Definition of ROLIE CSIRT Extension
draft-banghart-mile-rolie-csirt-01

Abstract

This document extends the Resource-Oriented Lightweight Information Exchange (ROLIE) core to add the information type categories and related requirements needed to support Computer Security Incident Response Team (CSIRT) use cases. The indicator and incident information types are defined as ROLIE extensions. Additional supporting requirements are also defined that describe the use of specific formats and link relations pertaining to the new information types.

Contributing to this document

The source for this draft is being maintained in GitHub. Suggested changes should be submitted as pull requests at <https://github.com/CISecurity/ROLIE>. Instructions are on that page as well. Editorial changes can be managed in GitHub, but any substantial issues need to be discussed on the MILE mailing list.

Status of This Memo

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

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

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

This Internet-Draft will expire on November 4, 2017.
Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the
document authors. All rights reserved.

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

Table of Contents

1. Introduction ............................................. 3
2. Terminology ............................................ 4
3. New information-types ................................. 4
   3.1. The "incident" information type .................. 4
   3.2. The "indicator" information type ................ 5
4. Usage of CSIRT Information Types in the Atom Publishing
   Protocol ................................................. 5
   4.1. / (forward slash) Resource URL ................. 5
5. Usage of CSIRT Information Types in the atom:feed element .. 5
6. Usage of CSIRT Information Types in an atom:entry .......... 6
   6.1. Use of the atom:link element .................... 6
   6.1.1. Link relations for the ‘incident’
          information-type .............................. 6
   6.1.2. Link relations for the ‘indicator’
          information-type .............................. 6
   6.1.3. Link relations for both information-types ... 7
   6.2. Use of the rolie:format element ................. 7
   6.2.1. IODEF Format ................................ 8
   6.2.2. STIX Format ................................ 8
   6.3. Use of the rolie:property element .............. 8
   6.3.1. urn:ietf:params:rolie:property:csirt:ID ...... 8
   6.4. Additional requirements for use of IODEF .......... 9
   6.4.1. The IODEF Document .......................... 9
   6.4.2. Category Element ............................. 9
   6.4.3. Entry Elements ................................ 9
   6.4.4. User Authorization ............................ 10
   6.4.5. Expectation and Impact Classes ............... 10
   6.4.6. Search ......................................... 10
7. IANA Considerations ..................................... 11
   7.1. information-type registrations ..................... 11
   7.1.1. incident information-type .................... 11
   7.1.2. indicator information-type ................. 11
   7.2. atom:category scheme registrations ................ 11
   7.2.1. category:csirt:iodef:purpose ............. 11
1. Introduction

Threats to computer security are evolving ever more rapidly as time goes on. As software increases in complexity, the number of vulnerabilities in systems and networks can increase exponentially. Threat actors looking to exploit these vulnerabilities are making more frequent and more widely distributed attacks across a large variety of systems. The adoption of liberal information sharing amongst attackers allows a discovered vulnerability to be shared and used to attack a vulnerable system within a narrow window of time. As the skills and knowledge required to identify and combat these attacks become more and more specialized, even a well established and secure system may find itself unable to quickly respond to an incident. Effective identification of and response to a sophisticated attack requires open cooperation and collaboration between defending operators, software vendors, and end-users. To improve the timeliness of responses, automation must be used to acquire, contextualize, and put to use shared computer security information.

CSIRTS share two primary forms of information: incidents and indicators. Using these forms of information, analysts are able to perform a wide range of activities both proactive and reactive to ensure the security of their systems.

Incident information describes a cyber security incident. Such information may include attack characteristics, information about the attacker, and attack vector data. Sharing this information helps analysts within the sharing community to inoculate their systems against similar attacks, providing proactive protection.

Indicator information describes the symptoms or necessary pre-conditions of an attack. Everything from system vulnerabilities to unexpected network traffic can help analysts secure systems and prepare for an attack. Making this information available for sharing
aids in the proactive defense of systems both within an operating unit but also for any CSIRTs that are part of a sharing consortium.

As a means to bring automation of content discovery and dissemination into the CSIRT domain, this specification provides an extension to the Resource-Oriented Lightweight Information Exchange (ROLIE) core [I-D.ietf-mile-rolie] designed to address CSIRT use cases. The primary purpose of this extension is to define two new information types: incident, and indicator, along with formats and link relations that support these information-types.

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. New information-types

This document defines the following two information types:

3.1. The "incident" information type

The "incident" information type represents any information describing or pertaining to a computer security incident. This document uses the definition of incident provided by [RFC4949]. Provided below is a non-exhaustive list of information that may be considered to be an incident information type.

- Timing information: start and end times for the incident and/or the response.
- Descriptive information: plain text or machine readable data that provides some degree of description of the incident itself.
- Response information: the methods and results of a response to the incident.
- Meta and contact information: data about the CSIRT that recorded the information, or the operator that enacted the response.
- Effect and result information: data that describes the effects of an incident, or what the final results of the incident are.
Note again that this list is not exhaustive, any information that is in the abstract realm of an incident should be classified under this information-type.

3.2. The "indicator" information type

The "indicator" information type represents computer security indicators or any information surrounding them. This document uses the definition of indicator provided by [RFC4949]. Some examples of indicator information is provided below, but note that indicator is defined in an abstract sense, to be understood as a flexible and widely-applicable definition.

- Specific vulnerabilities that indicate a vector for attack.
- Signs of malicious reconnaissance.
- Definitions of patterns of other indicators.
- Events that may indicate an attack and information regarding those events.
- Meta information about the collecting agent.

This list is intended to provide examples of the indicator information-type, not to define it.

4. Usage of CSIRT Information Types in the Atom Publishing Protocol

These requirements apply when a ROLIE repository contains any Collections with categories with scheme attributes of either CSIRT information type, or if the CSIRT information types appear in the Categories document.

4.1. / (forward slash) Resource URL

The forward slash resource URL MUST be supported as defined in Section 5.5 [I-D.ietf-mile-rolie]. Note that this is a stricter requirement than the core document.

5. Usage of CSIRT Information Types in the atom:feed element

This document does not define any additional requirements for Feeds.
6. Usage of CSIRT Information Types in an atom:entry

This document defines the following requirements for any Entries that are of the CSIRT information type categories.

6.1. Use of the atom:link element

These sections define requirements for atom:link elements in Entries. Note that the requirements are determined by the information type that appears in either the Entry or in the parent Feed.

6.1.1. Link relations for the 'incident' information-type

If the category of an Entry is the incident information type, then the following requirements MUST be followed for inclusion of atom:link elements.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Conformance</th>
</tr>
</thead>
<tbody>
<tr>
<td>indicators</td>
<td>Provides a link to a collection of zero or more instances of cyber security indicators that are associated with the resource.</td>
<td>SHOULD</td>
</tr>
<tr>
<td>evidence</td>
<td>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.</td>
<td>SHOULD</td>
</tr>
<tr>
<td>attacker</td>
<td>Provides a link to a collection of zero or more resources that provides a representation of the attacker.</td>
<td>SHOULD</td>
</tr>
<tr>
<td>vector</td>
<td>Provides a link to a collection of zero or more resources that provides a representation of the method used by the attacker.</td>
<td>SHOULD</td>
</tr>
</tbody>
</table>

Table 1: Link Relations for Resource-Oriented Lightweight Indicator Exchange

6.1.2. Link relations for the 'indicator' information-type

If the category of an Entry is the indicator information type, then the following requirements MUST be followed for inclusion of atom:link elements.
Table 2: Link Relations for Resource-Oriented Lightweight Indicator Exchange

6.1.3. Link relations for both information-types

If the category of an Entry is either information-type, the following requirements MUST be followed for inclusion of atom:link elements.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Conformance</th>
</tr>
</thead>
<tbody>
<tr>
<td>incidents</td>
<td>Provides a link to a collection of zero or more instances of incident</td>
<td>SHOULD</td>
</tr>
<tr>
<td></td>
<td>representations associated with the resource.</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Link Relations for Resource-Oriented Lightweight Indicator Exchange

6.2. Use of the rolie:format element

This document does not contain any additional requirements for the rolie:format element; the formats that follow are provided as examples of formats that describe the incident and indicator.
information type. The formats are in no particular order, and are not requirements, nor suggestions by the authors.

6.2.1. IODEF Format

The Incident Object Description Exchange Format (IODEF) is a format for representing computer security information commonly exchanged between Computer Security Incident Response Teams (CSIRTs) or other operational security teams.

IODEF conveys indicators, incident reports, response activities, and related meta-data in an XML serialization. This information is formally structured in order to support and encourage automated machine-to-machine security communication, as well as enhanced processing at the endpoint.

The full IODEF specification provides further high-level discussion and technical details.

The use of the IODEF format imposes additional requirements on the server. See Section 6.4.

6.2.2. STIX Format

STIX is a structured language for describing a wide range of security resources. STIX approaches the problem with a focus on flexibility, automation, readability, and extensibility.

The use of STIX as the content of an Entry does not impose any additional requirements on ROLIE implementations.

6.3. Use of the rolie:property element

This document provides new registrations for valid rolie:property names. These properties provide optional exposure point for valuable information in the linked content document. Exposing this information in a rolie:property element means that clients do not need to download the linked document to determine if it contains the information they are looking for.

6.3.1. urn:ietf:params:rolie:property:csirt:ID

Provides an exposure point for an identifier from the indicator or incident document. This value SHOULD be a uniquely identifying value for the document linked to in this entry’s atom:content element.
6.4. Additional requirements for use of IODEF

This section provides the normative requirements for usage of the IODEF format. These requirements SHOULD apply when an atom:entry has an IODEF format entity linked to by its atom:content element.

6.4.1. The IODEF Document

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 atom:link rel="history". The associated href MAY leverage OpenSearch to specify the required query.

6.4.2. Category Element

A collection or entry containing IODEF incident content MUST contain at least two additional <atom:category> elements. One category element MUST have the name attribute be equal to ‘urn:ietf:params:rolie:category:csirt:iodef:purpose’ and the other ‘urn:ietf:params:rolie:category:csirt:iodef:restriction’. This metadata provides valuable metadata for searching and organization of IODEF documents.

When the name attribute of this element is ‘urn:ietf:params:rolie:category:csirt:iodef:purpose’, the value attribute MUST be constrained as per section 3.2 of IODEF, e.g. traceback, mitigation, reporting, or other.

When the name attribute of this element is ‘urn:ietf:params:rolie:category:csirt:iodef:restriction’, the value attribute MUST be constrained as per section 3.2 of IODEF, e.g. public, need-to-know, private, default.

6.4.3. Entry Elements

An entry containing an IODEF payload MUST contain a <rolie:property> element with the following requirements:

The "name" attribute is urn:ietf:params:rolie:property:csirt:id.

The "value" attribute 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.

6.4.4. User Authorization

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 6.4.2.

6.4.5. Expectation and Impact Classes

It is frequently the case that an organization will need to triage their investigation and response activities based upon, e.g., the state of the current threat environment, or simply as a result of having limited resources.

In order to enable operators to effectively prioritize their response activity, it is RECOMMENDED that feed implementers provide Atom categories that correspond to the IODEF Expectation and Impact classes. The availability of these feed categories will enable clients to more easily retrieve and prioritize cyber security information that has already been identified as having a specific potential impact, or having a specific expectation.

Support for these categories may also enable efficiencies for organizations that already have established (or plan to establish) operational processes and workflows that are based on these IODEF classes.

6.4.6. Search

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.

7. IANA Considerations

7.1. information-type registrations

IANA has added the following entries to the "ROLIE Security Resource Information Type Sub-Registry" registry located at <https://www.iana.org/assignments/rolie/category/information-type>.

7.1.1. incident information-type

The entry is as follows:

name: incident
index: TBD
reference: This document, Section 3.1

7.1.2. indicator information-type

The entry is as follows:

name: indicator
index: TBD
reference: This document, Section 3.2

7.2. atom:category scheme registrations

IANA has added the following entries to the "ROLIE URN Parameters" registry located in <https://www.iana.org/assignments/rolie/>.

7.2.1. category:csirt:iodef:purpose

The entry is as follows:

name: category:csirt:iodef:purpose

Reference: This document, Section 6.4.2
Subregistry: None
7.2.2. category:csirt:iodef:restriction

The entry is as follows:

name: category:csirt:iodef:restriction

Extension IRI:

Reference: This document, Section 6.4.2

Subregistry: None

7.3. rolie:property name registrations

IANA has added the following entries to the "ROLIE URN Parameters" registry located in <https://www.iana.org/assignments/rolie/>.

7.3.1. property:csirt:id

The entry is as follows:

name: property:csirt:id


Reference: This document, section 6.3.1

Subregistry: None

8. Security Considerations

This document implies the use of ROLIE in high-security use cases, as such, added care should be taken to fortify and secure ROLIE repositories and clients using this extension. The guidance in the ROLIE core specification is strongly recommended, and implementers should consider adding additional security measures as they see fit.

When providing a private workspace for closed sharing, it is recommended that the ROLIE repository checks user authorization when the user sends a GET request to the service document. If the user is not authorized to send any requests to a given workspace or collection, that workspace or collection should be truncated from the service document in the response. In this way the existence of unauthorized content remains unknown to potential attackers, hopefully reducing attack surface.
9. Normative References


Appendix A. Non-Normative Examples

The following provide examples of some potential use cases of the CSIRT ROLIE extension, and provides a showcase for some of its benefits over traditional solutions.

The general non-normative examples provided in the core ROLIE document remain an excellent reference resource for typical ROLIE usage.

A.1. Use of Link Relations

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. Three 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.

A.1.1. Use Case: Incident Sharing

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

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

Example HTTP GET response for an incident entry:
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.
A.1.2. 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">
    <atom:title type="text">RID Use Case Requests</atom:title>
    <collection href="http://www.example.org/csirt/RID/InvestigationRequests">
      <atom:title type="text">Investigation Requests</atom:title>
      <accept>application/atom+xml; type=entry</accept>
    </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"
 xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0">
 <title>New Investigation Request</title>
 <id>http://www.example2.org/csirt/private/incidents/123456</id>
 <!-- id and updated not guaranteed to be preserved -->
 <!-- may want to profile that behavior in this document -->
 <updated>2012-08-12T11:08:22Z</updated>
 <author><name>Name of peer CSIRT</name></author>
 <rolie:format ns="urn:example:iodef"/>
 <content type="application/xml">
  <iodf:IODEF-Document lang="en"
   xmlns:iodf="urn:ietf:params:xml:ns:iodf-1.0">
   <iodf:Incident purpose="traceback" restriction="need-to-know">
    <iodf:IncidentID name="http://www.example2.org/csirt/private/incidents">123</iodf:IncidentID>
    <!-- ...additional incident data.... -->
   </iodf:Incident>
  </iodf: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"
   xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0">
<title>New Investigation Request</title>
<id>http://www.example.org/csirt/RID/InvestigationRequests/823</id>
<!-- id and updated not guaranteed to be preserved -->
<!-- may want to profile that behavior in this document -->
<updated>2012-08-12T11:08:30Z</updated>
<published>2012-08-12T11:08:30Z</published>
<author><name>Name of peer CSIRT</name></author>
<rolie:format ns="urn:example:iodef"/>
<content type="application/xml">
   <iodef:IODEF-Document lang="en"
      xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0">
      <iodef:Incident purpose="traceback" restriction="need-to-know">
         <iodef:IncidentID name="http://www.example.org/csirt/private/incidents">123</iodef:IncidentID>
         <!-- ...additional incident data.... -->
      </iodef:Incident>
   </iodef:IODEF-Document>
</content>
</entry>

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 any as appropriate.

A.1.3. 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, benchmarks, or...
any other related resources. In this use case, the repository is queried via HTTP GET, and the results that are returned to the client may optionally contain URL references to other cyber security resources that are known to be related. These related resources may also be persisted locally, or they may exist at another (remote) cyber data repository.

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

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

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

Example HTTP GET response for a DDOS feed:

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

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0"
      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>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>
  <author>
    <email>csirt@example.org</email>
    <name>EMC CSIRT</name>
  </author>

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

<entry>
  <id>http://www.example.org/csirt/repository/ddos/123456</id>
  <title>Sample DDOS Incident</title>
  <link href="http://www.example.org/csirt/repository/ddos/123456" rel="self"/>
  <link href="http://www.example.org/csirt/repository/ddos/123456" rel="alternate"/>
  <link href="http://www.example.org/csirt/repository/ddos/987654" rel="related"/>
  <link href="http://www.cyber-agency.gov/repository/indicators/1a2b3c" rel="related"/>
  <published>2012-08-04T18:13:51.0Z</published>
  <updated>2012-08-05T18:13:51.0Z</updated>
  <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 the IODEF Incident class, <description> element. </summary>
  <rolie:format ns="urn:example:iodef"/>
  <content href="http://www.example.org/ddos/123456/data"/>
</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 repository is not obligated to follow any particular URL template scheme. The repository available
at the hypothetical provider "www.example.com" uses a different URL pattern than the hypothetical repository available at "www.cyber-agency.gov". When a client de-references a link to resource that is located in a remote repository the client may be challenged for authentication credentials acceptable to that provider. If the two repository providers choose to support a federated identity scheme or some other form of single-sign-on technology, then the user experience can be improved for interactive clients (e.g., a human user at a browser). However, this is not required and is an implementation choice that is out of scope for this specification.

Authors’ Addresses

John P. Field
Pivotal Software, Inc.
625 Avenue of the Americas
New York, New York
USA

Phone: (646)792-5770
Email: jfield@pivotal.io

Stephen A. Banghart
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, Maryland
USA

Phone: (301)975-4288
Email: sab3@nist.gov
Definition of ROLIE CSIRT Extension
draft-banghart-mile-rolie-csirt-03

Abstract

This document extends the Resource-Oriented Lightweight Information Exchange (ROLIE) core to add the information type categories and related requirements needed to support Computer Security Incident Response Team (CSIRT) use cases. The indicator and incident information types are defined as ROLIE extensions. Additional supporting requirements are also defined that describe the use of specific formats and link relations pertaining to the new information types.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction ................................. 3
2. Terminology ................................. 4
3. Additional Requirements for the Atom Publishing Protocol .. 4
  3.1. Use of HTTP requests .................... 4
  3.1.1. / (forward slash) Resource URL ........... 4
4. Additional Requirements for the Atom Syndication Format ... 4
5. Information-type Extensions .................... 4
  5.1. The "incident" information type ............ 4
  5.2. The "indicator" information type .......... 5
  5.3. Use of the rolie:format element .......... 5
  5.3.1. IODEF Format .......................... 6
  5.3.2. STIX Format ........................... 6
6. rolie:property Extensions .................... 6
7. Use of the atom:link element ................ 6
  7.1. Link relations for the 'incident'
       information-type .......................... 7
  7.2. Link relations for the 'indicator'
       information-type ........................... 7
  7.3. Link relations for both information-types .... 8
8. Other Extensions ........................... 8
  8.1. Use of atom:category ...................... 8
  8.1.1. Newly registered category values .......... 8
  8.1.2. Expectation and Impact Classes .......... 9
9. IANA Considerations .......................... 9
  9.1. information-type registrations ............ 9
  9.1.1. incident information-type ............... 9
  9.1.2. indicator information-type .............. 9
  9.2. atom:category scheme registrations .......... 10
  9.2.1. category:csirt:iodef:purpose ............ 10
  9.2.2. category:csirt:iodef:restriction ......... 10
  9.3. rolie:property name registrations .......... 10
  9.3.1. property:csirt:id ...................... 10
10. Security Considerations ..................... 11
11. Normative References ....................... 11
Appendix A. Non-Normative Examples .............. 12
  A.1. Use of Link Relations .................... 12
  A.1.1. Use Case: Incident Sharing ............... 13
  A.1.2. Use Case: Collaborative Investigation .... 15
  A.1.3. Use Case: Cyber Data Repository ......... 17
Authors' Addresses ............................. 20
1. Introduction

Threats to computer security are evolving ever more rapidly as time goes on. As software increases in complexity, the number of vulnerabilities in systems and networks can increase exponentially. Threat actors looking to exploit these vulnerabilities are making more frequent and more widely distributed attacks across a large variety of systems. The adoption of liberal information sharing amongst attackers allows a discovered vulnerability to be shared and used to attack a vulnerable system within a narrow window of time. As the skills and knowledge required to identify and combat these attacks become more and more specialized, even a well established and secure system may find itself unable to quickly respond to an incident. Effective identification of and response to a sophisticated attack requires open cooperation and collaboration between defending operators, software vendors, and end-users. To improve the timeliness of responses, automation must be used to acquire, contextualize, and put to use shared computer security information.

CSIRTS share two primary forms of information: incidents and indicators. Using these forms of information, analysts are able to perform a wide range of activities both proactive and reactive to ensure the security of their systems.

Incident information describes a cyber security incident. Such information may include attack characteristics, information about the attacker, and attack vector data. Sharing this information helps analysts within the sharing community to inoculate their systems against similar attacks, providing proactive protection.

Indicator information describes the symptoms or necessary pre-conditions of an attack. Everything from system vulnerabilities to unexpected network traffic can help analysts secure systems and prepare for an attack. Making this information available for sharing aids in the proactive defense of systems both within an operating unit but also for any CSIRTs that are part of a sharing consortium.

As a means to bring automation of content discovery and dissemination into the CSIRT domain, this specification provides an extension to the Resource-Oriented Lightweight Information Exchange (ROLIE) core [RFC8322] designed to address CSIRT use cases. The primary purpose of this extension is to define two new information types: incident, and indicator, along with formats and link relations that support these information-types.
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. Additional Requirements for the Atom Publishing Protocol

This document specifies the following additional requirements for use of the Atom Publishing Protocol.[RFC5023]

3.1. Use of HTTP requests

This document defines the following requirements on HTTP request behavior:

3.1.1. / (forward slash) Resource URL

The forward slash resource URL SHOULD be supported as defined in Section 5.5 [RFC8322]. Note that this is a stricter requirement than [RFC8322].

4. Additional Requirements for the Atom Syndication Format

This document does not specify any additional requirements for the Atom Syndication Format.  [RFC4287]

5. Information-type Extensions

5.1. The "incident" information type

The "incident" information type represents any information describing or pertaining to a computer security incident. This document uses the definition of incident provided by [RFC4949]. Provided below is a non-exhaustive list of information that may be considered to be an incident information type.

o Timing information: start and end times for the incident and/or the response.

o Descriptive information: plain text or machine readable data that provides some degree of description of the incident itself.
5.2. The "indicator" information type

The "indicator" information type represents computer security indicators or any information surrounding them. This document uses the definition of indicator provided by [RFC4949]. Some examples of indicator information is provided below, but note that indicator is defined in an abstract sense, to be understood as a flexible and widely-applicable definition.

- Specific vulnerabilities that indicate a vector for attack.
- Signs of malicious reconnaissance.
- Definitions of patterns of other indicators.
- Events that may indicate an attack and information regarding those events.
- Meta information about the collecting agent.

This list is intended to provide examples of the indicator information-type, not to define it.

5.3. Use of the rolie:format element

This document does not contain any additional requirements for the rolie:format element; the formats that follow are provided as examples of formats that describe the incident and indicator information type. The formats are in no particular order, and are not requirements, nor suggestions by the authors.
5.3.1. IODEF Format

The Incident Object Description Exchange Format (IODEF) is a format for representing computer security information commonly exchanged between Computer Security Incident Response Teams (CSIRTs) or other operational security teams.

IODEF conveys indicators, incident reports, response activities, and related meta-data in an XML serialization. This information is formally structured in order to support and encourage automated machine-to-machine security communication, as well as enhanced processing at the endpoint.

The full IODEF specification provides further high-level discussion and technical details.

5.3.2. STIX Format

STIX is a structured language for describing a wide range of security resources. STIX approaches the problem with a focus on flexibility, automation, readability, and extensibility.

The use of STIX as the content of an Entry does not impose any additional requirements on ROLIE implementations.

6. rolie:property Extensions

This document provides new registrations for valid rolie:property names. These properties provide optional exposure point for valuable information in the linked content document. Exposing this information in a rolie:property element means that clients do not need to download the linked document to determine if it contains the information they are looking for.


Provides an XML element that can be populated with an identifier from the indicator or incident document linked to by an atom:content element. This value SHOULD be a uniquely identifying value for the document linked to in this entry’s atom:content element.

7. Use of the atom:link element

These sections define requirements for atom:link elements in Entries. Note that the requirements are determined by the information type that appears in either the Entry or in the parent Feed.
7.1. Link relations for the ‘incident’ information-type

If the category of an Entry is the incident information type, then the following requirements MUST be followed for inclusion of atom:link elements.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Conformance</th>
</tr>
</thead>
<tbody>
<tr>
<td>indicators</td>
<td>Provides a link to a collection of zero or more instances of cyber security indicators that are associated with the resource.</td>
<td>SHOULD</td>
</tr>
<tr>
<td>evidence</td>
<td>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.</td>
<td>SHOULD</td>
</tr>
<tr>
<td>attacker</td>
<td>Provides a link to a collection of zero or more resources that provides a representation of the attacker.</td>
<td>SHOULD</td>
</tr>
<tr>
<td>vector</td>
<td>Provides a link to a collection of zero or more resources that provides a representation of the method used by the attacker.</td>
<td>SHOULD</td>
</tr>
</tbody>
</table>

Table 1: Link Relations for Resource-Oriented Lightweight Indicator Exchange

7.2. Link relations for the ‘indicator’ information-type

If the category of an Entry is the indicator information type, then the following requirements MUST be followed for inclusion of atom:link elements.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Conformance</th>
</tr>
</thead>
<tbody>
<tr>
<td>incidents</td>
<td>Provides a link to a collection of zero or more instances of incident representations associated with the resource.</td>
<td>SHOULD</td>
</tr>
</tbody>
</table>

Table 2: Link Relations for Resource-Oriented Lightweight Indicator Exchange
7.3. Link relations for both information-types

If the category of an Entry is either information-type, the following requirements MUST be followed for inclusion of atom:link elements.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Conformance</th>
</tr>
</thead>
<tbody>
<tr>
<td>assessments</td>
<td>Provides a link to a collection of zero or more resources that represent the results of executing a benchmark.</td>
<td>MAY</td>
</tr>
<tr>
<td>reports</td>
<td>Provides a link to a collection of zero or more resources that represent RID reports.</td>
<td>MAY</td>
</tr>
<tr>
<td>traceRequests</td>
<td>Provides a link to a collection of zero or more resources that represent RID traceRequests.</td>
<td>MAY</td>
</tr>
<tr>
<td>investigationRequests</td>
<td>Provides a link to a collection of zero or more resources that represent RID investigationRequests.</td>
<td>MAY</td>
</tr>
</tbody>
</table>

Table 3: Link Relations for Resource-Oriented Lightweight Indicator Exchange

8. Other Extensions

This document defines additional extensions as follows:

8.1. Use of atom:category

8.1.1. Newly registered category values


When the name attribute of the category is ‘urn:ietf:params:rolie:category:csirt:iodef:purpose’, the value attribute SHOULD be constrained as per section 3.2 of IODEF [RFC7970], e.g. traceback, mitigation, reporting, or other.
When the name attribute of the category is
attribute SHOULD be constrained as per section 3.2 of IODEF
[RFC7970], e.g. public, need-to-know, private, default.

8.1.2. Expectation and Impact Classes

It is frequently the case that an organization will need to triage
their investigation and response activities based upon, e.g., the
state of the current threat environment, or simply as a result of
having limited resources.

In order to enable operators to effectively prioritize their response
activity, it is RECOMMENDED that feed implementers provide Atom
categories that correspond to the IODEF Expectation and Impact
classes. The availability of these feed categories will enable
clients to more easily retrieve and prioritize cyber security
information that has already been identified as having a specific
potential impact, or having a specific expectation.

Support for these categories may also enable efficiencies for
organizations that already have established (or plan to establish)
operational processes and workflows that are based on these IODEF
classes.

9. IANA Considerations

9.1. information-type registrations

IANA has added the following entries to the "ROLIE Security Resource
Information Type Sub-Registry" registry located at
<https://www.iana.org/assignments/rolie/category/information-type>
.

9.1.1. incident information-type

The entry is as follows:

name: incident
index: TBD
reference: This document, Section 5.1

9.1.2. indicator information-type

The entry is as follows:

name: indicator
9.2. atom:category scheme registrations

IANA has added the following entries to the "ROLIE URN Parameters" registry located in <https://www.iana.org/assignments/rolie/>.

9.2.1. category:csirt:iodef:purpose

The entry is as follows:

name: category:csirt:iodef:purpose


Reference: This document, Section 8.1.1

Subregistry: None

9.2.2. category:csirt:iodef:restriction

The entry is as follows:

name: category:csirt:iodef:restriction


Reference: This document, Section 8.1.1

Subregistry: None

9.3. rolie:property name registrations

IANA has added the following entries to the "ROLIE URN Parameters" registry located in <https://www.iana.org/assignments/rolie/>.

9.3.1. property:csirt:id

The entry is as follows:

name: property:csirt:id


Reference: This document, section 6.3.1
10. Security Considerations

This document implies the use of ROLIE in high-security use cases, as such, added care should be taken to fortify and secure ROLIE repositories and clients using this extension. The guidance in the ROLIE core specification is strongly recommended, and implementers should consider adding additional security measures as they see fit.

When providing a private workspace for closed sharing, it is recommended that the ROLIE repository checks user authorization when the user sends a GET request to the service document. If the user is not authorized to send any requests to a given workspace or collection, that workspace or collection should be truncated from the service document in the response. In this way the existence of unauthorized content remains unknown to potential attackers, hopefully reducing attack surface.

11. Normative References


Appendix A. Non-Normative Examples

The following provide examples of some potential use cases of the CSIRT ROLIE extension, and provides a showcase for some of its benefits over traditional solutions.

The general non-normative examples provided in the core ROLIE document remain an excellent reference resource for typical ROLIE usage.

A.1. Use of Link Relations

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. Three 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.

A.1.1. Use Case: Incident Sharing

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

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

Example HTTP GET response for an incident entry:
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.
A.1.2. 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">
    <atom:title type="text">RID Use Case Requests</atom:title>
    <collection href="http://www.example.org/csirt/RID/InvestigationRequests">
      <atom:title type="text">Investigation Requests</atom:title>
      <accept>application/atom+xml; type=entry</accept>
    </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"
   xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0">
   <title>New Investigation Request</title>
   <id>http://www.example2.org/csirt/private/incidents/123456</id>
   <!-- id and updated not guaranteed to be preserved -->
   <!-- may want to profile that behavior in this document -->
   <updated>2012-08-12T11:08:22Z</updated>
   <author><name>Name of peer CSIRT</name></author>
   <rolie:format ns="urn:example:iodef"/>
   <content type="application/xml">
      <iodef:IODEF-Document lang="en"
         xmlns:iodef="urn:ietf:params:xml:ns:iodef-1.0">
         <iodef:Incident purpose="traceback" restriction="need-to-know">
            <iodef:IncidentID name="http://www.example2.org/csirt/
                private/incidents">123</iodef:IncidentID>
            <!-- ...additional incident data... -->
         </iodef:Incident>
      </iodef:IODEF-Document>
   </content>
</entry>

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

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

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.

A.1.3. 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, benchmarks, or
In this use case, the repository is queried via HTTP GET, and the results that are returned to the client may optionally contain URL references to other cyber security resources that are known to be related. These related resources may also be persisted locally, or they may exist at another (remote) cyber data repository.

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

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

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

Example HTTP GET response for a DDOS feed:

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

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
     xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0"
     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>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>
  <author>
    <email>csirt@example.org</email>
    <name>EMC CSIRT</name>
  </author>
</feed>

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

Authors’ Addresses

Stephen A. Banghart
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, Maryland
USA

Phone: (301)975-4288
Email: sab3@nist.gov

John P. Field
Pivotal Software, Inc.
625 Avenue of the Americas
New York, New York
USA

Phone: (646)792-5770
Email: jfield@pivotal.io
Abstract

The Incident Object Description Exchange Format v2 [RFC7970] 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 security practitioners to develop tools that can leverage IODEF for incident sharing. This document provides guidelines for IODEF implementers. It also addresses how common security indicators can be represented in IODEF and use-cases of how IODEF is being used. This document aims 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.

Status of This Memo

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

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

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

This Internet-Draft will expire on November 23, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

The Incident Object Description Exchange Format v2 [RFC7970] 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
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 a plethora of data constructs that could potentially make it harder for IODEF implementers to decide which are important. 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, some IODEF classes are useful only in rare circumstances. This document tries to address how to avoid these concerns. It also addresses how common security indicators can be represented in IODEF. It points out the most important IODEF classes for an implementer and describe other ones that are not as important. Also, it presents some common challenges for IODEF implementers and how to address them. The end goal of this document is to make IODEF’s use by vendors easier and encourage wider adoption of the model by 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 a practitioner will come across and how to address them. Section 5 goes over some common uses of IODEF.

2. Terminology

The terminology used in this document follows the one defined in [RFC7970] 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 and Use Strategy

It is important for IODEF implementers to be able to distinguish how the IODEF classes will be used in incident information exchanges. To do that one has to follow a strategy according to which of the various IODEF classes will be implemented. It is also important to know the most common classes that will be used to describe common security incidents or indicators. Thus, this section describes the most important classes and factors an IODEF practitioner should take into consideration before using IODEF, or designing an implementation.
3.1. Minimal IODEF document

IODEF includes one mandatory classes. An IODEF document MUST include at least an Incident class, an xml:lang attribute that defines the supported language an the IODEF version attribute. An Incident MUST contain three minimal mandatory-to-implement classes. An Incident class needs to have a Generation time and IncidentID class and at least one Contact class. The structure of the minimal-style Incident class follows below.

```
+-------------------------+
<table>
<thead>
<tr>
<th>Incident</th>
</tr>
</thead>
</table>
| ENUM purpose            |<>----------[ IncidentID ]
|                         |<>----------[ GenerationTime ]
|                         |<--{1..*}--[ Contact ]
+-------------------------+
```

Minimal-style Incident class

The minimal Incident class needs to include a purpose attribute and the IncidentID, GenerationTime, and Contact elements.

The Contact class requires the type and role attributes, but no elements are required by the IODEF v2 specification. Nevertheless, at least one of the elements in the Contact class, such as Email class, SHOULD be implemented so that the IODEF document can be practical.

Implementers can refer to Appendix B and Section 7 of [RFC7970] for example IODEF v2 documents.

3.2. Information represented

There is no need for a practitioner to use or implement IODEF classes and fields other than the minimal ones (Section 3.1) and the ones that are necessary for her use-cases. The implementer should carefully look into the schema and decide classes to implement (or not).

For example, if we have Distributed Denial of Service (DDoS) as a potential use-case, then the Flow class and its included information are the most important classes to use. The Flow class describes information related to the attacker hosts and victim hosts, which information could help automated filtering or sink-hole operations.

Another potential use-case is malware command and control (c2). After modern malware infects a device, it usually proceeds to connect
to one or more c2 servers to receive instructions from its master and potentially exfiltrate information. To protect against such activity, it is important to interrupt the c2 communication by filtering the activity. IODEF can describe c2 activities using the Flow and the ServiceName classes.

For use-cases where indicators need to be described, the IndicatorData class its classes will be implemented instead of the EventData class.

In summary, an implementer SHOULD identify her use-cases and find the classes that are necessary to support in IODEF v2. Implementing and parsing all IODEF classes can be cumbersome in some occasions and is not always necessary. Other external schemata can also be used in IODEF to describe incidents or indicators which should be treated accordingly only if the implementer’s IODEF use-cases require external schema support.

IODEF supports multiple translations of free-form text in all ML_STRING classes [RFC7970]. That way text can be translated to different languages by using the same translation identifier in the class. Implementers SHOULD be able to parse iodef:MLStringType classes and extract only the information relevant to the language/s of interest.

3.3. IODEF Classes

[RFC7970] contains classes that can describe attack Methods, Events, Incidents, how they were discovered and the Assessment of the repercussions of the incident to the victim. It is important for IODEF users to know the distinction between these classes in order to decide which ones fulfill their use-cases.

An IndicatorData class depicts a threat indicator or observable that could be used to describe a threat that does not necessarily mean that a successful attack happened. For example, we could see an attack happening but it might have been prevented and not have resulted in an incident or security event. On the other hand an EventData class usually describes a security event and can be considered as an incident report of something that took place.

Classes like Discovery, Assessment, Method, and RecoveryTime are used in conjunction with EventData as they related to the incident report described in the EventData. The RelatedActivity class can reference an incident, an indicator or other related threat activity.

While deciding what classes are important for the needed use-cases, IODEF users SHOULD carefully evaluate the necessary classes and how
these are used in order to avoid unnecessary work. For example, if we want to only describe indicators in IODEF, the implementation of Method or Assessment might not be important.

4. Considerations

Implementers need to consider some common, standardized options for their IODEF use strategy.

4.1. External References

The IODEF format includes the Reference class that refers to externally defined information such as a vulnerability, Intrusion Detection System (IDS) alert, malware sample, advisory, or attack technique. To facilitate the exchange of information, the Reference class was extended to the Enumeration Reference Format [RFC7495]. The Enumeration Reference Format specifies a means to use external enumeration specifications (e.g. CVE) that could define an enumeration format, specific enumeration values, or both. As external enumerations can carry greatly, implementers SHOULD only support external enumerations that are expected to describe their specific use-cases.

4.2. Extensions

The IODEF data model ([RFC7970]) is extensible. Many attributes with enumerated values can be extended using the "ext-*" prefix. Additional classes 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]. Information about extending IODEF class attributes and enumerated values can be found in Section 5 of [RFC7970].

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 NOT consider using their own IODEF extensions unless data cannot be represented using existing standards or importing them in an IODEF document using [RFC7203] is not a suitable option.
4.3. Indicator predicate logic

An IODEF [RFC7970] document can describe incident reports and indicators. The Indicator class can include references to other indicators, observables and more classes that contain details about the indicator. When describing security indicators, it is often common to need to group them together in order to form a group of indicators that constitute a security threat. For example, a botnet might have multiple command and control servers. For that reason, IODEF v2 introduced the IndicatorExpression class that is used to add the indicator predicate logic when grouping more than one indicator or observable.

Implementations MUST be able to parse and apply the Boolean logic offered by an IndicatorExpression in order to evaluate the existence of an indicator. As explained in Section 3.29.5 of [RFC7970] the IndicatorExpression element operator defines the operator applied to all the child elements of the IndicatorExpression. If no operator is defined "and" SHOULD be assumed. IndicatorExpressions can also be nested together. Child IndicatorExpressions should be treated as child elements of their parent and they SHOULD be evaluated first before evaluated with the operator of their parent.

Users can refer to Appendix A for example uses of the IndicatorExpressions in an IODEF v2.

4.4. Disclosure level

The information conveyed in IODEF documents SHOULD be treated carefully since the content may be confidential. IODEF has a common attribute, called "restriction", which indicates the disclosure guideline to which the sender expects the recipient to adhere to for the information represented in the class and its children. That way, the sender can express the level of disclosure for each component of an IODEF document. Appropriate external measures could be implemented based on the restriction level. One example is when Real-time Inter-network Defense (RID) [RFC6545] is used to transfer the IODEF documents, it can provide policy guidelines for handling IODEF documents by using the RIDPolicy class.

The enforcement of the disclosure guidelines is out of scope for IODEF. The recipient of the IODEF document needs to follow the guidelines, but these guidelines themselves do not provide any enforcement measures. For that purpose, implementers SHOULD consider appropriate measures, technical or operational.
5. IODEF Uses

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. Implementations

In order to use IODEF, tools like IODEF parsers are necessary. [I-D.ietf-mile-implementreport] describes a set of IODEF implementations and uses by various vendors and Computer Security Incident Response Teams (CERT) organizations. The document does not specify any specific mandatory to implement (MTI) IODEF classes but provides a list of real world uses. Perl and Python modules (XML::IODEF, Iodef::Pb, iodeflib) are some examples. Section 7 also includes practical IODEF use guidelines. Implementers are encouraged to refer to [I-D.ietf-mile-implementreport]. [implementations], on the other hand, includes various vendor incident reporting products that can consume and export in IODEF format.

5.2. Inter-vendor and Service Provider Exercise

As an interoperability exercise, in 2013 a limited number of vendors organized and executed threat indicators exchanges in IODEF. The transport protocol used was RID. The threat information shared included indicators from DDoS attacks; and Malware and Spear-Phishing incidents. The results served as proof-of-concept (PoC) about how seemingly competing entities could use IODEF to exchange sanitized security information. As this was a PoC exercise only example information (no real threats) were shared as part of the exchanges.

| Vendor X | ------- | Vendor Y |
| RID Agent | Internet | RID Agent |

<--- 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-
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, Use of [RFC6545] and [RFC6546], were also proven in this exercise.

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 describe 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 a DDoS attack as explained in the following scenario: 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 CSIT contacts their ISP, Entity Y, and shares information with them about the attack traffic characteristics. Entity 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 describe 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.

(6) Entity Y shares information with other ISP Entities it has an established relationship with.

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 are 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 describe 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.

Appendix B includes some of the incident IODEF example information that was exchanged by the organizations’ RID Agents as part of this proof-of-concept.

5.3. Use-cases

Other use-cases of IODEF, other than the ones described above, 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.
When finding a certificate that should be revoked, a third-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

This document does not incur any new security issues, since it only talks about the usage of IODEFv2 defined RFC7970. Nevertheless, readers of this document SHOULD refer to the Security Considerations section of [RFC7970].

7. Updates

[ EDNOTE: To delete during last call. ]

version -10 updates:

(1) Fixed nits identified by Adam M.

(2) Added paragraph about language support in ML_STRING classes.

version -09 updates:

(1) Made changes according to suggestions in IETF-98.

version -08 updates:

(1) Updated Appendix IODEFv2 examples.

(2) Moved Predicate logic examples in appendix.

(3) Syntax and grammar fixes, clarifications, wording.

(4) Reorganized IODEF uses section and subsections.

version -07 updates:

(1) Updated examples in Appendix A to follow IODEFv2.

version -06 updates:

(1) Updated wording in various sections to make content clearer.
(2) Updated Predicate Logic section to reflect the latest IndicatorExpression logic in iodef-bis.

(3) Updated section to describe the difference between events and indicators and their use in IODEF v2.

version -05 updates:

(1) Changed section title from "Restrictions in IODEF" to "Disclosure level of IODEF" and added some description

(2) Mixed "Recommended classes to implement" section with "Unnecessary Fields" section into "Minimal IODEF document" section

(3) Added description to "Decide what IODEF will be used for" section, "Implementations" section, and "Security Considerations" section

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. 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,

[RFC5901] Cain, P. and D. Jevans, "Extensions to the IODEF-Document
Class for Reporting Phishing", RFC 5901,
DOI 10.17487/RFC5901, July 2010,

[RFC6545] Moriarty, K., "Real-time Inter-network Defense (RID)",
RFC 6545, DOI 10.17487/RFC6545, April 2012,

Incident Object Description Exchange Format (IODEF)
Extension for Structured Cybersecurity Information",
RFC 7203, DOI 10.17487/RFC7203, April 2014,

for the Incident Object Description Exchange Format
(IODEF)", RFC 7495, DOI 10.17487/RFC7495, March 2015,
8.2. Informative References

[I-D.ietf-mile-implementreport]
Inacio, C. and D. Miyamoto, "MILE Implementation Report",
draft-ietf-mile-implementreport-10 (work in progress),
November 2016.

[implementations]
"Implementations on IODEF",

[RFC6546] Trammell, B., "Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS", RFC 6546,
DOI 10.17487/RFC6546, April 2012,

Appendix A. Indicator predicate logic examples

In the following example the EventData class 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.
Similarly, the FileData Class can be an observable in an IndicatorExpression. The hash values of two files can be used to match against an indicator using Boolean "or" logic. In the following example the indicator consists of either of the two files with different hashes.
<IndicatorData>
  <Indicator>
    <IndicatorID name="csirt.example.com" version="1">
      A4399IWQ
    </IndicatorID>
    <Description>File hash watchlist</Description>
    <IndicatorExpression operator="or">
      <Observable>
        <FileData>
          <File>
            <FileName>dummy.txt</FileName>
            <HashData scope="file-contents">
              <Hash>
                <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <ds:DigestValue>
                  141accec23e7e5157de60853cb1e01bc38042d08f9086040815300b7f75c184
                </ds:DigestValue>
              </Hash>
            </HashData>
          </File>
        </FileData>
      </Observable>
      <Observable>
        <FileData>
          <File>
            <FileName>dummy2.txt</FileName>
            <HashData scope="file-contents">
              <Hash>
                <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <ds:DigestValue>
                  141accec23e7e5157de60853cb1e01bc38042d08f9086040815300b7f75c184
                </ds:DigestValue>
              </Hash>
            </HashData>
          </File>
        </FileData>
      </Observable>
    </IndicatorExpression>
  </Indicator>
</IndicatorData>
Appendix B. 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.

B.1. Malware Delivery URL

This example indicates malware and related URL for file delivery.
<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="2.00"
xmlns="urn:ietf:params:xml:ns:iodef-2.0"
xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
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:GenerationTime>2012-12-05T12:20:00+00:00</iodef:GenerationTime>
  <iodef:Description>Malware and related indicators</iodef:Description>
  <iodef:Assessment occurrence="potential">
    <iodef:SystemImpact severity="medium" type="breach-privacy">
      <iodef:Description>Malware with C&C</iodef:Description>
    </iodef:SystemImpact>
  </iodef:Assessment>
  <iodef:Contact role="creator" type="organization">
    <iodef:ContactName>example.com CSIRT</iodef:ContactName>
    <iodef:Email>
      <iodef:EmailTo>contact@csirt.example.com</iodef:EmailTo>
    </iodef:Email>
  </iodef:Contact>
  <iodef:EventData>
    <iodef:Flow>
      <iodef:System category="source">
        <iodef:Node>
          <iodef:Address category="ipv4-addr">192.0.2.200</iodef:Address>
        </iodef:Node>
        <iodef:NodeRole category="www"/>
        <iodef:Address category="site-uri"/>
      </iodef:System>
      /log-bin/lunch_install.php?aff_id=1&amp;lunch_id=1&amp;maddr=&amp;action=install
    </iodef:Flow>
  </iodef:EventData>
</iodef:Incident>
</IODEF-Document>

B.2. 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
[RFC7970].

<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="2.00"
    xmlns="urn:ietf:params:xml:ns:iodef-2.0"
    xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <iodef:Incident purpose="reporting" restriction="default">
        <iodef:IncidentID name="csirt.example.com">
            189701
        </iodef:IncidentID>
        <iodef:DetectTime>2013-02-05T01:15:45+00:00</iodef:DetectTime>
        <iodef:StartTime>2013-02-05T00:34:45+00:00</iodef:StartTime>
        <iodef:ReportTime>2013-02-05T01:34:45+00:00</iodef:ReportTime>
        <iodef:GenerationTime>2013-02-05T01:15:45+00:00</iodef:GenerationTime>
        <iodef:Description>DDoS Traffic Seen</iodef:Description>
        <iodef:Assessment occurrence="actual">
            <iodef:SystemImpact severity="medium" type="availability-system">
                <iodef:Description>DDoS Traffic</iodef:Description>
            </iodef:SystemImpact>
            <iodef:Confidence rating="high"/>
        </iodef:Assessment>
        <iodef:Contact role="creator" type="organization">
            <iodef:ContactName>Dummy Test</iodef:ContactName>
            <iodef:Email>
                <iodef:EmailTo>contact@dummytest.com</iodef:EmailTo>
            </iodef:Email>
        </iodef:Contact>
        <iodef:EventData>
            <iodef:Description>Dummy Test sharing with ISP1</iodef:Description>
            <iodef:Method>
                <iodef:Reference>
                    <iodef:URL>
                    </iodef:URL>
                </iodef:Reference>
                <iodef:Reference>
                    <iodef:URL>
                        http://en.wikipedia.org/wiki/Low_Orbit_Ion_Cannon
                    </iodef:URL>
                    <iodef:Description>
                        Low Orbit Ion Cannon User Agent
                    </iodef:Description>
                </iodef:Reference>
            </iodef:Method>
        </iodef:EventData>
    </iodef:Incident>
</IODEF-Document>
<iodef:Service ip-protocol="6">
  <iodef:Port>80</iodef:Port>
</iodef:Service>
</iodef:System>
<iodef:System category="sensor">
  <iodef:Node>
    <iodef:Node>
      <iodef:Description>
        Information provided in Flow class instance is from inspection of traffic from network tap
      </iodef:Description>
    </iodef:Node>
  </iodef:Node>
</iodef:System>
</iodef:Flow>
</iodef:EventData>
</iodef:IndicatorData>
</iodef:Indicator>
</iodef:IndicatorID name="csirt.example.com" version="1">
  G83345941
</iodef:IndicatorID>
</iodef:Description>
</iodef:Description>
</iodef:Observable>
</iodef:BulkObservable type="http-user-agent">
  <iodef:BulkObservableList>
    user-agent="Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10.5; en-US; rv:1.9.2.12) Gecko/20100126 Firefox/3.6.12">
  </iodef:BulkObservableList>
</iodef:BulkObservable>
</iodef:Observable>
</iodef:Indicator>
</iodef:IndicatorData>
</iodef:Incident>
</IODEF-Document>

B.3. 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 [RFC7970].

<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="2.00"
  xmlns="urn:ietf:params:xml:ns:iodef-2.0"
  xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
  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:DetectTime>2013-01-04T08:06:12+00:00</iodef:DetectTime>
  <iodef:StartTime>2013-01-04T08:01:34+00:00</iodef:StartTime>
  <iodef:EndTime>2013-01-04T08:31:27+00:00</iodef:EndTime>
  <iodef:ReportTime>2013-01-04T09:15:45+00:00</iodef:ReportTime>
  <iodef:GenerationTime>2013-01-04T09:15:45+00:00</iodef:GenerationTime>
  <iodef:Description>
    Zeus Spear Phishing E-mail with Malware Attachment
  </iodef:Description>
  <iodef:Assessment occurrence="potential">
    <iodef:SystemImpact severity="medium" type="takeover-system">
      <iodef:Description>
        Malware with Command and Control Server and System Changes
      </iodef:Description>
    </iodef:SystemImpact>
  </iodef:Assessment>
  <iodef:Contact role="creator" type="organization">
    <iodef:ContactName>example.com CSIRT</iodef:ContactName>
    <iodef:Email>
      <iodef:EmailTo>contact@csirt.example.com</iodef:EmailTo>
    </iodef:Email>
  </iodef:Contact>
  <iodef:EventData>
    <iodef:Description>
      Targeting Defense Contractors,
specifically board members attending Dummy Con
    </iodef:Description>
    <iodef:Method>
      <iodef:Reference observable-id="ref-1234">
        <iodef:Description>Zeus</iodef:Description>
      </iodef:Reference>
    </iodef:Method>
    <iodef:Flow>
      <iodef:System category="source">
        <iodef:Node>
          <iodef:Address category="site-uri">
            http://www.zeusevil.example.com
          </iodef:Address>
          <iodef:Address category="ipv4-addr">
            192.0.2.166
          </iodef:Address>
          <iodef:Address category="asn">
            65535
          </iodef:Address>
        </iodef:Node>
      </iodef:System category="source">
    </iodef:Flow>
  </iodef:EventData>
</iodef:Incident>
<iodef:Address category="ext-value"
    ext-category="as-name">
    EXAMPLE-AS - University of Example
</iodef:Address>

<iodf:Flow>
    <iodef:System category="source">
        <iodef:Node>
            <iodef:DomainData>
                <Name>mail1.evildave.example.com</Name>
            </iodef:DomainData>
            <iodef:Address category="ipv4-addr">
                198.51.100.6
            </iodef:Address>
            <iodef:Address category="asn">
                65534
            </iodef:Address>
            <iodef:Address category="ext-value"
                ext-category="as-name">
                EXAMPLE-AS - University of Example
            </iodef:Address>
            <iodef:DomainData>
                <iodef:Name>evildave.example.com</iodef:Name>
                <iodef:DateDomainWasChecked>2013-01-04T09:10:24+00:00</iodef:DateDomainWasChecked>
            </iodef:DomainData>
            <iodef:RelatedDNS dtype="string">
                evildave.example.com MX prefernce = 10, mail exchanger = mail1.evildave.example.com
            </iodef:RelatedDNS>
            <iodef:RelatedDNS dtype="string">
                mail1.evildave.example.com internet address = 198.51.100.6
            </iodef:RelatedDNS>
            <iodef:RelatedDNS dtype="string">
                zuesevil.example.com. IN TXT "v=spf1 a mx -all"
            </iodef:RelatedDNS>
        </iodef:Node>
        <iodef:NodeRole category="mail">
            <iodef:Description>
                Sending phishing mails
            </iodef:Description>
        </iodef:NodeRole>
    </iodef:System>
</iodef:Flow>

<iodef:System category="source">
    <iodef:Node>
        <iodef:DomainData>
            <Name>mail1.evildave.example.com</Name>
        </iodef:DomainData>
        <iodef:Address category="ipv4-addr">
            192.0.2.0/24
        </iodef:Address>
        <iodef:NodeRole category="malware-distribution"/>
    </iodef:Node>
</iodef:System>
</iodef:Flow>
<iodef:Description>
</iodef:Description>

</iodef:NodeRole>

</iodef:Service>

</iodef:EmailData>

</iodef:Service>

</iodef:EmailFrom>

emaildave@evildave.example.com
</iodef:EmailFrom>

</iodef:EmailSubject>

Join us at Dummy Con
</iodef:EmailSubject>

</iodef:EmailSubject>

</iodef:EmailX-Mailer>

StormRider 4.0
</iodef:EmailX-Mailer>
</iodef:EmailData>

</iodef:Service>

</iodef:System category="target">

</iodef:Node>

</iodef:Address category="ipv4-addr">

203.0.113.2
</iodef:Address>
</iodef:Node>
</iodef:System>
</iodef:Flow>

</iodef:Expectation action="other"/>

</iodef:Record>

</iodef:RecordData>

</iodef:FileData observable-id="fd-1234">

</iodef:File>

</iodef:FileName>

Dummy Con Sign Up Sheet.txt
</iodef:FileName>

</iodef:FileSize>

152
</iodef:FileSize>

</iodef:HashData scope="file-contents">

</iodef:Hash>

<ds:DigestMethod
 Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>

<ds:DigestValue>
141accecc23e7e5157de60853cb1e01bc38042d08f9086040815300b7fe75c184
</ds:DigestValue>
</iodef:Hash>
</iodef:RecordData>
</iodef:RecordData>
B.4. 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-Document version="2.00"
xmlns="urn:ietf:params:xml:ns:iodef-2.0"
xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
xmils:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmils:ds="http://www.w3.org/2000/09/xmldsig#">
    <iodef:Incident purpose="reporting">
        <iodef:IncidentID name="csirt.example.com">
            189234
        </iodef:IncidentID>
        <iodef:GenerationTime>2013-03-07T16:14:56.757+05:30</iodef:GenerationTime>
        <iodef:Description>
            Malware and related indicators identified
        </iodef:Description>
        <iodef:Assessment occurrence="potential">
            <iodef:SystemImpact severity="medium" type="breach-proprietary">
                Malware with Command and Control Server and System Changes
            </iodef:Description>
        </iodef:Assessment>
    </iodef:Incident>
</IODEF-Document>
Internet-Draft               IODEF Guidance                     May 2017

</iodef:SystemImpact>
</iodef:Assessment>
</iodef:Contact role="creator" type="organization">
</iodef:ContactName>example.com CSIRT</iodef:ContactName>
</iodef:Email>
</iodef:Email>
</iodef:Email>
</iodef:Contact>
</iodef:EventData>
</iodef:Method>
</iodef:Reference>
</iodef:URL>
</iodef:Description>Zeus</iodef:Description>
</iodef:Reference>
</iodef:EventData>
</iodef:Flow>
</iodef:System>
</iodef:Flow>
</iodef:RecordData>
</iodef:File>
</iodef:File>
</iodef:EventData>
</iodef:Flow>
</iodef:RecordData>
</iodef:File>
</iodef:File>
</iodef:Method>
</iodef:URL>
</iodef:Description>Zeus</iodef:Description>
</iodef:Reference>
</iodef:System>
</iodef:Flow>
</iodef:RecordData>
</iodef:File>
</iodef:File>
</iodef:Method>
</iodef:URL>
</iodef:Description>Zeus</iodef:Description>
</iodef:Reference>
</iodef:System>
</iodef:Flow>
</iodef:RecordData>
</iodef:File>
</iodef:File>

<iodef:HashData scope="file-contents">
  <iodef:Hash>
    <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#md5"/>
    <ds:DigestValue>
      MHgyRTg4ODA5ODBENjI0NDdFOTc5MEFGQTg5NTEzRjBBNA==
    </ds:DigestValue>
  </iodef:Hash>
</iodef:HashData>

<iodef:File>
  <iodef:FileData>
    <iodef:WindowsRegistryKeysModified observable-id="regkey-91011-001">
      <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:FileData>
</iodef:File>

<iodef:EventData>
  <iodef:Method>
    <iodef:Reference>
      <iodef:URL>
        http://www.threatexpert.example.com/report.aspx?md5=c3c528c939f9b176c883ae0ce5df0001
      </iodef:URL>
      <iodef:Description>Cridex</iodef:Description>
    </iodef:Reference>
  </iodef:Method>
  <iodef:Flow>
    <iodef:System category="source">
      <iodef:Node>
        <iodef:Address category="ipv4-addr" observable-id="addr-c2-91011-003">
          203.0.113.100
        </iodef:Address>
      </iodef:Node>
    </iodef:System>
  </iodef:Flow>
</iodef:EventData>
<iodef:Node>
  <iodef:NodeRole category="c2-server"/>
  <iodef:Service ip-protocol="6">
    <iodef:Port>8080</iodef:Port>
  </iodef:Service>
</iodef:System>
</iodef:Flow>
<iodef:Record>
  <iodef:RecordData>
    <iodef:FileData observable-id="file-91011-002">
      <iodef:File>
        <iodef:HashData scope="file-contents">
          <iodef:Hash>
            <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha1"/>
            <ds:DigestValue>
              MHg3MjYzRkUwRDNBMDk1RDU5QzhFMEM4OTBOUM1ODVFMzQzRTcxNDFD
            </ds:DigestValue>
          </iodef:Hash>
        </iodef:FileData>
      </iodef:File>
    </iodef:FileData>
    <iodef:FileData observable-id="file-91011-003">
      <iodef:File>
        <iodef:HashData scope="file-contents">
          <iodef:Hash>
            <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#md5"/>
            <ds:DigestValue>
              MHg0M0NEODUwRkNEQURFNDMzMEE1QkVBNkYxNkVFOTcxQw==
            </ds:DigestValue>
          </iodef:Hash>
        </iodef:FileData>
      </iodef:File>
    </iodef:FileData>
    <iodef:WindowsRegistryKeysModified observable-id="regkey-91011-002">
      <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:IndicatorData>
<iodef:Indicator>
  <iodef:IndicatorID name="csirt.example.com" version="1">
    ind-91011
  </iodef:IndicatorID>
  <iodef:Description>
    evil c2 server, file hash, and registry key
  </iodef:Description>
  <iodef:IndicatorExpression operator="or">
    <iodef:Observable>
      <iodef:Address category="site-uri" observable-id="addr-qrst">
        http://foo.example.com:12345/evil/cc.php
      </iodef:Address>
    </iodef:Observable>
    <iodef:Observable>
      <iodef:Address category="ipv4-addr" observable-id="addr-stuv">
        192.0.2.1
      </iodef:Address>
    </iodef:Observable>
    <iodef:Observable>
      <iodef:Address category="ipv4-addr" observable-id="addr-tuvw">
        198.51.100.1
      </iodef:Address>
    </iodef:Observable>
    <iodef:Observable>
      <iodef:Address category="ipv6-addr" observable-id="addr-uvwx">
        2001:db8:dead:beef::1
      </iodef:Address>
    </iodef:Observable>
    <iodef:ObservableReference uid-ref="addr-c2-91011-001"/>
    <iodef:ObservableReference uid-ref="addr-c2-91011-002"/>
    <iodef:ObservableReference uid-ref="addr-c2-91011-003"/>
  </iodef:IndicatorExpression>
  <iodef:IndicatorExpression operator="and">
    <iodef:Observable>
      <iodef:FileData observable-id="file-91011-000">
        <iodef:File>
          <iodef:HashData scope="file-contents">
            <iodef:Hash>
              <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlns#sha256"/>
              <ds:DigestValue>
                141accec23e7e5157de60853cbe01bc38042d08f9086040815300b7fe75c184
              </ds:DigestValue>
            </iodef:Hash>
          </iodef:HashData>
        </iodef:File>
      </iodef:FileData>
    </iodef:Observable>
  </iodef:IndicatorExpression>
<iodef:Observable>
  <iodef:WindowsRegistryKeysModified observable-id="regkey-91011-000">
    <iodef:Key registryaction="add-key"
      observable-id="regkey-vwxy">
      <iodef:KeyName>
        HKLM\SYSTEM\CurrentControlSet\Services\.Net CLR
      </iodef:KeyName>
    </iodef:Key>
    <iodef:Key registryaction="add-key"
      observable-id="regkey-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"
      observable-id="regkey-xyza">
      <iodef:KeyName>
        HKLM\SYSTEM\CurrentControlSet\Services\ServiceDll
      </iodef:KeyName>
      <iodef:Value>C:\bad.exe</iodef:Value>
    </iodef:Key>
    <iodef:Key registryaction="modify-value"
      observable-id="regkey-zabc">
      <iodef:KeyName>
        HKLM\SYSTEM\CurrentControlSet\Services\Parameters\Bar
      </iodef:KeyName>
      <iodef:Value>Baz</iodef:Value>
    </iodef:Key>
  </iodef:WindowsRegistryKeysModified>
</iodef:Observable>

<iodef:IndicatorExpression operator="or">
  <iodef:IndicatorExpression operator="and">
    <iodef:ObservableReference uid-ref="file-91011-001"/>
    <iodef:ObservableReference uid-ref="regkey-91011-001"/>
  </iodef:IndicatorExpression>
  <iodef:IndicatorExpression operator="and">
    <iodef:ObservableReference uid-ref="file-91011-002"/>
    <iodef:ObservableReference uid-ref="file-91011-003"/>
  </iodef:IndicatorExpression>
</iodef:IndicatorExpression>
B.5. IoT Malware

The IoT Malware test exchanged information that described a bad IP address of IoT malware and its scanned ports. This example information is extracted from alert messages of a Darknet monitoring system referred in [I-D.ietf-mile-implementreport]. The IODEF version used for the data representation was based on [RFC7970].
</iodef:Discovery>
<iodef:Flow>
  <iodef:System category="source">
    <iodef:Node>
      <iodef:Address category="ipv4-addr">
        192.0.2.210
      </iodef:Address>
    </iodef:Node>
    <iodef:NodeRole category="camera"/>
    <iodef:Service ip-protocol="6">
      <iodef:Port>23</iodef:Port>
    </iodef:Service>
    <iodef:OperatingSystem>
      <iodef:Description>
        Example Surveillance Camera OS 2.1.1
      </iodef:Description>
    </iodef:OperatingSystem>
  </iodef:System>
</iodef:Flow>
</iodef:EventData>

<iodef:Flow>
  <iodef:System category="target">
    <iodef:Node>
      <iodef:Address category="ipv4-addr">
        198.51.100.1
      </iodef:Address>
    </iodef:Node>
    <iodef:NodeRole category="honeypot"/>
    <iodef:Service ip-protocol="6">
      <iodef:Port>23</iodef:Port>
    </iodef:Service>
  </iodef:System>
</iodef:Flow>
</iodef:EventData>

<iodef:Flow>
  <iodef:System category="target">
    <iodef:Node>
      <iodef:Address category="ipv4-addr">
        198.51.100.94
      </iodef:Address>
    </iodef:Node>
    <iodef:NodeRole category="honeypot"/>
    <iodef:Service ip-protocol="6">
      <iodef:Port>23</iodef:Port>
    </iodef:Service>
  </iodef:System>
</iodef:Flow>
</iodef:EventData>
<iodef:EventData>
<iodef:Flow>
<iodef:System category="target">
<iodef:Node>
<iodef:Address category="ipv4-addr">
198.51.100.237
</iodef:Address>
</iodef:Node>
<iodef:NodeRole category="honeypot"/>
<iodef:Service ip-protocol="6">
<iodef:Port>2323</iodef:Port>
</iodef:Service>
</iodef:System>
</iodef:Flow>
</iodef:EventData>
</iodef:EventData>
</iodef:EventData>
</iodef:Incident>
</IODEF-Document>

Authors’ Addresses

Panos Kampanakis
Cisco Systems

Email: pkampana@cisco.com

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

Email: mio@nict.go.jp
Incident Object Description Exchange Format Usage Guidance

draft-ietf-mile-iodef-guidance-11

Abstract

The Incident Object Description Exchange Format (IODEF) v2 (RFC7970) defines a data representation that provides a framework for sharing information about computer security incidents commonly exchanged by Computer Security Incident Response Teams (CSIRTs). 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 security practitioners to develop tools that leverage IODEF for incident sharing. This document provides guidelines for IODEF implementers. It addresses how common security indicators can be represented in IODEF and use-cases of how IODEF is being used. This document aims to make IODEF’s adoption by vendors easier and encourage faster and wider adoption of the model by CSIRTs around the world.

Status of This Memo

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

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

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

This Internet-Draft will expire on March 11, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

The Incident Object Description Exchange Format (IODEF) v2 [RFC7970] defines a data representation that provides a framework for sharing computer security incident information commonly exchanged by Computer Security Incident Response Teams (CSIRTs). The IODEF data model
consists of multiple classes and data types that are defined in the IODEF XML schema.

The IODEF schema was designed to describe all the possible fields needed in a security incident exchange. Thus, IODEF contains a plethora of data constructs which could make it hard for IODEF implementers to decide which are important. 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, some IODEF classes are useful only in rare circumstances. This document tries to address these concerns. It also presents how common security indicators can be represented in IODEF. It points out the most important IODEF classes for an implementer and describes other ones that are not as important. Also, it presents some common pitfalls for IODEF implementers and how to address them. The end goal of this document is to make IODEF’s use by vendors easier and encourage wider adoption of the model by CSIRTs around the world.

Section 3 discusses the recommended classes and how an IODEF implementer should choose the classes to implement. Section 4 presents common considerations a practitioner will come across and how to address them. Section 5 goes over some common uses of IODEF.

2. Terminology

The terminology used in this document follows the one defined in [RFC7970] and [RFC7203].

3. Implementation and Use Strategy

It is important for IODEF implementers to distinguish how the IODEF classes will be used in incident information exchanges. It is also important to understand the most common IODEF classes that describe common security incidents or indicators. This section describes the most important classes and factors an IODEF practitioner should take into consideration before using IODEF or designing an implementation.

3.1. Minimal IODEF document

An IODEF document must include at least an Incident class, an xml:lang attribute that defines the supported language and the IODEF version attribute. An Incident must contain a purpose attribute and three mandatory-to-implement elements. These elements are Generation time class that describes the time of the incident, an IncidentID class and at least one Contact class. The structure of the minimal IODEF-Document is shown in Figure 1.
Figure 1: Minimal IODEF-Document class

The IncidentID class must contain at least a name attribute.

In turn, the Contact class requires the type and role attributes, but no elements are required by the IODEF v2 specification. Nevertheless, at least one of the elements in the Contact class, such as an Email class, should be implemented so that the IODEF document is useful.

Section 7.1 of [RFC7970] presents a minimal IODEF document with only the mandatory classes and attributes. Implementers can also refer to Section 7 of [RFC7970] and Appendix B for example IODEF v2 documents.

3.2. Information represented

There is no need for a practitioner to use or implement IODEF classes and fields other than the minimal ones (Section 3.1) and the ones necessary for her use-cases. The implementer should carefully look into the schema and decide which classes to implement (or not).

For example, if we have Distributed Denial of Service (DDoS) as a potential use-case, then the Flow class and its included information are the most important classes to use. The Flow class describes information related to the attacker and victim hosts, which information could help automated filtering or sink-hole operations.

Another potential use-case is malware command and control (c2). After modern malware infects a device, it usually proceeds to connect to one or more c2 servers to receive instructions from its master and potentially exfiltrate information. To protect against such
activity, it is important to interrupt the c2 communication by filtering the activity. IODEF can describe c2 activities using the Flow and the ServiceName classes.

For use-cases where indicators need to be described, the IndicatorData class will be implemented instead of the EventData class.

In summary, an implementer should identify her use-cases and find the classes that are necessary to support in IODEF v2. Implementing and parsing all IODEF classes can be cumbersome in some occasions and unnecessary. Other external schemata can also be used in IODEF to describe incidents or indicators. External schemata should be parsed accordingly only if the implementer’s IODEF use-cases require external schema information. But even when an IODEF implementation cannot parse an external schema, the IODEF report can still be valuable to an incident response team. The information can also be useful when shared further with content consumers able to parse this information.

IODEF supports multiple language translations of free-form, ML_STRING text in all classes [RFC7970]. That way, text in Description elements can be translated to different languages by using a translation identifier in the class. Implementers should be able to parse iodef:MLStringType classes and extract only the information relevant to languages of interest.

3.3. IODEF Classes

[RFC7970] contains classes that can describe attack Methods, Events, Incidents, Indicators, how they were discovered and the Assessment of the repercussions for the victim. It is important for IODEF users to know the distinction between these classes in order to decide which ones fulfill their use-cases.

An IndicatorData class depicts a threat indicator or observable that could be used to describe a threat that resulted in an attempted attack. For example, we could see an attack happening but it might have been prevented and not have resulted in an incident or security event. On the other hand, an EventData class usually describes a security event and can be considered as a report of something that took place.

Classes like Discovery, Assessment, Method, and RecoveryTime are used in conjunction with EventData as they related to the incident report described in the EventData. The RelatedActivity class can reference an incident, an indicator or other related threat activity.
While deciding what classes are important for the needed use-cases, IODEF users should carefully evaluate the necessary classes and how these are used in order to avoid unnecessary work. For example, if we want to only describe indicators in IODEF, the implementation of Method or Assessment might not be important.

4. IODEF usage considerations

Implementers need to consider some common, standardized options for their IODEF use strategy.

4.1. External References

The IODEF format includes the Reference class used for externally defined information such as a vulnerability, Intrusion Detection System (IDS) alert, malware sample, advisory, or attack technique. To facilitate the exchange of information, the Reference class was extended to the Enumeration Reference Format [RFC7495]. The Enumeration Reference Format specifies a means to use external enumeration specifications (e.g. CVE) that could define an enumeration format, specific enumeration values, or both. As external enumerations can vary greatly, implementers should only support the ones expected to describe their specific use-cases.

4.2. Extensions

The IODEF data model ([RFC7970]) is extensible. Many attributes with enumerated values can be extended using the "ext-*" prefix. Additional classes 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]. Information about extending IODEF class attributes and enumerated values can be found in Section 5 of [RFC7970].

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. [RFC7203] enhances the IODEF capabilities without further extending the data model.

IODEF implementers should not use their own IODEF extensions unless data cannot be represented using existing standards or importing them in an IODEF document using [RFC7203] is not a suitable option.
4.3. Indicator predicate logic

An IODEF [RFC7970] document can describe incident reports and indicators. The Indicator class can include references to other indicators, observables and more classes that contain details about the indicator. When describing security indicators, it is often common to need to group them together in order to form a group of indicators that constitute a security threat. For example, a botnet might have multiple command and control servers. For that reason, IODEF v2 introduced the IndicatorExpression class that is used to add the indicator predicate logic when grouping more than one indicators or observables.

Implementations must be able to parse and apply the Boolean logic offered by an IndicatorExpression in order to evaluate the existence of an indicator. As explained in Section 3.29.5 of [RFC7970] the IndicatorExpression element operator defines the operator applied to all the child element of the IndicatorExpression. If no operator is defined "and" should be assumed. IndicatorExpressions can also be nested together. Child IndicatorExpressions should be treated as child elements of their parent and they should be evaluated first before evaluated with the operator of their parent.

Users can refer to Appendix A for example uses of the IndicatorExpressions in an IODEF v2.

4.4. Disclosure level

Access to information in IODEF documents should be tightly locked since the content may be confidential. IODEF has a common attribute, called "restriction", which indicates the disclosure guideline to which the sender expects the recipient to adhere to for the information represented in the class and its children. That way, the sender can express the level of disclosure for each component of an IODEF document. Appropriate external measures could be implemented based on the restriction level. One example is when Real-time Inter-network Defense (RID) [RFC6545] is used to transfer the IODEF documents, it can provide policy guidelines for handling IODEF documents by using the RIDPolicy class.

The enforcement of the disclosure guidelines is out of scope for IODEF. The recipient of the IODEF document needs to follow the guidelines, but these guidelines themselves do not provide any enforcement measures. For that purpose, implementers should consider appropriate privacy control measures, technical or operational for their implementation.
5. IODEF Uses

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. Implementations

In order to use IODEF, tools like IODEF parsers are necessary. [RFC8134] describes a set of IODEF implementations and uses by various vendors and Computer Emergency Readiness Team (CERT) organizations. The document does not specify any specific mandatory to implement (MTI) IODEF classes but provides a list of real world uses. Perl and Python modules (XML::IODEF, Iodef::Pb, iodeflib) are some examples. Moreover, implementers are encouraged to refer to Section 7 of [RFC8134] practical IODEF usage guidelines. [implementations], on the other hand, includes various vendor incident reporting products that can consume and export in IODEF format.

5.2. Inter-vendor and Service Provider Exercise

As an interoperability exercise, in 2013 a limited number of vendors organized and executed threat indicators exchanges in IODEF. The transport protocol used was RID. The threat information shared included indicators from DDoS attacks; and Malware incidents and Spear-Phishing that targets specific individuals after harvesting information about them. The results served as proof-of-concept (PoC) about how seemingly competing entities could use IODEF to exchange sanitized security information. As this was a PoC exercise only example information (no real threats) were shared as part of the exchanges.

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

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

Figure 2: PoC peering topology
Figure 2 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 implementing [RFC6545] and [RFC6546] was also confirmed.

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 digital 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 describe 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 a DDoS attack as explained in the following scenario: 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 CSIT
contacts their ISP, Entity Y, and shares information with them about the attack traffic characteristics. Entity 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 digital certificates.

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

(6) Entity Y shares information with other ISP Entities it has an established relationship with.

One more use-case was sharing spear-phishing email information as explained in the following scenario: The board members of several defense contractors receive a targeted 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 are 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 phising 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 digital certificates.

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

Appendix B includes some of the incident IODEF example information that was exchanged by the organizations’ RID Agents as part of this proof-of-concept.

5.3. Use-cases

Other use-cases of IODEF, other than the ones described above, 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 third-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. IANA Considerations

This memo does not require any IANA actions.

7. Security Considerations

This document does not incur any new security issues, since it only talks about the usage of IODEFv2 defined RFC7970. Nevertheless, readers of this document should refer to the Security Considerations section of [RFC7970].

8. References

8.1. Normative References


8.2. Informative References

[implementations]
"Implementations on IODEF",

[RFC6546] Trammell, B., "Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS", RFC 6546,
DOI 10.17487/RFC6546, April 2012,

RFC 8134, DOI 10.17487/RFC8134, May 2017,

Appendix A. Indicator predicate logic examples

In the following example the EventData class evaluates as a Flow of one System with source address being (192.0.2.104 OR 192.0.2.106) AND target address 198.51.100.1.
Similarly, the FileData Class can be an observable in an IndicatorExpression. The hash values of two files can be used to match against an indicator using Boolean "or" logic. In the following example the indicator consists of either of the two files with different hashes.
<!-- ...XML code omitted... -->
<IndicatorData>
  <Indicator>
    <IndicatorID name="csirt.example.com" version="1">
      A4399IWQ
    </IndicatorID>
    <Description>File hash watchlist</Description>
    <IndicatorExpression operator="or">
      <Observable>
        <FileData>
          <File>
            <FileName>dummy.txt</FileName>
            <HashData scope="file-contents">
              <Hash>
                <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <ds:DigestValue>141accec23e7e5157de60853cb1e01bc38042d08f9086040815300b7fe75c184</ds:DigestValue>
              </Hash>
            </HashData>
          </File>
        </FileData>
      </Observable>
      <Observable>
        <FileData>
          <File>
            <FileName>dummy2.txt</FileName>
            <HashData scope="file-contents">
              <Hash>
                <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <ds:DigestValue>141accec23e7e5157de60853cb1e01bc38042d08f9086040815300b7fe75c184</ds:DigestValue>
              </Hash>
            </HashData>
          </File>
        </FileData>
      </Observable>
    </IndicatorExpression>
  </Indicator>
</IndicatorData>
<!-- ...XML code omitted... -->
Appendix B. 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.

B.1. Malware Delivery URL

This example indicates malware and related URL for file delivery.
<IODEF-Document version="2.00"
     xmlns="urn:ietf:params:xml:ns:iodef-2.0"
     xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
     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:GenerationTime>2012-12-05T12:20:00+00:00</iodef:GenerationTime>
       <iodef:Description>Malware and related indicators</iodef:Description>
       <iodef:Assessment occurrence="potential">
         <iodef:SystemImpact severity="medium" type="breach-privacy">
           <iodef:Description>Malware with C&amp;C</iodef:Description>
         </iodef:SystemImpact>
       </iodef:Assessment>
       <iodef:Contact role="creator" type="organization">
         <iodef:ContactName>example.com CSIRT</iodef:ContactName>
         <iodef:Email>
           <iodef:EmailTo>contact@csirt.example.com</iodef:EmailTo>
         </iodef:Email>
       </iodef:Contact>
       <iodef:EventData>
         <iodef:Flow>
           <iodef:System category="source">
             <iodef:Node>
               <iodef:Address category="ipv4-addr">192.0.2.200</iodef:Address>
             </iodef:Node>
             <iodef:Address category="site-uri">
               /log-bin/lunch_install.php?aff_id=1&amp;lunch_id=1&amp;maddr=&amp;action=install
             </iodef:Address>
           </iodef:System>
         </iodef:Flow>
       </iodef:EventData>
     </iodef:Incident>

   </IODEF-Document>

B.2. 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
[RFC7970].

<?xml version="1.0" encoding="UTF-8"?>
<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="2.00"
 xmlns="urn:ietf:params:xml:ns:iodef-2.0"
 xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <iodef:Incident purpose="reporting" restriction="default">
  <iodef:IncidentID name="csirt.example.com">189701</iodef:IncidentID>
  <iodef:DetectTime>2013-02-05T01:15:45+00:00</iodef:DetectTime>
  <iodef:StartTime>2013-02-05T00:34:45+00:00</iodef:StartTime>
  <iodef:ReportTime>2013-02-05T01:34:45+00:00</iodef:ReportTime>
  <iodef:GenerationTime>2013-02-05T01:15:45+00:00</iodef:GenerationTime>
  <iodef:Description>DDoS Traffic Seen</iodef:Description>
  <iodef:Assessment occurrence="actual">
    <iodef:SystemImpact severity="medium" type="availability-system">
      <iodef:Description>DDoS Traffic</iodef:Description>
    </iodef:SystemImpact>
    <iodef:Confidence rating="high"/>
  </iodef:Assessment>
  <iodef:Contact role="creator" type="organization">
    <iodef:ContactName>Dummy Test</iodef:ContactName>
    <iodef:Email>
      <iodef:EmailTo>contact@dummytest.com</iodef:EmailTo>
    </iodef:Email>
  </iodef:Contact>
  <iodef:EventData>
    <iodef:Description>Dummy Test sharing with ISP1</iodef:Description>
    <iodef:Method>
      <iodef:Reference>
        <iodef:URL>
        </iodef:URL>
      </iodef:Reference>
      <iodef:URL>
        http://en.wikipedia.org/wiki/Low_Orbit_Ion_Cannon
      </iodef:URL>
      <iodef:Description>
        Low Orbit Ion Cannon User Agent
      </iodef:Description>
    </iodef:Method>
  </iodef:EventData>
</iodef:Incident>
</IODEF-Document>
</iodef:Method>
<iodef:Flow>
  <iodef:System category="source" spoofed="no">
    <iodef:Node>
      <iodef:Address category="ipv4-addr">
        192.0.2.104
      </iodef:Address>
    </iodef:Node>
    <iodef:Service ip-protocol="6">
      <iodef:Port>1337</iodef:Port>
    </iodef:Service>
  </iodef:System>
  <iodef:System category="source" spoofed="no">
    <iodef:Node>
      <iodef:Address category="ipv4-addr">
        192.0.2.106
      </iodef:Address>
    </iodef:Node>
    <iodef:Service ip-protocol="6">
      <iodef:Port>1337</iodef:Port>
    </iodef:Service>
  </iodef:System>
  <iodef:System category="source" spoofed="yes">
    <iodef:Node>
      <iodef:Address category="ipv4-net">
        198.51.100.0/24
      </iodef:Address>
    </iodef:Node>
    <iodef:Service ip-protocol="6">
      <iodef:Port>1337</iodef:Port>
    </iodef:Service>
  </iodef:System>
  <iodef:System category="source" spoofed="yes">
    <iodef:Node>
      <iodef:Address category="ipv6-addr">
        2001:db8:dead:beef::1
      </iodef:Address>
    </iodef:Node>
    <iodef:Service ip-protocol="6">
      <iodef:Port>1337</iodef:Port>
    </iodef:Service>
  </iodef:System>
  <iodef:System category="target">
    <iodef:Node>
      <iodef:Address category="ipv4-addr">
        203.0.113.1
      </iodef:Address>
    </iodef:Node>
  </iodef:System>
</iodef:Flow>
B.3. 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 [RFC7970].
<iodef:Incident purpose="reporting">
  <iodef:IncidentID name="csirt.example.com">
    189601
  </iodef:IncidentID>
  
  <iodef:DetectTime>2013-01-04T08:06:12+00:00</iodef:DetectTime>
  <iodef:StartTime>2013-01-04T08:01:34+00:00</iodef:StartTime>
  <iodef:EndTime>2013-01-04T08:31:27+00:00</iodef:EndTime>
  <iodef:ReportTime>2013-01-04T09:15:45+00:00</iodef:ReportTime>
  <iodef:GenerationTime>2013-01-04T09:15:45+00:00</iodef:GenerationTime>
  
  <iodef:Description>
    Zeus Spear Phishing E-mail with Malware Attachment
  </iodef:Description>
  
  <iodef:Assessment occurrence="potential">
    <iodef:SystemImpact severity="medium" type="takeover-system">
      <iodef:Description>
        Malware with Command and Control Server and System Changes
      </iodef:Description>
    </iodef:SystemImpact>
  </iodef:Assessment>
  
  <iodef:Contact role="creator" type="organization">
    <iodef:ContactName>example.com CSIRT</iodef:ContactName>
  </iodef:Contact>
  
  <iodef:EventData>
    <iodef:Description>
      Targeting Defense Contractors, specifically board members attending Dummy Con
    </iodef:Description>
  </iodef:EventData>
  
  <iodef:Method>
    <iodef:Reference observable-id="ref-1234">
      <iodef:Description>Zeus</iodef:Description>
    </iodef:Reference>
  </iodef:Method>
  
  <iodef:Flow>
    <iodef:System category="source">
      <iodef:Node>
        <iodef:Address category="site-uri">
          http://www.zeusevil.example.com
        </iodef:Address>
        <iodef:Address category="ipv4-addr">
          192.0.2.166
        </iodef:Address>
        <iodef:Address category="asn">
          65535
        </iodef:Address>
      </iodef:Node>
    </iodef:System>
  </iodef:Flow>
</iodef:Incident>
<iodef:Flow>
  <iodef:Node category="malware-distribution"/>
</iodef:System>
</iodef:Flow>
</iodef:System category="source">
</iodef:Node>
</iodef:DomainData>
</iodef:Name>
</iodef:DateDomainWasChecked>
<!-- <iodef:RelatedDNS RecordType="MX"> -->
<iodef:RelatedDNS dtype="string">
  evildave.example.com MX preference = 10, mail exchanger
  = mail1.evildave.example.com
</iodef:RelatedDNS>
<iodef:RelatedDNS dtype="string">
  maill.evildave.example.com
  internet address = 198.51.100.6
</iodef:RelatedDNS>
<iodef:RelatedDNS dtype="string">
  zuesevil.example.com. IN TXT "v=spf1 a mx -all"
</iodef:RelatedDNS>
</iodef:DomainData>
</iodef:Node>
<iodef:NodeRole category="mail">
  Sending phishing mails
</iodef:Description>
</iodef:NodeRole>
</iodef:Service>

<iodef:EmailData>
<iodef:EmailFrom>
emaildave@evildave.example.com
</iodef:EmailFrom>
<iodef:EmailSubject>
Join us at Dummy Con
</iodef:EmailSubject>
<iodef:EmailX-Mailer>
StormRider 4.0
</iodef:EmailX-Mailer>
</iodef:EmailData>
</iodef:Service>
</iodef:System>
<iodef:System category="target">
<iodef:Node>
<iodef:Address category="ipv4-addr">
203.0.113.2
</iodef:Address>
</iodef:Node>
</iodef:System>
</iodef:Flow>
</iodef:Expectation action="other"/>
</iodef:Record>
</iodef:RecordData>
</iodef:FileData observable-id="fd-1234">
</iodef:File>
<iodef:FileName>
Dummy Con Sign Up Sheet.txt
</iodef:FileName>
<iodef:FileSize>
152
</iodef:FileSize>
<iodef:HashData scope="file-contents">
<iodef:Hash>
<ds:DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>
141accc23e7e5157de60853cb1e01bc38042d08f9086040815300b7fe75c184
</ds:DigestValue>
</iodef:Hash>
</iodef:HashData>
</iodef:File>
</iodef:RecordData>
B.4. 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-Document version="2.00"
xmlns="urn:ietf:params:xml:ns:iodef-2.0"
xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
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">189234</iodef:IncidentID>
    <iodef:GenerationTime>2013-03-07T16:14:56.757+05:30</iodef:GenerationTime>
    <iodef:Description>
      Malware and related indicators identified
    </iodef:Description>
    <iodef:Assessment occurrence="potential">
      <iodef:SystemImpact severity="medium" type="breach-proprietary">
        <iodef:Description>
          Malware with Command and Control Server and System Changes
        </iodef:Description>
      </iodef:SystemImpact>
    </iodef:Assessment>
  </iodef:Incident>
</IODEF-Document>
</iodef:SystemImpact>
</iodef:Assessment>
</iodef:Contact role="creator" type="organization">
</iodef:ContactName>example.com CSIRT</iodef:ContactName>
</iodef:Email>
</iodef:EmailTo>contact@csirt.example.com</iodef:EmailTo>
</iodef:Email>
</iodef:Contact>
</iodef:EventData>
</iodef:Method>
</iodef:Reference>
</iodef:URL>
</iodef:Method>
</iodef:Description>Zeus</iodef:Description>
</iodef:Reference>
</iodef:Flow>
</iodef:System category="source">
</iodef:Method>
</iodef:Address category="ipv4-addr" observable-id="addr-c2-91011-001">
203.0.113.200</iodef:Address>
</iodef:Address category="site-uri" observable-id="addr-c2-91011-002">
http://zeus.556677889900.example.com/log-bin/
lunch_install.php?aff_id=1&amp;amp;amp;
lunch_id=1&amp;amp;amp;maddr=&amp;amp;amp;
action=install
</iodef:Address>
</iodef:Node>
</iodef:NodeRole category="c2-server"/>
</iodef:System>
</iodef:Flow>
</iodef:Record>
</iodef:RecordData>
</iodef:FileData observable-id="file-91011-001">
</iodef:File>
</iodef:FileData scope="file-contents">
</iodef:Hash>
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha1"/>
<ds:DigestValue>
MHg2NzUxQTIlMzQ4M0E2N0Q4NkUwRjg0NzYwRjxRjxRjxEwQkJDQzJFREZG
</ds:DigestValue>
</iodef:Hash>
</iodef:FileData>
</iodef:File>
</iodef:File>

Kampanakis & Suzuki Expires March 11, 2018 [Page 25]
<iodef:HashData scope="file-contents">
  <iodef:Hash>
    <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#md5"/>
    <ds:DigestValue>
      MHgyRTg4ODA5ODBENjI0NDdFOTc5MEFGQTg5NTEzRjBBNA==
    </ds:DigestValue>
  </iodef:Hash>
</iodef:HashData>

</iodef:FileData>

</iodef:FileData>

</iodef:WindowsRegistryKeysModified observable-id="regkey-91011-001">
  <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:EventData>

</iodef:EventData>

</iodef:Method>

</iodef:Reference>

</iodef:URL>

http://www.threatexpert.example.com/report.aspx?md5=c3c528c939f9b176c883ae0ce5df0001

</iodef:URL>

</iodef:Description>Cridex</iodef:Description>

</iodef:Reference>

</iodef:Method>

</iodef:Flow>

</iodef:System category="source">
  <iodef:Node>
    <iodef:Address category="ipv4-addr" observable-id="addr-c2-91011-003">
      203.0.113.100
    </iodef:Address>
  </iodef:Node>
</iodef:System>
<iodef:Node>
    <iodef:NodeRole category="c2-server"/>
    <iodef:Service ip-protocol="6">
        <iodef:Port>8080</iodef:Port>
    </iodef:Service>
</iodef:System>
</iodef:Flow>
<iodef:Record>
    <iodef:RecordData>
        <iodef:FileData observable-id="file-91011-002">
            <iodef:File>
                <iodef:HashData scope="file-contents">
                    <iodef:Hash>
                        <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha1"/>
                        MHg3MjYzRkUwRDNBMDk1RDU5QzhFMEM4OTBOUM10DVFMzQzRTcxNDFD
                    </iodef:Hash>
                </iodef:HashData>
            </iodef:File>
        </iodef:FileData>
        <iodef:FileData observable-id="file-91011-003">
            <iodef:File>
                <iodef:HashData scope="file-contents">
                    <iodef:Hash>
                        <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#md5"/>
                        MHg0M0NEODUwRkNEQURFNDMzMEE1QkVBNkYxNkVFOTcxQw==
                    </iodef:HashData>
                </iodef:File>
            </iodef:FileData>
        </iodef:FileData>
        <iodef:WindowsRegistryKeysModified observable-id="regkey-91011-002">
            <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:IndicatorData>
<iodef:Indicator>
  <iodef:IndicatorID name="csirt.example.com" version="1">
    ind-91011
  </iodef:IndicatorID>
  <iodef:Description>
    evil c2 server, file hash, and registry key
  </iodef:Description>
  <iodef:IndicatorExpression operator="or">
    <iodef:Observable>
      <iodef:Address category="site-uri" observable-id="addr-qrst">
        http://foo.example.com:12345/evil/cc.php
      </iodef:Address>
    </iodef:Observable>
    <iodef:Observable>
      <iodef:Address category="ipv4-addr" observable-id="addr-stuv">
        192.0.2.1
      </iodef:Address>
    </iodef:Observable>
    <iodef:Observable>
      <iodef:Address category="ipv4-addr" observable-id="addr-tuvw">
        198.51.100.1
      </iodef:Address>
    </iodef:Observable>
    <iodef:Observable>
      <iodef:Address category="ipv6-addr" observable-id="addr-uvwx">
        2001:db8:dead:beef::1
      </iodef:Address>
    </iodef:Observable>
  </iodef:IndicatorExpression>
  <iodef:IndicatorExpression operator="and">
    <iodef:Observable>
      <iodef:FileData observable-id="file-91011-000">
        <iodef:File>
          <iodef:HashData scope="file-contents">
            <iodef:Hash>
              <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlnc#sha256"/>
              <ds:DigestValue>141accec23e7e5157de60853c1e01bc38042d08f9086040815300b7fe75c184</ds:DigestValue>
            </iodef:Hash>
          </iodef:HashData>
        </iodef:File>
      </iodef:FileData>
    </iodef:Observable>
  </iodef:IndicatorExpression>
</iodef:Indicator>
<iodef:Observable>
  <iodef:WindowsRegistryKeysModified observable-id="regkey-91011-000">
    <iodef:Key registryaction="add-key"
      observable-id="regkey-vwxy">
      <iodef:KeyName>
        HKLM\SYSTEM\CurrentControlSet\Services\Net CLR
      </iodef:KeyName>
    </iodef:Key>
    <iodef:Key registryaction="add-key"
      observable-id="regkey-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"
      observable-id="regkey-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"
      observable-id="regkey-zabc">
      <iodef:KeyName>
        HKLM\SYSTEM\CurrentControlSet\Services\Net CLR\Parameters\Bar
      </iodef:KeyName>
      <iodef:Value>Baz</iodef:Value>
    </iodef:Key>
  </iodef:WindowsRegistryKeysModified>
</iodef:Observable>

<iodef:IndicatorExpression operator="or">
  <iodef:IndicatorExpression operator="and">
    <iodef:ObservableReference uid-ref="file-91011-001"/>
    <iodef:ObservableReference uid-ref="regkey-91011-001"/>
  </iodef:IndicatorExpression>
  <iodef:IndicatorExpression operator="and">
    <iodef:ObservableReference uid-ref="file-91011-002"/>
    <iodef:ObservableReference uid-ref="file-91011-003"/>
  </iodef:IndicatorExpression>
</iodef:IndicatorExpression>
B.5. IoT Malware

The IoT Malware test exchanged information that described a bad IP address of IoT malware and its scanned ports. This example information is extracted from alert messages of a Darknet monitoring system referred in [RFC8134]. The IODEF version used for the data representation was based on [RFC7970].

<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="2.00"
    xmlns="urn:ietf:params:xml:ns:iodef-2.0"
    xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <iodef:Incident purpose="reporting">
        <iodef:IncidentID name="csirt.example.com">189802</iodef:IncidentID>
        <iodef:ReportTime>2017-03-01T01:15:00+09:00</iodef:ReportTime>
        <iodef:GenerationTime>2017-03-01T01:15:00+09:00</iodef:GenerationTime>
        <iodef:Description>IoT Malware and related indicators</iodef:Description>
        <iodef:Assessment occurrence="potential">
            <iodef:SystemImpact severity="medium" type="takeover-system">
                <iodef:Description>IoT Malware is scanning other hosts</iodef:Description>
            </iodef:SystemImpact>
        </iodef:Assessment>
        <iodef:Contact role="creator" type="organization">
            <iodef:ContactName>example.com CSIRT</iodef:ContactName>
        </iodef:Contact>
        <iodef:EventData>
            <iodef:Discovery source="nidps">
                <iodef:Description>Detected by darknet monitoring</iodef:Description>
            </iodef:Discovery>
        </iodef:EventData>
    </iodef:Incident>
</IODEF-Document>
</iodef:Discovery>

<iodef:Flow>
  <iodef:System category="source">
    <iodef:Node>
      <iodef:Address category="ipv4-addr">
        192.0.2.210
      </iodef:Address>
    </iodef:Node>
    <iodef:NodeRole category="camera"/>
    <iodef:Service ip-protocol="6">
      <iodef:Port>23</iodef:Port>
    </iodef:Service>
    <iodef:OperatingSystem>
      <iodef:Description>
        Example Surveillance Camera OS 2.1.1
      </iodef:Description>
    </iodef:OperatingSystem>
  </iodef:System>
</iodef:Flow>

<iodef:EventData>
  <iodef:Flow>
    <iodef:System category="target">
      <iodef:Node>
        <iodef:Address category="ipv4-addr">
          198.51.100.1
        </iodef:Address>
      </iodef:Node>
      <iodef:NodeRole category="honeypot"/>
      <iodef:Service ip-protocol="6">
        <iodef:Port>23</iodef:Port>
      </iodef:Service>
    </iodef:System>
  </iodef:Flow>
</iodef:EventData>

<iodef:EventData>
  <iodef:Flow>
    <iodef:System category="target">
      <iodef:Node>
        <iodef:Address category="ipv4-addr">
          198.51.100.94
        </iodef:Address>
      </iodef:Node>
      <iodef:NodeRole category="honeypot"/>
      <iodef:Service ip-protocol="6">
        <iodef:Port>23</iodef:Port>
      </iodef:Service>
    </iodef:System>
  </iodef:Flow>
</iodef:EventData>

Kampanakis & Suzuki Expires March 11, 2018 [Page 31]
</iodef:EventData>
</iodef:EventData>
</iodef:Flow>
</iodef:System category="target">
  <iodef:Node>
    <iodef:Address category="ipv4-addr">
      198.51.100.237
    </iodef:Address>
  </iodef:Node>
  <iodef:Node>
    <iodef:NodeRole category="honeypot"/>
    <iodef:Service ip-protocol="6">
      <iodef:Port>2323</iodef:Port>
    </iodef:Service>
  </iodef:Node>
</iodef:System>
</iodef:EventData>
</iodef:EventData>
</iodef:Incident>
</IODEF-Document>

Authors' Addresses

Panos Kampanakis
Cisco Systems
Email: pkampana@cisco.com

Mio Suzuki
NICT
4-2-1, Nukui-Kitamachi
Koganei, Tokyo 184-8795
JP
Email: mio@nict.go.jp
Abstract

This document defines a resource-oriented approach for security automation information publication, discovery, and sharing. Using this approach, producers may publish, share, and exchange representations of software descriptors, security incidents, attack indicators, software vulnerabilities, configuration checklists, and other security automation information as web-addressable resources. Furthermore, consumers and other stakeholders may access and search this security information as needed, establishing a rapid and on-demand information exchange network for restricted internal use or public access repositories. This specification extends the Atom Publishing Protocol and Atom Syndication Format to transport and share security automation resource representations.

Contributing to this document

The source for this draft is being maintained on GitHub. Suggested changes should be submitted as pull requests at <https://github.com/CISecurity/ROLIE>. Instructions are on that page as well. Editorial changes can be managed in GitHub, but any substantial issues need to be discussed on the MILE mailing list.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."
1. Introduction .................................................. 3
2. Terminology .................................................. 4
3. XML-related Conventions ....................................... 4
   3.1. XML Namespaces ............................................ 4
   3.2. RELAX NG Compact Schema ................................ 5
4. Background and Motivation ..................................... 5
5. ROLIE Requirements for the Atom Publishing Protocol ....... 6
   5.1. AtomPub Service Documents ............................... 7
   5.1.1. Use of the "app:workspace" Element .................. 7
   5.1.2. Use of the "app:collection" Element .................. 8
   5.1.3. Service Discovery ...................................... 9
   5.2. AtomPub Category Documents ............................. 9
   5.3. Transport Layer Security .................................. 10
   5.4. User Authentication and Authorization ................... 11
   5.5. / (forward slash) Resource URL ......................... 11
   5.6. HTTP methods ............................................. 11
6. ROLIE Requirements for the Atom Syndication Format .......... 12
   6.1. Use of the "atom:feed" element .......................... 12
     6.1.1. Use of the "atom:category" Element ................. 13
     6.1.2. Use of the "atom:link" Element ...................... 14
     6.1.3. Use of the "atom:updated" Element .................. 15
     6.2. Use of the "atom:entry" Element ...................... 15
     6.2.1. Use of the "atom:content" Element .................. 16
     6.2.2. Use of the "atom:link" Element ...................... 16
     6.2.3. Use of the "rolie:format" Element ................... 17
     6.2.4. Use of the rolie:property Element .................. 18
     6.2.5. Requirements for a Standalone Entry ................. 19
7. Available Extension Points Provided by ROLIE ................ 19
   7.1. The Category Extension Point ........................... 20
1. Introduction

This document defines a resource-oriented approach to security automation information sharing that follows the Representational State Transfer (REST) architectural style [REST]. In this approach, computer security resources are maintained in web-accessible repositories structured as Atom Syndication Format [RFC4287] Feeds. Within a given Feed, which may be requested by the consumer, representations of specific types of security automation information are organized, categorized, and described. Furthermore, all collections available to a given user are discoverable, allowing the consumer to search all available content they are authorized to view, and to locate and request the desired information resources. Through use of granular authentication and access controls, only authorized consumers may be permitted the ability to read or write to a given Feed.

The goal of this approach is to increase the communication and sharing of security information between providers and consumers that can be used to automate security processes (e.g., incident reports, vulnerability assessments, configuration checklists, and other
security automation information). Such sharing allows human operators and computer systems to leverage this standardized communication system to gather information that supports the automation of security processes.

In order for new types of security automation information and associated resource representations to be shared over time, this specification defines extension points that can be used to add support for new information types and associated resource representations by means of additional supplementary specification documents. Sections 5 and 6 of this document define the core requirements of all implementations of this specification, and is resource representation agnostic. An overview of the extension system is provided in Section 7. Implementers seeking to provide support for specific security automation information types should refer to the specification for that domain described by the IANA registry found in section 8.4.

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].

The previous key words are used in this document to define the conformance requirements for implementations of this specification. This document does not give any recommendations for the use of ROLIE, it only provides conformance requirements for implementations of this specification.

Definitions for some of the common computer security-related terminology used in this document can be found in Section 2 of [RFC7970].

The following terms are unique to this specification:

Information Type A class of security automation information having one or more associated data models. Often such security automation information is used in the automation of a security process. See section 7.1.2 for more information.

3. XML-related Conventions

3.1. XML Namespaces

This specification uses XML Namespaces [W3C.REC-xml-names-20091208] to uniquely identify XML element names. It uses the following namespace prefix mappings for the indicated namespace URI:
"app" is used for the "http://www.w3.org/2007/app" namespace defined in [RFC5023].

"atom" is used for the "http://www.w3.org/2005/Atom" namespace defined in [RFC4287].

"rolie" is used for the "urn:ietf:params:xml:ns:rolie:1.0" namespace defined in section 8.1 of this specification.

3.2. RELAX NG Compact Schema

Some sections of this specification are illustrated with fragments of a non-normative RELAX NG Compact schema [relax-NG]. The text of this specification provides the definition of conformance. Schema for the "http://www.w3.org/2007/app" and "http://www.w3.org/2005/Atom" namespaces appear in RFC5023 appendix B [RFC5023] and RFC4287 appendix B [RFC4287] respectively.

A complete informative RELAX NG Compact Schema for the new elements introduced by ROLIE is provided in Appendix A.

4. Background and Motivation

In order to automate security process, tools need access to sufficient sources of structured, security information that can be used to drive security processes. Thus, security information sharing is one of the core components of automating security processes. Vulnerabilities, configurations, software identification, security incidents, and patch data are just a few of the classes of information that are shared today to enable effective security on a wide scale. However, as the scale of defense broadens as networks become larger and more complex, and the volume of information to process makes humans-in-the-loop difficult to scale, the need for automation and machine-to-machine communication becomes increasingly critical.

ROLIE seeks to address this need by providing three major information sharing benefits:

Extensible information type categories and format agnosticism: ROLIE is not bound to any given data format or category of information. Instead, information categories are extensible, and entries declare the format of the referenced data. In cases where several formats or serializations are available, ROLIE can use link relations to communicate how a consumer can access these formats. For example, clients may request that a given resource representation be returned as XML, JSON, or in some other format or serialization. This approach allows the provider to support...
multiple, compatible formats allowing the consumer to select the most suitable version.

Open and distributed information sharing: Using the Atom Publishing Protocol, ROLIE feeds can easily aggregate feeds and accept information POSTed to them from other sources. Webs of communicating ROLIE servers form ad-hoc sharing communities, increasing data availability and the ability to correlate linked data across sources for participating consumers. ROLIE servers needn’t be distributed however, as large ROLIE repositories can function as a central or federated collections.

Stateless communication model: ROLIE, as a RESTful system, is stateless. That is, the server doesn’t keep track of client sessions, but rather uses link relations for state transitions. In practice, this means that any consumer can find and share information at any organizational level and at any time without needing to execute a long series of requests.

Information discovery and navigation: ROLIE provides a number of mechanisms to allow clients to programmatically discover and navigate collections of information in order to dynamically discover new or revised content. Extensible information types and other categories provide one way of determining content that is desirable. Link elements, each with a target URI and an established relationship type, provide a means for ROLIE providers to link other information that is relevant to the current entry or feed.

These benefits result in an information sharing protocol that is lightweight, interactive, open, and most importantly, machine readable.

The requirements in this specification are broken into two major sections, extensions to the Atom Publishing Protocol (AtomPub) [RFC5023], and extensions to the Atom Syndication Format [RFC4287]. All normative requirements in AtomPub and Atom Syndication are inherited from their respective specifications, and apply here unless the requirement is explicitly overridden in this document. In this way, this document may upgrade the requirement (e.g., make a SHOULD a MUST), but will never downgrade a given requirement (e.g., make a MUST a SHOULD).

5. ROLIE Requirements for the Atom Publishing Protocol

This section describes a number of restrictions of and extensions to the Atom Publishing Protocol (AtomPub) [RFC5023] that define the use of that protocol in the context of a ROLIE-based solution. The
normative requirements in this section are generally oriented towards client and server implementations. An understanding of the Atom Publishing Protocol specification [RFC5023] is helpful to understand the requirements in this section.

5.1. AtomPub Service Documents

As described in RFC5023 section 8 [RFC5023], a Service Document is an XML-based document format that allows a client to dynamically discover the Collections provided by a publisher. A Service Document consists of one or more app:workspace elements that may each contain a number of app:collection elements.

The general structure of a service document is as follows (from RFC5023 section 4.2 [RFC5023]):

```
Service
  o- Workspace
    |  o- Collection
    |   |  o- IRI, categories, media types
    |   o- ...
    o- Workspace
      |  o- Collection
      |   o- IRI, categories, media types
      o- ...
  o- ...
```

5.1.1. Use of the "app:workspace" Element

In AtomPub, a Workspace, represented by the "app:workspace" element, describes a group of one or more Collections. Building on the AtomPub concept of a Workspace, in ROLIE a Workspace represents an aggregation of Collections pertaining to security automation information resources. This specification does not impose any restrictions on the number of Workspaces that may be in a Service Document or the specific Collections to be provided within a given Workspace.

A ROLIE implementation can host Collections containing both public and private information entries. It is RECOMMENDED that
implementations segregate public and private Collections into different app:workspace elements. By doing this, Workspaces that contain private information can be ignored by clients or can be omitted from the Service Document provided to a client that lacks the appropriate privilege to access the set of Collections associated with the Workspace.

5.1.2. Use of the "app:collection" Element

In AtomPub, a Collection in a Service Document, represented by the "app:collection" element, provides metadata that can be used to point to a specific Atom Feed that contains information Entries that may be of interest to a client. The association between a Collection and a Feed is provided by the "href" attribute of the app:collection element. Building on the AtomPub concept of a Collection, in ROLIE a Collection represents a pointer to a group of security automation information resources pertaining to a given type of security automation information. Collections are represented as Atom Feeds as per RFC 5023. Atom Feed specific requirements are defined in section 6.1.

The following restrictions are imposed on the use of the app:collection element for ROLIE implementations:

- The atom:category elements contained in the app:categories element MUST be the same set of atom:categories used in the Atom Feed resource indicated by the app:collection "href" attribute value. This ensures that the category metadata associated with the Collection is discoverable in both the Feed and the corresponding Collection in the Service Document.

- An app:collection pertaining to a security automation information resource Feed MUST contain an app:categories element that minimally contains a single atom:category element with the "scheme" attribute value of "urn:ietf:params:rolie:category:information-type". This category MUST have an appropriate "term" attribute value as defined in section 7.1.1. This ensures that a given Collection corresponds to a specific type of security automation information.

- Any app:collection element that does not contain a descendant atom:category element with the "scheme" attribute value of "urn:ietf:params:rolie:category:information-type" MUST be considered a non-ROLIE Collection. This allows Collections pertaining to security automation information to co-exist alongside Collections of other non-ROLIE information within the same AtomPub instance.
The app:categories element in an app:collection MAY include additional atom:category elements using a scheme other than "urn:ietf:params:rolie:category:information-type". This allows other category metadata to be included.

5.1.3. Service Discovery

This specification requires that an implementation MUST publish an Atom Service Document that describes the set of security information Collections provided by the service. The Service Document MUST be retrievable using the standardized URI template "https://{host:port}/.well-known/rolie/servicedocument", where (host:port) is the authority portion of the URI. Dereferencing this URI MAY result in a redirect based on an appropriate HTTP 3xx status code to direct the client to the actual Service Document. This allows clients to have a well-known location to find a ROLIE service document, while giving implementations flexibility over how the service is deployed.

This document registers the "rolie/servicedocument" well-known URI as per [RFC5785] in Section 8.5.

A mechanism to determine which host and port to use is not specified in this document. Use of a mechanism such as DNS SRV Records [RFC2782] can provide a secure and reliable discovery mechanism for determining a specific host and port to use for retrieving a Service Document for a given DNS domain.

5.2. AtomPub Category Documents

As described in RFC5023 section 7 [RFC5023], a Category Document is an XML-based document format that allows a client to dynamically discover the Categories used within AtomPub Service Documents, and Atom Syndication Feed and Entry documents provided by a publisher. A Category Document consists of one app:categories element that contains a number of inline atom:category elements, or a URI referencing a Category Document.

A ROLIE implementation MUST publish a Category Document that describes the set of atomic:category elements and associated terms currently used by the service.

The Category Document MUST be retrievable using the standardized URI template "https://{host:port}/.well-known/rolie/categorydocument", where {host:port} is the authority portion of the URI. Dereferencing this URI MAY result in a redirect based on an appropriate HTTP 3xx status code to direct the client to the actual Category Document. This allows clients to have a well-known location to find a ROLIE
category document, while giving implementations flexibility over how the service is deployed.

This document registers the "rolie/categorydocument" well-known URI as per [RFC5785] in Section 8.5.

5.3. Transport Layer Security

ROLIE is intended to be handled with TLS. The following requirements have been derived from [RFC7589].

The most recent published version of TLS MUST be supported, and any mandatory-to-implement (MTI) cipher suites in that version MUST be supported as well.

The server MUST support certificate-based client authentication. The implementation MAY use any TLS cipher suite that supports mutual authentication.

During the TLS negotiation, the client MUST carefully examine the certificate presented by the server to determine if it meets the client’s expectations. Particularly, the client MUST check its understanding of the server hostname against the server’s identity as presented in the server Certificate message, in order to prevent man-in-the-middle attacks. Matching is performed according to the rules laid out in the Security Considerations section of [RFC4642].

If the match fails, the client MUST either ask for explicit user confirmation or terminate the connection and indicate the server’s identity is suspect. Additionally, clients MUST verify the binding between the identity of the servers to which they connect and the public keys presented by those servers. Client implementations SHOULD support an equivalent certificate validation approach to the what is defined in Section 6 of [RFC5280], but MAY supplement that algorithm with other validation methods that achieve equivalent levels of verification (such as comparing the server certificate against a local store of already-verified certificates and identity bindings). If the client has external information as to the expected identity of the server, the hostname check MAY be omitted.

The server MUST be capable of verifying the identity of the client with certificate-based authentication according to local policy to ensure that the incoming client request is legitimate before any configuration or state data is sent to or received from the client.
5.4. User Authentication and Authorization

Implementations MUST support user authentication. However, a given implementation MAY allow user authentication to be disabled on a feed by feed basis.

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

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.

5.5. / (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]. If the "/" resource is supported the following behavior MUST be also supported:

- Consistent with RFC6546 errata, a client requesting a GET on "/" SHOULD 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 is handled correctly as a RID request. 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.

If the "/" resource is unsupported, then a request for this resource MUST provide a 404 HTTP status code.

5.6. HTTP methods

Servers MAY accept request methods beyond those specified in this document.

Clients MUST be capable of recognizing and processing any standard HTTP status code, as defined in [RFC5023] Section 5.
6. ROLIE Requirements for the Atom Syndication Format

This section describes a number of restrictions of and extensions to
the Atom Syndication Format [RFC4287] that define the use of that
format in the context of a ROLIE-based solution. The normative
requirements in this section are generally oriented towards any
content to be published to a ROLIE server. An understanding of the
Atom Syndication Format specification [RFC4287] is helpful to
understand the requirements in this section.

6.1. Use of the "atom:feed" element

As described in RFC4287 section 4.1.1 [RFC4287], an Atom Feed is an
XML-based document format that describes a list of related
information items. The list of Atom Feeds provided by a ROLIE
service instance are listed in the service’s Service Document through
one or more app:collection elements. Each Feed document, represented
using the atom:feed element, contains a listing of zero or more
related information items individually called a "Member Entry" or
"Entry".

When applied to the problem domain of security automation information
sharing, an Atom Feed may be used to represent any meaningful
collection of security automation information resources. Each Entry
in an atom:feed represents an individual resource (e.g., a specific
checklist, a software vulnerability record). Additional Feeds can be
used to represent other collections of security automation resources.

The following Atom Feed definition represents a stricter definition
of the atom:feed element defined in RFC 4287 for use in a ROLIE
implementation. Any element not specified here inherits its
definition and requirements from [RFC4287].
atomFeed =
   element atom:feed {
      atomCommonAttributes,
      (atomAuthor* & atomCategory+ & atomContributor* & atomGenerator?
       & atomIcon? & atomId & atomLink+ & atomLogo? & atomRights?
       & atomSubtitle? & atomTitle & atomUpdated & extensionElement*),
      atomEntry*}

6.1.1. Use of the "atom:category" Element

An atom:feed can be categorized and can contain information from zero or more categories. In Atom the naming scheme and the semantic meaning of the terms used to identify an Atom category are application-defined.

The following restrictions are imposed on the use of the atom:category element when used in an atom:feed:

- An atom:feed element MUST minimally contain a single atom:category element with the "scheme" attribute value of "urn:ietf:params:rolie:category:information-type". This category MUST have an appropriate "term" attribute value as defined in section 7.1.1. This ensures that a given Feed corresponds to a specific type of security automation information. All member Entries in the Feed MUST represent security automation information records of the provided information type category or categories.

- Any atom:feed element that does not contain a child atom:category element with the "scheme" attribute value of "urn:ietf:params:rolie:category:information-type" MUST NOT be considered a ROLIE Collection. This allows Feeds pertaining to security automation information to co-exist alongside Feeds of other non-ROLIE information within the same AtomPub instance.

- An atom:feed MAY include additional atom:category elements using a scheme other than "urn:ietf:params:rolie:category:information-type". This allows other category metadata to be included.
6.1.2. Use of the "atom:link" Element

Link relations defined by the atom:link element are used to represent state transitions using a stateless approach. In Atom a type of link relationship can be defined using the "rel" attribute.

A ROLIE atom:feed MUST contain one or more atom:link elements with rel="service" and href attribute whose value is a IRI that points to an Atom Service Document associated with the atom:feed. If a client accesses a Feed without first accessing the service’s service document, a link with the "service" relationship provides a means to discover additional security automation information. The "service" link relationship is defined in the IANA Link Relations Registry [1].

An atom:feed can contain an arbitrary number of Entries. In some cases, a complete Feed may consist of a large number of Entries. Additionally, as new and updated Entries are ordered at the beginning of a Feed, a client may only be interested in retrieving the first N entries in a Feed to process only the Entries that have changed since the last retrieval of the Feed. As a practical matter, a large set of Entries will likely need to be divided into more manageable portions. Based on RFC5005 section 3 [RFC5005], link elements SHOULD be included in all Feeds to support paging using the following link relation types:

- "first" - Indicates that the href attribute value of the link identifies a resource IRI for the furthest preceding page of the Feed.
- "last" - Indicates that the href attribute value of the link identifies a resource IRI for the furthest following page of the Feed.
- "previous" - Indicates that the href attribute value of the link identifies a resource IRI for the immediately preceding page of the Feed.
- "next" - Indicates that the href attribute value of the link identifies a resource IRI for the immediately following page of the Feed.

For example:
Example Paged Feed

A reference to a historical Feed may need to be stable, and/or a Feed may need to be divided into a series of defined epochs. Implementations SHOULD support the mechanisms described in RFC5005 section 4 [RFC5005] to provide link-based state transitions for maintaining archiving of Feeds.

An atom:feed MAY include additional link relationships not specified in this document. If a client encounters an unknown link relationship type, the client MUST ignore the unrecognized link and continue processing as if the unrecognized link element did not appear. The definition of new Link relations that provide additional state transition extensions is discussed in section 7.3.

6.1.3. Use of the "atom:updated" Element

The atom:updated element MUST be populated with the current time at the instant the Feed representation was last updated by adding, updating, or deleting an Entry; or changing any metadata for the Feed.

6.2. Use of the "atom:entry" Element

Each Entry in an Atom Feed, represented by the atom:entry element, describes a single referenced information record, along with descriptive information about its format, media type, and other publication metadata. The following atom:entry schema definition represents a stricter representation of the atom:entry element defined in [RFC4287] for use in a ROLIE-based Atom Feed.
atomEntry =
    element atom:entry {
        atomCommonAttributes,
    }

6.2.1. Use of the "atom:content" Element

There MUST be exactly one atomContent element in the Entry. The content element MUST adhere to this definition, which is a stricter representation of the atom:content element defined in [RFC4287]:

atomContent =
    element atom:content {
        atomCommonAttributes,
        attribute type { atomMediaType },
        attribute src { atomUri },
        empty
    }

The type attribute MUST identify the serialization type of the content, for example, application/xml or application/json. A prefixed media type MAY be used to reflect a specific model used with a given serialization approach (e.g., application/rdf+xml). The src attribute MUST be an IRI that can be dereferenced to retrieve the related content data.

6.2.2. Use of the "atom:link" Element

Link relations can be included in an atom:entry to represent state transitions for the Entry.

If there is a need to provide the same information in different data models and/or serialization formats, separate Entry instances can be
An atom:feed MAY include additional link relationships not specified in this document. If a client encounters an unknown link relationship type, the client MUST ignore the unrecognized link and continue processing as if the unrecognized link element did not appear. The definition of new Link relations that provide additional state transition extensions is discussed in section 7.3.

6.2.3. Use of the "rolie:format" Element

As mentioned earlier, a key goal of this specification is to allow a consumer to review a set of published security automation information resources, and then identify and retrieve any resources of interest. The format of the data is a key criteria to consider when deciding what information to retrieve. For a given type of security automation information, it is expected that a number of different formats may be used to represent this information. To support this use case, both the serialization format and the specific data model expressed in that format must be known by the consumer.

The rolie:format element is used to describe the data model used to express the information referenced in the atom:content element of an atom:entry. It also allows a schema to be identified that can be used when parsing the content to verify or better understand the structure of the content.

There MUST be exactly one rolie:format element in an atom:entry. The element MUST adhere to this definition:

```xml
rolieFormat =
  element rolie:format {
    appCommonAttributes,
    attribute ns { atomURI },
    attribute version { text } ?,
    attribute schema-location { atomURI } ?,
    attribute schema-type { atomMediaType } ?,
    empty
  }
```

The rolie:format element MUST provide a "ns" attribute that identifies the data model of the resource referenced by the atom:content element. For example, the namespace used may be an XML namespace URI, or an identifier that represents a serialized JSON model. The URI used for the "ns" attribute value MUST be an absolute
or opaque URI. The resource identified by the URI need not be resolvable.

The rolie:format element MAY provide a "version" attribute that identifies the version of the format used for the related atom:content.

The rolie:format element MAY provide a "schema-location" attribute that is a URI that identifies a schema resource that can be used to validate the related atom:content.

The rolie:format element MAY provide a "schema-type" attribute, which is a mime type identifying the format of the schema resource identified by the "schema-location" attribute.

The following nominal example shows how these attributes describe the format of the content:

```xml
<rolie:format ns="urn:ietf:params:xml:ns:iodef-2.0"
    version="2.0"
    schema-location="https://www.iana.org/assignments/xml-registry/schema/iodef-2.0.xsd"
    schema-type="text/xml"/>
```

The previous element provides an indication that the content of the given entry is using the IODEF v2 format.

### 6.2.4. Use of the rolie:property Element

An atom:category element provides a way to associate a name/value pair of categorical information using the scheme and term attributes to represent the name, and the label attribute to represent the value. When used in this way an atom:category allows a specific label to be selected from a finite set of possible label values that can be used to further classify a given atom:entry or atom:feed. Within ROLIE, there may be a need to associate additional metadata with an atom:entry. In such a case, use of an atom:category is not practical to represent name/value data for which the allowed values are unbounded. Instead, ROLIE has introduced a new rolie:property element that can represent non-categorical metadata as name/value pairs. Examples include content-specific identifiers, naming data, and other properties that allow for unbounded values.

There MAY be zero or more rolie:property elements in an atom:entry.

The element MUST adhere to this definition:
rolieProperty =
  element rolie:property {
    app:appCommonAttributes,
    attribute name { atom:atomURI },
    attribute value { text }
  } empty

The name attribute provides a URI that identifies the namespace and
name of the property as a URI.

The value attribute is text that provides a value for the property
identified by the name attribute.

For example, the nominal element <rolie:property
name="urn:ietf:params:rolie:property:csirt-iodef-id" value="12345"/>
would expose an IODEF ID value contained in a given entry’s content.
The name used in the example also demonstrates the use of a
registered ROLIE property extension, which is described in
Section 7.4.

Implementations MAY use locally defined and namespaced elements in an
Entry in order to provide additional information. Clients that do
not recognize a property with an unregistered name attribute MAY
ignore the rolie:property.

6.2.5. Requirements for a Standalone Entry

If an Entry is ever shared as a standalone resource, separate from
its containing Feed, then the following additional requirements
apply:

- The Entry MUST have an atom:link element with rel="collection" and
  href="[IRI of the containing Collection]". This allows the Feed
  or Feeds for which the Entry is a member to be discovered, along
  with the related information the Feed may contain. In the case of
  the Entry have multiple containing Feeds, the Entry MUST have one
  atom:link for each related Feed.

- The Entry MUST declare the information type of the content
  resource referenced by the Entry (see Section 7.1.2).

7. Available Extension Points Provided by ROLIE

This specification does not require particular information types or
data formats; rather, ROLIE is intended to be extended by additional
specifications that define the use of new categories and link
relations. The primary point of extension is through the definition
of new information type category terms. Additional specifications can register new information type category terms with IANA that serve as the main characterizing feature of a ROLIE Collection/Feed or Resource/Entry. These additional specifications defining new information type terms, can describe additional requirements for including specific categories, link relations, as well as, use of specific data formats supporting a given information type term.

7.1. The Category Extension Point

The atom:category element, defined in RFC 4287 section 4.2.2 [RFC4287], provides a mechanism to provide additional categorization information for a content resource in ROLIE. The ability to define new categories is one of the core extension points provided by Atom. A Category Document, defined in RFC 5023 section 7 [RFC5023], provides a mechanism for an Atom implementation to make discoverable the atom:category terms and associated allowed values.

ROLIE further defines the use of the existing Atom extension category mechanism by allowing ROLIE specific category extensions to be registered with IANA, and additionally has assigned the "urn:ietf:params:rolie:category:information-type" category scheme that has special meaning for implementations of ROLIE. This allows category scheme namespaces to be managed in a more consistent way, allowing for greater interoperability between content producers and consumers.

The namespace "urn:ietf:params:rolie:category:local" has been reserved in the IANA ROLIE Parameters table for private use as defined in [RFC5226]. Any category whose "scheme" attribute uses this as a prefix MUST be considered private use. Implementations encountering such a category MUST parse the content without error, but MAY otherwise ignore the element.

Use of the "atom:category" element is discussed in the following subsections.

7.1.1. General Use of the "atom:category" Element

The atom:category element can be used for characterizing a ROLIE Resource. As discussed earlier in this document, an atom:category element has a "term" attribute that indicates the assigned category value, and a "scheme" attribute that provides an identifier for the category type. The "scheme" provides a means to describe how a set of category terms should be used and provides a namespace that can be used to differentiate terms provided by multiple organizations with different semantic meaning.
To further differentiate category types used in ROLIE, an IANA sub-registry has been established for ROLIE protocol parameters to support the registration of new category "scheme" attribute values by ROLIE extension specifications. Use of this extension point is discussed in section 8.3 using the name field with a type parameter of "category" to indicate a category extension.

7.1.2. Identification of Security Automation Information Types

A ROLIE specific extension point is provided through the atom:category "scheme" value "urn:ietf:params:rolie:category:information-type". This value is a Uniform Resource Name (URN) [RFC2141] that is registered with IANA as described in section 8.3. When used as the "scheme" attribute in this way, the "term" attribute is expected to be a registered value as defined in section Section 8.4. Through this mechanism a given security automation information type can be used to:

1. identify that an "app:collection" element in a Service Document points to an Atom Feed that contains Entries pertaining to a specific type of security automation information (see section 5.1.2), or

2. identify that an "atom:feed" element in an Atom Feed contains Entries pertaining to a specific type of security automation information (see section 6.1.1).

3. identify the information type of a standalone Resource (see section 6.2.5).

For example, the notional security automation information type "incident" would be identified as follows:

```xml
<atom:category
    scheme="urn:ietf:params:rolie:category:information-type"
    term="incident"/>
```

A security automation information type represents a class of information that represents the same or similar information model [RFC3444]. Notional examples of information types include:

indicator: Computing device- or network-related "observable features and phenomenon that aid in the forensic or proactive detection of malicious activity; and associated meta-data" (from [RFC7970]).

incident: Information pertaining to and "derived analysis from security incidents" (from [RFC7970]).
vulnerability reports: Information identifying and describing a vulnerability in hardware or software.

collection checklists: Content that can be used to assess the configuration settings related to installed software.

software tags: Metadata used to identify and characterize installable software.

This is a short list to inspire new engineering of information type extensions that support the automation of security processes.

This document does not specify any information types. Instead, information types in ROLIE are expected to be registered in extension documents that describe one or more new information types. This allows the information types used by ROLIE implementations to grow over time to support new security automation use cases. These extension documents may also enhance ROLIE Service, Category, Feed, and Entry documents by defining link relations, other categories, and Format data model extensions to address the representational needs of these specific information types. New information types are added to ROLIE through registrations to the IANA ROLIE Security Resource Information Type registry defined in section 8.4.

### 7.2. The "rolie:format" Extension Point

Security automation data pertaining to a given information type may be expressed using a number of supported formats. As described in section 6.2.3, the rolie:format element is used to describe the specific data model used to represent the resource referenced by a given "atom:entry". The structure provided by the rolie:format element, provides a mechanism for extension within the atom:entry model. ROLIE extensions MAY further restrict which data models are allowed to be used for a given information type.

By declaring the data model used for a given Resource, a consumer can choose to download or ignore the Resource, or look for alternate formats. This saves the consumer from downloading and parsing resources that the consumer is not interested in or resources expressed in formats that are not supported by the consumer.

### 7.3. The Link Relation Extension Point

This document uses several link relations defined in the IANA Link Relation Types registry [2]. Additional link relations can be registered in this registry to allow new relationships to be represented in ROLIE according to RFC 4287 section 4.2.7.2 [RFC4287]. Based on the preceding reference, if the link relation is too
specific or limited in the intended use, an absolute IRI can be used in lieu of registering a new simple name with IANA.

7.4. The "rolie:property" Extension Point

As discussed previously in Section 6.2.4, many formats contain unique identifying and characterizing properties that are vital for sharing information. In order to provide a global reference for these properties, this document establishes an IANA registry in Section 8.3 that allows ROLIE extensions to register named properties using the name field with a type parameter of "property" to indicate a property extension. Implementations SHOULD prefer the use of registered properties over implementation specific properties when possible.

ROLIE extensions are expected to register new and use existing properties to provide valuable identifying and characterizing information for a given information type and/or format.

The namespace "urn:ietf:params:rolie:property:local" has been reserved in the IANA ROLIE Parameters table for private use as defined in [RFC5226]. Any property whose "name" attribute uses this as a prefix MUST be considered private use. Implementations encountering such a property MUST parse the content without error, but MAY otherwise ignore the element.

This document also registers the "urn:ietf:params:rolie:property:content-author-name" property name. This property provides an exposure point for the person or organization that authored the content linked to in the "src" attribute of the entry’s atom:content element.

8. IANA Considerations

This document has a number of IANA considerations described in the following subsections.

8.1. XML Namespaces and Schema URNs

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

ROLIE XML Namespace The ROLIE namespace (rolie-1.0) has been registered in the "ns" registry.

URI: urn:ietf:params:xml:ns:rolie-1.0

Registrant Contact: IESG
XML: None. Namespace URIs do not represent an XML specification.

ROLIE XML Schema The ROLIE schema (rolie-1.0) has been registered in the "schema" registry.

URI: urn:ietf:params:xml:schema:rolie-1.0
Registrant Contact: IESG
XML: See section A of this document.

8.2. ROLIE URN Sub-namespace

IANA has added an entry to the "IETF URN Sub-namespace for Registered Protocol Parameter Identifiers" registry located at <http://www.iana.org/assignments/params/params.xml#params-1> as per RFC3553 [RFC3553].

The entry is as follows:

Registry name: rolie
Specification: This document
Repository: ROLIE URN Parameters. See Section 8.3 [TO BE REMOVED: This registration should take place at the following location: https://www.iana.org/assignments/rolie]
Index value: See Section 8.3

8.3. ROLIE URN Parameters

A new top-level registry has been created, entitled "Resource Oriented Lightweight Information Exchange (ROLIE) Parameters". [TO BE REMOVED: This registration should take place at the following location: https://www.iana.org/assignments/rolie]

In this top-level registry, a sub-registry entitled "ROLIE URN Parameters" has been created. Registration in this repository is via the Specification Required policy [RFC5226]. Designated Expert reviews should be routed through the MILE WG mailing list. Failing this, the Designated Expert will be assigned by the IESG.

Each entry in this sub-registry must record the following fields:

Name: A URN segment that adheres to the pattern {type}:{label}. The keywords are defined as follows:
(type): The parameter type. The allowed values are "category" or "property". "category" denotes a category extension as discussed in Section 7.1. "property" denotes a property extension as discussed in Section 7.4.

(label): A required US-ASCII string that conforms to the URN syntax requirements (see [RFC2141]). This string must be unique within the namespace defined by the {type} keyword. The "local" label for both the "category" and "property" types has been reserved for private use.

Extension IRI: The identifier to use within ROLIE, which is the full URN using the form: urn:ietf:params:rolie:{name}, where {name} is the "name" field of this registration.

Reference: A static link to the specification and section that the definition of the parameter can be found.

Sub-registry: An optional field that links to an IANA sub-registry for this parameter. If the {type} is "category", the sub-registry must contain a "name" field whose registered values MUST be US-ASCII. The list of names are the allowed values of the "term" attribute in the atom:category element. (See Section 7.1.2).

This repository has the following initial values:
<table>
<thead>
<tr>
<th>Name</th>
<th>Extension IRI</th>
<th>Reference</th>
<th>Sub-Registry</th>
</tr>
</thead>
<tbody>
<tr>
<td>category:information-type</td>
<td>urn:ietf:params:rolie:category :information-type</td>
<td>This document, Section 8.4</td>
<td>[TO BE REMOVED: This registration should take place at the following location: <a href="https://www.iana.org/assignments/rolie">https://www.iana.org/assignments/rolie</a> /category/information-type]</td>
</tr>
<tr>
<td>property:local</td>
<td>urn:ietf:params:rolie:property:local</td>
<td>This document, Section 7.4</td>
<td>None</td>
</tr>
<tr>
<td>category:local</td>
<td>urn:ietf:params:rolie:category:local</td>
<td>This document, Section 7.1</td>
<td>None</td>
</tr>
<tr>
<td>property:content-author-name</td>
<td>urn:ietf:params:rolie:property :content-author-name</td>
<td>This document, Section 7.4</td>
<td>None</td>
</tr>
</tbody>
</table>

### 8.4. ROLIE Security Resource Information Type Sub-Registry

A new sub-registry has been created to store ROLIE information type values.

**Name of Registry:** "ROLIE Information Types"

**Location of Registry:**
https://www.iana.org/assignments/rolie/category/information-type

**Fields to record in the registry:**

- **name:** The full name of the security resource information type as a string from the printable ASCII character set [RFC0020] with individual embedded spaces allowed. The ABNF [RFC5234] syntax for this field is:
1*VCHAR *(SP 1*VCHAR)

index: This is an IANA-assigned positive integer that identifies the registration. The first entry added to this registry uses the value 1, and this value is incremented for each subsequent entry added to the registry.

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

Allocation Policy: Specification required as per [RFC5226]

8.5. Well-Known URI Registrations

This document makes the follow two registrations to the Well-Known URI Registry at https://www.iana.org/assignments/well-known-uris/well-known-uris.xhtml.

Service Document registration:

URI suffix: rolie/servicedocument
Change controller: IETF
Specification document: This document, Section 5.1.3
Related information: None

Category Document registration:

URI suffix: rolie/categorydocument
Change controller: IETF
Specification document: This document, Section 5.2
Related information: None

9. Security Considerations

This document defines a resource-oriented approach for lightweight information exchange using HTTP over TLS, the Atom Syndication Format, and the Atom Publishing Protocol. As such, implementers must understand the security considerations described in those specifications. All that follows is guidance, more specific instruction is out of scope for this document.
All security measures SHOULD be enforced at the source, that is, a provider SHOULD NOT return any Feed content or member Entry content for which the client identity has not been specifically authenticated, authorized, and audited.

The approach described herein is based upon all policy enforcements being implemented at the point when a resource representation is created. As such, producers 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 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 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], 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 consortium, 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.

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 [3] 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. Note that 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.
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.

A service discovery mechanism is not explicitly specified in this document, and there are several approaches available for implementers. When selecting this mechanism, implementations need to ensure that their choice provides a means for authenticating the server. As described in the discovery section, DNS SRV Records are a possible secure solution to discovery.

10. Privacy Considerations

The optional author field may provide an identification privacy issue if populated without the author's consent. This information may become public if posted to a public feed. Special care should be taken when aggregating or sharing entries from other feeds, or when programmatically generating ROLIE entries from some data source that the author's personal info is not shared without their consent.

When using the Atom Publishing Protocol to POST entries to a feed, attackers may use correlating techniques to profile the user. The request time can be compared to the generated "updated" field of the entry in order to build out information about a given user. This correlation attempt can be mitigated by not using HTTP requests to POST entries when profiling is a risk, and rather use backend control of the feeds.

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

Overall, ROLIE introduces few privacy concerns above and beyond those present in any other HTTP protocol. Those that exist can be mitigated by following security considerations and carefully using the optional identifying elements.

11. Acknowledgements

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

12. References

12.1. Normative References


12.2. Informative References


12.3. URIs

[1] https://www.iana.org/assignments/link-relations/link-relations.xhtml

[2] https://www.iana.org/assignments/link-relations/link-relations.xhtml


Appendix A. Relax NG Compact Schema for ROLIE

This appendix is informative.

The Relax NG schema below defines the rolie:format element.
# -*- rnc -*-
# RELAX NG Compact Syntax Grammar for the rolie ns

namespace rolie = "urn:ietf:params:xml:ns:rolie-1.0"
namespace atom = "http://www.w3.org/2005/Atom"
namespace app = "http://www.w3.org/2007/app"

# rolie:format

rolieFormat =
  element rolie:format {
    app:appCommonAttributes,
    attribute ns { atom:atomURI },
    attribute version { text } ?,
    attribute schema-location { atom:atomURI } ?,
    attribute schema-type { atom:atomMediaType } ?,
    empty
  }

# rolie:property

rolieProperty =
  element rolie:property {
    app:appCommonAttributes,
    attribute name { atom:atomURI },
    attribute value { text }
    empty
  }

Appendix B.  Examples of Use

B.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 provider uses an Atom Service Document to enable authorized clients to determine what specific information the provider makes available to the community. While the Service Document is accessible at a pre-determined location, the Service Document can also be made accessible from any well known location, such as via a link from the producer’s home page.

A client may format an HTTP GET request to retrieve the service document from the specified location:
GET /.well-known/rolie/servicedocument
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 Collections that are provided.

Example HTTP GET response:

HTTP/1.1 200 OK
Date: Fri, 24 Aug 2016 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>
    <atom:title type="text">Vulnerabilities</atom:title>
    <collection href="http://example.org/provider/vulns">
      <atom:title type="text">Vulnerabilities Feed</atom:title>
      <categories fixed="yes">
        <atom:category
          scheme="urn:ietf:params:rolie:category:information-type"
          term="vulnerability"/>
      </categories>
    </collection>
  </workspace>
</service>

This simple Service Document example shows that the server provides
one workspace, named "Vulnerabilities". Within that workspace, the
server makes one Collection available.

A server may also offer a number of different Collections, each
containing different types of security automation information. In
the following example, a number of different Collections are
provided, each with its own category and authorization scope. This
categorization will help the clients to decide which Collections will
meet their needs.
<service xmlns="http://www.w3.org/2007/app"
         xmlns:atom="http://www.w3.org/2005/Atom">
<workspace>
<collection>
<atom:title>Public Vulnerabilities</atom:title>
<atom:link rel="service"
href="http://www.example.com/rolie/servicedocument"/>
<categories fixed="yes">
<atom:category
scheme="urn:ietf:params:rolie:category:information-type"
term="vulnerability"/>
</categories>
</collection>
</workspace>
<workspace>
<atom:title>Private Consortium Sharing</atom:title>
<collection>
<atom:title>Incidents</atom:title>
<atom:link rel="service"
href="http://www.example.com/rolie/servicedocument"/>
<categories fixed="yes">
<atom:category
scheme="urn:ietf:params:rolie:category:information-type"
term="incident"/>
</categories>
</collection>
</workspace>
</service>

In this example, the provider is making available a total of two Collections, organized into two different workspaces. The first workspace contains a Collection consisting of publicly available software vulnerabilities. The second workspace provides an incident Collection for use by a private sharing consortium. An appropriately authenticated and authorized client may then proceed to make HTTP requests for these Collections. The publicly provided vulnerability information may be accessible with or without authentication. However, users accessing the Collection restricted to authorized
members of a private sharing consortium are expected to authenticate before access is allowed.

B.2. Feed Retrieval

This section provides a non-normative example of a client retrieving an vulnerability Feed.

Having discovered the available security information sharing Collections, a client who is a member of the general public may be interested in receiving the Collection of public vulnerabilities. The client may retrieve the Feed for this Collection by performing an HTTP GET operation on the URL indicated by the Collection’s "href" attribute.

Example HTTP GET request for a Feed:

```
GET /provider/public/vulns
Host: www.example.org
Accept: application/atom+xml
```

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

Example HTTP GET response for a Feed:
HTTP/1.1 200 OK
Date: Fri, 24 Aug 2016 17:20:11 GMT
Content-Length: 2882
Content-Type: application/atom+xml;charset='utf-8'

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
     xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0"
     xml:lang="en-US">
  <id>2a7e265a-39bc-43f2-b711-b8fd9264b5c9</id>
  <title type="text">
    Atom formatted representation of a feed of XML vulnerability documents
  </title>
  <category
 scheme="urn:ietf:params:rolie:category:information-type"
 term="vulnerability"/>
  <updated>2016-05-04T18:13:51.0Z</updated>
  <link rel="self"
       href="http://example.org/provider/public/vulns" />
  <link rel="service"
       href="http://example.org/rolie/servicedocument"/>
  <entry>
    <rolie:format ns="urn:ietf:params:xml:ns:exampleformat"/>
    <id>dd786dba-88e6-440b-9158-b8fae67ef67c</id>
    <title>Sample Vulnerability</title>
    <published>2015-08-04T18:13:51.0Z</published>
    <updated>2015-08-05T18:13:51.0Z</updated>
    <summary>A vulnerability issue identified by CVE-...</summary>
    <content type="application/xml"
             src="http://www.example.org/provider/vulns/123456/data"/>
  </entry>

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

This Feed document has two Atom Entries, one of which has been elided. The first Entry illustrates an atom:entry element that provides a summary of essential details about one particular vulnerability. Based upon this summary information and the provided category information, a client may choose to do an HTTP GET request, on the content "src" attribute, to retrieve the full details of the vulnerability.
B.3. Entry Retrieval

This section provides a non-normative example of a client retrieving an vulnerability 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 vulnerability Entry by performing an HTTP GET operation on the URL indicated by the "src" attribute of the atom:content element.

Example HTTP GET request for an Entry:

GET /provider/public/vulns/123456
Host: www.example.org
Accept: application/atom+xml;type=entry

The corresponding HTTP response would be an XML document containing the Atom Entry for the vulnerability record:

Example HTTP GET response for an Entry:

HTTP/1.1 200 OK
Date: Fri, 24 Aug 2016 17:30:11 GMT
Content-Length: 713
Content-Type: application/atom+xml;type=entry;charset=UTF-8

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom"
xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0"
xml:lang="en-US">
  <id>f63aafa9-4082-48a3-9ce6-97a2d69d4a9b</id>
  <title>Sample Vulnerability</title>
  <published>2015-08-04T18:13:51.0Z</published>
  <updated>2015-08-05T18:13:51.0Z</updated>
  <category
    scheme="urn:ietf:params:rolie:category:information-type"
    term="vulnerability"/>
  <summary>A vulnerability issue identified by CVE-...</summary>
  <rolie:format ns="urn:ietf:params:xml:ns:exampleformat"/>
  <content type="application/xml"
    src="http://www.example.org/provider/vulns/123456/data">
  </content>
</entry>

The example response above shows an XML document referenced by the "src" attribute of the atom:content element. The client may retrieve the document using this URL.
Appendix C. Change History

Changes in draft-ietf-mile-rolie-07 since draft-ietf-mile-rolie-06 version, March 13, 2017 to TODO, 2017

Added /.well-known/ registration and requirement to service discovery.

Condensed and re-focused Sections 1 and 4 to be more concise.

Added privacy considerations section.

Made a number of editorial changes as per WGLC review.

Changes in draft-ietf-mile-rolie-06 since draft-ietf-mile-rolie-05 version, November 2, 2016 to March 13, 2017

Changed to standards track

Added the rolie:property element

Fixed references (Normative vs Informative)

Set Service and Category document URL template requirements

Fixed XML snippets in examples

Changes in draft-ietf-mile-rolie-05 since draft-ietf-mile-rolie-04 version, October 21, 2016 to November 2, 2016

Added ROLIE specific terminology to section 2

Added AtomPub Category Document in section 5.2

Edited document, improving consistency in terminology usage and capitalization of key terms, as well as enhancing clarity.

Removed unused format parameter type in section 8.3

Schema removed, the normative schema consists of the snippets in the requirements sections.

Changes in draft-ietf-mile-rolie-04 since draft-ietf-mile-rolie-03 version, July 8, 2016 to October 31, 2016

Further specification and clarification of requirements
IANA Considerations and extension system fleshed out and described.

Examples and References updated.

Schema created.

Fixed both internal section and external document referencing.

Removed XACML Guidance Appendix. This will be added to a future draft on ROLIE Authentication and Access Control.

Changes made in draft-ietf-mile-rolie-03 since draft-ietf-mile-rolie-02 version, May 27, 2016 to July 8, 2015:

Atom Syndication and Atom Pub requirements split and greatly expanded for increased justification and technical specification.

Reintroduction and reformatting of some use case examples in order to provide some guidance on use.

Established rough version of IANA table extension system along with explanations of said system.

Re-organized document to put non-vital information in appendices.

Changes made in draft-ietf-mile-rolie-02 since draft-field-mile-rolie-01 version, December, 2015 to May 27, 2016:

All CSIRT and IODEF/RID material moved to companion CSIRT document.

Recast document into a more general use perspective. The implication of CSIRTs as the defacto end-user has been removed where ever possible. All of the original CSIRT based use cases remain completely supported by this document, it has been opened up to support many other use cases.

Changed the content model to broaden support of representation.

Edited and rewrote much of sections 1, 2 and 3 in order to accomplish a broader scope and greater readability.

Removed any requirements from the Background section and, if not already stated, placed them in the requirements section.

Re-formatted the requirements section to make it clearer that it contains the lions-share of the requirements of the specification.
Changes made in draft-ietf-mile-rolie-01 since draft-field-mile-rolie-02 version, August 15, 2013 to December 2, 2015:

- Added section specifying the use of RFC5005 for Archive and Paging of Feeds.
- Added section describing use of atom categories that correspond to IODEF expectation class and impact classes. See: normative-expectation-impact
- Dropped references to adoption of a MILE-specific HTTP media type parameter.
- Updated IANA Considerations section to clarify that no IANA actions are required.

Authors’ Addresses

John P. Field  
Pivotal Software, Inc.  
625 Avenue of the Americas  
New York, New York  
USA  

Phone: (646)792-5770  
Email: jfield@pivotal.io

Stephen A. Banghart  
National Institute of Standards and Technology  
100 Bureau Drive  
Gaithersburg, Maryland  
USA  

Phone: (301)975-4288  
Email: sab3@nist.gov

David Waltermire  
National Institute of Standards and Technology  
100 Bureau Drive  
Gaithersburg, Maryland 20877  
USA  

Email: david.waltermire@nist.gov
Abstract

This document defines a resource-oriented approach for security automation information publication, discovery, and sharing. Using this approach, producers may publish, share, and exchange representations of software descriptors, security incidents, attack indicators, software vulnerabilities, configuration checklists, and other security automation information as web-addressable resources. Furthermore, consumers and other stakeholders may access and search this security information as needed, establishing a rapid and on-demand information exchange network for restricted internal use or public access repositories. This specification extends the Atom Publishing Protocol and Atom Syndication Format to transport and share security automation resource representations.

Contributing to this document

The source for this draft is being maintained on GitHub. Suggested changes should be submitted as pull requests at <https://github.com/CISecurity/ROLIE>. Instructions are on that page as well. Editorial changes can be managed in GitHub, but any substantial issues need to be discussed on the MILE mailing list.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."
This Internet-Draft will expire on June 17, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction ................................................. 3
2. Terminology ................................................. 4
3. XML-related Conventions ................................. 4
   3.1. XML Namespaces ......................................... 4
   3.2. RELAX NG Compact Schema .............................. 5
4. Background and Motivation .............................. 5
5. ROLIE Requirements for the Atom Publishing Protocol ...... 6
   5.1. AtomPub Service Documents ........................... 7
   5.1.1. Use of the "app:workspace" Element .............. 7
   5.1.2. Use of the "app:collection" Element ............ 8
   5.1.3. Service Document Discovery ...................... 9
   5.2. Category Documents ................................. 9
   5.3. Transport Layer Security ........................... 9
   5.4. User Authentication and Authorization .............. 10
   5.5. / (forward slash) Resource URL ..................... 10
   5.6. HTTP methods ........................................ 11
6. ROLIE Requirements for the Atom Syndication Format ...... 11
   6.1. Use of the "atom:feed" element .................... 11
   6.1.1. Use of the "atom:category" Element ........... 13
   6.1.2. Use of the "atom:link" Element ................. 13
   6.1.3. Use of the "atom:updated" Element ............. 14
   6.2. Use of the "atom:entry" Element .................... 15
   6.2.1. Use of the "atom:content" Element .......... 15
   6.2.2. Use of the "atom:link" Element ............... 16
   6.2.3. Use of the "rolie:format" Element ........... 16
   6.2.4. Use of the rolie:property Element .......... 18
   6.2.5. Requirements for a Standalone Entry .......... 19
7. Available Extension Points Provided by ROLIE .......... 19
   7.1. The Category Extension Point ..................... 20
1. Introduction

This document defines a resource-oriented approach to security automation information sharing that follows the Representational State Transfer (REST) architectural style [REST]. In this approach, computer security resources are maintained in web-accessible repositories structured as Atom Syndication Format [RFC4287] Feeds. Within a given Feed, which may be requested by the consumer, representations of specific types of security automation information are organized, categorized, and described. Furthermore, all collections available to a given user are discoverable, allowing the consumer to search all available content they are authorized to view, and to locate and request the desired information resources. Through use of granular authentication and access controls, only authorized consumers may be permitted the ability to read or write to a given Feed.

The goal of this approach is to increase the communication and sharing of security information between providers and consumers that can be used to automate security processes (e.g., incident reports, vulnerability assessments, configuration checklists, and other security automation information). Such sharing allows human
operators and computer systems to leverage this standardized communication system to gather information that supports the automation of security processes.

To support new types of security automation information being used as time goes on, this specification defines a number of extension points that can be used either privately or globally. These global extensions are IANA registered by ROLIE extension specifications, and provide enhanced interoperability for new use cases and domains. Sections 5 and 6 of this document define the core requirements of all implementations of this specification, and is resource representation agnostic. An overview of the extension system is provided in Section 7. Implementers seeking to provide support for specific security automation information types should refer to the specification for that domain described by the IANA registry found in Section 8.4.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The previous key words are used in this document to define the requirements for implementations of this specification. As a result, the key words in this document are not used for recommendations or requirements for the use of ROLIE.

Definitions for some of the common computer security-related terminology used in this document can be found in Section 2 of [RFC7970].

The following terms are unique to this specification:

Information Type  A class of security automation information having one or more associated data models. Often such security automation information is used in the automation of a security process. See Section 7.1.2 for more information.

3. XML-related Conventions

3.1. XML Namespaces

This specification uses XML Namespaces [W3C.REC-xml-names-20091208] to uniquely identify XML element names. It uses the following namespace prefix mappings for the indicated namespace URI:
"app" is used for the "http://www.w3.org/2007/app" namespace defined in [RFC5023].

"atom" is used for the "http://www.w3.org/2005/Atom" namespace defined in [RFC4287].

"rolie" is used for the "urn:ietf:params:xml:ns:rolie:1.0" namespace defined in Section 8.1 of this specification.

3.2. RELAX NG Compact Schema

Some sections of this specification are illustrated with fragments of a non-normative RELAX NG Compact schema [relax-NG]. The text of this specification provides the definition of conformance. Schema for the "http://www.w3.org/2007/app" and "http://www.w3.org/2005/Atom" namespaces appear in RFC5023 appendix B [RFC5023] and RFC4287 appendix B [RFC4287] respectively.

A complete informative RELAX NG Compact Schema for the new elements introduced by ROLIE is provided in Appendix A.

4. Background and Motivation

In order to automate security process, tools need access to sufficient sources of structured security information that can be used to drive security processes. Thus, security information sharing is one of the core components of automating security processes. Vulnerabilities, configurations, software identification, security incidents, and patch data are just a few of the classes of information that are shared today to enable effective security on a wide scale. However, as the scale of defense broadens as networks become larger and more complex, and the volume of information to process makes humans-in-the-loop difficult to scale, the need for automation and machine-to-machine communication becomes increasingly critical.

ROLIE seeks to address this need by providing four major information sharing benefits:

Extensible information type categories and format agnosticism: ROLIE is not bound to any given data format or category of information. Instead, information categories are extensible, and entries declare the format of the referenced data. In cases where several formats or serializations are available, ROLIE can use link relations to communicate how a consumer can access these formats. For example, clients may request that a given resource representation be returned as XML, JSON, or in some other format or serialization. This approach allows the provider to support
multiple isomorphic formats allowing the consumer to select the
most suitable version.

Open and distributed information sharing: Using the Atom Publishing
Protocol, ROLIE feeds can easily aggregate feeds and accept
information POSTed to them from other sources. Webs of
communicating ROLIE servers form ad-hoc sharing communities,
increasing data availability and the ability to correlate linked
data across sources for participating consumers. ROLIE servers
needn't be distributed however, as large ROLIE repositories can
function as a central or federated collections.

Stateless communication model: ROLIE, as a RESTful system, is
stateless. That is, the server doesn't keep track of client
sessions, but rather uses link relations for state transitions.
In practice, this means that any consumer can find and share
information at any organizational level and at any time without
needing to execute a long series of requests.

Information discovery and navigation: ROLIE provides a number of
mechanisms to allow clients to programmatically discover and
navigate collections of information in order to dynamically
discover new or revised content. Extensible information types and
other categories provide one way of determining content that is
desirable. Link elements, each with a target URI and an
established relationship type, provide a means for ROLIE providers
to link other information that is relevant to the current entry or
feed.

These benefits result in an information sharing protocol that is
lightweight, interactive, open, and most importantly, machine
readable.

The requirements in this specification are broken into two major
sections, extensions to the Atom Publishing Protocol (AtomPub)
[RFC5023], and extensions to the Atom Syndication Format [RFC4287].
All normative requirements in AtomPub and Atom Syndication are
inherited from their respective specifications, and apply here unless
the requirement is explicitly overridden in this document. In this
way, this document may upgrade the requirement (e.g., make a SHOULD a
MUST), but will never downgrade a given requirement (e.g., make a
MUST a SHOULD).

5. ROLIE Requirements for the Atom Publishing Protocol

This section describes a number of restrictions of and extensions to
the Atom Publishing Protocol (AtomPub) [RFC5023] that define the use
of that protocol in the context of a ROLIE-based solution. The
normative requirements in this section are generally oriented towards client and server implementations. An understanding of the Atom Publishing Protocol specification [RFC5023] is helpful to understand the requirements in this section.

5.1. AtomPub Service Documents

As described in RFC5023 section 8 [RFC5023], a Service Document is an XML-based document format that allows a client to dynamically discover the Collections provided by a publisher. A Service Document consists of one or more app:workspace elements that may each contain a number of app:collection elements.

The general structure of a service document is as follows (from RFC5023 section 4.2 [RFC5023]):

```
Service
  o- Workspace
    |   Collection
    |       o- URI, categories, media types
    |       o- ...
    o- Workspace
    |   Collection
    |       o- URI, categories, media types
    |       o- ...
    o- ...
```

Note that the IRIs in the original diagram have been replaced with URIs.

5.1.1. Use of the "app:workspace" Element

In AtomPub, a Workspace, represented by the "app:workspace" element, describes a group of one or more Collections. Building on the AtomPub concept of a Workspace, in ROLIE a Workspace represents an aggregation of Collections pertaining to security automation information resources. This specification does not restrict the number of Workspaces that may be in a Service Document or the specific Collections to be provided within a given Workspace.
A ROLIE implementation can host Collections containing both public and private information entries. It is suggested that implementations segregate Collections into different app:workspace elements by their client access requirements. With proper naming of workspaces, this reduces the amount of trial and error a human user would need to utilize to discover accessible Collections.

5.1.2. Use of the "app:collection" Element

In AtomPub, a Collection in a Service Document, represented by the "app:collection" element, provides metadata that can be used to point to a specific Atom Feed that contains information Entries that may be of interest to a client. The association between a Collection and a Feed is provided by the "href" attribute of the app:collection element. Building on the AtomPub concept of a Collection, in ROLIE a Collection represents a pointer to a group of security automation information resources pertaining to a given type of security automation information. Collections are represented as Atom Feeds as per RFC 5023. Atom Feed specific requirements are defined in Section 6.1.

ROLIE defines specialized data requirements for Collections, Feeds, and Entries containing security automation related data. The difference between a ROLIE and a non-ROLIE Collection defined in a Service Document can be determined as follows:

ROLIE Collection: An app:collection is considered a ROLIE Collection when it contains an app:categories element that contains only one atom:category element with the "scheme" attribute value of "urn:ietf:params:rolie:category:information-type". Further, this category has an appropriate "term" attribute value as defined in Section 7.1.1. This ensures that a given Collection corresponds to a specific type of security automation information.

Non-ROLIE Collection: An app:collection is considered a non-ROLIE Collection when it does not contain an atom:category element with a "scheme" attribute value of "urn:ietf:params:rolie:category:information-type".

By distinguishing between ROLIE and non-ROLIE Collections in this way, implementations supporting ROLIE can host Collections pertaining to security automation information alongside Collections of other non-ROLIE information within the same AtomPub instance.

The following are additional requirements on the use of the app:collection element for a ROLIE Collection:
The child atom:category elements contained in the app:categories element MUST be the same set of atom:category elements used in the Atom Feed resource referenced by the app:collection "href" attribute value. This ensures that the category metadata associated with the Collection and the associated Feed is discoverable in both of these resources.

The app:categories element in an app:collection MAY include additional atom:category elements using a scheme other than "urn:ietf:params:rolie:category:information-type". This allows other category metadata to be included.

### 5.1.3. Service Document Discovery

The Service Document serves as the "head" of a given ROLIE repository: from the Service Document all other repository content can be discovered. A client will need to determine the URL of this Service Document to discover the Collections provided by the repository. The client might determine the URL from a web page, based on out-of-band communication, or through a "service" link relation in a Feed or Entry document that the client has already retrieved. The latter is a typical scenario if the client learns of a specific feed or entry through an out-of-band mechanism, and wishes to discover additional information provided by the repository.

This document does not provide a fully automated discovery mechanism. A mechanism may be defined in the future that allows automated clients to discover the URL to use to retrieve a ROLIE Service Document representing the head of the ROLIE repository.

### 5.2. Category Documents

As described in RFC5023 section 7 [RFC5023], a Category Document is an XML-based document format that allows a client to dynamically discover the Categories used within AtomPub Service Documents, Atom Syndication Feeds, and Entry documents provided by a publisher. A Category Document consists of one app:categories element that contains a number of inline atom:category elements, or a URI referencing a Category Document.

### 5.3. Transport Layer Security

ROLIE is intended to be handled with TLS. TLS version 1.2 MUST be supported. TLS 1.2 SHOULD be implemented according to all recommendations and best practices present in [RFC7525].

It is RECOMMENDED that the most recent published version of TLS is supported. If this version is TLS 1.3 [I-D.ietf-tls-tls13] it is
suggested that 0-RTT (Zero Round Trip Time Resumption) is not used in order to prevent replay attacks. Replay attacks on PUT, POST, or DELETE requests can disrupt repository operation by modifying data unexpectedly.

For example, an automated ROLIE repository that updates very frequently may receive a PUT request against a given resource a few times an hour (or more). An attacker may store an early PUT request, and at the end of the resumption window replay the PUT request, reverting the resource to an old version. Not only could an attacker be doing this replay continuously to cause havoc on the server, but the client is completely unaware of the attack taking place.

Given the potentially sensitive nature of data handled by ROLIE, all appropriate precautions should be taken at the transport layer to protect forward secrecy and user privacy.

The server MUST implement certificate-based client authentication. This MAY be enabled on a workspace by workspace basis.

5.4. User Authentication and Authorization

Implementations MUST support user authentication. However, a given implementation MAY allow user authentication to be disabled on a Feed by Feed, or Workspace by Workspace basis.

It is recommended that servers participating in an information sharing consortium and supporting interactive user logins by members of the consortium support client authentication via a federated identity scheme.

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

5.5. / (forward slash) Resource URL

The "/" resource MAY be supported for compatibility with existing deployments that are using Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS [RFC6546]. The following requirements apply only to implementations supporting RFC 6546.

The following additional requirements only apply if a implementation is supporting the "/" resource as described above:
Consistent with RFC6546 errata, a client requesting a GET on the "/" resource SHOULD receive an HTTP status code 405 Method Not Allowed.

An implementation MAY provide full support for [RFC6546] such that a POST to the "/" resource containing a recognized RID message is handled correctly as a RID request. 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.

If RFC 6546 is unsupported, then a request for the "/" resource may be handled as deemed appropriate by the server.

5.6. HTTP methods

Servers MAY accept request methods beyond those specified in this document.

Clients MUST be capable of recognizing and processing any standard HTTP status code, as defined in [RFC5023] Section 5.

6. ROLIE Requirements for the Atom Syndication Format

This section describes a number of restrictions of and extensions to the Atom Syndication Format [RFC4287] that define valid use of the format in the context of a ROLIE implementation. An understanding of the Atom Syndication Format specification [RFC4287] is helpful to understand the requirements in this section.

6.1. Use of the "atom:feed" element

As described in RFC4287 section 4.1.1 [RFC4287], an Atom Feed is an XML-based document format that describes a list of related information items. The list of Atom Feeds provided by a ROLIE service are listed in the service's Service Document through one or more app:collection elements. Each Feed document, represented using the atom:feed element, contains a listing of zero or more Entries.

When applied to the problem domain of security automation information sharing, an Atom Feed may be used to represent any meaningful collection of security automation information resources. Each Entry in an atom:feed represents an individual resource (e.g., a specific checklist, a software vulnerability record). Additional Feeds can be used to represent other collections of security automation resources.
As discussed in Section 5.1.2, ROLIE defines specialized data requirements for Feeds containing security automation related data. The difference between a ROLIE and a non-ROLIE Feed can be determined as follows:

**ROLIE Feed:** For an atom:feed to be considered a ROLIE Feed, the atom:feed MUST contain only one child atom:category element with the "scheme" attribute value of "urn:ietf:params:rolie:category:information-type". This category MUST have an appropriate "term" attribute value as defined in Section 7.1.1. This ensures that a given Feed corresponds to a specific type of security automation information.

**Non-ROLIE Feed:** For an atom:feed to be considered a non-ROLIE Feed, the atom:feed MUST NOT contain an atom:category element with a "scheme" attribute value of "urn:ietf:params:rolie:category:information-type".

By distinguishing between ROLIE and non-ROLIE Feeds in this way, implementations supporting ROLIE can host Feeds pertaining to security automation information alongside Feeds of other non-ROLIE information within the same AtomPub instance. This is parallel to the handling of collections earlier in this specification in Section 5.1.2.

The following Atom Feed definition represents a stricter definition of the atom:feed element defined in RFC 4287 when used as a ROLIE Feed. Any element not specified here inherits its definition and requirements from [RFC4287].

```xml
atomFeed =
  element atom:feed {
    atomCommonAttributes,
    (atomAuthor*
     & atomCategory+
     & atomContributor*
     & atomGenerator?
     & atomIcon?
     & atomId
     & atomLink+
     & atomLogo?
     & atomRights?
     & atomSubtitle?
     & atomTitle
     & atomUpdated
     & extensionElement*),
    atomEntry*
  }
```
The following subsections contain requirements for a ROLIE Feed.

6.1.1. Use of the "atom:category" Element

An atom:feed can contain one or more atom:category elements. In Atom the naming scheme and the semantic meaning of the terms used to identify an Atom category are application-defined.

The following are additional requirements on the use of the atom:category element when used in a ROLIE Feed:

- All member Entries in the Feed MUST represent security automation information records of the provided information type category.
- An atom:feed MAY include additional atom:category elements using a scheme other than "urn:ietf:params:rolie:category:information-type". This allows other category metadata to be included.

6.1.2. Use of the "atom:link" Element

Link relations defined by the atom:link element are used to represent state transitions using a stateless approach. In Atom a type of link relationship can be defined using the "rel" attribute.

A ROLIE atom:feed MUST contain one or more atom:link elements with rel="service" and href attribute whose value is a URI that points to an Atom Service Document associated with the atom:feed. If a client accesses a Feed without first accessing the service’s service document, a link with the "service" relationship provides a means to discover additional security automation information. The "service" link relationship is defined in the IANA Link Relations Registry [1].

An atom:feed can contain an arbitrary number of Entries. In some cases, a complete Feed may consist of a large number of Entries. Additionally, as new and updated Entries are ordered at the beginning of a Feed, a client may only be interested in retrieving the first N entries in a Feed to process only the Entries that have changed since the last retrieval of the Feed. As a practical matter, a large set of Entries will likely need to be divided into more manageable portions, or pages. Based on RFC5005 section 3 [RFC5005], link elements SHOULD be included in all Feeds to support paging using the following link relation types:

- "first" - Indicates that the href attribute value of the link identifies a resource URI for the furthest preceding page of the Feed.
o "last" - Indicates that the href attribute value of the link identifies a resource URI for the furthest following page of the Feed.

o "previous" - Indicates that the href attribute value of the link identifies a resource URI for the immediately preceding page of the Feed.

o "next" - Indicates that the href attribute value of the link identifies a resource URI for the immediately following page of the Feed.

For example:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom">
  <id>b7f65304-b63b-4246-88e2-c104049c5fd7</id>
  <title>Paged Feed</title>
  <link rel="self" href="https://example.org/feedA?page=5"/>
  <link rel="first" href="https://example.org/feedA?page=1"/>
  <link rel="prev" href="https://example.org/feedA?page=4"/>
  <link rel="next" href="https://example.org/feedA?page=6"/>
  <link rel="last" href="https://example.org/feedA?page=10"/>
  <updated>2012-05-04T18:13:51.0Z</updated>
  <!-- remainder of feed elements -->
</feed>
```

Example Paged Feed

A reference to a historical Feed may need to be stable, and/or a Feed may need to be divided into a series of defined epochs. Implementations SHOULD support the mechanisms described in RFC5005 section 4 [RFC5005] to provide link-based state transitions for maintaining archiving of Feeds.

An atom:feed MAY include additional link relationships not specified in this document. If a client encounters an unknown link relationship type, the client MUST ignore the unrecognized link and continue processing as if the unrecognized link element did not appear. The definition of new Link relations that provide additional state transition extensions is discussed in Section 7.3.

6.1.3. Use of the "atom:updated" Element

The atom:updated element identifies the date and time that a Feed was last updated.
The atom:updated element MUST be populated with the current time at the instant the Feed was last updated by adding, updating, or deleting an Entry; or changing any metadata for the Feed.

6.2. Use of the "atom:entry" Element

Each Entry in an Atom Feed, represented by the atom:entry element, describes a single referenced information record, along with descriptive information about its format, media type, and other publication metadata. The following atom:entry schema definition represents a stricter representation of the atom:entry element defined in [RFC4287] for use in a ROLIE-based Atom Feed as defined in Section 6.1.1.

atomEntry =
element atom:entry {
  atomCommonAttributes,
}

The notable changes from [RFC4287] are the addition of rolieFormat and rolieProperty elements. Also the atomContent element is restricted to the atomOutOfLineContent formulation and is now REQUIRED.

The following subsections contain requirements for Entries in a ROLIE Feed.

6.2.1. Use of the "atom:content" Element

An atom:content element associates its containing Entry with a content resource identified by the src attribute.
There MUST be exactly one atom:content element in the Entry. The content element MUST adhere to this definition, which is a stricter representation of the atom:content element defined in [RFC4287]:

\[
\text{atomContent} = \\
\text{element atom:content} \{ \\
\text{atomCommonAttributes}, \\
\text{attribute type \{ atomMediaType \}}, \\
\text{attribute src \{ atomUri \}}, \\
\text{empty}
\}\n\]

This restricts atomContent in ROLIE to the atomOutofLine formulation presented in [RFC4287].

The type attribute MUST identify the serialization type of the content, for example, application/xml or application/json. A prefixed media type MAY be used to reflect a specific model used with a given serialization approach (e.g., application/rdf+xml). The src attribute MUST be an URI that can be dereferenced to retrieve the related content data.

6.2.2. Use of the "atom:link" Element

Link relations can be included in an atom:entry to represent state transitions for the Entry.

If there is a need to provide the same information in different data models and/or serialization formats, separate Entry instances can be included in the same or a different Feed. Such an alternate content representation can be indicated using an atom:link having a rel attribute with the value "alternate".

An atom:feed MAY include additional link relationships not specified in this document. If a client encounters an unknown link relationship type, the client MUST ignore the unrecognized link and continue processing as if the unrecognized link element did not appear. The definition of new Link relations that provide additional state transition extensions is discussed in Section 7.3.

6.2.3. Use of the "rolie:format" Element

As mentioned earlier, a key goal of this specification is to allow a consumer to review a set of published security automation information resources, and then identify and retrieve any resources of interest. The format of the data is a key criteria to consider when deciding what information to retrieve. For a given type of security automation information, it is expected that a number of different
formats may be used to represent this information. To support this use case, both the serialization format and the specific data model expressed in that format must be known by the consumer.

In the Atom Syndication format, a media type can be defined using the "type" attribute on the "atom:content" element of an atom:entry. The media type can be fully descriptive of the format of the linked document, such as "application/atom+xml". In some cases, however, a format specific media type may not be defined. An example might be when "application/xml" is used because there is no defined specific media type for the content. In such a case the exact data model of the content cannot be known without first retrieving the content.

In cases where a specific media type does not exist, the rolie:format element is used to describe the data model used to express the information referenced in the atom:content element. The rolie:format element also allows a schema to be identified that can be used when parsing the content to verify or better understand the structure of the content.

When it appears, the "rolie:format" element MUST adhere to this definition:

```
rolieFormat =
  element rolie:format {
    appCommonAttributes,
    attribute ns { atomURI },
    attribute version { text } ?,
    attribute schema-location { atomURI } ?,
    attribute schema-type { atomMediaType } ?,
    empty
  }
```

The rolie:format element MUST provide a "ns" attribute that identifies the data model of the resource referenced by the atom:content element. For example, the namespace used may be an XML namespace URI, or an identifier that represents a serialized JSON model. The URI used for the "ns" attribute MUST be absolute. The resource identified by the URI need not be resolvable.

The rolie:format element MAY provide a "version" attribute that identifies the version of the format used for the related atom:content.

The rolie:format element MAY provide a "schema-location" attribute that is a URI that identifies a schema resource that can be used to validate the related atom:content.
The rolie:format element MAY provide a "schema-type" attribute, which is a media type (as described in [RFC2045] identifying the format of the schema resource identified by the "schema-location" attribute.

The following nominal example shows how these attributes describe the format of the content:

<rolie:format ns="urn:ietf:params:xml:ns:iodef-2.0" version="2.0"
schema-location="https://www.iana.org/assignments/xml-registry/schema/iodef-2.0.xsd"
schema-type="text/xml"/>

The previous element provides an indication that the content of the given entry is using the IODEF v2 format.

6.2.4. Use of the rolie:property Element

An atom:category element provides a way to associate a name/value pair of categorical information using the scheme and term attributes to represent the name, and the label attribute to represent the value. When used in this way an atom:category allows a specific label to be selected from a finite set of possible label values that can be used to further classify a given atom:entry or atom:feed. Within ROLIE, there may be a need to associate additional metadata with an atom:entry. In such a case, use of an atom:category is not practical to represent name/value data for which the allowed values are unbounded. Instead, ROLIE has introduced a new rolie:property element that can represent non-categorical metadata as name/value pairs. Examples include content-specific identifiers, naming data, and other properties that allow for unbounded values.

There MAY be zero or more rolie:property elements in an atom:entry.

The element MUST adhere to this definition:

rolieProperty =
   element rolie:property {
      app:appCommonAttributes,
      attribute name { atom:atomURI },
      attribute value { text }
      empty
   }

The name attribute provides a URI that identifies the namespace and name of the property as a URI.
The value attribute is text that provides a value for the property identified by the name attribute.

For example, the nominal element `<rolie:property name="urn:ietf:params:rolie:property:content-id" value="12345"/>` would expose an IODEF ID value contained in a given entry’s content. The name used in the example also demonstrates the use of a registered ROLIE property extension, which is described in Section 7.4.

Implementations MAY use locally defined and namespaced elements in an Entry in order to provide additional information. Clients that do not recognize a property with an unregistered name attribute MUST ignore the rolie:property, that is, the client MUST NOT fail parsing content that contains an unrecognized property.

6.2.5. Requirements for a Standalone Entry

If an Entry is ever shared as a standalone resource, separate from its containing Feed, then the following additional requirements apply:

- The Entry MUST have an atom:link element with rel="collection" and href="[URI of the containing Collection]". This allows the Feed or Feeds for which the Entry is a member to be discovered, along with the related information the Feed may contain. In the case of the Entry have multiple containing Feeds, the Entry MUST have one atom:link for each related Feed.

- The Entry MUST declare the information type of the content resource referenced by the Entry (see Section 7.1.2).

7. Available Extension Points Provided by ROLIE

This specification does not require particular information types or data formats; rather, ROLIE is intended to be extended by additional specifications that define the use of new categories and link relations. The primary point of extension is through the definition of new information type category terms. Additional specifications can register new information type category terms with IANA that serve as the main characterizing feature of a ROLIE Collection/Feed or Resource/Entry. These additional specifications defining new information type terms, can describe additional requirements for including specific categories, link relations, as well as, use of specific data formats supporting a given information type term.
7.1. The Category Extension Point

The atom:category element, defined in RFC 4287 section 4.2.2 [RFC4287], provides a mechanism to provide additional categorization information for a content resource in ROLIE. The ability to define new categories is one of the core extension points provided by Atom. A Category Document, defined in RFC 5023 section 7 [RFC5023], provides a mechanism for an Atom implementation to make discoverable the atom:category terms and associated allowed values.

ROLIE further defines the use of the existing Atom extension category mechanism by allowing ROLIE specific category extensions to be registered with IANA, and additionally has assigned the "urn:ietf:params:rolie:category:information-type" category scheme that has special meaning for implementations of ROLIE. This allows category scheme namespaces to be managed in a more consistent way, allowing for greater interoperability between content producers and consumers.

Any category whose "scheme" attribute uses an unregistered scheme MUST be considered private use. Implementations encountering such a category MUST parse the content without error, but MAY otherwise ignore the element.

Use of the "atom:category" element is discussed in the following subsections.

7.1.1. General Use of the "atom:category" Element

The atom:category element can be used for characterizing a ROLIE Resource. As discussed earlier in this document, an atom:category element has a "term" attribute that indicates the assigned category value, and a "scheme" attribute that provides an identifier for the category type. The "scheme" provides a means to describe how a set of category terms should be used and provides a namespace that can be used to differentiate terms provided by multiple organizations with different semantic meaning.

To further differentiate category types used in ROLIE, an IANA sub-registry has been established for ROLIE protocol parameters to support the registration of new category "scheme" attribute values by ROLIE extension specifications. Use of this extension point is discussed in Section 8.3 using the name field with a type parameter of "category" to indicate a category extension.
7.1.2. Identification of Security Automation Information Types

A ROLIE specific extension point is provided through the
atom:category "scheme" value
"urn:ietf:params:rolie:category:information-type". This value is a
Uniform Resource Name (URN) [RFC8141] that is registered with IANA as
described in Section 8.3. When used as the "scheme" attribute in
this way, the "term" attribute is expected to be a registered value
as defined in Section 8.4. Through this mechanism a given security
automation information type can be used to:

1. identify that an "app:collection" element in a Service Document
   points to an Atom Feed that contains Entries pertaining to a
   specific type of security automation information (see
   Section 5.1.2), or

2. identify that an "atom:feed" element in an Atom Feed contains
   Entries pertaining to a specific type of security automation
   information (see Section 6.1.1).

3. identify the information type of a standalone Resource (see
   Section 6.2.5).

For example, the notional security automation information type
"incident" would be identified as follows:

```xml
<atom:category
    scheme="urn:ietf:params:rolie:category:information-type"
    term="incident"/>
```

A security automation information type represents a class of
information that represents the same or similar information model
[RFC3444]. Note that this document does not register any information
types, but offers the following as examples of potential information
types:

indicator: Computing device- or network-related "observable features
and phenomenon that aid in the forensic or proactive detection of
malicious activity; and associated meta-data" (from [RFC7970]).

incident: Information pertaining to and "derived analysis from
security incidents" (from [RFC7970]).

vulnerability reports: Information identifying and describing a
vulnerability in hardware or software.

configuration checklists: Content that can be used to assess the
configuration settings related to installed software.
software tags: Metadata used to identify and characterize installable software.

This is a short list to inspire new engineering of information type extensions that support the automation of security processes.

This document does not specify any information types. Instead, information types in ROLIE are expected to be registered in extension documents that describe one or more new information types. This allows the information types used by ROLIE implementations to grow over time to support new security automation use cases. These extension documents may also enhance ROLIE Service, Category, Feed, and Entry documents by defining link relations, other categories, and Format data model extensions to address the representational needs of these specific information types. New information types are added to ROLIE through registrations to the IANA ROLIE Security Resource Information Type registry defined in Section 8.4.

7.2. The "rolie:format" Extension Point

Security automation data pertaining to a given information type may be expressed using a number of supported formats. As described in Section 6.2.3, the rolie:format element is used to describe the specific data model used to represent the resource referenced by a given "atom:entry". The structure provided by the rolie:format element, provides a mechanism for extension within the atom:entry model. ROLIE extensions MAY further restrict which data models are allowed to be used for a given information type.

By declaring the data model used for a given Resource, a consumer can choose to download or ignore the Resource, or look for alternate formats. This saves the consumer from downloading and parsing resources that the consumer is not interested in or resources expressed in formats that are not supported by the consumer.

7.3. The Link Relation Extension Point

This document uses several link relations defined in the IANA Link Relation Types registry [2]. Additional link relations can be registered in this registry to allow new relationships to be represented in ROLIE according to RFC 4287 section 4.2.7.2 [RFC4287]. Based on the preceding reference, if the link relation is too specific or limited in the intended use, an absolute URI can be used in lieu of registering a new simple name with IANA.
7.4. The "rolie:property" Extension Point

As discussed previously in Section 6.2.4, many formats contain unique identifying and characterizing properties that are vital for sharing information. In order to provide a global reference for these properties, this document establishes an IANA registry in Section 8.3 that allows ROLIE extensions to register named properties using the name field with a type parameter of "property" to indicate a property extension. Implementations SHOULD prefer the use of registered properties over implementation specific properties when possible.

ROLIE extensions are expected to register new and use existing properties to provide valuable identifying and characterizing information for a given information type and/or format.

The namespace "urn:ietf:params:rolie:property:local" has been reserved in the IANA ROLIE Parameters table for private use as defined in [RFC8126]. Any property whose "name" attribute uses this as a prefix MUST be considered private use. Implementations encountering such a property MUST parse the content without error, but MAY otherwise ignore the element.

This document also registers a number of general use properties that can be used to expose content information in any ROLIE use case. The following are descriptions of how to use these registered properties:

urn:ietf:params:rolie:property:content-author-name The "value" attribute of this property is a text representation indicating the individual or organization that authored the content referenced by the "src" attribute of the entry’s atom:content element. This author may differ from the atom:author when the author of the content and the entry are different people or entities.

urn:ietf:params:rolie:property:content-id The "value" attribute of this property is a text representation of an identifier pertaining to or extracted from the content referenced by the "src" attribute of the entry’s atom:content element. For example, if the atom:entry’s atom:content element links to an IODEF document, the "content-id" value would be an identifier of that IODEF document.

urn:ietf:params:rolie:property:content-published-date The "value" attribute of this property is a text representation indicating the original publication date of the content referenced by the "src" attribute of the entry’s atom:content element. This date may differ from the published date of the ROLIE Entry because publication of the content and the ROLIE Entry represent different events. The date MUST be formatted as specified in [RFC3339].
urn:ietf:params:rolie:property:content-updated-date The "value" attribute of this property is a text representation indicating the date that the content, referenced by the "src" attribute of the entry’s atom:content element, was last updated. This date may differ from the updated date of the ROLIE Entry because updates made to the content and to the ROLIE Entry are different events. The date MUST be formatted as specified in [RFC3339].

8. IANA Considerations

This document has a number of IANA considerations described in the following subsections.

8.1. XML Namespaces and Schema URNs

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

ROLIE XML Namespace The ROLIE namespace (rolie-1.0) has been registered in the "ns" registry.

URI: urn:ietf:params:xml:ns:rolie-1.0

Registrant Contact: IESG

XML: None. Namespace URIs do not represent an XML specification.

ROLIE XML Schema The ROLIE schema (rolie-1.0) has been registered in the "schema" registry.

URI: urn:ietf:params:xml:schema:rolie-1.0

Registrant Contact: IESG

XML: See Appendix A of this document.

8.2. ROLIE URN Sub-namespace

IANA has added an entry to the "IETF URN Sub-namespace for Registered Protocol Parameter Identifiers" registry located at <http://www.iana.org/assignments/params/params.xml#params-1> as per RFC3553 [RFC3553].

The entry is as follows:

Registry name: rolie

Specification: This document
Repository: ROLIE URN Parameters.  See Section 8.3 [TO BE REMOVED: This registration should take place at the following location: https://www.iana.org/assignments/rolie]

Index value: See Section 8.3

8.3.  ROLIE URN Parameters

A new top-level registry has been created, entitled "Resource Oriented Lightweight Information Exchange (ROLIE) URN Parameters". [TO BE REMOVED: This registration should take place at the following location: https://www.iana.org/assignments/rolie]

Registration in the ROLIE URN Parameters registry is via the Specification Required policy [RFC8126]. Registration requests must be sent to both the MILE WG mailing list (mile@ietf.org) and IANA. IANA will forward registration requests to the Designated Expert.

Each entry in this sub-registry must record the following fields:

Name: A URN segment that adheres to the pattern {type}:{label}. The keywords are defined as follows:

{type}: The parameter type. The allowed values are "category" or "property". "category" denotes a category extension as discussed in Section 7.1. "property" denotes a property extension as discussed in Section 7.4.

{label}: A required US-ASCII string that conforms to the URN syntax requirements (see [RFC8141]). This string must be unique within the namespace defined by the {type} keyword. The "local" label for both the "category" and "property" types has been reserved for private use.

Extension URI: The identifier to use within ROLIE, which is the full URN using the form: urn:ietf:params:rolie:{name}, where {name} is the "name" field of this registration.

Reference: A static link to the specification and section that the definition of the parameter can be found.

Sub-registry: An optional field that links to an IANA sub-registry for this parameter. If the {type} is "category", the sub-registry must contain a "name" field whose registered values MUST be US-ASCII. The list of names are the allowed values of the "term" attribute in the atom:category element. (See Section 7.1.2).

This repository has the following initial values:
<table>
<thead>
<tr>
<th>Name</th>
<th>Extension URI</th>
<th>Reference</th>
<th>Sub-Registry</th>
</tr>
</thead>
<tbody>
<tr>
<td>category:information-type</td>
<td>urn:ietf:params:rolie:category:information-type</td>
<td>This document, Section 8.4</td>
<td>[TO BE REMOVED: This registration should take place at the following location: <a href="https://www.iana.org/assignments/rolie/category/information-type">https://www.iana.org/assignments/rolie/category/information-type</a>]</td>
</tr>
<tr>
<td>property:content-author-name</td>
<td>urn:ietf:params:rolie:property:content-author-name</td>
<td>This document, Section 7.4</td>
<td>None</td>
</tr>
<tr>
<td>property:content-id</td>
<td>urn:ietf:params:rolie:property:content-id</td>
<td>This document, Section 7.4</td>
<td>None</td>
</tr>
<tr>
<td>property:content-published-date</td>
<td>urn:ietf:params:rolie:property:content-published-date</td>
<td>This document, Section 7.4</td>
<td>None</td>
</tr>
<tr>
<td>property:content-updated-date</td>
<td>urn:ietf:params:rolie:property:content-updated-date</td>
<td>This document, Section 7.4</td>
<td>None</td>
</tr>
</tbody>
</table>

8.4. ROLIE Security Resource Information Type Sub-Registry

A new sub-registry has been created to store ROLIE information type values.

Name of Registry: "ROLIE Information Types"

Location of Registry: https://www.iana.org/assignments/rolie/category/information-type

Fields to record in the registry:
name: The full name of the security resource information type as a string from the printable ASCII character set [RFC0020] with individual embedded spaces allowed. This value must be unique in the context of this table. The ABNF [RFC5234] syntax for this field is:

```
1*VCHAR *(SP 1*VCHAR)
```

index: This is an IANA-assigned positive integer that identifies the registration. The first entry added to this registry uses the value 1, and this value is incremented for each subsequent entry added to the registry.

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

Allocation Policy: Specification required as per [RFC8126]

9. Security Considerations

This document defines a resource-oriented approach for lightweight information exchange using HTTP over TLS, the Atom Syndication Format, and the Atom Publishing Protocol. As such, implementers must understand the security considerations described in those specifications. All that follows is guidance, more specific instruction is out of scope for this document.

To protect the confidentiality of a given resource provided by a ROLIE implementation, requests for retrieval of the resource need to be authenticated to prevent unauthorized users from accessing the resource (see Section 5.4). It can also be useful to log and audit access to sensitive resources to verify that proper access controls remain in place over time.

Access control to information published using ROLIE should use mechanisms that are appropriate to the sensitivity of the information. Primitive authentication mechanisms like HTTP Basic Authentication [RFC7617] are rarely appropriate for sensitive information. A number of authentication schemes are defined in the HTTP Authentication Schemes Registry [3]. Of these, HOBA [RFC7486] and SCRAM-SHA-256 [RFC7804] provide improved security properties over HTTP Basic [RFC7617] and Digest [RFC7616] Authentication Schemes. 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 a risk-based or multi-factor approach.

Collaborating consortiums may benefit from the adoption of a federated identity solution, such as those based upon OAuth [RFC6749] with JWT [RFC7797], or SAML-core [SAML-core], 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 consortium, 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.

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 [4] 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. Note that 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.

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. Additionally, the underlying serialization approach used in the representation (e.g., XML, JSON) can offer encryption and message authentication capabilities. For example, XMLDSig [RFC3275] for XML, and JSON Web Encryption [RFC7516] and JSON Web Signature[RFC7515] for JSON can provide such mechanisms.

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.

A service discovery mechanism is not explicitly specified in this document, and there are several approaches available for implementers. When selecting this mechanism, implementations need to ensure that their choice provides a means for authenticating the server. As described in the discovery section, DNS SRV Records are a possible solution to discovery.

10. Privacy Considerations

The optional author field may provide an identification privacy issue if populated without the author’s consent. This information may become public if posted to a public feed. Special care should be taken when aggregating or sharing entries from other feeds, or when programmatically generating ROLIE entries from some data source that the author’s personal info is not shared without their consent.

When using the Atom Publishing Protocol to POST entries to a feed, attackers may use correlating techniques to profile the user. The request time can be compared to the generated "updated" field of the entry in order to build out information about a given user. This correlation attempt can be mitigated by not using HTTP requests to POST entries when profiling is a risk, and rather use backend control of the Feeds.

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

11. Acknowledgements

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

The authors would also like to thank the MILE Working Group, the SACM Working Group, and countless other people from both within the IETF community and outside of it for their excellent review and effort towards constructing this draft.

12. References

12.1. Normative References


12.2. Informative References


12.3. URIs

[1] https://www.iana.org/assignments/link-relations/link-relations.xhtml

[2] https://www.iana.org/assignments/link-relations/link-relations.xhtml


Appendix A. Relax NG Compact Schema for ROLIE

This appendix is informative.

The Relax NG schema below defines the rolie:format element.
namespace rolie = "urn:ietf:params:xml:ns:rolie-1.0"

# import the ATOM Syndication RELAX NG Compact Syntax Grammar
include "atomsynd.rnc"

# rolie:format
rolieFormat =
element rolie:format {
    atomCommonAttributes,
    attribute ns { atomUri }?,
    attribute version { text } ?,
    attribute schema-location { atomUri } ?,
    attribute schema-type { atomMediaType } ?,
    empty
}

# rolie:property
rolieProperty =
element rolie:property {
    atomCommonAttributes,
    attribute name { atomUri },
    attribute value { text },
    empty
}

Appendix B.  Examples of Use

B.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 provider uses an Atom Service Document to enable authorized clients to determine what specific information the provider makes available to the community.  The Service Document should be made accessible from an easily found location, such as a link from the producer’s home page.

A client may format an HTTP GET request to retrieve the service document from the specified location:
GET /rolie/servicedocument
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 Collections that are provided.

Example HTTP GET response:

HTTP/1.1 200 OK
Date: Fri, 24 Aug 2016 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>
        <atom:title type="text">Vulnerabilities</atom:title>
        <collection href="https://example.org/provider/vulns">
            <atom:title type="text">Vulnerabilities Feed</atom:title>
            <categories fixed="yes">
                <atom:category
                    scheme="urn:ietf:params:rolie:category:information-type"
                    term="vulnerability"/>
            </categories>
        </collection>
    </workspace>
</service>

This simple Service Document example shows that the server provides one workspace, named "Vulnerabilities". Within that workspace, the server makes one Collection available.

A server may also offer a number of different Collections, each containing different types of security automation information. In the following example, a number of different Collections are provided, each with its own category and authorization scope. This categorization will help the clients to decide which Collections will meet their needs.
<collection>
  <atom:link rel="service" href="https://example.org/rolie/servicedocument"/>
  <categories fixed="yes">
    <atom:category scheme="urn:ietf:params:rolie:category:information-type" term="vulnerability"/>
  </categories>
</collection>
</workspace>

<atom:title>Private Consortium Sharing</atom:title>
<collection>
  <atom:link rel="service" href="https://example.org/rolie/servicedocument"/>
  <categories fixed="yes">
    <atom:category scheme="urn:ietf:params:rolie:category:information-type" term="incident"/>
  </categories>
</collection>
</workspace>
</service>

In this example, the provider is making available a total of two Collections, organized into two different workspaces. The first workspace contains a Collection consisting of publicly available software vulnerabilities. The second workspace provides an incident Collection for use by a private sharing consortium. An appropriately authenticated and authorized client may then proceed to make HTTP requests for these Collections. The publicly provided vulnerability information may be accessible with or without authentication. However, users accessing the Collection restricted to authorized...
members of a private sharing consortium are expected to authenticate before access is allowed.

B.2. Feed Retrieval

This section provides a non-normative example of a client retrieving an vulnerability Feed.

Having discovered the available security information sharing Collections, a client who is a member of the general public may be interested in receiving the Collection of public vulnerabilities. The client may retrieve the Feed for this Collection by performing an HTTP GET operation on the URL indicated by the Collection’s “href” attribute.

Example HTTP GET request for a Feed:

```
GET /provider/public/vulns
Host: www.example.org
Accept: application/atom+xml
```

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

Example HTTP GET response for a Feed:
HTTP/1.1 200 OK
Date: Fri, 24 Aug 2016 17:20:11 GMT
Content-Length: 2882
Content-Type: application/atom+xml;charset="utf-8"

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
     xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0"
     xml:lang="en-US">
  <id>2a7e265a-39bc-43f2-b711-b8fd9264b5c9</id>
  <title type="text">
    Atom formatted representation of a feed of XML vulnerability documents
  </title>
  <category
    scheme="urn:ietf:params:rolie:category:information-type"
    term="vulnerability"/>
  <updated>2016-05-04T18:13:51.0Z</updated>
  <link rel="self"
        href="https://example.org/provider/public/vulns" />
  <link rel="service"
        href="https://example.org/rolie/servicedocument"/>
  <entry>
    <rolie:format ns="urn:ietf:params:xml:ns:exampleformat"/>
    <id>dd786dba-88e6-440b-9158-b8fae67ef67c</id>
    <title>Sample Vulnerability</title>
    <published>2015-08-04T18:13:51.0Z</published>
    <updated>2015-08-05T18:13:51.0Z</updated>
    <summary>A vulnerability issue identified by CVE-...</summary>
    <content type="application/xml"
             src="https://example.org/provider/vulns/123456/data"/>
  </entry>
  <!-- ...another entry... -->
<entry>
</feed>

This Feed document has two Atom Entries, one of which has been elided. The first Entry illustrates an atom:entry element that provides a summary of essential details about one particular vulnerability. Based upon this summary information and the provided category information, a client may choose to do an HTTP GET request, on the content "src" attribute, to retrieve the full details of the vulnerability.
B.3. Entry Retrieval

This section provides a non-normative example of a client retrieving an vulnerability 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 vulnerability Entry by performing an HTTP GET operation on the URL indicated by the "src" attribute of the atom:content element.

Example HTTP GET request for an Entry:

```
GET /provider/public/vulns/123456
Host: www.example.org
Accept: application/atom+xml;type=entry
```

The corresponding HTTP response would be an XML document containing the Atom Entry for the vulnerability record:

Example HTTP GET response for an Entry:

```
HTTP/1.1 200 OK
Date: Fri, 24 Aug 2016 17:30:11 GMT
Content-Length: 713
Content-Type: application/atom+xml;type=entry;charset=utf-8

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom"
  xmlns:rolie="urn:ietf:params:xml:ns:rolie-1.0"
  xml:lang="en-US">
  <id>f63aafa9-4082-48a3-9ce6-97a2d69d4a9b</id>
  <title>Sample Vulnerability</title>
  <published>2015-08-04T18:13:51.0Z</published>
  <updated>2015-08-05T18:13:51.0Z</updated>
  <category
    scheme="urn:ietf:params:rolie:category:information-type"
    term="vulnerability"/>
  <summary>A vulnerability issue identified by CVE-..."</summary>
  <rolie:format ns="urn:ietf:params:xml:ns:exampleformat"/>
  <content type="application/xml"
    src="https://example.org/provider/vulns/123456/data">
  </content>
</entry>
```

The example response above shows an XML document referenced by the "src" attribute of the atom:content element. The client may retrieve the document using this URL.
Appendix C. Change History

Changes in draft-ietf-mile-rolie-14 since draft-ietf-mile-rolie-13 revision:
- Removed /.well-known registration and updated Discovery text.
- Fixed small namespacing error in RNC schema.

Changes in draft-ietf-mile-rolie-13 since draft-ietf-mile-rolie-12 revision:
- Adjusted .well-known registration.
- Updated IANA Consideration text.

Changes in draft-ietf-mile-rolie-11 since draft-ietf-mile-rolie-09 revision:
- Incorporated ART last call review and AD review changes.

Changes in draft-ietf-mile-rolie-09 since draft-ietf-mile-rolie-08 revision:
- TLS requirements changed to clarify TLS versioning and recommendations
- Informative references and textual discussion added to Security Considerations around HTTP Authentication and content Signing/Encryption.
- IANA Expert review clarified.
- Editorial changes from AD review/WGLC.

Changes in draft-ietf-mile-rolie-08 since draft-ietf-mile-rolie-07 revision:
- Reworked "usage of app:collection" and "usage of atom:feed" sections to clarify ROLIE vs non-ROLIE collections/feeds
- Removed requirement from Security Considerations that was a duplicate of text earlier in the document
- TLS requirement clarifications around mutual authentication
- Clarified requirements around support for the "/" resource
Added IANA property registrations for content-id, content-published-date, and content-updated-date that can be used across all ROLIE extensions to increase consistency/interop

Assorted editorial changes

Changes in draft-ietf-mile-rolie-07 since draft-ietf-mile-rolie-06 revision:

Condensed and re-focused Sections 1 and 4 to be more concise.

Added /.well-known/ registration and requirement for service discovery.

Added local category, property namespace, and additional property registrations

Added privacy considerations section.

Made a number of editorial changes as per WGLC review.

Changes in draft-ietf-mile-rolie-06 since draft-ietf-mile-rolie-05 revision:

Changed to standards track

Added the rolie:property element

Fixed references (Normative vs Informative)

Set Service and Category document URL template requirements

Fixed XML snippets in examples

Changes in draft-ietf-mile-rolie-05 since draft-ietf-mile-rolie-04 revision:

Added ROLIE specific terminology to section 2

Added AtomPub Category Document in section 5.2

Edited document, improving consistency in terminology usage and capitalization of key terms, as well as enhancing clarity.

Removed unused format parameter type in section 8.3

Schema removed, the normative schema consists of the snippets in the requirements sections.
Changes in draft-ietf-mile-rolie-04 since draft-ietf-mile-rolie-03 revision:

- Further specification and clarification of requirements
- IANA Considerations and extension system fleshed out and described.
- Examples and References updated.
- Schema created.
- Fixed both internal section and external document referencing.
- Removed XACML Guidance Appendix. This will be added to a future draft on ROLIE Authentication and Access Control.

Changes made in draft-ietf-mile-rolie-03 since draft-ietf-mile-rolie-02 revision:

- Atom Syndication and Atom Pub requirements split and greatly expanded for increased justification and technical specification.
- Reintroduction and reformatting of some use case examples in order to provide some guidance on use.
- Established rough version of IANA table extension system along with explanations of said system.
- Re-organized document to put non-vital information in appendices.

Changes made in draft-ietf-mile-rolie-02 since draft-field-mile-rolie-01 revision:

- All CSIRT and IODEF/RID material moved to companion CSIRT document
- Recast document into a more general use perspective. The implication of CSIRTs as the defacto end-user has been removed where ever possible. All of the original CSIRT based use cases remain completely supported by this document, it has been opened up to support many other use cases.
- Changed the content model to broaden support of representation
- Edited and rewrote much of sections 1, 2 and 3 in order to accomplish a broader scope and greater readability
- Removed any requirements from the Background section and, if not already stated, placed them in the requirements section.
- Re-formatted the requirements section to make it clearer that it contains the lion's-share of the requirements of the specification.

Changes made in draft-ietf-mile-rolie-01 since draft-field-mile-rolie-02 revision:

- Added section specifying the use of RFC5005 for Archive and Paging of Feeds.
- Added section describing use of atom categories that correspond to IODEF expectation class and impact classes. See: normative-expectation-impact
- Dropped references to adoption of a MILE-specific HTTP media type parameter.
- Updated IANA Considerations section to clarify that no IANA actions are required.

Authors' Addresses

John P. Field
Pivotal Software, Inc.
625 Avenue of the Americas
New York, New York
USA

Phone: (646) 792-5770
Email: jfield@pivotal.io

Stephen A. Banghart
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, Maryland
USA

Phone: (301) 975-4288
Email: stephen.banghart@nist.gov
David Waltermire
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, Maryland  20877
USA

Email: david.waltermire@nist.gov
Abstract

This document describes how the Extensible Messaging and Presence Protocol (XMPP) [RFC7590] can be used as the framework as transport protocol for collecting and distributing any security telemetry information between any network connected device. As an example, this document describes how XMPP can be used to transport the Incident Object Description Exchange Format (IODEF) information.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 4, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of
Internet-Draft                  XMPP Grid                      July 2017

the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1. Introduction  . . . . . . . . . . . . . . . . . . . . . . . .   2
   1.1. Glossary of Terms . . . . . . . . . . . . . . . . . . . .   3
   1.2. Overview of XMPP-Grid . . . . . . . . . . . . . . . . . .   4
   1.3. Benefits of XMPP-Grid . . . . . . . . . . . . . . . . . .   5
2. XMPP-Grid Architecture . . . . . . . . . . . . . . . . . . . .   6
   2.1. Using XMPP . . . . . . . . . . . . . . . . . . . . . . . .   7
   2.2. XMPP-Grid Requirements for enabling Information Sharing .   8
3. Example use of XMPP-Grid for IODEF . . . . . . . . . . . . . . .   9
4. IANA Considerations . . . . . . . . . . . . . . . . . . . . . .  11
5. Security Considerations . . . . . . . . . . . . . . . . . . . .  11
   5.1. Trust Model . . . . . . . . . . . . . . . . . . . . . . . .  11
      5.1.1. Network . . . . . . . . . . . . . . . . . . . . . . . .  12
      5.1.2. XMPP-Grid Nodes . . . . . . . . . . . . . . . . . . .  12
      5.1.3. XMPP-Grid Controller . . . . . . . . . . . . . . . .  12
      5.1.4. Certification Authority . . . . . . . . . . . . . . .  12
   5.2. Threat Model . . . . . . . . . . . . . . . . . . . . . . .  13
      5.2.1. Network Attacks . . . . . . . . . . . . . . . . . . .  13
      5.2.2. XMPP-Grid Nodes . . . . . . . . . . . . . . . . . . .  14
      5.2.3. XMPP-Grid Controllers . . . . . . . . . . . . . . . .  15
      5.2.4. Certification Authority . . . . . . . . . . . . . . .  16
   5.3. Countermeasures . . . . . . . . . . . . . . . . . . . . . .  17
      5.3.1. Securing the XMPP-Grid Transport Protocol . . . . . .  17
      5.3.2. Securing XMPP-Grid Nodes . . . . . . . . . . . . . .  18
      5.3.3. Securing XMPP-Grid Controllers . . . . . . . . . . .  18
      5.3.4. Limit on search result size . . . . . . . . . . . . .  19
      5.3.5. Cryptographically random session-id and
              authentication checks for ARC . . . . . . . . . .  19
      5.3.6. Securing the Certification Authority . . . . . . . .  20
   5.4. Summary . . . . . . . . . . . . . . . . . . . . . . . . . .  20
6. Privacy Considerations . . . . . . . . . . . . . . . . . . . . .  21
7. Acknowledgements . . . . . . . . . . . . . . . . . . . . . . . .  21
8. References . . . . . . . . . . . . . . . . . . . . . . . . . . . .  21
   8.1. Normative References . . . . . . . . . . . . . . . . . . .  21
   8.2. Informative References . . . . . . . . . . . . . . . . . .  22
Authors’ Addresses . . . . . . . . . . . . . . . . . . . . . . . . .  23

1. Introduction

XMPP-Grid is intended for use as a secure transport and
communications ecosystem for devices, applications and organizations
to interconnect, forming an information grid for the exchange of
formatted data (e.g. XML, JSON, etc). This document describes how
XMPP [RFC7590] serves as the framework and protocols 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

Capability Provider

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

Publisher

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

Subscriber

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

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 of subscribed devices.

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 Node

Platform or device that implements XMPP to connect to XMPP-Grid and share or consume security data.
1.2. Overview of XMPP-Grid

XMPP-Grid employs publish/subscribe/query operations brokered by a controller, which enforces access control in the system. This XMPP-based architecture controls what platforms can connect to the "Grid" to share ("publish") and/or consume ("subscribe" or "query") contextual information ("Topics") such as security data needed to support MILE.

Leveraging the XMPP architecture, XMPP-Grid uses the XMPP server to act as a controller, affecting the authentication and authorization of participating XMPP-Grid nodes (Node). Security information may only be published or consumed by authenticated and authorized Nodes using the XMPP publish/subscribe extension defined in [XEP-0060].

The components of XMPP-Grid are:

- **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 Node (Node):** A Node is a platform or application that has mutually authenticated with the Controller and obtained authorization by the Controller 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) and a specified interface by which the data can be obtained.

As enabled by the XMPP architecture, 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:
1 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.

2 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.

3 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.

1.3 Benefits of XMPP-Grid

Currently, security information standards such as IODEF [RFC7970] defines a data models that has no explicit transport defined and typically are carried over HTTPS as defined in RID [RFC6545].

As security solutions are expanding to expose and share information asynchronously and across network boundaries there is a need for an architecture that facilitates federation, discovery of the different information available, the interfaces used to obtain the information and the need for near real-time exchange of data.

Based on XMPP, XMPP-Grid has been defined to meet those requirements.
2. XMPP-Grid Architecture

XMPP-Grid is an XMPP-based 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 (see figure below).

![XMPP-Grid Architecture Diagram]

Figure 1: XMPP-Grid Architecture
Nodes must connect to the XMPP-Grid controller to authenticate and establish appropriate authorizations, with appropriate authorization privileges. The control plane messaging is established through XMPP and shown as "A" (Control plane interface) in Figure 1. Authorized nodes may then share data either thru the XMPP-Grid Controller (shown as "B" in Figure 1) or directly (shown as "C" in Figure 1). The data messaging enable Nodes to:

- Receive real-time events of the published messages from the publisher through Topic subscriptions
- Make directed queries to other Nodes in the XMPP-Grid with appropriate authorization from the Controller
- Negotiate out-of-band secure file transfer channel with the peer

2.1. Using XMPP

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 the XMPP working group:


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

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

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

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.

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

* 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

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

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

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

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 [RFC7590] and the presence in [RFC6121]. A full list of XMPP Extension Protocols (XEPs) [RFC7590] can be found in http://xmpp.org/extensions/xep-0001.html.

2.2. XMPP-Grid Requirements for enabling Information Sharing

This section summarizes the requirements and the extensions used to facilitate the secure sharing of information using XMPP. Knowledge
of the XMPP Protocol and extensions is required to understand this section.

- **Authentication and Authorization**: Nodes participating in XMPP-Grid MUST mutually authenticate to the controller using XMPP’s authentication mechanisms. Authorization is affected by the controller.

- **Topic Discovery**: to facilitate dynamic discovery, Nodes SHOULD support the XMPP Service Discovery [XEP-0030].

- **Publish/Subscribe**: to facilitate unsolicited notifications to new or updated security information, Nodes MUST support the XMPP Publish/Subscribe protocol as defined in [RFC7590].

Once a Node has authenticated with the XMPP-Grid controller, it may further register a topic (e.g. information type) to be shared or use the discovery mechanism for determining topics to be consumed.

**Sharing Information**: security information may be shared using registered topics. An example for sharing or consuming the IODEF 1.0 is defined in [XEP-0268].

### 3. Example use of XMPP-Grid for IODEF

A Node follows the standard XMPP workflow for connecting to the Controller as well as using the XMPP discovery mechanisms to discover the availability to consume IODEF information. The general workflow is summarized in the figure below:
Figure 2: IODEF Example XMPP Workflow

An example XMPP discovery request for an IODEF 1.0 topic is shown below:

```
<iq type='get' from='iodefclientabc@company.com' to='pubsub.company.com' id='nodes1'>
```

An example XMPP discovery response for an IODEF 1.0 topic is shown below:
4. IANA Considerations

IODEF extensions as defined in [XEP-0268] may require IANA considerations and assignment thru the IODEF IANA rules.

5. Security Considerations

An 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:

- Perform best effort delivery of network traffic

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

- Provide confidentiality or integrity protection for messages sent over it
- Provide timely or reliable service

5.1.2. XMPP-Grid Nodes

Authorized XMPP-Grid Nodes are trusted to:

- 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:

- Broker requests for data and enforce authorization of access to this data throughout its lifecycle
- Perform service requests in a timely and accurate manner
- Create and maintain accurate operational attributes
- Only reveal data to and accept service requests from authorized parties

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

- 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:

- Ensure that only proper certificates are issued and that all certificates are issued in accordance with the CA’s policies
Revoke certificates previously issued when necessary

Regularly and securely distribute certificate revocation information

Promptly detect and report any violations of this trust so that they can be handled

The CA is not expected (trusted) to:

- Issue certificates that go beyond the XMPP-Grid needs or other constraints imposed by a relying party.

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.

- Network traffic may be passively monitored to glean information from any unencrypted traffic

- Even if all traffic is encrypted, valuable information can be gained by traffic analysis (volume, timing, source and destination addresses, etc.)

- Network traffic may be modified in transit

- Previously transmitted network traffic may be replayed

- New network traffic may be added

- Network traffic may be blocked, perhaps selectively

- 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
5.2.2. XMPP-Grid Nodes

An unauthorized XMPP-Grid Node (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:

- 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
- Omit important actions (such as posting incriminating data), resulting in incorrect access
- Use confidential information obtained from the XMPP-Grid Controller to enable further attacks (such as using endpoint health check results to exploit vulnerable endpoints)
- Advertise data crafted to exploit vulnerabilities in the XMPP-Grid Controller or in other XMPP-Grid Nodes, with a goal of compromising those systems
- 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
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 an 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 an 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:

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

- 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
o 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:

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

o 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.

o 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.

o 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’s certificate. 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 an 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, an XMPP-Grid Controller MAY respond to any request that lacks credentials with an error code similar to HTTP code 401. An 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, an 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, an 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 an 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 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 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 an 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

An XMPP-Grid Controller SHOULD ensure that the XMPP-Grid Node establishing an Authenticated Results Chain (ARC) is the same XMPP-Grid Node as the XMPP-Grid Node that established the corresponding Synchronization Source Identifier (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 an 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 an 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. In addition, we want to thank Takeshi Takahashi, Panos Kampanakis, Adam Montville and Chris Inacio for reviewing and providing valuable comments.

8. References

8.1. Normative References


Cam-Winget, et al.
Expires January 4, 2018
8.2. Informative References

[I-D.ietf-mile-rolie]

[I-D.ietf-uta-tls-bcp]


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
This document describes how to use the Extensible Messaging and Presence Protocol (XMPP) to collect and distribute security-relevant information between network-connected devices. To illustrate the principles involved, this document describes such a usage for the Incident Object Description Exchange Format (IODEF).

Status of This Memo

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

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

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

This Internet-Draft will expire on December 28, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must...
This document describes "XMPP-Grid": a method for using the Extensible Messaging and Presence Protocol (XMPP) [RFC6120] to collect and distribute security-relevant information among network platforms, endpoints, and any other network-connected device. Among other things, XMPP provides a publish-subscribe service [XEP-0060] that acts as a broker, enabling control-plane functions by which entities can discover available information to be published or consumed. Although such information can take the form of any structured data (XML, JSON, etc.), this document illustrates the principles of XMPP-Grid with examples that use the Incident Object Description Exchange Format (IODEF) [RFC7970].

This document uses XMPP terminology defined in [RFC6120] and [XEP-0060] as well as Security Automation and Continuous Monitoring (SACM) terminology defined in [I-D.ietf-sacm-terminology]. Because the intended audience for this document is those who implement and deploy security reporting systems, in general the SACM terms are used...
(however, mappings are provided for the benefit of XMPP developers and operators).

Broker: In SACM, a specific type of controller containing control plane functions; as used here, the term refers to an XMPP publish-subscribe service.

Broker Flow: In SACM, a method by which security-related information is published and consumed in a mediated fashion through a Broker. In this flow, the Broker handles authorization of Consumers and Providers to Topics, receives messages from Providers, and delivers published messages to Consumers.

Consumer: In SACM, an entity that contains functions to receive information from other components; as used here, the term refers to an XMPP publish-subscribe Subscriber.

Controller: In SACM, a "component containing control plane functions that manage and facilitate information sharing or execute on security functions"; as used here, the term refers to an XMPP server, which provides core message delivery [RFC6120] used by publish-subscribe entities.

Node: The XMPP term for a Topic.

Platform: Any entity that connects to the XMPP-Grid in order to publish or consume security-related data.

Provider: In SACM, an entity that contains functions to provide information to other components; as used here, the term refers to an XMPP publish-subscribe Publisher.

Publisher: The XMPP term for a Provider.

Publish-Subscribe Service: The XMPP term for the kind Broker discussed here.

Subscriber: The XMPP term for a Consumer.

Topic: A contextual information channel created on a Broker at which messages generated by a Provider are propagated in real time to one or more Consumers. Each Topic is limited to a specific type and format of security data (e.g., IODEF) and provides an XMPP interface by which the data can be obtained.

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].
3. Architecture

The following figure illustrates the architecture of XMPP-Grid.

Platforms connect to the Controller (XMPP server) to authenticate and then establish appropriate authorizations and relationships (e.g., Provider or Consumer) at the Broker. The control plane messaging is established through XMPP and shown as "A" (control plane interface) in Figure 1. Authorized nodes can then share data either thru the
Broker (shown as "B" in Figure 1) or in some cases directly (shown as "C" in Figure 1). This document focuses primarily on the Broker Flow for information sharing ("direct flow" interactions can be used for specialized purposes such as bulk data transfer, but methods for doing so are outside the scope of this document).

4. Workflow

A typical XMPP-Grid workflow is as follows:

a. A Platform with a source of security data requests connection to the XMPP-Grid via a Controller (XMPP server).

b. The Controller authenticates the Platform.

c. The Platform establishes authorized privileges (e.g. privilege to publish and/or subscribe to security data Topics) with a Broker.

d. The Platform can publish security-related data to a Topic, subscribe to a Topic, query a Topic, or any combination of these operations.

e. A Provider unicasts its Topic updates to the Grid in real time through a Broker. The Broker handles replication and distribution of the Topic to Consumers. A Provider can publish the same or different data to multiple Topics.

f. Any Platform on the Grid can subscribe to any Topics published to the Grid (as permitted by authorization policy), and as Consumers will then receive a continual, real-time stream of updates from the Topics to which it is subscribed.

The general workflow is summarized in the figure below:
Figure 2: IODEF Example Workflow
XMPP-Grid implementations MUST adhere to the mandatory-to-implement and mandatory-to-negotiate features as defined in [RFC6120]. Similarly, implementations MUST implement [XEP-0060] to facilitate the asynchronous sharing for information. The Service Discovery per [XEP-0030] SHOULD be implemented to facilitate the means to dynamically discover the available information (Topics) to be published or consumes.

The following sections provide protocol examples for the service discovery and publish-subscribe parts of the workflow.

5. Service Discovery

Using the XMPP service discovery extension [XEP-0030], a Controller enables Platforms to discover what information can be consumed through the Broker, and at which Topics. As an example, the Controller at ‘security-grid.example’ might provide a Broker at ‘broker.security-grid.example’ hosting a number of Topics. A Platform at ‘xmpp-grid-client@mile-host.example’ would query the Broker about its available Topics by sending an XMPP "disco#items" request to the Broker:

<iq type='get'
    from='xmpp-grid-client@mile-host.example/2EBE702A97D6'
    to='broker.security-grid.example'
    id='B3C17F7B-B9EF-4ABA-B08D-805DA9F34626'>
    <query xmlns='http://jabber.org/protocol/disco#items'/>
</iq>

The Broker responds with the Topics it hosts:

<iq type='result'
    from='broker.security-grid.example'
    to='xmpp-grid-client@mile-host.example/2EBE702A97D6'
    id='B3C17F7B-B9EF-4ABA-B08D-805DA9F34626'>
    <query xmlns='http://jabber.org/protocol/disco#items'>
        <item node='NEA1'
            name='Endpoint Posture Information'
            jid='broker.security-grid.example'/>
        <item node='MILEHost'
            name='MILE Host Data'
            jid='broker.security-grid.example'/>
    </query>
</iq>

In order to determine the exact nature of each Topic (i.e., in order to find topics that publish incidents in the IODEF format), a Platform would send an XMPP "disco#info" request to each Topic:
The Broker responds with the "disco#info" description, which SHOULD include an XMPP Data Form [XEP-0004] including a 'pubsub#type' field that specifies the supported namespace (in this example, the IODEF namespace defined in [RFC7970]):

6. Publish-Subscribe

Using the XMPP publish-subscribe extension [XEP-0030], a Consumer subscribes to a Topic and a Provider publishes information to that Topic, which the Broker then distributes to all subscribed Consumers.

First, a Provider would create a Topic as follows:

```xml
<iq type='set'
    from='datasource@provider.example/F12C2EFC9BB0'
    to='broker.security-grid.example'
    id='A67507DF-2F22-4937-8D30-88D2F7DBA279'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <create node='MILEHost'/>
  </pubsub>
</iq>
```
Note: The foregoing example is the minimal protocol needed to create a Topic with the default node configuration on the XMPP publish-subscribe service specified in the 'to' address of the creation request stanza. Depending on security requirements, the Provider might need to request a non-default configuration for the node; see [XEP-0060] for detailed examples.

Unless an error occurs (see [XEP-0060] for various error flows), the Broker responds with success:

```xml
<iq type='result' from='broker.security-grid.example' to='datasource@provider.example/F12C2EFC9BB0' id='A67507DF-2F22-4937-8D30-88D2F7DBA279'/>
```

Second, a Consumer would subscribe as follows:

```xml
<iq type='set' from='xmpp-grid-client@mile-host.example/2EBE702A97D6' to='broker.security-grid.example' id='9C6EEE9E-F09A-4418-8D68-3BA6AF852522'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <subscribe node='MILEHost' jid='xmpp-grid-client@mile-host.example'/>
  </pubsub>
</iq>
```

Unless an error occurs (see [XEP-0060] for various error flows), the Broker responds with success:

```xml
<iq type='result' from='broker.security-grid.example' to='xmpp-grid-client@mile-host.example/2EBE702A97D6' id='9C6EEE9E-F09A-4418-8D68-3BA6AF852522'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <subscription node='MILEHost' jid='xmpp-grid-client@mile-host.example' subscription='subscribed'/>
  </pubsub>
</iq>
```

Third, a Provider would publish an incident as follows:
<iq type='set' from='datasource@provider.example/F12C2EFC9BB0' to='broker.security-grid.example' id='2A17D283-0DAE-4A6C-85A9-C10B1B40928C'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <publish node='MILEHost'>
      <item id='8bh1g27skbga47fh9wk7'>
        <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="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>
                <EmailTo>contact@csirt.example.com</EmailTo>
              </Email>
            </Contact>
          </Incident>
        </IODEF-Document>
      </item>
    </publish>
  </pubsub>
</iq>

(The payload in the foregoing example is from [RFC7970]; payloads for additional use cases can be found in [RFC8274].)

The Broker would then deliver that incident report to all Consumers who are subscribe to the Topic:
<message
from='broker.security-grid.example'
to='xmpp-grid-client@mile-host.example/2EBE702A97D6'
id='37B3921D-4F7F-450F-A589-56119A88BC2E'>
<event xmlns='http://jabber.org/protocol/pubsub#event'>
<items node='MILEHost'>
=item id='iah37s61s964gguqy47aksbx9453ks77'>
<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="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>
<EmailTo>contact@csirt.example.com</EmailTo>
</Email>
</Contact>
</Incident>
</IODEF-Document>
</item>
</items>
</event>
</message>

7. IANA Considerations

This document has no actions for IANA.

8. Security Considerations

An XMPP-Grid Controller serves as an controlling broker for XMPP-Grid Platforms 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 Platforms 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 data transfer 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 can be mounted on the system), and the countermeasures (ways to address or mitigate the threats previously identified).

8.1. 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.

8.1.1. Network

The network used to carry XMPP-Grid messages (i.e., the underlying network transport layer over which XMPP runs) is trusted to:

- Perform best effort delivery of network traffic

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

- Provide confidentiality or integrity protection for messages sent over it
- Provide timely or reliable service

8.1.2. XMPP-Grid Platforms

Authorized XMPP-Grid Platforms are trusted to:

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

8.1.3. XMPP-Grid Controller

The XMPP-Grid Controller (including its associated Broker) is trusted to:

- Broker requests for data and enforce authorization of access to this data throughout its lifecycle
- Perform service requests in a timely and accurate manner
- 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

8.1.4. Certification Authority

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

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 the XMPP-Grid needs or other constraints imposed by a relying party.

8.2. Threat Model

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

8.2.1. Network Attacks

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

- Network traffic can be passively monitored to glean information from any unencrypted traffic
Even if all traffic is encrypted, valuable information can be gained by traffic analysis (volume, timing, source and destination addresses, etc.)

- Network traffic can be modified in transit
- Previously transmitted network traffic can be replayed
- New network traffic can be added
- Network traffic can be blocked, perhaps selectively
- A "Man In The Middle" (MITM) attack can 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
- Resist attacks (including denial of service and other attacks from XMPP-Grid Platforms)
- Undesired network traffic can be sent in an effort to overload an architectural component, thus mounting a denial of service attack

8.2.2. XMPP-Grid Platforms

An unauthorized XMPP-Grid Platform (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 Platform, on the other hand, can mount many attacks. These attacks might occur because the XMPP-Grid Platform is controlled by a malicious, careless, or incompetent party (whether because its owner is malicious, careless, or incompetent or because the XMPP-Grid Platform has been compromised and is now controlled by a party other than its owner). They might also occur because the XMPP-Grid Platform is running malicious software; because the XMPP-Grid Platform is running buggy software (which can fail in a state that floods the network with traffic); or because the XMPP-Grid Platform 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 can be mounted by an authorized XMPP-Grid Platform:

- 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

- Omit important actions (such as posting incriminating data), resulting in incorrect access
- Use confidential information obtained from the XMPP-Grid Controller to enable further attacks (such as using endpoint health check results to exploit vulnerable endpoints)
- Advertise data crafted to exploit vulnerabilities in the XMPP-Grid Controller or in other XMPP-Grid Platforms, with a goal of compromising those systems
- 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 Platform, and/or the network
- Establish a communication channel using another XMPP-Grid Platform’s session-id

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

8.2.3. XMPP-Grid Controllers

An unauthorized XMPP-Grid Controller (one which is not trusted by XMPP-Grid Platforms) 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 Platform case described above, these attacks might occur because the XMPP-Grid Controller is controlled by a malicious, careless, or incompetent party (either an 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 can 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 Platform 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 can include:

- Expose different data to different XMPP-Grid Platforms to mislead investigators or cause inconsistent behavior
- Mount an even more effective denial of service attack than a single XMPP-Grid Platform could
- Obtain and cache XMPP-Grid Platform credentials so they can be used to impersonate XMPP-Grid Platforms 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 can be exploited to obtain control of the XMPP-Grid Controller and effect these attacks.

8.2.4. Certification Authority

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

- Issue certificates for unauthorized parties, enabling them to impersonate authorized parties such as the XMPP-Grid Controller or an XMPP-Grid Platform. 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 can 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.
o Fail to regularly and securely distribute certificate revocation information. This can 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.

8.3. Countermeasures

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

8.3.1. Securing the XMPP-Grid Data Transfer Protocol

To address network attacks, the XMPP-Grid data transfer protocol described in this document requires that the XMPP-Grid messages MUST be carried over TLS (minimally TLS 1.2 [RFC5246]) as described in [RFC6120] and updated by [RFC7590]. The XMPP-Grid Platform MUST verify the XMPP-Grid Controller's certificate and determine whether the XMPP-Grid Controller is trusted by this XMPP-Grid Platform before completing the TLS handshake. The XMPP-Grid Controller MUST authenticate the XMPP-Grid Platform either using the SASL EXTERNAL mechanism or using the SASL SCRAM mechanism (with the SCRAM-SHA-256-PLUS variant being preferred over the SCRAM-SHA-256 variant and SHA-256 variants [RFC7677] being preferred over SHA-1 variants [RFC5802]). XMPP-Grid Platforms and XMPP-Grid Controllers using mutual certificate-based authentication SHOULD each verify the revocation status of the other party's certificate. All XMPP-Grid Controllers and XMPP-Grid Platforms MUST implement both SASL EXTERNAL and SASL SCRAM. The selection of which XMPP-Grid Platform authentication technique to use in any particular deployment is left to the administrator.

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 can be
employed. In addition, an XMPP-Grid Controller MAY automatically
disable a misbehaving XMPP-Grid Platform.

8.3.2. Securing XMPP-Grid Platforms

XMPP-Grid Platforms can be deployed in locations that are susceptible
to physical attacks. Physical security measures can be taken to
avoid compromise of XMPP-Grid Platforms, but these are not always
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 Platforms can 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 Platform as a DHCP server and
authorize only the operations and metadata types needed by a DHCP
server to be permitted for that XMPP-Grid Platform. These techniques
can reduce the negative impacts of a compromised XMPP-Grid Platform
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 Platform behavior. XMPP-Grid Controllers SHOULD
make it easy to revoke an XMPP-Grid Platform’s authorization when
necessary. Another way to detect attacks from XMPP-Grid Platforms is
to create fake entries in the available data (honeytokens) which
normal XMPP-Grid Platforms will not attempt to access. The XMPP-Grid
Controller SHOULD include auditable logs of XMPP-Grid Platform
activities.

To avoid compromise of XMPP-Grid Platform, XMPP-Grid Platform SHOULD
be hardened against attack and minimized to reduce their attack
surface. They should be well managed to minimize vulnerabilities in
the underlying platform and in systems upon which the XMPP-Grid
Platform depends. Personnel with administrative access should be
carefully screened and monitored to detect problems as soon as
possible.

8.3.3. Securing XMPP-Grid Controllers

Because of the serious consequences of XMPP-Grid Controller
compromise, XMPP-Grid Controllers need to be especially well hardened
against attack and minimized to reduce their attack surface. They
need to 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 can be used to monitor and limit traffic to and from the XMPP-Grid Controller. Personnel with administrative access ought to 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 ought to 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 Platforms). XMPP-Grid Platforms 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 can have been compromised should be reissued.

8.3.4. Broker Access Models for Topics

The XMPP publish-subscribe specification [XEP-0060] defines five access models for subscribing to Topics at a Broker: open, presence, roster, authorize, and whitelist. The first model allows uncontrolled access and the next two models are appropriate only in instant-messaging applications. Therefore, a Broker SHOULD support only the authorize model (under which the Topic owner needs to approve all subscription requests and only subscribers can retrieve data items) and the whitelist model (under which only preconfigured Platforms can subscribe or retrieve data items). In order to ease the deployment burden, subscription approvals and whitelist management can be automated (e.g., the Topic "owner" can be a policy server). The choice between "authorize" and "whitelist" as the default access model is a matter for local service policy.

8.3.5. Limit on Search Result Size

While XMPP-Grid is designed for high scalability to 100,000s of Platforms, 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 an XMPP-Grid Platform causing resource exhaustion by issuing a search or subscription that leads to an enormous result.
8.3.6. 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 Platforms 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 can 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 can still be consulted if a pinned certificate is revoked, since the CA and revocation process are not completely trusted.

8.4. 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 data transfer protocol provides strong protection against a variety of different attacks. In the event that an XMPP-Grid Platform 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.
9. Privacy Considerations

XMPP-Grid Platforms can publish information about endpoint health, network access, events (which can 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 can be queried by other XMPP-Grid Platforms and could potentially be used to correlate network activity to a particular end user.

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

Care needs to be taken by deployers of XMPP-Grid to ensure that the information published by XMPP-Grid Platforms does not violate agreements with end users or local and regional laws and regulations. This can be accomplished either by configuring XMPP-Grid Platforms to not publish certain information or by restricting access to sensitive data to trusted XMPP-Grid Platforms. 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.

10. Operations and Management Considerations

In order to facilitate the management of Providers and the onboarding of Consumers, it is helpful to generate the following ahead of time:

- Agreement between the operators of Provider services and the implementers of Consumer software regarding identifiers for common Topics (e.g., these could be registered with the XMPP Software Foundation’s registry of well-known nodes for service discovery and publish-subscribe located at <https://xmpp.org/registrar/nodes.html>).

- Security certificates (including appropriate certificate chains) for Controllers, including identification of any Providers associated with the Controllers (which might be located at subdomains).

- Consistent and secure access control policies for publishing and subscribing to Topics.
These matters are out of scope for this document but ought to be addressed by the XMPP-Grid community.

11. 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. In addition, we want to thank Takeshi Takahashi, Panos Kampanakis, Adam Montville, Chris Inacio, and Dave Cridland for reviewing and providing valuable comments.

12. References

12.1. Normative References


12.2. Informative References


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: syaml@cisco.com
Scott Pope
Cisco Systems
5400 Meadows Road
Suite 300
Lake Oswego, OR  97035
USA

Email: scottp@cisco.com

Peter Saint-Andre
Mozilla

Email: stpeter@mozilla.com