RFC2251
Lightweight Directory Access Protocol (v3)
分散システム特論 レポート

情報工学専攻 修士1年 田中宏輔

1999年2月8日

1 要約

我々の身の回りには、図書館のカード式目録や電話帳などさまざまなディレクトリ (Directory) があり、探したい情報を見つける手段を提供している。あるエンティティ(Entity: 人、サーバ、各種資源) に対する情報をひとまとめにし、Tree 構造で管理するデータベースをディレクトリと呼ぶ。ディレクトリサーバ (Directory Server) はこのディレクトリへのプロトコル要求を処理するサーバである。

LDAP(Lightweight Directory Access Protocol) は、インターネットのディレクトリプロトコルである。ディレクトリプロトコルは他にもあるが、LDAP はインターネット標準として良く使われている。LDAP は従来の X.500 ディレクトリアクセスプロトコルに対する軽いフロントエンドとして開発された。LDAP は、インターネット上の実装や開発には不向きであるという X.500 の欠点を補うものとして提案された。

LDAP の特徴は以下の通りである。

- プロトコル要素は、TCP やその他のトランスポート層上で直接伝えられる。TCP では 389 番ポートを使用する。
- 多くのプロトコルデータ要素、例えば識別名などは簡単な文字列形式で符号化される。そのため、X.500 の複雑に構造化された表現よりも、ずっと簡単に処理が行なえる。
- ネットワーク上にデータを流す際、X.500 のものと同じ BER 符号化規則の簡易版を使用する。

- LDAP では、X.500 でほとんど利用されない機能や余分なオペレーションを取り除き、X.500 の持つ機能を簡素化した。

2 LDAP のモデル

2.1 X.500 との関係

この RFC では、LDAP は X.500 のアクセス機構として X.500 の見地から定義されている。LDAP サーバは、ITU(International Telecommunications Union) 勧告の X.500 シリーズ (1993) に従ってサービスを提供しなければならない。しかし、X.500 プロトコルを必ずしも使用する必要はない。たとえば、X.500 のデータやサービスモデルが、LDAP で使われている時に LDAP のインタフェースの規則を破らない限り、LDAP をいくつものディレクトリシステムの上にマップすることができる。

2.2 プロトコルモデル

このプロトコルで採用された一般的なモデルは、クライアントの一つがサーバに対してプロトコルオペレーションを実行するというものである。クライアントはプロトコル要求をサーバに送信し、それを受けたサーバは、ディレクトリで必要なオペレーションを実行することに関して責任を持つことになる。つまり、サーバはプロトコル要求を自分で処理しなくてはならない。クライアントへは最終的な結果やエラーを返す。ディレクトリの利用にかかるコストを軽減するという目標の他に、クライアントの複雑さを最小限にとどめることも目標としている。

2.3 エントリと識別名

ディレクトリサーバの管理する情報の単位をエントリ (Entry) と呼ぶ。エントリの名前は識別名 (DN: Distinguished Name) で表す。以下に識別名の例を示す。

cn=Kousuke, ou=csce, o=Kyushu Univ., c=JP

識別名はいくつかの部分からなる。識別名の最初の部分 (この例では "cn=Kousuke" に相当) を 相対 DN (Relative DN) と呼び、2 番目以降の部分 ("ou=csce, o=Kyushu Univ., c=JP" に相当) を 親 DN(Parent DN) と呼ぶ。相対 DN と親 DN の関係を利用して、エントリは Tree 構造を形成している。

2.4　エントリの属性

すべてのエントリは何らかの情報を持っており、それらを属性 (attribute) と値 (value) という組の集合で表現する。図 1 はエントリの持つ属性と値の組の例である。

```
[attribute 名]:[value]

dn:cn=Kousuke, ou=csce, o=Kyushu Univ., c=JP
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
givenname: Kousuke
sn: Tanaka
cn: Kousuke
cn: Kousuke Tanaka
uid: kousuke
mail: kousuke@csce.kyushu-u.ac.jp
userpassword: secret
```

図 1: エントリの属性例

上の例における “objectclass” や “cn” のように、一つの属性に対しいくつかの値を定義することもできる。

各属性は、1 つのタイプ (type) と 1 つ以上の値を持っている。エントリとその属性や値との関係を表したものを図 2 に示す。

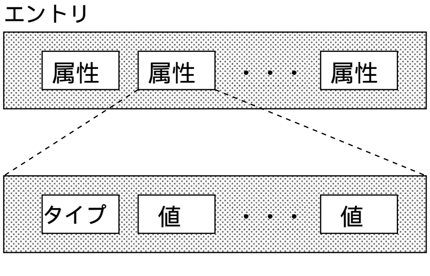

図 2: エントリの構造

各エントリは “objectclass” という特殊な属性を持っており、これはエントリごとにどのような属性を持つことができるかを規定する schema

にあたる。LDAPv2 まではこの objectclass で規定されていない属性を使うことは基本的にできなかったが、LDAPv3 では、objectclass 属性が extensibleObject を含んでいる時には、schema 規則に縛られずに他のどんな属性でもそのエントリ内で使うことができる。

3 LDAP のオペレーション

この RFC では LDAP のオペレーションとして以下の 9 つを定義している。それぞれを簡単に説明する。図 3 はディレクトリサーバとクライアント間で実行される一般的なオペレーションの様子である。

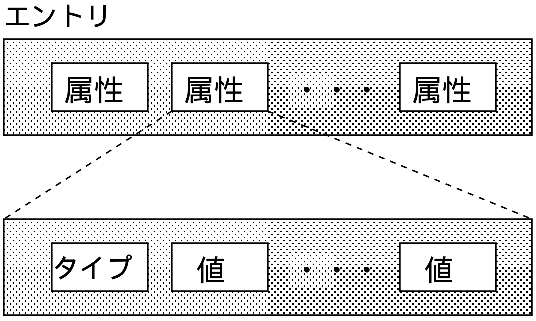

図 3: S/C 間で実行される一般的なオペレーション

3.1 Bind オペレーション

- クライアントがディレクトリサーバへの接続を行なう。サーバはクライアントの認証を行なう。
- バインド要求を受けたサーバは、クライアントに対しセッションを開始するためのステータスを返す。

3.1.1 LDAP の認証方式

クライアントの認証を行なう際に必要な情報として、simple, sasl という 2 つのオプションのうちどちらかを選択する。simple は平文 (cleartext) パスワードだけからなる認証フィールドの内容を使った、最小限の認証機能を提供する。しかし simple では使用するネットワークが暗号化されておらず安全でない場合は危険である。SSL(Secure Sockets Layer) プロ

トコルは、LDAP 上で渡されるパスワードを保護するために使われる。LDAPv3 では、SSL やその他のセキュリティオプションをサポートするために、SASL(Simple Authentication and Security Layer) と呼ばれる、拡張性のある認証フレームワークのサポートが追加されている。

3.2 Unbind オペレーション

- クライアントとディレクトリサーバとの接続を切断する。
- アンバインド要求に対する応答は定義されていない。

3.3 Search オペレーション

- クライアントの代わりにサーバに対しエントリの検索を要求するもので、検索条件に一致するすべてのエントリを取得する。
- サーバは検索要求を受けとると、成功か否かの詳細と取得したエントリからなる検索応答を、クライアントに返す。

3.4 Modify オペレーション

- クライアントの代わりにサーバにエントリの変更を要求するもので、エントリの持つ属性とその値を、追加 / 削除 / 置換する。たとえば追加を行なう場合、必要があれば属性を生成しながら、与えられた属性に、リストされた値を追加する。
- サーバは、修正が成功したか否か、を含む修正応答を、クライアントに返す。

3.5 Add オペレーション

- ディレクトリにエントリの追加を要求する。
- サーバは追加しようとした結果を追加応答としてクライアントに返す。

3.6 Delete オペレーション

- ディレクトリからエントリの削除を要求する。
- このオペレーションでは、ディレクトリの終端オブジェクトしか削除できない。

3.7 Modify DN オペレーション

- ディレクトリのエントリの識別名の変更を要求する。
- このオペレーションも、変更できるのは終端オブジェクトのみである。

3.8 Compare オペレーション

- 与えられた宣言とディレクトリのエントリを、クライアントが比較するためのオペレーションである。
- 比較の結果は比較応答としてクライアントに返される。

3.9 Abandon オペレーション

- 未処理オペレーションを破棄するようにサーバに要求する。
- 破棄要求をしたクライアントは、そのオペレーションを破棄されたものと考えてよい。