**Оглавление**

[Введение 5](#_Toc9522064)

[1 Анализ предметной области 7](#_Toc9522065)

[1.1 Описание предметной области 7](#_Toc9522066)

[1.2 Формулировка проблемы и актуальность 19](#_Toc9522067)

[1.3 Обзор аналогов 19](#_Toc9522068)

[2 Программная документация 21](#_Toc9522069)

[2.1 Техническое задание на программное обеспечение 21](#_Toc9522070)

[2.1.1 Общие сведения 21](#_Toc9522071)

[2.1.1.1 Наименование программы 21](#_Toc9522072)

[2.1.1.2 Плановые сроки начала и окончания работ 21](#_Toc9522073)

[2.1.1.3 Порядок оформления и предъявления заказчику результатов работ 21](#_Toc9522074)

[2.1.1.4 Перечень нормативно-технических документов, методических материалов, использованных при разработке ТЗ 21](#_Toc9522075)

[2.1.1.5 Определения, обозначения, сокращения 22](#_Toc9522076)

[2.1.2 Назначение и цели создания системы 22](#_Toc9522077)

[2.1.2.1 Назначение системы 22](#_Toc9522078)

[2.1.2.2 Цели создания системы 22](#_Toc9522079)

[2.1.3 Характеристика объектов автоматизации 23](#_Toc9522080)

[2.1.3.1 Краткие сведения об объекте автоматизации 23](#_Toc9522081)

[2.1.3.2 Существующие программное обеспечение 23](#_Toc9522082)

[2.1.4 Требования к системе 23](#_Toc9522083)

[2.1.4.1 Требование к структуре и функционированию системы 23](#_Toc9522084)

[2.1.4.1.1 Перечень подсистем, их назначение и основные характеристики 23](#_Toc9522085)

[2.1.4.1.2 Требования к способам и средствам связи для информационного обмена между компонентами системы 23](#_Toc9522086)

[2.1.4.1.3 Требования к режимам функционирования системы 24](#_Toc9522087)

[2.1.4.1.4 Перспективы развития, модернизации системы 24](#_Toc9522088)

[2.1.4.2 Требования к надежности 24](#_Toc9522089)

[2.1.4.3 Требования к безопасности 24](#_Toc9522090)

[2.1.4.4 Требования к эргономике и технической эстетике 24](#_Toc9522091)

[2.1.4.5 Требования к интерфейсу 24](#_Toc9522092)

[2.1.4.6 Требования к защите информации от несанкционированного доступа 24](#_Toc9522093)

[2.1.4.7 Требования по сохранности информации при авариях 25](#_Toc9522094)

[2.1.4.8 Требования по стандартизации и унификации 25](#_Toc9522095)

[2.1.4.9 Дополнительные требования 25](#_Toc9522096)

[2.1.5 Требования к функциям (задачам), выполняемым системой 25](#_Toc9522097)

[2.1.6 Требования к видам обеспечения 25](#_Toc9522098)

[2.1.6.1 Требования к лингвистическому обеспечению системы 25](#_Toc9522099)

[2.1.6.2 Требования к программному обеспечению системы 25](#_Toc9522100)

[2.1.6.3 Требования к техническому обеспечению 26](#_Toc9522101)

[2.2 Пояснительная записка к программному обеспечению 26](#_Toc9522102)

[2.2.1 Назначение и область применения 26](#_Toc9522103)

[2.2.2 Технические решения 26](#_Toc9522104)

[2.2.2.1 Постановка задачи на разработку программы 26](#_Toc9522105)

[2.2.2.2 Описание структуры программной системы 26](#_Toc9522106)

[2.2.2.3 Описание технологии программирования 26](#_Toc9522107)

[2.2.2.4 Описание взаимодействия с другими программами 27](#_Toc9522108)

[2.2.2.5 Описание и обоснование выбора метода организации входных и выходных данных 27](#_Toc9522109)

[2.2.2.6 Описание алгоритмов и функционирования программы 27](#_Toc9522110)

[2.2.3 Ожидаемые технико-экономические показатели 27](#_Toc9522111)

[2.3 Описание программы 27](#_Toc9522112)

[2.3.1 Общие сведения 27](#_Toc9522113)

[2.3.1.1 Необходимое программное обеспечение 27](#_Toc9522114)

[2.3.1.2 Используемые языки программирования 27](#_Toc9522115)

[2.3.2 Функциональное назначение 28](#_Toc9522116)

[2.3.3 Описание логической структуры 28](#_Toc9522117)

[2.3.4 Входные и выходные данные 28](#_Toc9522118)

[2.4 Программа и методика испытаний 28](#_Toc9522119)

[2.4.1 Объект испытаний 28](#_Toc9522120)

[2.4.2 Цель испытаний 28](#_Toc9522121)

[2.4.3 Требования к программе 28](#_Toc9522122)

[2.4.4 Средства и порядок испытаний 28](#_Toc9522123)

[2.4.5 Методы испытаний 28](#_Toc9522124)

[3 Эксплуатационная документация на программный продукт 30](#_Toc9522125)

[3.1 Руководство пользователя 30](#_Toc9522126)

[3.1.1 Назначение программного продукта 30](#_Toc9522127)

[3.1.2 Необходимые технические характеристики устройства 30](#_Toc9522128)

[3.1.3 Установка 30](#_Toc9522129)

[3.1.4 Начало работы 30](#_Toc9522130)

[3.1.5 Структура приложения 30](#_Toc9522131)

[3.1.6 Описание настроек 30](#_Toc9522132)

[4 Акт испытаний программного продукта 32](#_Toc9522133)

[5 Экономическое обоснование 33](#_Toc9522134)

[5.1 Краткая характеристика программного продукта 33](#_Toc9522135)

[5.2 Расчёт себестоимости разработки программной системы 33](#_Toc9522136)

[5.2.1 Материальные затраты 33](#_Toc9522137)

[5.2.2 Заработная плата с отчислением на социальные нужды 33](#_Toc9522138)

[5.2.4 Накладнее расходы 33](#_Toc9522139)

[5.2.5 Полная себестоимость 33](#_Toc9522140)

[5.3 Оценка экономической эффективности разработки программной системы 33](#_Toc9522141)

[5.4 Вывод 33](#_Toc9522142)

[Заключение 34](#_Toc9522143)

[Список литературы 35](#_Toc9522144)

[Приложения 36](#_Toc9522145)

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.

Пользователю крайне неудобно иметь десяток учётных записей. Поэтому всё чаще создаются системы с отдельным сервером, который отделяет процессы аутентификации от сервиса. Такой сервер может предоставлять свои услуги большому количеству сервисов.

Но и тут возникает проблема, так как существует несколько подобных решений и пользователям всё равно придётся иметь разные учётные записи для каждого из них. Так как они не совместимы между собой.

Очевидно, что заставить всех пользователей сменить свои аккаунты невозможно. Поэтому нужно научить сервисы понимать разные серверы аутентификации.

1 Анализ предметной области

1.1 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 - 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). (про крипто стойкость уточнить)

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 задачей 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.

Однако, 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. (добавить про центры сертификации)

Таким образом современные протоколы аутентификации и авторизации должны включать в себя множество механизмов обеспечения безопасности.

Протоколы в общем можно классифицировать по методу аутентификации:

1. По паролю
2. По сертификатам
3. По одноразовым паролям
4. По токенам

Самым простым методом является простая авторизация по паролю. Она предполагает наличие у пользователя и у сервиса пары логин-пароль, с помощью которых и происходит подтверждение личности. Является наиболее не безопасным, так как перехват пароля ведёт к полной потере конфиденциальности. Поэтому все коммуникации должны происходить в защищённой среде. Например, используя HTTPS. Одним из протоколов реализующих такой метод аутентификации является HTTP authentication.

Клиент

Сервер

Запрос ресурса

Запрос на аутентификацию

Учётные данные

Отчёт о выполнении

Рисунок 1. Пример работы HTTP аутентификации со схемой Basic

1. Клиент запрашивает ресурс, используя GET запрос
2. Сервер отвечает о необходимости аутентификации кодом 401
3. Клиент снова запрашивает ресурс и передаёю свои данные в заголовке Authorization в открытом виде.
4. Сервер проводит проверку данных пользователя и в случае успеха возвращает код 200.

Аутентификация по сертификатам предполагает наличие доверенного сервера certificate authority (CA). Этот сервер выступает гарантом подлинности сертификатов, которые он подписал. Сертификаты содержат данные идентифицирующие конкретного пользователя. Если сервис доверяет CA который подписал сертификат, то он принимает его для аутентификации.

Клиент

Сервис

Подпись сертификата

Подписанный сертификат

Рисунок 2. Аутентификация по сертификатам

Аутентификация по одноразовым паролям обычно является дополнительным способом повышения безопасности. С помощью её реализуется концепция two-factor authentication (2FA). Согласно ей, пользователь должен использовать два типа учётных данных. Например, обычный пароль к учётной записи и код из смс-сообщения. Для реализации одноразовых паролей могут использоваться как программные средства, так и специальные устройства.

Аутентификация по токенам чаще всего используется для построения больших распределённых систем. В ней функционал по аутентификации делегируется сервисами отдельному приложению – серверу аутентификации или identity provider (IdP). Это позволяет упростить структуру сервисов. А также открывает возможность реализовать технологию единого входа Single Sign-On (SSO). Она позволяет однажды аутентифицирующемуся пользователю иметь доступ ко всем сервисам, находящимся в зоне ответственности IdP. По такому же принципу работает вход через учётные записи социальных сетей.

Клиент

IdP

Сервис

Учётные данные

Токен сервиса

Токен сервиса

Отчёт о выполнении

Рисунок 3. Общая схема аутентификации по токену

В общем виде процесс аутентификации по токену выглядит следующим образом:

1. Клиент проводит аутентификацию с IdP. Способом может быть любым, но обычно это протоколы Kerberos, OpenID Connect и аналогичные.
2. Теперь Клиент может запрашивать к IdP токен для конкретного сервиса.
3. Используя токен, Клиент аутентифицируется в сервисе.

В некоторых ситуациях, например, когда клиентом является браузер невозможно выполнять сложные последовательности запросов. Для таких случаев разработана схема с пассивным клиентом.

Клиент

IdP

Сервис

Запрос учётных данных

Учётные данные

Запрос ресурса

Запрошенный ресурс

Токен

1

2

Перенаправление запроса

3

4

5

6

7

Рисунок 4. Схема аутентификации с пассивным клиентом, с перенаправлением запросов

Формат токенов очень важен при проведении аутентификации, поэтому стандартизирован. Основные используемые форматы токенов:

1. JSON Web Token (JWT)
2. Simple Web Token (SWT)

JWT – это открытии стандарт, который описывает формат сообщений в виде JSON файлов. Согласно нему токен состоит из заголовка, хранящего собственное описание, основной части, в которой хранится пользовательская информация, а также электронной подписи, удостоверяющей весь файл. В стандарте определены специальные имена, определяющие конкретные значения: iss, exp и т.д.

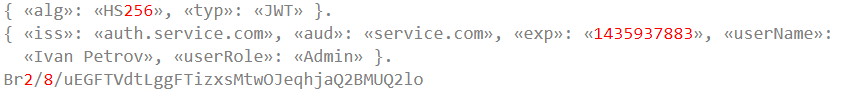


Рисунок 5. Пример не зашифрованного токена формата JWT

SWT – открытый стандарт, имеет простую структуру. Что позволяет просто использовать для передачи заголовки протоколов тапа HTTP. Имеет зарезервированные имена. Использует симметричное шифрование и цифровую подпись.



Рисунок 6. Пример не зашифрованного токена формата SWT

Из существующих протоколов аутентификации с токеном можно выделить наиболее популярные. 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.

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.

(Про распределение ключей)

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 токенов JWT или SWT. 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.

(дописать)

1.2 Формулировка проблемы и актуальность

Проблематика заключается в потребности пользователей сервисов ООО «БиАйВи» в агрегации различных протоколов аутентификации в одну систему. С предоставлением им единого интерфейса аутентификации.

Актуальность состоит в потребности ООО «БиАйВи» в повышении гибкости системы аутентификации. А также в предоставлении более удобного интерфейса взаимодействия с собственными сервисами.

1.3 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

2 Программная документация

2.1 Техническое задание на программное обеспечение

2.1.1 Общие сведения

2.1.1.1 Наименование программы

Полное наименование программы: Система централизованной авторизации на основе LDAP и современных протоколов аутентификации.

Краткое наименование: Системы централизованной авторизации.

2.1.1.2 Плановые сроки начала и окончания работ

Плановый срок начала работ по созданию системы централизованной авторизации – 01 ноября 2018 года. Плановый срок окончания работ по разработке системы централизованной авторизации – 15 июня 2019 года.

2.1.1.3 Порядок оформления и предъявления заказчику результатов работ

Система передается в виде функционирующего серверного приложения, установленного на оборудование заказчика в сроки, установленные договором. Совместно с системой производится сдача разработанного комплекта документации.

2.1.1.4 Перечень нормативно-технических документов, методических материалов, использованных при разработке ТЗ

При разработке автоматизированной системы и создании проектно-эксплуатационной документации Исполнитель должен руководствоваться требованиями следующих нормативных документов:

* ГОСТ 19.201-78. ТЕХНИЧЕСКОЕ ЗАДАНИЕ. ТРЕБОВАНИЯ К СОДЕРЖАНИЮ И ОФОРМЛЕНИЮ.
* ГОСТ 34.601-90. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.
* ГОСТ 34.201-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплексность и обозначение документов при создании автоматизированных систем.

2.1.1.5 Определения, обозначения, сокращения

Таблица 1 - Расшифровка определений, обозначений, сокращений

|  |  |  |
| --- | --- | --- |
| № | Наименование | Расшифровка |
| 1 | KDC | Key Distribution Center – центр выдачи ключей |
| 2 | ТЗ | Техническое задание |
| 3 | TGT | Ticket granting ticket – Билет на получение билета |

2.1.2 Назначение и цели создания системы

2.1.2.1 Назначение системы

Система централизованной авторизации предназначена для агрегации различных протоколов аутентификации и предоставления конечному пользователю единого интерфейса по доступу к сервисам ООО «БиАйВи» удобным ему методом.

2.1.2.2 Цели создания системы

Основной целью создания системы является упрощение процесса аутентификации для пользователей сервисов компании ООО «БиАйВи».

Для реализации поставленных целей система должна решать следующие задачи:

* Предоставление общего интерфейса для пользователей, использующих разные протоколы аутентификации
* Поддерживать работу с LDAP аутентификацией
* Поддерживать работу с протоколом OpenId Connect
* Поддерживать работу с протоколом SAML
* Реализация технологии Single Sign-On

2.1.3 Характеристика объектов автоматизации

2.1.3.1 Краткие сведения об объекте автоматизации

* Система реализует технологию Single Sign-On.
* Используемые протоколы аутентификации реализуют концепцию доверенной третьей стороны.
* Тикеты полученные от разных KDC, хранятся в одной базе данных.

2.1.3.2 Существующие программное обеспечение

На момент начала работы было…

2.1.4 Требования к системе

2.1.4.1 Требование к структуре и функционированию системы

2.1.4.1.1 Перечень подсистем, их назначение и основные характеристики

В состав системы централизованной авторизации должны входить следующие подсистемы:

* База данных для хранения полученных тикетов
* Подсистема анализа тикетов
* Подсистема диалога с пользователем
* Подсистема взаимодействия с сервисами

2.1.4.1.2 Требования к способам и средствам связи для информационного обмена между компонентами системы

Система централизованной авторизации должна взаимодействовать со своими компонентами с использованием канала Интернет, используя протокол HTTP. Настройки приложения должны храниться в локальном хранилище каждого из компонентов.

2.1.4.1.3 Требования к режимам функционирования системы

Для системы централизованной авторизации определены следующие режимы:

* 1
* 2

2.1.4.1.4 Перспективы развития, модернизации системы

В дальнейшем планируется увеличение числа поддерживаемых протоколов.

2.1.4.2 Требования к надежности

Система должна сохранять работоспособность и обеспечивать восстановление своих функций при возникновении следующих внештатных ситуаций:

* 1
* 2

2.1.4.3 Требования к безопасности

Безопасность

2.1.4.4 Требования к эргономике и технической эстетике

Следование стандартам реализации?

2.1.4.5 Требования к интерфейсу

Требования?

2.1.4.6 Требования к защите информации от несанкционированного доступа

Описание надёжности протоколов?

2.1.4.7 Требования по сохранности информации при авариях

Данные на сервере должны иметь резервные копии?

2.1.4.8 Требования по стандартизации и унификации

Требования?

2.1.4.9 Дополнительные требования

Пока нет?

2.1.5 Требования к функциям (задачам), выполняемым системой

Система централизованной авторизации должна обеспечивать:

1. Регистрацию нового пользователя на KDC.
2. Регистрацию нового сервиса на KDC.
3. Проведение аутентификации зарегистрированных пользователей.
4. Проведение авторизации аутентифицированных пользователей на зарегистрированных сервисах.
5. Управление политикой авторизации со стороны сервисов.
6. Фоновую поддержку авторизации.
7. …

2.1.6 Требования к видам обеспечения

2.1.6.1 Требования к лингвистическому обеспечению системы

В системе должны присутствовать русский или английский языковые пакеты.

2.1.6.2 Требования к программному обеспечению системы

Для работы системы необходим сервер аутентификации с установленным Windows Server 2008 и выше.

2.1.6.3 Требования к техническому обеспечению

Требования к техническим характеристикам сервера:

* 8 Гб ОЗУ
* 2 ядерный процессор
* Доступ к сети интернет

2.2 Пояснительная записка к программному обеспечению

2.2.1 Назначение и область применения

Система централизованной авторизации предназначена для агрегации поддержки различных протоколов аутентификации и предоставления конечным пользователям объединённого интерфейса для аутентификации в сервисах компании ООО «БиАйВи».

2.2.2 Технические решения

2.2.2.1 Постановка задачи на разработку программы

Разрабатываемое приложение должно решать следующие задачи:

* Предоставление общего интерфейса для пользователей, использующих разные протоколы аутентификации
* Поддержка работы с протоколом Kerberos
* Поддержка работы с протоколом OpenID Connect
* Поддержка работы с LDAP аутентификацией
* Реализация технологии Single Sign-On

2.2.2.2 Описание структуры программной системы

Программа состоит из?

2.2.2.3 Описание технологии программирования

Так как разработка проводилась с использованием объектно-ориентированного языка программирования, в разработке используется объектно-ориентированный метод программирования, представляющий программу в виде совокупности объектов, каждый из которого является экземпляром определенного класса.

2.2.2.4 Описание взаимодействия с другими программами

??

2.2.2.5 Описание и обоснование выбора метода организации входных и выходных данных

??

2.2.2.6 Описание алгоритмов и функционирования программы

Во время работы система аутентификации взаимодействует со следующими объектами:

1. Сервисы – ресурсы к которым обращаются клиенты.
2. Клиенты – устройство или программы используемая пользователем для доступа к сервисам.
3. Пользователь – реальный человек, имеющий учётную запись.
4. IdP -

Система аутентификации

БД

Клиент

IdP

1

2

3

4

5

6

7

Пользователь

888

2.2.3 Ожидаемые технико-экономические показатели

Результатом работы должно стать стабильное рабочее серверное приложение, позволяющее проводить аутентификацию пользователей, использующих различные протоколы аутентификации. Приложение должно поддерживать горизонтальную масштабируемость.

Расчёт экономической эффективности проекта приведён в разделе 5 отчёта.

2.3 Описание программы

2.3.1 Общие сведения

2.3.1.1 Необходимое программное обеспечение

Для корректной работы приложения на компьютере должна быть установлена операционная система Linux.

2.3.1.2 Используемые языки программирования

При разработке системы аутентификации использовался

объектно-ориентированный язык программирования Java SE 8.

2.3.2 Функциональное назначение

Централизованная система аутентификации предназначена для предоставления пользователям возможности получения доступа к сервисам ООО «БиАйВи» с использованием удобных им протоколов аутентификации.

2.3.3 Описание логической структуры

Схема взаимодействия и диограмма использования

2.3.4 Входные и выходные данные

???

2.4 Программа и методика испытаний

2.4.1 Объект испытаний

В рамках данного проекта объектами испытаний являются следующие модули: ???

2.4.2 Цель испытаний

Целью испытаний является проверка соответствия программного продукта функциональным требованиям и определение корректности работы системы с различными входными данными.

2.4.3 Требования к программе

??

2.4.4 Средства и порядок испытаний

Для проведения испытаний необходима тестовая учётная запись, ??

2.4.5 Методы испытаний

Таблица с методиками тестов

3 Эксплуатационная документация на программный продукт

3.1 Руководство пользователя

3.1.1 Назначение программного продукта

3.1.2 Необходимые технические характеристики устройства

3.1.3 Установка

3.1.4 Начало работы

3.1.5 Структура приложения

3.1.6 Описание настроек

4 Акт испытаний программного продукта

5 Экономическое обоснование

5.1 Краткая характеристика программного продукта

5.2 Расчёт себестоимости разработки программной системы

5.2.1 Материальные затраты

5.2.2 Заработная плата с отчислением на социальные нужды

5.2.4 Накладнее расходы

5.2.5 Полная себестоимость

5.3 Оценка экономической эффективности разработки программной системы

5.4 Вывод

Заключение

Список литературы

1. Хорев П.Б. Методы и средства защиты информации в компьютерных системах: Учеб. пособие для студ. высш. учеб. заведений -- М.: Издательский центр «Академия», 2005. -- 256 с.
2. Адаменко, Михаил Основы классической криптологии. Секреты шифров и кодов / Михаил Адаменко. - Москва: Машиностроение, 2014. - 256 c.
3. Бабаш, А. В. История криптографии. Часть I / А.В. Бабаш, Г.П. Шанкин. - М.: Гелиос АРВ, 2002. - 240 c.
4. Гупта, Арун Java EE 7. Основы / Арун Гупта. - М.: Вильямс, 2014. - 336 c
5. RFC 4120 – Стандарт службы сетевой аутентификации Kerberos (версия 5)
6. RFC 6113 - Обобщенный механизм предварительной аутентификации для Kerberos.
7. Инфраструктуры открытых ключей / Полянская О.Ю., Горбатов В.С. - M.: Интернет-университет информационных технологий - ИНТУИТ.ру, 2007
8. Обзор протокола Kerberos – режим доступа:  
   <https://pro-ldap.ru/tr/zytrax/tech/kerberos.html>

Приложения

LDAP – база данных для учётных данных (логины, права, и структура) (иерархическая (дерево)) + стандартные способы доступа.

Kerberos, OIDC… - протоколы (набор соглашений) по передаче данных между участниками(определёнными) сети.

НУЖНО – Сервер(KDC) реализующий(исполняющий) ПРОТОКОЛ, и хранящий данный в LDAP. (KDC выдаёт билеты и всё) (ИНФОРМАЦИЯ О РАСПОЛОЖЕНИИ KDC, алгоритмах шифрования в файле КОНФИГУРАЦИИ)

НУЖНО – Клиентская сторона способная создать нового USER’а и зарегистрироваться на KDC (ИНФОРМАЦИЯ О РАСПОЛОЖЕНИИ KDC, алгоритмах шифрования в файле КОНФИГУРАЦИИ)

НУЖНО – Пристройка(?!) к сервисам, которая может регистрироваться в KDC и принимать сформированные ПРОТОКОЛОМ билеты (ещё есть refresh ticket).

Какой сервер?? Как общаться с сервером?

(OIDC - общается json, Kerberos – не определенно (а как тогда? (GSSAPI ??)))

Критерии сравнения систем авторизации

1. Безопасность: количество участков системы, на которых возможен доступ третьих лиц, который может привести к потере безопасности (нужно описать все участки системы)
2. Удобство для пользователя (наличие SSO?) (кол-во действий для авторизации и необходимость повторной авторизации)
3. Удобство для ресурса (кол-во манипуляций, масштабируемость)
4. Стоимость?
5. Оборудование?
6. Поддержка работы? (выдача учёток)
7. Перспективность технологий (год? Популярность?)

Цель

Цель – оптимизация процессов авторизации и аутентификации в инфраструктуре сервисов компании ООО «БиАйВи»

Описание

Предметной областью является аутентификация/авторизация.

Это неотъемлемая часть всех современных программных продуктов. Необходима и для реализации конфиденциальности, и для безопасности, и для удобства пользователей.

Для успешного прохождения авторизации необходимы:

1. Идентификация – заявление о том, кем является пользователь
2. Аутентификация – проверка подлинности пользователя
3. Авторизация – выдача соответствующих прав доступа к запрашиваемому ресурсу

Одна из основных функций авторизации безопасность. Так как все процессы авторизации происходят удалённо, в рамках компьютерной сети, и защита таких сетей от несанкционированного доступа на сто процентов невозможна. Следовательно, необходимо обеспечить конфиденциальность данных с учётом этих уязвимостей. Первым этапом для решения этой задачи является выделение всех таких небезопасных участков системы.

* Программа, производящая авторизацию со стороны пользователя
* Каналы передачи данных
* Устройство (сервер) и программа производящие авторизацию
* ???

На каждом таком участке системы злоумышленник может производить:

1. Пассивное наблюдение - злоумышленник, перехватывая данные использует их для получения конфиденциальной информации.
2. Воздействие на обменную информацию - злоумышленник, перехватывая и манипулируя информацией, способен получить доступ к конфиденциальным данным/системам.
3. Пароли или иные удостоверяющие данные никогда не пересылаются по сети. Подразумевается, что сетевой трафик может быть прослушан, может произойти подмена сообщения или любая другая пакость.
4. Выдвигается обязательное требование, что информация о паролях/удостоверяющих данных хранится в единственном защищённом месте (Центре распределения ключей Kerberos). Поэтому удостоверяющие данные никогда не сохраняются на том хосте, который пользователь использует для входа/логина. После того, как произошёл первоначальный обмен в рамках аутентификации, этот хост должен забыть сведения о пароле.
5. Хосты/серверы приложений должны быть в состоянии подтвердить свою идентификационную сущность любому, кто запрашивает подобные доказательства.
6. Все коммуникации между аутентифицированными пользователями (клиентами) и сервисами приложений должны иметь возможность быть зашифрованными. С этой целью поддерживаются и могут применяться различные алгоритмы шифрования (все [симметричные](https://pro-ldap.ru/tr/zytrax/tech/encryption.html#symmetric)).

Для

Терминология Kerberos

Как и у всех серьёзных систем, у Kerberos есть своя уникальная терминология, а кое-какие общие термины в контексте Kerberos обретают новый смысл. Постараемся раскрыть их простыми словами (во возможности):

|  |  |
| --- | --- |
| Область действия (Realm) | Realm просто означает совокупность пользователей и серверов приложений, которые охватывает (или имеет о них информацию) Центр распределения ключей (KDC). Так, для того чтобы пользователь подсоединился (или вошёл) в Realm, у Сервера аутентификации (Authentication Server) этой области Realm должны быть сведения об удостоверяющих данных этого пользователя (и другая информация о нём), хранящиеся в защищённой базе данных безопасности того или иного вида (форма хранения не определяется в RFC). В терминологии Microsoft это будет называться Доменом (Domain). Области Realm могут доверять другим Realm, в этом случае доверяющие друг другу области должны быть взаимно аутентифицированными (Cross-Authenticated).  Форма имени Realm — ...@REALM (регистр символов имеет значение). Например, если Realm называется JOE, то его Realm-имя будет @JOE (что отличается от Realm-имени @joe), а если Realm называется EXAMPLE.COM, то его Realm-имя будет @EXAMPLE.COM. (Примечание: Согласно текущей рекомендации ([раздел 6.1 RFC 4120](http://tools.ietf.org/html/rfc4120#section-6.1)) в качестве имени REALM следует использовать доменное имя, которое часто преобразуется в верхний регистр.) Несмотря на то, что последняя форма может напоминать адрес электронной почты, никакого отношения к электронной почте она не имеет. Если буквы большие, это наверняка REALM, а не почта. |
| Принципал (Principal) | Принципал — это строка, полностью идентифицирующая пользователя службы Kerberos. Он имеет форму thing@REALM. Принципал может быть именем сервиса, выполняющегося на хосте (мы будем называть его принципалом сервиса (Service-Principal)), или именем пользователя (мы будем называть его принципалом пользователя (User-Principal)). Принципалы формируют индексное поле для информации об объекте, хранящейся в базе данных безопасности Kerberos (в Центре распределения ключей или KDC). Форматы принципалов для пользователей и сервисов различаются.  Имя принципала пользователя — это приблизительный эквивалент имени пользователя или имени учётной записи. Оно имеет форму principal-name[/instance-name]@REALM (где часть */instance-name* является опциональной). Например, если имя пользователя в принципале пользователя — alice, а Realm — joe, то полный принципал будет alice@joe. Расширение instance-name позволяет любому пользователю иметь более одного принципала. Так, если alice является администратором области Realm joe, имя её принципала будет alice/admin@joe, и у этого принципала будут другие права (и удостоверяющие данные).  Если речь идёт о принципале сервиса, то форма имени принципала становится service-name/QDN@REALM, где QDN — это доменное имя хоста (без точки в конце, как того требует FQDN), на котором работает сервис, а service-name — это специфичная для приложения строка, идентифицирующая сервис на этом хосте. Некоторые типы сервисов используют ключевое слово host. Так, для сервиса ftp, работающего на хосте с именем fileserver.example.com в области Realm @EXAMPLE.COM, имя принципала сервиса будет ftp/fileserver.example.com@EXAMPLE.COM. |
| Разрешение (Ticket) | Разрешение — это структура данных, содержимое которой известно только издателю этого разрешения, и какой-либо стороне или сторонам, к которым это разрешение имеет отношение. Промежуточные хосты, такие как клиентский хост, рассматривают эти разрешения как неразбираемый набор бит и просто передают их на конечный пункт назначения. В Kerberos разрешения могут быть либо Разрешениями на получение разрешения (Ticket Granting Tickets, TGT), — по сути представляют собой доказательства успешно пройденной аутентификации, — либо Сервисными разрешениями (Service Tickets, ST), — выдаются Службой выдачи разрешений (Ticket Granting Service, TGS) и позволяют пользователю получить доступ к требуемому Сервису приложений (Application Service). |

Полезные штуки

1. <https://pro-ldap.ru/tr/zytrax/tech/kerberos.html> - немного теории по Kerberos есть схема (С ссылками. Годно)
2. <https://www.intuit.ru/studies/courses/3580/822/lecture/30592?page=2> – Теория по криптографическим системам.
3. <https://it.wikireading.ru/59914> - про распределение ключей в Kerberos (+ там много всего)
4. <https://web.mit.edu/Kerberos/krb5-1.13/doc/admin/pkinit.html> - PKINIT (англ.)
5. <https://pro-ldap.ru/tr/zytrax/tech/ssl.html> - о x.509 и сертификатах.