MILE N. Cam-Winget, Ed.
Internet-Draft S. Appala
Intended status: Standards Track S. Pope
Expires: April 10, 2016 Cisco Systems
October 8, 2015

XMPP Protocol Extensions for Use with IODEF
draft-appala-mile-xmpp-grid-00

Abstract

This document describes the extensions made to Extensible Messaging and Presence Protocol (XMPP) [RFC6120]that enables the use of XMPP as a transport protocol for collecting and distributing any security telemetry information between and among network platforms, endpoints, and most any network connected device. Specifically, this document will focus on how these extensions can be used to transport the Incident Object Description Exchange Format (IODEF) information.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

XMPP-Grid is a set of standards-based XMPP [RFC6120] messages with extensions. It is intended for use as a secure transport and communications protocol ecosystem for devices and organizations to interconnect, forming an information grid for the exchange of formatted data (e.g. XML, JSON, etc). This document describes the extensions made to XMPP [RFC6120]that enables use of XMPP as a transport protocol for securely collecting and distributing security telemetry information between and among network platforms, endpoints, and most any network connected device.

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].

1.1. Glossary of Terms

AAA

Authentication, Authorization and Accounting.

CA

Certification Authority.

Capability Provider

Providers who are capable of sharing information on XMPP-Grid.

CMDB

Configuration Management Database.

IDS

Intrusion Detection System.

IPS

Intrusion Prevention System.

JID

Jabber Identifier, native address of an XMPP entity.

MDM

Mobile Device Management.

NAC

Network Admission Control.

PDP

Policy Decision Point.

PEP

Policy Enforcement Point.

Presence

XMPP-Grid node availability and online status on XMPP-Grid.

Publisher

A capability provider sharing contet information to other devices participating on XMPP-Grid.

SIEM

Security Information and Event Management.

Subscriber

A device participating in XMPP-Grid and subscribing or consuming information published by Publishers on XMPP-Grid.

Sub-Topics

Topic created by XMPP-Grid Controller under a capability provider's topic based on message filter criteria expressed by subscribers.

Topics

Contextual information channel created on XMPP-Grid where a published message by the Publisher will be propagated by XMPP in real-time to a set a subscribed devices.

VoIP

Voice over IP.

XMPP-Grid

Set of standards-based XMPP messages with extensions, intended for use as a transport and communications protocol framework between devices forming an information grid for sharing information.

XMPP-Grid Controller

Centralized component of XMPP-Grid responsible for managing all control plane operations.

XMPP-Grid Connection Agent

XMPP-Grid client library that a XMPP-Grid node implements to connect and exchange information with other vendor devices on XMPP-Grid.

XMPP-Grid Node

Platform or device that implements XMPP-Grid Connection Agent to connect to XMPP-Grid and share or consume security data.

1.2. What is XMPP-Grid?

XMPP-Grid is a set of standards-based XMPP messages with extensions. It is intended for use as a transport and communications protocol framework for devices that interconnect with each other, forming a secure information grid.

XMPP-Grid enables secure, bi-directional multi-vendor exchange of contextual information between IT infrastructure platforms such as security monitoring and detection systems, network policy platforms, asset and configuration management, identity and access management platforms. XMPP-Grid can serve to securely exchange any contextual information. XMPP-Grid is built on top of XMPP [RFC6120], [RFC6121] which is an open IETF standard messaging routing protocol used in commercial platforms such as Google Voice, Jabber IM, Microsoft Messenger, AOL IM and a variety of IoT and XML message routing services. XMPP is also being considered as a means to transport IODEF [RFC5070]. XMPP-Grid is designed for orchestration of data sharing between security platforms on a many-to-many basis for millions of end systems.

XMPP-Grid provides a security data sharing framework that enables multiple vendors to integrate to XMPP-Grid once, then both share and consume data bi-directionally with many IT infrastructure platforms and applications from a single consistent framework akin to a network-wide information bus. This reduces the need to develop to explicit, multiple platform-specific interfaces, thereby increasing the breadth of platforms that can interface and share security data. XMPP-Grid is also configurable thereby enabling partners to share only security data they want to share and consume only information relevant to their platform or use-case and to customize information shared without revising the interfaces. XMPP-Grid is data-agnostic enabling it to operate with virtually any data type such as IODEF [RFC5070].

1.3. Overview of XMPP-Grid

XMPP-Grid employs publish/subscribe/query operations brokered by a controller, which enforces access control in the system. This architecture controls what platforms can connect to the "grid" to share ("publish") and/or consume ("subscribe" or "query") contextual information ("Topics") (described in Section 3.3 and 3.5) such as security data needed to support MILE. The control of publish/subscribe/query operations is architecturally distinct from the actual sharing of the contextual information. Control functions are split into a logical control plane, whereas information exchange is considered a logical data plane. This separation enables scalability and customizability.

XMPP-Grid defines an infrastructure protocol that hides the nuances of the XMPP data plane protocol and makes the information sharing models extensible with simple intuitive interfaces. XMPP-Grid Nodes connect to the Grid using the XMPP-Grid Protocol. The XMPP-Grid Protocol makes use of the XMPP transport protocol and introduces an application layer protocol leveraging XML and XMPP extensions to define the protocol.

The components of XMPP-Grid are:

- o XMPP-Grid Controller (Controller): The Controller manages the control plane of XMPP-Grid operations. As such it authenticates and authorizes platforms connecting to the data exchange grid and controls whether or not they can publish, subscribe or query Topics of security data.

- XMPP-Grid Connection Agent (Connection Agent): The Connection Agent enables the adopting Node to communicate with the Controller and other vendor platforms that have adopted XMPP-Grid. Through this communication privileges of the connecting platform--authorization to connect, publish, subscribe, query--are established. The Connection Agent is typically implemented as a client library.

- XMPP-Grid Node (Node): A Node is a platform that has implemented the Connection Agent so that it can connect to an XMPP-Grid deployment to share and/or consume security data.

- Data Repository: This is the source of security data available on the Grid and may be a network security platform, management console, endpoint, etc. XMPP-Grid does not mandate a specific information model, but instead remains open to transport structured or unstructured data. Data may be supplied by the security platform itself or by an external information repository.

- Topic: An XMPP-Grid Topic defines a type of security data that a platform wants to share with other platform(s).

The operations carried out by XMPP-Grid to exchange security data are:

- Grid Connect: This is a Controller operation that authenticates a Node that has implemented the Connection Agent to establish a connection with the XMPP-Grid. Once authenticated, authorization policies on the Controller establish a Node’s privileges on the XMPP-Grid such as the right to undertake publish, subscribe or query operations explained below.

- Publish Topic: Security information is made available when a XMPP-Grid enabled platform "publishes" a "Topic". This operation is authorized by the Controller and communicated to the connecting platform via the Connection Agent.

- Topic Discovery: Nodes on a XMPP-Grid discover Topics of security data relevant to them by searching the Topic directory available within the XMPP-Grid deployment. The Controller maintains such a Topic directory for every instance of XMPP-Grid.

- Subscribe to Topic: A Node seeking to consume security information "subscribes" to a Topic that provides the security information it seeks to serve its use-case. This operation has its authorization checked by the Controller and communicated with the connecting platform via the Connection Agent.

- Query: This operation enables a Node to request a specific set of security data regarding a specific asset (such as a specific user endpoint) or bulk output history from a Topic over a specific span of time. Such queries can be carried out node-to-node or by querying a central data repository. Query structure is adaptable to match the information model in use.

XMPP-Grid is used to exchange security context data between systems on a 1-to-1, 1-to-many, or many-to-many basis. Security data shared between these systems may use pre-negotiated non-standard/native data formats or may utilize an optional common information repository with a standardized data format, such as IODEF. XMPP-Grid is data format agnostic and accommodates transport of whatever format the end systems agree upon.

XMPP-Grid can operate in the following deployment architectures:

- Broker-Flow: An XMPP-Grid control plane brokers the authorization and redirects the Topic subscriber to Topic publisher directly. In this architecture, the Controller only manages the connection; the security data flow is directly between Nodes using data formats negotiated out-of-band.

- Centralized Data-Flow: An XMPP-Grid maintains the data within its optional centralized database. In this architecture, the Controller provides a common information structure for use in formatting and storing security context data, such as IODEF, and directly responds to Node publish and Subscribe requests.

- Proxy-Flow: An XMPP-Grid is acting as proxy, collecting the data from the publisher(s) and presenting it to the subscriber directly. This is used for ad-hoc queries.

Within the deployment architecture, XMPP-Grid may be used in any combination of the following data exchange modes. The flexibility afforded by the different modes enables security information to be exchanged between systems in the method most suitable for serving a given use-case.

- Continuous Topic update stream: This mode delivers in real-time any data published to a Topic to the Nodes that are subscribed to that Topic.

- Directed query: This mode enables Nodes to request a specific set of security information regarding a specific asset, such as a specific user endpoint.

- Bulk historic data query: This mode enables Nodes to request transfer of past output from a Topic over a specific span of time.

1.4. Benefits of XMPP-Grid

Benefits of XMPP-Grid can be summarized on two fronts: 1) end-user benefits, 2) benefits for adopting vendors.

Benefits for end-users deploying security services based on XMPP-Grid security context information sharing capabilities are derived from the results that come with standardization including:

- Consolidating relevant security event data from multiple systems to the "right console at the right time".

- Cross-vendor interoperability out-of-the-box, when using a standard data format.

- Coordinated security response across multiple products from multiple vendors, ranging from endpoint security to AAA, NAC, IDS/IPS, Data Loss Prevention, firewalls to infrastructure such as SIEM, CMDB, physical access control systems.

- Customer product choice and flexibility. No need to buy all security products from one vendor.

Adopting XMPP-Grid security data sharing capabilities provides a number of benefits for adopting vendors, especially when compared to proprietary interfaces, such as:

- Integrate the XMPP-Grid Connection Agent once to interface with many platforms, simultaneously by subscribing or publishing relevant security data

- Security information shared is configurable (via Topics) based on relevance to specific use-cases and platforms

- Only sharing relevant data enables both publishing and subscribing platforms to scale their security data sharing by eliminating excess, irrelevant data

- Integrated authorization and security ensures only appropriate XMPP-Grid operations are executed by permitted platforms

- Ability to share security data in native or structured formats enables data model flexibility for adopting vendors

   o Flexibility, adaptability to evolve to address new use cases over
      time. Utilize data-agnostic transport protocol or the extensible
      schema that allows for easy support for vendor-specific data.

1.5. Example Workflow

```
                            ______________
                            | XMPP-Grid  |
                 Authorize /| Controller |\Authorize
                          / |____________| \
                    ________/      ^      \__________
"I have location"  |Location|      |        |   APP   |"I have app info"
"I need app & ID.."|________|\     |       /|________|"I need loc & dev"
                              \    |      /
                              _\___|_____/______
                               |Continuous Flow|
                         <-----|---------------|------>
                               | Directed Query|
                               --/------|-----\-
                                /       |      \
                               /        |       \
                    __________/         |        \__________
 "I have sec events"|  SIEM  |          V         |  PAP   |"I have ID & dev-typ
e"
   "I need ID & dev"|________|   ______________   |________|"I need loc & MDM"
                                 | Device Mgr |
                                 |____________|
                               "I have MDM Info!"
                              "I need location..."
```

   Figure 1: Typical XMPP-Grid Workflow

   a. XMPP-Grid Controller establishes a grid for platforms wanting to
       exchange security data.

   b. A platform (Node) with a source of security data requests
       connection to the Grid.

   c. Controller authenticates and establishes authorized privileges
       (e.g. privilege to publish and/or subscribe to security data
       Topics) for the requesting Node.

   d. Node may either publish a security data Topic, subscribe to a
       security data Topic, query a Node or Topic, or any combination of
       these operations.

   e. Publishing Nodes unicast Topic updates to the Grid in real-time.
       The Grid handles replication and distribution of the Topic to

subscribing Nodes. A Node may publish multiple Topics, thereby allowing for customized relevance of the security data shared.

f. Subscribing Nodes receive continuous real-time stream of updates to the Topic to which they are subscribed.

g. Any Node on the Grid may subscribe to any Topics published to the Grid (as permitted by authorization policy), thereby allowing for one-to-one, one-to-many and many-to-many meshed security data sharing between Nodes.

2. XMPP-Grid Architecture

XMPP-Grid is a communication fabric that facilitates secure sharing of information between network elements and networked applications connected to the fabric both in real time and on demand.

XMPP-Grid uses XMPP servers that operate as a cluster with message routing between them, for data plane communication. XMPP-Grid uses a control plane element, the XMPP-Grid Controller, that is an external component of XMPP for centralized policy-based control plane.

```
---------------         ---------------        ---------------
|  XMPP-Grid  |         |  XMPP-Grid  |        |  XMPP-Grid  |
|  Controller |         |  Controller |        |  Controller |
|             |         |             |        |             |
---------------         ---------------        ---------------
      |                         |                      |
      |                         |                      |
---------------         ---------------        ---------------
| XMPP Server |         | XMPP Server |        | XMPP Server |
|             |---------|             |--------|             |
|             |         |             |        |             |
---------------         ---------------        ---------------
```

Figure 2: XMPP Server and XMPP-Grid Cluster Architecture

The connected Nodes, with appropriate authorization privileges, can:

- o Receive real-time events of the published messages from the publisher through Topic subscriptions

- o Make directed queries to other Nodes in the XMPP-Grid with appropriate authorization from the Controller

- o Negotiate out-of-band secure file transfer channel with the peer

This model enables flexible API usage depending on the Nodes' contextual and time-sensitivity needs of security information.

2.1. XMPP Overview

XMPP is used as the foundation message routing protocol for exchanging security data between systems across XMPP-Grid. XMPP is a communications protocol for message-oriented middleware based on XML. Designed to be extensible, the protocol uses de-centralized client-server architecture where the clients connect to the servers securely and the messages between the clients are routed through the XMPP servers deployed within the cluster. XMPP has been used extensively for publish-subscribe systems, file transfer, video, VoIP, Internet of Things, Smart Grid Software Defined Networks (SDN) and other collaboration and social networking applications. The following are the 4 IETF specifications produced by XMPP working group:

- o [RFC6120] Extensible Messaging and Presence Protocol (XMPP): Core

- o [RFC6121] Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence

- o [RFC3922] Mapping the Extensible Messaging and Presence Protocol (XMPP) to Common Presence and Instant Messaging (CPIM)

- o [RFC3923] End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP)

XMPP offers several of the following salient features for building a security data interexchange protocol:

- o Open - standards-based, decentralized and federated architecture, with no single point of failure

- o Security - Supports domain segregations and federation. Offers strong security via Simple Authentication and Security Layer (SASL) [RFC4422] and Transport Layer Security (TLS) [RFC5246].

- o Real-time event management/exchange - using publish, subscribe notifications

- o Flexibility and Extensibility - XMPP is XML based and is easily extensible to adapt to new use-cases. Custom functionality can be built on top of it.

- o Multiple information exchanges - XMPP offers multiple information exchange mechanisms between the participating clients -

o

   * Real-time event notifications through publish and subscribe.

   * On-demand or directed queries between the clients communicated through the XMPP server

   * Facilitates out-of-band, direct communication between participating clients

o Bi-directional - avoids firewall tunneling and avoids opening up a new connection in each direction between client and server.

o Scalable - supports cluster mode deployment with fan-out and message routing

o Peer-to-peer communications also enables scale - directed queries and out-of-band file transfer support

o XMPP offers Node availability, Node service capability discovery, and Node presence within the XMPP network. Nodes ability to detect the availability, presence and capabilities of other participating nodes eases turnkey deployment.

The XMPP extensions used in XMPP-Grid are now part (e.g. publish/ subscribe) of the main XMPP specification [RFC6120] and the presence in [RFC6121]. A full list of XMPP Extension Protocols (XEPs) [RFC6120] can be found in http://xmpp.org/extensions/xep-0001.html .

2.2. XMPP-Grid Protocol Extensions to XMPP

XMPP-Grid defines an infrastructure protocol that hides the nuances of the XMPP data plane protocol and makes the information sharing models extensible with simple intuitive APIs. XMPP-Grid Nodes connect to the Grid using the XMPP-Grid Protocol. The XMPP-Grid Protocol makes use of the XMPP transport protocol and introduces an application layer protocol leveraging XML and XMPP extensions to define the protocol. The capability providers on the Grid extend the XMPP-Grid Protocol infrastructure model and define capability specific models and schemas, allowing a cleaner separation of infrastructure and capabilities that can run on the infrastructure.

2.3. XMPP-Grid Controller Protocol Flow

At the heart of the XMPP-Grid network, the XMPP-Grid Controller serves as the centralized policy-based control plane element managing all Node authentications, authorizations, capabilities/Topics and their subscription list. XMPP-Grid Controller manages all control aspects of the Node communication (including management) with the XMPP-Grid and other participating Nodes with mutual trust and authorizations' enforcement. XMPP-Grid Controller is a component of XMPP server and programs the data plane XMPP server with Node accounts, account status, XMPP Topics that are dynamically created and Topic subscriptions. This is analogous to File Transfer Protocol (FTP) that has control and data plane communication phases. Once the Node requests are authenticated and authorized in the control plane phase by the Controller, the Controller removes itself from the data flow. All data plane communication then occurs between the Nodes, publishers and subscribers of XMPP Topics happen at the XMPP data plane layer.

```
      ----------------      ----------------            ----------------
      | Publi- | Node |     | Grid  | XMPP  |           | Node | Sub-    |
      | sher   |client|     | Ctrlr | Srvr  |           |client| scriber|
      |        |      |     |       |       |           |      |        |
      -----------------     -----------------           -----------------
        | Authen & allow Grid Ctrlr Comm |                           |
        |<------------------------------>|                           |
        |                     |          |                           |
        |                     |Publisher |                           |
        |                     |   Auth   |                           |
        |                     |  Status  |                           |
        |                     |<---------|                           |
        |                     |          |                           |
        |                     |          | Authen & allow Grid Ctrlr |
        |                     |          | Communication             |
        |                     |          |<------------------------->|
        |                     |          |                           |
        |                     |Subscriber|                           |
        |                     |   Auth   |                           |
        |                     | Status & |                           |
        |                     | Account  |                           |
        |                     |<---------|                           |
        |                     |          |                           |
        | Author Publisher to |          |                           |
        |  Topic Sequence     |          |                           |
---     |<------------------->|          |                           |
 C |    |                     |          |                           |
 O |    |                     | Add      |                           |
 N |    |                     |Publisher |                           |
 T |    |                     | to Topic |                           |
 R |    |                     |--------->|                           |
 O |    |                     |          |                           |
 L |    |                     | Author Subscriber to Topic Sequence  |
---     |                     |<------------------------------------>|
```

```
        |                     |          |                           |
        |                     | Add      |                           |
        |                     |Subscriber|                           |
        |                     | to Topic |                           |
        |                     |--------->|                           |
----------------------------------------------------------------------
       |                      |          |                           |
       |    Publish Message to Topic     |                           |
---    |-------------------------------->|                           |
   |   |                      |          |                           |
 I |   |   Publish Success           Published Message to Subscriber |
 N |   |<----------------------------------------------------------->|
 F |   |                      |          |                           |
 R |   |                      |          |     Subscribe Success     |
 A |   |                      |          |<--------------------------|
   |   |                      |          |                           |
---    |                      | Topic & Publisher Discovery Request  |
       |                      |<-------------------------------------|
       |                      |          |                           |
       |                      | Topic & Publisher JID Response       |
       |                      |------------------------------------->|
       |                      |          |                           |
       |                      | Out-of-Band Bulk Dnld Query Reqeust  |
       |                      |<-------------------------------------|
       |                      |          |                           |
       |                      | Out-of-Band Bulk Dnld Query Author   |
       |                      |------------------------------------->|
       |                                 |                           |
       | Out-of-Band Data Bulk Byte Stream                           |
       |<----------------------------------------------------------->|
```

Figure 3: XMPP Controller Message Flow

Through a centralized authorization model, XMPP-Grid Controller provides -

o Visibility into "who is connecting", "who is accessing what"

o Node account management with provisions to add, delete or disable accounts, and with provisions to auto or manual approve Node account approval requests during the Node registration phase

o Centralized, policy-based authorization, providing "who can do what" for publish-subscribe, directed peer-to-peer queries or for bulk out-of-band transfers between participating Nodes

- Topics and subscription list management with provision to enable or disable Topics

- Dynamic creation of sub-Topics within the main Topic depending on attributes of interest from the requesting Node

- Ability to perform message filters on the published messages

2.4. XMPP-Grid Node Connection Protocol Flow

Nodes connecting to XMPP-Grid go through the phases of authentication, registration and authorization before they can participate in information exchange on XMPP-Grid.

2.4.1. Authentication

The communication between the Node and the XMPP-Grid Controller is cryptographically encrypted using TLS. XMPP-Grid uses X.509 certificate-based mutual authentication between the Nodes and Controller. Internally, XMPP uses Simple Authentication and Security Layer (SASL)[RFC4422] External mechanism to authenticate and establish secure tunnel with the Nodes, allowing the XMPP-Grid Controller to rely on this capability offered by XMPP. If the Node certificate does not pass the validation process, the connection establishment is terminated with the error messages defined by the XMPP standard. On successful authentication, XMPP SASL component extracts the Node certificate and Node username to the Controller for registration.

2.4.2. Registration

Once a Node has been authenticated and a secure tunnel has been successfully established, the Nodes will register their accounts with the Controller and Nodes provide their username to the Controller as part of the registration request. XMPP-Grid supports manual registration (requires explicit approval of the Node account) and mutual authentication trust-based auto-approval registration in order to provide additional trust and usability options to the administrator. The administrator may map the Nodes to the Node groups to add additional level of validation and trust, and enforce Node group based authorization. This allows the certificate-username-group trust to get uniquely establishment for each Node and duplicate registration requests using the same username will be rejected.

During the registration process, the Controller restricts all Node communication with the XMPP-Grid and only Node to Controller communication is allowed. Once the Node is successfully registered, the Controller lifts the restriction and allows the Nodes to communicate on XMPP-Grid after it passes the authorization phase. It should be noted that the registered and authorized Nodes could publish, subscribe or query to multiple XMPP Topics between login and logout to XMPP-Grid. Multiple Node applications running on a Node could use one XMPP-Grid Node to connect to XMPP-Grid. The XMPP-Grid Node should support Node applications' subscription to Topics and should multiplex messages on its connection to XMPP-Grid. If a Node application wants to be identified explicitly on XMPP-Grid, a new XMPP-Grid Node connection to XMPP-Grid is required.

```
       -----------------                         ---------------------------
        |               |                         |  Grid     |   XMPP      |
        |    Node       |                         | Controller|  Server     |
        |               |                         |           |             |
        -----------------                         ---------------------------
                |                                       |                |
           _____|                                       |                |
          |     |                                       |                |
Register  |     |                                       |                |
          |---->|                                       |                |
                | TLS Connect(username, cert)           |                |
                |<------------------------------------------------------>|
                |                                       |                |
                |                                       | Track          |
                |                                       |(username,cert) |
                |                                       |<---------------|
                |Register(username)                     |                |
                |-------------------------------------->|                |
                |                                       |___             |
                |                                       |   | Approve &  |
                |                                       |   | Authorize  |
                |                                       |<--| Account    |
                |                                       |                |
                |                                       |Create User     |
                |                                       |Account         |
                |                                       |(username)      |
                |                                       |--------------->|
                |        Registration Successful        |                |
                |<--------------------------------------|                |
                |                                       |                |
                |  Login()                              |                |
                |-------------------------------------->|                |
                |                                       |                |
                |  Pub/Sub/Query                        |                |
                |<------------------------------------------------------>|
                |                                       |                |
                |                                       |                |
                |  Logout()                             |                |
                |-------------------------------------->|                |
                |                                       |                |
```

Figure 4: XMPP-Grid Node Registration

2.4.3. Authorization

The registered Nodes send subscription requests to the Controller. The Controller, depending on the defined authorization privileges, grants permissions to subscribe and/or publish to a Topic at the registration time. The Controller updates the XMPP data plane server with the new subscriber information and its capability. Node identity extracted from the request, group to which the Node is assigned during account approval and Topic/capability to which the permission is sought could be some of the ways to authorize Nodes and their requests in XMPP-Grid. Similarly, the Controller authorizes directed peer-to-peer or out-of-band requests from a requesting peer. The destination peer has options to query back the Controller to retrieve and enforce granular authorizations such as read-only, write-only, read/write.

In a Query Authorization flow, the capability provider responding to the query is responsible for enforcing the authorization decision. It retrieves "is authorized" from the XMPP-Grid Controller. Based on the result, the service either allows or disallows the flow from continuing.

```
      -----------------        -----------------         -----------------
     |  Subscriber     |      |    XMPP-Grid    |       |   Publisher     |
     |                 |      |                 |       |                 |
      -----------------        -----------------         -----------------
              |                        |                         |
              |                        |                         |
              |                  query request                   |
              |------------------------------------------------->|
              |                        |                         |____
              |                        |                         |    | extract
              |                        |                         |    | identity
              |                        |                         |<---
              |                        |                         |
              |                        |     is authorized?      |
              |                        |   (identity, service)   |
              |                        |<------------------------|
              |                        |                         |
              |                  query response                  |
              |<-------------------------------------------------|
              |                        |                         |
```

Figure 5: Node Query Authorization Flow

For Publish Authorization, prior to allowing a publish request by a user, the XMPP-Grid Controller calls the rule evaluation engine directly for "is authorized". Based this result, the Controller either allows or disallowed the flow from continuing.

```
      -----------------        -----------------         -----------------
     |   Publisher     |      |    XMPP-Grid    |       |      XMPP       |
     |                 |      |    Controller   |       |     Server      |
      -----------------        -----------------         -----------------
              |                        |                         |
              |       publish          |                         |
              |----------------------->|____                     |
              |                        |   |  extract            |
              |                        |   |  identity           |
              |                        |<---                     |
              |                        |                         |
              |                        |____                     |
              |                        |   |   is authorized?    |
              |                        |   | (identity,publish)  |
              |                        |<---                     |
              |                        |                         |
              |                        |        publish          |
              |                        |------------------------>|
              |                        |                         |
```

Figure 6: Node Publish Authorization Flow

For Subscribe Authorization, prior to allowing a subscribe request by a user, the XMPP-Grid Controller calls the rule evaluation engine directly for "is authorized". Based this result, the Controller either allows or disallowed the flow from continuing.

```
      -----------------        -----------------          -----------------
     |   Subscriber    |      |    XMPP-Grid    |        |      XMPP       |
     |                 |      |    Controller   |        |     Server      |
      -----------------        -----------------          -----------------
              |                        |                          |
              |      subscribe         |                          |
              |----------------------->|____                      |
              |                        |    |  extract            |
              |                        |    |  identity           |
              |                        |<---                      |
              |                        |                          |
              |                        |____                      |
              |                        |    |   is authorized?    |
              |                        |    | (identity,publish)  |
              |                        |<---                      |
              |                        |                          |
              |                        |       subscribe          |
              |                        |------------------------->|
              |                        |                          |
```

Figure 7: Node Subscribe Authorization Flow

Bulk Data Query differs from other data transfer modes. Unlike with other modes of communication that operate in-band with the XMPP-Grid, bulk downloads occur out-of-band (over a different protocol, outside of the connection that was established with the XMPP-Grid Controller). Previously discussed authorization mechanisms are therefore not appropriate in this context.

```
      -----------------        -----------------          -----------------
     |   Subscriber    |      |    XMPP-Grid    |        |    Publisher    |
     |                 |      |    Controller   |        |                 |
      -----------------        -----------------          -----------------
              |                        |                         |
              |                     request                      |
              |------------------------------------------------->|
              |                        |                         |____ extract
              |                        |                         |    | cert
              |                        |                         |    | chain
              |                        |                         |<---
              |                        |    is authorized        |
              |                        | (cert chain, service)   |
              |                        |<------------------------|
              |                        |                         |
              |                     response                     |
              |<-------------------------------------------------|
              |                        |                         |
```

Figure 8: Node Bulk Data Query Flow

Instead the bulk download service sends the certificate chain used by a Node in the TLS connection to the XMPP-Grid Controller for purposes of authenticating and authorizing the Node. Upon receiving a request with a certificate chain, the Controller checks the issuing certificate against the trust store, looks up the identity associated with the certificate, evaluates the rules, and returns "is authorized" to the service. Then the service can either allow or disallow the flow from continuing.

2.5. XMPP-Grid Topics Protocol Flow

For each capability, XMPP-Grid supports extensibility through XML schemas where the providers (publishers) of the capabilities define the schemas for the data exchanged. The capability provider shall also define the version, the available queries and notifications that it can support. The capability provider publishes the messages to one or more XMPP Topics, that it requests XMPP-Grid to create dynamically, depending on:

a. If the capability provider has mutually exclusive schemas, different Topics will be created where the capability provider will be a publisher to each Topic with a separate schema.

b. For a given Topic, if the subscribers wants to receive filtered attributes or attribute values in capability provider’s published data, XMPP-Grid Controller creates sub Topics to the main Topic based on the message filters expressed. XMPP-Grid Controller enrolls the capability provider as the publisher and the requesting subscribers based on the message filter criteria they express. The capability provider will be the publisher to both the main Topic and the sub-Topics.

c. In the case mentioned in (b) above, it is possible for the capability provider to just publish on the main Topic and have the XMPP-Grid Controller filter the published messages on the Controller-side and deliver attributes and attribute values of interest to the subscribers. Controller-side message filter application and the specify mechanisms such as XPATH that can be used for parsing the messages is beyond the scope of this specification.

2.5.1. Topic Versioning

XMPP-Grid supports versioning to support forward and backward compatible information models. The providers of capability include the version number in the messages they publish and the receiving Nodes can interpret the Topic version and process the attributes accordingly. The expectation is any new version of a capability must be of additive updates only. In other words, existing elements and attributes cannot be changed, only new elements or attributes can be added. This will enable nodes with older capability be able to process newer version. The extra new elements or attributes will be ignored. Instead of using the same Topic for all versions, it is possible in XMPP-Grid to programmatically create separate Topics for each version and allow them to be discovered and subscribed by the Nodes.

In XMPP-Grid, versioning support applies equally to both publish/subscribe, directed and out-of-band queries.

2.5.2. Topic Discovery

The Nodes connected to XMPP-Grid can query the Controller and get the list of all capabilities/Topics running on XMPP-Grid. The XML samples provided in XMPP-Grid Protocol section above provide illustrations of Capability Query and Capability Provider Query.

2.5.3. Subtopics and Message Filters

XMPP-Grid supports semantic message filtering for Topics. The content being published by a provider can be semantically grouped into categories based on domain, location of endpoints for example. The provider of a capability specifies whether it supports semantic filtering or not to the Controller at the subscribe time to the Topic under consideration.

XMPP-Grid subscribers query the Controller and obtain the filtering options available for each capability, and express their message filtering criteria at subscription time. The Controller, for each unique filter criteria specified by the subscribers, creates a new sub Topic under the main capability Topic. All the subscribers with the same filtering criteria will be subscribed to the Subtopic. The set of filter criteria for a capability will be predefined by the capability provider and could be based on the well-defined attributes of the message.

```
  -----------------         ---------------------------        --------------
  |               |         |  Grid     |   XMPP      |        | Capability |
  |    Node       |         | Controller|  Server     |        |   Provider |
  |               |         |           |             |        |            |
  -----------------         ---------------------------        --------------
         |                    |                       |                |
         |Subscribe with filter                       |                |
         |------------------->|____                   |                |
         |                    |    | translate &      |                |
         |                    |    | validate         |                |
         |                    |<---- filter           |                |
         |                    |____                   |                |
         |                    |    | Check if sub-topic                |
         |                    |    | for filter       |                |
         |                    |<---  exists           |                |
         |                    |                       |                |
         |                    |Create subtopic if doesn't exist        |
         |                    |---------------------->|                |
         |                    |                       |                |
         |                    |Add Pub & Sub to Sub-Topic              |
         |                    |---------------------->|                |
         |                    |                       |                |
         |                    |Notify Publisher       |                |
         |                    |--------------------------------------->|
         | Subscribe Success  |                       |                |
         |<-------------------|                       |                |
         |                    |                       |                |
```

Figure 9: Subtopics and Information Filter Subscribe Operations Flow

The publisher will be responsible for applying the filter on the message and publishing the message on the Topic and the Subtopic based on the filter criteria. Filtering logic will be on the publisher, as the publisher understands the message content. XMPP-Grid fabric is oblivious to the message content.

To avoid proliferation of new Subtopics, the capability provider could express the configurable limit on the number of Subtopics that can be created for its capability at registration time. The XMPP-Grid Controller will perform periodic cleanup of Subtopics whenever their subscription list reduces to 0.

In XMPP-Grid, message filters are provided to all APIs i.e. publish/ subscribe and directed query.

```
  -----------------         ---------------------------        --------------
  |  Capability   |         |  Grid      |   XMPP      |        |            |
  |   Provider    |         | Controller|  Server      |        |   Node     |
  |               |         |           |              |        |            |
  -----------------         ---------------------------        --------------
         |                     |                       |                |
         | Register as Publisher                       |                |
         |-------------------->|                       |                |
         |                     |Add Publisher to main |                |
         |                     |topic & all subtopics |                |
         |                     |---------------------->|                |
         |Return registration |                       |                |
         |success & list of   |                       |                |
         |subtopics with      |                       |                |
         |filtering criteria  |                       |                |
         |<--------------------|                       |                |
         |                     |                       |                |
         |Publish message to  |                       |                |
         |main topic          |                       |                |
         |-------------------->|                       |                |
Check    |____                 |Publish message to    |                |
filtering|    |                |main topic            |                |
criteria &    |                |---------------------->|                |
identity |<---                 |                       |                |
         |                     |                       |                |
         |Publish message to  |                       |                |
         |subtopic that matched filter                 |                |
         |-------------------------------------------->|                |
         |                     |                       |   Notify       |
         |                     |                       |--------------->|
```

Figure 10: Subtopic Publish Operations Flow

2.6. XMPP-Grid Protocol Details

The XMPP-Grid Protocol provides an abstraction layer over and above XMPP messages with the intent to provide intuitive interfaces to the Nodes connecting to XMPP-Grid. Nodes connecting to XMPP-Grid use the following interfaces (provided as XML samples) offered by XMPP-Grid protocol to connect and participate in information exchange on XMPP-Grid:

o Register the Node to XMPP-Grid: Node identified as "Node2@domain.com/mac" sends the following Registration request to XMPP-Grid controller.

```
<iq id="ay0tK-4" to="grid_Controller.jabber"
    from="Node2@domain.com/syam-mac" type="get">
<grid xmlns='gi' type='request'>
<AccountQuery xmlns='com.domain.gi.gcl.Controller'>
<register></register></AccountQuery>
</grid>
</iq>
```

o Node login to XMPP-Grid: The following XML sample shows the Login request from Node "Node2@domain.com/mac" to XMPP-Grid controller and Login response returned by the XMPP-Grid controller to the Node.

```
// Request
<iq id="ay0tK-5" to="grid_Controller.jabber"
    from="Node2@domain.com/mac" type="get">
  <grid xmlns='gi' type='request'>
    <AccountQuery xmlns='com.domain.gi.gcl.Controller'>
      <login></login>
    </AccountQuery>
  </grid>
</iq>

// Response
<iq xmlns="jabber:client" to=" Node2@domain.com/mac"
    from="grid_Controller.jabber" type="result" id="ay0tK-5">
  <grid xmlns="gi" type="response">
    <AccountQuery xmlns="com.domain.gi.gcl.Controller">
      <login xmlns="">
        <value xmlns:ns2="gi" xmlns:xsi=" xsi:nil="true" />
      </login>
    </AccountQuery>
  </grid>
</iq>
```

o Node logout from XMPP-Grid: The following XML sample shows the Logout request sent by Node "Node2@domain.com/mac" to XMPP-Grid controller.

```
<iq id="o47m2-8" to="grid_Controller.jabber"
    from="Node2@domain.com/mac" type="get">
<grid xmlns='gi' type='request'>
<AccountQuery xmlns='com.domain.gi.gcl.Controller'>
<logout></logout>
</AccountQuery>
</grid>
</iq>
```

o Capability Discovery Query: The following XML sample shows the Capability Discovery query request from Node "Node2@domain.com/mac" to XMPP-Grid controller. The XMPP-Grid controller returns the list of capabilities supported by XMPP-Grid and their versions as a response to the Node's request.

```
// Request
<iq id="tVKqm-6" to="grid_Controller.jabber"
    from="Node2@domain.com/mac" type="get">
    <grid xmlns="xgrid" type="request">
        <ns2:getCapabilityListRequest xmlns:ns2=" xmlns:ns4="
                xmlns:ns3=" xmlns:ns5=" xmlns:ns6=" xmlns:ns7=" />
    </grid>
</iq>

// Response
<iq from="grid_Controller.jabber" id="tVKqm-6"
    to="Node2@domain.com/mac" type="result" xmlns="jabber:client">
    <grid type="response" xmlns="xgrid">
        <ns2:getCapabilityListResponse xmlns:ns2=" xmlns:ns3="
                xmlns:ns4=" xmlns:ns5=" xmlns:ns6=" xmlns:ns7=">
            <ns2:capability xmlns:xsi=
                        " xsi:type="ns5:TrustSecMetaDataCapability">
                <ns2:id>0</ns2:id>
                <ns2:name>TrustSecMetaDataCapability-1.0</ns2:name>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
            <ns2:capability
                xmlns:xsi=" xsi:type="ns5:EndpointProfileMetaDataCapability">
                <ns2:id>0</ns2:id>
                <ns2:name>
                     EndpointProfileMetaDataCapability-1.0</ns2:name>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
```

```
            <ns2:capability xmlns:xsi=
                            " xsi:type="ns5:IdentityGroupCapability">
                <ns2:id>0</ns2:id>
                <ns2:name>IdentityGroupCapability-1.0</ns2:name>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
            <ns2:capability xmlns:ns9=" xmlns:xsi="
                            xsi:type="ns9:TDAnalysisServiceCapability">
                <ns2:id>0</ns2:id>
                <ns2:name>TDAnalysisServiceCapability-1.0</ns2:name>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
            <ns2:capability xmlns:xsi=" xsi:type="
                            ns7:NetworkCaptureCapability">
                <ns2:id>0</ns2:id>
                <ns2:name>NetworkCaptureCapability-1.0</ns2:name>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
            <ns2:capability xmlns:xsi=
                            " xsi:type="ns6:EndpointProtectionServiceCapability"
>
                <ns2:id>0</ns2:id>
                <ns2:name>
                            EndpointProtectionServiceCapability-1.0</ns2:name>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
            <ns2:capability xmlns:xsi=
                          " xsi:type="ns4:GridControllerAdminServiceCapability">
                <ns2:id>0</ns2:id>
                <ns2:name>
                     GridControllerAdminServiceCapability-1.0</ns2:name>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
            <ns2:capability xmlns:xsi=
                            " xsi:type="ns5:SessionDirectoryCapability">
                <ns2:id>0</ns2:id>
                <ns2:name>SessionDirectoryCapability-1.0</ns2:name>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
        </ns2:getCapabilityListResponse>
    </grid>
</iq>
```

o Specific Capability Provider Query: The following XML sample shows the Capability Provider hostname query request from Node "Node2@domain.com/mac" to XMPP-Grid controller. XMPP-Grid controller returns the hostname of the specific Capability Provider as a response to the Node’s request.

```
// Request
<iq id="996IL-8" to="grid_Controller.jabber"
    from="Node2@domain.com/mac" type="get">
  <grid xmlns='gi' type='request'>
    <DiscoveryQuery xmlns='com.domain.gi.gcl.Controller'>
      <find><param xsi:type="xs:string" xmlns:ns2="gi" xmlns:xs
                   =" xmlns:xsi=">com.domain.ise.session.SessionQuery
      </param></find>
    </DiscoveryQuery>
  </grid>
</iq>

// Response
<iq from='grid_Controller.jabber' id='996IL-8'
    to='Node2@domain.com/mac' type='result'
    xmlns='jabber:client'>
  <grid type='response' xmlns='gi'>
    <DiscoveryQuery xmlns='com.domain.gi.gcl.Controller'>
      <find xmlns=''><value xmlns:ns3='http://jaxb.dev.java.net/array'
                            xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance
'
                            xsi:type='ns3:stringArray'>
      <item>ise@syam-06.domain.com/syam-mac</item></value></find>
    </DiscoveryQuery>
  </grid>
</iq>
```

o Register as a publisher to the Topic: The following XML sample shows the Register as a Publisher request from a Node "Node2@domain.com/mac" to XMPP-Grid controller.


```
<iq id="fD65a-6" to="grid_Controller.jabber"
    from="Node2@domain.com/mac" type="get">
    <grid xmlns="xgrid" type="request">
        <ns2:initPublishRequest xmlns:ns2=" xmlns:ns4=
                 " xmlns:ns3=" xmlns:ns5=" xmlns:ns6=" xmlns:ns7=">
            <ns2:capability xsi:type="ns5:SessionCapability"
                            xmlns:xsi=">
                <ns2:id>0</ns2:id>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
        </ns2:initPublishRequest>
    </grid>
</iq>
```

o Register as a subscriber to the Topic: The following XML sample shows a subscription request made by Node "Node2@domain.com/mac" for "SessionCapability" Topic to XMPP-Grid controller. On success, determined by the Node’s authorization privilege, XMPP-Grid controller returns the Topic name, version and the Publishers’ hostname as a response to the Node’s request.


```
// Subscribe Request
<iq id="lQJIT-6" to="grid_Controller.jabber"
    from="Node2@domain.com/mac" type="get">
    <grid xmlns="xgrid" type="request">
        <ns2:subscribeRequest xmlns:ns2=" xmlns:ns4=" xmlns:ns3
             =" xmlns:ns5=" xmlns:ns6=" xmlns:ns7=">
            <ns2:capability xsi:type="ns5:SessionCapability"
                        xmlns:xsi=">
                <ns2:id>0</ns2:id>
                <ns2:version>1.0</ns2:version>
            </ns2:capability>
        </ns2:subscribeRequest>
    </grid>
</iq>

// Subscribe Response
<iq from="grid_Controller.jabber" id=" lQJIT-6"
    to="Node2@domain.com/mac" type="result" xmlns="jabber:client">
    <grid type="response" xmlns="xgrid">
        <ns2:subscribeResponse xmlns:ns2=
                               " xmlns:ns3=" xmlns:ns4=" xmlns:ns5="
                               xmlns:ns6=" xmlns:ns7=">
            <ns2:topicName>SessionCapability-1.0</ns2:topicName>
            <ns2:xmppDetails>
                <ns2:jid>ise-mnt-XMPP-Grid-004@xgrid.domain.com/gcl
                </ns2:jid>
                <ns2:jid>ise-mnt-XMPP-Grid-005@xgrid.domain.com/gcl
                </ns2:jid>
            </ns2:xmppDetails>
        </ns2:subscribeResponse>
    </grid>
</iq>
```

o Peer-to-Peer Directed Query: The following XML sample shows a peer-to-peer directed query request made by Node "Node2@domain.com/mac" to other XMPP-Grid participating Node "grid_Controller.jabber", seeking identity group information for a specific user "user1". "grid_Controller.jabber" returns the list of identity groups "user1" belongs as a response to the request.

```
// Query Request
<iq id="kR0YY-8" to="grid_Controller.jabber"
      from="Node2@domain.com/mac" type="get">
    <grid xmlns="xgrid" type="request">
        <ns5:getIdentityGroupRequest xmlns:ns2=" xmlns:ns4="
                xmlns:ns3=" xmlns:ns5=" xmlns:ns6=" xmlns:ns7=">
            <ns5:user>
                <ns2:name>user1</ns2:name>
            </ns5:user>
        </ns5:getIdentityGroupRequest>
    </grid>
</iq>

// Query Response
<iq from="grid_Controller.jabber"
    id=" kR0YY-8" to="Node2@domain.com/mac" type="result">
    <grid type="response" xmlns="xgrid">
        <ns5:getIdentityGroupResponse xmlns:ns2=" xmlns:ns3=
            " xmlns:ns4=" xmlns:ns5=" xmlns:ns6=" xmlns:ns7=">
            <ns5:user>
                <ns2:name>user1</ns2:name>
                <ns3:groupList>
                    <ns3:object>
                        <ns2:name>User Identity Groups:Employee
                        </ns2:name>
                        <ns3:type>Identity</ns3:type>
                    </ns3:object>
                </ns3:groupList>
            </ns5:user>
        </ns5:getIdentityGroupResponse>
    </grid>
</iq>
```

3. XMPP-Grid Compatibility with IODEF

The Incident Object Description and Exchange Format (IODEF) [RFC5070] defines a common data format and common exchange procedures for sharing incidents and related data between CSIRTs. RFC5070 provides the information and data model for IODEF specified with XML schema.

XEP-0268 (http://xmpp.org/extensions/xep-0268.html), Incident Handling, defines ways for XMPP server deployments to share incident reports with each other using the IODEF format and handle attacks on the servers in real-time.

Providers of incident reports, across administrative domains, could participate as publishers to an XMPP topic (for example: IODEF). Trust is achieved through authentication, authorization and account approval as defined in Section 2.4. The providers could expose IODEF incident attributes such as Authority as message filter criteria for the topic in order for subscribing systems to subscribe to incident reports from administrative domains of interest. The providers could further expose other IODEF attributes such as Assessment, Method, Attacker etc as message filter criteria for subscribers to selectively choose events of interest that are published from administrative domain(s). Privacy and regulatory requirements of information shared across administrative domains is beyond the scope of this document.

4. IANA Considerations

IANA Considerations to be determined

5. Security Considerations

A XMPP-Grid Controller serves as an controlling broker for XMPP-Grid Nodes such as Enforcement Points, Policy Servers, CMDBs, and Sensors, using a publish-subscribe-search model of information exchange and lookup. By increasing the ability of XMPP-Grid Nodes to learn about and respond to security-relevant events and data, XMPP-Grid can improve the timeliness and utility of the security system. However, this integrated security system can also be exploited by attackers if they can compromise it. Therefore, strong security protections for XMPP-Grid are essential.

This section provides a security analysis of the XMPP-Grid transport protocol and the architectural elements that employ it, specifically with respect to their use of this protocol. Three subsections define the trust model (which elements are trusted to do what), the threat model (attacks that may be mounted on the system), and the countermeasures (ways to address or mitigate the threats previously identified).

5.1. Trust Model

The first step in analyzing the security of the XMPP-Grid transport protocol is to describe the trust model, listing what each architectural element is trusted to do. The items listed here are assumptions, but provisions are made in the Threat Model and Countermeasures sections for elements that fail to perform as they were trusted to do.

5.1.1. Network

The network used to carry XMPP-Grid messages is trusted to:

- o Perform best effort delivery of network traffic

The network used to carry XMPP-Grid messages is not expected (trusted) to:

- o Provide confidentiality or integrity protection for messages sent over it

- o Provide timely or reliable service

5.1.2. XMPP-Grid Nodes

Authorized XMPP-Grid Nodes are trusted to:

- o Preserve the confidentiality of sensitive data retrieved via the XMPP-Grid Controller

5.1.3. XMPP-Grid Controller

The XMPP-Grid Controller is trusted to:

- o Broker requests for data and enforce authorization of access to this data throughout its lifecycle

- o Perform service requests in a timely and accurate manner

- o Create and maintain accurate operational attributes

- o Only reveal data to and accept service requests from authorized parties

The XMPP-Grid Controller is not expected (trusted) to:

- o Verify the truth (correctness) of data

5.1.4. Certification Authority

The Certification Authority (CA) that issues certificates for the XMPP-Grid Controller and/or XMPP-Grid Nodes (or each CA, if there are several) is trusted to:

- o Protect the confidentiality of the CA’s private key
- o Ensure that only proper certificates are issued and that all certificates are issued in accordance with the CA’s policies
- o Revoke certificates previously issued when necessary
- o Regularly and securely distribute certificate revocation information
- o Promptly detect and report any violations of this trust so that they can be handled

The CA is not expected (trusted) to:

- o Issue certificates that go beyond name constraints or other constraints imposed by a relying party or a cross-certificate

5.2. Threat Model

To secure the XMPP-Grid transport protocol and the architectural elements that implement it, this section identifies the attacks that can be mounted against the protocol and elements.

5.2.1. Network Attacks

A variety of attacks can be mounted using the network. For the purposes of this subsection the phrase "network traffic" should be taken to mean messages and/or parts of messages. Any of these attacks may be mounted by network elements, by parties who control network elements, and (in many cases) by parties who control network-attached devices.

- o Network traffic may be passively monitored to glean information from any unencrypted traffic
- o Even if all traffic is encrypted, valuable information can be gained by traffic analysis (volume, timing, source and destination addresses, etc.)
- o Network traffic may be modified in transit
- o Previously transmitted network traffic may be replayed
- o New network traffic may be added
- o Network traffic may be blocked, perhaps selectively

- o A "Man In The Middle" (MITM) attack may be mounted where an attacker interposes itself between two communicating parties and poses as the other end to either party or impersonates the other end to either or both parties

- o Resist attacks (including denial of service and other attacks from XMPP-Grid Nodes)

- o Undesired network traffic may be sent in an effort to overload an architectural component, thus mounting a denial of service attack

5.2.2. XMPP-Grid Nodes

An unauthorized XMPP-Grid Nodes (one which is not recognized by the XMPP-Grid Controller or is recognized but not authorized to perform any actions) cannot mount any attacks other than those listed in the Network Attacks section above.

An authorized XMPP-Grid Node, on the other hand, can mount many attacks. These attacks might occur because the XMPP-Grid Node is controlled by a malicious, careless, or incompetent party (whether because its owner is malicious, careless, or incompetent or because the XMPP-Grid Node has been compromised and is now controlled by a party other than its owner). They might also occur because the XMPP-Grid Node is running malicious software; because the XMPP-Grid Node is running buggy software (which may fail in a state that floods the network with traffic); or because the XMPP-Grid Node has been configured improperly. From a security standpoint, it generally makes no difference why an attack is initiated. The same countermeasures can be employed in any case.

Here is a list of attacks that may be mounted by an authorized XMPP-Grid Node:

- o Cause many false alarms or otherwise overload the XMPP-Grid Controller or other elements in the network security system (including human administrators) leading to a denial of service or disabling parts of the network security system

- o Omit important actions (such as posting incriminating data), resulting in incorrect access

- o Use confidential information obtained from the XMPP-Grid Controller to enable further attacks (such as using endpoint health check results to exploit vulnerable endpoints)

- o Advertise data crafted to exploit vulnerabilities in the XMPP-Grid Controller or in other XMPP-Grid Nodes, with a goal of compromising those systems

- o Issue a search request or set up a subscription that matches an enormous result, leading to resource exhaustion on the XMPP-Grid Controller, the publishing XMPP-Grid Node, and/or the network

- o Establish a communication channel using another XMPP-Grid Node's session-id

Dependencies of or vulnerabilities of authorized XMPP-Grid Nodes may be exploited to effect these attacks. Another way to effect these attacks is to gain the ability to impersonate a XMPP-Grid Node (through theft of the XMPP-Grid Node's identity credentials or through other means). Even a clock skew between the XMPP-Grid Node and XMPP-Grid Controller can cause problems if the XMPP-Grid Node assumes that old XMPP-Grid Node data should be ignored.

5.2.3. XMPP-Grid Controllers

An unauthorized XMPP-Grid Controller (one which is not trusted by XMPP-Grid Nodes) cannot mount any attacks other than those listed in the Network Attacks section above.

An authorized XMPP-Grid Controller can mount many attacks. Similar to the XMPP-Grid Node case described above, these attacks might occur because the XMPP-Grid Controller is controlled by a malicious, careless, or incompetent party (either a XMPP-Grid Controller administrator or an attacker who has seized control of the XMPP-Grid Controller). They might also occur because the XMPP-Grid Controller is running malicious software, because the XMPP-Grid Controller is running buggy software (which may fail in a state that corrupts data or floods the network with traffic), or because the XMPP-Grid Controller has been configured improperly.

All of the attacks listed for XMPP-Grid Node above can be mounted by the XMPP-Grid Controller. Detection of these attacks will be more difficult since the XMPP-Grid Controller can create false operational attributes and/or logs that imply some other party created any bad data.

Additional XMPP-Grid Controller attacks may include:

- o Expose different data to different XMPP-Grid Nodes to mislead investigators or cause inconsistent behavior

- Mount an even more effective denial of service attack than a single XMPP-Grid Node could

- Obtain and cache XMPP-Grid Node credentials so they can be used to impersonate XMPP-Grid Nodes even after a breach of the XMPP-Grid Controller is repaired

- Obtain and cache XMPP-Grid Controller administrator credentials so they can be used to regain control of the XMPP-Grid Controller after the breach of the XMPP-Grid Controller is repaired

Dependencies of or vulnerabilities of the XMPP-Grid Controller may be exploited to obtain control of the XMPP-Grid Controller and effect these attacks.

5.2.4. Certification Authority

A Certification Authority trusted to issue certificates for the XMPP-Grid Controller and/or XMPP-Grid Nodes can mount several attacks:

- Issue certificates for unauthorized parties, enabling them to impersonate authorized parties such as the XMPP-Grid Controller or a XMPP-Grid Node. This can lead to all the threats that can be mounted by the certificate’s subject.

- Issue certificates without following all of the CA’s policies. Because this can result in issuing certificates that may be used to impersonate authorized parties, this can lead to all the threats that can be mounted by the certificate’s subject.

- Fail to revoke previously issued certificates that need to be revoked. This can lead to undetected impersonation of the certificate’s subject or failure to revoke authorization of the subject, and therefore can lead to all of the threats that can be mounted by that subject.

- Fail to regularly and securely distribute certificate revocation information. This may cause a relying party to accept a revoked certificate, leading to undetected impersonation of the certificate’s subject or failure to revoke authorization of the subject, and therefore can lead to all of the threats that can be mounted by that subject. It can also cause a relying party to refuse to proceed with a transaction because timely revocation information is not available, even though the transaction should be permitted to proceed.

- o Allow the CA’s private key to be revealed to an unauthorized party. This can lead to all the threats above. Even worse, the actions taken with the private key will not be known to the CA.

- o Fail to promptly detect and report errors and violations of trust so that relying parties can be promptly notified. This can cause the threats listed earlier in this section to persist longer than necessary, leading to many knock-on effects.

5.3. Countermeasures

Below are countermeasures for specific attack scenarios to the XMPP-Grid infrastructure.

5.3.1. Securing the XMPP-Grid Transport Protocol

To address network attacks, the XMPP-Grid transport protocol described in this document requires that the XMPP-Grid messages MUST be carried over TLS (minimally TLS 1.2 [RFC5246]) as described in [RFC2818]. The XMPP-Grid Node MUST verify the XMPP-Grid Controller’s certificate and determine whether the XMPP-Grid Controller is trusted by this XMPP-Grid Node before completing the TLS handshake. The XMPP-Grid Controller MUST authenticate the XMPP-Grid Node either using mutual certificate-based authentication in the TLS handshake or using Basic Authentication as described in IETF RFC 2617. XMPP-Grid Controller MUST use Simple Authentication and Security Layer (SASL), described in [RFC4422], to support the aforesaid authentication mechanisms. SASL offers authentication mechanism negotiations between the XMPP-Grid Controller and XMPP-Grid node during the connection establishment phase. XMPP-Grid Nodes and XMPP-Grid Controllers using mutual certificate-based authentication SHOULD each verify the revocation status of the other party. All XMPP-Grid Controllers and XMPP-Grid Nodes MUST implement both mutual certificate-based authentication and Basic Authentication. The selection of which XMPP-Grid Node authentication technique to use in any particular deployment is left to the administrator.

An XMPP-Grid Controller MAY also support a local, configurable set of Basic Authentication userid-password pairs. If so, it is implementation dependent whether a XMPP-Grid Controller ends a session when an administrator changes the configured password. Since Basic Authentication has many security disadvantages (especially the transmission of reusable XMPP-Grid Node passwords to the XMPP-Grid Controller), it SHOULD only be used when absolutely necessary. Per the HTTP specification, when basic authentication is in use, a XMPP-Grid Controller MAY respond to any request that lacks credentials with an error code similar to HTTP code 401. A XMPP-Grid Node SHOULD avoid this code by submitting basic auth credentials with every request when basic authentication is in use. If it does not do so, a XMPP-Grid Node MUST respond to this code by resubmitting the same request with credentials (unless the XMPP-Grid Node is shutting down).

As XMPP uses TLS as the transport and security mechanisms, it is understood that best practices such as those in [I-D.ietf-uta-tls-bcp] are followed.

These protocol security measures provide protection against all the network attacks listed in the above document section except denial of service attacks. If protection against these denial of service attacks is desired, ingress filtering, rate limiting per source IP address, and other denial of service mitigation measures may be employed. In addition, a XMPP-Grid Controller MAY automatically disable a misbehaving XMPP-Grid Node.

5.3.2. Securing XMPP-Grid Nodes

XMPP-Grid Nodes may be deployed in locations that are susceptible to physical attacks. Physical security measures may be taken to avoid compromise of XMPP-Grid Nodes, but these may not always be practical or completely effective. An alternative measure is to configure the XMPP-Grid Controller to provide read-only access for such systems. The XMPP-Grid Controller SHOULD also include a full authorization model so that individual XMPP-Grid Nodes may be configured to have only the privileges that they need. The XMPP-Grid Controller MAY provide functional templates so that the administrator can configure a specific XMPP-Grid Node as a DHCP server and authorize only the operations and metadata types needed by a DHCP server to be permitted for that XMPP-Grid Node. These techniques can reduce the negative impacts of a compromised XMPP-Grid Node without diminishing the utility of the overall system.

To handle attacks within the bounds of this authorization model, the XMPP-Grid Controller MAY also include rate limits and alerts for unusual XMPP-Grid Node behavior. XMPP-Grid Controllers SHOULD make it easy to revoke a XMPP-Grid Node’s authorization when necessary. Another way to detect attacks from XMPP-Grid Nodes is to create fake entries in the available data (honeytokens) which normal XMPP-Grid Nodes will not attempt to access. The XMPP-Grid Controller SHOULD include auditable logs of XMPP-Grid Node activities.

To avoid compromise of XMPP-Grid Node, XMPP-Grid Node SHOULD be hardened against attack and minimized to reduce their attack surface. They SHOULD go through a TNC handshake to verify the integrity of the XMPP-Grid Node, and SHOULD, if feasible, utilize a Trusted Platform Module (TPM) for identity and/or integrity measurements of the XMPP-

Grid Node within a TNC handshake. They should be well managed to minimize vulnerabilities in the underlying platform and in systems upon which the XMPP-Grid Node depends. Personnel with administrative access should be carefully screened and monitored to detect problems as soon as possible.

5.3.3. Securing XMPP-Grid Controllers

Because of the serious consequences of XMPP-Grid Controller compromise, XMPP-Grid Controllers SHOULD be especially well hardened against attack and minimized to reduce their attack surface. They SHOULD go through a regular TNC handshake to verify the integrity of the XMPP-Grid Controller, and SHOULD utilize a Trusted Platform Module (TPM) for identity and/or integrity measurements of the XMPP-Grid Node within a TNC handshake. They should be well managed to minimize vulnerabilities in the underlying platform and in systems upon which the XMPP-Grid Controller depends. Network security measures such as firewalls or intrusion detection systems may be used to monitor and limit traffic to and from the XMPP-Grid Controller. Personnel with administrative access should be carefully screened and monitored to detect problems as soon as possible. Administrators should not use password-based authentication but should instead use non-reusable credentials and multi-factor authentication (where available). Physical security measures SHOULD be employed to prevent physical attacks on XMPP-Grid Controllers.

To ease detection of XMPP-Grid Controller compromise should it occur, XMPP-Grid Controller behavior should be monitored to detect unusual behavior (such as a reboot, a large increase in traffic, or different views of an information repository for similar XMPP-Grid Nodes). XMPP-Grid Nodes should log and/or notify administrators when peculiar XMPP-Grid Controller behavior is detected. To aid forensic investigation, permanent read-only audit logs of security-relevant information (especially administrative actions) should be maintained. If XMPP-Grid Controller compromise is detected, a careful analysis should be performed of the impact of this compromise. Any reusable credentials that may have been compromised should be reissued.

5.3.4. Limit on search result size

While XMPP-Grid is designed for high scalability to 100,000s of Nodes, an XMPP-Grid Controller MAY establish a limit to the amount of data it is willing to return in search or subscription results. This mitigates the threat of a XMPP-Grid Node causing resource exhaustion by issuing a search or subscription that leads to an enormous result.

5.3.5. Cryptographically random session-id and authentication checks for ARC

A XMPP-Grid Controller SHOULD ensure that the XMPP-Grid Node establishing an ARC is the same XMPP-Grid Node as the XMPP-Grid Node that established the corresponding SSRC. The XMPP-Grid Controller SHOULD employ both of the following strategies:

- session-ids SHOULD be cryptographically random

- The HTTPS transport for the SSRC and the ARC SHOULD be authenticated using the same credentials. SSL session resumption MAY be used to establish the ARC based on the SSRC SSL session.

5.3.6. Securing the Certification Authority

As noted above, compromise of a Certification Authority (CA) trusted to issue certificates for the XMPP-Grid Controller and/or XMPP-Grid Nodes is a major security breach. Many guidelines for proper CA security have been developed: the CA/Browser Forum's Baseline Requirements, the AICPA/CICA Trust Service Principles, etc. The CA operator and relying parties should agree on an appropriately rigorous security practices to be used.

Even with the most rigorous security practices, a CA may be compromised. If this compromise is detected quickly, relying parties can remove the CA from their list of trusted CAs, and other CAs can revoke any certificates issued to the CA. However, CA compromise may go undetected for some time, and there's always the possibility that a CA is being operated improperly or in a manner that is not in the interests of the relying parties. For this reason, relying parties may wish to "pin" a small number of particularly critical certificates (such as the certificate for the XMPP-Grid Controller). Once a certificate has been pinned, the relying party will not accept another certificate in its place unless the Administrator explicitly commands it to do so. This does not mean that the relying party will not check the revocation status of pinned certificates. However, the Administrator may still be consulted if a pinned certificate is revoked, since the CA and revocation process are not completely trusted.

5.4. Summary

XMPP-Grid's considerable value as a broker for security-sensitive data exchange distribution also makes the protocol and the network security elements that implement it a target for attack. Therefore, strong security has been included as a basic design principle within the XMPP-Grid design process.

The XMPP-Grid transport protocol provides strong protection against a variety of different attacks. In the event that a XMPP-Grid Node or XMPP-Grid Controller is compromised, the effects of this compromise have been reduced and limited with the recommended role-based authorization model and other provisions, and best practices for managing and protecting XMPP-Grid systems have been described. Taken together, these measures should provide protection commensurate with the threat to XMPP-Grid systems, thus ensuring that they fulfill their promise as a network security clearing-house.

6. Privacy Considerations

XMPP-Grid Nodes may publish information about endpoint health, network access, events (which may include information about what services an endpoint is accessing), roles and capabilities, and the identity of the end user operating the endpoint. Any of this published information may be queried by other XMPP-Grid Nodes and could potentially be used to correlate network activity to a particular end user.

Dynamic and static information brokered by a XMPP-Grid Controller, ostensibly for purposes of correlation by XMPP-Grid Nodes for intrusion detection, could be misused by a broader set of XMPP-Grid Nodes which hitherto have been performing specific roles with strict well-defined separation of duties.

Care should be taken by deployers of XMPP-Grid to ensure that the information published by XMPP-Grid Nodes does not violate agreements with end users or local and regional laws and regulations. This can be accomplished either by configuring XMPP-Grid Nodes to not publish certain information or by restricting access to sensitive data to trusted XMPP-Grid Nodes. That is, the easiest means to ensure privacy or protect sensitive data, is to omit or not share it at all.

Another consideration for deployers is to enable end-to-end encryption to ensure the data is protected from the data layer to data layer and thus protect it from the transport layer.

7. Acknowledgements

The authors would like to acknowledge the contributions, authoring and/or editing of the following people: Joseph Salowey, Lisa Lorenzin, Clifford Kahn, Henk Birkholz, Jessica Fitzgerald-McKay, Steve Hanna, and Steve Venema.

8. References

8.1. Normative References

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

[RFC3922] Saint-Andre, P., "Mapping the Extensible Messaging and Presence Protocol (XMPP) to Common Presence and Instant Messaging (CPIM)", RFC 3922, DOI 10.17487/RFC3922, October 2004, <http://www.rfc-editor.org/info/rfc3922>.

[RFC3923] Saint-Andre, P., "End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP)", RFC 3923, DOI 10.17487/RFC3923, October 2004, <http://www.rfc-editor.org/info/rfc3923>.

[RFC4422] Melnikov, A., Ed. and K. Zeilenga, Ed., "Simple Authentication and Security Layer (SASL)", RFC 4422, DOI 10.17487/RFC4422, June 2006, <http://www.rfc-editor.org/info/rfc4422>.

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, DOI 10.17487/RFC6120, March 2011, <http://www.rfc-editor.org/info/rfc6120>.

[RFC6121] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", RFC 6121, DOI 10.17487/RFC6121, March 2011, <http://www.rfc-editor.org/info/rfc6121>.

8.2. Informative References

[I-D.ietf-uta-tls-bcp]
Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of TLS and DTLS", draft-ietf-uta-tls-bcp-11 (work in progress), February 2015.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, DOI 10.17487/RFC5070, December 2007, <http://www.rfc-editor.org/info/rfc5070>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

Authors' Addresses

Nancy Cam-Winget (editor)
Cisco Systems
3550 Cisco Way
San Jose, CA 95134
USA

Email: ncamwing@cisco.com

Syam Appala
Cisco Systems
3550 Cisco Way
San Jose, CA 95134
USA

Email: syam1@cisco.com

Scott Pope
Cisco Systems
5400 Meadows Road
Suite 300
Lake Oswego, OR 97035
USA

Email: scottp@cisco.com

MILE Implementation Report
draft-ietf-mile-implementreport-06

Abstract

This document is a collection of implementation reports from vendors, consortiums, and researchers who have implemented one or more of the standards published from the IETF INCident Handling (INCH) and Management Incident Lightweight Exchange (MILE) working groups.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document is a collection of implementation reports from vendors and researchers who have implemented one or more of the standards published from the INCH and MILE working groups. The standards include:

- o Incident Object Description Exchange Format (IODEF) v1, RFC5070,
- o Incident Object Description Exchange Format (IODEF) v2, RFC5070-bis,

- o Extensions to the IODEF-Document Class for Reporting Phishing, RFC5901

- o Sharing Transaction Fraud Data, RFC5941

- o IODEF-extension for Structured Cybersecurity Information, RFCXXXX

- o Real-time Inter-network Defense (RID), RFC6545

- o Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS, RFC6546.

- o Incident Object Description Exchange Format (IODEF) Extension for Structured Cybersecurity Information, RFC7203

The implementation reports included in this document have been provided by the team or product responsible for the implementations of the mentioned RFCs. Additional submissions are welcome and should be sent to the draft editor. A more complete list of implementations, including open source efforts and vendor products, can also be found at the following location:

http://siis.realmv6.org/implementations/

2. Consortiums and Information Sharing and Analysis Centers (ISACs)

2.1. Anti-Phishing Working Group

Anti-Phishing Working Group (APWG) is one of the biggest coalition against cybercrime, especially phishing. In order to collect threat information in a structured format, APWG provides a phishing and cybercrime reporting tool which sends threat information to APWG by tailoring information with IODEF format, based on RFC5070 and RFC5901.

2.2. Advanced Cyber Defence Centre

The Advanced Cyber Defense Centre (ACDC), is EU-wide activity to fight against botnets. ACDC provides a solutions to mitigate on-going attacks, as well as consolidating information provided by various stakeholders into a pool of knowledge. Within ACDC, IODEF is one of the supported schema for exchanging the information.

2.3. Research and Education Networking Information Sharing and Analysis Center

Research and Education Networking Information Sharing and Analysis Center (REN-ISAC) is a private community of the research and higher education members fro sharing threat information, and employs IODEF formatted-message to exchange information.

REN-ISAC also recommends to use of the IODEF attachment provided with the notification email be processed rather than relying on parsing of the email body text. The interface provided by REN-ISAC are designed for dealing with such email.

http://www.ren-isac.net/notifications/using_iodef.html

3. Open Source Implementations

3.1. EMC/RSA RID Agent

The EMC/RSA RID agent is an open source implementation of the Internet Engineering Task Force (IETF) standards for the exchange of incident and indicator data. The code has been released under an MIT license and development will continue with the open source community at the Github site for RSA Intelligence Sharing:

https://github.com/RSAIntelShare/RID-Server.git

The code implements the RFC6545, Real-time Inter-network Defense (RID) and RFC6546, Transport of RID over HTTP/TLS protocol. The code supports the evolving RFC5070-bis Incident Object Description Exchange Format (IODEF) data model from the work in the IETF working group Managed Incident Lightweight Exchange (MILE).

3.2. NICT IODEF-SCI implementation

Japan’s National Institute of Information and Communications Technology (NICT) Network Security Research Institute implemented open source tools for exchanging, accumulating, and locating IODEF-SCI documents.

Three tools are available in GitHub. They assist the exchange of IODEF-SCI documents between parties. IODEF-SCI is the IETF draft that extends IODEF so that IODEF document can embed structured cybersecurity information (SCI). For instance, it can embed MMDEF, CEE, MAEC in XML and CVE identifiers.

The three tools are generator, exchanger, and parser. The generator generates IODEF-SCI document or appends an XML to existing IODEF document. The exchanger sends the IODEF document to its correspondent node. The parser receives, parses, and stores the IODEF-SCI document. It also equips the interface that enable users to locate IODEF-SCI documents it has ever received. The code has been released under an MIT license and development will continue here.

Note that users can enjoy this software with their own responsibility.

Available Online:

https://github.com/TakeshiTakahashi/IODEF-SCI

3.3. n6

n6 is a platform for processing security-related information, developed by NASK, CERT Polska. Its API provides a common and unified way of representing data across the different sources that participate in knowledge management.

n6 exposes a REST-ful API over HTTPS with mandatory authentication via TLS client certificates, to ensure confidential and trustworthy communications. Moreover, it uses an event-based data model for representation of all types of security information.

Each event is represented as a JSON object with a set of mandatory and optional attributes. It also supports alternative output data formats for keeping compatibility with existing systems - IODEF and CSV - although they lack some of the attributes that may be present in the native JSON format.

Available Online:

https://github.com/CERT-Polska/n6sdk

4. Vendor Implementations

4.1. Deep Secure

Deep-Secure Guards are built to protect a trusted domain from:

- o releasing sensitive data that does not meet the organisational security policy

- o applications receiving badly constructed or malicious data which could exploit a vulnerability (known or unknown)

Deep-Secure Guards support HTTPS and XMPP (optimised server to server protocol) transports. The Deep-Secure Guards support transfer of XML based business content by creating a schema to translate the known good content to and from the intermediate format. This means that the Deep-Secure Guards can be used to protect:

- IODEF/RID using the HTTPS transport binding (RFC 6546)
- IODEF/RID using an XMPP binding
- ROLIE using HTTPS transport binding (draft-field-mile-rolie-02)
- STIX/TAXII using the HTTPS transport binding

Deep-Secure Guards also support the SMTP transport and perform deep content inspection of content including XML attachments. The Mail Guard supports S/MIME and Deep Secure are working on support for the upcoming PLASMA standard which enables information centric policy enforcement of data.

4.2. IncMan Suite, DFLabs

The Incident Object Description Exchange Format, documented in the RFC 5070, defines a data representation that provides a framework for sharing information commonly exchanged by Computer Security Incident Response Teams (CSIRTs) about computer security incidents. IncMan Suite implements the IODEF standard for exchanging details about incidents, either for exporting and importing activities. This has been introduced to enhance the capabilities of the various CSIRT, to facilitate collaboration and sharing of useful experiences, conveying awareness on specific cases.

The IODEF implementation is specified as an XML schema, therefore all data are stored in an xml file: in this file all data of an incident are organized in a hierarchical structure to describe the various objects and their relationships.

IncMan Suite relies on IODEF as a transport format, composed by various classes for describing the entities which are part of the incident description: for instance the various relevant timestamps (detect time , start time, end time, report time), the techniques used by the intruders to perpetrate the incident, the impact of the incident, either technical and non-technical (time and monetary) and obviously all systems involved in the incident.

4.2.1. Exporting Incidents

Each incident defined in IncMan Suite can be exported via a User Interface feature and it will populate an xml document. Due to the nature of the data processed, the IODEF extraction might be considered privacy sensitive by the parties exchanging the information or by those described by it. For this reason, specific care needs to be taken in ensuring the distribution to an appropriate audience or third party, either during the document exchange and subsequent processing.

The xml document generated will include description and details of the incident along with all the systems involved and the related information. At this stage it can be distributed for import into a remote system.

4.2.2. Importing Incidents

IncMan Suite provides a functionality to import incidents stored in files and transported via IODEF-compliant xml documents. The importing process comprises of two steps: firstly, the file is inspected to validate if well formed, then all data are uploaded inside the system.

If an incident is already existing in the system with the same incident id, the new one being imported will be created under a new id. This approach prevents from accidentally overwriting existing info or merging inconsistent data.

IncMan Suite includes also a feature to upload incidents from emails.

The incident, described in xml format, can be stored directly into the body of the email message or transported as an attachment of the email. At regular intervals, customizable by the user, IncMan Suite monitors for incoming emails, filtered by a configurable white-list and black-list mechanism on the sender’s email account, then a parser processes the received email and a new incident is created automatically, after having validated the email body or the attachment to ensure it is a well formed format.

4.3. Surevine Proof of Concept

XMPP is enhanced and extended through the XMPP Extension Protocols (or XEPs). XEP-0268 (http://xmpp.org/extensions/xep-0268.html) describes incident management (using IODEF) of the XMPP network itself, effectively supporting self-healing the XMPP network. In order to more generically cover incident management of a network and over a network, XEP-0268 requires some updates. We are working on these changes together with a new XEP that supports "social networking" over XMPP, enhancing the publish-and-subscribe XEP (XEP-0060). This now allows nodes to publish any type of content and subscribe to and therefore receive the content. XEP-0268 will be used to describe IODEF content. We now have an alpha version of the server-side software and client-side software required to demonstrate the "social networking" capability and are currently enhancing this to support Cyber Incident management in real-time.

4.4. MANTIS Cyber-Intelligence Management Framework

MANTIS provides an example implementation of a framework for managing cyber threat intelligence expressed in standards such as STIX, CybOX, IODEF, etc. The aims of providing such an example implementation are:

- To aide discussions about emerging standards such as STIX, CybOX et al. with respect to questions regarding tooling: how would a certain aspect be implemented, how do changes affect an implementation? Such discussions become much easier and have a better basis if they can be lead in the context of example tooling that is known to the community.

- To lower the entrance barrier for organizations and teams (esp. CERT teams) in using emerging standards for cyber-threat intelligence management and exchange.

- To provide a platform on the basis of which research and community-driven development in the area of cyber-threat intelligence management can occur.

5. Vendors with Planned Support

5.1. Threat Central, HP

HP has developed HP Threat Central, a security intelligence platform that enables automated, real-time collaboration between organizations to combat today’s increasingly sophisticated cyber attacks. One way automated sharing of threat indicators is achieved is through close integration with the HP ArcSight SIEM for automated upload and consumption of information from the Threat Central Server. In addition HP Threat Central supports open standards for sharing threat information so that participants who do not use HP Security Products can participate in the sharing ecosystem. General availability of Threat Central will be in 2014. It is planned that future versions also support IODEF for the automated upload and download of threat information.

5.2. DAEDALUS, NICT

DAEDALUS is a real-time alert system based on a large-scale darknet monitoring facility that has been deployed as a part of the nicter system of NICT, Japan. DAEDALUS consists of an analysis center (i.e., nicter) and several cooperate organizations. Each organization installs a darknet sensor and establishes a secure channel between it and the analysis center, and continuously forwards darknet traffic toward the center. In addition, each organization registers the IP address range of its livenet at the center in advance. When these distributed darknet sensors observe malware activities from the IP address of a cooperate organization, then the analysis center sends an alert to the organization. The future version of DAEDALUS will support IODEF for sending alert messages to the users.

6. Other Implementations

6.1. Collaborative Incident Management System

Collaborative Incident Management System (CIMS) is a proof-of-concept system for collaborative incident handling and for the sharing of cyber defence situational awareness information between the participants, developed for the Cyber Coalition 2013 (CC13) exercise organized by NATO. CIMS was implemented based on Request Tracker (RT), an open source software widely used for handling incident response by many CERTs and CSIRTs.

One of the functionality implemented in CIMS was the ability to import and export IODEF messages in the body of emails. The intent was to verify the suitability of IODEF to achieve the objective of collaborative incident handling. The customized version of RT could be configured to send an email message containing an IODEF message whenever an incident ticket was created, modified or deleted. These IODEF messages would then be imported into other incident handling systems in order to allow participating CSIRTs to use their usual means for incident handling, while still interacting with those using the proof-of-concept CIMS. Having an IODEF message generated for every change made to the incident information in RT (and for the system to allow incoming IODEF email messages to be associated to an existing incident) would in some way allow all participating CSIRTs to actually work on a "common incident ticket", at least at the conceptual level. Of particular importance was the ability for users to exchange information between each other concerning actions taken in the handling of a particular incident, thus creating a sort of common action log, as well as requesting/tasking others to provide information or perform specified action and correlating received responses to the original request or tasking. As well, a specific "profile" was developed to identify a subset of the IODEF classes that would be used during the exercise, in an attempt to channel all users into a common usage pattern of the otherwise flexible IODEF standard.

6.2. Automated Incident Reporting - AirCERT

AirCERT was implemented by CERT/CC of Carnegie Mellon’s Software Engineering Institute CERT division. AirCERT was designed to be an Internet-scalable distributed system for sharing security event data. The AirCERT system was designed to be an automated collector of flow and IDS alerts. AirCERT would collect that information into a relational database and be able to share reporting using IODEF and IDMEF. AirCERT additionally used SNML to exchange information about the network. AirCERT was implemented in a combination of C and perl modules and included periodic graphing capabilities leveraging RRDTool.

AirCERT was intended for large scale distributed deployment and eventually the ability to sanitize data to be shared across administrative domains. The architecture was designed to allow collection of data at a per site basis and to allow each site to create data sharing based on its own particular trust relationships.

6.3. US Department of Energy CyberFed

The CyberFed system was implemented and deployed by Argonne National Laboratory to automate the detection and response of attack activity against Department of Energy (DoE) computer networks. CyberFed automates the collection of network alerting activity from various perimeter network defenses and logs those events into its database. CyberFed then automatically converts that information into blocking information transmitted to all participants. The original implementation used IODEf messages wrapped in an XML extension to manage a large array of indicators. The CyberFed system was not designed to describe a particular incident as much as to describe a set of current network blocking indicators that can be generated and deployed machine-to-machine.

CyberFed is primarily implemented in Perl. Included as part of the CyberFed system are scripts which interact with a large number of firewalls, IDS/IPS devices, DNS systems, and proxies which operate to implement both the automated collection of events as well as the automated deployment of blacking.

Currently CyberFed supports multiple exchange formats including IODef and STIX. OpenIOC is also a potential exchange format that DoE is considering.

7. Implementation Guide

The section aims at sharing the tips for development of IODEF-capable systems.

7.1. Code Generators

For implementing IODEF-capable systems, it is feasible to employ code generators for XML Schema Document (XSD). The generators are used to save development costs since they automatically create useful libraries for accessing XML attributes, composing messages, and/or validating XML objects. The IODEF XSD was defined in section 8 of RFC 5070, and is availabe at http://www.iana.org/assignments/xml-registry/schema/iodef-1.0.xsd.

However, there still remains some problem. Due to the complexity of IODEF XSD, some code generators could not generate from the XSD file. The tested code generators were as follows.

- XML::Pastor [XSD:Perl] (Perl)
- RXSD [XSD:Ruby] (Ruby)
- PyXB [XSD:Python] (Python)
- JAXB [XSD:Java] (Java)
- CodeSynthesis XSD [XSD:Cxx] (C++)
- Xsd.exe [XSD:CS] (C#)

For instance, we have used XML::Pastor, but it could not properly understand its schema due to the complexity of IODEF XSD. The same applies to RXSD and JAXB. Only PyXB, CodeSynthesis XSD and Xsd.exe were able to understand the schema.

There is no recommended workaround, however, a double conversion of XSD file is one option to go through the situation; it means XSD is serialized to XML, and it is again converted to XSD. The resultant XSD was process-able by the all tools above.

It should be noted that IODEF uses '-' (hyphen) symbols in its classes or attributes, listed as follows.

- IODEF-Document Class; it is the top level class in the IODEF data model described in section 3.1 of [RFC5070].

- o The vlan-name and vlan-num Attribute; according to section 3.16.2 of [RFC5070], they are the name and number of Virtual LAN and are the attributes for Address class.

- o Extending the Enumerated Values of Attribute; according to section 5.1 of [RFC5070], it is a extension techniques to add new enumerated values to an attribute, and has a prefix of "ext-", e.g., ext-value, ext-category, ext-type, and so on.

According to the language specification, many programing language prohibit to contain '-' symbols in the name of class. The code generators must replace or remove '-' when building the librarlies. They should have the name space to restore '-' when outputting the XML along with IODEF XSD.

7.2. iodeflib

iodeflib is an open source implementation written in Python. This provides a simple but powerful APIs to create, parse and edit IODEF documents. It was designed in order to keep its interface as simple as possible, whereas generated libraries tend to inherit the complexity of IODEF XSD. As well as the interface, iodeflib involves functions of hiding some unnecessarily nested structures of the IODEF schema, and adding more convenient shortcuts.

This tool is available through the following link:

http://www.decalage.info/python/iodeflib

7.3. iodefpm

IODEF.pm is an open source implementation written in Perl. This also provides a simple interface for creating and parsing IODEF documents, in order to facilitate the translation of the a key-value based format to the IODEF representation. The module contains a generic XML DTD parser and includes a simplified node based representation of the IODEF DTD. It can hence easily be upgraded or extended to support new XML nodes or other DTDs.

This tool is available through the following link:

http://search.cpan.org/˜saxjazman/

7.4. Usability

Here notes some tips to avoid problems.

- o IODEF has category attribute for NodeRole class. Though various categories are described, they are not enough. For example, in the case of web mail servers, you should choose either "www" or "mail". One suggestion is selecting "mail" as the category attribute and adding "www" for another attirbute.

- o The numbering of Incident ID needs to be considered. Otherwise, information, such as the number of incidents within certain period could be observed by document receivers. For instance, we could randomize the assignment of the numbers.

8. Acknowledgements

The MILE Implementation report has been compiled through the submissions of implementers of INCH and MILE working group standards. A special note of thanks to the following contributors:

John Atherton, Surevine

Humphrey Browning, Deep-Secure

Dario Forte, DFLabs

Tomas Sander, HP

Ulrich Seldeslachts, ACDC

Takeshi Takahashi, National Institute of Information and Communications Technology Network Security Research Institute

Kathleen Moriarty, EMC

Bernd Grobauer, Siemens

Dandurand Luc, NATO

Pawel Pawlinski, NASK

9. IANA Considerations

This memo includes no request to IANA.

10. Security Considerations

This draft provides a summary of implementation reports from researchers and vendors who have implemented RFCs and drafts from the MILE and INCH working groups. There are no security considerations added in this draft because of the nature of the document.

11. Informative References

[RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, DOI 10.17487/RFC5070, December 2007, <http://www.rfc-editor.org/info/rfc5070>.

[RFC5901] Cain, P. and D. Jevans, "Extensions to the IODEF-Document Class for Reporting Phishing", RFC 5901, DOI 10.17487/RFC5901, July 2010, <http://www.rfc-editor.org/info/rfc5901>.

[RFC5941] M'Raihi, D., Boeyen, S., Grandcolas, M., and S. Bajaj, "Sharing Transaction Fraud Data", RFC 5941, DOI 10.17487/RFC5941, August 2010, <http://www.rfc-editor.org/info/rfc5941>.

[RFC6545] Moriarty, K., "Real-time Inter-network Defense (RID)", RFC 6545, DOI 10.17487/RFC6545, April 2012, <http://www.rfc-editor.org/info/rfc6545>.

[RFC6546] Trammell, B., "Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS", RFC 6546, DOI 10.17487/RFC6546, April 2012, <http://www.rfc-editor.org/info/rfc6546>.

[XSD:CS] Microsoft, "XML Schema Definition Tool (Xsd.exe)", <http://www.microsoft.com/>.

[XSD:Cxx] CodeSynthesis, "XSD - XML Data Binding for C++", <http://www.codesynthesis.com/>.

[XSD:Java]
Project Kenai, "JAXB Reference Implementation", <https://jaxb.java.net/>.

[XSD:Perl]
Ulsoy, A., "XML::Pastor", <http://search.cpan.org/˜aulusoy/XML-Pastor-1.0.4/>.

[XSD:Python]
Bigot, P., "PyXB: Python XML Schema Bindings", <https://pypi.python.org/pypi/PyXB>.

[XSD:Ruby]
Morsi, M., "RXSD - XSD / Ruby Translator", <https://github.com/movitto/RXSD>.

Authors' Addresses

Chris Inacio
Carnegie Mellon University
4500 5th Ave., SEI 4108
Pittsburgh, PA 15213
US

Email: inacio@andrew.cmu.edu

Daisuke Miyamoto
The Univerisity of Tokyo
2-11-16 Yayoi, Bunkyo
Tokyo 113-8658
JP

Email: daisu-mi@nc.u-tokyo.ac.jp

MILE Working Group M. Suzuki
Internet-Draft NICT
Intended status: Informational P. Kampanakis
Expires: April 21, 2016 Cisco Systems
October 19, 2015

IODEF Usage Guidance
draft-ietf-mile-iodef-guidance-04

Abstract

The Incident Object Description Exchange Format [RFC5070] defines a data representation that provides a framework for sharing information commonly exchanged by Computer Security Incident Response Teams (CSIRTs) about computer security incidents. Since the IODEF model includes a wealth of available options that can be used to describe a security incident or issue, it can be challenging for implementers to develop tools that can Leverage IODEF for incident sharing. This document provides guidelines for IODEF implementers. It will also address how common security indicators can be represented in IODEF and use-cases of how IODEF is being used so far.The goal of this document is to make IODEF’s adoption by vendors easier and encourage faster and wider adoption of the model by Computer Security Incident Response Teams (CSIRTs) around the world.

Table of Contents

1. Introduction

The Incident Object Description Exchange Format in [RFC5070] defines a data representation that provides a framework for sharing information commonly exchanged by Computer Security Incident Response

Teams (CSIRTs) about computer security incidents. The IODEF data model consists of multiple classes and data types that are used in the IODEF XML schema.

The IODEF schema was designed to be able to describe all the possible fields that would be needed in a security incident exchange. Thus, IODEF contains plenty data constructs that could potentially make it harder for IODEF implementers to decide which are the most important ones. Additionally, in the IODEF schema, there exist multiple fields and classes which do not necessarily need to be used in every possible data exchange. Moreover, there are fields that are useful only in data exchanges of non-traditional security events. This document tries to address the issues above. It will also address how common security indicators can be represented in IODEF. It will point out the most important IODEF classes for an implementer and describe other ones that are not as important. Also, it addresses some common challenges for IODEF implementers and how they should be addressed. The end goal of this document is to make IODEF's adoption by vendors easier and encourage faster and wider adoption of the model by Computer Security Incident Response Teams (CSIRTs) around the world.

Section 3 discusses the recommended classes and how an IODEF implementer should chose the classes to implement. Section 4 presents common considerations and implementer will come across and how to address them. Section 5 goes over some basic security concepts and how they can be expressed in IODEF.

2. Terminology

The terminology used in this document follows the one defined in [RFC5070] and [RFC7203].

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. Implementation Strategy

It is important for IODEF implementers to be able to distinguish how the IODEF classes will be used for incident information exchanges. It is critical for an implementer to follow a strategy according to which he will chose to implement various IODEF classes. It is also important to know what the most common classes that will be used to describe common security incident or indicators. Thus, this section will describe the most important classes and factors an IODEF implementer should take into consideration before designing the implementation or tool.

3.1. Recommended classes to implement

This section explains the mandatory to implement IODEF classes that are required more than once and also are useful. The mandatory top level class is the IODEF-Document class. One or more Incident class that represents the incident information need to be arranged under the IODEF-Document class. IncidentID, ReportTime, Assessment, and Contact under Incident class are mandatory classes. EventData, Record, Assessment, ReportTime, DetectTime, StartTime, ReportTime Description, Discovery

[TODO: More to be added...]

3.2. Decide what IODEF will be used for

This section describes that there is no need to implement all fields of IODEF, the ones that are necessary for your use-cases. The implementer should look into the schema and decide classes to implement (or not) Also it explains that other external schemata might be needed to describe incidents or indicators, based on SCI draft extensions.

[TODO: More to be added...]

4. IODEF considerations and how to address them

4.1. Unnecessary Fields

This section talks about fields that do not always play in important role like Assessment, Impact

[TODO: More to be added...]

4.2. External References

The IODEF format has the Reference class that refers to external information such as a vulnerability, Intrusion Detection System (IDS) alert, malware sample, advisory, or attack technique. However, due to insufficiency of the capability of the Reference class itself to describe external enumeration specifications, the Enumeretion Reference Format needs to be used with. The Enumeration Reference Format[RFC7495] specifies a format to include enumeration values from external data representations into IODEF, and manages references to external representations using IANA registry.

[TODO: More to be added...]

4.3. Extensions

The IODEF data model ([RFC5070]) is extensible. Many class attributes and their values can be extended using using the "ext-*" prefix. Additional classed can also be defined by using the AdditionalData and RecordItem classes. An extension to the AdditionalData class for reporting Phishing emails is defined in [RFC5901].

Additionally, IODEF can import existing schemata by using an extension framework defined in [RFC7203]. The framework enables IODEF users to embed XML data inside an IODEF document using external schemata or structures defined by external specifications. Examples include CVE, CVRF and OVAL. Thus, [RFC7203] enhances the IODEF capabilities without further extending the data model.

IODEF implementers should consider using their own IODEF extensions only for data that cannot be described using existing standards or importing them in and IODEF document using [RFC7203] is not a suitable option.

4.4. Predicate logic

IODEF [I-D.ietf-mile-rfc5070-bis] allows for nesting of incident information. For example, a EventData Class could include multiple Flows or Records. In turn, a Flow could consist of many Nodes and a Record of many RecordData classes. To ensure consistency, IODEF presumes certain predicate logic.

An EventData class that contains multiple EventData classes depicts an Event that consists of smaller events. For the parent event to take place, all the children EventData events SHOULD take place.

An EventData class with multiple Flows means that all the information defined in the flows need to exist for the event described to take place. For Records, the Records in an event just add more context to the event, they do not all need to be present for the event to take place. A Record in an EventData class with three RecordData in it, means that either of these RecordData classes needs to be present for the event described to take place.

In [RFC5070], if a Flow Class contained two System classes that have "source" and "target" as the category attributes, both Systems SHOULD be present in order for the Flow to be true and thus marked as an event. There SHOULD NOT be more than one "source" or "watchlist-source" and one "target" or "watchlist-target" Systems per Flow.

In Node class, Node information grouped together under a System class depicts different representations of the same System. For example, if a System consists of different Nodes with an IPv4 address, a domain-name and an IPv6 address, they all represent the same system. Of course, different representations could also be grouped under the same Node class.

[I-D.ietf-mile-rfc5070-bis] defined the HashData Class that describes a file's hash information as also described in [RFC5901]. Similar to the Node, if a HashData class consists of many digital signatures, the signatures represent alternative hash algorithms for the same signature. For example, if the HashData type is file-hash, then the signatures represent MD5, SHA1, SHA2 etc hashes.

For grouped Key classes the logic changes. Multiple Key classes in a WindowsRegistryKeysModified class represent necessary Windows Registry Keys that constitute an indicator. All SHOULD be present in order for the indicator to be present. Multiple WindowsRegistryKeysModified classes grouped under the same RecordData class represent alternatives for the same indicator. For example, if a RecordData class included two WindowsRegistryKeysModified classes, if either of the classes was true the RecordData class would be true.

4.5. Predicate Logic for watchlist of indicators

Multiple indicators occasionally need to be combined in an IODEF document. For example, a botnet might have multiple command and control servers. A consistent predicate logic for indicators SHOULD be followed in order to present such relationships in IODEF.

[I-D.ietf-mile-rfc5070-bis] defines two new category attributes in the System Class that can enhance the IODEF predicate logic functionality. These are watchlist-source and watchlist-target and they serve for watchlist indicator groupings. A watchlist of Systems means that the information is ORed with the information in the Flow section. In other words, if a Flow Class consists of multiple Systems with watchlist-source or watchlist-target attributes the Systems of the same watchlist type are ORed in the Flow Class. Multiple Flows in the EventData Class follow AND logic. There SHOULD NOT be more than one "watchlist-source" and one "watchlist-target" Systems per Flow. In the following example the EventData class will evaluates as a Flow of one System with source address being (10.10.10.104 OR 10.10.10.106) AND target address 10.1.1.1

```
<!-- ...XML code omitted... -->
<iodef:EventData>
   <iodef:Flow>
      <iodef:System category="watchlist-source" spoofed="no">
         <iodef:Node>
            <iodef:Address category="ipv4-addr">
               10.10.10.104
            </iodef:Address>
         </iodef:Node>
         <iodef:Node>
            <iodef:Address category="ipv4-addr">
               10.10.10.106
            </iodef:Address>
         </iodef:Node>
      </iodef:System>
      <iodef:System category="target">
         <iodef:Node>
            <iodef:Address category="ipv4-addr">
               10.1.1.1
            </iodef:Address>
         </iodef:Node>
      </iodef:System>
   </iodef:Flow>
</iodef:EventData>
<!-- ...XML code omitted... -->
```

Similarly, the HashData Class includes a type attribute that introduces watchlist groupings (i.e. PKI_email_ds_watchlist, PGP_email_ds_watchlist, file_hash_watchlist, email_hash_watchlist). Two HashData classes that contain a watchlist type attribute follow OR logic in a RecordData class. In the following example the RecordData class consists of either of the two files with two different hashes.

```
<!-- ...XML code omitted... -->
<iodef:RecordData>
   <iodef:HashData type="file-hash-watchlist">
      <iodef:FileName>dummy.txt</iodef:FileName>
      <ds:Reference>
         <ds:DigestMethod Algorithm=
           "http://www.w3.org/2001/04/xmlenc#sha256"/>
         <ds:DigestValue>
           141accec23e7e5157de60853cb1e01bc38042d
           08f9086040815300b7fe75c184
         </ds:DigestValue>
      </ds:Reference>
   </iodef:HashData>
   <iodef:HashData type="file-hash-watchlist">
      <iodef:FileName>dummy2.txt</iodef:FileName>
      <ds:Reference>
         <ds:DigestMethod Algorithm=
          "http://www.w3.org/2001/04/xmlenc#sha256"/>
         <ds:DigestValue>
           141accec23e7e5157de60853cb1e01bc38042d
           08f9086040815300b7fe75c184
         </ds:DigestValue>
      </ds:Reference>
   </iodef:HashData>
</iodef:RecordData>
<!-- ...XML code omitted... -->
```

Similarly, [I-D.ietf-mile-rfc5070-bis] introduces the WindowsRegistryKeyModified Class which consists of Key Classes. Key has an optional type attribute which has watchlist as an option in order to include the ability to group Keys. Multiple Keys of the same watchlist of indicators SHOULD be grouped in the same WindowsRegistryKeysModified Class. These Keys follow OR logic.

4.6. Indicator identifiers

[I-D.ietf-mile-rfc5070-bis] defines attrbutes indicator-set-id and indicator-uid. These are data elements that are commonly used as indicators. They are used in mutliple IODEF classes. Their purpose is to be able to define indicator relationships and reference respectively. The indicator-uid is used as a unique indicator identifier. Practitioners MAY use them to establish that a class represents an indicator that is different than other IODEF contextual information.

On the other hand, an IODEF report could contain multiple indicators that are part of the same or different indicator group. For example, an IP source address, a target address, that consitute a Flow and a

RecordData class respectively could be representing indicators of a virous and the traffic it generates. In such a situation, the indicator-set-id for all the classes (Address, RecordData) MUST be the same. Unrelated indicators MUST contain different indicator-set-id attributes or no indicator-set-id attributes.

Similarly,

4.7. Restrictions in IODEF

This section describes how Restriction can pose challenges

[TODO: More to be added...]

5. Current uses of IODEF

IODEF is currently used by various organizations in order to represent security incidents and share incident and threat information between security operations organizations.

5.1. Inter-vendor and Service Provider Exercise

Various vendors organized and executed an exercise where multiple threat indicators were exchanged using IODEF. The transport protocol used was RID. The threat information shared included incidents like DDoS attacks. Malware and Spear-Phishing. As this was a proof-of-concept (PoC) exercise only example information (no real threats) were shared as part of the exchanges.

```
      ____________                             ____________
     | Vendor X   |                           | Vendor Y   |
     | RID Agent  |_______-------------________| RID Agent |
     |___________|       | Internet  |        |___________|
                          -------------

                ---- RID Report message --->
                -- carrying IODEF example ->
                --------- over TLS -------->

                <----- RID Ack message -----
                <--- in case of failure ----
```

PoC peering topology

The figure above shows how RID interactions took place during the PoC. Participating organizations were running RID Agent software on-premises. The RID Agents formed peering relationships with other participating organizations. When Entity X had a new incident to exchange it would package it in IODEF and send it to Entity Y over TLS in a RID Report message. In case there was an issue with the message, Entity Y would send an RID Acknowledgement message back to Entity X which included an application level message to describe the issue. Interoperability between RID agents and the standards, [RFC6545] and [RFC6546], was also proven in this exercise. Appendix A includes some of the incident IODEF example information that was exchanged by the organizations' RID Agents as part of this proof-of-concept.

The first use-case included sharing of Malware Data Related to an Incident between CSIRTs. After Entity X detected an incident, she would put data about malware found during the incident in a backend system. Entity X then decided to share the incident information with Entity Y about the malware discovered. This could be a human decision or part of an automated process.

Below are the steps followed for the malware information exchange that was taking place:

(1) Entity X has a sharing agreement with Entity Y, and has already been configured with the IP address of Entity Y's RID Agent

(2) Entity X's RID Agent connects to Entity Y's RID Agent, and mutual authentication occurs using PKI certificates.

(3) Entity X pushes out a RID Report message which contains information about N pieces of discovered malware. IODEF is used in RID to discribe the

    (a) Hash of malware files

    (b) Registry settings changed by the malware

    (c) C&C Information for the malware

(4) Entity Y receives RID Report message, sends RID Acknowledgement message

(5) Entity Y stores the data in a format that makes it possible for the back end to know which source the data came from.

Another use-case was sharing Distributed Denial of Service (DDoS) as presented below information: Entity X, a Critical Infrastructure and Key Resource (CIKR) company detects that their internet connection is saturated with an abnormal amount of traffic. Further investigation determines that this is an actual DDoS attack. Entity X's computer incident response team (CIRT) contacts their ISP and shares information with them about the attack traffic characteristics. In addition, Entity X has an information sharing relationship with Entity Y. It shares information with Entity Y on characteristics of the attack to watch for. Entitty X's ISP is being overwhelmed by the amount of traffic, so it shares attack signatures and IP addresses of the most prolific hosts with its adjacent ISPs.

Below are the steps followed for a DDoS information exchange:

(1) Entity X has a sharing agreement with Entity Y, and has already been configured with the IP address of Entity Y's RID Agent

(2) Entity X's RID Agent connects to Entity Y's RID Agent, and mutual authentication occurs using PKI certificates.

(3) Entity X pushes out a RID Report message which contains information about the DDoS attack. IODEF is used in RID to discribe the

  (a) Start and Detect dates and times

  (b) IP Addresses of nodes sending DDoS Traffic

  (c) Sharing and Use Restrictions

  (d) Traffic characteristics (protocols and ports)

  (e) HTTP User-Agents used

  (f) IP Addresses of C&C for a botnet

(4) Entity Y receives RID Report message, sends RID Acknowledgement message

(5) Entity Y stores the data in a format that makes it possible for the back end to know which source the data came from.

One more use-case was sharing spear-phishing email information as explained in the following scenario: The board members of several defense contractors receive an email inviting them to attend a conference in San Francisco. The board members are asked to provide their personally identifiable information such as their home address, phone number, corporate email, etc in an attached document which came with the email. The board members were also asked to click on a URL which would allow them to reach the sign up page for the conference. One of the recipients believes the email to be a phishing attempt and forwards the email to their corporate CSIRT for analysis. The CSIRT identifies the email as an attempted spear phishing incident and distributes the indicators to their sharing partners.

Below are the steps followed for a spear-phishing information exchange between CSIRTs that was part of this PoC.

(1) Entity X has a sharing agreement with Entity Y, and has already been configured with the IP address of Entity Y's RID Agent

(2) Entity X's RID Agent connects to Entity Y's RID Agent, and mutual authentication occurs using PKI certificates.

(3) Entity X pushes out a RID Report message which contains information about the spear-phishing email. IODEF is used in RID to discribe the

    (a) Attachment details (file Name, hash, size, malware family

    (b) Target description (IP, domain, NSLookup)

    (c) Email information (From, Subject, header information, date/ time, digital signature)

    (d) Confidence Score

(4) Entity Y receives RID Report message, sends RID Acknowledgement message

(5) Entity Y stores the data in a format that makes it possible for the back end to know which source the data came from.

5.2. Implementations

[TODO: Mention and quickly describe [I-D.ietf-mile-implementreport] ...]

5.3. Other

IODEF is also used in various projects and products to consume and share security information. Various vendor incident reporting products have the ability to consume and export in IODEF format [implementations]. Perl and Python modules (XML::IODEF, Iodef::Pb, iodeflib) exist in order to parse IODEF documents and their extensions. Additionally, some worldwide CERT organizations are already able to use receive incident information in IODEF.

Future use-cases of IODEF could be:

(1) ISP notifying a national CERT or organization when it identifies and acts upon an incident and CERTs notifying ISPs when they are aware of incidents.

(2) Suspected phishing emails could be shared amongst organizations and national agencies. Automation could validate web content that the suspicious emails are pointing to. Identified malicious content linked in a phishing email could then be shared using IODEF. Phishing campaigns could thus be subverted much faster by automating information sharing using IODEF.

(3) When finding a certificate that should be revoked, a thrid-party would forward an automated IODEF message to the CA with the full context of the certificate and the CA could act accordingly after checking its validity. Alternatively, in the event of a compromise of the private key of a certificate, a third-party could alert the certificate owner about the compromise using IODEF.

6. Security Considerations

7. Updates

version -04 updates:

(1) Expanded on the Extensions section using Take’s suggestion.

(2) Moved Future use-cases under the Other section.

(3) CIF and APWG were consolidated in one "Implementation" section

(4) Added abstract of RFC7495 to the "External References" section

(5) Added Kathleen’s example of malware delivery URL to "Appendix"

(6) Added a little description to "Recommended classes to implement" section

version -03 updates:

(1) Added "Updates" section.

(2) Added details about the flow of information exchanges in "Inter-vendor and Service Provider Exercise" section. Also updated the usecases with more background information.

(3) Added future use-cases in the "Collective Intelligence Framework" section

(4) Updated Perl and Python references with the actual module names. Added IODEF implementation reference "implementations".

(5) Added Predicate logic section

(6) Updated Logic of watchlist of indicators section to simplify the logic and include examples.

(7) Renamed Externally defined indicators section to Indicator reference and elaborated on the use of indicator-uid and indicator-set-uid attribute use.

version -02 updates:

(1) Updated the "Logic for watchlist of indications" section to clarify the logic based on community feedback.

(2) Added "Inter-vendor and Service Provider Exercise" section.

(3) Added Appendix to include actual use-case IODEF examples.

8. Acknowledgements

9. Security Considerations

10. References

10.1. Normative References

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

[RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, DOI 10.17487/RFC5070, December 2007, <http://www.rfc-editor.org/info/rfc5070>.

[RFC5901] Cain, P. and D. Jevans, "Extensions to the IODEF-Document Class for Reporting Phishing", RFC 5901, DOI 10.17487/RFC5901, July 2010, <http://www.rfc-editor.org/info/rfc5901>.

[RFC6545] Moriarty, K., "Real-time Inter-network Defense (RID)", RFC 6545, DOI 10.17487/RFC6545, April 2012, <http://www.rfc-editor.org/info/rfc6545>.

[RFC6546] Trammell, B., "Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS", RFC 6546, DOI 10.17487/RFC6546, April 2012, <http://www.rfc-editor.org/info/rfc6546>.

[RFC7203] Takahashi, T., Landfield, K., and Y. Kadobayashi, "An Incident Object Description Exchange Format (IODEF) Extension for Structured Cybersecurity Information", RFC 7203, DOI 10.17487/RFC7203, April 2014, <http://www.rfc-editor.org/info/rfc7203>.

[RFC7495] Montville, A. and D. Black, "Enumeration Reference Format for the Incident Object Description Exchange Format (IODEF)", RFC 7495, DOI 10.17487/RFC7495, March 2015, <http://www.rfc-editor.org/info/rfc7495>.

10.2. Informative References

[APWG] "APWG", <http://apwg.org/>.

[CIF] "CIF", <http://csirtgadgets.org/collective-intelligence-framework/>.

[I-D.ietf-mile-implementreport] Inacio, C. and d. daisu-mi@nc.u-tokyo.ac.jp, "MILE Implementation Report", draft-ietf-mile-implementreport-06 (work in progress), October 2015.

[I-D.ietf-mile-rfc5070-bis] Danyliw, R. and P. Stoecker, "The Incident Object Description Exchange Format v2", draft-ietf-mile-rfc5070-bis-15 (work in progress), October 2015.

[implementations] "Implementations on IODEF", <http://siis.realmv6.org/implementations/>.

Appendix A. Inter-vendor and Service Provider Exercise Examples

Below some of the incident IODEF example information that was exchanged by the vendors as part of this proof-of-concept Inter-vendor and Service Provider Exercise.

A.1. Malware

In this test, malware information was exchanged using RID and IODEF. The information included file hashes, registry setting changes and the C&C servers the malware uses.

```
<?xml version="1.0" encoding="UTF-8"?>
          <iodef:IODEF-Document xmlns:ds="
              http://www.w3.org/2000/09/xmldsig#"
              xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.41">
     <iodef:Incident purpose="reporting">
       <iodef:ReportID name="EXAMPLE CSIRT">
          189234
       </iodef:ReportID>
       <iodef:ReportTime>
          2013-03-07T16:14:56.757+05:30
       </iodef:ReportTime>
       <iodef:Description>
          Malware and related indicators identified
       </iodef:Description>
       <iodef:Assessment occurrence="potential">
         <iodef:Impact severity="medium" type="info-leak">
            Malware with Command and Control Server
            and System Changes
         </iodef:Impact>
       </iodef:Assessment>
       <iodef:Contact role="creator" type="organization">
         <iodef:ContactName>EXAMPLE CSIRT</iodef:ContactName>
         <iodef:Email>emccirt@emc.com</iodef:Email>
       </iodef:Contact>
       <iodef:EventData>
         <iodef:Method>
           <iodef:Reference>
             <iodef:ReferenceName>Zeus</iodef:ReferenceName>
             <iodef:URL>
                 http://www.threatexpert.com/report.aspx?
                 md5=e2710ceb088dacdcb03678db250742b7
             </iodef:URL>
           </iodef:Reference>
         </iodef:Method>
         <iodef:Flow>
           <iodef:System category="watchlist-source">
             <iodef:Node>
               <iodef:Address category="ipv4-addr">
                  192.168.2.200
               </iodef:Address>
               <iodef:Address category="site-uri">
                  http://zeus.556677889900.com/log-bin/
                  lunch_install.php?aff_id=1&amp;amp;
                  lunch_id=1&amp;amp;maddr=&amp;amp;
                  action=install
               </iodef:Address>
               <iodef:NodeRole attacktype="c2-server"/>
             </iodef:Node>
```

```
          </iodef:System>
        </iodef:Flow>
        <iodef:Record>
          <iodef:RecordData>
            <iodef:HashData>
              <ds:Reference>
                <ds:DigestMethod Algorithm="
                   http://www.w3.org/2001/04/xmlenc#sha1"/>
                <ds:DigestValue>
                     MHg2NzUxQTI1MzQ4M0E2N0Q4NkUwRjg0NzYwRj
                     YxRjEwQkJDQzJFREZG</ds:DigestValue>
              </ds:Reference>
            </iodef:HashData>
            <iodef:HashData>
              <ds:Reference>
                <ds:DigestMethod Algorithm="
                    http://www.w3.org/2001/04/xmlenc#md5"/>
                <ds:DigestValue>
                    MHgyRTg4ODA5ODBENjI0NDdFOTc5MEFGQTg5NTE
                    zRjBBNA==
                </ds:DigestValue>
              </ds:Reference>
            </iodef:HashData>
            <iodef:WindowsRegistryKeysModified>
              <iodef:Key registryaction="add_value">
                <iodef:KeyName>
                    HKLM\Software\Microsoft\Windows\
                    CurrentVersion\Run\tamg
                </iodef:KeyName>
                <iodef:Value>
                   ?\?\?%System%\wins\mc.exe\?\??
                </iodef:Value>
              </iodef:Key>
              <iodef:Key registryaction="modify_value">
                <iodef:KeyName>HKLM\Software\Microsoft\
                   Windows\CurrentVersion\Run\dqo
                </iodef:KeyName>
                <iodef:Value>"\"\"%Windir%\Resources\
                    Themes\Luna\km.exe\?\?"
                </iodef:Value>
              </iodef:Key>
            </iodef:WindowsRegistryKeysModified>
          </iodef:RecordData>
        </iodef:Record>
      </iodef:EventData>
      <iodef:EventData>
        <iodef:Method>
          <iodef:Reference>
```

```
                    <iodef:ReferenceName>Cridex</iodef:ReferenceName>
                    <iodef:URL>
                        http://www.threatexpert.com/report.aspx?
                        md5=c3c528c939f9b176c883ae0ce5df0001
                    </iodef:URL>
                  </iodef:Reference>
                </iodef:Method>
                <iodef:Flow>
                  <iodef:System category="watchlist-source">
                    <iodef:Node>
                      <iodef:Address category="ipv4-addr">
                         10.10.199.100
                      </iodef:Address>
                      <iodef:NodeRole attacktype="c2-server"/>
                    </iodef:Node>
                    <iodef:Service ip_protocol="6">
                      <iodef:Port>8080</iodef:Port>
                    </iodef:Service>
                  </iodef:System>
                </iodef:Flow>
                <iodef:Record>
                  <iodef:RecordData>
                    <iodef:HashData>
                      <ds:Reference>
                        <ds:DigestMethod Algorithm="
                           http://www.w3.org/2001/04/xmlenc#sha1"/>
                        <ds:DigestValue>
                           MHg3MjYzRkUwRDNBMDk1RDU5QzhFMEM4OTVBOUM
                           1ODVFMzQzRTcxNDFD
                        </ds:DigestValue>
                      </ds:Reference>
                      <ds:Reference>
                        <ds:DigestMethod Algorithm="
                           http://www.w3.org/2001/04/xmlenc#md5"/>
                        <ds:DigestValue>MHg0M0NEODUwRkNEQURFNDMzMEE1
                           QkVBNkYxNkVFOTcxQw==</ds:DigestValue>
                      </ds:Reference>
                    </iodef:HashData>
                    <iodef:HashData>
                      <ds:Reference>
                        <ds:DigestMethod Algorithm="
                            http://www.w3.org/2001/04/xmlenc#md5"/>
                        <ds:DigestValue>MHg0M0NEODUwRkNEQURFNDMzMEE
                            1QkVBNkYxNkVFOTcxQw==</ds:DigestValue>
                      </ds:Reference>
                      <ds:Reference>
                        <ds:DigestMethod Algorithm="http://www.w3.org/
                            2001/04/xmlenc#sha1"/>
```

```
                         <ds:DigestValue>MHg3MjYzRkUwRDNBMDk1RDU5QzhFME
                             M4OTVBOUM1ODVFMzQzRTcxNDFD</ds:DigestValue>
                       </ds:Reference>
                     </iodef:HashData>
                     <iodef:WindowsRegistryKeysModified>
                       <iodef:Key registryaction="add_value">
                         <iodef:KeyName>
                            HKLM\Software\Microsoft\Windows\
                            CurrentVersion\Run\KB00121600.exe
                         </iodef:KeyName>
                         <iodef:Value>
                            \?\?%AppData%\KB00121600.exe\?\?
                         </iodef:Value>
                       </iodef:Key>
                     </iodef:WindowsRegistryKeysModified>
                   </iodef:RecordData>
                 </iodef:Record>
               </iodef:EventData>
               <iodef:EventData>
                 <iodef:Expectation action="other"/>
                   <iodef:Flow>
                     <iodef:System category="source"
                           indicator-set-id="91011">
                       <iodef:Node>
                           <iodef:Address category="url"
                              indicator-uid="qrst">
                                 http://foo.com:12345/evil/cc.php
                           </iodef:Address>
                       <iodef:NodeName indicator-uid="rstu">
                           evil.com
                       </iodef:NodeName>
                           <iodef:Address category="ipv4-addr"
                              indicator-uid="stuv">
                                 1.2.3.4</iodef:Address>
                           <iodef:Address category="ipv4-addr"
                              indicator-uid="tuvw">
                                 5.6.7.8 </iodef:Address>
                           <iodef:Address category="ipv6-addr"
                              indicator-uid="uvwx">
                                  2001:dead:beef::</iodef:Address>
                      <iodef:NodeRole category="c2-server"/>
                      </iodef:Node>
                   </iodef:System>
                 </iodef:Flow>
                 <iodef:Record>
                   <iodef:RecordData indicator-set-id="91011">
                     <iodef:HashData>
                        <ds:Reference>
```

```
                           <ds:DigestMethod Algorithm=
                               "http://www.w3.org/2001/04/xmlenc
                                #sha256"/>
                            <ds:DigestValue>
                             141accec23e7e5157de60853cb1e01bc3804
                             2d08f9086040815300b7fe75c184
                            </ds:DigestValue>
                        </ds:Reference>
                       </iodef:HashData>
                     <iodef:WindowsRegistryKeysModified
                       indicator-set-id="91011">
                       <iodef:Key registryaction="add_key"
                       indicator-uid="vwxy">
                         <iodef:KeyName>
                             HKLM\SYSTEM\CurrentControlSet\
                             Services\.Net CLR
                         </iodef:KeyName>
                       </iodef:Key>
                       <iodef:Key registryaction="add_key"
                         indicator-uid="wxyz">
                         <iodef:KeyName>
                             HKLM\SYSTEM\CurrentControlSet\
                             Services\.Net CLR\Parameters
                         </iodef:KeyName>
                         <iodef:Value>
                             \"\"%AppData%\KB00121600.exe\"\"
                         </iodef:Value>
                       </iodef:Key>
                       <iodef:Key registryaction="add_value"
                         indicator-uid="xyza">
                         <iodef:KeyName>
                            HKLM\SYSTEM\CurrentControlSet\Services\
                            .Net CLR\Parameters\ServiceDll
                         </iodef:KeyName>
                         <iodef:Value>C:\bad.exe</iodef:Value>
                       </iodef:Key>
                       <iodef:Key registryaction="modify_value"
                         indicator-uid="zabc">
                         <iodef:KeyName>
                             HKLM\SYSTEM\CurrentControlSet\
                             Services\.Net CLR\Parameters\Bar
                         </iodef:KeyName>
                         <iodef:Value>Baz</iodef:Value>
                       </iodef:Key>
                     </iodef:WindowsRegistryKeysModified>
                   </iodef:RecordData>
                 </iodef:Record>
               </iodef:EventData>
```

```
             </iodef:Incident>
           </iodef:IODEF-Document>
```

A.2. Malware Delivery URL

[TODO: fix indicator_uid attribute to Indicator class]

This example indicates malware and related URL for file delivery.


```
<?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:xsi="http://www.w3.org/2001/XMLSchema-instance">
<iodef:Incident purpose="reporting">
      <iodef:IncidentID name="csirt.example.com">
          189801
      </iodef:IncidentID>
      <iodef:ReportTime>2012-12-05T12:20:00+00:00</iodef:ReportTime>
      <iodef:Description>Malware and related indicators</iodef:Description>
          <iodef:Assessment occurrence="potential">
              <iodef:Impact severity="medium" type="info-leak">Malware with C&am
p;C </iodef:Impact>
           </iodef:Assessment>
                   <iodef:Contact role="creator" type="organization">
                <iodef:ContactName>example.com CSIRT
                </iodef:ContactName>
                <iodef:Email>contact@csirt.example.com</iodef:Email>
            </iodef:Contact>
            <iodef:EventData>
            <iodef:Flow>
<iodef:System category="source">
                  <iodef:Node>
                    <iodef:Address indicator_uid="xxxxx" category="ipv4-addr">19
2.168.2.200</iodef:Address>
                    <iodef:Address indicator_uid="xxxxx" category="url">
                      http://zeus.556677889900.example.com/log-bin/lunch_install
.php?aff_id=1&amp;lunch_id=1&amp;maddr=&amp;action=install</iodef:Address>
                    <iodef:NodeRole category="c2-server"/>
                 </iodef:Node>
              </iodef:System>
            </iodef:Flow>
        </iodef:EventData>
    </iodef:Incident>
</IODEF-Document>
```


A.3. DDoS

The DDoS test exchanged information that described a DDoS including protocols and ports, bad IP addresses and HTTP User-Agent fields.

The IODEF version used for the data representation was based on [I-D.ietf-mile-rfc5070-bis]


```
<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="1.00" lang="en"
    xmlns="urn:ietf:params:xml:ns:iodef-1.41"
    xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.41"
    xmlns:iodef-sci="urn:ietf:params:xml:ns:iodef-sci-1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
     <iodef:Incident purpose="reporting" restriction="default">
         <iodef:IncidentID name="csirt.example.com">
             189701
         </iodef:IncidentID>
         <iodef:StartTime>2013-02-05T00:34:45+00:00</iodef:StartTime>
         <iodef:DetectTime>2013-02-05T01:15:45+00:00</iodef:DetectTime>
         <iodef:ReportTime>2013-02-05T01:34:45+00:00</iodef:ReportTime>
         <iodef:description>DDoS Traffic Seen</iodef:description>
            <iodef:Assessment occurrence="actual">
                <iodef:Impact severity="medium" type="dos">
                DDoS Traffic</iodef:Impact>
                <iodef:Confidence rating="numeric">90
                </iodef:Confidence>
            </iodef:Assessment>
            <iodef:Contact role="creator" type="organization">
                <iodef:ContactName>Dummy Test</iodef:ContactName>
                <iodef:Email>contact@dummytest.com</iodef:Email>
            </iodef:Contact>
            <iodef:EventData>
               <iodef:Description>
                   Dummy Test sharing with ISP1
               </iodef:Description>
            <iodef:Expectation action="other"/>
            <iodef:Method>
               <iodef:Reference>
                  <iodef:ReferenceName>
                      Low Orbit Ion Cannon User Agent
                  </iodef:ReferenceName>
                  <iodef:URL>
                     http://blog.spiderlabs.com/2011/01/loic-ddos-
                     analysis-and-detection.html
                  </iodef:URL>
                  <iodef:URL>
                     http://en.wikipedia.org/wiki/Low_Orbit_Ion_Cannon
                  </iodef:URL>
               </iodef:Reference>
            </iodef:Method>
            <iodef:Flow>
```

```
                <iodef:System category="watchlist-source" spoofed="no">
                   <iodef:Node>
                     <iodef:Address category="ipv4-addr">
                        10.10.10.104</iodef:Address>
                   </iodef:Node>
                   <iodef:Node>
                     <iodef:Address category="ipv4-addr">
                        10.10.10.106</iodef:Address>
                                    </iodef:Node>
                   <iodef:Node>
                     <iodef:Address category="ipv4-net">
                      172.16.66.0/24</iodef:Address>
                      </iodef:Node>
                   <iodef:Node>
                     <iodef:Address category="ipv6-addr">
                      2001:db8:dead:beef::</iodef:Address>
                                  </iodef:Node>
    <iodef:Service ip_protocol="6">
       <iodef:Port>1337</iodef:Port>
       <iodef:Application user-agent="Mozilla/5.0 (Macintosh; U;
              Intel Mac OS X 10.5; en-US; rv:1.9.2.12) Gecko/
              20101026 Firefox/3.6.12">
       </iodef:Application>
    </iodef:Service>
               </iodef:System>
                <iodef:System category="target">
                   <iodef:Node>
                     <iodef:Address category="ipv4-addr">
    10.1.1.1</iodef:Address>
                  </iodef:Node>
                  <iodef:Service ip_protocol="6">
                    <iodef:Port>80</iodef:Port>
                  </iodef:Service>
                </iodef:System>
                <iodef:System category="sensor"><iodef:Description>
                    Information provided in FLow class instance is from
                    Inspection of traffic from network tap
                </iodef:Description></iodef:System>
             </iodef:Flow>
         </iodef:EventData>
     </iodef:Incident>
 </IODEF-Document>
```

A.4. Spear-Phishing

The Spear-Phishing test exchanged information that described a Spear-Phishing email including DNS records and addresses about the sender, malicious attached file information and email data. The IODEF version used for the data representation was based on [I-D.ietf-mile-rfc5070-bis].


```
<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="1.00" lang="en"
    xmlns="urn:ietf:params:xml:ns:iodef-1.41"
    xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.41"
    xmlns:iodef-sci="urn:ietf:params:xml:ns:iodef-sci-1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
     <iodef:Incident purpose="reporting">
         <iodef:IncidentID name="csirt.example.com">
            189601
         </iodef:IncidentID>
         <iodef:StartTime>2013-01-04T08:01:34+00:00</iodef:StartTime>
         <iodef:StopTime>2013-01-04T08:31:27+00:00</iodef:StopTime>
         <iodef:DetectTime>2013-01-04T08:06:12+00:00</iodef:DetectTime>
         <iodef:ReportTime>2013-01-04T09:15:45+00:00</iodef:ReportTime>
         <iodef:description>
            Zeus Spear Phishing E-mail with Malware Attachment
         </iodef:description>
            <iodef:Assessment occurrence="potential">
                <iodef:Impact severity="medium" type="info-leak">
                Malware with Command and Control Server and System
                Changes</iodef:Impact>
            </iodef:Assessment>
            <iodef:Contact role="creator" type="organization">
                <iodef:ContactName>example.com CSIRT
                </iodef:ContactName>
                <iodef:Email>contact@csirt.example.com</iodef:Email>
            </iodef:Contact>
            <iodef:EventData>
              <iodef:Description>Targeting Defense Contractors,
                 specifically board members attending Dummy Con
              </iodef:Description>
            <iodef:Expectation action="other"/>
            <iodef:Method>
               <iodef:Reference indicator_uid="1234">
                  <iodef:ReferenceName>Zeus</iodef:ReferenceName>
                  </iodef:Reference>
            </iodef:Method>
            <iodef:Flow>
               <iodef:System category="source">
                  <iodef:Node>
                    <iodef:Address category="url">
                          http://www.zeusevil.com</iodef:Address>
                    <iodef:Address category="ipv4-addr">
                          10.10.10.166</iodef:Address>
```

```
                    <iodef:Address category="as">
                           225</iodef:Address>
                     <iodef:Address category="ext-value"
                       ext-category="as-name">
                       EXAMPLE-AS - University of Example"
                       </iodef:Address>
                     <iodef:Address category="ext-value"
                       ext-category="as-prefix">
                       172.16..0.0/16
                       </iodef:Address>
                     <iodef:NodeRole category="www"
                       attacktype="malware-distribution"/>
                 </iodef:Node>
              </iodef:System>
            </iodef:Flow>
            <iodef:Flow>
               <iodef:System category="source">
                  <iodef:Node>
                   <iodef:NodeName>mail1.evildave.com</iodef:NodeName>
                    <iodef:Address category="ipv4-addr">
                          172.16.55.6</iodef:Address>
                   <iodef:Address category="asn">
                          225</iodef:Address>
                    <iodef:Address category="ext-value"
                      ext-category="as-name">
                      EXAMPLE-AS - University of Example
                    </iodef:Address>
    <iodef:DomainData>
       <iodef:Name>evildaveexample.com</iodef:Name>
       <iodef:DateDomainWasChecked>2013-01-04T09:10:24+00:00
       </iodef:DateDomainWasChecked>
       <iodef:RelatedDNS RecordType="MX">
           evildaveexample.com MX prefernce = 10, mail exchanger
           = mail1.evildave.com</iodef:RelatedDNS>
       <iodef:RelatedDNS RecordType="A">
           mail1.evildaveexample.com
           internet address = 172.16.55.6</iodef:RelatedDNS>
       <iodef:RelatedDNS RecordType="SPF">
            zuesevil.com. IN TXT \"v=spf1 a mx -all\"
       </iodef:RelatedDNS>
    </iodef:DomainData>
                 <iodef:NodeRole category="mail"
                       attacktype="spear-phishing"/>
                 </iodef:Node>
                 <iodef:Service>
                    <iodef:EmailInfo>
                       <iodef:Email>emaildave@evildaveexample.com
                       </iodef:Email>
```

```
                    <iodef:EmailSubject>Join us at Dummy Con
                    </iodef:EmailSubject>
                    <iodef:X-Mailer>StormRider 4.0
                    </iodef:X-Mailer>
                 </iodef:EmailInfo>
              </iodef:Service>
           </iodef:System>
           <iodef:System category="target">
              <iodef:Node>
                 <iodef:Address category="ipv4">
                   192.168.54.2</iodef:Address>
              </iodef:Node>
            </iodef:System>
         </iodef:Flow>

         <iodef:Record>
           <iodef:RecordData>
             <iodef:HashData type="file_hash"
                indicator_uid="1234">
                <iodef:FileName>Dummy Con Sign Up Sheet.txt
                </iodef:FileName>
                <iodef:FileSize>152</iodef:FileSize>
             <ds:Reference>
                <ds:DigestMethod Algorithm=
                   "http://www.w3.org/2001/04/xmlenc#sha256"/>
                   <ds:DigestValue>
                     141accec23e7e5157de60853cb1e01bc38042d
                     08f9086040815300b7fe75c184
                   </ds:DigestValue>
             </ds:Reference>
           </iodef:HashData>
          </iodef:RecordData>
          <iodef:RecordData>
           <iodef:HashData type="PKI_email_ds" valid="0">
             <ds:Signature>
              <ds:KeyInfo>
                 <ds:X509Data>
                    <ds:X509IssuerSerial>
                       <ds:X509IssuerName>FakeCA
                    </ds:X509IssuerName>
                    </ds:X509IssuerSerial>
                    <ds:X509SubjectName>EvilDaveExample
                    </ds:X509SubjectName>
                 </ds:X509Data>
              </ds:KeyInfo>
                 <ds:SignedInfo>
                  <ds:Reference>
                     <ds:DigestMethod Algorithm=
```

```
                           "http://www.w3.org/2001/04/xmlenc#sha256"/>
                         <ds:DigestValue>
                          352bddec13e4e5257ee63854cb1f05de48043d09f9
                          076070845307b7ce76c185
                         </ds:DigestValue>
                     </ds:Reference>
                   </ds:SignedInfo>
                 </ds:Signature>
              </iodef:HashData>
           </iodef:RecordData>
          </iodef:Record>
         </iodef:EventData>
     </iodef:Incident>
 </IODEF-Document>
```


Authors' Addresses

Mio Suzuki
NICT
4-2-1, Nukui-Kitamachi
Koganei, Tokyo 184-8795
JP

Email: mio@nict.go.jp

Panos Kampanakis
Cisco Systems
170 West Tasman Dr.
San Jose, CA 95134
US

Email: pkampana@cisco.com

MILE Working Group R. Danyliw
Internet-Draft CERT
Obsoletes: 5070 (if approved) P. Stoecker
Intended status: Standards Track RSA
Expires: April 18, 2016 October 16, 2015

The Incident Object Description Exchange Format v2
draft-ietf-mile-rfc5070-bis-15

Abstract

The Incident Object Description Exchange Format (IODEF) defines a data representation 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

Copyright Notice

Table of Contents

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:

- cyber intelligence to characterize threats;
- cyber incident reports to document particular cyber security events or relationships between events;
- cyber event mitigation to request proactive and reactive mitigation approaches to cyber intelligence or incidents; and
- cyber information sharing meta-data so that these various classes of information can be exchanged among parties.

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:

- increased automation in processing of incident data, since the resources of security analysts to parse free-form textual documents will be reduced;
- decreased effort in normalizing similar data (even when highly structured) from different sources; and

- 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

This document contains changes with respect to its predecessor RFC5070.

- All of the RFC5070 Errata was implemented.

- Imported the xmlns:ds namespace to include digital signature hash classes.

- The following classes were added to IODEF-Document: AdditionalData.

- The following class and attribute was added to Incident: IndicatorData and @status.

- The following classes were added to Incident and EventData: GenerationTime and Discovery.

- The following classes and attributes were added to the Service class: EmailData, DomainData, AssetID, ApplicationHeader @virtual, and @ownership. Service@ip_protocol was renamed to @ip-protocol.

- The following classes were added to the Record class: HashData and WindowsRegistryKeysModified.

- The following classes were added to the RelatedActivity class: ThreatActor, Campaign, Confidence, Description, and AdditionalData.

- The following classes were added to Assessment: IncidentCategory, SystemImpact, BusinessImpact, IntendedImpact and MitigatingFactor.

- The following classes were added to Node: PostalAddress and DomainData. The following classes were removed from Node: Removed NodeName and DateTime.

- The following classes were added to the Contact class: ContactTitle.

- The following classes were added to Expectation and HistoryItem: DefinedCOA.

- The following classes were added to Service: ServiceName

- The following classes were added to Reference: ReferenceName (replaced Name).

- The following attributes were added to Counter: type and unit.

- Additional enumerated values were added to the following attributes: @restriction, {Expectation, HistoryItem}@action, NodeRole@category, Incident@purpose, Contact@role, AdditionalData@dtype, System@spoofed.

- Added option for public extension of enumerated attributes with an IANA registry and added @ext-restriction.

- Removed Impact class in favor of using SystemImpact and IncidentCategory.

- iodef:MLStringType uses xml:lang and @translation-id.

- Incident/ReportTime and Assessment are longer mandatory.

- Incident/GenerateTime is mandatory.

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].

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

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 complementary. The data model of the Intrusion

Detection Message Exchange Format [RFC4765] 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) [refs.requirements].

1.5. About the IODEF Implementation

The IODEF implementation is specified as an Extensible Markup Language (XML) [W3C.XML] Schema [W3C.SCHEMA].

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 [W3C.SCHEMA.DTYPES]) 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" in [W3C.SCHEMA.DTYPES].

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" in [W3C.SCHEMA.DTYPES].

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" in [W3C.SCHEMA.DTYPES].

2.4. Multilingual Strings

A character string that needs to be represented in a language different than the default encoding of the document is of the ML_STRING data type.

ML_STRING data type is implemented as the "iodef:MLStringType" type in the schema. This type extends the "xs:string" to include two attributes. The body of any class that uses this type is the multilingual string.

```
+-----------------------+
| iodef:MLStringType    |
+-----------------------+
| ENUM xml:lang         |
| STRING translation-id |
|                       |
+-----------------------+
```

Figure 1: The iodef:MLStringType Type

Classes of the iodef:MLStringType type have two attributes:

xml:lang
: Optional. ENUM. A language identifier per Section 2.12 of [W3C.XML] whose values and form are described in [RFC5646]. The interpretation of this code is described in Section 6.

translation-id
: Optional. STRING. An identifier to relate other instances of this class with the same parent as translations of this text. The scope of this identifier is limited to all of the direct, peer child classes of a given parent class.

Using this class enables representing translation of the same text in multiple language. Each translation is a distinct instance of this class with a common parent. This relationship between multiple classes being translated instances of the same text is indicated by a common identifier set in the translation-id attribute. The language of a given class of this type is set by the xml:lang attribute.

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" in [W3C.SCHEMA.DTYPES].

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" in [W3C.SCHEMA.DTYPES].

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 [ISO8601] documented in [RFC3339].

The DATETIME data type is implemented as an "xs:dateTime" 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 [W3C.SCHEMA.DTYPES]. 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 [RFC4519]. 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 [RFC4519].

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 [RFC4519].

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 [RFC5322].

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 [RFC3986].

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

2.16. Identifiers and Identifier References

An identifier unique to the Document is represented by the ID data type. A reference to this identifier is represented by the IDREF data type. The acceptable format of ID and IDREF is documented in Section 3.3.8 and 3.3.9 of [W3C.SCHEMA.DTYPES].

The ID and IDREF data types are implemented as "xs:ID" and "xs:IDREF" 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.

```
+-------------------------+
| IODEF-Document          |
+-------------------------+
| STRING version          |<>--{1..*}--[ Incident       ]
| ENUM xml:lang           |<>--{0..*}--[ AdditionalData ]
| STRING format-id        |
| STRING private-enum-name |
| STRING private-enum-id  |
+-------------------------+
```

Figure 2: IODEF-Document Class

The aggregate class that constitute IODEF-Document is:

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

AdditionalData

Zero or more. Mechanism by which to extend the data model. See Section 3.9

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 "2.00"

xml:lang
: Optional. ENUM. A language identifier per Section 2.12 of [W3C.XML] whose values and form are described in [RFC5646]. The interpretation of this code is described in Section 6.

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

private-enum-name
: Optional. STRING. A globally unique identifier for the CSIRT generating the document to deconflict private extensions used in the Document. The fully qualified domain name associated with the CSIRT MUST be used as the identifier.

private-enum-id
: Optional. STRING. An organizationally unique identifier for an extension used in the Document. If this attribute is set, the private-enum-name MUST also be set.

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 status            |<>--{0..*}--[ RelatedActivity ]
| STRING ext-status      |<>--{0..1}--[ DetectTime      ]
| ENUM xml:lang          |<>--{0..1}--[ StartTime       ]
| ENUM restriction       |<>--{0..1}--[ EndTime         ]
| STRING ext-restriction |<>--{0..1}--{ RecoveryTime    ]
| STRING observable-id   |<>--{0..1}--[ ReportTime      ]
|                        |<>----------[ GenerationTime  ]
|                        |<>--{0..*}--[ Description     ]
|                        |<>--{0..*}  [ Discovery       ]
|                        |<>--{0..*}--[ Assessment      ]
|                        |<>--{0..*}--[ Method          ]
|                        |<>--{1..*}--[ Contact         ]
|                        |<>--{0..*}--[ EventData       ]
|                        |<>--{0..*}--[ IndicatorData   ]
|                        |<>--{0..1}--[ History         ]
|                        |<>--{0..*}--[ AdditionalData  ]
+------------------------+
```

Figure 3: 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 more. Related activity and attribution of this activity.

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.

RecoveryTime

Zero or one. The time the site recovered from the incident.

ReportTime
Zero or one. The time the incident was reported.

GenerationTime
One. The time the content in this Incident class was generated.

Description
Zero or more. ML_STRING. A free-form textual description of the incident.

Discovery
Zero or more. The means by which this incident was detected.

Assessment
Zero 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.

IndicatorData
Zero or more. Description of indicators.

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 eight 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 3.17). These values are maintained in the "Incident-purpose" IANA registry per Table 1. 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. watch. The document was sent to convey indicators to watch for particular activity.

5. other. The document was sent for purposes specified in the Expectation class.

6. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-purpose
Optional. STRING. A means by which to extend the purpose attribute. See Section 5.1.1.

status
Optional. ENUM. The status attribute conveys the state in a workflow where the incident is currently found. These values are maintained in the "Incident-status" IANA registry per Table 1. This attribute is defined as an enumerated list:

1. new. The document is newly reported and has not been actioned.

2. in-progress. The contents of this document are under investigation.

3. forwarded. The document has been forwarded to another party for handling.

4. resolved. The investigation into the activity in this document has concluded.

5. future. The described activity has not yet been detected.

6. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-status
Optional. STRING. A means by which to extend the status attribute. See Section 5.1.1.

xml:lang
Optional. ENUM. A language identifier per Section 2.12 of [W3C.XML] whose values and form are described in [RFC5646]. The interpretation of this code is described in Section 6.

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

3.3. Common Attributes

There are a number of recurring attributes used by the data model. They are documented in this section.

3.3.1. restriction Attribute

The restriction 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.

These values are maintained in the "Restriction" IANA registry per Table 1.

1. public. The information can be freely distributed without restriction.

2. partner. The information may be shared within a closed community of peers, partners, or affected parties, but cannot be openly published.

3. need-to-know. The information may be shared only within the organization with individuals that have a need to know.

4. private. The information may not be shared.

5. default. The information can be shared according to an information disclosure policy pre-arranged by the communicating parties.

6. white. Same as 'public'.

7. green. Same as 'partner'.

8. amber. Same as 'need-to-know'.

9. red. Same as 'private'.

10. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

3.3.2. observable-id Attribute

Information included in an incident report may be an observable relevant to an indicator. The observable-id attribute provides a unique identifier in the scope of the document for this observable. This identifier can then be used to reference the observable with an ObservableReference class to define an indicator in the IndicatorData class.

3.4. 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.

```
+------------------------+
| IncidentID             |
+------------------------+
| STRING                 |
|                        |
| STRING name            |
| STRING instance        |
| ENUM restriction       |
| STRING ext-restriction |
+------------------------+
```

Figure 4: The IncidentID Class

The IncidentID class has four 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. See Section 3.3.1. The default value is "public".

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.5. 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 in 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 must never be considered an AlternativeID.

```
+------------------------+
| AlternativeID          |
+------------------------+
| ENUM restriction       |<>--{1..*}--[ IncidentID ]
| STRING ext-restriction |
+------------------------+
```

Figure 5: 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 two attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.6. RelatedActivity Class

The RelatedActivity class relates the information described in the rest of the IODEF document to previously observed incidents or activity; and allows attribution to a specific actor or campaign.

```
+------------------------+
| RelatedActivity        |
+------------------------+
| ENUM restriction       |<>--{0..*}--[ IncidentID     ]
| STRING ext-restriction |<>--{0..*}--[ URL            ]
|                        |<>--{0..*}--[ ThreatActor    ]
|                        |<>--{0..*}--[ Campaign       ]
|                        |<>--{0..1}--[ Confidence     ]
|                        |<>--{0..*}--[ Description    ]
|                        |<>--{0..*}--[ AdditionalData ]
+------------------------+
```

Figure 6: 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.

ThreatActor
One or more. The threat actor to whom the described activity is attributed.

Campaign
One or more. The campaign of a given threat actor to whom the described activity is attributed.

Confidence
Zero or one. An estimate of the confidence in attributing this RelatedActivity to the event described in the document.

Description
Zero or more. ML_STRING. A description of how these relationships were derived.

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

RelatedActivity MUST at least have one instance of IncidentID, URL, ThreatActor, or Campaign.

The RelatedActivity class has two attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.7. ThreatActor Class

The ThreatActor class describes a given actor.

```
+------------------------+
| ThreatActor            |
+------------------------+
| ENUM restriction       |<>--{0..1}--[ ThreatActorID  ]
| STRING ext-restriction |<>--{0..*}--[ Description    ]
|                        |<>--{0..*}--[ AdditionalData ]
+------------------------+
```

Figure 7: ThreatActor Class

The aggregate classes that constitutes ThreatActor are:

ThreatActorID
One or more. STRING. An identifier for the ThreatActor.

Description
One or more. ML_STRING. A description of the ThreatActor.

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

ThreatActor MUST have at least one instance of a ThreatActorID or Description.

The ThreatActor class has two attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.8. Campaign Class

The Campaign class describes a campaign of attacks by a threat actor.

```
+------------------------+
| Campaign               |
+------------------------+
| ENUM restriction       |<>--{0..1}--[ CampaignID     ]
| STRING ext-restriction |<>--{0..*}--[ Description    ]
|                        |<>--{0..*}--[ AdditionalData ]
+------------------------+
```

Figure 8: Campaign Class

The aggregate classes that constitutes Campaign are:

CampaignID
One or more. STRING. An identifier for the Campaign.

Description
One or more. ML_STRING. A description of the Campaign.

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

Campaign MUST have at least one instance of a Campaign or Description.

The Campaign class has two attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.9. 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. 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.

```
+------------------------+
| AdditionalData         |
+------------------------+
| ANY                    |
|                        |
| ENUM dtype             |
| STRING ext-dtype       |
| STRING meaning         |
| STRING formatid        |
| ENUM restriction       |
| STRING ext-restriction |
+------------------------+
```

Figure 9: The AdditionalData Class

The AdditionalData class has six 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". These values are maintained in the "AdditionalData-dtype" IANA registry per Table 1.

1. boolean. The element content is of type BOOLEAN.

2. byte. The element content is of type BYTE.

3. bytes. The element content is of type HEXBIN.

4. character. The element content is of type CHARACTER.

5. date-time. The element content is of type DATETIME.

6. ntpstamp. Same as date-time.

7. integer. The element content is of type INTEGER.

8. portlist. The element content is of type PORTLIST.

9. real. The element content is of type REAL.

10. string. The element content is of type STRING.

11. file. The element content is a base64 encoded binary file encoded as a BYTE[] type.

12. path. The element content is a file-system path encoded as a STRING type.

13. frame. The element content is a layer-2 frame encoded as a HEXBIN type.

14. packet. The element content is a layer-3 packet encoded as a HEXBIN type.

15. ipv4-packet. The element content is an IPv4 packet encoded as a HEXBIN type.

16. ipv6-packet. The element content is an IPv6 packet encoded as a HEXBIN type.

17. url. The element content is of type URL.

18. csv. The element content is a common separated value (CSV) list per Section 2 of [RFC4180] encoded as a STRING type.

19. winreg. The element content is a Windows registry key encoded as a STRING type.

20. xml. The element content is XML. See Section 5.

21. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-dtype
Optional. STRING. A means by which to extend the dtype attribute. See Section 5.1.1.

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. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.10. 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.

```
+------------------------+
| Contact                |
+------------------------+
| ENUM role              |<>--{0..*}--[ ContactName    ]
| STRING ext-role        |<>--{0..*}--[ ContactTitle   ]
| ENUM type              |<>--{0..*}--[ Description    ]
| STRING ext-type        |<>--{0..*}--[ RegistryHandle ]
| ENUM restriction       |<>--{0..1}--[ PostalAddress  ]
| STRING ext-restriction |<>--{0..*}--[ Email          ]
|                        |<>--{0..*}--[ Telephone      ]
|                        |<>--{0..1}--[ Fax            ]
|                        |<>--{0..1}--[ Timezone       ]
|                        |<>--{0..*}--[ Contact        ]
|                        |<>--{0..*}--[ AdditionalData ]
+------------------------+
```

Figure 10: The Contact Class

The aggregate classes that constitute the Contact class are:

ContactName
Zero or more. ML_STRING. The name of the contact. The contact may either be an organization or a person. The type attribute disambiguates the semantics.

ContactTitle
Zero or more. ML_STRING. The title for the individual named in the ContactName.

Description
Zero or more. 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 more. A handle name into the registry of the contact.

PostalAddress
Zero or one. The postal address of the contact.

Email
Zero or more. The email address of the contact.

Telephone
Zero or more. 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 more. 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 more. 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 six attributes:

role
Required. ENUM. Indicates the role the contact fulfills. This attribute is defined as an enumerated list. These values are maintained in the "Contact-role" IANA registry per Table 1.

1. creator. The entity that generate the document.

2. reporter. The entity that reported the information.

3. admin. An administrative contact or business owner for an asset or organization.

4. tech. An entity responsible for the day-to-day management of technical issues for an asset or organization.

5. provider. An external hosting provider for an asset.

6. zone. An entity with authority over a DNS zone.

7. user. An end-user of an asset or part of an organization.

8. billing. An entity responsible for billing issues for an asset or organization.

9. legal. An entity responsible for legal issue related to an asset or organization.

10. irt. An entity responsible for handling security issues for an asset or organization.

11. abuse. An entity responsible for handling abuse originating from an asset or organization.

12. cc. An entity that is to be kept informed about the events related to an asset or organization.

13. cc-irt. A CSIRT or information sharing organization coordinating activity related to an asset or organization.

14. leo. A law enforcement organization supporting the investigation of activity affecting an asset or organization.

15. vendor. The vendor that produces an asset.

16. vendor-support. A vendor that provides services.

17. victim. A victim in the incident.

18. victim-notified. A victim in the incident who has been notified.

19. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-role
Optional. STRING. A means by which to extend the role attribute. See Section 5.1.1.

type
Required. ENUM. Indicates the type of contact being described. This attribute is defined as an enumerated list. These values are maintained in the "Contact-type" IANA registry per Table 1.

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 5.1.1.

ext-type
Optional. STRING. A means by which to extend the type attribute. See Section 5.1.1.

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.10.1. 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.

```
+---------------------+
| RegistryHandle      |
+---------------------+
| STRING              |
|                     |
| ENUM registry       |
| STRING ext-registry |
+---------------------+
```

Figure 11: The RegistryHandle Class

The RegistryHandle class has two attributes:

registry
Required. ENUM. The database to which the handle belongs. These values are maintained in the "RegistryHandle-registry" IANA registry per Table 1. 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 5.1.1.

ext-registry
Optional. STRING. A means by which to extend the registry attribute. See Section 5.1.1.

3.10.2. PostalAddress Class

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

```
+---------------------+
| PostalAddress       |
+---------------------+
| POSTAL              |
|                     |
| STRING meaning      |
| ENUM xml:lang       |
+---------------------+
```

Figure 12: The PostalAddress Class

The PostalAddress class has two attributes:

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

xml:lang
Optional. ENUM. A language identifier per Section 2.12 of [W3C.XML] whose values and form are described in [RFC5646]. The interpretation of this code is described in Section 6.

3.10.3. Email Class

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

```
+--------------+
| Email        |
+--------------+
| EMAIL        |
|              |
| ENUM meaning |
+--------------+
```

Figure 13: The Email Class

The Email class has one attribute:

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

3.10.4. 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).

```
+--------------------+
| {Telephone | Fax } |
+--------------------+
| PHONE              |
|                    |
| ENUM meaning       |
+--------------------+
```

Figure 14: 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).

3.11. Time Classes

The data model uses six 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).

```
+-----------------+
| StartTime       |
| EndTime         |
| ReportTime      |
| DetectTime      |
| GenerationTime  |
| DateTime        |
+-----------------+
| DATETIME        |
+-----------------+
```

Figure 15: The Time Classes

3.11.1. StartTime Class

The StartTime class represents the time the incident began.

3.11.2. EndTime Class

The EndTime class represents the time the incident ended.

3.11.3. DetectTime Class

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

3.11.4. ReportTime Class

The ReportTime class represents the time the incident was reported.

3.11.5. GenerationTime Class

The GenerationTime class represents the time when the IODEF document was produced. This timestamp MUST be the time at which the IODEF document was generated.

3.11.6. DateTime

The DateTime class is a generic representation of a timestamp. Infer its semantics from the parent class in which it is aggregated.

3.12. Discovery Class

The Discovery class describes how an incident was detected.

```
+------------------------+
| Discovery              |
+------------------------+
| ENUM source            |<>--{0..*}--[ Description      ]
| STRING ext-source      |<>--{0..*}--[ Contact          ]
| ENUM restriction       |<>--{0..*}--[ DetectionPattern ]
| STRING ext-restriction |
+------------------------+
```

Figure 16: The Discovery Class

The Discovery class is composed of three aggregate classes.

Description
: Zero or more. ML_STRING. A free-form text description of how this incident was detected.

Contact
Zero or more. Contact information for the party that discovered the incident.

DetectionPattern
Zero or more. Describes an application-specific configuration that detected the incident.

The Discovery class has four attribute:

source
Optional. ENUM. Categorizes the techniques used to discover the incident. These values are partially derived from Table 3-1 of [NIST800.61rev2]. These values are maintained in the "Discovery-source" IANA registry per Table 1.

1. nidps. Network Intrusion Detection or Prevention system.

2. hips. Host-based Intrusion Prevention system.

3. siem. Security Information and Event Management System.

4. av. Antivirus or and antispam software.

5. third-party-monitoring. Contracted third-party monitoring service.

6. incident. The activity was discovered while investigating an unrelated incident.

7. os-log. Operating system logs.

8. application-log. Application logs.

9. device-log. Network device logs.

10. network-flow. Network flow analysis.

11. passive-dns. Passive DNS analysis.

12. investigation. Manual investigation initiated based on notification of a new vulnerability or exploit.

13. audit. Security audit.

14. internal-notification. A party within the organization reported the activity

15. external-notification. A party outside of the organization reported the activity.

16. leo. A law enforcement organization notified the victim organization.

17. partner. A customer or business partner reported the activity to the victim organization.

18. actor. The threat actor directly or indirectly reported this activity to the victim organization.

19. unknown. Unknown detection approach.

20. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-source
Optional. STRING. A means by which to extend the source attribute. See Section 5.1.1.

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.12.1. DetectionPattern Class

The DetectionPattern class describes a configuration or signature that can be used by an IDS/IPS, SIEM, anti-virus, end-point protection, network analysis, malware analysis, or host forensics tool to identify a particular phenomenon. This class requires the identification of the target application and allows the configuration to be describes in either free-form or machine readable form.

```
+-------------------------+
| DetectionPattern        |
+-------------------------+
| ENUM restriction        |<>----------[ Application             ]
| STRING ext-restriction |<>--{0..*}--[ Description             ]
|                         |<>--{0..*}--[ DetectionConfiguration ]
+-------------------------+
```

Figure 17: The DetectionPattern Class

The DetectionPattern class is composed of three aggregate classes.

Application
One. The application for which the DetectionConfiguration or Description is being provided.

Description
Zero or more. ML_STRING. A free-form text description of how to use the Application or provided DetectionConfiguration.

DetectionConfiguration
Zero or more. STRING. A machine consumable configuration to find a pattern of activity.

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

The DetectionPattern class has two attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.13. Method Class

The Method class describes the tactics, techniques, procedures or underlying issue used by the intruder in the incident. This class consists of both a list of references describing the attack methods and weaknesses and a free form description.

```
+------------------------+
| Method                 |
+------------------------+
| ENUM restriction       |<>--{0..*}--[ Reference         ]
| STRING ext-restriction |<>--{0..*}--[ Description       ]
|                        |<>--{0..*}--[ sci:AttackPattern ]
|                        |<>--{0..*}--[ sci:Vulnerability ]
|                        |<>--{0..*}--[ sci:Weakness      ]
|                        |<>--{0..*}--[ AdditionalData    ]
+------------------------+
```

Figure 18: The Method Class

The Method class is composed of six aggregate classes.

enum:Reference

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

Description
Zero or more. ML_STRING. A free-form text description of techniques, tactics, or procedures used by the intruder.

sci:AttackPattern
Zero or more. A reference to an pattern of attack or exploitation per [RFC-SCI]

sci:Vulnerability
Zero or more. A reference to a vulnerability per [RFC-SCI]

sci:Weakness
Zero or more. A reference to the exploited weakness per [RFC-SCI]

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

An instance of one of these child MUST be present.

The Method class has two attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.13.1. Reference Class

The Reference class is an external reference to relevant information such 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.

```
+------------------------+
| Reference              |
+------------------------+
| ID observable-id       |<>--{0..1}--[ enum:ReferenceName ]
|                        |<>--{0..*}--[ URL                ]
|                        |<>--{0..*}--[ Description        ]
+------------------------+
```

Figure 19: The Reference Class

The aggregate classes that constitute Reference:

ReferenceName
Zero or one. Reference identifier per [RFC-ENUM].

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

Description
Zero or more. ML_STRING. A free-form text description of this reference.

At least one of these classes MUST be present.

The Reference class has one attribute.

observable-id
Optional. ID. See Section 3.3.2.

3.14. Assessment Class

The Assessment class describes the repercussions of the incident to the victim.

```
+------------------------+
| Assessment             |
+------------------------+
| ENUM occurrence        |<>--{0..*}--[ IncidentCategory ]
| ENUM restriction       |<>--{0..*}--[ SystemImpact     ]
| STRING ext-restriction |<>--{0..*}--[ BusinessImpact   ]
| ID observable-id       |<>--{0..*}--[ TimeImpact       ]
|                        |<>--{0..*}--[ MonetaryImpact   ]
|                        |<>--{0..*}--[ IntendedImpact   ]
|                        |<>--{0..*}--[ Counter          ]
|                        |<>--{0..*}--[ MitigatingFactor ]
|                        |<>--{0..*}--[ Cause            ]
|                        |<>--{0..1}--[ Confidence       ]
|                        |<>--{0..*}--[ AdditionalData   ]
+------------------------+
```

Figure 20: Assessment Class

The aggregate classes that constitute Assessment are:

IncidentCategory
Zero or more. ML_STRING. A free-form text description categorizing the type of Incident.

SystemImpact
Zero or more. Technical characterization of the impact of the activity on the victim’s enterprise.

BusinessImpact
Zero or more. Impact of the activity on the business functions of the victim organization.

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

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

IntendedImpact
Zero or more. Intended impact to the victim by the attacker. Identically defined as Section 3.14.2 but describes intent rather than the realized impact.

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

MitigatingFactor
Zero or more. ML_STRING. A description of a mitigating factor an impact.

Cause
Zero or more. ML_STRING. A description of the underlying cause of the impact.

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

AdditionalData
Zero or more. 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. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

3.14.1. SystemImpact Class

The SystemImpact class describes the technical impact of the incident to the systems on the network.

This class is based on [RFC4765].

```
+-----------------------+
| SystemImpact          |
+-----------------------+
| ML_STRING             |
|                       |
| ENUM xml:lang         |
| STRING translation-id |
| ENUM severity         |
| ENUM completion       |
| ENUM type             |
| STRING ext-type       |
+-----------------------+
```

Figure 21: SystemImpact Class

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

The SystemImpact class has six attributes:

xml:lang
Optional. ENUM. A language identifier. See Section 6.

translation-id

Optional. STRING. An identifier to relate other instances of this class as translations of this text. See 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 impact. The permitted values are shown below. The default value is "unknown". These values are maintained in the "SystemImpact-type" IANA registry per Table 1.

1. takeover-account. Control was taken of a given account (e.g., a social media account).

2. takeover-service. Control was taken of a given service.

3. takeover-system. Control was taken of a given system.

4. cps-manipulation. A cyber physical system was manipulated.

5. cps-damage. A cyber physical system was damaged.

6. availability-data. Access to particular data was degraded or denied.

7. availability-account. Access to an account was degraded or denied.

8. availability-service. Access to a service was degraded or denied.

9. availability-system. Access to a system was degraded or denied.

10. damaged-system. Hardware on a system was irreparably damaged.

11. damaged-data. Data on a system was deleted.

12. breach-proprietary. Sensitive or proprietary information was accessed or exfiltrated.

13. breach-privacy. Personally identifiable information was accessed or exfiltrated.

14. breach-credential. Credential information was accessed or exfiltrated.

15. breach-configuration. System configuration or data inventory was access or exfiltrated.

16. integrity-data. Data on the system was modified.

17. integrity-configuration. Application or system configuration was modified.

18. integrity-hardware. Firmware of a hardware component was modified.

19. traffic-redirection. Network traffic on the system was redirected

20. monitoring-traffic. Network traffic emerging from a host was monitored.

21. monitoring-host. System activity (e.g., running processes, keystrokes) were monitored.

22. policy. Activity violated the system owner’s acceptable use policy.

23. unknown. The impact is unknown.

24. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-type
Optional. STRING. A means by which to extend the type attribute. See Section 5.1.1.

3.14.2. BusinessImpact Class

The BusinessImpact class describes and characterizes the degree to which the function of the organization was impacted by the Incident.

The element body describes the impact to the organization as a free-form text string. The two attributes characterize the impact.

```
+------------------------+
| BusinessImpact         |
+------------------------+
| ML_STRING              |
|                        |
| ENUM xml:lang          |
| STRING translation-id  |
| ENUM severity          |
| STRING ext-severity    |
| ENUM type              |
| STRING ext-type        |
+------------------------+
```

Figure 22: BusinessImpact Class

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

The BusinessImpact class has four attributes:

xml:lang
Optional. ENUM. A language identifier. See Section 6.

translation-id
Optional. STRING. An identifier to relate other instances of this class as translations of this text. See Section 6.

severity
Optional. ENUM. Characterizes the severity of the incident on business functions. The permitted values are shown below. They were derived from Table 3-2 of [NIST800.61rev2]. The default value is "unknown". These values are maintained in the "BusinessImpact-severity" IANA registry per Table 1.

1. none. No effect to the organization’s ability to provide all services to all users.

2. low. Minimal effect as the organization can still provide all critical services to all users but has lost efficiency.

3. medium. The organization has lost the ability to provide a critical service to a subset of system users.

4. high. The organization is no longer able to provide some critical services to any users.

5. unknown. The impact is not known.

6. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-severity
Optional. STRING. A means by which to extend the severity attribute. See Section 5.1.1.

type
Required. ENUM. Characterizes the effect this incident had on the business. The permitted values are shown below. There is no default value. These values are maintained in the "BusinessImpact-type" IANA registry per Table 1.

1. breach-proprietary. Sensitive or proprietary information was accessed or exfiltrated.

2. breach-privacy. Personally identifiable information was accessed or exfiltrated.

3. breach-credential. Credential information was accessed or exfiltrated.

4. loss-of-integrity. Sensitive or proprietary information was changed or deleted.

5. loss-of-service. Service delivery was disrupted.

6. theft-financial. Money was stolen.

7. theft-service. Services were misappropriated.

8. degraded-reputation. The reputation of the organization’s brand was diminished.

9. asset-damage. A cyber-physical system was damaged.

10. asset-manipulation. A cyber-physical system was manipulated.

11. legal. The incident resulted in legal or regulatory action.

12. extortion. The incident resulted in actors extorting the victim organization.

13. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-type
Optional. STRING. A means by which to extend the type attribute. See Section 5.1.1.

3.14.3. 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.

```
+---------------------+
| TimeImpact          |
+---------------------+
| REAL                |
|                     |
| ENUM severity       |
| ENUM metric         |
| STRING ext-metrics  |
| ENUM duration       |
| STRING ext-duration |
+---------------------+
```

Figure 23: 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. These values are maintained in the "TimeImpact-metric" IANA registry per Table 1.

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 5.1.1.

ext-metric
Optional. STRING. A means by which to extend the metric attribute. See Section 5.1.1.

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". These values are maintained in the "TimeImpact-duration" IANA registry per Table 1.

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 5.1.1.

ext-duration

Optional. STRING. A means by which to extend the duration attribute. See Section 5.1.1.

3.14.4. 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.

```
+------------------+
| MonetaryImpact   |
+------------------+
| REAL             |
|                  |
| ENUM severity    |
| STRING currency  |
+------------------+
```

Figure 24: 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 "Codes for the representation of currencies and funds" of [ISO4217]. There is no default value.

3.14.5. Confidence Class

The Confidence class represents a best estimate of the validity and accuracy of the described impact (see Section 3.14) of the incident activity. This estimate can be expressed as a category or a numeric calculation.

This class if based upon [RFC4765].

```
+------------------+
| Confidence       |
+------------------+
| REAL             |
|                  |
| ENUM rating      |
+------------------+
```

Figure 25: 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 MUST 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.

3.15. 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.

```
+------------------------+
| History                |
+------------------------+
| ENUM restriction       |<>--{1..*}--[ HistoryItem ]
| STRING ext-restriction |
+------------------------+
```

Figure 26: 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 two attributes:

restriction
: Optional. ENUM. See Section 3.3.1. The default value is "default".

ext-restriction
: Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.15.1. HistoryItem Class

The HistoryItem class is an entry in the History (Section 3.15) 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       ]
| STRING ext-restriction  |<>--{0..1}--[ IncidentId     ]
| ENUM action             |<>--{0..1}--[ Contact        ]
| STRING ext-action       |<>--{0..*}--[ Description    ]
| ID observable-id        |<>--{0..*}--[ DefinedCOA     ]
|                         |<>--{0..*}--[ AdditionalData ]
+-------------------------+
```

Figure 27: 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 more. ML_STRING. A free-form textual description of the action or event.

DefinedCOA
Zero or more. ML_STRING. A unique identifier meaningful to the sender and recipient of this document that references a course of action. This class MUST be present if the action attribute is set to "defined-coa".

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

The HistoryItem class has five attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

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 action 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 3.17.

ext-action
Optional. STRING. A means by which to extend the action attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

3.16. 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.

```
+-------------------------+
| EventData               |
+-------------------------+
| ENUM restriction        |<>--{0..*}--[ Description    ]
| STRING ext-restriction  |<>--{0..1}--[ DetectTime     ]
| ID observable-id        |<>--{0..1}--[ StartTime      ]
|                         |<>--{0..1}--[ EndTime        ]
|                         |<>--{0..1}--[ RecoveryTime   ]
|                         |<>--{0..1}--[ ReportTime     ]
|                         |<>--{0..*}--[ Contact        ]
|                         |<>--{0..*}--[ Discovery      ]
|                         |<>--{0..1}--[ Assessment     ]
|                         |<>--{0..*}--[ Method         ]
|                         |<>--{0..*}--[ Flow           ]
|                         |<>--{0..*}--[ Expectation    ]
|                         |<>--{0..1}--[ Record         ]
|                         |<>--{0..*}--[ EventData      ]
|                         |<>--{0..*}--[ AdditionalData ]
+-------------------------+
```

Figure 28: 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.

RecoveryTime
Zero or one. The time the site recovered from the event.

ReportTime
One. The time the event was reported.

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

Discovery
Zero or more. The means by which the event was detected.

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 three attributes:

restriction
Optional. ENUM. See Section 3.3.1. The default value is "default".

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

3.16.1. 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 Incident.

3.16.2. Cardinality of EventData

The EventData class is 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 are grouped around these common properties.

The recursive definition of the EventData class (the EventData class is aggregated into the EventData class) provides a way to relate 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 29.

```
+------------------+
| EventData        |
+------------------+
|                  |<>----[ Contact    ]
|                  |
|                  |<>----[ EventData  ]<>----[ Flow       ]
|                  |      [            ]<>----[ Assessment ]
|                  |
|                  |<>----[ EventData  ]<>----[ Flow       ]
|                  |      [            ]<>----[ Assessment ]
+------------------+
```

Figure 29: Recursion in the EventData Class

3.17. 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.

```
+-------------------------+
| Expectation             |
+-------------------------+
| ENUM restriction        |<>--{0..*}--[ Description ]
| STRING ext-restriction  |<>--{0..*}--[ DefinedCOA  ]
| ENUM severity           |<>--{0..1}--[ StartTime   ]
| ENUM action             |<>--{0..1}--[ EndTime     ]
| STRING ext-action       |<>--{0..1}--[ Contact     ]
| ID observable-id        |
|                         |
+-------------------------+
```

Figure 30: The Expectation Class

The aggregate classes that constitute Expectation are:

Description
   Zero or more. ML_STRING. A free-form description of the desired action(s).

DefinedCOA
Zero or more. ML_STRING. A unique identifier meaningful to the sender and recipient of this document that references a course of action. This class MUST be present if the action attribute is set to "defined-coa".

StartTime
Zero or one. The time at which the sender would like the action performed. A timestamp that is earlier than the ReportTime specified in the Incident class denotes that the sender would like the action performed as soon as possible. The absence of this element indicates no expectations of when the recipient would like the action performed.

EndTime
Zero or one. The time by which the sender expects the recipient to complete the action. If the recipient cannot complete the action before EndTime, the recipient MUST NOT carry out the action. Because of transit delays, clock drift, and so on, the sender MUST be prepared for the recipient to have carried out the action, even if it completes past EndTime.

Contact
Zero or one. The expected actor for the action.

The Expectations class has six attributes:

restriction
Optional. ENUM. See Section 3.3.1. The default value is "default".

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

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 dependent.

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". These values are maintained in the "Expectation-action" IANA registry per Table 1.

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. redirect-traffic. Redirect traffic from intended recipient for further analysis.

13. honeypot. Redirect traffic to a honeypot for further analysis.

14. upgrade-software. Upgrade or patch the software or firmware on an asset.

15. rebuild-asset. Reinstall the operating system or applications on an asset.

16. harden-asset. Change the configuration an asset (e.g., reduce the number of services or user accounts) to reduce the attack surface.

17. remediate-other. Remediate the activity in a way other than by rate limiting or blocking.

18. status-triage. Conveys receipts and the triaging of an incident.

19. status-new-info. Conveys that new information was received for this incident.

20. watch-and-report. Watch for the described activity and share if seen.

21. training. Train user to identify or mitigate a threat.

22. defined-coa. Perform a predefined course of action (COA). The COA is named in the DefinedCOA class.

23. other. Perform some custom action described in the Description class.

24. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-action
Optional. STRING. A means by which to extend the action attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

3.18. Flow Class

The Flow class groups related the source and target hosts.

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

Figure 31: The Flow Class

The aggregate class that constitutes Flow is:

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

The Flow class has no attributes.

3.19. 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            ]
| STRING ext-restriction |<>--{0..*}--[ NodeRole        ]
| ENUM category         |<>--{0..*}--[ Service         ]
| STRING ext-category   |<>--{0..*}--[ OperatingSystem ]
| STRING interface      |<>--{0..*}--[ Counter         ]
| ENUM spoofed          |<>--{0..*}--[ AssetID         ]
| ENUM virtual          |<>--{0..*}--[ Description     ]
| ENUM ownership        |<>--{0..*}--[ AdditionalData  ]
| STRING ext-ownership  |
|                       |
+-----------------------+
```

Figure 32: The System Class

The aggregate classes that constitute System are:

Node
One. A host or network involved in the incident.

NodeRole
Zero or more. The intended purpose of the system.

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.

AssetID
Zero or more. An asset identifier for the System.

Description
Zero or more. ML_STRING. A free-form text description of the System.

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

The System class has nine attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

category
Optional. ENUM. Classifies the role the host or network played in the incident. These values are maintained in the "System-category" IANA registry per Table 1. 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. intermediate. The System was an intermediary in the event.

4. sensor. The System was a sensor monitoring the event.

5. infrastructure. The System was an infrastructure node of IODEF document exchange.

6. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-category
Optional. STRING. A means by which to extend the category attribute. See Section 5.1.1.

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.

virtual
Optional. ENUM. Indicates whether this System is a virtual or physical device. The default value is "unknown". The possible values are:

1. yes. The System is a virtual device.

2. no. The System is a physical device.

3. unknown. It is not known if the System is virtual.

ownership
Optional. ENUM. Describes the ownership of this System relative to the sender of the IODEF document. These values are maintained in the "System-ownership" IANA registry per Table 1. The possible values are:

1. organization. The System is owned by the organization.

2. personal. The System is owned by employee or affiliate of the organization.

3. partner. The System is owned by a partner of the organization.

4. customer. The System is owned by a customer of the organization.

5. no-relationship. The System is owned by an entity that has no known relationship with the organization.

6. unknown. The ownership of the System is unknown.

7. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-ownership
Optional. STRING. A means by which to extend the ownership attribute. See Section 5.1.1.

3.20. Node Class

The Node class names an asset or network.

This class was derived from [RFC4765].

```
+---------------+
| Node          |
+---------------+
|               |<>--{0..*}--[ DomainData    ]
|               |<>--{0..*}--[ Address       ]
|               |<>--{0..1}--[ PostalAddress ]
|               |<>--{0..*}--[ Location      ]
|               |<>--{0..1}--[ DateTime      ]
|               |<>--{0..*}--[ Counter       ]
+---------------+
```

Figure 33: The Node Class

The aggregate classes that constitute Node are:

DomainData
Zero or more. The detailed domain (DNS) information associated with this Node. If an Address is not provided, at least one DomainData MUST be specified.

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

PostalAddress
Zero or one. The postal address of the asset.

Location

Zero or more. ML_STRING. A free-form description of the physical location of the Node. This description may provide a more detailed description of where in the PostalAddress this Node is found (e.g., room number, rack number, slot number in a chassis).

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

The Node class has no attributes.

3.20.1. Address Class

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

This class was derived from [RFC4765].

```
+------------------------+
| Address                |
+------------------------+
| ENUM category          |
| STRING ext-category    |
| STRING vlan-name       |
| INTEGER vlan-num       |
| ID observable-id       |
+------------------------+
```

Figure 34: 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". These values are maintained in the "Address-category" IANA registry per Table 1.

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 (i.e., a.b.c.d/nn)

6. ipv4-net-mask. IPv4 network address in dotted-decimal notation, slash, network mask in dotted-decimal notation (i.e., 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 (i.e., a:b:c:d:e:f)

11. site-uri. A URL or URI for a resource.

12. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-category
Optional. STRING. A means by which to extend the category attribute. See Section 5.1.1.

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.

observable-id
Optional. ID. See Section 3.3.2.

3.20.2. NodeRole Class

The NodeRole class describes the function performed by a particular .

```
+---------------------+
| NodeRole            |
+---------------------+
| ENUM category       |
| STRING ext-category |
| ENUM xml:lang       |
+---------------------+
```

Figure 35: The NodeRole Class

The NodeRole class has three attributes:

category
Required. ENUM. Functionality provided by a node. These values are maintained in the "NodeRole-category" IANA registry per Table 1.

1. client. Client computer

2. client-enterprise. Client computer on the enterprise network

3. client-partner. Client computer on network of a partner

4. client-remote. Client computer remotely connected to the enterprise network

5. client-kiosk. Client computer is serves as a kiosk

6. client-mobile. Client is a mobile device

7. server-internal. Server with internal services

8. server-public. Server with public services

9. www. WWW server

10. mail. Mail server

11. webmail. Web mail server

12. messaging. Messaging server (e.g., NNTP, IRC, IM)

13. streaming. Streaming-media server

14. voice. Voice server (e.g., SIP, H.323)

15. file. File server (e.g., SMB, CVS, AFS)

16. ftp. FTP server

17. p2p. Peer-to-peer node

18. name. Name server (e.g., DNS, WINS)

19. directory. Directory server (e.g., LDAP, finger, whois)

20. credential. Credential server (e.g., domain controller, Kerberos)

21. print. Print server

22. application. Application server

23. database. Database server

24. backup. Backup server

25. dhcp. DHCP server

26. assessment. Assessment server (e.g., vulnerability scanner, end-point assessment)

27. source-control. Source code control server

28. config-management. Configuration management server

29. monitoring. Security monitoring server (e.g., IDS)

30. infra. Infrastructure server (e.g., router, firewall, DHCP)

31. infra-firewall. Firewall

32. infra-router. Router

33. infra-switch. Switch

34. camera. Camera and video system

35. proxy. Proxy server

36. remote-access. Remote access server

37. log. Log server (e.g., syslog)

38. virtualization. Server running virtual machines

39. pos. Point-of-sale device

40. scada. Supervisory control and data acquisition system

41. scada-supervisory. Supervisory system for a SCADA

42. sinkhole. Traffic sinkhole destination

43. honeypot. Honeypot server

44. anonymization. Anonymization server (e.g., Tor node)

45. c2. Malicious command and control server

46. malware-distribution. Server that distributes malware

47. drop-server. Server to which exfiltrated content is uploaded.

48. hop-point. Intermediary server used to get to a victim.

49. reflector. A system used in a reflector attacker.

50. phishing-site. Site hosting phishing content

51. spear-phishing-site. Site hosting spear-phishing content

52. recruiting-site. Site to recruit

53. fraudulent-site. Fraudulent site.

54. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-category
Optional. STRING. A means by which to extend the category attribute. See Section 5.1.1.

xml:lang
Optional. ENUM. A language identifier per Section 2.12 of [W3C.XML] whose values and form are described in [RFC5646]. The interpretation of this code is described in Section 6.

3.20.3. 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 dependent based on the class in which the Counter is aggregated.

```
+---------------------+
| Counter             |
+---------------------+
| REAL                |
|                     |
| ENUM type           |
| STRING ext-type     |
| ENUM unit           |
| STRING ext-unit     |
| STRING meaning      |
| ENUM duration       |
| STRING ext-duration |
+---------------------+
```

Figure 36: The Counter Class

The Counter class has seven attribute:

type
Required. ENUM. Specifies the type of counter specified in the element content. These values are maintained in the "Counter-type" IANA registry per Table 1.

1. count. The Counter class value is a counter.

2. peak. The Counter class value is a peak value.

3. average. The Counter class value is an average.

4. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-type
Optional. STRING. A means by which to extend the type attribute. See Section 5.1.1.

unit
Required. ENUM. Specifies the units of the element content. These values are maintained in the "Counter-unit" IANA registry per Table 1.

1. byte. Bytes transferred.

2. mbit. Megabits (Mbits) transfered.

3. packet. Packets.

4. flow. Network flow records.

5. session. Sessions.

6. alert. Notifications generated by another system (e.g., IDS or SIM).

7. message. Messages (e.g., mail messages).

8. event. Events.

9. host. Hosts.

10. site. Site.

11. organization. Organizations.

12. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-unit
Optional. STRING. A means by which to extend the unit attribute. See Section 5.1.1.

meaning
Optional. STRING. A free-form description of the metric represented by the Counter.

duration
Optional. ENUM. If present, the Counter class represents a rate. This attribute specifies unit of time over which the rate whose units are specified in the unit attribute is being conveyed. This attribute is the the denominator of the rate (where the unit attribute specified the nominator). The possible values of this attribute are defined in Section 3.14.3

ext-duration
Optional. STRING. A means by which to extend the duration attribute. See Section 5.1.1.

3.21. DomainData Class

The DomainData class describes a domain name and meta-data associated with this domain.

```
+-------------------------+
| DomainData              |
+-------------------------+
| ENUM system-status      |<>----------[ Name                 ]
| STRING ext-system-status |<>--{0..1}--[ DateDomainWasChecked ]
| ENUM domain-status      |<>--{0..1}--[ RegistrationDate     ]
| STRING ext-domain-status |<>--{0..1}--[ ExpirationDate       ]
| ID observable-id        |<>--{0..*}--[ RelatedDNS           ]
|                         |<>--{0..*}--[ Nameservers          ]
|                         |<>--{0..1}--[ DomainContacts       ]
|                         |
+-------------------------+
```

Figure 37: The DomainData Class

The aggregate classes that constitute DomainData are:

Name
One. STRING. The domain name of the Node (e.g., fully qualified domain name).

DateDomainWasChecked
Zero or one. DATETIME. A timestamp of when the Name was resolved.

RegistrationDate
Zero or one. DATETIME. A timestamp of when domain listed in Name was registered.

ExpirationDate
Zero or one. DATETIME. A timestamp of when the domain listed in Name is set to expire.

RelatedDNS
Zero or more. Additional DNS records associated with this domain.

Nameservers
Zero or more. The name servers identified for the domain listed in Name.

DomainContacts
Zero or one. Contact information for the domain listed in Name supplied by the registrar or through a whois query.

The DomainData class has five attribute:

system-status

Required. ENUM. Assesses the domain’s involvement in the event. These values are maintained in the "DomainData-system-status" IANA registry per Table 1.

1. spoofed. This domain was spoofed.

2. fraudulent. This domain was operated with fraudulent intentions.

3. innocent-hacked. This domain was compromised by a third party.

4. innocent-hijacked. This domain was deliberately hijacked.

5. unknown. No categorization for this domain known.

6. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-system-status
Optional. STRING. A means by which to extend the system-status attribute. See Section 5.1.1.

domain-status
Required. ENUM. Categorizes the registry status of the domain at the time the document was generated. These values and their associated descriptions are derived from Section 3.2.2 of [RFC3982]. These values are maintained in the "DomainData-domain-status" IANA registry per Table 1.

1. reservedDelegation. The domain is permanently inactive.

2. assignedAndActive. The domain is in a normal state.

3. assignedAndInactive. The domain has an assigned registration but the delegation is inactive.

4. assignedAndOnHold. The domain is under dispute.

5. revoked. The domain is in the process of being purged from the database.

6. transferPending. The domain is pending a change in authority.

7. registryLock. The domain is on hold by the registry.

8. registrarLock. Same as "registryLock".

9. other. The domain has a known status but it is not one of the redefined enumerated values.

10. unknown. The domain has an unknown status.

11. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-domain-status
Optional. STRING. A means by which to extend the domain-status attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

3.21.1. RelatedDNS

The RelatedDNS class describes additional record types associated with a given domain name. The record type is described in the record-type attribute and the value of the record is the element content. ... TODO Issue #39 ...

```
+----------------------+
| RelatedDNS           |
+----------------------+
| STRING               |
|                      |
| ENUM record-type     |
+----------------------+
```

Figure 38: The RelatedDNS Class

The RelatedDNS class has one attribute:

record-type
Required. ENUM. The DNS record type. ... TODO values need to be listed ...

3.21.2. Nameservers Class

The Nameservers class describes the name servers associated with a given domain.

```
+--------------------+
| Nameservers        |
+--------------------+
|                    |<>----------[ Server  ]
|                    |<>--{1..*}--[ Address ]
+--------------------+
```

Figure 39: The Nameservers Class

The aggregate classes that constitute Nameservers are:

Server
One. STRING. The domain name of the name server.

Address
One or more. The address of the name server. See Section 3.20.1.

3.21.3. DomainContacts Class

The DomainContacts class describes the contact information for a given domain provided either by the registrar or through a whois query.

This contact information can be explicitly described through a Contact class or a reference can be provided to a domain with identical contact information. Either a single SameDomainContact MUST be present or one or many Contact classes.

```
+--------------------+
| DomainContacts     |
+--------------------+
|                    |<>--{0..1}--[ SameDomainContact ]
|                    |<>--{1..*}--[ Contact ]
+--------------------+
```

Figure 40: The DomainContacts Class

The aggregate classes that constitute DomainContacts are:

SameDomainContact
Zero or one. STRING. A domain name already cited in this document or through previous exchange that contains the identical contact information as the domain name in question. The domain contact information associated with this domain should be used instead of an explicit definition with the Contact class.

Contact

One or more. Contact information for the domain. See Section 3.10.

3.22. 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 [RFC4765].

```
+-------------------------+
| Service                 |
+-------------------------+
| INTEGER ip-protocol     |<>--{0..1}--[ ServiceName       ]
| ID observable-id        |<>--{0..1}--[ Port              ]
|                         |<>--{0..1}--[ Portlist          ]
|                         |<>--{0..1}--[ ProtoCode         ]
|                         |<>--{0..1}--[ ProtoType         ]
|                         |<>--{0..1}--[ ProtoField        ]
|                         |<>--{0..*}--[ ApplicationHeader ]
|                         |<>--{0..1}--[ EmailData         ]
|                         |<>--{0..1}--[ Application       ]
+-------------------------+
```

Figure 41: The Service Class

The aggregate classes that constitute Service are:

ServiceName
Zero or one. Identifies the the observed service.

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 transport layer (layer 4) protocol-specific code field (e.g., ICMP code field).

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

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

ApplicationHeader
Zero or more. An application layer (layer 7) protocol header. See Section 3.22.2.

EmailData
Zero or one. Headers associated with an email. See Section 3.24.

Application
Zero or one. The application bound to the specified Port or Portlist. See Section 3.22.3.

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

When a given System classes with category="source" and another with category="target" are aggregated into a single Flow class, and each of these System classes has a Service and Portlist class, an implicit relationship between these Portlists exists. If N ports are listed for a System@category="source", and M ports are listed for System@category="target", the number of ports in N must be equal to M. Likewise, the ports MUST be listed in an identical sequence such that the n-th port in the source corresponds to the n-th port of the target. If N is greater than 1, a given instance of a Flow class MUST only have a single instance of a System@category="source" and System@category="target".

The Service class has two attributes:

ip-protocol
Required. INTEGER. The IANA assigned IP protocol number per [IANA.Protocols].

observable-id
Optional. ID. See Section 3.3.2.

3.22.1. ServiceName Class

The ServiceName class names an application protocol. It can be described by referencing an IANA registered protocol, a URL or with free-form text.

```
+--------------------+
| ServiceName        |
+--------------------+
|                    |<>--{0..1}--[ IANAService       ]
|                    |<>--{0..*}--[ URL               ]
|                    |<>--{0..*}--[ Description       ]
+--------------------+
```

Figure 42: The ServiceName Class

The aggregate classes that constitute ServiceName:

IANAService
Zero or one. The name of the service per the "Service Name" field of the [IANA.Ports] registry.

URL
Zero or more. URL. A URL describing the service.

Description
Zero or more. ML_STRING. A free-form text description of the service.

At least one of these classes MUST be present.

The ServiceName class has no attributes.

3.22.2. ApplicationHeader Class

The ApplicationHeader class allows the representation of arbitrary fields from an application layer protocol header and its corresponding value.

```
+--------------------------+
| ApplicationHeader        |
+--------------------------+
| ANY                      |
|                          |
| INTEGER proto            |
| STRING proto-name        |
| STRING field             |
| ENUM dtype               |
| STRING ext-dtype         |
| ID observable-id         |
+--------------------------+
```

Figure 43: The ApplicationHeader Class

The ApplicationHeader class has six attributes:

proto
Optional. INTEGER. The IANA assigned port number per the "Protocol Number" field of the [IANA.Ports] registry corresponding to the application layer protocol whose field will be represented.

proto-name
Optional. STRING. The IANA assigned service name per the "Service Name" field of the the [IANA.Ports] registry corresponding to the application layer protocol whose field will be represented.

field
Required. STRING. The name of the protocol field whose value will be found in the element body.

dtype
Required. ENUM. The data type of the element content. The permitted values for this attribute are shown below. The default value is "string". These values are maintained in the "ApplicationHeader-proto-dtype" IANA registry per Table 1.

1. boolean. The element content is of type BOOLEAN.

2. byte. The element content is of type BYTE.

3. bytes. The element content is of type HEXBIN.

4. character. The element content is of type CHARACTER.

5. date-time. The element content is of type DATETIME.

6. integer. The element content is of type INTEGER.

7. portlist. The element content is of type PORTLIST.

8. real. The element content is of type REAL.

9. string. The element content is of type STRING.

10. file. The element content is a base64 encoded binary file encoded as a BYTE[] type.

11. path. The element content is a file-system path encoded as a STRING type.

12. xml. The element content is XML. See Section 5.

13. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-dtype
Optional. STRING. A means by which to extend the dtype attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

Either the proto or proto-name attribute MUST be set. If both are set, they MUST correspond to the same entry in the registry.

3.22.3. Application Class

The Application class describes a software application. It can be described by using formal reference, a URL or with free-form text.

```
+--------------------+
| Application        |
+--------------------+
|                    |<>--{0..1}--[ SoftwareReference ]
|                    |<>--{0..*}--[ URL               ]
|                    |<>--{0..*}--[ Description       ]
+--------------------+
```

Figure 44: The Application Class

The aggregate classes that constitute Application:

SoftwareReference
Zero or one. Reference to a software application.

URL
Zero or more. URL. A URL associated with the application.

Description
Zero or more. ML_STRING. A free-form text description of this application.

At least one of these classes MUST be present.

The Application class has no attributes.

3.22.4. SoftwareReference Class

The Application class describes a software application. It can be described by using formal reference, a URL or with free-form text.

```
+---------------------+
| SoftwareReference   |
+---------------------+
| ANY                 |
|                     |
| ENUM spec-name      |
| STRING ext-spec-name |
| ENUM dtype          |
| STRING enum-dtype   |
+---------------------+
```

Figure 45: The SoftwareReference Class

The element body of this class varies according to the value of the spec-name attribute.

The SoftwareReference class has four attributes:

spec-name
Required. ENUM. Identifies the format and semantics of the element body of this class. Formal standards and specifications can be referenced as well as free-form description with user-provided data-types. These values are maintained in the "SoftwareReference-spec-id" IANA registry per Table 1

1. custom. The element content is free-form and of the data type specified by the dtype attribute. If this value is selected, then the dtype attribute MUST be set.

2. cpe. The element content describes a Common Platform Enumeration (CPE) entry [fix me. reference].

3. swid. The element content describes a software identification (SWID) tag per ISO/IEC 19770-2:2009 [fix me. reference].

4. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-spec-name
Optional. STRING. A means by which to extend the spec-name attribute. See Section 5.1.1.

dtype

Optional. ENUM. The data type of the element content. The permitted values for this attribute are shown below. The default value is "string". These values are maintained in the "SoftwareReference-dtype" IANA registry per Table 1.

1. bytes. The element content is of type HEXBIN.

2. integer. The element content is of type INTEGER.

3. real. The element content is of type REAL.

4. string. The element content is of type STRING.

5. xml. The element content is XML. See Section 5.

6. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-dtype
Optional. STRING. A means by which to extend the dtype attribute. See Section 5.1.1.

3.23. OperatingSystem Class

The OperatingSystem class describes the operating system running on a System. The definition is identical to the Application class (Section 3.22.3).

3.24. EmailData Class

The EmailData class describes headers from an email message. Common headers have dedicated classes, but arbitrary headers can also be described.

```
+--------------------------+
| EmailData                |
+--------------------------+
| ID observable-id         |<>--{0..1}--[ EmailFrom        ]
|                          |<>--{0..1}--[ EmailSubject     ]
|                          |<>--{0..1}--[ EmailX-Mailer    ]
|                          |<>--{0..*}--[ EmailHeaderField ]
|                          |<>--{0..*}--[ HashData         ]
|                          |<>--{0..*}--[ SignatureData    ]
+--------------------------+
```

Figure 46: EmailData Class

The aggregate class that constitutes EmailData are:

EmailFrom
Zero or one. The value of the "From:" header field in an email. See Section 3.6.2 of [RFC5322].

EmailSubject
Zero or one. The value of the "Subject:" header field in an email. See Section 3.6.4 of [RFC5322].

EmailX-Mailer
Zero or one. The value of the "X-Mailer:" header field in an email.

EmailHeaderField
Zero or one. The value of an arbitrary header field in the email. See Section 3.22.2. The attributes of EmailHeaderField MUST be set as follows: proto="25" or proto-name="smtp", or both can be set; and dtype="string". The name of the email header field MUST be set in the field attribute.

HashData
Zero or One. Hash(es) associated with this email.

SignatureData
Zero or One. Signature(s) associated with this email.

The EmailData class has one attribute:

observable-id
Optional. ID. See Section 3.3.2.

3.25. 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 substantiate the activity described in the document.

```
+------------------------+
| Record                 |
+------------------------+
| ENUM restriction       |<>--{1..*}--[ RecordData ]
| STRING ext-restriction |
+------------------------+
```

Figure 47: 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 two attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.25.1. 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               ]
| STRING ext-restriction |<>--{0..*}--[ Description            ]
| ID observable-id       |<>--{0..1}--[ Application            ]
|                        |<>--{0..*}--[ RecordPattern          ]
|                        |<>--{0..*}--[ RecordItem             ]
|                        |<>--{0..*}--[ FileData               ]
|                        |<>--{0..*}--[ CertificateData        ]
|                        |<>--{0..*}--
|                        |       [ WindowsRegistryKeysModified ]
|                        |<>--{0..*}--[ AdditionalData         ]
+------------------------+
```

Figure 48: 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
Zero or more. Log, audit, or forensic data.

FileData
Zero or one. 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 attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

3.25.2. 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.

```
+-----------------------+
| RecordPattern         |
+-----------------------+
| STRING                |
|                       |
| ENUM type             |
| STRING ext-type       |
| INTEGER offset        |
| ENUM offsetunit       |
| STRING ext-offsetunit |
| INTEGER instance      |
+-----------------------+
```

Figure 49: 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 six attributes:

type
Required. ENUM. Describes the type of pattern being specified in the element content. The default is "regex". These values are maintained in the "RecordPattern-type" IANA registry per Table 1.

1. regex. regular expression as defined by POSIX Extended Regular Expressions (ERE) in Chapter 9 of [IEEE.POSIX].

2. binary. Binhex encoded binary pattern, per the HEXBIN data type.

3. xpath. XML Path (XPath) [W3C.XPATH]

4. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-type
Optional. STRING. A means by which to extend the type attribute. See Section 5.1.1.

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". These values are maintained in the "RecordPattern-offsetunit" IANA registry per Table 1.

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 5.1.1.

ext-offsetunit
Optional. STRING. A means by which to extend the offsetunit attribute. See Section 5.1.1.

instance
Optional. INTEGER. Number of types to apply the specified pattern.

3.25.3. 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 3.9).

3.26. WindowsRegistryKeysModified Class

The WindowsRegistryKeysModified class describes Windows operating system registry keys and the operations that were performed on them. This class was derived from [RFC5901].

```
+----------------------------+
| WindowsRegistryKeysModified |
+----------------------------+
| ID observable-id           |<>--{1..*}--[ Key ]
+----------------------------+
```

Figure 50: The WindowsRegistryKeysModified Class

The aggregate class that constitutes the WindowsRegistryKeysModified class is:

Key
One or many. The Window registry key.

The WindowsRegistryKeysModified class has one attribute:

observable-id

Optional. ID. See Section 3.3.2.

3.26.1. Key Class

The Key class describes a particular Windows operating system registry key name and value pair, and the operation performed on it.

```
+---------------------------+
| Key                       |
+---------------------------+
| ENUM registryaction       |<>----------[ KeyName ]
| STRING ext-registryaction |<>--{0..1}--[ KeyValue ]
| ID observable-id          |
+---------------------------+
```

Figure 51: The Key Class

The aggregate classes that constitutes Key are:

KeyName
One. STRING. The name of the Windows operating system registry key (e.g.,[HKEY_LOCAL_MACHINE\Software\Test\KeyName])

KeyValue
Zero or one. STRING. The value of the associated registry key encoded as in Microsoft .reg files [KB310516].

The Key class has three attributes:

registryaction
Optional. ENUM. The type of action taken on the registry key. These values are maintained in the "Key-registryaction" IANA registry per Table 1.

1. add-key. Registry key added.

2. add-value. Value added to registry key.

3. delete-key. Registry key deleted.

4. delete-value. Value deleted from registry key.

5. modify-key. Registry key modified.

6. modify-value. Value modified for registry key.

7. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-registryaction
Optional. STRING. A means by which to extend the registryaction attribute. See Section 5.1.1.

observable-id
Optional. ID. See Section 3.3.2.

3.27. CertificateData Class

The CertificateData class describes X.509 certificates.

```
+------------------------+
| CertificateData        |
+------------------------+
| ID observable-id       |<>--{1..*}--[ Certificate    ]
| ENUM restriction       |
| STRING ext-restriction |
+------------------------+
```

Figure 52: The CertificateData Class

The aggregate classes that constitutes CertificateData are:

Certificate
One or more. A certificate.

The CertificateData class has three attributes:

observable-id
Optional. ID. See Section 3.3.2.

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.27.1. Certificate Class

The Certificate class describes a given X.509 certificate or certificate chain.

```
+-------------------------+
| Certificate             |
+-------------------------+
| ID observable-id        |<>----------[ ds: X509Data   ]
|                         |<>--{0..*}--[ Description    ]
+-------------------------+
```

Figure 53: The Certificate Class

The aggregate classes that constitutes Certificate are:

ds:X509Data
One. A given X.509 certificate or chain. See Section 4.4.4 of [W3C.XMLSIG].

Description
Zero or more. ML_STRING. Free-form textual description explaining the context of this certificate.

The Certificate class has one attribute:

observable-id
Optional. ID. See Section 3.3.2.

3.28. FileData Class

The FileData class describes files of interest identified during the analysis of an incident.

```
+-----------------------+
| FileData              |
+-----------------------+
| ID observable-id      |<>--{1..*}--[ File      ]
| ENUM restriction      |
| STRING ext-restriction |
+-----------------------+
```

Figure 54: The FileData Class

The aggregate class that constitutes FileData is:

File
One or more. A description of a file.

The FileData class has three attributes:

observable-id
Optional. ID. See Section 3.3.2.

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction
Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.28.1. File Class

The File class describes a file and its associated meta data.

```
+-----------------------+
| File                  |
+-----------------------+
| ID observable-id      |<>--{0..1}--[ FileName            ]
|                       |<>--{0..1}--[ FileSize            ]
|                       |<>--{0..1}--[ FileType            ]
|                       |<>--{0..*}--[ URL                 ]
|                       |<>--{0..1}--[ HashData            ]
|                       |<>--{0..1}--[ SignatureData       ]
|                       |<>--{0..1}--[ AssociatedSoftware ]
|                       |<>--{0..*}--[ FileProperties      ]
+-----------------------+
```

Figure 55: The File Class

The aggregate classes that constitutes File are:

FileName
Zero or One. STRING. The name of the file.

FileSize
Zero or One. INTEGER. The size of the file in bytes.

FileType
Zero or One. STRING. The type of file per the IANA Media Types Registry [IANA.Media]. Valid values correspond to the text in the "Template" column (e.g., "application/pdf").

URL
Zero or more. A URL reference to the file.

HashData
Zero or One. Hash(es) associated with this file.

SignatureData
Zero or One. Signature(s) associated with this file.

AssociatedSoftware
Zero or One. The software application or operating system to which this file belongs. See Section 3.22.3 for the definition.

FileProperties
Zero or more. Mechanism by which to extend the data model to describe properties of the file. See Section 3.9.

The File class has one attribute:

observable-id
Optional. ID. See Section 3.3.2.

3.29. HashData Class

The HashData class describes different types of hashes on an given object (e.g., file, part of a file, email).

```
+--------------------------+
| HashData                 |
+--------------------------+
| ENUM scope               |<>--{0..1}--[ HashTarget   ]
|                          |<>--{0..*}--[ Hash         ]
|                          |<>--{0..*}--[ FuzzyHash    ]
+--------------------------+
```

Figure 56: The HashData Class

The aggregate classes that constitutes HashData are:

HashTarget
Zero or One. An identifier that references a a subset of the object per the @scope attribute.

Hash
Zero or more. The hash generated on the object.

FuzzyHash
Zero or more. The fuzzy hash of the object.

A single instance of Hash or FuzzyHash MUST be present.

The HashData class has one attribute:

scope
Required. ENUM. Describes the scope of the hash on a type of object. These values are maintained in the "HashData-scope" IANA registry per Table 1.

1. file-contents. A hash computed over the entire contents of a file.

2. file-pe-section. A hash computed on a given section of a Windows Portable Executable (PE) file. If set to this value, the HashTargetId class MUST identify the section being hashed. This section is identified by an ordinal number (starting at 1) corresponding to the the order in which the given section header was defined in the Section Table of the PE file header.

3. file-pe-iat. A hash computed on the Import Address Table (IAT) of a PE file. As IAT hashes are often tool dependent, if this value is set, the HashTargetId class MUST specify the tool used to generate the hash.

4. file-pe-resource. A hash computed on a given resource in a PE file. If set to this value, the HashTargetId class MUST identify the resource being hashed. This resource is identified by an ordinal number (starting at 1) corresponding to the oder in which the given resource is declared in the Resource Directory of the Data Dictionary in the PE file header.

5. file-pdf-object. A hash computed on a given object in a Portable Document Format (PDF) file. If set to this value, the HashTargetId class MUST identify the object being hashed. This object is identified by its offset in the PDF file.

6. email-hash. A hash computed over the headers and body of an email message.

7. email-headers-hash. A hash computed over all of the headers of an email message.

8. email-body-hash. A hash computed over the body of an email message.

9. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-scope
Optional. STRING. A means by which to extend the scope attribute. See Section 5.1.1.

3.29.1. Hash Class

The Hash class describes a specific hash value, algorithm, and an application used to generate it.

```
+----------------+
| Hash           |
+----------------+
|                |<>----------[ ds:DigestMethod            ]
|                |<>----------[ ds:DigestValue             ]
|                |<>--{0..1}--[ ds:CannonicalizationMethod ]
|                |<>--{0..1}--[ Application                ]
+----------------+
```

Figure 57: The Hash Class

The aggregate classes that constitutes Hash are:

ds:DigestMethod
One. The hash algorithm used to generate the hash. See Section 4.3.3.5 of [W3C.XMLSIG]

ds:DigestValue
One. The computed hash value. See Section 4.3.3.6 of [W3C.XMLSIG].

ds:CannonicalizationMethod
Zero or one. The canonicalization method used for the has. See Section 4.3.1 of [W3C.XMLSIG].

Application
Zero or One. The application used to calculate the hash.

The HashData class has no attribute:

3.29.2. FuzzyHash Class

The FuzzyHash class describes a fuzzy hash (in an extensible way) and the application used to generate it.

```
+--------------------------+
| FuzzyHash                |
+--------------------------+
|                          |<>--{0..*}--[ AdditionalData ]
|                          |<>--{0..1}--[ Application    ]
+--------------------------+
```

Figure 58: The FuzzyHash Class

The aggregate classes that constitutes FuzzyHash are:

AdditionalData
: Zero or more. Mechanism by which to extend the data model. See Section 3.9.

Application
: Zero or One. The application used to calculate the hash.

The FuzzyData class has no attribute:

3.30. SignatureData Class

The SignatureData class describes different signatures on an given object.

```
+-------------------------+
| SignatureData           |
+-------------------------+
|                         |<>--{1..*}--[ ds:Signature ]
+-------------------------+
```

Figure 59: The SignatureData Class

The aggregate classes that constitutes SignatureData are:

Signature
: One or more. An given signature. See Section 4.2 of [W3C.XMLSIG]

The SignatureData class has no attribute:

3.31. IndicatorData Class

The IndicatorData class describes the indicators identified from analysis of an incident.

```
+-------------------------+
| IndicatorData           |
+-------------------------+
|                         |<>--{1..*}--[ Indicator      ]
+-------------------------+
```

Figure 60: The IndicatorData Class

The aggregate class that constitutes IndicatorData is:

Indicator
: One or more. An indicator from the incident.

The IndicatorData class has no attributes.

3.32. Indicator Class

The Indicator class describes a cyber indicator. An indicator consists of observable features and phenomenon that aid in the forensic or proactive detection of malicious activity, and associated meta-data. This indicator can be described outright or reference observable features and phenomenon described elsewhere in the incident information. Portions of an incident description can be composed to define an indicator, as can the indicators themselves.

```
+------------------------+
| Indicator              |
+------------------------+
| ENUM restriction       |<>----------[ IndicatorID            ]
| STRING ext-restriction |<>--{0..1}--[ AlternativeIndicatorID ]
|                        |<>--{0..*}--[ Description            ]
|                        |<>--{0..1}--[ StartTime              ]
|                        |<>--{0..1}--[ EndTime                ]
|                        |<>--{0..1}--[ Confidence             ]
|                        |<>--{0..*}--[ Contact                ]
|                        |<>--{0..1}--[ Observable             ]
|                        |<>--{0..1}--[ ObservableReference    ]
|                        |<>--{0..1}--[ IndicatorExpression    ]
|                        |<>--{0..1}--[ IndicatorReference     ]
|                        |<>--{0..*}--[ AdditionalData         ]
+------------------------+
```

Figure 61: The Indicator Class

The aggregate classes that constitute Indicator are:

IndicatorID
One. An identifier for this indicator. See Section 3.32.1

AlternativeIndicatorID
Zero or one. An alternative identifier for this indicator. See Section 3.32.2

Description
Zero or more. ML_STRING. A free-form textual description of the indicator.

StartTime
Zero or one. DATETIME. A timestamp of the start of the time period during which this indicator is valid.

EndTime
Zero or one. DATETIME. A timestamp of the end of the time period during which this indicator is valid.

Confidence
Zero or one. An estimate of the confidence in the quality of the indicator. See Section 3.14.5.

Contact
Zero or more. Contact information for this indicator. See Section 3.10.

Observable
Zero or one. An observable feature or phenomenon of this indicator. See Section 3.32.3.

ObservableReference
Zero or one. A reference to a feature or phenomenon defined elsewhere in the document. See Section 3.32.6.

IndicatorExpression
Zero or one. A composition of observables. See Section 3.32.4.

IndicatorReference
Zero or one. A reference to an indicator.

AdditionalData
Zero or more. Mechanism by which to extend the data model. See Section 3.9

The Indicator class MUST have exactly one instance of an Observable, IndicatorExpression, ObservableReference, or IndicatorReference class.

The StartTime and EndTime classes can be used to define an interval during which the indicator is valid. If both classes are present, the indicator is consider valid only during the described interval. If neither class is provided, the indicator is considered valid during any time interval. If only a StartTime is provided, the indicator is valid anytime after this timestamp. If only an EndTime is provided, the indicator is valid anytime prior to this timestamp.

The Indicator class has two attributes:

restriction
Optional. ENUM. See Section 3.3.1.

ext-restriction

Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.32.1. IndicatorID Class

The IndicatorID class identifies an indicator with a globally unique identifier. The combination of the name and version attributes, and the element content form this identifier. Indicators generated by given CSIRT MUST NOT reuse the same value unless they are referencing the same indicator.

```
+------------------+
| IndicatorID      |
+------------------+
| ID               |
|                  |
| STRING name      |
| STRING version   |
+------------------+
```

Figure 62: The IndicatorID Class

The IndicatorID class has two attributes:

name
: Required. STRING. An identifier describing the CSIRT that created the indicator. In order to have a globally unique CSIRT name, the fully qualified domain name associated with the CSIRT MUST be used. This format is identical to the IncidentID@name attribute in Section 3.4.

version
: Required. STRING. A version number of an indicator.

3.32.2. AlternativeIndicatorID Class

The AlternativeIndicatorID class lists alternative identifiers for an indicator.

```
+-------------------------+
| AlternativeIndicatorID  |
+-------------------------+
| ENUM restriction        |<>--{1..*}--[ IndicatorReference ]
| STRING ext-restriction  |
+-------------------------+
```

Figure 63: The AlternativeIndicatorID Class

The aggregate class that constitutes AlternativeIndicatorID is:

IndicatorReference
   One or more. A reference to an indicator.

The AlternativeIndicatorID class has two attributes:

restriction
   Optional. ENUM. See Section 3.3.1.

ext-restriction
   Optional. STRING. A means by which to extend the restriction attribute. See Section 5.1.1.

3.32.3. Observable Class

The Observable class describes a feature and phenomenon that can be observed or measured for the purposes of detecting malicious behavior.

```
+------------------+
| Observable       |
+------------------+
|                  |<>--{0..1}--[ Address                     ]
|                  |<>--{0..1}--[ DomainData                  ]
|                  |<>--{0..1}--[ Service                     ]
|                  |<>--{0..1}--[ EmailData                   ]
|                  |<>--{0..1}--[ ApplicationHeader           ]
|                  |<>--{0..1}--[ WindowsRegistryKeysModified ]
|                  |<>--{0..1}--[ FileData                    ]
|                  |<>--{0..1}--[ CertificateData             ]
|                  |<>--{0..1]--[ RegistryHandle              ]
|                  |<>--{0..1}--[ RecordData                  ]
|                  |<>--{0..1}--[ EventData                   ]
|                  |<>--{0..1}--[ Incident                    ]
|                  |<>--{0..*}--[ Expectation                 ]
|                  |<>--{0..*}--[ Reference                   ]
|                  |<>--{0..1}--[ Assessment                  ]
|                  |<>--{0..1}--[ HistoryItem                 ]
|                  |<>--{0..1}--[ BulkObservable              ]
|                  |<>--{0..*}--[ AdditionalData              ]
+------------------+
```

Figure 64: The Observable Class

The aggregate classes that constitute Observable are:

Address

Zero or One. An Address observable. See Section 3.20.1.

DomainData
Zero or One. A DomainData observable. See Section 3.21.

Service
Zero or One. A Service observable. See Section 3.22.

EmailData
Zero or One. A EmailData observable. See Section 3.24.

ApplicationHeader
Zero or One. An ApplicationHeader observable. See Section 3.22.2.

WindowsRegistryKeysModified
Zero or One. A WindowsRegistryKeysModified observable. See Section 3.26.

FileData
Zero or One. A FileData observable. See Section 3.28.

CertificateData
Zero or One. A CertificateData observable. See Section 3.27.

RegistryHandle
Zero or One. A RegistryHandle observable. See Section 3.10.1.

RecordData
Zero or One. A RecordData observable. See Section 3.25.1.

EventData
Zero or One. An EventData observable. See Section 3.16.

Incident
Zero or One. An Incident observable. See Section 3.2.

EventData
Zero or One. An EventData observable. See Section 3.16.

Expectation
Zero or One. An Expectation observable. See Section 3.17.

Reference
Zero or One. A Reference observable. See Section 3.13.1.

Assessment
Zero or One. An Assessment observable. See Section 3.14.

HistoryItem
Zero or One. A HistoryItem observable. See Section 3.15.1.

BulkObservable
Zero or One. A bulk list of observables. See Section 3.32.3.1.

AdditionalData
Zero or more. Mechanism by which to extend the data model. See Section 3.9.

The Observable class MUST have exactly one of the possible child classes.

The Observable class has no attributes.

3.32.3.1. BulkObservable Class

The BulkObservable class allows the bulk enumeration of single type of observables without requiring each one to be encoded individually in multiple instances of the same class. The type attribute describes the type observable listed in the child BulkObservableList class. The BulkObservableFormat class optionally provides additional meta-data.

```
+---------------------------+
| BulkObservable            |
+---------------------------+
| ENUM type                 |<>--{0..1}--[ BulkObservableFormat ]
| STRING ext-type           |<>----------[ BulkObservableList   ]
|                           |<>--{0..*}--[ AdditionalData       ]
+---------------------------+
```

Figure 65: The BulkObservable Class

The aggregate classes that constitutes BulkObservable are:

BulkObservableFormat
Zero or one. Provides additional meta-data about the observables enumerated in the BulkObservableList class.

BulkObservableList
One. STRING. A list of observables, one per line. Each line is separated with either a LF character or CR-and-LF characters. The type attribute will specify the which observables will be listed.

AdditionalData
Zero or more. Mechanism by which to extend the data model. See Section 3.9.

The BulkObservable class has two attributes:

type
Optional. ENUM. The type of the observable listed in the child ObservableList class. These values are maintained in the "BulkObservable-type" IANA registry per Table 1.

1. asn. Autonomous System Number (per the Address@category attribute).

2. atm. Asynchronous Transfer Mode (ATM) address (per the Address@category attribute).

3. e-mail. Electronic mail address (RFC 822) (per the Address@category attribute).

4. ipv4-addr. IPv4 host address in dotted-decimal notation (e.g., 192.0.2.1) (per the Address@category attribute).

5. ipv4-net. IPv4 network address in dotted-decimal notation, slash, significant bits (e.g., 192.0.2.0/24) (per the Address@category attribute).

6. ipv4-net-mask. IPv4 network address in dotted-decimal notation, slash, network mask in dotted-decimal notation (i.e., 192.0.2.0/255.255.255.0) (per the Address@category attribute).

7. ipv6-addr. IPv6 host address (e.g., 2001:DB8::3) (per the Address@category attribute).

8. ipv6-net. IPv6 network address, slash, significant bits (e.g., 2001:DB8::/32) (per the Address@category attribute).

9. ipv6-net-mask. IPv6 network address, slash, network mask (per the Address@category attribute).

10. mac. Media Access Control (MAC) address (i.e., a:b:c:d:e:f) (per the Address@category attribute).

11. site-uri. A URL or URI for a resource (per the Address@category attribute).

12. fqdn. Fully qualified domain name.

13. domain-name. A fully qualified domain name or part of a name. (e.g., fqdn.example.com, example.com).

14. domain-to-ipv4. A fqdn-to-IPv4 address mapping specified as a comma separated list (e.g., "fqdn.example.com, 192.0.2.1").

15. domain-to-ipv6. A fqdn-to-IPv6 address mapping specified as a comma separated list (e.g., "fqdn.example.com, 2001:DB8::3").

16. domain-to-ipv4-timestamp. Same as domain-to-ipv4 but with a timestamp (in the DATETIME format) of the resolution (e.g., "fqdn.example.com, 192.0.2.1, 2015-06-11T00:38:31-06:00").

17. domain-to-ipv6-timestamp. Same as domain-to-ipv6 but with a timestamp (in the DATETIME format) of the resolution (e.g., "fqdn.example.com, 2001:DB8::3, 2015-06-11T00:38:31-06:00").

18. ipv4-port. An IPv4 address, port and protocol tuple (e.g., 192.0.2.1, 80, tcp). The protocol name corresponds to the "Keyword" column in the [IANA.Protocols] registry.

19. ipv6-port. An IPv6 address, port and protocol tuple (e.g., 2001:DB8::3, 80, tcp). The protocol name corresponds to the "Keyword" column in the [IANA.Protocols] registry.

20. windows-reg-key. A Microsoft Windows Registry key.

21. file-hash. A file hash. The format of this hash is described in the Hashclass that MUST be present in a sibling BulkObservableFormat class.

22. email-x-mailer. An X-Mailer field from an email.

23. email-subject. An email subject line.

24. http-user-agent. A User Agent field from an HTTP request header (e.g., "Mozilla/5.0 (Windows NT 6.3; WOW64; rv:38.0) Gecko/20100101 Firefox/38.0").

25. http-request-uri. The Request URI from an HTTP request header.

26. mutex. The name of a system mutex.

27. file-path. A file path (e.g., "/tmp/local/file", "c:\windows\system32\file.sys")

28. user-name. A username.

29. ext-value. An escape value used to extend this attribute. See Section 5.1.1.

ext-type
Optional. STRING. A means by which to extend the type attribute. See Section 5.1.1.

3.32.3.1.1. BulkObservableFormat Class

The ObservableFormat class specifies meta-data about the format of an observable enumerated in a sibling BulkObservableList class.

```
+--------------------------+
| BulkObservableFormat     |
+--------------------------+
|                          |<>--{0..1}--[ Hash             ]
|                          |<>--{0..*}--[ AdditionalData   ]
+--------------------------+
```

Figure 66: The BulkObservableFormat Class

The aggregate classes that constitutes BulkObservableFormat are:

Hash
Zero or one. Describes the format of a hash.

AdditionalData
Zero or more. Mechanism by which to extend the data model. See Section 3.9.

The BulkObservableFormat class has no attributes.

Either Hash or AdditionalData MUST be present.

3.32.4. IndicatorExpression Class

The IndicatorExpression describes an expression composed of observed phenomenon or features, or indicators. Elements of the expression can be described directly, reference relevant data from other parts of a given IODEF document, or reference previously defined indicators.

All child classes of a given instance of IndicatorExpression form a boolean algebraic expression where the operator between them is determined by the operator attribute.

```
+-------------------------+
| IndicatorExpression     |
+-------------------------+
| ENUM operator           |<>--{0..*}--[ IndicatorExpression  ]
|                         |<>--{0..*}--[ Observable           ]
|                         |<>--{0..*}--[ ObservableReference  ]
|                         |<>--{0..*}--[ IndicatorReference   ]
|                         |<>--{0..*}--[ AdditionalData       ]
+-------------------------+
```

Figure 67: The IndicatorExpression Class

The aggregate classes that constitute IndicatorExpression are:

IndicatorExpression
Zero or more. An expression composed of other observables or indicators.

Observable
Zero or more. A description of an observable.

ObservableReference
Zero or more. A reference to another observable.

IndicatorReference
Zero or more. A reference to another indicator.

AdditionalData
Zero or more. Mechanism by which to extend the data model. See Section 3.9

The IndicatorExpression class has one attribute:

operator
Optional. ENUM. The operator to be applied between the child elements. The default value is "and". These values are maintained in the "IndicatorExpression-operator" IANA registry per Table 1.

1. not. negation operator.

2. and. conjunction operator.

3. or. disjunction operator.

4. xor. exclusive disjunction operator.

3.32.5. Expressions with IndicatorExpression

Boolean algebraic expressions can be used specify relationships between observables and indicator. These expressions are constructed through the use of the operator attribute and parent-child relationships in IndicatorExpressions. These expressions should be parsed as follows:

1. The operator specified by the operator attribute is applied between each of the child elements of the immediate parent IndicatorExpression element. If no operator attribute is specified, it should be assumed to be an AND.

2. A nested IndicatorExpression element with a parent IndicatorExpression is the equivalent of a parentheses in the expression.

The following four examples illustrate these parsing rules:

```
1     : <IndicatorExpression>
2 [O1]:    <Observable>..</Observable>
3 [O2]:    <Observable>..</Observable>
4     : </IndicatorExpression>
```

Equivalent expression: (O1 AND O2)

Figure 68: Nested elements in an IndicatorExpression without an operator attribute specified

```
1     : <IndicatorExpression operator="or">
2 [O1]:    <Observable>..</Observable>
3 [O2]:    <Observable>..</Observable>
4     : </IndicatorExpression>
```

Equivalent expression: (O1 OR O2)

Figure 69: Nested elements in an IndicatorExpression with an operator attribute specified

```
1     : <IndicatorExpression operator="or">
2     :    <IndicatorExpression operator="or">
2 [O1]:      <Observable>..</Observable>
3 [O2]:      <Observable>..</Observable>
4     :    </IndicatorExpression>
2 [O3]:    <Observable>..</Observable>
4     : </IndicatorExpression>

Equivalent expression: ((O1 OR O2) OR O3)
```

Figure 70: Nested elements with a recursive IndicatorExpression with an operator attribute specified

```
1     : <IndicatorExpression operator="not">
2     :    <IndicatorExpression operator="and">
2 [O1]:      <Observable>..</Observable>
3 [O2]:      <Observable>..</Observable>
4     :    </IndicatorExpression>
4     : </IndicatorExpression>

Equivalent expression: (NOT (O1 AND O2))
```

Figure 71: A recursive IndicatorExpression with an operator attribute specified

Invalid algebriac expressions while valid XML, MUST not be specified.

3.32.6. ObservableReference Class

The ObservableReference describes a reference to an observable feature or phenomenon described elsewhere in the document.

This class has no content.

```
+------------------------+
| ObservableReference    |
+------------------------+
| EMPTY                  |
|                        |
| IDREF uid-ref          |
+------------------------+
```

Figure 72: The ObservableReference Class

The ObservableReference class has one attribute:

uid-ref

Required. IDREF. An identifier that serves as a reference to a class in the IODEF document. The referenced class will have this identifier set in the observable-id attribute.

3.32.7. IndicatorReference Class

The IndicatorReference describes a reference to an indicator. This reference may be to an indicator described in the IODEF document or in a previously exchanged IODEF document.

```
+--------------------------+
| IndicatorReference       |
+--------------------------+
| EMPTY                    |
|                          |
| IDREF uid-ref            |
| STRING euid-ref          |
| STRING version           |
+--------------------------+
```

Figure 73: The IndicatorReference Class

The IndicatorReference class has one attribute:

uid-ref
Optional. IDREF. An identifier that serves as a reference to an Indicator class in the IODEF document. The referenced Indicator class will have this identifier set in the IndicatorID class.

euid-ref
Optional. STRING. An identifier that references an IndicatorID not in this IODEF document.

version
Optional. STRING. A version number of an indicator.

Either the uid-ref or the euid-ref attribute MUST be set.

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. The character encoding MUST also be explicitly specified. UTF-8 [RFC3629] SHOULD be used unless UTF-16

[RFC2781] is necessary. Encodings other than UTF-8 and UTF-16 SHOULD NOT be used. 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 [RFC2978].

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 "&", "<", ">", """, and "'" respectively.

4.2. IODEF Namespace

The IODEF schema declares a namespace of "urn:ietf:params:xml:ns:iodef-2.0" and registers it per [W3C.XMLNS]. Each IODEF document MUST 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="2.00" lang="en-US"
   xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
   xsi:schemaLocation="urn:ietf:params:xmls:schema:iodef-2.0"
```

4.3. Validation

The IODEF documents MUST be well-formed XML. It is RECOMMENDED that recipients validate the document 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 "xml: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 [RFC5646].

- o The MonetaryImpact@currency attribute is declared as an "xs:string", but the list of valid values as defined in [ISO4217].

- 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 3.22.

- o The enumerated values present in this document are a static list that will be incomplete over time as select attributes can be extended by a corresponded IANA registry. See Table 1. Hence, the schema to validate a given document MUST be dynamically generated from these registry values.

4.4. Incompatibilities with v1

Version 2 of the IODEF data model makes a number of changes to [RFC5070]. Largely, these changes were additive in nature -- classes and enumerated values were added. The following is a list of incompatibilities where the data model has changed between versions:

- The IODEF-Document@version attribute is set to "2.0".

- The Service@ip_protocol attribute was renamed to @ip-protocol.

- The Node/NodeName class was removed in favor of representing domain names with Node/DomainData/Name class. The Node/DataTime class was also removed so that the Node/DomainData/DateDomainWasChecked class can represent the time at which the name to address resolution occurred.

- The Node/NodeRole class was moved to System/NodeRole.

- The Reference class is now defined by [RFC-ENUM].

- Attributes with enumerated values can now also be extended with IANA registries.

- The data previously represented in the Impact class is now in the SystemImpact and IncidentCategory classes. The Impact class has been removed.

- The Description class has been redefined to use xml:lang and @translation-id. IODEF-document also uses xml:lang.

- The semantics of Counter@type in v1 are now represented in Counter@unit.

- The IODEF-Document@formatid attribute has been renamed to @format-id.

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 base 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

Enumerated values of select attributes can be extended for private use through specially marked attributes with the "ext-" prefix. Likewise, each extensible attribute has a corresponding IANA registry to which to added public extensions.

5.1.1. Private Extension of Enumerated Values

The data model supports a means by which to add new enumerated values to an attribute without public registration. For each attribute that supports this extension technique, there is a corresponding attribute in the same element whose name is identical but with 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 enumerated values. This particular value serves as an escape sequence to the implementor to signal that the extension attribute value should be read. Otherwise, this value 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 SystemImpact class would look as follows:

```
<SystemImpact 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".

5.1.2. Public Extension of Enumerated Values

Select enumerated value of the attributes defined in the data model can be extended by adding entries to the corresponding IANA registry. Table 1 enumerates these registries. Section 4.3 discusses the XML Validation implications of these types of extensions.

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-". This recommendation makes readability of the document easier by allowing the reader to infer which namespaces relate to IODEF by inspection.

3. It is RECOMMENDED that extension schemas follow the naming convention of the IODEF data model. This makes reading an extended IODEF document look like any other IODEF document. The names of all elements are capitalized. For elements with composed names, a capital letter is used for each word. Attribute names are lower case. Attributes with composed names are separated by a hyphen.

4. Parsers that encounter an unrecognized element in a namespace that they do support MUST reject the document as a syntax error.

5. There are security and performance implications in requiring implementations to dynamically download schemas at run time. Thus, implementations SHOULD NOT download schemas at runtime, unless implementations take appropriate precautions and are prepared for potentially significant network, processing, and time-out demands.

6. Some users of the IODEF may have private schema definitions that might not be available on the Internet. In this situation, if a IODEF document leaks out of the private use space, references to some of those document schemas may not be resolvable. This has two implications. First, references to private schemas may never resolve. As such, in addition to the suggestion that implementations do not download schemas at runtime mentioned

above, recipients MUST be prepared for a schema definition in an IODEF document never to resolve.

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:schema: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="2.00" 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>
       </Incident>
 </IODEF-Document
```

5.3. Deconflicting Private Extensions

Private extensions used in a document can be labeled to attribute their original specifier using the private-enum-name and private-enum-id attributes. This allows a recipient of a document to disambiguate private extensions. Only a single private extension can be identified in a given IODEF-Document.

If a CSIRT has only a single private extension, then only the private-enum-name attribute needs to be specified. Additional, multiple distinct private extensions or versioning of a single extension can be accomplished by also setting the corresponding private-num-id attribute.

The following XML excerpt demonstrates the specification of a private extension from "example.com" with an identifier of "13".

```
     <IODEF-Document
          version="2.00" lang="en-US"
          private-enum-name="example.com"
          private-enum-id="13"
         ...
 </IODEF-Document>
```

If an unrecognized private extension is encountered in processing, the recipient MAY reject the entire document as a syntax error.

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 and threat information shared. 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 [W3C.XML]) in the top-level element (i.e., IODEF-Document) 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 [RFC5646].

The data model supports multiple translations of free-form text. For classes where free-text is used for descriptive purposes (e.g., classes of the iodef:MLStringType type such as the Description class), the given class always has a one-to-many cardinality to its parent. 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.

Related instances of a given iodef:MLStringType class that are translations of each other are identified by a common identifier set in the translation-id attribute. The example below shows three instances of a Description class expressed in three difference languages. The relationship between these three instances of the Description class is conveyed by the common value of "1" in the translation-id attribute.

```
<IODEF-Document version="2.00" xml:lang="en" ...
  <Incident purpose="reporting">
    ...
    <Description translation-id="1"
                 xml:lang="en">English</Description>
    <Description translation-id="1"
                 xml:lang="de">Englisch</Description>
    <Description translation-id="1"
                 xml:lang="fr">Anglais</Description>
```

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 IODEF documents. These examples do not necessarily represent the only way to encode particular information.

7.1. Minimal Example

A document containing only the mandatory elements and attributes.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Minimum IODEF document -->
<IODEF-Document version="2.00" xml:lang="en"
   xmlns="urn:ietf:params:xml:ns:iodef-2.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation=
"http://www.iana.org/assignments/xmlregistry/schema/
iodef-2.0.xsd">
  <Incident purpose="reporting" restriction="private">
    <IncidentID name="csirt.example.com">492382</IncidentID>
    <GenerationTime>2015-07-18T09:00:00-05:00</GenerationTime>
    <Contact type="organization" role="creator">
      <Email>contact@csirt.example.com</Email>
    </Contact>
    <!-- Add more fields to make the document useful -->
  </Incident>
</IODEF-Document>
```


7.2. Indicators from a Campaign

An example of C2 domains from a given campaign.


```
<?xml version="1.0" encoding="UTF-8"?>
<!-- A list of C2 domains associated with a campaign -->
<IODEF-Document version="2.00" xml:lang="en"
   xmlns="urn:ietf:params:xml:ns:iodef-2.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation=
   "http://www.iana.org/assignments/xml-registry/schema/
    iodef-2.0.xsd">
  <Incident purpose="watch" restriction="green">
    <IncidentID name="csirt.example.com">897923</IncidentID>
      <RelatedActivity>
        <ThreatActor>
          <ThreatActorID>
          TA-12-AGGRESSIVE-BUTTERFLY
          </ThreatActorID>
          <Description>Aggressive Butterfly</Description>
        </ThreatActor>
        <Campaign>
          <CampaignID>C-2015-59405</CampaignID>
          <Description>Orange Giraffe</Description>
        </Campaign>
      </RelatedActivity>
      <GenerationTime>2015-10-02T11:18:00-05:00</GenerationTime>
      <Description>Summarizes the Indicators of Compromise
        for the Orange Giraffe campaign of the Aggressive
```

```
          Butterfly crime gang.
        </Description>
        <Assessment>
          <BusinessImpact type="breach-proprietary"/>
        </Assessment>
        <Contact type="organization" role="creator">
          <ContactName>CSIRT for example.com</ContactName>
          <Email>contact@csirt.example.com</Email>
        </Contact>
        <IndicatorData>
          <Indicator>
            <IndicatorID name="csirt.example.com" version="1">
            G90823490
            </IndicatorID>
            <Description>C2 domains</Description>
            <StartTime>2014-12-02T11:18:00-05:00</StartTime>
            <Observable>
              <BulkObservable type="fqdn">
              <BulkObservableList>
                kj290023j09r34.example.com
                09ijk23jfj0k8.example.net
                klknjwfjiowjefr923.example.org
                oimireik79msd.example.org
              </BulkObservableList>
            </BulkObservable>
          </Observable>
        </Indicator>
      </IndicatorData>
    </Incident>
  </IODEF-Document>
```

7.3. Incident Report

An example of an incident report.

... TODO ...

8. The IODEF Schema

```
<xs:schema targetNamespace="urn:ietf:params:xml:ns:iodef-2.0"
           xmlns="urn:ietf:params:xml:ns:iodef-2.0"
           xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
           xmlns:enum="urn:ietf:params:xml:ns:iodef-enum-1.0"
           xmlns:sci="urn:ietf:params:xml:ns:iodef-sci-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-xmldsig-core-20020212/xmldsig-core-schema.xsd"/>
<xs:import namespace="urn:ietf:params:xml:ns:iodef-enum-1.0"
           schemaLocation="http://www.iana.org/assignments/
xml-registry/schema/iodef-enum-1.0.xsd" />
<xs:import namespace="urn:ietf:params:xml:ns:iodef-sci-1.0"
           schemaLocation="http://www.iana.org/assignments/
xml-registry/schema/iodef-sci-1.0.xsd" />
<xs:import namespace="http://www.w3.org/XML/1998/namespace"
           schemaLocation="http://www.w3c.org/2001/xml.xsd" />
  <xs:annotation>
    <xs:documentation>
       Incident Object Description Exchange Format v2.0, RFC5070bis
    </xs:documentation>
  </xs:annotation>

  <!--
   ==================================================================
   == IODEF-Document class                                         ==
   ==================================================================
  -->
    <xs:element name="IODEF-Document">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="iodef:Incident"
                      maxOccurs="unbounded"/>
          <xs:element ref="iodef:AdditionalData"
                      minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="version"
                      type="xs:string" fixed="2.00"/>
        <xs:attribute ref="xml:lang" />
        <xs:attribute name="format-id"
                      type="xs:string" use="optional"/>
        <xs:attribute name="private-enum-name"
                      type="xs:string" use="optional"/>
        <xs:attribute name="private-enum-id"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>
  <!--
   ==================================================================
   ===  Incident class                                            ===
   ==================================================================
  -->
    <xs:element name="Incident">
```

```
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="iodef:IncidentID"/>
          <xs:element ref="iodef:AlternativeID"
                      minOccurs="0"/>
          <xs:element ref="iodef:RelatedActivity"
                      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:RecoveryTime"
                      minOccurs="0"/>
          <xs:element ref="iodef:ReportTime"
                      minOccurs="0"/>
          <xs:element ref="iodef:GenerationTime"/>
          <xs:element ref="iodef:Description"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Discovery"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Assessment"
                      minOccurs="0" 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:IndicatorData"
                      minOccurs="0" />
          <xs:element ref="iodef:History"
                      minOccurs="0"/>
          <xs:element ref="iodef:AdditionalData"
                      minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="purpose" use="required"
                      type="incident-purpose-type"/>
        <xs:attribute name="ext-purpose"
                     type="xs:string" use="optional"/>
        <xs:attribute name="status" type="incident-status-type"/>
        <xs:attribute name="ext-status"
                     type="xs:string" use="optional"/>
        <xs:attribute ref="xml:lang" />
        <xs:attribute name="restriction"
                      type="iodef:restriction-type"
                      default="private"/>
```

```
        <xs:attribute name="ext-restriction"
                     type="xs:string" use="optional"/>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="incident-purpose-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="traceback"/>
        <xs:enumeration value="mitigation"/>
        <xs:enumeration value="reporting"/>
        <xs:enumeration value="watch" />
        <xs:enumeration value="other"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="incident-status-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="new"/>
        <xs:enumeration value="in-progress"/>
        <xs:enumeration value="forwarded"/>
        <xs:enumeration value="resolved" />
        <xs:enumeration value="future"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

  <!--
   ===================================================================
   ==  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:attribute name="ext-restriction"
                        type="xs:string" use="optional"/>
        </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:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>
  <!--
   =================================================================
   ==  RelatedActivity class                                      ==
   =================================================================
  -->
    <xs:element name="RelatedActivity">
      <xs:complexType>
        <xs:sequence>
          <xs:choice maxOccurs="unbounded">
            <xs:element ref="iodef:IncidentID"
                        maxOccurs="unbounded"/>
            <xs:element ref="iodef:URL"
                        maxOccurs="unbounded"/>
            <xs:element ref="iodef:ThreatActor"
                        maxOccurs="unbounded"/>
            <xs:element ref="iodef:Campaign"
                        maxOccurs="unbounded"/>
          </xs:choice>
          <xs:element ref="iodef:Confidence"
                      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="ext-restriction"
```

```
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

  <!--
   =================================================================
   ==  ThreatActor class                                          ==
   =================================================================
  -->
    <xs:element name="ThreatActor">
      <xs:complexType>
        <xs:sequence>
          <xs:choice>
            <xs:sequence>
              <xs:element ref="iodef:ThreatActorID" />
              <xs:element ref="iodef:Description"
                          minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="iodef:Description"
                        minOccurs="1" maxOccurs="unbounded"/>
          </xs:choice>
          <xs:element ref="iodef:AdditionalData"
                      minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="restriction"
                      type="iodef:restriction-type"/>
        <xs:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:element name="ThreatActorID" type="xs:string"/>

  <!--
   =================================================================
   ==  Campaign class                                             ==
   =================================================================
  -->
    <xs:element name="Campaign">
      <xs:complexType>
        <xs:sequence>
          <xs:choice>
            <xs:sequence>
              <xs:element ref="iodef:CampaignID"/>
              <xs:element ref="iodef:Description"
                        minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="iodef:Description"
```

```
                     minOccurs="1" maxOccurs="unbounded"/>
          </xs:choice>
          <xs:element ref="iodef:AdditionalData"
                      minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="restriction"
                      type="iodef:restriction-type"/>
        <xs:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="CampaignID" type="xs:string"/>

  <!--
   ==================================================================
   ==  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" maxOccurs="unbounded"/>
          <xs:element ref="iodef:ContactTitle"
                      minOccurs="0" maxOccurs="unbounded"/>
          <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"
                      type="contact-role-type"/>
        <xs:attribute name="ext-role"
                      type="xs:string" use="optional"/>
        <xs:attribute name="type" use="required"
                      type="contact-type-type"/>
        <xs:attribute name="ext-type"
                      type="xs:string" use="optional"/>
        <xs:attribute name="restriction"
                      type="iodef:restriction-type"/>
        <xs:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="contact-role-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="creator"/>
        <xs:enumeration value="reporter"/>
        <xs:enumeration value="admin"/>
        <xs:enumeration value="tech"/>
        <xs:enumeration value="provider"/>
        <xs:enumeration value="zone"/>
        <xs:enumeration value="user"/>
        <xs:enumeration value="billing"/>
        <xs:enumeration value="legal"/>
        <xs:enumeration value="abuse"/>
        <xs:enumeration value="irt"/>
        <xs:enumeration value="cc"/>
        <xs:enumeration value="cc-irt"/>
        <xs:enumeration value="leo"/>
        <xs:enumeration value="vendor"/>
        <xs:enumeration value="vendor-services"/>
        <xs:enumeration value="victim"/>
        <xs:enumeration value="victim-notified"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="contact-type-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="person"/>
        <xs:enumeration value="organization"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
```

```
    </xs:simpleType>

    <xs:element name="ContactName"
                type="iodef:MLStringType"/>
    <xs:element name="ContactTitle"
                type="iodef:MLStringType"/>
    <xs:element name="RegistryHandle">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="xs:string">
            <xs:attribute name="registry"
                          type="registryhandle-registry-type"/>
            <xs:attribute name="ext-registry"
                          type="xs:string" use="optional"/>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="registryhandle-registry-type">
      <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: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="RecoveryTime"
                type="xs:dateTime"/>
    <xs:element name="GenerationTime"
                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:attribute name="ext-restriction"
```

```
                      type="xs:string" use="optional"/>
      </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:DefinedCOA"
                      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="ext-restriction"
                      type="xs:string" use="optional"/>
        <xs:attribute name="action"
                      type="iodef:action-type" use="required"/>
        <xs:attribute name="ext-action"
                      type="xs:string" use="optional"/>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="DefinedCOA" type="iodef:MLStringType"/>

  <!--
   ==================================================================
   ==  Expectation class                                           ==
   ==================================================================
  -->
    <xs:element name="Expectation">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="iodef:Description"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:DefinedCOA"
                      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="ext-restriction"
                      type="xs:string" use="optional"/>
        <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"/>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
      </xs:complexType>
    </xs:element>

  <!--
   ===================================================================
   ==  Discovery class                                              ==
   ===================================================================
  -->
    <xs:element name="Discovery">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="iodef:Description"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Contact"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:DetectionPattern"
                      minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="source"
                      type="discovery-source-type"
                      use="optional" default="unknown"/>
        <xs:attribute name="ext-source"
                      type="xs:string" use="optional"/>
        <xs:attribute name="restriction"
                      type="iodef:restriction-type"/>
        <xs:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="discovery-source-type">
      <xs:restriction base="xs:NMTOKEN">
```

```
        <xs:enumeration value="nidps"/>
        <xs:enumeration value="hips"/>
        <xs:enumeration value="siem"/>
        <xs:enumeration value="av"/>
        <xs:enumeration value="third-party-monitoring"/>
        <xs:enumeration value="incident"/>
        <xs:enumeration value="os-log"/>
        <xs:enumeration value="application-log"/>
        <xs:enumeration value="device-log"/>
        <xs:enumeration value="network-flow"/>
        <xs:enumeration value="passive-dns"/>
        <xs:enumeration value="investigation"/>
        <xs:enumeration value="audit"/>
        <xs:enumeration value="internal-notification"/>
        <xs:enumeration value="external-notification"/>
        <xs:enumeration value="leo"/>
        <xs:enumeration value="partner"/>
        <xs:enumeration value="actor"/>
        <xs:enumeration value="unknown"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:element name="DetectionPattern">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="iodef:Application"/>
          <xs:element ref="iodef:Description"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="DetectionConfiguration"
                      type="xs:string"
                      minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="restriction"
                      type="iodef:restriction-type"/>
        <xs:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

  <!--
   ==================================================================
   ==  Method class                                                ==
   ==================================================================
  -->
    <xs:element name="Method">
      <xs:complexType>
        <xs:sequence>
```

```
          <xs:element ref="iodef:Reference"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Description"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="sci:AttackPattern"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="sci:Vulnerability"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="sci:Weakness"
                      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="ext-restriction"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

  <!--
   ==================================================================
   ==  Reference class                                             ==
   ==================================================================
  -->
    <xs:element name="Reference">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="enum:ReferenceName"
                      minOccurs="0" />
          <xs:element ref="iodef:URL"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Description"
                      minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
      </xs:complexType>
    </xs:element>

  <!--
   ==================================================================
   ==  Assessment class                                            ==
   ==================================================================
  -->
    <xs:element name="Assessment">
      <xs:complexType>
        <xs:sequence>
```

```
          <xs:element ref="iodef:IncidentCategory"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:choice maxOccurs="unbounded">
            <xs:element ref="iodef:SystemImpact"/>
            <xs:element ref="iodef:BusinessImpact"/>
            <xs:element ref="iodef:TimeImpact"/>
            <xs:element ref="iodef:MonetaryImpact"/>
            <xs:element ref="iodef:IntendedImpact"/>
          </xs:choice>
          <xs:element ref="iodef:Counter"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:MitigatingFactor"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Cause"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Confidence" minOccurs="0"/>
          <xs:element ref="iodef:AdditionalData"
                      minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <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"/>
        <xs:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:element name="IncidentCategory" type="iodef:MLStringType"/>
    <xs:element name="BusinessImpact"
                type="iodef:BusinessImpactType"/>
    <xs:element name="IntendedImpact"
                type="iodef:BusinessImpactType"/>
    <xs:element name="MitigatingFactor"
                type="iodef:MLStringType"/>
    <xs:element name="Cause"
                type="iodef:MLStringType"/>
    <xs:element name="SystemImpact">
      <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"
                          type="systemimpact-type-type"
                          use="optional"/>
            <xs:attribute name="ext-type"
                          type="xs:string" use="optional"/>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="systemimpact-type-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="admin"/>
        <xs:enumeration value="takeover-account"/>
        <xs:enumeration value="takeover-service"/>
        <xs:enumeration value="takeover-system"/>
        <xs:enumeration value="cps-manipulation"/>
        <xs:enumeration value="cps-damage"/>
        <xs:enumeration value="availability-data"/>
        <xs:enumeration value="availability-account"/>
        <xs:enumeration value="availability-service"/>
        <xs:enumeration value="availability-system"/>
        <xs:enumeration value="damaged-system"/>
        <xs:enumeration value="damaged-data"/>
        <xs:enumeration value="breach-proprietary"/>
        <xs:enumeration value="breach-privacy"/>
        <xs:enumeration value="breach-credential"/>
        <xs:enumeration value="breach-configuration"/>
        <xs:enumeration value="integrity-data"/>
        <xs:enumeration value="integrity-configuration"/>
        <xs:enumeration value="integrity-hardware"/>
        <xs:enumeration value="traffic-redirection"/>
        <xs:enumeration value="monitoring-traffic"/>
        <xs:enumeration value="monitoring-host"/>
        <xs:enumeration value="policy"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
```

```
    </xs:simpleType>

    <xs:complexType name="BusinessImpactType">
      <xs:simpleContent>
        <xs:extension base="iodef:MLStringType">
          <xs:attribute name="severity"
                        type="businessimpact-severity-type"
                        use="optional"/>
          <xs:attribute name="ext-severity"
                        type="xs:string" use="optional"/>
          <xs:attribute name="type"
                        type="businessimpact-type-type"
                        use="optional"/>
          <xs:attribute name="ext-type"
                        type="xs:string" use="optional"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>

    <xs:simpleType name="businessimpact-severity-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="none"/>
        <xs:enumeration value="low"/>
        <xs:enumeration value="medium"/>
        <xs:enumeration value="high"/>
        <xs:enumeration value="unknown"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="businessimpact-type-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="breach-proprietary"/>
        <xs:enumeration value="breach-privacy"/>
        <xs:enumeration value="breach-credential"/>
        <xs:enumeration value="loss-of-integrity"/>
        <xs:enumeration value="loss-of-service" />
        <xs:enumeration value="theft-financial"/>
        <xs:enumeration value="theft-service"/>
        <xs:enumeration value="degraded-reputation"/>
        <xs:enumeration value="asset-damage"/>
        <xs:enumeration value="asset-manipulation"/>
        <xs:enumeration value="legal"/>
        <xs:enumeration value="extortion"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>
```

```
    <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"
                          type="timeimpact-metric-type"/>
            <xs:attribute name="duration"
                          type="iodef:duration-type"/>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="timeimpact-metric-type">
      <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: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"
                      type="confidence-rating-type"/>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="confidence-rating-type">
      <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>

  <!--
   ==================================================================
   == 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:RecoveryTime"
                      minOccurs="0"/>
          <xs:element ref="iodef:ReportTime"
                      minOccurs="0"/>
          <xs:element ref="iodef:Contact"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Discovery"
                      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"/>
```

```
        <xs:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
        <xs:attribute name="observable-id"
                      type="xs:ID" 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" maxOccurs="unbounded"/>
          <xs:element ref="iodef:NodeRole"
                      minOccurs="0" maxOccurs="unbounded" />
          <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 name="AssetID" type="xs:string"
                      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="ext-restriction"
                      type="xs:string" use="optional"/>
        <xs:attribute name="category" type="system-category-type"/>
```

```
        <xs:attribute name="ext-category"
                      type="xs:string" use="optional"/>
        <xs:attribute name="interface"
                      type="xs:string"/>
        <xs:attribute name="spoofed" type="yes-no-unknown-type"
                      default="unknown" />
        <xs:attribute name="virtual" type="yes-no-unknown-type"
                      use="optional" default="unknown"/>
        <xs:attribute name="ownership" type="system-ownership-type"
                      use="optional" />
        <xs:attribute name="ext-ownership"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="system-category-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="source"/>
        <xs:enumeration value="target"/>
        <xs:enumeration value="intermediate"/>
        <xs:enumeration value="sensor"/>
        <xs:enumeration value="infrastructure"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="system-ownership-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="organization"/>
        <xs:enumeration value="personal"/>
        <xs:enumeration value="partner"/>
        <xs:enumeration value="customer"/>
        <xs:enumeration value="no-relationship"/>
        <xs:enumeration value="unknown"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

  <!--
   ==================================================================
   == Node class                                                   ==
   ==================================================================
  -->
    <xs:element name="Node">
      <xs:complexType>
        <xs:sequence>
          <xs:choice maxOccurs="unbounded">
            <xs:element ref="iodef:DomainData" minOccurs="0"
```

```
                    maxOccurs="unbounded"/>
            <xs:element ref="iodef:Address"
                        minOccurs="0" maxOccurs="unbounded"/>
          </xs:choice>
          <xs:element ref="iodef:PostalAddress"
                      minOccurs="0"/>
          <xs:element ref="iodef:Location"
                      minOccurs="0" maxOccurs="unbounded"/>
          <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"
                          type="address-category-type"
                          default="ipv4-addr"/>
            <xs:attribute name="ext-category"
                          type="xs:string" use="optional"/>
            <xs:attribute name="vlan-name"
                          type="xs:string"/>
            <xs:attribute name="vlan-num"
                          type="xs:integer"/>
            <xs:attribute name="observable-id"
                          type="xs:ID" use="optional"/>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="address-category-type">
      <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"/>
        <xs:enumeration value="site-uri"/>
```

```
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:element name="Location" type="iodef:MLStringType"/>

    <xs:element name="NodeRole">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="iodef:MLStringType">
            <xs:attribute name="category"
                          type="noderole-category-type"
                          use="required"/>
            <xs:attribute name="ext-category"
                          type="xs:string" use="optional"/>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="noderole-category-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="client"/>
        <xs:enumeration value="client-enterprise"/>
        <xs:enumeration value="client-partner"/>
        <xs:enumeration value="client-remote"/>
        <xs:enumeration value="client-kiosk"/>
        <xs:enumeration value="client-mobile"/>
        <xs:enumeration value="server-internal"/>
        <xs:enumeration value="server-public"/>
        <xs:enumeration value="www"/>
        <xs:enumeration value="mail"/>
        <xs:enumeration value="webmail" />
        <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="backup"/>
        <xs:enumeration value="dhcp"/>
        <xs:enumeration value="assessment"/>
```

```
        <xs:enumeration value="source-control"/>
        <xs:enumeration value="config-management"/>
        <xs:enumeration value="monitoring"/>
        <xs:enumeration value="infra"/>
        <xs:enumeration value="infra-firewall"/>
        <xs:enumeration value="infra-router"/>
        <xs:enumeration value="infra-switch"/>
        <xs:enumeration value="camera"/>
        <xs:enumeration value="proxy"/>
        <xs:enumeration value="remote-access"/>
        <xs:enumeration value="log"/>
        <xs:enumeration value="virtualization"/>
        <xs:enumeration value="pos"/>
        <xs:enumeration value="scada"/>
        <xs:enumeration value="scada-supervisory"/>
        <xs:enumeration value="sinkhole"/>
        <xs:enumeration value="honeypot"/>
        <xs:enumeration value="anonymization"/>
        <xs:enumeration value="c2-server"/>
        <xs:enumeration value="malware-distribution"/>
        <xs:enumeration value="drop-server"/>
        <xs:enumeration value="hop-point"/>
        <xs:enumeration value="reflector"/>
        <xs:enumeration value="phishing-site"/>
        <xs:enumeration value="spear-phishing-site"/>
        <xs:enumeration value="recruiting-site"/>
        <xs:enumeration value="fraudulent-site"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

  <!--
   ===================================================================
   ==  Service Class                                                ==
   ===================================================================
  -->
    <xs:element name="Service">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="iodef:ServiceName"
                             minOccurs="0"/>
          <xs:choice minOccurs="0">
            <xs:element ref="iodef:Port"/>
            <xs:element ref="iodef:Portlist"/>
          </xs:choice>
          <xs:element ref="iodef:ProtoType" minOccurs="0"/>
          <xs:element ref="iodef:ProtoCode" minOccurs="0"/>
          <xs:element ref="iodef:ProtoField" minOccurs="0"/>
```

```
          <xs:element ref="iodef:ApplicationHeader"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:EmailData" minOccurs="0"/>
          <xs:element ref="iodef:Application"
                      minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="ip-protocol"
                      type="xs:integer" use="required"/>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:element name="Port" type="xs:integer"/>
    <xs:element name="Portlist" type="iodef:PortlistType"/>
    <xs:element name="ProtoType" type="xs:integer"/>
    <xs:element name="ProtoCode" type="xs:integer"/>
    <xs:element name="ProtoField" type="xs:integer"/>
    <xs:element name="ApplicationHeader"
                type="iodef:ApplicationHeaderType"/>

    <xs:element name="ServiceName">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="iodef:IANAService"/>
          <xs:element ref="iodef:URL"
                            minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:Description"
                            minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>

    <xs:element name="IANAService" type="xs:string"/>

  <!--
   ===================================================================
   ==  Counter class                                                ==
   ===================================================================
  -->
    <xs:element name="Counter">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="xs:double">
            <xs:attribute name="type"
                          type="counter-type-type"
                          use="required"/>
            <xs:attribute name="unit"
```

```
                          type="counter-unit-type"
                          use="required"/>
            <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>

    <xs:simpleType name="counter-type-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="counter"/>
        <xs:enumeration value="rate"/>
        <xs:enumeration value="average"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="counter-unit-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="byte"/>
        <xs:enumeration value="mbit"/>
        <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>

  <!--
   ===================================================================
   ==  EmailData class                                              ==
   ===================================================================
  -->
   <xs:element name="EmailData">
     <xs:complexType>
```

```
       <xs:sequence>
         <xs:element ref="iodef:EmailFrom" minOccurs="0"/>
         <xs:element ref="iodef:EmailSubject" minOccurs="0"/>
         <xs:element ref="iodef:EmailX-Mailer" minOccurs="0"/>
         <xs:element ref="iodef:EmailHeaderField" minOccurs="0"/>
         <xs:element ref="iodef:HashData"
                     minOccurs="0" />
         <xs:element ref="SignatureData"
                     minOccurs="0" />
       </xs:sequence>
       <xs:attribute name="observable-id"
                     type="xs:ID" use="optional"/>
     </xs:complexType>
   </xs:element>

   <xs:element name="EmailFrom" type="iodef:MLStringType"/>
   <xs:element name="EmailSubject" type="iodef:MLStringType"/>
   <xs:element name="EmailX-Mailer" type="iodef:MLStringType"/>
   <xs:element name="EmailHeaderField"
               type="iodef:ApplicationHeaderType"/>

  <!--
   ==================================================================
   ==   DomainData class - from RFC5901                            ==
   ==================================================================
  -->
  <xs:element name="DomainData">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:Name" maxOccurs="1" />
        <xs:element ref="iodef:DateDomainWasChecked"
                    minOccurs="0" maxOccurs="1" />
        <xs:element ref="iodef:RegistrationDate"
                    minOccurs="0" maxOccurs="1" />
        <xs:element ref="iodef:ExpirationDate"
                    minOccurs="0" maxOccurs="1" />
        <xs:element ref="iodef:RelatedDNS"
                    minOccurs="0" maxOccurs="unbounded" />
        <xs:element ref="iodef:Nameservers"
                    minOccurs="0" maxOccurs="unbounded" />
        <xs:element ref="iodef:DomainContacts"
                    minOccurs="0" maxOccurs="1" />
      </xs:sequence>
      <xs:attribute name="system-status"
                    type="domaindata-system-status-type"/>
      <xs:attribute name="ext-system-status"
                    type="xs:string" use="optional"/>
      <xs:attribute name="domain-status"
```

```
                    type="domaindata-domain-status-type"/>
      <xs:attribute name="ext-domain-status"
                    type="xs:string" use="optional"/>
      <xs:attribute name="observable-id"
                    type="xs:ID" use="optional"/>
    </xs:complexType>
  </xs:element>

  <xs:element name="Name" type="xs:string"/>
  <xs:element name="DateDomainWasChecked" type="xs:dateTime"/>
  <xs:element name="RegistrationDate" type="xs:dateTime"/>
  <xs:element name="ExpirationDate" type="xs:dateTime"/>

  <xs:simpleType name="domaindata-system-status-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="spoofed"/>
      <xs:enumeration value="fraudulent"/>
      <xs:enumeration value="innocent-hacked"/>
      <xs:enumeration value="innocent-hijacked"/>
      <xs:enumeration value="unknown"/>
      <xs:enumeration value="ext-value"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="domaindata-domain-status-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="reservedDelegation"/>
      <xs:enumeration value="assignedAndActive"/>
      <xs:enumeration value="assignedAndInactive"/>
      <xs:enumeration value="assignedAndOnHold"/>
      <xs:enumeration value="revoked"/>
      <xs:enumeration value="transferPending"/>
      <xs:enumeration value="registryLock"/>
      <xs:enumeration value="registrarLock"/>
      <xs:enumeration value="other"/>
      <xs:enumeration value="unknown"/>
      <xs:enumeration value="ext-value"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:element name="RelatedDNS">
    <xs:complexType>
     <xs:simpleContent>
      <xs:extension base="xs:string">
      <xs:attribute name="record-type" use="optional">
        <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="A"/>
```

```
            <xs:enumeration value="AAAA"/>
            <xs:enumeration value="AFSDB"/>
            <xs:enumeration value="APL"/>
            <xs:enumeration value="AXFR"/>
            <xs:enumeration value="CAA"/>
            <xs:enumeration value="CERT"/>
            <xs:enumeration value="CNAME"/>
            <xs:enumeration value="DHCID"/>
            <xs:enumeration value="DLV"/>
            <xs:enumeration value="DNAME"/>
            <xs:enumeration value="DNSKEY"/>
            <xs:enumeration value="DS"/>
            <xs:enumeration value="HIP"/>
            <xs:enumeration value="IXFR"/>
            <xs:enumeration value="IPSECKEY"/>
            <xs:enumeration value="LOC"/>
            <xs:enumeration value="MX"/>
            <xs:enumeration value="NAPTR"/>
            <xs:enumeration value="NS"/>
            <xs:enumeration value="NSEC"/>
            <xs:enumeration value="NSEC3"/>
            <xs:enumeration value="NSEC3PARAM"/>
            <xs:enumeration value="OPT"/>
            <xs:enumeration value="PTR"/>
            <xs:enumeration value="RRSIG"/>
            <xs:enumeration value="RP"/>
            <xs:enumeration value="SIG"/>
            <xs:enumeration value="SOA"/>
            <xs:enumeration value="SPF"/>
            <xs:enumeration value="SRV"/>
            <xs:enumeration value="SSHFP"/>
            <xs:enumeration value="TA"/>
            <xs:enumeration value="TKEY"/>
            <xs:enumeration value="TLSA"/>
            <xs:enumeration value="TSIG"/>
            <xs:enumeration value="TXT"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
     </xs:extension>
    </xs:simpleContent>
   </xs:complexType>
   </xs:element>

   <xs:element name="Nameservers">
     <xs:complexType>
       <xs:sequence>
         <xs:element ref="iodef:Server"/>
```

```
         <xs:element ref="iodef:Address" maxOccurs="unbounded"/>
       </xs:sequence>
     </xs:complexType>
   </xs:element>

   <xs:element name="Server" type="xs:string"/>

   <xs:element name="DomainContacts">
     <xs:complexType>
        <xs:choice>
          <xs:element ref="iodef:SameDomainContact"/>
          <xs:element ref="iodef:Contact"
                      maxOccurs="unbounded" minOccurs="1"/>
        </xs:choice>
     </xs:complexType>
   </xs:element>

   <xs:element name="SameDomainContact" type="xs:string"/>

  <!--
   =================================================================
   ==  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:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
      </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"/>
          <xs:element ref="iodef:FileData"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:WindowsRegistryKeysModified"
                      minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="iodef:CertificateData"
                      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="ext-restriction"
                      type="xs:string" use="optional"/>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:element name="RecordPattern">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="xs:string">
            <xs:attribute name="type"
                          type="recordpattern-type-type"
                          use="required"/>
            <xs:attribute name="ext-type"
                          type="xs:string" use="optional"/>
            <xs:attribute name="offset"
                          type="xs:integer" use="optional"/>
            <xs:attribute name="offsetunit"
                          type="recordpattern-offsetunit-type"
                          use="optional" default="line"/>
            <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:simpleType name="recordpattern-type-type">
      <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:simpleType name="recordpattern-offsetunit-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="line"/>
        <xs:enumeration value="byte"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:element name="RecordItem"
                type="iodef:ExtensionType"/>
<!--
 ==================================================================
 ==  Class to describe Windows Registry Keys                     ==
 ==================================================================
-->
    <xs:element name="WindowsRegistryKeysModified">
      <xs:complexType>
        <xs:sequence>
           <xs:element ref="iodef:Key"
                       maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
       </xs:complexType>
     </xs:element>

     <xs:element name="Key">
       <xs:complexType>
         <xs:sequence>
           <xs:element ref="iodef:KeyName"/>
           <xs:element ref="iodef:Value" minOccurs="0"/>
         </xs:sequence>
         <xs:attribute name="registryaction"
                       type="key-registryaction-type"/>
         <xs:attribute name="ext-registryaction"
                       type="xs:string" use="optional"/>
       </xs:complexType>
     </xs:element>

     <xs:element name="KeyName" type="xs:string"/>
     <xs:element name="Value" type="xs:string"/>

     <xs:simpleType name="key-registryaction-type">
       <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>

<!--
=================================================================
==  Classes to describe a file                                 ==
=================================================================
-->

   <xs:element name="FileData">
     <xs:complexType>
       <xs:sequence>
         <xs:element ref="iodef:File"
                     minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="observable-id"
                      type="xs:ID" use="optional"/>
        <xs:attribute name="restriction"
                      type="iodef:restriction-type"/>
        <xs:attribute name="ext-restriction"
                      type="xs:string" use="optional"/>
        </xs:complexType>
   </xs:element>

   <xs:element name="File">
     <xs:complexType>
       <xs:sequence>
        <xs:element ref="iodef:FileName" minOccurs="0" />
        <xs:element ref="iodef:FileSize" minOccurs="0" />
        <xs:element ref="FileType" minOccurs="0" />
        <xs:element ref="iodef:URL"
                    minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="iodef:HashData"
                    minOccurs="0" />
        <xs:element ref="ds:Signature"
                    minOccurs="0" />
        <xs:element name="Application"
                    type="iodef:SoftwareType" minOccurs="0"/>
        <xs:element ref="iodef:FileProperties"
                    minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="observable-id"
```

```
                    type="xs:ID" use="optional"/>
        </xs:complexType>
   </xs:element>

   <xs:element name="FileName" type="xs:string"/>
   <xs:element name="FileSize" type="xs:integer"/>
   <xs:element name="FileType" type="xs:integer"/>
   <xs:element name="FileProperties" type="iodef:ExtensionType"/>

<!--
================================================================
==  Classes to describe a hash                                ==
================================================================
-->

  <xs:element name="HashData">
  <xs:complexType>
    <xs:sequence>
        <xs:element ref="iodef:HashTarget" minOccurs="0"/>
        <xs:element ref="iodef:Hash"
                    minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="iodef:FuzzyHash"
                    minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="scope"
                    type="hashdata-scope-type"
                    use="required"/>
      <xs:attribute name="ext-scope"
                    type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>

  <xs:element name="HashTarget" type="iodef:MLStringType"/>

  <xs:simpleType name="hashdata-scope-type">
    <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="file-contents"/>
      <xs:enumeration value="file-pe-section"/>
      <xs:enumeration value="file-pe-iat"/>
      <xs:enumeration value="file-pe-resource"/>
      <xs:enumeration value="file-pdf-object"/>
      <xs:enumeration value="email-hash"/>
      <xs:enumeration value="email-headers-hash"/>
      <xs:enumeration value="email-body-hash"/>
      <xs:enumeration value="ext-value"/>
    </xs:restriction>
  </xs:simpleType>
```

```
  <xs:element name="Hash">
   <xs:complexType>
    <xs:sequence>
        <xs:element ref="ds:DigestMethod" />
        <xs:element ref="ds:DigestValue" />
        <xs:element ref="ds:CanonicalizationMethod" />
        <xs:element ref="iodef:Application"
                    minOccurs="0"/>
      </xs:sequence>
   </xs:complexType>
  </xs:element>

  <xs:element name="FuzzyHash">
   <xs:complexType>
    <xs:sequence>
        <xs:element ref="iodef:AdditionalData" />
        <xs:element ref="iodef:Application"
                    minOccurs="0"/>
      </xs:sequence>
   </xs:complexType>
  </xs:element>

<!--
=================================================================
==  Classes to describe a signature                            ==
=================================================================
-->

  <xs:element name="SignatureData">
   <xs:complexType>
    <xs:sequence>
        <xs:element ref="ds:Signature"
                    maxOccurs="unbounded" />
      </xs:sequence>
   </xs:complexType>
  </xs:element>

<!--
=================================================================
==  Classes to describe a certficate                           ==
=================================================================
-->

  <xs:element name="CertificateData">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:Certificate"
                    maxOccurs="unbounded"/>
```

```
      </xs:sequence>
      <xs:attribute name="observable-id"
                    type="xs:ID" use="optional"/>
      <xs:attribute name="restriction"
                    type="iodef:restriction-type"/>
      <xs:attribute name="ext-restriction"
                    type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>

  <xs:element name="Certificate">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="ds:X509Data" />
        <xs:element ref="iodef:Description"
                    minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
     <xs:attribute name="observable-id"
                   type="xs:ID" use="optional"/>
    </xs:complexType>
  </xs:element>

  <!--
   ==================================================================
   ==  Classes that describe software                              ==
   ==================================================================
  -->
    <xs:complexType name="SoftwareType">
      <xs:sequence>
        <xs:element ref="iodef:SoftwareReference"
                    minOccurs="0"/>
        <xs:element ref="iodef:URL"
                    minOccurs="0" maxOccurs="unbounded" />
        <xs:element ref="iodef:Description"
                    minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>

    <xs:element name="SoftwareReference">
      <xs:complexType>
        <xs:sequence>
          <xs:any namespace="##any" processContents="lax"
                  minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="spec-name"
                      type="softwarereference-spec-name-type"
                      use="required"/>
        <xs:attribute name="ext-spec-name"
```

```
                      type="xs:string" use="optional"/>
        <xs:attribute name="dtype"
                      type="softwarereference-dtype-type"
                      use="optional"/>
        <xs:attribute name="ext-dtype"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="softwarereference-spec-name-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="custom"/>
        <xs:enumeration value="cpe"/>
        <xs:enumeration value="swid"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="softwarereference-dtype-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="bytes"/>
        <xs:enumeration value="integer"/>
        <xs:enumeration value="real"/>
        <xs:enumeration value="string"/>
        <xs:enumeration value="xml"/>
        <xs:enumeration value="ext-value"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:element name="Application" type="iodef:SoftwareType"/>
    <xs:element name="OperatingSystem" type="iodef:SoftwareType"/>

  <!--
   ==================================================================
   == IndicatorData classes                                       ==
   ==================================================================
-->
   <xs:element name="IndicatorData">
     <xs:complexType>
       <xs:sequence>
         <xs:element ref="iodef:Indicator"
                     minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
     </xs:complexType>
   </xs:element>

   <xs:element name="Indicator">
     <xs:complexType>
```

```
      <xs:sequence>
        <xs:element ref="iodef:IndicatorID" />
        <xs:element ref="iodef:AlternativeIndicatorID"
                    minOccurs="0" maxOccurs="unbounded"/>
        <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:Confidence"
                    minOccurs="0" />
        <xs:element ref="iodef:Contact"
                    minOccurs="0" maxOccurs="unbounded"/>
        <xs:choice>
          <xs:element ref="iodef:Observable" />
          <xs:element ref="iodef:ObservableReference" />
          <xs:element ref="iodef:IndicatorExpression" />
          <xs:element ref="iodef:IndicatorReference" />
        </xs:choice>
      </xs:sequence>
      <xs:attribute name="restriction"
                    type="iodef:restriction-type"/>
      <xs:attribute name="ext-restriction"
                    type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>

  <xs:element name="IndicatorID">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:ID">
          <xs:attribute name="name"
                        type="xs:string" use="required"/>
          <xs:attribute name="version"
                        type="xs:string" use="required"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name="AlternativeIndicatorID">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="iodef:IndicatorID"
                    maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="restriction"
```

```
                     type="iodef:restriction-type"/>
       <xs:attribute name="ext-restriction"
                     type="xs:string" use="optional"/>
     </xs:complexType>
   </xs:element>

   <xs:element name="Observable">
     <xs:complexType>
       <xs:sequence>
         <xs:element ref="iodef:Address"
                     minOccurs="0"/>
         <xs:element ref="iodef:DomainData"
                     minOccurs="0"/>
         <xs:element ref="iodef:EmailData"
                     minOccurs="0"/>
         <xs:element name="ApplicationHeader"
                     type="iodef:ApplicationHeaderType"
                     minOccurs="0"/>
         <xs:element ref="iodef:WindowsRegistryKeysModified"
                     minOccurs="0"/>
         <xs:element ref="iodef:FileData"
                     minOccurs="0"/>
         <xs:element ref="iodef:CertificateData"
                     minOccurs="0"/>
         <xs:element ref="iodef:RegistryHandle"
                     minOccurs="0"/>
         <xs:element ref="iodef:RecordData"
                     minOccurs="0"/>
         <xs:element ref="iodef:EventData"
                     minOccurs="0"/>
         <xs:element ref="iodef:Incident"
                     minOccurs="0"/>
         <xs:element ref="iodef:Expectation"
                     minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="Reference"
                     minOccurs="0" maxOccurs="unbounded"/>
         <xs:element ref="iodef:Assessment"
                     minOccurs="0"/>
         <xs:element ref="iodef:HistoryItem"
                     minOccurs="0"/>
         <xs:element ref="iodef:BulkObservable"
                     minOccurs="0"/>
         <xs:element ref="iodef:AdditionalData"
                     minOccurs="0"/>
       </xs:sequence>
       <xs:attribute name="restriction"
                     type="iodef:restriction-type"/>
       <xs:attribute name="ext-restriction"
```

```
                     type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:element name="BulkObservable">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="iodef:BulkObservableFormat"
                      minOccurs="0"/>
          <xs:element name="BulkObservableList"
                      type="xs:string" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="type"
                      type="observable-type-type"
                      use="required"/>
        <xs:attribute name="ext-type"
                      type="xs:string" use="optional"/>
      </xs:complexType>
    </xs:element>

    <xs:simpleType name="observable-type-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="asn"/>
        <xs:enumeration value="atm"/>
        <xs:enumeration value="e-mail"/>
        <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"/>
        <xs:enumeration value="mac"/>
        <xs:enumeration value="site-uri"/>
        <xs:enumeration value="fqdn"/>
        <xs:enumeration value="doman-name"/>
        <xs:enumeration value="domain-to-ipv4"/>
        <xs:enumeration value="domain-to-ipv6"/>
        <xs:enumeration value="domain-to-ipv4-timestamp"/>
        <xs:enumeration value="domain-to-ipv6-timestamp"/>
        <xs:enumeration value="ipv4-port"/>
        <xs:enumeration value="ipv6-port"/>
        <xs:enumeration value="windows-reg-key"/>
        <xs:enumeration value="file-hash"/>
        <xs:enumeration value="email-x-mailer"/>
        <xs:enumeration value="email-subject"/>
        <xs:enumeration value="http-user-agent"/>
        <xs:enumeration value="http-request-uri"/>
        <xs:enumeration value="mutex"/>
```

```
       <xs:enumeration value="file-path"/>
       <xs:enumeration value="user-name"/>
     </xs:restriction>
   </xs:simpleType>

   <xs:element name="BulkObservableFormat">
     <xs:complexType>
       <xs:sequence>
         <xs:element ref="iodef:Hash"
                     minOccurs="0"/>
         <xs:element ref="iodef:AdditionalData"
                     minOccurs="0" maxOccurs="unbounded"/>
       </xs:sequence>
     </xs:complexType>
   </xs:element>

   <xs:element name="IndicatorExpression">
     <xs:complexType>
       <xs:sequence>
         <xs:choice>
           <xs:element ref="iodef:IndicatorExpression"
                       minOccurs="0"/>
           <xs:element ref="iodef:Observable"
                       minOccurs="0" />
           <xs:element ref="iodef:ObservableReference"
                       minOccurs="0"/>
           <xs:element ref="iodef:IndicatorReference"
                       minOccurs="0"/>
         </xs:choice>
         <xs:element ref="iodef:AlternativeIndicatorID"
                     minOccurs="0" maxOccurs="unbounded"/>
       </xs:sequence>
       <xs:attribute name="operator"
                     type="indicatorexpression-operator-type"
                     use="optional"
                     default="and">
      </xs:attribute>
     </xs:complexType>
   </xs:element>

   <xs:simpleType name="indicatorexpression-operator-type">
     <xs:restriction base="xs:NMTOKEN">
       <xs:enumeration value="not"/>
       <xs:enumeration value="and"/>
       <xs:enumeration value="or"/>
       <xs:enumeration value="xor"/>
     </xs:restriction>
   </xs:simpleType>
```

```
   <xs:element name="ObservableReference">
     <xs:complexType>
       <xs:attribute name="uid-ref"
                     type="xs:IDREF" use="required"/>
     </xs:complexType>
   </xs:element>

   <xs:element name="IndicatorReference">
     <xs:complexType>
       <xs:attribute name="uid-ref"
                     type="xs:IDREF" use="optional"/>
       <xs:attribute name="euid-ref"
                     type="xs:string" use="optional"/>
       <xs:attribute name="version"
                     type="xs:string" use="optional"/>
     </xs:complexType>
   </xs:element>
<!--
   ==================================================================
   == Miscellaneous simple classes                                 ==
   ==================================================================
  -->
    <xs:element name="Description" type="iodef:MLStringType"/>
    <xs:element name="URL" type="xs:anyURI"/>
  <!--
   ==================================================================
   == IODEF Basic 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="translation-id"
                        type="xs:string" use="optional"/>
          <xs:attribute ref="xml:lang" />
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>

    <xs:simpleType name="PortlistType">
      <xs:restriction base="xs:string">
        <xs:pattern value="\d+(\-\d+)?(,\d+(\-\d+)?)*"/>
```

```
      </xs:restriction>
    </xs:simpleType>

    <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="meaning"
                    type="xs:string"/>
      <xs:attribute name="formatid"
                    type="xs:string"/>
      <xs:attribute name="restriction"
                    type="iodef:restriction-type"/>
      <xs:attribute name="ext-restriction"
                    type="xs:string" use="optional"/>
    </xs:complexType>

    <xs:complexType name="ApplicationHeaderType" mixed="true">
      <xs:sequence>
         <xs:any namespace="##any" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="proto"
                    type="xs:integer" use="optional"/>
      <xs:attribute name="proto-name"
                    type="xs:integer" use="optional"/>
      <xs:attribute name="field"
                    type="xs:string" use="required"/>
      <xs:attribute name="dtype"
                    type="iodef:proto-dtype-type"
                    use="required"/>
      <xs:attribute name="observable-id"
                    type="xs:ID" use="optional"/>
    </xs:complexType>

  <!--
   ==================================================================
   == Global attribute type declarations                           ==
   ==================================================================
  -->
    <xs:simpleType name="yes-no-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="yes"/>
        <xs:enumeration value="no"/>
      </xs:restriction>
```

```
    </xs:simpleType>

    <xs:simpleType name="yes-no-unknown-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="yes"/>
        <xs:enumeration value="no"/>
        <xs:enumeration value="unknown"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="restriction-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="default"/>
        <xs:enumeration value="public"/>
        <xs:enumeration value="partner"/>
        <xs:enumeration value="need-to-know"/>
        <xs:enumeration value="private"/>
        <xs:enumeration value="white"/>
        <xs:enumeration value="green"/>
        <xs:enumeration value="amber"/>
        <xs:enumeration value="red"/>
        <xs:enumeration value="ext-value"/>
      </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="redirect-traffic"/>
        <xs:enumeration value="honeypot"/>
        <xs:enumeration value="upgrade-software"/>
        <xs:enumeration value="rebuild-asset"/>
        <xs:enumeration value="harden-asset"/>
        <xs:enumeration value="remediate-other"/>
        <xs:enumeration value="status-triage"/>
        <xs:enumeration value="status-new-info"/>
        <xs:enumeration value="watch-and-report"/>
        <xs:enumeration value="defined-coa"/>
        <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="bytes"/>
        <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="proto-dtype-type">
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="boolean"/>
        <xs:enumeration value="byte"/>
        <xs:enumeration value="bytes"/>
        <xs:enumeration value="character"/>
        <xs:enumeration value="date-time"/>
        <xs:enumeration value="integer"/>
        <xs:enumeration value="real"/>
        <xs:enumeration value="string"/>
        <xs:enumeration value="xml"/>
        <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.

Executable content could be embedded into the IODEF document directly or through an extension. The IODEF parser MUST handle this content with care to prevent unintentional automated execution.

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 [RFC6545] and its associated transport binding IODEF/RID over HTTP/TLS [RFC6546] 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 registers a namespace, XML schema, and a number of registries that map to enumerated values defined in the schema.

10.1. Namespace and Schema

This document uses URNs to describe an XML namespace and schema conforming to a registry mechanism described in [RFC3688]

Registration for the IODEF namespace:

- URI: urn:ietf:params:xml:ns:iodef-2.0

- Registrant Contact: See the first author of the "Author’s Address" section of this document.

- XML: None. Namespace URIs do not represent an XML specification.

Registration for the IODEF XML schema:

- URI: urn:ietf:params:xml:schema:iodef-2.0

- Registrant Contact: See the first author of the "Author’s Address" section of this document.

- XML: See the "IODEF Schema" in Section 8 of this document.

10.2. Enumerated Value Registries

This document creates xx identically structured registries to be managed by IANA:

- Name of the parent registry: "Incident Object Description Exchange Format v2 (IODEF)"

- o URL of the registry: http://www.iana.org/assignments/iodef2
- o Namespace format: A registry entry consists of:
  - * Value. An enumerated value for a given IODEF attribute.
  - * Description. A short description of the enumerated value.
  - * Reference. An optional list of URIs to further describe the value.
- o Allocation policy: Expert Review per [RFC5226]

The registries to be created are named in the table below in the "Registry Name" column. The initial values for the Value and Description fields of a given registry are listed in the "IV (Value)" and "IV (Description)" columns respectively. The "IV (Value)" points to a given schema attribute or type per Section 8. Each enumerated value in the schema gets a corresponding entry in a given registry. The "IV (Description)" points to a section in the text of this document. The initial value of the Reference field of every registry entry described below should be this document.

Registry Name	IV (Value)	IV (Description)
Restriction	iodef-restriction-type	Section 3.3.1
Incident-purpose	Incident@purpose	Section 3.2
Incident-status	Incident@status	Section 3.2
Contact-role	Contact@role	Section 3.10
Contact-type	Contact@type	Section 3.10
RegistryHandle-registry	RegistryHandle@registry	Section 3.10.1
Expectation-action	iodef:action-type	Section 3.17
Discovery-source	Discovery@source	Section 3.12
SystemImpact-type	SystemImpact@type	Section 3.14.1
BusinessImpact-severity	BusinessImpact@severity	Section

		3.14.2
BusinessImpact-type	BusinessImpact@type	Section 3.14.2
TimeImpact-metrics	TimeImpact@metric	Section 3.14.3
TimeImpact-duration	iodef:duration-type	Section 3.14.3
NodeRole-category	NodeRole@category	Section 3.20.2
System-category	System@category	Section 3.19
System-ownership	System@ownership	Section 3.19
Address-category	Address@category	Section 3.20.1
Counter-type	Counter@type	Section 3.20.3
Counter-unit	Counter@unit	Section 3.20.3
DomainData-system-status	DomainData@system-status	Section 3.21
DomainData-domain-status	DomainData@domain-status	Section 3.21
RelatedDNS-record-type	RelatedDNS@record-type	Section 3.21.1
RecordPattern-type	RecordPattern@type	Section 3.25.2
RecordPattern-offsetunit	RecordPattern@offsetunit	Section 3.25.2
Key-registryaction	Key@registryaction	Section 3.26.1
HashData-scope	HashData@scope	Section 3.29
BulkObservable-type	BulkObservable@type	Section

		3.32.3.1
IndicatorExpression-operator	IndicatorExpression@operator	Section 3.32.4
AdditionalData-dtype	iodef:dtype-type	Section 3.9
ApplicationHeader-proto-dtype	iodef:proto-dtype-type	Section 3.22.2
SoftwareReference-dtype	SoftwareReference	Section 3.22.4

Table 1: IANA Enumerated Value Registries

11. Acknowledgments

The following groups and individuals, listed alphabetically, contributed substantially to this document and should be recognized for their efforts.

... TODO ...

12. References

12.1. Normative References

[W3C.XML] 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>.

[W3C.SCHEMA] World Wide Web Consortium, "XML XML Schema Part 1: Structures Second Edition", W3C Recommendation , October 2004, <http://www.w3.org/TR/xmlschema-1/>.

[W3C.SCHEMA.DTYPES] World Wide Web Consortium, "XML Schema Part 2: Datatypes Second Edition", W3C Recommendation , October 2004, <http://www.w3.org/TR/xmlschema-2/>.

[W3C.XMLNS] World Wide Web Consortium, "Namespaces in XML", W3C Recommendation , January 1999, <http://www.w3.org/TR/REC-xml-names/>.

[W3C.XPATH]
World Wide Web Consortium, "XML Path Language (XPath) 2.0", W3C Candidate Recommendation , June 2006, <http://www.w3.org/TR/xpath20/>.

[W3C.XMLSIG]
World Wide Web Consortium, "XML Signature Syntax and Processing 2.0", W3C Candidate Recommendation , June 2008, <http://www.w3.org/TR/xmldsig-core/>.

[IEEE.POSIX]
Institute of Electrical and Electronics Engineers, "Information Technology - Portable Operating System Interface (POSIX) - Part 1: Base Definitions", IEEE 1003.1, June 2001.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC5646] Philips, A. and M. Davis, "Tags for Identifying of Languages", RFC 5646, September 2009.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 3986, January 2005'.

[RFC2978] Freed, N. and J. Postel, "IANA Charset Registration Procedures", BCP 2978, October 2000.

[RFC4519] Sciberras, A., "Schema for User Applications", RFC 4519, June 2006.

[RFC5322] Resnick, P., "Internet Message Format", RFC 5322, October 2008.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

[RFC-ENUM]
Montville, A. and D. Black, "IODEF Enumeration Reference Format", RFC ENUM, January 2015.

[RFC-SCI] Takahashi, T., Landfield, K., and Y. Kadobayashi, "An Incident Object Description Exchange Format (IODEF) Extension for Structured Cybersecurity Information", RFC 5901, April 2014.

[ISO8601] International Organization for Standardization, "International Standard: Data elements and interchange formats - Information interchange - Representation of dates and times", ISO 8601, Second Edition, December 2000.

[ISO4217] International Organization for Standardization, "International Standard: Codes for the representation of currencies and funds, ISO 4217:2001", ISO 4217:2001, August 2001.

[RFC3688] Mealling, M., "The IETF XML Registry", RFC 3688, January 2004.

[IANA.Ports] Internet Assigned Numbers Authority, "Service Name and Transport Protocol Port Number Registry", January 2014, <http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.txt>.

[IANA.Protocols] Internet Assigned Numbers Authority, "Assigned Internet Protocol Numbers", January 2014, <http://www.iana.org/assignments/protocol-numbers/protocol-numbers.txt>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", RFC 3629, November 2003.

[RFC2781] Hoffman, P. and F. Yergeau, "UTF-16, an encoding of ISO 10646", RFC 2781, February 2000.

[IANA.Media] Internet Assigned Numbers Authority, "Media Types", March 2015, <http://www.iana.org/assignments/media-types/media-types.xhtml>.

12.2. Informative References

[RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "Incident Object Description Exchange Format", RFC 5070, December 2007.

[refs.requirements] Keeni, G., Demchenko, Y., and R. Danyliw, "Requirements for the Format for Incident Information Exchange (FINE)", Work in Progress, June 2006.

[RFC4765] Debar, H., Curry, D., Debar, H., and B. Feinstein, "Intrusion Detection Message Exchange Format", RFC 4765, March 2007.

[RFC6545] Moriarty, K., "Real-time Inter-network Defense (RID)", RFC 6545, April 2012.

[RFC6546] Trammell, B., "Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS", RFC 6546, April 2012.

[RFC5901] Cain, P. and D. Jevans, "Extensions to the IODEF-Document Class for Reporting Phishing", RFC 5901, July 2010.

[NIST800.61rev2] Cichonski, P., Millar, T., Grance, T., and K. Scarfone, "NIST Special Publication 800-61 Revision 2: Computer Security Incident Handling Guide", January 2012, <http://csrc.nist.gov/publications/nistpubs/800-61rev2/SP800-61rev2.pdf>.

[RFC3982] Newton, A. and M. Sanz, "IRIS: A Domain Registry (dreg) Type for the Internet Registry Information Service (IRIS)", RFC 3982, January 2005.

[KB310516] Microsoft Corporation, "How to add, modify, or delete registry subkeys and values by using a registration entries (.reg) file", December 2007.

[RFC4180] Shafranovich, Y., "Common Format and MIME Type for Comma-Separated Values (CSV) File", RFC 4180, October 2005.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, May 2008.

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

MILE Working Group J. Field
Internet-Draft EMC
Intended status: Informational December 5, 2014
Expires: June 8, 2015

Resource-Oriented Lightweight Indicator Exchange
draft-ietf-mile-rolie-00.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

Copyright Notice

Table of Contents

1. Introduction

This document defines a resource-oriented approach to cyber security information sharing that follows the REST (Architectural Styles and t he 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 been 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 Publ
ishing 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/name
space">
      <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" t
erm="public" />
                  <atom:category scheme="http://example.org/csirt/purpose" term=
"reporting" />
                </categoies>
                <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" t
erm="public" />
                  <atom:category scheme="http://example.org/csirt/purpose" term=
"reporting" />
                </categoies>
                <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" term=
"traceback, mitigation, reporting" />
                  <atom:category scheme="http://example.org/csirt/restriction" t
erm="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/atom.
xsd
                              urn:ietf:params:xml:ns:iodef-1.0 file:/C:/schemas/
iodef-1.0.xsd"
          xml:lang="en-US">

          <generator version="1.0" xml:lang="en-US">emc-csirt-iodef-feed-service
</generator>
          <id xml:lang="en-US">http://www.example.org/csirt/private/incidents</i
d>
          <title type="text" xml:lang="en-US">Atom formatted representation of 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="self"
/>

          <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 attr
ibutes -->
              <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>
          </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 alterntive to the RID investigation request messaage, 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-s
pecific) link relationships -->
        <!-- Typical operations that can be performed on this IODEF message incl
ude 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 an
ything 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:io
def-1.0">
            <iodef:Incident purpose="traceback" restriction="need-to-know">

              <!-- Note that the ID is assigned using a namespace that is our ba
se URL, so that it can also be leveraged as an Atom link -->
              <iodef:IncidentID name="http://www.example.org/csirt/private/incid
ents">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>
                      <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 attack
                    </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 retreives 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 o
f this incident, and so on.... -->
        <link href="http://www.example.org/csirt/private/incidents/123456/relati
onships/indicators" rel="indicators"/>
        <link href="http://www.example.org/csirt/private/incidents/1234456/relat
ionships/history" rel="history"/>
        <link href="http://www.example.org/csirt/private/incidents/1234456/relat
ionships/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:io
def-1.0">
            <iodef:Incident purpose="traceback" restriction="need-to-know">
              <iodef:IncidentID name="http://www.example.org/csirt/private/incid
ents">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/name
space">
      <atom:title type="text">RID Use Case Requests</atom:title>
      <collection  href="http://www.example.org/csirt/RID/InvestigationReq
uests">
         <atom:title type="text">Investigation Requests</atom:title>
         <accept>application/atom+xml; type=entry</accept>  <!-- perhaps w
e 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 and u
pdated not guranteed to be preserved -->
  <updated>2012-08-12T11:08:22Z</updated>                          <!-- may want
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">1
23</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>  <!-- id a
nd updated not guranteed to be preserved -->
  <updated>2012-08-12T11:08:30Z</updated>                              <!-- may
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">12
3</iodef:IncidentID>
        <!-- ...additional incident data.... -->
      </iodef:Incident>
    </iodef:IODEF-Document>
  </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 messaage, 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/atom.
xsd
                        urn:ietf:params:xml:ns:iodef-1.0 file:/C:/schemas/
iodef-1.0.xsd"
    xml:lang="en-US">

    <link href="http://www.example.org/opensearchdescription.xml" rel="sea
rch"
            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 searc
h interface</Description>
  <Tags>example csirt indicator search</Tags>
  <Contact>admin@example.org</Contact>
  <!-- ...optionally, other elements, as per OpenSearch specificat
ion... -->
  <Url type="application/opensearchdescription+xml" rel="self" tem
plate="http://www.example.com/csirt/opensearchdescription.xml"/>
  <Url type="application/atom+xml" rel="results" template="http://
www.example.org/csirt?q={searchTerms}&amp;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 respository.

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/atom.
xsd
                              urn:ietf:params:xml:ns:iodef-1.0 file:/C:/schemas/
iodef-1.0.xsd"
          xml:lang="en-US">

          <generator version="1.0" xml:lang="en-US">emc-csirt-iodef-feed-service
</generator>
          <id xml:lang="en-US">http://www.example.org/csirt/repository/ddos</id>
          <title type="text" xml:lang="en-US">Atom formatted representation of 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" r
el="self"/>          <!-- by convention -->
              <link href="http://www.example.org/csirt/repository/ddos/123456" r
el="alternate"/>     <!-- required by Atom spec -->
              <link href="http://www.example.org/csirt/repository/ddos/987654" r
el="related"/>       <!-- link to a related DDOS resource in this repository -->
              <link href="http://www.cyber-agency.gov/repository/indicators/1a2b
3c" rel="related"/>  <!-- link to a related DDOS resource in another repository
-->
              <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 attr
ibutes -->
              <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 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 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 repostory 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 consotium 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 resources that provides a representation of the attacker.	SHOULD
vector	Provides a link to a	SHOULD

	collection of zero or more resources that provides a representation of the method used by the attacker.	
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

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 rek 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 unkown type the client MUST ignore the offending link and continue processing the remaining resource representation as if the offending link element did not appear.

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 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?

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>.

[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.

10.3. URIs

[1] http://csrc.nist.gov/groups/SNS/rbac/

Appendix A. Change Tracking

Changes since -00 version, September 5, 2012 to Feb 15, 2013:

- o Fixed a small number of typographical errors and a few misspellings throughout.

- o Added a number of missing internal cross references to improve readability.

- o Updated the text in the Introduction section for improved brevity and clarity of goal. See: Section 1

- o Added new non-normative text describing the use of HTTP 4xx status codes for authorization. See: Section 3.3.2

- o Added a new non-normative example illustrating a persistent repository use case. See: Section 4.2.4.4

- o Added new normative text recommending use of SAML2 for authentication of interactive end users who are members of a sharing consortium. See: Section 5.2

- o Added new normative text describing requirements for user authorization. See: Section 5.3

- o Added non-normative appendix for change tracking. See: Appendix A

- o Added non-normative appendix describing a suggested approach to a XACML profile. See: Appendix B

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 aproach finds support in the community then this section (or a new draft, as a seperate document) could provide a more complete XACML 3.0 compliant example.

Author’s Address

John P. Field
EMC Corporation
1133 Westchester Avenue
White Plains, New York
USA

Phone: 914-461-3522
Email: jfield@pivotal.io