Implementace LDAP serveru do předmětu ISA, ak. r. 2023

Adam Gabrys, xgabry01

10.11.2023

Obsah

1 Základní přehled teorie

1.1 Co je to LDAP?

LDAP, neboli *Lightweight Directory Access Protocol*, je protokol navržený pro přístup a úpravy informací v adresářových službách. Jedná se o otevřený standard pro správu distribuovaných informací, a to zejména pro ukládání informací o uživatelích, zařízeních a dalších objektech v síti.

1.2 Základní Principy LDAP

1. **Adresářová Struktura:** LDAP udržuje informace ve formě hierarchické adresářové struktury. Každý objekt v adresáři je identifikován unikátním názvem nazývaným *Distinguished Name* (DN).

2. **Schéma:** Definuje typy objektů a atributy, které mohou být v adresáři uloženy. Schéma je klíčové pro validaci a strukturování dat.

3. **Operace:** LDAP definuje operace pro vyhledávání, zápis, aktualizaci a mazání dat v adresáři. Tyto operace jsou posílány pomocí LDAP příkazů. V tomto projektu je řešeno pouze vyhledávání.

Pro sekci 1.2 a 1.1 jsem čerpal ze zdrojů: [7] [6]

1.3 Použitá verze LDAP

LDAPv2 je ranou verzí protokolu LDAP. Poskytuje standardizovanou metodu přístupu k informacím adresáře a jejich správy. Organizacím se doporučuje přejít na LDAPv3 kvůli lepší funkčnosti a souladu s moderními standardy, *ovšem pro tento projekt byla naimplementována verze LDAPv2 dostačující.*

Pro tuto sekci jsem čerpal ze zdroje: [8]

1.4 TCP

TCP (*Transmission Control Protocol*) hraje v LDAP (*Lightweight Directory Access Protocol*) klíčovou roli, protože LDAP využívá TCP jako transportní protokol pro komunikaci mezi klientem a serverem.

1. **Transportní Protokol:** LDAP běžně využívá TCP jako svůj transportní protokol. TCP poskytuje spojovaný a spolehlivý přenos dat, což je důležité pro zajištění integritu a doručení zpráv.

2. **Porty pro LDAP:** Standardní port pro LDAP komunikaci je 389 pro nešifrované spojení a 636 pro spojení šifrovaná pomocí TLS/SSL. Tyto porty jsou výchozí, ale mohou být změněny konfigurací serveru.

3. **TCP Handshake:** Při navazování spojení mezi klientem a LDAP serverem je realizován TCP handshake, zahrnující tři fáze: odeslání SYN (synchronizačního) signálu, přijetí SYN a odeslání ACK (potvrzovacího) signálu.

4. **Spolehlivý Přenos Dat:** TCP garantuje spolehlivý přenos dat, což je klíčové pro správné fungování LDAP, kde dochází k výměně informací mezi klientem a serverem.

5. **Přenos v Rámci TCP Spojujícího Kanálu:** V rámci navázaného TCP spojení dochází k výměně LDAP zpráv, kódovaných pomocí BER (*Basic Encoding Rules*), jako datových toků přes TCP spojení.

Informace pto tuto sekci jsem čerpal ze zdrojů [5] [6]

1.5 Kódování BER

BER (*Basic Encoding Rules*) je způsob kódování dat pro přenos v počítačových sítích, představující binární kódovací schéma v kontextu LDAP.

1. **Binární Kódování:** BER je binární kódovací formát, který představuje prvky dat jako posloupnost oktetů (bytů).

2. **Struktura Tag-Délka-Hodnota (TLV):** BER organizuje data do trojic TLV, kde každá trojice sestává z tagu, délky a hodnoty. Tato struktura umožňuje kódování a dekódování složitých datových struktur flexibilním způsobem.

3. **LDAP Zprávy:** LDAP zprávy jsou kódovány pomocí BER. Každá LDAP zpráva je vytvořena podle formátu TLV, s konkrétními tagy označujícími typ zprávy a její komponenty.

4. **Efektivita a Kompaktnost:** Binární formát BER je navržen pro efektivitu kódování a dekódování, vytvářející kompaktní reprezentaci dat, což je důležité pro optimalizaci komunikace v síti.

5. **Zpětná Kompatibilita:** Kódování BER v LDAP umožňuje zpětnou kompatibilitu s dřívějšími verzemi protokolu, poskytující standardizovaný způsob reprezentace LDAP zpráv v binární podobě.

6. **Identifikátory Tagů:** Tagy v BER slouží k identifikaci typu kódovaných dat. LDAP specifikuje specifické tagy pro různé typy zpráv a prvků LDAP, zajišťující konzistentní interpretaci napříč implementacemi LDAP.

1.6 Struktura TLV

V kontextu LDAP znamená TLV (*Tag-Length-Value*) struktura základní metodu kódování a reprezentace dat v binární formě. TLV je klíčovým prvkem v BER.

Vysvětlení zkratky TLV:

1. **Tag (Značka):** Označuje typ dat, která jsou kódována. V LDAP jsou definovány specifické tagy pro různé typy zpráv a prvků, umožňující jednotné a srozumitelné kódování.

2. **Length (Délka):** Určuje délku hodnoty (value) v oktetech, umožňující dekodéru efektivně určit, kde končí aktuální prvek a začíná další.

3. **Value (Hodnota):** Obsahuje samotná data, která jsou kódována. Tato část může být libovolné délky a obsahuje informace specifické pro daný prvek (např. hodnotu atributu v LDAP).

Použití TLV v LDAP umožňuje efektivní a flexibilní kódování do binární podoby vhodné pro přenos v síti. Díky této struktuře je možné standardizovaně a účinně reprezentovat a přenášet data mezi LDAP klientem a serverem.

Informace pto sekci 1.5 a 1.6 jsem čerpal ze zdrojů [3] [11], [12].

1.7 Filtry

V rámci LDAPv2 jsou filtry klíčovým prvkem pro vyhledávání informací v adresáři. Zde jsou popisy jednotlivých typů filtrů:

1. **And (0):**
   - **Syntax:** `(& (filter1) (filter2) ...)`
   - **Popis:** Logický "a". Vyhledávání vrátí položky, které vyhovují všem zadaným podmínkám filtrů.
   - **Nebylo implementováno**

2. **Or (1):**
   - **Syntax:** `(| (filter1) (filter2) ...)`
   - **Popis:** Logický "nebo". Vyhledávání vrátí položky, které vyhovují alespoň jedné zadané podmínce filtrů.

- **Nebylo implementováno**

3. **Not (2):**
   - **Syntax:** `(! (filter))`
   - **Popis:** Logický "ne". Vyhledávání vrátí položky, které nesplňují zadanou podmínku filtru.
   - **Nebylo implementováno**
4. **Equality Match (3):**
   - **Syntax:** '(attribute=value)'
   - **Popis:** Vyhledávání vrátí položky, kde hodnota určeného atributu je rovna zadané hodnotě. Například '(cn=xlogin00)' vrátí položky, kde hodnota atributu 'cn' je "xlogin00".
5. **Substrings (4):**
   - **Syntax:** '(attribute=*substring*)'
   - **Popis:** Vyhledávání vrátí položky, kde hodnota určeného atributu obsahuje zadaný podřetězec.
     - Například '(cn=xlogin0*)' vrátí položky, kde hodnota atributu 'cn' obsahuje podřetězec začínající "xlogin0".
     - Například '(cn=*login00)' vrátí položky, kde hodnota atributu 'cn' obsahuje podřetězec končící "login01".
     - Například '(cn=*login0*)' vrátí položky, kde hodnota atributu 'cn' obsahuje podřetězec "login01" kdekoli.

Zdroje o filtrech jsem čerpal ze zdrojů: [9].

1.8 Typy LDAP zpráv

- **BindRequest (Přihlašovací požadavek):** Tato zpráva slouží k autentizaci klienta u LDAP serveru. Ovšem v této implementaci řešíme pouze simple-bind, tedy žádnou autentizaci.
- **BindResponse (Přihlašovací odpověď):** Tato zpráva je odpovědí na přihlašovací požadavek. Obsahuje informace o úspěchu nebo neúspěchu přihlášení.
- **SearchRequest (Vyhledávací požadavek):** Tato zpráva slouží k hledání informací v adresáři. Klient v ní specifikuje kritéria vyhledávání.
- **SearchResEntry (Výsledek hledání - Položka):** Tato zpráva obsahuje informace o položce, která odpovídá kritériím hledání. Výsledky hledání jsou vráceny ve formě těchto položek.

- **SearchResDone (Výsledek hledání - Dokončeno):** Tato zpráva oznamuje konec výsledků hledání. Obsahuje informace o celkovém počtu nalezených položek a návratový kód `Success` či `SizeLimitExceeded`.

Informace pro typy zpráv jsem čerpal ze zdrojů: [10] [11]

2 Struktura programu a popis souborů

2.1 handle_search_request.cpp

V tomto souboru je zpracováno celkové řešení dekompozice a kontroly zprávy `SearchRequest`, následované voláním funkcí `handle_search_res_done` a `handle_search_res_entry`, spolu s funkcemi na filtry.

2.2 handle_search_res_done.cpp

Tento soubor obsahuje zpracování a odesílání zprávy `SearchResDone`. Obsahuje také implementaci funkce `sendSizeLimitExceededMessage`, která informuje klienta o překročení limitu `SizeLimit`.

2.3 handle_search_res_entry.cpp

V tomto souboru se nachází funkce `search_entry`, která je součástí třídy `ldap_functions`. Funkce prochází každým filtrem, aplikuje je a odesílá výsledky vyhledávání klientovi. Pokud počet položek překročí limit velikosti, informuje klienta o překročení tohoto limitu.

2.4 isa-ldapserver-main.cpp

Hlavní vstupní bod LDAP serveru. Obsahuje pouze funkci `main`.

2.5 ldap_filters.cpp

Tento soubor obsahuje dvě hlavní funkce třídy `ldap_functions`:

- `get_filter()`: Zpracovává filtr z klientovy zprávy. Kontroluje typ filtru, získává délku filtru a zpracovává popis a hodnotu atributu. Pro filtry typu `EQUALITY_MATCH` nebo `SUBSTRING` probíhá další zpracování.
- `performSearch(ldap_filters f)`: Provádí vyhledávání v databázi na základě filtru. Pro filtry typu `EQUALITY_MATCH` hledá přesné shody. Pro filtry typu `SUBSTRING` hledá shody pomocí regulárních výrazů. Výsledky vyhledávání jsou vráceny jako sada vektorů řetězců.

2.6 ldap_functions.cpp

Tento soubor obsahuje implementaci třídy `ldap_functions`, která zpracovává LDAP zprávy.

- `Konstruktor ldap_functions`: Inicializuje třídu s klientovým socketem a daty databáze.
- `check_ldap_FSM_state()`: Kontroluje stav konečného automatu LDAP. Čte po bytech z klientovy zprávy a kontroluje, zda je to správný typ LDAP packetu. Pokud je vše v pořádku, volá funkci `choose_ldap_message()`.
- `choose_ldap_message()`: Na základě typu zprávy volá odpovídající funkci pro zpracování zprávy.
- `handleBindRequest()`: Zpracovává `BindRequest` zprávu od klienta. Pokud je vše v pořádku, odesílá `BindResponse` zpět klientovi pomocí funkce `sendBindResponse()`.
- `sendBindResponse()`: Odesílá `BindResponse` zpět klientovi.

2.7 server.cpp

Tento soubor obsahuje definice metod třídy `server` a několik dalších funkcí:

- `sigint_handler(int signum)`: Obslužná rutina pro zacházení se signálem SIGINT.
- `client_handler(void* arg, set<vector<string>> database)`: Funkce pro obsluhu klienta.
- `server::server(int port)`: Konstruktor třídy `server`, který nastavuje server.
- `server::parse_database(string input_file)`: Metoda pro zpracování databáze.
- `server::connect_clients()`: Metoda pro připojení klientů.
- `server::trim(string s)`: Metoda pro odstranění bílých mezer ze stringu.

Soubor také obsahuje definici proměnné `exit_signal`, která je použita pro zacházení se signálem SIGINT.

2.8 processing_help_functions.cpp

Tento soubor obsahuje pomocné funkce pro třídu `ldap_functions`, které pomáhají při zpracování LDAP zpráv.

- `next_byte(int client_message, size_t amount)`: Čte určitý počet bytů z klientovy zprávy.

- `next_byte_content_equals_to(int hex_value)`: Kontroluje, zda je obsah příštího bytu roven dané hexadecimální hodnotě.
- `this_byte_content_equals_to(int hex_value)`: Kontroluje, zda je obsah aktuálního bytu roven dané hexadecimální hodnotě.
- `next_byte_content_bigger_than(int hex_value)`: Kontroluje, zda je obsah příštího bytu větší než daná hexadecimální hodnota.
- `get_mess_length()`: Získává délku zprávy.
- `get_base_content(int base_content_legth)`: Získává obsah base_objectu.
- `get_string(int length)`: Získává řetězec z klientovy zprávy.
- `get_limit()`: Získává limit z klientovy zprávy.
- `debug_print_constructed_response(int bind_data_length, char* bind_response)`: Tiskne debugovací informace o odeslané odpovědi.
- `LV_string(string s)`: Převádí řetězec na formát LV (Length Value).
- `LV_id(int num)`: Převádí číslo na formát LV (Length Value).

2.9 Hlavičkové soubory

2.9.1 ldap_functions.hpp

Obsahuje deklaraci třídy `ldap_functions`, která je zodpovědná za zpracování LDAP zpráv.

2.9.2 server.hpp

Obsahuje deklaraci třídy `server`, která je zodpovědná za nastavení a správu serveru.

- `server(int port)`: Konstruktor třídy, který nastavuje server.
- `connect_clients()`: Vytváří vlákna a navazuje spojení s klienty.
- `parse_database(string input_file)`: Zpracovává data z CSV souboru.
- `trim(string s)`: Odstraňuje bílé mezery ze stringu.
- `client_handler`: Deklarace funkce, která zpracovává více klientů najednou a volá funkci `handleBindRequest`.
- `SIGINT handling`: Definice pro zacházení se signálem SIGINT.

3 Zajímavé úseky kódu

Server byl implementován v jazyce C++. Popis jednotlivých funkcí a vytvořených maker je v hlavičkových souborech .h.

- Pro usnadění přehlednosti funkcí a omezení vysvětlování, proč bylo použité dané číslo v hex formátu byla naimplementována makra. Ukázka maker v následujícím úseku kódu.

```
    // ASN TAGS
    #define ASN_TAG_EOC 0x00
    #define ASN_TAG_BOOL 0x01
    #define ASN_TAG_INTEGER 0x02
    #define ASN_TAG_BIT_STRING 0x03
    #define ASN_TAG_OCTETSTRING 0x04
    #define BER_TAG_SEQUENCE 0x30
    #define ASN_TAG_ENUMERATED 0x0A

    // LDAP MESSAGE TYPES
    #define BINDREQUEST 0x60
    #define BINDRESPONSE 0x61
    #define SEARCHREQUEST 0x63
    #define SEARCHRESENTRY 0x64
    #define SEARCHRESDONE 0x65
    #define UNBINDREQUEST 0x42

```

- Makro pro debugging ušetřilo mnoho psaní a zvýšilo přehlednost kódu.

```
    #ifdef DEBUG
        #define DEBUG_PRINT(message) std::cout << message << std::endl
        #define DEBUG_PRINT_BYTE_CONTENT DEBUG_PRINT("byte content: " << hex << byte_content);
    #else
        #define DEBUG_PRINT(message) // Define as nothing when debugging is disabled
        #define DEBUG_PRINT_BYTE_CONTENT DEBUG_PRINT();
    #endif

```

4 Testování

Testování serveru proběhlo na referenčním fakultním serveru merlin.fit.vutbr.cz s testovacím klientem **ldapsearch** dostupným na fakultním serveru eva.fit.vutbr.cz, pro testování s klientem ldapsearch pro verzi LDAPv2 bylo použito referenční prostředí na lokálním zařízení s OS Ubuntu 20.0.4. Testy byly provedeny na obou zařízeních a jsou ve formátu textového výstupu terminálu. Pro testování a vývoj byl také využit program Wireshark a Tshark pro přehledné zobrazení komunikace a detailů paketů. Testy se nachází ve složce `tests` a mají příponu souboru .txt, kdy se na prvním řádku soborů nachází příkaz, který byl pro test použit.

4.1 Použitá databáze

Použitá databáze byla CSV soubor se studenty VUT FIT poskytnutý ve specifikaci zadání projektu [1].

4.2 Použitý klient

Jako klient byl na testování použit nástroj ldapsearch [2] doporučený zadáním.

4.2.1 Důležité přepínače klienta

1. **-x:** Tento přepínač zajišťuje, že bude použit jednoduchý (simple) autentizační mechanismus při komunikaci s LDAP serverem.
2. **-h:** Slouží k určení cílového hostitelského serveru nebo adresy LDAP serveru. Pokud není specifikováno, nástroj `ldapsearch` použije výchozí hodnotu.
3. **-z:** Určuje limit počtu vrácených výsledků (počet položek), což může být užitečné při omezení velikosti odpovědi serveru.
4. **-b:** Nastavuje kořenový (base) DN (Distinguished Name), který specifikuje výchozí umístění pro vyhledávání ve stromu LDAP. Tímto lze omezit rozsah hledání.

Zdroje pro přepínače jsem čerpal odsud: [2]

4.3 jednotlivé testy

4.3.1 `ipv6_loopback_interface.txt`

Co bylo testováno: Byla provedena testovací operace pomocí nástroje `ldapsearch` s následujícími parametry:

- Příkaz: `ldapsearch -H ldap://[::1]:389 -x -z 100 -b "dc=vutbr,dc=cz" 'mail=xgal*.cz' uid -P 2 > tests/ipv6_loopback_interface`
- Parametry:
  - `-H ldap://[::1]:389`: Specifikace cílového LDAP serveru na IPv6 loopback rozhraní.
  - `-x`: Použití jednoduché autentizace.
  - `-z 100`: Omezení počtu vrácených výsledků na 100.
  - `-b "dc=vutbr,dc=cz"`: Určení kořenového DN pro vyhledávání ve stromu LDAP - zanedbán.
  - `'mail=xgal*.cz'`: Filtr pro hledání položek s e-mailem odpovídajícím vzoru.
  - `uid`: Požadovaný atribut - uid - zanedbán.

- `-P 2`: Použití LDAPv2 protokolu.
- `> tests/ipv6_loopback_interface`: Uložení výstupu do souboru `ipv6_loopback_interface` ve složce `tests`.

Proč to bylo testováno: Test byl proveden k ověření funkcionality serveru pomocí nástroje `ldapsearch` při použití s IPv6 loopback rozhraním na specifickém LDAP serveru s filtrováním na základě e-mailové adresy.

Jak to bylo testováno: Byl proveden LDAP vyhledávací dotaz na server s využitím specifických parametrů pro ověření správného fungování. Nástroj `ldapsearch` byl spuštěn s konkrétními přepínači a filtrováním pro získání určitých atributů.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: LDAP příkaz s uvedenými přepínači a parametry.

Očekávané výstupy:

1. Seznam tří uživatelských záznamů s odpovídajícími atributy (dn, cn, mail).
2. Výsledek hledání se status kódem Success (0).

Skutečné výstupy: Bylo nalezeno tři odpovídající záznamy, které byly vypsány spolu s odpovídajícími atributy (dn, cn, mail). Výsledek hledání měl status kód Success (0).

4.3.2 `loopback_interface.txt`

Co bylo testováno: Byla provedena testovací operace pomocí nástroje `ldapsearch` s následujícími parametry:

- Příkaz: `ldapsearch -H ldap://127.0.0.1:389 -x -b "dc=vutbr,dc=cz" 'mail=xgal*.cz' -P 2 > tests/loopback_interface.txt`
- Parametry:
  - `-H ldap://127.0.0.1:389`: Specifikace cílového LDAP serveru na loopback rozhraní.
  - `-x`: Použití jednoduché autentizace.
  - `-b "dc=vutbr,dc=cz"`: Určení kořenového DN pro vyhledávání ve stromu LDAP.
  - `'mail=xgal*.cz'`: Filtr pro hledání položek s e-mailem odpovídajícím vzoru.
  - `-P 2`: Použití LDAPv2 protokolu.
  - `> tests/loopback_interface.txt`: Uložení výstupu do souboru `loopback_interface.txt` ve složce `tests`.

Proč to bylo testováno: Test byl proveden k ověření funkcionality serveru pomocí nástroje `ldapsearch` s konkrétními parametry, zejména odzkoušení IPV4 loopback rozhraní.

Jak to bylo testováno: Byl proveden LDAP vyhledávací dotaz na server s využitím specifických parametrů pro ověření správného fungování. Nástroj `ldapsearch` byl spuštěn s konkrétními přepínači a filtrováním pro získání určitých atributů.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: LDAP příkaz s uvedenými přepínači a parametry.

Očekávané výstupy:

1. Seznam tří uživatelských záznamů s odpovídajícími atributy (dn, cn, mail).
2. Výsledek hledání se status kódem Success (0).

Skutečné výstupy:

1. Seznam tří uživatelských záznamů s odpovídajícími atributy (dn, cn, mail).
2. Výsledek hledání se status kódem Success (0).

4.3.3 `mail_equalityMatch.txt`

Co bylo testováno: Byla provedena testovací operace pomocí nástroje `ldapsearch` s následujícími parametry:

- Příkaz: `ldapsearch -H ldap://localhost:389 -x -b "dc=vutbr,dc=cz" 'mail=xgabry01@stud.fit.vutbr.cz' -P 2 > tests/mail_equalityMatch.txt`
- Parametry:
  - `-H ldap://localhost:389`: Specifikace cílového LDAP serveru na lokálním rozhraní.
  - `-x`: Použití jednoduché autentizace.
  - `-b "dc=vutbr,dc=cz"`: Určení kořenového DN pro vyhledávání ve stromu LDAP.
  - `'mail=xgabry01@stud.fit.vutbr.cz'`: Filtr pro hledání položky s konkrétní e-mailovou adresou.
  - `-P 2`: Použití LDAPv2 protokolu.
  - `> tests/mail_equalityMatch.txt`: Uložení výstupu do souboru `mail_equalityMatch.txt` ve složce `tests`.

Proč to bylo testováno: Test byl proveden k ověření funkcionality serveru pomocí nástroje `ldapsearch` s konkrétními parametry, zejména s filtrováním na základě e-mailové adresy.

Jak to bylo testováno: Byl proveden LDAP vyhledávací dotaz na server s využitím specifických parametrů pro ověření správného fungování. Nástroj `ldapsearch` byl spuštěn s konkrétními přepínači a filtrováním pro získání určitých atributů.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: LDAP příkaz s uvedenými přepínači a parametry.

Očekávané výstupy:

1. Záznam pro uživatele `xgabry01` s atributy `dn: uid=xgabry01`, `cn: Gabrys Adam`, `mail: xgabry01@stud.fit.vutbr.cz`.
2. Výsledek hledání se status kódem Success (0).

Skutečné výstupy:

- Záznam pro uživatele `xgabry01` s atributy `dn: uid=xgabry01`, `cn: Gabrys Adam`, `mail: xgabry01@stud.fit.vutbr.cz`.
- Výsledek hledání s status kódem Success (0).

4.3.4 `mail_multiple_substring.txt`

Co bylo testováno: Byla provedena testovací operace pomocí nástroje `ldapsearch` s následujícími parametry:

- Příkaz: `ldapsearch -H ldap://localhost:389 -x -z 100 -b "dc=vutbr,dc=cz" 'mail=xgal*.cz' uid -P 2 > tests/mail_multiple_substring.txt`
- Parametry:
  - `-H ldap://localhost:389`: Specifikace cílového LDAP serveru.
  - `-x`: Použití jednoduché autentizace.
  - `-z 100`: Omezení počtu vrácených výsledků na 100.
  - `-b "dc=vutbr,dc=cz"`: Určení kořenového DN pro vyhledávání ve stromu LDAP.
  - `'mail=xgal*.cz'`: Filtr pro hledání položek s e-mailem odpovídajícím vzoru.
  - `uid`: Požadovaný atribut uid - zanedbán.
  - `-P 2`: Použití LDAPv2 protokolu.
  - `> tests/mail_multiple_substring.txt`: Uložení výstupu do souboru `mail_multiple_substring.txt` ve složce `tests`.

Proč to bylo testováno: Test byl proveden k ověření funkcionality serveru pomocí nástroje `ldapsearch` s konkrétními parametry, zejména pro hledání položek s e-mailem obsahujícím daný vzor a otestování filtru substring pro různé atributy.

Jak to bylo testováno: Byl proveden LDAP vyhledávací dotaz na server s využitím specifických parametrů pro ověření správného fungování. Nástroj `ldapsearch` byl spuštěn s konkrétními přepínači a filtrováním pro získání určitých atributů.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: LDAP příkaz s uvedenými přepínači a parametry.

Očekávané výstupy:

1. Seznam tří uživatelských záznamů s odpovídajícím atributem uid.
2. Výsledek hledání se status kódem Success (0).

Skutečné výstupy:

1. Záznamy tří uživatelů s odpovídajícím atributem uid.
2. Výsledek hledání se status kódem Success (0).

4.3.5 `size_limit_exceeded.txt`

Co bylo testováno: Byla provedena testovací operace pomocí nástroje `ldapsearch` s následujícími parametry:

- Příkaz: `ldapsearch -H ldap://localhost:389 -x -z 10 'uid=xga*' -P 2 > tests/size_limit_exceeded.txt`
- Parametry:
  - `-H ldap://localhost:389`: Specifikace cílového LDAP serveru.
  - `-x`: Použití jednoduché autentizace.
  - `-z 10`: Omezení počtu vrácených výsledků na 10.
  - `'uid=xga*'`: Filtr pro hledání položek s uid odpovídajícím vzoru.
  - `-P 2`: Použití LDAPv2 protokolu.
  - `> tests/size_limit_exceeded.txt`: Uložení výstupu do souboru `size_limit_exceeded.txt` ve složce `tests`.

Proč to bylo testováno: Test byl proveden k ověření chování serveru v případě překročení limitu velikosti výsledku hledání.

Jak to bylo testováno: Byl proveden LDAP vyhledávací dotaz na server s využitím specifických parametrů pro ověření reakce na překročení limitu velikosti výsledku. Nástroj `ldapsearch` byl spuštěn s konkrétními přepínači a filtrováním pro získání určitých atributů.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: LDAP příkaz s uvedenými přepínači a parametry.

Očekávané výstupy:

1. Seznam deseti uživatelských záznamů odpovídající filtru.
2. Výsledek hledání se status kódem Size limit exceeded (4).

Skutečné výstupy:

1. Záznamy deseti uživatelů s odpovídající filtru.
2. Výsledek hledání se status kódem Size limit exceeded (4).

4.3.6 `uid_multiple_substring.txt`

Co bylo testováno: Byla provedena testovací operace pomocí nástroje `ldapsearch` s následujícími parametry:

- Příkaz: `ldapsearch -H ldap://localhost:389 -x -b "dc=vutbr,dc=cz" 'uid=xga*' uid -P 2 > tests/uid_multiple_substring.txt`
- Parametry:
  - `-H ldap://localhost:389`: Specifikace cílového LDAP serveru.
  - `-x`: Použití jednoduché autentizace.
  - `-b "dc=vutbr,dc=cz"`: Určení kořenového DN pro vyhledávání ve stromu LDAP.
  - `'uid=xga*'`: Filtr pro hledání položek s uid odpovídajícím vzoru.
  - `uid`: Požadovaný atribut uid - zanedbán.
  - `-P 2`: Použití LDAPv2 protokolu.
  - `> tests/uid_multiple_substring.txt`: Uložení výstupu do souboru `uid_multiple_substring.txt` ve složce `tests`.

Proč to bylo testováno: Test byl proveden k ověření funkcionality serveru při hledání položek s více než jednou hodnotou. Otestování filtru substring a správné volání searchResEntry.

Jak to bylo testováno: Byl proveden LDAP vyhledávací dotaz na server s využitím specifických parametrů pro ověření správného fungování. Nástroj `ldapsearch` byl spuštěn s konkrétními přepínači a filtrováním pro získání určitých atributů.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: LDAP příkaz s uvedenými přepínači a parametry.

Očekávané výstupy:

1. Seznam jedenácti uživatelských záznamů s odpovídajícím atributem uid.
2. Výsledek hledání se status kódem Success (0).

Skutečné výstupy:

1. Záznamy jedenácti uživatelů s odpovídajícím atributem uid.
2. Výsledek hledání se status kódem Success (0).

4.3.7 `Řádné ukončení pomocí signálu SIGINT_test`

Co bylo testováno: Řádné ukončení serveru po poslání signálu SIGINT(CTRL+C).

Jak to bylo testováno: Pomocí DEBUG printů na výstup terminálu a průchodů programem ve funkci `void server::connect_clients()` kdy tato funkce běžela v while cyklu dokud nebyl poslán signál SIGINT a následně proběhlo ukončení.

```
volatile sig_atomic_t exit_signal = 0;

void sigint_handler(int signum) {
    exit_signal = 1;
}
```

```
    while (!exit_signal) {
    /// rest of the code
    }
    printf("\nClosing server socket\n");
    close(server_socket);
    printf("Server closed and exiting gracefully\n");
```

Očekávaný a skutečný výstup na terminálu byly totožné:

```
    LDAP server is listening on port 389...
    ^C
    Closing server socket
    Server closed and exiting gracefully
```

4.3.8 `parallelismus_test`

Co bylo testováno: Byla provedena testovací operace na ověření paralelismu nezávislých úloh pomocí funkce `sleep` k simulaci nezávislých operací ve funkci `bool ldap_functions::handleBindRequest()`.

Proč to bylo testováno: Cílem testu bylo ověřit schopnost systému zpracovávat úlohy paralelně, zejména zjištění, zda nezávislé operace probíhají současně a nejsou ovlivněny čekáním na dokončení jiných úloh.

Jak to bylo testováno: Díky funkce `sleep` bylo možné spustit několik klientů zároveň a pomocí DEBUG printů pozorovat, že obsloužení klientů probíhá nezávisle.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: Několik spuštění stejného příkazu pomocí klientů `ldapsearch`[2] v rychlém sledu.

Očekávané výstupy:

- Zprávy o začátku a dokončení každé úlohy.
- Očekává se, že úlohy budou probíhat nezávisle a paralelně, což bude viditelné ve správném časovém rozvržení výstupů.

Skutečné výstupy:

- Zprávy o začátku a dokončení každé úlohy.
- Úlohy probíhaly nezávisle a paralelně, což bylo potvrzeno správným časovým rozvržením výstupů.

4.3.9 `cn_basic.txt`

Co bylo testováno: Byla provedena testovací operace pomocí nástroje `ldapsearch` s následujícími parametry:

- Příkaz: `ldapsearch -H ldap://[::1]:389 -x -z 100 -b "dc=vutbr,dc=cz" 'cn=Gabry*' -P 2 > tests/cn_basic.txt`
- Parametry:
  - `-H ldap://[::1]:389`: Specifikace cílového LDAP serveru na loopback rozhraní pomocí IPv6 adresy.
  - `-x`: Použití jednoduché autentizace.
  - `-z 100`: Omezení počtu vrácených výsledků na 100.

  - `-b "dc=vutbr,dc=cz"`: Určení kořenového DN pro vyhledávání ve stromu LDAP.
  - `'cn=Gabry*'`: Filtr pro hledání položek s jménem (common name) odpovídajícím vzoru.
  - `-P 2`: Použití LDAPv2 protokolu.
  - `> tests/cn_basic.txt`: Uložení výstupu do souboru `cn_basic.txt` ve složce `tests`.

Proč to bylo testováno: Test byl proveden k ověření funkcionality serveru při vyhledávání položek s konkrétním jménem (common name) pomocí filtru.

Jak to bylo testováno: Byl proveden LDAP vyhledávací dotaz na server s využitím specifických parametrů pro ověření správného fungování. Nástroj `ldapsearch` byl spuštěn s konkrétními přepínači a filtrováním pro získání určitých atributů.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: LDAP příkaz s uvedenými přepínači a parametry.

Očekávané výstupy:

1. Záznam uživatele s odpovídajícím jménem (common name) a atributem uid.
2. Výsledek hledání se status kódem Success (0).

Skutečné výstupy:

1. Záznam uživatele s odpovídajícím jménem (common name) a atributem uid.
2. Výsledek hledání se status kódem Success (0).

4.3.10 `no_base.txt`

Co bylo testováno: Byla provedena testovací operace pomocí nástroje `ldapsearch` s následujícími parametry:

- Příkaz: `ldapsearch -H ldap://[::1]:389 -x 'mail=xgabr*.cz' uid -P 2 > tests/no_base.txt`
- Parametry:
  - `-H ldap://[::1]:389`: Specifikace cílového LDAP serveru na loopback rozhraní pomocí IPv6 adresy.

- `-x`: Použití jednoduché autentizace.
- `'mail=xgabr*.cz'`: Filtr pro hledání položek s e-mailem odpovídajícím vzoru.
- `uid`: Požadovaný atribut uid - zanedbán.
- `-P 2`: Použití LDAPv2 protokolu.
- `> tests/no_base.txt`: Uložení výstupu do souboru `no_base.txt` ve složce `tests`.

Proč to bylo testováno: Test byl proveden k ověření chování serveru při hledání položek bez specifikovaného kořenového DN (base DN).

Jak to bylo testováno: Byl proveden LDAP vyhledávací dotaz na server s využitím specifického filtru pro hledání položek s e-mailem odpovídajícím vzoru. Nebyl specifikován konkrétní kořenový DN (base DN), což mělo otestovat implicitní chování serveru.

Jaké byly vstupy, očekávané výstupy a skutečné výstupy:

Vstupy: LDAP příkaz s uvedenými přepínači a parametry.

Očekávané výstupy:

1. Záznam uživatele s odpovídajícím e-mailem a atributem uid.
2. Výsledek hledání se status kódem Success (0).

Skutečné výstupy:

1. Záznam uživatele s odpovídajícím e-mailem a atributem uid.
2. Výsledek hledání se status kódem Success (0).

Test byl úspěšně proveden, což potvrzuje, že server správně zpracovává dotazy bez specifikovaného kořenového DN.

5 Použitá bibliografie

References

[1] Vladimír Veselý, *LDAP server (Vladimír Veselý)*, Dostupné z `https://www.vut.cz/studis/student.phtml?sn=zadani_detail&apid=268266&zid=54268`, 2023 [cit. 2023-15-11].

[2] *The ldapsearch Command-Line Tool*, Dostupné z `https://docs.ldap.com/ldap-sdk/docs/tool-usages/ldapsearch.html`,[cit. 2023-11-15].

[3] *LDAPv3 Wire Protocol Reference: The ASN.1 Basic Encoding Rules*, [online]. Dostupné z https://ldap.com/ldapv3-wire-protocol-reference-asn1-ber/, [cit. 2023-11-03].

[4] *LDAPv3 Wire Protocol Reference: The LDAPMessage Sequence* [online]. Dostupné z https://ldap.com/ldapv3-wire-protocol-reference-ldap-message/, [cit. 2023-11-15].

[5] J. Sermersheim, Ed., *LDAP: The Protocol - Bind Operation* [online][sekce 5.2]. Dostupné z https://docs.ldap.com/specs/rfc4511.txt, [cit. 2023-11-03].

[6] *Lightweight Directory Access Protocol* [online]. Dostupné z https://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol[section Protocol overview], [last edited: 24 August 2023], [cit. 2023-10-20].

[7] *OpenLDAP Administrator's Guide*[online], OpenLDAP Foundation, © Copyright 2011-2022, Dostupné z https://www.openldap.org/doc/admin26/intro.html, [cit. 2023-11-15].

[8] *LDAP v3*, Oracle Corporation, Dostupné z https://docs.oracle.com/javase/jndi/tutorial/ldap/models/v3.html, [cit. 2023-10-30].

[9] *LDAP Filters*. [online]. 2023. Dostupné z: https://ldap.com/ldap-filters/, [cit. 2023-11-05].

[10] *LDAPv3 Wire Protocol Reference: Bind Operation* [online]. Dostupné z https://ldap.com/ldapv3-wire-protocol-reference-bind/, [cit. 2023-11-03].

[11] *LDAPv3 Wire Protocol Reference: The LDAPMessage Sequence* [online]. Dostupné z https://ldap.com/ldapv3-wire-protocol-reference-ldap-message/, [cit. 2023-11-03].

[12] *ASN.1 Made Simple - ASN.1 Quick Reference* [online]. Dostupné z https://www.oss.com/asn1/resources/asn1-made-simple/asn1-quick-reference/basic-encoding-rules.html,2023 © OSS Nokalva, Inc., [cit. 2023-11-12].