DNS HA

A multimaster DNS configuration for Disaster Recovery and Business Continuity environment

Riccardo.Veraldi@cnaf.infn.it

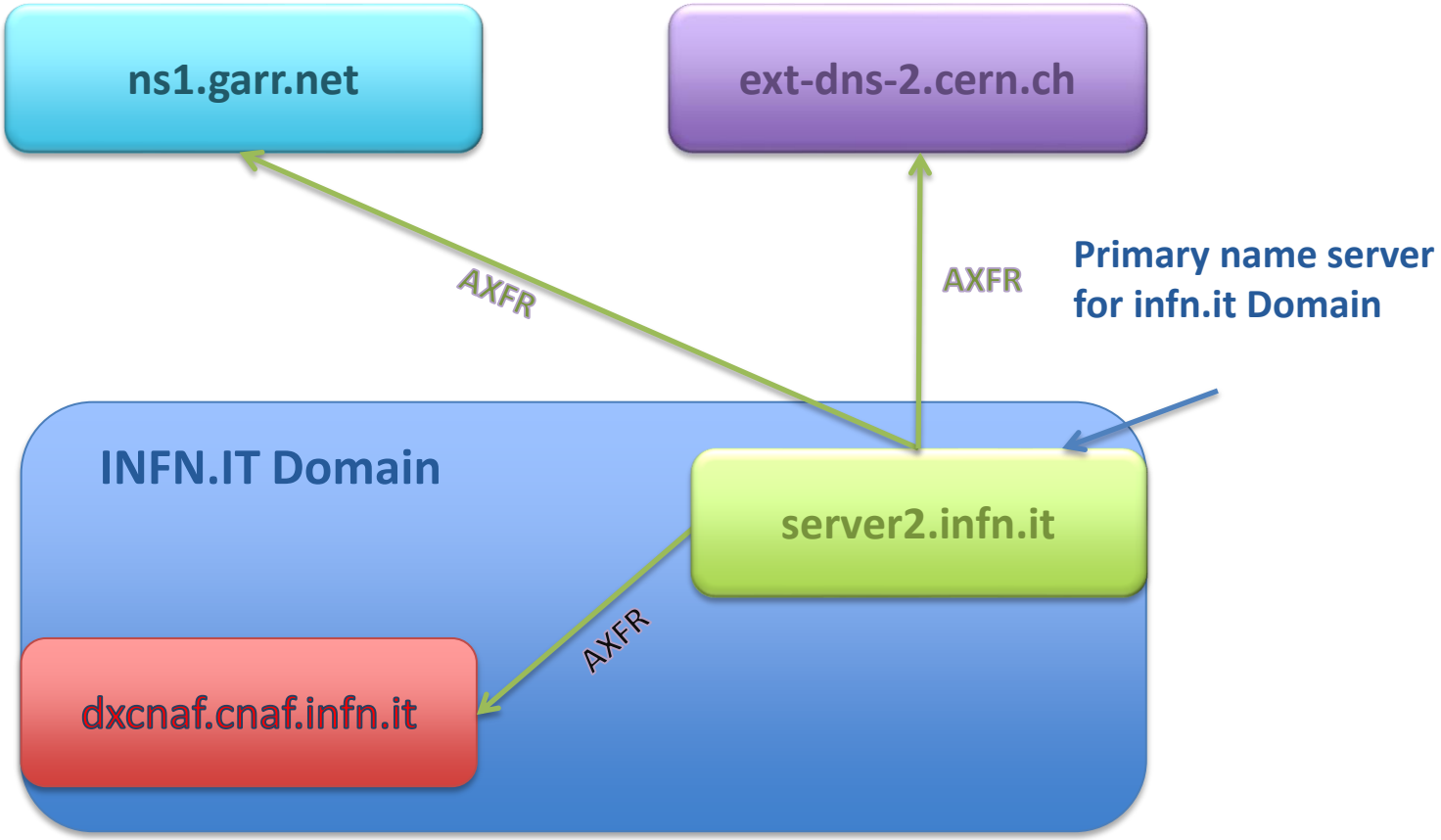
HA DNS General Concepts

It is a fundamental component of a Disaster Recovery strategy

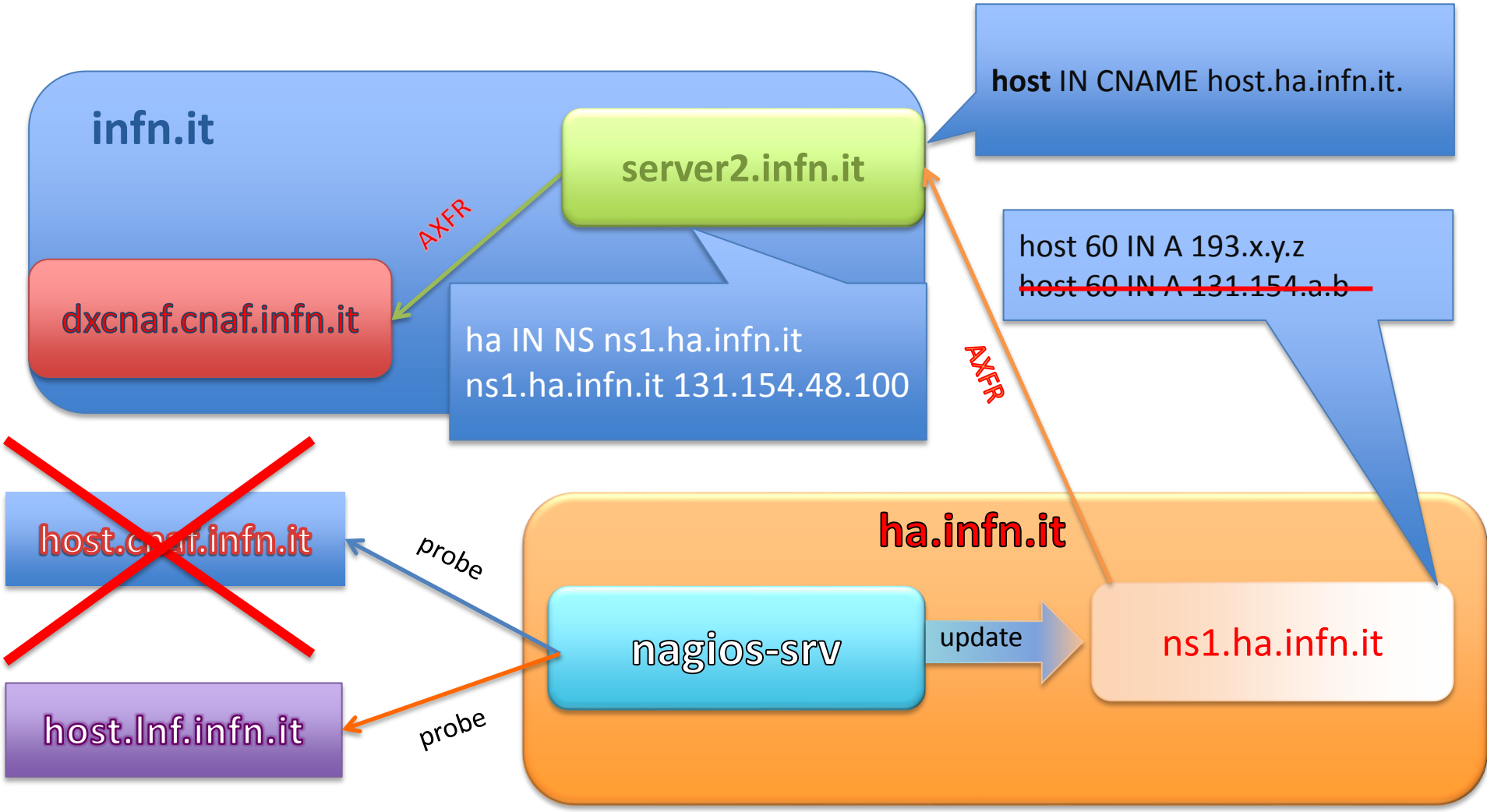
Objectives:

- Build a resilient DNS infrastructure able to guarantee the full functionality in case of "Disaster" in any of the INFN Computing centers.
- The system should be able to modify the IP of a service also during the "down time" of a site hosting one of the authoritative DNS servers for the "INFN.IT" Domain.
- In general INFN national services addresses are of this type: **<ServiceName>.INFN.IT**

Current INFN DNS infrastructure



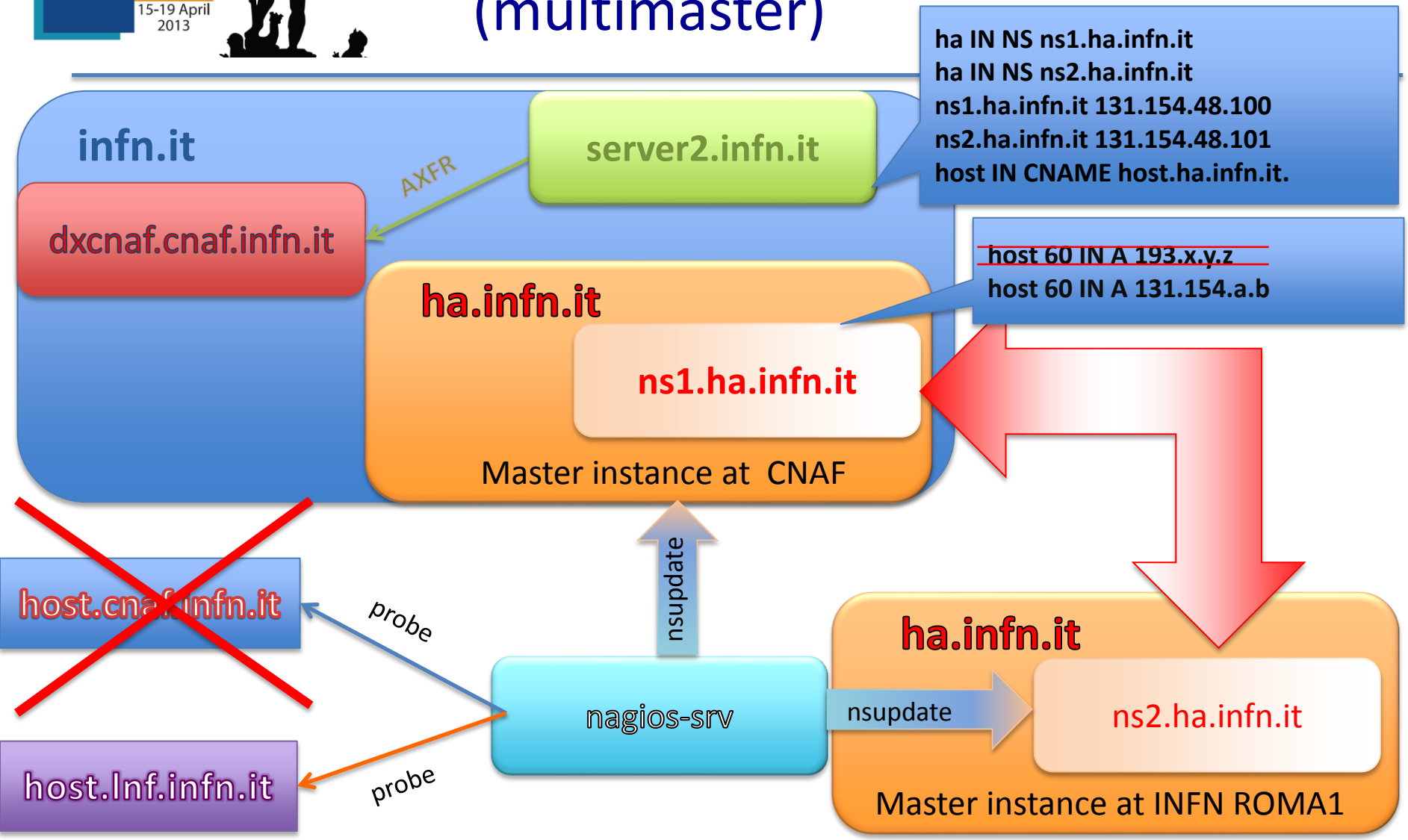
DNS HA Architecture (single master)



HA DNS Architecture (Single Master implementation)

- National Services (**host.infn.it**) are geographically replicated in 2 sites for Example: CNAF in Bologna and LNF (Frascati National Laboratories) in Frascati
- The **ha.infn.it** sub domain defined in a name server **ns1.ha.infn.it** placed in a third site geographically far from the two sites hosting the National Services.
  - A DNS delegation is set up on server2.infn.it (infn.it primary DNS) defining ns1.ha.infn.it as the primary NS for the **ha.infn.it** domain. The host names of the HA Services are registered in server2.infn.it as CNAME pointing to the ha.infn.it domain entries.
- The host name defined in ns1.ha.infn.it points to the IP address of one of the instances of the service with TTL 60
- A **nagios** server (Installed for example in the site hosting ns1.ha.infn.it) probes the different servers implementing the service instances.
  - If the main server doesn't answer to the probe, nagios triggers the nsupdate procedure on ns1.ha.infn.it in order to point to the active instance of the service.
- Using the same CNAME defined in server2.infn.it the service will always be reached in the site where it is Up and Running.

DNS HA architecture (multimaster)



HA DNS multimaster

- National Services (**host.infn.it**) are geographically replicated in 2 sites for Example: CNAF in Bologna and LNF (Frascati National Laboratories) in Frascati
- The ha.infn.it sub domain is now implemented with a Multimaster architecture in two different INFN department for example CNAF and INFN ROMA1
  - A DNS delegation is set up on server2.infn.it (infn.it primary DNS) defining ns1.ha.infn.it and ns2.ha.infn.it as NS for the **ha.infn.it** domain. The host names of the HA services are registered in server2.infn.it as CNAME pointing to the hostnames defined in ha.infn.it domain.
- The host names defined in ns1.ha.infn.it and ns2.ha.infn.it point to the IP address of one of the instances of the service with TTL 60.
- A **nagios** server (Installed in the site hosting ns2.ha.infn.it) probes the different servers implementing the service instances.
  - If the main server doesn't answer to the probe, nagios triggers the update procedure to modify the IP address in ns2.ha.infn.it or in ns1.ha.infn.it if the first is not reachable in order to point to the active instance of the service.
- Using the same CNAME defined in server2.infn.it the service will always be reachable in the site where it is Up and Running.

BIND9 limits

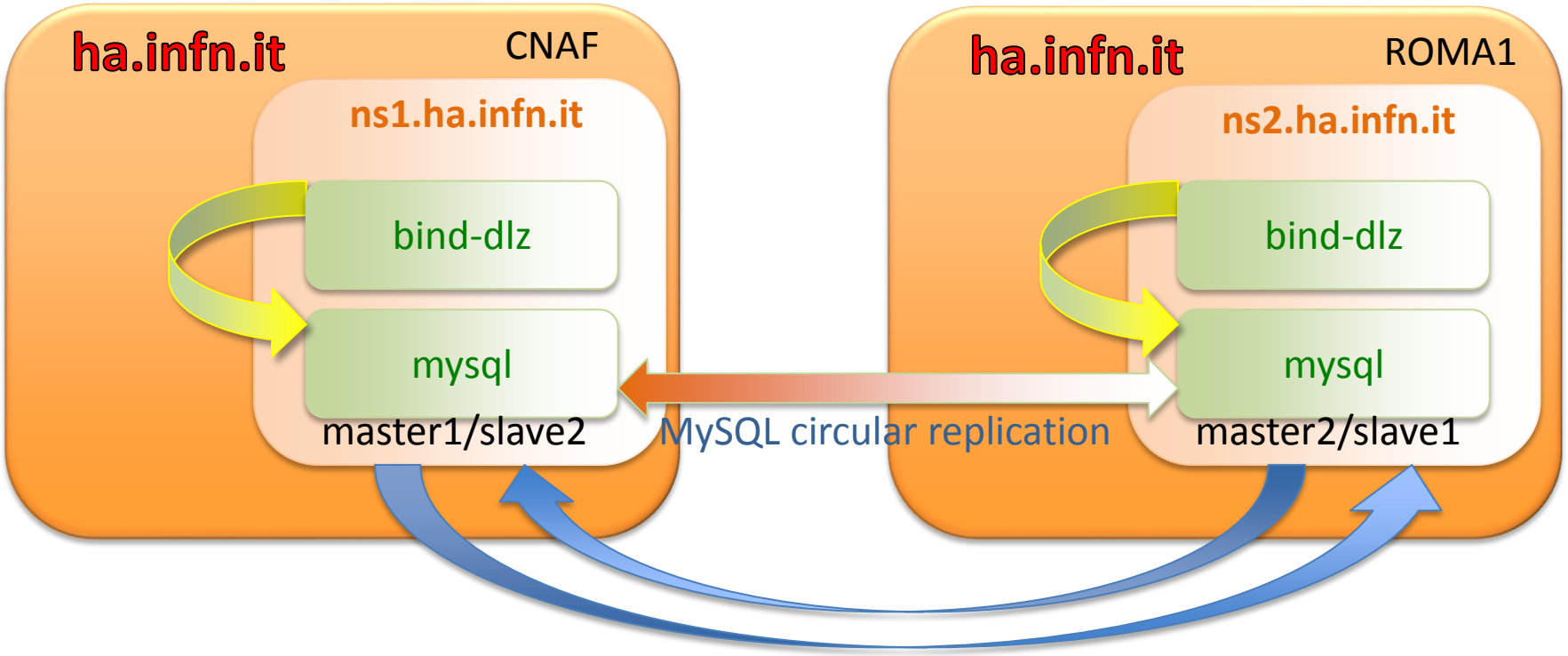
- BIND9
  - reads DNS data from text files. It is very easy to make a mistake when editing a file causing it to be mis-read or made un-readable by BIND
  - stores all DNS data in RAM. If your DNS server is authoritative for a large number of zones, you may have to rebuild the kernel on your machine in order to support BIND's memory needs
  - parses all of its zone files at startup. For a large number of zones this can be time-consuming
  - If you change any information in those zone files, you must reload or restart BIND before those changes take effect. “Do this often enough and BIND could spend more time reloading data than answering queries!”
  - does not support multi-master architecture

BIND-DLZ

- BIND-DLZ is a patch for BIND9
  - It allows you to store your zone data in a database
    - PostgreSQL
    - MySQL
    - Berkeley DB
    - ODBC
    - LDAP

    or
    - FS hierarchical structure
  - Changes in your database are immediately reflected in BIND's response to DNS queries, no need to restart **named** when DNS info changes
  - Il dynamically load zones when needed
  - Very flexible: you can have standard BIND zones and DLZ zones configured in named.conf

BIND-DLZ + MySQL



CentOS + BIND-DLZ

- CentOS 6.4
  - bind-sdb-9.8.2-0.17.rc1.el6 `(yum install bind-sdb)`
  - mysql-server-5.1.67-1.el6
- Centos 5.x
  - it works but bind src.rpm from el6 must be re-built on el5 bind from el5 rpm does not include DLZ extensions
- Any mysql version can be used: tested on 5.0, 5.1, 5.6

named configuration

- A chrooted version of named is always a suggested best practice:
  - bind-9.8.2-0.17.rc1.el6_4.4.x86_64
  - bind-utils-9.8.2-0.17.rc1.el6_4.4.x86_64
  - bind-libs-9.8.2-0.17.rc1.el6_4.4.x86_64
  - bind-sdb-9.8.2-0.17.rc1.el6_4.4.x86_64
  - bind-chroot-9.8.2-0.17.rc1.el6_4.4.x86_64
- DLZ zone in *named.conf*

```
dlz "ha.infn.it zone" {
         database "mysql
   {host=127.0.0.1 dbname=named user=named pass=NameD}
   {select zone from dns_records where zone = '$zone$'}
   {select ttl, type, mx_priority, case when lower(type)='txt' then concat('\"', data, '\"')
        else data end from dns_records where zone = '$zone$' and host = '$record$'
        and not (type = 'SOA' or type = 'NS')}
   {select ttl, type, mx_priority, data, resp_person, serial, refresh, retry, expire, minimum
        from dns_records where zone = '$zone$' and (type = 'SOA' or type='NS')}
   {select ttl, type, host, mx_priority, data, resp_person, serial, refresh, retry, expire,
        minimum from dns_records where zone = '$zone$' and not (type = 'SOA' or type = 'NS')}
   {select zone from xfr_table where zone = '$zone$' and client = '$client$'}
   {update data_count set count = count + 1 where zone ='$zone$'}";
};
```

MySQL Schema

- DLZ driver does not impose a specific schema, It accepts SQL queries with a few special tokens as parameters. These queries are then parsed and the tokens removed. When a query is run, the token is replaced with the appropriate value. This allows a variety of database schemas to be used without modification to the driver's code
  - Limitations:
    - The query must return the appropriate data types in the correct order.
    - The query must use the correct pre-defined tokens.
    - Named must not be multi-threaded

MySQL example schema

```
+-------------+-----------------------------------------+------+-----+---------+-------
+
| Field       | Type                                    | Null | Key | Default | Extra
|
+-------------+-----------------------------------------+------+-----+---------+-------
+
| zone        | varchar(255)                            | NO   | MUL | NULL    |
|
| host        | varchar(255)                            | NO   | MUL | NULL    |
|
| type        | enum('SOA','NS','MX','A','CNAME','PTR') | NO   | MUL | NULL    |
|
| data        | varchar(255)                            | YES  |     | NULL    |
|
| ttl         | int(11)                                 | NO   |     | NULL    |
|
| mx_priority | varchar(10)                             | YES  |     | NULL    |
|
| refresh     | int(11)                                 | YES  |     | NULL    |
|
| retry       | int(11)                                 | YES  |     | NULL    |
|
| expire      | int(11)                                 | YES  |     | NULL    |
|
| minimum     | int(11)                                 | YES  |     | NULL    |
|
| serial      | bigint(20)                              | YES  |     | NULL    |
```

INDEXES: zone, host, type

Step1: on MySQL Master1/Slave2

- Master 1/Slave 2 ip: 131.154.48.100
- Master 2/Slave 1 ip : 131.154.48.101

```
[mysqld]
server-id = 1
#
log-bin=mysql-bin
binlog-do-db=named
binlog-ignore-db=mysql
binlog-ignore-db=test
```

- Create a replication slave account in mysql

```
mysql> grant replication slave on *.* to 'replication'@131.154.48.101 \
identified by '*****';
```

- Restart mysql Master1

Step2: on MySQL Slave1/Master2

```
[mysqld]
server-id = 2
master-host = 131.154.48.100
master-user = replication
master-password = *******
master-port = 3306
```

- Restart mysql slave1/master2 then start slave process

```
mysql> start slave;
mysql> show slave status\G;
```

- Slave_IO_Running and Slave_SQL_Running must be Yes in the output

Step3: on MySQL Master1/Slave2

```
mysql> show master status;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000007 |     5248 | named        | mysql,test       |
+------------------+----------+--------------+------------------+
1 row in set (0.00 sec)
```

This is the simple master-slaves scenario

Step4: on MySQL Slave1/Master2

- Configure Slave1 as a master (Master2)

```
[mysqld]
#information for becoming master
log-bin=mysql-bin
binlog-do-db=named
binlog-ignore-db=mysql
binlog-ignore-db=test
```

- Create a replication slave account on Slave1/Master2 for Master1

```
mysql> grant replication slave on *.* to \
'replication'@131.154.48.100 identified by '********';
```

Step5: on MySQL Master1/Slave2

- Edit my.cnf on Master1 for information of its master.

```
[mysqld]
#information for becoming slave.
master-host = 131.154.48.101
master-user = replication
master-password = **********
master-port = 3306
```

- Restart both Master1 and Master2

- On Master1

```
mysql> start slave;
```

Step6: on MySQL Master1 and Master2

- Checking slave and master status both on Master1 and Master2

```
mysql> show slave status\G;

mysql> show master status;
```

- Both MySQL servers are configured as slave/master to each other allowing a circular replication
- To Avoid auto incement key clashing
  - On Master 1

```
[mysqld]
auto_increment_increment = 2
auto_increment_offset = 1
```

  - On Master2

```
[mysqld]
auto_increment_increment = 2
auto_increment_offset = 2
```

conclusions

- With the introduction of a new domain **ha.infn.it** with two nameservers (multimaster) installed in different sites with the correct CNAME configuration and delegation on the top level domain nameserver is possible to build a DNS architecture for Geographically redundant services.
- The Multimaster implementation permits to change the ip addresses even if one of the two sites hosting the DNS service is down.
  - The technologies used in this implementation of the multimaster DNS is based on Bind9-DLZ + mysql backebd