DNS and BIND Best Practices Checklist

Infoblox
CONTROL YOUR NETWORK

The following 12-item checklist should help ensure that your solution is more secure, easier to manage, and more reliable. The DNSone® appliance-based solution from Infoblox can help you implement these best practices simply and effectively.

1. Use Forwarders Sparingly

Most corporate DNS architectures rely on forwarders. Some use of forwarders is inevitable: It's a bad idea to expose all of your internal name servers directly to the Internet, so you funnel queries for Internet domain names through a smaller group of forwarders. However, many companies create Byzantine name resolution architectures in which one name server forwards to another, which in turn forwards to another. These designs are almost unavoidably brittle: The failure of any forwarder in the chain slows or breaks name resolution. Even during normal operation, the forwarders handle a disproportionate share of the resolution load, and resolution paths are needlessly inefficient.

2. Use Hidden Primary Masters

A hidden primary master name server is one that doesn't appear in the NS records for its zones and doesn't serve any resolvers. So what good is it? Its only responsibility is to serve zone transfers to slave name servers. (Well, it also sends them NOTIFY messages and handles their refresh queries, and it may process dynamic updates, too.)

Running a hidden primary master gives you flexibility in its administration. A primary master that handles queries from resolvers and other name servers can't be brought down for long without causing query timeouts. But if a hidden primary takes a few minutes to reload a large zone, there's no adverse effect. If you muff a configuration change and the name server won't start, you'll have the full expiration interval of your zones to fix the problem.

3. Don't Place All Your Eggs in One Basket

In a similar vein, there are advantages to separating a name server's recursive and authoritative functions. Most name servers, of course, can process both recursive and non-recursive queries simultaneously. But splitting the two functions between two sets of name servers has pronounced advantages.

First, you can more effectively secure each set of name servers. On the authoritative-only name servers, you can disable recursion. This helps the name servers resist denial-of-service attacks, since a BIND name server's capacity for processing non-recursive queries dwarfs its capacity for handling recursive queries. Authoritative-only name servers also don't cache, so you won't need to worry about your server's data segment growing out of control.

You can configure the recursive name servers to deny name service to all but authorized queriers. This should make it difficult for hackers to attack the name server without the collusion of one of your users.
This separation also helps compartmentalize breaches and failures of name servers. If a hacker finds a vulnerability in the query processing code in BIND, he may be able to crash your authoritative name servers, but won't be able to compromise your recursive name servers, which won't accept his queries.

4. Call for Backup

Despite your best efforts, your name servers will sometimes go down. They may crash. They may become overloaded and stop processing queries. If your resolvers only know about one name server and it's down or unresponsive, their queries will fail and your users will get angry. So configure your resolvers with a backup name server in their configurations so that if the first fails, they'll move on.

5. Keep Watch over Your Flock

Like I said, sometimes your name servers will go down. If you're not monitoring them, you may not find out right away. Someone else may have to tell you. Like your boss.

DNS and BIND Best Practices Checklist

Infoblox
CONTROL YOUR NETWORK

If you set up a system to monitor your name servers, you'll be the first to know. Any respectable monitor should check not only that your name servers are responding, but also that they're authoritative for the zones assigned to them. The free mon software includes a DNS service monitor.

You can also monitor the operation of your name servers by collecting their syslog output on a single loghost and using tools such as swatch to alert you when a name server logs a significant message. This would enable you to detect a slave name server's repeated failures to transfer a zone, for example.

6. Location, Location, Location

Most modern name servers have sophisticated internal mechanisms for locking onto the fastest of a set of authoritative name servers. If you know where the name servers that query your zone data are, place authoritative name servers near them. They'll automatically gravitate to the closest name server. This is especially important if your name servers serve zone data to Internet name servers; make sure that some are close to your connections to the Internet, or ask your ISP to configure one of their name servers as a slave for your zones.

7. Tread Lightly

Many administrators reflexively restart a name server after changing its named.conf file or editing a zone data file. This is akin to using a bulldozer when a shovel would do. The name server loses its cache (including round-trip time data) and must re-read all of its zone data files. When administering a name server, use the least disruptive option that accomplishes what you want. If you've only edited one zone data file, reload just that zone. If you've added a zone statement to named.conf but made no other changes, use reconfig, not reload or restart.

8. Use NOTIFY to Speed Zone Propagation

The NOTIFY mechanism allows the primary master name server for a zone to alert the slaves to changes in the zone's data. With NOTIFY, changes are normally propagated within minutes. Without NOTIFY, the slaves poll periodically to determine whether zone data has changed, and those changes may take hours to percolate.

NOTIFY is on by default, which is usually a good thing, but there's no way for a zone's primary master to automatically discover stealth slave name servers (those not listed in the zone's NS records). Those require explicit configuration on the primary master (e.g., via BIND's also-notify substatement).

By default, slave name servers also send NOTIFY messages in case they serve as masters to other slaves. This is a waste of resources if a slave doesn't act as a master, so turn NOTIFY off for these name servers.

9. Prefer TSIG to IP Addresses

Use Transaction Signatures (TSIG) instead of IP address-based access control whenever possible. TSIG provides cryptographic authentication of DNS messages and integrity checking; source IP addresses provide only a weak form of authentication—one easily subverted in UDP-based transactions. Some types of UDP-based messages are particularly dangerous in the wrong hands—think dynamic updates—so you should be as careful with them as you can.

10. Hide behind an Alias

Do you dread moving network servers because of the client reconfiguration it entails? Moving an IMAP or NTP server shouldn't cause angst. Create a functional alias for the server, say imap.<domain>, that points to the real domain name of the host currently offering the service. Make sure all client configurations use the alias, not the server's real name. Then, if you're forced to move the host, just change the target of the alias.

DNS and BIND Best Practices Checklist

Infoblox
CONTROL YOUR NETWORK

Beware those services that require canonical domain names, though. Mail servers listed in MX records and name servers listed in NS records, in particular, must have address records associated with them. However, you can work around this by adding an address record to the functional name (e.g., smtp.<domain>) that points to the host's current address. If you move the SMTP server, just change the address record.

11. Always Test Changes

Test modified configurations and zone data before putting them into production. Errors you inadvertently introduce will cause outages: Syntax errors in zone data files will prevent the name server from loading the zone, while syntax errors in a name server's configuration file will keep the name server from starting.

After modifying zone data, you can use named-checkzone (a standard part of BIND 9) to check for syntax errors. If you don't have named-checkzone available, you can configure a new named process, listening on a non-standard port, to load (and thereby test) the zone data file. (Just look in the name server's syslog output for errors.) Similarly, you can use named-checkconf or a new namedprocess to check a named.conf file for errors.

12. Plan Ahead

When changing the IP address of an important host, plan ahead. If you change the host's address record after making the change, it may take as long as the old record's TTL until other resolvers and name servers time out the old address—which means a potential outage.

If you reduce the TTL of the address record you'll be changing ahead of time, you can minimize the chance of an outage. One TTL cycle before the change (e.g., if the record's TTL is one hour, do this at least one hour before the change), reduce the TTL to a very small value, say five seconds. Then, when you make the change, any remote resolvers and name servers that have the old address cached will have it cached for no longer than five seconds. When you modify the record to reflect the new address, you can restore the TTL to its original value.

This separation also helps compartmentalize breaches and failures of name servers. If a hacker finds a vulnerability in the query processing code in BIND, he may be able to crash your authoritative name servers, but won't be able to compromise your recursive name servers, which won't accept his queries.

Corporate Headquarters: +1.408.986.4000 1.866.463.6256 (toll-free, U.S. and Canada) info@infoblox.com www.infoblox.com