SLAVE NAME SERVERS

A Slave DNS gets its zone data using a zone transfer operation (typically from a zone master) and it will respond as authoritative for those zones for which it is defined to be a 'slave' and for which it has a currently valid zone configuration. It is impossible to determine from a query result that it came from a zone master or slave.

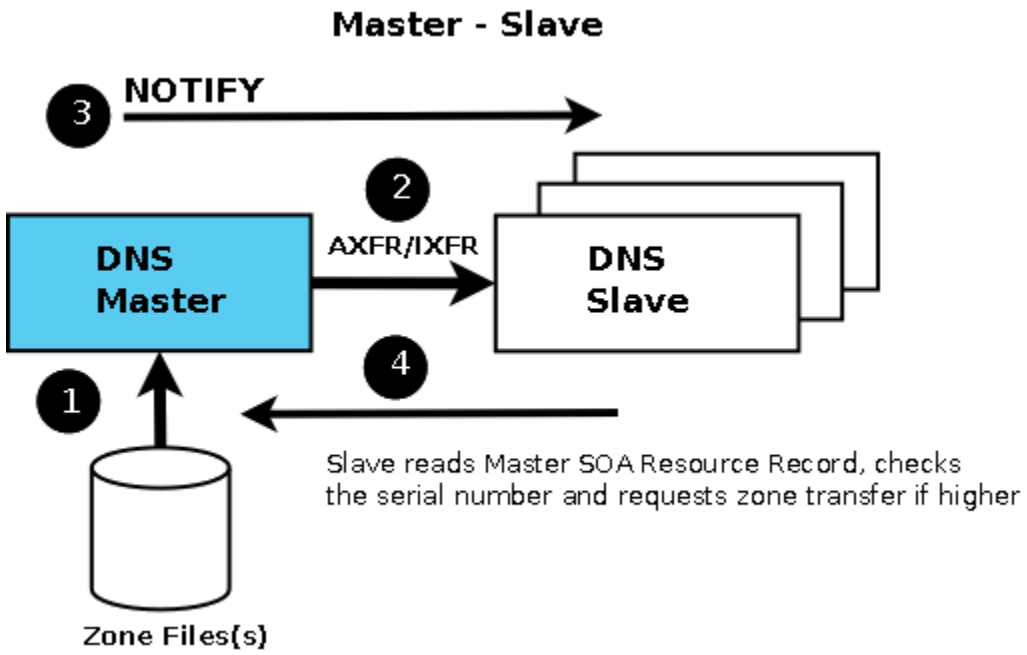

Diagram 2 DNS Slave Server

The term 'slave' was introduced with BIND 8.x and replaced the term 'secondary'.

There can be any number of slave DNS's for any given zone.

Slave status is defined in BIND by including 'type slave' in the zone declaration section of the named.conf file as shown by the following fragment.

```
// example.com fragment from named.conf
// defines this server as a zone slave
zone "example.com" in{
        type slave;
        file "sec/sec.example.com";
        masters {192.168.23.17;};
};
```

Notes:

1. The master DNS for each zone is defined in the 'masters' statement of the zone clause and allows slaves to refresh their zone record when the 'expiry' parameter of the SOA Record is reached. If a slave cannot reach the master DNS when the 'expiry' time has been reached it will stop responding to requests for the zone. It will not use time-expired data.
2. The file parameter is optional and allows the slave to write the transferred zone to disc and hence if BIND is restarted before the 'expiry' time the Slave server will use the saved data. In large DNS systems this can save a considerable amount of network traffic.

Assuming NOTIFY is allowed in the master DNS for the zone (the default behaviour) then zone changes are propagated to all the servers defined with NS Records in the zone file. Other acceptable NOTIFY sources can be defined using the also-notify parameter in named.conf.

But Slaves can also be Masters:

This section can get a bit confusing. Read it only when accompanied by your favorite keep-me-awake-cos-I-can't-take-anymore-of-this-stuff beverage.

The definition of a slave server is simply that it gets its zone data via zone transfer, whereas a master gets its zone data from a local file system. The source of the zone transfer could just as easily be another slave as a master.

So what sane human would want to do that?

1. Assume you want to hide your master servers in, say, a stealth configuration then at least one slave server will sit on the public side of a firewall, or similar configuration, providing perimeter defence. To provide resilience you would need two or more such public slaves. The second slave can be updated from the same master as the first or it could be updated from the slave server - we'll call it the 'boss' slave to avoid getting into tortuous terminology (is it a master-slave or a slave-master?).

To configure this miracle the second slave server would define the 'boss' slave's IP in its masters statement. When the 'boss' slave has sucessfully transfered a zone file (from the master) it will send out NOTIFY messages (the default) unless configured not to do so. This type of configuration will marginally increase latency for updating the zone on the second slave - but that may be more than offset by increased stealth.

2. In a DNSSEC environment the master will likely have all kinds of whizzo dodads concerned with keeping keys secure. Whereas DNSSEC slaves simply send the data in the zone file in response to queries and have no requirements for secure key maintenance. Hidden master configurations will become increasingly the norm in this environment.

Source: http://www.zytrax.com/books/dns/ch4/