LDAP Linux HOWTO

Luiz Ernesto Pinheiro Malere, malere@yahoo.com v1.04, 2001년 2월 28일

서정룡, s_ryong@hotmail.com 2001년 3월 7일

이 문서는 LDAP(Lightweight Directory Access Protocol) 서버 설치, 설정, 구동 및 유지 보수에 대한 정보를 설명한다. LDAP 데이타베이스 생성, 데이타베이스내의 정보 갱신과 삭제, 로우밍 엑세스 수행 및 넷스케이프 주소록 사용 방법에 대한 세부사항도 설명한다. 이 문서는 대부분이 미시간 대학의 정보 페이지와 OpenLDAP 관리자 지침에 근거한다.

목 차

제 1 절 소개

이 문서는 LDAP 디렉토리 서버를 설치하여 사용하는데 있어 도움을 주기 위한 것으로 LDAP 서버의 설치, 설정, 구동 및 유지 보수 방법을 배운다. 그 후 LDAP 클라이언트와 유틸리티를 사용하여 디렉토리내에 정보를 저장, 검색 및 갱신하는 방법을 배운다. LDAP 디렉토리 서버 데몬, *slapd*은 여러가지 많은 UNIX 플랫폼에서 작동된다.

LDAP 서버들간의 사본을 다루는 다른 데몬, *slurpd*이 있는데, 당분간은 염두에 둘 필요가 없다. 이 문서에서는 사본없이, 즉 slurpd 데몬없이, 단지 로컬 도메인에 대해 디렉토리 서비스를 제공하는 slapd 데몬만을 작동한다.

이 문서는 간단한 서버 설정을 다루는데 처음 시작을 위해 유익하며 추후 필요시 다른 설정으로 갱신하는 것은 쉽다. 이 문서의 정보는 LDAP 프로토콜 사용에 대한 정확한 초기화를 설명한다. 아마도 이 문서를 읽은 후에는 리눅스 서버 능력의 확장뿐만 아니라 기존의 사용가능한 C, C++ 과 자바 개발 도구(JDKs)를 사용하여 자신의 클라이언트를 작성할 수 있을 것이다.

1.1 LDAP란 무엇인가?

LDAP는 디렉토리 서비스 엑세스를 위한 클라이언트-서버 프로토콜로 초기에는 X.500의 전위(front-end)로 사용되었으나 스탠드 얼론(stand-alone) 및 다른 종류의 디렉토리 서버들과 함께 사용될 수 있다.

1.2 디렉토리 서비스란 무엇인가?

디렉토리는 데이타베이스와 유사하지만 더욱 설명적이고 속성에 기초한 정보를 갖고 있다. 디렉토리내의 정보는 일반적으로 쓰기보다는 읽기 작업에 더욱 빈번히 이용된다. 따라서, 디렉토리는 통상적으로 정규 데이타베이스들이 다량의 복잡한(high-volume complex) 갱신을 위해 사용하는 복잡한 처리(transaction) 또는 롤백 계획(프로그램에 따라 바로 전의 체크포인트로 돌아가기, roll-back)을 수행하지 않는다. 디렉토리는 일반적으로, 적어도 허용된다면, 전부 갱신되거나 아무 것도 변경되지 않는다.

디렉토리는 다량의 순람(lookup) 또는 검색 연산에 대해 빠르게 응답하기 위해 조정된다. 디렉토리는 응답시간을 감소시키는 반면 가용성과 신뢰성을 증대시키기 위해 정보를 널리 복제할 수 있다. 디렉토리 정보가 복제될 때 복제된 정보들 사이의 일시적인 불일치는 결국 일치된다면 무방할 것이다.

디렉토리 서비스를 제공하는 많은 다른 방법이 있다. 각각의 방법들은 다양한 종류의 정보가 디렉토리에 저장되는 것을 허용하며, 그러한 정보가 어떻게 참조, 질의 및 갱신될 수 있는지 또는 허가받지 않은 엑세스로부터 어떻게 보호되는지 등에 대한 여러가지 요건을 둔다. 어떤 디렉토리 서비스는 제한된 상황(예를 들면 단독 머신에서 finger 서비스)에 대해서 서비스를 제공하는 지역적인 반면 다른 서비스는 더욱 넓은 상황에 대해서 서비스를 제공하는 전체적이다.

1.3 LDAP는 어떻게 작동하는가?

LDAP 디렉토리 서비스는 클라이언트-서버 모델에 기초하는데, 하나 또는 그 이상의 LDAP 서버들이 LDAP 디렉토리 트리 또는 백엔드(backend) 데이타베이스를 구성하는 자료를 갖고 있다. LDAP 클라이언트는 LDAP 서버에 연결해 질의하며, 서버는 답 또는 클라이언트가 더 많은 정보를 얻을 수 있는 포인터(일반적으로 다른 LDAP서버)를 갖고 응답한다. 클라이언트는 어떤 LDAP 서버에 연결하던지 간에 동일한 디렉토리 구조를 본다; 한 LDAP 서버에 보내지는 이름은 다른 LDAP에 있을 수 있는 동일한 엔트리를 참조하며 이것이 LDAP와 같은 전체적인 디렉토리 서비스의 중요한 특징이다.

1.4 LDAP 백엔드, 객체와 속성

Slapd는 세개의 다른 백엔드 데이타베이스 중에서 하나를 선택할 수 있다; 고성능 디스크에 기초한 데이타베이스 LDBM, 임의의 유닉스 명령어 또는 쉘 스크립트에 대한 데이타베이스 인터페이스 SHELL, 간단한 패스워드 파일 데이타베이스인 PASSWD.

이 문서에서는 LDBM 데이타베이스가 선택된다고 가정한다.

LDBM 데이타베이스는 데이타베이스내의 각 엔트리에 콤팩트한 4 바이트의 고유한 식별자를 할당함으로써 작동한다. 데이타베이스는 엔트리 고유 식별자(entry's unique identifier, EID)를 엔트리 자체를 표현하는 텍스트로 사상해주는 id2entry라는 하나의 주 인덱스 파일로 구성되는데 다른 인덱스 파일들도 마찬가지로 유지된다.

LDAP에 기초한 디렉토리 서버들간의 디렉토리 정보를 import 및 export 하거나 디렉토리에 적용되어지려고 하는 일련의 변경들을 기술하기 위해 LDIF(LDAP Data Interchange Format) 파일 포맷이 일반적으로 사용된다. LDIF 파일은 엔트리의 객체지향 계층 구조내에 정보를 저장하는데 LDAP 소프트웨어 패키지에는 LDIF 파일을 LDBM 포맷으로 변화해주는 유틸리티가 있다.

일반적인 LDIF 파일은 다음처럼 보인다:

```
dn: o=TUDelft, c=NL
```

```
o: TUDelft
objectclass: organization
dn: cn=Luiz Malere, o=TUDelft, c=NL
cn: Luiz Malere
sn: Malere
mail: malere@yahoo.com
objectclass: person
```

위에서 보듯이 각 엔트리는 구별되는 이름(DN, distinguished name)에 의해 고유하게 식별되며 DN은 엔트리의 이름과 엔트리를 통해 디렉토리 계층 구조의 최상위를 찾는 이름들의 경로로 구성된다.

LDAP에서 객체 클래스는 엔트리를 정의하는데 사용될 수 있는 속성 집합을 정의한다. LDAP 표준은 다음과 같은 기본 형태의 객체 클래스를 제공한다:

- 개별 객체 또는 객체 그룹의 정렬되지 않은 리스트를 포함하고 있는 디렉토리내 그룹
- 국가명과 설명(description)과 같은 장소
- 디렉토리내 조직
- 디렉토리내 사람

엔트리는 하나 이상의 객체 클래스에 속할 수 있는데, 예를 들면 사람에 대한 엔트리는 *person* 객체 클래스에 의해 정의되지만 inetOrgPerson, groupOfNames와 Organization 객체 클래스내의 속성에 의해서도 정의될 수 있다. 서버의 객체 클래스 구조(즉 스키마, schema)는 특별한 엔트리에 대해 필수 및 허용 속성들의 총 리스트를 결정한다.

디렉토리 자료는 속성-값 쌍으로 표현되는데 정보의 어떤 특정 부분은 설명적 속성과 연결된다.

예를 들면, commonName 또는 cn 속성은 사람의 이름을 저장하는데 사용된다. Jonas Salk라는 이름을 갖는 사람은 디렉토리내에서 다음과 같이 표현될 수 있다.

```
cn: Jonas Salk
```

디렉토리내에 들어가는 각 사람은 *person* 객체 클래스내의 속성 집합에 의해 정의되는데 이 엔트리를 정의하기 위해 사용하는 다른 속성들은 다음을 포함할 수 있다.

```
givenname: Jonas
surname: Salk
mail: jonass@airius.com
```

필수 속성들은 객체 클래스를 사용하는 엔트리에 존재해야 하는 속성들을 포함하는데 모든 엔트리들은 엔트리가 속하는 객체 클래스가 리스트되어 있는 objectClass 속성을 필요로 한다.

허용 속성들은 객체 클래스를 사용하는 엔트리에 존재할 수 있는 속성들을 포함하는데 예를 들면 person 객체 클래스에서 cn 과 sn 속성은 필수지만 description, telephoneNumber, seeAlso와 userpassword 속성들은 필수가 아닌 허용된 것들이다.

각 속성은 속성에 의해 제공되는 정보 유형을 기술하는 해당 구문(syntax) 정의를 갖는다:

- bin binary (바이너리)
- ces case exact string (case는 비교시 일치해야 한다)
- cis case ignore string (case는 비교시 무시된다)
- tel telephone number string (cis와 같지만 비교시 공백과 대시기호 '-' 는 무시된다)
- dn distinguished name (구별되는 이름)

Objectclass와 속성 정의들이 시스템의 어디에 있는지를 알기 위해서는 3 ()의 첫 문단으로 가라.

1.5 문서의 새로운 버전

이 문서는 독자들에 의해 제공되는 피드백에 기초해 수정과 갱신될 수 있다. 이 문서의 새로운 버전에 대해서는 다음을 참조해야 한다:

`<http://www.mobilesoft.com.br/HOWTO/LDAP-HOWTO.html>`

1.6 의견과 제안

이 문서에서 사용할 수 있는 어떤 정보에 대해 어떤 의구심이 있다면 다음 전자우편 주소로 메일을 보내주길 바란다.

`malere@yahoo.com`

주석과/또는 제안이 있다면 또한 저자에게 알려주길 바란다.

1.7 문서 발표 기록

이 절은 시기별로 분류된 본 발표 문서를 열거한다. 각 발표 문서는 초기 버전에 도입된 변경, 새로운 추가사항 및 수정을 포함한다.

v1.0: 20 June 1999, Initial version.

v1.01: 15 February 2000, added the following sections:

- LDAP Migration Tools
- Authentication using LDAP
- Graphical LDAP tools
- RFCs

v1.02: 13 September 2000, correction of typos and addition of the following section:

- History of Releases

v1.03: 28 September 2000, presenting OpenLDAP 2.0, which comprises Ldap v3, defined on the *RFC2251* `<ftp://ftp.isi.edu/in-notes/rfc2251.txt>`. v1.04: 28 Februrary 2001, correction of more typos and update on the following sections:

- Roaming Access
- Authentication using LDAP

1.8 Acknowledgments

이 문서는 Netherlands의 TUDelft 대학에서 수행한 인턴십의 결과이다. 저자는 이 문서를 쓰도록 용기를 준 많은 분들에게 고마움을 표현하고 싶다: Rene van Leuken과 Wim Tiwon에 매우 고맙게 생각한다. 그들은 또한 저자와 같은 리눅스 팬들이다. 저자는 이 문서에 기여한 독일어판 LDAP HOWTO 문서의 저자인 Thomas Bendler와 LDP 프로젝트의 위대한 지원자인 Joshua에게 고마움을 표현하고 싶다.

1.9 Copyright and Disclaimer

If you want a translation, for example to Portuguese, you can send me an email about it too.

No liability for the contents of this document can be accepted. I have no responsability about the consequences of following the steps provided in this document.

If you have questions, please contact, the Linux HOWTO coordinator, at

`linux-howto@metalab.unc.edu`

제 2 절 LDAP 서버 설치하기

LDAP 서버 설치는 선행 필수 패키지 설치(설치되어 있지 않을 경우), 서버 다운로드, 소프트웨어 압축해체, Makefile 설정과 서버 구축의 다섯 단계를 통해 이루어진다.

2.1 선행 필수, Pre-Requirements

LDAP 버전 3을 완전히 따르기 위해 OpenLDAP 클라이언트와 서버는 약간의 부가적인 패키지의 설치를 필요로한다:

OpenSSL TLS libraries

어떤 운영체제가 이 라이브러리를 기본 시스템의 부분 또는 선택적인 소프트웨어 컴포넌트로 제공할 수 있지만, OpenSSL은 종종 별도의 설치를 필요로 한다. OpenSSL은 다음 사이트에서 구할 수 있다.

`<http://www.openssl.org/>`

Kerberos Authentication Services

OpenLDAP 클라이언트와 서버는 Kerberos에 기초한 인증 서비스를 지원하는데 특히 Heimdal 또는 MIT Kerberos V 패키지를 이용한 SASL/GSAPI 인증 기구를 지원한다. Kerberos에 기초한 SASL/GSSAPI 인증을 사용하고 싶다면 Heimdal 또는 MIT Kerberos V를 설치해야 한다. Heimdal Kerbero는 `<http://www.pdc.kth.se/heimdal>`로부터 구할 수 있다.

MIT Kerberos는 `<http://web.mit.edu/kerberos/www>`로부터 구할 수 있다. Kerberos가 제공하는 것과 같은 강력한 인증 서비스의 사용을 강력히 추천한다.

Cyrus's Simple Authentication and Security Layer Libraries

어떤 운영체제가 이 라이브러리를 기본 시스템의 부분 또는 선택적인 소프트웨어 컴포넌트로 제공할 수 있지만, Cyrus SASL은 종종 별도의 설치를 필요로 한다. Cyrus SASL은 `<http://asg.web.cmu.edu/sasl/sasl-library.html>`로부터 구할 수 있다. Cyrus SASL은 OpenSSL과 Kerberos/GSSAPI 라이브러리가 미리 설치되어 있다면 이들을 사용할 것이다.

Database Software

OpenLDAP의 slapd 기본 데이터베이스 백엔드인 LDBM은 엔트리 저장을 위해 호환성 데이타베이스 패키지를 필요로 한다. LDBM은 Sleepycat 소프트웨어의 BerkeleyDB(추천) 또는 자유 소프트웨어 재단(FSF, Free Software Foundation)의 GNU 데이타베이스 매니저(GDBM)와 호환된다. 이러한 패키지들을 설정시 이용할 수 없다면 기본 데이타베이스 백엔드 지원을 하는 slapd 데몬을 구축할 수 없을 것이다.

운영체제가 기본 시스템내에서 또는 선택적인 소프트웨어 컴포넌트로써 두 패키지 중 하나를 제공할 수 있는데 이러한 소프트웨어를 구해서 설치할 필요가 있다.

BerkekeyDB 는 Sleepycat 소프트웨어의 다운로드 페이지 `<ttp://www.sleepycat.com/download.html>`로부터 구할 수 있다. 여러 버전을 이용할 수 있는데 이 문서를 작성하는 시점에는 최신 배포본 버전 3.1 이 추천된다.

GDBM은 FSF의 다운로드 사이트 `<ftp://ftp.gnu.org/pub/gnu/gdbm>`로부터 구할 수 있는데 이 문서를 작성하는 시점에는 버전 1.8 이 최신 배포본이다.

Threads

OpenLDAP는 쓰레드를 이용할 수 있도록 설계되어 있는데 POSIX pthreads, Mach CThreads와 많은 다른 변형들을 지원한다. *configure* 스크립트가 적합한 쓰레드 하위 시스템을 찾을 수 없다면 에러 메세지를 출력하는데 이런 경우 OpenLDAP FAQ의 소프트웨어 - 설치 - 플랫폼 힌트 절을 참조하길 바란다.

`<http://www.openldap.org/faq>`

TCP Wrappers

*slapd*는 TCP wrappers(IP 레벨 엑세스 제어 필터)가 이미 설치되어 있다면 이를 지원하는데 개인적인 비공개 정보를 포함하는 서버 보안을 위해 TCP wrappers 또는 다른 IP 레벨 엑세스 필터(IP 레벨 방화벽이 제공하는 것과 같은) 사용을 추천한다.

2.2 패키지 다운로드 받기

LDAP 서버로는 Michigan 대학 LDAP 서버와 OpenLDAP 서버 두 종류의 자유로이 배포되는 LDAP 서버가 있으며 또한 넷스케이프 디렉토리 서버도 어떤 조건하에서는 자유로이 사용할 수 있다(예를 들면 교육기관은 자유로이 얻을 수 있다). OpenLDAP 서버는 Michigan 대학 LDAP 서버의 최신 버전에 기초하는데 그 버전에 대해 이용할 수 있는 메일링 리스트와 부가적 문서가 있다. 이 문서는 OpenLDAP 서버의 사용을 가정한다.

최신 tar gzipped 버전은 다음 주소에서 구할 수 있다:

`<http://www.openldap.org/>`

Michigan 대학 LDAP 서버의 최신 버전은 다음 주소에서 구할 수 있다:

`<ftp://terminator.rs.itd.umich.edu/ldap>`

이 문서를 작성하기 위해 저자는 최신 안정 버전 1.2.11과 새로이 배포된 2.0.4 버전 두가지 OpenLDAP 패키지를 사용했다. 운영체제는 커널 2.2.13인 슬랙웨어 리눅스이다.

OpenLDAP 사이트에서 늘 OpenLDAP 서버의 최신 개발 및 안정 버전을 찾을 수 있다. 이 문서가 갱신되었던 시점까지 최신 안정 버전과 최신 개발 버전은 각각 openldap-stable-20000704.tgz과 openldap-2.0.4.tgz였다.

2.3 패키지 압축해제 하기

tar gzipped 패키지를 구한 후 압축해제한다.

우선 패키지를 원하는 디렉토리, 예를 들면 /usr/local, 로 복사하길 바란다.

그리고 다음의 명령을 사용하길 바란다:

```
tar xvzf openldap-stable.tgz
```

마찬가지로 다음 명령을 사용할 수도 있다:

```
gunzip openldap-stable.tgz | tar xvf -
```

2.4 소프트웨어 설정하기

웹사이트에 최적의 소프트웨어를 설치할 수 있도록 개별화(customization)할 수 있는 여러 옵션들이 있다. 소프트웨어는 단지 두 단계에 의해 설정된다:

- 소프트웨어를 압축해제한 디렉토리 밑의 하위 디렉토리 *include*에 위치한 *ldapconfig.h.edit* 파일을 편집한다
- configure 스크립트를 실행시킨다 (호기심이 많다면 configure 스크립트를 실행시키는 대신 Make-common 파일을 편집할 수 있다)

include/ldapconfig.h.edit 파일안에서 slapd와 slurpd 데몬의 위치와 같은 옵션을 설정할 수 있다. 파일자체가 잘 주석처리 되어 있고 default 설정은 대부분의 일반적인 관리자 선택을 반영하고 있는데 급하다면 이 단계를 지나칠 수 있다:

```
vi include/ldapconfig.h.edit
```

OpenLDAP 서버 소스는 설치 디렉토리, 컴파일러와 링커 플래그와 같은 옵션 설정을 위한 설정 스크립트와 함께 배포되는데 소프트웨어를 압축해제한 디렉토리에서 다음 명령을 실행시킨다:

```
./configure --help
```

이 명령은 소프트웨어를 설치하기 전에 configure 스크립트를 갖고 개별화할 수 있는 모든 선택사항을 출력할 것이다. 설치 디렉토리 설정에 관한 유용한 옵션은 –prefix=pref, –exe-prefix=eprefix 와 –bindir=dir 가 있다. 일반적으로 옵션없이 configure를 실행시킨다면 스크립트가 적절한 설정을 자동적으로 인지해서 default로 공통 위치에 설치하기 위해 준비할 것이다. 단지 다음과 같이 실행시킨다:

```
./configure
```

모든 것이 잘 진행되는지 보기 위해 화면에 출력되는 내용을 보길 바란다.

2.5 서버 구축하기

소프트웨어를 설정한 후 구축을 시작하는데 우선 다음 명령을 이용하여 의존성을 구축한다:

```
make depend
```

다음 명령을 이용하여 서버를 구축한다:

```
make
```

모든 것이 잘 진행된다면 서버는 설정된대로 구축될 것이다. 그렇지 않다면 설정 사항을 검토하기 위해 이전 단계로 돌아가길 바란다. 플랫폼에 관계되는 특수한 지시를 검사해야 하는데 소프트웨어를 압축해제한 디렉토리밑의 doc/install/hints 에 있다.

바이너리와 man 페이지를 설치한다. 어디에 설치하느냐에 따라 슈퍼유저일 필요가 있다.

```
su
make install
```

설치가 완료되었으며 서버 바이너리와 여러 다른 유틸리티들이 생성되었을 것이다. LDAP 서버 작동 설정 방법을 보기 위해서는 3 (다음)으로 가길 바란다.

OpenLDAP 2.0 서버의 바이너리는 *slapd*이다. OpenLDAP 2.0은 공식적으로 8월 30일 발표되었는데 RFC 2251에 정의된 바와 같이 Ldap 프로토콜 v2을 포함하고 있다.

OpenLDAP 2.0 의 주된 특징은 다음과 같다:

- LDAPv2 and LDAPv3 Support (RFC2251-2256,2829-2831)
- Maintenance of interoperability with existing clients
- IPv4 and IPv6 support
- Strong Authentication (SASL) (RFC2829)
- Start TLS (RFC2830)
- Language Tags (RFC2596)
- DNS-based service location (RFC2247+"locate" I-D)
- Enhanced Standalone Server
- Named References/ManageDsaIT ("nameref" I-D)
- Enhanced Access Control subsystem
- Thread pooling
- Preemptive threading support
- Multiple listener support
- LDIFv1 (RFC2849)
- Improved platform/subsystem detection

Note: LDP(Linux Document Projext)에 LDAP Implementation HOWTO 문서가 있을 것이다. 이 문서는 OpenLDAP 2.0 의 새로운 특징을 이용하길 원하는 사람들에게 많은 자료를 제공할 것이다. 배포 날짜는 2000년 12월 즈음이다.

OpenLDAP 패키지 최신 버전에서는 구축된 바이너리를 시험하는 것 또한 가능 한데 다음 명령을 이용하여 시험 스크립트를 실행시킬 수 있다:

```
make test
```

어떤 것이 잘 되지 않는다면 Ctrl-C를 눌러 중간에 정지시킬 수 있다. 저자의 경우 스크립트가 완전히 끝나기 전에 중간에 멈추었는데 어쨌든 OpenLDAP 설정에 대한 성공적인 메시지를 볼 수 있었다.

제 3 절 LDAP 서버 설정하기

소프트웨어의 설치 및 구축 완료후 사이트에 적합하게 설정할 수 있는데 모든 slapd 런타임 설정은 설정 스크립트에서 지정한 prefix 디렉토리 또는 default로 /usr/local/etc/openldap 디렉토리에 설치된 *slapd.conf* 파일을 통해 이루어진다.

이 절은 *slapd.conf* 파일내의 일반적으로 사용되는 설정 지시(configuration directive)들을 세부적으로 설명한다. 완전한 리스트를 위해서는 slap.conf(5) 매뉴얼 페이지를 보기 바란다. 설정 파일 지시들은 전역적, 백엔드 특정적 및 데이타 특정적인 부문으로 분류된다. 이 절에서 지시의 설명과 그들의 default 값들(존재한다면) 및 그들 사용의 예를 볼 수 있다.

3.1 설정 파일 포맷

Slapd.conf 파일은 전역적, 백엔드 특정적 및 데이타베이스 특정적인 세가지 유형의 설정 정보로 구성되는데 전역적정보, 개개의 백엔드 형태와 관련된 정보와 개별적 데이타베이스 인스턴스와 관련된 정보순으로 세부적으로 설명된다.

전역적 지시는 백엔드와/또는 데이터베이스 지시에서 무효화될 수 있으며, 백엔드 지시는 데이터베이스 지시에 의해 무효화될 수있다.

공라인과 '#'로 시작하는 주석라인은 무시되며 white space로 시작하는 라인은 이전 라인의 연속으로 고려된다. Slapd.conf 파일의 일반적 포맷은 다음과같다:

```
# global configuration directives
<global config directives>

# 백엔 definition
백엔 <typeA>
<백엔-specific directives>

# first database definition & config directives
database <typeA>
<database-specific directives>

# second database definition & config directives
database <typeB>
<database-specific directives>

# second database definition & config directives
database <typeA>
<database-specific directives>

# subsequent backend & database definitions & config directives
...
```

설정 지시는 인수를 취할 수 있는데 white space에 의해 구분한다. 인수가 white space를 포함한다면 "like this"와 같이 이중 인용부호로 에워싸야 한다. 인수가 이중 인용부호 또는 역슬래쉬문자 '\'를 포함하면 역슬래쉬 문자가 이들 문자 앞에 있어야 한다.

배포판은 /usr/local/etc/openldap 디렉토리에 설치될 수 있는 설정 파일의 예를 포함한다. 스키마 정의(속성 형태와 객체 클래스)를 포함한 다수의 파일들은 /usr/local/etc/openldap/schema 디렉토리에 설치된다.

3.2 전역적 지시

이 절에 기술된 지시들은 백엔드 또는 데이타베이스 정의에서 특별히 무효화되지 않는다면 모든 백엔드에 적용된다. 실제 텍스트에 의해 대체되는 인수들은 <> 괄호내에 있다.

access to <what> [by <who> <accesslevel> <control>]+

> 이 지시는 한명 또는 그 이상의 요청자(<who>에 의해 지정된)에 의한 일련의 엔트리와/또는 속성(<what>에 의해 지정된)에 대한 엑세스(<accesslevel>에 의해 지정된)를 허용한다. 세부사항을 알고 싶다면 엑세스 제어 예를 보길 바란다.

attributetype <RFC2252 Attribute Type Description>

> 이 지시는 속성 형태를 정의한다.

defaultaccess { none | compare | search | read | write }

이 지시는 엑세스 지시가 지정되지 않았을 때 요청자에게 허용된 default 엑세스를 지정한다. 임의의 주어진 엑세스 레벨은 모든 하위 엑세스 레벨을 내포한다 (예, 읽기 엑세스는 검색과 비교를 내포하지만 쓰기를 내포하지는 않는다).

```
Default:
defaultaccess read
```

idletimeout <integer>

유휴 클라이언트 연결을 강제로 종료하기 전에 기다리는 시간(초)을 지정한다. Default로 0의값의 ideltimeout은 이 특징을 작동시키지 않는다.

include <filename>

이 지시는 현재 파일의 다음 라인을 계속 진행하기 전에 slapd 데몬이 주어진 파일로부터 부가적인 설정 정보를 읽어야 함을 지정한다. Included된 파일은 일반적인 slapd config 파일 포맷을 따라야한다. 파일은 일반적으로 스키마 명세사항(specification)을 갖고있는 파일들을 포함하기위해 사용된다.

Note: 내포 include 지시의 수에 제한이 없으며 루프(loop) 탐지가 행해지지 않기 때문에 이 지시를 사용할 때는 주의해야 한다.

loglevel <integer>

본 지시는 디버깅 보고서(statement)와 operation 통계가 syslogged(현재는 syslogd(8) LOCAL4로 로그되어있다) 되어야 하는 레벨을 지정한다. 이것이 잘 작동되도록(늘 작동되고 있는 두 통계 레벨을 제외하고) OpenLDAP 를
--enable-debug(default) 옵션을 갖고 설정했어야 한다. 로그 레벨은 부가적이다. 어떤 숫자가 어떤 종류의 디버깅에 해당되는 지를 출력하기위해 -? 을 옵션과 함께 slapd를 실행시키거나 아래의 테이블을 참고하길 바란다. <integer>에 가능한 값들은 다음과 같다:

```
-1 enable all debugging
0 no debugging
1 trace function calls
2 debug packet handling
4 heavy trace debugging
8 connection management
16 print out packets sent and received
32 search filter processing
64 configuration file processing
128 access control list processing
256 stats log connections/operations/results
512 stats log entries sent
1024 print communication with shell backends
2048 print entry parsing debugging
```

예:
```
loglevel 255 or loglevel -1
```
이는 매우 많은 디버깅 정보가 syslogged되게 할 것이다.
```
Default: loglevel 256
```

objectclass <RFC2252 Object Class Description>

이 지시는 객체 클래스를 정의한다.

referral <URI>

이 지시는 요청을 처리하는 로컬 데이타베이스를 찾을수 없을 때 돌려보내는 referral을 지정한다.

예:
```
referral ldap://root.openldap.org
```

이는 non-local 질의에 대해 OpenLDAP 프로젝트의 전역 루트 LDAP 서버를 참조하라는 것을 의미한다. Smart LDAP 클라이언트는 그 서버에 질의를 재차 요청할 것이지만, 대부분의 클라이언트는 호스트 부분과 선택적으로 구별되는 이름 부분을 포함하는 간단한 LDAP URLs을 처리하는 방법을 알려고 한다는 것을 주목하길 바란다.

sizelimit <integer>

이 지시는 검색 연산시 리턴되는 최대 엔트리수를 지정한다.

Default:
```
sizelimit 500
```

timelimit <integer>

이 지시는 slapd가 검색 요청에 답변하기위해 쓸 수 있는 최대시간(실제시간, 초)을 지정한다. 요청이 이 시간내에 종결되지 않는다면 초과된 timelimit을 지적하는 결과가 리턴될 것이다.

Default:
```
timelimit 3600
```

3.3 일반적인 백엔드 옵션

이 절의 지시들은 오로지 그들이 정의된 백엔드에만 적용되는데 모든 종류의 백엔드가 이 지시들을 지원한다. 백엔드 지시는 같은 유형의 모든 데이터베이스 인스턴스에 적용되지만 어떤 지시냐에 따라 데이터베이스 지시에 의해 무효화될 수 있다.

backend <type>

이 지시는 백엔드 정의의 시작을 나타낸다. <type>은 ldbm, shell, passwd 또는 다른 지원되는 백엔드 유형 중 하나여야 한다.

3.4 일반적인 데이타베이스 지시

이 절의 지시들은 오로지 그들이 정의된 데이타베이스에만 적용되는데 모든 종류의 데이타베이스가 이 지시들을 지원한다.

database <type>

이 지시는 새로운 데이터베이스 인스턴스 정의의 시작을 나타낸다. <type>은 ldbm, shell, passwd 또는 다른 지원되는 백엔드 형태중의 하나여야한다.

예:
```
database ldbm
```

이는 LDBM 백엔 데이터베이스 인스턴스 정의의 시작을 나타낸다.

readonly { on | off }

이 지시는 데이터베이스를 "read-only" 모드로 만든다. 데이터베이스를 수정하려는 모든 시도는 "unwilling to perform"" 에러를 출력할 것이다.

```
Default:
readonly off
```

replica host=<hostname>[:<port>] [bindmethod={ simple | kerberos | sasl }] ["binddn=<DN>"] [mech=<mech>] [authcid=<identity>] [authzid=<identity>] [credentials=<password>] [srvtab=<filename>]

이 지시는 데이터베이스의 복사본 사이트를 지정한다. host= 변수는 호스트와 옵션으로 slave slapd 인스턴스를 찾을 수 있는 포트를 지정한다. <hostname>에는 도메인 네임 또는 IP 주소가 사용될 수 있다. <port>가 지정되지 않으면 표준 LDAP 포트 넘버(389)가 사용된다.

binddn= 변수는 slave slapd에 갱신을 위해 bind할 DN을 준다. 이는 일반적으로 slave의 config 파일에 rootdn으로 주어지는데 slave slapd 데이터베이스에 대한 읽기/쓰기 엑세스를 갖는 DN 이어야 한다. 또한 slave slapd config 파일내의 updatedn 지시와 일치해야 한다. DN은 중간에 space를 포함할 수 있기때문에 전체 "binddn=<DN>" 문자열은 이중 인용부호로 에워싸야 한다.

bindmethod는 slave slapd에 연결할 때 간단한 패스워드에 기초한 인증, Kerberos 인증 또는 SASL 인증이 사용되는지에 따라 simple, kerveros 또는 sasl 이다.

Simple 인증은 적절한 무결성과 프라이버시 보호가 적당(예, TLS 또는 IPSEC)
하지 않다면 사용하지 않아야 한다.
Simple 인증은 binddn과 credential 변수의 명세 사항을 필요로 한다.

Kerberos 인증은 SASL 인증 기구, 특히 KERBEROUS_V4와 GSSAPI 기구에 비해 그다지 지지받지 못하고 있다. Kerberos 인증은 binddn과 srvtab 변수를 필요로 한다.

SASL 인증이 일반적으로 추천되는데 mech 변수를 사용하는 기구의 명세 사항을 필요로 한다. 메카니즘에 따라 인증 identity 와/또는 credentials는 각각 authcid와 credentials을 사용하여 지정할 수 있다. authzid 변수가 인가(authorization) identity를 지정하기 위해 사용될 수도 있다.

replogfile <filename>

이 지시는 slapd가 변경사항들을 기록할 복사본 로그 파일의 이름을 지정한다. replication 로그파일은 일반적으로 slapd에 의해 작성되며 slurpd에 의해 읽혀진다. 보통 이 지시는 slurpd가 데이터베이스를 복사하기 위해 사용되는 경우만 사용된다. 그러나 slurpd가 작동되지 않더라도 트랜잭션(transaction) 로그를 생성하기 위해 이를 사용할 수 있다. 이 경우 파일이 무한정 커질 수 있기 때문에 주기적으로 파일을 truncate할 필요가 있다.

rootdn <dn>

이 지시는 데이터베이스에서 작업을 하기 위한 엑세스 제어 또는 관리상의 한계 제한을 필요로 하지않는 DN
을
지정한다. DN은 디렉토리의 엔트리를 참조할 필요가 없다. DN은 SASL identity를 참조할 수 있다.

```
Entry-based Example:
rootdn "cn=Manager, dc=example, dc=com"
```

```
SASL-based Example:
rootdn "uid=root@EXAMPLE.COM"
```

rootpw <password>

이 지시는 주어진 DN을 가진 엔트리가 존재하는지 또는 패스워드를 갖는지에 상관없이 위에서 주어진 DN
에 대해
항상 작용할 패스워드를 지정한다. 이 지시는 SASL에 기초한 인증에 비해 그다지 지지받지 못하고 있다.

예:
```
rootpw secret
```

suffix <dn suffix>

이 지시는 백엔드 데이터베이스에 보내질 질의의 DN 접미사를 지정한다. 다중 접미사 라인이 주어질 수 있으며
각 데이터베이스 정의를 위해 적어도 하나가 필요하다.

예:
```
suffix "dc=example, dc=com"
```

"dc=example, dc=com"로 끝나는 질의가 이 백엔로 보내질 것이다.

Note: 질의를 넘겨줄 백엔드가 선택될 때 각 데이터베이스 정의에서 주어진 순서대로 suffix 라인을 찾는다.
따라서 한 데이터베이스 suffix가 다른 것의 prefix라면 config 파일에서 나중에 나타나야 한다.

updatedn <dn>

이 지시는 단지 slave slapd에만 해당된다. 이는 replica 변경이 허용된 DN
을 지정한다. 이는 replica를 변경할 때
slurpd(8)가 bind할 DN 또는 SASL identity와 관련된 DN 일 수 있다.

Entry-based Example:
```
updatedn "cn=Update Daemon, dc=example, dc=com"
```

SASL-based Example:
```
updatedn "uid=slurpd@EXAMPLE.COM"
```

updateref <URL>

이 지시는 단지 slave slapd 에만 해당된다. 이는 replica에 대한 갱신 요청을 제출하는 클라이언트에 답변하는
URL을 지정한다. 여러번 지정하려면 각각의 URL을 놓는다.

예: `update  ldap://master.example.net`

3.5 LDBM 백엔드 특정적 지시

이 항목의 지시는 단지 LDBM 백엔 데이터베이스에 적용되며 "database ldbm" 라인 뒤 및 어떤 다른 "database" 라인 앞에 놓여야 한다.

cachesize <integer>

이 지시는 LDBM 백엔드 데이터베이스 인스턴스에 의해 유지되는 in-memory cache 엔트리의 크기를 지정한다.

Default:
```
cachesize 1000
```

dbcachesize <integer>

이 지시는 각 오픈 인덱스 파일과 관련된 in-memory cache의 바이트 크기를 지정한다. 기본적인 데이터베이스 방법이 지원하지 않는다면 주석처리없이 무시된다. 이 숫자의 증가는 더많은 메모리 사용을 의미하지만 특히 인덱스 변경중 또는 구축할 때 효과적인 성능향상을 가져올 것이다.

```
Default:
dbcachesize 100000
```

dbnolocking

이 옵션은 존재한다면 데이터베이스 locking 기능을 억제한다. 이 옵션을 작동시키면 데이터 보안을 낮추면서 성능을 향상시킬 것이다.

dbnosync

이 옵션은 디스크상의 데이터베이스 내용이 메모리내에서 바뀌는 변경에 즉각적으로 동기화되지 않도록 한다. 이 옵션의 활성화는 데이터 보안을 낮추지만 성능을 향상시킬 것이다.

directory <directory>

이 지시는 데이터베이스와 관련 인덱스를 포함하는 LDBM 파일이 놓이는 디렉토리를 지정한다.

```
Default:
directory /usr/local/var/openldap-ldbm
```

index {<attrlist> | default} [pres,eq,approx,sub,none]

이 지시는 주어진 속성에 대해 유지할 인덱스를 지정한다. 단지 <attrlist>가 주어진다면 디폴트 인덱스가 유지된다.

```
예:
index default pres,eq
index objectClass,uid
index cn,sn eq,sub,approx
```

첫 번째 라인은 present와 equality에 유지할 디폴트 인덱스 집합을 설정한다. 두 번째 라인은 디폴트(pret,eq) 인덱스 집합이 objectClass와 uid 속성 형태를 위해 유지되도록 한다. 세 번째 라인은 equality, substring와 approximate 인덱스들이 cn과 sn 속성 형태를 위해 유지되도록 한다.

mode <integer>

이 지시는 새로이 생성된 데이터베이스 인덱스 파일이 가져야 하는 파일 보호 모드를 지정한다.

```
Default:
mode 0600
```

3.6 다른 백엔드 데이타베이스

*slapd*는 디폴트 LDBM 이외에도 많은 백엔드 데이터베이스 형태를 지원한다:

- ldbm: Berkeley or GNU DBM compatible backend
- passwd: Provides read-only access to /etc/passwd
- shell: Shell (extern program) backend
- sql: SQL Programmable backend

세부사항을 알기 위해서는 slapd.conf(5) 매뉴얼 페이지를 보길 바란다.

3.7 엑세스 제어 예

3.2 (3.2절)에 설명한 엑세스 제어 지시는 매우 강력한데 이 절은 엑세스 제어 사용의 몇 예를 보여준다. 우선, 약간의 간단한 예들:

```
access to * by * read
```

이 엑세스 지시는 모든 사람에게 읽기 엑세스를 허용한다. 이 지시가 단독으로 나타나면 다음의 defaultaccess 라인과 같다.

```
defaultaccess read
```

다음 예는 순서가 중요한 두 엑세스 지시에서 DN에 의해 엔트리를 선택하는 정규 표현 사용의 예를 보여준다.

```
access to dn=".*, o=U of M, c=US"
by * search
access to dn=".*, c=US"
by * read
```

검색 엑세스가 허용된 "o=University of Michigan, c=US" 하위 트리하의 엔트리를 제외한 c=Us 하위 트리하의 엔트리에 읽기 엑세스가 허용된다. 이러한 엑세스 지시의 순서가 반전되면, 모든 U-M 엔트리와 c=US 엔트리가 동일하기 때문에 U-M 특정적 지시는 절대로 부합될 수 없다.

다음 예 또한 엑세스 지시와 "by" 절(clause)의 순서의 중요성을 보여주는데 특정 속성과 다양한 <who> 선택자에 대한 엑세스를 허용하는 속성 선택자 사용의 예를 보여준다.

```
access to dn=".*, o=U of M, c=US" attr=homePhone
by self write
by dn=".*, o=U of M, c=US" search
by domain=.*\.umich\.edu read
by * compare
access to dn=".*, o=U of M, c=US"
by self write
by dn=".*, o=U of M, c=US" search
by * none
```

이 예는 "o=U of M, c=US" 하부 트리내의 엔트리에 적용된다. homePhone를 제외한 모든 속성들에 대해 엔트리가 속성들을 쓸 수 있고, 다른 U-M 엔트리는 속성들에 의해 검색되며 어느 누구도 엑세스를 하지 못한다. homePhone 속성은 엔트리에 의해 쓸 수 있고, 다른 U-M 엔트리에 의해 검색할 수 있고, umich.edu 도메인 상에서 연결하는 클라이언트에 의해 읽을 수 있고, 다른 모든 사람에 의해 비교할 수 있다.

때때로 특정 DN에 자신의 속성을 추가 또는 삭제할 수 있는 권한을 주는 것이 유용하다. 예를 들어 그룹을 생성해서 사람들로 하여금 member 속성에서 그들 소유의 DN을 추가 및 삭제할 수 있게 허용하고 싶다면 다음의 엑세스 지시를 이용해 수행할 수 있다:

```
access to attr=member,entry
by dnattr=member selfwrite
```

dnattr <who> 선택자는 엑세스가 member 속성에 리스트된 엔트리에 적용됨을 말해준다. selfwrite 엑세스 선택자는 그 member들이 다른 값을 제외한 그들의 DN 값만을 속성에 추가 및 삭제할 수 있음을 말해준다. 어떤 엔트리 속성의 엑세스를 위해서는 엔트리 엑세스가 필요하기 때문에 엔트리 속성의 추가가 필요하다.

<what>절에서 attr=member 는 "dn=* attr=member" 절(즉 이것은 모든 엔트리에 member 속성과 부합된다)의 속기임을 주목하기 바란다.

Note:Ldap의 엑세스 제어에 대해 더 많은 것을 배우기 위해 `<http://openldap.org/>`의 OpenLDAP 관리자 지침을 보라.

3.8 설정 파일 예

다음은 설명 텍스트가 들어있는 설정 파일 예제이다. 이는 X.500 트리의 여러 부분들을 다루기 위해 LDBM 데이타베이스 인스턴스인 두 개의 데이터베이스를 정의한다. 라인 숫자는 참조를 위한 것으로 실제 파일에는 없다. 우선 전역적 설정 부분:

- 1. # example config file - global configuration section
- 2. include /usr/local/etc/schema/core.schema
- 3. referral ldap://root.openldap.org
- 4. access to * by * read

라인 1은 주석이다. 라인 2는 핵심 스키마 정의를 갖는 다른 config 파일을 포함한다. 라인 3의 referral 지시는 밑에 정의된 데이터베이스 중 하나에 지역적이 아닌 질의는 root.openldap.org 호스트의 표준 포트(389)에서 작동되는 LDAP 서버를 참조할 것임을 의미한다.

라인 4는 전역적 엑세스 제어로 부합되는 데이터베이스 엑세스 제어가 없거나 또는 타겟 객체가 Root DSE와 같은 임의의 데이터베이스의 제어하에 없을때만 사용된다.

설정 파일의 다음 부분은 트리의 "dc=example,dc=com" 부분에 있는 내용에 대한 질의를 다룰 LDBM 백엔드를 정의한다. 데이터베이스는 각각 truelies와 judgementday 두 개의 slapd 에 복사될 것이다. 인덱스는 여러 속성을 위해 유지되며 userPassword 속성은 인가받지 못한 엑세스에 대해 보호된다.

- 5. # ldbm definition for the example.com
- 6. database ldbm
- 7. suffix "dc=example, dc=com"
- 8. directory /usr/local/var/openldap
- 9. rootdn "cn=Manager, dc=example, dc=com"
- 10. rootpw secret
- 11. # replication directives
- 12. replogfile /usr/local/var/openldap/slapd.replog
- 13. replica host=slave1.example.com:389
- 14. binddn="cn=Replicator, dc=example, dc=com"
- 15. bindmethod=simple credentials=secret
- 16. replica host=slave2.example.com
- 17. binddn="cn=Replicator, dc=example, dc=com"
- 18. bindmethod=simple credentials=secret

- 19. # indexed attribute definitions
- 20. index uid pres,eq
- 21. index cn,sn,uid pres,eq,approx,sub
- 22. index objectClass eq
- 23. # ldbm access control definitions
- 24. access to attr=userPassword
- 25. by self write
- 26. by anonymous auth
- 27. by dn="cn=Admin,dc=example,dc=com" write
- 28. by * none
- 29. access to *
- 30. by self write
- 31. by dn="cn=Admin,dc=example,dc=com" write
- 32. by * read

라인 5는 주석이다. 라인 6의 데이터베이스 키워드에 의해 데이터베이스 정의가 시작된다. 라인 7은 이 데이터베이스에 보내질 질의에 대한 DN suffix를 지정한다. 라인 8은 데이터베이스 파일이 놓일 디렉토리를 지정한다.

라인 9와 10은 데이터베이스 "super user" 엔트리와 관련 패스워드를 지정한다. 이 엔트리는 엑세스 제어 또는 크기 또는 시간 한계 제한을 필요로 하지 않는다.

라인 11-18은 복사본에 대한 것으로 라인 11은 복사본 로그 파일을 지정한다 (데이터베이스에 대한 변경 사항이 기록되는데 slapd 에 의해 쓰여지고 slurpd 에 의해 읽혀진다). 라인 12-14는 복사된 호스트에 대한 호스트 네임과 포트, 갱신할때의 bind할 DN, binddn에 대해 bind 방법(간략)및 credentials(패스워드)를 지정한다. 라인 15-18은 두 번째 복사본 사이트를 지정한다.

라인20-22는 다양한 속성에 대해 유지되는 인덱스를 가리킨다.

라인 24-32는 데이터베이스내의 엔트리에 대한 엑세스 제어를 지정한다. 모든 엔트리에 대해 userPassword는 엔트리 자체 및 "admin" 엔트리에 의해 쓸 수 있다. 이는 인증/인가 목적에 사용될 수 있지만 그렇지 않은 경우 읽을 수 없다. 모든 다른 속성은 엔트리와 "admin" 엔트리에 의해 쓸 수 있지만 인증받은 사용자에 의해 읽힐 수 있다.

설정 파일 예의 다음 부분은 다른 LDBM 데이터베이스를 정의하는데 이 데이터베이스는 dc=example,dc=net 하위 트리를 포함한 질의를 처리한다. 라인 38이 없다면 라인 4에 있는 전역적 엑세스 규칙 때문에 읽기 엑세스가 허용될 수 있음을 주목하기 바란다.

- 33. # ldbm definition for example.net
- 34. database ldbm
- 35. suffix "dc=example, dc=net"
- 36. directory /usr/local/var/ldbm-example-net
- 37. rootdn "cn=Manager, dc=example, dc=com"
- 38. access to * by users read

제 4 절 LDAP 서버 구동하기

*slapd*는 스탠드 얼론 서버로서 작동되도록 설계되어 있어 서버가 캐싱 이용, 기본 데이터베이스와의 동시 작용 문제 처리 및 시스템 자원 보호를 할 수 있다. inetd(8)로부터의 작동은 옵션이 아니다.

4.1 Command Line Options

*slapd*는 메뉴얼 페이지에 상세히 설명된 바와 같이 많은 command-line 옵션을 지원한다. 이 절은 일반적으로 자주 사용되는 약간의 옵션을 상세히 설명한다:

-f <filename>

이 옵션은 slapd에 대한 대체 구성 파일을 지원한다. 디폴트는 보통 /usr/local/etc/openldap/slapd.conf 파일이다.

-h <URLs>

이 옵션은 대체 listener 구성을 지정한다. 디폴트는 ldap:/// 로 디폴트 LDAP
포트 389의 TCP 인터페이스를 갖는
LDAP를 의미한다. 호스트-포트 쌍 및 ldaps:// 또는 ldapi:// 와 같은 다른 프로토콜 계획을 지정할 수 있다.
예를들어, -h "ldaps:// ldap:/127.0.0.1:667"은 두 개의 listener를 생성할 것이다: 하나는 디폴트 LDAP/SSL 포트
636의 모든 인터페이스에서 SSL을 이용하는 LDAP이고 다른 하나는 포트 667의 로컬 호스트(루프백,loopback)에서
TCP를 이용한 LDAP. 호스트는 IPv4 dotted-decimal 형태 또는 호스트 네임을 사용하여 지정될 수 있다.
포트값은 수치여야 한다.

-n <service-name>

이 옵션은 로깅과 다른 목적을 위해 사용되는 서비스 이름을 지정한다. 디폴트 서비스 이름은 slapd이다.

-l <syslog-local-user>

이 옵션은 syslog(8) 에 대한 로컬 사용자를 지정한다. LOCAL0, LOCAL1, LOCAL2,..., 와 LOCAL7이 값이 있다.
디폴트는 LOCAL4이다. 이 옵션은 모든 시스템에서 지원되지 않을 수 있다.

-u user -g group

이 옵션들은 각각 서버를 작동하는 사용자와 그룹을 지정한다. 사용자와 그룹은 각각 사용자 및 그룹 이름과
uid 및 gid 일 수 있다.

-r directory

이 옵션은 런타임 디렉토리를 지정한다. slapd는 listener을 오픈한 후 그렇지만 어떤 구성파일을 읽기 전 또는
어떤 백엔드를 초기화하기 전에 이 디렉토리로 chroot(2) 할 것이다.

-d <level> | ?

이 옵션은 slapd 디버그 레벨을 <level>로 설정한다. 레벨이 '?' 문자일 때는 선택한 옵션에 상관없이 다양한
디버깅 레벨이 출력되며 slapd 는 종료된다. 현재 디버깅 레빌은 다음과 같다:

```
-1  enable all debugging
0  no debugging
1  trace function calls
2  debug packet handling
4  heavy trace debugging
8  connection management
16  print out packets sent and received
32  search filter processing
64  configuration file processing
128  access control list processing
256  stats log connections/operations/results
512  stats log entries sent
1024  print communication with shell 백엔s
2048  print entry parsing debugging

각각의 원하는 레벨에 대해 디버그 옵션을 지정함으로써 다중 레벨을 작동시킬 수 있다. 또한  디버깅 레벨은
부가적이기 때문에 스스로 레벨을 계산할 수 있다. 즉, function call을 tracing 해서 config 파일이
프로세싱되는 것을 보려고 한다면 이러한 두 레벨의 합(이 경우 -d 65)으로 레벨을 설정할 수 있을 것이다.
또는 slapd 가 계산을 하도록 할 수 있다(예, -d 1 -d 64). 더 많은 세부사항을 알기 위해서는 <ldap.h> 파일을
참고하길 바란다.

Note: slapd가  두 stats 레벨 이상의 임의의 디버깅 정보를 이용할 수 있도록 정의된 -DLDAP_DEBUG 옵션을
갖고 컴파일되어있어야 한다.
```

4.2 LDAP 서버 시작하기

일반적으로 slapd 는 다음과 같이 구동시킨다:

/usr/local/etc/libexec/slapd [<option>]*

/usr/local/etc/libexec는 configure에 의해 결정되며 <option>은 위에서 설명한 옵션(또는 slapd(8)) 중의 하나이다. 레벨 0를 포함하여 디버깅 레벨을 지정하지 않는다면 slapd는 자동적으로 분기(fork)하여 그 자신의 제어 터미널로부터 분리해서 백그라운드에서 실행된다.

4.3 LDAP 서버 중지하기

slapd를 안전하게 종료시키기 위해 다음의 명령을 실행시켜야 한다:

kill -TERM 'cat $(ETCDIR)/slapd.pid'

더욱 과감한 방법으로 slapd를 종료하는 것은 그것이 종료전에 다양한 버퍼를 flush 할 필요가 있을 수 있기 때문에 LDBM 데이터베이스를 손상시킬 수 있다. slapd는 자신의 pid를 slapd.conf 파일에 설정했던 디렉토리(예를들어 /usr/local/var/slapd.pid)내의 slapd.pid 파일에 쓴다는 것을 주목해라.

include/ldapconfig.h.edit 파일의 SLAD_PIDFILE 변수를 변경함으로써 이 pid 파일의 위치를 변경할 수 있다.

Slapd 는 slapd.conf 파일에 설정했던 디렉토리(예를들어 /usr/local/var/slapd.args)내의 slapd.args 파일에 또한 인수들을 쓸 것이다.

include/ldapconfig.h.edit. 파일의 SLAPD_ARGSFILE 변수를 변경함으로써 args 파일의 위치를 변경할 수 있다.

제 5 절 데이타베이스 생성과 유지 보수

이 절에서는 scratch로부터 slapd 데이타베이스를 생성하는 방법에 대해 논의한다. 데이타베이스는 두가지 방법으로 생성할 수 있다. 첫째, LDAP를 이용하여 온라인상에서 데이터베이스를 생성할 수 있는데 간단히 slapd를 구동하고 선택한 LDAP 클라이언트를 이용하여 엔트리를 추가해주면 된다. 이 방법은 비교적 작은 데이터베이스에 대해서는 더할 나위없이 좋다 (요구에 따라 수백 또는 수천개의 엔트리).

두 번째는 인덱스 생성 도구를 이용하여 오프라인상에서 데이터베이스를 생성하는 것인데 LDAP 방법을 이용할 때 매우 오랜 시간이 소요될 수 있는 방대한 엔트리 생성 또는 데이터베이스가 생성되는 동안 엑세스되지 않기를 원할 경우 좋은 방법이다.

5.1 온라인상에서 데이타베이스 생성하기

OpenLDAP 소프트웨어 패키지에는 LDAP 서버 작동중에 엔트리를 추가하는데 사용하는 ldapadd 유틸리티를 포함하고 있다. 온라인상에서 데이터베이스를 생성하려고 한다면 엔트리 추가를 위해 ldapadd 도구를 사용할 수 있다. 첫 번째 엔트리를 추가한 후 더 많은 엔트리를 추가하기 위해 ldapadd를 사용할 수 있다. slapd를 구동하기 전에 slapd.conf 파일에 다음 옵션이 설정되어 있음을 확인해야 한다.

suffix <dn>

3 (3절)에 설명한 바와 같이 이 옵션은 어떤 엔트리가 이 데이타베이스에 들어있는지를 말해주는데 이를 생성하려고 하는 하위 트리의 루트 DN으로 설정해야 한다:

suffix "o=TUDelft, c=NL"

인덱스 파일이 생성되어 놓이는 디렉토리 지정을 확실히 해주어야 한다:

directory <directory>

예:

directory /usr/local/tudelft

엔트리를 추가할 수 있는 허가권을 가진 누군가로 slapd 에 연결할 수 있도록 설정할 필요가 있는데 이는 데이터베이스 정의에서 다음 두 옵션을 통해 이루어진다:

rootdn <dn>

rootpw <passwd> /* 암호화된 패스워드를 사용하는 것을 기억해라 !!! */

이 옵션들은 데이터베이스의 슈퍼유저 엔트리(어떤 작업이든 할 수 있는 엔트리)를 인증하는데 사용될 수 있는 DN과 password를 지정한다. 여기서 지정한 DN과 password는 실제 이름을 갖는 엔트리가 존재하든지 또는 엔트리가 지정된 패스워드를 갖는지 상관없이 늘 작동한다. 이는 아직 어떤 엔트리가 존재하기도 전에 어떻게 인증을 하고 어떻게 엔트리를 추가하는지의 chicken and egg (병아리가 먼저냐 닭이 먼저냐 하는 식의) 문제를 해결한다.

마지막으로 데이터베이스 정의가 원하는 인덱스 정의를 포함하는지를 확인해야 한다:

index {<attrlist> | default} [pres,eq,approx,sub,none]

예를들어 cn, sn, uid 와 objectclass 속성을 인덱스하기 위해 다음 인덱스 설정 라인이 사용될 수 있다.

index cn,sn,uid

index objectclass pres,eq

index default none

취향에 맞게 구성했다면 slapd를 구동하고 LDAP 클라이언트로 연결하여 엔트리 추가를 시작해라. 예를들어 ldapadd 도구를 이용하여 TUDelft 엔트리와 Postmaster 엔트리를 순차적으로 추가하기 위해 다음 내용을 갖는 /tmp/newentry 파일을 생성할 수 있다:

```
o=TUDelft, c=NL
objectClass=organization
description=Technical University of Delft Netherlands

cn=Postmaster, o=TUDelft, c=NL
objectClass=organizationalRole
cn=Postmaster description= TUDelft postmaster - postmaster@tudelft.nl
```

그리고나서 엔트리를 실제 생성하기 위해 다음 명령을 사용한다:

```
ldapadd -f /tmp/newentry -D "cn=Manager, o=TUDelft, c=NL" -w secret
```

위 명령은 rootdn을 "cn=Manager, o=TUDelft, c=NL" 으로 rootpw를 "secret" 설정했다고 가정한다. command-line 상에서 패스워드를 타이핑하길 원하지 않는다면 -w "password" 대신 ldapadd 명령에 대해 -W 옵션을 사용해라. 패스워드를 입력하는 프롬프트를 볼 수 있을 것이다:

```
ldapadd -f /tmp/newentry -D "cn=Manager, o=TUDelft, c=NL" -W
Enter LDAP Password:
```

5.2 오프라인상에서 데이타베이스 생성하기

데이터베이스를 생성하는 두 번째 방법은 다음에 설명된 인덱스 생성 도구를 이용하여 오프라인상에서 작업을 하는 것인데 LDAP 방법을 이용할 때 매우 오랜 시간이 소요될 수 있는 방대한 엔트리 생성 또는 데이터베이스가 생성되는 동안 엑세스되지 않기를 원할 경우 좋은 방법이다. 이 도구는 slapd 설정 파일과 추가되는 엔트리의 텍스트 표현을 포함하는 입력 LDIF 파일을 읽어들이는데 LDBM 인덱스 파일을 직접적으로 생성한다. config 파일 데이터베이스 정의에서 우선적으로 확인 및 설정하길 원하는 여러 중요한 설정 옵션이 있다:

suffix <dn>

이전 절에서 설명한 바와 같이, 이 옵션은 어떤 엔트리가 이 데이타베이스에 들어 있는지를 말해주는데 이를 생성하려고 하는 하부 트리의 루트 DN에 설정해야 한다. 예:

suffix "o=TUDelft, c=NL"

인덱스 파일이 생성되어 놓이는 디렉토리 지정을 확실히 해주어야 한다.

directory <directory>

예:

directory /usr/local/tudelft

다음 아마도 각 오픈 인덱스 파일이 사용하는 in-core 캐시의 크기를 증가시키길 원할 수 있는데 인덱스 생성중 최상의 성능을 위해 전체 인덱스가 메모리상에 놓여야 한다. 데이터가 메모리에 올리지 못할 정도로 방대하거나 또는 메모리가 너무 작다면 메모리 크기를 증가시키거나 페이징 시스템을 작동시킬 수 있다. 이 크기는 다음 옵션에 의해 설정된다:

dbcachesize <integer>

예: dbcachesize 50000000

이 옵션은 꽤 큰(Michigan 대학에서 데이터베이스는 대략 125K 엔트리를 가지며 가장 큰 인덱스 파일은 대략 45MB 이다) 50MB 크기의 캐시를 생성할 것이다. 시스템이 어떤 옵션 값에서 최상으로 작동하는 가를 살펴보기 위해 이 비트수와 아래서 설명되는 parallelism 정도를 변화시키면서 실험해봐라. 인덱스 파일이 일단 생성되면 slapd를 실행시키기 전에 이 값을 감소시키는 것을 잊지 마라.

마지막으로 어떤 인덱스를 구축하길 원하는지 지정할 필요가 있는데 이는 하나 또는 그 이상의 인덱스 옵션에 의해 지정된다:

index {<attrlist> | default} [pres,eq,approx,sub,none]

예:

index cn,sn,uid pres,eq,approx

index default none

이는 cn, sn과 uid 속성에 대해 presence, equality 와 approximate 인덱스를 생성하며 나머지 다른 속성에 대해서는 인덱스를 생성하지 않을 것이다. 이 옵션에 대해 더 많은 정보를 얻기 위해서는 3 (3절)의 설정 파일을 보라.

취향에 맞게 설정했다면 slapadd(8) 프로그램을 실행시킴으로써 기본 데이터베이스와 관련 인덱스를 생성한다:

slapadd -l <inputfile> -f <slapdconfigfile> [-d <debuglevel>] [-n <integer>|-b <suffix>]

인수들은 다음의 의미를 갖는다:

-l <inputfile>

텍스트 형태로 추가되는 엔트리를 포함한 LDIF 입력 파일을 지정한다(다음 절을 보라).

-f <slapdconfigfile>

인덱스를 어디에 생성하는지, 어떤 인덱스를 생성하는지 등을 말해주는 slapd 설정 파일을 지정한다.

-d <debuglevel>

<debuglevel>에 의해 지정된 디버깅을 작동시킨다. 디버그 레벨은 slapd 에 대한 레벨과 같다. 4.1 절의 옵션을 보라.

-n <databasenumber>

어떤 데이터베이스가 수정되는가를 지정하는 선택적 인수로 설정 파일에 명시된 첫 번째 데이터베이스는 1, 두 번째는 2 등으로 표현된다. 디폴트로 설정 파일의 첫 번째 ldbm 데이터베이스가 사용된다. -b 옵션과 함께 사용되서는 안된다.

-b <suffix>

어떤 데이터베이스가 수정되는가를 지정하는 선택적 인수로 데이터베이스 넘버를 결정하기 위한 데이터베이스 suffix 지시와 부합되지 않는다. -n 옵션과 함께 사용되서는 안된다.

slapd.conf(5) 파일을 수정한 후와 같이 때때로 인덱스들을 재 생성할 필요가 있을 수 있는데 이는 slapindex(8) 프로그램을 이용하여 가능하다. slapindex 는 다음과 같이 실행시킨다:

slapindex -f <slapdconfigfile> [-d <debuglevel>] [-n <databasenumber>|-b <suffix>]

-f, -d, -n 과 -b 옵션은 slapadd(1) 프로그램에 대한 옵션과 동일하다. slapindex는 현재 데이터베이스 내용에 기초한 모든 인덱스들을 재 구축한다.

데이터베이스를 LDIF 파일로 덤프(dump)하는데 사용되는 slapcat 이라는 프로그램이 있는데 이는 데이터베이스를 읽을 수 있는(human-readable) 백업을 할때나 데이터베이스를 오프라인 상에서 편집하려고 할 때 유용하다. 이 프로그램은 다음과 같이 실행시킨다:

slapcat -l <filename> -f <slapdconfigfile> [-d <debuglevel>] [-n <databasenumber>|-b <suffix>]

-n 또는 -b 옵션은 -f를 사용하여 지정된 slapd.conf(5)내의 데이터베이스를 선택하는데 사용된다. 해당 LDIF 출력은 표준 출력 또는 -l 옵션을 사용하여 지정된 파일에 작성된다.

5.3 More on the LDIF format

LDAP Data Interchange Format (LDIF)은 간단한 텍스트 포맷으로 LDAP 엔트리를 표현하기 위해 사용된다. 엔트리의 기본 폼은:

```
#comment
dn: <distinguished name>
<attrdesc>; <attrvalue>
<attrdesc>; <attrvalue>
...
```

'#' 문자로 시작하는 라인들은 주석이다. 속성 설명(attrdesc)은 cn 또는 objectClasse 또는 1.2.3(속성 형태와 관련된 OID)과 같은 간단한 속성형태 이거나 cn:lang_en_US 또는 userCertificate;binay와 같은 옵션을 포함할 수 있다.

라인은 single space 또는 tab 문자로 다음 라인을 시작함으로써 계속될 수 있다. 예를들어:

```
dn: cn=Barbara J Jensen, dc=example, dc=
 com
cn: Barbara
      Jensen
```

은

```
dn: cn=Barbara J Jensen, dc=example, dc=com
cn: Barbara J Jensen
```

과 동일하다.

다중 속성 값들은 별개 라인에서 지정된다. 예:

```
cn: Barbara J Jensen
cn: Babs Jensen
```

<attrvalue>가 출력되지 않는 문자들을 포함하거나 또는 space, 콜론(':') 또는 '<' 으로 시작된다면, <attrdesc>다음에 이중 콜론과 base64 로 암호화된 값이 온다. 예를들어 "space로 시작"되는 값은 다음과 같이 암호화될 것이다:

```
 cn:: IGJlZ2lucyB3aXRoIGEgc3BhY2U=
```

속성값을 포함하는 URL을 지정할 수 있다. 예를들어, 다음은 jpegPhoto 값이 /path/to/file.jpg 파일로부터 얻어야 함을 지정한다.

```
cn:<file://path/to/file.jpeg
```

동일한 LDIF 파일내의 다중 엔트리들은 blank 라인으로 구별된다. 세 개의 엔트리를 갖는 LDIF 파일의 예는 다음과 같다:

```
# Barbara's Entry
dn: cn=Barbara J Jensen, dc=example, dc=com
cn: Barbara J Jensen
cn: Babs Jensen
objectClass: person
sn: Jensen
```

```
# Bjorn's Entry
dn: cn=Bjorn J Jensen, dc=example, dc=com
cn: Bjorn J Jensen
cn: Bjorn Jensen
objectClass: person
sn: Jensen
# Base64 encoded JPEG photo
jpegPhoto:: /9j/4AAQSkZJRgABAAAAAQABAAD/2wBDABALD
A4MChAODQ4SERATGCgaGBYWGDEjJR0oOjM9PDkzODdASFxOQ
ERXRTc4UG1RV19iZ2hnPk1xeXBkeFxlZ2P/2wBDARESEhgVG

# Jennifer's Entry
dn: cn=Jennifer J Jensen, dc=example, dc=com
cn: Jennifer J Jensen
cn: Jennifer Jensen
objectClass: person
sn: Jensen
# JPEG photo from file
jpegPhoto:<file://path/to/file.jpeg
```

Bjorn 엔트리의 jpegPhoto는 base64로 암호화되어 있고 Jennifer 엔트리의 jpegPhoto는 URL이 가리키는 위치로부터 얻어짐을 주목하라.

trailing space는 LDIF 파일의 값들로부터 정리되지 않으며 또한 내부의 다중 space는 압축되지도 않는다. 데이터내에 trailing과 내부 space를 원하지 않는다면 거기에 그들을 넣지 마라.

5.4 The ldapsearch, ldapdelete and ldapmodify utilities

ladpsearch - ldapsearch 는 ldap_search(3) 라이브러리 콜에 대해 쉘이 엑세스할 수 있는 인터페이스로 LDAP 데이터베이스 백엔의 엔트리를 검색하기 위해 사용한다.

ldapsearch를 호출하기 위한 개요는 다음과 같다 (각 옵션의 의미를 알기 위해 ldapsearch man 페이지를 찾아보라):

```
ldapsearch [-n] [-u] [-v] [-k] [-K] [-t] [-A] [-B] [-L] [-R] [-d debuglevel] [-F sep] [-f file]
[-D binddn] [-W] [-w bindpasswd] [-h ldaphost] [-p ldapport] [-b searchbase] [-s base|one|sub]
[-a never|always|search|find] [-l timelimit] [-z sizelimit] filter [attrs...]
```

ladpsearch는 LDAP 서버에 연결, 바인드해 필터를 이용하여 검색을 수행한다. 필터는 RFC 1558에 정의된 것과 같은 LDAP 필터에 대한 문자열 표현을 따라야 한다. ldapsearch가 하나 또는 그 이상의 엔트리를 찾으면 attrs에 의해 지정된 속성들이 검색되어 엔트리와 값이 표준 출력으로 출력된다. attrs가 명시되지 않으면 모든 속성들을 되돌려준다.

다음은 ldapsearch 사용 예이다:

```
ldapsearch -b 'o=TUDelft,c=NL' 'objectclass=*'

ldapsearch -b 'o=TUDelft,c=NL' 'cn=Rene van Leuken'

ldasearch -u -b 'o=TUDelft,c=NL' 'cn=Luiz Malere' sn mail
```

-b 옵션은 searchbase(initial search point, 초기 검색 지점)을 -u 옵션은 사용자에 편리한(userfriendly) 출력 정보를 나타낸다.

ldapdelete - ldapdelete는 ldap_delete(3) 라이브러리 콜에 대해 쉘이 엑세스할 수 있는 인터페이스로 LDAP 데이터베이스 백엔드의 엔트리를 삭제하기 위해 사용한다.

ldapdelete를 호출하기 위한 개요는 다음과 같다 (각 옵션의 의미를 알기 위해 ldapdelete man 페이지를 찾아보라):

```
ldapdelete [-n] [-v] [-k] [-K] [-c] [-d debuglevel] [-f file] [-D binddn] [-W] [-w passwd]
[-h ldaphost] [-p ldapport] [dn]...
```

ldapdelete는 LDAP 서버에 연결, 하나 또는 그 이상의 엔트리를 바인드해 삭제한다. 하나 또는 그 이상의 dn 인수가 제공되면 이러한 DN을 갖는 엔트리는 삭제된다. 각 dn은 RFC 1779에 정의된 것과 같은 문자열로 표현된 DN 이어야 한다. dn 인수가 없다면 DN의 리스트가 표준 입력(또는 -f flag가 사용된다면 파일)으로부터 읽혀진다.

다음은 ldapdelete 사용 예이다:

```
ldapdelete 'cn=Luiz Malere,o=TUDelft,c=NL'

ldapdelete -v 'cn=Rene van Leuken,o=TUDelft,c=NL' -D 'cn=Luiz malere,o=TUDelft,c=NL' -W
```

-v 옵션은 verbose 모드, -D 옵션은 Binddn(인증되어야 하는 dn), -W 옵션은 패스워드 프롬프트를 나타낸다.

ldapmodify - ldapmodify 는 ldap_modify(5)와 ldap_add 라이브러리 콜에 대해 쉘이 엑세스할 수 있는 인터페이스로 LDAP 데이터베이스 백엔드의 엔트리를 수정하기 위해 사용한다.

ldapmodify를 호출하기 위한 개요는 다음과 같다(각 옵션의 의미를 알기 위해 ldapmodify man 페이지를 찾아보라)

```
ldapmodify [-a] [-b] [-c] [-r] [-n] [-v] [-k] [-d debuglevel] [-D binddn] [-W] [-w passwd]
[-h ldaphost] [-p ldapport] [-f file]

ldapadd [-b] [-c] [-r] [-n] [-v] [-k] [-K] [-d debuglevel] [-D binddn] [-w passwd] [-h ldaphost]
[-p ldapport] [-f file]
```

ldapadd는 ldapmodify 도구에 대해 하드 링크되어 수행되는데 ldapadd가 실행될 때 ldapmodify의 -a (새로운 엔트리를 추가) flag 가 자동적으로 설정된다. ldapmodify는 LDAP 서버에 연결, 바인드해 엔트리를 수정 또는 추가한다. 엔트리 정보는 표준 입력 또는 -f 옵션을 사용시 파일로부터 읽혀진다.

다음은 ldapmodify의 사용 예이다:

/tmp/entrymods 가 존재하고 다음 내용을 갖고 있다고 가정한다:

```
dn: cn=Modify Me, o=University of Michigan, c=US
changetype: modify
replace: mail
mail: modme@terminator.rs.itd.umich.edu
-
add: title
title: Grand Poobah
-
add: jpegPhoto
jpegPhoto: /tmp/modme.jpeg
-
delete: description
-
```

명령:

```
ldapmodify -b -r -f /tmp/entrymods
```

이는 "Modify Me" 엔트리의 메일 속성 내용을 "modme@terminator.rs.itd.umich.edu"로 대체하고, "Grand Poobah"을 타이틀에 /tmp/modme.jpeg 파일의 내용을 jpegPhoto로 추가하며, description 속성을 완전히 삭제한다.

위와 동일한 수정은 이전 ldapmodify 입력 포맷을 이용하여 수행할 수 있다:

```
cn=Modify Me, o=University of Michigan, c=US mail=modme@terminator.rs.itd.umich.edu
+title=Grand Poobah
+jpegPhoto=/tmp/modme.jpeg
-description
```

다음 명령을 실행시킨다: ldapmodify -b -r -f /tmp/entrymods

/tmp/newentry 파일이 존재하고 다음 내용을 갖는다고 가정한다:

```
dn: cn=Barbara Jensen, o=University of Michigan, c=US
objectClass: person
cn: Barbara Jensen
cn: Babs Jensen
sn: Jensen
title: the world's most famous manager
mail: bjensen@terminator.rs.itd.umich.edu
uid: bjensen
```

다음 명령을 실행시킨다:

```
ldapadd -f /tmp/entrymods
```

/tmp/newentry 파일이 존재하고 다음 내용을 갖는다고 가정한다:

```
dn: cn=Barbara Jensen, o=University of Michigan, c=US
changetype: delete
```

다음 명령은 Babs Jensen의 엔트리를 삭제한다:

```
ldapmodify -f /tmp/entrymods
```

-f 옵션은 파일(표준 입력대신 파일로부터 수정 정보를 읽는), -b 옵션은 바이너리(입력 파일의 '/'로 시작되는 모든 값들은 바이너리로 해석된다), -r 옵션은 대체(디폴트로 기존 값을 대체한다)를 나타낸다.

제 6 절 부가적 정보와 특징

이 절은 디렉토리를 질의하는데 사용할 수 있는 LDAP 클라이언트인 Netscape Address Book에 관한 정보를 다룬다. 또한 넷스케이프 네비게이터 버전 4.5 또는 그 이상과 LDAP 서버를 이용하여 로우밍 엑세스(roaming access)를 수행하는 방법에 대한 세부사항도 설명한다. 로우밍 엑세스는 다 수행되지는 않기 때문에 OpenLDAP 메일링 리스트에 매우 많이 논의되고 있는데 대부분의 사용자들은 LDAP 서버에 다운로드 및 업로드를 하는 반면 넷스케이프 네비게이터가 LDAP 서버와 함께 작동되는 방식을 좋아하지 않는다. 따라서 절을 읽은 후 로우밍 엑세스가 원하는 방식대로 작동하지 않는다 하더라도 괘념치 말기를 바란다. 많은사람들이 이러한 과정을 이미 거쳐 왔다. 이 절은 사람들에게 LDAP 프로토콜의 가능성에 대한 아이디어를 제공하기 위해 이런 특징을 소개한다. slapd 프로세스를 안전하게 종료하는 것과 slapd 로그에 대한 다소의 정보가 제공된다.

6.1 로우밍 엑세스(Roaming Access)

로우밍 엑세스를 사용하면 넷상의 어디에 있던지 넷스케이프 네비게이터와 LDAP서버를 이용하여 북마크, preference, 메일 필터 등을 가져올 수 있는데 이는 매우 멋진 특징이다. 어디서 웹에 엑세스하던지 브라우저에 대한 고유의 설정을 가질 수 있다고 상상해보라. 여행중에 로컬 북마크에 저장된 통화 사이트에 엑세스할 필요가 있다면 걱정하지 마라. 북마크와 다른설정 파일들을 LDAP 서버에 업로드해서 추후 어느 장소에 있던지 그들을 다 가져올 수 있다.

로우밍 엑세스를 수행하기위해 다음 단계를 따라야 한다:

- slapd.conf 설정파일에 새로운 스키마(schema) 파일을 포함한다
- slapd.conf 설정파일의 데이타베이스 부분에 변경 필드를 설정한다
- 로우밍 엑세스의 사용을 원하는 사용자들에 대한 프로파일 엔트리를 첨가함으로써 Ldif 파일을 변경한다
- LDAP 서버를 로우밍 엑세스 서버로 사용하기 위해 넷스케이프 네비게이터를 설정한다
- 새로운 설정 사항으로 LDAP 서버를 재시작한다

- 새로운 스키마 파일 포함하기: 밑의 부분을 복사 및 붙여넣기 한후 .schema 확장자를 갖는 텍스트 파일로 저장한다. 대개 이 파일은 /usr/local/etc/openldap/schema 디렉토리내에 저장될 것이다. 원한다면 파일을 `<http://home.kabelfoon.nl/~hvdkooij/mull.schema>`로부터 다운받을 수 있다. slapd.conf 파일에 다음과 같이 core.schema 정의 파일을 포함해야 함을 명심하라:

```
include /usr/local/etc/schema/core.schema

#       이 스키마는 core 스키마가 적재되는 것을 전제로 한다

# 넷스케이프 로우밍 프로파일 정보를 OpenLDAP v2 내로 저장하는데 사용
# 이는 실제 프로파일 이름을 데이타베이스내로 저장한다.
attributeType ( 1.3.6.1.4.1.7081.1.1.1
        NAME 'nsLIProfileName'
        DESC 'Store Netscape Roaming Profile name'
        EQUALITY caseIgnoreMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# 넷스케이프 로우밍 프로파일 정보를 OpenLDAP v2 내로 저장하는데 사용
attributeType ( 1.3.6.1.4.1.7081.1.1.2
        NAME 'nsLIPrefs'
        DESC 'Store Netscape Roaming Profile preferences'
        EQUALITY caseExactIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

# 넷스케이프 로우밍 프로파일 정보를 OpenLDAP v2 내로 저장하는데 사용
attributeType ( 1.3.6.1.4.1.7081.1.1.3
        NAME 'nsLIElementType'
        DESC ''
        EQUALITY caseIgnoreMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# 넷스케이프 로우밍 프로파일 정보를 OpenLDAP v2 내로 저장하는데 사용
attributeType ( 1.3.6.1.4.1.7081.1.1.4
        NAME 'nsLIData'
        DESC 'Store the actual data blocks'
```

```
        EQUALITY bitStringMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# 넷스케이프 로우밍 프로파일 정보를 OpenLDAP v2 내로 저장하는데 사용
attributeType ( 1.3.6.1.4.1.7081.1.1.5
        NAME 'nsLIVersion'
        DESC 'Store Netscape Roaming Profile version'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

# 넷스케이프 로우밍 프로파일 정보를 OpenLDAP v2 내로 저장하는데 사용
# 이는 로우밍 프로파일의 기본 사용자로 LDAP 데이타베이스내에 정보를
# 저장하기전에 생성되어야 한다.
objectClass ( 1.3.6.1.4.1.7081.1.2.1
        NAME 'nsLIProfile'
        DESC 'Base holder of the NetScape Roaming Profile'
        SUP top
        MUST ( objectClass $ nsLIProfileName )
        MAY ( nsLIPrefs $ uid $ owner )
        )

# 넷스케이프 로우밍 프로파일 정보를 OpenLDAP v2 내로 저장하는데 사용
# 이 객체 클래스는 실제 데이타를 저장할 것이다.
objectClass ( 1.3.6.1.4.1.7081.1.2.2
        NAME 'nsLIProfileElement'
        DESC 'Contains the actual Roaming Profile data'
        SUP top
        MUST ( objectClass $ nsLIElementType )
        MAY ( owner $ nsLIData $ nsLIVersion )
        )

# EOF
```

- 변경 필드 설정하기: 넷스케이프가 프로파일 데이타의 지역적 복사본을 LDAP 서버와 비교할 수 있음을 확인하기 위해 데이타베이스내의 변경 시간을 설정할 필요가 있다. slapd.conf 파일의 데이타베이스 부분에 다음과 같은 간단한 라인을 첨가하는 것으로 충분하다:

```
lastmod on
```

- Ldif 파일 변경하기: 넷스케이프의 로우밍 엑세스 특징을 이용하려고 하는 각 사용자들은 Ldif 파일에 프로파일 엔트리를 필요로 한다. 프로파일 엔트리를 갖는 간단한 Ldif 파일의 예를 보라:

```
dn: o=myOrg,c=NL
o: myOrg
objectclass: organization

dn: cn=seallers,ou=People,o=myOrg,c=NL
cn: seallers
userpassword: myPassword
objectclass: top
objectclass: person

dn: nsLIProfileName=seallers,ou=Roaming,o=myOrg,c=NL
nsLIProfileName: seallers
```

```
owner: cn=seallers,ou=People,o=myOrg,c=NL
objectclass: top
objectclass: nsLIProfile
```

이 엔트리들은 5.4 (ldapadd) 프로그램을 이용하여 추가할 수 있다. 아마도 단지 로우밍 프로파일 (dn: nsLIProfileName=...)과 일치하는 엔트리만 추가할 필요가 있을 것이다.

- 넷스케이프 네비게이터 설정하기: 다음 단계는 LDAP 서버에 대한 로우밍 엑세스가 작동되도록 넷스케이프를 설정하는 것이다. 다음 순서를 따르라:

Go to Menu Edit -> Preferences -> Roaming User

이 옵션에 해당하는 체크박스를 클릭함으로써 이 프로파일에 대해 처음으로 로우밍 엑세스를 작동시켜야 한다.

적절한 값을 username 박스에 쓰는데 이는 LDIF 파일의 사용자 프로파일 엔트리의 nsLIProfileName= 부분과 일치해야 한다. 예: sealers

로우밍 엑세스의 하위 옵션을 보기 위해 Preferences Window 왼편에 있는 로우밍 사용자 옵션 화살표를 클릭한다.

서버 정보를 클릭하고 LDAP 서버 옵션을 활성화시키며 다음 정보를 박스에 쓴다.

Address: ldap://myHost/nsLIProfileName=$USERID,ou=Roaming,o=myOrg,c=NL

User DN: cn=$USERID,ou=People,o=myOrg,c=NL

IMPORTANT: 넷스케이프는 브라우저를 실행시키기 전에 자동적으로 $USERID를 선택한 프로파일 이름으로 대체한다. 그래서 프로파일 sealler을 선택하면 $USERID를 sealler로 대체하고 프로파일 gonzales를 선택하면 $USERID를 gonzales로 대체한다. 프로파일에 익숙치 않다면 넷스케이프 커뮤니케이터 패키지에 있는 프로파일 매니저 어플리케이션을 실행시켜라. 이는 동일 머신에서 여러사람이 브라우저를 사용할 수 있도록 설계된 어플리케이션으로 브라우저에 대해 각자 자신의 고유한 설정을 가질 수 있다.

마지막은 서버를 재시작하는 것이다. 이를 안전하게 하는 방법과 다시 시작하는 방법은 각각 4.2 (4.2절)과 4 (4절)을 보길 바란다.

6.2 넷스케이프 주소록

LDAP 서버를 구동하고 있다면 여러가지 많은 클라이언트(예를들어 ldapsearch command line utiltity)를 이용하여 서버에 엑세스할 수 있는데 매우 흥미로운 것이 넷스케이프 주소록이다. 이는 넷스케이프 4.x 버전부터 이용할 수 있지만 LDAP 서버와의 안정된 상호운영을 위해서는 4.5 또는 그 이상의 버전을 사용해야 한다.

다음 순서를 따르라:

Open Netscape Navigator -> Go to Communicator Menu -> Address Book

넷스케이프 주소록은 어떤 default LDAP 디렉토리와 함께 시작될 수 있는데 각자의 LDAP 디렉토리를 추가해야 한다.

Go to File Menu -> New Directory

서버 정보를 박스에 쓴다. 예를들면:

- Description: TUDelft

- LDAP Server: dutedin.et.tudelft.nl

- Server Root: o=TUDelft, c=NL

Default LDAP 포트는 389인데 서버를 구축할때 이 옵션을 변경하지 않았다면 이를 변경하지말기 바란다.

box Show Names Containing를 이용해 간략한 질의 또는 Search for button을 이용해 진보된 질의를 해봐라.

6.3 LDAP Migration Tools

LDAP 이동 도구는 설정 파일을 LDIF 포맷으로 변환하는데 사용되는 펄 스크립트 모음인데 PADL 소프트웨어 회사에서 제공된다. 저자는 이를 사용하기전에, 자유로이 사용할 수 있음에도, 라이센스를 읽어 보길 권한다. 사용자를 인증하기 위해 LDAP 서버를 이용할 계획이라면 이 도구는매우 유용할 것이다. NIS 또는 password 아카이브들을 LDAP 서버와 호환되게 하는 LDIF 포맷으로 변환하기 위해 이동 도구를 사용하라. 또한 사용자, 그룹, 앨리어스(aliases), 호스트, 넷그룹, 프로토콜, RPCs와 서비스를 기존 네임 서비스(NIS, flat 파일과 NetInfo)로부터 LDIF 포맷으로 이동하기 위해 스크립트를 적용하라. LDAP 이동 도구의 다운로드와 더욱 많은 정보를 얻기 위해서 다음 주소로 가라:

`<http://www.padl.com/tools.html>`

패키지에 README 파일이있는데 스크립트 파일 이름은 직관적인데 우선 README 파일을 읽고 난후 스크립트를 실행하길 바란다.

6.4 LDAP를 사용한 인증

LDAP 서비스를 사용하기 위해 LDAP 클라이언트는 서비스에 대해 인증을 받아야한다. 즉, 클라이언트가 보기 및 작업이 허용된 것이 무엇인지를 서버가 결정할 수 있도록 클라이언트는 서버에게 데이타를 엑세스하려고 한다라고 말을 해야한다. 클라이언트가 LDAP 서버에 대해 성공적으로 인증받는다면 서버가 후에 클라이어트로부터 요청을 받을때 클라이언트가 요청을 수행하도록 허용되었는지 여부를 검사할 것이다. 이 프로세스를 엑세스 제어라 한다.

LDAP에서 인증은 "bind" 연산에서 지원되는데 Ldapv3는 anonymous, simple 및 SASL의 세가지 유형의 인증을 지원한다."bind" 연산없이 LDAP 요청을 보내는 클라이언트는 anonymous 클라이언트로 처리된다. Simple 인증은 LDAP 서버에 클라이언트(사용자)의 FQDN(Fully Qualified Domain Name)과 암호화되지 않은 패스워드를 보내는 것으로 이루어진다. 이 기구는 패스워드를 네트워크상에서 읽을 수 있기 때문에 보안 문제를 갖고 있다. 이러한 패스워드 노출을 피하기 위해 LDAP 서버에 의해 지원된다면 SSL과 같은 암호화된 채널내에서 simple 인증 기구를 사용할 수 있다.

마지막으로 SASL은 Simple Authentication and Security Layer (RFC 2222)로 인증 및 그 다음의 통신이 수행되는 보안 계층의 확립을 위해 데이타가 클라이언트와 서버간에 교환이 이루어지는 요구-응답 (challenge-response) 프로토콜을 지정한다. SASL을 사용함으로써 LDAP는 LDAP 클라이언트와 서버에 의해 합의된 모든 유형의 인증을 지원할 수 있다. SASL 사용은 Cyrus SASL 라이브러리의 설치가 중요하기 때문에 이 하우투 문서의 다음 버전에 설명될 것이다.

더구나 디렉토리 트리의 정보를 엑세스하는 사용자를 인증함과 동시에 LDAP 서버는 다른 서비스 (Sendmail, Login, Ftp, 등등)에 대해 사용자를 인증할 수 있다. 이는 특정 사용자 정보를 LDAP 서버로 옮겨 PAM (Pluggable Authentication Module) 기구를 사용하여 수행된다.

유닉스 초창기 이후로 사용자 인증은 사용자가 패스워드를 입력하고 입력된 패스워드를 시스템이 /etc/passwd 파일에 저장되어 있는 암호화된 공식 패스워드에 해당하는지 검사하는 방법을 통해 이루어졌다. 이러한 방법은 초창기에 행해졌는데 그 후 /etc/passwd 파일의 보다 복잡한 대체 및 스마트 카드라 불리는 하드웨어 디바이스를 포함하여 인증하는 많은 새로운 방법이 통용되었다. 그러나 새로운 인증 스키마가 개발될때마다 모든 필요한 프로그램(login, ftp 등)이 이를 지원하기 위해 새로 작성되어야 하는 것이 문제점이여다. PAM은 인증 계획에 상관없이 프로그램을 개발할 수 있는 방법을 제공한다. 이러한 프로그램들이 작동하기 위해 런타임시 프로그램에 부착되는 인증 모듈을 필요로 한다.

LDAP에 대한 인증 모듈은 다음 주소에서 tar ball 형태로 이용할 수 있다:

`<http://www.padl.com/pam_ldap.html>`

저자는 리눅스 배포판에 PAM이 설치되어 있다고 가정한다. 이렇지 않은 경우 다음 주소 `<http://www.kernel.org/pub/linux/libs/pam>` 보길 바란다. 다양한 리눅스 배포판들은 PAM과 관련하여 서로 다른

표준 설정을 사용한다. 대개 PAM 설정 파일은 /etc/pam.d 디렉토리내에 존재한다. 이 디렉토리에서 리눅스 박스에서 운영되고 있는 각 서비스에 대한 파일을 발견할 수 있다. 예를들어 리눅스가 부팅된 후 사용자들의 로그인에 LDAP 서버를 사용하길 원한다면 LDAP PAM 모듈을설치하고 /etc/pam.d 디렉토리내 login 파일에 다음 내용을 편집하길바란다:

```
#%PAM-1.0
auth          required     /lib/security/pam_securetty.so
auth          required     /lib/security/pam_nologin.so
auth          sufficient   /lib/security/pam_ldap.so
auth          required     /lib/security/pam_unix_auth.so try_first_pass
account       sufficient   /lib/security/pam_ldap.so
account       required     /lib/security/pam_unix_acct.so
password      required     /lib/security/pam_cracklib.so
password      required     /lib/security/pam_ldap.so
password      required     /lib/security/pam_pwdb.so use_first_pass
session       required     /lib/security/pam_unix_session.so
```

6.5 그래픽 LDAP 도구

- Kldap

  Kldap는 KDE 환경의 그래픽 LDAP 클라이언트로 우수한 인터페이스를 갖으며 디렉토리에 저장된 모든 정보 트리를 볼 수 있다. 어플리케이션으로부터 약간의 screenshot를 검사할 수 있으며 다음 주소에서 다운로드 받을 수 있다.

  <http://www.mountpoint.ch/oliver/kldap/>

- GQ

  GQ는 보다 간단한 인터페이스를 갖는 다른 그래픽 LDAP 클라이언트로 GNOME 환경을 위해작성되었지만 KDE 환경에서도 작동된다. Kldap도 또한 GNOME 환경에서 작동한다. 다운로드 및 더욱 많은 정보를 얻기 위해서는 다음 주소로 가길 바란다:

  <http://biot.com/gq/>

6.6 로그

Slapd는 로그를 발생시키기 위해 syslog(8)를 사용하는데 이 유틸리티의 default 사용자는 LOCAL4 이지만 LOCAL0, LOCAL1에서 LOCAL7까지의 값이 허용된다.

로그 발생을 동작시키기 위해 대개 /etc 디렉토리내에있는 syslog.conf 파일을 편집해야 한다.

다음과 같은 라인을 추가하길 바란다:

local4.* /usr/adm/ldalog

이는 syslog에 대해 LOCAL4 default 사용자를 이용할 것이다. 위 라인의 구문에 익숙치않다면 syslog, syslog.conf 와 syslogd man 페이지를 보라. Default 사용자 변경 및 생성되는 로그 레벨을 지정하길 원한다면 slapd를 구동할때 다음 옵션을 지정하길바란다.

-s syslog-level 옵션은 slapd에게 어떤 레벨의 디버깅 보고서가 syslog(8)로 로그되어야 하는 지를 말해준다. 레벨은 메시지의 엄밀정도를 기술하는데 다음의 정열된 리스트(높은 수준에서 낮은 수준)의 키워드이다: emerg, alert, crit, err, warning, notice, info, and debug. 예: slapd -f myslapd.conf -s debug

-l syslog-local-user syslog(8)의 로컬 사용자를 선택한다. LOCAL0, LOCAL1,..., LOCAL7 까지의 값을 가질 수 있다. 디폴트는 LOCAL4 이다. 그러나 이 옵션은 로컬 사용자에게 syslogd(8)를 지원하는 시스템에만 단지 허용된다.

생성된 로그를 보라. 이들은 질의, 갱신, 바인딩 등과 관계된 문제를 해결하는데 상당한 도움을 제공할 것이다.

제 7 절 참고 문헌

On this section you will find additional documentation about LDAP: useful URLs, cool books and definition RFCs.

7.1 URLs

Here are the URLs that contain very useful information about LDAP. From these URLs, this HOWTO was made, so if after reading this document you need more specific information, you probably will find here:

- University of Michigan LDAP Page: <http://www.umich.edu/~dirsvcs/ldap/index.html>
- University of Michigan LDAP Documentation Page: <http://www.umich.edu/~dirsvcs/ldap/doc/>
- OpenLDAP Administrator's Guide: <http://www.openldap.org/doc/admin>
- Manually Implementing Roaming Access: <http://help.netscape.com/products/client/communicator/manual_roaming2.html>
- Customizing LDAP Settings for Communicator 4.5: <http://developer.netscape.com/docs/manuals/communicator/ldap45.htm>
- Introducing to Directory Service (X.500): <http://www.nic.surfnet.nl/surfnet/projects/x500/introducing/>
- Linux Directory Service: <http://www.rage.net/ldap/>

7.2 서적

These are the most popular and useful books about LDAP:

- Implementing LDAP by Mark Wilcox
- LDAP: Programming Directory-Enabled Applications with Lightweight Directory Access Protocol by Howes and Smith
- Understanding and Deploying LDAP Directory Servers by Howes, Smith, and Good

7.3 RFCs

The RFCsw that support the LDAP development efforts:

- RFC 1558: A String Representation of LDAP Search Filters
- RFC 1777: Lightweight Directory Access Protocol
- RFC 1778: The String Representation of Standard Attribute Syntaxes
- RFC 1779: A String Representation of Distinguished Names

- RFC 1781: Using the OSI Directory to Achieve User Friendly Naming
- RFC 1798: Connectionless LDAP
- RFC 1823: The LDAP Application Programming Interface
- RFC 1959: An LDAP URL Format
- RFC 1960: A String Representation of LDAP Search Filters
- RFC 2251: Lightweight Directory Access Protocol (v3)
- RFC 2307: LDAP as a Network Information Service RFC 1558: A String Representation of LDAP Search Filters