# BOB and LDAP

Version: 0.2

Date: October 26, 2003

39355 California Street, Suite 307 • Fremont, CA 94538 Tel: 510.744.4007 • Fax: 510.608.5917 • www.gamingstandards.com

DRAFT

39355 California Street, Suite 307 • Fremont, CA 94538
Tel: 510.744.4007 • Fax: 510.608.5917 • www.gamingstandards.com

# Table of Contents

39355 California Street, Suite 307 • Fremont, CA 94538
Tel: 510.744.4007 • Fax: 510.608.5917 • www.gamingstandards.com

# 1 Introduction

This paper covers how LDAP fits into the BOB standard.

LDAP will be used to provide the following functionality:

1. Centralized Configuration Management
2. SSL Session Management

This paper will define the LDAP schema to support this functionality and then will describe how LDAP will be used to deliver on this functionality.

# 2 Justification

The BOB standard consists of two major components:

1. Flow of run time information between EGMs and Host Systems.
2. The boot process for EGMs.

LDAP addresses the second component.

The goal of including LDAP in the BOB standard is borne out of the following requirements:

1. In order to foster innovation the issue of configuration information, in particular the configuration information for EGMs, must be addressed.
2. Therefore the BOB standard must address the issue of configuration information storage.
3. The BOB standard must also address the issue of access (both read and write) of configuration information.
4. The BOB standard must propose a solution that is **vendor neutral** in regards to configuration information.
5. And the solution must be:
    a. Highly scaleable
    b. Platform neutral
    c. Inherently secure
    d. Redundant
    e. Easy for operators to implement and operate

Therefore the BOB committee must select a technology that meets these requirements. There are two courses here:

1. Invent something new that meets all of these requirements.
2. Use an existing, open technology that meets these requirements.

It is our recommendation that the BOB committee select LDAP.

The **benefits** of going with LDAP are:

1. Open standard for storage
2. Open standard for data access (the LDAP API)
3. Security control
4. Scaleable
5. Redundancy built in

The **draw backs** of LDAP are:

1. New API for the vendors to learn and implement.
2. New server for the operator to deploy and operate.
3. New server for the vendors to learn.
4. Some change to the economic proposition for the first vendor to sell the LDAP server to the operator.

# 3 Schema

An LDAP Schema defines what logical objects exist in the LDAP server. Schemas from multiple vendors can (usually) exist in the same tree.

At this point in time BOB committee has not been able to nail down the LDAP Schema. But we'll start with what we do know.

Right now we know we need three schema objects:

1. GSAEGM
2. GSAHostSystem
3. GSABackOffSystem

We will now talk about these objects.

## 3.1 Schema Object: GSAEGM

This LDAP entry defines an EGM.

| Attribute | Name | Type | Description |
|---|---|---|---|
| gmid | EGM ID | String | A unique identifier. |
| | | | |
| | | | |
| | | | |

## 3.2 Schema Object: GSAHostSystem

This LDAP entry defines a Host System. There is nothing yet in the BOB standard specifically addressing Host Systems. But future versions will. This object is a place holder for that work. For right now we won't define any attributes.

| Attribute | Name | Type | Description |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

## 3.3 Schema Object: GSAEGMReference

This LDAP entry defines a reference to an EGM entry. That is, an object of this type points to a **GSAEGM** somewhere else in the LDAP tree.

| Attribute | Name | Type | Description |
|---|---|---|---|
| Gmid | EGM ID | String | A unique identifier |
| Ref | Reference DN | DN | Pointer to where the EGM exists in the LDAP tree. |
| | | | |
| | | | |

# 4 LDAP Tree

The organization of the tree is completely deployment dependent. The standard must address this fact.

The solution is for the BOB standard to define a Reference Model. A Reference Model is an architecture for an LDAP server in which the client does a double query. The first query is to a known location in the tree. The response contains a pointer to the real data. Using this response the client issues a second query which gets the real information.

This is also sometimes known as a Referral Model. Referral Models use the Referral feature of the LDAPv2 standard. However, Referrals are not well implemented across all LDAP Servers and all LDAP API implementations. So at this point it makes sense to have the client do a manual double query operation.

39355 California Street, Suite 307 • Fremont, CA 94538
Tel: 510.744.4007 • Fax: 510.608.5917 • www.gamingstandards.com

Having said that, we can diagram a possible LDAP Server implementation. In this example we have two roots: **o=CasinoRoot** and **o=GSARoot**.

- o=CasinoRoot
  - o=EGMs
    - cn=EGM 1
      - gmid=GSA.com@123456
      - object: top
      - object: GSAEGM
    - cn=EGM 2
      - gmid=GSA.com@876543
      - object: top
      - object: GSAEGM
    - cn=EGM 3
      - gmid=GSA.com@999999
      - object: top
      - object: GSAEGM
  - o=Host Systems
    - Host System 1
      - object: top
      - object: GSAHostSystem
    - Host System 2
      - object: top
      - object: GSAHostSystem
    - Host System 3
      - object: top
      - object: GSAHostSystem
- o=GSARoot
  - EGM Group
    - Entry 1
      - gmid=GSA.com@123456
      - ref=cn=EGM 1, o=EGMs, o=CasinoRoot
      - object: top
      - object: GSAEGMRef

- Entry 2
  - gmid=GSA.com@876543
  - ref=cn=EGM 2, o=EGMs, o=CasinoRoot
  - object: top
  - object: GSAEGMRef
- Entry 3
  - gmid=GSA.com@999999
  - ref=cn=EGM 3, o=EGMs, o=CasinoRoot
  - object: top
  - object: GSAEGMRef

# 5 Searching during Bootstrap

The principal operation performed by the LDAP Server, at least from the perspective of the EGM, is to locate the EGM entry in the LDAP Server. That is, when an EGM boots up it must find its entry in the LDAP Server.

When you search the LDAP tree for an EGM entry you need two pieces of information:

- LDAP Search Root
- LDAP Search Statement

The BOB standard should dictate that every casino LDAP Server have a branch that starts with **o=GSARoot**. That is, for the casino deployment to be BOB Compliant the local LDAP Server must have an LDAP root that starts with **o=GSARoot**.

Once you have the tree standard established, you will do something like this:

(gmid=GSA.com@837853535)

The response is then an LDAP entry which contains the attribute **ref** which points to where the actual EGM record resides. So the EGM will fetch a **GSAEGMRef** object and then use the **dn** attribute in there to find the actual **GSAEGM** entry.

**Justification**: many people will ask why we do it this way. It is for a couple of reasons:

1. It is impossible for the GSA to impose a shape on the LDAP server of the casino operator. Maybe not today but in the future the casino operator will use LDAP to keep all of their configuration information, including player information. All vendors that store information in the local LDAP server have to coexist. The GSA must do so as well.
2. The flexibility imposed by this scheme relieves the operator from having a brittle LDAP server. A reference Model is much more flexible. The target data can be moved anywhere in the tree and the EGM bootstrap sequence does not change.