Open Pluggable Edge Services
Internet-Draft
Expires: April 16, 2006

M. Stecher
CyberGuard
C. Perz
All About IT
October 13, 2005

SMTP adaptation with OPES
draft-ietf-opes-smtp-00

Status of this Memo

Copyright Notice

Abstract

Open Pluggable Edge Services (OPES) framework documents several application-agnostic mechanisms such as OPES tracing, OPES bypass, and OPES callout protocol. This document extends those generic mechanisms for Simple Mail Transfer Protocol (SMTP) adaptation. Together, application-agnostic OPES documents and this SMTP profile constitute a complete specification for SMTP adaptation with OPES.

Table of Contents

1. Scope

The Open Pluggable Edge Services (OPES) framework documents several application-agnostic mechanisms such as OPES processor and endpoints communications [RFC3897] or OPES callout protocol [RFC4037]. This document extends those generic mechanisms for adaptation of a specific application protocol, SMTP [RFC2821]. Together, application-agnostic OPES documents and this SMTP profile constitute a complete specification for SMTP adaptation with OPES.

The primary sections of this document specify SMTP-specific extensions for the corresponding application-agnostic mechanisms documented elsewhere.

2. OPES Document Map

This document belongs to a large set of OPES specifications produced by the IETF OPES Working Group. Familiarity with the overall OPES approach and typical scenarios is often essential when trying to comprehend isolated OPES documents. This section provides an index of OPES documents to assist the reader with finding "missing" information.

- The document on "OPES Use Cases and Deployment Scenarios" [RFC3752] describes a set of services and applications that are considered in scope for OPES and have been used as a motivation and guidance in designing the OPES architecture.

- Usecases specifically for SMTP that are motivation for the SMTP adaptation described in this document have been lised in "OPES SMTP Use Cases" [I-D.ietf-opes-smtp-use-cases].

- The OPES architecture and common terminology are described in "An Architecture for Open Pluggable Edge Services (OPES)" [RFC3835].

- "Policy, Authorization and Enforcement Requirements of OPES" [RFC3838] outlines requirements and assumptions on the policy framework, without specifying concrete authorization and enforcement methods.

- "Security Threats and Risks for OPES" [RFC3837] provides OPES risk analysis, without recommending specific solutions.

- "OPES Treatment of IAB Considerations" [RFC3914] addresses all architecture-level considerations expressed by the IETF Internet Architecture Board (IAB) when the OPES WG was chartered.

- At the core of the OPES architecture are the OPES processor and the callout server, two network elements that communicate with each other via an OPES Callout Protocol (OCP). The requirements for such protocol are discussed in "Requirements for OPES Callout Protocols" [RFC3836].

- "OPES Callout Protocol Core" [RFC4037] specifies an application agnostic protocol core to be used for the communication between OPES processor and callout server.

- "OPES entities and end points communications" [RFC3897] specifies generic tracing and bypass mechanisms for OPES.

- The OCP Core and Communications documents are independent from the application protocol being adapted by OPES entities. Their

generic mechanisms have to be complemented by application-specific profiles. This document, SMTP adaptation with OPES, is such an application profile for SMTP. It specifies how application-agnostic OPES mechanisms are to be used and augmented in order to support adaptation of SMTP messages.

- Finally, "P: Message Processing Language" [I-D.ietf-opes-rules-p] defines a language for specifying what OPES adaptations (e.g, translation) must be applied to what application messages (e.g., e-mail from bob@example.com). P language is meant for configuring application proxies (OPES processors).

3. Callout Protocol

This section documents the SMTP profile for the OPES Callout Protocol (OCP) Core [RFC4037]. Familiarity with OCP Core is required to understand the SMTP profiles. This section uses OCP Core conventions, terminology, and mechanisms.

The OPES processor communicates its desire to adapt SMTP messages via a Negotiation Offer (NO) message with SMTP-specific feature identifiers documented in Section 3.2. SMTP-specific OCP optimization mechanisms can be negotiated at the same time. A callout server that supports adaptation of SMTP messages has a chance to negotiate what SMTP message parts will participate in adaptation, including SMTP commands and email body elements as application message parts documented in Section 3.1.

3.1 Application Message Parts

An SMTP message may have several well-known parts: SMTP commands such as HELO, MAIL, RCPT and email content sent after the DATA command. The email content has a header and a body; and the email body again can be divided into several sections.

An SMTP OPES processor has to have information about at least some SMTP message parts (of those SMTP commands it supports). It may or may not be able to divide the email content into parts. A callout server may want to ask the OPES processor to do email content preprocessing for a more efficient OCP handling. The agents will negotiate the capabilities of the OPES processor and the needs of the callout server using the application message parts in additional parameters that will be defined for the Negotiation Offer (NO) and Negotiation Response (NR) messages in Section 3.2.1.

The following is the declaration of am-part (application message part) type using OCP Core Protocol Element Type Declaration Mnemonic (PETDM):

```
am-part:  extends atom;
am-parts: extends list of am-part;
```

Figure 1

The following "am-part" atoms are valid values:

EHLO: The argument of the Extended HELLO command as defined in section 4.1.1.1 of [RFC2821].

HELO: The argument of the HELLO command as defined in section 4.1.1.1 of [RFC2821].

MAIL: The argument of the MAIL command as defined in section 4.1.1.2 of [RFC2821].

RCPT: The argument of the RECIPIENT command as defined in section 4.1.1.3 of [RFC2821].

VRFY: The argument of the VERIFY command as defined in section 4.1.1.6 of [RFC2821].

EXPN: The argument of the EXPAND command as defined in section 4.1.1.7 of [RFC2821].

RAWDATA: The complete mail data which is sent AFTER the DATA command (see section 4.1.1.4 of [RFC2821]).

ALLHEADERS: The header of the email data including the empty line at the end as defined in section 2.1 of [RFC2822].

SINGLEHEADERS: Some or all header fields of the email data, each to be sent in a separate OCP message.

BODY: The body of the email data as defined in section 2.3 of [RFC2822].

SECTIONS: Sections of the email body (for example MIME sections), each to be sent in a separate OCP message.

The list of calid "am-part" atoms can be extended, especially to represent commands that have been defined in extension to [RFC2821]. A callout server MUST ignore unknown "am-part" atoms.

Some commands defined in [RFC2821] MUST NOT be used as application message parts for OCP. These include DATA, RSET and QUIT.

3.2 Application Profile Features

This document defines two SMTP profiles for OCP: sender and receiver profiles. These two profiles are described below. Each profile has a unique feature identifier:

profile ID: http://iana.org/opes/ocp/SMTP/sender

profile ID: http://iana.org/opes/ocp/SMTP/receiver

The sender profile is used by the OPES processor while or just before he is sending a message to another peer using the SMTP protocol, the receiver defines the opposite: the OPES processor is receiving or has just received a message from another peer using the SMTP protocol.

The scope of a negotiated profile is the OCP connection (default) or the service group specified via the SG parameter.

3.2.1 Profile Parts

If an SMTP OPES processor receives a valid RSET or QUIT command from its SMTP peer it MUST terminate the corresponding OCP transaction. The order of application message parts in OCP transaction MUST follow the order of commands in the SMTP transaction as defined in section 4.1.4 of [RFC2821]. An OCP agent SHOULD terminate OCP transactions with incorrect application message part orders.

Some application message parts are mutually exclusive within one OCP transaction: RAWDATA and any message part of the following list are mutually exclusive: ALLHEADERS, SINGLEHEADERS, BODY, SECTIONS. ALLHEADERS and SINGLEHEADERS are mutually exclusive. BODY, SECTIONS are mutually exclusive.

3.2.2 Profile Structure

An SMTP application profile feature extends semantics of the feature type of OCP Core while adding the following named parameters to that type:

- Adaptive-Parts (Section 3.2.3)
- Informative-Parts (Section 3.2.4)
- Header-List (Section 3.2.5)

The definition of the SMTP profile feature structure using PETDM follows:

```
SMTP-Profile: extends Feature with {
    [Adaptive-Parts: am-parts];
    [Informative-Parts: am-parts];
    [Header-List: headernames];
};
```

Figure 2

An SMTP profile structure can be used in feature lists of Negotiation Offer (NO) messages and as anonymous parameter of an Negotiation Response (NR) mesage. All profile parameters apply to any OCP transaction within profile scope.

3.2.3 Adaptive-Parts

The list of application message parts negotiated as Adaptive-Parts specify those parts that the OPES processor allows the callout server to change and that the callout server expects to adapt while running the specified callout service.

The OPES processor MUST NOT list application message parts as Adaptive-Commands for which it is not willing or able to accept changes or error responses from the callout server.

A callout server SHOULD only list those message parts in the Adaptive-Parts parameter of the Negotiation Response (NR) message that it may eventually change. Those message parts that the callout server needs for information only SHOULD be moved to the Informative-Parts list.

The callout server MUST NOT add a message part to the Adaptive-Parts list of the Negotiation Response (NR) message if that part was not contained in the Adaptive-Parts list of the Negotiation Offer (NO) message. The OPES processor MUST close a connection if a message part is returned in the Negotiation Response (NR) message that it does not support.

The OPES processor is not bound to the mutual exclusive rule for application message parts as defined in Section 3.2.1 and can list any application message parts it supports. The callout server MUST NOT list application message parts in the Negotiation Response (NR) message that are mutually exclusive.

The Adaptive-Parts parameter can be omitted which then means an empty list of am-part atoms.

3.2.4 Informative-Parts

In addition to the Adaptive-Parts the OCP agents can negotiate the inclusion of auxiliary application message parts by using the Informative-Parts parameter. Message parts of this list will not be adaptable by the callout server but provide meta information that is needed for the service.

All application message parts that the OPES processor has offered as Adaptive-Parts can also be returned as Informative-Parts by the

callout server. The OPES processor SHOULD NOT list application message parts as Informative-Parts again if the part was already listed under Adaptive-Parts although this does not define a semantical difference. A callout server MUST NOT list application message parts as Informative-Parts if it already lists them as Adaptive-Parts; the OPES processor MUST close the connection is such a case.

The callout server MUST NOT add a message part to the Informative-Parts list of the Negotiation Response (NR) message if that part was neither contained in the Adaptive-Parts list nor in the Informative-Parts list of the Negotiation Offer (NO) message. The OPES processor MUST close a connection if a message part is returned in the Negotiation Response (NR) message that it does not support.

The OPES processor is not bound to the mutual exclusive rule for application message parts as defined in Section 3.2.1 and can list any application message parts it supports. The callout server MUST NOT list application message parts in the Negotiation Response (NR) message that are mutually exclusive.

The Informative-Parts parameter can be omitted which then means an empty list of am-part atoms.

3.2.5 Header-List

The callout server can use the Header-List parameter in an Negotiation Response (NR) OCP message if it listed the SINGLEHEADERS message part token in either Adaptive-Parts or Informative-Parts. With this parameter it specifies which single headers it wants to receive during the transactions. Before this the OPES processor has indicated its ability to sent separated email headers by listing SINGLEHEADERS in the Adaptive-Parts or Informative-Parts of the Negotiation Offer (NO) message.

The following is the declaration of headernames (list of header names) type using OCP Core Protocol Element Type Declaration Mnemonic (PETDM); the type is used as a the value type for the Header-List parameter:

```
headername:  atom / "*";
headernames: extends list of headername;
```

Figure 3

[Note: May not be a good idea to allow "*" as a token as this is not a bare atom; may be better to defined headername as atom and use the wildcard in the quoted form "1:*"]

The value of headername is the name of a case insensitive email header that should be transmitted to the callout server. The wildcard "*" requests all available headers.

The requested header lines are transmitted in separated OCP messages, using one message for each header. Requestes headers are only sent if they exist in the original message.

The callout server MUST use a Header-List parameter if it listed SINGLEHEADERS as an Informative- or Adaptive-Part and it MUST NOT use the Header-List parameter if SINGLEHEADERS is not listed as such a part.

3.2.6 Negotiation Examples

In this example the OPES processor offers recipient information and the message data as Adaptive-Parts, which means that it is willing to accept changes to these items. As auxiliary information, it can send IP, HELO and MAIL command arguments. The callout server replies that it only needs DATA, MAIL and RCPT and it might only change the DATA part of the message.

```
Example 1: P=OPES processor, S=Callout Server
P: NO ({"38:http://iana.org/opes/ocp/SMTP/receiver"
   Adaptive-Commands: (RCPT,DATA)
   Informative-Commands: (IP,HELO,MAIL)
   })
   SG: 25
   ;
S: NR {"38:http://iana.org/opes/ocp/SMTP/receiver"
   Adaptive-Commands: (DATA)
   Informative-Commands: (MAIL,RCPT)
   }
   SG: 25
   ;
```

Figure 4

In the second example the callout server accepts all items offered by the OPES processor and requests some single headers from processed messages.

```
Example 2: P=OPES processor, S=Callout Server
P: NO ({"38:http://iana.org/opes/ocp/SMTP/receiver"
   Adaptive-Commands: (MAIL,RCPT)
   Informative-Commands: (IP,EHLO,SINGLEHEADERS)
   })
   SG: 25
   ;
S: NR {"38:http://iana.org/opes/ocp/SMTP/receiver"
   Adaptive-Commands: (MAIL,RCPT)
   Informative-Commands: (IP,EHLO,SINGLEHEADERS)
   Header-List: (From,To,Reply-To,Received)
   }
   SG: 25
   ;
```

Figure 5

3.3 DUM and DUY Messages

The SMTP application profiles extend semantics of the DUM and DUY messages defined in OCP Core by adding the following named parameters:

- AM-Part (Section 3.3.1)
- Allow (Section 3.3.2)
- SMTP-Error (Section 3.3.3)
- Advice (Section 3.3.4)
- Add-Header (Section 3.3.5)

The following parameters are defined for DUM messages:

```
AM-Part: am-part
Allow: supported-params
SMTP-Error: atom
Advice: advice
Add-Header: atom
```

Figure 6

The following parameters are defined for DUY messages:

```
Advice: advice
Add-Header: atom
```

Figure 7

3.3.1 AM-Part Parameter

An OCP agent MUST send an AM-Part parameter with every DUM message that is a part of an OCP transaction with an SMTP profile. The AM-Part parameter value is a single am-part token. As implied by the syntax, a DUM message can only contain data of a single application message part.

Only the following message parts can be fragmented into any number of DUM messages with the same AM-Part parameter: RAWDATA, ALLHEADERS, BODY, SECTIONS. All other application message parts MUST each be sent in a single DUM message.

The following example shows four DUM messages containing an abridged SMTP response transaction. The RAWDATA part is fragmented and sent within two DUM messages.

```
DUM 72 1 0
Kept: 0
AM-Part: MAIL

19:<steve@example.org>
;
DUM 72 1 19
Kept: 19
AM-Part: RCPT

18:<paul@example.com>
;
DUM 72 1 37
Kept: 37
AM-Part: RAWDATA

49:From: steve@example.org
To: sandra@example.com

;
DUM 72 1 86
Kept: 86
AM-Part: RAWDATA

41:Subject: Test

Hi, this is a test!
.

;
```

Figure 8

3.3.2 Allow Parameter

The Allow parameter is used by the OPES processor to indicate to the callout server which optional parameters are supported by the OPES processor when receiving DUM and DUY messages from the callout server. This information can be important for the callout server. For example: A callout server may prefer to have the OPES processor to send an SMTP error in response to email message data in order to reject the message but if the OPES processor is not able to do this (or not willing to do this) the callout server may want to exchange the email body content by an error message.

The following is the declaration of supported-params type (value type of the Allow Parameter) using OCP Core Protocol Element Type

```
Declaration Mnemonic (PETDM):
supported-param:  extends atom;
supported-params: extends list of supported-param;
```

Figure 9

The following "supported-param" atoms are defined by this document:

SMTP-Error: The OPES processor will accept an SMTP error response for this application message part. The callout server MAY use the SMTP-Error parameter as defined in Section 3.3.3.

Advice: The OPES processor will accept an advice from the callout server what to do with this message. The callout server MAY use the Advice parameter as defined in Section 3.3.4.

Add-Header: The OPES processor will accept an Add-Header directive. The callout server MAY use the Add-Header parameter as defined in Section 3.3.5.

The list of possible values for the "supported-param" atom can be extended, especially by new parameter names of DUM and DUY messages that will be defined in extensions to this document. Therefore the callout server MUST ignore unknown atoms in the supported-params list. The "supported-params" list MAY be empty.

The Allow parameter MUST NOT be sent by the callout server. The OPES processor SHOULD add an Allow parameter to a DUM message if it is able and willing to accept some of the additional parameters in DUM and DUY messages that the callout server sends in response to the DUM message of the OPES processor carrying the Allow parameter (i.e. DUM messages with the same AM-Part and DUY messages referencing the data of DUM messages). The Allow parameter MAY be omitted which is semantically equal to an empty "supported-params" list. The Allow parameter can be used for Adaptive-Parts as well as for Informative-Parts.

3.3.3 SMTP-Error Parameter

The SMTP-Error parameter can be used by the callout server if it wants the OPES processor to reply with an SMTP error to the SMTP command that is encapsulated in the DUM message that the callout server received.

By changing the payload of a DUM message the callout server adapts the corresponding SMTP argument or email data. When adding a SMTP-Error parameter the callout server SHOULD send an empty payload.

The OPES processor MUST NOT send SMTP-Error parameters with DUM messages, the callout server SHOULD NOT send SMTP-Error parameters with a DUM message if SMTP-Error was not in the list of "supported-params" in the Allow-Parameter of the DUM message it responds to. The SMTP-Error parameter MUST NOT be sent in a DUY message.

The value atom of the SMTP-Error parameter is a quoted-value with an SMTP reply as defined in s ection 4.2 of [RFC2821] but without the final CRLF characters at the end of the (last) reply line.

```
Example: P=OPES processor, S=Callout Server
P: DUM 72 1 0
   Kept: 0
   AM-Part: RCPT
   Allow: (SMTP-Error)

   18:<paul@example.com>
   ;
S: DUM 72 1 0
   AM-Part: RCPT
   SMTP-Error: "21:550 No such user here"

   0:
   ;
```

Figure 10

3.3.4 Advice Parameter

Advice can be s.th. like "Discard" or "Quarantine". More definitions are needed here.

To be done

3.3.5 Add-Header Parameter

The Add-Header parameter can be used by the callout server to have the OPES processor to add an additional header line to the email data. This can be an interesting side effect especially if the callout server does not intend to modify the email content application parts for other purposes.

The OPES processor MUST NOT send Add-Header parameters with DUM messages, the callout server SHOULD NOT send Add-Header parameters with DUM or DUY messages if Add-Header was not in the list of "supported-params" in the Allow-Parameter of the DUM message it

responds to.

The value atom of the Add-Header parameter is a quoted-value with an SMTP header lines as defined in s ection 2.2 of [RFC2822] but without the terminating CRLF characters.

```
Example: P=OPES processor, S=Callout Server
P: DUM 72 1 0
   Kept: 0
   AM-Part: RCPT
   Allow: (SMTP-Error,Add-Header)

   18:<paul@example.com>
   ;
S: DUM 72 1 0
   AM-Part: RCPT
   Add-Header: "33:X-Original-User: paul@example.com"

   21:<newuser@example.com>
   ;
```

Figure 11

3.4 SMTP Extension handling

Section 2.2 of [RFC2821] describes the Extension Model for SMTP, "that permits the client and server to agree to utilize shared functionality beyond the original SMTP requirements". Extensions can add or replace SMTP keywords and may add additional parameters to the SMTP MAIL and RCPT commands.

Further, extensions could create new meta information, that is not part of the extension, but relevant for certain actions of a callout server. Authentication within an SMTP session is an extension, but the information wether the authentication process was successful or not is in the domain of the MTA.

An MTA will use extensions during an SMTP session, regardless if a callout server knows about them or not. But the callout server could benefit from any data, that was exchanged using it.

Thus an OPES SMTP profile needs three features to deal with SMTP extensions.

- The callout server must be able to tell the OPES processor, what extensions it knows about (Section 3.4.1)
- The OPES processor needs a method how to submit the extension data to the callout server (Section 3.4.2)
- Keywords for exchanging meta information must be defined (Section 3.4.3)

3.4.1 Negotiating SMTP Extensions

The callout server will add a named parameter "Extensions" to the NR message with a list of keywords of extensions it supports. It uses the the EHLO response keyword value associated with the extension as defined in the underlying RFC.

```
Example:   Extensions are AUTH and SIZE
P: NO ({"38:http://iana.org/opes/ocp/SMTP/receiver"
   Adaptive-Commands: (RCPT,DATA)
   Informative-Commands: (IP,HELO,MAIL)
   })
   SG: 25
   ;
S: NR {"38:http://iana.org/opes/ocp/SMTP/receiver"
   Adaptive-Commands: (DATA)
   Informative-Commands: (MAIL,RCPT)
   Extensions: (AUTH,SIZE)
   }
   SG: 25
   ;
```

Figure 12

3.4.2 Transfer of extension information

The OPES processor can then send the data in the same way to the callout server as it would do to SMTP receivers that indicated their extension support with the keywords in the EHLO response.

These can be parameters with the MAIL and RCPT command arguments or additional commands (which have also been negotiated in the am-parts lists of the NO/NR messages.

```
Example 1:   SIZE is listed as supported extension
P: DUM 72 1 0
   AM-Part: MAIL
   Allow: (SMTP-Error,Advice)
   18:<paul@example.com> SIZE=256
   ;
```

Figure 13

```
Example 2: Fictive extension VIPEXT,
           that defines the new keyword VIPNAME:
P: NO ({"38:http://iana.org/opes/ocp/SMTP/receiver"
   Adaptive-Commands: (RCPT,DATA)
   Informative-Commands: (IP,HELO,MAIL,VIPNAME)
   })
   SG: 25
   ;
S: NR {"38:http://iana.org/opes/ocp/SMTP/receiver"
   Adaptive-Commands: (DATA)
   Informative-Commands: (MAIL,RCPT,VIPNAME)
   Extensions: (AUTH,SIZE,VIPEXT)
   }
   SG: 25
   ;
P: DUM 72 1 0
   AM-Part: VIPNAME
   Allow: (SMTP-Error,Advice)

   10:"John Doe"
   ;
```

Figure 14

3.4.3 Extension meta information

Addtional information as a result of using extensions between two SMTP peers can be transferred by putting the AM-OPT parameter in a DUM message. Any data an OPES processor knows about can be send in the described way. The callout server SHOULD provide a mechanism to map the processors keywords to its internal names.

```
            P: DUM 33 27
               AM-Part: MAIL
               AM-OPT: ({authtype cram-md5},{authen true})

               53:sender@example.org SIZE=33423 AUTH=sender@example.org
               ;
```

Figure 15

3.5 Examples

To be done.

4. Tracing

[RFC3897] defines application-agnostic tracing facilities in OPES. Compliance with this specification requires compliance with [RFC3897].

4.1 Tracing Headers

When adapting SMTP, trace entries are supplied using header lines for the message content. The following extension headers are defined to carry trace entries. Their definitions are given using BNF notation; the definition for "absoluteURI" is taken from [RFC2396].

```
OPES-System = "OPES-System" ":" #trace-entry
OPES-Via    = "OPES-Via" ":" #trace-entry

trace-entry = opes-agent-id *( ";" parameter )
opes-agent-id = absoluteURI
```

Figure 16

An OPES System MUST add its trace entry to the OPES-System header. Other OPES agents MUST use the OPES-Via header if they add their tracing entries. All OPES agents MUST append their entries. Informally, OPES-System is the only required OPES tracing header while OPES-Via provides optional tracing details; both headers reflect the order of trace entry additions.

An OPES System MUST NOT change OPES-System and OPES-VIA headers that were previously added to the message header. OPES agents MUST prepend OPES-System and OPES-VIA lines before all existing OPES-System and OPES-VIA lines; they MUST NOT change the order of existing lines or insert OPES-System and OPES-VIA lines in any other location. If an OPES System is using both headers, it MUST add identical trace entries except it MAY omit some or all trace-entry parameters from the the OPES-Via header. Informally, the OPES System entries in the OPES-Via header are used to delimit and group OPES-Via entries from different OPES Systems without having a priory knowledge about OPES System identifiers.

For example, here is an email message header after OPES adaptations have been applied by two OPES processors, the first executing 10 OPES services:

```
Received: from gateway.example.com ([192.0.2.138])
 by mail.example.com with testserver;
 Mon, 10 Oct 2005 05:37:19 +0200
Received: from mail2.example.org [192.0.2.99]
 by gateway.example.com id 33W9WIMC;
 Mon, 10 Oct 2005 05:35:55 +0200
OPES-System: http://mail.example.com/opes?id=33W9WIMC
OPES-System: http://gateway.example.com/opes?session=33W9WIMC
OPES-Via: http://gateway.example.com/opes?session=33W9WIMC,
          http://www.opes-services-4u.com/cat/?sid=123,
          http://www.opes-services-4u.com/cat/?sid=124,
          http://www.opes-services-4u.com/cat/?sid=125 ; mode=A
Subject: Test
From: "Steve" <steve@example.org>
To: "Sandra" <sandra@example.com>
```

Figure 17

In the above example, the first OPES processor has not included its trace entry or its trace entry was replaced by an OPES system trace entry. Only 3 out of 10 services are traced. The remaining services did not include their entries or their entries were removed by OPES system or processor. The last traced service included a "mode" parameter. The second OPES system has added its trace line before the other header lines. Various identifiers in trace entries will probably have no meaning to the recipient of the message, but may be decoded by OPES System software.

OPES entities MAY place optional tracing entries in the message content.

4.2 SMTP Trace Extension

An OPES processor MUST support the SMTP Service Extension for OPES Trace and Bypass [to be done in external document; referenced from here].

To request notitifications about actions taken by OPES intermediates while a message is relayed to its recipients, the sender of the message adds the parameter "opestrace" to the MAIL FROM SMTP command.

The requested information is sent as a separate message and is generated by each OPES system in the chain. The message MUST contain all message headers, including the trace headers. It MUST NOT contain any parts of the body of the original message.

[Comment to discuss on email list: Note that DSN (RFC1891) has not been chosen as a method for sending trace information back to the

sender as that would offer an attacker to also send the email content body with potential malicious content to a faked sender address. Another candidate to evaluate is Message Tracking (RFC3885); maybe this can be used rather than defining yet another method.]

5. Bypass

An OPES processor MUST support the SMTP Service Extension for OPES Trace and Bypass [to be done in external document; referenced from here] which allows for OPES system bypass as defined in [RFC3897].

As SMTP does not include client requests an OPES bypass for SMTP is triggered by asking the email message sender to initiate the bypass on behalf of the recipient.

[Extension be done in external document; referenced from here. Draft idea: New keyword to add to EHLO responses is "OPES", allows two new extensions for the MAIL FROM command argument: (1) "opestrace" to send trace information to the sender of the message and (2) "opesbypass=("*" | 1#bypass-entry)" for a list of OPES agent IDs to bypass.]

The decision, if a bypass request is fulfilled must be taken by the OPES processors in a chain and is their responsibility, because it is possible, that executing a bypass request results in an undelivarable message. If one OPES processor cannot fulfill the request, non-OPES content is not available.

Especially for client centric OPES adaptations, other bypass methods may be added that allow direct bypass requests from the recipients to the OPES systems; that needs to be implemented outside of the normal SMTP message flow (as SMTP traffic is not request/response) and is therefore out of the scope of this document.

[Comment to discuss on email list: The whole bypass idea is an issue for protocols that do not have client requests. Is a general out-of-band solution required?]

6. IAB Considerations

OPES treatment of IETF Internet Architecture Board (IAB) considerations [RFC3238] are documented in "OPES Treatment of IAB Considerations" [RFC3914].

Section 5.3 of [RFC3914] talks about trace information in application message requests and that "some application protocols may not have explicit requests". This is fact for SMTP and therefore the MUST requirement for a new SMTP extension has been added for OPES processors for SMTP (Section 4.2); non-OPES SMTP servers are also encouraged to implement that extension. If some SMTP relays on the way of the email from or to the OPES processors do not support the opestrace extension, the trace notifications may not be delivered. This lack of trace notifications is not a problem that has been introduced by OPES but a general SMTP issue. In such a case the email sender can still contact the recipients and ask to provide email headers of the message in question or similar messages in order to get trace information of the OPES systems involved.

OPES bypass will also fail if some SMTP relays on the way of the email do not supports the OPES SMTP extension. The sender will detect whether an SMTP server supports this extension by parsing the EHLO response. With information from the trace notification the sender can try to contact the first OPES processor which MUST support the bypass extension according to Section 5.

7. Security Considerations

Application-independent security considerations are documented in application-agnostic OPES specifications [RFC3837].

Requests to bypass OPES agents (Section 5) are sent by the email message sender on request of the recipient. A malicious sender could try to activate the bypass of OPES security services; it is important that implementations of the bypass feature include a policy that defines which OPES agents can be bypassed and which cannot.

8. Compliance

Compliance with OPES mechanisms is defined in corresponding application-agnostic specifications. SMTP profiles for these mechanisms use corresponding compliance definitions from these specifications, as if each profile was incorporated into the application-agnostic specification it profiles.

Appendix A. Acknowledgments

Appendix B. Change Log

Internal WG revision control ID: $Id: smtp.xml,v 1.6 2005/10/13 15:18:54 martin Exp $

2005/10/13

* Filled Header-List section.
* Added negotiation examples
* Filled Extension sections

2005/10/12

* Clemens' corrections and additions to Tracing and Bypass.
* Added hint to evaluate Message Tracking (RFC3885)

2005/10/11

* Removed DSN requirements, added new requirement for Trace/Bypass SMTP extension.

2005/10/10

* Tracing section and additional to IAB considerations.
* Corrections provided by Clemens.

2005/09/29

* Added DUM/DUY sections with parameters.
* Started SMTP Extension Handling section.
* Updated RFC numbers in document map and References section.

2005/09/23

* Initial revision.

9. References

9.1 Normative References

[RFC2821] Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April 2001.

[RFC2822] Resnick, P., "Internet Message Format", RFC 2822, April 2001.

[RFC3897] Barbir, A., "Open Pluggable Edge Services (OPES) Entities and End Points Communication", RFC 3897, September 2004.

[RFC4037] Rousskov, A., "Open Pluggable Edge Services (OPES) Callout Protocol (OCP) Core", RFC 4037, March 2005.

9.2 Informative References

[RFC2396] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[RFC3238] Floyd, S. and L. Daigle, "IAB Architectural and Policy Considerations for Open Pluggable Edge Services", RFC 3238, January 2002.

[RFC3752] 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.

[RFC3835] Barbir, A., Penno, R., Chen, R., Hofmann, M., and H. Orman, "An Architecture for Open Pluggable Edge Services (OPES)", RFC 3835, August 2004.

[RFC3836] Beck, A., Hofmann, M., Orman, H., Penno, R., and A. Terzis, "Requirements for Open Pluggable Edge Services (OPES) Callout Protocols", RFC 3836, August 2004.

[RFC3837] 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.

[RFC3838] 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.

[RFC3914] Barbir, A. and A. Rousskov, "Open Pluggable Edge Services

(OPES) Treatment of IAB Considerations", RFC 3914, October 2004.

[I-D.ietf-opes-smtp-use-cases]
Barbir, A. and M. Stecher, "OPES SMTP Use Cases", draft-ietf-opes-smtp-use-cases-03 (work in progress), July 2005.

[I-D.ietf-opes-rules-p]
Rousskov, A., "P: Message Processing Language", draft-ietf-opes-rules-p-02 (work in progress), October 2003.

Authors' Addresses

Martin Stecher
CyberGuard Corporation
Webwasher Division
Vattmannstr. 3
33100 Paderborn
Germany

Email: martin.stecher@webwasher.com
URI: http://www.cyberguard.com/

Clemens Perz
All About It Systems S.A.
16, rue du Parc
L-6684 Mertert
Luxembourg

Email: cperz@allaboutit.lu
URI: http://www.allaboutit.lu/

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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