Impact type="recon" completion="succeeded" />
        </Assessment>
        <Contact type="organization" role="creator">
            <ContactName>CSIRT for example.com</ContactName>
            <Email>contact@csirt.example.com</Email>
        </Contact>
        <!-- Separate <EventData> used to convey different <Expectation> -->
        <EventData>
            <Flow>
                <System category="source">
                    <Node>

```

```

        <Address category="ipv4-addr">192.0.2.53</Address>
      </Node>
      <Description>Source of numerous attacks</Description>
    </System>
  </Flow>
  <!-- Expectation class indicating that sender of list would like
        to be notified if activity from the host is seen -->
    <Expectation action="contact-sender" />
</EventData>
<EventData>
  <Flow>
    <System category="source">
      <Node>
        <Address category="ipv4-net">192.0.2.16/28</Address>
      </Node>
      <Description>
        Source of heavy scanning over past 1-month
      </Description>
    </System>
  </Flow>
  <Flow>
    <System category="source">
      <Node>
        <Address category="ipv4-addr">192.0.2.241</Address>
      </Node>
      <Description>C2 IRC server</Description>
    </System>
  </Flow>
  <!-- Expectation class recommends that these networks
        be filtered -->
    <Expectation action="block-host" />
</EventData>
</Incident>
</IODEF-Document>

```

## 8. The IODEF Schema

```

<xs:schema targetNamespace="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">
  <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"

```

```
    schemaLocation="http://www.w3.org/TR/2002/REC-xmlsig-core-20020212/xmlsig-
core-schema.xsd"/>
    <xs:annotation>
      <xs:documentation>
        Incident Object Description Exchange Format v1.10, RFC5070-bis
      </xs:documentation>
    </xs:annotation>

<!-- CHANGE: See above addition of xmlns:ds and import of same
      namespace. This is to use the digital signature hash inclusion
      of a file by referencing the existing standard as was done in
      RFC5901, RFC3275 is the reference, see RFC5901 section 5.9.5.2
-->
<!--
=====
===  List of changes                                     ===
=====
CHANGE - new indicator values in the schema

The purpose of the proposed changes is to include commonly shared
indicators in the base IODEF schema. This class will contain
indicators from the list below that are not represented elsewhere
in the schema. IODEF extensions or embedded schemas via the SCI
classes will be required to include additional data types.
A table could be maintained through IANA to extend or change this
class in between IODEF revisions.

RFC5901 provides a method to include an entire email, the following
included indicators are ones commonly used when you do not need the
entire email
The following are in the Service Class:
    Email address
    Email subject
    X-Mailer
The following are in the Record class:
    File Name
    File Hash - 5.9.5.2 - using ds:reference
    WindowsRegistryKey - using method from RFC5901
The following are now in the Node class as a proposed location.
    URL
    HTTPUserAgent is included as a SoftwareType
    HTTP User Agent String
The following are already represented elsewhere in the schema (Node):
    IP address
    Network CIDR / ASN
    Host Name
    Domain Name (additional options for RFC5901 were not included in
    this revision - can include point-in-time dig info)
-->
```

```

<!--
=====
== IODEF-Document class                                     ==
=====
-->
  <xs:element name="IODEF-Document">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:Incident"
                      maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version"
                    type="xs:string" fixed="1.00"/>
      <xs:attribute name="lang"
                    type="xs:language" use="required"/>
      <xs:attribute name="formatid"
                    type="xs:string"/>
    </xs:complexType>
  </xs:element>
<!--
=====
=== Incident class                                         ===
=====
-->
  <xs:element name="Incident">
    <xs:complexType>
      <xs:sequence>
        <xs:choice>
          <xs:element ref="iodef:IncidentID"/>
          <!-- CHANGE - the incidentID can still be used,
               but when you have a set of indictaors or include
               a watch list, a ReportID may be preferred. If
               this is agreed upon, do we make them both unique
               so the same key can be used in databases? This
               should not be used as your index value unless you
               are the issuing entity. -->
          <xs:element name="ReportID" type="IncidentIDType"/>
        </xs:choice>
        <xs:element ref="iodef:AlternativeID"
                      minOccurs="0"/>
        <xs:element ref="iodef:RelatedActivity"
                      minOccurs="0"/>
        <xs:element ref="iodef:DetectTime"
                      minOccurs="0"/>
        <xs:element ref="iodef:StartTime"
                      minOccurs="0"/>
        <xs:element ref="iodef:EndTime"
                      minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

```

    <xs:element ref="iodef:ReportTime"/>
    <xs:element ref="iodef:Description"
      minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="iodef:Assessment"
      maxOccurs="unbounded"/>
    <xs:element ref="iodef:Method"
      minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="iodef:Contact"
      maxOccurs="unbounded"/>
    <xs:element ref="iodef:EventData"
      minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="iodef:History"
      minOccurs="0"/>
    <xs:element ref="iodef:AdditionalData"
      minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="purpose" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="traceback"/>
        <xs:enumeration value="mitigation"/>
        <xs:enumeration value="reporting"/>
        <xs:enumeration value="other"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="ext-purpose"
    type="xs:string" use="optional"/>
  <xs:attribute name="lang"
    type="xs:language"/>
  <xs:attribute name="restriction"
    type="iodef:restriction-type" default="private"/>
  <!-- CHANGE - adding an attribute to mark sets of indicators -->
  <xs:attribute name="indicator-set-id"
    type="xs:string" use="optional"/>
</xs:complexType>
</xs:element>
<!--
=====
== IncidentID class ==
=====
-->
  <xs:element name="IncidentID" type="iodef:IncidentIDType"/>
  <xs:complexType name="IncidentIDType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="name"

```



```

        type="xs:string" use="required"/>
      <xs:attribute name="instance"
        type="xs:string" use="optional"/>
      <xs:attribute name="restriction"
        type="iodef:restriction-type" default="public"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<!--
=====
==  AlternativeID class                                ==
=====
-->
  <xs:element name="AlternativeID">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:IncidentID"
          maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="restriction"
        type="iodef:restriction-type"/>
    </xs:complexType>
  </xs:element>
<!--
=====
==  RelatedActivity class                                ==
=====
-->
  <xs:element name="RelatedActivity">
    <xs:complexType>
      <xs:choice>
        <xs:element ref="iodef:IncidentID"
          maxOccurs="unbounded"/>
        <xs:element ref="iodef:URL"
          maxOccurs="unbounded"/>
      </xs:choice>
      <xs:attribute name="restriction"
        type="iodef:restriction-type"/>
    </xs:complexType>
  </xs:element>
<!--
=====
===  AdditionalData class                                ===
=====
-->
  <xs:element name="AdditionalData" type="iodef:ExtensionType"/>
<!--

```

```
=====
===  Contact class                                     ===
=====

-->
<xs:element name="Contact">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="iodef:ContactName"
        minOccurs="0"/>
      <xs:element ref="iodef:Description"
        minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="iodef:RegistryHandle"
        minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="iodef:PostalAddress"
        minOccurs="0"/>
      <xs:element ref="iodef:Email"
        minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="iodef:Telephone"
        minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="iodef:Fax"
        minOccurs="0"/>
      <xs:element ref="iodef:Timezone"
        minOccurs="0"/>
      <xs:element ref="iodef:Contact"
        minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="iodef:AdditionalData"
        minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="role" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
          <xs:enumeration value="creator"/>
          <xs:enumeration value="admin"/>
          <xs:enumeration value="tech"/>
          <xs:enumeration value="irt"/>
          <xs:enumeration value="cc"/>
          <xs:enumeration value="ext-value"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="ext-role"
      type="xs:string" use="optional"/>
    <xs:attribute name="type" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
          <xs:enumeration value="person"/>
          <xs:enumeration value="organization"/>
          <xs:enumeration value="ext-value"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

```
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="ext-type"
      type="xs:string" use="optional"/>
    <xs:attribute name="restriction"
      type="iodef:restriction-type"/>
  </xs:complexType>
</xs:element>
<!-- CHANGE - UML states the type disambiguates the type of Name
person or organization. Do we want this added to the schema? -->
<xs:element name="ContactName"
  type="iodef:MLStringType"/>
<xs:element name="RegistryHandle">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="registry">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="internic"/>
              <xs:enumeration value="apnic"/>
              <xs:enumeration value="arin"/>
              <xs:enumeration value="lacnic"/>
              <xs:enumeration value="ripe"/>
              <xs:enumeration value="afrinic"/>
              <xs:enumeration value="local"/>
              <xs:enumeration value="ext-value"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ext-registry"
          type="xs:string" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name="PostalAddress">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="iodef:MLStringType">
        <xs:attribute name="meaning"
          type="xs:string" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

```
<xs:element name="Email" type="iodef:ContactMeansType"/>
<xs:element name="Telephone" type="iodef:ContactMeansType"/>
<xs:element name="Fax" type="iodef:ContactMeansType"/>

<xs:complexType name="ContactMeansType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="meaning"
                    type="xs:string" use="optional"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<!--
=====
===  Time-based classes                                     ===
=====
-->
  <xs:element name="DateTime"
              type="xs:dateTime"/>
  <xs:element name="ReportTime"
              type="xs:dateTime"/>
  <xs:element name="DetectTime"
              type="xs:dateTime"/>
  <xs:element name="StartTime"
              type="xs:dateTime"/>
  <xs:element name="EndTime"
              type="xs:dateTime"/>
  <xs:element name="Timezone"
              type="iodef:TimezoneType"/>
  <xs:simpleType name="TimezoneType">
    <xs:restriction base="xs:string">
      <xs:pattern value="Z|[\+\-](0[0-9]|1[0-4]):[0-5][0-9]"/>
    </xs:restriction>
  </xs:simpleType>
<!--
=====
===  History class                                         ===
=====
-->
  <xs:element name="History">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:HistoryItem"
                      maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="restriction"
                    type="iodef:restriction-type" default="default"/>
    </xs:complexType>
  </xs:element>
```

```

    </xs:complexType>
  </xs:element>
  <xs:element name="HistoryItem">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:DateTime"/>
        <xs:element ref="iodef:IncidentID"
          minOccurs="0"/>
        <xs:element ref="iodef:Contact"
          minOccurs="0"/>
        <xs:element ref="iodef:Description"
          minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="iodef:AdditionalData"
          minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="restriction"
        type="iodef:restriction-type"/>
      <xs:attribute name="action"
        type="iodef:action-type" use="required"/>
      <xs:attribute name="ext-action"
        type="xs:string" use="optional"/>
      <!-- CHANGE: Including a unique ID for indicators, may be
        used to connect indicators in different representations
      -->
      <xs:attribute name="indicator-uid"
        type="xs:string" use="optional"/>
      <!-- CHANGE: Including an indicator set ID that may be used
        to detail changes in the history class as it relates to
        indicators or sets.
      -->
      <xs:attribute name="indicator-set-id"
        type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>
<!--
=====
===  Expectation class                                ===
=====
-->
  <xs:element name="Expectation">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:Description"
          minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="iodef:StartTime"
          minOccurs="0"/>
        <xs:element ref="iodef:EndTime"
          minOccurs="0"/>

```

```

        <xs:element ref="iodef:Contact"
                    minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="restriction"
                  type="iodef:restriction-type" default="default"/>
    <xs:attribute name="severity"
                  type="iodef:severity-type"/>
    <xs:attribute name="action"
                  type="iodef:action-type" default="other"/>
    <xs:attribute name="ext-action"
                  type="xs:string" use="optional"/>
    <!-- CHANGE - adding indicator set id to connect the reference
           to the appropriate set of indicators -->
    <xs:attribute name="indicator-set-id"
                  type="xs:string" use="optional"/>
    <!-- CHANGE: Including a unique ID for indicators, may be
           used to connect indicators in different representations
    -->
    <xs:attribute name="indicator-uid"
                  type="xs:string" use="optional"/>
  </xs:complexType>
</xs:element>
<!--
=====
===  Method class                                ===
=====
-->
<xs:element name="Method">
  <xs:complexType>
    <xs:sequence>
      <xs:choice maxOccurs="unbounded">
        <xs:element ref="iodef:Reference"/>
        <xs:element ref="iodef:Description"/>
      </xs:choice>
      <xs:element ref="iodef:AdditionalData"
                    minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="restriction"
                  type="iodef:restriction-type"/>
  </xs:complexType>
</xs:element>
<!--
=====
===  Reference class                                ===
=====
-->
<xs:element name="Reference">
  <xs:complexType>

```

```

<xs:sequence>
  <xs:element name="ReferenceName"
    type="iodef:MLStringType"/>
  <xs:element ref="iodef:URL"
    minOccurs="0" maxOccurs="unbounded"/>
  <xs:element ref="iodef:Description"
    minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<!-- CHANGE: Do we want an indicator_set_id here to connect
data in the reference class to specific indicators?
is there a better way to do this?
Should the indicator_uid be used to mark data so that
you have a way to limit who you share that data with
in products?
-->
<xs:attribute name="indicator-set-id"
type="xs:string" use="optional"/>
<!-- CHANGE: Including a unique ID for indicators, may be
used to connect indicators in different representations
-->
<xs:attribute name="indicator-uid"
type="xs:string" use="optional"/>
<!-- Adding in Attack Type -->
<xs:attribute name="attacktype" type="att-type" use="required">
</xs:attribute>
<xs:attribute name="ext-attacktype"
type="xs:string" use="optional"/>
</xs:complexType>
</xs:element>
<!--
=====
===  Assessment class  ===
=====
-->
<xs:element name="Assessment">
  <xs:complexType>
    <xs:sequence>
      <xs:choice maxOccurs="unbounded">
        <xs:element ref="iodef:Impact"/>
        <xs:element ref="iodef:TimeImpact"/>
        <xs:element ref="iodef:MonetaryImpact"/>
      </xs:choice>
      <xs:element ref="iodef:Counter"
        minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="iodef:Confidence" minOccurs="0"/>
      <xs:element ref="iodef:AdditionalData"
        minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```
<xs:attribute name="occurrence">
  <xs:simpleType>
    <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="actual"/>
      <xs:enumeration value="potential"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
<xs:attribute name="restriction"
type="iodef:restriction-type"/>
<!-- CHANGE: Including an indicator set ID for indicators, may be
used to connect indicators in different representations
-->
<xs:attribute name="indicator-set-id"
type="xs:string" use="optional"/>
<!-- CHANGE: Including a unique ID for indicators, may be
used to connect indicators in different representations.
May need separate confidence ratings for different indicators.
-->
<xs:attribute name="indicator-uid"
type="xs:string" use="optional"/>
</xs:complexType>
</xs:element>
<xs:element name="Impact">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="iodef:MLStringType">
        <xs:attribute name="severity"
type="iodef:severity-type"/>
        <xs:attribute name="completion">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="failed"/>
              <xs:enumeration value="succeeded"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="type"
use="optional" default="unknown">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <!-- CHANGE question: do we want to allow multiple values
to be selected in case it is a combination? -->
              <xs:enumeration value="admin"/>
              <xs:enumeration value="dos"/>
              <xs:enumeration value="extortion"/>
              <xs:enumeration value="file"/>
              <xs:enumeration value="info-leak"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```



```
        <xs:enumeration value="misconfiguration"/>
        <xs:enumeration value="recon"/>
        <xs:enumeration value="policy"/>
        <xs:enumeration value="social-engineering"/>
        <xs:enumeration value="user"/>
        <xs:enumeration value="unknown"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="ext-type"
    type="xs:string" use="optional"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="TimeImpact">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="iodef:PositiveFloatType">
        <xs:attribute name="severity"
          type="iodef:severity-type"/>
        <xs:attribute name="metric"
          use="required">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="labor"/>
              <xs:enumeration value="elapsed"/>
              <xs:enumeration value="downtime"/>
              <xs:enumeration value="ext-value"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ext-metric"
          type="xs:string" use="optional"/>
        <xs:attribute name="duration"
          type="iodef:duration-type"/>
        <xs:attribute name="ext-duration"
          type="xs:string" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
<xs:element name="MonetaryImpact">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="iodef:PositiveFloatType">
        <xs:attribute name="severity"
```

```

        type="iodef:severity-type"/>
        <xs:attribute name="currency"
            type="xs:string"/>
    </xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="Confidence">
    <xs:complexType mixed="true">
        <xs:attribute name="rating" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="low"/>
                    <xs:enumeration value="medium"/>
                    <xs:enumeration value="high"/>
                    <xs:enumeration value="numeric"/>
                    <xs:enumeration value="unknown"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>
</xs:element>
<!--
=====
===  EventData  class                                     ===
=====
-->
<xs:element name="EventData">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="iodef:Description"
                minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="iodef:DetectTime"
                minOccurs="0"/>
            <xs:element ref="iodef:StartTime"
                minOccurs="0"/>
            <xs:element ref="iodef:EndTime"
                minOccurs="0"/>
            <xs:element ref="iodef:Contact"
                minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="iodef:Assessment"
                minOccurs="0"/>
            <xs:element ref="iodef:Method"
                minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="iodef:Flow"
                minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="iodef:Expectation"
                minOccurs="0" maxOccurs="unbounded"/>

```

```

    <xs:element ref="iodef:Record"
        minOccurs="0"/>
    <xs:element ref="iodef:EventData"
        minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="iodef:AdditionalData"
        minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="restriction"
    type="iodef:restriction-type" default="default"/>
<!-- CHANGE - adding an attribute to mark sets of indicators -->
<xs:attribute name="indicator-set-id"
    type="xs:string" use="optional"/>
</xs:complexType>
</xs:element>
<!--
=====
===  Flow class                                     ===
=====
-->
    <xs:element name="Flow">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="iodef:System"
                    maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
<!--
=====
===  System class                                     ===
=====
-->
    <xs:element name="System">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="iodef:Node"/>
                <xs:element ref="iodef:Service"
                    minOccurs="0" maxOccurs="unbounded"/>
                <xs:element ref="iodef:OperatingSystem"
                    minOccurs="0" maxOccurs="unbounded"/>
                <xs:element ref="iodef:Counter"
                    minOccurs="0" maxOccurs="unbounded"/>
                <xs:element ref="iodef:Description"
                    minOccurs="0" maxOccurs="unbounded"/>
                <xs:element ref="iodef:AdditionalData"
                    minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="restriction"

```

```

        type="iodef:restriction-type"/>
<xs:attribute name="interface"
    type="xs:string"/>
<xs:attribute name="category">
    <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="source"/>
            <xs:enumeration value="target"/>
            <!-- CHANGE - adding two new values to cover watchlist groups -->
            <xs:enumeration value="watchlist-source"/>
            <xs:enumeration value="watchlist-target"/>
            <xs:enumeration value="intermediate"/>
            <xs:enumeration value="sensor"/>
            <xs:enumeration value="infrastructure"/>
            <xs:enumeration value="ext-value"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
<xs:attribute name="ext-category"
    type="xs:string" use="optional"/>
<!-- CHANGE - adding an attribute to mark sets of indicators -->
<xs:attribute name="indicator-set-id"
    type="xs:string" use="optional"/>
<xs:attribute name="spoofed"
    default="unknown">
    <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="unknown"/>
            <xs:enumeration value="yes"/>
            <xs:enumeration value="no"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
<!--
=====
=== Node class                                     ===
=====
-->
<xs:element name="Node">
    <xs:complexType>
        <xs:sequence>
            <xs:choice maxOccurs="unbounded">
                <xs:element name="NodeName"
                    type="iodef:MLStringType" minOccurs="0"/>
                <xs:element ref="iodef:Address"
                    minOccurs="0" maxOccurs="unbounded"/>
            </xs:choice>
        </xs:sequence>
    </xs:complexType>
</xs:element>

```

```
<!-- Proposed CHANGE: include a URI indicator.
Common complaint that URIs were only in the
IODEF schema as references and not part of the
incident or included indicators.
Included right now as an address type, below is a
second option for how to add it.
<xs:element ref="iodef:URL"
minOccurs="0" maxOccurs="unbounded"/>
-->
</xs:choice>
<xs:element ref="iodef:Location"
minOccurs="0"/>
<xs:element ref="iodef:DateTime"
minOccurs="0"/>
<xs:element ref="iodef:NodeRole"
minOccurs="0" maxOccurs="unbounded"/>
<xs:element ref="iodef:Counter"
minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Address">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="category" default="ipv4-addr">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="asn"/>
              <xs:enumeration value="atm"/>
              <xs:enumeration value="e-mail"/>
              <xs:enumeration value="mac"/>
              <xs:enumeration value="ipv4-addr"/>
              <xs:enumeration value="ipv4-net"/>
              <xs:enumeration value="ipv4-net-mask"/>
              <xs:enumeration value="ipv6-addr"/>
              <xs:enumeration value="ipv6-net"/>
              <xs:enumeration value="ipv6-net-mask"/>
              <!-- CHANGE - added uri type for site url/uris -->
              <xs:enumeration value="site-uri"/>
              <xs:enumeration value="ext-value"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ext-category"
          type="xs:string" use="optional"/>
        <xs:attribute name="vlan-name"
          type="xs:string"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

```
        <xs:attribute name="vlan-num"
                      type="xs:integer"/>
<!-- CHANGE: Including a unique ID for indicators, may be
         used to connect indicators in different representations
-->
        <xs:attribute name="indicator-uid"
                      type="xs:string" use="optional"/>

    </xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="Location" type="iodef:MLStringType"/>
<xs:element name="NodeRole">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="iodef:MLStringType">
        <xs:attribute name="category" use="required">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="client"/>
              <xs:enumeration value="server-internal"/>
              <xs:enumeration value="server-public"/>
              <xs:enumeration value="www"/>
              <xs:enumeration value="mail"/>
              <xs:enumeration value="messaging"/>
              <xs:enumeration value="streaming"/>
              <xs:enumeration value="voice"/>
              <xs:enumeration value="file"/>
              <xs:enumeration value="ftp"/>
              <xs:enumeration value="p2p"/>
              <xs:enumeration value="name"/>
              <xs:enumeration value="directory"/>
              <xs:enumeration value="credential"/>
              <xs:enumeration value="print"/>
              <xs:enumeration value="application"/>
              <xs:enumeration value="database"/>
              <xs:enumeration value="infra"/>
              <xs:enumeration value="log"/>
              <xs:enumeration value="ext-value"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ext-category"
                      type="xs:string" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
```

```

    </xs:element>
<!--
=====
===  Service Class  ===
=====
-->
  <xs:element name="Service">
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs="0">
          <xs:element name="Port"
            type="xs:integer"/>
          <xs:element name="Portlist"
            type="iodef:PortlistType"/>
        </xs:choice>
        <xs:element name="ProtoType"
          type="xs:integer" minOccurs="0"/>
        <xs:element name="ProtoCode"
          type="xs:integer" minOccurs="0"/>
        <xs:element name="ProtoField"
          type="xs:integer" minOccurs="0"/>
        <xs:element ref="iodef:Application"
          minOccurs="0"/>
<!-- CHANGE - email from address indicator, may be better as a sub class?
      Would only make sense with the service set to email ports
      or none at all here or a new class. -->
        <xs:element ref="Email" minOccurs="0"/>
        <xs:element name="EmailSubject"
          type="iodef:MLStringType" minOccurs="0"/>
        <xs:element name="X-Mailer"
          type="iodef:MLStringType" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="ip_protocol"
        type="xs:integer" use="required"/>
<!-- CHANGE: Including a unique ID for indicators, may be
      used to connect indicators in different representations
-->
      <xs:attribute name="indicator-uid"
        type="xs:string" use="optional"/>
<!-- CHANGE: Including an indicator set ID that may be used
      to detail changes in the history class as it relates to
      indicators or sets.
-->
      <xs:attribute name="indicator-set-id"
        type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>
  <xs:simpleType name="PortlistType">

```

```

    <xs:restriction base="xs:string">
      <xs:pattern value="\d+(\-\d+)?(,\d+(\-\d+)?)*"/>
    </xs:restriction>
  </xs:simpleType>
<!--
=====
=== Counter class                                     ===
=====
-->
<xs:element name="Counter">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:double">
        <xs:attribute name="type" use="required">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="byte"/>
              <xs:enumeration value="packet"/>
              <xs:enumeration value="flow"/>
              <xs:enumeration value="session"/>
              <xs:enumeration value="event"/>
              <xs:enumeration value="alert"/>
              <xs:enumeration value="message"/>
              <xs:enumeration value="host"/>
              <xs:enumeration value="site"/>
              <xs:enumeration value="organization"/>
              <xs:enumeration value="ext-value"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ext-type"
                      type="xs:string" use="optional"/>
        <xs:attribute name="meaning"
                      type="xs:string" use="optional"/>
        <xs:attribute name="duration"
                      type="iodef:duration-type"/>
        <xs:attribute name="ext-duration"
                      type="xs:string" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
<!--
=====
=== Record class                                     ===
=====
-->
<xs:element name="Record">

```



```

    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:RecordData"
          maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="restriction"
        type="iodef:restriction-type"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="RecordData">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:DateTime"
          minOccurs="0"/>
        <xs:element ref="iodef:Description"
          minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="iodef:Application"
          minOccurs="0"/>
        <xs:element ref="iodef:RecordPattern"
          minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="iodef:RecordItem"
          maxOccurs="unbounded"/>

        <!-- CHANGE: File name and hash of file indicator information -->
        <xs:element name="FileName"
          type="iodef:MLStringType" minOccurs="0"/>
        <!-- Represent file hash information via digsig schema
          Reference class -->
        <xs:element ref="ds:Reference" minOccurs="0"/>
        <!-- CHANGE: Windows Registry Key Modifications:
          Here, we include the classes from iodef-phish, to
          prevent the need to pull in the full schema.
          Ensure reference to RFC5901 Section 5.9.7 remains
          included in UML description.
        -->
        <xs:element name="WindowsRegistryKeysModified" type="RegistryKeyModifie
d"
          minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="iodef:AdditionalData"
          minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="restriction"
        type="iodef:restriction-type"/>
        <!-- CHANGE: Including a unique ID for an indicator.
        -->
        <xs:attribute name="indicator-uid"
          type="xs:string" use="optional"/>
        <!-- CHANGE: Including a unique ID for sets of indicators, may be
          used to connect indicators in different representations

```

```

-->
  <xs:attribute name="indicator-set-id"
                type="xs:string" use="optional"/>
</xs:complexType>
</xs:element>

<xs:element name="RecordPattern">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="type" use="required">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="regex"/>
              <xs:enumeration value="binary"/>
              <xs:enumeration value="xpath"/>
              <xs:enumeration value="ext-value"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ext-type"
                      type="xs:string" use="optional"/>
        <xs:attribute name="offset"
                      type="xs:integer" use="optional"/>
        <xs:attribute name="offsetunit"
                      use="optional" default="line">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="line"/>
              <xs:enumeration value="byte"/>
              <xs:enumeration value="ext-value"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ext-offsetunit"
                      type="xs:string" use="optional"/>
        <xs:attribute name="instance"
                      type="xs:integer" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
<xs:element name="RecordItem"
            type="iodef:ExtensionType"/>
<!--
=====
===  Class to describe Windows Registry Keys  ===
=====

```

```

-->
    <xs:complexType name="RegistryKeyModified">
      <xs:sequence>
        <xs:element name="Key" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <!-- Allows for the value to be optional for cases such as
,
              the registry key was deleted -->
              <xs:element name="KeyName" type="xs:string"/>
              <xs:element name="Value" type="xs:string" minOccurs="0"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:attribute name="registryaction">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="add_key"/>
              <xs:enumeration value="add_value"/>
              <xs:enumeration value="delete_key"/>
              <xs:enumeration value="delete_value"/>
              <xs:enumeration value="modify_key"/>
              <xs:enumeration value="modify_value"/>
              <xs:enumeration value="ext-value"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ext-category"
          type="xs:string" use="optional"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <!-- CHANGE: Including a unique ID for indicators, may be
    used to connect indicators in different representations
-->
    <xs:attribute name="indicator-uid"
      type="xs:string" use="optional"/>
  </xs:complexType>
</xs:element>
<!-- CHANGE: Including an indicator set ID that may be used
to detail changes in the history class as it relates to
indicators or sets.
-->
  <xs:attribute name="indicator-set-id"
    type="xs:string" use="optional"/>
</xs:sequence>
</xs:complexType>
<!-- CHANGE: Should this be broken out as another class
for WindowsRegistryKeyModified and add attributes
for indicator_ID and action - add_value, removes_value, etc.
as is demonstrated?
-->

```

```

<!--
=====
===  Classes that describe software                                ===
=====
-->
  <xs:complexType name="SoftwareType">
    <xs:sequence>
      <xs:element ref="iodef:URL"
        minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="swid"
      type="xs:string" default="0"/>
    <xs:attribute name="configid"
      type="xs:string" default="0"/>
    <xs:attribute name="vendor"
      type="xs:string"/>
    <xs:attribute name="family"
      type="xs:string"/>
    <xs:attribute name="name"
      type="xs:string"/>
    <!-- CHANGE: Should UserAgent or HTTPUserAgent fit in
      SoftwareTypes? This is typically intended to mean
      servers, but the category seems more appropriate
      than others.
    -->
    <xs:attribute name="user-agent"
      type="xs:string"/>
    <xs:attribute name="version"
      type="xs:string"/>
    <xs:attribute name="patch"
      type="xs:string"/>
  </xs:complexType>
  <xs:element name="Application"
    type="iodef:SoftwareType"/>
  <xs:element name="OperatingSystem"
    type="iodef:SoftwareType"/>

<!--
=====
===  Miscellaneous simple classes                                ===
=====
-->
  <xs:element name="Description"
    type="iodef:MLStringType"/>
  <xs:element name="URL"
    type="xs:anyURI"/>
<!--
=====

```

```

=== Data Types                                     ===
=====
-->
<xs:simpleType name="PositiveFloatType">
  <xs:restriction base="xs:float">
    <xs:minExclusive value="0"/>
  </xs:restriction>
</xs:simpleType>
<xs:complexType name="MLStringType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="lang"
                    type="xs:language" use="optional"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="ExtensionType" mixed="true">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax"
            minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="dtype"
                type="iodef:dtype-type" use="required"/>
  <xs:attribute name="ext-dtype"
                type="xs:string" use="optional"/>
  <xs:attribute name="meaning"
                type="xs:string"/>
  <xs:attribute name="formatid"
                type="xs:string"/>
  <xs:attribute name="restriction"
                type="iodef:restriction-type"/>
</xs:complexType>
<!--
=====
=== Global attribute type declarations             ===
=====
-->
<xs:simpleType name="restriction-type">
  <xs:restriction base="xs:NMTOKEN">
    <xs:enumeration value="default"/>
    <xs:enumeration value="public"/>
    <xs:enumeration value="need-to-know"/>
    <xs:enumeration value="private"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="severity-type">
  <xs:restriction base="xs:NMTOKEN">

```

```
<xs:enumeration value="low"/>
<xs:enumeration value="medium"/>
<xs:enumeration value="high"/>
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="duration-type">
  <xs:restriction base="xs:NMTOKEN">
    <xs:enumeration value="second"/>
    <xs:enumeration value="minute"/>
    <xs:enumeration value="hour"/>
    <xs:enumeration value="day"/>
    <xs:enumeration value="month"/>
    <xs:enumeration value="quarter"/>
    <xs:enumeration value="year"/>
    <xs:enumeration value="ext-value"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="action-type">
  <xs:restriction base="xs:NMTOKEN">
    <xs:enumeration value="nothing"/>
    <xs:enumeration value="contact-source-site"/>
    <xs:enumeration value="contact-target-site"/>
    <xs:enumeration value="contact-sender"/>
    <xs:enumeration value="investigate"/>
    <xs:enumeration value="block-host"/>
    <xs:enumeration value="block-network"/>
    <xs:enumeration value="block-port"/>
    <xs:enumeration value="rate-limit-host"/>
    <xs:enumeration value="rate-limit-network"/>
    <xs:enumeration value="rate-limit-port"/>
    <xs:enumeration value="remediate-other"/>
    <xs:enumeration value="status-triage"/>
    <xs:enumeration value="status-new-info"/>
    <xs:enumeration value="other"/>
    <xs:enumeration value="ext-value"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="dtype-type">
  <xs:restriction base="xs:NMTOKEN">
    <xs:enumeration value="boolean"/>
    <xs:enumeration value="byte"/>
    <xs:enumeration value="character"/>
    <xs:enumeration value="date-time"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="ntpstamp"/>
    <xs:enumeration value="portlist"/>
```

```
<xs:enumeration value="real"/>
<xs:enumeration value="string"/>
<xs:enumeration value="file"/>
<xs:enumeration value="path"/>
<xs:enumeration value="frame"/>
<xs:enumeration value="packet"/>
<xs:enumeration value="ipv4-packet"/>
<xs:enumeration value="ipv6-packet"/>
<xs:enumeration value="url"/>
<xs:enumeration value="csv"/>
<xs:enumeration value="winreg"/>
<xs:enumeration value="xml"/>
<xs:enumeration value="ext-value"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="att-type">
  <xs:restriction base="xs:NMTOKEN">
    <!-- CHANGE - added two values per July IETF discussion
      adding command and control server and sink hole
      bringing forward the 'FraudType' from RFC5901
    -->
    <xs:enumeration value="c2-server"/>
    <xs:enumeration value="sink-hole"/>
    <xs:enumeration value="malware-distribution"/>
    <xs:enumeration value="phishing"/>
    <xs:enumeration value="spear-phishing"/>
    <xs:enumeration value="recruiting"/>
    <xs:enumeration value="fraudulent-site"/>
    <xs:enumeration value="dns-spoof"/>
    <xs:enumeration value="other"/>
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="ext-value"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```

## 9. Security Considerations

The IODEF data model itself does not directly introduce security issues. Rather, it simply defines a representation for incident information. As the data encoded by the IODEF might be considered privacy sensitive by the parties exchanging the information or by those described by it, care needs to be taken in ensuring the appropriate disclosure during both document exchange and subsequent processing. The former must be handled by a messaging format, but the latter risk must be addressed by the systems that process, store,

and archive IODEF documents and information derived from them.

The contents of an IODEF document may include a request for action or an IODEF parser may independently have logic to take certain actions based on information that it finds. For this reason, care must be taken by the parser to properly authenticate the recipient of the document and ascribe an appropriate confidence to the data prior to action.

The underlying messaging format and protocol used to exchange instances of the IODEF MUST provide appropriate guarantees of confidentiality, integrity, and authenticity. The use of a standardized security protocol is encouraged. The Real-time Inter-network Defense (RID) protocol [18] and its associated transport binding IODEF/RID over SOAP [19] provide such security.

In order to suggest data processing and handling guidelines of the encoded information, the IODEF allows a document sender to convey a privacy policy using the restriction attribute. The various instances of this attribute allow different data elements of the document to be covered by dissimilar policies. While flexible, it must be stressed that this approach only serves as a guideline from the sender, as the recipient is free to ignore it. The issue of enforcement is not a technical problem.

#### 10. IANA Considerations

This document uses URNs to describe an XML namespace and schema conforming to a registry mechanism described in [15]

Registration for the IODEF namespace:

- o URI: urn:ietf:params:xml:ns:iodef-1.0
- o Registrant Contact: See the first author of the "Author's Address" section of this document.
- o XML: None. Namespace URIs do not represent an XML specification.

Registration for the IODEF XML schema:

- o URI: urn:ietf:params:xml:schema:iodef-1.0
- o Registrant Contact: See the first author of the "Author's Address" section of this document.
- o XML: See the "IODEF Schema" in Section 8 of this document.



## 11. Acknowledgments

The following groups and individuals, listed alphabetically, contributed substantially to this document and should be recognized for their efforts.

- o Patrick Cain, Cooper-Cain Group, Inc.
- o The eCSIRT.net Project
- o The Incident Object Description and Exchange Format Working-Group of the TERENA task-force (TF-CSIRT)
- o Glenn Mansfield Keeni, Cyber Solutions, Inc.
- o Hiroyuki Kido, NARA Institute of Science and Technology
- o Kathleen Moriarty, EMC Corporation
- o Brian Trammell, ETH Zurich
- o Jan Meijer, SURFnet bv
- o Yuri Demchenko, University of Amsterdam

## 12. References

### 12.1. Normative References

- [1] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C Recommendation , October 2000, <<http://www.w3.org/TR/2000/REC-xml-20001006>>.
- [2] World Wide Web Consortium, "XML Schema Part 1: Structures Second Edition", W3C Recommendation , October 2004, <<http://www.w3.org/TR/xmlschema-1/>>.
- [3] World Wide Web Consortium, "XML Schema Part 2: Datatypes Second Edition", W3C Recommendation , October 2004, <<http://www.w3.org/TR/xmlschema-2/>>.
- [4] World Wide Web Consortium, "Namespaces in XML", W3C Recommendation , January 1999, <<http://www.w3.org/TR/REC-xml-names/>>.
- [5] World Wide Web Consortium, "XML Path Language (XPath) 2.0", W3C Candidate Recommendation , June 2006, <<http://www.w3.org/TR/xpath20/>>.

- [6] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.
- [7] Philips, A. and M. Davis, "Tags for Identifying of Languages", RFC 4646, September 2006.
- [8] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 3986, January 2005`.
- [9] Freed, N. and J. Postel, "IANA Charset Registration Procedures", BCP 2978, October 2000.
- [10] Sciberras, A., "Schema for User Applications", RFC 4519, June 2006.
- [11] Resnick, P., "Internet Message Format", RFC 2822, April 2001.
- [12] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.
- [13] International Organization for Standardization, "International Standard: Data elements and interchange formats - Information interchange - Representation of dates and times", ISO 8601, Second Edition, December 2000.
- [14] International Organization for Standardization, "International Standard: Codes for the representation of currencies and funds, ISO 4217:2001", ISO 4217:2001, August 2001.
- [15] Mealling, M., "The IETF XML Registry", RFC 3688, January 2004.

## 12.2. Informative References

- [16] Keeni, G., Demchenko, Y., and R. Danyliw, "Requirements for the Format for Incident Information Exchange (FINE)", Work in Progress, June 2006.
- [17] Debar, H., Curry, D., Debar, H., and B. Feinstein, "Intrusion Detection Message Exchange Format", RFC 4765, March 2007.
- [18] Moriarty, K., "Real-time Inter-network Defense", Work in Progress, April 2007.
- [19] Moriarty, K. and B. Trammell, "IODEF/RID over SOAP", Work in Progress, April 2007.
- [20] Shafranovich, Y., "Common Format and MIME Type for Comma-

Separated Values (CSV) File", RFC 4180, October 2005.

Authors' Addresses

Roman Danyliw  
CERT - Software Engineering Institute  
Pittsburgh, PA  
USA

EMail: rdd@cert.org

Paul Stoecker  
RSA  
Reston, VA  
USA

EMail: paul.stoecker@rsa.com