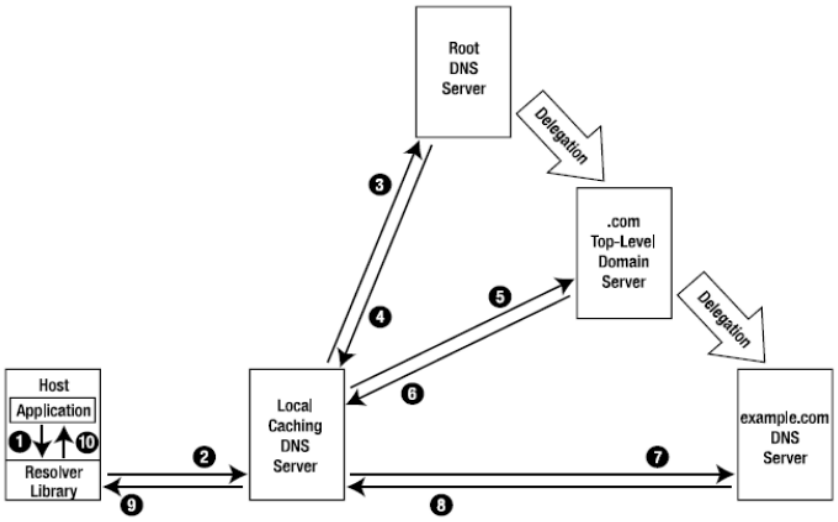
BAB 5 - Domain Name System (DNS)

Bagi kita sebagai manusia, sangan sulit bila harus menggunakan alamat IP terutama dengan alamat Ipv6 untuk melakukan akses pada suatu jaringan. Tetapi kini Domain Name System (DNS) memungkinkan kita untuk melakukan akses jaringan degan lebih mudah yaitu dengan menggunakan nama simbolik. Jadi, DNS akan menerjemahkan IP tersebut (dan juga sebaliknya). Suhingga DNS itu sendiri juga perlu diperbarui umtuk mendukung Ipv6. Pada bab ini akan dibahas mengenai Berkeley Internet Name Domain (BIND) perangkat lunak server DNS. Tetapi sebelum itu akan dijelaskan terlebih dahulu mengenai bagaimana DNS bekerja dan perubahan yang diperlukan dan yang tidak diperlukan untuk Ipv6. Gambar 5-1 menunjukkan bagaimana interaksi antara bagian pada Domain Name System.



Gambar 5-1. Mencari informasi dalam Domain Name System

Keterangan :

- Langkah 1

  Bila suatu aplikasi ingin berkomuniaksi melalui jaringan, maka dibutuhkan nama lengkap tujuan. Contoh : www.example.com. Ini sering disebut “Fully Qualified Domain Name” (FQDN). Selain itu, aplikasi harus menemukan alamat yang cocok di DNS dengan cara memanggil *resolver library*.

- Langkah 2

  Resolver library ini akan bekerja sebagai pengirim permintaan untuk informasi yang diperlukan untuk “caching” atau “rekursif” server DNS menggunakan protokol DNS.

- Langkah 3
  Jika server caching belum menemukan jawaban atas permintaan sebelumnya dalam memori, maka tidak dapat diketahui di mana untuk menemukan informasi yang diminta. Maka akan menghubungi salah satu DNS Root server. DNS Root server tidak tahu alamat www.example.com, tetapi DNS Root server memiliki pointer ke

server DNS yang bertanggung jawab untuk semua “too\p-level domain” (TLDs),seperti .Com.

- Langkah 4
  Root server mengirimkan kembali pointer ke com TLD nameserver. Informasi yang diterima dari nameserver akan disimpan di local cache.
- Langkah 5
  Hasil name server caching berguna untuk menghubungi salah satu com TLD server dan mengulangi pertanyaan tentang alamat yang dicari(www.example.com).
- Langkah 6
  Aeperti root server, server TLD juga tidak mengetahui, tetapi server TLD akan memasok pointer ke name server yang bertanggung jawab untuk nama domain example.com.
- Langkah 7
  Server caching kembali meminta informasi alamat untuk www.example.com.
- Langkah 8
  Hasil dari langkah 7 adalah server caching mendapat jawaban yang berisi informasi yang diminta
- Langkah 9
  Caching server dapat mengirim kembali respom ke *resolver library*.
- Langkah 10
  Aplikasi memiliki informasi yang dibutuhkan (misal : mengatur koneksi TCP ke www.example.com)

Merepresentasikan Informasi Ipv6 di DNS

Setiap permintaan DNS selalu melibatkan name server root,server TLD, name server tujuan, dan name server dari inisiator dan/atau ISP destinasi. Hal ini menyebabkan semua name server wajib ditingkatkan untuk mendukung Ipv6 agar dapat mencari alamat Ipv6 dalam DNS . Untungnya, hal ini tidak diperlukan. Pada tahun 1995, RFC 1886 menjelaskan cara yang sangat mudah untuk memplubikasikan informasi Ipv6 dalam DNS yang menyediakan jalur upgrade yang mudah. Namun, pada tahun 2000 diterbitkan RFC 2974. Ini merupakan mekanisme baru yang baru sebagian dilaksanakan. Pada tahun 2001, IETF mulai bergerak menjauh dari metode yang baru, dan pada tahun 2003, RFC 1886 diangkat kembali dan dilakukan sedikit perubahan (RFC 3596).

RFC 1886:AAAA and ip6.int

Pada Ipv4, alamat disimpan dalam record,dan pemetaan reverse dilakukan dengan membuat nama domain khusus yang terdiri dari nilai-nilai byte individu dalam alamat dengan urutan terbalik,diikuti dengan in-addr.arpa. Contoh:

Listing 5-1Alamat Ipv4 di DNS

```
www.example.com.          IN A   192.0.2.17
17.2.0.192.in-addr.arpa. IN PTR www.example.com.
```

Sedangkan alamat Ipv6 disimpan dalam AAAA (“quad A”). Reverse Mapping dilakukan dengan mengambil angka heksadesimal dari alamat Ipv6 dalam urutan terbalik dan menambahkan ip6.int,seperti pada Listing 5-2.

Listing 5-2. *An IPv6 Address in the DNS According to RFC 1886*

```
www.example.com. IN AAAA 2001:db8:1bff:c001::390
0.9.3.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.c.f.f.b.1.8.b.d.0.1.0.0.2.ip6.int. IN PTR ➥
www.example.com.
```

RFC 2874 : A6, DNAME Bitlabels, dan ip6.arpa

Masalah terbesar Ipv6 yang tidak terpecahkan adalah routing. Satu hal yang akan membuat situasi routing yang jauh lebih baik adalah penomoran ulang yang cepat. Jika penomoran ulang sederhana, orang akan lebih mungkin untuk menggunakan ruang alamat yang dapat dikumpulkan oleh ISP sehingga tabel routing tetap kecil.

Penamaan A6 sampai Pemetaan Alamat

Penggambaran dari Metode AAAA RFC 1886, yaitu hanya memberikan apa yang diminta saja. Lain halnya dengan RFC 2874, ia akan memberikan jalan melalui hirarki pengalamatan.Listing 5-3 menunjukkan hirarki A6 di DNS.

Listing 5-3. Rekaman A6 di DNS

```
www          IN  A6  64  ::0000:0000:0000:0390  subnet-a
subnet-a     IN  A6  48  0000:0000:0000:c001::  prefix-isp1
subnet-a     IN  A6  48  0000:0000:0000:c001::  prefix-isp2
prefix-isp1  IN  A6  0   2001:0db8:1bff::
prefix-isp2  IN  A6  0   3ffe:9500:003c::
```

Berbeda dengan contoh sebelumnya, semua nama domain pada Listing 5-3 relatif. Dengan asumsi mereka berada di zona file example.com, nameserver akan menambah .example.com setelah setiap nama. Setiap record A6 memberikan bagian dari alamat dan pointer ke mana sisa alamat dapat ditemukan. Catatan A6 pertama (satu untuk www) daun 64 bit kosong harus didefinisikan kemudian dan terus memberikan alamat IPv6 :: 390 untuk mengisi 128-64 = 64 bit yang tidak menentukan. Alamat bit ditentukan dalam catatan A6 di tangan yang disalin dari tempat masing-masing di alamat yang tercantum. Bagian dari alamat yang tidak ditentukan di sini harus diatur ke nol di alamat yang diberikan dalam file zona. Nama mengikuti alamat IPv6- seperti nilai dalam catatan poin A6 ke tempat dalam hirarki DNS dimana sisa alamat dapat ditemukan, dalam hal ini, dengan nama subnet-a. Dan memang, di bawah subnet-adalah rekor A6 lain atau, lebih tepatnya, dua di antaranya. Mereka berdua mengatur bit antara 48 dan 64 untuk C001 (sisa bit adalah nol) dan arahkan ke awalan-ISP1 dan awalan-ISP2, masing-masing, untuk sisa alamat. Berdasarkan nama-nama, bit yang tersisa 0-48 disediakan, dan pointer ke tempat lain tidak diperlukan, sebagai alamat sekarang lengkap. Karena subnet-memiliki dua pointer untuk atas 48 bit, hasil seluruh prosedur di dua alamat lengkap: 2001: db8: 1bff: C001 :: 390 dan 3ffe: 9500:3 c: C001 :: 390.
Dengan catatan A6, mengupdate DNS ketika ada penomoran ulang dengan mudah yaitu tidak harus mengubah semua alamat di semua domain untuk situs, tetapi hanya A6 tunggal yang harus diubah. Misalnya, jika 3ffe: 9500:3 c :: / 48 prefix pada Listing 5-1 itu harus diubah menjadi 2007:4580:73 :: / 48, ini hanya akan membutuhkan update awalan-ISP2 catatan, all catatan A6 yang mengarah ke hal itu otomatis mencerminkan informasi baru.

Bitlabel dan Pengalamatan DNAME sampai Memetakan Nama

RFC 2874 juga menentukan cara baru untuk melakukan reverse mapping dari alamat ke nama. Ini menggunakan dua mekanisme yang didefinisikan dalam RFC 2672 dan 2673, masing-masing: DNAME dan bitlabels. Rekor DNAME agak mirip dengan CNAME. Namun, tidak seperti CNAME yang hanya memberikan alias untuk satu nama, DNAME dapat memberikan alias untuk seluruh cabang dalam pohon DNS: domain atau subdomain. Listing 5-4 menunjukkan tindakan DNAME.

Listing 5-4. The DNAME Record

```
research.example.com          IN DNAME  r-and-d.example.com.
www.plastics.r-and-d.example.com.  IN A      192.0.2.1
www.biotech.r-and-d.example.com.   IN A      192.0.2.2
```

Dengan catatan DNAME berlaku,dan semua catatan subdomain dibawah r-dan-d.example.com juga hadir dalam research.example.com. Jadi mencari www.biotech.research.example.com memiliki hasil yang sama seperti mencari www.biotech.r-dan-d.example.com. Agar kompatibel, nameserver akan "mensintesis" data CNAME untuk informasi yang diminta, bersama dengan memberikan catatan DNAME sebenarnya. Tetapi beberapa resolver library yang lebih tua tidak akan menangani catatan DNAME dengan benar.

Gagasan di balik bitlabels (juga kadang-kadang disebut "binary labels") adalah bahwa tradisional ... 4.3.2.1.in-addr.arpa atau ... mekanisme delegasi eff3.ip6.int kurang sempurna karena hanya memungkinkan delegasi pada tanggal 8 - atau batas 4-bit, masing-masing. Jadi secara konseptual, sebuah bitlabel merupakan ekspresi dari sebuah nama domain yang sangat panjang dengan bit yang dipisahkan oleh periode. (Dalam sistem nama domain, data antara dua periode disebut "label.") Namun, dalam protokol DNS, bitlabel yang dinyatakan sebagai sepotong tunggal data biner, terlepas dari jumlah bit yang dikandungnya, daripada daftar panjang label ASCII individu. Dalam file zona DNS, bitlabels dapat ditentukan baikbiner, oktal, desimal, atau heksadesimal, dengan nilai eksplisit menunjukkan panjang dalam bit. Listing 5-5 menunjukkan beberapa bitlabel representasi dari informasi yang sama.

Listing 5-5. Bitlabels di File Zona

```
\[xf0d2b496785a3c1e]              IN PTR www.example.com.
\[b11110000110100101011010010010110011110000101101000111100000011110] IN PTR ➥
www.example.com.
\[o074151264454741321701 70/64]      IN PTR www.example.com.
\[120.90.60.30].\[240.210.180.150] IN PTR www.example.com.
```

Bit tabel pertama dalam heksadesimal,seperti dilambangkan dengan x awal. Yang kedua adalah dalam biner (b) dan yang ketiga adalah dalam octal (o). Karena salah satu oktal digit mewakili tiga bit, string 22-karakter biasanya akan menentukan 66 bit. Eksplisit / 64 menunjukkan bahwa hanya 64 bit harus dianggap sebagai bagian dari bitlabel tersebut. Baris terakhir adalah dalam notasi desimal bertitik-quad. Karena notasi ini terbatas pada 32 bit, kita perlu menggabungkan dua bitlabels untuk tiba di penuh 64 bit. Perhatikan bahwa meskipun dalam setiap bitlabel notasi paling signifikan-dengan-paling tidak-signifikan lebih wajar digunakan, nama sistem domain paling-signifikan-untuk-paling signifikan memesan label datang kembali ketika concatenating bitlabels. Jadi, jika 64-bit desimal bertitik-quads diizinkan, bahwa versi bitlabel pada Listing 5-5 akan terlihat seperti \[240.210.180.150.120.90.60.30]. Bersama-sama, DNAME dan bitlabels memungkinkan informasi pemetaan terbalik untuk didelegasikan seperti pada Listing 5-6.

Listing 5-6. Membalikkan Pemetaan dengan DNAME dan Bitlabels

```
\[x20010DB81BFF/48].ip6.arpa.  IN  DNAME  rev.example.com.
\[xC001/16].rev.example.com    IN  DNAME  srvrs.rev.example.com.
\[x0000000000000390/64].srvrs.rev.example.com. IN PTR www.example.com.
www.example.com.         IN  AAAA   2001:db8:1bff:c001::390
```

Dalam kehidupan nyata, baris pertama harus menjadi delegasi oleh ISP, sehingga akan berada di zona file ISP. Namun, jalur lain bisa semua berada di zona file yang sama (satu untuk example.com, misalnya), atau mereka dapat tersebar di beberapa zona untuk menambah fleksibilitas dan kemudahan penomoran ulang.

RFC 1886 vs. RFC 2874

Ketika sebuah host yang mengimplementasikan RFC 1886 pada alamat IPv6 dalam DNS, resolver library akan mengirimkan permintaan aaaa. Server DNS terakhir dalam rantai harus memahami apa yang ada, tetapi server DNS menengah (lihat Gambar 5-1) tidak, mereka hanya melihat jenis catatan sumber daya bahwa mereka tidak mengakui, tapi format karena format yang familiar sehingga mereka tahu bagaimana untuk memproses informasi. Mencari informasi terbalik dalam domain ip6.int bahkan lebih mudah, karena dengan nameserver, tidak ada yang khusus tentang domain ini.

Hal yang sama berlaku untuk memproses A6 record: ini semua dilakukan oleh resolver, jadi sekali lagi, nameserver di tengah tidak harus memahami semantik A6. Namun, untuk mendapatkan alamat dengan cara ini membutuhkan proses dan nameserver cukup terlibat, terutama jika pointer dari satu record A6 dengan lompatan berikutnya antara server yang berbeda dalam domain yang berbeda, atau ketika rantai pointer A6 sangat panjang. Fakta bahwa prosedur ini harus dijalankan oleh resolver dan bukan nameserver caching mengharuskan penulisan ulang yang besar pada perangkat lunak BIND dan penambahan daemon resolver. Resolver library tradisional tidak benar-benar siap untuk menangani tugas-tugas kompleks tersebut.

Meskipun dukungan penuh untuk catatan DNAME membutuhkan perubahan dengan caching nameserver serta nameserver hosting informasi DNAME, tambahan sintesis CNAME memungkinkan resolvers dimodifikasi dan caching nameserver untuk bekerja dengan DNAME. Hal-hal yang berbeda untuk bitlabels, namun. DNS query yang berisi bitlabels membutuhkan pengolahan yang berbeda dari pertanyaan yang hanya berisi label ASCII tradisional, sehingga selain resolver dan nameserver memegang informasi bitlabel, semua nameserver di antara (caching nameserver, akar, dan server TLD) harus memahami bitlabels.

RFC 3596: AAAA and ip6.arpa

Argumen utama dalam mendukung RFC 2874 adalah fleksibilitas dan dukungan untuk penomoran ulang cepat. Argumen melawan menggunakan catatan A6 untuk menyimpan alamat IPv6 dalam DNS, dan bitlabels untuk melakukan pemetaan terbalik, adalah bahwa mereka menambahkan kompleksitas dan meningkatkan waktu yang diperlukan untuk mencari informasi (jika itu tersebar di beberapa nameserver). RFC 3364 juga mencatat bahwa A6 record yang "dioptimalkan untuk menulis" ataupun "membaca" informasi DNS jauh lebih sering daripada mengubahnya. Ini juga sulit dengan membayangkan cara di mana informasi dalam DNS akan tetap sinkron dengan alamat yang sebenarnya digunakan oleh host selama acara penomoran ulang. Akhirnya, hal ini menyebabkan kesimpulan bahwa catatan AAAA akan menjadi cara terbaik untuk menyimpan alamat IPv6 dalam DNS, dan metode bite merupakan cara yang lebih disukai untuk melakukan pemetaan terbalik. Namun, dalam waktu yang berarti, pada tahun 2001, Internet Architecture Board (IAB) telah menerbitkan RFC 3172, yang menyatakan preferensi arah. ARPA (sekarang "Alamat dan Routing Parameter area") sebagai infrastruktur domain tingkat atas, rumit kembali lengkap RFC 1886 dan ip6.int. Selanjutnya, penggunaan ip6.int itu "ditinggalkan" dalam mendukung ip6.arpa di Best Current Practice (BCP) Dokumen 49, juga dikenal sebagai RFC 3152. Semua ini mencapai puncaknya pada RFC 3596 (2003), yang standarisasi penggunaan aaaa record dan metode bite untuk reverse lookup di bawah ip6.arpa.

Instalasi dan Konfigurasi BIND

Kebanyakan sistem operasi UNIX tidak perlu melakukan install BIND karena pada umumnya BIND versi 9.x dan versi 9.2.x disertakan dalam suatu sistem . BIND terdiri dari sejumlah program utama dan fitur pendukung. Untuk program utama berupa biner. Ini merupakan daemon nama server yang sebenarnya.

Instalasi BIND

Untuk mendapatkan software BIND versi yang lain, dapat diakses pada website resmi ISC(Internet Software Consortium) dengan alamat http://www.isc.org/sw/bind/. Untuk pendukung penuh IPV6, termasuk transportasi IPV6, maka harus memilih salah satu BIND versi 9.x. Pada BIND versi 9.2.x digunakan untuk bit label dan pendukung A6. Sebuah distribusi biner software BIND tersedia untuk Windows yang ada seperti windows NT, 2000, XP dan 2003. Distribusi tersebut belum mendukung transportasi IPv6. Jika anda ingin melakukan instalasi BIND pada Windows, maka harus membaca file readme1st.txt dengan baik. Untuk file tersebut terletak pada lokasi yang berbeda, jika tidak, BIND pada windows tersebut hampir sama dengan BIND pada sistem operasi UNIX.

Memulai BIND pada Boot Time

Pada **Red Hat Linux** sudah terdapat skrip startup untuk named, tetapi tidak diaktifkan secara default. Halinidapatdilakukandenganmenampilkan (sebagairoot)yang bernama perintah chkconfig , yang mana dilakukan untukmenciptakanlinksimbolikyang diperlukanuntuk/etc/ init.d/named Script.Tidak heran apabila chkconfig dalam keadaanoff akan menghapuslinkdanberhentidarinama yang dimulaipadastart upsistem. Perintah chkconfig- listyang mempunyai skripstart upakan dieksekusi ketikamengubah run level.
Pada sistem operasi FreeBSDmulaidinamaipadasaatbootdilakukan, yaitu denganmenambahkanduabariske / etc/rc.conf,sepertiditunjukkanpadaListing 5-7. Adabaiknya, jika named daemon diletakkan pada /usr /sbin/.

Listing5-7. Mengaktifkan nameddi/etc/rc.confpadaOSFreeBSD

```
named_program=”/usr/sbin/named”
named_enable=”YES”
```

KonfigurasiBIND

Semua pilihan konfigurasi yangluas pada BIND, yang m a n a dijelaskan padaBIND Administrator ReferenceManualdengansumberatau distribusibiner. BINDtidaksulituntuk dijalankan hanya saja perlumengetik mulai *named*daemon.Karenakebutuhan*named* diakseske TCPdanUDPport53, harus dijalankan(setidaknyaseperti awalnya)sebagairoot. Sebuah metode alternatifuntuk mengendalikan *named* serveryaitu denganmengatur jarak jauh *named*daemon pada programrndc.rndcterhubung ke *named*melaluiTCP,sehinggaperintah tersebut jugadapatdigunakanuntukmengontrol jarakjauh *named*server,sesuainamanya.Secaradefault,*named*melakukankoneksiinputdari rndcpadaport953dilocalhostalamatIPv4danIPv6,tetapihalini dapatdiubahdengan perintah konfigurasikontroldifilekonfigurasi*named*.rndcmengharapkansebuahfile konfigurasidi/ etc/rndc.conf, tetapi itu jauhlebihmudahhanyauntukmengeksekusirndc-confgenuntukmembuatfile/ etc/rndc.key. Lokasidirektori*named*danfile*named*.rootharus terdaftardalamfilekonfigurasibernama itu.Listing5-8 menunjukkan file dasarnamed.conf.

Listing5-8. Isifile/etc/named.conf

```
Options {
        directory “/var/named”;
        allow-recursion { 192.0.2.0/24;
 2001:db8:1bff::/48;} ;
        listen-on {192.0.2.106;}
        listen-on-v6 { any; };

# forward first;
# forwarders { 192.0.2.53; };
};
```

```
zone "." {
        type hint;
        file "named.root";
};

zone "0.0.12.IN-ADDR.ARPA"{
        type master;
        file "localhost.rev";
};

zone "example.com" {
        type slave;
        file "example.com";
        masters {192.0.2.53; };
};

zone "0.0.0.0.f.f.b.1.8.b.d.0.1.0.0.2.ip6.arpa."
        {
                type master;
                file "db.2001:db8:1bff:0";
};
```

Filedimulai dengandirektifpilihan,diikutidenganbeberapapilihan.Koma mengakhiripernyataanatau itemdalamdaftar.Opsipertama menentukandirektori*named*untukmencarifile. Pilihan memungkinkan-rekursimendefinisikan*named* klien yang akan melakukan query rekursif. Dalam kasus ini, klien dengan alamat awalan 192.0.2.0/24 dan 2001:db8:1bff ::/48. Meskipun tidakadasalahnya langsungdalam kemungkinan permintaan rekursif untuk seluruh Internet,dan memungkinkan untuk debugging lebihmudah, mendapatbiayabandwidthekstra,pengolahanoverheaddanmemori jika sisainternetmulaimenggunakanserversecaramassal.Banyakmasalahkeamanan yang ditemukandiBINDselamabertahun-tahun, yang mana hanyabisadimanfaatkanolehorang-orang maupun server mana yang akanmelakukanquery rekursif. JikaAndainginmembatasihal-haltertentuke localhost dan menyadari bahwa localhost hanya kata kunci yang berarti pada alamat localhost IPv4di filenamed.conf.AlamatlocalhostIPv6 harusterdaftarsecaraeksplisitsebagai::1,jika hal tersebut diinginkan.

Duabarisberikutnyaadalahkomentar.Selain komentar dalam tipeshell,*named*juga menerima komentartipeCdanC++,tetapitidak menerimatitikkomasebagaiawalkomentar,seperti dalamsebuahfilezona. Selanjutnya, ada empatzonaspesifikasi yaitu :

1. Zona "Dot"adalah "petunjuk"zonadan poin kefilenamed.root.
2. Zona0.0.127.in-addr.arpaadalahzonareverseuntukalamatlocalhost,dan,menjadi zonautama, informasiotoritatifterdapatdalamfilelocalhost.rev.
3. Zonaexample.comadalahzonaslave,dan dataotoritatifditransfersecaraberkaladari nameserver dialamat192.0.2.53 dan disimpan dalamfileexample.com.
4. Zona terakhiradalah zona ip6.arpanibble-styleuntuk 2001:db8:1bff::/48.

MenambahkanInformasi IPv6ke File Zona

Sebelummenjalankan tekseditor favoritAndadanmulaimenambahkancatatan AAAAke semuafilezona. Terlebihdahulu Andaharusmempertimbangkanimplikasi.Ketika alamatIPv6 host terdaftar diDNS, aplikasiIPv6 padahost lain IPv6 umumnyaakan lebih

memilihuntukterhubung melaluiIPv6bukanIPv4.Ketikalebih menghubungkan, IPv6tidak bekerja, aplikasimungkinatautidak mungkin jatuhkembalipadaIPv4.Sayangnya, konektivitas IPv6 masih sering lambat dan kurang dapat diandalkan dibandingkan konektivitasIPv4.

Untukdownloadfile melaluiHTTPatauFTP merupakanideyangbaiksecaraeksplisitmendaftaralamatIPv4dan IPv6,sehinggaorangdapatmemilih,sepertitransferfileyang merupakan salahsatuaplikasiyang palingrentan terhadap keterbatasan bandwidth.

Jikaserveryang adatidakbisamenanganiIPv6,mungkinperluuntukmendirikansebuah serveryangberbedaatauclusterserveruntukmemberikanlayananyang adamelaluiIPv6. Kemudian,Anda menunjuk satu atau lebih catatan AkeserverIPv4 atau server, dan satu atau lebihcatatanAAAA keserverIPv6atauserver lain.

Perekaman AAAA

Listing 5-9 menunjukkan zona file catatan AAAA digunakan berbeda untuk layanan yang brbedaberbeda.

Listing 5-9. Sebuah Zona dengan record AAAA

```
;   20041215    IvB created
;   20050209    IvB added AAAA records

$TTL 86400

@   IN  SOA ns1.example.com. root.example.com. (
            2005020900      ; Serial
            28800           ; Refresh (8 hours)
            7200            ; Retry (2 hours)
            604800          ; Expire (7 days)
            86400 )         ; Minimum (1 day)

            IN  NS      ns1.example.com.
            IN  NS      ns2.beispiel.de.

            IN  MX      100 smtp.example.com.
            IN  MX      200 smtp.ipv4.example.com.

            IN  A       192.0.2.80
            IN  AAAA    2001:db8:31:1:201:2ff:fe29:2640

ns1         IN  A       192.0.2.80
            IN  AAAA    2001:db8:31:53::53
            IN  A6      0 2001:db8:31:53::53

www         IN  A       192.0.2.80
www         IN  AAAA    2001:db8:31:1:201:2ff:fe29:2640
www.ipv4    IN  A       192.0.2.80
www.ipv6    IN  AAAA    2001:db8:31:1:201:2ff:fe29:2640

smtp        IN  A       192.0.2.25
smtp        IN  AAAA    2001:db8:31:1:20a:95ff:fecd:987a
smtp.ipv4   IN  A       192.0.2.25

pop         IN  A       192.0.2.25
popv4v6     IN  A       192.0.2.25
popv4v6     IN  AAAA    2001:db8:31:1:20a:95ff:fecd:987a
```

Di "awal otoritas" (SOA) record pertama berisi daftar nama untuk nameserver utama untuk zona ini (ns1.example.com) dan alamat email kontak dengan tanda at digantikan oleh periode (alamat

email root@example.com). Catatan SOA berlanjut pada lima baris berikutnya sampai kurung tutup.

Nama www memiliki kedua IPv4 dan alamat IPv6 dan dilengkapi dengan IPv4 dan IPv6-versi hanya (www.ipv4 dan www.ipv6, masing-masing). Memiliki nama IPv6-hanya berguna untuk tes reachability IPv6 cepat: jika beban halaman, IPv6 diaktifkan dan bekerja. layanan POP adalah salah satu yang penting, dan itu sulit bagi pengguna email untuk sementara mengubah alamat untuk server POP mereka ketika ada masalah konektivitas IPv6, nama yang bersangkutan untuk layanan ini hanya memiliki satu alamat IPv4. Sintaks dari file zona dan file konfigurasi sebelum reload nameserver, seperti pada Listing 5-10.

```
# named-checkzone example.com /var/named/example.com
zone example.com/IN: loaded serial 2005020900
OK
# named-checkconf
# rndc reload
```

Reverse Mapping

Zona terbalik pemetaan yang pada umumnya sama dengan zona biasa, kecuali bahwa mereka hanya berisi catatan PTR, kecuali untuk SOA awal dan NS catatan.Zona terbalik pemetaan yang pada umumnya sama dengan zona biasa, kecuali bahwa mereka hanya berisi catatan PTR, kecuali untuk SOA awal dan NS catatan. Listing 5-11 menunjukkan perintah host dan Pencatatan 5-12 file zona reverse yang dihasilkan.

```
host 2001:db8:31:1:201:2ff:fe29:2640
Host 0.4.6.2.9.2.e.f.f.f.2.0.1.0.2.0.1.0.0.0.1.3.0.0.8.b.d.0.1.0.0.2.ip6.arpa not ➥
found: 3(NXDOMAIN)
```

Jika perintah host mencoba untuk melakukan query bitlabel, program ini datang dengan BIND versi 9.x sebelum 9.3. Gunakan host-n sebagai gantinya, dan ia akan melakukan permintaan ip6.int.

```
;   20050209    IvB     created

$TTL 86400

@   IN  SOA ns1.example.com. root.example.com. ( 2005020900 28800 7200 604800 ➥
86400 )

            IN  NS      ns1.example.com.
            IN  NS      ns2.beispiel.de.

$ORIGIN 3.5.0.0.1.3.0.0.8.b.d.0.1.0.0.2.ip6.arpa.
; ns1       IN  AAAA    2001:db8:31:53::53
3.5.0.0.0.0.0.0.0.0.0.0.0.0.0.0  IN  PTR     ns1.example.com.

$ORIGIN 1.0.0.0.1.3.0.0.8.b.d.0.1.0.0.2.ip6.arpa.
; www       IN  AAAA    2001:db8:31:1:201:2ff:fe29:2640
0.4.6.2.9.2.e.f.f.f.2.0.1.0.2.0  IN  PTR     www.example.com.

; smtp      IN  AAAA    2001:db8:31:1:20a:95ff:fecd:987a
a.7.8.9.d.c.e.f.f.f.5.9.a.0.2.0  IN  PTR     smtp.example.com.

$ORIGIN 1.3.0.0.8.b.d.0.1.0.0.2.ip6.arpa.
0.0.2.c     IN  NS      ns.research.example.com.
```

RFC 1886 and 2874 Reverse Mapping Hacks

Beberapa perpustakaan penyelesai gagal dalam cara yang jelek karena bitlabel ip6.arpa informasi yang mereka cari tidak ada. Untuk menghindari masalah ini, Anda mungkin ingin mengatur informasi pemetaan palsu sebaliknya bagi mereka. Hal ini dilakukan dalam Daftar 5-13 (zona bitlabel) dan 5-14 (bagian yang relevan dari named.conf).

```
$TTL 86400

@   IN  SOA ns1.example.com. root.example.com. ( 2005020900 28800 7200 604800 ➥
86400 )
                    IN  NS      ns1.example.com.

*.\[x2/3].ip6.arpa. IN  PTR     bit.label.ip6.arpa.

\[x20010db800310001020a95fffecd987a/128].ip6.arpa. IN CNAME a.7.8.9.d.c.e.f.f.f.5. ➥
9.a.0.2.0.1.0.0.0.1.3.0.0.8.b.d.0.1.0.0.2.ip6.arpa.

zone "\[x2/3].ip6.arpa." {
  type master;
  file "bitlabel.ip6.arpa";
};

zone "ip6.int." {
  type master;
  file "ip6.int";
};

$TTL 86400

@   IN  SOA ns1.example.com. root.example.com. ( 2005020900 28800 7200 604800 ➥
86400 )
        IN  NS      ns1.example.com.

@       IN  DNAME   ip6.arpa.
```

Dynamic DNS Updates

RFC 2136 memperkenalkan konsep "update DNS dinamis." Mekanisme ini memungkinkan klien untuk meminta server otoritatif untuk menambahkan informasi ke zona atau menghapus informasi yang ada dari zona tersebut. Mekanisme update dinamis memungkinkan host yang menerima alamat baru melalui DHCP atau autoconfiguration stateless untuk memperbarui catatan DNS mereka sendiri sehingga mereka tetap bisa dicapai dengan nama mereka, meskipun perubahan alamat.

Untuk alasan yang jelas, itu tidak mungkin untuk hanya klien untuk memodifikasi setiap dan semua zona. Daftar pengguna yang berwenang mungkin dalam bentuk rentang alamat IP, atau mungkin menentukan satu atau lebih kunci yang melindungi update. Ketika zona diatur untuk update dinamis, bernama mengambil kendali dari zona file dan itu tidak mungkin lagi untuk mengedit file tanpa terlebih dahulu mematikan bernama daemon.