

JANET Technical Guides

JANET Technical Guides are one of a series of user guides available to JANET customers. Technical Guides contain detailed technical information and are intended for technical support staff at JANET sites, network specialists or those with a particular interest in the specialist area.

If you have any queries or comments about the Technical Guides, or would like to obtain copies, please contact:

JANET Customer Service,
UKERNA,
Atlas Centre, Chilton, Didcot,
Oxfordshire, OX11 0QS

Tel: 01235 822 212
Fax: 01235 822 397
E-mail: service@janet.ac.uk

This guide is also available from:

http://www.ja.net/documents/tg_dns.pdf

The following titles are also currently available in this series:

Security Matters
GD/JANET/TECH/001

The Use of Firewalls in an Academic Environment
GD/JANET/TECH/002
(A re-print of the original JTAP report)

Up to date details of all UKERNA's publications are available at:

<http://www.ja.net/documents/>

Contents

1	Building Stable Foundations for Your Network	5
2	An Overview of the Domain Name System	9
2.1	Hierarchical	11
2.2	Distributed.....	11
2.3	Subdomains.....	13
2.4	Domains and Zones	13
3	Registering First and Second Level Domain Names	15
3.1	Choosing Domain Names	17
3.2	Second Level Domains for the UK.....	17
3.3	Registering Domain Names with UKERNA	18
3.4	Registering Domain Names with Nominet UK.....	19
3.5	Registering Names beneath Generic Top Level Domains	19
4	A Review of DNS Software	21
4.1	Software for Microsoft® Windows®.....	23
4.2	Software for UNIX.....	23
5	Constructing Forward Zone Files	25
5.1	Types of Resource Record	27
5.2	Abbreviations in Zone Files	32
5.3	Directives in Zone Files	32
6	Constructing Reverse Zone Files	35
6.1	Reverse Domains.....	37
6.2	Classless Reverse Domains	38
7	Delegation: Creating Child Domains	41
7.1	Child Domains without Delegation.....	43
7.2	Delegating Child Domains.....	43
8	The BIND Configuration File.....	45
9	DNS for Microsoft® Windows®	49
10	Masters, Slaves and Zone Transfers	55
10.1	Where to Locate a Slave.....	57
10.2	Configuring a Master Server.....	57
10.3	Configuring a Slave Server	57
10.4	Communications between Master and Slave Servers	58

11	DNS Queries and Querying Tools.....	61
11.1	Types of Query	63
11.2	The Query Process.....	63
11.3	Querying Tools.....	65
12	Dynamic DNS.....	67
13	Securing a Public DNS Server	71
13.1	Restrict Zone Transfers	73
13.2	Restrict Dynamic Updates	73
13.3	Restrict Recursive Queries	74
14	Common Errors	77
14.1	Illegal Characters	79
14.2	Failing to Increment the Serial Number.....	79
14.3	Inappropriate Numerical Parameters in the SOA Resource Record.....	79
14.4	Lame Delegations	79
14.5	Inclusion of Child Records in Parent Domain’s Zone File	80
14.6	Missing Terminating Periods.....	80
14.7	Aliases in the RDATA Portion of Resource Records.....	80
14.8	Including IP Addresses or Wildcards in Resource Records	80
14.9	Advertising Private IP Addresses	80
	Appendix 1: UKERNA Templates	81
A1.1	Template for Requesting a New Domain Name Under ac.uk/ gov.uk	83
A1.2	Template for Requesting Modification to a Domain Name Under ac.uk/ gov.uk..	83
	Appendix 2: Complete Zone Files for Sunny College	85
A2.1	Forward Zone File	87
A2.2	Reverse Zone File	87
A2.3	BIND Configuration File	88
	Appendix 3: Glossary of Terms	89

Building Stable Foundations for Your Network

1

Building Stable Foundations for Your Network

The Domain Name System (DNS) is used to translate between human friendly names, such as `www.ja.net`, and the numeric IP addresses that computers themselves use to communicate. DNS information can also be used to direct the operation of some Internet services, notably electronic mail. Any organization with an Internet presence should be concerned about DNS for two reasons: to ensure that accurate information about its own services is available to others; and to give local users the ability to resolve host and service names at other locations in the global DNS.

Until recently, many network managers had little need to concern themselves with the intricacies of the Domain Name System (DNS). Internet connections, purchased from a commercial Internet Service Provider (ISP), are invariably bundled with a variety of value-added services that include maintenance of the customer's domain(s).

With the recent connection of Further Education (FE) colleges to JANET and the introduction of DNS as the name resolution method for local area networks based around Microsoft® Windows® 2000, the requirement for a technical guide to commissioning and maintaining DNS servers specifically aimed at JANET connected sites has become apparent.

The author has not sought to supplant the many excellent texts on this subject, and the interested reader is encouraged to explore further. However, it is hoped that a guide in which procedures specific to JANET connected sites are emphasised will be an accessible and useful addition to the arsenal of information on DNS which is already available

An Overview of the Domain Name System

2

Hierarchical

Distributed

Subdomains

Domains and Zones

An Overview of the Domain Name System

The Domain Name System was originally conceived as a worldwide database capable of storing many types of data. These early expectations were not, however, realised and the system is now used exclusively for mapping domain names to IP addresses. The corollary of the hierarchical and distributed nature of the DNS database is that it is very flexible to operate.

2.1 Hierarchical

The database may be represented as an inverted tree (Figure 1) with its hierarchical nature requiring that any given node be addressed relative to its parent.

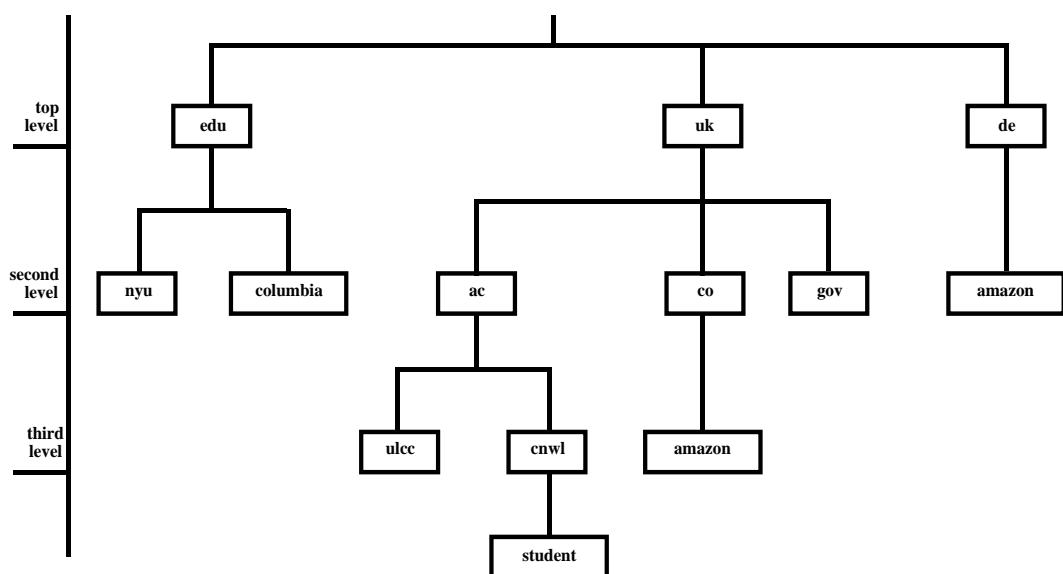


Figure 1: The Domain Name Space

In practice, therefore, the tree is traversed by systematically moving downwards from the root node until the desired child node is reached. Providing sibling nodes have unique names, representing the data hierarchically eliminates the problem of *name collision*. Thus, although any two nodes may have the same name, the fully qualified domain name for each will be completely different due to the unique parentage of each child.

2.2 Distributed

There is no single authority responsible for the entire database. Each node's data can be maintained by an individual or organization to whom authority for that node has been delegated by the maintainer of the parent domain. There is no requirement for the structure of the database to be replicated in the arrangement of servers which host the data. It is entirely feasible for any given name server to hold authoritative data for random, and widely separated, nodes in the database. This implies that an organization with legal title over a domain name can in principle nominate any name server to hold the data for that domain.

2.2.1 Top Level Domains

Each node within the tree belongs to a level of increasing specificity. Immediately beneath the root node are the top-level domains, of which there are two classes: generic and country code.

Generic

The original (US-centric) domains, known as the *generic top-level domains* (gTLD), are summarised in *Table 1*.

Domain	Restricted to	Example Registration
.aero	Air transport industry	baa.aero
.biz	Business organizations	
.com	Commercial organizations	cisco.com
.coop	Co-operative associations	ncb.coop
.edu	US higher educational organizations	nyu.edu
.gov	US governmental organizations	whitehouse.gov
.info	Unrestricted registrations	mta.info
.int	International bodies	nato.int
.mil	US military	army.mil
.museum	Museums worldwide	ashmolean.oxford.museum
.name	Individuals	
.net	Networking organizations	ja.net
.org	Not-for-profit organizations	eff.org

Table 1: Generic Top-Level Domains

Registrations beneath some of these domains are restricted to qualifying organizations, whilst others are freely available. The original three-letter domains have recently been extended with a variety of new registrations such as *.info* and *.biz*.

Country Code

The international nature of the modern Internet is reflected by the presence of *country code top-level domains* (ccTLD). These domain names are based on an international standard (ISO 3166), which assigns a two-letter code to each country in the world. Exceptionally, the United Kingdom does not use its ISO code of *gb* as its top-level domain.

2.2.2 Second Level Domains

The second-level registrations beneath the ccTLDs are treated differently depending upon the parent domain. Some countries, such as Germany, permit organizations to register at the second level whilst others, such as the United Kingdom, only permit registrations beneath defined second-level domains. This difference is depicted in *Figure 1*. As with some of the gTLDs, registrations beneath certain second level domains such as *ac.uk* and *gov.uk* are not freely available, being restricted to qualifying organizations.

2.3 Subdomains

Once an organization has registered a domain name at the appropriate point in the DNS, the administrators of the parent domain then *delegate authority* for the new registration. This means that maintenance of the data pertaining to the new registration becomes the responsibility of the registrant, who performs this task without reference to any higher authority. Changes made to the domain's records automatically propagate through the DNS. Later it will be seen how this process of delegation is accomplished, but with reference to *Figure 1*, it is worth noting that the administrators of third-level registrations are at liberty to define subdomains using exactly the same methods by which all higher-level domains are created.

2.4 Domains and Zones

An appreciation of the difference between a domain and a zone is vital to understanding the processes by which the global namespace is managed and maintained. The process of delegation involves breaking a domain into separately managed units known as zones. A domain encompasses both the parent zone (for example, *ac.uk*) and all child zones for which authority has been delegated to some other body. By delegating the *ulcc.ac.uk* zone to the University of London Computer Centre, UKERNA, which manages the *ac.uk* zone, does not have to spend time managing the DNS data for ULCC. As can be seen by examining *Figure 2*, almost all the *ac.uk* domain has been delegated away from UKERNA with the effect that the data for this zone comprise little more than the delegation information for all the child zones.

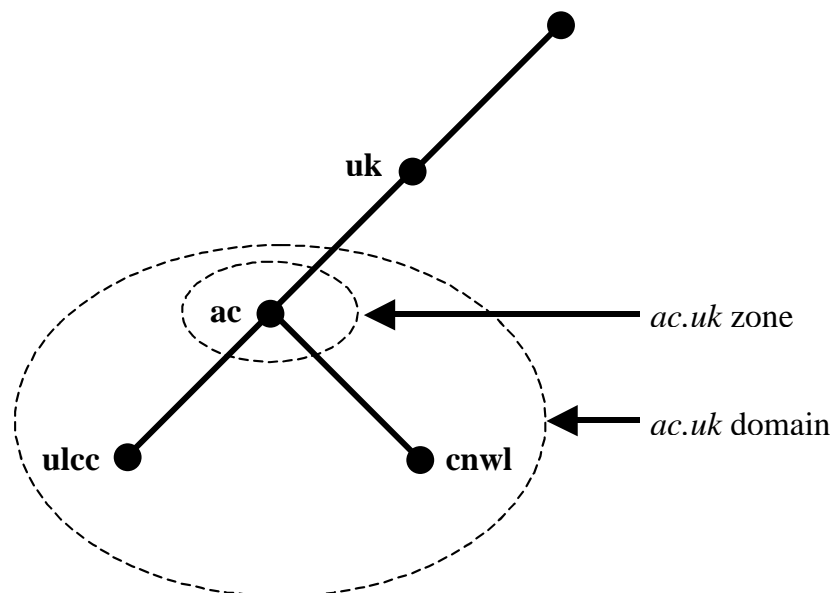


Figure 2: Domains and Zones

It should be apparent, therefore, that the terms *domain* and *zone* should not be used interchangeably, as they refer to different aspects of the DNS tree.

Registering First and Second Level Domain Names

3

Choosing Domain Names

Second Level Domains for the UK

Registering Domain Names with UKERNA

Registering Domain Names with Nominet UK

Registering Names beneath Generic Top Level Domains

Registering First and Second Level Domain Names

In order successfully to register a new domain name, two conditions must be met:

- The administrators of the intended parent domain must create the delegation in the parent's zone file.
- The registrant must have available a name server to accept the delegation. This server may be operated by the registrant or by an ISP on his or her behalf.

This section discusses procedures for registering domain names beneath the generic and .uk top level domains

3.1 Choosing Domain Names

Many considerations apply when selecting an effective domain name. Some registries (see below) impose local requirements, which must be taken into account. However, there are some generic rules governing the construction of domain names that must always be followed. These are as follows:

- Only letters, numbers and the hyphen character are permitted.
- A label must begin with a letter, and must end with a letter or a number.
- A maximum of 63 characters is permitted.
- The fully qualified domain name is limited to 255 characters (including the separating periods).
- No more than 127 separate levels are permitted within the DNS tree (unlikely to be a problem!).

The astute reader will recognise that the second rule in the above list is sometimes flouted (e.g., *3com.com*) but starting a label with a number is the only allowed exception to these rules.

3.2 Second Level Domains for the UK

Responsibility for managing the defined second level domains beneath .uk is assigned to various bodies, as shown in *Table 2*. Registrations within all of these domains are restricted to qualifying bodies except for *.co.uk.*, *.me.uk.* and *.org.uk.*

Responsible Body	Nominet UK	UKERNA	Various
Domain	co.uk me.uk org.uk plc.uk and ltd.uk net.uk sch.uk	ac.uk gov.uk	nhs.uk police.uk mod.uk

Table 2: Responsible Registration Bodies

3.3 Registering Domain Names with UKERNA

Registrations beneath the *ac.uk* domain are restricted to organizations with a permanent physical presence in the UK and which conduct education at the tertiary level or are in receipt of government funding to conduct research, the results of which are released into the public domain.

A qualifying organization may register multiple domain names, but each proposed new name is subject to scrutiny by the naming committee and is subject to the following guidelines:

- Names must exceed two characters in length.
- Names may not be the same as an existing higher-level name (e.g., *.co.ac.uk* would be disallowed).
- The requested domain name must either be representative of the requesting organization's name or a detailed justification must accompany the application.
- The requested name must not present a risk of confusion with existing registrations.
- Domain names pertaining to a project or service (as opposed to an organization) must be centrally funded, of wide relevance to the academic community and be of greater than two years duration.

Subject to these constraints, names are awarded on a 'first come, first served' basis. Once an application has been considered, the response will be either an acceptance of the new name, or a rejection accompanied by the reasons for it. A rejected proposal may be resubmitted if the grounds for the rejection are satisfactorily addressed. Applications may be submitted by organizations with a primary JANET connection or from ISPs that have registered under UKERNA's 'approved ISP' scheme.

The current charging scheme for registrations and modifications is outlined in *Table 3*.

	JANET Site	Approved ISP
Registration	£94	£94
Biennial Maintenance	Free	£47

Table 3: Charging Scheme for Domain Names Registered with UKERNA

All JANET connected sites are allowed to register their first domain name for free. Domain names registered before 1 April 1999 are not subject to the maintenance charge unless a modification is required, at which point the ISP is required to join the maintenance scheme and pay the associated charge. The first two years' maintenance is included in the initial registration charge.

Applications for new domain names should be submitted to:

naming-ac@ukerna.ac.uk

Modifications to existing domain names should be sent to:

naming-admin@ukerna.ac.uk

The relevant templates to be used in both cases are given in *Appendix 1*.

The reader is referred to UKERNA's website for the most current information regarding domain names:

http://www.ja.net/documents/naming/names_ac_gov.html

3.4 Registering Domain Names with Nominet UK

Nominet UK is a not-for-profit company established in 1996. Their website may be viewed at :

<http://www.nic.uk/>

Nominet have a membership scheme with various benefits, including voting rights and access to exclusive information, for members who pay a joining fee and annual subscription. A Nominet tag holder benefits from being able to use Nominet's automated registration system. This enables registrations and modifications to be processed swiftly with minimal human intervention by sending PGP-signed e-mail to the automaton. Members who are also tag holders are entitled to reduced domain name registration fees. It is most unlikely that any JANET connected site would need to avail themselves of these two schemes. They are primarily aimed at ISPs that manage large numbers of domain names on behalf of customers.

For organizations that are not tag holders, there are two methods of registering domain names through Nominet. The preferred method is via an ISP, and UKERNA is able to register domain names with Nominet on behalf of JANET connected organizations. A charge is levied for this service. Under these circumstances, the registrant does not have any direct contact with Nominet. Alternatively, it is possible to register domain names directly with Nominet providing the registering organization tenders a credit card, is able to provide the IP addresses of two name servers that will host the domain and employs staff competent to deal with the process and associated technical requirements.

All domain names registered with Nominet are subject to a biennial renewal fee.

3.5 Registering Names beneath Generic Top Level Domains

The body ultimately responsible for the maintenance of the various generic top-level domains is the Internet Corporation for Assigned Names and Numbers (ICANN). Two types of generic top-level domain are recognised; sponsored and unsponsored. A sponsor is an organization to which ICANN delegates some ongoing policy formulation governing the manner in which the sponsored gTLD will be operated. Registrations beneath these domains are restricted to qualifying bodies. An unsponsored gTLD operates under policies established by the global Internet community directly through the ICANN process. Names beneath these domains may be registered through any one of numerous competing commercial organizations (known as 'registrars'). Each registrar sets its own charges and regulations governing renewals etc. Registrars must be accredited by ICANN, but they may offer their services through approved resellers which provide some assistance with the registration process. It is possible to move a registered domain name between registrars at any time more than sixty days after initial registration. A list of registrars is available on the ICANN website:

<http://www.icann.org/>

A Review of DNS Software

4

Software for Microsoft® Windows®

Software for UNIX

A Review of DNS Software

There are DNS server packages available for every modern network operating system. The standard implementation is BIND on UNIX, and many of the commercial packages are based on the open-source BIND software. The machine nominated as the primary name server for an organization should be reliable, stable and permanently visible on the Internet via a static IP address. Remember that if the DNS server is unreachable (because it keeps crashing) then local users may not be able to resolve names, and remote machines may experience difficulties making contact for e-mail delivery and so forth. The additional load placed on a machine by virtue of its role as a name server is not normally significant; it is perfectly feasible to run other services on the same server. For example, the college's mail, proxy or web servers could also act as name servers. Precious funds should not be spent on a powerful new server just for running DNS! Be aware, however, that multiple DNS servers cannot be run on the same host.

4.1 Software for Microsoft® Windows®

There are several possible DNS server solutions for sites running either Windows NT or Windows 2000 (Advanced Server). The port of BIND for Windows will run on any flavour of NT or Windows 2000, however this software is not updated nearly as frequently as the parent UNIX version. For more information on setting up BIND/NT see:

<http://bind8nt.meiway.com/>

Microsoft also bundle a DNS server with their network operating systems.

4.1.1 Windows NT4

There is a known problem with the Microsoft offering in NT4 related to running the reverse zone for a subnetted (classless) network (see *Section 6.2*). Whilst the underlying server (running as an NT service) is capable of handing the extensions to the original specification for classful networks, the DNS Manager Graphical User Interface (GUI) program, used to provide a visual editor for the zone files, is not. The corollary is that sites that use classless subnets will find that they have to create and edit the zone files with a text editor, and not the GUI application.

4.1.2 Windows 2000 (Advanced Server)

A revamped DNS server is provided by Microsoft with Windows 2000. The author has performed several installations with it and found that the deficiencies in the earlier offering under NT4 have been remedied. A user interface consistent with other administrative tools is available courtesy of the Microsoft Management Console (MMC).

4.2 Software for UNIX

If a DNS server is to run on a Linux (or other UN*X box) then the obvious choice is BIND, although Dan Bernstein (of qmail fame) does offer an alternative in which the DNS server

and cache manager run as separate processes. The binaries for the latter have a very small footprint and read their data from compiled (not text) files.

The BIND software may be included with the operating system software, but if not, it is available from the Internet Software Consortium site:

<http://www.isc.org/>

Constructing Forward Zone Files

5

Types of Resource Record

Abbreviations in Zone Files

Directives in Zone Files

Constructing Forward Zone Files

Lurking behind every registered domain name are one or more computers, referred to as *name servers*, running DNS server software, which answer incoming queries about the domain. Data pertaining to the domain are stored in a *zone file* using ASCII text format. Zone files are composed of *resource records*, (also referred to as RRs) consisting of up to five columns (angled brackets indicate optional entries with defined default values):

<owner> **<class>** **type** **<TTL>** **RDATA**

The ordering of resource records within a zone file is not significant, and need not be preserved by name servers or resolvers. White space is commonly used to provide a legible format to the zone file, and is ignored by the server.

Owner

The *owner* column is a fully-qualified domain name with an associated 'answer' (of specified class and type) stored in the *RDATA* column.

Class

There are three possible *classes* of resource record, but only the Internet (IN) class (see below) is significant. The Hesiod (HS) class is for an information service from MIT's Project Athena. It is used to share information pertaining to various systems databases (users, printers, *etc.*) and will not be discussed further. ChaosNet, a LAN protocol created in the mid-1970s, is still sometimes seen on LISP stations and other hardware in the artificial intelligence community; zone data for it can be specified with the *CHAOS* (CH) class. It may also be used to query the version of BIND running on a name server. It is not required to specify a resource record's class as the IN class is assumed.

TTL

The *TTL* (time to live) field dictates the length of time in seconds that a resource record may be retained in a resolver's cache. This need not be specified for each individual record, in which case a global value is applied.

The *RDATA* field contains the data that are presented in the 'answer' section of a query that has been answered by the name server.

5.1 Types of Resource Record

The IN class of resource record is composed of a variety of types, some of which every site will deploy in the zone files. A selection of types, and the number of the associated Request for Comment (RFC), are given in *Table 4*. Those types that will likely be required in any site's zone files are shaded.

Not all are shown; deprecated and some rarely-used experimental types have been omitted. The following sections describe the important types of resource record, with examples drawn from the zone file of an imaginary institute called Sunny College.

Type	Value	Meaning	RFC
A	1	host address	1035
NS	2	authoritative name server	1035
CNAME	5	canonical name for an alias	1035
SOA	6	marks the start of a zone of authority	1035
NULL	10	anything	1035
WKS	11	a well known service description	1035
PTR	12	a domain name	1035
HINFO	13	host information	1035
MX	15	mail exchange	1035
TXT	16	text strings	1035
RP	17	responsible person	1183
ISDN	20	direct dial ISDN number	1183
AAAA	28	IPv6 host address	1886
LOC	29	geographical location	1876
SRV	33	protocols and services	2052

Table 4: Common Resource Record Types

5.1.1 SOA Records

It is required that a start of authority (SOA) resource record is present at the top of every zone file. This indicates that the name server is the authoritative source of information for that domain. The following SOA record is present in the fictional zone file for Sunny College:

```
sunny.ac.uk. SOA dns.sunny.ac.uk. hostmaster.sunny.ac.uk. (
    2000120801 ;serial
    28800 ;refresh
    7200 ;retry
    604800 ;expire
    86400) ;minimum TTL
```

The RDATA section of this resource record is quite complex, containing as it does seven separate items. These comprise the name of the DNS server (dns.sunny.ac.uk.) followed by the e-mail address of the individual responsible for the domain (hostmaster.sunny.ac.uk). Note how the '@' sign that usually separates the addressee from the domain is here replaced with a period character and that the addressee must not contain any periods. For these reasons, unaliased e-mail addresses are normally used. The provision of such contact information within the SOA resource record allows for human communications pertaining to the domain.

All but one of the numerical parameters govern how the master server communicates with its slaves. This is discussed fully in *Section 10*. The values shown should be suitable for any JANET site.

The parentheses allow the RDATA section to span multiple lines and the semicolons are used to insert comments into the zone file.

5.1.2 NS Records

Any zone file should also include resource records of the NS type, which indicate the authoritative name servers for the domain.

There should be an NS record for the master and each of the slave servers. Our

example organization, Sunny College, operates its own master server with the slave being provided by JANET and so the NS records should be as follows:

```
sunny.ac.uk.      NS      dns.sunny.ac.uk.
sunny.ac.uk      NS      ns3.ja.net.
```

5.1.3 A Records

The most readily-understood use of the DNS is mapping domain names to IP addresses. This is simply done using resource records of the 'A' type.

```
; Host Addresses
gw.sunny.ac.uk.      A      212.219.172.89
garnet.sunny.ac.uk.  A      212.219.172.91

; Multi-homed Devices
ftp.sunny.ac.uk.     A      212.219.172.92
ftp.sunny.ac.uk.     A      212.219.172.93

; Aliases
dns.sunny.ac.uk.     A      212.219.172.91
mail.sunny.ac.uk.    A      212.219.172.91
```

Every node on a network that has been assigned a real-world (as opposed to private) IP address should have an entry in the organization's DNS zone file. This rule applies equally to nodes with a real-world address bound to a network interface card or those with entries in network address translation (NAT) tables. Do not include IP numbers from the 'private' ranges in externally-visible DNS zone files. Remember that the server is there to inform the world about services being offered and if the mail server is advertised with an unreachable IP address, no e-mail will be received regardless of how well-configured the DNS might be!

This example has been deliberately constructed so as illustrate the following points.

Multi-homed devices

Although less usual, it is also permissible for the same hostname to be mapped to multiple IP addresses. By these means, an elementary load-balancing system can be achieved. Companies whose websites are subjected to very high traffic sometimes mirror their site across multiple servers and map the hostname 'www' to the multiple IP addresses comprising the server farm. This is illustrated in the example where the college's ftp service is balanced across two servers.

Aliases

It is perfectly legal for multiple hostnames to be mapped to one IP address. If a host offers several different services (e.g. it is a mail server and a name server), then many DNS administrators will assign a name to the physical host along with 'standard' hostnames that refer to each of the services the host offers. Although it is permissible for the equivalent number of corresponding PTR records to exist in the reverse zone, the author's inclination is to include a single PTR record for the host's physical name only. Note in the above example how the station called 'garnet' has two additional names alluding to the Internet services offered by the host.

5.1.4 CNAME Records

It is the author's opinion that resource records of this type should be restricted to hosts sited on remote networks that need to be included in the site's DNS for some reason. A common example would be an externally-hosted website. While the hostname of 'www' could legitimately be defined in an 'A' record, this is not best practice as the ISP are at liberty to change their servers' IP numbers at any time. This would then cause lookups of 'www.sunny.ac.uk' to fail.

```
www.sunny.ac.uk.          CNAME          webfarm1.happyhosters.co.uk.
```

A preferable solution would be to include a CNAME record to the fully-qualified domain name of the remote server as shown. If multiple names need to be matched to a single IP address within the site's own network, then multiple 'A' type records should be used. Deploying CNAME types in these circumstances is pointless; it simply forces a resolver to issue two queries.

5.1.5 MX Records

Mail delivery to a domain is controlled by the presence of the MX type of resource record in the DNS. These records specify a mail exchanger for the domain that will either process the mail it receives or forward it to another mail exchanger. The RDATA section of MX records includes two parameters; the domain name of the mail exchanger and a 16-bit unsigned integer called a *preference value*.

```
sunny.ac.uk.             MX             10 mail.sunny.ac.uk.
```

The latter allows for a series of alternative mail exchangers to be defined. A remote mailer will attempt to deliver mail to the mail exchanger with the *lowest* preference value. If the most preferred mail exchanger fails to respond, then alternative mail exchangers (if present in the DNS) are contacted in order of increasing preference value. If more than one MX type of resource record is defined for a domain, a special algorithm is deployed so as to prevent mail routing loops, which can arise as a result of mailers forwarding mail either to themselves or to a mailer with a higher preference value.

5.1.6 PTR Records

This type of resource record is only used in *reverse* zone files, the purpose of which are to transfer IP addresses into domain names. The PTR type of resource record may be regarded as the 'opposite' of an 'A' record and is fully discussed in *Section 6*.

5.1.7 SRV Records

The SRV type of resource record may be regarded as a generalised MX record in which the identity of a server for a particular protocol and domain is given. Multiple servers for a given protocol may be defined with an associated preference value. As with the MX type of resource record, a resolver should always select the server with the lowest preference value.

```
;backup web server running on port 8080 on ISP's network
http.tcp.www.sunny.ac.uk.      SRV 10 0 80   www.sunny.ac.uk.
http.tcp.www.sunny.ac.uk.      SRV 20 0 8080 www.ip-provider.net.
```

```

;load-balanced ftp service with ¾ of logons directed to 'ftp2'
server
ftp.tcp.sunny.ac.uk.          SRV 0 1 21 ftp1.sunny.ac.uk.
ftp.tcp.sunny.ac.uk.          SRV 0 3 21 ftp2.sunny.ac.uk.

```

An additional unsigned 16-bit integer, called the weight, is defined to allow for load-balancing amongst servers for a given protocol with the same priority value. This figure should not be taken too literally as the load on most servers will vary too quickly to be reflected in the caches of DNS resolvers. In effect, the weight parameter gives an indication of the relative performance of different servers.

The translation of service names to port numbers is currently performed by the client (on UN*X systems, the file `/etc/services` contains these data). Moving this to the DNS allows for the development of new services without updating system files on all client machines and makes it easy to map standard services to non-standard ports, as shown in the example.

The widespread use of the SRV type of record over the Internet is a future development, however Active Directory technology on Microsoft's Windows 2000 operating system now uses DNS as the name resolution method, WINS having been abandoned. This would not have been feasible without recourse to two facilities (dynamic DNS and the SRV type) in the DNS standard. The use of DNS on a Windows-based LAN is covered fully in *Section 9*.

5.1.8 LOC Records

It is possible to encode precise geographical data about domains and networks in the DNS using this type of resource record. The location is expressed using latitude, longitude (both in degrees, minutes and seconds) and altitude (in meters). It is possible to obtain latitude and longitude for any UK postal code courtesy of a number of websites, for example:

<http://www.streetmap.co.uk/>

However, precise altitudes may be more difficult to obtain accurately.

```
sunny.ac.uk          LOC          51 31 25 N 0 7 7 W 68m
```

5.1.9 HINFO Records

This type of record is used to record the hardware type and operating system of a host in the DNS and is strictly informational, not functional. There is a security risk inherent in unnecessarily exposing this type of information in the DNS, and it is recommended that this type of resource record is not used without a compelling reason.

5.1.10 RP Records

The mandatory SOA record at the head of the zone file contains contact information to which queries regarding the domain may be sent. It is possible to include finer-grained contact information using this type of resource record. Using this type of resource record allows for contact information to be attached to a non-delegated sub-domain or host.

5.2 Abbreviations in Zone Files

The amount of text within a zone file may be reduced by making use of two common abbreviations. It is important to remember that inappropriate use of these may cause the origin of one or more records to be incorrect.

5.2.1 Omit Owner Field

The first field within a resource record is optional. If it is omitted, then the owner of the previous record is assumed to hold.

```
;multi-homed addresses
ftp.sunny.ac.uk      A           212.219.172.92
                    A           212.219.172.91
```

In contrast with the example in Section 5.1.3, the 'owner' field of the second record may be replaced with tabs. Queries about the host concerned would generate the same answer in both cases.

5.2.2 Appended Origins

Unless the owner field in a resource record is terminated with a period, the current origin will be appended. This will have been learnt either from the server's configuration file, or by using the \$ORIGIN directive within the zone file. The following two 'A' records are equivalent:

```
;fully-qualified owner field
garnet.sunny.ac.uk.  A           212.219.172.91

;trailing period omitted
garnet      A           212.219.172.91
```

Both records are read as garnet.sunny.ac.uk. Had the terminating period in the first example been omitted, the domain name would be read as garnet.sunny.ac.uk.sunny.ac.uk. This convention can cause the most problems when including domain names in the right-hand side of a record:

```
;note the (correctly) applied terminating period
www      CNAME      webfarm1.happyhosters.co.uk.

;this would be read as webfarm1.happyhosters.co.uk.sunny.ac.uk
www      CNAME      webfarm1.happyhosters.co.uk
```

Note how failing to terminate the domain name on the right-hand side of the resource record caused a nonsensical result.

5.3 Directives in Zone Files

In addition to resource records, two directives may also be included in zone files. The origin may be changed at any point in a zone file by use of the \$ORIGIN directive. This is particularly useful when constructing sub-domain records.

```

garnet                A                212.219.172.91

;change the origin to student.sunny.ac.uk
$ORIGIN student
cinnabar              A                212.219.172.94

```

Note how neither of the resource records, nor the directive, are terminated with a period character in the above example. The effect of this is to cause the second 'A' record to correspond to *cinnabar.student.sunny.ac.uk*. All records following the directive will have the new origin appended unless they are terminated with periods.

Other files may be included at any point by use of the \$INCLUDE directive. This could be used to maintain records pertaining to a undelegated sub-domain in a separate file.

```

garnet                A                212.219.172.91

;records pertaining to the student subdomain kept in a separate file
$INCLUDE student.sunny.ac.uk.dns

```

The author has seen cases where the SOA record was maintained in a separate file and this directive was used to include it at the top of every zone file. This practice is not recommended as it means that the serial number cannot be incremented for an individual zone file.

Constructing Reverse Zone Files

6

Reverse Domains

Classless Reverse Domains

Constructing Reverse Zone Files

The most common purpose of the DNS is to resolve domain names into IP addresses to which packets may be routed. There are occasions where the reverse is required and a known IP address needs to be resolved to the corresponding domain name. Such *reverse lookups* are often performed as part of an automated security check. A common example of this concerns mail exchangers.

It is a requirement of connection to JANET that sites maintain a correctly-delegated reverse zone file for each block of IP addresses that have been assigned. The same basic format is adopted for the zone files except that only four types of resource record are permitted. These are SOA, NS, CNAME and PTR.

6.1 Reverse Domains

As seen previously, the global domain name space is indexed by name, not by IP address. A scheme is therefore required to provide a reverse lookup facility that does not require trawling through the entire number space until the desired IP address is located (a clearly impossible task). By generating domain name labels from the octets in IP addresses (expressed using dotted quad notation), it is possible to create 'names' out of IP addresses and thereby provide a reverse lookup facility. This portion of the DNS is located beneath the *in-addr.arpa* domain, as shown in Figure 3.

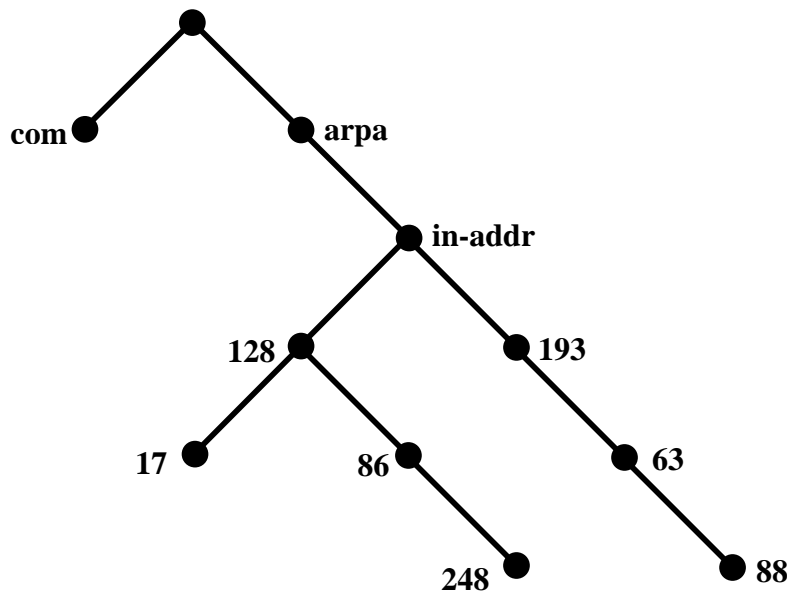


Figure 3: Reverse Domain Name Space

Specificity within an IP address increases from left to right. In other words, the number of hosts within a class A network (16777216) is far greater than the number of hosts within a class C network (256). For domain names, the level of specificity decreases as one approaches the root node. For consistency, therefore, the quads comprising an IP address are written in reverse order within the DNS. Each node beneath *in-addr.arpa* corresponds to a network and so the depth of the tree beyond this domain extends to three levels (delegations do not extend to individual IP addresses, or the fourth quad).

As an example, consider the host `calypso.ulcc.ac.uk`, which has the IP address `193.63.88.9`. The domain `193.in-addr.arpa` is delegated to the Réseaux IP Européens (RIPE), who either delegate authority for the 255 child domains or retain the delegation. Addresses from the `193.63.0.0/16` block are not allocated directly by the RIPE but by JANET, which is a national registry. Therefore, it makes sense for authority for the domain `63.193.in-addr.arpa` to be delegated by the RIPE to JANET as well. JANET has assigned the class C network `193.63.88.0/24` to ULCC and so JANET also delegates authority for the corresponding reverse domain to ULCC. In general, it follows that the body to whom a block of IP addresses has been allocated also accepts the delegation for the reverse domain.

6.2 Classless Reverse Domains

As can be seen from the preceding section, the scheme for reverse delegations does not allow for assignment of a block of IP numbers smaller than a class C network. Due to the depletion of global address space, it is nowadays comparatively rare for organizations to be allocated an entire class C network. Instead, smaller blocks of IP numbers are assigned by ISPs to their customers. Subnetting networks on non-octet boundaries is known as *classless inter domain routing* (CIDR).

As an example, imagine that the example organization, Sunny College, has been assigned the IP numbers in the range `212.219.172.88 – 212.219.172.95`. This block comprises eight addresses in which 29 of the 32 bits in each address specify the network, with the remaining 3 bits specifying the host address within the network. The range of numbers may therefore alternatively be expressed as `212.219.172.88/29`. The reverse domain corresponding to the parent class C network from which this range is derived is `172.219.212.in-addr.arpa`. Evidently, this domain cannot be delegated to Sunny College as all but eight of the addresses are assigned to other organizations. There are two approaches to solving this paradox.

The IP registry (in this example, JANET) could retain the delegation for the class C network's reverse domain. However, JANET would then assume responsibility for maintaining all the address to domain name mappings; an impossible task given the considerable size of the address space assigned to JANET connected organizations and the inherent delays sites would experience whenever changes were required.

An approach exists in which IP address space can be assigned in smaller blocks than 24-bit prefixes and authority for the corresponding reverse name space can be delegated. The `in-addr.arpa` tree must be extended downwards by creating additional nodes to which authority may be delegated. This requires that:

- Alternative reverse lookup names are created according to a standard scheme.
- A static translation is maintained between the 'new' name and the normal name in the zone file for the parent network.
- The new names are delegated to the relevant organization.

Extending our example of Sunny College's network shows how this works. JANET retains authority for the reverse zone (`172.219.212.in-addr.arpa`) corresponding to the class C network from which the college's IP addresses are drawn. The new reverse lookup names are created from an origin, which takes the form:

P/Q.Z.Y.X.in-addr.arpa.

in which

P = Fourth quad of network number
 Q = Network mask length
 Z, Y, X = Third, second and first quads of network number

The relationship between an IP address, reverse name and classless reverse name are readily visualised by consulting *Table 5*, which lists some of the hosts on Sunny College's network.

Hostname	IP Address	Reverse Domain Name	Classless Reverse Domain Name
gw.sunny.ac.uk.	212.219.172.89	89.172.219.212.in-addr.arpa.	89.88/29.172.219.212.in-addr.arpa.
garnet.sunny.ac.uk.	212.219.172.91	91.172.219.212.in-addr.arpa.	91.88/29.172.219.212.in-addr.arpa.
ftp.sunny.ac.uk.	212.219.172.92	92.172.219.212.in-addr.arpa.	92.88/29.172.219.212.in-addr.arpa.
	212.219.172.93	93.172.219.212.in-addr.arpa.	93.88/29.172.219.212.in-addr.arpa.

Table 5: Derivation of Classless Reverse Domain Names

The mappings between the two reverse names for each IP address are created in the zone file for the *172.219.212.in-addr.arpa* domain, which remains delegated to JANET. This is achieved by means of CNAME records. A portion of this (imaginary) zone file is shown below.

```

$ORIGIN 172.219.212.in-addr.arpa.

@          SOA          ns1.ja.net. hostmaster.ja.net. (
                2002042301 ;serial
                28800      ;refresh
                7200       ;retry
                604800     ;expire
                86400)    ;minimum TTL

                NS        ns0.ja.net.
                NS        ns4.ja.net.

;the domain 88/29.172.219.212.in-addr.arpa. is redelegated
88/29      NS        dns.sunny.ac.uk.
                NS        ns3.ja.net.

;reverse names are mapped to the classless form here
89        CNAME     89.88/29.172.219.212.in-addr.arpa.
90        CNAME     90.88/29.172.219.212.in-addr.arpa.
91        CNAME     91.88/29.172.219.212.in-addr.arpa.
92        CNAME     92.88/29.172.219.212.in-addr.arpa.

```

A reverse query will initially be received by the primary name server for the parent class C network. The CNAME mapping will then cause a second query to be issued for the classless reverse domain name. Because the parent reverse zone file contains delegation information for this domain, the second query will be directed to the zone file (shown below) operated by the organization.

```

$ORIGIN 88/29.172.219.212.in-addr.arpa.

@      SOA      dns.sunny.ac.uk. hostmaster.sunny.ac.uk. (
                2002042301      ;serial
                2880           ;refresh
                7200           ;retry
                604800         ;expire
                86400          ;minimum TTL

                NS      dns.sunny.ac.uk.
                NS      ns3.ja.net.

89     PTR      gw.sunny.ac.uk.
91     PTR      garnet.sunny.ac.uk.
92     PTR      ftp.sunny.ac.uk.
93     PTR      ftp.sunny.ac.uk.
    
```

Note how the right-hand side of every resource record comprises a fully-qualified domain name with a terminating period. Forgetting that the origin of a reverse zone file is quite different from that of the forward file is a very common mistake.

An example of this process in action is depicted in *Figure 4*. An organization's local name server has received a request from a client on the LAN to resolve the IP address 212.219.172.91 to the corresponding fully-qualified domain name. As shown, this requires two queries to be dispatched. The first involves the local name server seeking the PTR

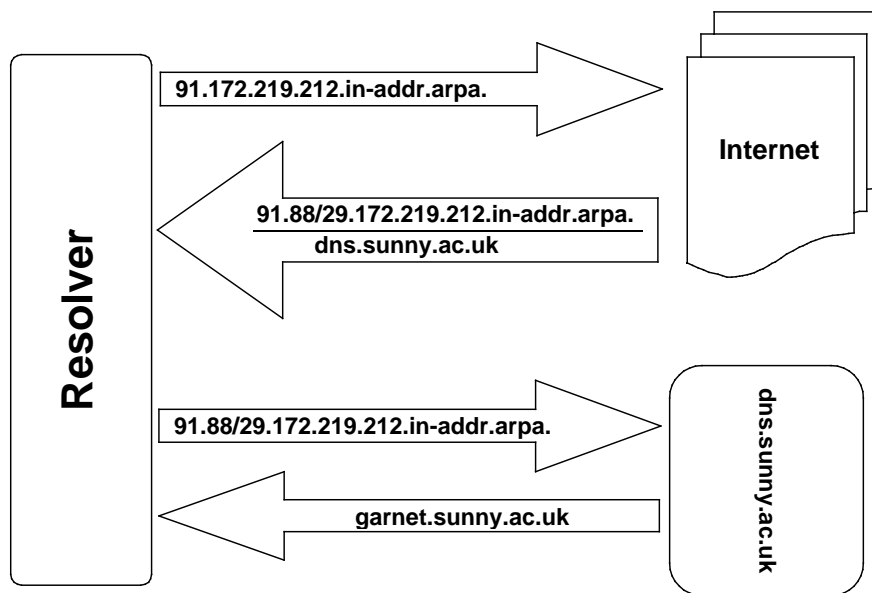


Figure 4: The Reverse Query Process

record for the 91.172.219.212.in-addr.arpa domain name and is resolved in the usual fashion (see *Section 11*) by means of a series of iterative queries posted to relevant name servers on the Internet. When the name server for the parent class C network receives this query, it returns as an 'answer' the right-hand side of the CNAME resource record in its zone file. The delegation information for the 88/29.172.219.212.in-addr.arpa domain is also returned in the 'authority' section of the response, as shown in the diagram. The local resolver then dispatches a single recursive query to the name server to which the classless network has been delegated, and receives either the domain name or an error if no PTR record is found.

Delegation: Creating Child Domains

7

Child Domains without Delegation

Delegating Child Domains

Delegation: Creating Child Domains

Management of an organization's namespace often involves the creation of subdomains, in exactly the same sense as domains are created within the *ac.uk* namespace. Sometimes a subdomain is required simply to indicate an administrative function with the organization. If the subdomain is being assigned to an autonomous organization, it may be appropriate for authority for the subdomain to be delegated away from the parent name servers. Both of these cases will be examined with the aid of examples.

7.1 Child Domains without Delegation

Sunny College have decided that e-mail addresses should distinguish between staff and student users by creating a subdomain of *student.sunny.ac.uk*. Mail sent to these addresses will be handled by the same mail server as for the parent domain and the college systems staff will manage all aspects of the new subdomain. In these circumstances, there is clearly no need for authority for the child domain to be delegated away from Sunny College's primary name server. In order to implement the new policy, a single additional MX resource record within the zone file for the *sunny.ac.uk* domain thus:

```
student.sunny.ac.uk.           MX           10 mail.sunny.ac.uk.
```

Some time later, the student union are granted permission to operate their own web server on the college's network. This requires the addition of another record:

```
www.student.sunny.ac.uk.      A           212.219.172.94
```

A more concise method of expressing these records is given in the complete zone file (see *Appendix 2*), but the key point is that the subdomain and its resource records were created without delegation of authority to a different name server.

7.2 Delegating Child Domains

When Sunny College originally applied to UKERNA for their domain name, this was created by delegating authority for the subdomain within the *ac.uk* zone file. In order to create the delegation, one or more NS resource records are required to map the domain name to its authoritative name servers:

```
sunny.ac.uk.                 NS           dns.sunny.ac.uk.  
                             NS           ns3.ja.net.
```

While these records create the delegated domain, they are insufficient in their own right because the name server for the parent *ac.uk* domain cannot answer iterative queries about the child domain without knowing the IP addresses of the name servers, and the first of these can apparently only be provided by Sunny College's primary name server. This paradox is resolved by including an 'A' record that maps the first of the delegated name servers to an IP address:

```
dns.sunny.ac.uk.             A           212.219.172.91
```

This is known as a *glue* record; *glue* records are required whenever a domain is delegated to a name server that resides within the delegated domain. Clearly, a glue record is not required for the second name server as it resides outside the delegated domain. Although recent versions of BIND will ignore unnecessary glue records, they should never be included.

The BIND Configuration File

8

The BIND Configuration File

Once all the required forward and reverse zone files have been created, the BIND daemon or service must be configured. When the daemon starts, it reads a configuration file which informs it about the authoritative zones and the location of the corresponding zone files in the local file system. In addition to mapping the names of authoritative zones to the corresponding files, various options may be specified in the configuration. These include the file system directory where the zone files are stored, allowing names of the zone files to be specified relative to this base directory and those options concerned primarily with securing the server; these are discussed fully in *Section 13*.

The syntax of the configuration file is radically different for the two major versions (4 and 8) of BIND. In the interests of brevity, this document discusses only the BIND 8 syntax. Comments may be inserted in the file using any of three different styles.

```
/* This is a C-style comment */  
// This is a C++-style comment  
# This is a shell-style comment
```

A basic configuration file for Sunny College is shown below. The full version is given in *Appendix 2*, which includes the security-related options alluded to in *Section 13*.

```
options {  
    directory "c:\winnt\system32\dns\etc";    /* Windows file system */  
    directory "/usr/local/named";          /* UNIX file system */  
};  
  
// location of root name server names and addresses  
zone "." IN {  
    type hint;  
    file "root.dns";  
};  
  
// main forward zone file  
zone "sunny.ac.uk" IN {  
    type master;  
    file "sunny.ac.uk.dns";  
};  
  
//classless reverse zone file  
zone "88/29.172.219.212.in-addr.arpa" IN {  
    type master;  
    file "88-29.172.219.212.in-addr.arpa.dns";  
};
```

Note how each section is enclosed within curly braces and how each statement (and section) is terminated with a semi-colon character. The class of each zone must also be specified.

If a server is to resolve Internet domain names, then it must know the names and IP addresses of the root name servers. These have been stored in a file called 'root.dns', and the root zone has been mapped to this file. Because the server is neither a master nor a slave for this zone, the special 'hint' type is used.

The two zones (forward and reverse) for which this server is the authoritative master are listed together with the filenames of the corresponding files.

DNS for Microsoft® Windows®

9

DNS for Microsoft® Windows®

Almost all of the preceding material is of direct relevance to sites intending to operate a DNS server under Microsoft Windows, however a dedicated discussion of the user interface in the Windows 2000 DNS server is merited.

The same zone files are required on both platforms. On Windows 2000 they are stored within the %systemroot%\system32\dns directory. The zone files may be manually constructed or they can be compiled by means of the graphical user interface, shown in *Figure 5*.

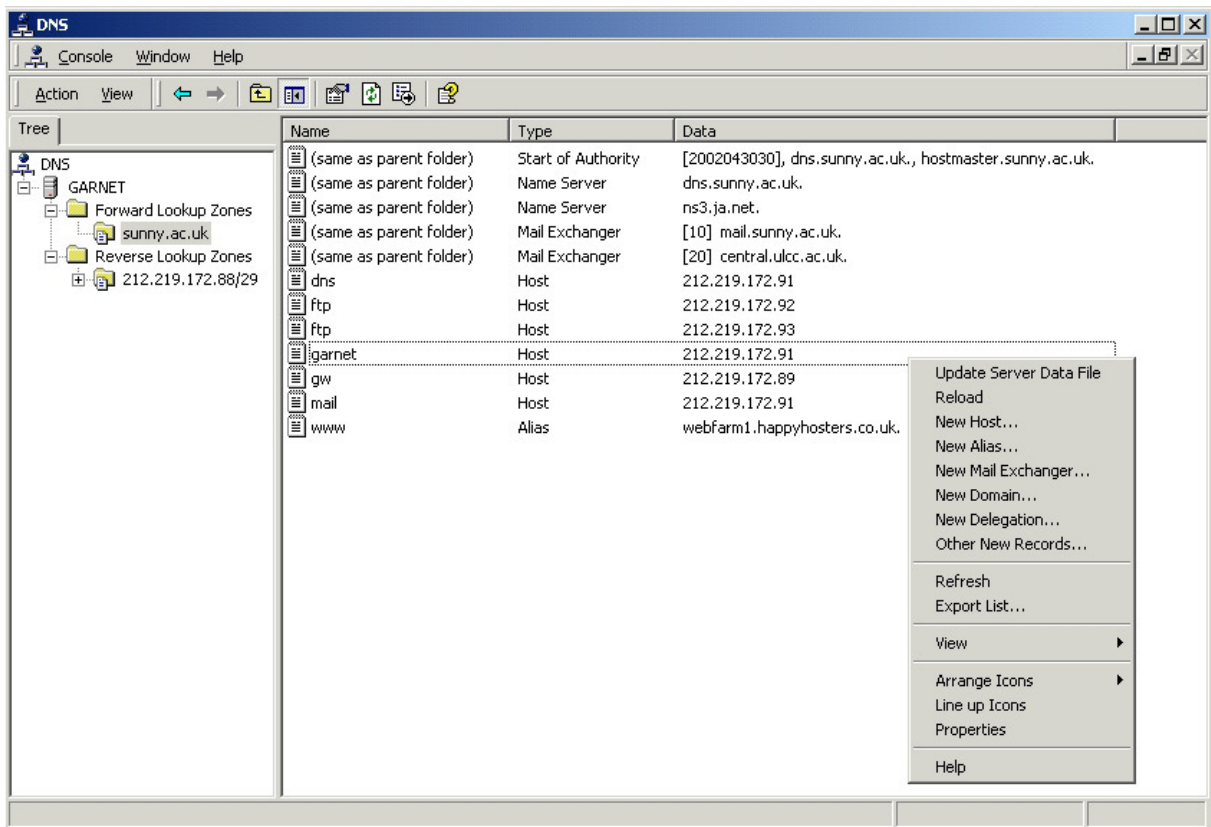


Figure 5: The Windows 2000 DNS Server

A zone is created by right-clicking on either the 'Forward' or 'Reverse' folder and selecting the 'New Zone' option from the pop-up menu. While this is trivial for a forward zone, the correct origin has to be supplied when creating a classless reverse zone (*Figure 6*).

Creating PTR resource records within a classless reverse zone file is no more intuitive under Windows 2000 than when manually creating zone files, because the file's origin has to be supplied. No tools are available to construct this from an IP address written using CIDR terminology (*Figure 6*).

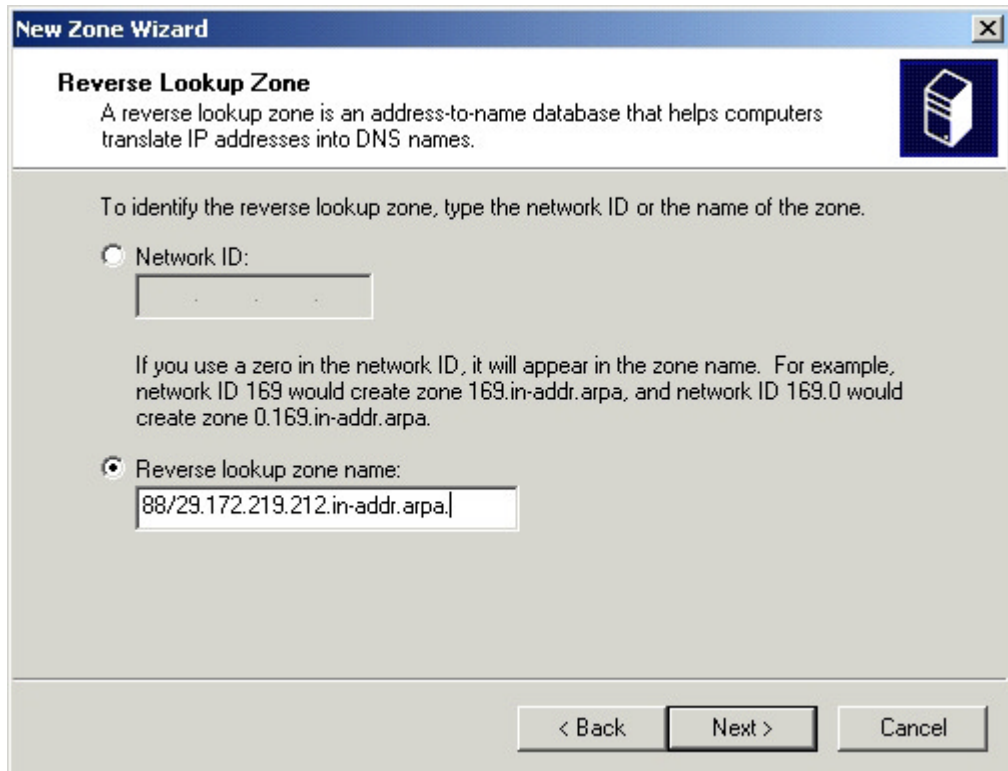


Figure 6: Creating a Classless Reverse Zone in Windows

Once the zone has been created, records must be added by right-clicking on the zone name and selecting the desired type of resource record from the pop-up menu which can be seen in *Figure 5*. As records are created, they are listed in the right-hand pane of the user interface. Most records are trivial to create, however the author has noticed that the default values for the numerical parameters in the SOA record are all too small for most normal purposes. These may be altered at any time by displaying the SOA tab on the zone's 'Properties' page. Often-used values are shown in *Figure 7*.

The image shows a Windows dialog box titled "sunny.ac.uk Properties" with a help icon and a close button in the top right corner. The dialog has five tabs: "General", "Start of Authority (SOA)", "Name Servers", "WINS", and "Zone Transfers". The "Start of Authority (SOA)" tab is selected. The dialog contains the following fields and controls:

- Serial number:** A text box containing "2002043030" and an "Increment" button to its right.
- Primary server:** A text box containing "dns.sunny.ac.uk." and a "Browse..." button to its right.
- Responsible person:** A text box containing "hostmaster.sunny.ac.uk." and a "Browse..." button to its right.
- Refresh interval:** A numeric input box with "8" and a dropdown menu set to "hours".
- Retry interval:** A numeric input box with "2" and a dropdown menu set to "hours".
- Expires after:** A numeric input box with "7" and a dropdown menu set to "days".
- Minimum (default) TTL:** A text box containing "1 :0 :0 :0".
- TTL for this record:** A text box containing "1 :0 :0 :0".

At the bottom of the dialog are three buttons: "OK", "Cancel", and "Apply".

Figure 7: Zone Properties Page

Masters, Slaves and Zone Transfers

10

Where to Locate a Slave

Configuring a Master Server

Configuring a Slave Server

Communications between Master and Slave Servers

Masters, Slaves and Zone Transfers

This section discusses the nature of the relationship between a domain's master name server (of which there must only be one) and its slaves (of which there may be none, or many). The master or primary server is that which is acknowledged to hold the authoritative data for a domain. Any number of slave or secondary servers may be defined for a domain. These machines, if queried, will always answer from their own authoritative data but obtain these data from the master server by a process called a zone transfer.

The *type* statement is used within the BIND configuration file to identify the server as either a master or a slave server for the domain.

10.1 Where to Locate a Slave

Delegating authority for a domain to more than one name server is advantageous because it increases resilience. In the event of one, or more, of the authoritative name servers failing to respond to a query, the resolver will post the query to another of the domain's authoritative servers. There is little additional benefit accruing from running multiple name servers on the same network segment. In the event of failure of the leased line that provides JANET connectivity, the additional server would not have added any benefit. A slave server should operate on a network independent of the segment upon which the master is running. For the smaller JANET connected organizations (e.g., FE colleges), UKERNA are prepared to run a slave name server. Alternatively, a neighbouring organization may be prepared to run a slave. For more information on UKERNA's Secondary Nameserver Service see:

http://www.ja.net/services/secondary_nameserver.html

10.2 Configuring a Master Server

Within the BIND configuration file, a server is defined as a master for a defined domain as follows:

```
zone "sunny.ac.uk" IN {
    type master;
    file "sunny.ac.uk.dns";
};
```

Note that the zone keyword is followed by the name and class of the zone. The following two fields specify that this server is the master name server for the zone and identify the zone file.

10.3 Configuring a Slave Server

Our example organization, Sunny College, has agreed to configure their name server as a slave for the neighbouring school, Riverbank High School. This is achieved by defining a new zone in Sunny College's configuration file.

```
zone "riverbank.hillingdon.sch.uk" IN {
    type slave;
    file "riverbank.hillingdon.sch.uk.dns";
    masters { 194.238.189.13 };
};
```

The numeric IP address of the zone's master name server must be specified so that the slave is able to initiate communications with the master.

10.4 Communications between Master and Slave Servers

The numerical parameters contained within a zone file's SOA record dictate how a slave name server communicates with the master. It is vital that a slave name server knows when the data on the master have been modified so that the modified zone file is copied to the slave. This process is termed a *zone transfer* and ensures that the master's and slaves' copies of the data remain reconciled.

The *refresh* interval dictates how often a slave makes contact with the domain's master server. Once a TCP virtual circuit has been established, the slave will compare the serial number in the SOA record of its copy of the data with the corresponding serial number from the master. If the master server's serial number is greater than that of the slave, this indicates to the slave that the data have changed. A *zone transfer* is then initiated in which the master server transfers a complete copy of its data to the slave. It is essential, therefore, that any modifications to a domain's data on the master server are followed by incrementing the serial number, otherwise the slaves will not register the changes. The Microsoft DNS server will automatically increment the serial number following any changes to a zone file. Users of the BIND software need to remember to do this manually. A common technique is to create the serial number by writing the date of the last modification to the zone file in reverse and appending a two digit number thus:

2002042301

If more than one change is made in a single day, then the last two digits can be incremented (allowing for 99 separate changes in one day!). By using this technique, the serial number will always increase and a casual glance can reveal the date the zone file was last modified.

If a slave is unable to establish a connection with its master for some reason, it will continually poll the master at an interval dictated by the *retry* value in the SOA record. If the master server has been unavailable for a lengthy period of time, then it may be preferable for the slave not to have any data pertaining to a given domain at all, rather than to answer queries based on (presumed) outdated zone files. If the slave has failed to establish a connection with the domain's master for a period of time equal to that specified in the *expire* interval, the slave will delete the data it holds on the domain.

10.4.1 DNS Notify

An alternative to the 'polling' mechanism exists whereby a master server informs its slaves when zone file data has been modified. When a slave receives a 'notify' request from the master server, it sends back a response which informs the master that it can cease sending the request messages. The slave then behaves as if the refresh timer had expired, and compares serial numbers with the master to determine whether a zone transfer should be initiated, just as described above.

Receiving a notify request from the zone's master server is insufficient to cause the slave to initiate a zone transfer, because this would render name servers vulnerable to an obvious denial-of-service attack in which a third party could dispatch numerous forged notify request messages to a target zone's slaves. This would in turn generate a flurry of zone transfers and thereby overwhelm the master name server.

The JANET name servers do not respond to DNS Notify.

DNS Queries and Querying Tools

11

Types of Query

The Query Process

Querying Tools

DNS Queries and Querying Tools

Understanding how queries are resolved is vital in order to troubleshoot problems relating to a DNS server. Several tools exist which enable the user manually to compose a query and examine the precise response to that query. This is invaluable when determining the reason for a DNS server failing to return the expected data. This section briefly describes how DNS queries are resolved and how to use common query tools.

11.1 Types of Query

There are two types of query that a resolver may issue; *recursive* and *iterative* queries. When a name server receives a recursive query, it is obliged to either return an answer or to state categorically that the data do not exist. In contrast, an server receiving an iterative query may either answer directly, or refer the device that issued the query to a source of more authoritative information.

11.2 The Query Process

The progress of a DNS query is depicted in *Figure 8*.

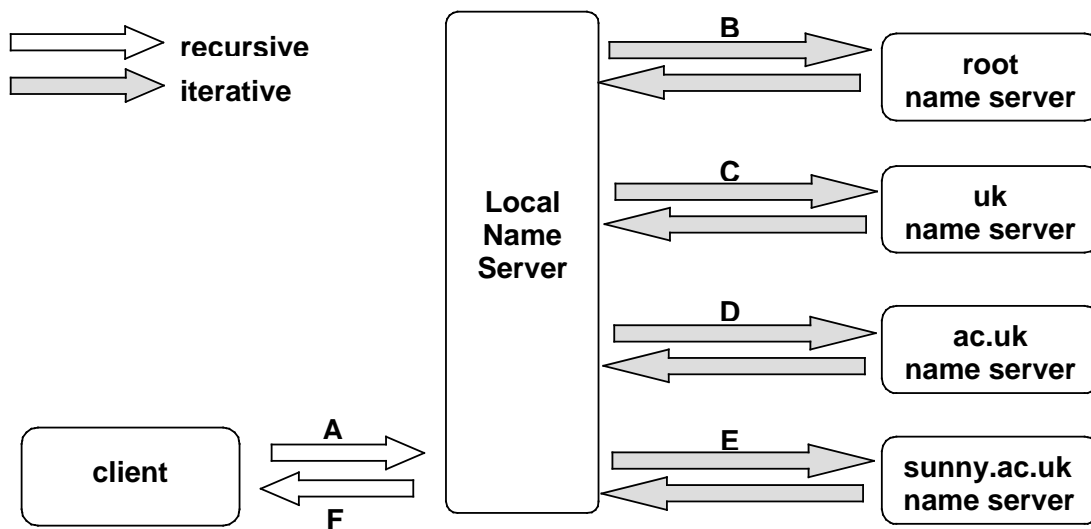


Figure 8: The Query Resolution Process

Each stage of the process may be annotated as follows.

- A** A client machine on the local network issues a query to the local DNS server it has been configured to use. This is always a recursive query, and so the server is obliged to return either an answer or an error stating there are no data available in the DNS to provide an answer (i.e., the client asked about a non-existent domain).
- B** Once the query has been received by the local DNS server, it is obliged to seek out the answer, and this is done by means of a series of iterative queries. The first of these is sent to one of the root name servers, which know the identity of the authoritative name servers for each of the top-level domains. If the root server is unable to answer the query from its own data, it advises the local name server to consult the authorita-

tive server for the top-most domain in the query (which is *uk.* in the example).

- C The original query is then posted to this second name server which will either answer from its own data, or advise of a 'closer' name server for the second-level domain (here *ac.uk.*).
- D If the name server for the *ac.uk.* domain is unable to answer the query from its own data, it advises the local name server of a server that is closer still to the domain being queried.
- E This is the authoritative server for the domain being queried, and so a definitive answer is returned to the querying name server.
- F Finally, the local name server is now able to answer the client machine's recursive query.

The process by which an iterative query may be answered with a referral to a 'closer', or more authoritative, name server is clearly possible because the answering name server holds the delegation information (or NS records) that point to the name servers for the delegated child domain.

11.2.1 Caching

This description of the query resolution process has ignored the effect of caching. In practice, each name server participating in the resolution process will retain any information received from a query in its cache for a period of time. Should a subsequent similar query be received, the server will then provide an answer from the cache. Many queries can be resolved with fewer network queries than in the complete example given. This greatly reduces the load on the root and top-level domain name servers.

Referring to the example in *Figure 8*, the local name server will retain the identities of the authoritative name servers for the *uk.*, *ac.uk.* and *sunny.ac.uk.* in its cache. These responses will remain in the cache for a period set by the TTL field for that record. Suppose that a client machine subsequently dispatches a recursive query for the MX records for the domain *megacorp.co.uk.* The local name server will not need to consult the root name servers as it already knows the identity of the authoritative name servers for the *uk.* domain. The query process will therefore start with the local name server querying one of the *uk.* name servers for the authoritative name servers of the *co.uk.* domain.

11.2.2 Root Name Servers

It should be apparent that queries cannot be resolved unless the local name server is provided with the names and IP addresses of the root name servers in advance. These data may be readily obtained by anonymous ftp to the server *ftp.rs.internic.net* and retrieving the file called *named.root* from the *domain* subdirectory. Once loaded into memory, these data are treated differently from ordinary cache data inasmuch as the root name server identities are not discarded once the TTL drops to zero.

11.3 Querying Tools

It is often useful to be able manually to compose queries and inspect the results as part of troubleshooting DNS-related problems. Two common command line interface (CLI) tools are discussed in this section. Both tools, while having slightly different syntax and presentation of the results, perform the same task. They allow the class and type of record to be specified and queries may be directed at a particular name server.

11.3.1 nslookup

This tool is in widespread use as it is distributed with all common network operating systems. It may be run in an interactive or non-interactive fashion, only the latter is discussed here. The syntax of the command is as follows:

```
nslookup <-cl=class> <-q=type> query <server>
```

Parameters in angle brackets are optional. If the type of RR is not specified then an 'A' record is assumed if the query is a domain name and a PTR is assumed if the query is an IP address. The output of a typical query is:

```
nslookup -q=MX ja.net
Server:  andromache.ulcc.ac.uk
Address: 128.86.248.193

Non-authoritative answer:
ja.net  MX preference = 1, mail exchanger = rimmer.ja.net

ja.net  nameserver = NS0.ja.net
ja.net  nameserver = NS1.CS.UCL.AC.UK
ja.net  nameserver = NS1.SURFNET.NL
ja.net  nameserver = NS4.ja.net
rimmer.ja.net  internet address = 128.86.8.39
NS0.ja.net    internet address = 128.86.1.20
NS0.ja.net    internet address = 193.63.94.20
NS1.CS.UCL.AC.UK    internet address = 128.16.5.32
NS1.SURFNET.NL internet address = 192.87.106.101
NS4.ja.net    internet address = 193.62.157.66
```

The query has asked for the MX records for the domain ja.net. The first two lines of the output specify the name and address of the server that returned the result. Note how the next line states that the answer is 'non-authoritative'. This means that the local name server has provided the answer from its cache, rather than by issuing iterative queries over the Internet. The next line provides the answer to the query. The following lines list the NS records for the domain and all the IP addresses of all the relevant hosts. This section demonstrates one aspect of nslookup that some people dislike; the returned data are not particularly well partitioned and formatted.

If the command 'nslookup' is issued without any parameters, then one can operate with the program interactively. Typing 'exit' at the prompt returns the user back to their OS shell.

11.3.2 dig

Users of the BIND server software will also have access to the domain internet proper (dig). The syntax is subtly different from nslookup:

```
dig          @server  query  <type>  <class>
```

The output from a 'dig' query (dig @localhost ja.net MX) is given below.

```
; <<>> DiG 8.3 <<>> @localhost ja.net MX
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 6
;; QUERY SECTION:
;;      ja.net, type = MX, class = IN

;; ANSWER SECTION:
ja.net.                22h52m32s IN MX  1 rimmer.ja.net.

;; AUTHORITY SECTION:
ja.net.                18h29m43s IN NS  NS0.ja.net.
ja.net.                18h29m43s IN NS  NS1.CS.UCL.AC.UK.
ja.net.                18h29m43s IN NS  NS1.SURFNET.NL.
ja.net.                18h29m43s IN NS  NS4.ja.net.

;; ADDITIONAL SECTION:
rimmer.ja.net.        22h52m32s IN A   128.86.8.39
NS0.ja.net.          19h31m38s IN A   128.86.1.20
NS0.ja.net.          19h31m38s IN A   193.63.94.20
NS1.CS.UCL.AC.UK.    4d2h52m32s IN A   128.16.5.32
NS1.SURFNET.NL.     2d20h9m23s IN A   192.87.106.101
NS4.ja.net.          19h31m38s IN A   193.62.157.66

;; Total query time: 0 msec
;; FROM: andromache to SERVER: localhost 127.0.0.1
;; WHEN: Thu Apr 25 15:40:54 2002
;; MSG SIZE  sent: 24  rcvd: 237
```

It is interesting to note that the format of this output very closely resembles a zone file. Note how the TTL value associated with each record indicates the length of time it will remain in the local name server's cache. If the same query is posted repeatedly to the same resolver, these values will be seen to drop each time (as expected).

Dynamic DNS

12

Dynamic DNS

Traditionally, zone files have been static objects in which the resource records are modified by hand. For a public, Internet-facing server that contains relatively few records, this is acceptable. However, when DNS is used for internal hostname resolution, its inherently static nature can prove to be a drawback.

If a DHCP server is used to distribute IP addresses to client machines on the local network, it is desirable that each machine is registered automatically with the network's name resolver once an IP address has been issued. Networks based on Windows NT4 have recourse to WINS servers with which client machines automatically register, and are then free to query the database. If a different network operating system is in use, then DNS may be the only option for internal hostname resolution. In these circumstances, it is possible to instruct a name server to accept automatic updates from a specified host. Often, this will be a DHCP server which updates the internal DNS immediately after successfully issuing a new lease on an IP address. The BIND software is able to accept dynamic updates, but must be configured to do so:

```
/* Internal zone file */
zone "sunny-college.ac.uk" {
    type master;
    file "sunny-college.ac.uk.dns";
    allow-update { 192.168.100.23; };
};
```

Note how this zone will accept updates from a single specified host. There is a tool called *nsupdate*, which is part of the BIND distribution, with which updates may be created manually.

Within Windows 2000, dynamic DNS may either be enabled or disabled. Additional access-control list security is not available unless the DNS server is integrated into an active directory. Under these circumstances, an additional security tab will become available on the server's properties page. The security implications of accepting dynamic DNS updates are discussed in *Section 13.2*.

Securing a Public DNS Server

13

Restrict Zone Transfers

Restrict Dynamic Updates

Restrict Recursive Queries

Securing a Public DNS Server

A name server that is necessarily exposed to the public Internet should be subject to some elementary security considerations. Having compromised an organization's name server a malicious third party could be able to modify DNS resource records, causing traffic to the organization's other servers (e.g., web and mail) to be redirected elsewhere. This redirection would likely be to hosts under the control of the attacker. Most importantly, security flaws that have been remedied in later versions of the software must be corrected with immediate effect. All network managers should ensure that they are in receipt of security advisories from JANET-CERT, and their operating system manufacturers, and that operating systems are patched in accordance with the manufacturer's guidelines. Apart from these general observations, there are several actions that may be taken to improve the security of your name servers.

13.1 Restrict Zone Transfers

A name server should never accede to a request for a zone transfer from just any device on the Internet. Generally speaking, a master server should only perform zone transfers with its slaves. A slave name server should not be configured to respond to any zone transfers requests at all. Clearly, circumstances may dictate that other machines should be able to transfer zones from a name server. These could include other hosts on the local network (for troubleshooting purposes) and possibly from other recognised organizations for support purposes (e.g., Regional Support Centres).

The BIND software allows for simple access-control lists to be constructed in which the source IP address of an inbound connection may be restricted. Zone transfers may be restricted to listed source IP addresses using the 'allow-transfer' option in a BIND configuration file.

```
allow-transfer { 128.86/16; 193.63.106.103; };
```

In the above example, zone transfers will only be permitted to the official slave and to machines with an IP address from the listed class B network. As shown in the complete configuration file for Sunny College in *Appendix 2*, this may be configured as a global option, or specific to a particular zone. Restricting zone transfers in this fashion prevents crackers from deliberately taxing a name server or from listing the contents of zone files to identify targets.

Under Windows 2000, zone transfers may be restricted to either the official slave name servers or to a list of IP addresses. This facility may be found under the *Zone Transfers* tab of the zone's properties page.

13.2 Restrict Dynamic Updates

A name server that is exposed to the Internet should not generally accept dynamic updates. If, for some reason, this is unavoidable then the server should never accept updates from an unknown source. In fact, BIND cannot be configured to accept dynamic updates without specifying a list of recognised hosts from which these updates will be accepted.

13.3 Restrict Recursive Queries

An Internet-visible name server is vulnerable to spoofing attacks if it answers recursive queries from any source. In this type of attack, the cracker directs a query about a zone under his control to the name server he wishes to compromise. The target name server is then forced to query the cracker's server and receives bogus data, which it stores in its cache. Sites may also wish to protect their network resources by prohibiting their name server from acting as a general resolver for anybody on the Internet.

The BIND software offers three strategies to restrict a server's willingness to answer recursive queries.

13.3.1 Disable Recursion

The effect of this is to instruct a name server never to dispatch queries on behalf of other name servers or resolvers. The server will only respond to iterative queries received as part of the normal resolution cycle. In other words, recursion can only be disabled on a name server whose only function is to answer from its own authoritative data. A server configured in this way will not be able to respond to queries about other domains, even if they come from internal clients.

```
options {
    directory "/var/named";
    recursion no;
    fetch-glue no;
};
zone "sunny.ac.uk" {
    type master;
    file "sunny.ac.uk.dns";
    allow-transfer { 193.63.106.103; };
};
```

In this example, note the global option which turns recursion off altogether. This name server will decline to answer all queries, except those about its authoritative zones. The *fetch-glue* option is used to control whether a server resolves NS records for which it does not have the corresponding A records. This should be disabled, as in the example, for any name server configured only to answer from its own authoritative data. It follows that a name server configured in this fashion will not build up a cache.

13.3.2 Control Queries

Most organizations will wish to use their name server(s) to both host authoritative data about their own domains and to resolve queries about any domain name on behalf of their own internal hosts.

Queries about an authoritative zone can potentially come from any source on the Internet. In contrast, queries about non-authoritative zones should predictably come from only internal hosts. Judicious deployment of the *allow-query* option at key points within the configuration file can effect this set-up.

```

acl internal { 212.219.172.88/29; };
options {
    directory "/var/named";
    allow-transfer { 193.63.106.103; };
    allow-query { internal; };
};
zone "sunny.ac.uk" {
    type master;
    file "sunny.ac.uk.dns";
    allow-query { any; };
};

```

This example shows how the global option, restricting queries to a defined list of internal hosts, can be overridden for selected authoritative zones where queries received from any source should be answered. It also demonstrates how source IP addresses may be encapsulated into a named access-control list, which can be referred to at any subsequent point in the configuration file. Note that the *allow-transfer* statement could also have been accompanied by an ACL defining the list of slave servers.

13.3.3 Restrict Recursive Queries

This is a slightly different technique, which has the same effect as previous example. The aim is again to prevent the server from answering recursive queries emanating from hosts outside the local network. By selectively enabling recursion for a defined list of hosts, the server will only answer queries from the Internet if they concern the server's authoritative zones.

```

acl internal { 212.219.172.88/29; };
options {
    directory "/var/named";
    allow-recursion { internal; };
};
zone "sunny.ac.uk" {
    type master;
    file "sunny.ac.uk.dns";
    allow-transfer { 193.63.106.103; };
};

```

The Windows 2000 DNS server is less flexible than BIND inasmuch as recursion may either be turned on or off. It is therefore not possible to selectively answer recursive queries for nominated querying hosts.

Common Errors

14

Illegal Characters

Failing to Increment the Serial Number

Inappropriate Numerical Parameters in the SOA Resource Record

Lame Delegations

Inclusion of Child Records in Parent Domain's Zone File

Missing Terminating Periods

Aliases in the RDATA Portion of Resource Records

Including IP Addresses or Wildcards in Resource Records

Advertising Private IP Addresses

Common Errors

A useful summary of common mistakes and oversights follows.

14.1 Illegal Characters

As mentioned earlier (see *Section 3.1*), labels may be composed solely of letters, digits and hyphens and must not exceed 63 characters in length. The label must begin with a letter and may end with a letter or a digit.

14.2 Failing to Increment the Serial Number

It is all too easy to edit a zone file, and then to forget to increment the file's serial number. The domain's slaves will not then retrieve a new copy of the modified zone file. The Microsoft DNS servers will automatically perform this task, but users of BIND will have to remember to do this manually.

A related oversight is failing to reload (`kill -HUP `cat /etc/named.pid``) a modified zone file. When a name server process is started, it parses the zone files and reads them into memory. Queries are then answered from the data stored in volatile RAM. If the files on disc are modified, the server must be instructed to read them back into memory.

14.3 Inappropriate Numerical Parameters in the SOA Resource Record

The refresh, retry, expire and TTL parameters should be assigned values of a magnitude that reflect the parameter's purpose. The author recommends that whatever the absolute values, the first three should be related as follows.

`expire > refresh > retry`

High TTL values will result in out of date information remaining in caches all over the Internet. Sensible defaults are given in *Table 6*.

Parameter	Length	Value (seconds)
refresh	8 hours	28800
retry	2 hours	7200
expire	7 days	604800
TTL	1 day	86400

Table 6: Common Values for SOA Parameters

14.4 Lame Delegations

These are domains that have been delegated either to non-existent name servers or to servers that are not running the appropriate zone file. Often, delegations fail in this fashion because the name servers for a domain are moved to stations with a different hostname or IP address, and the administrator of the parent is not warned of the change

in advance. Consequently, the NS and any associated glue records which create the delegation in the parent domain's zone file become outdated. It is less common (although not unknown!) for a delegation to be lame from the outset.

14.5 Inclusion of Child Records in Parent Domain's Zone File

When a child domain is delegated, it is important that the only records pertaining to the child within the parent zone file are those concerned with the delegation. Including any other data in the parent's file will cause the equivalent records in the child file to be inactive. Given that the rationale for the delegation is that the delegated domain will be maintained independently of the parent, this error defeats the purpose of delegating authority.

14.6 Missing Terminating Periods

Failing to deploy terminating periods when they are required can quickly cause havoc, with a zone file's origin appended to fully-qualified domain names in a resource record. The two primary examples concern reverse zone files and CNAME resource records.

14.7 Aliases in the RDATA Portion of Resource Records

The right-hand side of a resource record should never point to an alias (a label that is defined by a CNAME record) but to a name that is defined by an 'A' record.

	MX 10	mail
	MX 20	obsidian
obsidian	A	212.219.172.91
garnet	A	212.219.172.93
mail	CNAME	garnet

In this example, the first MX record is incorrectly mapped to an alias, while the second MX record is correctly defined. In the case of MX records, mail routing loops can ensue if the RDATA portion of the RR maps to an alias.

14.8 Including IP Addresses or Wildcards in Resource Records

The RDATA portion of NS and MX records should point to a name, not an IP address. In fact, if an IP address is mistakenly used, it will not be treated as such by the name server software but as a name (and an illegal one at that!).

14.9 Advertising Private IP Addresses

Sites that deploy private IP addresses (10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16) internally with network address translation on the border router should take care that 'A' records in the public (Internet-facing) DNS do not map to the unreachable private IP addresses.

Appendix 1: UKERNA Templates

Template for Requesting a New Domain Name Under ac.uk/gov.uk

Template for Requesting Modification to a Domain Name Under ac.uk/gov.uk

UKERNA Templates

These pro-forma templates may be found on the world wide web under:

http://www.ja.net/documents/naming/names_ac_gov.html

Applications for new domain names should be based on the first template, and submitted to **naming-ac@ukerna.ac.uk** for .ac.uk names or to **naming-gov@ukerna.ac.uk** for .gov.uk names. Modifications to an existing name should be based on the second template and submitted instead to **naming-admin@ukerna.ac.uk**. Enquiries may also be sent to this second mail address.

A1.1 Template for Requesting a New Domain Name Under ac.uk/gov.uk

requested name:
to represent :
domain owner:

descr:

admin-c:
address:
phone:
fax-no:
e-mail:

tech-c:
address:
phone:
fax-no:
e-mail:

nserver name & ip address:
nserver name & ip address:

payment method:

Approved ISP Mem.No:

notes:

A1.2 Template for Requesting Modification to a Domain Name Under ac.uk/gov.uk

domain name:
representing:
name owner:

admin-c:
address:
phone:
fax-no:
e-mail:

tech-c:
address:
phone:
fax-no:
e-mail:

nserver name & ip address:
nserver name & ip address:

Approved ISP Mem.No:

Previous ISP (if known):

notes:

Appendix 2: Complete Zone Files for Sunny College

Forward Zone File

Reverse Zone File

BIND Configuration File

Complete Zone Files for Sunny College

Portions of the zone files for the example organization Sunny College have appeared in the text. The complete files are given here for ease of reference. In contrast with the examples within the text, the following abbreviations have been used where appropriate:

Appended domains. The file's origin is appended to all domain names missing a terminating period.

@ Notation. Where the domain name is the same as the origin, the name may be specified as '@'. As shown here, this is most often seen in SOA records.

Omitted Owner Field. If the 'owner' (first column) of a resource record consists of white space then the name from the last record is used.

A2.1 Forward Zone File

```
$ORIGIN sunny.ac.uk.
@ SOA dns.sunny.ac.uk.
hostmaster.sunny.ac.uk. (
                                2000120801 ;serial
                                28800      ;refresh
                                7200       ;retry
                                604800     ;expire
                                86400)     ;minimum TTL
                                NS dns
                                NS ns3.ja.net.
                                MX 10 mail
student MX 10 mail
;host addresses
gw A 212.219.172.89
garnet A 212.219.172.91
www.student A 212.219.172.94
;multi-homed addresses
ftp A 212.219.172.92
A 212.219.172.93
;aka
dns A 212.219.172.91
mail A 212.219.172.91
;aliases
www CNAME webfarm1.happyhosters.co.uk.
```

A2.2 Reverse Zone File

```
$ORIGIN 88/29.172.219.212.in-addr.arpa.
@ SOA dns.sunny.ac.uk.
hostmaster.sunny.ac.uk. (
                                2000120801 ;serial
                                28800      ;refresh
                                7200       ;retry
                                604800     ;expire
                                86400)     ;minimum TTL
                                NS dns.sunny.ac.uk.
```

	NS	ns3.ja.net.
89	PTR	gw.sunny.ac.uk.
91	PTR	garnet.sunny.ac.uk.
92	PTR	ftp.sunny.ac.uk.
93	PTR	ftp.sunny.ac.uk.
94	PTR	www.student.sunny.ac.uk.

A2.3 BIND Configuration File

```
acl internal { 212.219.172.88/29; };
options {
    directory "/var/named";
    allow-recursion { internal; };
    allow-transfer { 128.86/16; 193.63.106.103; };
};
zone "." IN {
    type hint;
    file "root.dns";
};
zone "sunny.ac.uk" IN {
    type master;
    file "sunny.ac.uk.dns";
};
zone "88/29.172.219.212.in-addr.arpa." IN {
    type master;
    file "88-29.172.219.212.in-addr.arpa.dns";
};
zone "riverbank.hillingdon.sch.uk" IN {
    type slave;
    file "riverbank.hillingdon.sch.uk.dns";
    masters { 194.238.189.13 };
};
zone "0.0.127.in-addr.arpa" IN {
    type master;
    file "0.0.127.in-addr.arpa.dns";
};
```

Appendix 3: Glossary of Terms

Appendix 3: Glossary of Terms

This glossary is designed to accompany the text of the Guide; it should not be regarded as a comprehensive listing of all terms and acronyms related to DNS.

BIND	<i>Berkeley Internet Name Domain</i> . The most popular implementation of DNS. It has been ported to most flavours of UN*X and Windows NT.
Class	Resource records, which contain the data associated with domain names, belong to a particular <i>class</i> , each of which pertains to a particular network protocol or a type of software. Only the Internet (IN) class is of relevance to DNS.
Classful	<i>Classful networks</i> are those in which the network and host portions of an IP number lie on the octet boundaries of the address.
Classless	Networks in which any number of contiguous bits of an IP address are assigned to the network portion of the address, causing the network and host portions of an IP address to cross octet boundaries.
Delegation	The process by which authority for a subdomain is assigned to a different organization's name servers.
Domain	A portion of the domain name space encompassing a node and all of the child nodes that derive from it.
Dynamic	The method by which records are automatically inserted into a zone file by, for example, a DHCP server.
Iterative	One of two types of DNS query. Upon receipt of an iterative query, a name server that is not authoritative for the domain being queried will respond with a referral to a more local name server. <i>See also 'Recursive'</i> .
Label	The name of a single node within the DNS tree. Domain names are composed of labels separated by period characters.
Master	A master (also known as <i>primary</i>) name server obtains its authoritative data for a zone from a file stored on its host which is maintained either by hand or by dynamic insertion of resource records. <i>See also 'Slave'</i> .
Origin	The domain that is represented by a zone file. Hostnames and subdomains within a zone file are all relative to this starting point.
Owner	The first column of a resource record, representing the domain name.
Recursive	One of two types of DNS query. A name server that receives a recursive query must respond with either an answer (from its authoritative data or cache) or with an error stating no data are available. Referrals to other name servers are not permitted. <i>See also 'Iterative'</i> .
Root	The topmost node of the domain name system. Delegation information for the top-level domains (the immediate children of the root node such as <i>.com</i> and <i>.uk</i>) are maintained on the root node's name servers.

Slave	A slave (also know as <i>secondary</i>) name server obtains its authoritative data for a zone from a file stored on its host which is updated by means of automated zone transfers from the zone's master server. <i>See also 'Master'</i> .
Type	Resource records of a given class are categorised by their type (e.g., CNAME, MX, <i>etc.</i>), which effectively defines the purpose of the record.
Wildcard	A single resource record that will match any label, except those for which specific data is available within the zone file.
Zone	A single node within the domain name space for which a zone file, running upon a master name server, exists.

Information Services welcome feedback on the documentation issued by UKERNA. If you have any comments on the content, presentation, etc. of this guide please e-mail:

documentation@ukerna.ac.uk

or write to:

Information Services, UKERNA, Atlas Centre, Chilton, Didcot, Oxon, OX11 0QS

UKERNA manages the networking programme on behalf of the higher and further education and research community in the United Kingdom. JANET, the United Kingdom's academic and research network, is funded by the Joint Information Systems Committee (JISC).

For further information please contact:

JANET Customer Service

UKERNA

Atlas Centre, Chilton, Didcot

Oxfordshire, OX11 0QS

Tel: +44 (0) 1235 822 212

Fax: +44 (0) 1235 822 397

E-mail: service@janet.ac.uk

Copyright:

This document is copyright The JNT Association trading as UKERNA. Parts of it, as appropriate, may be freely copied and incorporated unaltered into another document unless produced for commercial gain, subject to the source being appropriately acknowledged and the copyright preserved. The reproduction of logos without permission is expressly forbidden. Permission should be sought from JANET Customer Service.

Screen shots reprinted by permission from Microsoft Corporation.

Trademarks:

JANET®, SuperJANET® and UKERNA® are registered trademarks of the Higher Education Funding Councils for England, Scotland and Wales. The JNT Association is the registered user of these trademarks.

Disclaimer:

The information contained herein is believed to be correct at the time of issue, but no liability can be accepted for any inaccuracies.

The reader is reminded that changes may have taken place since issue, particularly in rapidly changing areas such as internet addressing, and consequently URLs and e-mail addresses should be used with caution.

The JNT Association cannot accept any responsibility for any loss or damage resulting from the use of the material contained herein.

Availability:

Further copies of this document may be obtained from JANET Customer Service at the above address.

This document is also available electronically from:

http://www.ja.net/documents/tg_dns.pdf



© The JNT Association 2002
GD/JANET/TECH/003 (02/06)

**Joint Information
Systems Committee**

