Why Directories?

Фикрат Х.

Introduction

"The Directory is a collection of open systems which cooperate to hold a logical database of information about a set of objects in the real world." Directories have some properties that set them apart from relational databases

- Directories are organized in an object-oriented and hierarchical way. Information about a real-world object is stored in the entry that represents this object in

  The directory. To mirror the relationships of their respective objects, entries can be organized in a tree structure.
- Directory services provide a common schema for what can/must be stored for a certain class of objects and a standard access protocol, which greatly facilitates interoperability.
- Directories services offer a fine-grained security model. For example, access restrictions can be specified for one entry and then inherited by all entries below this entry in the directory tree.
- Directory services do not support transactions. Instead, they adopt a loose-consistency model. This allows for improved local availability of the service in a distributed environment.

The most common areas of application for directories are *white pages* and *yellow ages* services. In white-pages services such as a phone book, information about an object is accessed by the object's name, whereas yellow pages allow for information to earched or browsed by specifying categories. Due to their flexibility, directories are being used. In other applications as well, for example, as information repository for users and resources in computer networks.

The successful standardization efforts on directories have created a highly interoperable software landscape. Support for directory protocols has become a standard feature in many programs. Standard-compliant general-purpose directories are therefore often used to consolidate information germane to multiple applications into a singe repository. In an economic viability-analysis the impact of the introduction of a directory based white-pages and Single Sign-On service in six banks and insurance companies has been researched. This case study estimated that "the benefit of the directory would be 11 times that of its cost" which would lead to savings of about 23 Mil. DM

2. Directory Services Overview

Directories contain information about real world objects. Information concerning one Object is stored in an *entry*—the basic building block of a

directory .

An entry is a collection of name-value pairs called *attributes*. There can be more than One value associated with an attribute name—a person might have multiple first names, for example. However, some attributes may only have one value—a person definitely would only have one date of birth. Attributes of that kind are said to be *single-valued.*

The others are called *multi-valued.*

Each attribute has a *syntax*, which describes the type of information that can be stored in this attribute. This might be a character string, a number, a photo or some complex data type, e.g. a certificate for digital signatures.

The definition of an attribute alsolists its *matching rules*.

These rules govern how values of this attribute type should becompared. It is possible to specify rules for *equality* and *substring* matching as well as specifying rules that determine how values of this attribute should be ordered—the *ordering* matching rule.

Objects in the real world often show similarities and are therefore said to be of a particular kind. This grouping principle is met in the directory world by the notion of *object classes*. Object classes specify the attributes that an entry must or may contain. Attributes that an entry must contain are called *mandatory*, attributes that may be present *optional*. It is possible to derive an object class from another one. This allows the design of an object class hierarchy. Subclasses inherit all mandatory and optional attributes of their superior class and can incorporate additional ones. It is also possible to declare an inherited optional attribute as mandatory for the subclass.

Three types of object classes exist:*abstract*, *structural* and *auxiliary*. Object classes of the abstract type are used to form the upper levels of an object class hierarchy.

Entries can only be added to the directory if they meet the requirements of at least one structural object class. Structural object classes reflect the principal fabric of an object. Common structural object classes are, e.g. "person" or "organization". The object classes to which an entry conforms are listed in its "objectClass" attribute. The "objectClass" attribute is introduced as a mandatory attribute by the "top" object class.

"top" forms the root of the object class hierarchy. All other object classes are directly or indirectly derived from it. This ensures that every entry has at least one object class.

Sometimes the need arises to store additional data that is not strictly tied to the structure of an object or which may not be present for all objects of a particular class. This additional data can be stored in attributes, which are governed by auxiliary object classes.

With auxiliary classes, attributes can be introcuded as a mandatory requirement to a subset of entries that have the same structural class. An auxiliary class can also be added to entries that have a different structural classes, e.g. both a person and an organization could have a homepage, which would be stored in the common "labeledUri"attribute.

When more applications with partly different needs employ the directory as their data storage, the advantage of introducing auxiliary classes over deriving from structural.

3. A History of Directory Standards

3.1 Early Electronic Directories
3.2 Special Purpose Directories
3.4 Lightweight X.500 – LDAP
3.5 LDAPv3
3.6 Text-based Standards

Access Control and Security Layers

Not all directories provide a read-only white-pages service that should be publicly available. Access might be limited to subscribers of the service or only available on a pay-per-view basis, in which case appropriate accounting mechanisms would have to be in place. The data contained in the directory could be of a sensitive nature so that access by non-authorized parties would not be allowed.

The security model for directory services relies on the fact that the protocols, which are used to access the directory, are connection-oriented. It is assumed, that peer entities do not change after a connection has been established. To ensure this in an insecure environment, *security layers* have to be deployed. All operations, which are initiated over a connection, can rely on data that was exchanged earlier. This is used to implement
access control mechanisms. At the beginning of a connection the client's identity is established during the *authentication* stage. For each subsequent operation requested by the client, the server checks whether the client is *authorized* to do so.

- One-way-hash-functions
- Symmetric cryptosystems
- Asymmetric cryptosystems
- Authentication Methods
- Anonymous
- Plain Text Passwords
- Challenge-Response Mechanisms
- Kerberos
- X.509
- Authorization
- Security Layers
- Authentication Methods for LDAP

Conclusions and Future Work

Directory services are designed and developed for managing different aspects of modern computer networks. In this thesis, the use of directory services as central authentication backend and as central repository for provisioning email applications has been presented and analyzed with regard to its resource requirements. The benchmarks that have been conducted in the frame of this thesis show that current directory servers running on commonly available hardware can easily keep up with the performance requirements of a typical academic institution .

As the performance requirements are met by a number of products, other criteria

become more important in the decision making process for a particular product. If costs are a primary concern, products that do not incur any license costs are likely to be chosen.
This would include OpenLDAP and—when released—SecureWay Directory. Organizations, which own hardware from Sun Microsystems, could consider the Netscape Directory Server instead. The free binary license of Solaris 8 for the SPARC platform contains a 200,000-entry production license. If the intended use of the directory goes beyond user management, flexibility and full compliance to the standards becomes an issue. While Active Directory in its current version does a good job in fulfilling its primary task as NOS specific directory, it is not as capable and compliant as the standalone directory servers. As a result, a migration of the AMBIX directory, this offers a white-pages service for the world academic community, to Active Directory as provided in Windows 2000 is not feasible.Many world universities provide heterogeneous Windows/Linux computer pools to their students. Networks that consist of client PCs running Windows are generally connected to either a Windows NT/2000 or a NetWare server. In such an environment, integrating the Linux client PCs with the native Windows—pending an upgrade to Windows 2000, if Windows NT is still used—or NetWare directory services seems feasible1. This allows for investments, which have been put into the infrastructure in. If the Linux clients in a heterogeneous Windows/Linux network should not to access Active Directory directly, but rather authenticate to a Linux based directory server, passwords need to be synchronized to provide users with a unified login. A prototype software that accomplishes this task has been developed in the frame of this thesis. It is comprised of two modules—one for each direction of synchronization
Future work might concentrate on adapting Nss_ldap to Active Directory. Nss_ldap is a name service module, which allows the use of LDAP instead of a local /etc/password file. This would allow Linux clients to retrieve user information besides passwords directly from Active Directory and would eliminate the need to maintain a second directory server on Linux. The reverse approach, i.e. integrating a Unix Kerberos distribution with OpenLDAP, would also offer interesting possibilities.

Additionally, research could be carried out into the area of certificate based authenti cation mechanisms. As part of this work, the SASL EXTERNAL mechanism could be implemented in OpenLDAP. To make effective use of this mechanism, ways to associate a certificate with a directory entry need to be specified and implemented—including a check of the certificate's validity. One way of establishing this relationship would be to map the subject's name form the certificate to the distinguished name of an
entry in the directory. Another possibility would be to store data that uniquely identifies a certificate in the associated entry. As part of this analysis, the role of the directory service as part of a Public Key Infrastructure could to be investigated.

Summary

In this research I want to explain that what is a directory in real world and in computer and if need directory which operation system is good for using directories?