Network Working Group
Internet-Draft
Expires: November 1, 2002

Abbie. Barbir
Nortel Networks
R. Chen
AT&T Labs
M. Hofmann
Bell Labs/Lucent Technologies
H. Orman
The Purple Streak Development
R. Penno
Nortel Networks
G. Tomlinson
Cacheflow
May 3, 2002

An Architecture for Open Pluggable Edge Services (OPES)
draft-ietf-opes-architecture-00

Status of this Memo

Copyright Notice

Abstract

This memo defines an architecture for a cooperative application service in which a data provider, a data consumer, and zero or more

application entities cooperatively realize a data stream service.

Table of Contents

1. Introduction

When realizing a data stream service between a provider and a consumer, the need may arise to provision the use of other application entities, in addition to the provider and consumer. For example, some party may wish to customize a data stream as a service to a consumer, e.g., a service might customize the data based on the customer's geographical locality (e.g., language) or resource availability (e.g., display capabilities).

In some cases it may be impossible to offer the customization service at either the provider or the consumer applications. In this case, one or more additional application entities may participate in the data stream service. There are many possible provisioning scenarios which make a data stream service attractive. The reader is referred to [1] for a description of several scenarios.

The document presents the architectural components of Open Pluggable Edge Services (OPES) that are needed in order to perform a data stream service. The architecture addresses the IAB considerations described in [2]. These considerations are covered in the various parts of the document, specifically with respect to tracing (Section 2.6) and security considerations (Section 3).

The document is organized as follows: Section 2 introduces the OPES architecture. Section 3 discusses security considerations. Section 4 provides a summary of the architecture and the requirements for interoperability.

2. The Architecture

The architecture of Open Pluggable Edge Services (OPES) can be described in terms of three interrelated concepts, mainly:

- OPES entities: processes operating in the network;
- OPES flows: data flows that are cooperatively realized by the OPES entities; and,
- OPES rules: these determine how a given data flow is modified by an OPES entity.

2.1 OPES Entities

An OPES entity is an application that operates on a data flow between a data provider application and a data consumer application. OPES entities can be one of the following:

- an OPES service application, which analyzes and possibly transforms messages exchanged between the data provider application and the data consumer application; or,
- a data dispatcher, which invokes an OPES service application based on filtering rules and application-specific knowledge.

In the network, OPES entities reside inside OPES processors. The cooperative behavior of OPES entities introduces additional functionality for each data flow provided that it matches the OPES rules.

The OPES architecture is largely independent of the protocol that is used by the OPES entities to exchange data. However, this document selects HTTP [4] as the example protocol to be used for realizing a data flow. In this regard, the "protocol" stack of an OPES entity is summarized in Figure 1.

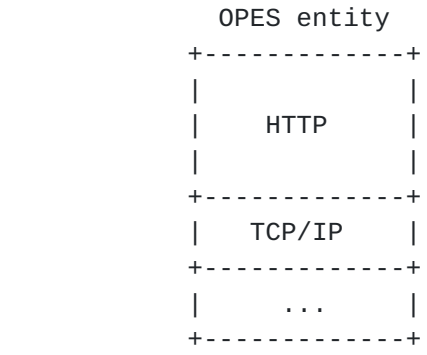

Figure 1: An OPES protocol stack

Figure 1 depicts a "minimal" stack for OPES. However, other protocols may be present, depending on the functions that are performed by the application.

2.2 OPES Flows

An OPES flow is a cooperative undertaking between a data provider application, a data consumer application, one or more OPES service applications, and one or more data dispatchers.

In order to understand the trust relationships between OPES entities, each is labeled as residing in an administrative domain. However, depending on provisioning decisions, the entities associated with a given OPES flow may reside in one or more administrative domains.

For example, Figure 2 depicts a data flow (also known as an "OPES flow"), that spans two administrative domains.

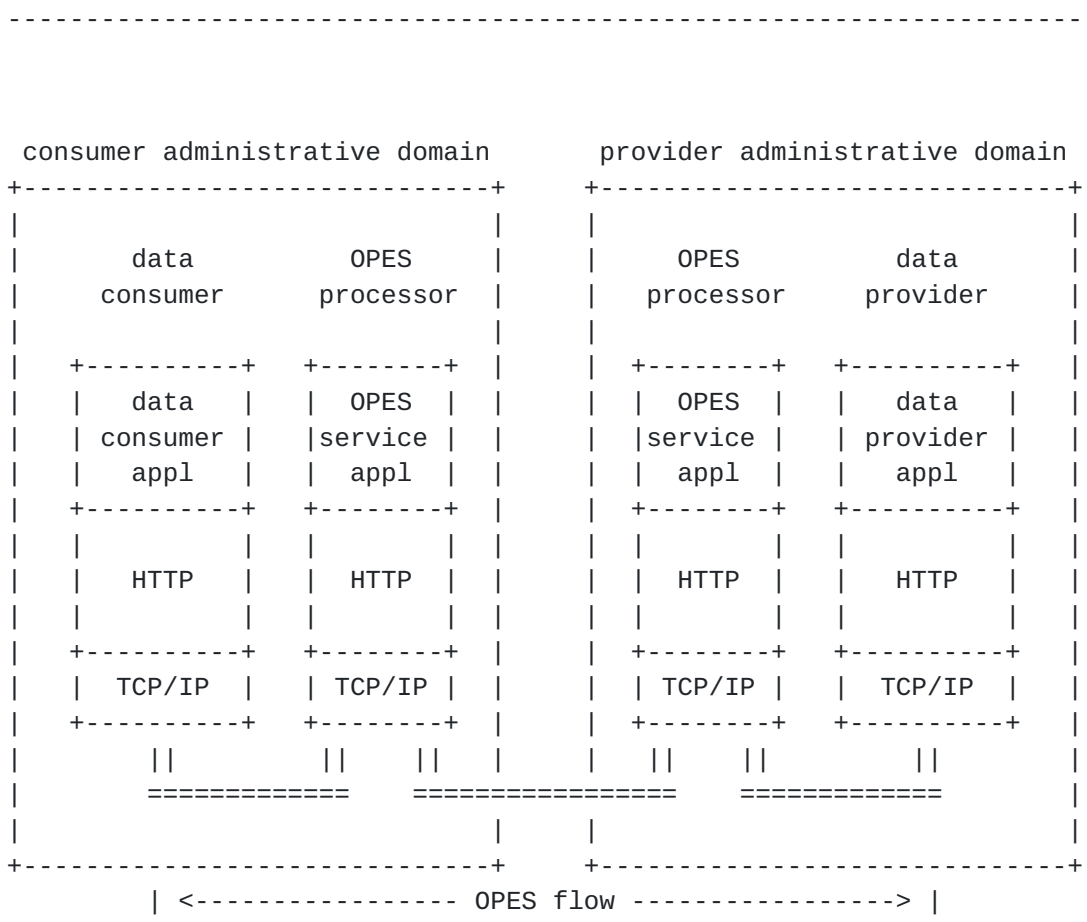

Figure 2: An OPES flow

Figure 2 depicts two data dispatchers that are present in the OPES flow. However, the architecture allows for zero or more data dispatchers to be present in any flow.

2.3 OPES Rules

The behavior of data dispatchers is governed by a set of filtering rules, consisting of a set of conditions and related actions. The ruleset is the superset of all OPES rules on the processor. Data dispatchers invoke OPES service applications that may perform modifications on an OPES flow. In this model, all data filters are invoked for all data.

In order to ensure predictable behavior the OPES architecture requires the use of a standardized schema for the purpose of defining and interpreting the ruleset. The OPES architecture does not require

a mechanism for configuring a ruleset into a data dispatcher. This is treated as a local matter for each implementation (e.g., through the use of a text editor, secure upload protocol, and so on). Future revisions of the architecture may introduce such a requirement.

2.4 Callout Servers

The OPES ruleset is executed within a data dispatcher, which triggers the execution of local OPES service applications. How the ruleset is executed is not the subject of the architecture. However, in some cases it may be useful for the OPES processor to distribute the responsibility of service execution by communicating with one or more callout servers (cf., [7]). The situation is illustrated in Figure 3, which shows a data dispatcher communicating with multiple callout servers as it processes an OPES flow.

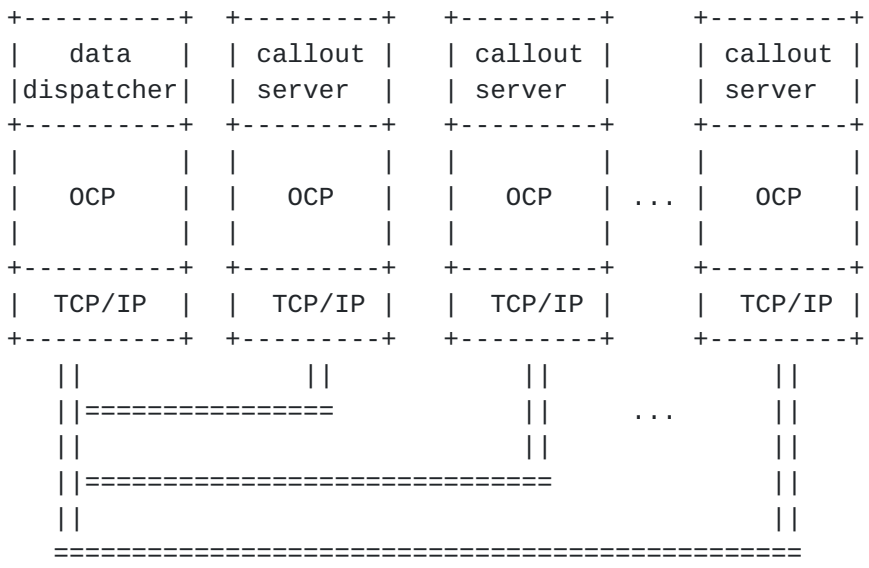

Figure 3: An OPES flow with Callout servers

In Figure 3, a data dispatcher invokes the services of a callout server by using the OPES callout protocol (OCP). The requirements for the OCP are given in [7]. The OCP is application-agnostic, being unaware of the semantics of the encapsulated application protocol (e.g., HTTP). However, the OCP must incorporate a service aware vectoring capability that parses the data flow according to the ruleset and delivers the data to the OPES service application that

can be local or remote.

2.5 Policy Enforcement

Data dispatchers include a ruleset that can be compiled from several sources and must resolve into an unambiguous result. The compiled ruleset enables an OPES processor to detremine which service applications to invoke for which data flow. Accordingly, the data dispatcher consitutes an enhanced Policy Enforcement Point (PEP), where policy rules are executed, data vectoring and connection management is performed, and service-specific data handlers and state information are maintained, as depicted in Figure 4.

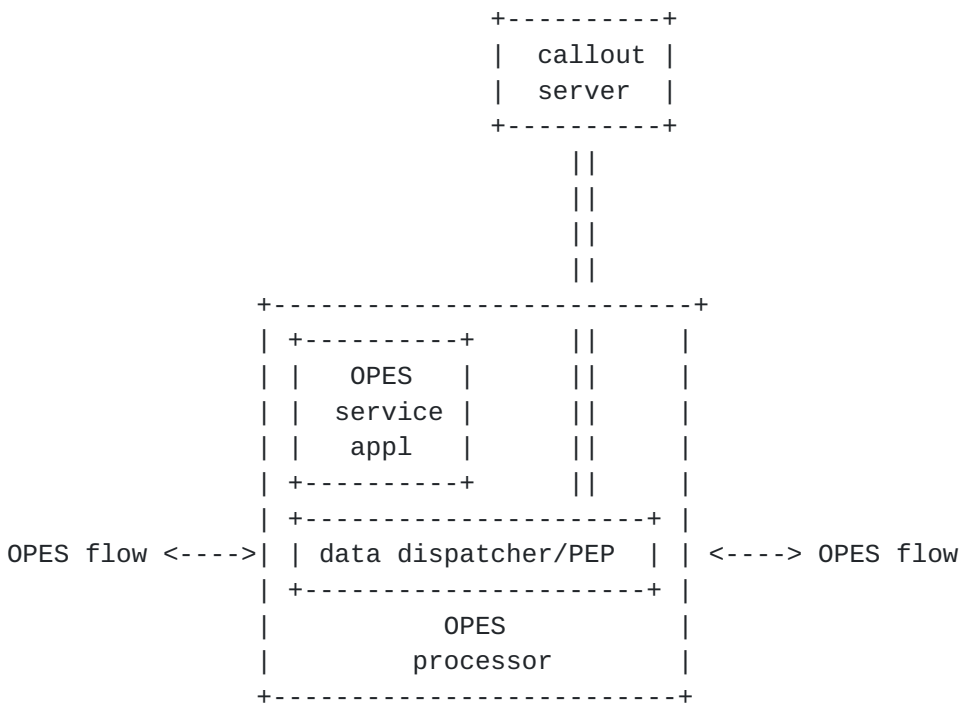

Figure 4: Data Dispatchers and Policy Enforcement Point

The architecture allows more than one PEP to be present on an OPES flow.

2.6 Tracing Facility

The architecture makes no requirements as to how an OPES flow is negotiated, provided that it is consistent with the security policy

(Section 3) of each administrative domain that hosts the OPES entities that are associated with the flow.

The OPES architecture requires that each data dispatcher provides tracing facilities that allow the appropriate verification of its operation. The OPES architecture requires that tracing be feasible on the OPES flow per OPES processor using in-band annotation through header extensions on the application protocol that is used (e.g., HTTP). One of those annotations could be a URL with more detailed information on the transformation that occurred to the data on the OPES flow. Future revisions of the architecture may provide for a tracing facility to be used for subsequent out-of-band analysis.

3. Security and Privacy Considerations

Each data flow must be secured in accordance with several policies. The primary stakeholders are the data consumer and the data provider. The secondary stakeholders are the entities to which they may have delegated their trust. The other stakeholders are the owners of the callout servers. Any of these parties may be participants in the OPES architecture.

These parties must have a model, explicit or implicit, describing their trust policy; which of the other parties are trusted to operate on data, and what security enhancements are required for communication. The trust might be delegated for all data, or it might be restricted to granularity as small as an application data unit.

All parties that are involved in enforcing policies must communicate the policies to the parties that are involved. These parties are trusted to adhere to the communicated policies.

In order to delegate fine-grained trust, the parties must convey policy information by implicit contract, by a setup protocol, by a dynamic negotiation protocol, or in-line with application data headers.

3.1 Trust Domains

The delegation of authority starts at either a data consumer or data provider and moves to more distant entities in a "stepwise" fashion. Stepwise means A delegates to B and B delegates to C and so force. The entities thus "colored" by the delegation are said to form a trust domain with respect to the original delegating party. Here, "Colored" means that if the first step in the chain is the data provider, then the stepwise delegation "colors" the chain with that data "provider" color. The only colors that are defined are data the "provider" and the data "consumer". An OPES processor may be in several trust domains at any time. There is no restriction on whether the OPES processors are authorized by data consumers and/or data providers. The original party has the option of forbidding or limiting redelegation.

An OPES processor must have a representation of its trust domain memberships that it can report in whole or in part for tracing purposes. It must include the name of the party which delegated each privilege to it.

3.2 Primary data flow

The primary data flow occurs between the data provider and the data consumer. OPES must not interfere with the capability of these parties to use end-to-end authentication and confidentiality.

If the primary parties want the assurance that their data does not appear in plaintext on network links, but they will permit use of plaintext on OPES processors and/or callout servers, then the OPES processors must use authentication and encryption between "hops".

A separate security association must be used for each channel established between two OPES processors.

3.3 Callout protocol

The determination of whether or not OPES processors will use the measures that are described in the previous section during their communication with callout servers depends on the details of how the primary parties delegated trust to the OPES processors and the trust relationship between the OPES processors and the callout server. If the OPES processors are in a single administrative domain with strong confidentiality guarantees, then encryption may be optional. In other cases, encryption and strong authentication would be at least strongly recommended.

If the delegation mechanism names the trusted parties and their privileges in some way that permits authentication, then the OPES processors will be responsible for enforcing the policy and for using authentication as part of that enforcement.

The callout servers must be aware of the policy governing the communication path. They must not, for example, communicate confidential information to auxiliary servers outside the trust domain.

A separate security association must be used for each channel established between an OPES processor and a callout server. The channels must be separate for different primary parties.

3.4 Privacy

Some data from data consumers is considered "private" or "sensitive", and OPES processors must both advise the primary parties of the their privacy policy and respect the policies of the primary parties. The privacy information must be conveyed on a per-flow basis.

The callout servers must also participate in handling of private

data, and they must be prepared to announce their own capabilities and to enforce the policy required by the primary parties.

3.5 Establishing trust

The OPES processor will have configuration policy specifying what privileges the callout servers have and how they are to be identified. This is especially critical for third-party (fourth-party, etc.) callout servers. OPES uses standard protocols for authenticating and otherwise security communication with callout servers.

An OPES processor will have a trusted method for receiving configuration information such as rules for the data dispatcher, trusted callout servers, primary parties that opt-in or opt-out of individual services, etc.

3.6 End-to-end Integrity

Digital signature techniques can be used to mark data changes in such a way that a third-party can verify that the changes are or are not consistent with the originating party's policy. This requires an inline manner of specifying policy and its binding to data, a trace of changes and the party making the changes, and strong identities and authentication methods.

Strong end-to-end integrity can fulfill some of the functions required by "tracing".

4. Summary

Although the architecture supports a wide range of cooperative transformation services, it has few requirements for interoperability.

The necessary and sufficient elements are specified in the following documents:

- the OPES ruleset schema [6] which defines the syntax and semantics of the rules interpreted by a data dispatcher; and,

- the OPES callout protocol (OCP) [7] which defines the protocol between a data dispatcher and a callout server.

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] 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.

[4] 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.

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

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

[7] OPES working group, "OPES Callout Protocol and Tracing Protocol Requirements", Internet-Draft TBD, May 2002.

Authors' Addresses

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

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

Robin Chen
AT&T Labs
Room E219, 180 Park Avenue
Florham Park, NJ 07932
US

Phone: +1 973 360 8653
EMail: chen@research.att.com

Markus Hofmann
Bell Labs/Lucent Technologies
Room 4F-513
101 Crawfords Corner Road
Holmdel, NJ 07733
US

Phone: +1 732 332 5983
EMail: hofmann@bell-labs.com

Hilarie Orman
The Purple Streak Development

EMail: ho@alum.mit.edu

Reinaldo Penno
Nortel Networks
2305 Mission College Boulevard
San Jose, CA 95134
US

EMail: rpenno@nortelnetworks.com

Gary Tomlinson
Cacheflow

EMail: gary@tomlinsongroup.net

Appendix A. Acknowledgements

The authors gratefully acknowledge the contributions of: Marshall T. Rose. (more to come, soon...)

Full Copyright Statement

Acknowledgement

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