Министерство образования и науки РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рыбинский государственный авиационный технический университет

имени П.А. Соловьева»

Факультет радиоэлектроники и информатики

Кафедра Математическое и программное обеспечение ЭВС

**ОТЧЕТ**

**по производственной практике**

Студент группы ИПБ-15 Козин О.И.

*(Код) (Подпись, дата) (Фамилия И. О.)*

Руководитель Волков М.А.

*(Уч. степень, звание) (Подпись, дата) (Фамилия И. О.)*

Оценка

*(Подпись, дата)*

Рыбинск 2018

Оглавление

[Введение 4](#_Toc9414537)

[1.1 Формулировка темы выпускной квалификационной работы 5](#_Toc9414538)

[1.2 Анализ предметной области 6](#_Toc9414539)

[1 Анализ предметной области 6](#_Toc9414540)

[1.1 Описание предметной области 6](#_Toc9414541)

[1.2 Формулировка проблемы и актуальность 13](#_Toc9414542)

[1.3 Обзор аналогов 13](#_Toc9414543)

[1.3 Обзор аналогов 15](#_Toc9414544)

[1.4 Актуальность 16](#_Toc9414545)

[1.5 Цели и задачи 17](#_Toc9414546)

[2.1 Описание функциональных и технологических требований 18](#_Toc9414547)

[Техническое задание на разработку ПО 19](#_Toc9414548)

[3.1 Постановка задачи 19](#_Toc9414549)

[3.2 Функциональные требования 19](#_Toc9414550)

[3.2.1 Клиентская часть 19](#_Toc9414551)

[3.2.2 Серверная часть 19](#_Toc9414552)

[3.3 Технические требования 19](#_Toc9414553)

[3.3.1 Системные требования к серверу 19](#_Toc9414554)

[3.3.2 Системные требования к клиенту 19](#_Toc9414555)

[3.3.3 Требования отказоустойчивости 19](#_Toc9414556)

[3.4 План выполнения практики 19](#_Toc9414557)

[4 Описание методики тестирования 21](#_Toc9414558)

[Приложения 22](#_Toc9414559)

[5 Вывод 27](#_Toc9414560)

[6 Список литературы 28](#_Toc9414561)

# Введение

В современном мире, с его бурным развитием информационных технологий, с каждым днём становится всё больше и больше сфер подверженных автоматизации с помощью программных продуктов.

Количество информации растёт, её ценность повышается. Вопрос защиты информации от несанкционированного доступа важен как никогда. Кроме того, увеличение количества инструментов на основе программных продуктов требует усовершенствования механизмов доступа к ним пользователей с целью повышения удобства и безопасности.

Особенно это актуально в сфере коммерческого программного обеспечения, в которой необходимо обеспечение защищённости информации как от внешних угроз, так и поддержание ограничений на доступ к данным по причине разделения полномочий внутри круга доверенных лиц.

Для решения этих проблем существуют различные протоколы аутентификации и авторизации. Каждый из них имеет свои преимущества и недостатки, а подходы к реализации могут кардинально отличатся. Поэтому в условиях сложной инфраструктуры, большого количества пользователей и разнообразных ресурсов, а также наличия устаревших решений, несколько протоколов могут существовать одновременно. А это, в свою очередь усложняет работу пользователей из-за необходимости иметь несколько учётных записей.

1 Формулировка темы выпускной квалификационной работы

Тема выпускной квалификационной работы: Разработка системы централизованной авторизации на основе LDAP и современных протоколов аутентификации.

2 Анализ предметной области

2.1 Описание предметной области

Процессы аутентификации и авторизации позволяют реализовать разделения прав доступа и защиту конфиденциальной информации от внешних угроз. Это необходимо в большинстве программных продуктов, особенно коммерческих. Также реализация конкретной системы может значительно повысит удобство работы пользователей, что зачастую ведёт ещё большему увеличению безопасности.

Для успешного прохождения авторизации необходимо пройти несколько этапов:

1) Идентификация – заявление о том, кем является пользователь, обычно выражается в предоставлении своего уникального идентификатора-имени.

2) Аутентификация – проверка подлинности пользователя. То есть действительно ли пользователь тот, за кого себя выдаёт. Самый распространённый способ — это подтверждение с помощью секретного пароля. Кроме того, аутентификация может быть односторонней и двусторонней (взаимной). При односторонней, проверки требует только тот, кто запрашивает доступ. При двусторонней запрашиваемый ресурс тоже должен подтвердить себя.

3) Авторизация – выдача соответствующих прав доступа к запрашиваемому ресурсу. Этот процесс позволяет использовать результаты работы аутентификации и выдать пользователю только то, что ему разрешается согласно его статусу.

Успешное последовательное прохождение этих этапов позволяет гарантировать защищённость информации, к которой осуществляется доступ.

Но, так как информация имеет ценность, то найдутся люди, желающие ей завладеть. При этом они могут использовать различные методы чтобы нарушить нормальное функционирование этапов авторизации или получить доступ к конфиденциальной информации напрямую. Поэтому появляется необходимость в полной защите данных: как при хранении, так и при передаче.

На практике процессы авторизации обычно происходят с использованием компьютерной сети, которая не может быть полностью ограждена от физического доступа злоумышленника. В таком случае приходится учитывать, что любая информация, передаваемая по сети может быть известна кому угодно. Следовательно, необходимо использовать методы, позволяющие преобразить данные так, что даже при их перехвате неавторизированным пользователем конфиденциальность не будет нарушена.

Такие методы предоставляет криптография. Криптография - это наука об обеспечении безопасности данных. Она занимается поисками решений четырех важных проблем безопасности - конфиденциальности, аутентификации, целостности и контроля участников взаимодействия. Основным способом решения этих проблем является шифрование. Шифрование. Шифрование - это преобразование данных определённым способом-алгоритмом в нечитабельную форму, с возможностью возвращения исходного состояния, используя специальные ключи шифрования.

На практике приходится считаться со сложными структурами данных и с аппаратные ограничениями, поэтому отдельных алгоритмов шифрования недостаточно. Нужны комплексные решения описывающие все этапы передачи данных, способные надёжно и эффективно решать проблему защиты информации. Такими решениями являются криптосистемы.

Они включают в себя набор правил и алгоритмов, регламентирующих манипуляции с данными, а также способы создания и распространения ключей. Криптографические системы можно классифицировать по типу используемых алгоритмов шифрования:

1. На основе симметричных алгоритмов.
2. На основе асимметричных алгоритмов.

Симметричные криптосистемы, также называемые системами с секретным ключом – способ шифрования, в котором один ключ используется как для шифрования, так и для расшифровывания. Является первым подходом к сокрытию информации изобретённым человеком.

Главной особенностью симметричной криптосистемы является то, что что ключ алгоритма шифрования должен оставаться в секрете от посторонних. Поэтому должна обеспечиваться защита ключа на протяжении всего его существования.

Кроме возможности потери конфиденциальности ключа, также существует опасность применения криптоанализа к зашифрованным данным. Поэтому выбранный алгоритм шифрования должен быть достаточно стойким, для недопущения расшифровки информации за приемлемое время.

Для симметричных алгоритмов уровень криптостойкости, обычно, определяется размером ключа шифрования, что эквивалентно полному перебору его значений.

Для современных систем желателен ключ длинной не менее 128 двоичных разрядов (128 бит).

Асимметричные криптосистемы в своей работе используют пару ключей: закрытый и открытый. Открытый ключ применяется для шифрования и может передаваться по незащищённому каналу связи. Расшифровка производится с помощью закрытого ключа, который должен оставаться в секрете.

Реализация асимметричных алгоритмов шифрования основана на применении односторонних функций, то есть таких функций, что, зная несложно найти значение F(x), но вычисление x по F(x) невозможно за приемлемое время. Теоретически, зная F(x), x можно подобрать, перебирая все возможные значения x и применять к ним функцию, до тех пор, пока результат не совпадёт с F(x). Однако практически, при достаточной размерности множества X, выполнить такой перебор, за приемлемое время, невозможно.

Классический пример такой функции – это функция основанные на принципе обратного модульного возведения в степень, также называемые задаче дискретного логарифмирования.

Но получается, что односторонняя функция не может использоваться в качестве функции шифрования. Так как хоть она и позволяет надёжно зашифровать данные, но расшифровать их некто не сможет. Для решения этой проблемы используются односторонние функции с секретом. Это особый вид функций, которые имеют некоторый секрет, позволяющий быстро вычислить обратное значение функции. Именно на этом принципе основаны большинство асимметричных криптографических систем.

В итоге оба подхода имеют свои преимущества и недостатки. Симметричные системы проще в реализации, за счёт более простых операций, а скорость шифрования в среднем на несколько порядков выше, чем у аналогичных по защищённости систем с открытым ключом. Единственным серьёзным недостатком является сложность обмена секретного ключа. Поэтому для применения симметричных систем на практике необходимо решить проблему надёжного распределения ключей между участниками процесса передачи данных.

При работе в публичных сетях, требуется быть уверенным в подлинности ключа. А именно в том, что полученный от пользователя ключ действительно сгенерирован им. Такой функционал может предоставить механизм Электронной подписи.

Обычно, в большинстве реальных систем, электронная подпись основана на принципах асимметричного шифрования, только применяет их иначе. Так же генерируется пара ключей: открытый, закрытый. Но для шифрования (подписи) данных применяется закрытый, соответственно для расшифровки (проверки подписи) открытый. Такой подход позволяет любому владельцу открытого ключа проверять подпись, а возможность подписать остаётся только за хранителем закрытого ключа. Таким образом проведя подписание каких-либо информации, мы генерируем электронную подпись и можем отправлять её вместе с исходными данными. Далее, зная отправителя и его открытый ключ, можно применить его к подписи и получить исходный файл.

Но получается, что каждый обладатель открытого ключа может получить из подписи оригинальные данные, а они должны быть секретны. Кроме того, подписанный файл по размеру будет сопоставим с оригинальным, что удвоит нагрузку на передающую среду. Поэтому в электронная подпись невозможна без хеш-функций.

Хеш-функция – осуществляет преобразование (хеширование) набора входных данных произвольной длинны, в выходную последовательность определённой длинны. При этом для каждого набора результат хеширования будет уникальным, а провести обратное преобразование невозможно.

Таким образом подписывая не сами данные, а их хеш, общедоступным окажется именно хеш, что безопасно. И размер подписи теперь возможно уменьшить, выбрав подходящую хещ-функцию.

Одной только проверки подлинности ключей недостаточно для сохранения конфиденциальности информации. Кроме этого, очевидно, необходимо скрывать сами значения ключей. Это можно сделать если использовать асимметричное шифрование на ключи. Так как размеры ключей небольшие, то потеря ресурсов на зашифровку-расшифровку будет незначительной.

Используя цифровую подпись и асимметричное шифрование можно реализовать безопасное распределение ключей для симметричных криптосистем.

Но такая система связи остаётся уязвимой для злоумышленника, который представляется доверенным пользователем, но отдаёт свой открытый ключ асимметричного шифрования, таким образом подменяя его.

Для решения этой проблемы открытый ключ доверенного пользователя вместе с сопроводительной информацией: именем, сроком действия и прочим подписывается центром сертификации. Предполагается, что центр сертификации честный и не подпишет ключ злоумышленника. А также то, что центр сертификации распространяет свой открытый ключ максимально широко, и желающие пройти аутентификацию, ещё до инициализации процесса обмена ключами, будут иметь открытый ключ такого центра, и злоумышленник ничего не сможет с этим поделать.

Этот процесс называется сертификацией. А подписанные центром сертификации открытые ключи, включающие в себя также прочие данные о владельце, называются цифровыми сертификатами.

Таким образом современные протоколы аутентификации и авторизации должны включать в себя множество механизмов обеспечения безопасности. Одним из таких является Kerberos.

Kerberos может предоставлять как сетевую аутентификацию, так и безопасную авторизацию без необходимости повторного ввода пароля или предоставления других удостоверяющих данных. Поэтому он способет реализовать технологию единого входа (Single Sign-on).

Особенностью Kerberos является то, что пароли или иные удостоверяющие данные никогда не пересылаются по сети. Это достигается благодаря наличию третьей доверенной стороны – центра распространения ключей (key distribution center (KDC)). KDC – хранит все учётные данные пользователей и сервисов, и предоставляет функции аутентификации. Для обеспечения безопасности протоколом предусмотрен следующий формат проведения аутентификации:

1. Пользователь, желая аутентифицироваться, обращается к KDC и передаёт ему в открытом виде данные о себе и о желаемом ресурсе.
2. KDC, посылает в ответ разрешение на получение разрешения (TGT), которое интерпретируется клиентом как неразбираемый набор бит, а также ключ сессии. Этот ключ генерируется случайным образом и зашифровывается секретным ключом пользователя, хранящимся в базе данных KDC.
3. Теперь пользователь, применяя свой закрытый ключ может получить сессионный ключ. И зашифровав им TGT, отправляет обратно на KDC.
4. KDC применяет сохранённый сессионный ключ получает TGT и если он соответствует исходному, значит пользователь тот за кого себя выдаёт. Генерируется новое сообщение. В него записываются данные необходимые для аутентификации и авторизации (опционально) пользователя сервисом – Service Ticket (ST). А также генерируется новый сессионный ключ, но на этот раз для взаимодействия сервиса и пользователя. Этот ключ вместе с ST шифруется закрытым ключом это сервиса. А чтобы пользователь мог получить доступ к нему, то его копия шифруется сессионным ключом пользователя. Всё вместе отправляется пользователю.
5. Пользователь, расшифровав ключ для общения с сервисом запоминает его. И посылает ST сервису.
6. Сервис расшифровывает ST своим секретным ключом, оттуда же берёт и сессионный ключ. По полученным данным сервис может решать какой доступ предоставить.

OpenId Connect – это надстройка над протоколом авторизации OAuth 2.0, предоставляющая функции аутентификации. OIDC предполагает наличие отдельного доверенного сервера аутентификации, благодаря чему поддерживает технологию SSO, позволяет отделить функционал по аутентификации от конкретных приложений и гарантирует безопасность. Очень популярный протокол, поддерживается практически всеми крупными компаниями, например: Google, Facebook.

Для передач данных в процессе аутентификации используется стандарт JSON Web Token (JWT). Это открытии стандарт, описывает формат сообщений в виде JSON файлов, называемых токенами. Согласно нему токен состоит из заголовка, хранящего собственное описание, основной части, в которой хранится пользовательская информация, а также электронной подписи, удостоверяющей весь файл.

Процесс авторизации, в общем случае, выглядит следующим образом:

1. Пользователь запрашивает ресурс.
2. Ресурс перенаправляет пользователя на сервер аутентификации.
3. Пользователь вводит свои учётные данные на форме сервера аутентификации.
4. Сервер аутентификации проверяет данные пользователя по своей базе, и вы даёт токен доступа (access token).
5. Имея access token пользователь может получить доступ к ресурсу.

2.2 Формулировка проблемы и актуальность

Проблематика заключается в необходимость ООО «БиАйВи» в агрегации различных протоколов аутентификации в одну систему. С предоставлением пользователям этих протоколов единого интерфейса аутентификации.

Актуальность состоит в потребности ООО «БиАйВи» в повышении гибкости системы аутентификации. А также повышенные безопасности путём внедрения современных протоколов аутентификации.

2.3 Обзор аналогов

KeyCloak - это open-source сервер аутентификации и управления учетными записями, построенный на базе спецификаций OAuth 2.0, Open ID Connect, JSON Web Token и SAML 2.0. Разрабатывается JBoss.

Преимущества:

* Встроенный механизм управления учетными записями
* Поддержка большого количества платформ
* Поддержка каталогов LDAP
* Поддержка протоколов SAML 2 и OpenID Connect

Недостатки:

* Не поддерживает Kerberos
* Ограниченные возможности по расширению функционала

OpenidDict – это open-sour библиотека, позволяющая разворачивать сервер аутентификации. Поддерживает OAuth 2.0, Open ID Connect.

Преимущества:

* Легко расширяется и дополняется
* Поддержка механизма управления учётными записями

Недостатки:

* Поддерживает только OpenID Connect
* Поддерживает только платформу ASP.NET

3 Цели и задачи

Цель – агрегация различных протоколов аутентификации в инфраструктуре сервисов компании ООО «БиАйВи».

Задачи:

1. Реализовать хранилище учётных данных на основе LDAP
2. Реализовать поддержку протоколов Kerberos и OpenID connect

4 Описание функциональных и технологических требований

# Техническое задание на разработку ПО

## 3.1 Постановка задачи

Разработать сервер SSO аутентификации на основе протокола LDAP, используя Active Directory.

## 3.2 Функциональные требования

### 3.2.1 Клиентская часть

Консольное Windows приложение на платформе Java, способное производить запрос на аутентификацию.

### 3.2.2 Серверная часть

Серверная часть должна обрабатывать запросы, приходящие от клиентской стороны, и производить аутентификацию.

## 3.3 Технические требования

### 3.3.1 Системные требования к серверу

ПК с ОС Windows версии не ниже Windows Server 2012, Жесткий диск объемом не менее 2000Мб, ОЗУ не менее 4096Мб, с тактовой частотой не ниже 2 Ггц.

### 3.3.2 Системные требования к клиенту

Для работы Windows приложения необходима установленная JDK версии 6 и выше.

### 3.3.3 Требования отказоустойчивости

Фиксация ошибок.

## 3.4 План выполнения практики

План выполнения практики представлен в таблице 1.

|  |  |  |
| --- | --- | --- |
| Наименование работ | Период выполнения  (дата начала – дата окончания) | Форма отчетности |
| Ознакомление с организацией работ по разработке программного обеспечения на различных этапах жизненного цикла | 2.07.2018 - 7.07.2018 | Формализованное описание сформулированной в индивидуальном задании на практику проблемы. |
| Изучение и формализация задачи, сформулированной в индивидуальном задании. Выбор методов и алгоритмов | 9.07.2018 - 11.07.2018 | Модель предметной области, описание методов и алгоритмов |
| Разработка и тестирование программного обеспечения | 12.07.2018 – 18.07.2018 | Программный код и результаты тестирования разработанных модулей |
| Подготовка отчетных материалов | 18.07.2018 – 20.07.2018 | Отчет по практике, презентация |

Таблица 1. План выполнения практики.

# 4 Описание методики тестирования

Необходимо проверить, что выполнены все функциональные требования, а также требования к отказоустойчивости. Проверка происходить при помощи ввода необходимых данных. Описание тестов и результатов тестирования приведено в Таблица 2.

Таблица 2 - Тестирование программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| № | Проверяемая функциональность | Входные данные / действия | Ожидаемый результат |
| 1 | Аутентификация существующего пользователя | Логин, пароль | Разрешён вход |
| 2 | Аутентификация несуществующего пользователя | Логин, пароль | Вход не разращён |

В результате тестирования результат соответствует ожидаемому.

# Приложения

**Листинг 1. Подключение к LDAP серверу**

package com.ldap;

import java.util.Hashtable;

import javax.naming.Context;

import javax.naming.NamingEnumeration;

import javax.naming.NamingException;

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 ConnectLdap {

public String dom;

public String id;

public String pass;

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.print("dom");

dom = in.NextLine();

System.out.print("Id");

id = in.NextLine();

System.out.print("pass");

pass = in.NextLine();

Hashtable<String, String> env = new Hashtable<String, String>();

env.put(Context.INITIAL\_CONTEXT\_FACTORY,"com.sun.jndi.ldap.LdapCtxFactory");

env.put(Context.PROVIDER\_URL, dom);

env.put(Context.SECURITY\_PRINCIPAL, id);

env.put(Context.SECURITY\_CREDENTIALS, pass);

DirContext ctx = null;

NamingEnumeration results = null;

try {

ctx = new InitialDirContext(env);

SearchControls controls = new SearchControls();

controls.setSearchScope(SearchControls.ONELEVEL\_SCOPE);

// results = ctx.search("cn=users,DC=domain,DC=companyname,DC=com",

// "(objectclass=top)", controls);

results = ctx.search(

"ou=Associates,DC=domain,DC=companyname,DC=com",

"(objectclass=top)", controls);

int count = 0;

while (results.hasMore()) {

SearchResult searchResult = (SearchResult) results.next();

Attributes attributes = searchResult.getAttributes();

// System.out.println(attributes+"\n");

Attribute samaccountname = attributes.get("samaccountname");

if (samaccountname != null) {

String cn = (String) samaccountname.get();

System.out.println("Count = " + count++);

System.out.println("samaccountname = " + cn);

}

Attribute mail = attributes.get("mail");

if (mail != null) {

String cn = (String) mail.get();

System.out.println("mail id = " + cn + "\n");

} else {

System.out.println("mail id = \n");

}

}

} catch (NamingException e) {

throw new RunIvaneException(e);

} finally {

if (results != null) {

try {

results.close();

} catch (Exception e) {

}

}

if (ctx != null) {

try {

ctx.close();

} catch (Exception e) {

}

}

}

}

}

**Листинг 2. Пример хранения данных в LDAP каталоге**

# suffix ibm.com

dn: dc=ibm,dc=com

objectclass: top

objectclass: domain

dc: ibm

# employees directory

dn: cn=employees,dc=ibm,dc=com

objectclass: top

objectclass: container

cn: employees

# employee Ivan Qwe

dn: cn=Ivan Qwe,cn=employees,dc=ibm,dc=com

objectclass: top

objectclass: person

objectclass: organizationalPerson

objectclass: inetOrgPerson

objectclass: publisher

objectclass: ePerson

cn: Ivan Qwe

cn: "Qwe, Ivan"

sn: Qwe

givenname: Ivan

telephonenumber: 555-555-1234

mail: Qwe@ibm.com

# employee John Smith

dn: cn=John Smith,cn=employees,dc=ibm,dc=com

objectclass: top

objectclass: person

objectclass: organizationalPerson

objectclass: inetOrgPerson

objectclass: publisher

objectclass: ePerson

cn: John Smith

cn: "Smith, John"

sn: Smith

givenname: John

telephonenumber: 555-555-1235

mail: jsmith@ibm.com

# 5 Вывод

При прохождении практики я ознакомился с организацией работ по разработке программного обеспечения в компании «ООО «Би Ай Ви». Изучил и проанализировал методы, технологии и инструментарии, используемые при разработке реальных программных систем. При выполнении поставленной задачи были получены новые знания с языком Java и новыми средами разработки. Так же за время практики была изучена материал по теме SSO авторизации. Результатом работы стала разработка своего сервиса авторизации. За все время прохождения, место практики оставило положительные эмоции.

# 6 Список литературы

1. Гупта, Арун Java EE 7. Основы / Арун Гупта. - М.: Вильямс, 2014. - 336 c.

2. Шилдт, Герберт Java 8. Руководство для начинающих / Герберт Шилдт. - М.: Вильямс, 2015. - 720 c.

3. Эккель, Брюс Философия Java / Брюс Эккель. - М.: Питер, 2016. - 809 c.

4. Хабибуллин, Ильдар Самоучитель Java / Ильдар Хабибуллин. - М.: БХВ-Петербург, 2014. - 768 c.

5. Гонсалвес, Энтони Изучаем Java EE 7 / Энтони Гонсалвес. - М.: Питер, 2016. - 640 c.