Chapter 5

Practical assignment 5

5.1 Background

LDAP is a protocol that provides access to a distributed directory service. It is standardised by RFC 4511. RFC 4510 provides a road map of a number of standards that may be relevant for this assignment (and indicates where RFC 4511 fits into the bigger picture). The Wikipedia entry on LDAP provides a nice overview of the protocol and its use; it may assist you to navigate RFC 4511 (and other relevant standards).

At an early stage in your preparation for this assignment it is important to understand how an LDAP tree (a *Directory Information Tree* is structured. Sections 2.1 to 2.3 of RFC 4512 (LDAP Models) should provide you with all the essential details.

In order to gain some experience with the use of LDAP install `OpenLDAP` on your server computer. Although one should almost always use the most secure options when configuring LDAP, for this assignment it is acceptable to send passwords as cleartext and not to use SSL/TLS. Without encryption, sniffing traffic is possible and much can be learnt from such sniffing.

It is possible to access the LDAP tree using the commands (on the command line) that are installed along with `OpenLDAP`. However, rather install `phpLDAPadmin` (often abbreviated as PLA) as well. It provides one with a web-based interface through which one is able to add, delete and modify LDAP entries.

Note that PLA executes on the server and you will not learn much about the operation of LDAP by sniffing traffic between your browser and PLA. Many LDAP clients exist that use the LDAP protocol. You are encouraged to find such a client and sniff traffic between it and your server.

Note that the protocols required to complete this assignment use different approaches to specify the representation of messages. You will encounter (at least)

ASN.1 and Augmented Backus-Naur Form (ABNF). Use this as a learning opportunity.

Also note that, where previous assignments used protocols where requests and responses were sent as plain text, using LDAP will often require you to send messages encoded in binary.

5.2 Your assignment

While LDAP is intended to serve as a directory of people, it may also be used to store other information. In this assignment we will use LDAP in a rather atypical manner. A typical directory may use countries as the first nodes in the directory tree. Organisations within those countries may be placed at the next level. Departments (or 'organisation units') within each company may be at the next level. Finally, details of employees working in such a department (such as their names, work addresses and telephone numbers) are stored as the leaves. It is , however, not necessary to implement the tree using all the levels. A company that wants to create a directory may, for example, deem the company to be at the root, and put departments (and with departmental information) one link lower in the tree.

Similar to the directory described above, the DNS forms a tree, starting at the root nodes.

Now suppose, just because we are able to do it, we want to store DNS information about the organisations within the `za.` TLD in LDAP. (This is an educational exercise — do not think too hard about the logic of what we are doing.)

Hence, the root will be the TLD. Our 'organisational units' in the tree will be the second-level domains, such as `ac`, `co` and `gov`. The organisations about which we want to store details will be one level lower — where details if individuals would fit more naturally.

The information we want to record about our (individual) organisations is the information stored on the third layer of the DNS tree about them. To use `up.ac.za` as an example, the following information is relevant:

```
C:\>nslookup

> set type=any
> up.ac.za

Non-authoritative answer:
up.ac.za
    primary name server = ns2.up.ac.za
    responsible mail addr = dnsadmin.up.ac.za
    serial  = 3991808205
```

```
    refresh = 10800 (3 hours)
    retry   = 3600 (1 hour)
    expire  = 1728000 (20 days)
    default TTL = 900 (15 mins)
up.ac.za    internet address = 137.215.10.220
up.ac.za    MX preference = 1, mail exchanger = aspmx.l.google.com
up.ac.za    MX preference = 10, mail exchanger = alt3.aspmx.l.google.com
up.ac.za    MX preference = 5, mail exchanger = alt2.aspmx.l.google.com
up.ac.za    MX preference = 10, mail exchanger = alt4.aspmx.l.google.com
up.ac.za    nameserver = ns2.up.ac.za
up.ac.za    nameserver = ns3.up.ac.za
up.ac.za    nameserver = ns1.up.ac.za
>
```

(Not all the information returned by this query is included above.)

The information provided clearly derives from SOA, A, MX and NS resource records in the authoritative zone file. For the assignment you only have to include information that pertain to A, NS and MX records. One would expect a single IPv4 address (if any). Record only the MX record with the highest priority (lowest number) — if MX records are present. Use any one NS record — if NS records are present.

Create the Directory Information Tree on your server and populate it with three or four second-level domains. Pick any that you like. Pick any of these second-level domains and add information about three to four organisations at the appropriate places in the tree. Use the pre-existing tools discussed above to populate this (tiny) part of the tree.

Now write a client that will ask you for an organisation name, query the LDAP database, and display the resource records (if the organisation exists in your tree).

If `up.ac.za` is in your tree, are you able to query it using just `up`? Are you able to query it using `up.ac.za`? The underlying question is the following: If Jane Doe works in the design department of ACME, will we (where LDAP is used for its intended purpose) be able to search for Jane Doe, Jane Doe at ACME, the employees at ACME, or any other such complete or incomplete information.

Note that your client should establish a connection with port 389 on your server, formulate and write all requests to the socket, and read and interpret all responses from the server. As always you are not allowed to use library functions or any other mechanisms that handles communication between the client and server on a more abstract level. To be more specific, your client should construct the query packet and write it to the appropriate socket using byte or string level operations. Similarly, your client should read a response packet from the appropriate port, unpack it using string or byte array operations, and process the unpacked message as appropriate.

5.3 Assessment

A working program will be awarded 8 out of 10. To earn a higher mark your program has to do more than just the basics - in particular should it demonstrate that you have some deeper knowledge of the protocols as they are defined in the various LDAP-related RFCs.