Network Working Group
Internet-Draft
Expires: March 12, 2004

A. Barbir
Nortel Networks

September 12, 2003

OPES processor and end points communications
draft-ietf-opes-end-comm-01

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

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

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

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

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

This Internet-Draft will expire on March 12, 2004.

Copyright Notice

Abstract

This memo documents tracing requirements for Open Pluggable Edge Services (OPES).

Table of Contents

1. Introduction 3
2. OPES Domain and OPES System 4
3. OPES Tracing 6
3.1 What is traceable in an OPES Flow? 6
3.2 Requirements for Information Related to Traceable Entities? 7
4. Requirements for OPES processors 8
5. Requirements for callout servers 9
6. Privacy considerations 10
6.1 Tracing and Trust Domains 10
7. How to Support Tracing 11
7.1 Tracing and OPES System Granularity 11
7.2 Requirements for In-Band Tracing 12
7.2.1 Tracing Information Granularity and Persistence levels Requirements 12
7.3 Protocol Binding 13
7.4 Tracing scenarios and examples 13
8. Optional Notification 14
9. IANA considerations 16
10. Security Considerations 17
Normative References 18
Informative References 19
Author's Address 19
A. Acknowledgements 20
Intellectual Property and Copyright Statements 21

1. Introduction

The Open Pluggable Edge Services (OPES) architecture [8] enables cooperative application services (OPES services) between a data provider, a data consumer, and zero or more OPES processors. The application services under consideration analyze and possibly transform application-level messages exchanged between the data provider and the data consumer.

The execution of such services is governed by a set of rules installed on the OPES processor. The rules enforcement can trigger the execution of service applications local to the OPES processor. Alternatively, the OPES processor can distribute the responsibility of service execution by communicating and collaborating with one or more remote callout servers. As described in [8], an OPES processor communicates with and invokes services on a callout server by using a callout protocol.

The work specify the requirements for providing tracing functionality for the OPES architecture [8]. This document specifies tracing mechanisms that the OPES architecture could provide that enable data provider application to detect inappropriate clinet centric actions by OPES entities. The work focus on developing tracing requirements that can be used to fulfil the notification and Non-Blocking requirements [2].

In the OPES architecture document [8], there is a requirement of relaying tracing information in-band. This work investigates this possibility and discusses possible methods that could be used to detect faulty OPES processors or callout servers by end points in an OPES flow.

The document is organized as follows: Section 2 defines OPES Domain and OPES System. Section 3 discusses entities that are traceable in an OPES Flow. Sections 4 and 5 discuss tracing requirements for OPES systems and callout servers. Section 6 focus on Tracing and Trust Domains. Section 7 discusses how to support tracing and provides uses cases. Section 8 examines Optional Notofication. Section 9 looks into IANA considerations. Section 10 examines security considerations.

2. OPES Domain and OPES System

This sections clarifies the terms OPES system and OPES Domain [8]. These terms are needed in order to define what is traceable in an OPES Flow [8].

An OPES domain describes the collection of OPES entities that a single provider operates. OPES domains can be based on trust or other operational boundaries. All elements of an "OPES Domain" MUST be in the same trust domain. This would be independent of any specific OPES flow.

An OPES system consists of a limited set of OPES entities, parts of a single or of multiple OPES operators domains, organized by (or on behalf) of either a data provider application or a data consumer application to perform authorized services on a given application message. Each OPES entity in an OPES system MUST be directly addressable on IP level by a data consumer application.

An OPES system can be formed in a recursive manner. An OPES system can start with either a data provider application or a data consumer application (for a given message). The OPES system then includes any OPES entity trusted by (accepting authority from) an entity that is already in the OPES system. The trust and authority delegation is viewed in the context of the given application message.

As implied by the above definition, some OPES entities in the system may not participate in the processing of a given message.

An OPES domain MUST not be an OPES sub-system. An OPES domain MUST require external resources to provide services. An OPES domain is a part of an OPES system belonging to a given operator. OPES domains have no incidence on the structure of an OPES system, but they may influence its organization for different reasons such as security, payment, quality of service, delivery parameters among others.

In Figure 1 an OPES Flow is shown that traverses across various OPES Domains. A data consumer application MUST be able to recive tracing information on per message basis that enable it to determine the set of transformations that were perfomed on the data for a particular OPES Flow. The formation of an OPES flow can be static or dynamic, meaning that the determination of which OPES Domains will participate in a given OPES Flow (per message basis) can be a function of business arrangements.

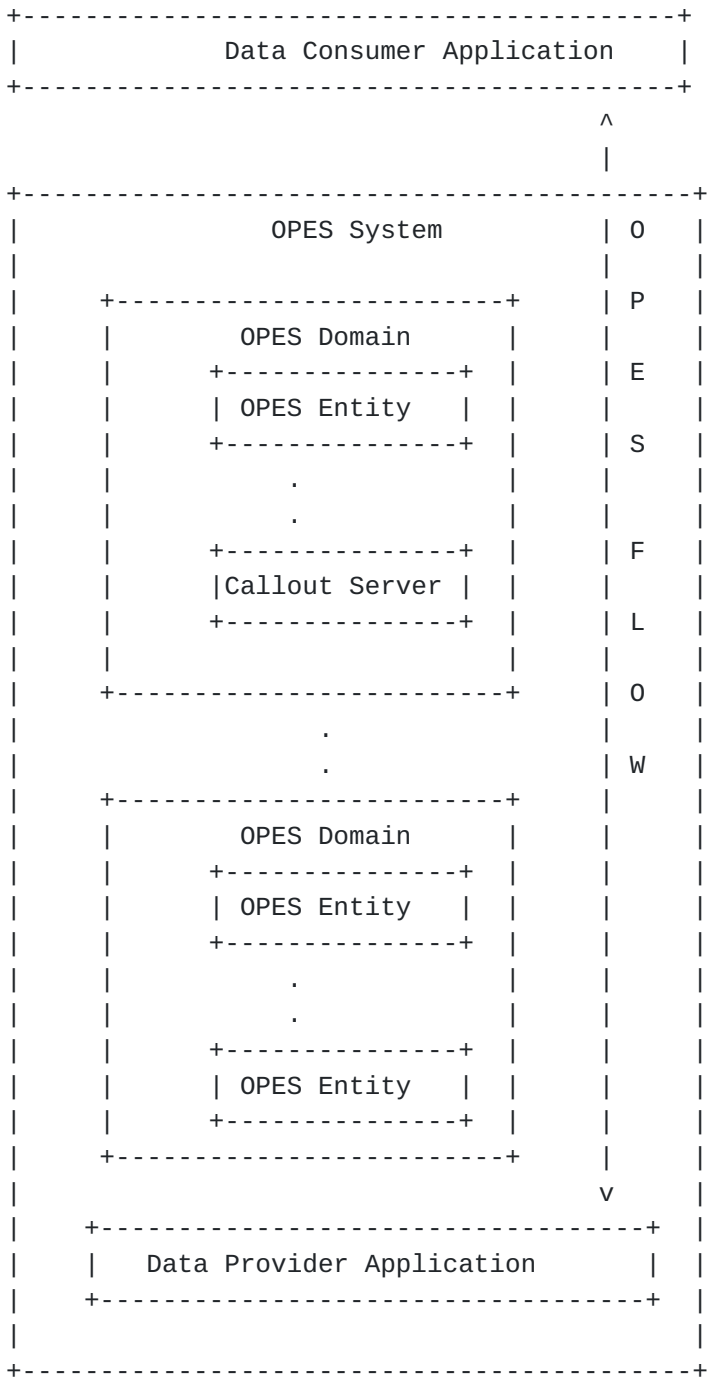

Figure 1: OPES System

3. OPES Tracing

Before discussing what is traceable in an OPES flow, it is beneficial to define what tracing means. Tracing is defined as the inclusion of necessary information within a message in an OPES flow that could be used to identify the set of transformations or adpatations that have been performed on its content in an OPES System before its delivery to an end point (the data consumer application).

- OPES trace: application message information about OPES entities in an OPES System that adapted that message.

- OPES tracing: the process of including, manipulating, and interpreting an OPES trace in an OPES System.

To emphasize, the above definition means that OPES tracing SHOULD be performed on per message basis. Trace format is dependent on the application protocol that is being adapted by OPES. Data consumer application can use OPES trace to infer the actions that have been performed by the OPES system. The architecture document requires [8] that tracing be supported in-band.

In an OPES System the task of providing tracing information, must take into account the following considerations:

- Providers may be hesitant to reveal information about their internal network infrastructure.

- Within a service provider network, OPES processors may be configured to use non-routable, private IP addresses.

- A Data consumer applications would prefer to have a single point of contact regarding the trace information.

- TBD

3.1 What is traceable in an OPES Flow?

This section focuses on identifying the traceable entities in an OPES Flow. Tracing information MUST be able to provide a data consumer application with useful information without tracing the exact OPES Processor or callout servers that adapted the data. It is up to the OPES service provider to have maintained appropriate internal detailed traces to find the answer to the data consumer applications inquiry.

At the implementation level, for a given trace, an OPES entity

involved in handling the corresponding application message is "traceable" or "traced" if information about it appears in that trace. OPES entities have different levels of traceability requirements. Specifically,

- An OPES system MUST add its entry to the trace.
- An OPES processor SHOULD add its entry to the trace.
- An OPES service SHOULD add its entry to the trace.
- An OPES entity MAY manage trace information from entities that are under its control. For example, an OPES processor may add or remove callout service entries in order to manage the size of a trace. Other considerations include:
  - The OPES processor may have a fixed configuration that enable it to respond to tracing inquires.
  - The OPES processor may insert a summary of the services that it controls. The summary can be used to respond to tracing inquiries.
  - The OPES processor may package tracing information related to the entities that it control based on the policy of a given OPES System.

From an OPES context, a good tracing approach is similar to a trouble ticket ready for submission to a known address. The trace in itself is not necessarily a detailed description of what has happened. It is the resposibility of the operator to resolve the problems.

3.2 Requirements for Information Related to Traceable Entities?

The requirements for information as related to entities that are terceable in an OPES flow are:

- The privacy policy at the time it dealt with the message
- Identification of the party responsible for setting and enforcing that policy
- Information pointing to a technical contact
- Information that identifies, to the technical contact, the OPES processors involved in processing the messag
- TBD

4. Requirements for OPES processors

In order to facilitate compliance, the concept of an "OPES system" being traceable, requires that each OPES processor MUST support tracing. Policy can be set that defines which domain has authorization to turn on tracing and its granularity. An OPES provider can have its private policy for trace information, but it MUST support tracing mechanisms and it MUST reveal it's policy.

The requirements for OPES processors that are applicatble to tracing are:

- Each OPES processor MUST belong to a single OPES Domain.
- Each OPES processor MUST have a Unique Identity in that Domain.
- Each OPES processor MUST support tracing, policy can be used to turn tracing on and.to determine granuality.
- TBD

5. Requirements for callout servers

If it is the task of an OPES processor to add trace records to application messages, then callout servers that uses the OCP protocol are not affected by tracing requirements. In order for an OCP protocol to be tracing neutral, the OPES server SHOULD be able to meet the following requirements:

- Callout services adapt payload regardless of the application protocol in use and leave header adjustment to OPES processor.
- OPES processor SHOULD be able to trace it's own invocation and service(s) execution since they understand the application protocol.
- Callout servers MAY be able to add their own OPES trace records to application level messages.
- TBD

6. Privacy considerations

6.1 Tracing and Trust Domains

A trust domain may include several OPES systems and entities. Within a trust domain, there MUST be at least support for one trace entry per system. Entities outside of that system may or may not see any traces, depending on domain policies or configuration. For example, if an OPES system is on the content provider "side", end-users are not guaranteed any traces. If an OPES system is working inside end-user domain, the origin server is not guaranteed any traces related to user requests.

7. How to Support Tracing

In order to support tracing, the following aspects must be addressed:

- There MUST be a System Identifier that identify a domain that is employing an OPES system.
- An OPES processor MUST be able to be uniquely identified (MUST have an Identifier) within a system.
- An OPES processor MUST add its identification to the trace.
- An OPES processor SHOULD add to the trace identification of every callout service that received the application message.
- An OPES processor MUST add to the trace identification of the "system/entity" it belongs to. "System" ID MUST make it possible to access "system" privacy policy.
- An OPES processor MAY group the above information for sequential trace entries having the same "system/entity" ID. In other words, trace entries produced within the same "system/entity" MAY be merged/aggregated into a single less detailed trace entry.
- An OPES processor MAY delegate trace management to a callout service within the same "system/entity".

TBD

7.1 Tracing and OPES System Granularity

There are two distinct uses of traces. First, is to SHOULD enable the "end (content producer or consumer) to detect OPES processor presence within end's trust domain. Such "end" should be able to see a trace entry, but does not need to be able to interpret it beyond identification of the trust domain(s).

Second, the domain administrator SHOULD be able to take a trace entry (possibly supplied by an "end? as an opaque string) and interpret it. The administrator must be able to identify OPES processor(s) involved and may be able to identify applied adaptation services along with other message-specific information. That information SHOULD help to explain what OPES agent(s) were involved and what they did. It may be impractical to provide all the required information in all cases. This document view a trace record as a hint, as opposed to an exhaustive audit.

Since the administrators of various trust domains can have various

ways of looking into tracing, they MAY require the choice of freedom in what to put in trace records and how to format them. Trace records should be easy to extend beyond basic OPES requirements. Trace management algorithms should treat trace records as opaque data to the extent possible.

It is not expected that entities in one trust domain to be able to get all OPES-related feedback from entities in other trust domains. For example, if an end-user suspects that a served is corrupted by a callout service, there is no guarantee that the use will be able to identify that service, contact its owner, or debug it _unless_ the service is within my trust domain. This is no different from the current situation where it is impossible, in general, to know the contact person for an application on an origin server that generates corrupted HTML; and even if the person is known, one should not expect that person to respond to end-user queries.

7.2 Requirements for In-Band Tracing

The OPES architecture [8] states that traces must be in-band. The support of this design specification is dependent on the specifics of the message application level protocol that is being used in an OPES flow. In-band tracing limits the type of application protocols that OPES can support. The details of what a trace record can convey is also dependent on the choice of the application level protocol.

For these reasons, the work will document requirements for application protocols that need to support OPES traces. However, the architecture does not prevent implementers of developing out-of-band protocols and techniques to address the above limitation.

7.2.1 Tracing Information Granularity and Persistence levels Requirements

In order to be able to trace entities that have acted on an application message in an OPES flow, there may be requirements to keep information that is related to the following:

- Message-related informatio: All data that describes specific actions performed on the message SHOULD be provided with that message, as there is no other way to find message level details later.

- Session related information: Session level data MUST be preserved for the duration of the session. OPES processor is responsible for inserting notifications if session-level information changes.

- End-point related data: What profile is activated? Where to get

profile details? Where to set preferences?

- TBD

7.3 Protocol Binding

How tracing is added is application protocol-specific and will be documented in separate drafts. This work documents what tracing information is required and some common tracing elements.

7.4 Tracing scenarios and examples

TBD

8. Optional Notification

This section examines IAB [2] considerations (3.1) and (3.2) regarding notification in an OPES architecture.

Notification propagates in opposite direction of tracing and cannot be attached to application messages that it notifies about. Notification can be done out-band and may require the development of a new protocol. The direction of data flow for tracing and notification are depicted in Figure 2.

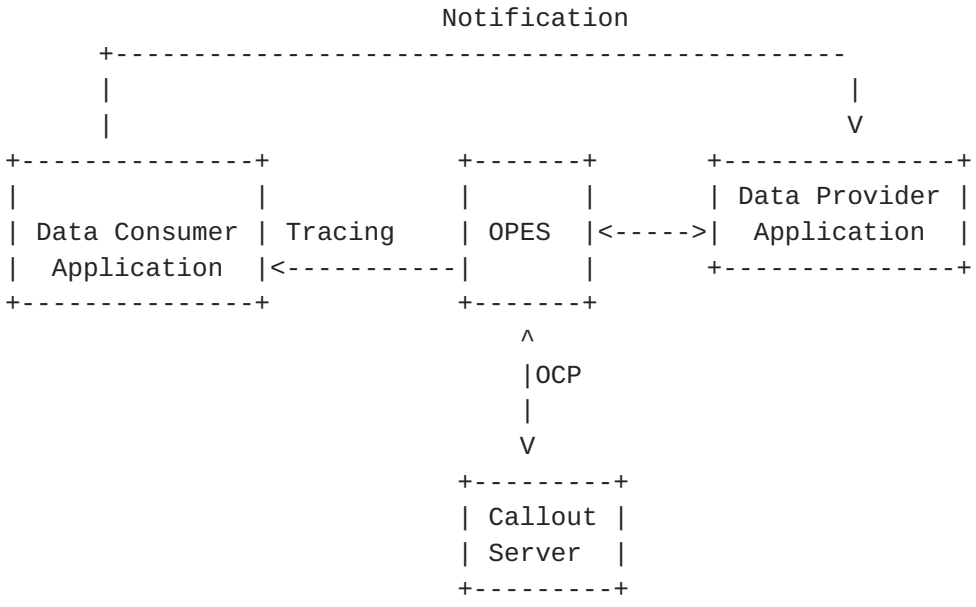

Figure 2: Notification Flow

In [9] it was argued that Notification is an expensive approach for providing tracing information. However, the current work does not prevent an OPES System from publishing policy and specifications that allow Optional Notification. For example, an OPES System can adopt a mechanism that uses a flag that would allow a data consumer and a data provider application to signal to each other that they are interested to receive an explicit notification if an OPES service is applied to a specific message. The value of this optional flag/field can be a URI that identifies notification method plus parameters. If a processor understands the method, it would be able to further decode the field and send a notification. The specification of the field name and format for an application protocol can be stated in the associated binding document. The details of the notification

protocol is beyond the scope of this Working Group.

For example, the following HTTP header:

- OPES-Notify: URI *(pname=pvalue)

Or,

- My-OPES-Notify: foo=bar q=0.5

can be used.

9. IANA considerations

TBD

10. Security Considerations

TBD

Normative References

[1] McHenry, S., et. al, "OPES Scenarios and Use Cases", Internet-Draft TBD, May 2002.

[2] Floyd, S. and L. Daigle, "IAB Architectural and Policy Considerations for Open Pluggable Edge Services", RFC 3238, January 2002.

[3] Fielding, R., Gettys, J., Mogul, J., Nielsen, H., Masinter, L., Leach, P. and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[4] OPES working group, "OPES Service Authorization and Enforcement Requirements", Internet-Draft TBD, May 2002.

[5] OPES working group, "OPES Ruleset Schema", Internet-Draft TBD, May 2002.

[6] A. Beck et al., "Requirements for OPES Callout Protocols", Internet-Draft http://www.ietf.org/internet-drafts/draft-ietf-opes-protocol-reqs-03.txt, December 2002.

[7] A. Barbir et al., "Security Threats and Risks for Open Pluggable Edge Services", Internet-Draft http://www.ietf.org/internet-drafts/draft-ietf-opes-threats-00.txt, October 2002.

[8] A. Barbir et al., "An Architecture for Open Pluggable Edge Services (OPES)", Internet-Draft http://www.ietf.org/internet-drafts/draft-ietf-opes-architecture-04, December 2002.

[9] A. Barbir et al., "OPES Treatment of IAB Considerations", Internet-Draft http://www.ietf.org/internet-drafts/draft-ietf-opes-iab-01.txt, February 2004.

Informative References

[10] Westerinen, A., Schnizlein, J., Strassner, J., Scherling, M., Quinn, B., Herzog, S., Huynh, A., Carlson, M., Perry, J. and S. Waldbusser, "Terminology for Policy-Based Management", RFC 3198, November 2001.

[11] L. Cranor, et. al, "The Platform for Privacy Preferences 1.0 (P3P1.0) Specification", W3C Recommendation 16 http://www.w3.org/TR/2002/REC-P3P-20020416/ , April 2002.

[12] "Hit Metering", RFC .

Author's Address

Abbie Barbir
Nortel Networks
3500 Carling Avenue
Nepean, Ontario K2H 8E9
Canada

Phone: +1 613 763 5229
EMail: abbieb@nortelnetworks.com

Appendix A. Acknowledgements

TBD

Intellectual Property Statement

Full Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.