MILE Working Group J. Field
Internet-Draft Pivotal
Intended status: Standards Track S. Banghart
Expires: November 25, 2017 D. Waltermire
NIST
May 24, 2017

Resource-Oriented Lightweight Information Exchange
draft-ietf-mile-rolie-07

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 will expire on November 25, 2017.

Copyright Notice

Table of Contents

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 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 programmtically 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.

- o 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 atom: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:

```
<?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="http://example.org/feedA?page=5"/>
    <link rel="first" href="http://example.org/feedA?page=1"/>
    <link rel="prev" href="http://example.org/feedA?page=4"/>
    <link rel="next" href="http://example.org/feedA?page=6"/>
    <link rel="last" href="http://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 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,
         (atomAuthor*
         & atomCategory*
         & atomContent
         & atomContributor*
         & atomId
         & atomLink*
         & atomPublished?
         & atomRights?
         & atomSource?
         & atomSummary?
         & atomTitle
         & atomUpdated
         & rolieFormat
         & rolieProperty*
         & extensionElement*)
     }
```

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 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.

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:

```
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:

```
<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:

```
<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.

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 specific 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:

Name	Extension IRI	Reference	Sub-Registry
category:information-type	urn:ietf:params:rolie:category:information-type	This document, Section 8.4	[TO BE REMOVED: This registration should take place at the following location: https://www.iana.org/assignments/rolie/category/information-type]
property:local	urn:ietf:params:rolie:property:local	This document, Section 7.4	None
category:local	urn:ietf:params:rolie:category:local	This document, Section 7.1	None
property:content-author-name	urn:ietf:params:rolie:property:content-author-name	This document, Section 7.4	None

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

[RFC0020] Cerf, V., "ASCII format for network interchange", STD 80, RFC 20, DOI 10.17487/RFC0020, October 1969, <http://www.rfc-editor.org/info/rfc20>.

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://www.rfc-editor.org/info/rfc3688>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC4287] Nottingham, M., Ed. and R. Sayre, Ed., "The Atom Syndication Format", RFC 4287, DOI 10.17487/RFC4287, December 2005, <http://www.rfc-editor.org/info/rfc4287>.

[RFC5005] Nottingham, M., "Feed Paging and Archiving", RFC 5005, DOI 10.17487/RFC5005, September 2007, <http://www.rfc-editor.org/info/rfc5005>.

[RFC5023] Gregorio, J., Ed. and B. de hOra, Ed., "The Atom Publishing Protocol", RFC 5023, DOI 10.17487/RFC5023, October 2007, <http://www.rfc-editor.org/info/rfc5023>.

[RFC7970] Danyliw, R., "The Incident Object Description Exchange Format Version 2", RFC 7970, DOI 10.17487/RFC7970, November 2016, <http://www.rfc-editor.org/info/rfc7970>.

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

[RFC3553] Mealling, M., Masinter, L., Hardie, T., and G. Klyne, "An IETF URN Sub-namespace for Registered Protocol Parameters", BCP 73, RFC 3553, DOI 10.17487/RFC3553, June 2003, <http://www.rfc-editor.org/info/rfc3553>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[W3C.REC-xml-names-20091208]
Bray, T., Hollander, D., Layman, A., Tobin, R., and H. Thompson, "Namespaces in XML 1.0 (Third Edition)", World Wide Web Consortium Recommendation REC-xml-names-20091208, December 2009, <http://www.w3.org/TR/2009/REC-xml-names-20091208>.

[RFC7589] Badra, M., Luchuk, A., and J. Schoenwaelder, "Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication", RFC 7589, DOI 10.17487/RFC7589, June 2015, <http://www.rfc-editor.org/info/rfc7589>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC4642] Murchison, K., Vinocur, J., and C. Newman, "Using Transport Layer Security (TLS) with Network News Transfer Protocol (NNTP)", RFC 4642, DOI 10.17487/RFC4642, October 2006, <http://www.rfc-editor.org/info/rfc4642>.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, DOI 10.17487/RFC5785, April 2010, <http://www.rfc-editor.org/info/rfc5785>.

[relax-NG] Clark, J., Ed., "RELAX NG Compact Syntax", 11 2002, <https://www.oasis-open.org/committees/relax-ng/compact-20021121.html>.

12.2. Informative References

[RFC2141] Moats, R., "URN Syntax", RFC 2141, DOI 10.17487/RFC2141, May 1997, <http://www.rfc-editor.org/info/rfc2141>.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <http://www.rfc-editor.org/info/rfc2782>.

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <http://www.rfc-editor.org/info/rfc3444>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[SAML-core] Cantor, S., Kemp, J., Philpott, R., and E. Maler, "Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard saml-core-2.0-os, March 2005, <http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>.

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

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

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

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

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

[3] http://csrc.nist.gov/groups/SNS/rbac/

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 "Vunerabilities". 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.

```
HTTP/1.1 200 OK
Date: Fri, 24 Aug 2016 17:10:11 GMT
Content-Length: 1912
Content-Type: application/atomsvc+xml;charset="utf-8"

<?xml version="1.0" encoding=’utf-8’?>
<service xmlns="http://www.w3.org/2007/app"
    xmlns:atom="http://www.w3.org/2005/Atom">
  <workspace>
    <atom:title>Public Security Information Sharing</atom:title>
    <collection
        href="http://example.org/provider/public/vulns">
      <atom:title>Public Vulnerabilities</atom:title>
      <atom:link rel="service"
        href="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
        href="http://example.org/provider/private/incidents">
      <atom:title>Incidents</atom:title>
      <atom:link rel="service"
        href="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>

  <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