СЛУЖБА КАТАЛОГОВ «СЕЛЕНГА»

Обеспечение программное для управления объектами каталога

Руководство программиста

Прикладной программный интерфейс, протокол LDAP

Версия системы 1.1

Редакция 1

Листов 68

2025

АННОТАЦИЯ

Настоящий документ «Обеспечение программное для управления объектами каталога. Руководство программиста» предназначен для ознакомления с функциями прикладного программного интерфейса (ППИ) службы каталогов «Селенга». Документ разработан в соответствии с ГОСТ 19.504-79 «Единая система программной документации. Руководство программиста».

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Назначение программы

Программное изделие Служба каталогов «Селенга» представляет собой службу каталогов LDAP.

1.2. Функции программы

Программное изделие Служба каталогов «Селенга» представляет собой службу каталогов LDAP и реализующую следующие целевые функции:

1) Предоставление прикладного программного интерфейса, по протоколу LDAP V3;
2) Создание, изменение, удаление и поиск объектов каталога;
3) Хранение данных объектов каталога.
4) Управление доступом к объектам каталога;
5) Управление парольной политикой;
6) Управление схемой данных каталога;
7) Управление репликацией объектов каталога с несколькими серверами;

Среди особенностей Службы каталогов «Селенга» является поддержка следующих служб и спецификаций:

1) Поставщик протокола сетевого времени (NTP), в соответствии со спецификацией RFC 2030;
2) Протокол определения местоположения службы (SLP), в соответствии со спецификацией RFC 2608;
3) Поставщик протокола сетевой аутентификации «Kerberos V5», в соответствии со спецификацией RFC 1510 для службы сетевой аутентификации Kerberos V5.

2. УСЛОВИЯ ПРИМЕНЕНИЯ ПРОГРАММЫ

2.1. Требуемые характеристики технических средств

2.1.1 Требуемые характеристики средств вычислительной техники коллективного пользования

Требуемые характеристики СВТ коллективного пользования:

1) Минимальные требования:

– Центральный процессор:

(1) Архитектура — Intel x86-64.

(2) Число ядер, не менее — 2.

(3) Тактовая частота, ГГц, не менее — 2.

(4) Поддержка набора команд SSE4.2.

– ОЗУ:

(1) Емкость, Гбайт, не менее — 4.

– Накопитель данных:

(1) Емкость, Гбайт, не менее — 8.

2) Рекомендуемые требования:

– Центральный процессор:

(1) Архитектура — Intel x86-64.

(2) Число ядер, не менее — 4 (6 при репликации данных между ЦОД).

(3) Тактовая частота, ГГц, не менее — 3.

(4) Поддержка набора команд SSE4.2.

– ОЗУ:

(1) Емкость, Гбайт, не менее — 16.

– Накопитель данных:

(1) Емкость, Гбайт, не менее — 16.

2.1.2 Требуемые характеристики средств вычислительной техники индивидуального пользования

Требуемые характеристики СВТ индивидуального пользования (АРМ оператора):

1) При разработке программных изделий на базе функционала, предоставляемого Службой каталогов «Селенга»:

– ПЭВМ с сетевым адаптером, обеспечивающим инфокоммуникационный канал, и характеристиками, соответствующими рекомендуемым требованиям операционной системы:

(1) Microsoft Windows версии не ниже 10.

(2) Apple macOS версии не ниже 11 «Big Sur».

(3) Linux с ядром версии 5.4 и выше и графическим интерфейсом.

2) При эксплуатации программных изделий на базе функционала, предоставляемого Службой каталогов «Селенга»:

– ПЭВМ с сетевым адаптером, обеспечивающим инфокоммуникационный канал, и характеристиками, соответствующими рекомендуемым требованиям операционных систем, обеспечивающих функционирование браузеров:

(1) В среде ОС семейства Microsoft Windows:

– Google Chrome версии не ниже 67.

– Microsoft Edge версии не ниже 80.

– Mozilla Firefox версии не ниже 67.

(2) В среде ОС семейства Apple macOS:

– Google Chrome версии не ниже 67.

– Apple Safari версии не ниже 11.1.

– Mozilla Firefox версии не ниже 67.

(3) В середе Linux:

– Google Chrome версии не ниже 67.

– Microsoft Edge версии не ниже 80.

– Mozilla Firefox версии не ниже 67.

2.2. Программное обеспечение, необходимое для функционирования программы

2.2.1 Программное обеспечение, устанавливаемое на СВТ коллективного пользования

Для эксплуатации Службы каталогов «Селенга»: необходимо следующее программное обеспечение, устанавливаемое на СВТ коллективного пользования:

1) Операционная система — Альт Линукс.

2) SSH Server (режим аутентификации по имени и паролю).

3) Пакеты утилит командной строки и общесистемных программных средств — bash, ifconfig, sysctl, curl, yum, systemctl, yum-config-manager, unzip.

Для установки Службы каталогов «Селенга»: необходимо установить операционную систему Linux, настроить SSH Server, пакеты утилит командной строки и общесистемных программных средств, указанных в перечне выше.

2.2.2 Программное обеспечение, устанавливаемое на СВТ индивидуального пользования

Для эксплуатации Службы каталогов «Селенга»: необходимо следующее программное обеспечение, устанавливаемое на СВТ индивидуального пользования (АРМ оператора):

1) При разработке программных изделий на базе функционала, предоставляемого Службой каталогов «Селенга»:

– Операционные системы:

(1) Microsoft Windows версии не ниже 10.

(2) Apple macOS версии не ниже 11 «Big Sur».

(3) Linux с ядром версии 5.4 и выше и графическим интерфейсом.

2) При эксплуатации программных изделий на базе функционала, предоставляемого Службой каталогов «Селенга»:

– Сочетание операционных систем и браузеров:

(1) В среде ОС семейства Microsoft Windows:

– Google Chrome версии не ниже 67.

– Microsoft Edge версии не ниже 80.

– Mozilla Firefox версии не ниже 67.

(2) В среде ОС семейства Apple macOS:

– Google Chrome версии не ниже 67.

– Apple Safari версии не ниже 11.1.

– Mozilla Firefox версии не ниже 67.

(3) В середе Linux:

– Google Chrome версии не ниже 67.

– Microsoft Edge версии не ниже 80.

– Mozilla Firefox версии не ниже 67.

3. ХАРАКТЕРИСТИКИ ПРОГРАММЫ

3.1. Временные характеристики

Программное изделие Служба каталогов «Селенга»: предоставляет средства доступа к хранящимся внутри каталогов данным, таким образом, при его функционировании должны обеспечиваться временные характеристики, перечень которых отображает Таблица 1.

Таблица 1 — Перечень временных характеристик, которым должна соответствовать Служба каталогов «Селенга»

Временные характеристики, которым должна соответствовать Служба каталогов «Селенга»[1)]	Значение характеристик
Количество операций чтения для твердотельного накопителя, операций в секунду	Не менее 200 000
Количество операций чтения для ОЗУ, операций в секунду	Не менее 200 000
Количество операций записи для твердотельного накопителя, операций в секунду	Не менее 150 000
Количество операций записи для ОЗУ, операций в секунду	Не менее 300 000
Средняя задержка, мкс	Не более 4 000
Минимальная задержка, мкс	Не более 200
Максимальная задержка, мкс	Не более 300 000
Задержка, в которую уложились 95% операций, мс	Не более 12
Задержка, в которую уложились 99% операций, мс	Не более 22

———————

[1)] Указанные временные характеристики достигаются при использовании средств вычислительной техники со следующими показателями производительности:

Процессор Intel Xeon, число ядер, не менее — 8.

Емкость ОЗУ, Гбайт, не менее — 32.

Емкость твердотельного накопителя данных (суммарно 4 накопителя), Гбайт, не менее — 120.

3.2. Режим работы

Режим функционирования Службы каталогов «Селенга» — круглосуточный круглогодичный (24/7/365).

3.3. Средства контроля правильности выполнения и самовосстанавливаемости программы

3.3.1 Средства контроля правильности выполнения

Контроль правильности выполнения Службы каталогов «Селенга» осуществляется посредством:

1) Внутренних средств диагностики.

3.3.2 Средства самовосстанавливаемости

Самовосстанавливаемость Службы каталогов «Селенга» осуществляется посредством внутренних репликаций.

1) Внутренних средств восстановления.
2) Сторонних средств автоматического резервного копирования.
3) Встроенных инструментов операционной системы.

4. ОБРАЩЕНИЕ К ПРОГРАММЕ

4.1. Введение

Каталог — это "ряд открытых систем, взаимодействующих друг с другом для предоставления сервисов каталога" [X.500]. Пользователь каталога, человек или другая сущность, получает доступ к каталогу посредством клиента (или пользовательского агента каталога (Directory User Agent, DUA)). Этот клиент от имени пользователя каталога взаимодействует с одним или несколькими серверами (или системными агентами каталога (Directory System Agent, DSA)). Клиенты взаимодействуют с серверами с помощью протокола доступа к службам каталогов.

В данном случае используется протокол Lightweight Directory Access Protocol (LDAP).

4.2. Безопасность

Данная версия протокола предоставляет возможности простой аутентификации с использованием паролей в открытом виде, а также аутентификации с использованием любого механизма SASL. Установление уровней SASL и/или TLS может обеспечить целостность и другие сервисы безопасности информации.

Также разрешается возврат сервером клиенту своих учётных данных для аутентификации, если он захочет это сделать.

Использование паролей в открытом виде настоятельно не рекомендуется там, где используемый транспортный сервис не может гарантировать конфиденциальности и это может привести к раскрытию пароля посторонними лицами.

Считается правильным, когда серверы предотвращают изменения каталога клиентами, осуществляющими доступ анонимно (См. RFC4513).

Соображения безопасности для методов аутентификации, механизмов SASL и TLS описаны в RFC4513.

Обмен аутентификационной информацией SASL не обеспечивает конфиденциальности информации и защиты целостности для полей version или name запроса BindRequest, полей resultCode, diagnosticMessage или referral ответа BindResponse, а также для какой-либо информации, содержащейся в элементах управления, вложенных в запросы и ответы Bind. Таким образом, не следует помещать в эти поля важную информацию, если она не защищена другим способом (таким, как установка защиты на транспортном уровне).

Различные факторы безопасности (в том числе аутентификационная и авторизационная информация и сервисы безопасности данных) могут меняться в ходе сессии LDAP или даже во время выполнения конкретной операции. Например, может закончиться срок действия учётных данных, могут измениться авторизационные сущности или правила контроля доступа, либо уровень (уровни) обеспечения безопасности, поверх которых работает сессия LDAP, могут быть заменены или их работа завершена. Реализации должны обеспечить достаточную надёжность при обработке изменений факторов безопасности.

В некоторых случаях может быть целесообразно продолжить работу даже в свете изменений фактора безопасности. Например, может быть целесообразно продолжить операцию Abandon независимо от произошедших изменений, или продолжить выполнение какой-либо операции, когда изменение привело к повышению (или установлению) фактора безопасности. В других случаях может быть целесообразно завершить неудачей либо изменить обработку производимой операции. Например, при снятии защиты конфиденциальности может быть целесообразно либо завершить запрос на получение конфиденциальных данных неудачей, либо, как минимум, исключить такие данные из возвращаемого результата.

Реализации, кэширующие полученные посредством LDAP атрибуты и записи, должны обеспечить поддержку контроля доступа при предоставлении такой информации нескольким клиентам, поскольку у серверов могут иметься политики контроля доступа, предотвращающие возвращение записей и атрибутов в результатах операции Search, за исключением конкретных клиентов, прошедших аутентификацию. К примеру, информацию из кэша можно выдавать только тем клиентам, в результате запроса которых она и была закэширована.

Серверы могут возвращать отсылки либо ссылки-продолжения в результате операции Search, перенаправляющие клиентов на другие серверы. Недобросовестные приложения имеют возможность внедрить подобные отсылки в поток данных, пытаясь тем самым перенаправить клиента на недобросовестный сервер. Клиентам рекомендуется учитывать это и, по возможности, отклонять отсылки, если защита конфиденциальности не обеспечена.

Содержимое полей matchedDN и diagnosticMessage, а также некоторые значения результирующих кодов resultCode (например, attributeOrValueExists и entryAlreadyExists) могут раскрывать наличие или отсутствие конкретных данных в каталоге, защищаемых средствами контроля доступа и других административных ограничений. Реализациям сервера следует ограничивать доступ к защищаемой информации в равной степени как в нормальных условиях, так и при возникновении ошибок.

Стороны протокола должны быть готовы обрабатывать неверные кодировки протокола и кодировки произвольной длины. Неверные кодировки протокола включают в себя: исключения кодировки BER, исключения формата строки и кодировки UTF-8, исключения переполнения, исключения целочисленных значений, а также исключения флага on/off бинарного режима.

В случае обнаружения сторонами протокола какой-либо атаки, которая может привести к плохим последствиям при продолжении взаимодействия на любом уровне в рамках сессии LDAP, им следует немедленно прекратить эту сессию LDAP.

4.3. Общие элементы

Данный раздел описывает формат блока данных протокола "Конверт сообщения LDAP" (LDAPMessage envelope Protocol Data Unit (PDU)), а также определения типов данных, которые используются в операциях протокола.

4.3.1 Конверт сообщения

В целях обмена сообщениями протокола, все операции протокола инкапсулируются в общий конверт LDAPMessage. Определение LDAPMessage отображает Рисунок 1.

```
LDAPMessage ::= SEQUENCE {
             messageID       MessageID,
             protocolOp      CHOICE {
                  bindRequest           BindRequest,
                  bindResponse          BindResponse,
                  unbindRequest         UnbindRequest,
                  searchRequest         SearchRequest,
                  searchResEntry        SearchResultEntry,
                  searchResDone         SearchResultDone,
                  searchResRef          SearchResultReference,
                  modifyRequest         ModifyRequest,
                  modifyResponse        ModifyResponse,
                  addRequest            AddRequest,
                  addResponse           AddResponse,
```

```
                delRequest              DelRequest,
                delResponse             DelResponse,
                modDNRequest            ModifyDNRequest,
                modDNResponse           ModifyDNResponse,
                compareRequest          CompareRequest,
                compareResponse         CompareResponse,
                abandonRequest          AbandonRequest,
                extendedReq             ExtendedRequest,
                extendedResp            ExtendedResponse,
                ...,
                intermediateResponse    IntermediateResponse },
           controls       [0] Controls OPTIONAL }

       MessageID ::= INTEGER (0 .. maxInt)

       maxInt INTEGER ::= 2147483647 -- (2^^31 - 1) -
```

Конверт сообщения LDAPMessage
Рисунок 1

Назначение LDAPMessage — предоставить конверт, содержащий общие поля, требуемые во всех обменах сообщениями протокола. В настоящий момент общими полями являются только messageID и controls.

Если сервер получает от клиента LDAPMessage, в котором конструкция LDAPMessage SEQUENCE не может быть распознана, либо messageID не может быть разобран, либо значение в поле protocolOp не распознаётся как запрос, либо обнаружено, что закодированная структура или длина полей данных некорректны, то серверу следует вернуть Notice of Disconnection (Уведомление об отключении) с результирующим кодом resultCode, установленным в protocolError, после чего он должен немедленно завершить сессию LDAP.

В остальных случаях, когда клиент или сервер не могут разобрать LDAP PDU, им следует немедленно завершить сессию LDAP, если дальнейшее взаимодействие (в том числе предоставление уведомления) было бы пагубным. В противном случае реализации сервера должны возвращать соответствующий ответ на запрос, с кодом resultCode, установленным в protocolError.

Поля конверта сообщения LDAPMessage определяет Таблица 2.

Таблица 2 – Поля конверта сообщения

Наименование атрибута	Описание	Тип

Наименование атрибута	Описание	Тип
MessageID	Поле должно иметь ненулевое значение, отличное от messageID любого другого запроса в течение одной и той же сессии LDAP. Нулевое значение зарезервировано для сообщений незапрошенных уведомлений (unsolicited notification message). Обычно клиенты при каждом запросе увеличивают счётчик на единицу. Клиенту недопустимо посылать запрос с тем же messageID, что и в предыдущих запросах, в рамках одной и той же сессии LDAP, за исключением тех случаев, когда есть возможность определить, что эти предыдущие запросы больше не обслуживаются сервером. В противном случае поведение не определено.	Integer
protocolOp	Перечень всех операций протокола	List
controls	Элементы управления предоставляют механизм, с помощью которого семантика и аргументы существующих операций LDAP могут быть расширены. К одному сообщению LDAP могут быть присоединены один или несколько элементов управления. Действие элемента управления распространяется только на семантику того сообщения, к которому он присоединён. Элементы управления, посылаемые клиентом, называются "элементами управления запроса" ("request controls"), а посылаемые сервером — "элементами управления ответа" ("response controls"). Структуру поля отображает Рисунок 2.	

```
Controls ::= SEQUENCE OF control Control

      Control ::= SEQUENCE {
           controlType             LDAPOID,
           criticality             BOOLEAN DEFAULT FALSE,
           controlValue            OCTET STRING OPTIONAL }
```

Структура поля Controls
Рисунок 2

Поля элемента управления Controls отображает Таблица 3.

Таблица 3 – Поля элемента управления Controls

Наименование атрибута	Описание	Тип
controlType	Точечно-цифровое представление идентификатора объекта OBJECT IDENTIFIER, уникально идентифицирующего данный элемент управления. Таким образом обеспечивается однозначность именования элементов управления. Часто элемент (элементы) управления ответа, предоставляемые в ответ на элемент управления запроса, разделяют с этим элементом управления запроса значения controlType.	LDAPOID
criticality	Поле criticality имеет смысл только в элементах управления, присоединяемых к сообщениям запроса (за исключением UnbindRequest). Для элементов управления, присоединяемых к сообщениям ответа и UnbindRequest, поле criticality должно быть установлено в FALSE, и должно быть проигнорировано принимающей стороной протокола. Значение TRUE указывает на то, что выполнение операции без применения семантики элемента управления является неприемлемым. Конкретнее, обработка поля criticality выполняется следующим образом: Если сервер не распознаёт тип элемента управления, определяет, что тот не соответствует выполняемой операции, или по какой-то другой причине не желает исполнять операцию с элементом управления, и если поле criticality установлено в TRUE, сервер не должен выполнять эту операцию и, для операций, у которых есть ответное сообщение, он должен вернуть сообщение с результирующим кодом resultCode, установленным в unavailableCriticalExtension. Если сервер не распознаёт тип элемента управления, определяет, что тот не соответствует выполняемой операции, или по какой-то другой причине не желает исполнять операцию с элементом управления, и если поле criticality установлено в FALSE, сервер должен проигнорировать этот элемент управления. Независимо от критичности, если элемент управления применяется к операции, он применяется последовательно и без исключений ко всей операции целиком. Значение по умолчанию: false	Boolean

Наименование атрибута	Описание	Тип
controlValue	Данное поле может содержать информацию, связанную с типом controlType. Формат этого поля определяется спецификацией элемента управления. Реализации протокола должны быть готовы обрабатывать строку октетов controlValue произвольного содержимого, в том числе нулевой длины. Это поле отсутствует только в случае, когда с элементом управления определённого типа не связано никакой информации в виде значения. Когда значение controlValue определено в терминах ASN.1 и закодировано BER, оно также следует правилам расширяемости. Серверы перечисляют поддерживаемые ими типы controlType элементов управления запроса в атрибуте "supportedControl" записи root DSE. Не следует объединять элементы управления, кроме случаев, когда семантика такого объединения была определена. Семантики объединения элементов управления, если таковые определены, как правило можно найти в спецификации элемента управления, опубликованной позднее остальных. Если встречается объединение элементов управления, семантика которого неверна, не определена (или неизвестна), сообщение считается плохо сформированным; таким образом, операция завершается неудачей с результирующим кодом protocolError. Элементы управления с полем criticality, установленным в FALSE, могут быть проигнорированы с целью получения допустимого сочетания. Кроме того, если в спецификации не указаны семантики, зависящие от порядка следования элементов управления, порядок комбинации элементов управления в последовательности SEQUENCE игнорируется. Там, где порядок следования элементов управления должен быть проигнорирован, но сервер не может его проигнорировать, сообщение считается плохо сформированным и операция завершается неудачей с результирующим кодом protocolError. Опять же, элементы управления с полем criticality, установленным в FALSE, могут быть проигнорированы с целью получения допустимого сочетания.	Octet String

4.4. Операции

4.4.1 Операция подсоединения Bind

Функция операции Bind — разрешить обмен аутентификационной информацией между клиентом и сервером. Операция Bind должна рассматриваться как операция "аутентификации". Операционные, аутентификационные и связанные с безопасностью семантики данной операции даны в RFC4513.

4.4.1.1. Запрос

Структуру запроса Bind отображает Рисунок 3.

```
BindRequest ::= [APPLICATION 0] SEQUENCE {
     version                 INTEGER (1 .. 127),
     name                    LDAPDN,
     authentication          AuthenticationChoice }

AuthenticationChoice ::= CHOICE {
     simple                  [0] OCTET STRING,
                             -- 1 и 2 зарезервированы
     sasl                    [3] SaslCredentials,
     ... }

SaslCredentials ::= SEQUENCE {
     mechanism               LDAPString,
     credentials             OCTET STRING OPTIONAL }
```

Структура запроса
Рисунок 3

Поля запроса Bind отображает Таблица 4.

Таблица 4 – Поля запроса Bind

Наименование атрибута	Описание	Тип
version	Номер версии, указывающий версию протокола, которая будет использоваться на уровне сообщений LDAP. В этом документе описывается версия 3 протокола. Согласования версий не производится. Клиент устанавливает версию в этом поле по своему желанию. Если сервер не поддерживает указанную версию, он должен ответить сообщением BindResponse, в котором результирующий код resultCode установлен в protocolError.	Integer

Наименование атрибута	Описание	Тип
name	Если поле не пустое, оно содержит имя объекта каталога, от которого клиент хочет произвести подсоединение. Это поле может принимать нулевое значение (строка нулевой длины) в целях анонимного подсоединения ([RFC4513], раздел 5.1) или при использовании SASL-аутентификации [RFC4422] ([RFC4513], раздел 5.2). При попытке найти поименованный объект, серверу не следует выполнять разыменование псевдонимов.	LDAPDN
authentication	Информация, используемая для аутентификации. Эта часть сообщения является расширяемой, как определено в разделе 3.7 [RFC4520]. Серверы, не поддерживающие предоставленный клиентом вариант аутентификационной информации, возвращают сообщение BindResponse с результирующим кодом resultCode, установленным в authMethodNotSupported. Параметры поля отображает Таблица 5.	

Таблица 5 – Параметры поля authentication

Наименование атрибута	Описание	Тип
simple	Текстовые пароли (состоящие из последовательности символов с известным набором символов и кодировкой), передаваемые на сервер с использованием варианта simple конструкции AuthenticationChoice, должны передаваться как Unicode, закодированный UTF-8 RFC3629. Перед передачей клиенту следует подготовить текстовые пароли как строки запроса "query" путём применения к ним профиля SASLprep RFC4013 алгоритма stringprep RFC3454. Пароли, состоящие из других данных (такие, как случайный набор октетов), не должны изменяться. Определение того, является ли пароль текстовым, возлагается на клиента.	Octet String
sasl	Конструкция, представленная значениями полей mechanism и credentials.	

Соединение LDAP может быть установлено либо во время создания объекта, либо как отдельный шаг. Точно так же аутентификация может выполняться для соединения во время его создания, во время его установления или как отдельный процесс. Пример запроса Bind на языке Java отображает Рисунок 4.

```
// Создаётся новое неустановленное соединение. Затем подключается и выполняется
// простое присоединение отдельной операцией.
LDAPConnection c = new LDAPConnection();
c.connect(address, port);
BindResult bindResult = c.bind(bindDN, password);

// Создаётся новое соединение, которое устанавливается во время создания, и затем
// проходит отдельно аутентификацию, используя простую аутентификацию
c = new LDAPConnection(address, port);
BindResult bindResult = c.bind(bindDN, password);

// Создаётся новое соединение, которое устанавливается и связывает, используя простую
// аутентификацию, всё в одном шаге.
c = new LDAPConnection(address, port, bindDN, password);
```

Пример запроса Bind
Рисунок 4

4.4.1.2. Ответ

Структуру ответа Bind отображает Рисунок 5.

```
BindResponse ::= [APPLICATION 1] SEQUENCE {
         COMPONENTS OF LDAPResult,
         serverSaslCreds     [7] OCTET STRING OPTIONAL }
```

Структура ответа
Рисунок 5

Поля ответа на запрос Bind отображает Таблица 6.

Таблица 6 – Поля ответа на запрос Bind

Наименование атрибута	Описание	Тип

Наименование атрибута	Описание	Тип
LDAPResult	Об удачном завершении операции Bind свидетельствует BindResponse с результирующим кодом resultCode, установленным в success. При ином исходе в BindResponse устанавливается соответствующий результирующий код. Результирующий код protocolError в BindResponse может использоваться для указания на то, что предоставленный клиентом номер версии не поддерживается. Если клиент получает сообщение BindResponse, в котором resultCode установлено в protocolError, он должен считать, что сервер не поддерживает эту версию LDAP. Клиент может быть способен продолжить сессию с другой версией данного протокола (при этом может потребоваться или не потребоваться закрывать и вновь устанавливать транспортное соединения), однако описание того, как можно продолжить сессию с другой версией данного протокола, выходит за рамки этого документа. Клиентам, которые не в состоянии или не желают продолжать сессию, следует завершить эту сессию LDAP.	
serverSaslCreds	Используется как часть определённого в SASL механизма подсоединения, чтобы позволить клиенту аутентифицировать сервер, с которым он взаимодействует, либо выполнить аутентификацию типа "вызов-ответ" ("challenge-response"). Если клиент выполнял подсоединение с механизмом simple, либо механизм SASL не требует, чтобы сервер возвращал информацию клиенту, то включать это поле в BindResponse не нужно.	OCTET STRING OPTIONAL

4.4.2 Операция отсоединения Unbind

Назначение операции Unbind — завершение сессии LDAP. Операция Unbind не является противоположностью операции Bind, как можно предположить из названия. Наименования этих операций являются историческими. Операция Unbind должна рассматриваться как операция выхода "quit".

4.4.2.1. Запрос

Структуру запроса отсоединения Unbind отображает Рисунок 6.

```
UnbindRequest ::= [APPLICATION 2] NULL
```

Структура запроса
Рисунок 6

Примечание: клиент при передаче UnbindRequest и сервер при получении UnbindRequest должны корректно завершить сессию LDAP. Обработка незавершённых операций производится следующим образом: при закрытии транспортного соединения любые незавершённые операции на уровне сообщений LDAP отбрасываются (если это возможно), либо завершаются без передачи ответа (когда отказаться от их выполнения невозможно). Также, при закрытии транспортного соединения, клиент не должен подразумевать, что любые незавершившиеся операции обновления были выполнены успешно или неуспешно.

Пример запроса Unbind на языке Java отображает Рисунок 7.

```
public static void closeConnection( LdapNetworkConnection conn ) throws LdapException,
IOException
{
  if ( conn != null )
  {
    conn.unBind();
    conn.close();
  }
}
```

Пример запроса
Рисунок 7

Пример запроса Unbind на языке Java с использованием перехвата и обработки исключений отображает Рисунок 8.

```
@Override
public void close(final RequestControl[] controls)
 throws LdapException
{
 if (controls != null) {
  throw new UnsupportedOperationException("Provider does not support unbind with controls");
 }
 if (connection != null) {
  try {
   if (connection.isConnected()) {
    connection.unBind();
   }
  } catch (org.apache.directory.api.ldap.model.exception.LdapException e) {
   logger.error("Error unbinding from LDAP", e);
```

```
  }
  try {
   connection.close();
  } catch (IOException e) {
   throw new LdapException(e);
  } finally {
   connection = null;
  }
 }
}
```

Пример запроса
Рисунок 8

4.4.3 Операция поиска Search

Операция Search используется для того, чтобы запросить сервер вернуть (после проверки контроля доступа и других ограничений) набор записей, соответствующих комплексному критерию поиска. Она может быть использована для получения атрибутов единственной записи, записей, непосредственно подчинённых какой-либо конкретной записи, либо всего поддерева записей.

4.4.3.1. Запрос

Структуру запроса Search отображает Рисунок 9.

```
SearchRequest ::= [APPLICATION 3] SEQUENCE {
            baseObject      LDAPDN,
            scope           ENUMERATED {
                 baseObject              (0),
                 singleLevel             (1),
                 wholeSubtree            (2),
                 ... },
            derefAliases    ENUMERATED {
                 neverDerefAliases       (0),
                 derefInSearching        (1),
                 derefFindingBaseObj     (2),
                 derefAlways             (3) },
            sizeLimit       INTEGER (0 .. maxInt),
            timeLimit       INTEGER (0 .. maxInt),
            typesOnly       BOOLEAN,
            filter          Filter,
            attributes      AttributeSelection }

       AttributeSelection ::= SEQUENCE OF selector LDAPString
                       -- строка LDAPString, ограниченная конструкцией
                       -- <attributeSelector> из раздела 4.5.1.8

       Filter ::= CHOICE {
```

```
            and             [0] SET SIZE (1..MAX) OF filter Filter,
            or              [1] SET SIZE (1..MAX) OF filter Filter,
            not             [2] Filter,
            equalityMatch   [3] AttributeValueAssertion,
            substrings      [4] SubstringFilter,
            greaterOrEqual  [5] AttributeValueAssertion,
            lessOrEqual     [6] AttributeValueAssertion,
            present         [7] AttributeDescription,
            approxMatch     [8] AttributeValueAssertion,
            extensibleMatch [9] MatchingRuleAssertion,
            ... }

       SubstringFilter ::= SEQUENCE {
            type           AttributeDescription,
            substrings     SEQUENCE SIZE (1..MAX) OF substring CHOICE {
                 initial [0] AssertionValue,  -- может вкючаться только один раз
                 any     [1] AssertionValue,
                 final   [2] AssertionValue } -- может вкючаться только один раз
            }

       MatchingRuleAssertion ::= SEQUENCE {
            matchingRule    [1] MatchingRuleId OPTIONAL,
            type            [2] AttributeDescription OPTIONAL,
            matchValue      [3] AssertionValue,
            dnAttributes    [4] BOOLEAN DEFAULT FALSE }
```

Запрос
Рисунок 9

Поля запроса Search отображает Таблица 7.

Таблица 7 – Поля запроса Search

Наименование атрибута	Описание	Тип
baseObject	Имя записи базового объекта (или, возможно, корневой записи), относительно которой должна быть выполнена операция поиска Search.	LDAPDN
scope	Указывает диапазон выполняемой операции Search. Семантика определённых для данного поля значений (как описано в [X.511]): baseObject: диапазон ограничен записью, указанной в baseObject. singleLevel: диапазон ограничен записями, непосредственно подчинёнными записи, указанной в baseObject. wholeSubtree: диапазон ограничен записью, указанной в baseObject, и всеми подчинёнными ей записями.	Enumerated

Наименование атрибута	Описание	Тип
derefAliases	Индикатор того, должны или нет записи-псевдонимы (определённые в [RFC4512]) разыменовываться на этапах операции поиска Search. Процедура разыменования псевдонимов включает в себя рекурсивное разыменование псевдонимов, которые ссылаются на другие псевдонимы. Серверы должны определять зацикливание в процессе разыменования псевдонимов в целях предотвращения атак типа "отказ от обслуживания" подобного рода. Семантика определённых для данного поля значений: neverDerefAliases: не разыменовывать псевдонимы при поиске или при определении местонахождения базового объекта поиска. derefInSearching: при поиске среди подчинённых записей базового объекта, разыменовывать любые псевдонимы в рамках поискового диапазона. Разыменованные объекты становятся вершинами дальнейших диапазонов поиска, на которые также распространяется эта операция поиска Search. Если диапазон поиска — wholeSubtree, операция поиска Search продолжается по поддереву (поддеревьям) любого разыменованного объекта. Если диапазон поиска — singleLevel, операция поиска применяется к любым разыменованным объектам и не применяется к подчинённым им записям. Серверам следует исключить дублирующиеся записи, появляющиеся в процессе разыменования псевдонимов при поиске. derefFindingBaseObj: разыменовывать псевдонимы при определении местонахождения базового объекта поиска, но не при поиске подчинённых записей этого объекта. derefAlways: разыменовывать псевдонимы и при поиске, и при определении местонахождения базового объекта поиска.	Enumerated

Наименование атрибута	Описание	Тип
sizeLimit	Ограничение по размеру, устанавливающее максимальное количество записей, которое будет возвращено в качестве результата операции Search. Значение ноль в этом поле означает, что никакие запрашиваемые клиентом ограничения по размеру не распространяются на данную операцию Search. Сервер также может принудительно установить максимальное количество записей, которое он будет возвращать.	Integer
timeLimit	Ограничение по времени, устанавливающее максимальное время (в секундах), которое отводится на выполнение операции Search. Значение ноль в этом поле означает, что никакие запрашиваемые клиентом ограничения по времени не распространяются на данную операцию Search. Сервер также может принудительно установить максимальное время выполнения операции Search.	Integer
typesOnly	Индикатор того, должны ли результаты операции Search содержать и описания атрибутов, и значения, либо только описания атрибутов. Установка этого поля в TRUE приведёт к возврату только описаний атрибутов (без значений). Установка этого поля в FALSE приведёт к возврату и описаний атрибутов, и значений.	Boolean

Наименование атрибута	Описание	Тип
filter	Фильтр, определяющий условия, которые должны быть соблюдены для того, чтобы операция Search привела к нахождению требуемых записей. Для формирования комбинаций фильтров могут быть использованы пункты "and", "or" и "not". Как минимум один элемент фильтра должен присутствовать при использовании пунктов "and" или "or". Элементы фильтра предназначены для нахождения совпадений значений индивидуальных атрибутов записей в диапазоне поиска. Сервер должен оценить фильтры в соответствии с трехзначной логикой согласно пункта 7.8.1 стандарта X.511 (1993). Если коротко, фильтр оценивается как "TRUE", "FALSE" или "Undefined". Если фильтр оценивается как TRUE для конкретной записи, то атрибуты такой записи возвращаются как часть результата операции Search (после проверки на соответствие любым применимым к записи ограничениям контроля доступа). Если фильтр оценивается как FALSE или Undefined, то для этой операции Search запись игнорируется. Фильтр оценивается как Undefined, когда сервер не в состоянии определить, соответствует ли значение утверждения записи. Возможные примеры: - Описание атрибута в фильтрах equalityMatch, substrings, greaterOrEqual, lessOrEqual, approxMatch или extensibleMatch не распознаётся сервером. - Запрашиваемое правило соответствия не определено в типе атрибута. - Идентификатор MatchingRuleId в фильтре extensibleMatch не распознаётся сервером или не является действительным для данного типа атрибута. - Тип запрашиваемого фильтра не реализован. - Значение утверждения не верно.	

Наименование атрибута	Описание	Тип
filter.equalityMatch	Правило соответствия для фильтра equalityMatch определяется правилом соответствия EQUALITY для типа или подтипа атрибута. Фильтр оценивается как TRUE, когда правило EQUALITY, при применении его к типу или подтипу атрибута и заявленному значению, возвращает TRUE.	
filter.substrings	Должно быть не более одного пункта "initial" и не более одного пункта "final" в последовательности "substrings" конструкции SubstringFilter. Если присутствует пункт "initial", он должен быть первым элементом последовательности "substrings". Если присутствует пункт "final", он должен быть последним элементом последовательности "substrings". Правило соответствия для AssertionValue в пункте фильтра substrings определяется правилом соответствия SUBSTR для типа или подтипа атрибута. Фильтр оценивается как TRUE, когда правило SUBSTR, при применении его к типу или подтипу атрибута и заявленному значению, возвращает TRUE. AssertionValue в пункте фильтра substrings соответствует синтаксису утверждения правила соответствия EQUALITY для типа атрибута, а не синтаксису утверждения правила соответствия SUBSTR для типа атрибута. Концептуально, перед применением правила вся конструкция SubstringFilter преобразуется в значение утверждения согласно правилу соответствия substrings.	
filter.greaterOrEqual	Правило соответствия для фильтра greaterOrEqual определяется правилом соответствия ORDERING для типа или подтипа атрибута. Фильтр оценивается как TRUE, когда правило ORDERING, при применении его к типу или подтипу атрибута и заявленному значению, возвращает FALSE.	
filter.lessOrEqual	Правило соответствия для фильтра lessOrEqual определяется правилами соответствия ORDERING и EQUALITY для типа или подтипа атрибута. Фильтр оценивается как TRUE, когда либо правило ORDERING, либо правило EQUALITY, при применении их к типу или подтипу атрибута и заявленному значению, возвращают TRUE.	

Наименование атрибута	Описание	Тип
filter.present	Фильтр present оценивается как TRUE, когда в записи присутствует тип или подтип атрибута, соответствующий указанному описанию типа атрибута AttributeDescription; и как FALSE, когда в записи не присутствует тип или подтип атрибута, соответствующий указанному описанию типа атрибута, в противном случае такой фильтр оценивается как Undefined.	
filter.approxMatch	Фильтр approxMatch оценивается как TRUE, когда присутствует значение типа или подтипа атрибута, для которого некоторый локально определённый алгоритм нахождения приблизительного соответствия (например, по вариантам написания, по фонетическому соответствию, и т.п.) возвращает TRUE. Если значение соответствует фильтру equality, оно также удовлетворяет соответствию approximate. Если для атрибута не поддерживается нахождение соответствия approximate, этот пункт фильтра должен интерпретироваться как equalityMatch.	

Наименование атрибута	Описание	Тип
filter.extensibleMatch	Поля пункта фильтра extensibleMatch оцениваются следующим образом: При отсутствии поля matchingRule должно присутствовать поле type, и для этого типа атрибута type выполняется сравнение по соответствию equality. При отсутствии поля type и наличии matchingRule, значение в поле matchValue сравнивается со значениями всех атрибутов записи, поддерживающих это правило соответствия matchingRule. При наличии и поля type, и поля matchingRule, значение в поле matchValue сравнивается со значениями указанного типа атрибута и его подтипов. Если поле dnAttributes установлено в TRUE, то сравнение дополнительно производится по всем парам атрибут-значение AttributeValueAssertions в уникальном имени (distinguished name) записи. В этом случае фильтр оценивается как TRUE, если в уникальном имени есть хотя бы один тип или подтип атрибута, для которого данный пункт фильтра оценивается как TRUE. Поле dnAttributes введено для того, чтобы избежать необходимости определения нескольких версий фильтров с общим правилом соответствия (например, нахождения соответствия слову), когда одна версия применяется к записям, а другая — к записям и атрибутам DN. Используемое для оценки правило соответствия matchingRule определяет синтаксис для значения утверждения. После того, как были определены matchingRule и атрибут (атрибуты), пункт фильтра оценивается как TRUE, если найдено соответствие хотя бы с одним типом или подтипом атрибута в записи; как FALSE, если не найдено соответствие ни с одним типом или подтипом атрибута в записи; и как Undefined, если правило соответствия matchingRule не распознано, не может быть использовано с указанным типом type, либо значение assertionValue не верно.	

Наименование атрибута	Описание	Тип
attributes	Список атрибутов, которые должны быть возвращены для каждой записи, соответствующей поисковому фильтру. Атрибуты, которые являются подтипами перечисленных атрибутов, также неявно включаются в этот список. На строковые значения LDAPString этого поля накладываются ограничения по следующей расширенной форме Бэкуса-Наура (ABNF) [RFC4234]: attributeSelector = attributedescription / selectorspecial selectorspecial = noattrs / alluserattrs noattrs = %x31.2E.31 ; "1.1" alluserattrs = %x2A ; asterisk ("*") Конструкция <attributedescription> определена в разделе 2.5 [RFC4512]. В списке атрибутов могут фигурировать три особых случая: 1. Запросы с пустым списком (без атрибутов) возвращают все пользовательские атрибуты. 2. Запросы со списком, содержащим "*" (с нулём или более описаний атрибутов), возвращают все пользовательские атрибуты в дополнение к другим перечисленным (операционным) атрибутам. 3. Список, содержащий только OID "1.1", указывает на то, что никаких атрибутов возвращено не будет. Если кроме "1.1" предоставляются другие значения attributeSelector, значение attributeSelector "1.1" игнорируется. Данный OID был выбран потому, что он не соответствует (и не может соответствовать) никакому используемому атрибуту. Операционные атрибуты описаны в [RFC4512]. Атрибуты в записи возвращаются не более одного раза. Если описание атрибута указывается в списке более одного раза, повторно встретившиеся имена игнорируются. Если описание атрибута в списке не распознаётся, оно игнорируется сервером.	

Пример запроса Search на языке Java отображает Рисунок 10 и Рисунок 11.

```
    @Test
    public void testSearch() throws Exception
    {
        EntryCursor cursor = connection.search( "ou=system", "(objectclass=*)", SearchScope.ON
ELEVEL );

        try
        {
            for ( Entry entry : cursor )
            {
                assertNotNull( entry );
                System.out.println( entry );
            }
        }
        finally
        {
            cursor.close();
        }
    }
```

Пример запроса
Рисунок 10

Пример запроса Search на языке Java с выводом результатов на экран отображает Рисунок 11.

```
import java.util.Hashtable;

import javax.naming.Context;
import javax.naming.NamingEnumeration;
import javax.naming.directory.Attribute;
import javax.naming.directory.Attributes;
import javax.naming.directory.DirContext;
import javax.naming.directory.InitialDirContext;
import javax.naming.directory.SearchControls;
import javax.naming.directory.SearchResult;

public class LdapSearch {
  public static void main(String[] args) throws Exception {
    Hashtable env = new Hashtable();

    String sp = "com.sun.jndi.ldap.LdapCtxFactory";
    env.put(Context.INITIAL_CONTEXT_FACTORY, sp);

    String ldapUrl = "ldap://localhost:389/dc=yourName, dc=com";
    env.put(Context.PROVIDER_URL, ldapUrl);

    DirContext dctx = new InitialDirContext(env);

    String base = "ou=People";
```

```
    SearchControls sc = new SearchControls();
    String[] attributeFilter = { "cn", "mail" };
    sc.setReturningAttributes(attributeFilter);
    sc.setSearchScope(SearchControls.SUBTREE_SCOPE);

    String filter = "(&(sn=W*)(l=Criteria*))";

    NamingEnumeration results = dctx.search(base, filter, sc);
    while (results.hasMore()) {
      SearchResult sr = (SearchResult) results.next();
      Attributes attrs = sr.getAttributes();

      Attribute attr = attrs.get("cn");
      System.out.print(attr.get() + ": ");
      attr = attrs.get("mail");
      System.out.println(attr.get());
    }
    dctx.close();
  }
}
```

Пример запроса
Рисунок 11

4.4.3.2. Ответ

Результаты операции Search возвращаются в виде нуля или более сообщений SearchResultEntry и/или сообщений SearchResultReference, за которыми следует единственное сообщение SearchResultDone.

Структуру ответа на запрос Search отображает Рисунок 12.

```
SearchResultEntry ::= [APPLICATION 4] SEQUENCE {
     objectName      LDAPDN,
     attributes      PartialAttributeList }

PartialAttributeList ::= SEQUENCE OF
                     partialAttribute PartialAttribute

SearchResultReference ::= [APPLICATION 19] SEQUENCE
                          SIZE (1..MAX) OF uri URI

SearchResultDone ::= [APPLICATION 5] LDAPResult
```

Структура ответа
Рисунок 12

Поля ответа на запрос Search отображает Таблица 8.

Таблица 8 – Поля ответа на запрос Search

Наименование атрибута	Описание	Тип
SearchResultEntry	Каждое сообщение представляет собой запись, найденную во время поиска. Каждая запись, возвращаемая в сообщении SearchResultEntry, будет содержать все соответствующие атрибуты, указанные в поле attributes запроса Search, по результатам применения к ним контроля доступа и другой административной политики.	
PartialAttributeList	Последовательность может содержать ноль элементов. Такое может произойти, когда ни один из атрибутов записи не был запрошен или не может быть возвращён. Набор значений partialAttribute может содержать ноль элементов. Такое может произойти при запросе с выставленным полем typesOnly, если контроль доступа не допускает возврата значений или по другим причинам.	Sequence
SearchResultReference	Каждое сообщение представляет собой область, которая ещё не была изучена во время поиска. Если сервер смог определить расположение записи, на которую указывает поле baseObject запроса, но не может или не желает осуществлять поиск одной или нескольких нелокальных записей, он может вернуть одно или несколько сообщений SearchResultReference, каждое из которых содержит ссылку на другой набор серверов для продолжения операции. Сервер не должен возвращать какие-либо сообщения SearchResultReference, если он не определил расположение объекта baseObject, и, следовательно, не нашёл ни одной записи. В этом случае он должен вернуть сообщение SearchResultDone, содержащее либо отсылку, либо результирующий код noSuchObject (в зависимости от того, обладает ли сервер знаниями относительно записи, указанной в baseObject).	Sequence
SearchResultDone	Содержит индикацию успешного завершения, либо детали любых произошедших ошибок.	String

Для примера, предположим, что на сервере, к которому происходит обращение (hosta), хранятся записи <DC=Example,DC=NET> и <CN=Manager,DC=Example,DC=NET>. Этот сервер знает, что на двух других LDAP-серверах (hostb) и (hostc) (один из них главный (master), а второй — теневая копия) хранится <OU=People,DC=Example,DC=NET>, а на LDAP-совместимом сервере (hostd) хранится поддерево <OU=Roles,DC=Example,DC=NET>. Если на сервере, к которому происходит обращение, был запрошен поиск Search по <DC=Example,DC=NET> с диапазоном wholeSubtree.

Пример ответа на запрос Search отображает Рисунок 13.

```
SearchResultEntry for DC=Example,DC=NET
SearchResultEntry for CN=Manager,DC=Example,DC=NET
SearchResultReference {
  ldap://hostb/OU=People,DC=Example,DC=NET??sub
  ldap://hostc/OU=People,DC=Example,DC=NET??sub }
SearchResultReference {
  ldap://hostd/OU=Roles,DC=Example,DC=NET??sub }
SearchResultDone (success)
```

Пример ответа
Рисунок 13

4.4.4 Операция модификации Modify

Операция модификации Modify позволяет клиенту запросить, чтобы сервер выполнил модификацию записи от его имени.

4.4.4.1. Запрос

Структуру запроса Modify отображает Рисунок 14.

```
ModifyRequest ::= [APPLICATION 6] SEQUENCE {
     object          LDAPDN,
     changes         SEQUENCE OF change SEQUENCE {
          operation       ENUMERATED {
               add     (0),
               delete  (1),
               replace (2),
               ... },
          modification    PartialAttribute } }
```

Структура запроса
Рисунок 14

Поля запроса Modify отображает Таблица 9.

Таблица 9 – Поля запроса Modify

Наименование атрибута	Описание	Тип
object	Значение этого поля содержит имя записи, которая будет модифицирована. Серверу не нужно выполнять какие-либо разыменования псевдонимов для определения объекта, который требуется модифицировать.	LDAPDN
changes	Список модификаций, которые будут произведены над записью. Все модификации из списка должны быть выполнены в порядке их перечисления как одна атомарная операция. Хотя отдельные модификации могут нарушать определенные аспекты схемы данных каталога (такие, как определение объектного класса и правило содержимого информационного дерева каталога (Directory Information Tree, DIT)), результирующая запись после выполнения всех модификаций из списка должна удовлетворять требованиям модели каталога и управляющей схемы [RFC4512].	Sequence
operation	Используется для указания типа выполняемой модификации. Каждый тип операции воздействует на тот атрибут, который указан в следующем за полем operation поле modification. Значения этого поля имеют следующие семантики (соответственно): add: добавить перечисленные значения к атрибуту, указанному в поле modification, при необходимости создать этот атрибут. delete: удалить перечисленные значения из атрибута, указанного в поле modification. Если не было перечислено никаких значений или если были перечислены все текущие значения данного атрибута, атрибут удаляется полностью. replace: заменить все существующие значения атрибута, указанного в поле modification на новые (перечисленные), создать атрибут, если его ещё не существовало. В случае, если новые значения не были перечислены, замена сводится к удалению атрибута целиком, если он существует, и к игнорированию операции, если он не существует.	Enumerated

Наименование атрибута	Описание	Тип
modification	Частичный атрибут PartialAttribute (у которого может быть пустой набор значений). Это поле используется для указания того типа атрибута или типа атрибута вместе со значениями, которые будут модифицироваться.	

Для модификации записи необходимо знать, какую запись нужно изменить, то есть знать ее Dn. Затем необходимо создать экземпляр Modification, который применяется к записи. Для применения нескольких модификаций необходимо создать несколько экземпляров модификации перед вызовом метода.

Пример запроса Modify на языке Java отображает Рисунок 15.

```
    ...
    Modification addedGivenName = new DefaultModification( ModificationOperation.ADD_ATTRIBUTE
, "givenName",
        "John", "Peter" );
    Modification addedInitials = new DefaultModification( ModificationOperation.ADD_ATTRIBUTE,
 "initials",
        "JD" );

    connection.modify( "uid=Doe,dc=acme,dc=com", addedGivenName, addedInitials );
    ...
```

Пример запроса
Рисунок 15

4.4.4.2. Ответ

После получения запроса Modify сервер пытается выполнить необходимые модификации в DIT и возвращает результат в ответе Modify Response.

Структуру ответа на запрос Modify отображает Рисунок 16.

```
ModifyResponse ::= [APPLICATION 7] LDAPResult
```

Структура ответа
Рисунок 16

Примечание: сервер возвращает клиенту единственный ответ Modify, сообщающий либо об успешной модификации DIT, либо причину неудачного завершения модификации. В связи с требованием атомарности в применении списка изменений в запросе Modify Request, клиент вправе ожидать, что в случае получения ответа Modify,

указывающего на ошибку любого рода, никаких модификаций DIT произведено не было, а в случае получения ответа Modify, указывающего на успешное завершение операции, все запрошенные модификации были произведены. Клиент не может определить, была или нет выполнена модификация, если ответ Modify не был получен (например, в случае прерывания сессии LDAP или отказа от этой операции Modify).

Серверы должны обеспечить, чтобы записи удовлетворяли правилам пользовательской и системной схемы данных, а также другим ограничениям модели данных. Операция Modify не может быть использована для удаления из записи каких-либо её уникальных (отличительных) значений, то есть тех значений, которые формируют относительное уникальное имя записи. Попытка сделать это приведёт к тому, что сервер вернёт результирующий код notAllowedOnRDN. Для переименования записи используется операция модификации уникального имени (Modify DN).

Типы атрибутов, для которых не определено соответствие equality, подчиняются правилам из раздела 2.5.1 [RFC4512].

В связи со сделанными в LDAP упрощениями, нет прямого отображения изменений посредством запроса LDAP ModifyRequest в изменения посредством операции DAP ModifyEntry, и различные реализации шлюзов LDAP-DAP могут использовать различные средства представления этих изменений. В случае успешного завершения изменений, окончательный эффект от выполнения этих операций над записью должен быть идентичен.

4.4.5 Операция добавления Add

Операция Add позволяет клиенту запросить добавление записи в каталог.

4.4.5.1. Запрос

Структуру запроса Add отображает Рисунок 17.

```
AddRequest ::= [APPLICATION 8] SEQUENCE {
           entry           LDAPDN,
           attributes      AttributeList }

      AttributeList ::= SEQUENCE OF attribute Attribute
```

Структура запроса
Рисунок 17

Поля запроса Add отображает Таблица 10.

Таблица 10 – Поля запроса Add

Наименование атрибута	Описание	Тип
entry	Имя добавляемой записи. Серверу не нужно выполнять какие-либо разыменования псевдонимов для определения местоположения записи, которая будет добавлена.	LDAPDN
attributes	Список атрибутов, которые, наряду с атрибутами из RDN, составляют содержимое добавляемой записи. Клиенты могут включать, либо могут не включать в этот список атрибут (атрибуты) RDN. Клиенты не должны предоставлять те атрибуты, которые нельзя модифицировать пользователю (NO-USER-MODIFICATION), такие как атрибуты createTimestamp или creatorsName, поскольку сервер выставляет их автоматически.	

Серверы должны обеспечить, чтобы записи удовлетворяли правилам пользовательской и системной схемы данных, а также другим ограничениям модели данных. Типы атрибутов, для которых не определено соответствие equality, подчиняются правилам из раздела 2.5.1 [RFC4512] (это относится к атрибуту именования и, кроме того, к любым добавляемым многозначным атрибутам).

Для успешного выполнения запроса AddRequest запись, имя которой указано в поле entry этого запроса, не должна существовать. Непосредственно вышестоящая (родительская) запись добавляемой записи объекта или псевдонима должна существовать. Например, если клиент пытается добавить <CN=JS,DC=Example,DC=NET>, запись <DC=Example,DC=NET> не существует, а запись <DC=NET> существует, то сервер вернёт результирующий код noSuchObject, а поле matchedDN конструкции LDAPResult будет содержать <DC=NET>.

Пример запроса Add на языке Java с указанием её DN и списком её атрибутов отображает Рисунок 18.

```
@Test
public void testAddLdif1() throws Exception
{
    connection.add(
        new DefaultEntry(
            "cn=testadd,ou=system", // The Dn
            "ObjectClass: top",
            "ObjectClass: person",
            "cn: testadd_cn",
```

```
                "sn: testadd_sn" ) );

        assertTrue( connection.exists( "cn=testadd,ou=system" ) );
    }
```

Пример запроса
Рисунок 18

4.4.5.2. Ответ

При получении запроса Add сервер попытается добавить указанную запись. Результат попытки добавления будет возвращён клиенту в ответе Add Response.

Структуру ответа на запрос Add отображает Рисунок 19.

```
AddResponse ::= [APPLICATION 9] LDAPResult
```

Структура ответа
Рисунок 19

Ответ с кодом success указывает на то, что новая запись была добавлена в каталог.

4.4.6 Операция удаления Delete

Операция Delete позволяет клиенту запросить удаление записи из каталога.

4.4.6.1. Запрос

Структуру запроса Delete отображает Рисунок 20.

```
DelRequest ::= [APPLICATION 10] LDAPDN
```

Структура запроса
Рисунок 20

Запрос Delete состоит из имени записи, которую требуется удалить. Серверу не нужно выполнять разыменования псевдонимов при определении имени целевой записи для удаления.

С помощью этой операции могут быть удалены только листовые записи (у которых нет нижестоящих (подчинённых) записей).

Пример запроса Delete на языке Java с предоставлением DN записи отображает Рисунок 21.

```
@Test
public void testDeleteLeafNode() throws Exception
{
    assertTrue( session.exists( "cn=child1,cn=parent,ou=system" ) );

    try
    {
        connection.delete( "cn=child1,cn=parent,ou=system" );
    }
    catch ( LdapException le )
    {
        fail( le.getMessage() );
    }

    assertFalse( session.exists( "cn=child1,cn=parent,ou=system" ) );
}
```

Пример запроса
Рисунок 21

4.4.6.2. Ответ

При получении запроса Delete сервер попытается выполнить удаление указанной записи и результат будет возвращён клиенту в ответе Delete Response.

Структуру ответа на запрос Delete отображает Рисунок 22.

```
DelResponse ::= [APPLICATION 11] LDAPResult
```

Структура ответа
Рисунок 22

4.4.7 Операция модификации уникального имени Modify DN

Операция Modify DN позволяет клиенту изменить относительное уникальное имя (Relative Distinguished Name, RDN) записи в каталоге и/или переместить поддерево записей в новое местоположение в каталоге.

4.4.7.1. Запрос

Структуру запроса Modify DN отображает Рисунок 23.

```
ModifyDNRequest ::= [APPLICATION 12] SEQUENCE {
     entry           LDAPDN,
     newrdn          RelativeLDAPDN,
     deleteoldrdn    BOOLEAN,
     newSuperior     [0] LDAPDN OPTIONAL }
```

Структура запроса
Рисунок 23

Поля запроса Modify DN отображает Таблица 11.

Таблица 11 – Поля запроса Modify DN

Наименование атрибута	Описание	Тип
entry	Имя записи, которую требуется изменить. У этой записи могут быть, а могут и не быть нижестоящие (подчинённые) записи.	LDAPDN
newrdn	Новое RDN записи. Если запись перемещается к новой вышестоящей записи без изменения RDN, в этом поле предоставляется старое значение RDN. Значения атрибутов нового RDN, не совпавшие ни с одним из имеющихся в записи значением этих атрибутов, добавляются в эту запись, и, если добавление завершилось неудачно, возвращается соответствующая ошибка.	RelativeLDAPDN
deleteoldrdn	Логическое поле, управляющее тем, будут ли старые значения атрибутов RDN сохранены как атрибуты записи, или будут удалены из записи.	Boolean
newSuperior	При наличии этого поля в нём находится имя существующей записи-объекта, которая становится непосредственной вышестоящей (родительской) записью перемещаемой записи.	LDAPDN

Примечание: серверу не нужно выполнять какие-либо разыменования псевдонимов для определения местоположения объектов, имена которых указаны в полях entry или newSuperior.

Пример запроса ModifyDN на языке Java, где выполняется операция изменения DN с предоставленной информацией, используя соединение для записи из этого пула соединений отображает Рисунок 24.

```
public LDAPResult modifyDN(final String dn, final String newRDN,
              final boolean deleteOldRDN,
              final String newSuperiorDN)
    throws LDAPException
{
 return writePool.modifyDN(dn, newRDN, deleteOldRDN, newSuperiorDN);
}
```

Пример запроса
Рисунок 24

4.4.7.2. Ответ

При получении запроса ModifyDNRequest сервер попытается выполнить изменение имени и вернуть результат в ответе Modify DN Response.

Структуру ответа на запрос Modify DN отображает Рисунок 25.

```
ModifyDNResponse ::= [APPLICATION 13] LDAPResult
```

Структура ответа
Рисунок 25

Например, если имя записи, указанное в поле entry, было <cn=John Smith,c=US>, в поле newrdn было <cn=John Cougar Smith>, а поле newSuperior отсутствовало, то данная операция выполнила бы попытку переименовать запись в <cn=John Cougar Smith,c=US>. Если запись с таким именем уже существовала, операция бы закончилась неудачей с результирующим кодом entryAlreadyExists.

Серверы должны обеспечить, чтобы записи удовлетворяли правилам пользовательской и системной схемы данных, а также другим ограничениям модели данных. Типы атрибутов, для которых не определено соответствие equality, подчиняются правилам из раздела 2.5.1 [RFC4512] (это относится к полям newrdn и deleteoldrdn).

Объект, имя которого указано в newSuperior должен существовать. Например, если клиент пытается добавить <CN=JS,DC=Example,DC=NET>, а запись <DC=Example,DC=NET> не существует, но существует запись <DC=NET>, то сервер вернёт результирующий код noSuchObject, а поле matchedDN конструкции LDAPResult будет содержать <DC=NET>.

Если поле deleteoldrdn установлено в TRUE, значения атрибутов, формировавшие старое RDN (но не входящие в новое RDN), удаляются из записи. Если поле deleteoldrdn установлено в FALSE, значения атрибутов, формировавшие старое RDN, будут сохранены как неотличительные значения атрибутов записи.

Имейте ввиду, что стандарт X.500 накладывает ограничение: операция ModifyDN может влиять только на перемещение записей в пределах одного сервера. Если сервер LDAP отображается на DAP, то будет применяться это ограничение, и при возникновении ошибки вследствие его нарушения будет возвращаться результирующий код affectsMultipleDSAs. В общем случае, клиенты не должны рассчитывать на то, что они смогут выполнить произвольное перемещение записей и поддеревьев между серверами или контекстами именования.

4.4.8 Операция сравнения Compare

Операция Compare позволяет клиенту сравнить значение утверждения со значениями конкретного атрибута конкретной записи в каталоге.

4.4.8.1. Запрос

Структуру запроса Compare отображает Рисунок 26.

```
CompareRequest ::= [APPLICATION 14] SEQUENCE {
     entry           LDAPDN,
     ava             AttributeValueAssertion }
```

Структура запроса
Рисунок 26

Поля запроса Compare отображает Таблица 12.

Таблица 12 – Поля запроса Modify DN

Наименование атрибута	Описание	Тип
entry	Имя записи, с атрибутом которой будет производиться сравнение. Серверу не нужно выполнять какие-либо разыменования псевдонимов для определения местоположения этой записи.	LDAPDN
ava	Содержит утверждение значения атрибута, по которому будет производиться сравнение.	

Пример запроса Compare на языке Java с проверкой правильности предоставленного пароля пользователя, где password – пароль, passwordField – наименование поля LDAP, содержащего пароль отображает Рисунок 27. В случае действительного пароля функция возвращает true, иначе – false.

```
public boolean checkPassword(String userDN, String password, String passwordField)
```

```
{
  try {
    LDAPAttribute attribute = new LDAPAttribute(passwordField, password);
    return this.connection.compare(userDN, attribute);
  } catch (LDAPException e) {
    if (e.getResultCode() == LDAPException.NO_SUCH_OBJECT) {
      if (LOG.isDebugEnabled()) {
        LOG.debug("Unable to locate user_dn:" + userDN, e);
      }
    } else if (e.getResultCode() == LDAPException.NO_SUCH_ATTRIBUTE) {
      if (LOG.isDebugEnabled()) {
        LOG.debug("Unable to verify password because userPassword attribute not found.", e);
      }
    } else {
      if (LOG.isDebugEnabled()) {
        LOG.debug("Unable to verify password", e);
      }
    }
  }
  return false;
}
```

Пример запроса
Рисунок 27

4.4.8.2. Ответ

При получении запроса Compare сервер попытается выполнить заданное сравнение и вернуть результат в ответе Compare Response.

Структуру ответа на запрос Compare отображает Рисунок 28.

```
CompareResponse ::= [APPLICATION 15] LDAPResult
```

Структура ответа
Рисунок 28

Результирующий код resultCode может быть устновлен в compareTrue, compareFalse или в значение, соответствующее возникшей ошибке. Код compareTrue указывает на то, что значение утверждения в поле ava совпадает со значением типа или подтипа атрибута согласно правилу соответствия EQUALITY этого атрибута. Код compareFalse указывает на то, что значение утверждения в поле ava и значение типа или подтипа атрибута не совпадают. Другие результирующие коды указывают либо на то, что результат сравнения был Undefined, либо на возникновение какой-либо ошибки.

Имейте ввиду, что некоторые системы каталогов могут устанавливать контроль доступа, разрешающий сравнения со значениями некоторых атрибутов (таких как userPassword), при этом другие действия с этими атрибутами могут быть запрещены.

4.4.9 Операция отказа Abandon

Назначение операции Abandon — позволить клиенту запросить сервер отказаться от выполнения незавершённой операции.

4.4.9.1. Запрос

Структуру запроса Abandon отображает Рисунок 29.

```
AbandonRequest ::= [APPLICATION 16] MessageID
```

Структура запроса
Рисунок 29

Поля запроса Abandon отображает Таблица 13.

Таблица 13 – Поля запроса Abandon

Наименование атрибута	Описание	Тип
MessageID	Идентификатор сообщения той операции, которая была запрошена ранее на данном уровне сообщений LDAP. У самого запроса Abandon есть свой собственный идентификатор сообщения MessageID. Он отличается от идентификатора MessageID ранее запрошенной операции, от выполнения которой требуется отказаться.	Integer

Пример запроса Abandon на языке Java отображает Рисунок 27.

```
public boolean cancel(final boolean mayInterruptIfRunning)
{
 // Если операция уже завершена, то мы не можем ее отменить
 if (isDone())
 {
  return false;
 }
 // Отправка запроса на отмену операции
 try
 {
  cancelRequested.set(true);
  result.compareAndSet(null,
```

```
      new LDAPResult(messageID, ResultCode.USER_CANCELED,
        INFO_ASYNC_REQUEST_USER_CANCELED.get(), null,
        StaticUtils.NO_STRINGS, StaticUtils.NO_CONTROLS));
   connection.abandon(this);
  }
  catch (final Exception e)
  {
   Debug.debugException(e);
  }
  return true;
}
```

Пример запроса
Рисунок 30

4.4.9.2. Ответ

В операции Abandon не предусмотрено ответа. При получении запроса AbandonRequest сервер может отказаться от выполнения операции, идентифицируемой по MessageID. Поскольку клиент не может отличить операцию, отказ от которой выполнен успешно, от незавершенной операции, применение операции Abandon ограничено теми случаями, когда клиенту не требуется индикация результатов операции.

4.4.10 Расширенная операция Extended

Операция Extended позволяет определить дополнительные операции помимо тех, которые уже определены в протоколе; например, добавить операции для установки Transport Layer Security.

Операция Extended позволяет клиентам выполнять запросы и получать ответы с предопределёнными синтаксисами и семантиками. Эти синтаксисы и семантики могут быть определены в RFC, либо определяться в частном порядке для конкретных реализаций.

4.4.10.1. Запрос

Структура запроса Extended отображает Рисунок 31.

```
ExtendedRequest ::= [APPLICATION 23] SEQUENCE {
     requestName      [0] LDAPOID,
     requestValue     [1] OCTET STRING OPTIONAL }
```

Структура запроса
Рисунок 31

Поля запроса Extended отображает Таблица 14.

Таблица 14 – Поля запроса Extended

Наименование атрибута	Описание	Тип
requestName	содержит точечно-цифровое представление соответствующего запросу уникального идентификатора объекта OBJECT IDENTIFIER.	LDAPOID
requestValue	Содержит информацию, форма которой определяется этим запросом, инкапсулированная в строку октетов OCTET STRING.	Octet String

Пример запроса Extended на языке Java отображает Рисунок 32.

```
import netscape.ldap.*;
import java.util.*;
import java.io.*;

public class ExtOpt {
    private static String OID = "1.2.3.4";
    public static void main(String[] args) {
        try {
            UserArgs userArgs = new UserArgs("ExtOpt", args, true);
            LDAPConnection ld = new LDAPConnection();
            ld.connect(userArgs.getHost(), userArgs.getPort());
            ld.authenticate(3, userArgs.getBindDN(),userArgs.getPassword());
            System.out.println("Authenticated to directory.");

            /* Создать расширенную операцию */
            String myval = "My Value";
            byte vals[] = myval.getBytes("UTF8");
            LDAPExtendedOperation exop =
                new LDAPExtendedOperation(OID, vals);

            /* Запросить расширенную операцию с сервера */
            LDAPExtendedOperation exres = ld.extendedOperation(exop);
            System.out.println("Performed extended operation.");

            /*Получить данные из ответа, отправленного сервером */
            System.out.println("OID returned: " + exres.getID());
            String retValue = new String(exres.getValue(),"UTF8");
            System.out.println("Value returned: " + retValue);

            ld.disconnect();
        }
        catch(LDAPException e) {
            System.out.println("Error: " + e.toString());
```

```
        }
        catch(UnsupportedEncodingException e) {
            System.out.println("Error: UTF8 not supported");
        }
    }
}
```

Пример запроса
Рисунок 32

4.4.10.2. Ответ

На этот запрос сервер ответит сообщением LDAPMessage, содержащим ответ ExtendedResponse.

Структуру ответа на запрос Extended отображает Рисунок 33.

```
ExtendedResponse ::= [APPLICATION 24] SEQUENCE {
     COMPONENTS OF LDAPResult,
     responseName     [10] LDAPOID OPTIONAL,
     responseValue    [11] OCTET STRING OPTIONAL }
```

Структура ответа
Рисунок 33

Поля ответа на запрос Extended отображает Таблица 15.

Таблица 15 – Поля ответа на запрос Extended

Наименование атрибута	Описание	Тип
responseName	При его наличии, содержит идентификатор LDAPOID, уникальный для данной расширенной операции или её ответа. Это поле является опциональным (даже когда в спецификации расширения определён LDAPOID для использования в этом поле). Это поле будет отсутствовать, когда сервер не может или не желает определить, какой именно LDAPOID вернуть, например, если невозможно разобрать requestName, либо его значение не распознано. Если не распознан идентификатор в requestName, сервер возвращает protocolError. (Сервер может вернуть protocolError и в других случаях.)	LDAPOID

Наименование атрибута	Описание	Тип
responseValue	Содержит связанную с операцией информацию. Формат поля определяется спецификацией операции-расширения Extended. Реализации должны быть готовы обработать произвольное содержимое поля, в том числе и нулевой длины. Значения, которые определяются в терминах ASN.1 и закодированы BER также следуют правилам расширяемости.	Octet String

4.4.11 Операция StartTLS

Предназначение операции Start Transport Layer Security (StartTLS) — инициировать установление уровня TLS. Операция StartTLS определена с использованием механизма расширенной операции Extended.

4.4.11.1. Запрос

Клиент запрашивает установление TLS путём отправки серверу сообщения с запросом StartTLS. Запрос StartTLS определяется в терминах запроса операции-расширения ExtendedRequest. Идентификатор requestName — "1.3.6.1.4.1.1466.20037", а поле requestValue всегда отсутствует.

Клиент не должен посылать вслед за этим запросом каких-либо LDAP PDU на данном уровне сообщений LDAP до получения ответа операции-расширения StartTLS, и, в случае ответа с успешным статусом, до завершения переговоров TLS.

При обнаружении проблем с последовательностью выполнения операций (в частности тех, которые описаны в разделе 3.1.1 [RFC4513]) результирующий код resultCode ответа должен быть установлен в operationsError.

Если сервер не поддерживает TLS (в силу конструктивных особенностей или текущих настроек), он возвращает ответ с результирующим кодом resultCode, установленным в protocolError.

Запускает протокол Transport Layer Security (TLS) в этом соединении для включения конфиденциальности сеанса.

Пример запроса запуска Transport Layer Security (TLS) в этом соединении для включения конфиденциальности сеанса на языке Java отображает Рисунок 34.

```
import netscape.ldap.*;
import java.util.*;
import java.io.*;
```

```
public class ExtOpt {
    private static String OID = "1.2.3.4";
    public static void main(String[] args) {
        try {
            UserArgs userArgs = new UserArgs("ExtOpt", args, true);
            LDAPConnection ld = new LDAPConnection();
            ld.connect(userArgs.getHost(), userArgs.getPort());
            ld.authenticate(3, userArgs.getBindDN(),userArgs.getPassword());
            System.out.println("Authenticated to directory.");

            /* Создать расширенную операцию */
            String myval = "My Value";
            byte vals[] = myval.getBytes("UTF8");
            LDAPExtendedOperation exop =
                new LDAPExtendedOperation(OID, vals);

            /* Запросить расширенную операцию с сервера */
            LDAPExtendedOperation exres = ld.extendedOperation(exop);
            System.out.println("Performed extended operation.");

            /*Получить данные из ответа, отправленного сервером */
            System.out.println("OID returned: " + exres.getID());
            String retValue = new String(exres.getValue(),"UTF8");
            System.out.println("Value returned: " + retValue);

            ld.disconnect();
        }
        catch(LDAPException e) {
            System.out.println("Error: " + e.toString());
        }
        catch(UnsupportedEncodingException e) {
            System.out.println("Error: UTF8 not supported");
        }
    }
}
```

Пример запроса
Рисунок 34

4.4.11.1.1. Снятие уровня TLS

Как клиент, так и сервер может снять уровень TLS (прекратить его использование) и продолжить работу с "чистым" уровнем сообщений LDAP путём отправки и получения оповещения о закрытии TLS (TLS closure alert).

Сторона-инициатор посылает оповещение о закрытии TLS и должна ожидать, пока не получит оповещение о закрытии TLS от другой стороны, прежде чем посылать дальнейшие LDAP PDU.

Когда одна из сторон получает начальное оповещение о закрытии TLS, она может избрать вариант продолжения работы с "чистым" уровнем сообщений LDAP. В этом случае она должна немедленно передать оповещение о закрытии TLS. Вслед за этим она может посылать и принимать LDAP PDU.

Стороны могут завершить данную сессию LDAP после отправки и получения оповещения о закрытии TLS.

4.4.11.2. Ответ

При получении запроса StartTLS, сервер, поддерживающий данную операцию, должен вернуть запрашивающему клиенту ответное сообщение StartTLS. Идентификатор responseName, если таковой предоставляется (смотрите раздел 4.12), — "1.3.6.1.4.1.1466.20037". Поле responseValue всегда отсутствует.

Если сервер желает и способен вести переговоры TLS, он возвращает ответ StartTLS, результирующий код resultCode которого установлен success. При получении клиентом ответа StartTLS с успешным статусом стороны могут начать переговоры TLS как описано в разделе 3 [RFC4513].

В противном случае, если сервер не желает или не способен выполнить данную операцию, он должен вернуть соответствующий результирующий код, указывающий на характер проблемы. Например, если подсистема TLS в настоящее время не доступна, сервер может указать на это путём возврата сообщения с результирующим кодом resultCode, установленным в unavailable. В тех случаях, когда возвращён ответ с неуспешным результирующим кодом, сессия LDAP продолжается без уровня TLS.

4.5. Коды ошибок

LDAPResult — это конструкция, используемая в данном протоколе для возврата индикации успешного или неудачного завершения операции от сервера клиенту. На различные запросы сервер будет возвращать ответы, содержащие элементы из конструкции LDAPResult, для индикации финального статуса запроса операции протокола.

Коды LDAPResult отображает Рисунок 35.

```
LDAPResult ::= SEQUENCE {
     resultCode         ENUMERATED {
          success                      (0),
          operationsError              (1),
          protocolError                (2),
```

```
          timeLimitExceeded            (3),
          sizeLimitExceeded            (4),
          compareFalse                 (5),
          compareTrue                  (6),
          authMethodNotSupported       (7),
          strongerAuthRequired         (8),
               -- 9 зарезервирован --
          referral                     (10),
          adminLimitExceeded           (11),
          unavailableCriticalExtension (12),
          confidentialityRequired      (13),
          saslBindInProgress           (14),
          noSuchAttribute              (16),
          undefinedAttributeType       (17),
          inappropriateMatching        (18),
          constraintViolation          (19),
          attributeOrValueExists       (20),
          invalidAttributeSyntax       (21),
               -- 22-31 не используются --
          noSuchObject                 (32),
          aliasProblem                 (33),
          invalidDNSyntax              (34),
               -- 35 зарезервирован для неопределённого isLeaf --
          aliasDereferencingProblem    (36),
               -- 37-47 не используются --
          inappropriateAuthentication  (48),
          invalidCredentials           (49),
          insufficientAccessRights     (50),
          busy                         (51),
          unavailable                  (52),
          unwillingToPerform           (53),
          loopDetect                   (54),
               -- 55-63 не используются --
          namingViolation              (64),
          objectClassViolation         (65),
          notAllowedOnNonLeaf          (66),
          notAllowedOnRDN              (67),
          entryAlreadyExists           (68),
          objectClassModsProhibited    (69),
               -- 70 зарезервирован для CLDAP --
          affectsMultipleDSAs          (71),
               -- 72-79 не используются --
          other                        (80),
          ... },
     matchedDN          LDAPDN,
     diagnosticMessage  LDAPString,
     referral           [3] Referral OPTIONAL }
```

Коды LDAPResult
Рисунок 35

Поля кода LDAPResult отображает Таблица 16.

Таблица 16 – Поля кода LDAPResult

Наименование атрибута	Описание	Тип
LDAPResult	Нумерация кодов resultCode является расширяемой, как определено в разделе 3.8 RFC4520. Если сервер обнаруживает при выполнении операции несколько ошибок, возвращается только один результирующий код. Сервер должен вернуть результирующий код, который наилучшим образом отражает характер возникшей ошибки. Серверы могут возвращать подмененные результирующие коды для предотвращения несанкционированного сбора сведений злоумышленниками. Значение перечисленных результирующих кодов дано в таблице ниже.	ENUMERATED
diagnosticMessage	Поле данной конструкции diagnosticMessage может, по усмотрению сервера, быть использовано для возврата строки, содержащей читабельное текстовое диагностическое сообщение (в нём следует избегать символов управления терминалом и форматирования страницы). Поскольку данное диагностическое сообщение не стандартизировано, реализации не должны полагаться на возвращаемые в этом поле значения. Обычно диагностические сообщения дополняют коды resultCode дополнительной информацией. Если сервер решает не возвращать текстовой диагностики, поле diagnosticMessage должно быть пустым.	LDAPString

Наименование атрибута	Описание	Тип
matchedDN	Для некоторых результирующих кодов (как правило, noSuchObject, aliasProblem, invalidDNSyntax и aliasDereferencingProblem, но не ограничиваясь только ими), в поле matchedDN помещается (по результатам контроля доступа) имя последней записи (объекта или псевдонима), которое используется для нахождения целевого (или базового) объекта. Это будет усечённая форма предоставленного имени, либо, если при попытке нахождения записи был разыменован псевдоним, то результирующего имени. В противном случае поле matchedDN остаётся пустым.	LDAPDN

4.5.1 Отсылка (Referral)

Результирующий код referral указывает на то, что запрашиваемый сервер не может или не желает исполнять операцию, и что один или несколько других серверов могут быть в состоянии это сделать. Причинами этого могут быть:

Целевая запись запроса не хранится локально, но у сервера есть сведения о её возможном существовании в другом месте.

На данном сервере установлено ограничение на такие операции, возможно производится попытка изменить копию записи, предназначенную только для чтения.

Поле referral присутствует в конструкции LDAPResult, если поле resultCode установлено в referral, при всех остальных результирующих кодах оно отсутствует. Оно содержит одну или несколько отсылок на один или несколько серверов или сервисов, которые могут быть доступны посредством LDAP или других протоколов. Отсылки могут быть возвращены в ответ на запрос какой-либо операции (за исключением Unbind и Abandon, которые не возвращают ответов). В поле referral должен присутствовать по меньшей мере один URI.

Во время операции поиска Search, после нахождения базового объекта baseObject и оценки записей, отсылка referral не возвращается. Вместо этого, если для завершения операции требуется обратиться к другим серверам, возвращаются ссылки-продолжения (continuation references), описанные в разделе 4.5.3.

Структуру Referral отображает Рисунок 36.

```
Referral ::= SEQUENCE SIZE (1..MAX) OF uri URI

URI ::= LDAPString     -- ограничена набором символов,
                       -- разрешённых в URI
```

Структура Referral
Рисунок 36

Если клиент желает продолжить выполнение операции, он обращается к одному из поддерживаемых сервисов, перечисленных в отсылке. Если присутствует несколько URI, клиент предполагает, что для продолжения операции можно использовать любой из предоставленных URI.

Клиенты, следующие по отсылкам, должны принимать меры по предотвращению зацикливания между серверами. Они не должны повторно обращаться к одному и тому же серверу с одними и теми же запросами с указанием в них одинаковых параметров. Некоторые клиенты используют счётчик, увеличивающийся на единицу каждый раз, когда во время операции происходит обработка отсылки. Клиенты такого типа должны быть способны обрабатывать по крайней мере десять вложенных отсылок во время выполнения операции.

URI для сервера, реализующего LDAP и доступного по TCP/IP (v4 или v6) [RFC793][RFC791], записывается в виде LDAP URL в соответствии с [RFC4516].

Значения Referral, которые являются LDAP URL, подчиняются следующим правилам:

Если происходит разыменование псевдонима, в LDAP URL должна присутствовать часть <dn> с именем нового целевого объекта.

Во избежание неоднозначности рекомендуется (RECOMMENDED) наличие части <dn>.

Если часть <dn> присутствует, клиент использует это имя в следующем запросе, выполняемом для продолжения операции, а если она не присутствует, клиент использует то же имя, что и в оригинальном запросе.

Некоторые серверы (например, участвующие в распределённом индексировании) могут предоставлять различные фильтры в URL отсылок для операций поиска Search.

Если в LDAP URL присутствует часть <filter>, клиент использует этот фильтр в следующем запросе, выполняемом для продолжения данной операции Search, а если она не присутствует, клиент использует тот же самый фильтр, который использовался для этой операции Search.

Для операций поиска Search рекомендуется наличие части <scope> во избежание неоднозначности.

Если часть <scope> отсутствует, при продолжении операции клиент использует диапазон, указанный в оригинальной операции Search.

Другие аспекты нового запроса могут как совпадать, так и отличаться от параметров того запроса, который породил отсылку.

Могут быть возвращены и другие типы URI. Синтаксис и семантика таких URI оставлены как предмет будущих спецификаций. Клиенты могут игнорировать неподдерживаемые ими URI.

Закодированные в UTF-8 символы, добавляемые в строковое представление DN, поискового фильтра или других полей в значении поля referral, могут быть неразрешёнными для использования в URI (например, пробелы) и должны быть экранированы с помощью % по методу, описанному в [RFC3986].

4.5.2 Результирующие коды LDAP

Значения кодов ошибок отображает Таблица 17.

Таблица 17 – Значения кодов ошибок

Код	Значение
success (0)	Указывает на успешное выполнение операции. Примечание: этот код не используется с операцией Compare. Смотрите compareFalse (5) и compareTrue (6).
operationsError (1)	Указывает на то, что операция нарушает последовательность выполнения по отношению к другим операциям (того же или другого типа). Например, данный код возвращается, когда клиент пытается выполнить StartTLS [RFC4346], в то время как другие незавершенные операции ещё выполняются или уровень TLS уже был установлен.

Код	Значение
protocolError (2)	Указывает на то, что сервер получил неправильно сформированные данные. Применительно к операции Bind этот код также используется для указания на то, что сервер не поддерживает запрашиваемую версию протокола. Применительно к операции Extended этот код также используется для указания на то, что сервер не поддерживает (в силу конструктивных особенностей или текущих настроек) ассоциированную с полем requestName операцию-расширение. Применительно к запросам операций с указанием нескольких элементов управления, этот код может использоваться для индикации того, что сервер не может игнорировать порядок, в котором указаны эти элементы управления, либо того, что комбинация указанных элементов управления неверна или не определена.
timeLimitExceeded (3)	Указывает на то, что определённое клиентом ограничение по времени было превышено до завершения операции.
sizeLimitExceeded (4)	Указывает на то, что определённое клиентом ограничение по размеру было превышено до завершения операции.
compareFalse (5)	Указывает на то, что операция Compare успешно выполнена и утверждение оценено как FALSE или Undefined.
compareTrue (6)	Указывает на то, что операция Compare успешно выполнена и утверждение оценено как TRUE.
authMethodNotSupported (7)	Указывает на то, что метод или механизм аутентификации не поддерживается.
strongerAuthRequired (8)	Указывает, что для выполнения операции сервер требует более строгой аутентификации. При использовании с уведомлением об отключении данный код указывает на то, что сервер определил неожиданный обрыв или компрометацию установленного защищённого соединения между клиентом и сервером.
referral (10)	Указывает на то, что для выполнения операции необходимо проследовать по отсылке (Referral).

Код	Значение
adminLimitExceeded (11)	Указывает на то, что были превышены административные ограничения.
unavailableCriticalExtension (12)	Указывает, что критичный элемент управления не распознан (Controls).
confidentialityRequired (13)	Указывает на то, что требуется защита конфиденциальности данных.
saslBindInProgress (14)	Указывает, что для продолжения процесса аутентификации сервер требует от клиента нового запроса на подсоединение с тем же самым механизмом SASL.
noSuchAttribute (16)	Указывает на то, что запись с заданным именем не содержит указанного атрибута или значения атрибута.
undefinedAttributeType (17)	Указывает на то, что описание атрибута в поле запроса не распознано.
inappropriateMatching (18)	Указывает на то, что была предпринята попытка использовать (например, в утверждении) правило соответствия, не определённое для того типа атрибута, который участвует в операции.
constraintViolation (19)	Указывает на то, что клиент предоставил значение атрибута, не удовлетворяющее ограничениям, налагаемым на него моделью данных. Например, данный код возвращается, когда для атрибута с ограничением SINGLE-VALUE было предоставлено несколько значений.
attributeOrValueExists (20)	Указывает на то, что клиент предоставил для добавления в запись атрибут или значение, но эти атрибут или значение уже существуют.
invalidAttributeSyntax (21)	Указывает на то, что предполагаемое значение атрибута не соответствует синтаксису атрибута.
noSuchObject (32)	Указывает на то, что такого объекта в DIT не существует.
aliasProblem (33)	Указывает на то, что возникла проблема с псевдонимом. Например, данный код может использоваться для индикации того, что полученное в результате разыменования псевдонима имя не указывает на объект.

Код	Значение
invalidDNSyntax (34)	Указывает на то, что значения поля типа LDAPDN или RelativeLDAPDN запроса (например, search base, target entry, ModifyDN newrdn, и т.п.) не удовлетворяют требуемому синтаксису или содержат значения атрибутов, не удовлетворяющие синтаксису этих типов атрибутов.
aliasDereferencingProblem (36)	Указывает на то, что при разыменовании псевдонима возникла проблема. Как правило, псевдоним встретился в ситуации, когда это не позволяется, либо доступ к нему запрещён.
inappropriateAuthentication (48)	Указывает, что сервер требует от клиента, который попытался подсоединиться анонимно или без предоставления данных аутентификации, предоставить эти данные в той или иной форме.
invalidCredentials (49)	Указывает на то, что предоставленные данные аутентификации (например, имя пользователя и пароль) неверны.
insufficientAccessRights (50)	Указывает на то, что у клиента нет достаточных прав доступа для выполнения данной операции.
busy (51)	Указывает на то, что сервер слишком занят для обслуживания данной операции.
unavailable (52)	Указывает на то, что сервер находится в стадии выключения или необходимая для выполнения операции подсистема недоступна.
unwillingToPerform (53)	Указывает на то, что сервер не желает исполнять данную операцию.
loopDetect (54)	Указывает на то, что сервер обнаружил зацикливание (например, в процессе разыменования псевдонимов или при выполнении сцепления).
namingViolation (64)	Указывает на то, что имя записи нарушает ограничения именования.
objectClassViolation (65)	Указывает на то, что запись нарушает ограничения объектного класса.
notAllowedOnNonLeaf (66)	Указывает на то, что операция выполняет ненадлежащее действие над нелистовой записью.

Код	Значение
notAllowedOnRDN (67)	Указывает на то, что операция производит неуместную попытку удалить значение, формирующее относительное уникальное имя записи.
entryAlreadyExists (68)	Указывает на то, что запрос (на добавление, перемещение или переименование) не может быть исполнен, поскольку целевая запись уже существует.
objectClassModsProhibited (69)	Указывает на то, что попытка модификации объектного класса (классов) в атрибуте "objectClass" записи не разрешается. Например, данный код возвращается при попытке клиента модифицировать структурный объектный класс записи.
affectsMultipleDSAs (71)	Указывает на то, что операция не может быть выполнена, поскольку она затрагивает несколько серверов (DSA).
other (80)	Указывает, что на сервере произошла внутренняя ошибка.

ПРИЛОЖЕНИЕ

ПЕРЕЧЕНЬ ИЗМЕНЕНИЙ ПРИКЛАДНОГО ПРОГРАММНОГО ИНТЕРФЕЙСА

Редакция	*Дата*	*Версия*	*Список внесенных изменений*
1	04.03.2022	1	Первая редакция

ПЕРЕЧЕНЬ ТЕРМИНОВ

В настоящем документе использованы следующие термины:

1) Программное изделие — программа на носителе данных, являющаяся продуктом промышленного производства.

2) Средство вычислительной техники (СВТ) — ПЭВМ (персональная электронно-вычислительная машина) либо другое вычислительное оборудование (мэйнфрейм, мини-ЭВМ, микро-ЭВМ, КПК (карманный персональный компьютер), компьютерный терминал).

3) СВТ индивидуального пользования — вычислительное оборудование, обеспечивающее доступ отдельного пользователя к информационным сервисам, предоставляемым программным изделием:

– Сервер (стоечный или отдельно стоящий).

– Многомашинный вычислительный комплекс (ММВК), то есть серверный кластер.

4) СВТ коллективного пользования — вычислительное оборудование, предназначенное для реализации программным изделием информационных сервисов, предоставляемых всем пользователям, имеющим доступ:

– Автоматизированное рабочее место (АРМ) на базе ПЭВМ.

– Портативный компьютер (ноутбук).

5) Мобильное СВТ — вычислительное оборудование повышенной портативности:

– Карманный персональный компьютер (КПК).

– КПК со встроенным модулем мобильной связи — смартфоны и коммуникаторы.

6) Система управления базами данных — совокупность программных и языковых средств, обеспечивающих управление базами данных.

7) Прикладной программный интерфейс — совокупность методов, позволяющих стороннему программному средству получить доступ к функциям программного изделия, обеспечивающим обработку данных, управление ОС, контроль над СВТ и т.д. Английское наименование термина — application programming interface, сокращенно ППИ.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

АРМ	Автоматизированное рабочее место
АС	Автоматизированная система
АСУТП	Автоматизированная система управления технологическим процессом
БД	База данных
КПК	Карманный персональный компьютер
ММВК	Многомашинный вычислительный комплекс
ОС	Операционная система
ППИ	Прикладной программный интерфейс
ПК	Программный комплекс
ПО	Программное обеспечение
ПС	Программное средство
ПЭВМ	Персональная электронно-вычислительная машина
СВТ	Средство вычислительной техники
СПО	Специальное программное обеспечение
СУБД	Система управления базами данных
ППИ	Application programming interface, программный интерфейс приложения
LDAP	(Lightweight Directory Access Protocol — «легковесный протокол доступа к каталогам») — протокол прикладного уровня для доступа к службе каталогов X.500

ПЕРЕЧЕНЬ РИСУНКОВ

ПЕРЕЧЕНЬ ТАБЛИЦ