# LDAP

Łukasz Zubkowicz

25 listopada 2009

# W skrócie

- *Lightweight Directory Access Protocol*

# W skrócie

- *Lightweight Directory Access Protocol*
- standard **IETF** (*The Internet Engineering Task Force*)
- otwarta specyfikacja (**RFC4510**, ..., **RFC4519**)

# W skrócie

- *Lightweight Directory Access Protocol*
- standard **IETF** (*The Internet Engineering Task Force*)
- otwarta specyfikacja (**RFC4510**, ..., **RFC4519**)
- uproszczona wersja **DAP** (wzbogacona o obsługę **TCP/IP**)

# W skrócie

- *Lightweight Directory Access Protocol*
- standard **IETF** (*The Internet Engineering Task Force*)
- otwarta specyfikacja (**RFC4510**, ..., **RFC4519**)
- uproszczona wersja **DAP** (wzbogacona o obsługę **TCP/IP**)
- protokół stanowy, utrzymujący połączenie

# W skrócie

- *Lightweight Directory Access Protocol*
- standard **IETF** (*The Internet Engineering Task Force*)
- otwarta specyfikacja (**RFC4510**, ..., **RFC4519**)
- uproszczona wersja **DAP** (wzbogacona o obsługę **TCP/IP**)
- protokół stanowy, utrzymujący połączenie
- architektura klient-serwer

# Przeznaczenie

Zapewnienie dostępu do usług katalogowych.

- zdefiniowanie struktury katalogu (opartej na **OSI X.500**)

# Przeznaczenie

Zapewnienie dostępu do usług katalogowych.

- zdefiniowanie struktury katalogu (opartej na **OSI X.500**)
- opis metod dostępu do danych

# Przeznaczenie

Zapewnienie dostępu do usług katalogowych.

- zdefiniowanie struktury katalogu (opartej na **OSI X.500**)
- opis metod dostępu do danych
- wyspecyfikowanie zabezpieczeń

# Pojęcie katalogu

Wyspecjalizowana forma bazy danych.

- założenie: organizowanie i porządkowanie informacji

# Pojęcie katalogu

Wyspecjalizowana forma bazy danych.

- założenie: organizowanie i porządkowanie informacji
- elastyczność (ze względu na chęć katalogowania "wszystkiego")

# Pojęcie katalogu

Wyspecjalizowana forma bazy danych.

- założenie: organizowanie i porządkowanie informacji
- elastyczność (ze względu na chęć katalogowania "wszystkiego")
- optymalizacja pod kątem wyszukiwania i szybkości odczytu

# Pojęcie katalogu

Wyspecjalizowana forma bazy danych.

- założenie: organizowanie i porządkowanie informacji
- elastyczność (ze względu na chęć katalogowania "wszystkiego")
- optymalizacja pod kątem wyszukiwania i szybkości odczytu
- wyrafinowane metody filtrowania danych

# Atrybut

Przykład atrybutu wraz z wartościami:

```
cn: Piotr Waldemar Kowalski
cn: Piotr W. Kowalski
cn: Piotr Kowalski
```

Odpowiednik pola tabeli z baz relacyjnych.

- globalnie unikalne nazwa i OID (*Object Identifier*)

# Atrybut

Przykład atrybutu wraz z wartościami:

```
cn: Piotr Waldemar Kowalski
cn: Piotr W. Kowalski
cn: Piotr Kowalski
```

Odpowiednik pola tabeli z baz relacyjnych.

- globalnie unikalne nazwa i OID (*Object Identifier*)
- możliwość dopuszczenia wielu wartości

# Atrybut

Przykład atrybutu wraz z wartościami:

```
cn: Piotr Waldemar Kowalski
cn: Piotr W. Kowalski
cn: Piotr Kowalski
```

Odpowiednik pola tabeli z baz relacyjnych.

- globalnie unikalne nazwa i OID (*Object Identifier*)
- możliwość dopuszczenia wielu wartości
- składnia (*syntax*), odpowiednik typu danych (definiuje wyrażenie regularne weryfikujące wartość atrybutu)

# Atrybut

Przykład atrybutu wraz z wartościami:

```
cn: Piotr Waldemar Kowalski
cn: Piotr W. Kowalski
cn: Piotr Kowalski
```

Odpowiednik pola tabeli z baz relacyjnych.

- globalnie unikalne nazwa i OID (*Object Identifier*)
- możliwość dopuszczenia wielu wartości
- składnia (*syntax*), odpowiednik typu danych (definiuje wyrażenie regularne weryfikujące wartość atrybutu)
- zasady porównywania (*matching*), używane przy wyszukiwaniu

# Atrybut

Przykład atrybutu wraz z wartościami:

```
cn: Piotr Waldemar Kowalski
cn: Piotr W. Kowalski
cn: Piotr Kowalski
```

Odpowiednik pola tabeli z baz relacyjnych.

- globalnie unikalne nazwa i OID (*Object Identifier*)
- możliwość dopuszczenia wielu wartości
- składnia (*syntax*), odpowiednik typu danych (definiuje wyrażenie regularne weryfikujące wartość atrybutu)
- zasady porównywania (*matching*), używane przy wyszukiwaniu
- opis

# Wpis (*entry*)

Przykład wpisu:

```
dn: ou=seminaria,dc=mimuw,dc=edu,dc=pl
objectClass: top
objectClass: organizationalUnit
```

Odpowiednik rekordu z baz relacyjnych.

- obiekt pewnej ilości klas
- klasa to zbiór atrybutów (podzielonych na obowiązkowe i opcjonalne)

# Wpis (*entry*)

Przykład wpisu:

```
dn: ou=seminaria,dc=mimuw,dc=edu,dc=pl
objectClass: top
objectClass: organizationalUnit
```

Odpowiednik rekordu z baz relacyjnych.

- obiekt pewnej ilości klas
- klasa to zbiór atrybutów (podzielonych na obowiązkowe i opcjonalne)
- niejawne dziedziczenie po *top*, wymaga zdefiniowania wartości dla atrybutów `dn` (*Distinguished Name*) oraz `objectClass`

# Wpis (*entry*)

Przykład wpisu:

```
dn: ou=seminaria,dc=mimuw,dc=edu,dc=pl
objectClass: top
objectClass: organizationalUnit
```

Odpowiednik rekordu z baz relacyjnych.

- obiekt pewnej ilości klas
- klasa to zbiór atrybutów (podzielonych na obowiązkowe i opcjonalne)
- niejawne dziedziczenie po *top*, wymaga zdefiniowania wartości dla atrybutów `dn` (*Distinguished Name*) oraz `objectClass`
- ułożenie klas w hierarchię (drzewo z jednym korzeniem)

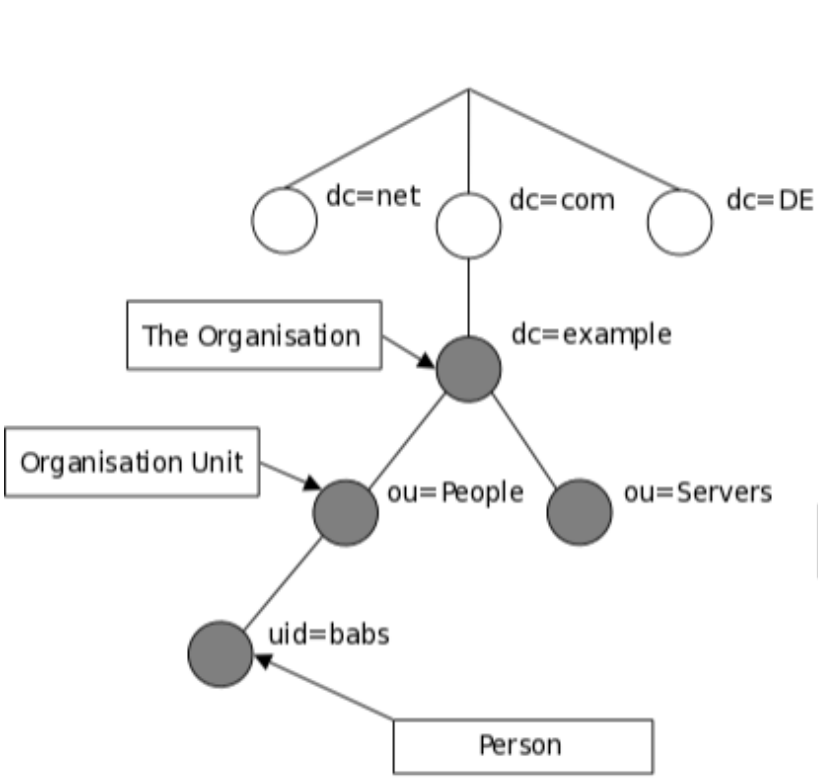
# Schema

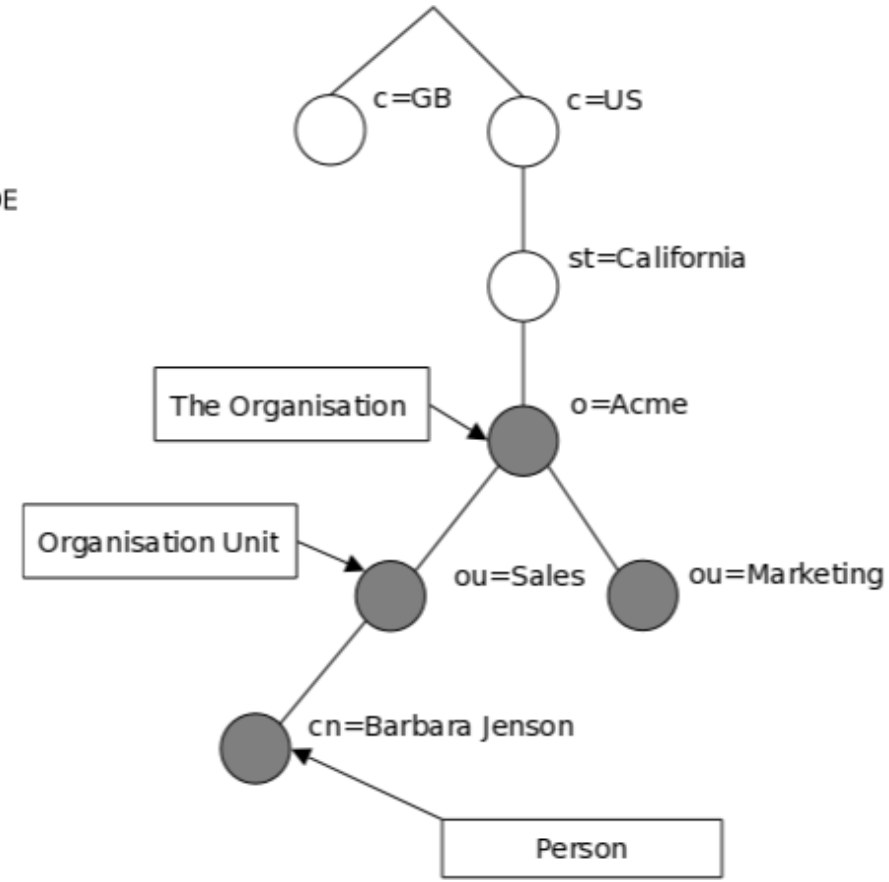
- rozszerzalny zbiór danych o atrybutach (czyli też składniach, zasadach porównywania) i klasach obiektów

# Schema

- rozszerzalny zbiór danych o atrybutach (czyli też składniach, zasadach porównywania) i klasach obiektów
- możliwość definiowania własnych atrybutów i klas

# DIT (*Directory Information Tree*)




(z podręcznika do OpenLDAP, *http://www.openldap.org*)

# DIT (*Directory Information Tree*)

- każdy węzeł to dokładnie jeden wpis (zbiór par atrybut-wartość)

# DIT (*Directory Information Tree*)

- każdy węzeł to dokładnie jeden wpis (zbiór par atrybut-wartość)
- RDN (*Relative Distinguished Name*) to nazwa węzła, która tak naprawdę jest jedną z par atrybut-wartość (np. `ou=Marketing`)

# DIT (*Directory Information Tree*)

- każdy węzeł to dokładnie jeden wpis (zbiór par atrybut-wartość)
- RDN (*Relative Distinguished Name*) to nazwa węzła, która tak naprawdę jest jedną z par atrybut-wartość (np. `ou=Marketing`)
- RDN musi być unikalny wśród rodzeństwa węzła, ale nie musi być tego samego typu (np. `cn=Manager` i `ou=Sales` mogą mieć wspólnego rodzica)

# DIT (*Directory Information Tree*)

- każdy węzeł to dokładnie jeden wpis (zbiór par atrybut-wartość)
- RDN (*Relative Distinguished Name*) to nazwa węzła, która tak naprawdę jest jedną z par atrybut-wartość (np. `ou=Marketing`)
- RDN musi być unikalny wśród rodzeństwa węzła, ale nie musi być tego samego typu (np. `cn=Manager` i `ou=Sales` mogą mieć wspólnego rodzica)
- ciąg nazw RDN po ścieżce do korzenia jednoznacznie identyfikuje węzeł

# DIT (*Directory Information Tree*)

- każdy węzeł to dokładnie jeden wpis (zbiór par atrybut-wartość)
- RDN (*Relative Distinguished Name*) to nazwa węzła, która tak naprawdę jest jedną z par atrybut-wartość (np. `ou=Marketing`)
- RDN musi być unikalny wśród rodzeństwa węzła, ale nie musi być tego samego typu (np. `cn=Manager` i `ou=Sales` mogą mieć wspólnego rodzica)
- ciąg nazw RDN po ścieżce do korzenia jednoznacznie identyfikuje węzeł
- wyżej wymieniony ciąg oddzielony przecinkami to DN (*Distinguished Name*), np.
  `dn: ou=seminaria,dc=mimuw,dc=edu,dc=pl`

# LDIF (*LDAP Data Interchange Format*)

- plik tekstowy listujący wpisy z katalogu

# LDIF (*LDAP Data Interchange Format*)

- plik tekstowy listujący wpisy z katalogu
- każda para `atrybut: wartość` musi zajmować całą linię
- kolejność atrybutów jest częściowo ustalona (najpierw `dn`, potem `objectClass`, za nimi reszta w dowolnej kolejności)

# LDIF (*LDAP Data Interchange Format*)

- plik tekstowy listujący wpisy z katalogu
- każda para `atrybut: wartość` musi zajmować całą linię
- kolejność atrybutów jest częściowo ustalona (najpierw `dn`, potem `objectClass`, za nimi reszta w dowolnej kolejności)
- znak # (jako pierwszy w nowej linii) rozpoczyna komentarz
- linia zaczynająca się od białego znaku to kontynuacja poprzedniej linii (także w przypadku komentarza)
- pusta linia oddziela poszczególne wpisy

# LDIF (*LDAP Data Interchange Format*)

Fragment przykładowego pliku:

```
dn: ou=proseminaria,dc=mimuw,dc=edu,dc=pl
objectClass: organizationalUnit
description: Rok III

dn: ou=seminaria,dc=mimuw,dc=edu,dc=pl
objectClass: organizationalUnit
description: Lata IV i V

# najfajniejsze seminarium :)
        bo nasze :p
dn: ou=Systemy Rozproszone,ou=seminaria,dc=mimuw,dc=edu,dc=pl
objectClass: organizationalUnit

dn: ou=przedmioty,dc=mimuw,dc=edu,dc=pl
objectClass: organizationalUnit

dn: ou=monograficzne,ou=przedmioty,dc=mimuw,dc=edu,dc=pl
objectClass: organizationalUnit

dn: ou=obieralne,ou=przedmioty,dc=mimuw,dc=edu,dc=pl
objectClass: organizationalUnit
```

# Zastosowania katalogów

- autentykacja, autoryzacja
- adresy, telefony, zasoby
- dane o zbiorach (grupy użytkowników, listy dyskusyjne)
- konfiguracje (sprzętu, oprogramowania)
- reprezentacja struktury (organizacji, kapitału)
- ...

# W skrócie

- projekt zapoczątkowany przez **University of Michigan** (twórców LDAP)
- otwarty kod, własna licencja

# W skrócie

- projekt zapoczątkowany przez **University of Michigan** (twórców LDAP)
- otwarty kod, własna licencja
- referencyjna implementacja protokołu LDAP
- w pełni wystarczająca do pracy w środowisku produkcyjnym

# W skrócie

- projekt zapoczątkowany przez **University of Michigan** (twórców LDAP)
- otwarty kod, własna licencja
- referencyjna implementacja protokołu LDAP
- w pełni wystarczająca do pracy w środowisku produkcyjnym
- skład: serwer usług katalogowych *slapd*, narzędzia, biblioteki

# Cechy serwera

- LDAPv3

# Cechy serwera

- LDAPv3
- wieloplatformowy, wsparcie dla Unicode

# Cechy serwera

- LDAPv3
- wieloplatformowy, wsparcie dla Unicode
- SASL (*Simple Authentication and Security Layer*)
- TLS (*Transport Layer Security*)

# Cechy serwera

- LDAPv3
- wieloplatformowy, wsparcie dla Unicode
- SASL (*Simple Authentication and Security Layer*)
- TLS (*Transport Layer Security*)
- możliwość wyboru bazy danych, np. BDB, HDB (obie oparte na Oracle Berkeley DB), pliki tekstowe (LDIF)
- każdy katalog może używać innej bazy

# Cechy serwera

- LDAPv3
- wieloplatformowy, wsparcie dla Unicode
- SASL (*Simple Authentication and Security Layer*)
- TLS (*Transport Layer Security*)
- możliwość wyboru bazy danych, np. BDB, HDB (obie oparte na Oracle Berkeley DB), pliki tekstowe (LDIF)
- każdy katalog może używać innej bazy
- możliwość rozszerzania za pomocą modułów (dzięki *Generic modules API*), istnieje wiele gotowych rozszerzeń

# Cechy serwera

- LDAPv3
- wieloplatformowy, wsparcie dla Unicode
- SASL (*Simple Authentication and Security Layer*)
- TLS (*Transport Layer Security*)
- możliwość wyboru bazy danych, np. BDB, HDB (obie oparte na Oracle Berkeley DB), pliki tekstowe (LDIF)
- każdy katalog może używać innej bazy
- możliwość rozszerzania za pomocą modułów (dzięki *Generic modules API*), istnieje wiele gotowych rozszerzeń
- wielowątkowy, możliwość pracy w trybie proxy, wysoce konfigurowalny

# Replikacja

- minimalizuje wpływ awarii (rozwiązanie z serwerami zapasowymi)

# Replikacja

- minimalizuje wpływ awarii (rozwiązanie z serwerami zapasowymi)
- skraca czas dostępu do danych (każdy dział firmy może mieć swoją kopię katalogu, która będzie “bliżej”)

# Replikacja

- minimalizuje wpływ awarii (rozwiązanie z serwerami zapasowymi)
- skraca czas dostępu do danych (każdy dział firmy może mieć swoją kopię katalogu, która będzie “bliżej”)
- replikacji nie musi podlegać cały katalog (tylko np. gałąź z książką adresową)

# Replikacja

- minimalizuje wpływ awarii (rozwiązanie z serwerami zapasowymi)
- skraca czas dostępu do danych (każdy dział firmy może mieć swoją kopię katalogu, która będzie “bliżej”)
- replikacji nie musi podlegać cały katalog (tylko np. gałąź z książką adresową)
- synchronizacja działa na zasadzie producent-konsument, do wyboru metody push lub pull (producent powiadamia konsumentów o zmianach w katalogu lub konsumenci odpytują o nie co jakiś czas)

# Kontrola dostępu

ACL (*Access Control Lists*)

- wszechmogący administrator

# Kontrola dostępu

ACL (*Access Control Lists*)

- wszechmogący administrator
- użytkowników można poprzydzielać do węzłów, wyspecyfikować poprzez wyrażenie regularne

# Kontrola dostępu

ACL (*Access Control Lists*)

- wszechmogący administrator
- użytkowników można poprzydzielać do węzłów, wyspecyfikować poprzez wyrażenie regularne
- istnieją różne prawa: odczyt, zapis, wyszukiwanie, ...

# Kontrola dostępu

ACL (*Access Control Lists*)

- wszechmogący administrator
- użytkowników można poprzydzielać do węzłów, wyspecyfikować poprzez wyrażenie regularne
- istnieją różne prawa: odczyt, zapis, wyszukiwanie, ...
- dostęp można określać dla całych wpisów jak i poszczególnych atrybutów

# Żywy przykład: książka adresowa

Chcecie go zobaczyć?

# Dziękuję za uwagę

Czy są jakieś pytania?

# Bibliografia


- http://www.openldap.org/doc/admin24
- http://ldap.akbkhome.com
- http://www.ldapman.org/articles/intro_to_ldap.html
- http://www.ldapman.org/articles/attributes.html
- http://www.ldapman.org/articles/tree_design.html
- http://onlamp.com/pub/a/onlamp/2001/08/16/ldap.html
- http://linuxdevcenter.com/pub/a/linux/2001/11/08/ldap.html
- http://onlamp.com/pub/a/onlamp/2002/10/17/essentialsysadmin.html
- http://onlamp.com/pub/a/onlamp/2003/03/27/ldap_ab.html