

The X.500 Directory Standard: A Key Component of Identity Management

ERIK ANDERSEN



Erik Andersen is an independent consultant with the company Andersen's L-Service

New things generally fascinate people. This is especially true within the field of Information and Communications Technology (ICT). In science, it is good practice to make a literature search when starting a research project. This is a less common practice within ICT. Right now Identity Management (IdM) and Next Generation Network (NGN) are attracting great interest. Before developing directory like specifications within these areas, current works should be considered. It appears that the X.500 standard could play a major role here.

Introduction

X.500 is a directory standard and is therefore a specification for how information about entities (objects) is stored, retrieved, updated, deleted, managed and protected. X.500 has been developed and continuously extended for the last 23 years and this process continues. Many highly skilled people have contributed to the work over the years. The X.500 standard is an extensive specification consisting of ten documents (X.500, X.501, X.509, X.511, X.518, X.519, X.520, X.521, X.525 and X.530). X.509 is widely known as the basis for digital signatures, public-key infrastructure (PKI), etc. X.500 is an introductory document, but the term X.500 is here used as a synonym for the whole series. The X.500 standard is developed jointly between ITU-T and ISO/IEC. ISO/IEC labels the standard ISO/IEC 9594, Parts 1 to 10.

The Internet Engineering Task Force (IETF) has developed the Lightweight Directory Access Protocol (LDAP) for accessing X.500 directories. It is part of the X.500 family.

Identity Management (IdM) and Next Generation Network (NGN) use the term *entity* about things that need to be identified and named. X.500 uses the term *object*. This is the term used in the following.

You can find more information about X.500 and the standardization process with links to documents, etc. at <http://www.x500standard.com/>. It is a good site to visit.

X.500 is many things. It defines protocols, procedures, replication, etc., and it provides models for naming structure, information structure, protection, management, etc. Modelling is a major part of the X.500 standard.

Directory Information and Naming Model

An object, say, a human being, only has an identity if it has a name (and possibly other identifiable attributes) that uniquely identifies that object, at least within a specific context. An object may have several names, but no two objects should have the same name. X.500 has a naming model that assures unique naming if proper naming authorities are in place. The naming model recognises that objects are hierarchically organised by having a hierarchical naming structure. As an example, a person within an organisation may have a name consisting of a sequence of name components, like a country name component, an organisation name component, an organisational unit name component and finally a personal name component. We call such a name a *Distinguished Name*.

An *entry* within a directory represents an object and entries are viewed as having the same hierarchy as the corresponding objects. This hierarchy of entries is called the *Directory Information Tree* (DIT). A rather simple DIT is shown in Figure 1.

Information about an object is modelled as *attributes* within its entry. Attributes are of different types according to the kind of information.

Figure 2 illustrates the entry and attribute concepts. An attribute is of a specific type, for example, a telephone number. An attribute may have multiple values.

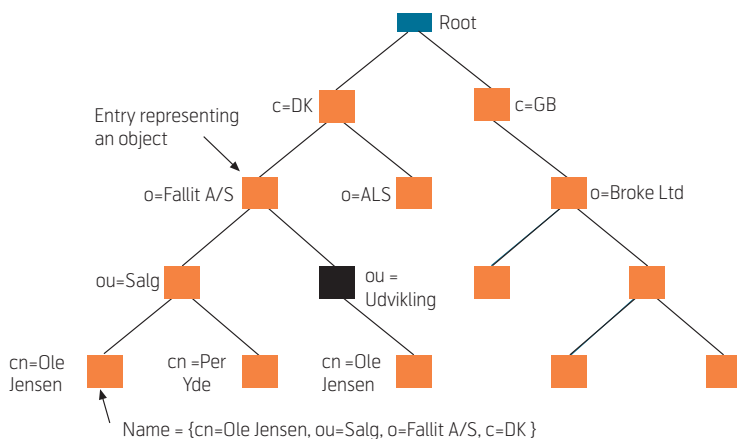


Figure 1 Small sample DIT

Recent editions of the X.500 standard allow an entry to contain so-called child entries, which allow attributes to be grouped according to different purposes. A person having different roles may have its attributes grouped according to those roles. Identity Management talks about different contexts for an entity. Child entries are the support for this concept.

Directory Schema

It is important that the information in a directory be orderly organised. An administrator can impose a directory *schema* that determines the types of objects for which the directory may hold entries, where such entries should be placed in the hierarchy, what kind of attribute types that may be associated with each type of object, etc.

Directory Components and Protocols

Let us get some terminology in place. A Directory System Agent (*DSA*) is a directory server holding directory information. Several interconnected DSAs may form a distributed directory. A Directory User Agent (*DUA*) is a client system supporting the user access to a directory by connecting to a DSA within the directory. X.500 defines several protocols. The Directory Access Protocol (*DAP*) supports communication between a DUA and a DSA. This protocol is used to invoke directory operation. The Directory System Protocol (*DSP*) is used between DSAs when an operation requires the involvement of more than one DSA. The Directory Information Shadowing Protocol (*DISP*) is used when replicating information from one DSA to another DSA. The Directory Operational Binding Management Protocol (*DOP*) is used to establish a common understanding between two DSA, for example, about knowledge of each other.

The Lightweight Directory Access Protocol (*LDAP*) was initially developed within the Internet Engineering Task Force (IETF) as an X.500 access protocol. Later it has also developed LDAP directory server specifications based on the X.500 model.

Directory Database

A DSA needs some kind of underlying database to support directory information handling. X.500 does not specify any particular database technique, as the type of database does not affect interoperability among different systems. How entries and attributes are reflected in the database is a pure implementation issue. However, an efficient database technology with an efficient data structure is essential for performance. Database performance is one of the more critical issues.

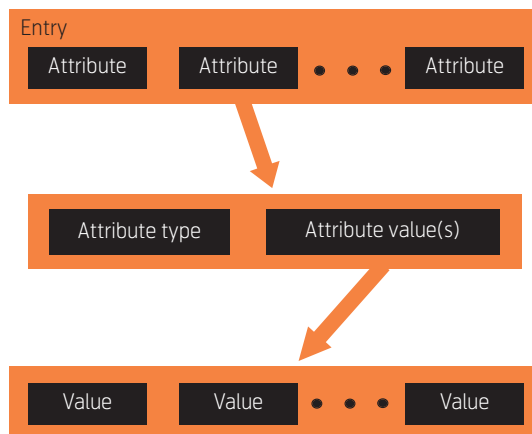


Figure 2 Entry and attribute structure

Distribution and Replication

From the very beginning, it was realised that the X.500 standard should allow *distribution* of information among different systems. Later, it also became evident that *replication* is a major requirement. Distribution is important, as it allows a service provider to maintain its own directory system and then interconnect that system with systems of other service providers to establish an integrated global system. This is illustrated in Figure 4. Users may utilise the combined service of all the interconnected DSAs.

Replication of directory information is illustrated in Figure 5. It provides availability and load sharing. Replication is also called *shadowing*. An administrator may decide that only a part of the information held by a DSA may be replicated to another DSA. Even individual entries and attributes may be excluded. The DSA receiving the shadow may in addition hold information of its own.

Replication and in particular distribution require elaborate procedure specifications to allow proper interworking among different vendor implementations to provide a seamless service.

The X.500 naming structure allows navigation within a distributed and replicated directory providing the

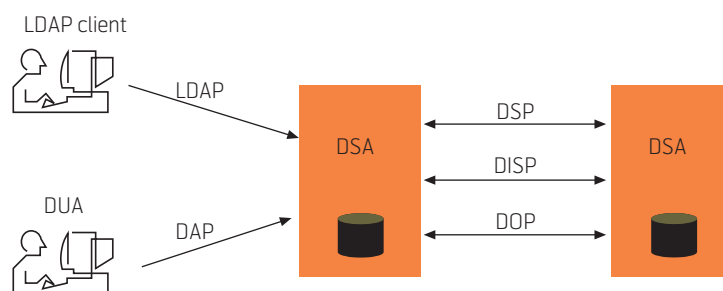


Figure 3 Distributed directory

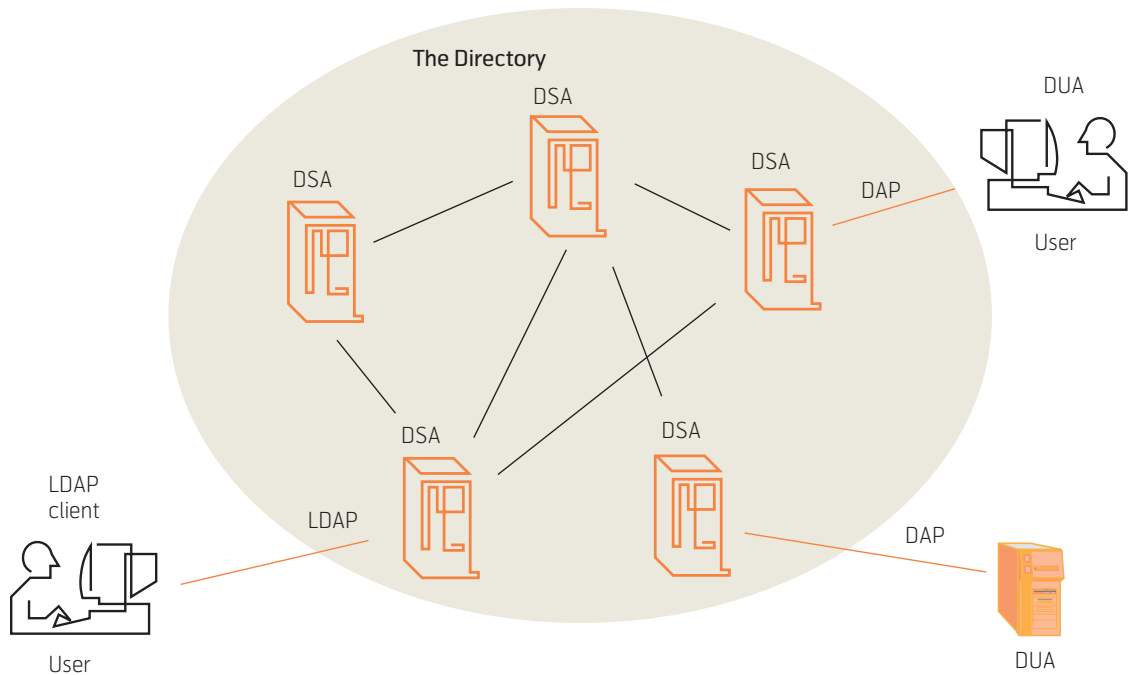


Figure 4 Distributed directory

different systems maintain knowledge of each other's network addresses and name domains. Procedures are defined for how such knowledge can be maintained. X.500 is the only directory standard that has specifications for distribution and replication.

Deployment of Directories

An X.500 directory may be employed whenever it is possible to establish a consistent naming structure. Network Directory Server is one such important deployment of an X.500 directory, and in more simple cases by an LDAP directory (Microsoft's Active Directory is an important example).

Directory Operations

A user may invoke directory *operations*. There are operation types for reading a named entry, searching, adding, updating and deleting entries, etc.

The search operation is the primary operation for retrieving information. It is the most complex operation, as it requires many entries to be checked and it requires matching between the information supplied by the DUA (search criteria) and the information within entries. Matching can be rather straightforward or it can be quite complicated. X.500 defines a range of matching rules from quite simple rules to very sophisticated rules.

Data Protection

Data protection is much in focus at the time being. It is a major part of Identity Management. Data protection is primarily a privacy issue, but may also be a way to protect the assets the data may represent. Almost from the beginning, data protection features have been an important part of the X.500 standard. X.500 is the only directory specification having these important features. A common data protection model is important for several reasons. It gives a consistent service to users, as a systems reaction is not depending on the brand of system. In a replicated multi-vendor environment, it is necessary that data protection information be replicated together with the data allowing the receiving system to protect such information.

Different levels of data protection are required for different types of accessing users. The authentication level of a user also affects the access right. X.500 provides for several levels of authentication, ranging from none, only name, name and password, name and encrypted password, and finally strong authentication based on digital signatures.

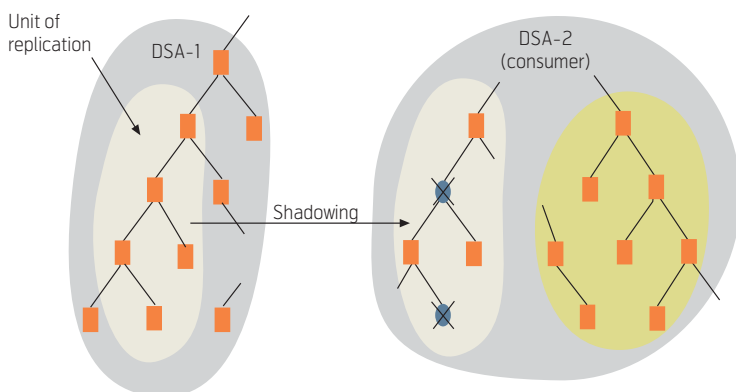


Figure 5 Replication

An elaborate access control has been part of the X.500 standard almost from the beginning. Access control is a question about who may do what or may not do what based on the level of authentication. Retrieving sensitive information or updating entries may require higher level of authentication than just retrieving non-sensitive information.

Access control is about the *right to know*. However need-to-know goes beyond access control, as the right-to-know should not be sufficient to retrieve information if a need-to-know is not established. For this reason, a service administration concept has been developed. A directory system without service administration could be compared to a buffet restaurant, while a directory system with service administration could be compared with an à la carte restaurant. An administrator may define a final set of distinct services, specify what kind of input is required and what type of output is relevant. Such services may be tailored differently for different user groups. In addition, the input restriction may be tailored to prevent devious searches and to prevent data trawling.

Distributed Management

An X.500 directory is designed to be distributed and to be owned by different service providers and organisations. The administration of the directory can be distributed by defining *autonomous administrative areas*. Such areas could follow DSA boundaries, but may not necessarily do so. Several organisations or organisational units could share a single DSA by dividing the DSA into multiple autonomous administrative areas. An autonomous administrative area may also span DSAs.

Within an autonomous administrative area, an administrator may define its own directory schema, its own access control, its own service administration, etc. An autonomous administrative area may be subdivided for different purposes, for example, to support fragmented access control. When service administration is applied, an administrative area acts as a mini directory. A search initiated outside the administrative area will not migrate into that area. Likewise, a search initiating within the area will not migrate out of the area.

X.509 and PKI

X.509 is a subject of its own. It also has a life outside the remainder of X.500. It is the basis for many other specifications, such as the Secure Socket Layer (SSL). X.509 is about message integrity, authentication and authorisation. X.500 protocols use the fea-

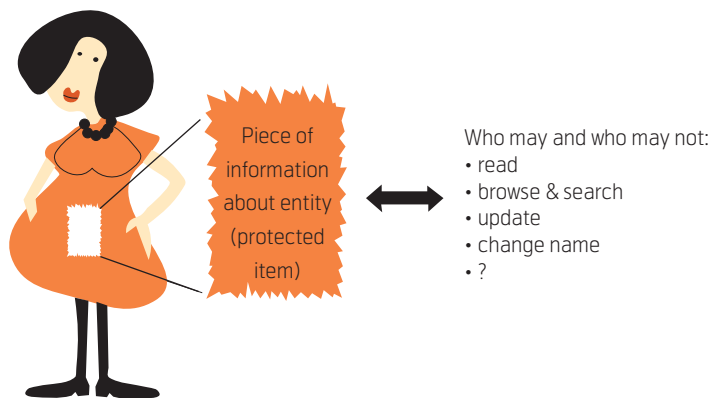


Figure 6 Access control

tures of X.509 to provide security functions not available in other types of directory specifications.

Authentication requires the use of hashing and asymmetric encryption. Hashing is the transformation of a message into a usually shorter fixed-length value string that represents the original string using some kind of algorithm. The algorithm must have the characteristic that it is virtually impossible to create a message resulting in a given hash value. A hash value will typically change considerably if just one bit is changed in the original message. This allows for message integrity. If the hash value is attached to the message when transmitted, the receiver can create its own hash value and compare it to the one attached. If different, the message has been changed and should be discarded.

Asymmetric encryption requires the use of an encryption key pair consisting of a private key and a corresponding public key. A message encrypted using one of these keys can only be decrypted using the other key. The owner of the key pair is in the position of the private key. Copies of the public key may be distributed to several parties. A message encrypted by a public key can only be decrypted by the holder of the private key. This can be used, for example, to encrypt e-mails sent to the holder of the private key.

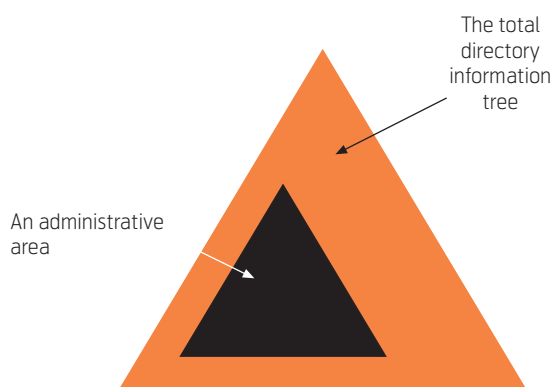


Figure 7 Administrative areas

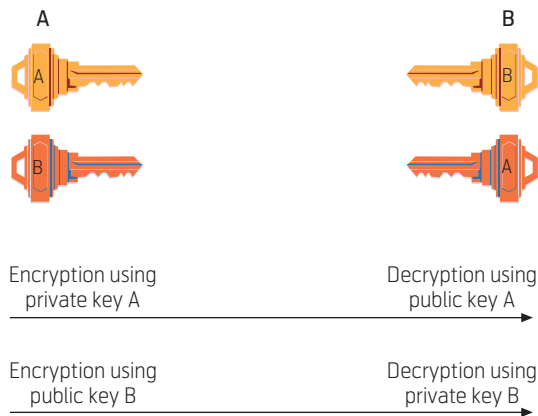


Figure 8 Asymmetric key encryption

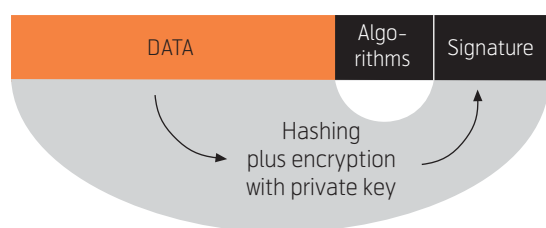


Figure 9 Digital signature

On the other hand, a message encrypted by the private key can be decrypted by anyone holding a copy of the public key. If decryption is possible, only the holder of the private key could have sent this message. This technique is used to create digital signatures as shown in Figure 9.

When a message is to be digitally signed, a hash of the message is created. The hash is encrypted using the private key and appended to the message as a digital signature. As indicated in Figure 9 also information about the used algorithms for hashing and encryption is attached. The receiver decrypts the signature using the public key. It then creates its own hash of the message. If the two hashes are identical, the receiver knows that the message has been transmitted unchanged and that the sender's identity is known with a high level of certainty. This gives an end-to-end security also in a distributed environment.

This sounds quite easy, but is actually quite complicated. The holder of a public key needs to be reasonably sure about the identity of the holder of the corresponding private key. The binding between a public key and the identity of the holder of the corresponding private key is provided in a so-called *public-key certificate*. For it to be trustworthy, a *Certification Authority (CA)* needs to verify the content of the public-key certificate and sign it. If the public key of the CA is known and the CA is trustworthy, we should be safe. However, there are further complications. The owner of a private key must protect it. If somebody else gets hold of the private key, it can be misused. If suspension of something like that has happened, the certificate must be revoked and put on a revocation list to be accessed. As a further complication, there may be many CAs out there. All this requires the establishment of a *Public Key Infrastructure (PKI)*. X.500 or LDAP directories are typically applied to establish a PKI.

An *attribute certificate* is a way to bind a privilege to a specific object. Such an attribute certificate is signed by an *Attribute Authority*. As for public-key certificates, attribute certificates require the establishment of a *Privilege Management Infrastructure (PMI)*.

X.500 products

Several vendors have implemented the X.500 standard. A list of X.500 vendors and short product descriptions may be found at <http://www.x500standard.com/>. For each vendor there is a link to more detailed product information. Many vendors refer to their systems as *Information Management systems*.

Summary and Conclusions

X.500 is a very comprehensive directory standard having excellent support for the requirements of our time. It is open-ended and extensible. It is under continuous development to cope with future challenges. When it comes to features, distribution, replication and security, it is the only game in town.

Erik Andersen holds a Master of Science in Electronics and Physics from the Danish Technical University. He has worked for IBM for 27 years as an expert on data communications and software architecture. He was IBM's representative in data communication standardisation work, including X.500. Since 1995 he has been the owner of Erik Andersen's L-Service consultancy company with continued participation in the X.500 standardisation work. He is now the main project editor for X.500 and the ITU-T Rapporteur for the Directory question. For the Association for the Directory Information Industry (EIDQ) he has developed several ITU-T Recommendations for the Directory Assistance area. He has been member and chair-person of European workshops on Directory.

email: era@x500.eu