

Inhalt:

dynamische DNS Zone Infrastruktur

- ▷ DNS Zones in MySQL
- ▷ Verteilung der Zones an Slaves
- ▷ Aktualisierung der Slaves

Aufgabe:

Ein Netzwerk mit ständig neuen, veränderten und nicht mehr benötigten DNS Einträgen zu einer schnellen, flexiblen und pflegbaren Umgebung umzubauen. In dem neuen Setup soll mit einem Browser Änderungen an den umgestellten Zonen möglich sein. Dabei soweit als möglich auf bewährten Komponenten aufsetzen. Ein caching der Zones soll ebenso wie die Anbiertung von statischen Zonen weiterhin möglich sein. Für interne Zonen soll eine ACL den Zugriff auf die IP Adressen des Unternehmens beschränken.

Nicht gemeint ist bei dieser Beschreibung die dynamische DNS Konfiguration, wie sie z.B. DHCP Server benutzen um Clients im DNS einzutragen.

Umgebung:

- DNS (bind)
- Webserver (apache)
- RDBMS (mysql)

Ansätze:

1. Nutzung der BIND Erweiterung SDB für die direkte Anbindung eines bind9 Servers an eine MySQL Backend Datenbank
2. Erzeugung von Zone Files aus einer DB und restart von BIND bei Aktualisierungen
3. Nutzung eines DNS Servers der native mit einer MySQL Datenbank arbeitet

Lösung 1: BIND MySQL SDB

In BIND ist die Schnittstelle zur Datenspeicherung in die sogenannte SDB gekapselt und es gibt mehrere Projekte die sich daran gemacht haben eine MySQL Anbindung zu realisieren. Keines der Projekte hat es bisher geschafft in die Quellen von BIND aufgenommen zu werden und teilweise sind die notwendigen Patches nicht für die aktuelle Version.

Den besten Eindruck macht BIND DLZ, da es hinsichtlich der Backend Systeme auch weitere Datenbanken unterstützt. Nach der FAQ: postgresQL, MySQL, Berkeley DB, ODBC, LDAP es ist eine weitere Betrachtung sicher Wert, dieses Dokument geht allerdings erstmal nicht weiter darauf ein.

MySQL BIND - <http://mysql-bind.sourceforge.net/> - Version 0.1 vom 12.03.2004

BIND MySQL - <http://gw.netbastards.org/bm/> - Version für BIND 9.2.2 vom 10.03.2003
Projekt wird vom maintainer nicht mehr supported

BIND Dynamically Loadable Zones - Version 0.6 vom 1.11.2003
<http://www.nlnet.nl/project/bind-dlz/> bzw. <http://bind-dlz.sourceforge.net/>

Lösung 2: dump von Zone Files ins Filesystem und restart des DNS

Aufgrund der in der Zwischenzeit vorhandenen Alternativen wird auf diese Realisierung gänzlich verzichtet. Es macht, für den Autor, keinen erkennbaren Sinn auf ein derartiges System zu wechseln.

Lösung 3: nativer MySQL DNS Server

Da noch keines der SDB Projekte die entgültige Reife erreicht hat und Bestandteil von BIND wurde ist dieser Zwischenschritt interessant. Mit einer derartigen Lösung kann zu einem späteren Zeitpunkt leicht auf eine SDB Anbindung innerhalb von BIND gewechselt werden kann. Zwischenschritt deshalb, weil hinsichtlich Security und Features der BIND die Referenzimplementation ist und die meisten Sites im Internet an einer Weiterentwicklung dieses Projektes interessiert sind. Die hier genannten Lösungen sollten daher nur im Hintergrund für die dynamischen Zonen sorgen und die Auslieferung der Anfragen BIND oder djbdns überlassen. Vor allem hinsichtlich ACLs und Zonetransfers, aber teilweise auch bei RR Typen ist die Implementierung nicht vollständig.

Anbieten tun sich für dieses Vorhaben:

MyDNS - Version 0.10.3 vom 12.03.2004
<http://mydns.bboy.net/>

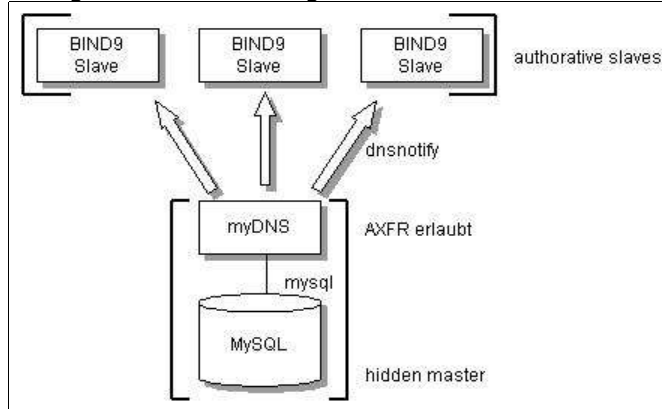
POWERDNS - Version 2.9.16
<http://www.powerdns.com/>

<http://www.onsite4u.de/>
E-Mail: info@onsite4u.de
Telefon: 02203 800 305



Realisierung

Es wird die Umsetzung der Lösung 3 beschrieben, hierbei können die vorhandenen BIND9 Nameserver als Slaves für die dynamischen Zonen die Verteilung übernehmen. Es wurde der myDNS verwendet, weil er zusätzlich zu allen notwendigen Features auch bereits ein fertiges Webinterface mitbringt. Es ergibt sich in etwa folgendes Bild:



Ein myDNS Server realisiert die Anbindung an MySQL und dient als AXFR Quelle für beliebige Slave Nameserver. Ein nettes Feature von myDNS ist, dass er sein benötigtes Datenbank Schema direkt in sich trägt und z.B. über `mydns -create | mysql -h xxx -u xxx -p xxx` werden in der Datenbank 2 Tabellen erzeugt, das ist alles was für den Betrieb benötigt wird (`rr / soa`). Als absolut nützlich hat sich der Import vorhandener Zonen per AXFR gezeigt, so kann eine vorhandene Zone von jedem beliebigen Nameserver (der dem myDNS das erlaubt) die Zonen übernommen werden, es dauert nur Sekunden. Da myDNS selbst kein `dnsnotify` unterstützt, aber die BIND Slaves auch vor Ablauf der Cache Zeiten bei einem `dnsnotify` sich von Ihrem Master Server ein Zone Update holen sollen, wurde dies mit einem kleinen Shell Skriptes realisiert. myDNS muss dazu der AXFR transfer erlaubt werden. Allerdings verwendet BIND9 als Standard Einstellung in der Zwischenzeit IXFR (incremental transfers), so dass eine Aktualisierung solange scheitern muss, bis den Slaves fest konfiguriert wird, dass sie von dem myDNS Master nur AXFR Zonentransfers durchführen können.

Hier die notwendigen Konfigurationen:

```
/etc/mydns.conf
allow-axfr = yes           # Should AXFR be enabled?
```

```
/etc/named.conf
server 192.168.x.x {
    request-ixfr no;
};
zone "intern.xxx.de" in {
    type slave;
    file "slave/intern.xxx.de";
    masters { 192.168.X.X; };
};
```

Mit dem `masters` Eintrag erfährt der Slave wo er die Zone bekommt und mit dem `server` Eintrag erfährt er dass er keinen IXFR Transfer starten soll.

Das `dnsnotify` Perl Script sieht wie folgt aus und verrät beim Aufruf ohne Parameter wie es gerne bedient werden möchte:

```
#!/usr/bin/perl
my ($domain, $slave) = @ARGV;
die "usage: $0 domain.to.refresh slave.I.P.addr\n" unless $slave;
use Socket; socket
SOCK,AF_INET,SOCK_DGRAM,getprotobyname('udp');
$to = sockaddr_in($3,pack"C4",split/\./,$slave));
$pkt=pack("nnnnna*nn",0,0x2400,1,0,0,0,
    join(" ",map{chr(length)}.${split/\./,$domain}."00",6,1));
send(SOCK,$pkt,0,$to);
recv(SOCK,$buf,10240,0);
```

Das Skript gibt es hier:
<http://powerman.sky.net.ua/Projects/dnsNOTIFY/dnsNOTIFY>

Ein Screenshot von dem aktuell mitgelieferten PHP Admin Interface für myDNS:



Eine Einschränkung ist zum Schluss bei Tests der Lösung noch aufgefallen. Der myDNS kennt in der aktuellen Version noch keine DNAME RR Types. Eine Untersuchung auf IPv6 wurde nicht vorgenommen.

Fazit: Mit dieser Lösung dauert es eine Minute von der Eintragung eines Servers in die MySQL Datenbank bis zur Verfügbarkeit im LAN. Die Zeit lässt sich durch eine direkt Integration des `dnsnotify` Scriptes in ein verbessertes Frontend weiter optimieren. Aktuell wird nach einer Änderung an einer Zone in der DB mittels Cron ein Update der Slaves gestartet. Achtung! - die Slaves erkennen Updates an den Zonen an der Serial Nummer, d.h. nach jeder Änderung muss diese im Frontend ebenfalls aktualisiert werden. Die Lösung ist durch die `refresh / retry / expire` Zeiten gegen Ausfall des hidden masters geschützt. Der hidden master kann selbstverständlich zusätzlich redundant aufgebaut werden. Die Trennung der MySQL Datenbank vom Server des Masters ist möglich und bei Nutzung einer verfügbaren Datenbank im Unternehmen sicher eine Option. Die Datenbankbelastung ist in diesem Szenario minimal, da Clients nicht bis auf die Datenbank durchgereicht werden.

