Global Information Assurance Certification Paper

GIAC Advanced Incident Handling and Hacker Exploits Practical Assignment

By
Beth Hemmelgarn

Vulnerability

Name: Vulnerabilities of LDAP and X.500
Variants: observation, sniffing
Operating System: All OS's are impacted
Protocols/Services: DAP and LDAP
Brief Description: This paper discusses potential vulnerabilities with the Lightweight Directory Access Protocol (LDAP) and the Directory Access Protocols (DAP). Also discussed are applications that use the directory for passwords and configuration information and how this information passed over the network is vulnerable.

Protocol Description

The Directory Access Protocol (DAP) and the Lightweight Directory Access Protocol (LDAP) allow applications, users and other directories to communicate. LDAP is often used by clients such as Outlook and Netscape to store user information for e-mail systems. The use of the LDAP and DAP protocols are by no means restricted to the use of e-mail. It is being used for more and more applications for user authentication as well as application configuration. This paper will illustrate how the directory is utilized and it's potential for exploitation.

Description of variants

The directory contains information about users such as those using e-mail as well as application information. Applications previously used proprietary mechanisms for this information. That made sharing this information difficult and deferred crackers from even caring about these data repositories.

How the exploit works

There are several ways to gain information from an LDAP or DAP based directory. The first is to sniff the information off of the network. The second is to gain access to the server that contains the directory itself. A third mechanism is to simply write a program that issues queries to the directory. If you sniff the information off the network, it will be necessary to decode the LDAP or DAP protocol. Once this is done passwords contained in the sniffed data can be hacked using simple decryption methods. If the actual files are acquired, the information is usually in clear text and can be used as is. If there is encrypted information it can be decrypted using simple decryption methods.

Unix Password file as a user information repository

The Unix password file is/was used by many IMAP implementations. (at least the first implementations I know about). The user name constituted the left hand side of the address and the servers hostname and domainname constituted the right hand side of the address. The GCOS field was used to create a "friendly name" for the user. (Berkeley Mail and many other Berkeley utilities were based on the password file) Because these IMAP implementations used the password file, if the password file was compromised or a user name was acquired the account could be violated. Current password acquiring

methods exploit operating system frailties. In Unix systems that could be any number of mechanisms for acquiring access and the password file.

Acquiring password files

In order to acquire a password file, it is necessary to gain access to the system in some way. This can be done through many well known exploits. IP Address spoofing, circumventing IDS, using sniffers, hijacking sessions, social engineering are all mechanisms for acquiring access to password files and other system information.

Password file from a UNIX system

If the hacker has your password file and a comprehensive dictionary all that needs to be done is run the crack program against the file. This will take some time depending on the system resources the crack program has, but will eventually crack your passwords.

Crack is a very comprehensive tool. It uses the dictionary but also takes the cracking process further by including common and not so common techniques frequently used by individuals to create passwords. Users substitute numbers and other characters for letters in the password. This is the same technique used for license plates on cars. For instance **theregr8** forms an intelligible phrase that is easy to remember. The cracker program takes these types of variations into consideration. The bottom line is if the hacker has a means of cracking your passwords such as your password file, the hacker will have access to your system. However, once the hacker has user name/password pairs the hacker must have access to the system in order to use them. For instance if the user used a cgi-bin exploit to acquire the password file, that system must also have a means to logon. This can be by means of a telnet session (which should not be running on a web server!), or physical access to the system.

LDAP and DAP Directory

Directories such as DAP and LDAP allow applications to use a centrally managed repository for information. For some applications this includes passwords. The password is stored in the directory in an attribute determined by the application program. A well known directory attribute is “password”. Some LDAP implementations, by default, store passwords in a weak XOR based encryption so that it (or applications using it) can facilitate challenge based authentication methods such as CHAP and CRAM. Therefore, unlike a password file which you would then have to go crack, if you obtain the underlying data in a directory you have all the passwords instantly. Centralized directories may become a big security threat to operational environments. Improvements to the design and implementation of LDAP directories is critical to the roll out and bulletproofing directories that are primed for e-commerce on the Internet. Many ISP’s and corporations use IMAP based messaging that involves an LDAP directory. The remainder of this paper will discuss the mechanism by which applications access the directory and how to trace interactions from the application server such as IMAP to the directory.

IMAP utilization of an LDAP directory

This section shows the use of an LDAP directory by an IMAP server. It illustrates the information that is passed between the various components involved in the use of e-mail.

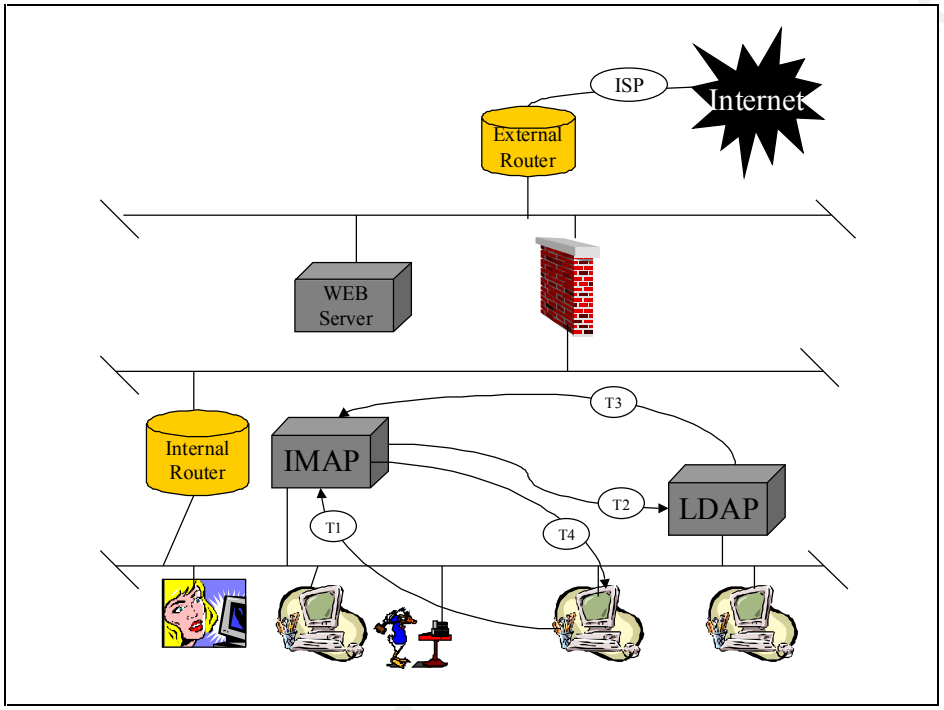

Figure 1 Typical IMAP Interaction with the Directory

Figure 1 shows the interaction between a protocol such as IMAP and the LDAP protocol. The box labeled IMAP is probably a message store (contains user mailboxes). Once the user is authenticated the IMAP server allows the user access to their mailboxes. During T1 the user makes a request to the IMAP server to logon (USER: joe;PASS:joey). The IMAP server then queries the LDAP server for the user name and password, (T2 IMAP server ask for information in the user entry identified by the following Distinguished Name (DN) cn=joe,ou=users,o=acme mining,c=us). During T3 the LDAP server finds the user entry and pulls out the password for the user and sends it back to the IMAP server (passwd: Lkwjeiae). Depending on the implementation of the IMAP service the LDAP server may give the IMAP server much more information such as the location of the user's mailboxes, filtering rules and access control information. But once the IMAP server decrypts the password and compares it to the one the user originally used to login and it is valid the user now has access to his mail.

Time	Protocol	Direction	Data passed over network

T1	IMAP	User → IMAP	Username, Password
T2	LDAP	IMAP → LDAP	DN of user requesting Password or a search string for the user
T3	LDAP	LDAP → IMAP	Users Password
T4	IMAP	IMAP → User	Access is granted

Table 1 Protocols used during the authentication process with IMAP using LDAP

Identifying and investigating an intrusion

Using an LDAP service for user identification and application configuration constitutes a distributed application. The IMAP application illustrated in Figure 2 "Typical IMAP interaction with the directory" there must be 3 software components involved. The first is a User Agent (Netscape or Outlook for instance) that is used by the client to access his mailbox. The application server component, in this example, the IMAP server and the LDAP service. All three of these components can run on a single system, but they may be hosted on different servers possibly for performance reasons. Assuming these components run on separate servers the log files from each component must be investigated when a break-in is suspected. This is a good argument for using the network time protocol (ntp) to keep the timestamps in these log files in sync if at all possible.

Application

The first place to look when a break-in is suspected is the application server. In a hypothetical situation where an e-mail user complains he is missing messages that colleagues claim they have been sending e-mail to Road but Road has not seen them. The IMAP server is the first place to look. Looking through the IMAP log shows that Road has logged in and logged off between the hour of 12:00PM and 12:30PM. Road claims that he was at lunch during this time and could not have been reading e-mail. So now we have a time frame in which to investigate this problem. I would look through the IMAP log to determine what IMAP transactions Road did during this time. This investigation may show just what messages were removed and read. The next place to look would be to look at the syslog file (where sendmail logs all messages coming from and going to the IMAP server). With this information you can determine the originator and recipient addresses so you can pinpoint the user sending the message to Road that the intruder has deleted. But… This does not explain how the intruder gained access to Road's e-mail.

```
08-29-2000.09:35:24 imsd-204: 438 Starting session
08-29-2000.09:35:24 imsd-204: 438 host name: XTRM0991
08-29-2000.09:35:24 imsd-204: 438 >> * OK grr-002.CP.ACME.COM IMAP4 server ready
08-29-2000.09:35:24 imsd-204: 438 << 00000000 CAPABILITY
```

```
08-29-2000.09:35:24 imsd-204: 438 >> 00000000 OK CAPABILITY complete
08-29-2000.09:35:24 imsd-204: 438 << 00000001 AUTHENTICATE CRAM-MD5
08-29-2000.09:35:24 imsd-204: 438 >> 00000001 OK AUTHENTICATE complete
08-29-2000.09:35:24 imsd-204: 438 authenticated as "Road.Runner" (domain=<none>, user=
<none>)
08-29-2000.09:35:24 imsd-204: 438 << 00000002 MYRIGHTS INBOX/One
08-29-2000.09:35:24 imsd-204: 438 >> 00000002 OK MYRIGHTS complete
08-29-2000.09:41:45 imsd-204: 438 << 00000003 LOGOUT
08-29-2000.09:41:45 imsd-204: 438 >> 00000003 OK LOGOUT complete
08-29-2000.09:41:45 imsd-204: 438 closed (IMAP4)
08-29-2000.09:41:45 imsd-204: 438 Ending session
```

This is an example of a user logging into their IMAP server to read e-mail. This example shows that the user logged in to see if there was any e-mail. In this case there was no new mail. This user stayed logged in for 6:21 more minutes and then logged out.

LDAP Server (X.500 backend)

The LDAP server logs should be investigated to determine is there is an LDAP brute force login attack. This could be an attack to acquire the directory's manager password giving access to any information in the directory or simply the user's password that the intruder is interested in. The intruder may have gained access to Road's user information by breaking into the LDAP server. This may occur by using various algorithms to crack the users password attribute in the directory.

```
08/29 09:34:14 GDServer 00269 (root ) X500 DAP context association (364): Internet=1
0.87.61.222+1512
08/29 09:34:14 GDServer 00269 (root ) Bind (364) (simple): c=GB@o=Acme@cn=Appl ication
Services@ou=Hubs@ou=Explosives@ou=grr-002@cn=Manager
```

This X.500 log segment shows that the IMAP server binds (logs on to the) X.500 directory using the manager Distinguished Name (X.500 DN) "c=GB@o=Acme@cn=Appl ication Services@ou=Hubs@ou=Explosives@ou=grr-002@cn=Manager". This gives the IMAP server all rights to the directory the this user is granted. Since it is the manager, it probably has all rights to the directory. This implementation of X.500 tracks bind's with an internal number, in this example the internal number is 364. Once the directory is bound, the only reference for this connection is this internal number. So note the IP address! If this IP address is not that of the IMAP server, it warrants further investigation.

```
08/29 09:34:45 GDServer 00269 (root ) Search (364):
c=GB@o=Acme@ou=Users@ou=Explosives@ou=ExplosivesNorthEast
08/29 09:34:45 GDServer 00269 (root ) Search service controls: preferChaining copyShallDo
08/29 09:34:45 GDServer 00269 (root ) Search (hash) subtree (cn=Road Runner)
```

```
08/29 09:34:45 GDServer 00269 (root  )  Using exact hash matching for equality on: Road Runner.
08/29 09:34:45 GDServer 00269 (root  )  Search result: 1 of 1 matching entry found (0); Response within
limits.
08/29 09:34:45 GDServer 00269 (root  )  Result sent (364)
```

The above clip of the directory log shows when the request for this user information came in. This shows the directory sending the user information to the IMAP server so that the users password can be compared to the one that the user typed in.

This same process should be used when investigating an intrusion. The first thing I do is to reduce the logs to a reasonable size. I do this by using the Unix "grep" command. If I have to deal with NT log files, I move them to a Unix system to do this. If I know the name of the user I am chasing, then I look through the log files for that information, first on the application server.

% grep –i runner imsd.log >/tmp/imsd.runner.log

If I know a time frame then I narrow it down to an hour or two. By doing this you can see what the user was doing. I can then look at the directory logs during the time that the incident took place.

Making LDAP directory transactions more secure

Use a reasonably strong encryption algorithym for passwords. This is an obvious and small part of securing the directory. Remember that the directory can and does hold configuration information not only for users but also for applications. This information in the wrong hands could allow for more devastating consequences than just being able to access an individuals information.

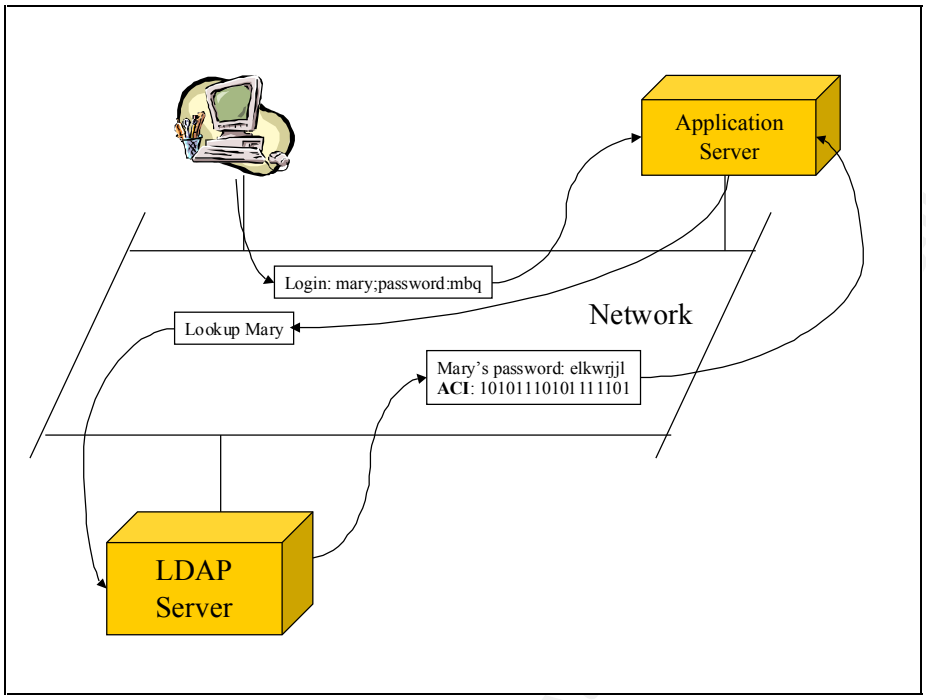

Figure 2 Information on the network during a login transaction

The information that is going over the network can be obtained by using the same techniques as you would use to obtain passwords over the network. But now, in addition to obtaining the password the cracker can also get additional information about the user and the application the user is accessing. Note the Access Control Information (ACI) that is also passed to the application server over the network. This could be used by the application for any number of things.
The next step would be to secure the communications over the network using a technique such as Secure Socket Layering (SSL).

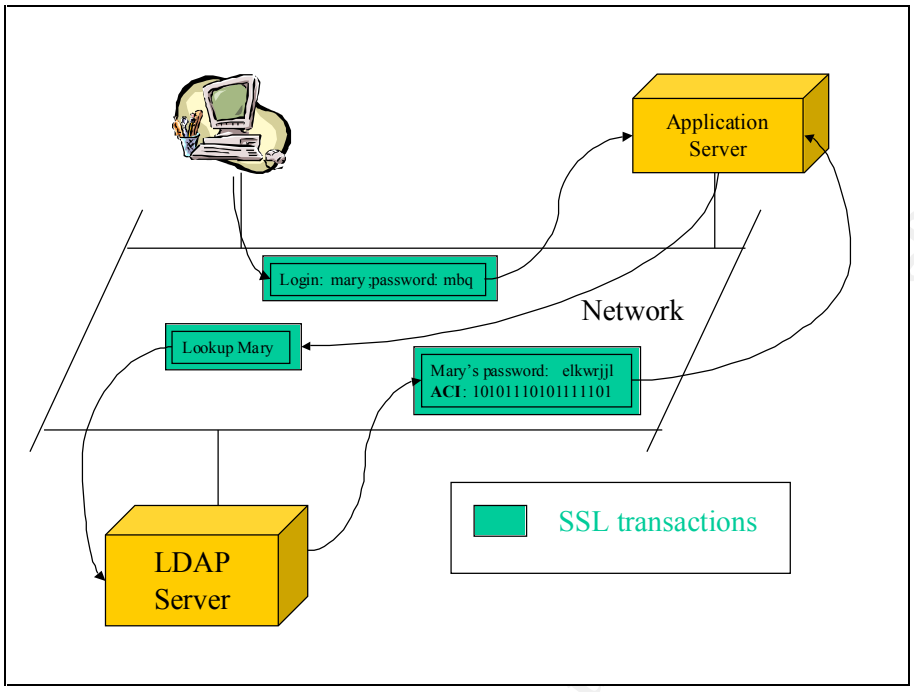

Figure 3 LDAP and Application Information is Secured using SSL

The information that is being passed over the network is shown being wrapped in a secure socket layer (SSL). This improves the security of these transactions. However, this does not address the issue of someone gaining access to the files as they are stored in a file system. So I say, encrypt those as well.

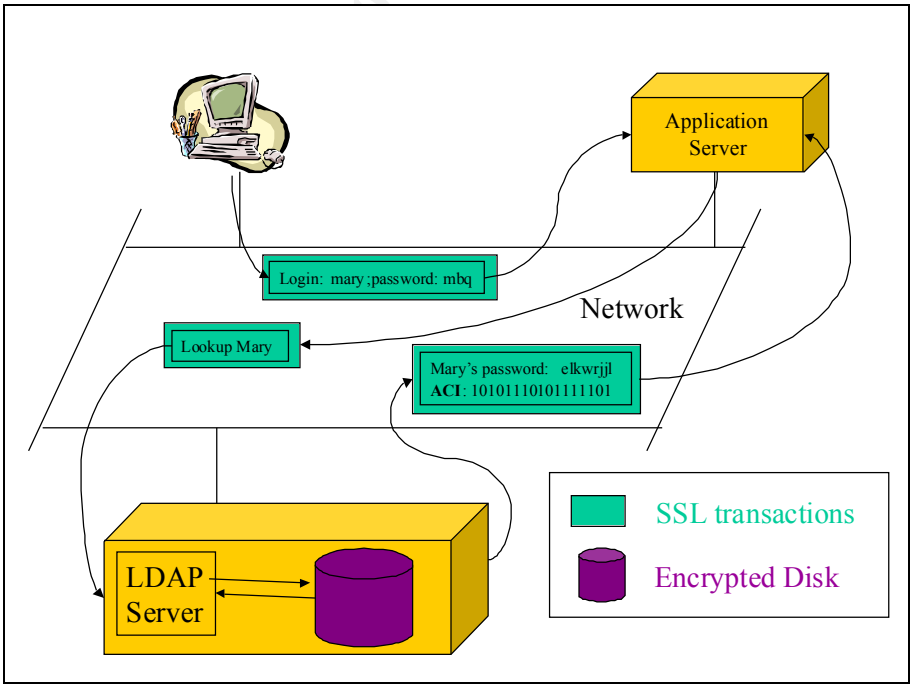

Figure 4 Encrypt Information Stored on Disk

Figure 4 shows the directory being encrypted on the disk on the LDAP Server. This provides another level of security in the event the LDAP Server is compromised.

In addition to encryption, access to the directory may also be restricted through the use of Access Control Information (ACI's). Depending on the implementation of the directory access can be controlled to any branch of the tree and sometimes down to the attribute level.

Applications and Vendor Use of LDAP directories

There are many other applications that use an LDAP directory for user information. More and more well established and widely used operating systems and applications use the LDAP protocol. Many vendors are including the ability to store and retrieve configuration information in an LDAP directory. This makes the directory critical component of the enterprise that uses it.
Vendors capable of using LDAP include CISCO (routers), Checkpoint (Firewall), Entrust (Public Key Encryption) to name a few. If you will notice each of these vendors products are commonly used to secure and protect the enterprise. This is the reason that securing the LDAP protocol and information in the directory is paramount for the secure use of LDAP.

Figure 8 shows a picture of application servers, each a separate entity requiring user authentication. In addition to separate logons and passwords for each application, they may all have different criteria for creating usernames and passwords. One application may require the username to be 10 characters while the next one requires only 8. In addition the passwords could range from any 8 character to a pass phrase. This requires the user to write down their passwords and store them on yellow sticky things on their computer monitor.

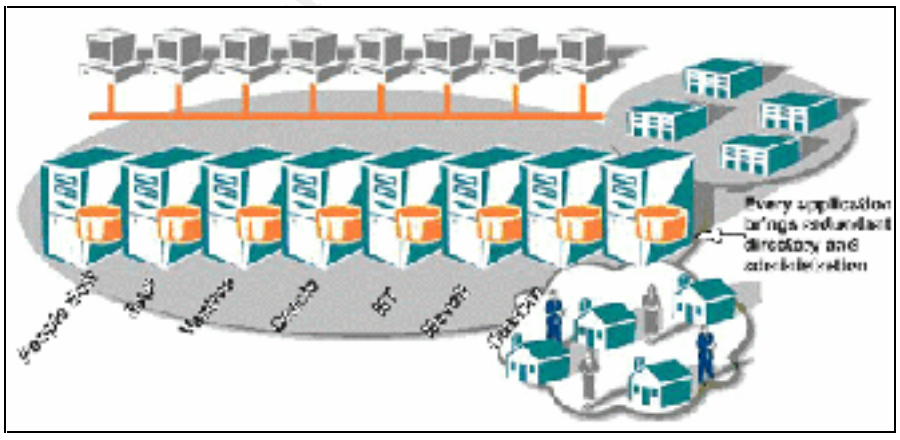

Figure 5 Individual Applications Using Proprietary Auth Methods

Figure 7 shows a hypothetical implementation of the same applications using the LDAP directory. Depending how these applications were implemented, they may use user information as well as configuration information.

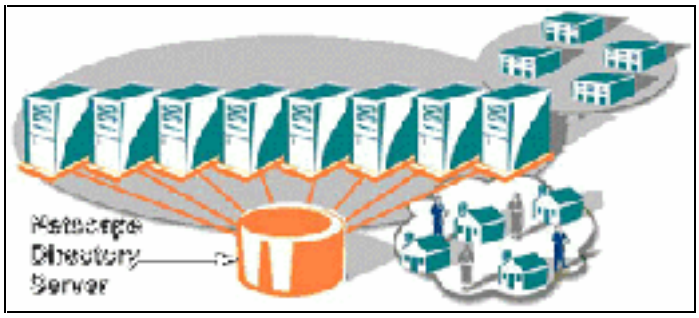

Figure 6 Illustrates Individual Apps Using LDAP

This is all fine and well, but why not take it a step further and not require the user to login more than one time? With the implementation of Single Sign On (SSO) technologies this is possible. It is also beyond the topic of this paper.

Other Considerations When Using LDAP

Some attacks on LDAP directories have already occurred. As the use of LDAP becomes more and more prevalent attacks against this protocol will increase. To make matters more complicated, an LDAP directory is often the front end to a distributed X.500 directory.

Differences between the X.500 DAP protocol and the LDAP protocol

First lets talk about the difference between the X.500 Directory Access Protocol (DAP) and the Lightweight Directory Access Protocol (LDAP). Two important differences are that LDAP will not make referrals and it will not replicate directory information. As far as LDAP is concerned, all information in the directory is owned (mastered) on the system it is making queries to. With an X.500 directory the information in the directory may be shadowed or copied. Some applications that use this directory information require the information from the true source. This may require a referral to another directory that masters the data. This is why, when you use LDAP as a front end to an X.500 (DAP) directory there can be many machines involved when investigating an LDAP breakin. An LDAP directory can have a cascading affect as shown in Figure 11.

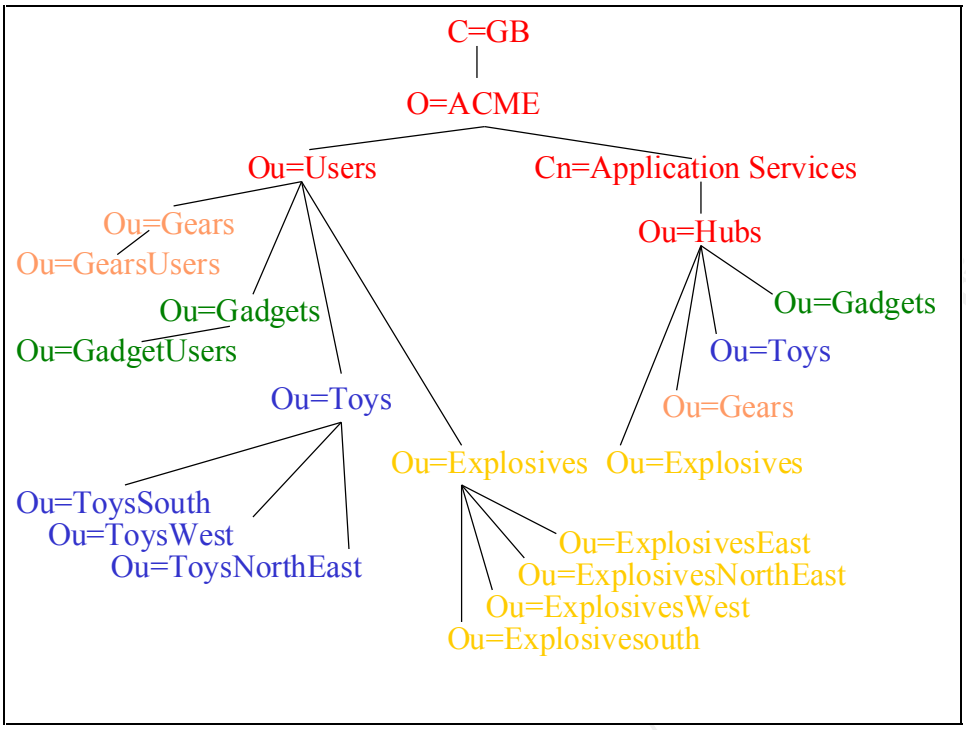

Figure 7 Disitributed X.500 Directory

Lets assume that each color of this X.500 directory resides on a different server. And that each directory owns or masters their part of the directory. The top level of the directory pictured in red is a superior reference for each of the other 4 subordinate directories. The Superior directory is consulted when the local directory does not have an authoritative answer. This directory can then broadcast or refer a query for the information to one or all directories it knows about. In this example, Gears, Gadgets, Explosives and Toys . For example, if the following LDAP query came into the Gadgets server depending on the way the X.500 directory is implemented, the following could happen.

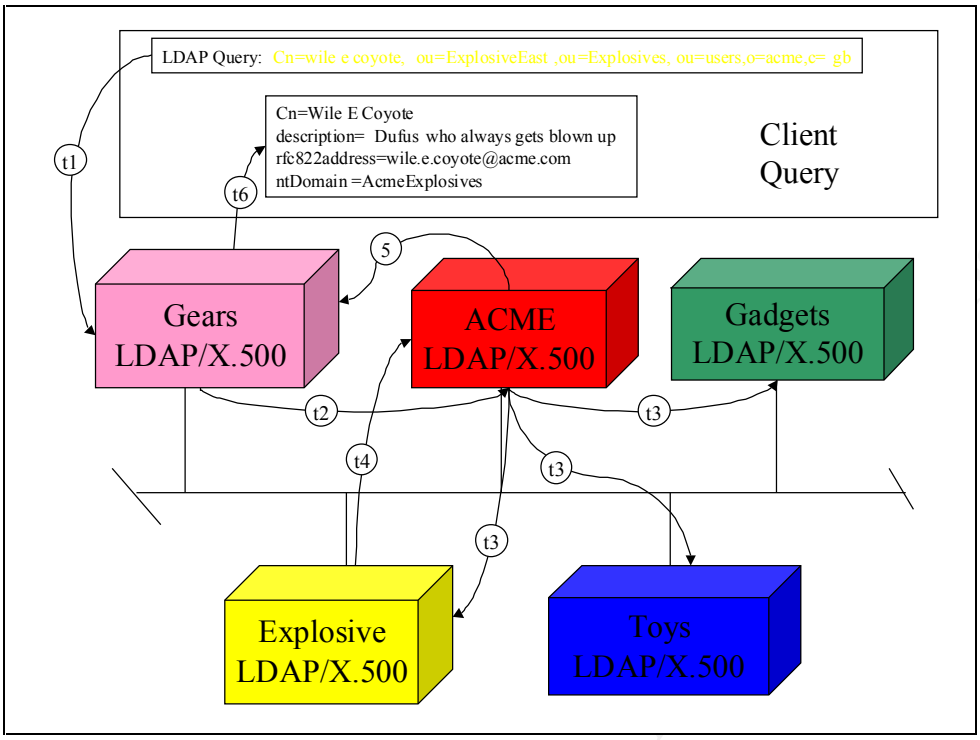

Figure 8 LDAP Query with an X.500 Backend

This LDAP query originates on the Gadgets X.500 directory server Cn=wile e coyote, ou=ExplosiveEast,ou=Explosives,ou=users,o=acme,c=gb Suppose the Gadgets directory does not know about the other three directories including the Explosive directory, this means that the Gadgets directory must refer to the top level directory shown in red. The red directory will see this request and contact the Explosive directory. The Explosive directory can then be asked to return the results to the red directory or return the results directly to the Gadgets directory. The point is that in the event of a break in it will be necessary to investigate each of the LDAP and X.500 logs to confirm or deny an attack on the directory happened. It is also very dependent on how the directories are configured as to how they request data from other directories.

References

There are numerous RFC's that discuss the X.500 directory and the LDAP directories. There are also RFC's that suggest the use of X.500 directories for uses such as Domain Name Services (DNS) and other applications.

RFC	TITLE
RFC2253	Lightweight Directory Access Protocol (v3)
RFC2256	A Summary of the X.500(96) User Schema for use with LDAPv3
RFC2307	An Approach for Using LDAP as a Network Information Service

RFC2425	A MIME Content-Type for Directory Information
RFC2459	Internet X.509 Public Key Infrastructure Certificate and CRL Profile
RFC2589	Lightweight Directory Access Protocol (v3)

Table 2 Related Request For Comments (RFC's)

LDAP Vulnerabilities identified to date (securityfocus.com)

The following vulnerabilities were documented at the securityfocus web site. I have copied the descriptions for each of the vulnerabilities from the securityfocus web site. As more and more applications are configured and implemented to use an LDAP directory there will be more and more exploits targeted at LDAP. These implementations appear to not have distributed directories. That is the application and the LDAP server run on the same system. The following show 6 such intrusions that occurred since March of 1999.

1 - NT IMail LDAP Buffer Overflow DoS Vulnerability

The IMail ldap service has an unchecked buffer, resulting in a classic buffer overflow vulnerability. While it does not crash the service, it drives CPU utilization up rendering the system essentially unusable.

2 - Lotus Notes Domino Server 4.6 NLDAP DoS Vulnerability

The component of Notes that handles the LDAP protocol, NLDAP, has an unchecked buffer. If too much data is sent to the ldap_search function it will cause a PANIC error in the Domino Server, halting all Domino services on the target machine.

3 - Check Point Firewall-1 LDAP Authentication Vulnerability

With FireWall-1 Version 4.0 Checkpoint introduced support for the Lightweight Directory Access Protocol (LDAP) for user authentication. It looks like there's a bug in Checkpoint's ldap code which under certain circumstances can lead to unauthorized access to protected systems behind the firewall.
A user can authenticate himself at the firewall providing a valid username and password. The firewall acts as a ldap client, validating the credentials by a directory server using the ldap protocol. After successful authentication access will be granted to systems protected by the firewall.
In contrast to authentication using the Radius or SecurID protocol, after successful authentication the directory server can supply the firewall with additional ldap attributes for the user like the time and day of a week a user is allowed to login, the source addresses a user can run a client from, or the system behind the firewall a user is allowed to access. This can be done individual for each user.
In general I think that's a great idea but it seems Checkpoint made something wrong interpreting the ldap attribute 'fw1allowed-dst' which is supposed to

control in detail which protected network object a user can access. It seems this attribute is ignored by the firewall software, granting access to all protected network objects instead.

Example:

------ Server 'Foo'

|

Internet --- FW-1 ---|

|

------ Server 'Bar'

Supposed there's a user 'Sid' with access only to Server 'Foo', and a second user 'Nancy' with access restricted to Server 'Bar', both controlled by the ldap protocol, using the ldap attribute 'fw1allowed-dst'. The bug will cause that both, Sid and Nancy, will have access to Foo and to Bar.
[Quoted from the post by Olaf Selke with permission]

4 - Shiva Access Manager World Readable LDAP Password Vulnerability

The Shiva Access Manager is a solution for centralized remote access authentication, authorization, and accounting offered by Intel. It runs on Solaris and Windows NT. Shiva Access Manager is vulnerable to a default configuration problem in its Solaris version (and possibly for NT as well, though uncomfirmed). When configuring the Access Manager for LDAP, it prompts for the root "Distinguished Name" and password. It stores this information in a textfile that is owned by root and set world readable by default, $SHIVA_HOME_DIR/insnmgmt/shiva_access_manager/radtac.ini. This file also contains information such as the LDAP server's hostname and server port. This information can be used to completely compromise the LDAP server.

5 - Netscape Professional Services FTP Server Vulnerability

Certain versions of the LDAP-aware Netscape Professional Services FTP Server (distributed with Enterprise Web Server) have a serious vulnerability which may lead to a remote or local root compromise. The vulnerability in essence is a failure of the FTP server to enforce a restricted user environment (chroot). By failing to do this an FTP (anonymous or otherwise) user may download any file on the system (/etc/passwd etc.) as well as upload files at will at the privilege level of the FTP daemon.

Furthermore (quoted from the original attached message) this FTP server supports LDAP users; different LDAP accounts are served on single physical UID. This means, any user can access and eventually overwrite files on other accounts; as it's used in cooperation with webserver, typically virtual web servers are affected.

6 - OpenLDAP 'ud' Group Writable Vulnerability

The "Interactive LDAP Directory Server query program", ud, which ships with OpenLDAP, is installed by default mode 775. Depending on the group it is installed as this could present a security issue and possibly be used to elevate privileges.