Optimizing the Email Gateway for Outbound Mail Delivery Using D-Mode

This chapter contains the following sections:

Feature Summary: D-Mode for Optimized Outbound Delivery

D-Mode is a feature key-enabled feature that optimizes certain email gateways for outbound email delivery. Features specific to inbound mail handling are disabled in D-Mode.

Features Unique to D-Mode-Enabled Email Gateways

- 256 Virtual Gateway Addresses - The Cisco Virtual Gateway technology allows you to configure enterprise mail gateways for all domains you host — with distinct IP addresses, hostname and domains — and create separate corporate email policy enforcement and anti-spam strategies for those domains, while hosted within the same physical email gateway. See information about “Customizing Listeners” in Configuring the Gateway to Receive Email
- IronPort Mail Merge (IPMM) - IronPort Mail Merge (IPMM) removes the burden of generating individual personalized messages from customer systems. By removing the need to generate thousands of individual messages and transmit them between message generating systems and the email gateway, users benefit from the decreased load on their systems and increased throughput of email delivery. For more information, see Sending Bulk Mail Using IronPort Mail Merge (IPMM), on page 4.
- Resource-conserving bounce setting - You can configure D-Mode-enabled email gateways to detect potential blocked destinations and bounce all messages bound for that destination. For more information, see Configuring Resource-Conserving Bounce Settings, on page 3.
- Enhanced performance for outbound delivery

Standard Features Disabled in D-Mode-Enabled Email Gateways

- IronPort anti-spam scanning and on or off box spam quarantining — Because anti-spam scanning pertains mostly to incoming mail, the IronPort Anti-Spam scanning engine is disabled. The Anti-Spam chapter is, therefore, not applicable.
- Outbreak Filters — Because the Outbreak Filters feature is used to quarantine incoming mail, this feature is disabled on D-Mode-enabled email gateways. Information in the Outbreak Filters chapter is, therefore, not applicable.
- Service Logs - The Service Logs feature is disabled on D-mode enabled email gateways.
- Reporting — Reporting is limited. Some reports are not available, and the reporting that does occur is set to run at a very limited level for performance reasons.

Note The totals shown in the Email Security Monitor Overview report for D-Mode-enabled email gateways may erroneously include spam and suspect spam counts, even though these features are disabled on D-Mode-enabled email gateways.

- Data Loss Prevention — DLP scanning for outgoing messages is disabled on D-Mode-enabled email gateways.

Standard Features Applicable to D-Mode-Enabled Email Gateways

Table 1: AsyncOS Features Included in D-Mode Enabled Email Gateways

Feature	More Information
Anti-virus scanning	See Anti-Virus
Domain Key signing	DKIM/Domain Keys is a method for verifying authenticity of email based on a signing key used by the sender. See Email Authentication
Centralized management	See Centralized Management Using Clusters
Delivery throttling	For each domain, you can assign a maximum number of connections and recipients that will never be exceeded by the system in a given time period. This "good neighbor" table is defined through the destconfig command. For more information, see Controlling Email Delivery Using Destination Controls.
Bounce Verification	Verify the authenticity of bounce messages. See Bounce Verification.
Delegated administration	See Distributing Administrative Tasks
Trace (debug)	See Debugging Mail Flow Using Test Messages: Trace.
VLAN, NIC-pairing	See Advanced Network Configuration
Optional Anti-virus engine	You can add optional anti-virus scanning to ensure the integrity of your outbound messages. See Anti-Virus Scanning Overview.

Setting Up the Email Gateway for Optimized Outbound Mail Delivery

Procedure

Step 1 Apply the provided feature key. You will need to apply the key to your email gateway *prior to running the system setup wizard* (prior to configuring the email gateway). Apply the key via the System Administration > Feature Key page or by issuing the featurekey command in the CLI.

Note The preceding feature keys include a sample 30 day Sophos or McAfee Anti-Virus license you can use to test anti-virus scanning on outbound mail.

Step 2 Reboot the email gateway.

Step 3 Run the system setup wizard (GUI or CLI) and configure your email gateway.

Keep in mind that email gateways that are optimized for outbound delivery do not include anti-spam scanning or the Outbreak Filters feature. (Please ignore these chapters.)

Note In a clustered environment, you cannot combine email gateways that are configured with the D-Mode feature key with AsyncOS email gateways that are not configured with the delivery performance package.

Configuring Resource-Conserving Bounce Settings

Once the email gateway is configured for optimized outbound mail delivery, you can configure the system to detect potential delivery problems and bounce all messages for a destination.

Note Using this setting will bounce all messages in the queue for a destination domain that is deemed undeliverable. You will need to re-send the message once the delivery issues have been resolved.

Example of Enabling Resource-Conserving Bounce Settings

```
mail3.example.com> bounceconfig

Choose the operation you want to perform:

- NEW - Create a new profile.

- EDIT - Modify a profile.

- DELETE - Remove a profile.

- SETUP - Configure global bounce settings.

[]> setup
```

```
Do you want to bounce all enqueued messages bound for a domain if the host is down? [N]>

y
```

When using this feature, a host is considered “down” after at least 10 consecutive connection attempts fail. AsyncOS scans for down hosts every 15 minutes, so it is possible that more than 10 attempts will be made before the queue is cleared.

Sending Bulk Mail Using IronPort Mail Merge (IPMM)

Note IronPort Mail Merge is available only on email gateways that are D-Mode-enabled.

Overview of IronPort Mail Merge

IronPort Mail Merge removes the burden of generating individual personalized messages from customer systems. It removes the need to generate thousands of individual messages and transmit them between message generating systems and the email gateway, resulting in decreased load on your systems and increased throughput of email delivery.

With IPMM, a single message body is created with variables representing locations in the message to be replaced for personalization. For each individual message recipient, only the recipient email address and the variable substitutions need to be transmitted to the email gateway. In addition, IPMM can be used to send certain recipients specific “parts” of the message body, while excluding certain parts from others recipients. (For example, suppose you needed to include a different copyright statements at the end of your messages to recipients in two different countries.)

Benefits of the Mail Merge Function

- Ease of use for the mail administrator. The complexities of creating personalized messages for each recipient are removed, as IPMM provides variable substitution and an abstracted interface in many common languages.
- Reduced load on message generation systems. By requiring one copy of the message body and a table of required substitutions, most of the message generation “work” is off-loaded from message generation systems and moved to the email gateway that is configured for optimized outbound mail delivery.
- Increased delivery throughput. By reducing the resources necessary to accept and queue thousands of incoming messages, the email gateway can significantly increase out-bound delivery performance.
- Queue storage efficiency. By storing less information for each message recipient, users can achieve orders-of- magnitude, better use of queue storage on the D-Mode enabled email gateway.

Using Mail Merge

SMTP Injection

IPMM extends SMTP as the transport protocol. There is no special configuration that needs to be made to the email gateway. (By default, IPMM can be enabled for private listeners and disabled for public listeners on the D-Mode-enabled email gateway.) However, if you are not currently using SMTP as your injection protocol, you must create a new private listener that utilizes SMTP through the D-Mode enabled email gateway interface.

Use the setipmm subcommand of listenerconfig to enable IPMM on the listener. For more information, see Configuring the Gateway to Receive Email

IPMM modifies SMTP by altering two commands — MAIL FROM and DATA — and adding another: XDFN . The MAIL FROM command is replaced with XMRG FROM and, the DATA command is replaced with XPRT .

To generate a Mail Merge message, the commands used to generate the message need to be issued in a particular sequence.

1. The initial EHLO statement, identifying the sending host.
2. Each message starts with an XMRG FROM: statement, indicating the sender address.
3. Each recipient is then defined:
4. One or more XDFN variable allocation statements are made, including defining the parts (XDFN *PART=1,2,3...), and any other recipient specific variables.
5. The recipient email address is defined with the RCPT TO: statement. Any variable allocations prior to the RCPT TO:, but after the prior XMRG FROM, or RCPT TO command, will be mapped to this recipient email address.
6. Each part is defined using the XPRT n command, with each part terminated by a period (.) character similar to the DATA command. The last part is defined by the XPRT n LAST command.

Variable Substitution

Any part of the message body, including message headers, can contain variables for substitution. Variables can appear in HTML messages, as well. Variables are user-defined and must begin with the ampersand (&) character and end with the semi-colon character (;). Variable names beginning with an asterisk (*) are reserved and cannot be used.

Reserved Variables

IPMM contains five special “reserved” variables that are predefined.

Table 2: IPMM: Reserved Variables

`*FROM`	The reserved variable *FROM is derived from the “Envelope From” parameter. The “Envelope From” parameter is set by the “XMRG FROM:” command.
`*TO`	The reserved variable *TO is derived from the envelope recipient value, as set by the “RCPT TO:” command.
`*PARTS`	The reserved variable *PARTS holds a comma separated list of parts. It is set prior to defining a recipient with the “RCPT TO:” and determines which of the “XPRT n” message body blocks a given user will receive.

*DATE	The reserved variable *DATE is replaced with the current date stamp.
*DK	The reserved variable *DK is used to specify a DomainKeys Signing profile (this profile must already exist in AsyncOS). For more information about creating DomainKeys Signing profiles, see Email Authentication

Example Message 1

The following example message body (including headers) contains four distinct variables and five substitution locations that will be replaced in the final message. Note that the same variable may be used more than once in the message body. Also, the reserved variable &*TO; is used, which will be replaced with the recipient email address. This reserved variable does not need to be passed in as a separate variable. The variables in the example appear in bold.

```
From: Mr.Spacely <spacely@example.com>

To: &first_name;&last_name;&*TO;


Subject: Thanks for Being an Example.Com Customer

Dear &first_name;,
Thank you for purchasing a &color; sprocket.
```

This message needs only be injected once into the email gateway. For each recipient, the following additional information is required:

- A recipient email address
- Name-value pairs for the variable substitution

Part Assembly

Where SMTP uses a single DATA command for each message body, IPMM uses one or many XPRT commands to comprise a message. Parts are assembled based upon the order specified per-recipient. Each recipient can receive any or all of the message parts. Parts can be assembled in any order.

The special variable *PARTS holds a comma separated list of parts.

For example, the following example message contains two parts.

The first part contains the message headers and some of the message body. The second part contains an offer that can be variably included for specific customers.

Example Message 2, Part 1

```
From: Mr. Spacely <spacely@example.com>


To: &first_name; &last_name; &*TO;

Subject: Thanks for Being an Example.Com Customer

Dear &first_name;,

Thank you for purchasing a &color; sprocket.
```

Example Message 2, Part 2

```
Please accept our offer for 10% off your next sprocket purchase.
```

The message parts need only be injected once into the email gateway. In this case, each recipient requires the following additional information:

- The ordered list of parts to be included in the final message
- A recipient email address
- Name value pairs for the variable substitution

IPMM and DomainKeys Signing

IPMM does support DomainKeys Signing. Use the *DK reserved variable to specify a DomainKeys profile. For example:

```
XDFN first_name="Jane" last_name="User" color="red" *PARTS=1,2 *DK=mass_mailing_1
```

In this example, “mail_mailing_1” is the name of a previously configured DomainKeys profile.

Command Descriptions

When a client injects IPMM messages to the listener, it uses extended SMTP with the following key commands.

XMRG FROM

Syntax:

```
XMRG FROM: <sender email address>
```

This command replaces the SMTP MAIL FROM: command and indicates that what follows is an IPMM message. An IPMM job is initiated with the XMRG FROM: command.

XDFN

Syntax:

```
XDFN <KEY=VALUE> [KEY=VALUE]
```

The XDFN command sets the per-recipient metadata. Note that key-value pairs can optionally be enclosed in angle brackets or square brackets.

*PARTS is a special reserved variable that indicates the index number as defined by the XPRT command (described below). The *PARTS variable is split as a comma-delimited list of integers. The integers match the body parts to be sent as defined by the XPRT commands. The other reserved variables are: *FROM , *TO , and *DATE .

XPRT

Syntax:

```
XPRT index_number LAST

Message

.
```

The XPRT command replaces the SMTP DATA command. The command accepts the transfer of the message part after the command is issued. The command is completed with a single period on a line followed by a return (which is the same way an SMTP DATA command is completed).

The special keyword **LAST** indicates the end of the mail merge job and must be used to specify the final part that will be injected.

After the LAST keyword is used, the message is queued, and delivery begins.

Notes on Defining Variables

- When you define variables with the XDFN command, note that the actual command line cannot exceed the physical limit of the system. In the case of the D-Mode-enabled email gateway, this limit is 4 kilobytes per line. Other host systems may have lower thresholds. Use caution when defining multiple variables on very large lines.
- You can escape special characters using the forward slash " / " character when defining variables key-value pairs. This is useful if your message body contains HTML character entities that might be mistakenly replaced with variable definitions. (For example, the character entity ™ defines the HTML character entity for a trademark character. If you created the command XDFN trade=foo and then created a IPMM message containing the HTML character entity " ™ " the assembled message would contain the variable substitution (" foo ") instead of the trademark character. The same concept is true for the ampersand character "&" which is sometimes used in URLs containing GET commands.

Example IPMM Conversation

The following is an example IPMM conversation of Example Message #2 (shown above). The message will be sent to two recipients in this example: "Jane User" and "Joe User."

In this example, the type in **bold** represents what you would type in a manual SMTP conversation with the D-Mode-enabled email gateway, type in `monospaced type` represents the responses from the SMTP server, and *italic type* represents comments or variables.

A connection is established:

```
220 ESMTP


EHLO foo


250 - ehlo responses from the listener enabled for IPMM
```

The conversation is started:

```
XMRG FROM:<user@domain.com> [Note: This replaces the MAIL FROM: SMTP command.]

250 OK
```

Variables and parts are set for each recipient:

```
XDFN first_name="Jane" last_name="User" color="red" *PARTS=1,2

[Note: This line defines three variables (first_name, last_name, and color) and then
uses the *PARTS reserved variable to define that the next recipient defined will receive
message parts numbers 1 and 2.]
```

```
250 OK

RCPT TO:<jane@company.com>

250 recipient <jane@company.com> ok

XDFN first_name="Joe" last_name="User" color="black" *PARTS=1

[Note: This line defines three variables (first_name, last_name, and color) and then
uses the *PARTS reserved variable to define that the next recipient defined will receive
message parts numbers 1 only.]

RCPT TO:<joe@company1.com>

250 recipient <joe@company1.com> ok
```

Next, part 1 is transmitted:

```
XPRT 1 [Note: This replaces the DATA SMTP command.]

354 OK, send part

From: Mr. Spacely <spacely@example.com>

To: &first_name; &last_name; &*TO;

Subject: Thanks for Being an Example.Com Customer

&*DATE;

Dear &first_name;,

Thank you for purchasing a &color; sprocket.

.
```

And then part 2 is transmitted. Note that the LAST keyword is used to identify Part 2 as the final part to assemble:

```
XPRT 2 LAST

Please accept our offer for 10% off your next sprocket purchase.

.

250 Ok, mailmerge message enqueued
```

The "`250 Ok, mailmerge message queued`" notes that the message has been accepted.

Based on this example, recipient Jane User will receive this message:

```
From: Mr. Spacely <spacely@example.com>


To: Jane User <jane@company.com>


Subject: Thanks for Being an Example.Com Customer


message date
```

```
Dear Jane,


Thank you for purchasing a red sprocket.

Please accept our offer for 10% off your next sprocket purchase.
```

Recipient Joe User will receive this message:

```
From: Mr. Spacely <spacely@example.com>

To: Joe User <joe@company1.com>

Subject: Thanks for Being an Example.Com Customer

message date

Dear Joe,

Thank you for purchasing a black sprocket.
```

Example Code

Cisco has created libraries in common programming languages to abstract the task of injecting IPMM messages into the email gateway listener enabled for IPMM. Contact Cisco Customer Support for examples of how to use the IPMM library. The code is commented extensively to explain its syntax.