Directory Services

Timothy Myers
Tennessee Tech
344 East 12th Street Apt. 4
Cookeville, TN 38501
931-206-2255
temyers240@gmail.com

ABSTRACT

This paper is an attempt at a brief overview of Directory Services in general and will, in that, include some discussion of the X.500 protocol and its lightweight derivative LDAP. At the top level, a directory service like Active Directory is a system that was designed to make certain information available to people all over the world. At the basic level this could be just information about people, in a manner similar to the phone book, but based upon the structure of the directory, this could also include information on groups of people, groups of computers and even entire organizations. The X.500 series of protocols and papers were created to assist in the creation of one of these directories, and the work done on this area through the X.500 Directory Access Protocol (DAP) was very complete, but it was also quite costly in terms of the resources to implement and run the protocol. In order to cut down on the cost of directory access, the Lightweight Directory Access Protocol (LDAP) was created, a slimmed down version of DAP implementing the most important and most widely used features found in the original.

Keywords

DAP, LDAP, Directory Services

1. INTRODUCTION

An argument can easily be made that a large number of the software innovations today have been created out of laziness. Directory services, for me, fall squarely into this category. Before the conception of directory services there were valid ways of storing and retrieving information about people and organizations that was stored on an individual computer, however, these utilities, like the unix program finger, required the person doing the searching to know quite a bit of information at the start of the process[2]. As an example, to finger a person on unix, the searcher would need to know not only the name of at least one machine where that person had an account, but also that person's username on that particular machine[2]. Also, for those people who had multiple accounts on multiple machines, if they wanted their information to be discoverable they were required to update their information manually across all accounts on all machines, potentially a time-consuming and annoying process. To help locate and store information not only on users, but also about organizations, groups of users, computers, and even other devices on the network, a series of papers occasionally collectively referred to as X.500 were created[3], these papers spoke of a concept called directory services as well as something called Directory Access Protocol (DAP) which was a way of speaking with these directories. Unfortunately DAP, while being very well designed and very complete, was just too much to handle for many of the commonly used computer systems at the time. Because of this another of protocol was created called the Lightweight DAP (LDAP) this protocol did essentially the same thing as DAP, but it cut out the (mostly) unused features, thus making LDAP a much leaner and more applicable protocol.

2. X.500

The X.500 series of papers is where directory services and also DAP were originally introduced. Again, staying similar to much in the field of computer science, these papers from the mid 1980's still describe the basic structure of modern directory service systems today. As far as information storage and retrieval are concerned the information is stored into logical units called entries, each of which has one or more attributes, each attribute of which has an “attribute syntax[2]” which describes exactly what type of data can be stored in each attribute. For example, a user entry could have a name attribute which could be any string, a id number attribute which could only be a number, a work number attribute which could only be a phone number(in one of a few formats) and others as well. With the X.500 model, there is also the concept of object classes[2]. These require each entry to have an attribute called objectClass which specifies which object class that entry belongs to. These object class entries can then inherit from each other in a very C++/Java sort of way. Children object classes inherit all attributes both mandatory and optional from their ancestor object classes. These entries can then be added to a tree structure that is found in the global namespace of the directory. These entries are placed in the tree in a way that makes logical sense, or at least they should be, with people entries being children of groups entries or with computer entries being associated with a particular domain[2].

Also included in the X.500 series is an unfortunately large number of three letter acronym's (TLA's) such as the DIT, DIB, DSA, DUA, DAP, DSP, DOP, DMD, DMO, and DISP[1]. DAP is the Directory Access Protocol that the end user uses to query information, add entries, remove entries, and perform other activities upon the information stored in the directory. The standards body that came up with this model did a very good job of building into the system the ability to deal with arbitrarily complex queries[4] and scalability, as well as things possibly unseen at the time, unfortunately, due to the built in complexity and robustness, the implementation of the protocol was a bit hard on the hardware readily available to the majority of users at the time of implementation. This fact, as well as the fact that all of the protocols (DAP, DSP, and DOP) in the X.500 model used the OSI transport stack kept the DAP and other protocols and other model accoutrements from becoming the standard in the long term for directory services. Currently this role is being held by DAP's lighter cousin LDAP, which we will speak about in a bit.

3. Active Directory

Active Directory (AD) is the name of Microsoft's version of directory services. This implementation was added with Windows Server 2000 and was included as an aid for large businesses. The system used before Active Directory was fine for small to medium sized businesses, but lacked scalability[1]. Microsoft wanted Active directory to be similar enough to their old NT system that they could be compatible and work with each other. To this goal Active Directory has the idea of domains within the directory, which are logical units that each have their own users, computers and other entities. There can also be trust among these domains, which allows for users of the trusted domain to be able to use the resources of the trusting domain, as long as those resources requested are within the permission level of that particular user. Each of these domains have their separate namespace, security policies, and information. One of the reasons that this is done is the sheer size of many of the organizations that would be using Active Directory, if everything were placed under one unit, the database holding the information would get large enough as to make searching for anything rather expensive as far as the time needed to find the information[1].

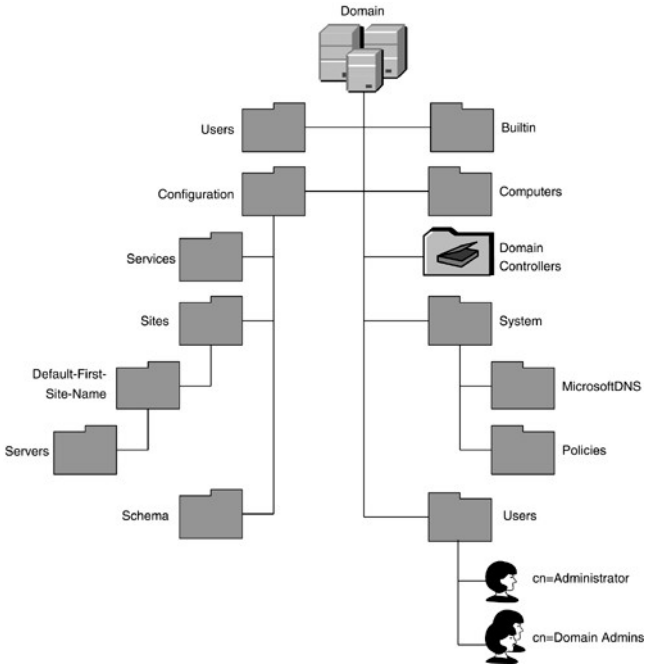

Illustration 1: Typical Object Tree in Active Directory[1]

Active Directory also treats the objects a bit differently as well. In Active Directory, there are really only three different object designators. The most common designator is the Common Name (CN), this type includes all of the objects depicted in Illustration 1 except for the Domain and Domain Controllers portions. CN is just a general way of giving objects a name within the directory. The domain section of Illustration 1 uses the designator Domain Component (DC) which is used for the name of the domain and is typically what would need to be searched for using DNS, an example being dc=CSC3550-3,dc= LOCAL. The final part is the Domain Controllers object, this object has a designation of Organizational Unit (OU) and is classified as such because the domain controllers need separate group policies, which are not available to the other designators[1]. It is the combination of the names of these entities wrapped in the designators as well as their position within the directories database tree that gives a Distinguished Name (DN). This DM is a unique way of identifying an entry within the active directory. As an example, the DN for the author's user object is CN=Timothy E. Myers,CN= Users,DC=CSC3550-3,DC=LOCAL.

4. LDAP

The Lightweight Directory Access Protocol or LDAP seems to currently be the most common way of accessing directory services systems today, even Microsoft makes large use of LDAP in their Active Directory[3]. LDAP was created at the University of Michigan in an attempt to take the best of the system explained in X.500 and make it into something that was actually worthwhile to implement on the client computers available at the time. In order to do this, several of the features of DAP were removed, such as some redundant features and items that were very rarely used, LDAP only used a subset of the encoding that DAP was designed to use, instead preferring to send data as simple text bytes, wrapped in a binary message for speed, as opposed to a purely binary data transmission[3]. But possibly the largest factor in it's eventual widespread use was the fact that LDAP ran on the TCP/IP transport stack. The system modeled in the X.500, because of it's origin in the ISO standards committee was designed to work on the OSI transport stack, a system that was not, and still is not, widespread because of its complexity. In the end TCP/IP won out as the transport layers of choice, and with that LDAP won out as well because of it's use of the widespread speedy stack of TCP/IP.

LDAP originally started as simply a translator protocol and service. There was a LDAP daemon (ldapd) which processed LDAP requests by changing them to DAP requests and sending them to a DAP server, and then would receive the answer as a DAP response, translate it back to LDAP and then return the LDAP response to the client. This was much better in terms of the requirements on the clients, but the servers were still complex and hard to implement efficiently[3]. Eventually, it was noticed that the vast majority of activity to the X.500 based directories were being performed though LDAP. At this point there was created a standalone LDAP daemon (sldapd) which was a server implementation of LDAP that needed no interaction with a DAP server, it was able to fully respond to queries by it self. This brought the past client accomplishment to the server, reducing the complexity and bringing the efficiency up[3].

However, LDAP is not perfect. Today as the need for security on the internet rises, LDAP is running into some rough patches. Certificates are used in our online environment to determine if the site that we are viewing is really who it says it is, and certificates, just like most other types of data, can be stored within a directory. The issue is that not all certificates contain the same information, and LDAP is not designed with very flexible information storage in mind[4]. Due to the subset of the encoding mentioned earlier, LDAP would need a new set of syntax rules for each type of certificate, which is not feasible in the long run[4].

5. CONCLUSION

While I believe that Directory Services are here partially because human beings tend to be lazy, another large reason they exist is because they are very useful and convenient. With a working implementation of Directory Services we are able to discover information about someone or something down the hall or around the world. The best part is it seems just like the information is

sitting on your own computer, instead of being on a server in another state possibly. The dream of the creators of X.500 was not just Directories but *the* Directory, with the ability to discover the world, and whether that happens or not, one thing is certain. Directory services are here to stay.

6. REFERENCES

[1] Boswell, William. Understanding Active Directory Services 2003 Retrieved November 14, 2010 from informIT http://www.informit.com/articles/article.aspx?p=101405

[2] C. Wieder, J. Reynolds, S. Heker "Techinical Overview of Directory Services Using x.500 Protocol" RFC 1309, March 1992. [Online]. Available: http://www.faqs.org/rfcs/rfc1309.html

[3] Howes, Timothy; Smith, Mark; Good, Gordon. *Understanding and Deploying LDAP Directory Services*. Addison-Wesley Professional, 2003

[4] Sang Seok Lim, Jong Hyuk Choi, and Kurt D. Zeilenga. 2004. Secure and flexible certificate access in WS-security through LDAP component matching. In Proceedings of the 2004 workshop on Secure web service (SWS '04). ACM, New York, NY, USA, 87-96. DOI=10.1145/1111348.1111358 http://doi.acm.org/10.1145/1111348.1111358