Networking Services

Hands-On Unix System Administration DeCal

Lab 7 — 15 October 2012

Due 22 October at 6:10 PM

1 DNS queries

(Hint: The commands `host` and `dig` will be useful for the following questions)

1. Briefly lookup/define the following terms. These are common DNS record types.
   (a) A record
   (b) CNAME record
   (c) NS record
   (d) MX record
2. Use `host` to look up the MX record for berkeley.edu. Show the syntax you used.
3. Briefly describe the role of a name server.
4. What are `berkeley.edu's` NS records?

2 Playing with LDAP

2.1 The CalNet LDAP Directory Service

Recall that LDAP (the Lightweight Directory Access Protocol) is the protocol used to access information stored in an information directory (aka an LDAP directory). If that sounded confusing, you could think of an LDAP information directoy as a kind of database, that can store useful information such as (say, in the context of a running a company) customer contact information, email addresses, employee information, etc. It is important to remember that it is *not* a relational database (like MySQL or PostgreSQL).

The CalNet LDAP Directory contains information about students, faculty, and staff, as well as other campus affiliates. In the following exercises, you'll look up your own LDAP entry stored in the CalNet Directory. (Or...feel free to be a stalker...) Before we begin, let's review some important concepts.

The LDAP entry data structure:

LDAP directories store their data hierarchically (similar to how the Unix file directory is structured). Data is represented as a data structure called an *entry*. An entry has a set of *attributes* that hold the data for that entry. If you're familiar with databases, then this is analgous to fields in a database record.

The structure of an LDAP directory tree:

Like I said before, LDAP directories are structured hierarchically. The top level of the LDAP directory tree is referred to as the "*base DN*". A base DN could take various forms; it could be derived from DNS domain name, for example, in the case of the Calnet directory, the base DN would be "`dc=berkeley,dc=edu`" (dc just stands for "domain component").

Under the directory's base (analogous to the root directory in a filesystem), a directory can have numerous "containers" that logically partition the data. Most LDAP directories set these containers as *OU entries*, which stands for "Organizational Unit". You can see how CalNet has organized its data by looking at this page:

`https://wikihub.berkeley.edu/display/calnet/How+is+LDAP+organized`

Data in the CalNet directory is basically split up into seven Organizational Units under the top level Directory root "dc=berkeley, dc=edu", like Students/Staff/Affiliates, Alumni, Admitted students, Guests, etc etc, you get the idea. For the majority of you (if not all of you), you'd be interested in the People OU(`ou=people,dc=berkeley,dc=edu`), since is the branch of the CalNet Directory that contains all the ldap entries representing students, staff, and campus affiliates. You can see an example of what a 'People' Entry looks like here:

`https://wikihub.berkeley.edu/display/calnet/People+Entry+Example`

We can search the CalNet Directory with a command-line utility called `ldapsearch`. `ldapsearch` basically opens a connection to an LDAP server and performs a search using parameters you specify. (See `man ldapsearch`)

Here's an example of how one could use `ldapsearch` to search the CalNet LDAP Directory:

```
ldapsearch -x -h ldap.berkeley.edu -p 389 -b 'ou=people,dc=berkeley,dc=edu' "(uid=XXXXXX)"
```

You'll notice that you did not have to input any credentials to access the directory. LDAP generally offers the possibility to logon to a directory without any user credentials. This is because we'll be working with attributes marked 'Public' can be accessed using an 'anonymous bind'. No special account needs to be set up.

All of the above options (excluding the search filter at the end of the command) are necessary to perform a simple, anonymous bind to the LDAP server.

-h hostname

-p port number (standard is 389)

-x tells ldapsearch to perform a simple_authentication (yes, you need this even for anonymous bind)

-b baseDN

The search filter is specified at the end, in parentheses. In this case, I'm looking for ldap entries with the specified attribute (uid) and value (XXXXXX, of course this would be your own uid). You could specify other attributes as well, like last name (the attribute `sn`, which stands for surname), but `uid` is pretty darn specific, whereas in the case of `sn` you might have multiple people with the same last name! Feel free to experiment!

1. Now try running `ldapsearch` (on your OCF account) to look for your ldap entry in the CalNet directory. What search filter(s) did you use?

2. What other attributes are listed in the ldap entry? Just name a few.

3. What organizational unit does the entry belong in? (specified by the `ou` attribute). I pretty much gave this away earlier.

4. You'll see numerous "`objectClass`" attributes. Any idea what these mean? (hint, LDAP is object-oriented.)

3 SSH

1. Describe how you would generate a public/private SSH key pair and install the public key onto a remote server.

2. Briefly explain the role of `ssh-agent` and `ssh-add`. How do they enable noninteractive logins?