Network Working Group
Request for Comments: 3835
Category: Informational

A. Barbir
R. Penno
Nortel Networks
R. Chen
AT&T Labs
M. Hofmann
Bell Labs/Lucent Technologies
H. Orman
Purple Streak Development

August 2004

An Architecture for Open Pluggable Edge Services (OPES)

Status of this Memo

Copyright Notice

Abstract

This memo defines an architecture that enables the creation of an application service in which a data provider, a data consumer, and zero or more application entities cooperatively implement a data stream service.

Table of Contents

1. Introduction

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

In some cases it may be beneficial to provide a customization service at a network location between the provider and consumer host rather than at one of these endpoints. For certain services performed on

behalf of the end-user, this may be the only option of service deployment. In this case, zero 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 OPES Use Cases and Deployment Scenarios [1] document provides examples of OPES services. The document discusses services that modify requests, services that modify responses, and services that create responses. It is recommended that the document on OPES Use Cases and Deployment Scenarios [1] be read before reading this document.

This 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 various parts of the document. Section 2.5 addresses tracing; section 3 addresses security considerations. Section 4 provides a summary of IAB considerations and how the architecture addresses them.

The document is organized as follows: Section 2 introduces the OPES architecture. Section 3 discusses OPES security and privacy considerations. Section 4 addresses IAB considerations for OPES. Section 5 discusses security considerations. Section 6 addresses IANA considerations. Section 7 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 specify when and how to execute OPES services.

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:

- an OPES service application, which analyzes and possibly transforms messages exchanged between the data provider application and the data consumer application;

- a data dispatcher, which invokes an OPES service application based on an OPES ruleset and application-specific knowledge.

The cooperative behavior of OPES entities introduces additional functionality for each data flow provided that it matches the OPES rules. In the network, OPES entities reside inside OPES processors. In the current work, an OPES processor MUST include a data dispatcher. Furthermore, the data provider and data consumer applications are not considered as OPES entities.

To provide verifiable system integrity (see section 3.1 on trust domains below) and to facilitate deployment of end-to-end encryption and data integrity control, OPES processors MUST be:

- explicitly addressable at the IP layer by the end user (data consumer application). This requirement does not preclude a chain of OPES processors with the first one in the chain explicitly addressed at the IP layer by the end user (data consumer application).

- consented to by either the data consumer or data provider application. The details of this process are beyond the scope of the current work.

The OPES architecture is largely independent of the protocol that is used by the data provider application and the data consumer application to exchange data. However, this document selects HTTP [3] as the example for the underlying protocol in OPES flows.

2.1.1. Data Dispatcher

Data dispatchers include a ruleset that can be compiled from several sources and MUST resolve into an unambiguous result. The combined ruleset enables an OPES processor to determine which service applications to invoke for which data flow. Accordingly, the data dispatcher constitutes an enhanced policy enforcement point, where policy rules are evaluated and service-specific data handlers and state information are maintained, as depicted in Figure 1.

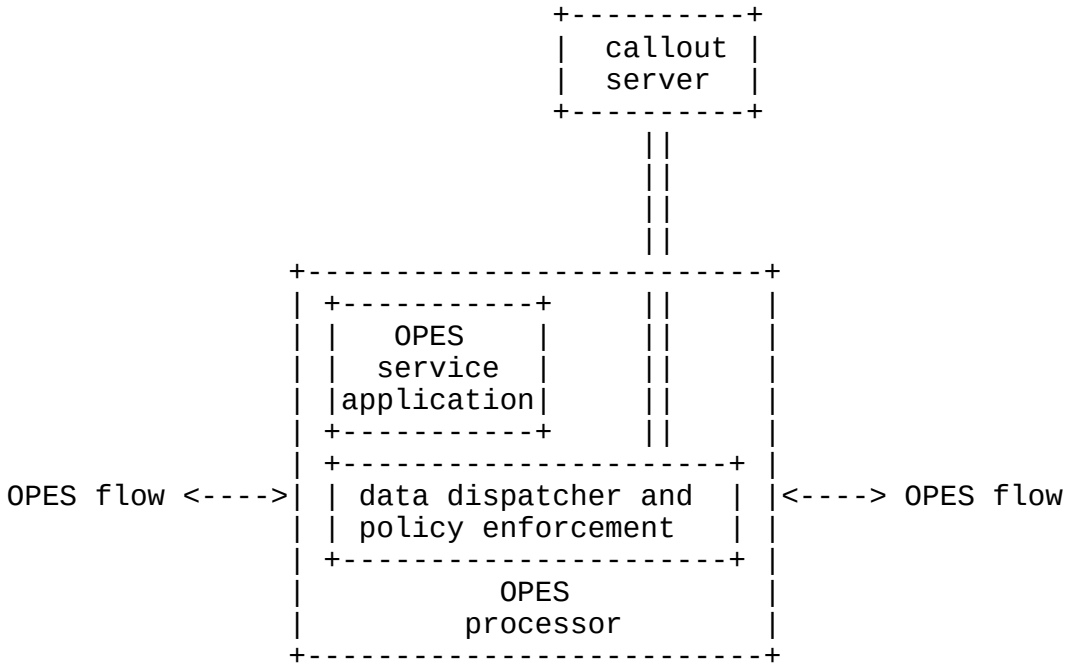

Figure 1: Data Dispatchers

The architecture allows for more than one policy enforcement point to be present on an OPES flow.

2.2. OPES Flows

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

Since policies are enforced by data dispatchers, the presence of at least one data dispatcher is required in the OPES flow.

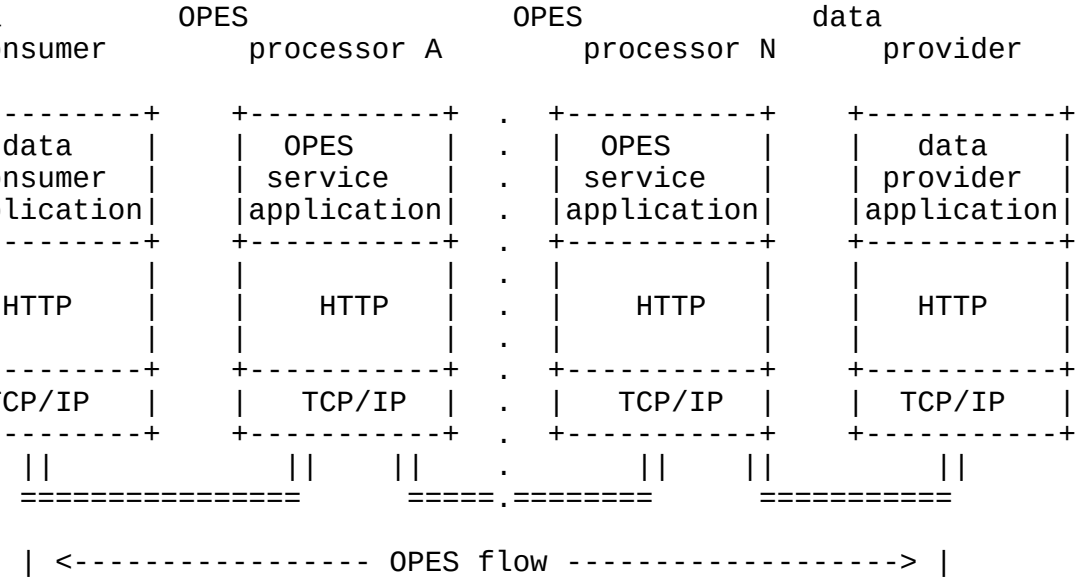

Figure 2: An OPES flow

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

2.3. OPES Rules

OPES' policy regarding services and the data provided to them is determined by a ruleset consisting of OPES rules. The rules consist of a set of conditions and related actions. The ruleset is the superset of all OPES rules on the processor. The OPES ruleset determines which service applications will operate on a data stream. In this model, all data dispatchers are invoked for all flows.

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 or a secure upload protocol), as long as such a mechanism complies with the requirements set forth in section 3.

2.4. Callout Servers

The evaluation of the OPES ruleset determines which service applications will operate on a data stream. How the ruleset is evaluated is not the subject of the architecture, except to note that it MUST result in the same unambiguous result in all implementations.

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. 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 [5]. The OCP is application-agnostic, being unaware of the semantics of the encapsulated application protocol (e.g., HTTP). However, the data dispatcher MUST incorporate a service aware vectoring capability that parses the data flow according to the ruleset and delivers the data to either the local or remote OPES service application.

The general interaction situation is depicted in Figure 3, which illustrates the positions and interaction of different components of OPES architecture.

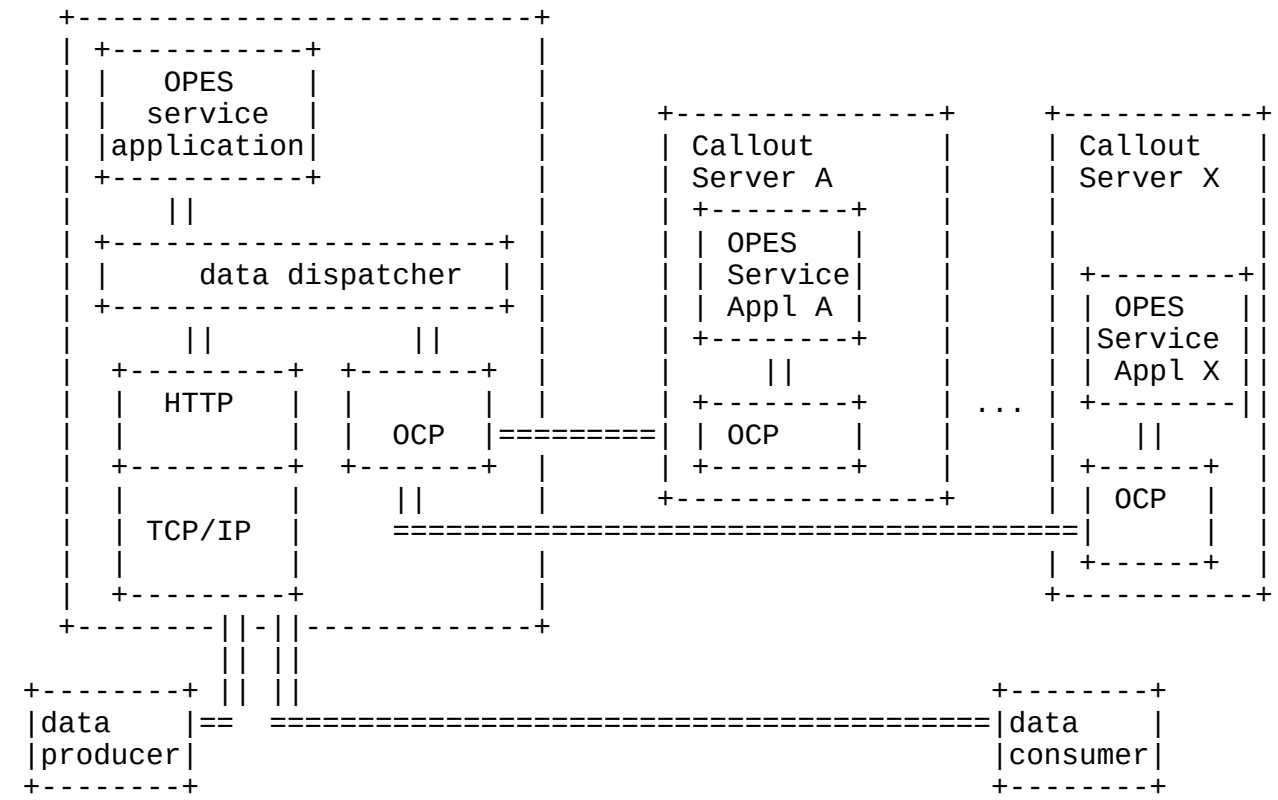

Figure 3: Interaction of OPES Entities

2.5. Tracing Facility

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. One of those annotations could be a URI with more detailed information on the OPES services being executed in the OPES flow.

Providing the ability for in-band annotation MAY require header extensions on the application protocol that is used (e.g., HTTP). However, the presence of an OPES processor in the data request/response flow SHALL NOT interfere with the operations of non-OPES aware clients and servers. Non-OPES clients and servers need not support these extensions to the base protocol.

OPES processors MUST obey tracing, reporting, and notification requirements set by the center of authority in the trust domain to which an OPES processor belongs. As part of these requirements, the OPES processor may be instructed to reject or ignore such requirements that originate from other trust domains.

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 flow.

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 forth. 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 defined are the data "provider" and the data "consumer". Delegation of authority (coloring) propagates from the content producer start of authority or from the content consumer start of authority, which may be different from the end points in the data flow.

Figure 4 illustrates administrative domains, out-of-band rules, and policy distribution.

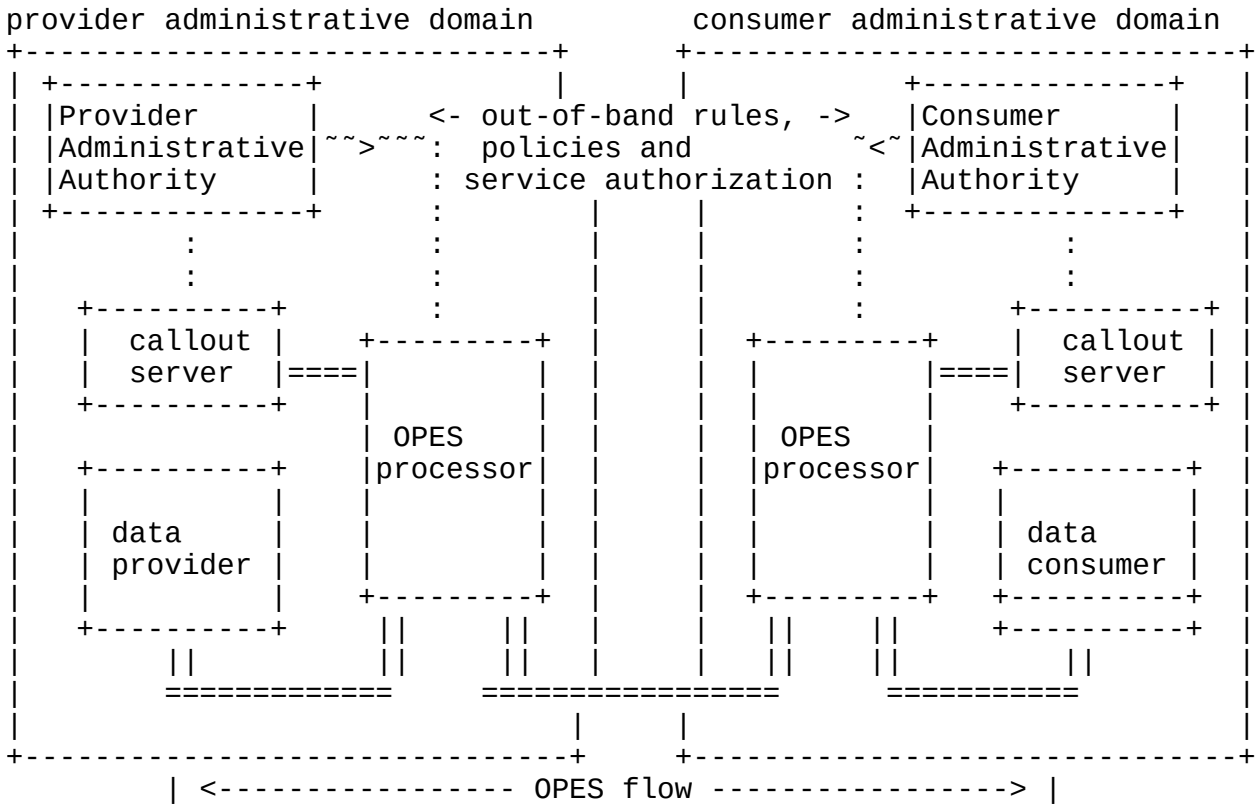

Figure 4: OPES administrative domains and policy distribution

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

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 that delegated each privilege to it.

3.2. Establishing Trust and Service Authorization

The OPES processor will have a configuration policy specifying what privileges the callout servers have and how they are to be identified. OPES uses standard protocols for authentication and other 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.

Protocol(s) for policy/rule distribution are out of scope for this document, but the OPES architecture assumes the existence of such a mechanism.

Requirements for the authorization mechanism are set in a separate document [4].

Service requests may be done in-band. For example, a request to bypass OPES services could be signalled by a user agent using an HTTP header string "Bypass-OPES". Such requests MUST be authenticated. The way OPES entities will honor such requests is subordinate to the authorization policies effective at that moment.

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. Strong authentication, message authentication codes, and encryption SHOULD be used. If the OPES processors are in a single administrative domain with strong confidentiality and integrity guarantees, then cryptographic protection is recommended but optional.

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 OPES flow endpoints is considered "private" or "sensitive", and OPES processors MUST advise the primary parties of their privacy policy and respect the policies of the primary parties. The privacy information MUST be conveyed on a per-flow basis. This can be accomplished by using current available privacy techniques such as P3P [7] and HTTP privacy capabilities.

The callout servers MUST also participate in the handling of private data, they MUST be prepared to announce their own capabilities, and enforce the policy required by the primary parties.

3.5. 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 method to specify policy and its binding to data, a trace of changes and the identity of the party making the changes, and strong identification and authentication methods.

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

4. IAB Architectural and Policy Considerations for OPES

This section addresses the IAB considerations for OPES [2] and summarizes how the architecture addresses them.

4.1. IAB Consideration (2.1) One-Party Consent

The IAB recommends that all OPES services be explicitly authorized by one of the application-layer end-hosts (that is, either the data consumer application or the data provider application).

The current work requires that either the data consumer application or the data provider application consent to OPES services. These requirements have been addressed in sections 2 (section 2.1) and 3.

4.2. IAB Consideration (2.2) IP-Layer Communications

The IAB recommends that OPES processors must be explicitly addressed at the IP layer by the end user (data consumer application).

This requirement has been addressed in section 2.1, by the requirement that OPES processors be addressable at the IP layer by the data consumer application.

4.3. IAB Consideration (3.1 and 3.2) Notification

The IAB recommends that the OPES architecture incorporate tracing facilities. Tracing enables data consumer and data provider applications to detect and respond to actions performed by OPES processors that are deemed inappropriate to the data consumer or data provider applications.

Section 3.2 of this document discusses the tracing and notification facilities that must be supported by OPES services.

4.4. IAB Consideration (3.3) Non-Blocking

The OPES architecture requires the specification of extensions to HTTP. These extensions will allow the data consumer application to request a non-OPES version of the content from the data provider application. These requirements are covered in Section 3.2.

4.5. IAB Consideration (4.1) URI Resolution

This consideration recommends that OPES documentation must be clear in describing OPES services as being applied to the result of URI resolution, not as URI resolution itself.

This requirement has been addressed in sections 2.5 and 3.2, by requiring OPES entities to document all the transformations that have been performed.

4.6. IAB Consideration (4.2) Reference Validity

This consideration recommends that all proposed services must define their impact on inter- and intra-document reference validity.

This requirement has been addressed in section 2.5 and throughout the document whereby OPES entities are required to document the performed transformations.

4.7. IAB Consideration (4.3) Application Addressing Extensions

This consideration recommends that any OPES services that cannot be achieved while respecting the above two considerations may be reviewed as potential requirements for Internet application addressing architecture extensions, but must not be undertaken as ad hoc fixes.

The current work does not require extensions of the Internet application addressing architecture.

4.8. IAB Consideration (5.1) Privacy

This consideration recommends that the overall OPES framework must provide for mechanisms for end users to determine the privacy policies of OPES intermediaries.

This consideration has been addressed in section 3.

5. Security Considerations

The proposed work has to deal with security from various perspectives. There are security and privacy issues that relate to data consumer application, callout protocol, and the OPES flow. In [6], there is an analysis of the threats against OPES entities.

6. IANA Considerations

The proposed work will evaluate current protocols for OCP. If the work determines that a new protocol needs to be developed, then there may be a need to request new numbers from IANA.

7. 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, which defines the syntax and semantics of the rules interpreted by a data dispatcher; and,
- the OPES callout protocol (OCP) [5], which defines the requirements for the protocol between a data dispatcher and a callout server.

8. References

8.1. Normative References

[1] Barbir, A., Burger, E., Chen, R., McHenry, S., Orman, H., and R. Penno, "Open Pluggable Edge Services (OPES) Use Cases and Deployment Scenarios", RFC 3752, April 2004.

[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., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[4] Barbir, A., Batuner, O., Beck, A., Chan, T., and H. Orman, "Policy, Authorization, and Enforcement Requirements of the Open Pluggable Edge Services (OPES)", RFC 3838, August 2004.

[5] Beck, A., Hofmann, M., Orman, H., Penno, R., and A. Terzis, "Requirements for Open Pluggable Edge Services (OPES) Callout Protocols", RFC 3836, August 2004.

[6] Barbir, A., Batuner, O., Srinivas, B., Hofmann, M., and H. Orman, "Security Threats and Risks for Open Pluggable Edge Services (OPES)", RFC 3837, August 2004.

8.2. Informative References

[7] Cranor, L. 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.

9. Acknowledgements

This document is the product of OPES WG. Oskar Batuner (Independent consultant) and Andre Beck (Lucent) are additional authors that have contributed to this document.

Earlier versions of this work were done by Gary Tomlinson (The Tomlinson Group) and Michael Condry (Intel).

The authors gratefully acknowledge the contributions of: John Morris, Mark Baker, Ian Cooper and Marshall T. Rose.

10. Authors' Addresses

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

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

Yih-Farn Robin Chen
AT&T Labs - Research
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
Purple Streak Development

EMail: ho@alum.mit.edu

Reinaldo Penno
Nortel Networks
600 Technology Park Drive
Billerica, MA 01821
USA

EMail: rpenno@nortelnetworks.com

11. Full Copyright Statement

Intellectual Property

Acknowledgement

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