Editor: Siobhán Clarke • siobhan.clarke@cs.tcd.ie

LDAP: Framework, Practices, and Trends

Vassiliki Koutsonikola and Athena Vakali • *Aristotle University*

Directory services facilitate access to information organized under a variety of frameworks and applications. The Lightweight Directory Access Protocol is a promising technology that provides access to directory information using a data structure similar to that of the X.500 protocol. IBM Tivoli, Novell, Sun, Oracle, Microsoft, and many other vendors feature LDAP-based implementations. The technology's increasing popularity is due both to its flexibility and its compatibility with existing applications.

A directory service is a searchable database repository that lets authorized users and services find information related to people, computers, network devices, and applications. Given the increasing need for information – particularly over the Internet – directory popularity has grown over the last decade and is now a common choice for distributed applications.

The Lightweight Directory Access Protocol[1] is an open industry standard that's gaining wide acceptance as a directory-access method. As the name suggests, LDAP is the lightweight version of the Directory Access Protocol and is a direct descendent of the heavyweight X.500, the most common directory-management protocol. Although they use a similar structure for data representation, LDAP and X.500 have several fundamental differences.[2]

- LDAP operates over the TCP/IP stack, whereas X.500 uses the OSI stack.
- LDAP's protocol element encoding is less complex than that of X.500.
- Each LDAP server uses a referral mechanism: if it can't satisfy a client's information request, it specifies the URL of an alternate LDAP server containing the requested information. In contrast, an X.500 server locates any missing data itself and serves it to the client without indicating the actual source server.

Many software vendors support LDAP due to its flexibility and the fact that it integrates with an increasing number of data retrieval and management applications. LDAP is thus an evolving ground for research on new and existing data management practices. Here, based on our own research, we offer an overview of LDAP, followed by a discussion of LDAP implementations and what's ahead for this technology.

LDAP Overview

To date, numerous LDAP-based servers have been released on the market, ranging from megascale public servers such as BigFoot (http://search.bigfoot.com/index.jsp) and Infospace (www.infospace.com) to small, workgroup-based LDAP servers. In between are the many universities and enterprises that have installed and configured directory servers to provide information about faculty, staff, and students in a way that works with the organizations' mail service, authentication systems, and application- and resource-access control. A list of public directory interfaces is available at the pan-European Delivery of Advanced Network Technology to Europe (DANTE) research network (http://archive.dante.net/nameflow/national.html).

Table 1 presents some of the most common Web-based services that use LDAP and summarizes the functionalities gained by integrating LDAP into existing data-related applications such as email, file transfer, and videoconferencing.

Data typically stored under LDAP includes

 Published by the IEEE Computer Society 1089-7801/04/$20.00 © 2004 IEEE

Table 1. Integrating LDAP with Web-based services.

Web-based services	Protocols and APIs for LDAP integration	LDAP-enabled functionalities
Web services	Secure Sockets Layer, Apache mod plug-ins	• Provides user-authentication mechanism • Defines restrictions and access-control lists
Web-powered databases	MySQL, PostgreSQL, Oracle 9i, IBM DB2, and so on	• Offers common access for multiple database-management systems
Domain Name Service	DNS protocol	• Delegates DNS service • Eliminates need for multiple DNS servers
Email	Simple Mail Transfer Protocol, POP3, Internet Message Access Protocol, WebMail	• Defines the user's `mailhost`, `maildir`, quotas, mail forwarding address, and mail aliases • Implements mailing list service • Provides user authentication
File transfers	FTP, WebDAV	• Defines user quotas for maximum space and file ownership • Defines home directory and server for storing files • Provides user authentication
Application environments	Java, XML, C/C++, Active Server Pages, Perl, Python, Hypertext Preprocessor (PHP), common gateway interface	• Supports multiple programming languages
Public-key infrastructure	OpenSSL	• Offers simple access to digital certificates and central storage of users' certificates • Provides automatic retrieval of LDAP client certificates from LDAP servers • Avoids useless data replication
Videoconferencing	H.323, H.320, Session Initiation Protocol	• Offers central storage of users' voice, video, and collaborative multimedia information • Scales up video and voice-over-IP operations from a few hundred endpoints to full enterprise deployments • Links account configuration details, authentication, and authorization to the enterprise directory

configuration files for network device drivers, user entries, application preferences, user certificates, and access control lists. LDAP's flexibility lets administrators create new attributes that can better serve their applications. With mail services, for example, a typical LDAP entry might contain attributes such as the `mailLocalAddress`, `mailHost`, `UserCertificate` (which stores the user's certificate in binary form), `ipLoginPort`, and `ipLoginHost` (for when the user makes a dial-up connection).

The LDAP Framework

LDAP operations are based on the client–server model. Each LDAP client uses the LDAP protocol, which runs over TCP/IP, to retrieve data stored in a directory server's database. LDAP clients are either directly controlled by an LDAP-installed server or managed by an LDAP-collaborating application. Figure 1 offers an overview of the LDAP framework, in which many devices (such as printers and routers) and servers (such as mail servers) can access data stored in a given LDAP server database. LDAP clients accessing LDAP servers should be authorized through authentication mechanisms, which can imple-

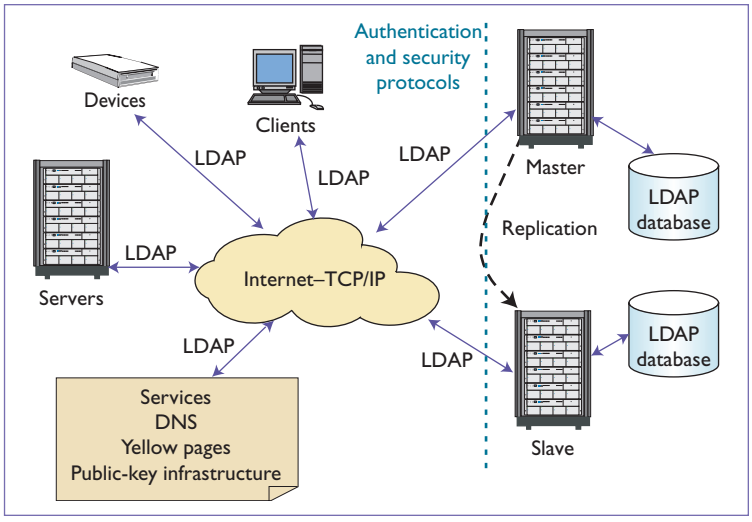

Figure 1. The LDAP framework. Devices and servers use the LDAP protocol to access data stored in LDAP server databases.

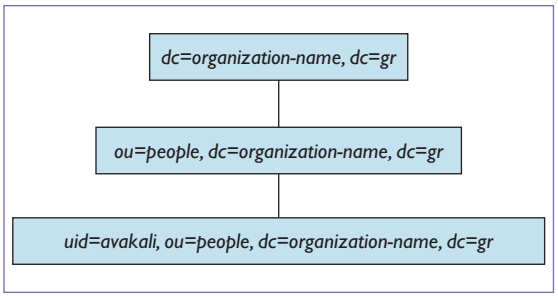

Figure 2. Example LDAP hierarchy. A distinguished name identifies each LDAP entry and declares its position in the hierarchy.

ment various security protocols. As Figure 1 shows, replication — in which a primary LDAP server (master) sends updates to a read-only replica server (slave) — is common among collaborating LDAP servers.

Two components are crucial to the LDAP framework: the LDAP-tailored database, or *directory*, and the data-representation format, which is based on XML.

LDAP Directory

LDAP directories are databases arranged as hierarchical information trees representing the organizations they describe. Figure 2 shows an example of a three-level hierarchy. Each LDAP entry is identified by a distinguished name (DN) that declares its position in the hierarchy. The hierarchy's structure forms the directory information tree (DIT), which originates from a root (RootDN). In the basic LDAP notations, *dc* stands for domain component, *ou* for organizational unit, and *uid* for user id. For example, the RootDN of the DIT that maintains user data for an organization in Greece would be `dc=organization-name, dc=gr`, while the DN of an authorized user's entry would be `uid=avakali, ou=people, dc=organization-name, dc=gr`.

The underlying LDAP database differs from typical relational databases in three key areas: data representation and structure, querying and transactions, and operational benefits and costs.

Data representation and structure. In relational databases, users define the schema; in LDAP, a fixed core schema controls the directory hierarchy. Also, whereas LDAP objects are nested in hierarchies, relational database objects are related to each other via primary and foreign keys that connect data items. Finally, LDAP data types and structure are flexible and extensible.

Querying and transactions. In relational databases, the query processor is aware of the relationships among database objects, whereas in LDAP, the corresponding relationships are extracted during the querying process. LDAP queries can also vary according to the DIT level at which (query-response) searching begins. For example, we might have the following two query types:

```
Query 1:
ldapsearch -h localhost -b
"dc=organization-name,dc=gr"
"uid=avakali"

Query 2:
ldapsearch -h localhost -b
"ou=people,dc=organization-name,dc=gr"
"businesscategory=Assistant Professor"
```

Here, the parameter `-h` declares the executing host and `-b` declares where searching will begin in the hierarchy. Therefore, Query 1 refers to the entry of the user with `uid=avakali` (searching starts from the node with DN "`dc=organization-name,dc=gr`"), whereas Query 2 refers to all the entries for which "`businesscategory=Assistant Professor`" (searching starts from the node with DN "`ou=people, dc=organization-name, dc=gr`").

Unlike relational databases, LDAP doesn't provide protocol-level support for relational integrity, transactions, and other relational database management system (RDBMS) features. For example, because an LDAP entry can have a simple value or multiple unordered values, LDAP supports multivalued data fields — that is, it "breaks" the first normalization rule, which states that fields with repeating values must be placed into separate tables. Finally, LDAP does not support relational database features such as views and joins.

Operational benefits and costs. In relational databases, write transactions and reading performance are critical, whereas LDAP directories are used mostly for reads. In addition:

- Most LDAP servers are simple to install and maintain, whereas RDBMS support demands considerable administrative effort;
- LDAP directories can be highly distributed, whereas relational databases are typically centralized; and
- LDAP servers can replicate some or all of their data using a built-in and easily configured

replication technology. Many RDBMS vendors consider such functionality "extra" and charge accordingly.

Finally, although relational databases efficiently support complex relationships between objects, in LDAP directories, it can be difficult to represent nonhierarchical relationships between objects.

XML and LDAP Tuning

XML is the dominant standard for Web data representation. Given their widespread adoption and integration with many Web-based applications, directory services tend to exploit XML's power and flexibility. Although current LDAP servers are not XML-enabled, the two technologies are quite similar in structure.

Directory Services Markup Language (http://xml.coverpages.org/dsml.html) is a new proposal for representing directory information in XML and thus bridging the gap between directory services and XML-enabled applications. DSML lets different vendors' directory services work together by describing their contents in XML syntax. An XML-enabled application can retrieve directory information in DSML by sending a request to the Web application server hosting the DSML service. DSML is defined using a document content description, which specifies the rules and constraints on XML documents' structure and content (for more on DCD, see www.w3.org/TR/NOTE-dcd).

Figure 3 shows a typical transaction in which the DSML service converts LDAP entries into DSML. This produces a DSML entry, such as:

```
<?xml version="1.0" ?>
- <entry dn="uid=avakali,ou=people,
      dc=auth,dc=gr">
- <objectclass>
  <oc-value>top</oc-value>
  <oc-value>person</oc-value>
  <oc-value>organizationalPerson
      </oc-value>
  <oc-value>inetOrgPerson</oc-value>
  </objectclass>
  <attr name="dn" />
- <attr name="businesscategory">
  <value>Assistant Professor</value>
  </attr>
- <attr name="mail">
  <value>avakali@csd.auth.gr</value>
  </attr>
- <attr name="ou">
```

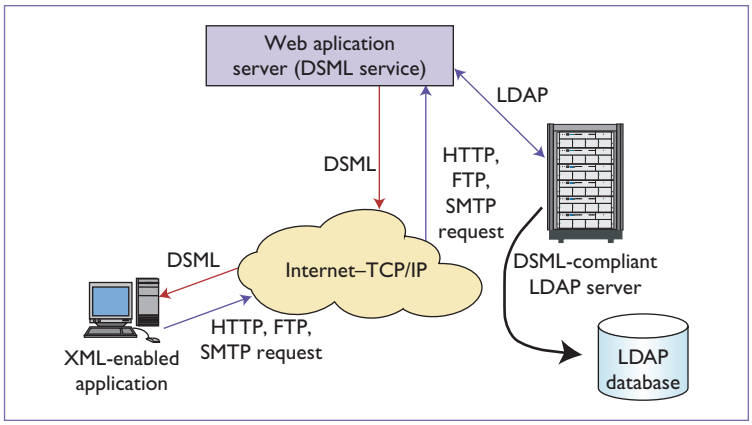

Figure 3. A transaction tailored to Directory Services Markup Language. The DSML service converts LDAP entries to DSML for XML-enabled applications.

```
  <value>Computer Science
      Department</value>
  </attr>
- <attr name="cn">
  <value>Vakali Athena</value>
  </attr>
- <attr name="sn">
  <value>Vakali</value>
  </attr>
- <attr name="givenname">
  <value>Athena</value>
  </attr>
- <attr name="uid">
  <value>avakali</value>
  </attr>
  </entry>
```

Coupling directories and XML imposes new data-storage and -retrieval requirements. There are several existing proposals for effective XML data storage and retrieval based on the LDAP technology. Some schema-driven approaches,[3,4] for example, involve mapping XML Document Object Model (DOM) nodes to LDAP entries (and vice versa). Such a process is based on an object processor that maps XML objects to LDAP objects by defining new LDAP object classes (to which XML nodes, elements, and attributes will be mapped). Another approach[5] maps XML DOM nodes to LDAP entries using LDAP object class definitions for XML nodes.

Because of structural similarities, specific modules can translate XPath queries to LDAP queries. More specifically, researchers have proposed a query model based on an evaluation algorithm that transforms any XPath query into a series of LDAP queries that achieve the original

Table 2. Key LDAP server features.

Features	Open LDAP	Sun ONE	Novell eDirectory	IBM	Oracle	Active Directory
Platform	Linux, Windows NT, AIX, BSD, Solaris	Solaris, Linux, HP-UX, IBM AIX, Windows	Linux, Windows, Solaris, AIX, NetWare/HP-UX	AIX, Solaris, Windows, Linux, HP-UX	AIX, Windows, Solaris, HP-UX, Unix/Linux	Windows
Security-authentication protocols	Kerberos, SSL/TLS, Cleartext, SHA/MD5 passwd, SASL	SASL, SSL/TLS, X.509 v3	Kerberos, smart cards, PKI/X.509, SSL, SASL/GSS API	Kerberos, SHA/MD5 passwd, PKI, biometric, smart cards	SSL/TLS, SASL, certificate	Kerberos, SSL, smart cards, PKI/X.509, SASL/GSSAPI
Backends	Postgress, Berkeley DB, SQL, Shell, passwd	Sybase, Berkeley DB	Flaim	IBM DB2	Oracle	MS SQL
Multimaster replication		✓	✓	✓	✓	✓
DSML support	✓	✓	✓	✓	✓	✓
Directory-enabled networking	✓	✓	✓	✓	✓	✓
SNMP monitoring		✓	✓	✓	✓	✓

AIX = Advanced Interactive Executive; BSD = Berkeley Software Distribution; GSS = Generic Security Service; HP-UX = Hewlett-Packard Unix; PKI = Public-key infrastructure; SASL = Simple Authentication Security Layer; SHA/MD5 = Secure Hash Algorithm/Message Digest 5; SNMP = Simple Network-Management Protocol; SQL = Structured Query Language; SSL/TLS = Secure Sockets Layer/Transport Layer Security

query's goal.[4] In another approach[6], users formulate XPath queries that are transformed to LDAP by an XML2LDAP component, and then an LDAP2XML component transforms the result from LDAP into XML. An XML parser can also translate LDAP data to XML.[7]

LDAP in Practice

Developers have long articulated the need for an industry-standard directory, and their need has been reinforced by numerous (and continuously evolving) applications that operate under the Directory Enabled Network (DEN) framework, including network-management applications that communicate with existing network devices, system-configuration files, voice-over-IP, videoconferencing, and so on.

The DEN specification concentrates on building a robust and extensible infrastructure that can model different network elements and services for easy storage and retrieval from LDAP-based directories and data stores. Interesting DEN initiatives include DEN-enabled switches (http://carol.science.uva.nl/~handree/DEN/D1/index_en.html) and directory services middleware for multimedia conferencing (http://metric.it.uab.edu/vnet/cookbook/v1.0).

LDAP Implementations

Today, many companies support LDAP-based directory services, and the directory market is becoming quite competitive. Standalone directory vendors such as IBM Tivoli, Novell, Sun Microsystems, Oracle, and Microsoft feature mature and effective LDAP-based implementations with robust multivendor integration capabilities. OpenLDAP (www.openldap.org), a suite of open-source directory software, is becoming competitive with these commercial directory servers as well.

Table 2 shows the main features of the dominant LDAP servers, which are similar in their broad range of platform support and authentication and encryption protocols, as well as in their DEN framework. Of the six major servers, all but OpenLDAP offer support for multimaster replication, in which two suppliers (primary LDAP servers that push changes) can accept updates, synchronize with each other, and update all consumers (replicated LDAP servers). These consumers, in turn, can refer update requests to both masters. OpenLDAP is also the only major server that doesn't provide Simple Network Management Protocol monitoring through a built-in agent or component, which gives network applications information about the directory server's status and activity.

LDAP services' wide adoption of XML is evident from the integration of DSML in many LDAP vendor-specific approaches.

- Novell has stated its support for DSML and has implemented DirXML (www.novell.com/products/dirxml), which offers a way to interface with eDirectory data and uses an XML interface to surface the data and change events. Essentially, DirXML support lets eDirectory expose valuable directory data to other applications using XML.
- IBM's Standalone LDAP HTTP API (Slaphapi) can return output in text, HTML, or DSML, accessing LDAP directories via HTTP. IBM has also developed XML Data Mediator (formerly XML Integrator; see www.alphaworks.ibm.com/tech/XI), a tool for bidirectional data conversion between XML and structured formats such as relational or LDAP data.
- Sun Microsystems' Java naming and directory interface API supports DSML (http://developer.java.sun.com/developer/earlyAccess/jndi).
- Unlike the gateway design used by most LDAP directories, the Sun ONE directory server offers native DSML support. By using DSML over HTTP/SOAP, ONE frees applications from LDAP reliance, thus letting non-LDAP clients interact with directory data.
- Microsoft provides DSML support for Active Directory and is also working on a mechanism that maps directory data into a DOM structure that can be accessed via XPath.
- LDAPProcessor (http://cocoon.apache.org/1.x/ldap.html) is a Cocoon processor that performs LDAP queries, translates the result set into an XML fragment, and inserts the fragment in the original document.
- LDAPHTTP (www.onjava.com/pub/a/onjava/2003/07/16/ldaphttp.html) translates XML to LDAP.
- The XMLDAP gateway (http://xml.coverpages.org/ni2001-03-02-a.html) is a standards-based solution that lets developers present LDAP directory data in multiple formats.

Such widespread LDAP support offers potential LDAP clients many choices. However, it's important to clarify and prioritize criteria before selecting a particular implementation.

Choosing an LDAP Server

Various experiments comparing LDAP server performance offer potential customers a list of important criteria and metrics to consider.

- *Time requirements*. Typical benchmarks compare LDAP servers' execution time for reading, searching, writing, and loading operations. To increase the reliability of results, experiments generally refer to more than one database load. Several researchers have tested time-critical applications;[8-10] others have analyzed query-response time in combination with aggregate throughput and latency.[11,12]
- *Binding information*. In LDAP interactions, bind operations are crucial: they initiate LDAP server connections by sending the client's authentication information. Metrics related to bind operations — including bind response time, number of bind requests, and bind errors — can significantly delay the overall LDAP operation. The bind response time depends on the authentication method, as Wang and colleagues note in their article on LDAP bind operations and connection latency.[12]
- *Search functionality*. This criterion includes search requests and errors, the average number and size of search results, search response time, and current searches. Search response times depend on several factors, including: query filtering; where the search starts in the data hierarchy; the number of query-requested attributes; and whether the query involves indexed attributes. Many organizations supporting LDAP servers regularly collect statistics on their search operations so they can monitor server performance; example organizations include the University of Vermont (http://mouse.uvm.edu/ldapstats/fishercat.uvm.edu-Search.html) and the University of Toronto (www.utoronto.ca/ns/stats/ldap.html).
- *Cache management*. Measurements here are important because directory servers use directory caches to improve response times. Researchers have explored the idea of using LDAP-related caches and have proposed an algorithm for improving responsiveness.[13] Cache-management metrics involve directory cache hits compared to overall directory cache requests; in LDAP cache services, users typically define the cache size.
- *Data load*. The data load is the number of bytes transmitted and entries sent between the LDAP server and its clients. Various metrics affect the data load, including connection requests, current connections, average connection length, and the average size of search results. LDAP server administrators can use various tools — such as the Mirabai-LDAP Metrics software

tool (www.forsitesolutions.com/mirabai.html) — to monitor the server data load, especially the near-real-time status.

Overviews of technical differences in LDAP servers, such as support for LDAPv3, access-control lists, multimaster replication, security protocols, and data integrity, are available through vendors (see wwws.sun.com/software/products/directory_srvr/wp_directory.pdf) and public institutions.[14]

LDAP Evolution: What's Next?

LDAP is currently in version 3, and we expect its ongoing evolution to address interconnection with X.500 directory services and thus facilitate the construction of a global Internet directory. Metadirectories, which manage integration and data flow between directory servers, offer one step toward the "marriage" of X.500 and LDAP servers. Many LDAP vendors, including Sun, Novell, and Microsoft, support metadirectories, and such support seems to be a trend for LDAP-based applications.

LDAP data management, particularly storage and retrieval, could improve significantly by tuning XML's integration with LDAP. Earlier efforts in XML data caching using the LDAP framework support this trend. As an example, HLCaches, an LDAP-based hierarchical distributed caching system for semistructured documents, has shown promising improvements by integrating caching in an XML- and LDAP-tuned environment.[15] This approach implemented an XMLDAP cache based on the OpenLDAP server and showed that the average access times have improved in comparison to more conventional caching schemes.

Current LDAP momentum is quite promising in terms of an Internet-wide LDAP adoption for data management frameworks involved in querying, indexing, caching, and security.

References

1. M. Wahl, T. Howes, and S. Kille, "Lightweight Directory Access Protocol (v3)," IETF RFC 2251, Dec. 1997; www.ietf.org/rfc/rfc2251.
2. T.A. Howes, *The Lightweight Directory Access Protocol: X.500 Lite*, tech. report TR-95-8, Center for Information Technology Integration, Univ. of Michigan, 1995.
3. XLNT Software, "Handling XML Documents Using Traditional Databases," Aug. 2002; www.surfnet.nl/innovatie/surfworks/xml/xml-databases.pdf.
4. P.J. Marron and G. Lausen, "On Processing XML in LDAP," *Proc. 27th Int'l Conf. Very Large Databases*, ACM Press, 2001, pp. 601–610.
5. C.R. Ey, *Managing Content with Directory Servers*, diploma thesis, Dept. Business Info. Systems, Karlsruhe Univ. of Applied Sciences, 2000.
6. L. Ahmedi and G. Lausen, "Ontology-Based Querying of Linked XML Documents," *Proc. Semantic Web Workshop, 11th World Wide Web Conf.*, 2002; http://semanticweb2002.aifb.uni-karlsruhe.de/proceedings/research/ahmedi.pdf.
7. K.L.E. Law, "XML on LDAP Network Database," *Proc. IEEE Canadian Conf. Electrical and Computer Eng.* (CCECE '00), IEEE Press, 2000, pp. 469–473.
8. Isode, *Comparative Performance Benchmarking of Isode M-Vault R10.1*, white paper, Oct. 2003, www.isode.com/whitepapers/m-vault-benchmarking.htm.
9. E.J. Thornton, D.P. Mundy, and D.W. Chadwick, "A Comparative Performance Analysis of Seven LDAP Directories," *Proc. Conf. Terena Networking*, 2003; www.terena.nl/conferences/tnc2003/programme/papers/p1d1.pdf.
10. N. Klasen, *Directory Services for Linux, in Comparison with Novell NDS and Microsoft Active Directory*, master's thesis, Dept. Computer Science, RWTH Aachen Univ., 2001.
11. W. Dixon et al., *An Analysis of LDAP Performance Characteristics*, tech. report TR-2002GRC154, GE Global Research, 2002.
12. X. Wang et al., "Measurement and Analysis of LDAP Performance," *Proc. Int'l Conf. Sigmetrics,* ACM Press, 2000, pp 156–165.
13. S. Cluet, O. Kapitskaia, and D. Srivastava, "Using LDAP Directory Caches," *Proc. Symp. Principles of Database Systems* (PODS), ACM Press, 1999, pp. 273–284.
14. J. Hanck and J. Pingenot, *LDAP Product Research Results*, Computing and Network Services, Kansas State Univ., Apr. 2002; www.ksu.edu/cns/projects/Archived-Projects/KEAS-Phase-1/Documents/Research/LDAPreport.pdf.
15. P.J. Maron and G. Lausen, *HLCaches: An LDAP-Based Distributed Cache Technology for XML*, tech. report TR-147, Inst. for Computer Science, Univ. Freiburg, 2001.

Vassiliki Koutsonikola is a PhD student at Aristotle University of Thessaloniki, where she was a technical staff member of the Network Operation Center. Her research interests include directory services and network-based data organization. She received a BS in computer science from Aristotle University and an MS in information systems from the University of Macedonia, Greece. Contact her at vkoutson@csd.auth.gr.

Athena Vakali is an assistant professor at the Aristotle University of Thessaloniki. Her research interests include Web data management, data caching, content delivery, and Web data clustering. She received a PhD in disk subsystems performance from Aristotle University and an MS in computer science from Purdue University. She is a member of the IEEE Computer Society and the ACM. Contact her at avakali@csd.auth.gr.