インターネットのツボを押さえる!

# DNSの仕組み完全解説

1

馬場達也

## DNSの役割と仕組み

コンピュータのホスト名とIPアドレスとを対応づけるDNS（Domain Name System）は、現在のインターネットで欠かすことのできない重要なインフラである。しかし、DNSの仕組みは必ずしも正しく理解されているとはいえないのが現状だ。本連載では、DNSの仕組みと機能について具体例を交えながら紹介する。そして、管理者がDNSを正しく安全に運用できるようになることを目標とする。連載1回目の今回は、DNSの役割と仕組み、DNSを実現するソフトウェアを紹介しよう。

### ホスト名をアドレスに変換するのがDNSの役割

インターネットの世界では、相手のコンピュータを識別するために、「192.168.0.20」のような32ビット（IPv4の場合）のIPアドレスが使用される。しかし、アクセスする先のコンピュータのIPアドレスをいくつも覚えるのは簡単ではない。さらにIPv6ではIPアドレスが128ビットに拡張されており、覚えるという選択肢はないに等しい。そこで、通常は、「192.168.0.20」のようなIPアドレスではなく、「www.example.com」のような人間が覚えやすい名称を使ってコンピュータにアクセスする。このようなコンピュータの名称はFQDN（Fully Qualified Domain Name）と呼ばれており、ホスト名（この例でいうと「www」の部分）に、ホストが属するドメイン名（「example.com」の部分）を組み合わせて表記する（今後は、断りがないかぎりFQDNをホスト名と呼ぶことにする）。しかし、インターネットでは、IPアドレスを使用してパケットを配送するため、結局、このホスト名をIPアドレスに変換しなければならない。DNS（Domain Name System）は、このホスト名（FQDN）とIPアドレスとの対応を管理する重要なシステムである。

DNSは、基本的に「ホスト名をIPアドレスに変換する」機能を提供するシステムである。ホスト名からIPアドレスを得ることを「名前解決」と呼ぶが、DNSには、逆にIPアドレスからホスト名を得るための機能や、ある組織のユーザーにメールを送る際に、その組織のメールサーバのホスト名とIPアドレスを教えてくれる機能もあり、DNSの役割は名前解決の範囲にとどまらない。DNSは、インターネットで使用するアプリケーションのすべてが使用しているといっても過言ではない、インターネットのインフラとも言うべき重要なシステムだ。いくらWebサーバやメールサーバをきちんと管理していても、DNSがダウンしてしまったら、これらのサービスは利用できなくなってしまう。

### 分散データベースとして機能するDNS

インターネットにDNSが導入されたのは1984年のことで、まだインターネット上のホストの数が1,000程度しかなかった時代である。以来、インターネットの規模は爆発的に増加し続け、NetNames International Ltd.の調査（http://www.domainstats.com/）によると、現在登録されている汎用ドメイン名の数は3,000万を超える。この規模でDNSが動作しているのは奇跡であるとも言われている。

DNSの基本的な仕様は、RFC 1034、1035に記述されている。DNSは、「ルートネームサーバ」を頂点としたツリー状の分散データベースシステムとなっており、ネットワークを管理する組織が、それぞれ、自組織に関する情報を格納した「ネームサーバ」を管理している。ホスト名に対応するIPアドレスを知りたい場合は、まずルートネームサーバに問い合わせれば、そのホスト名とIPアドレスの対応を管理している組織のネームサーバまでたどりつくことができる仕組みになっている。

このルートネームサーバは、障害などによって完全に停止することがないように13台設置されている。そのうちの1台が東京にあり、ロンドンとストックホルムに1台ずつ、そして残りの10台が米国にある（表1）。このルートネームサーバの一覧は、ftp://ftp.rs.internic.net /domain/named.rootから入手できる。

それぞれのネームサーバは、「ゾーン」と呼ばれる範囲の情報（これをゾーンデータという）を管理しており、

| ルートネームサーバ | 公表所在地 | 運用組織 |
|---|---|---|
| a.root-servers.net | 米国バージニア州 | VeriSign |
| b.root-servers.net | 米国カリフォルニア州 | 南カリフォルニア大学情報科学研究所（ISI） |
| c.root-servers.net | 米国バージニア州 | Cogent Communications,Inc.（旧PSINet） |
| d.root-servers.net | 米国メリーランド州 | メリーランド大学 |
| e.root-servers.net | 米国カリフォルニア州 | 米国航空宇宙局（NASA） |
| f.root-servers.net | 米国カリフォルニア州 | Internet Software Consortium（ISC） |
| g.root-servers.net | 米国バージニア州 | 米国国防情報システム局（DISA） |
| h.root-servers.net | 米国メリーランド州 | 米国陸軍研究所（ARL） |
| i.root-servers.net | ストックホルム | NORDUnet |
| j.root-servers.net | 米国バージニア州 | VeriSign Global Registry Services |
| k.root-servers.net | ロンドン | RIPE NCC |
| l.root-servers.net | 米国カリフォルニア州 | 南カリフォルニア大学情報科学研究所（ISI） |
| m.root-servers.net | 東京 | WIDEプロジェクト |

表1● ルートネームサーバの所在地と運用組織

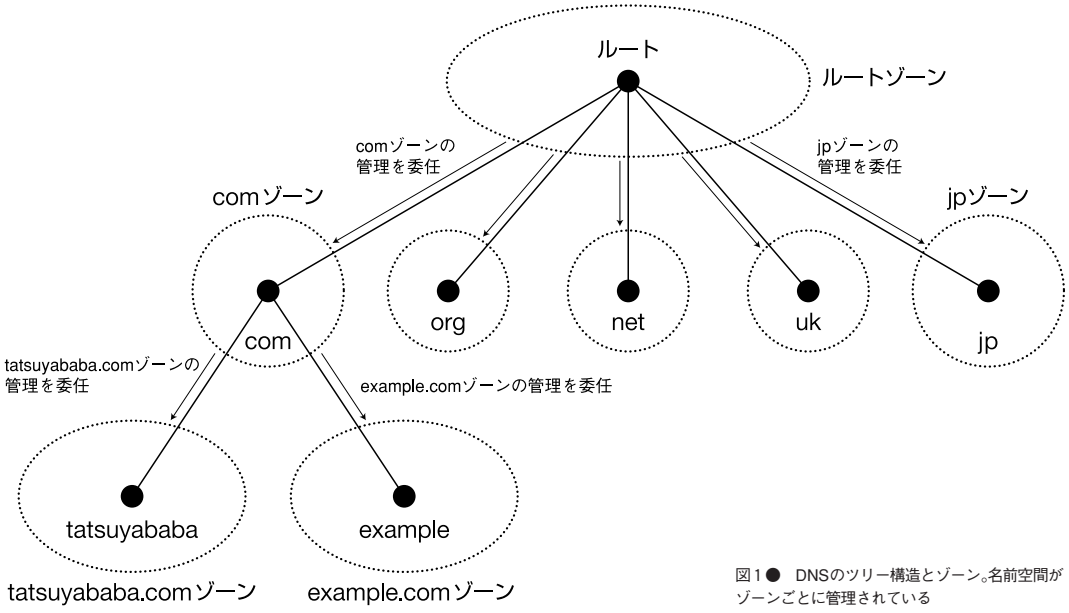



図1● DNSのツリー構造とゾーン。名前空間がゾーンごとに管理されている

そのゾーンに関する権威（authority）を持っている。例えば、ルートネームサーバは、図1のルートゾーンを管理しており、ルートゾーンに関する権威を持っている。このルートゾーンのゾーンデータには、com、net、orgなどのgTLD（Generic Top Level Domain）やjp、ukなどの国別のccTLD（Country Code Top Level Domain）などのトップレベルドメインのゾーンを管理するネームサーバの名前とIPアドレスが書かれており、各ゾーンの管理をほかのネームサーバに任せている（これは通例的に「委任する」と表現される）。

例えば、VeriSignの管理するgTLDネームサーバは、com、net、orgの3つのゾーンの管理をルートネームサーバから委任され、これらのゾーンに関する権威を持っている。そして、さらにcomゾーンを管理するgTLDネームサーバは、tatsuyababa.comやexample.comといった下位レベルのゾーンの管理をほかのネームサーバに委任している。このように、DNSでは、名前空間をゾーンという単位に区切って分散管理している。ちなみにルートネームサーバからjpゾーンの管理を委任されたネームサーバは6台あり、JPNIC（日本ネットワークインフォメーションセンター）およびJPRS（日本レジストリサービス）によって2台（ns0.nic.ad.jpとns-jp.nic.ad.jp）が、IIJ（ns0.iij.ad.jp）、国立情報学研究所（ns-jp.sinet.ad.jp）、JENS（dns0.spin.ad.jp）、WIDEプロジェクト（ns.wide.ad.jp）によってそれぞれ1台ずつが運用されている。


## リゾルバとネームサーバで構成されるDNS

それでは、実際にDNSがどのように動作しているのかを説明しよう。まず、DNSを構成する要素を紹介する。DNSは、リゾルバとネームサーバ（ローカルネームサーバおよび権威ネームサーバ）によって構成され

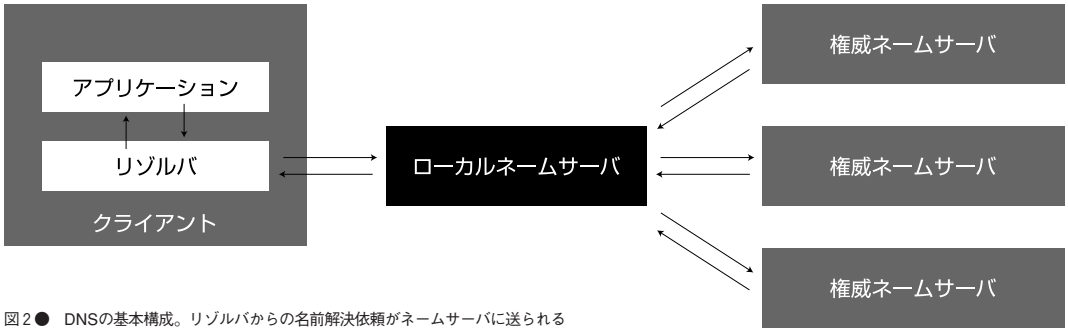



図2● DNSの基本構成。リゾルバからの名前解決依頼がネームサーバに送られる

ている（図2）。

**■リゾルバ**

リゾルバは、ネームサーバにアクセスして名前解決を行うクライアント上のプログラムである。名前解決を必要とするアプリケーションは、まず、リゾルバに対して名前解決を依頼する。すると、リゾルバがネームサーバに対して名前解決の問い合わせを行う。ただし、多くのリゾルバは、自分自身だけで名前解決を行う機能はなく、あらかじめ指定されたローカルネームサーバに対して名前解決を依頼するだけである。このようなリゾルバは、特に「スタブリゾルバ」と呼ばれる。

**■ネームサーバ**

ネームサーバ（「ドメインネームサーバ」や「DNSサーバ」と呼ばれることも多い）は、リゾルバやほかのネームサーバからの名前解決の問い合わせをポート53番で待ち受ける機能を持つサーバである。ネームサーバは、その役割に応じて、権威ネームサーバやローカルネームサーバなどと呼ばれる。

権威ネームサーバ（authoritative name server）は、名前空間の中のあるゾーンを管理するネームサーバである。権威ネームサーバは、ゾーンデータのマスタファイルを管理するプライマリマスタと、プライマリマスタが保持するゾーンデータをコピーして管理するセカンダリマスタ（スレーブともいう）に分けられる。例えば、13台あるルートネームサーバのうち、a.root-servers.netがプライマリマスタで、ほかの12台のルートネームサーバがセカンダリマスタとなっている。つまり、新しいトップレベルドメインができたら、プライマリマスタであるa.root-servers.netのゾーンデータを更新し、ほかの12台のセカンダリマスタは、a.root-servers.netから更新されたゾーンデータを転送してもらい、そのコピーを管理するのである（このように、ゾーンデータをプライマリマスタからセカンダリマスタに転送することを「ゾーン転送」と呼ぶ）。同じゾーンを管理するネームサーバを複数台設置することにより、問い合わせの負荷を分散させたり、そのうちの1台が障害などによりダウンしても、ほかのネームサーバを使用してサービスを継続できるようにしているのである。外から見た場合は、プライマリマスタもセカンダリマスタも同等の権威を持ったネームサーバであることに変わりはなく、クライアントが区別する必要はない。

ローカルネームサーバは、クライアントから最も近い位置にあるネームサーバで、直接リゾルバから名前解決の依頼を受けるネームサーバである。ローカルネームサーバは、リゾルバからの名前解決の依頼を受けると、自身でその情報を持っていない場合には、リゾルバの代わりに名前解決の責任を負ってほかのネームサーバに対して問い合わせを行う。ローカルネームサーバは、クライアントが属するゾーンを管理する権威ネームサーバであることが多いが、ゾーンを管理せず、ほかのネームサーバに問い合わせた結果をキャッシュするだけのキャッシュネームサーバであってもよい（後述するdjbdnsはこの形態である）。


## DNSの問い合わせの仕組みを知る

それでは、www.example.comというWebサーバにアクセスする場合を、図3を例に説明しよう。Webブラウザに URLとして「http://www.example.com/」と入力してアクセスしようとすると、まず、Webブラウザからリゾルバに向けて名前解決の依頼が発生する。リゾルバは、あらかじめ設定された近くのローカルネームサーバに対して、名前解決の依頼を行う（①）。通常、クライアントのリゾルバは名前解決をすべてローカルネームサーバに任せ、自身はほかのネームサーバに問い合わせることはしない。このような問い合わせ形態は「再帰問い合わせ」と呼ばれる。もしこのローカルネームサーバにwww.example.comに対するIP

アドレスの情報があれば、ここでIPアドレスが判明するので、その結果がクライアントに返され、名前解決が完了する。しかし、ローカルネームサーバに情報がない場合は、ほかのネームサーバに問い合わせを行う必要がある。

この場合、ローカルネームサーバは、最初にルートゾーンを管理している13台のルートネームサーバのうちの1台に対して問い合わせを送る(②)。この13台のルートネームサーバのIPアドレスは、あらかじめローカルネームサーバに設定されている。すると、ルートネームサーバは、comゾーンを管理しているネームサーバのホスト名とIPアドレスをいくつか(現在は13台)返してくる (③)。ローカルネームサーバは、comゾーンを管理しているネームサーバのうちの1台を選択して、再びwww.example.comのIPアドレスを問い合わせる(④)。すると、comゾーンを管理しているネームサーバは、example.comゾーンを管理しているネームサーバのホスト名とIPアドレスを返答する(⑤)。そこで、ローカルネームサーバは、example.comゾーンを管理しているネームサーバに対して再び問い合わせを行う(⑥)。example.comゾーンを管理しているネームサーバは、www.example.comのIPアドレスを管理しているため、そのIPアドレスを返答する(⑦)。ローカルネームサーバは、判明したwww.example .comのIPアドレスを問い合わせ元であるクライアントのリゾルバに返答する(⑧)。このように、リゾルバからの再帰問い合わせを受け付けたローカルネームサーバは、その名前解決の結果を得るまで、ルートネームサーバからDNSツリーをたどりながらさまざまなネームサーバに問い合わせを行う。このような問い合わせ形態を「反復問い合わせ」と呼ぶ。最後に、Webブラウザは、判明したIPアドレスを使用して、Webサーバに対してアクセスを行うわけである (⑨、⑩)。

しかし、このような問い合わせを毎回行っていると、DNSのトラフィックだけでネットワークが混雑してしまう。そこで、特に、リゾルバからの再帰問い合わせを受け付けるローカルネームサーバなどでは、名前解決の結果を一定期間キャッシュする機能を持っている。つまり、一度問い合わせを行えば、その情報はローカルネームサーバにキャッシュされるため、次回は同じ問い合わせに対して再びルートネームサーバから順に問い合わせをする必要がなくなるのである。このキャッシュの有効期間は、ゾーンデータ中でTTL(Time To Live)として設定することができる。

また、問い合わせた情報が存在しなかった場合に、再び同じ問い合わせを行わないようにするために、RFC 2308で規定されている「ネガティブキャッシュ」という機能も実装されている。ある問い合わせに失敗した場合に「情報が存在しない」ことをキャッシュすることで、次回以降同じ問い合わせを行って失敗する

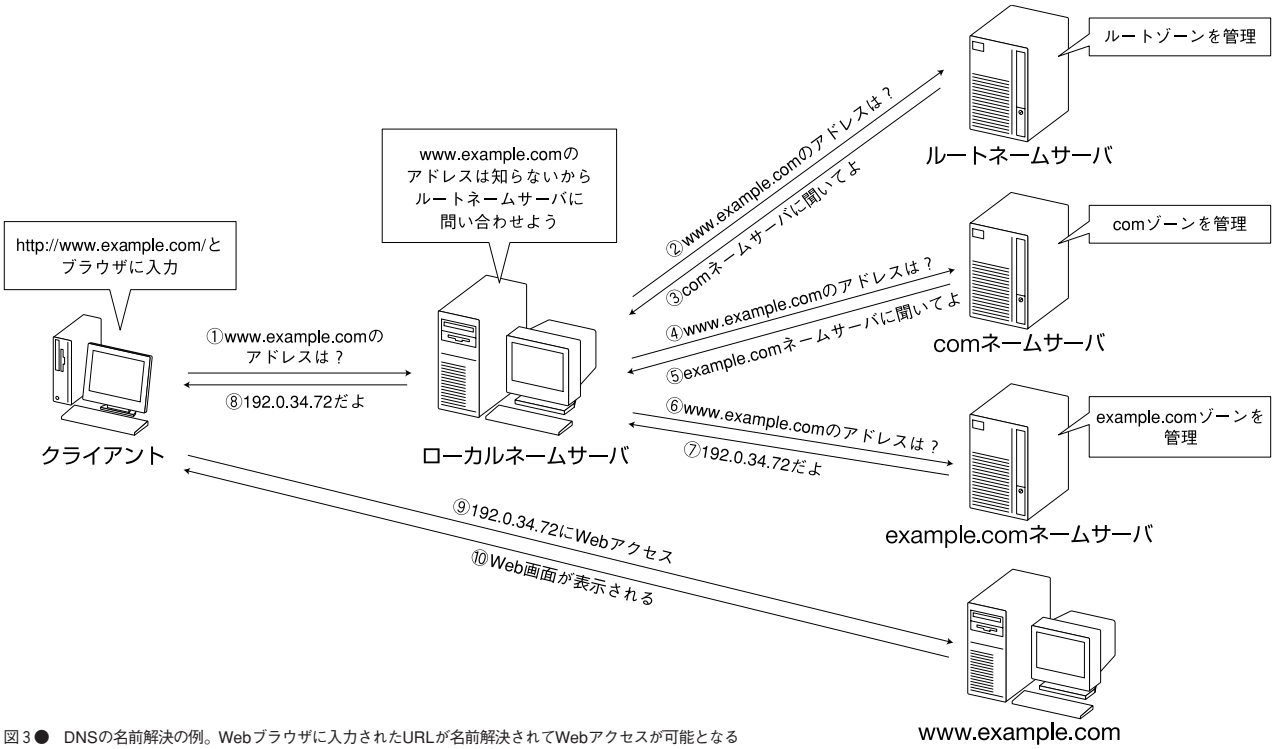



図3● DNSの名前解決の例。Webブラウザに入力されたURLが名前解決されてWebアクセスが可能となる

のを回避できる。

## DNSの機能を提供するソフトウェア

それでは、これまでに紹介したDNSの機能を提供するソフトウェアを紹介しよう。

### ■リゾルバ

リゾルバは、通常OS付属のライブラリとして提供されているため、特別な機能を持ったリゾルバが必要でないかぎり、ユーザーがインストールする必要はない。ユーザーは、リゾルバが再帰問い合わせを発行するローカルネームサーバのIPアドレスをあらかじめ設定しておくだけでよい。

この設定は、Unix系OSでは「/etc/resolv.conf」ファイルでリスト1のように行う。

Windows2000の場合は、スタートメニューから「設定」→「ネットワークとダイヤルアップ接続」で、「ローカルエリア接続」（LAN接続の場合）の「プロパティ」を開き、さらに「インターネットプロトコル（TCP/IP）」のプロパティを開くと、ローカルネームサーバのIPアドレスを設定できる（画面1）。ここでは、「優先DNSサーバ」で、通常使用するローカルネームサーバのIPアドレスを設定し、「代替DNSサーバ」で、通常使用するローカルネームサーバがダウンしていた場合に使用するネームサーバのIPアドレスを設定する。

### ■ネームサーバ

ネームサーバを構築するためには、専用のソフトウェアが必要である。現在、世界中で最も普及しているのがISC（Internet Software Consortium）がメンテナンスしている「BIND」である。そのほか、最近注目されてきた「djbdns」や、WindowsNT/2000 Serverに付属している「Microsoft DNS Server」（以下、本稿ではWindowsNT/2000 Serverに付属するDNSサーバプログラムをこう呼ぶ）などがある。

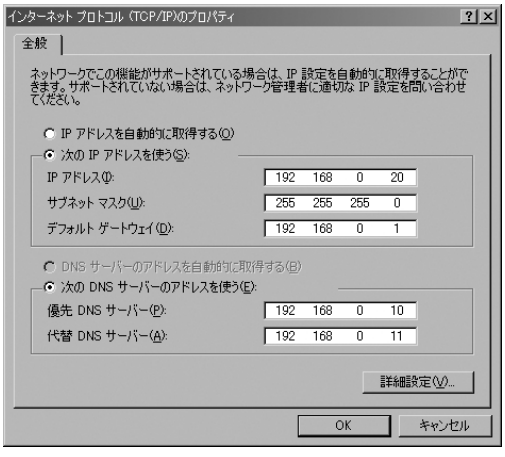
代表的なネームサーバソフトウェアの特徴を表2に示す。

#### ・BIND

BIND（Berkeley Internet Name Domain）は、4.3BSD Unix用に開発されて以来、多くのサイトで使用されてきた、現在最も普及しているオープンソースのDNS実装である。13あるルートネームサーバでもすべ

```
$ cat /etc/resolv.conf
domain example.com          ← 自ホストが所属しているドメイン名
nameserver 192.168.0.10     ← 通常使用するローカルネームサーバのIPアドレス
nameserver 192.168.0.11     ← 上記のサーバがダウンしていた場合に使用するローカルネームサーバのIPアドレス
```

リスト1● Unix系OSのリゾルバ設定は/etc/resolv.confファイルで行う




画面1● Windows2000では、リゾルバの設定は「インターネットプロトコル（TCP/IP）」のプロパティで行う

| | BIND | djbdns | Microsoft DNS Server |
|---|---|---|---|
| 最新バージョン | 9.2.1 | 1.05 | Windows2000 Server付属 |
| 開発元 | ISC | D.J.Bernstein | マイクロソフト |
| TSIG | ○ | × | ○ |
| DNSSEC | ○ | × | × |
| ダイナミックアップデート | ○ | × | ○ |
| 差分ゾーン転送 | ○ | △（独自方式） | ○ |
| IPv6（AAAAレコード） | ○ | ×（※） | ○ |
| ネガティブキャッシュ | ○ | ○ | ○ |
| 対応OS | Linux、FreeBSD、SolarisなどのUnix系OSおよびWindowsNT/2000 | Linux、FreeBSD、SolarisなどのUnix系OS | WindowsNT/2000 Server |

表2● 主なネームサーバの実装の比較

※第三者により提供されているパッチにより対応可能（http://www.fefe.de/dns/）

てこのBINDが使用されており、商用のネームサーバプログラムもBINDを基に作られているものが多い。

BINDの最新バージョンであるBIND 9では、DNSSEC（DNS Security Extensions）やTSIG（Transaction Signatures）などのセキュリティ機能を実装し、IPv6、ダイナミックアップデート、差分ゾーン転送（IXFR：Incremental Zone Transfer）などにも対応している。

BINDは、Linux、FreeBSD、Solarisなどの多くのUnix系OSに加えて、WindowsNT/2000などでも動作する。BINDのソースプログラムはhttp://www.isc.org/products/BIND/から入手でき、WindowsNT/2000用のバイナリプログラムもここから入手できる。

BINDでは、namedというプログラムがゾーンデータの管理や問い合わせ結果のキャッシュ、再帰問い合わせの受け付けなどのすべての機能を提供しており、キャッシュ機能を使用するローカルネームサーバでもゾーンを管理する権威ネームサーバでも同じようにnamedを動作させる（図4）。

**・djbdns**

BINDの代替として最近注目されてきているのがdjbdnsである。djbdnsは、SMTPサーバプログラム「qmail」の作者でもあるD.J.Bernstein氏によって作成されたオープンソースのDNS実装である。djbdnsは、qmailと同様にセキュリティ上の欠点が少なくなるように設計されている点と、設定ファイルの記述がシンプルである点が特徴である。

djbdnsは、Linux、FreeBSD、Solarisなどの多くのUnix系OS上で動作する。djbdnsのソースプログラムは、http://cr.yp.to/djbdns.html から入手できる（バイナリでの配布はセキュリティ上の観点から認められていないので、必ずソースからコンパイルする必要がある）。

djbdnsでは、ゾーンを管理する権威ネームサーバプログラム（tinydns）とキャッシュネームサーバプログラム（dnscache）が別になっている。ほかのネームサーバに対する問い合わせの処理は、キャッシュネームサーバであるdnscacheが行い、権威ネームサーバであるtinydnsは、自身が管理しているゾーンへの問い合わせにしか応答しない。このため、図5のように、クライアントのリゾルバは、まず、dnscacheが動作しているキャッシュネームサーバに対して再帰問い合わせを行い、問い合わせた内容がキャッシュとして保持されていなければ、dnscacheがほかのネームサーバ（tinydnsやBINDのnamedなど）に対して問い合わせを行う。また、tinydns自身にはゾーン転送

図4● BINDを使用したネームサーバの構成。namedプログラムにプロセスが集中している

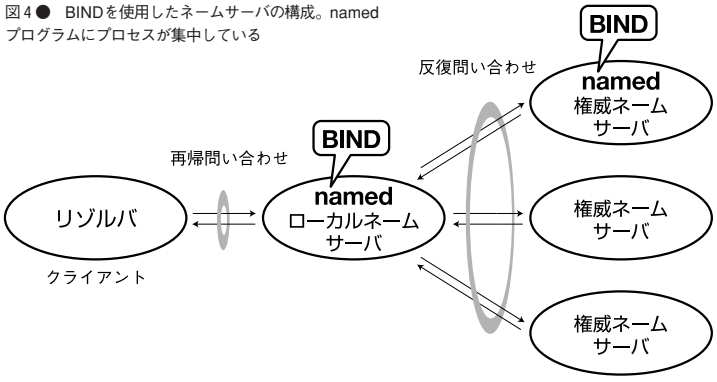
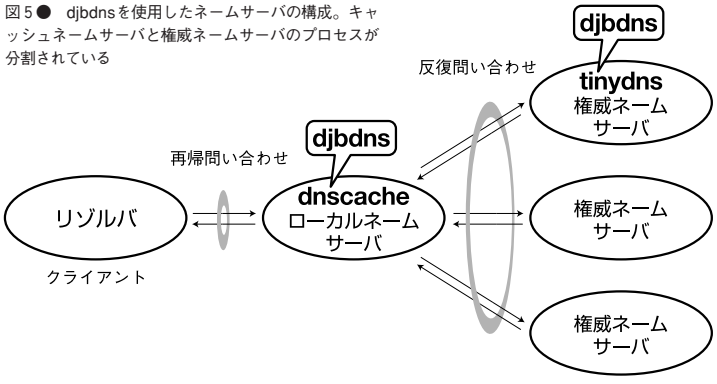



図5● djbdnsを使用したネームサーバの構成。キャッシュネームサーバと権威ネームサーバのプロセスが分割されている




の機能はないため、BINDなどのほかのネームサーバとの間でゾーン転送を行うためには、付属のaxfrdnsというプログラムを使用する。

**・Microsoft DNS Server**

Microsoft DNS Serverは、WindowsNT ServerおよびWindows2000 Serverに付属しているネームサーバプログラムである。Windows2000 Serverに付属しているMicrosoft DNS Serverでは、TSIGやダイナミックアップデート、差分ゾーン転送、IPv6アドレスの問い合わせ（AAAAレコード）などにも対応しており、実用に十分な機能が提供されている。

次回からは、ネームサーバが管理するゾーンデータの内容について詳しく説明していく。

NTTデータ　馬場達也

**●今回の内容に関連するRFC**

RFC 1034 "Domain Names - Concepts and Facilities"
RFC 1035 "Domain Names - Implementation and Specification"
RFC 2308 "Negative Caching of DNS Queries (DNS NCACHE)"