              Open Pluggable Edge Services (OPES) SMTP Use Cases

Status of This Memo

Copyright Notice

Abstract

   The Open Pluggable Edge Services (OPES) framework is application
   agnostic. Application-specific adaptations extend that framework.
   This document describes OPES SMTP use cases and deployment scenarios
   in preparation for SMTP adaptation with OPES.

Table of Contents

1. Introduction

   The Open Pluggable Edge Services (OPES) architecture [1] 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 OPES processor can distribute
   the responsibility of service execution by communicating and
   collaborating with one or more remote callout servers.

   The execution of such services is governed by a set of rules
   installed on the OPES processor. The rule evaluation can trigger the
   execution of service applications local to the OPES processor or on a
   remote callout server.

   Use cases for OPES based on HTTP [8] are described in [2]. This work
   focuses on OPES for SMTP [7] use cases, whereby additional use cases
   and enhancements to the types of OPES services defined in [2] are
   provided.

   In SMTP, the OPES processor may be any agent participating in SMTP
   exchanges, including a Mail Submission Agent (MSA), a Mail Transfer
   Agent (MTA), a Mail Delivery Agent (MDA), and a Mail User Agent
   (MUA). This document focuses on use cases in which the OPES
   processor is a MTA.

   SMTP is a store-and-forward protocol. Current email filtering
   systems either operate during the SMTP exchange or on messages that
   have already been received, after the SMTP connection has been closed
   (for example, in an MTA's message queue).

   This work focuses on SMTP-based services that want to modify command
   values or want to block SMTP commands. In order to block a command,
   the service will provide an error message that the MTA should use in
   response to the command it received. An OPES MTA will be involved in
   SMTP command modification and command satisfaction, analogous to
   request modification and request satisfaction from HTTP [8].

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 [6]. When used with
   the normative meanings, these key words will be all uppercase.
   Occurrences of these words in lowercase comprise normal prose usage,
   with no normative implications.

3. Brief Overview of SMTP Architecture

   The SMTP design, taken from RFC 2821 [7], is shown in Figure 1. When
   an SMTP client has a message to transmit, it establishes a two-way
   transmission channel to an SMTP server. The responsibility of an
   SMTP client is to transfer mail messages to one or more SMTP servers,
   or report its failure to do so.

```
                +----------+                 +----------+
    +------+     |          |                 |          |
    | User |<-->|          |      SMTP       |          |
    +------+     |  Client  |Commands/Replies | Server   |
    +------+     |   SMTP   |<--------------->|  SMTP    |    +------+
    | File |<-->|          |     and Mail    |          |<-->| File |
    |System|     |          |                 |          |    |System|
    +------+     +----------+                 +----------+    +------+
              SMTP client                  SMTP server
```
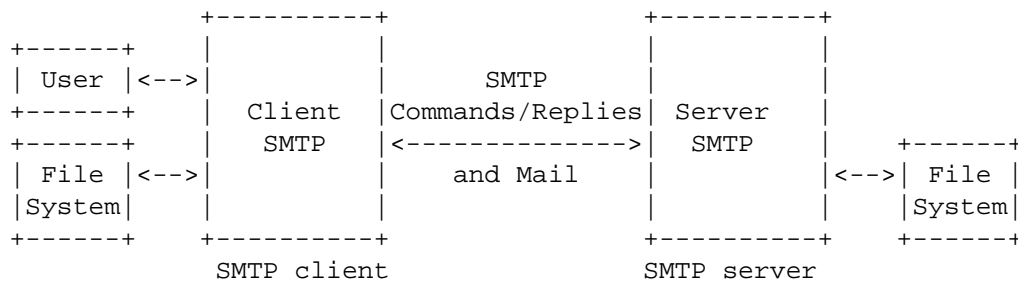
                         Figure 1: SMTP Design

   In some cases, the domain name(s) transferred to, or determined by,
   an SMTP client will identify the final destination(s) of the mail
   message. In other cases, the domain name determined will identify an
   intermediate destination through which those mail messages are to be
   relayed.

   An SMTP server may be either the ultimate destination or an
   intermediate "relay" or "gateway" (that is, it may transport the
   message further using some protocol other than SMTP or using again
   SMTP and then acting as an SMTP client).

   SMTP commands are generated by the SMTP client and sent to the SMTP
   server. SMTP responses are sent from the SMTP server to the SMTP
   client in response to the commands. SMTP message transfer can occur
   in a single connection between the original SMTP sender and the final
   SMTP recipient, or it can occur in a series of hops through
   intermediary systems. SMTP clients and servers exchange commands and
   responses and eventually the mail message body.

   Figure 2 expands on the mail flow in an SMTP system. Further
   information about the architecture of email in the Internet may be
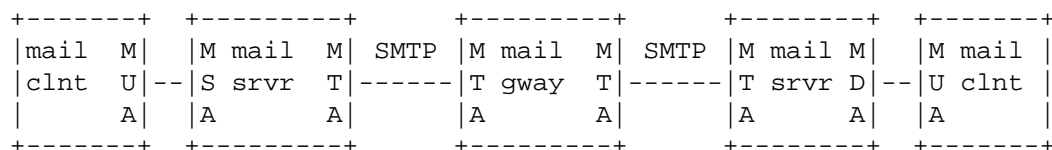   found in [9].

```
+-------+  +---------+          +---------+          +--------+  +-------+
|mail  M|  |M mail  M| SMTP |M mail  M| SMTP |M mail M|  |M mail |
|clnt  U|--|S srvr  T|------|T gway  T|------|T srvr D|--|U clnt |
|      A|  |A       A|          |A       A|          |A      A|  |A      |
+-------+  +---------+          +---------+          +--------+  +-------+
```

                    Figure 2: Expanded SMTP Flow

   In this work, the OPES processor may be any agent that is
   participating in SMTP exchanges, including an MSA, MTA, MDA, and MUA.
   However, this document focuses on use cases in which the OPES
   processor uses the SMTP protocol or one of the protocols derived from
   SMTP Message Submission (SUBMIT) [10] and the Local Mail Transfer
   Protocol (LMTP) [11]).

3.1. Operation Flow of an OPES SMTP System

   In this work, an MTA is the OPES processor device that sits in the
   data stream of the SMTP protocol. The OPES processor gets enhanced
   by an OCP (OPES callout protocol) [3] client that allows it to vector
   out data to the callout server. The filtering functionality is on
   the callout server.

   A client (a mail user) starts with an email client program (MUA).
   The user sends email to an outgoing email server. In the email
   server, there is an MSA (mail submission agent) that is waiting to
   receive email from the user. The MSA uses an MTA within the same
   server to forward the user email to other domains. (Communication
   between the MUA and MSA may be via SMTP, SUBMIT [10], or something
   else such as MAPI).

   The MTA in the user email server may directly contact the email
   server of the recipient or may use other intermediate email gateways.
   The sending email server and all intermediate gateway MTAs usually
   communicate using SMTP. Communication with the destination email
   server usually uses SMTP or its derivative, LMTP [11].

   In the destination email server, a mail delivery agent (MDA) may
   deliver the email to the recipient's mailbox. The email client
   program of the recipient might use a different protocol (such as the
   Post Office Protocol version 3 (POP3) or IMAP) to access the mailbox
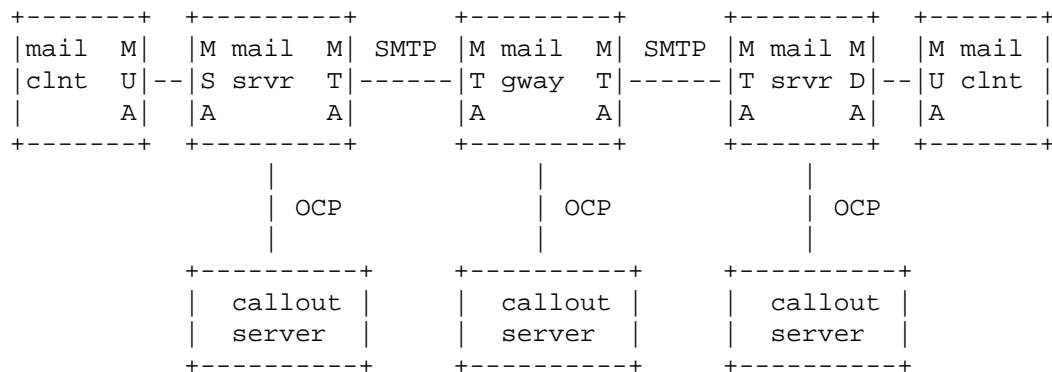   and retrieve/read the messages.

```
+-------+  +---------+          +---------+          +--------+  +-------+
|mail  M|  |M mail  M| SMTP     |M mail  M| SMTP     |M mail M|  |M mail |
|clnt  U|--|S srvr  T|------    |T gway  T|------    |T srvr D|--|U clnt |
|      A|  |A       A|          |A       A|          |A      A|  |A      |
+-------+  +---------+          +---------+          +--------+  +-------+
                |                    |                    |
                |  OCP               |  OCP               |  OCP
                |                    |                    |
           +----------+        +----------+        +----------+
           | callout  |        | callout  |        | callout  |
           | server   |        | server   |        | server   |
           +----------+        +----------+        +----------+
```

                     Figure 3: OPES SMTP Flow

   From Figure 3, the MTA (the OPES processor) is either receiving or
   sending an email (or both) within an email server/gateway. An OPES
   processor might be the sender's SMTP server, the destination SMTP
   server, or any intermediate SMTP gateway. (Which building block
   belongs to which authoritative domain is an important question but
   different from deployment to deployment.) Note that this figure
   shows multiple OPES deployment options in a typical chain of mail
   servers and gateways with different roles as MSA, MTA, and MDA; the
   OPES standard case, however, will only have a single OPES processor
   and a single callout server in the message flow.

3.1.1. OPES SMTP Example

   A typical (minimum) SMTP dialog between two OPES SMTP processors
   (MTA) will consist of the following (C: means client, S: means
   server):

      S: 220 mail.example.com Sample ESMTP MAIL Service, Version: 1.2
      ready at Thu, 20 Jan 2005 11:24:40+0100
      C: HELO [192.0.2.138]
      S: 250 mail.example.com Hello [192.0.2.138]
      C: MAIL FROM:<steve@example.org>
      S: 250 2.1.0 steve@example.org....Sender OK
      C: RCPT TO:<paul@example.com>
      S: 250 2.1.5 paul@example.com
      C: DATA
      S: 354 Start mail input; end with "CRLF"."CRLF"
      C: From: steve@example.org
      C: To: sandra@example.com
      C: Subject: Test
      C:
      C: Hi, this is a test!
      C: .

```
   S: 250 2.6.0 "MAIL0m4b1f@mail.example.com" Queued mail for
   delivery
   C: QUIT
   S: 221 2.0.0 mail.example.com Service closing transmission channel
```

The client (C:) is issuing SMTP commands and the server (S:) is
generating responses. All responses start with a status code and
then some text. At minimum, 4 commands are needed to send an email.
Together, all commands and responses to send a single email message
form "the dialog". The mail message body is the data sent after the
"DATA" command. An OPES processor could see that as command
modification.

If a callout service wants to adapt the email message body, it is
mainly interested in this part of the dialog:

```
   From: steve@example.org
   To: sandra@example.com
   Subject: Test

   Hi, this is a test!
```

The callout service may need to examine values of previous commands
of the same dialog. For example, the callout service needs to
examine the value of the RCPT command (it is "paul@example.com"),
which is different from the "sandra@example.com" that the email
client displays in the visible "To" field. That might be an
important fact for some filters such as spam filters (Section 4.2).

4. OPES/SMTP Use Cases

In principle, all filtering that is deployed at SMTP gateways today
and tomorrow defines use cases for OPES callout filtering. An
OCP/SMTP callout protocol will enable an SMTP gateway to vector out
(parts of) an SMTP message or parts of the SMTP dialog to a callout
server that is then performing actions on behalf of the gateway.
(OCP/SMTP would be a profile defined for OCP analogous to the
OCP/HTTP profile [4] that has been defined earlier.)

Here is a collection of some typical use cases describing different
filtering areas and different actions caused by those filters.

4.1. Security Filters Applied to Email Messages

These filters concentrate on the email message body and usually
filter the email sections one by one. Email sections (attachments)
that violate the security policy (e.g., because they contain a virus

or contain an unwanted mime type) define an event that can cause a
combination of different actions to be performed:

o The attachment is replaced by an error message.

o The email is marked by inserting a warning into the subject or the
   email body.

o An additional header is added for post-processing steps.

o The email storage is advised to put the email into quarantine.

o Notifications are sent to sender, recipients, and/or
   administrators.

o The incident is reported to other tools such as intrusion
   detection applications.

These kinds of filters usually do not require working with elements
of the SMTP dialog other than the email message body. An exception
to this is the need to map email senders and recipients to different
security sub-policies that are used for a particular message. A
security filter may therefore require receiving the information of
the RCPT TO and MAIL FROM commands as meta data with the email
message body it examines.

4.2. Spam Filter

Next to security filters, spam filters are probably the most wanted
filtering application today. Spam filters use several methods. They
concentrate most on the email message body (that also includes the
email headers), but many of these filters are also interested in the
values of the other SMTP commands in order to compare the SMTP
sender/recipients with the visible From/To fields. They may even
want to get the source IP of the connected SMTP client as meta
information to verify this against lists of open relays, known
spammers, etc.

These are typical actions that could be performed when a message has
been classified as spam:

o Add a mark to the subject of the email.

o Add an additional header for post-processing steps.

o The email storage is advised to put the email into a spam queue.

o The email is rejected or returned to the sender.

4.3. Logging and Reporting Filters

   The nature of these kinds of filters is not to modify the email
   message. Depending on what is being logged or reported on, the
   filter may need access to any part of the SMTP dialog. Most wanted
   is the sender and recipient information. Depending on the ability of
   the OPES processor to pre-calculate and transfer information about
   the message body, the callout filter may want to see the email
   message body itself or just that meta info; an example is the email
   size. This information would be typical logging and reporting
   information that is easy for the SMTP gateway to calculate although
   not a direct parameter of the SMTP dialog. Transferring the complete
   email message body only because the callout server wants to calculate
   its size would be a waste of network resources.

4.4. Access Control Filters

   These filters operate on the values of the MAIL FROM and RCPT TO
   commands of the SMTP dialog. They run an access control policy to
   determine whether a sender is currently allowed to send a message to
   the given recipients. The values of HELO/EHLO, AUTH, and STARTTLS
   commands may also be applied. The result of this filter has a direct
   influence on the SMTP response that the OPES processor has to send to
   its peer for the filtered SMTP command.

4.5. Secure Email Handling

   Filters of this kind can support an email gateway to centrally encode
   and decode email, and to set and to verify email signatures. They
   will therefore modify the email message body to encrypt, decrypt,
   verify, or sign the message, or use an action as specified in the
   "Security Filter" (Section 4.1) section if the decryption or
   signature verification fails.

   Sending the SMTP sender and recipient information as meta data to
   these filters is mission critical because these filters may not trust
   the information found in the header section of the email message
   body.

4.6. Email Format Normalization

   SMTP messages may be sent with an illegal or uncommon format; this
   may have happened by a buggy SMTP application or on purpose in order
   to exploit vulnerabilities of other products. A normalization filter
   can correct the email format. The format correction can be done for
   the email body and for the value of other SMTP commands. An example
   for the email body format correction would be a strange length of
   UUencoded lines or unusual names of MIME sections. Command values

may be analysed against buffer overflow exploits; a rewrite will not
always be possible in this case (cannot simply rewrite an email
address that is very long) but will require that the callout server
tells the OPES processor to send an error response in reply to such a
command.

4.7. Mail Rerouting and Address Rewriting

A corporation with multiple locations may want to deploy a central
gateway that receives all email messages for all employees and then
determines at which location the mail storage of the employee
resides. The callout server will then need the RCPT TO command value
and it will look up the location in the corporate directory service.
It then tells either the OPES processor where the next SMTP server is
(i.e., the next SMTP server to connect to) or it rewrites the
recipient address; in the first case, the SMTP servers at the
different locations accept emails of the same domain as the central
gateway does; in the second case, the other locations will probably
use the sublocation of the original domain (joe@example.org ->
joe@fr.example.org or joe@de.example.org).

4.8. Block Email during SMTP Dialog

In a first step, the callout server will check the sender and
recipient information that was transmitted in the SMTP dialog; that
information again maps to a policy that will deny all messages either
from that sender or to that recipient, or it checks the body of the
email and classifies it (maybe just by looking for some words in the
subject or by doing in-depth content analysis), which can then also
lead to the decision to deny the message.

Unlike previous examples, this use case wants to deny the email while
the SMTP dialog is still active, i.e., before the OPES processor
finally accepted the message. Depending on the exact policy, the
error response should then be sent in reply to the MAIL FROM, RCPT
TO, or DATA command.

4.9. Convert Attachments to HTTP Links

This use case will only modify the email message body without any
other influence on the SMTP dialogs, mail routing, etc. Larger
sections (attachments) are removed from the email, and the content is
stored on a web server. A link to that new URL is then added into
the text of a first section that is likely to be displayed by an
email client. Storing the attachments onto the web server is not in
the scope of the OPES/SMTP scenario and needs to be implemented by
the callout server.

5. Security Considerations

   Application-independent security considerations are documented in
   application-agnostic OPES specifications [5]. This document contains
   only use cases and defines no protocol operations. Security
   considerations for protocols that appear in these use cases are
   documented in the corresponding protocol specifications.

   Use case "Secure Email Handling" (Section 4.5) is special in this
   regard because it requires the extension of the end-to-end encryption
   model and a secure handling of private cryptographic keys when
   creating digital signatures or when decrypting messages. Both are
   out of scope of OPES protocol specifications. An implementation of
   such a service raises security issues (such as availability and
   storage of cryptographic keys) that must be addressed regardless of
   whether the implementation happens within an MTA or within an OPES
   callout server.

6. References

6.1. Normative References

   [1]   Barbir, A., Penno, R., Chen, R., Hofmann, M., and H. Orman, "An
         Architecture for Open Pluggable Edge Services (OPES)", RFC 3835,
         August 2004.

   [2]   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.

   [3]   Rousskov, A., "Open Pluggable Edge Services (OPES) Callout
         Protocol (OCP) Core", RFC 4037, March 2005.

   [4]   Rousskov, A. and M. Stecher, "HTTP Adaptation with Open
         Pluggable Edge Services (OPES)", RFC 4236, November 2005.

   [5]   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.

   [6]   Bradner, S., "Key words for use in RFCs to Indicate Requirement
         Levels", BCP 14, RFC 2119, March 1997.

6.2. Informative References

   [7]   Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April
         2001.

   [8] 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.

   [9] Crocker, D., "Internet Mail Architecture", Work in Progress,
        March 2005.

   [10] Gellens, R. and J. Klensin, "Message Submission", RFC 2476,
        December 1998.

   [11] Myers, J., "Local Mail Transfer Protocol", RFC 2033, October
        1996.

Acknowledgements

Authors' Addresses

   Martin Stecher
   Secure Computing Corporation
   Vattmannstr. 3
   33100 Paderborn
   Germany

   EMail: martin.stecher@webwasher.com
   URI: http://www.securecomputing.com/

   Abbie Barbir
   Nortel
   3500 Carling Avenue
   Ottawa, Ontario
   CA

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