The importance of LDAP

Past, present and future of a battered protocol

Tom Jackiewicz

All that you know about Lightweight Directory Access Protocol (LDAP) is wrong. From its inception to perceived usefulness, and ultimately, until the marketing department got a hold of it, LDAP has grown. It started as a useful protocol and a data structuring methodology (known by only a few), and became the latest and greatest way to synergize your *action items* and find parity with your eMarketing growth plan.

The freedom given to those who choose to deploy LDAP has ultimately led to problems in interoperability

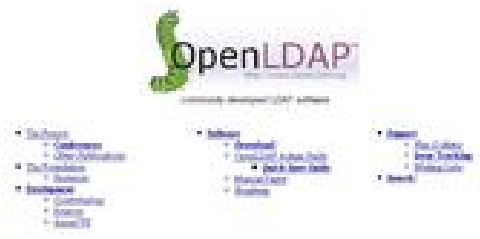
What does this mean? Hopefully your answer is the same as mine - nothing. The importance and growth of LDAP should be based on how the protocol has adapted in the past, and how we keep on innovating and adapting it as technology grows. Unfortunately for us, the marketing department has already taken a hold of LDAP, eaten it up, and - if we are not careful - it will spit it out.

The origins of LDAP

LDAP was first implemented at the University of Michigan in 1992 as a way of creating an interface to DAP over TCP/IP. This eventually evolved into a stand-alone system utilizing data structures based on types stemming from X.500. Over time, the popularity of OSI waned and TCP/IP became the de facto standard accepted for networking. One of the reasons for the failure of OSI (and its directory solution – DAP) was the complexity of the data definition and the infrastructure required to use it. No flexibility was given in any of the OSI standards and it became extremely cumbersome to deploy.

With OSI's failure and TCP/IP's acceptance, the LDAP "community" (which at the time consisted of college age engineers meeting over pizza and beer) became more ambitious and tried to invade the space of directories and started to innovate instead of just following X.500's lead. Following the original RFC's for X.500 and LDAP, it can be seen that while X.500 was trying to solve problems that didn't exist (i.e. how to replace yp, how to create a more structured information standard that was even more painful to implement), LDAP was clarifying how to access information and proposing formats for URL standardization and search filters.

Fig. 1: The OpenLDAP web site



Because of its simplicity, ease of implementation, availability, and a little bit of luck, LDAP became a good way of centralizing information. LDAP provided an easy solution where synchronization scripts and custom code were used to keep information, on large systems, consistent. Within a relatively short time, email systems were relying on LDAP as the single authoritative source. Authentication systems were no longer relying on their own customized solutions and flat files. Netscape looked to LDAP to provide the central repository for their initial suite of products.

Following the original RFC's for X.500 and LDAP, it can be seen that while X.500 was trying to solve problems that didn't exist, LDAP was clarifying how to access information and proposing formats for URL standardization and search filters

All of a sudden, LDAP was the quick and easy solution to many of the problems that system administrators faced. The community, large and growing, was updating standards in order to help solve the problems that the users were facing. Once other vendors were replacing their back-end databases with LDAP, it was obvious that this little experiment had finally gained acceptance.

If it ain't broke, don't fix it

One can easily conclude that the success of LDAP can be attributed to how fast technology evolved during the latter part of the 20th century and the general laziness of the information technology community. While new products, solutions, ideas and toys where becoming available, it was too difficult (and too expensive) to implement a database (such as Oracle) for storing profile information for all of a system's users. It was equally frustrating, migrating flat files across different systems, which required similar information. Deploying an LDAP directory didn't require much in the way of an investment of time. It could also be used by some of the new products that were being released. One LDAP directory provided the same authentication and profile information for all of the new toys that system administrators wanted to experiment with. So if it didn't live up to their expectations, only hours were wasted (instead of the resources required to deploy an instance of Oracle).

Unfortunately, in today's fast-paced world, a product that isn't constantly updated with all the new bells and whistles is not trendy or exciting. Vendors often remove features just to add them again a few revisions later. The LDAP community, through RFC's, fell into this trap and started solving problems that didn't exist. RFC's were proposed to adapt DNS information into LDAP. Even complicated schema was proposed to store Java objects. LDAP was slowly moving into areas that it didn't need to exist, and that just made it more complicated. These were the same types of endeavours that destroyed OSI, X.500 and DAP. Those who fail to learn from past mistakes are destined to repeat them.

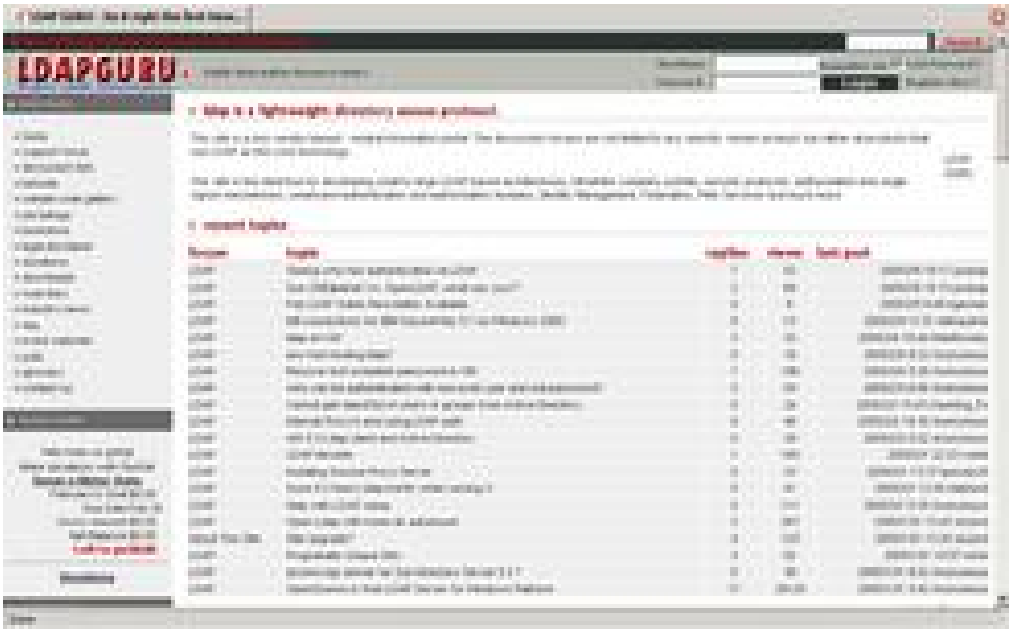
Vendor interpretation

The acceptance of a technology by a vendor is a blessing for many. For others, it can be a fast spiral towards obscurity. Initially, LDAP gained widespread acceptance by the information technology community because of its use within Netscape's suite of products. However, as other vendors started to tap into the possibility of LDAP, their proprietary system background began to invade LDAP's space. LDAP gained popularity because of well established standards and the ability to be protocol dependent and vendor (or implementation) independent. The data stored in the original University of Michigan LDAP server could be exported and put into OpenLDAP with ease. However, vendors (like Sun, Novell, and Microsoft) chose to implement good (but proprietary) features that required only the use of their implementation. Direct calls for authentication and authorization via LDAP became requirements of proprietary plug-ins and new integration layers. While some of these features went beyond the scope of LDAP, standards should have been written, and these new feature sets could have easily become part of the standard LDAP feature set. Instead, the question quickly went from "Are you using LDAP?" to "Which LDAP are you using?".

To make things worse, the way vendors added LDAP to their offerings was questionable.

Early adapters integrated all of their product offerings with LDAP - despite their dependence on proprietary features of their LDAP implementations.

Fig. 2: The site LDAPGURU.COM is helping to provide the right LDAP information



Late LDAP adapters quickly added inferior LDAP support to their products. Instead of having their products query LDAP directly for information, they decided to synchronize their products daily with an existing LDAP server and then pump the data into a database. Alas, their marketing literature could now exclaim "We support LDAP!"

Other vendors decided to use LDAP directly, but their schemas were ported from previous database-centric products and these directories could exclusively be used by them. This led to having multiple LDAP servers with the requirement for synchronization – one of the problems that LDAP was supposed to solve!

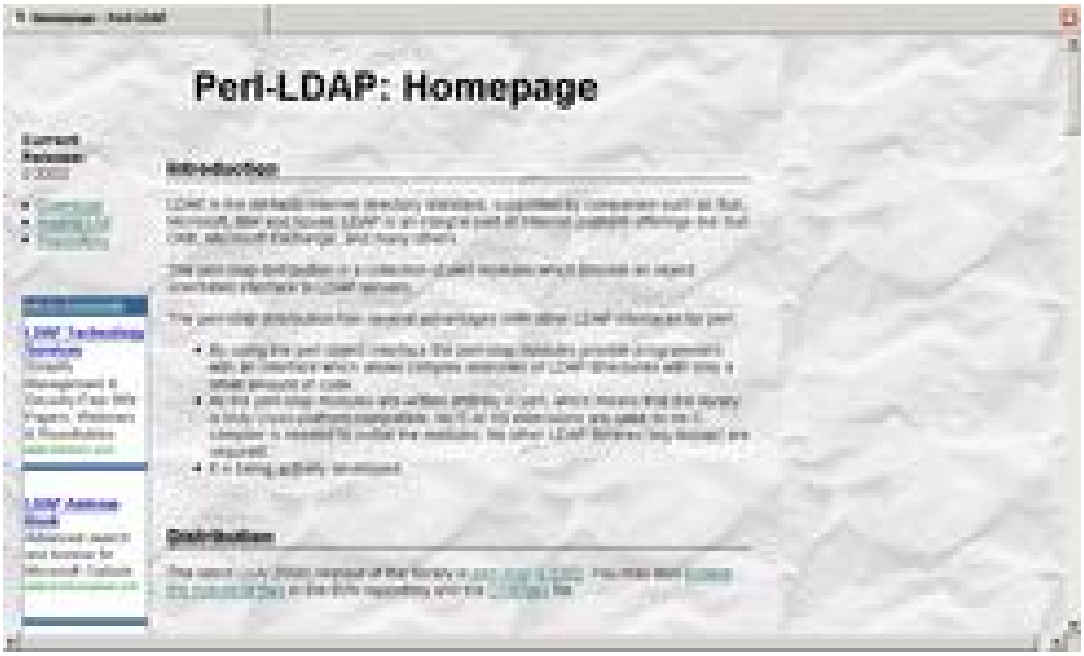
Today, some of these problems are overlooked (it is, after all, an open standard) and LDAP is a popular (and exposed) protocol. The problem now is that people who see the bad side of LDAP often fail to see how important it really is.

User interpretation

The freedom given to those who choose to deploy LDAP has ultimately led to problems in interoperability. DAP was largely ignored because too much time needed to be invested to plan an appropriate deployment strategy. Too much information would also have been needed to create a schema compliant user profile. LDAP left almost everything to the imagination. To quickly deploy LDAP and create user profiles, no planning was required. There were no standards for information (this would have been a hindrance initially and too close to X.500) and no best practices were provided. As with all new technologies, no one went far enough to have made any significant mistakes. Whoops.

A quick fix or temporary solution turned into an internal standard. When "Tom Jackiewicz" was created by different LDAP administrators in different environments, I could authenticate as *tom*, *tjackiewicz*, my badge number, *tjackiewicz* followed by my badge number, or *tjack* when the name was just too difficult to spell. While many of these were deployed in test environments, they were quickly adopted and became corporate standards that could not be easily changed to meet real needs within the environment. It was realized, that by choosing these naming standards, without any forethought, it would become difficult to integrate LDAP with other directories, databases, or data sources. Even the short-sighted deployment of a directory information tree (used to create branches within a flat directory) hurt integration efforts. At the top of the tree might be definitions for the whole internal user base. However, as LDAP is needed in other areas (such as external customers), or is used to store other data types, the lack of planning for the directory information tree causes problems when applying access controls or even setting appropriate search filters.

Fig. 3: Perl-LDAP provides a standard way of utilizing Perl to access your LDAP directory



Conclusion

The future of computing is currently in the hands of marketing departments and corporations. It has been pulled from the hands of the universities and computer scientists, innovating for the sake of doing what is right. What we must do, as a community, is insist on standards. Good yet proprietary ideas created by the vendors must be cherry picked and be turned into well-defined standards. We shouldn't let LDAP lose its simplicity and, ultimately, the reason we are using it in the first place. We should also make it clear to the vendors that we won't base our LDAP deployments on their systems. We want the ability to use whatever implementation we choose without having to conform to their ideas of what LDAP should be.

Bibliography

[1] Jackiewicz, Tom "Deploying OpenLDAP", Apress:2004

Copyright information

About the author

Tom Jackiewicz is currently responsible for global LDAP and email architecture at a Fortune 100 company. Over the past 12 years, he has worked on the email and LDAP capabilities of the Palm VII, helped architect many large scale ISP's servicing millions of active email users, and audited security for many Fortune 500 companies. Jackiewicz has held management, engineering, and consulting positions at Applied Materials, Motorola, and Winstar GoodNet. Jackiewicz has also published articles on network security and monitoring, IT infrastructure, Solaris, Linux, DNS, LDAP, and LDAP security. He lives in San Francisco's Mission neighborhood, where he relies on public transportation and a bicycle to transport himself to the office -fashionably late. He is the author of Deploying OpenLDAP, published by Apress in November 2004.