

Scaling DBMail with MySQL

Guido A.J. Stevens
© 2007 NFG Net Facilities Group
All rights reserved

<http://www.dbmail.eu>
sales@dbmail.eu
T +31.43.3618933
F +31.43.3561655

Scaling DBMail

DBMail is a fast, scalable, database-powered email application. NFG Net Facilities Group leads DBMail development and provides consultancy and support services for DBMail¹.

DBMail currently supports three SQL back ends: SQLite, PostgreSQL and MySQL. All of these pose limits to the scalability of a DBMail system, because none of these provides true database clustering capabilities as of yet.

NFG has done some preliminary work on an Ingres driver. Ingres² is a GPL-licensed database system that provides full clustering capabilities. No DBMail-Ingres driver has been released yet.

Pending support for clustering databases, one has to work around the limitations of the currently available database systems for DBMail. This paper details a design based on MySQL.

Disclaimer

NFG does not warrant or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, design, product, or process disclosed. The authors assume no responsibility whatsoever for any damage to hardware and/or software, or any loss of data resulting from the procedures outlined in this document.

Scaling DBMail with MySQL

MySQL is rather unique in that it supports different storage engines³. The capabilities and limitations of the specific storage engine chosen greatly impact the capabilities and limitations of MySQL in a concrete deployment.

Limitations

The MySQL server system offers asynchronous replication⁴ (multiple storage engines), synchronous replication in the form of MySQL Cluster⁵ (NDB storage engine only), partitioning⁶ (rather crippled⁷) and federated storage⁸.

Asynchronous replication is not truly multi-master. It is possible to create a highly available MySQL cluster using crossed asynchronous replication but one should not load balance write actions to both nodes. That is, only one node at any given time should be the 'true' master⁹. Thus: HA is OK, load balancing is not. This limits scalability to the write load a single node can handle.

Synchronous replication using MySQL cluster relies on the NDB storage engine. This is a in-memory database engine, limiting the maximum size of the total database to the total available RAM size. MySQL Cluster is actively developed and shows some progress towards a disk-storage solution¹⁰, but is not very mature¹¹ yet.

Partitioning and federated storage as implemented provide some additional resource allocation capabilities for low-end installations, but do not contribute towards realizing highly available, highly scalable environments.

Workaround

MySQL effectively scales to a single node. One can work around this limitation by creating multiple DBMail "islands".

An island is a combination of a storage back end cluster with a DBMail front end cluster. Users are separated into several groups; any given user belongs to a specific island and will only be served from that island. LDAP user attributes are used to configure, which island serves which user.

SMTP is configured to perform a lookup on the addressee to determine, which island handles this specific user's email. The MTA then relays the email through LMTP to the appropriate DBMail cluster, which performs the actual database injection on the Storage back end of that island.

POP and IMAP clients connect to a highly available proxy server, that defers the actual handling of the connection to the DBMail POP or IMAP server of the designated island for this specific user. Web mail uses the same proxy server.

1 <http://www.dbmail.eu>

2 <http://www.ingres.com/>

3 <http://dev.mysql.com/doc/refman/5.1/en/storage-engine-overview.html>

4 <http://dev.mysql.com/doc/refman/5.1/en/replication.html>

5 <http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster.html>

6 <http://dev.mysql.com/doc/refman/5.1/en/partitioning-overview.html>

7 <http://dev.mysql.com/doc/refman/5.1/en/partitioning-limitations.html>

8 <http://dev.mysql.com/doc/refman/5.1/en/federated-storage-engine.html>

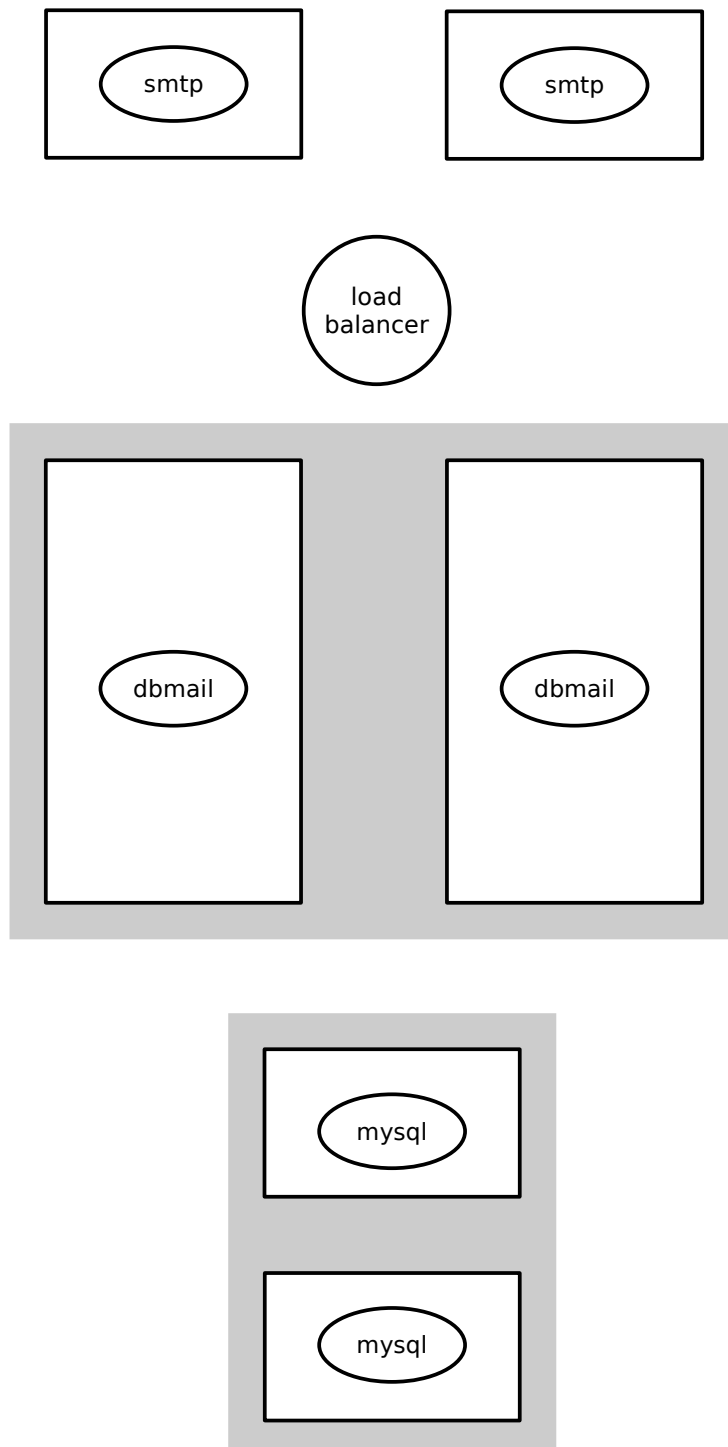
9 <http://dev.mysql.com/doc/refman/5.1/en/replication-faq.html#qandaitem-7-3-4-5>

10 <http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster-disk-data.html>

11 <http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster-limitations.html>

Basic HA architecture

- 6 servers
- 6 services
- 1 dbmail island



1. Basic HA architecture

SMTP is served by 2 nodes `smtp-left` and `smtp-right`. These nodes need not be linked in a HA cluster. The load balancer balances incoming IMAP and POP3 connections towards the DBMail cluster. HA can be achieved without load balancer.

DBMail IMAP, POP3 and LMTP are served by two nodes `dbmail-left` and `dbmail-right`, linked in a HA cluster. The HA cluster is configured such, that each node will take over the ip number of it's peer in case the peer goes down.

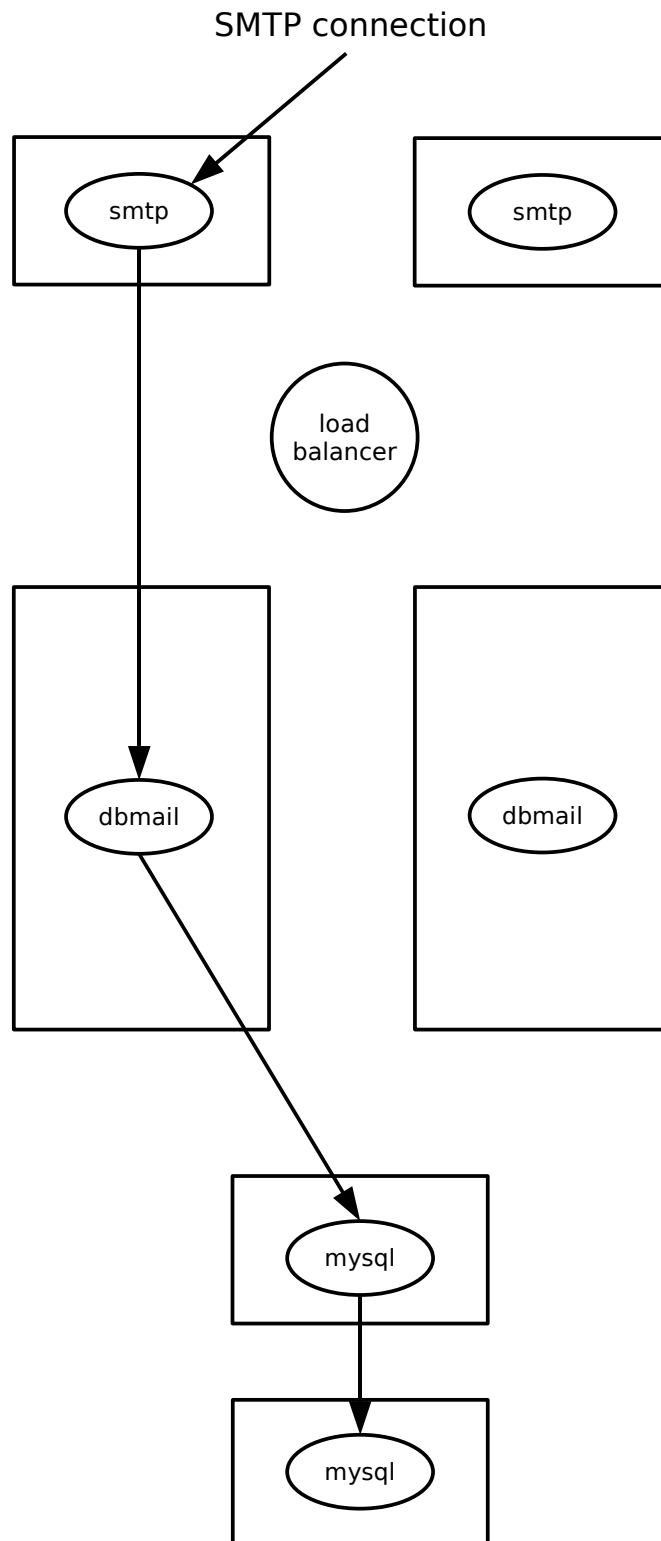
MySQL is served by two nodes `mysql-front` and `mysql-back`, linked in a HA cluster. This HA cluster is configured in another way. Only one 'active' node serves mysql requests, to prevent transaction collisions: this is the `mysql-active` ip number. DBMail connects to `mysql-active` only, not to `mysql-front` or `mysql-back`.

MySQL replication: `mysql-front` (master) replicates to `mysql-back` (slave). `mysql-back` (master) replicates to `mysql-front` (slave). Usually, `mysql-front` serves as `mysql-active`. In this default case only the front to back replication channel is actively used.

If `mysql-front` goes down, `mysql-back` takes over the `mysql-active` ip number. If `mysql-front` comes back online, it does **not** immediately take over the `mysql-active` ip number. Rather, it catches up on the replication from `mysql-back`. In this case only the back to front channel is actively used.

Basic HA architecture - SMTP

- 6 servers
- 6 services
- 1 dbmail island



2. Basic HA architecture: SMTP (left)

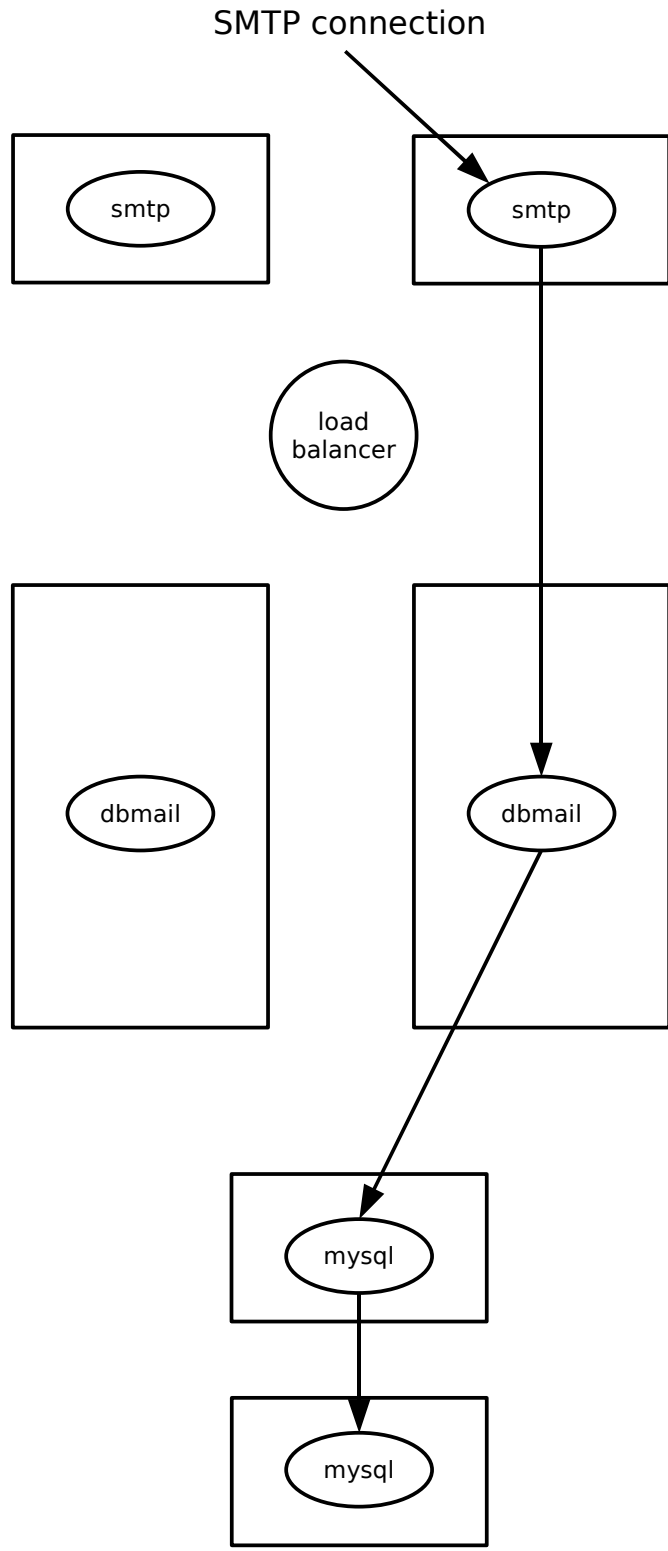
Both HA and load balancing are achieved by pointing DNS MX records for all domains to both nodes.

```
clientdomain.com IN MX 10 smtp-left.bigisp.net  
clientdomain.com IN MX 10 smtp-right.bigisp.net
```

An incoming SMTP relay may select `smtp-left` to deliver a message. The SMTP server relays the message to `dbmail-left` via LMTP. DBMail performs the database injection against the active MySQL node.

Basic HA architecture - SMTP

- 6 servers
- 6 services
- 1 dbmail island

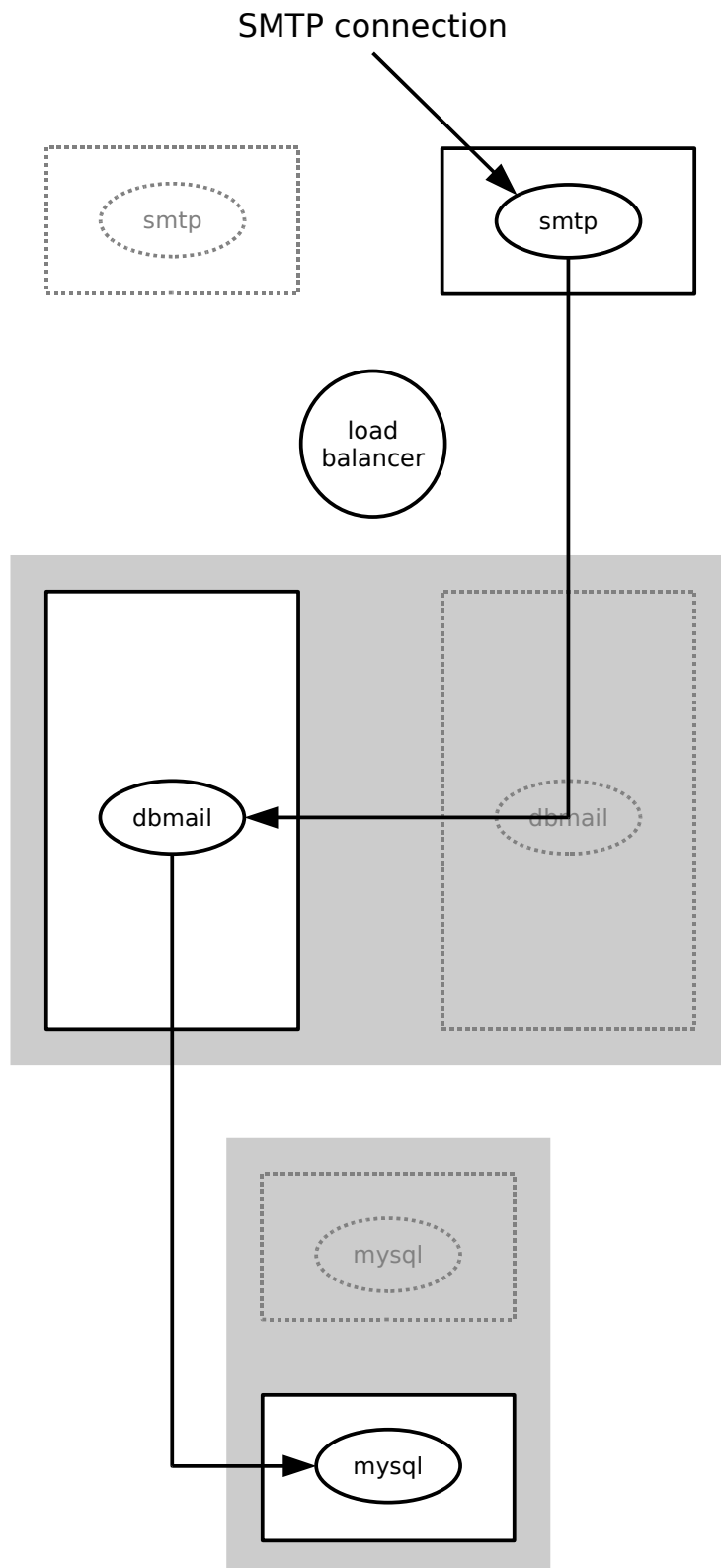


3. Basic HA architecture: SMTP (right)

An incoming SMTP relay may choose `smtp-right` to deliver a message. With equal MX priorities, the chances of an incoming relay choosing either `smtp-left` or `smtp-right` are equal.

Basic HA architecture – 3x degraded mode: SMTP

- 6 servers
- 6 services
- 1 dbmail island



4. Basic HA architecture – 3x degraded mode: SMTP

Figure 4 shows the worst-case scenario that can be tolerated without service disruption.

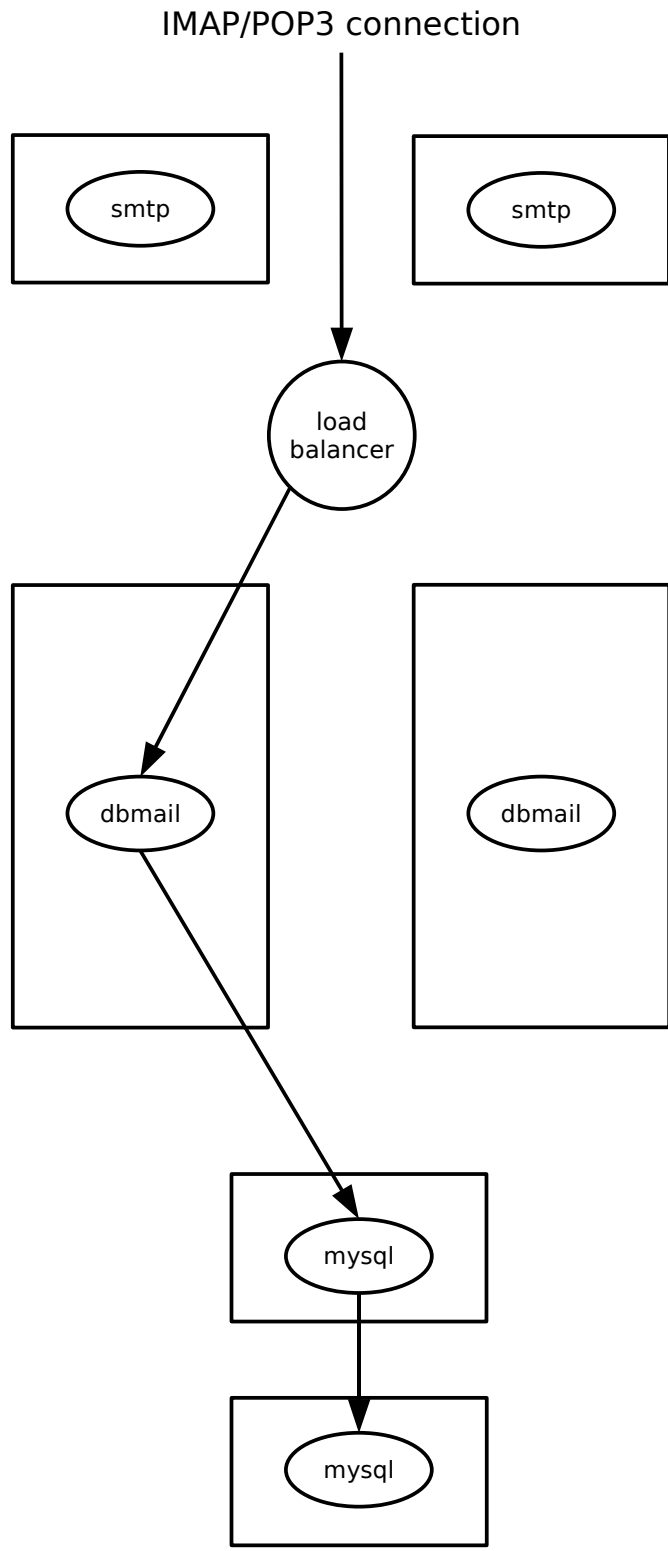
smtp-left is down. Incoming SMTP relays that initially choose smtp-left cannot deliver there and will subsequently deliver at smtp-right.

smtp-right usually delivers at dbmail-right. Though dbmail-right is down, smtp-right doesn't know this, because dbmail-left has taken over the ip number of dbmail-right. smtp-right happily delivers the message to the dbmail-left service listening on the ip number of dbmail-right.

dbmail-left inserts the message into the database on the active mysql node mysql-active. It doesn't know or care that mysql-front is down: mysql-back has taken over the mysql-active ip number (from mysql-front).

Basic HA architecture: IMAP/POP3

- 6 servers
- 6 services
- 1 dbmail island

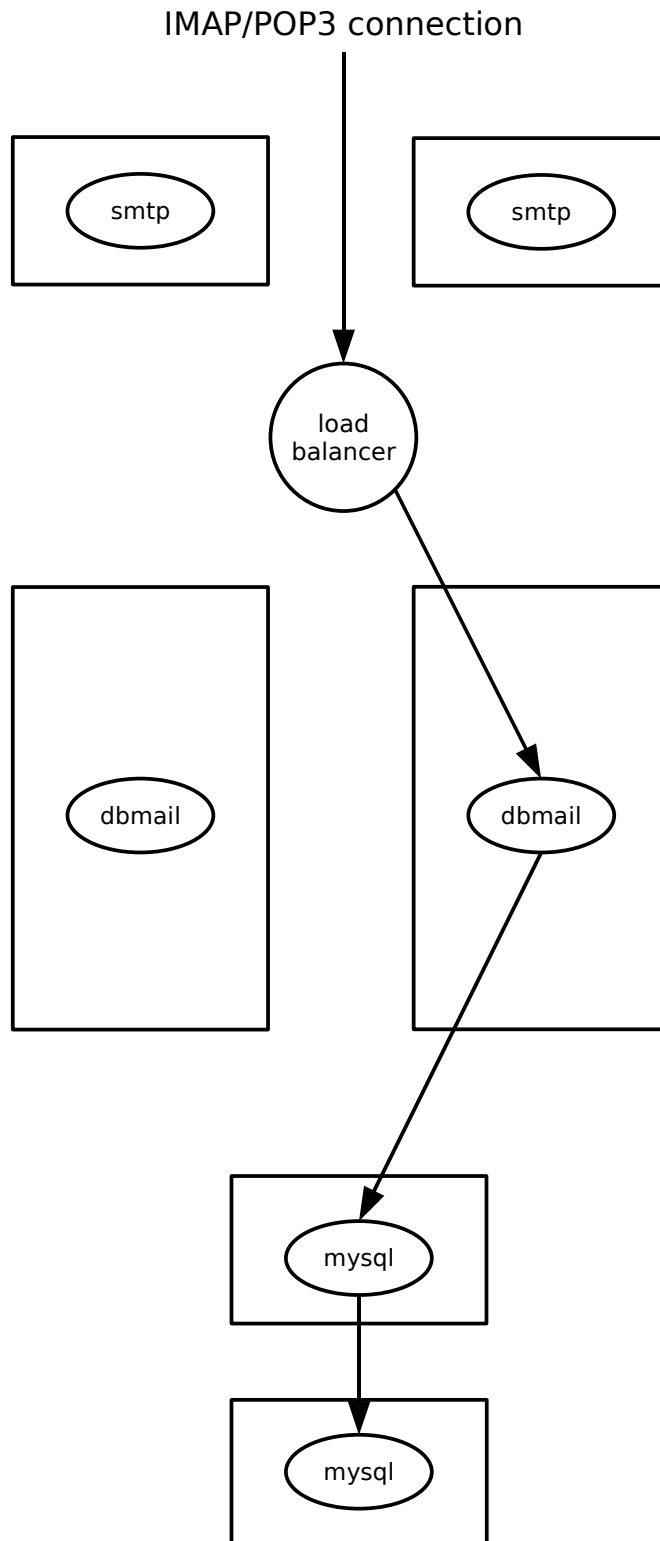


5. Basic HA architecture: IMAP/POP3 (left)

IMAP and POP3 connections are load balanced. In figure 5, the connection is directed to `dbmail-left`. `dbmail-left` interacts with `mysql-active`, which in non-degraded mode is served by `mysql-front`. Any state modifications are replicated to `mysql-back`.

Basic HA architecture: IMAP/POP3

- 6 servers
- 6 services
- 1 dbmail island

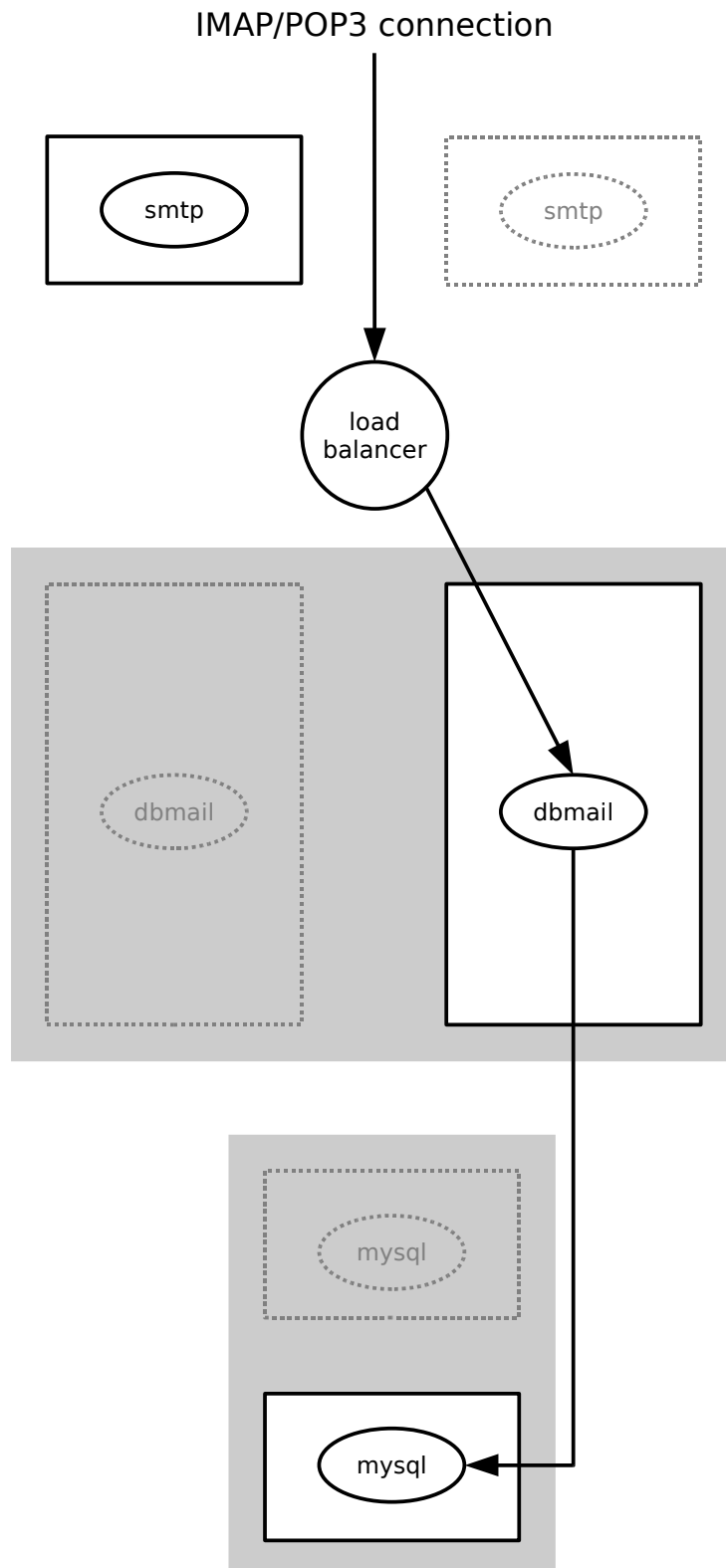


6. Basic HA architecture: IMAP/POP3 (right)

The load balancer can also direct an incoming IMAP/POP3 connection to `dbmail-right`.

Basic HA architecture – 3x degraded mode: IMAP/POP3

6 servers
6 services
1 dbmail island



7. Basic HA architecture – 3x degraded mode: IMAP/POP3

Figure 7 again shows serious degradation with 3 nodes down.

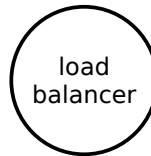
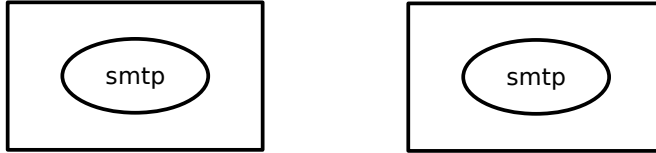
The load balancer directs all IMAP and POP3 connections to `dbmail-right`.

`dbmail-right` connects to `mysql-active` as described above.

Not depicted here is failover without load balancer. The load balancer is not required to achieve high availability. Analogous to the mysql setup, DBMail can be served from a virtual `dbmail-active` ip number which is either served by `dbmail-left` or `dbmail-right` and which is taken over in case of outage. This alternative configuration preserves high availability but is not load-balanced for the DBMail services.

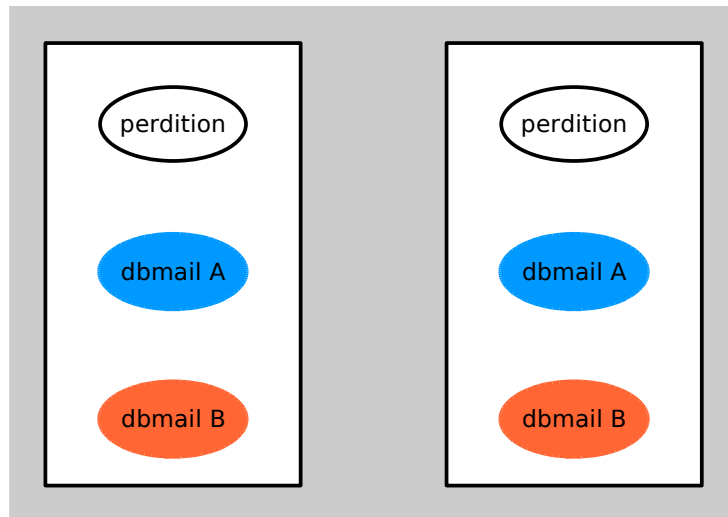
7 servers
12 services
2 dbmail islands

island-aware
smtp server

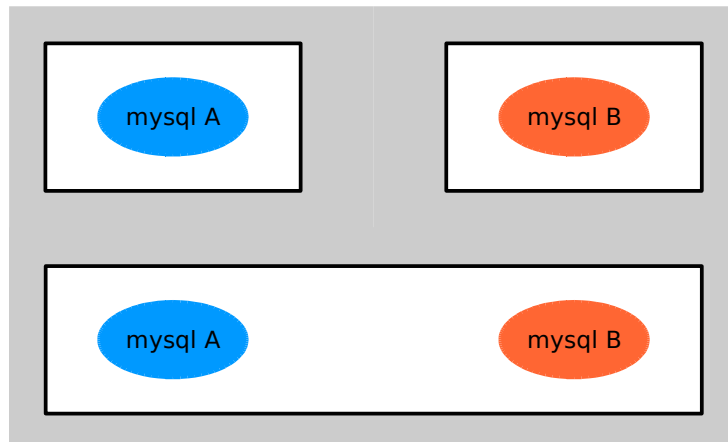


island-aware
IMAP/POP3 proxy

2x DBMail:
island A, island B



2x mysql active node



1x mysql passive node
2x mysql daemons: A, B

8. Upscaled HA architecture

Figure 8 introduces extra complexity to enable scalability.

MySQL is the scalability bottleneck. One extra server `mysql-front-b` is installed to create additional database serving capacity. All other reconfigurations are needed to accommodate this extra server.

Let's call the original mysql database `mysql-a`. An additional `mysql-b` configuration is installed on both `mysql-front-b` and `mysql-back`. This configuration has a separate database, a separate TCP port, etc. `mysql-back` now runs two database installations in parallel: it runs `mysql-a` slaved from `mysql-front-a` and `mysql-b` slaved from `mysql-front-b`.

Also, to prepare for recovery after outages, return replication channels are configured that replicate `mysql-back-a` (master) to `mysql-front-a` (slave) and `mysql-back-b` (master) to `mysql-front-b` (slave). These channels are latent and not actively used in a non-degraded configuration, as described above.

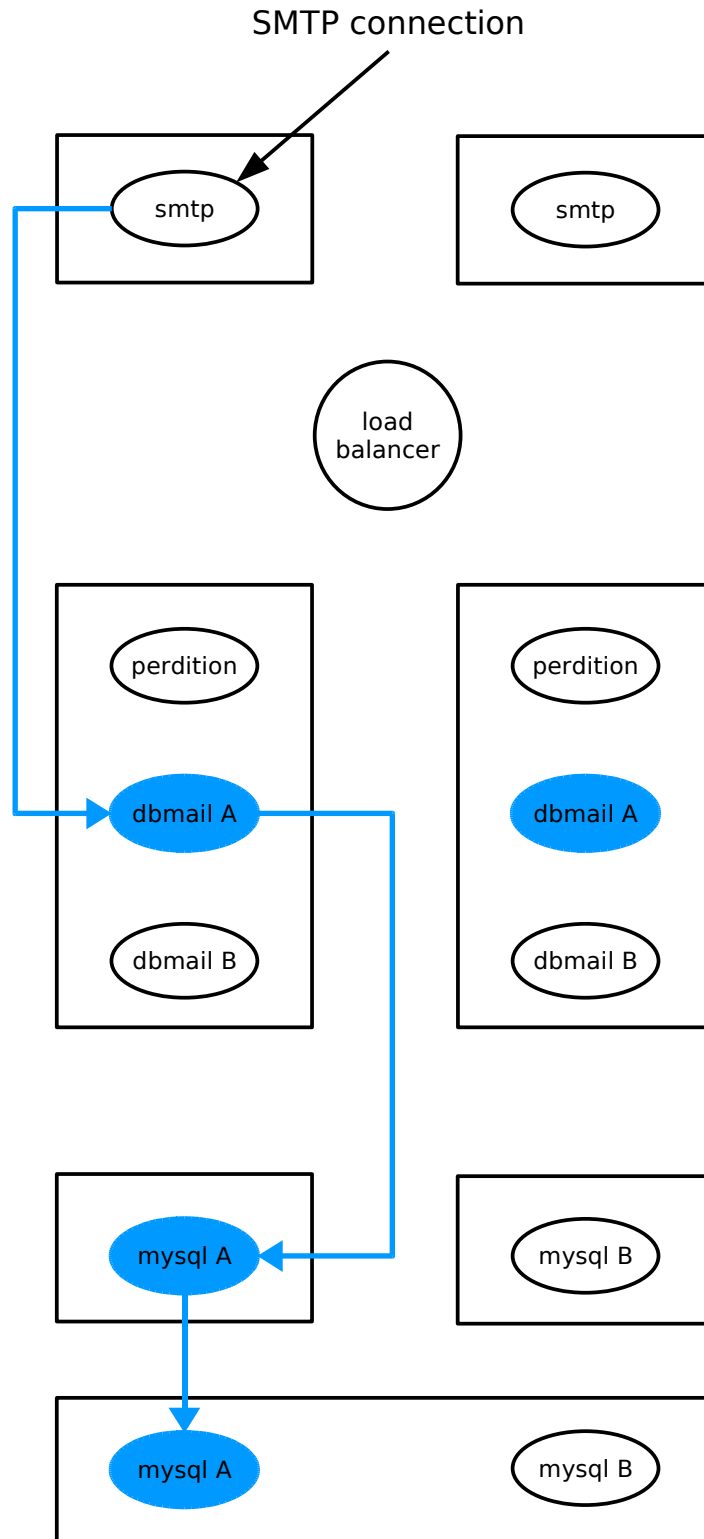
DBMail is not capable of connecting to multiple database backends. DBMail is split into two separate configurations `dbmail-a` (using `mysql-active-a`) and `dbmail-b` (using `mysql-active-b`). These two `dbmail` configurations can live side by side on the same platform, just like `mysql-back-a` and `mysql-back-b` coexist on the same server `mysql-back`.

To preserve high availability, `dbmail-a` is served as `dbmail-left-a` and `dbmail-right-a`, while `dbmail-b` is served as `dbmail-left-b` and `dbmail-right-b`.

Users are configured as belonging to either the `dbmail-a` island or the `dbmail-b` island through an LDAP attribute.

Upscaled HA architecture: SMTP

7 servers
12 services
2 dbmail islands



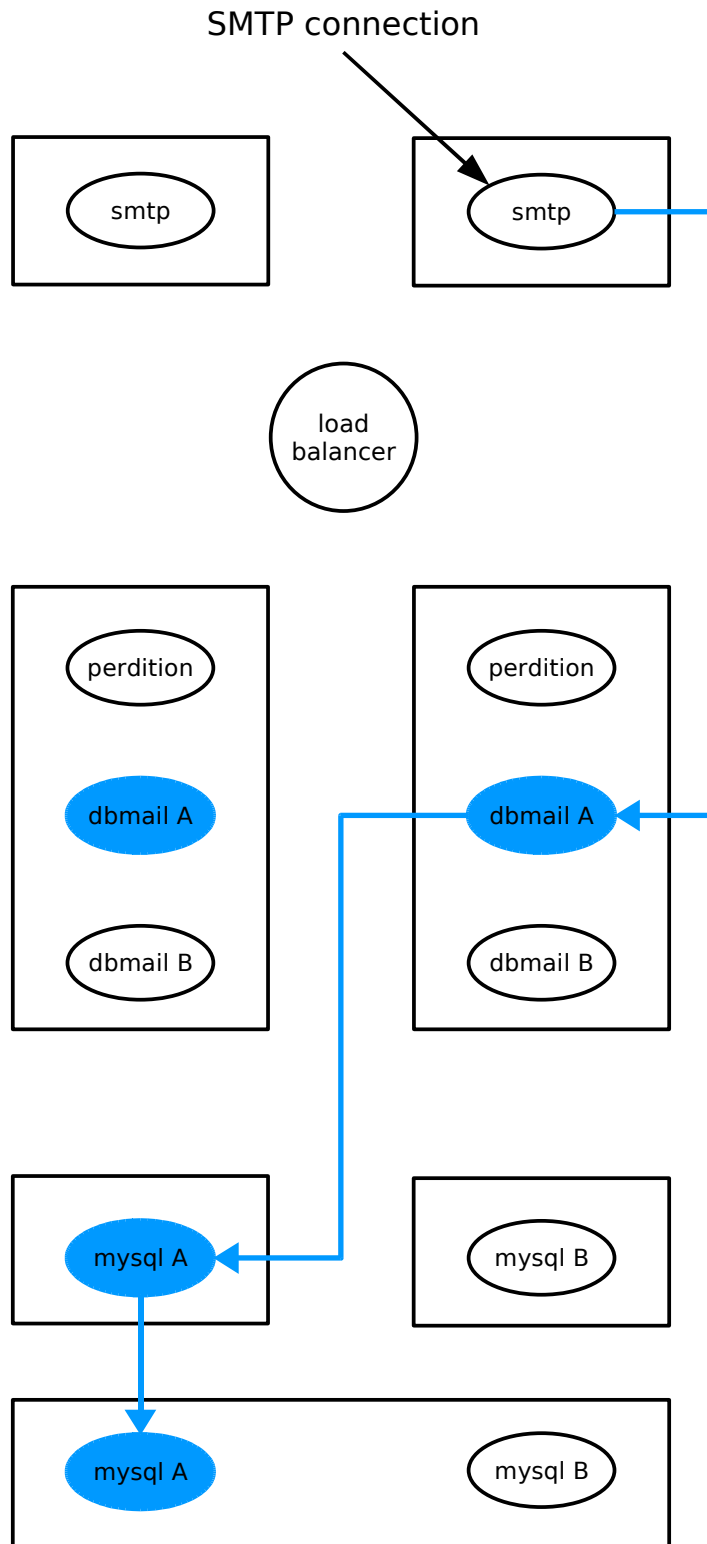
9. Upscaled HA architecture: SMTP (left A)

Postfix is able to perform an LDAP lookup to determine whether an incoming message should be delivered to either dbmail island A or to dbmail island B.

In figure 9, the incoming relay chooses `smtp-left`, which determines that the To: address belongs to island A and delivers at `dbmail-left-a`, which inserts into `mysql-active-a`, which replicates to `mysql-back-a`.

Upscaled HA architecture: SMTP

7 servers
12 services
2 dbmail islands

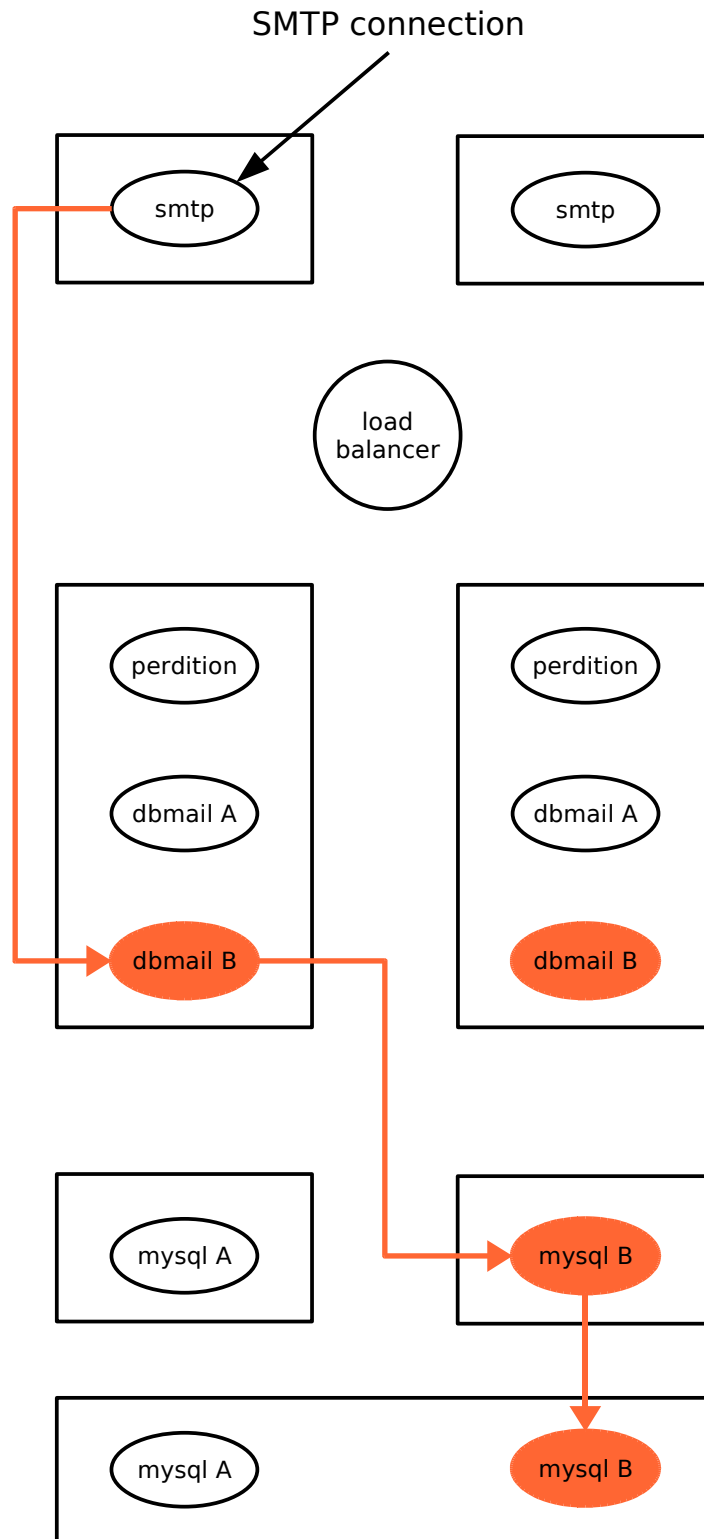


10. Upscaled HA architecture: SMTP (right A)

smtp-right delivers messages belonging to island A at dbmail-right-a, which inserts into mysql-active-a, which replicates to mysql-back-a.

Upscaled HA architecture: SMTP

7 servers
12 services
2 dbmail islands

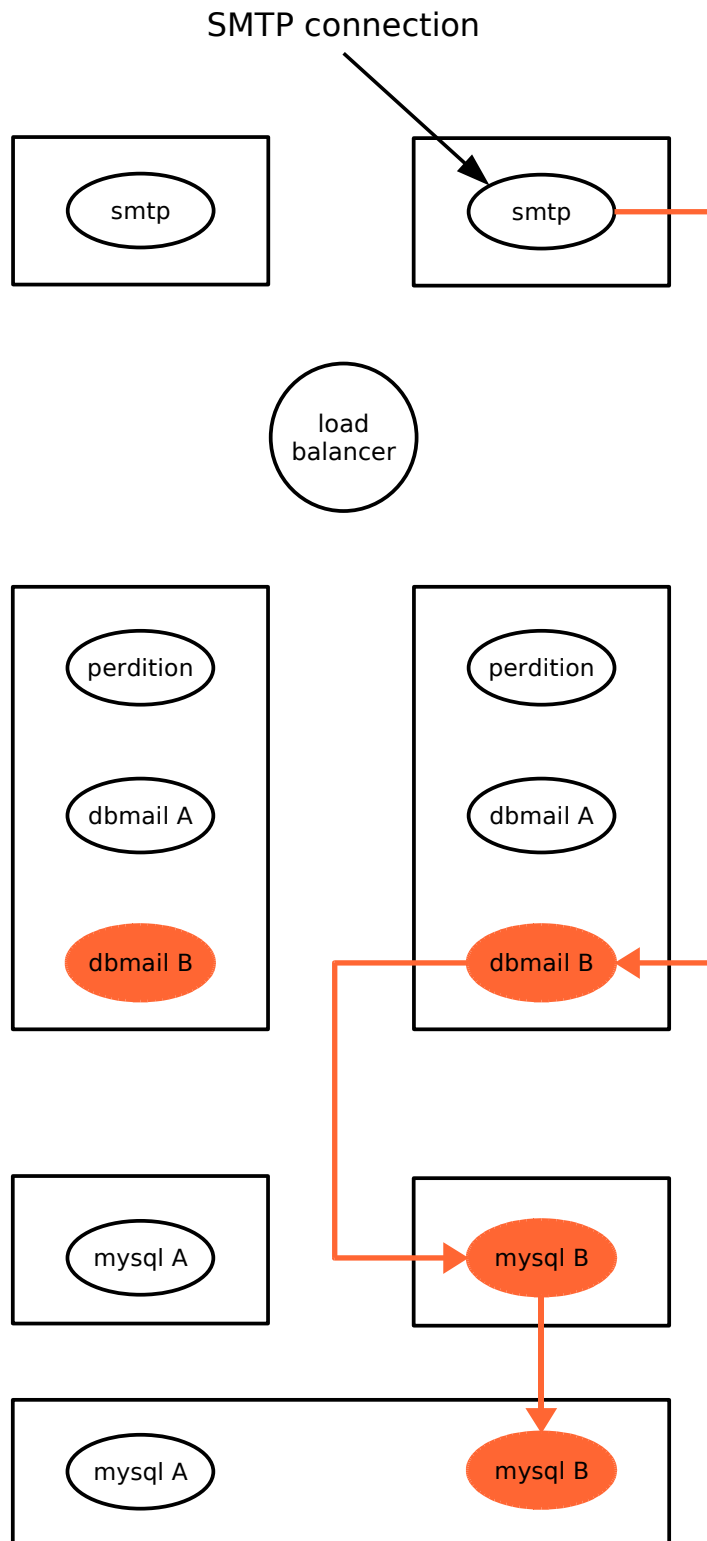


11. Upscaled HA architecture: SMTP (left B)

smtp-left delivers addresses belonging to island B at dbmail-left-b, which inserts into mysql-active-b, which replicates to mysql-back-b.

Upscaled HA architecture: SMTP

7 servers
12 services
2 dbmail islands



12. Upscaled HA architecture: SMTP (right B)

smtp-right delivers addresses belonging to island B at dbmail-right-b, which inserts into mysql-active-b, which replicates to mysql-back-b.

13. Upscaled HA architecture – 3x degraded mode: SMTP (A)

14. Upscaled HA architecture – 3x degraded mode: SMTP (B)

15. Upscaled HA architecture: IMAP/POP3 (left A)

Perdition is a IMAP/POP3 proxy that is able to lookup this LDAP attribute and direct an incoming user to the right dbmail instance.

16. Upscaled HA architecture: IMAP/POP3 (right A)

17. Upscaled HA architecture: IMAP/POP3 (left B)

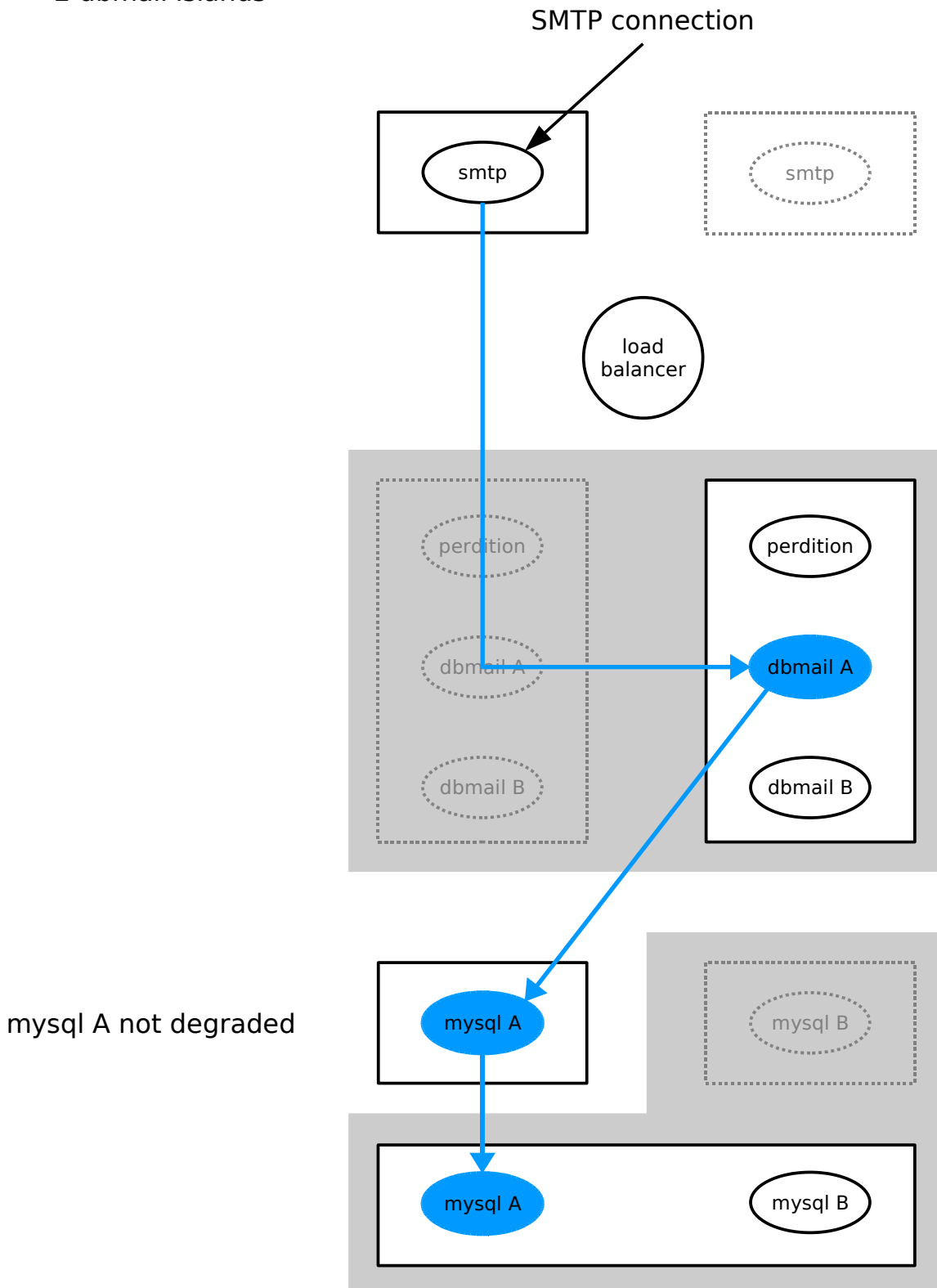
18. Upscaled HA architecture: IMAP/POP3 (right B)

19. Upscaled HA architecture – 3x degraded mode: IMAP/POP3 (A)

20. Upscaled HA architecture – 3x degraded mode: IMAP/POP3 (B)

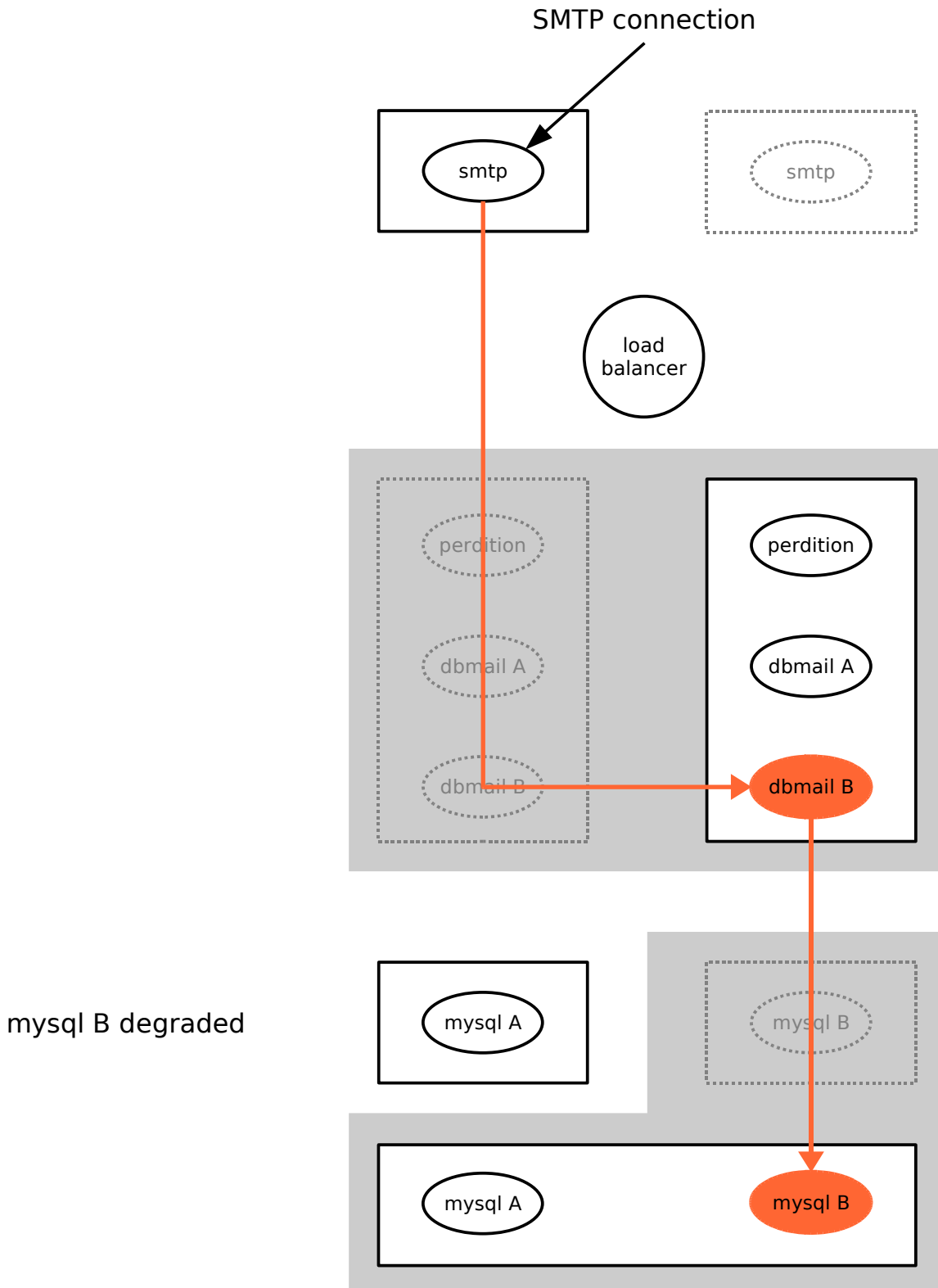
Upscaled HA architecture – 3x degraded mode: SMTP

7 servers
 12 services
 2 dbmail islands



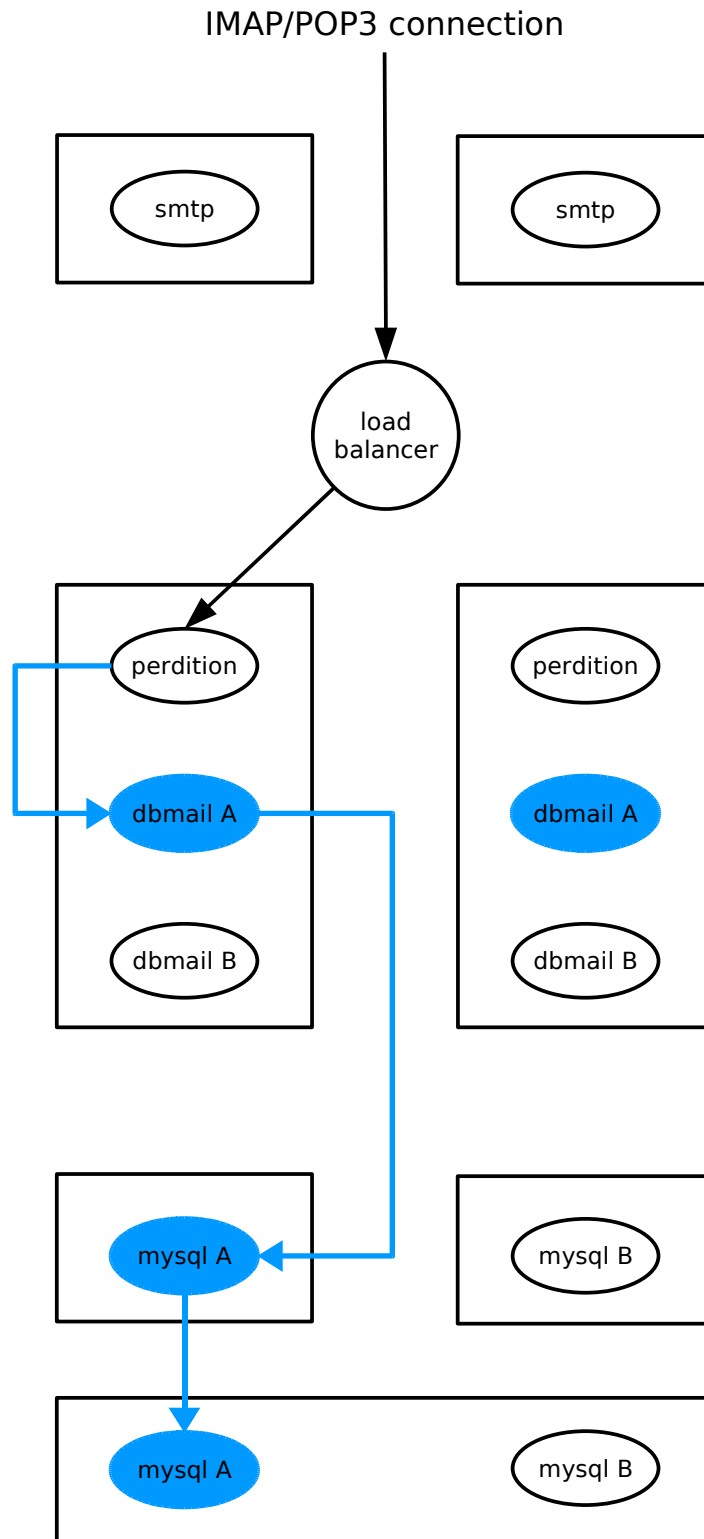
Upscaled HA architecture – 3x degraded mode: SMTP

7 servers
 12 services
 2 dbmail islands



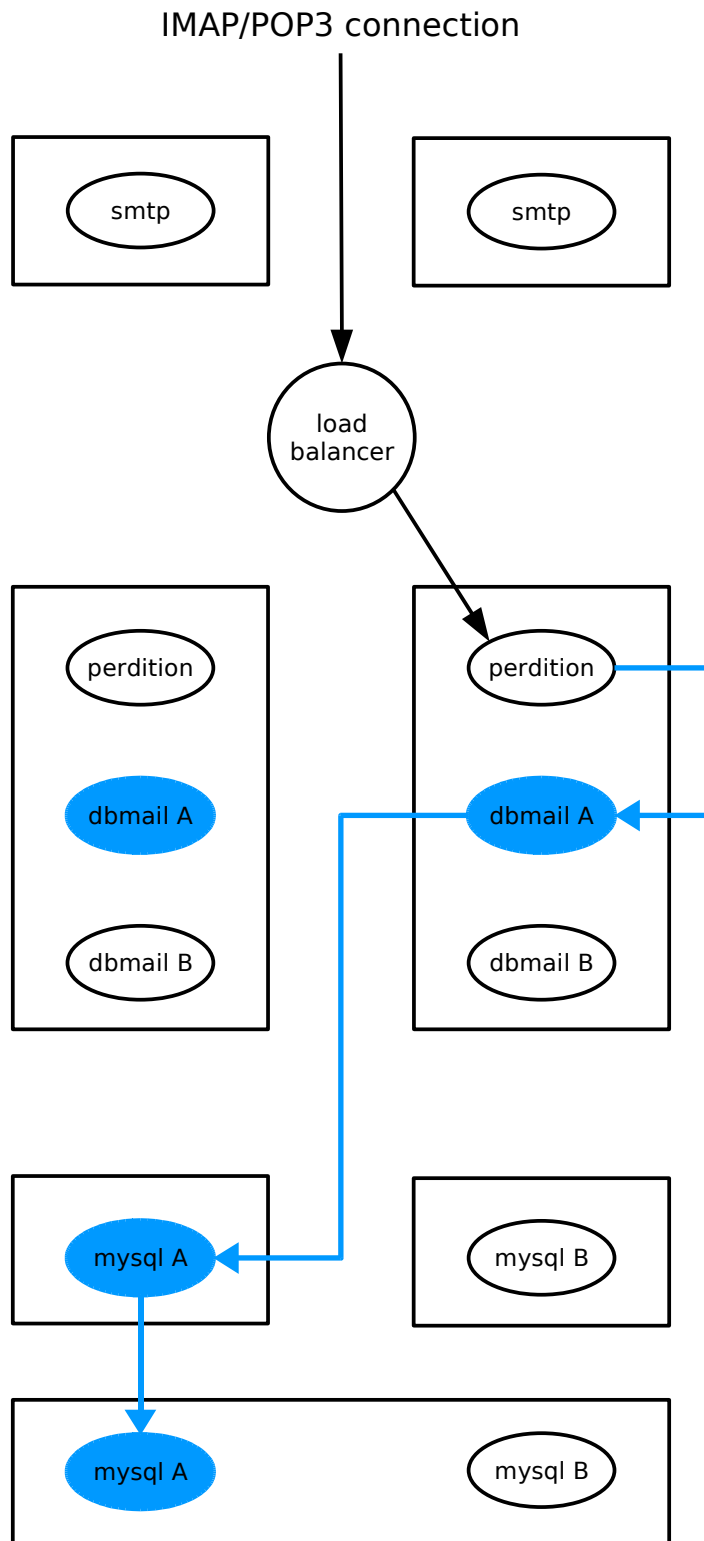
Upscaled HA architecture: IMAP/POP3

7 servers
12 services
2 dbmail islands



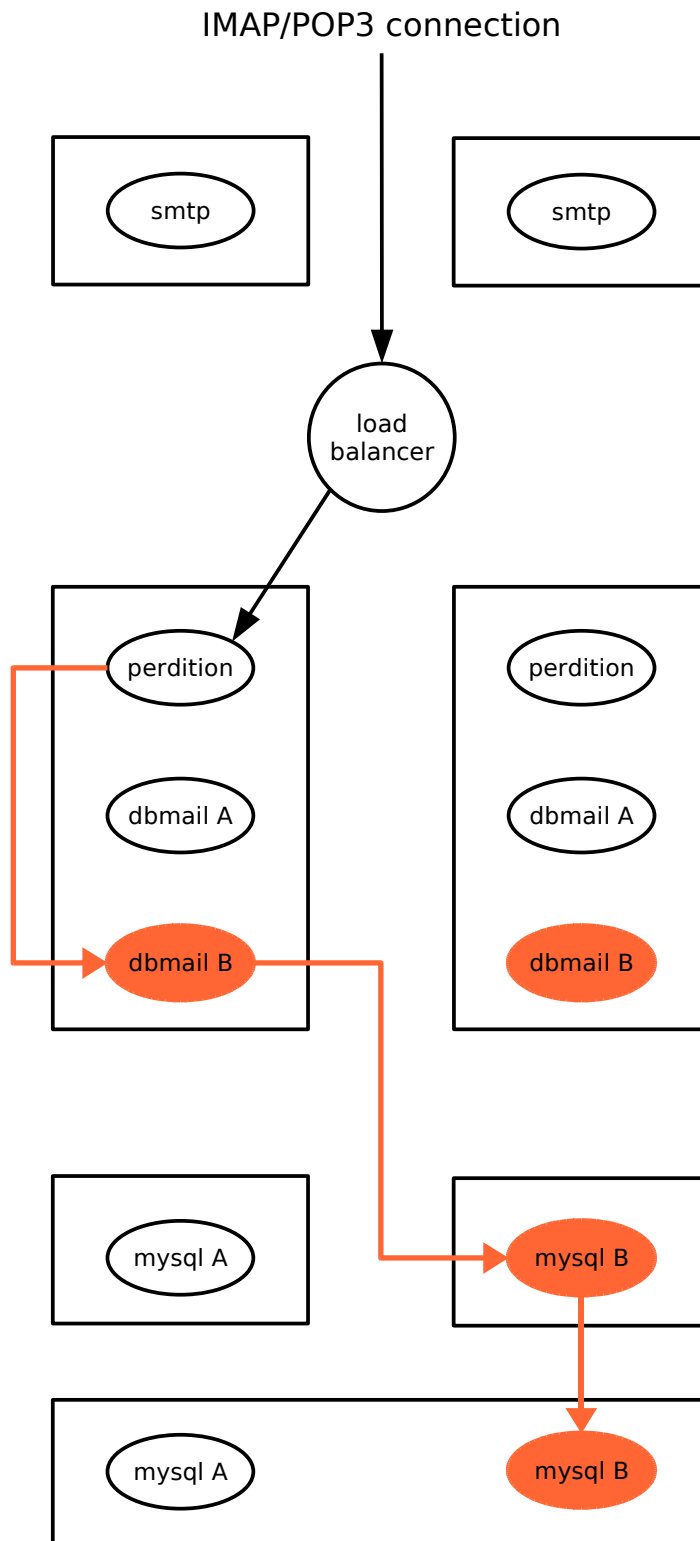
Upscaled HA architecture: IMAP/POP3

7 servers
12 services
2 dbmail islands



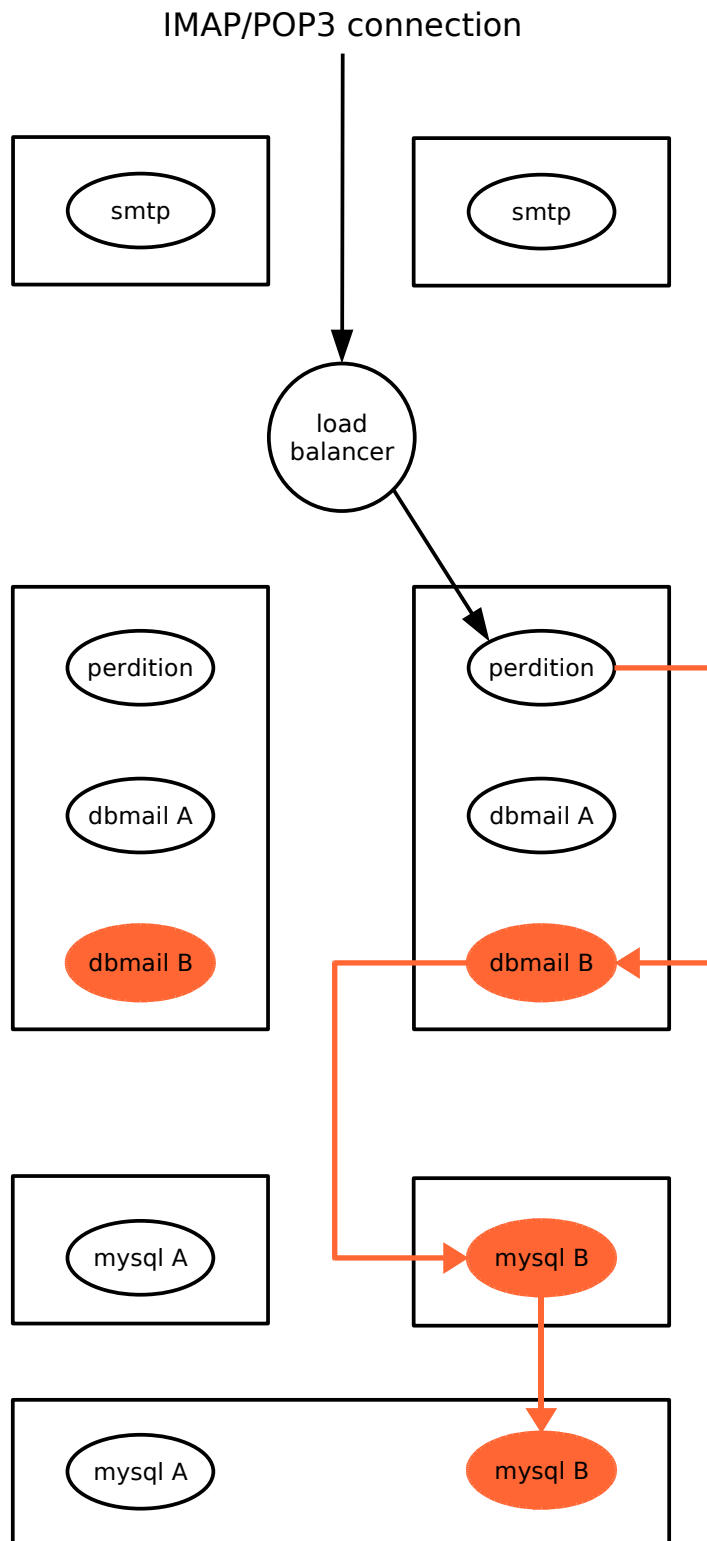
Upscaled HA architecture: IMAP/POP3

7 servers
12 services
2 dbmail islands



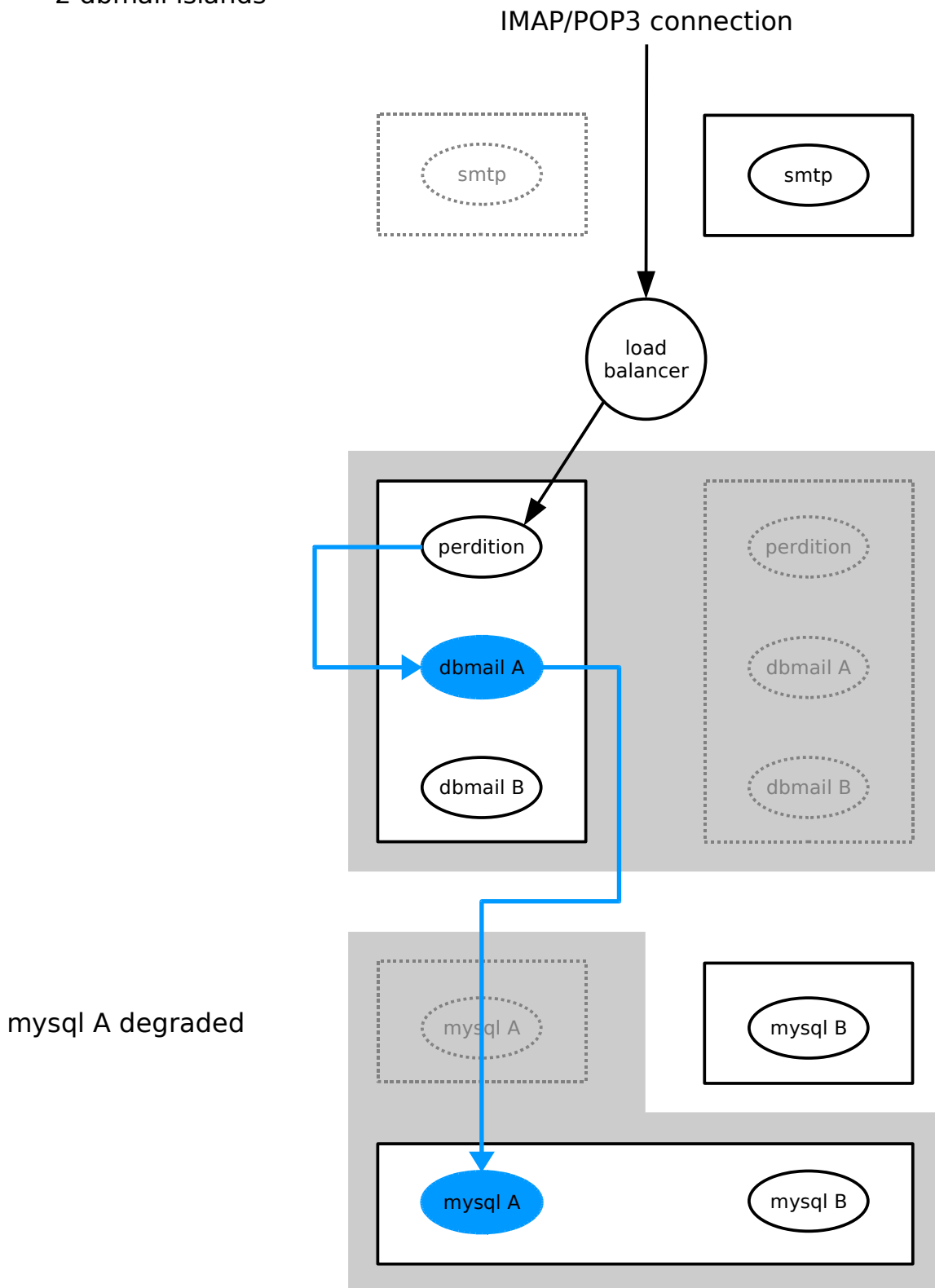
Upscaled HA architecture: IMAP/POP3

7 servers
12 services
2 dbmail islands



Upscaled HA architecture - 3x degraded mode: IMAP/POP3

7 servers
12 services
2 dbmail islands



Upscaled HA architecture - 3x degraded mode: IMAP/POP3

7 servers
12 services
2 dbmail islands

