IDML: Identity Management Language

COMS W4115

Programming Languages and Translators

Columbia University

Department of Computer Science

Ashraf Motiwala

2.5.2005

Introduction

The purpose of this language is to provide a simple way to perform common Identity Management (IdM) related tasks. Examples of these tasks include provisioning users to a data repository, deprovisioning users from a data repository, changing a user's passwords, and synchronizing user data from one repository to its corresponding entry in another repository.

Currently, in order to perform the tasks above, much code and knowledge is needed regarding the underlying data repository and how to access it. With IDML, significantly fewer lines of code and far less knowledge about the underlying data repository will be required by the developer. This will allow developers to quickly create scripts to do common IdM tasks instead of buying expensive third party tools that often provide far more functionality than may be needed.

Background

Identity Management is quite simply, the management of digital identities. A digital identity is the representation of human identity that is used in a distributed network and primarily consists of two parts: who one is and the credentials one holds. Regardless of the legal and technical implications of a digital identity, the simplest digital identity consists of a username and password.

Managing digital identities can quickly become a cumbersome task in corporate environments. Quite a bit of administration is required with the hiring and termination of employees, their constant changing of roles within the organization, the existence multiple identities for the same user in various repositories, and the task of ensuring users identities are not lost or stolen.

Experts have generally divided IdM into 4 components:

- **Authentication**: allows users to identify themselves when accessing resources
- **Access Management**: policies and rules to manage access to resource
- **User Management**: includes password management, self-service mechanisms, provisioning, delegated administration tools
- **Data Synchronization**: includes Directory Services and metadirectory processes

Purpose

Currently, there exist only bulky GUI tools to accomplish the aforementioned tasks. Unfortunately, most small to mid size companies fail to manage digital identities well because of the high costs of these tools on the one hand, and on the other hand - most system administrators are not equipped to write the necessary code to accomplish the tasks above.

IDML will abstract the unneeded details, thereby simplifying the task of managing identities. This will allow the developer to provision a user to a data repository in a few lines of code that will abstract the details of the data repository.

Limitations

Due to the vastness of IdM, I have decided to implement IDML with the following limitations:

- The data repository will be limited to LDAP servers
- Users can only be of objectclass: `inetorgperson`
- I will only be addressing one of the four components of IdM, namely User Management

Language Details

- Data Types: due to the specific domain of the language, the data types will be User, Repository, File, String, Boolean
- Operators: existsin, !, =, >, <

- Control Flow: An if statement can be used to perform conditional checks. Read loops can be used read a flat file line by line.
- Comments: will be lines starting with //

Sample Code

```
REPOSITORY::Repository1,(“servername”,”bind_dn”,”bind_pw”,”people_ou”);
read(<filein>,comma) { //read from comma separated file
      USER::User1 from <filein> as (uid,firstname,lastname,emplnum);
      If(User1 !existin Repository1)
            Add: User, Repository1;
}
```

These few lines of code would read line by line through a comma-separated file of user entries, check if the user already exists in the repository or not, and if the user does not exist in the defined repository, the user gets added.