Thing-to-Thing Research Group
Internet-Draft
Intended status: Experimental
Expires: November 10, 2020

K. Hartke
Ericsson

May 9, 2020

Thing-to-Thing Data Hub
draft-hartke-t2trg-data-hub-06

Abstract

A "Thing-to-Thing Data Hub" is a RESTful, hypermedia-driven Web application that can be used in Thing-to-Thing communications to share data items such as thing descriptions, configurations, resource descriptions, or firmware updates at a central location.

Table of Contents

1. Introduction

In Thing-to-Thing communication, there is often a need to share data items of common interest through a central location. For example, the Resource Directory [I-D.ietf-core-resource-directory] aggregates descriptions of Web resources held on constrained nodes, which enables other nodes to easily discover these resources; a Thing Directory [W3C.CR-wot-architecture-20190516] stores metadata of IoT devices, allowing clients to discover interaction affordances and supported protocol bindings of Things; a Firmware Server [I-D.ietf-suit-architecture] stores firmware images and manifests, making this data available to deployed devices, commissioning tools, and other services.

As more and more Thing-to-Thing applications are implemented, it becomes increasingly important being able to not only share resource descriptions and firmware updates but also many other kinds of data, such as default configurations for new devices, service locations, or certificate revocation lists. Resource directories and firmware servers are not a good fit for these kinds of data, as they're specialized to their use cases and generally not accepting any other

kinds of data. The creation of new, specialized applications for every type of data is not practical in the long term.

This document defines a simple "data hub" application, a RESTful Web application with a machine-understandable hypermedia API. A "data hub" generalizes the concept of a central repository for different applications and is suitable for constrained environments [RFC7228]. Specifically, it enables clients to share data items in any format and provides means for creating, reading, observing, updating, deleting, and finding data items at a data hub server.

Data hubs are primarily intended to be accessible over the Constrained Application Protocol (CoAP) [RFC7252].

Features:

- General

  The data hub generalizes the concept of a directory or repository to data items of any Internet media type. This means that applications using the data hub aren't stuck forever with the same media types or limited to just resource descriptions or firmware updates.

- Searchable

  Clients can retrieve a subset of data items from a data hub based on item metadata.

- Observable

  Data items published to a data hub are exposed as resources. As such, they can be observed for changes [RFC7641] over CoAP. This allows clients to stay informed of information that other clients update over time. As a result, the data hub functions similar to a Publish-Subscribe Broker [I-D.ietf-core-coap-pubsub].

- Evolvable

  The key differentiator of the data hub compared to Resource Directory [I-D.ietf-core-resource-directory] and CoAP Publish-Subscribe Broker [I-D.ietf-core-coap-pubsub] lies in the evolvability of the application -- the ability to respond effectively to the need for changes without negatively impacting existing and new clients.

  Data hubs enable fine-grained evolvability by driving all interactions by machine-understandable hypermedia elements.

Features can be added, changed or removed in a safe, backwards-compatible way simply by updating the data hub representation to expose appropriate links and forms.

1.1. Notational Conventions

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 [RFC2119].

2. Data Hubs

The "Thing-to-Thing Data Hub" application consists of two types of resources: a "data collection" and a number of "data items" that have been shared (Figure 1).

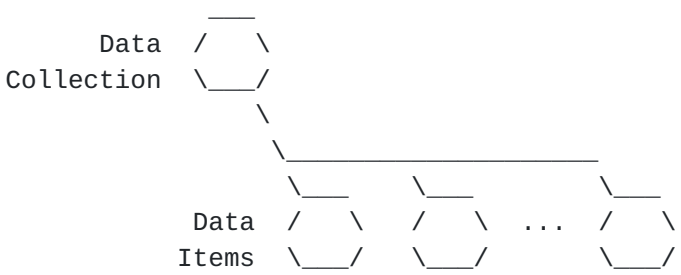

Figure 1: Resources of a Data Hub

2.1. Data Items

A data item is a resource that is a member of the data collection resource.

2.1.1. Data Item Representation

The representation of a data item can be of any media type. However, a data hub can restrict the media types it accepts for publication (see Section 2.2.1).

2.2. Data Collections

A data collection is a collection resource that contains data item resources.

Design Note: In this version of this document, a data hub has only a depth of one level; i.e., all data item resources are organized directly under the top-level data collection resource. This could be extended to multiple levels in a future version.

2.2.1. Data Collection Representation

The representation of a data collection is a CoRAL document [I-D.ietf-core-coral] containing the following elements:

- A form of type <http://coreapps.org/collections#create> containing the following fields:
  - For each content format accepted for publication, a form field of type <http://coreapps.org/coap#accept> indicating that content format. The absence of any form fields of this type indicates that any content format is accepted.
- A form of type <http://coreapps.org/base#search>.
- For each data item in the data collection, a link of type <http://www.iana.org/assignments/relation/item> [RFC6573] targeting the data item and containing the following elements:
  - A form of type <http://coreapps.org/base#update>.
  - A form of type <http://coreapps.org/collections#delete>.
  - Optionally, a (complete or partial) embedded representation of the data item.

The document MAY additionally contain other links and forms not described in this document. For example, a document could contain a link with the <http://www.iana.org/assignments/relation/alternate> link relation type [W3C.REC-html52-20171214] that references an alternate representation of a data item.

Any of the links and forms MUST be omitted if following or submitting it can never lead to a successful outcome, for example, because the client is not authorized or the server does not support the feature.

2.2.2. Filter Query Representation

TODO.

2.3. Data Hub Discovery

In this version of this document, clients are assumed to be pre-configured with the URI of a data collection at a data hub.

2.4. Interactions

2.4.1. Getting All Data Items

A client can list all data items in a data collection by making a GET request to the data collection URI (e.g., after discovering the data hub as described in Section 2.3).

On success, the server returns a 2.05 (Content) response with a representation of the collection. As specified in Section 2.2.1 above, this representation includes links to (and, optionally, representations of) the data items in the data collection as well as forms for creating, updating, deleting, and finding data items.

2.4.2. Getting Data Items by Metadata

If the representation of a data collection contains a form of type <http://coreapps.org/base#search>, the client can filter the data collection by submitting this form with a search query (see Section 2.2.2).

Implementations of this version of this document MUST use the method implied by the <http://coreapps.org/base#search> operation type, i.e., the FETCH method [RFC8132]. Any form indicating a different method MUST be ignored.

On success, the server returns a 2.05 (Content) response with a representation of a list of data items in the collection (see Section 2.2.1) that match the query.

2.4.3. Creating a Data Item

If the representation of a data collection contains a form of type <http://coreapps.org/collections#create>, the client can create a new data item in the data collection by submitting this form with a representation in one of the acceptable media types. The acceptable media types are indicated by form fields of type <http://coreapps.org/coap#accept>.

Implementations of this version of this document MUST use the method implied by the <http://coreapps.org/collections#create> operation type, i.e., the POST method [RFC7252]. Any form indicating a different method MUST be ignored.

On success, the server returns a 2.01 (Created) response. The location of the created data item is conveyed using the Location-Path and Location-Query options [RFC7252].

2.4.4. Reading a Data Item

A client can read a data item by following a link with the <http://www.iana.org/assignments/relation/item> link relation type in the representation of the data collection.

2.4.5. Observing a Data Item

A client can observe a data item by following a link with the <http://www.iana.org/assignments/relation/item> link relation type in the representation of the data collection and observing the target resource as specified in RFC 7641 [RFC7641].

2.4.6. Updating a Data Item

If the representation of a data collection includes a form of type <http://coreapps.org/base#update> nested within the link to a data item, a client can update the data item by submitting this form with a representation of the updated data item.

Implementations of this version of this document MUST use the method implied by the <http://coreapps.org/base#update> operation type, i.e., the PUT method [RFC7252]. Any form indicating a different method MUST be ignored.

On success, the server returns a 2.04 (Changed) response.

2.4.7. Deleting a Data Item

If the representation of a data collection includes a form of type <http://coreapps.org/collections#delete> nested within the link to a data item, the client can delete the data item by submitting this form.

Implementations of this version of this document MUST use the method implied by the <http://coreapps.org/collections#delete> operation type, i.e., the DELETE method [RFC7252]. Any form indicating a different method MUST be ignored.

On success, the server returns a 2.02 (Deleted) response.

3. Security Considerations

The data hub application relies on a Web transfer protocol like CoAP to exchange representations in a CoRAL serialization format. See Section 11 of RFC 7252 [RFC7252] and Section 7 of RFC 7641 [RFC7641] for security considerations relating to CoAP. See Section XX of RFC

XXXX [I-D.ietf-core-coral] for security considerations relating to CoRAL.

The data hub application does not define any specific mechanisms for protecting the confidentiality and integrity of messages exchanged between a data hub and a client. It is recommended that implementations employ application layer or transport layer mechanisms for interactions with a data hub.

The data hub application does not define any specific mechanisms for protecting the confidentiality and integrity of representations of data items shared through a data hub. For scenarios where end-to-end security matters, such as for firmware updates [I-D.ietf-suit-information-model], implementations should employ an object security mechanism.

4. IANA Considerations

This document has no IANA actions.

5. References

5.1. Normative References

[I-D.ietf-core-coral]
Hartke, K., "The Constrained RESTful Application Language (CoRAL)", draft-ietf-core-coral-03 (work in progress), March 2020.

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

[RFC6573] Amundsen, M., "The Item and Collection Link Relations", RFC 6573, DOI 10.17487/RFC6573, April 2012, <https://www.rfc-editor.org/info/rfc6573>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7641] Hartke, K., "Observing Resources in the Constrained Application Protocol (CoAP)", RFC 7641, DOI 10.17487/RFC7641, September 2015, <https://www.rfc-editor.org/info/rfc7641>.

[RFC8132] van der Stok, P., Bormann, C., and A. Sehgal, "PATCH and FETCH Methods for the Constrained Application Protocol (CoAP)", RFC 8132, DOI 10.17487/RFC8132, April 2017, <https://www.rfc-editor.org/info/rfc8132>.

5.2. Informative References

[I-D.ietf-core-coap-pubsub]
Koster, M., Keranen, A., and J. Jimenez, "Publish-Subscribe Broker for the Constrained Application Protocol (CoAP)", draft-ietf-core-coap-pubsub-09 (work in progress), September 2019.

[I-D.ietf-core-resource-directory]
Shelby, Z., Koster, M., Bormann, C., Stok, P., and C. Amsuess, "CoRE Resource Directory", draft-ietf-core-resource-directory-24 (work in progress), March 2020.

[I-D.ietf-suit-architecture]
Moran, B., Tschofenig, H., Brown, D., and M. Meriac, "A Firmware Update Architecture for Internet of Things", draft-ietf-suit-architecture-08 (work in progress), November 2019.

[I-D.ietf-suit-information-model]
Moran, B., Tschofenig, H., and H. Birkholz, "An Information Model for Firmware Updates in IoT Devices", draft-ietf-suit-information-model-05 (work in progress), January 2020.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[W3C.CR-wot-architecture-20190516]
Kovatsch, M., Matsukura, R., Lagally, M., Kawaguchi, T., Toumura, K., and K. Kajimoto, "Web of Things (WoT) Architecture", World Wide Web Consortium Candidate Recommendation CR-wot-architecture-20190516, May 2019, <https://www.w3.org/TR/2019/CR-wot-architecture-20190516>.

[W3C.REC-html52-20171214]
Faulkner, S., Eicholz, A., Leithead, T., Danilo, A., and S. Moon, "HTML 5.2", World Wide Web Consortium Recommendation REC-html52-20171214, December 2017, <https://www.w3.org/TR/2017/REC-html52-20171214>.

Acknowledgements

Thanks to Christian Amsuess and Jaime Jimenez for helpful comments and discussions that have shaped the document.

Author's Address

Klaus Hartke
Ericsson
Torshamnsgatan 23
Stockholm SE-16483
Sweden

Email: klaus.hartke@ericsson.com