Chapter 18

Domain Name System (DNS)

CONTENTS

- NAME SPACE
- DOMAIN NAME SPACE
- DISTRIBUTION OF NAME SPACE
- DNS IN THE INTERNET
- RESOLUTION
- DNS MESSAGES
- TYPES OF RECORDS
- COMPRESSION
- EXAMPLES
- DDNS
- ENCAPSULATION

18.1

NAME SPACE

- the names must be unique because the addresses are unique. A name space that maps each address to a unique name can be organized in two ways: flat or hierarchical.

- 1. Flat
- 2.Hierarchical

Flat:

- In a flat name space, a name is assigned to an address. A name in this space is a sequence of characters without structure
- The main disadvantage of a flat name space is that it cannot be used in a large system such as the Internet because it must be centrally controlled to avoid ambiguity and duplication.

- Hierarchical:In a hierarchical name space, each name is made of several parts. The first part can define the nature of the organization, the second part can define the name of an organization, the third part can define departments in the organization, and so on. In this case, the authority to assign and control the name spaces can be decentralized. A central authority can assign the part of the name that defines the nature of the organization and the name of the organization. The responsibility of the rest of the name can be given to the organization itself.
- For example, assume two colleges and a company call one of their computers challenger. The first college is given a name by the central authority such as fhda.edu, the second college is given the name berkeley.edu, and the company is given the name smart.com. When each of these organizations adds the name challenger to the name they have already been given, the end result is three distinguishable names: challenger.fhda.edu, challenger.berkeley.edu, and challenger.smart.com. The names are unique without the need for assignment by a central authority. The central authority controls only part of the name, not the whole.

18.2

DOMAIN NAME SPACE

- To have a hierarchical name space, a domain name space was designed. In this design the names are defined in an inverted-tree structure with the root at the top. The tree can have only 128 levels: level 0 (root) to level 127

Figure 18-1

Domain name space

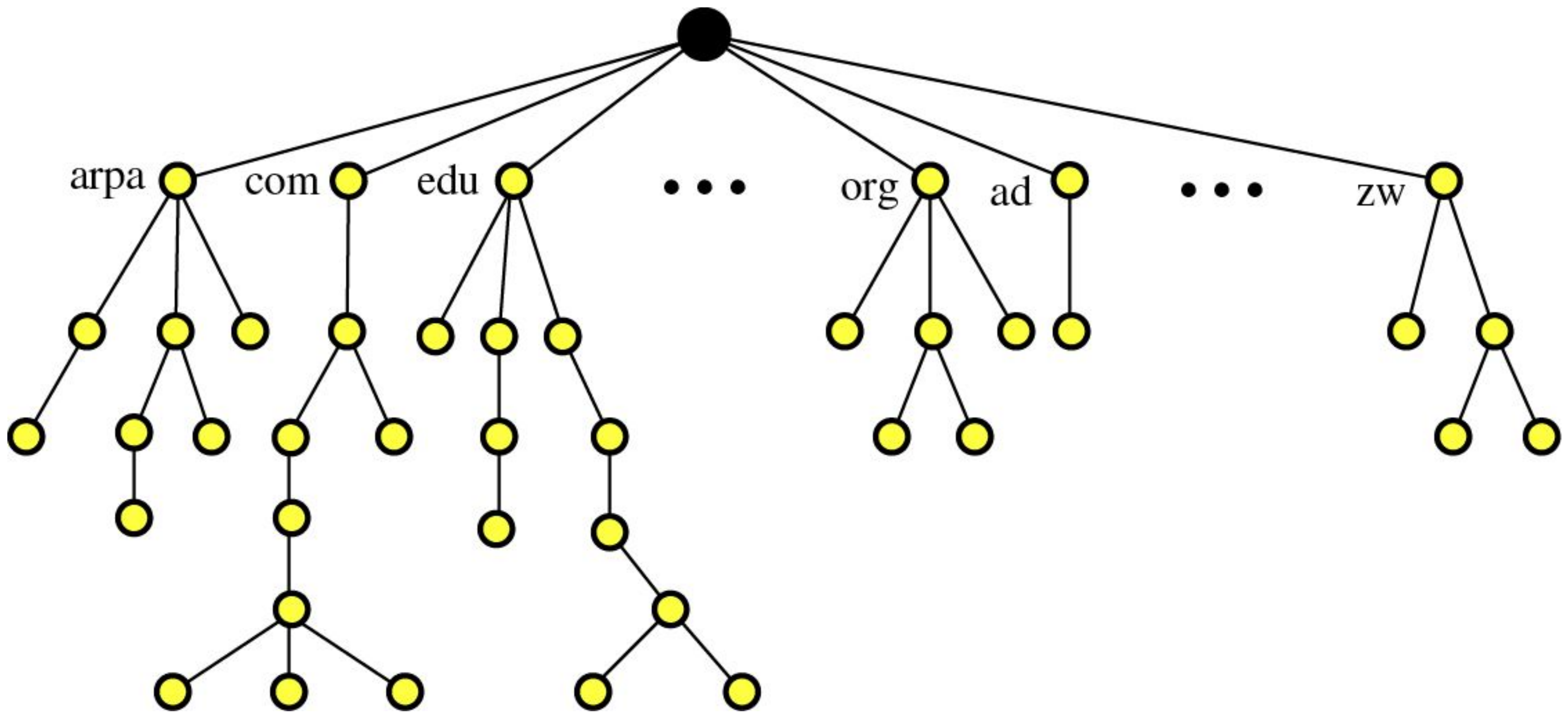
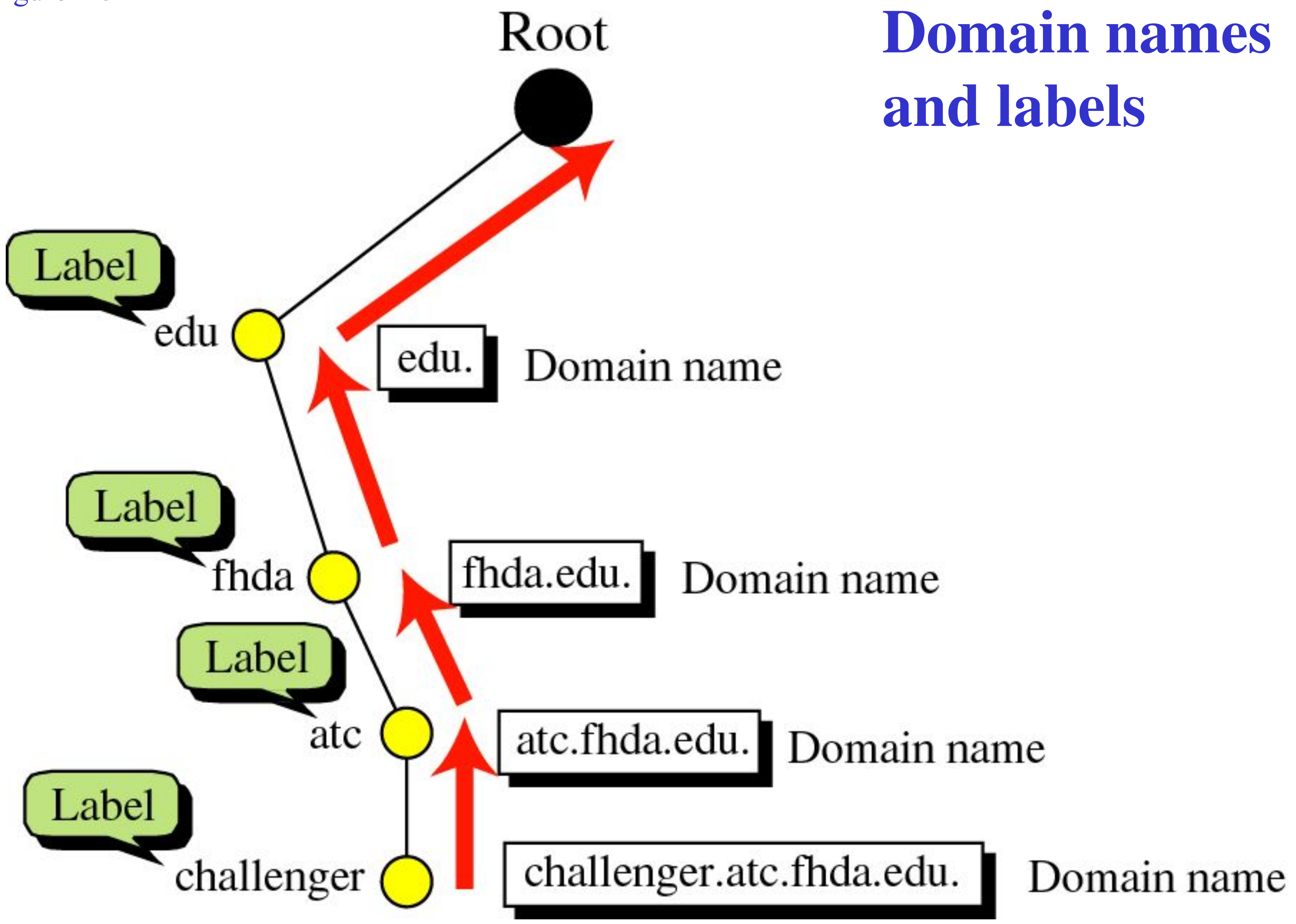


Figure 18-2

Domain names and labels



FQDN and PQDN

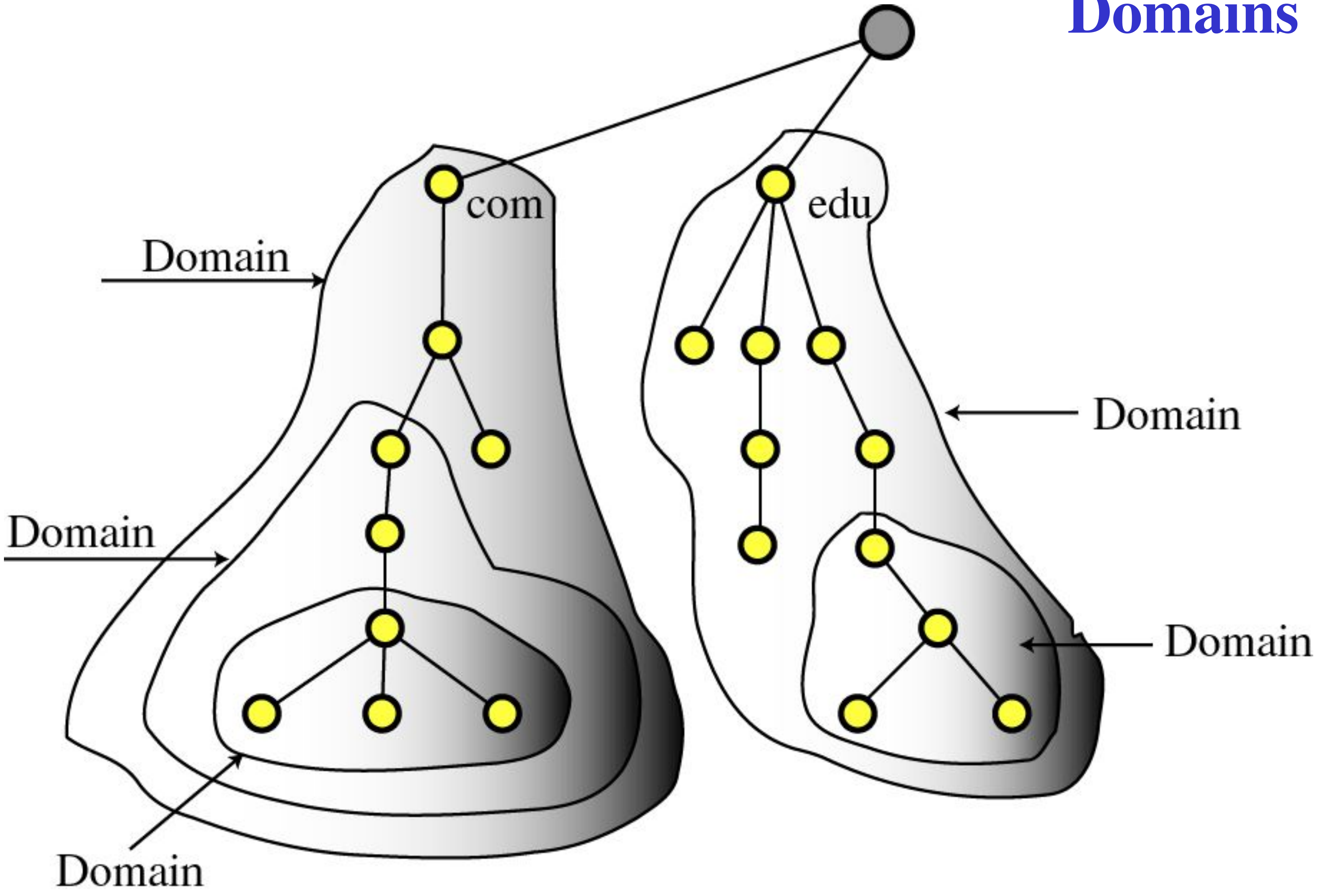
FQDN	PQDN
challenger.atc.fhda.edu.	challenger.atc.fhda.edu
cs.hmme.com.	cs.hmme
www.funny.int.	www

- If a label is not terminated by a null string, it is called a partially qualified domain name (PQDN).

- FQDN is a domain name that specifies the exact location of host in a tree hierarchy of the DNS.
- If the name isnt fully specified and is missing certain components it is called PQDN

Figure 18-4

Domains



- A domain is a subtree of the domain name space. The name of the domain is the name of the node at the top of the subtree. Note that a domain may itself be divided into domains (or subdomains as they are sometimes called).

18.3

DISTRIBUTION OF NAME SPACE

Figure 18-5

Hierarchy of name servers

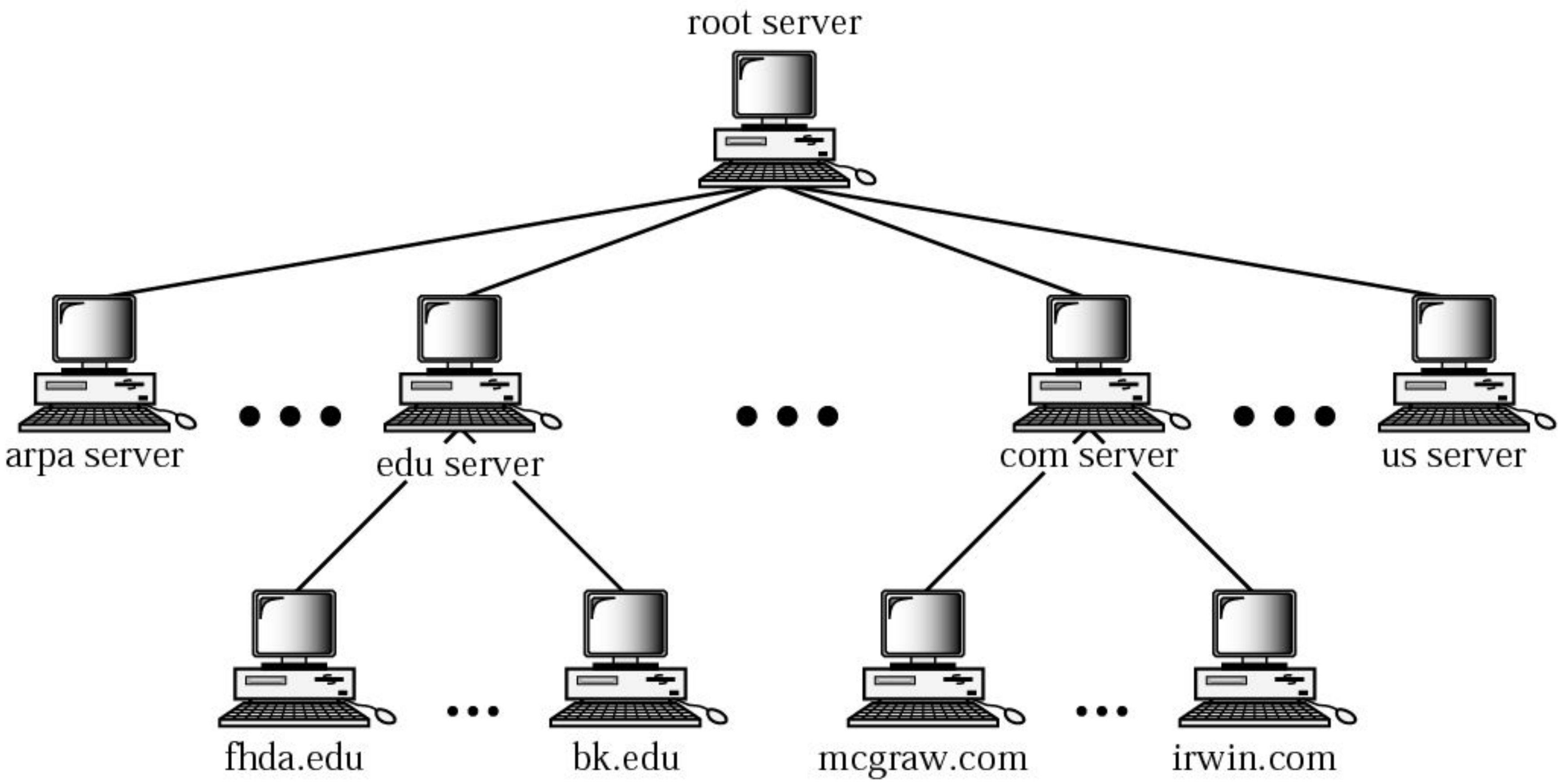
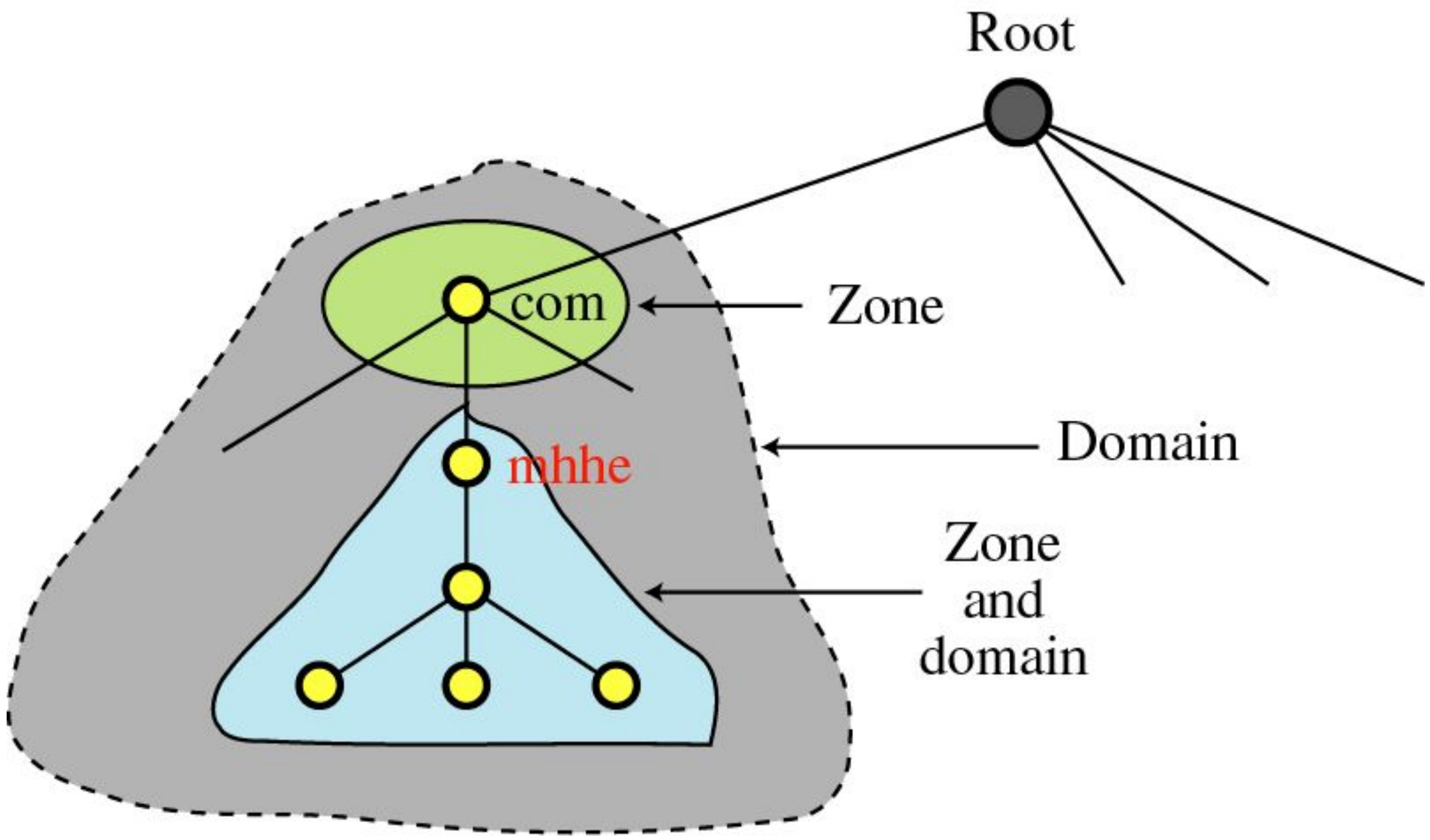


Figure 18-6

Zones and domains



- Zone :
- Since the complete domain name hierarchy cannot be stored on a single server, it is divided among many servers. What a server is responsible for or has authority over is called a zone. We can define a zone as a contiguous part of the entire tree.
- If a server accepts responsibility for a domain and does not divide the domain into smaller domains, the "domain" and the "zone" refer to the same thing.
- The server makes a database called a zone file and keeps all the information for every node under that domain. However, if a server divides its domain into subdomains and delegates part of its authority to other servers, "domain" and "zone" refer to different things.
- The information about the nodes in the subdomains is stored in the servers at the lower levels, with the original server keeping some sort of reference to these lower-level servers. Of course the original server does not free itself from responsibility totally: It still has a zone, but the detailed information is kept by the lower-level servers

- **DNS Zone** Levels
- The **Domain** Name System (**DNS**) defines a **domain** namespace, which specifies

Top Level **Domains** (such as ".com"),

second-level **domains**, (such as "acme.com")
lower-level **domains**, also called subdomains (such as "support.acme.com"). Each of these levels can be a **DNS zone**.

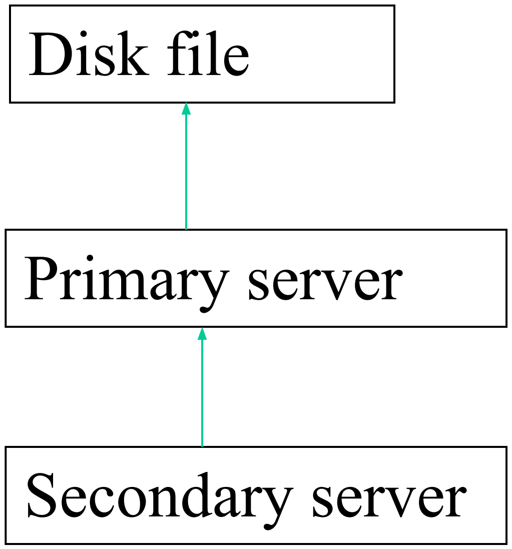
- ***Root Server*** :A root server is a server whose zone consists of the whole tree. A root server usually does not store any information about domains but delegates its authority to other servers, keeping references to those servers. There are several root servers, each covering the whole domain name space. The root servers are distributed all around the world.
- ***Primary and Secondary Servers*** :DNS defines two types of servers: primary and secondary.
- A primary server is a server that stores a file about the zone for which it is an authority. It is responsible for creating, maintaining, and updating the zone file. It stores the zone file on a local disk.
- A secondary server is a server that transfers the complete information about a zone from another server and stores the file on its local disk. The secondary server neither creates nor updates the zone files. If updating is required, it must be done by the primary server, which sends the updated version to the secondary.
- The primary and secondary servers are both authoritative for the zones they serve. The idea is not to put the secondary server at a lower level of authority but to create redundancy for the data so that if one server fails, the other can continue serving clients. Note also that a server can be a primary server for a specific zone and a secondary server for another zone. Therefore, when we refer to a server as a primary or secondary server, we should be careful about which zone we refer to.

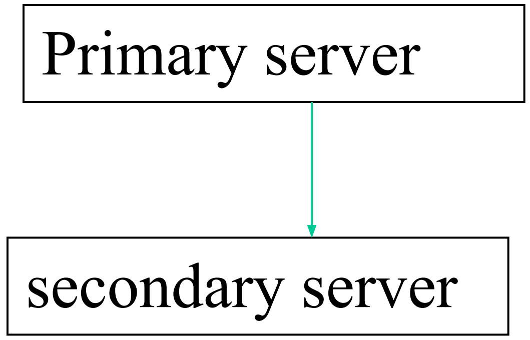
Note

A primary server loads all information from the disk file;

the secondary server loads all information from the the primary server.

When the primary downloads information from the secondary, it is called zone transfer.





Zone transfer

18.4

DNS IN THE INTERNET

Figure 18-7

DNS in the Internet

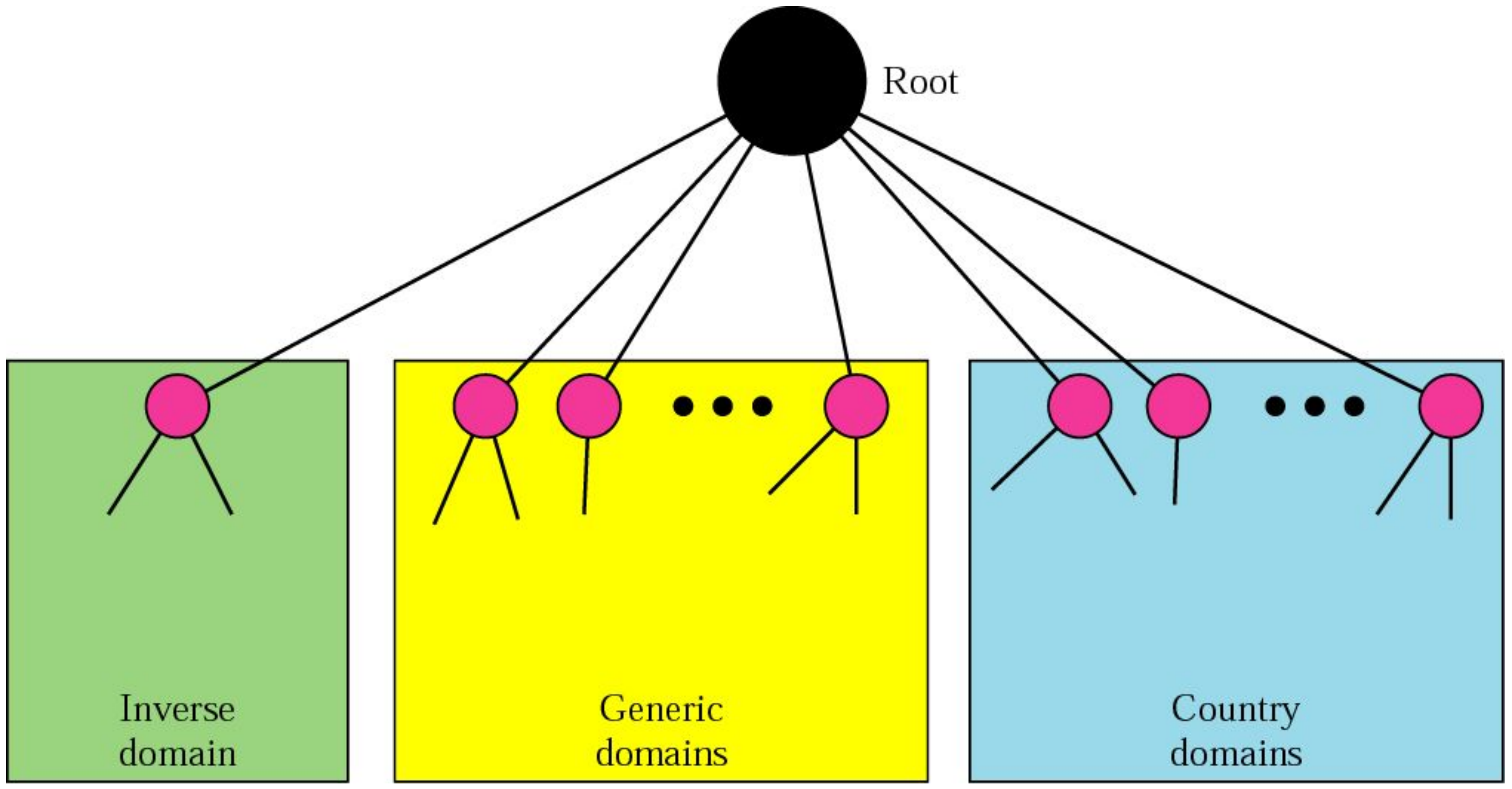
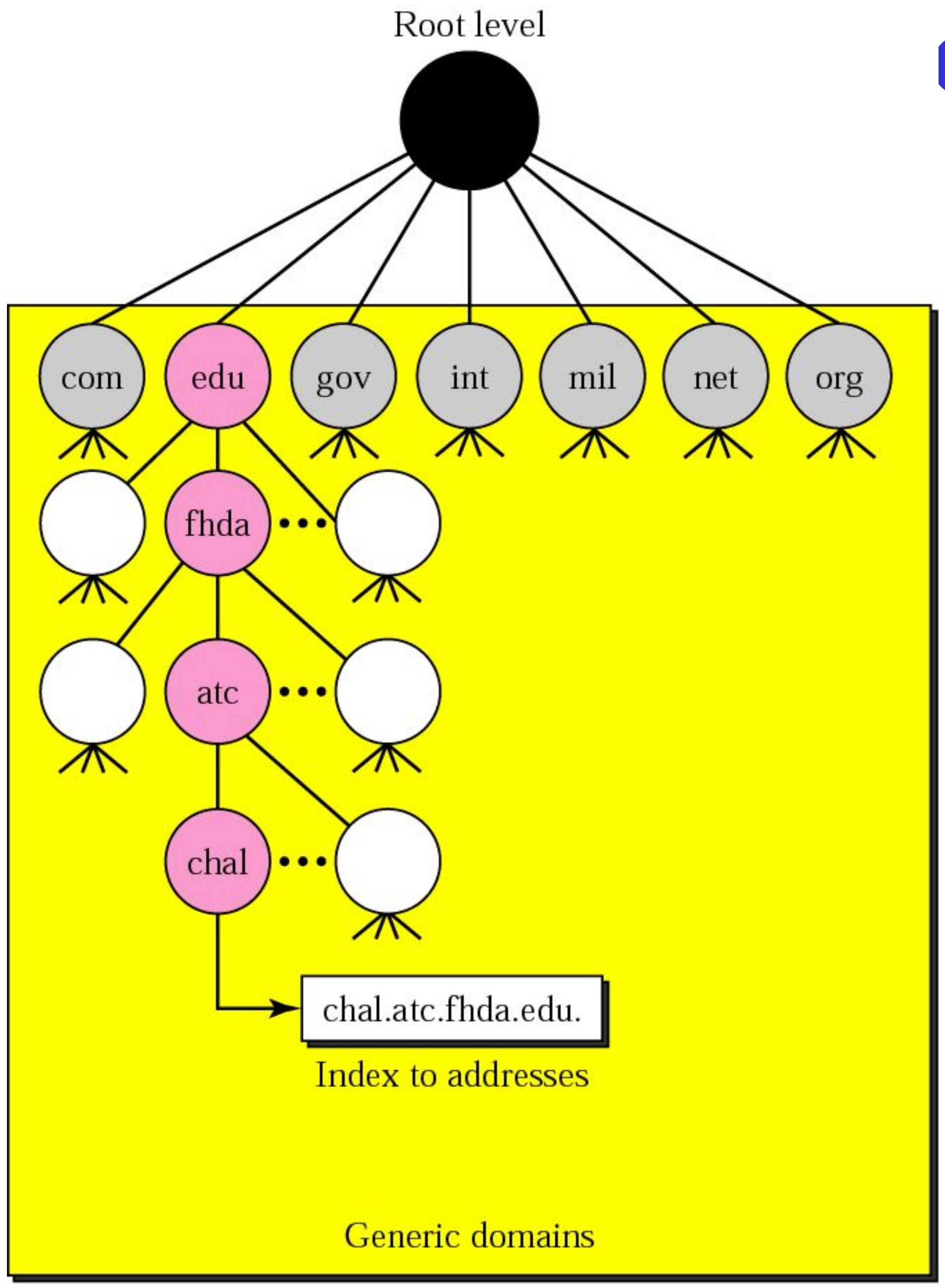


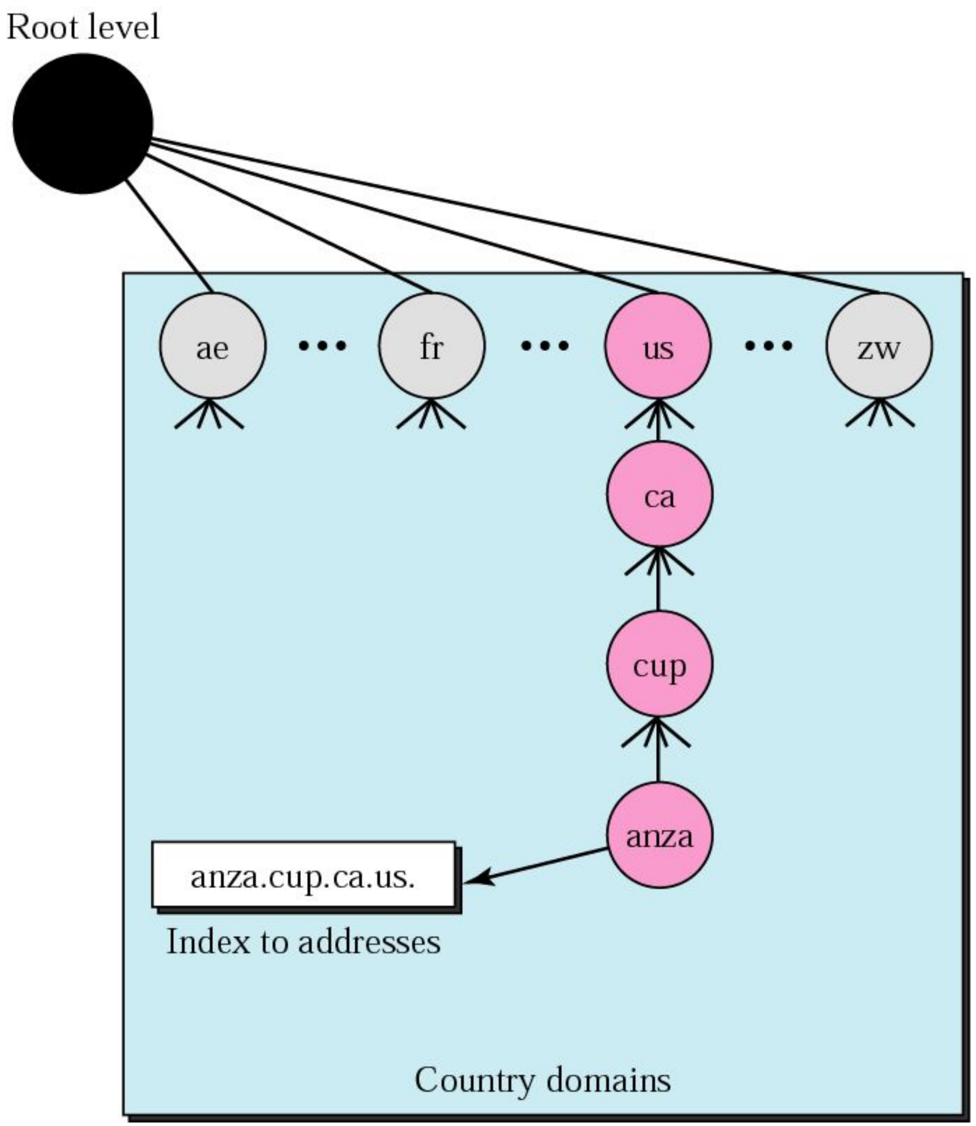
Figure 18-8

Generic domains



- The generic domains define registered hosts according to their generic behavior. Each node in the tree defines a domain, which is an index to the domain name space database

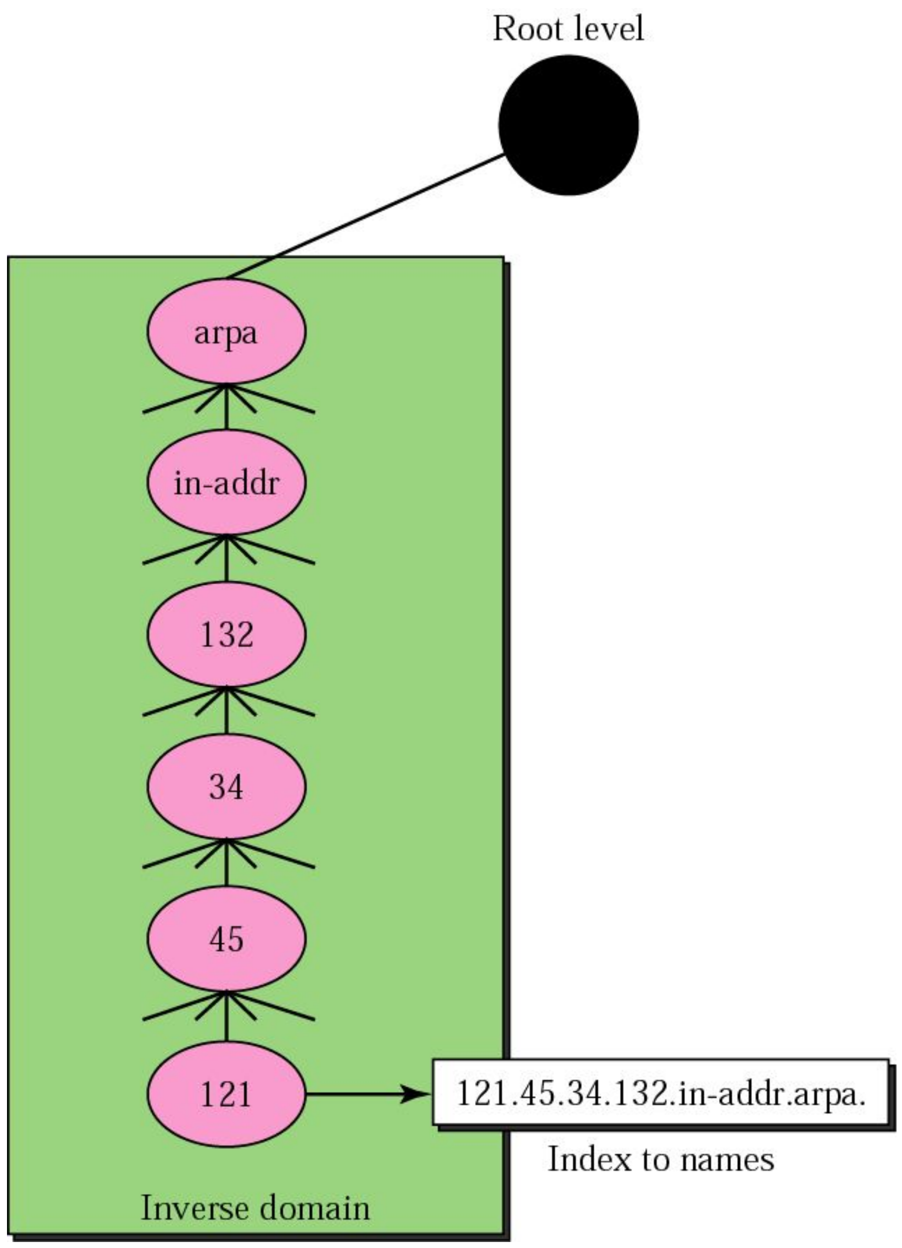
Country domains



- The address anza.cup.ca.us can be translated to De Anza College in Cupertino in California in the United States.

Figure 18-10

Inverse domain



- Inverse Domain
- The inverse domain is used to map an address to a name. This may happen, for example, when a server has received a request from a client to do a task. Although the server has a file that contains a list of authorized clients, only the IP address of the client (extracted from the received IP packet) is listed. The server asks its resolver to send a query to the DNS server to **map an address to a name** to determine if the client is on the authorized list. This type of query is called an inverse or pointer (PTR) query. To handle a pointer query, the inverse domain is added to the domain name space with the first-level node called arpa (for historical reasons). The second level is also one single node named in-addr (for inverse address). The rest of the domain defines IP addresses.

18.5

RESOLUTION

Figure 18-11

Recursive resolution

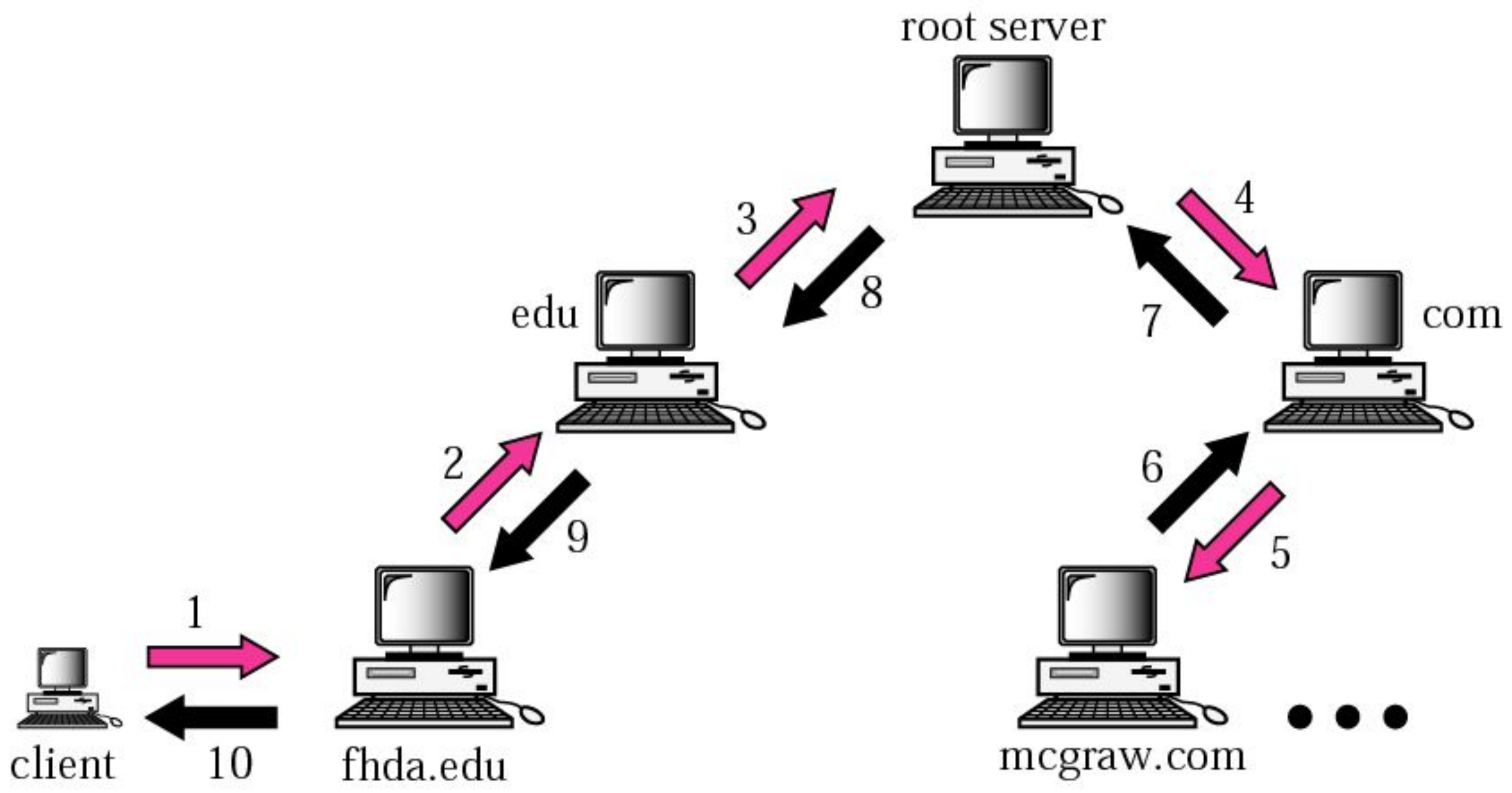
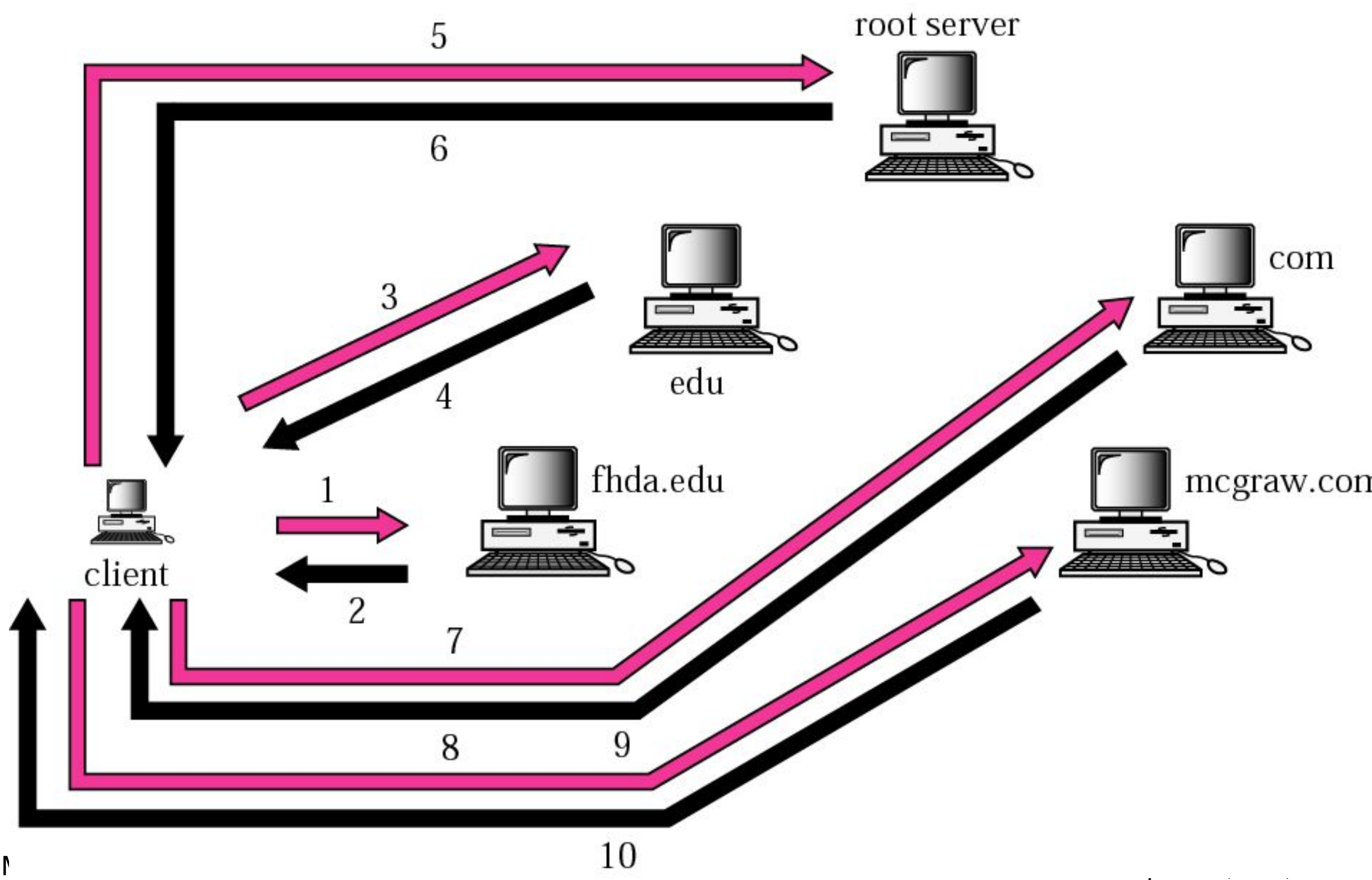


Figure 18-12

Iterative resolution



18.6

DNS MESSAGES

Figure 18-13

DNS messages

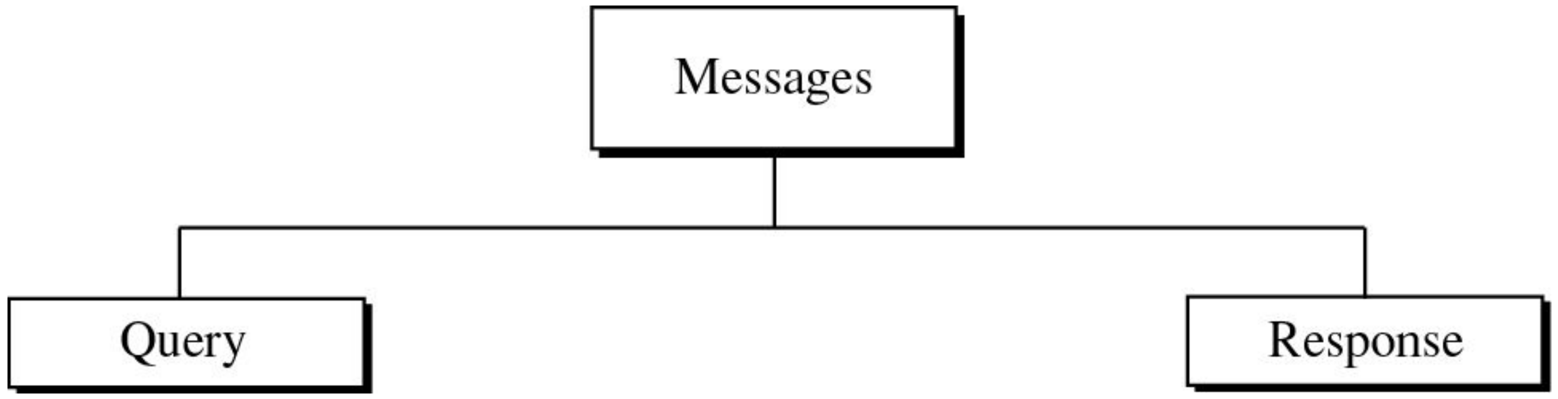
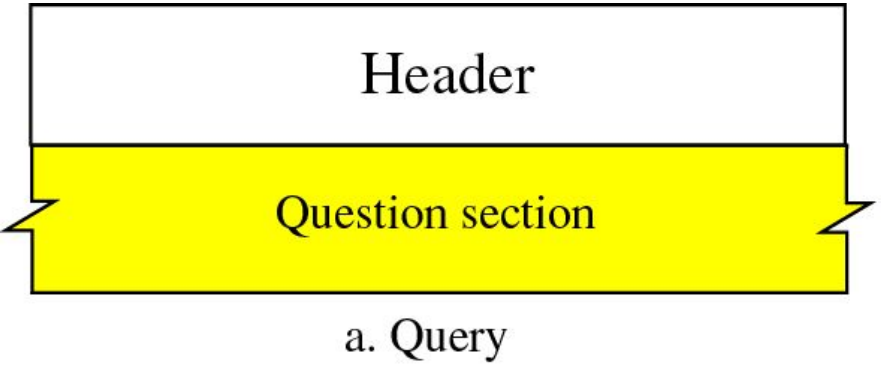
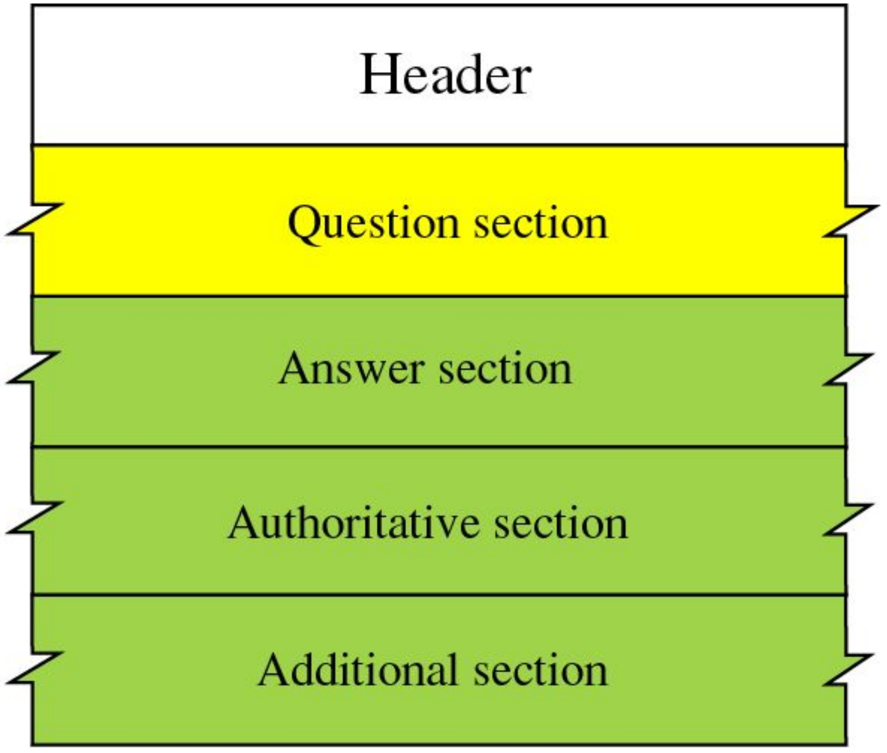


Figure 18-14

Query and response messages



a. Query



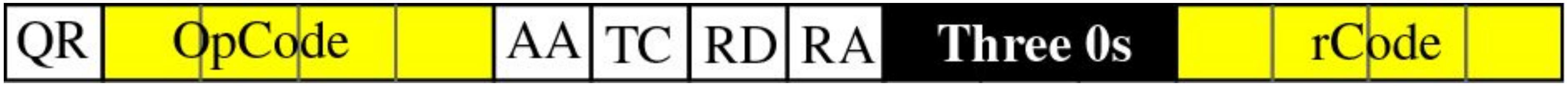
b. Response

Figure 18-15

Header format

Identification	Flags
Number of question records	Number of answer records (All 0s in query message)
Number of authoritative records (All 0s in query message)	Number of additional records (All 0s in query message)

Flags fields

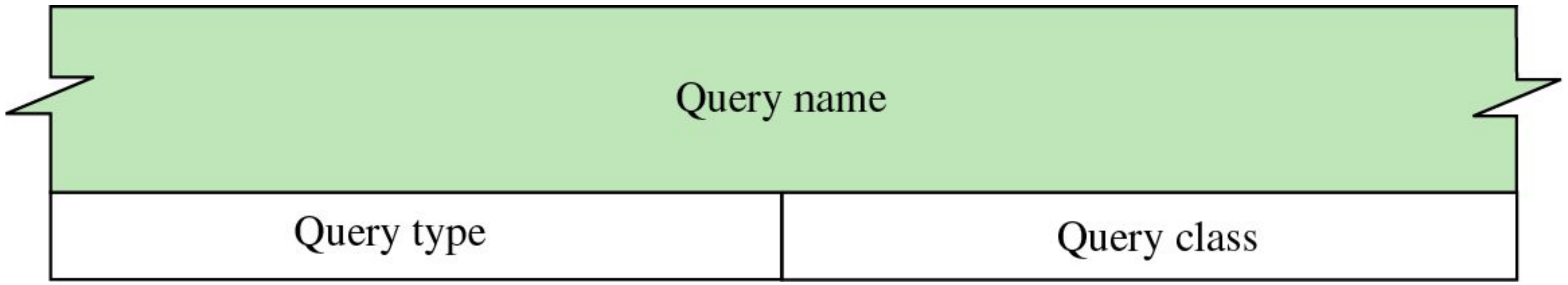


QR: Query/Response
OpCode: 0 standard, 1 inverse, 2 server status
AA: Authoritative
TC: Truncated
RD: Recursion Desired
RA: Recursion Available
rCode: Status of the error

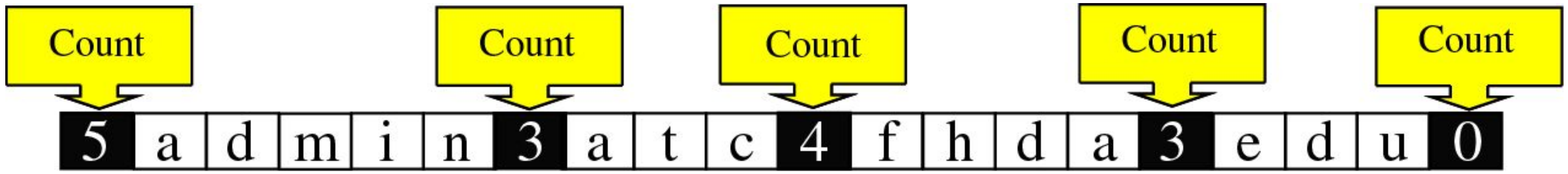
18.7

TYPES OF RECORDS: qsn record and response record

Question record format



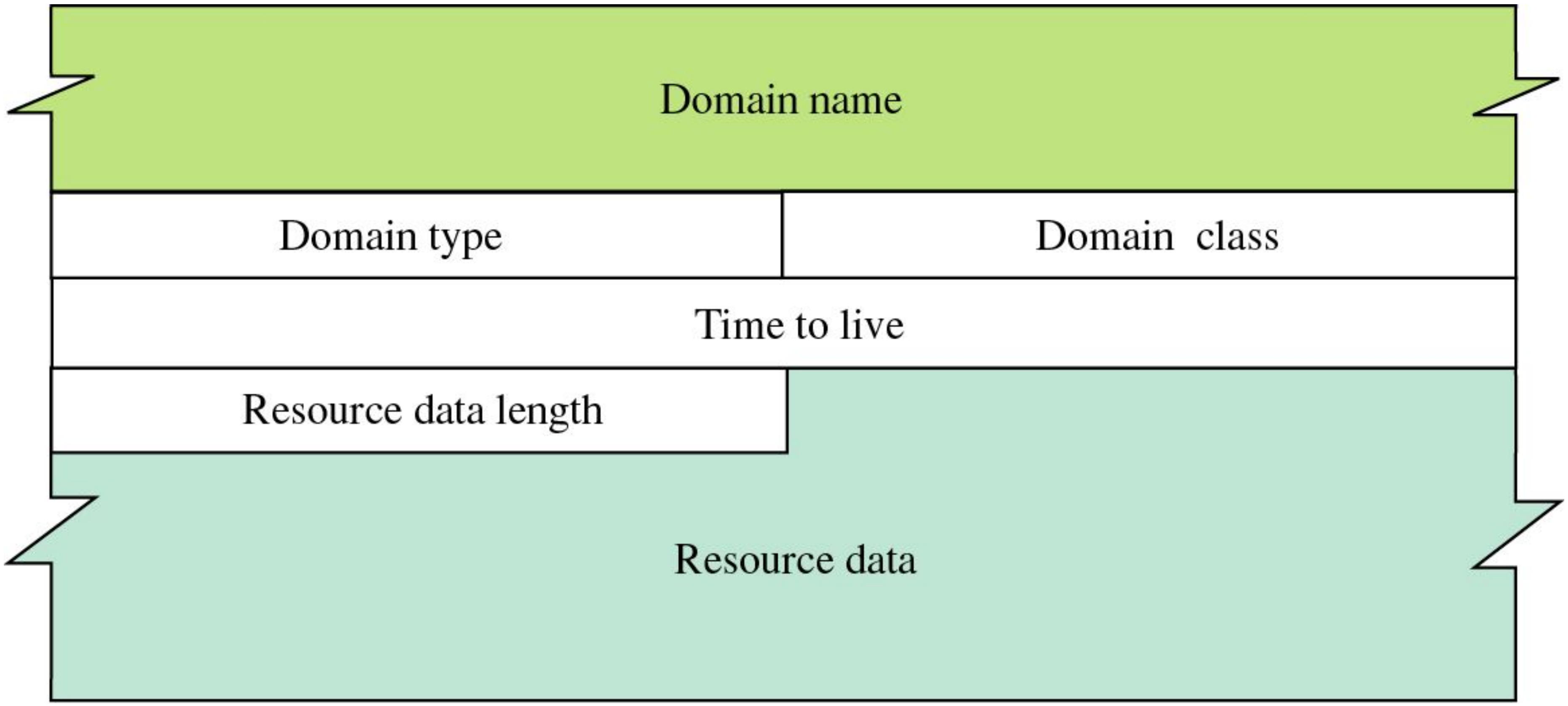
Query name format



admin.atc.fhda.edu.

Figure 18-19

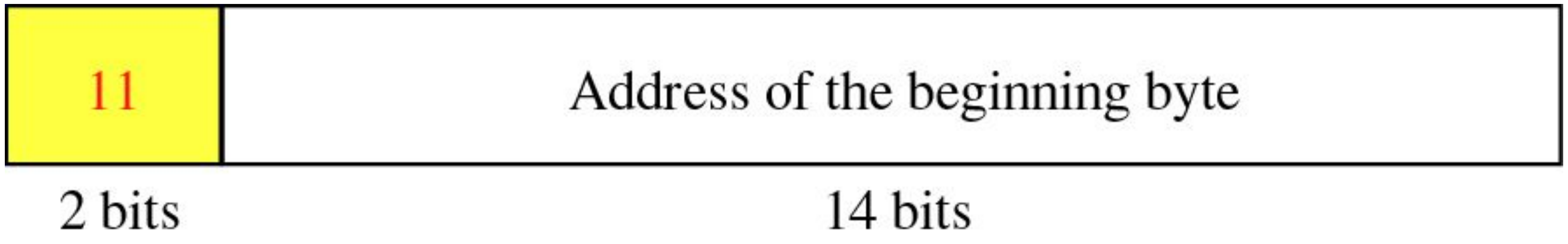
Resource record format



18.8

COMPRESSION

Format of an offset pointer



18.9

EXAMPLES

Example 1

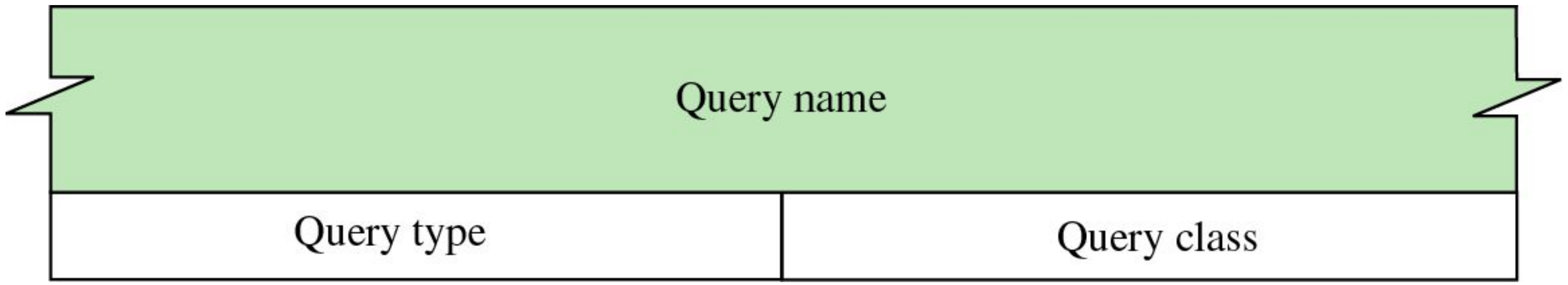
A resolver sends a query message to a local server to find the IP address for the host "*chal.fhda.edu.*". We discuss the query and response messages separately.

Figure 18-15

Header format

Identification	Flags
Number of question records	Number of answer records (All 0s in query message)
Number of authoritative records (All 0s in query message)	Number of additional records (All 0s in query message)

Question record format



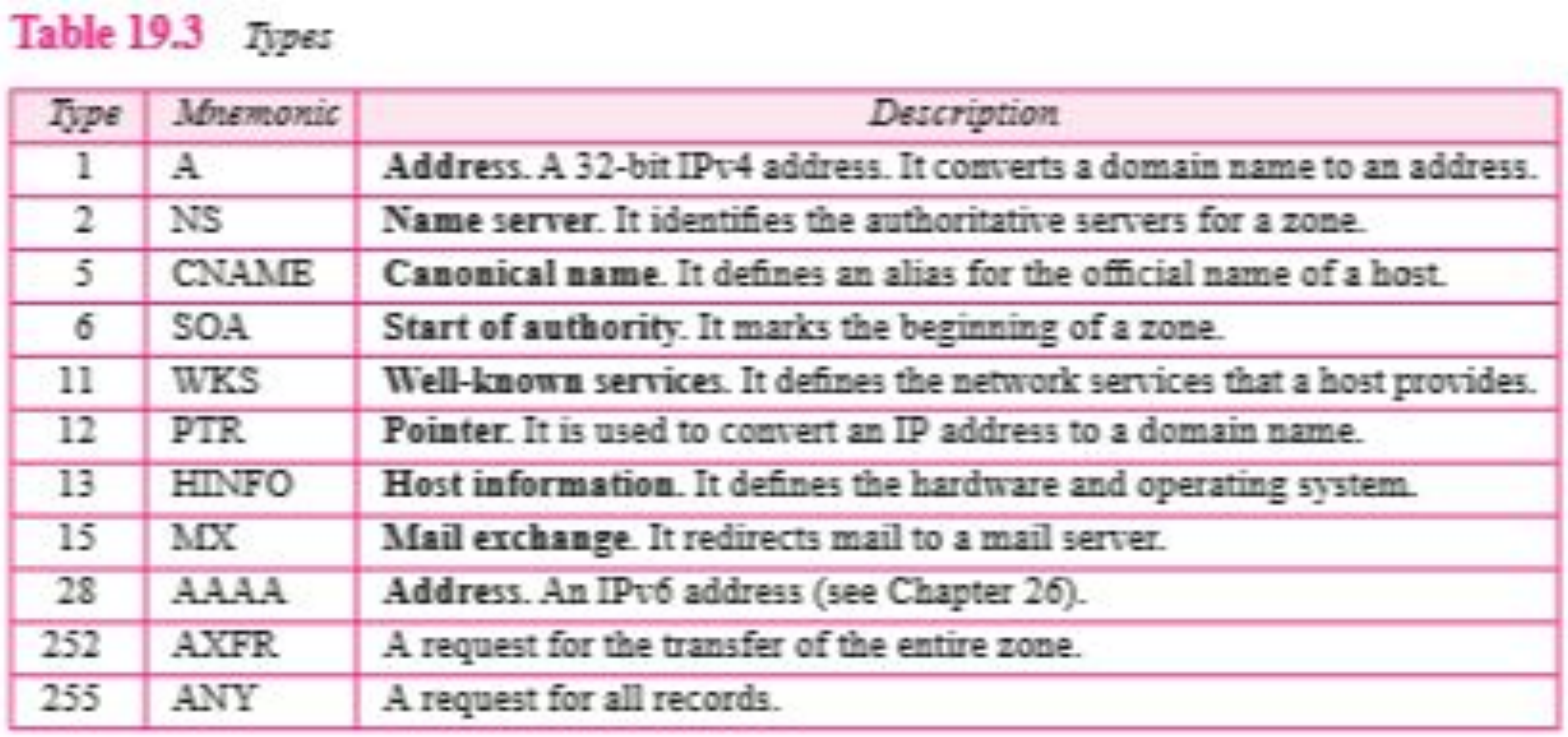


Table 19.3 *Types*

Type	*Mnemonic*	*Description*
1	A	**Address**. A 32-bit IPv4 address. It converts a domain name to an address.
2	NS	**Name server**. It identifies the authoritative servers for a zone.
5	CNAME	**Canonical name**. It defines an alias for the official name of a host.
6	SOA	**Start of authority**. It marks the beginning of a zone.
11	WKS	**Well-known services**. It defines the network services that a host provides.
12	PTR	**Pointer**. It is used to convert an IP address to a domain name.
13	HINFO	**Host information**. It defines the hardware and operating system.
15	MX	**Mail exchange**. It redirects mail to a mail server.
28	AAAA	**Address**. An IPv6 address (see Chapter 26).
252	AXFR	A request for the transfer of the entire zone.
255	ANY	A request for all records.

Table 19.4 *Classes*

Class	*Mnemonic*	*Description*
1	IN	Internet
2	CSNET	CSNET network (obsolete)
3	CS	The COAS network
4	HS	The Hesiod server developed by MIT

Figure 18-21

Example of a query message

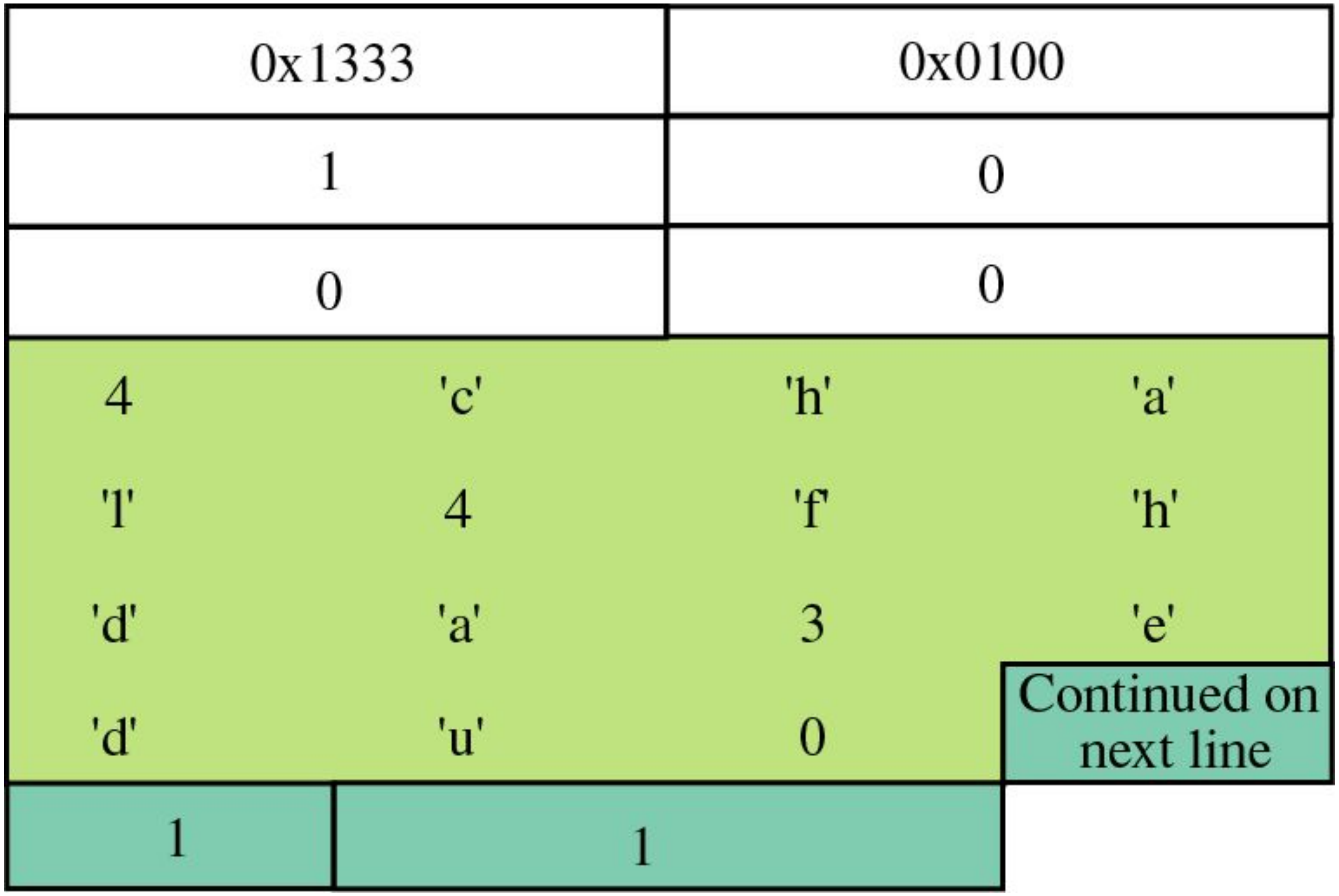
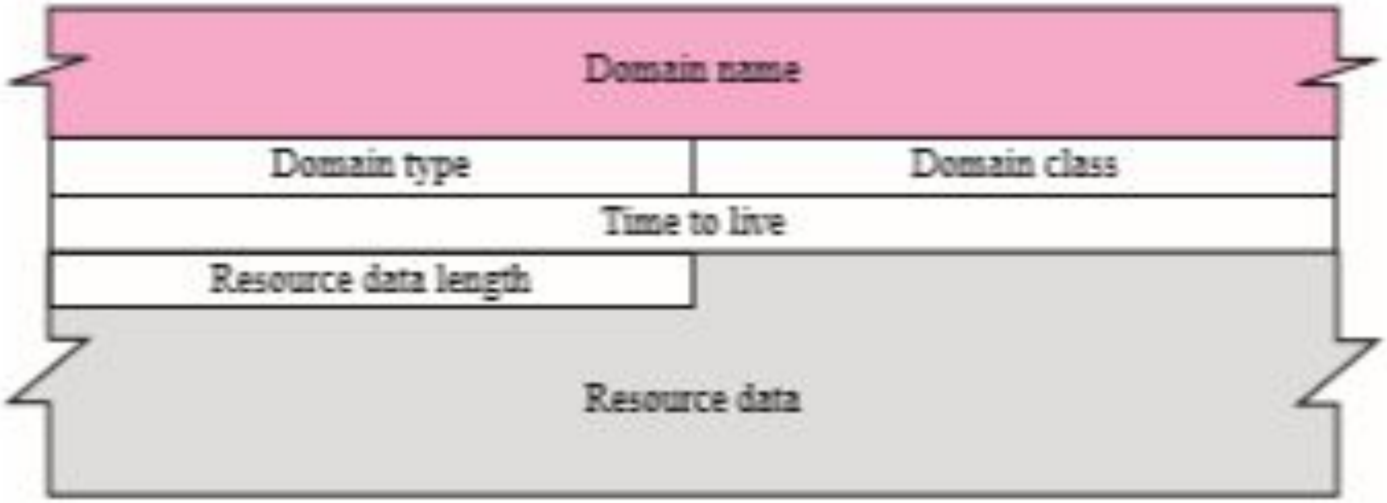


Figure 19.19 *Resource record format*



- Domain name. This is a variable-length field containing the domain name. It is a duplication of the domain name in the question record. Since DNS requires the use of compression everywhere a name is repeated, this field is a pointer offset to the corresponding domain name field in the question record.
- ❑ Domain type. This field is the same as the query type field in the question record except the last two types are not allowed.
- ❑ Domain class. This field is the same as the query class field in the question record
- ❑ Time-to-live. This is a 32-bit field that defines the number of seconds the answer is valid. The receiver can cache the answer for this period of time. A zero value means that the resource record is used only in a single transaction and is not cached. ❑ Resource data length. This is a 16-bit field defining the length of the resource data.

- **Resource data.** This is a variable-length field containing the answer to the query (in the answer section) or the domain name of the authoritative server (in the authoritative section) or additional information (in the additional information section). The format and contents of this field depend on the value of the type field. It can be one of the following:

  a. **A number.** This is written in octets. For example, an IPv4 address is a 4-octet integer and an IPv6 address is a 16-octet integer.

  b. **A domain name.** Domain names are expressed as a sequence of labels. Each label is preceded by a 1-byte length field that defines the number of characters in the label. Since every domain name ends with the null label, the last byte of every domain name is the length field with the value of 0. To distinguish between a length field and an offset pointer (as we will discuss later), the two high-order bits of a length field are always zero (00). This will not create a problem because the length of a label cannot be more than 63, which is a maximum of 6 bits (111111).

  c. **An offset pointer.** Domain names can be replaced with an offset pointer. An offset pointer is a 2-byte field with each of the 2 high-order bits set to 1 (11).

  d. **A character string.** A character string is represented by a 1-byte length field followed by the number of characters defined in the length field. The length field is not restricted like the domain name length field. The character string can be as long as 255 characters (including the length field).

Figure 19.19 *Resource record format*

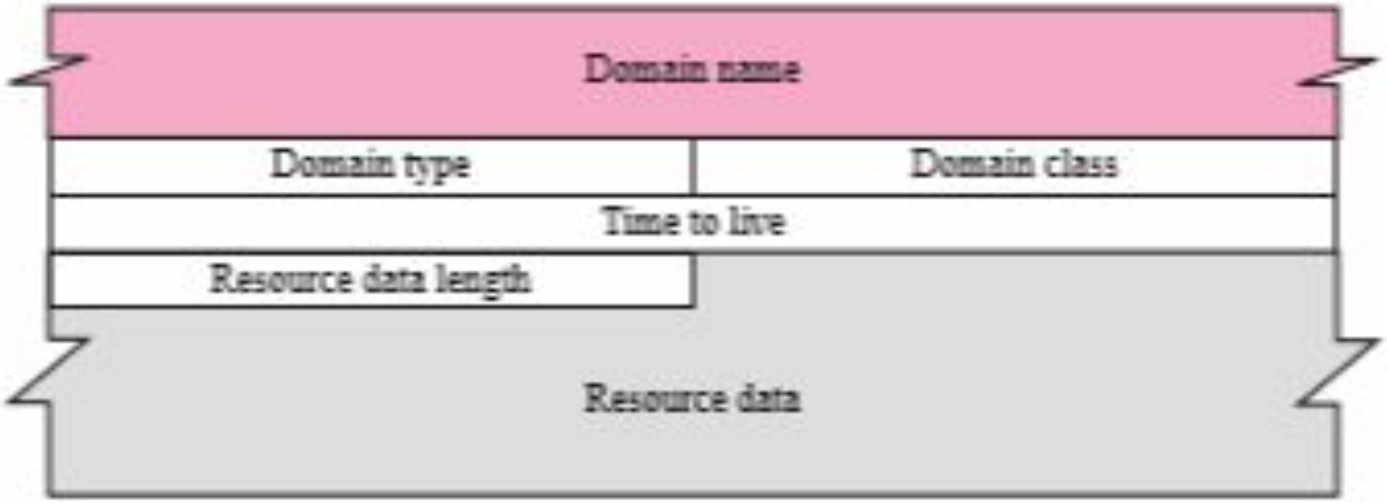
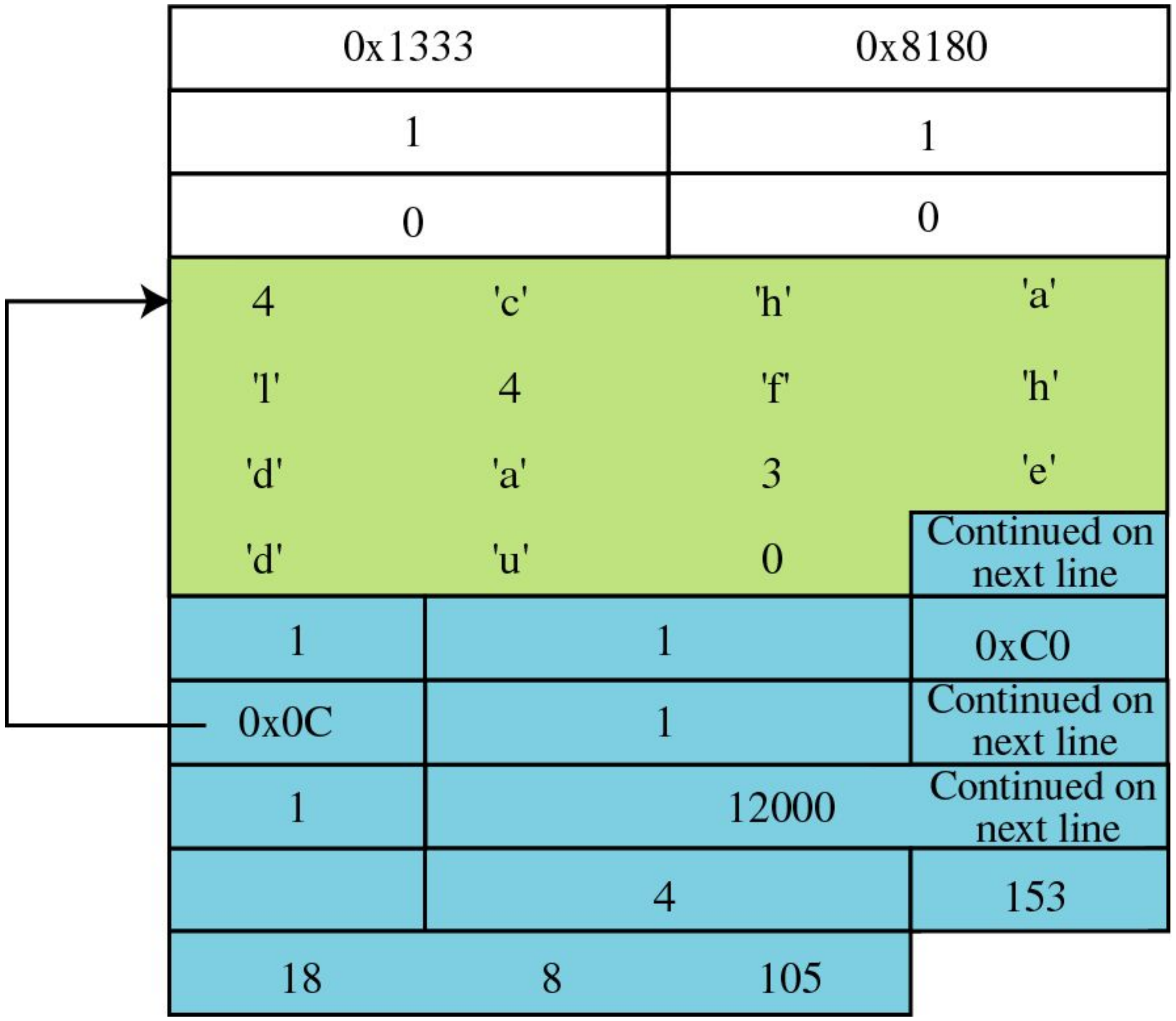


Figure 18-22

Example of a response message



Example 2

An FTP server has received a packet from an FTP client with IP address 153.2.7.9. The FTP server wants to verify that the FTP client is an authorized client.

Example of inverse query message

0x1200		0x0900	
1		0	
0		0	
1	'9'	1	'7'
1	'2'	3	'1'
'5'	'3'	7	'i'
'n'	'-'	'a'	'd'
'd'	'r'	4	'a'
'r'	'p'	'a'	0
12		1	

153.2.7.9.

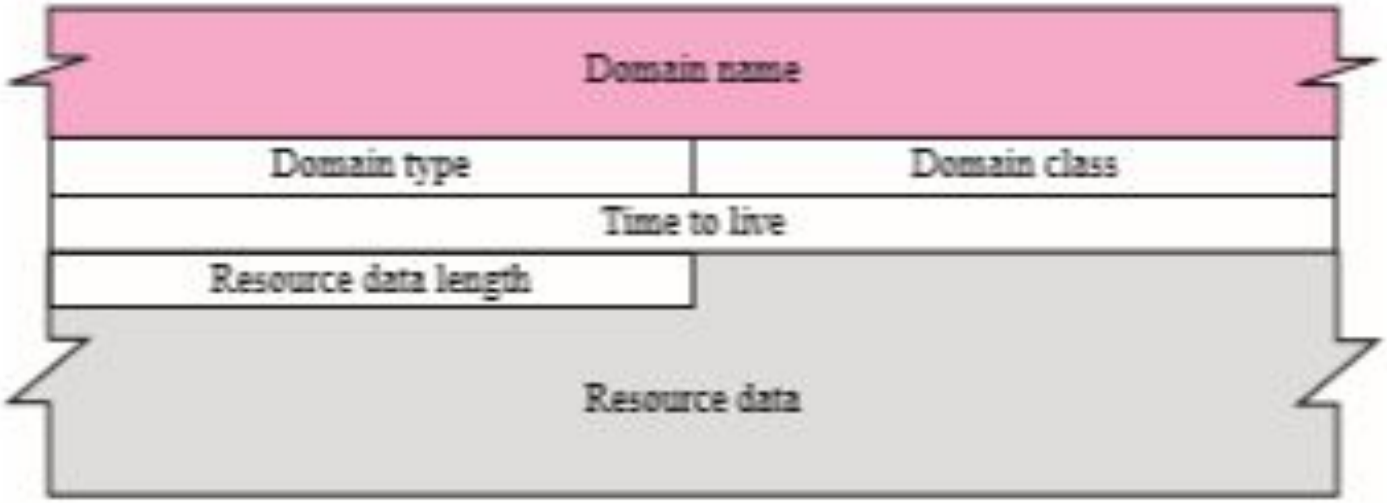
3 1 1 1

in-addr	7
arpa	4

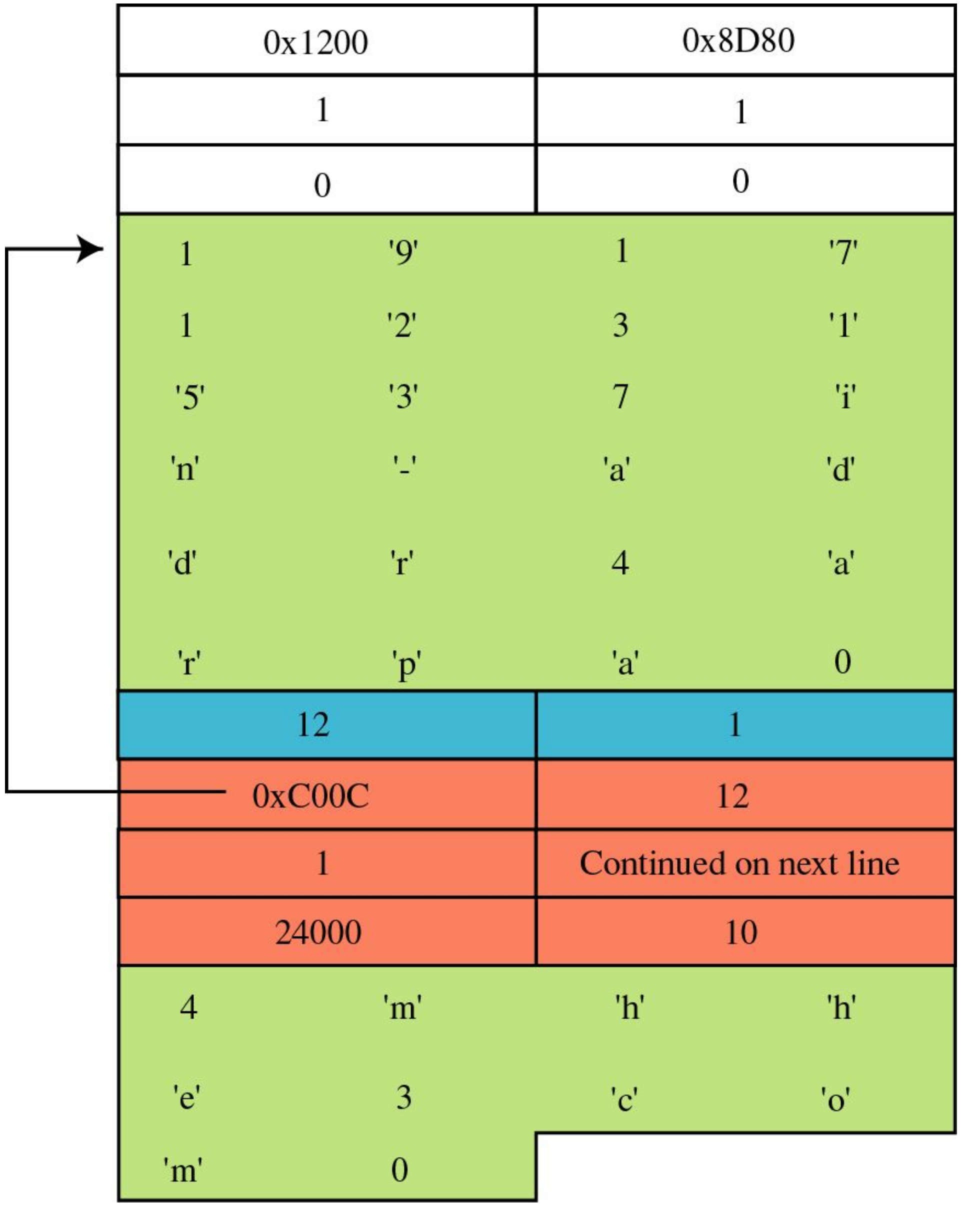
Table 19.3 *Types*

Type	*Mnemonic*	*Description*
1	A	**Address**. A 32-bit IPv4 address. It converts a domain name to an address.
2	NS	**Name server**. It identifies the authoritative servers for a zone.
5	CNAME	**Canonical name**. It defines an alias for the official name of a host.
6	SOA	**Start of authority**. It marks the beginning of a zone.
11	WKS	**Well-known services**. It defines the network services that a host provides.
12	PTR	**Pointer**. It is used to convert an IP address to a domain name.
13	HINFO	**Host information**. It defines the hardware and operating system.
15	MX	**Mail exchange**. It redirects mail to a mail server.
28	AAAA	**Address**. An IPv6 address (see Chapter 26).
252	AXFR	A request for the transfer of the entire zone.
255	ANY	A request for all records.

Figure 19.19 *Resource record format*



Example of inverse response message



18.10

DDNS

- When the DNS was designed, no one predicted that there would be so many address changes. In DNS, when there is a change, such as adding a new host, removing a host, or changing an IP address, the change must be made to the DNS master file.
- These types of changes involve a lot of manual updating. The size of today's Internet does not allow for this kind of manual operation. The DNS master file must be updated dynamically.
- The Dynamic Domain Name System (DDNS) therefore was devised to respond to this need. In DDNS, when a binding between a name and an address is determined, the information is sent, usually by DHCP to a primary DNS server. The primary server updates the zone. The secondary servers are notified either actively or passively. In active notification, the primary server sends a message to the secondary servers about the change in the zone, whereas in passive notification, the secondary servers periodically check for any changes. In either case, after being notified about the change, the secondary requests information about the entire zone (zone transfer). To provide security and prevent unauthorized changes in the DNS records, DDNS can use an authentication mechanism.

18.11

ENCAPSULATION

- DNS can use either UDP or TCP. In both cases the well-known port used by the server is port 53. UDP is used when the size of the response message is less than 512 bytes because most UDP packages have a 512-byte packet size limit. If the size of the response message is more than 512 bytes, a TCP connection is used. In that case, one of two scenarios can occur:
- ❑ If the resolver has prior knowledge that the size of the response message is more than 512 bytes, it uses the TCP connection. For example, if a secondary name server (acting as a client) needs a zone transfer from a primary server, it uses the TCP connection because the size of the information being transferred usually exceeds 512 bytes.
- ❑ If the resolver does not know the size of the response message, it can use the UDP port. However, if the size of the response message is more than 512 bytes, the server truncates the message and turns on the TC bit (See Figure 19.16). The resolver now opens a TCP connection and repeats the request to get a full response from the server.

Note

DNS can use the services of UDP or TCP using the well-known port 53.