Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Работа допущена к защите
зав. кафедрой

_____ В.М. Ицыксон

«____» _____ 2018 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА

Система формирования целевой рекламы на основе данных общественной Wi-Fi сети

по направлению 09.04.01 «Информатика и вычислительная техника»
по образовательной программе
09.04.01_15 «Технологии проектирования системного и прикладного программного обеспечения»

Выполнил студент гр. 23541/3 _____ С.В. Васильев

Научный руководитель,
к. т. н., доц. _____ Н.В. Богач

Консультант по нормоконтролю,
к. т. н., доц. _____ А.Г. Новопашенный

Санкт-Петербург
2018

РЕФЕРАТ

На 86 с., 37 рисунков, 1 приложение.

БЕЗОПАСНОСТЬ ДАННЫХ, ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ, ОБЛАЧНЫЕ ВЫЧИСЛЕНИЯ, ОБЩЕСТВЕННАЯ WI-FI СЕТЬ, ЦЕЛЕВАЯ РЕКЛАМА

Данная работа посвящена разработке системы формирования рекламных сообщений. В качестве исходных данных система использует данные, указанные клиентами при регистрации в открытой Wi-Fi сети. Для выполнения задачи формирования рекламных сообщений используется информация о местах, посещенных зарегистрированными клиентами. Эту информацию позволяют собирать специально созданные Wi-Fi сканеры. В работе рассмотрена структура интеллектуальных систем, как способ организации генератора рекламных сообщений. Рассмотрено и обосновано применение облачных технологий в качестве платформы для анализа данных. Разработана и реализована структура системы и каждого ее компонента. Проведено тестирование полученной системы.

THE ABSTRACT

86 pages, 37 pictures, 1 appendix.

CLOUD COMPUTING, DATA SECURITY, INTELLIGENT SYSTEMS, PUBLIC WI-FI NETWORK, TARGETED ADVERTISING

This thesis deals with the development of an advertising message forming system. The system uses data provided by clients during registration in an open Wi-Fi network as the initial data. To form an advertising message, the system uses the information about the places visited by the registered clients. This information is collected by specially created Wi-Fi scanners. The thesis considers the structure of intelligent systems as a way of organizing an advertising message generator. The use of cloud technologies as a platform for data analysis is considered and justified. The structure of the system and each of its components have been developed and implemented. The obtained system has been tested.

СОДЕРЖАНИЕ

СПИСОК ОБОЗНАЧЕНИЙ И СОКРАЩЕНИЙ

CRM	Customer Relationship Management, Система управления взаимоотношениями с клиентами
SaaS	Software as a service, программное обеспечение как сервис
БД	база данных
ИС	интеллектуальная система
ОС	операционная система
ПО	программное обеспечение
ЭС	экспертная система
ЯП	язык программирования

ВВЕДЕНИЕ

В современном мире реклама встречается практически на каждом шагу. Ее показывают по телевизору, распространяют листовками, почтовыми сообщениями, развешивают по городу в виде банеров. Существует реклама и в сети интернет. Однако, в отличии от обычной рекламы, у интернет рекламы есть одно большое преимущество: она может быть целенаправленной. Это значит, что рекламные сообщения, которые увидит пользователь, не будут выбраны случайным образом из общего числа заявок, а будут использовать специальный алгоритм, позволяющий конкретизировать аудиторию. Целевая или таргетированная реклама становится возможной благодаря тому, что интернет в буквальном смысле запоминает все проделанные человеком действия. Будь то посещение тематических сайтов, совершение покупок в интернет магазине или просмотр видео — все действия остаются в видет информации на серверах компаний и при необходимости могут быть использованы для выявления предпочтений человека. Таргетированной рекламой целенаправленно занимаются такие компании как Google, Яндекс и Microsoft. Спрос на такой вид услуг никогда не иссякнет, т.к. любой компании выгодно привлечь именно ту аудиторию, которую заинтересует их услуга или товар.

Дальнейшее развити целевой рекламы — это вынос ее возможностей за пределы интернета. Возможность привлечь посетителя, который просто проходит каждый день мимо вашего заведения — это уже новый уровень таргетинга. Сделать это можно на основе данных, которые вещаются в эфир мобильным устройством потенциального клиента. К таким данным, например, относится информация от включенного Wi-Fi модуля, который есть в каждом современном смартфоне.

Целью данной работы является проектирование и разработка системы, которая позволила бы пользователям составлять правила таргетирования рекламы, на основе данных о посещении клиентами определенных географических мест. Для создания клиентской базы используется общественная Wi-Fi сеть с открытым доступом и требующая регистрации клиентов. А для формирования информации о географических точках, посещенных клиентом — специальные комплексы, включающие в себя Wi-Fi модуль и позволяющие выявлять проносимые мимо устройства.

Работа состоит из шести разделов. В первом разделе рассмат-

риваются вопросы связанные с таргетированной рекламой в целом. Рассматриваются уже существующие методы таргетирования и виды используемой информации. Решается вопрос о законности использования персональных данных.

Во втором разделе осуществляется обзор видов интеллектуальных систем и их типовая структура. Определяется конфигурация приложения, которая может быть использована в целях таргетирования рекламы.

Третий раздел посвящен облачным вычислениям и их видам. Рассматривается вопрос о размещении компонента генерации рекламных сообщений в облачной инфраструктуре.

В четвертом разделе описывается общая архитектура системы. Рассматриваются технология Captive Portal для реализации компонента регистрации пользователей. В разделе описываются возможности стандарта 802.11, позволяющие выявить новые устройства в зоне действия Wi-Fi модуля.

Пятый раздел посвящен проектированию и разработке каждого компонента. В разделе приведены архитектуры всех компонентов, описываются используемые при разработке инструменты и библиотеки.

Раздел номер шесть посвящен тестированию разработанного прототипа. Осуществляется как тестирование каждого компонента по отдельности, так и целостное тестирование системы.

1. РОЛЬ И МЕСТО ТЕХНОЛОГИЙ В ЦЕЛЕВОЙ РЕКЛАМЕ. ПОСТАНОВКА ЗАДАЧИ

Таргетированная реклама - это рекламные текстовые сообщения, дополненные изображениями, анимацией или видео роликами, демонстрирующиеся только лицам, которые удовлетворяют критериям рекламодателя.

1.1. Развитие целевой рекламы

Первое онлайн объявление создано в 1994 году, когда интернетом пользовалось всего 30 миллионов человек. В веб версии магазина HotWired 27 октября 1994 года было впервые запущено баннерное объявление для компании AT&T [18].

Впервые интернет реклама, имеющая целевой характер, появилась в поисковой системе goto.com в 1998 году. В настоящее время компания известна под названием Overture и принадлежит Yahoo!. Идея заключалась в том, чтобы продавать ссылки, которые видит пользователь в результатах поиска. Сами ссылки подбираются исходя из контекста поисковых запросов, введенных пользователем. Идея принадлежит американскому финансисту Биллу Гроссу, который является основателем компании Idealab [27].

Такой подход в продвижении товаров и брендов был выгоден не только рекламодателям, но и создателям поисковых систем, которые могли получать от этого прибыль. Данный вид рекламы в дальнейшем был назван контекстной рекламой.

В дальнейшем идею контекстной рекламы стали реализовывать и другие компании. Например компания Google создала сервис Google AdWords, в который позже была добавлена возможность проведения торгов за рекламное место.

В настоящее время за рубежом лидерами в области контекстной рекламы являются такие компании как: Google AdWords, Yahoo! Search Marketing и Microsoft Advertising. В нашей стране широкую популярность имеет сервис Яндекс Директ.

Сильный толчок развитию таргетированной рекламы придало по-

явление социальных сетей. Именно в них начали использовать такую информацию о клиенте как возраст, пол, семейное положение, геолокацию, круг общения.

В настоящее время информация о потенциальных клиентах собирается не только на сайтах с персональными данными, но и самими браузерами или техническими устройствами, такими как мобильный телефон или умный телевизор. Например, браузеру известна информация о посещаемых пользователем сайтах и времени, в течении которого он был запущен. Собранная информация в дальнейшем используется сторонними компаниями, которым выгодно найти клиента с определенной моделью поведения.

Для того чтобы понять как работает и как формируется целевая реклама, необходимо сначала разобраться в интернет рекламе в целом. Для этого рассмотрим виды уже существующих решений.

1.2. Технологии в интернет рекламе

Рекламу в интернете разделяют на несколько видов. Критерием для разделения служит используемая технология, позволяющая тем или иным способом довести контент до клиента.

Контекстная реклама

Этот вид рекламы использует контекст просматриваемой страницы. Рекламные сообщения или баннеры подбираются исходя из ключевых слов, которым соответствует содержание посещаемого интернет ресурса. Этот вид рекламы можно увидеть на большинстве современных сайтов. Например, при посещении сайта по организации туристических походов, можно увидеть рекламу магазина, продающего туристическое снаряжение.

Маркетинг в поисковых системах

Частным случаем контекстной рекламы являются рекламные ссылки в поисковых системах. Это маркетинговая практика, при которой рекламные объявления встраиваются в результаты поисковых запросов. Рекламодатель заранее указывает ключевые слова, которые будут использованы для выявления целевой аудитории. При выдаче клиенту результатов поиска в них добавляются рекламные сообщения, которые по ключевым словам соответствуют поисковому запросу.

При таком методе распространения рекламодатель оплачивает

только ту рекламу, которой воспользовался клиент. Такой подход также называется “оплата по клику”. Рекламные сообщения могут быть как небольшими текстовыми фразами, так и более информативными блоками с описанием товара, его цены и ссылкой на обзор.

Главной особенностью поискового маркетинга является то, что он предоставляет рекламодателям возможность размещать свои объявления перед мотивированными клиентами, которые получают рекламу в тот момент, когда сами готовы совершать покупки. Никакой другой тип распространения рекламы не может обеспечить подобного. Поэтому маркетинг в поисковых системах является одним из самых эффективных видов целевой рекламы.

Таргетинг в социальных сетях

Это еще один канал интернет маркетинга. В социальных сетях пользователи сами оставляют о себе полезную для рекламодателя информацию, такую как возраст, пол, сфера деятельности, интересы. Такая информация позволяет достаточно точно подбирать целевые группы и определять где эти группы сосредоточены. Целевая аудитория, выделенная по множеству критериев, более лояльно отнесется к товару из близкой ей тематики.

Нативная реклама

Нативная реклама - это реклама завуалированная под развлекательную или обзорную статью. Такая статья имеет привычный читателю вид и своим заголовком обещает рассмотреть какой-либо вопрос. Однако на самом деле ее целью является реклама определенного товара. Как правило, такие статьи имеют шаблонные заголовки: "Как выбрать хороший X "Топ 5 X"или "Обзор популярных X". Данный тип рекламы является относительно дорогим, но при этом и достаточно эффективным.

Все выше изложенные способы доведения рекламы до клиента используют определенную информацию для выделения групп клиентов, заинтересованных в том или ином товаре. Таким образом следует рассмотреть какая информация может быть полезной для формирования клиентской базы. В дальнейшем клиентская база позволит выделять целевые объекты по их интересам.

1.3. Виды целевой рекламы по типу используемой информации

Рассмотрим виды целевой рекламы, которые разделены по типу информации, используемой для формирования целевой аудитории.

Таргетинг по интересам

Таргетинг по интересам подразумевает показ рекламы, соответствующей интересам целевого объекта. Интересы клиента могут быть установлены, например, по тематике посещаемой в данные момент веб-страницы.

Временной таргетинг

Временной таргетинг позволяет рекламодателям уточнять время, в которое будут задействованы их рекламные сообщения. Чаще всего время привязано к локальному времени конечного клиента. Это позволяет использовать рекламу более эффективно, т.к. сокращает число показов в часы, когда потенциальные клиенты не пользуются интернет ресурсами [3].

Демографический таргетинг

Данный вид маркетинга использует в работе демографическую информацию, такую как пол, возраст, доход, семейное положение и другое. Эти данные позволяют значительно сократить целевую аудиторию и рекламировать товар, только тем людям, которые потенциально могут быть в нем заинтересованы. В большинстве случаев такая информация может быть собрана при заполнении регистрационной формы на сайте рекламодателя.

Географический таргетинг

Географический таргетинг позволяет формировать группы клиентов исходя из их местоположения. Критериями такой рекламы может быть страна, регион, город или почтовый индекс. Например, если компания ведет свою деятельность только на территории одного города, то для нее не имеет смысл рекламировать свои услуги в других городах. Информация для географического таргетинга может быть получена как при регистрации клиента на сайте рекламодателя, так и из других источников [11].

Поведенческий таргетинг

Данный вид формирования рекламы основан на интересах посетителя.Эти интересы устанавливаются исходя из поведения пользователя в браузере, ранее просмотренного контента, выполненного поиска, посещенных сайтов. Например, если установлено, что поль-

зователь ранее просматривал туристические туры, то в рекламных баннерах могут быть предложения на покупку авиа или ж/д билетов.

Геоповеденческий таргетинг

Исходными данными для геоповеденческого таргетинга является информация о перемещениях клиента и его остановках. Используя эту информацию можно точно определить привычки и пристрастия объекта. Например, если клиент часто посещает магазины со спорттоварами, то и реклама компаний, занимающихся распространением спортивной амуниции его заинтересует.

1.4. Постановка задачи и выбор способов решения

Постановка задачи.

Рассмотрение видов целевой рекламы и используемой в ней информацию позволяет определиться с целью и задачами, решаемыми в данной работе.

Целью работы является проектирование и разработка системы, которая позволяет пользователю формировать правила для генерации таргетированной рекламы, использующей данные демографического и геоповеденческого таргетинга.

Далее под пользователем подразумевается лицо, которое заинтересовано в привлечении клиентов и использует для этого разрабатываемую систему. Клиентом считается лицо, зарегистрировавшееся в системе, и являющееся потребителем услуг пользователя.

Решаемые задачи:

- спроектировать и реализовать модуль, генерации рекламных событий на основании информации о клиентах;
- реализовать часть системы, позволяющую пользователям взаимодействовать с системой и, в первую очередь, формировать правила для генерации целевой рекламы;
- спроектировать и реализовать модуль, который позволит пополнять базу клиентов и демографическую информацию о них;
- спроектировать и реализовать модуль, позволяющий собирать геоповеденческие данные о клиентах.

Выбор способов решения.

Для модуля генерации рекламных сообщений характерна постоянная работа по обработке клиентских данных. Модуль использует правил, сформированные пользователем и информацию, известную о клиентах, для формирования рекламных сообщений. Такой вид работы похож на работу интеллектуальных систем. Поэтому первым этапом диссертации станет рассмотрение применимости интеллектуальных систем в условиях решаемой задачи.

Другой частью системы является модуль формирования правил. Для реализации его функциональности следуют разработать интерфейс пользователя, который позволит справляться с поставленной задачей. Этот интерфейс более правильно сделать доступным из сети Интернет, как и остальной интерфейс пользователя. При этом появляется возможность сделать модуль генерации реклманых сообщений централизованным и лишь, предоставить к нему доступ из вне. Для реализации такой идеи следует рассмотреть вариант использования облачных технологий, как платформы для генератора целевой рекламы.

Для пополнения клиентской базы следует использовать общественную Wi-Fi сеть. Это позволит не только собирать демографическую информацию, но и установить идентификатор устройства пользователя. Получанный идентификатор будет использован в модуле, собирающем геоповеденческую информацию, и позволит «узнавать» клиента. Назовем этот модуль Wi-Fi сканером.

Прежде чем приступить к выполнению поставленных задач, следует разобраться с еще одним вопросом. Перечисленные ранее виды таргетинга подразумевают сбор и использование определенного вида информации о клиенте. Использование какой-то части такой информации может быть запрещено законодательством РФ. Это значит, что прежде чем создавать собственную систему формирования целевой рекламы, необходимо разобраться с вопросом о законности хранения и использования видов данных, которые будут использованы в создаваемой системе.

1.5. Юридический аспект. Проблема хранения и использования персональных данных

Следует отметить, что юридический аспект будет рассмотрен непосредственно в привязке к разрабатываемой системе. Это снизит

количество лишних вопросов и позволит более подробно рассмотреть важные моменты.

Одним из компонентов разрабатываемой системы является точка доступа с публичной Wi-Fi сетью. В связи с Федеральным законом(ФЗ) №126 «О связи», каждому поставщику услуг необходимо идентифицировать клиента, которому предоставляется доступ в интернет. Штраф за несоблюдение закона описан в Кодексе РФ об административных правонарушениях (КоАП), последние поправки в который, были внесены в апреле 2018 года. Сумма штрафа может составить от 30 до 75 тысяч рублей [9].

Для того чтобы выполнить требования законодательство, достаточно запросить у клиента номер мобильного телефона. В Российской Федерации сотовые номера выдаются при предъявлении оригинала паспорта, а информация о соответствии номера и телефона хранится у оператора связи [5]. Таким образом, в случае необходимости, правовые органы, смогут идентифицировать личность человека, совершавшего противозаконную деятельность из публичной Wi-Fi сети.

Для разрабатываемой системы является важным вопрос о возможности хранения и использования дополнительных данных о клиенте, таких как пол и возраст. Для того чтобы с полученной персональной информацией можно было осуществлять манипуляции, достаточно составить пользовательское соглашения, описывающее возможные варианты использования клиентских данных. Согласно ч.1 ст.9 ФЗ «О персональных данных» пользовательское соглашение может быть подтверждено клиентом в разных формах:

«Согласие на обработку персональных данных может быть дано субъектом персональных данных или его представителем в любой позволяющей подтвердить факт его получения форме, если иное не установлено федеральным законом»

Этот пункт позволяет клиенту согласиться с пользовательским соглашением в процессе регистрации в выданной ему веб-форме. Необходимо лишь уведомить пользователя о том, что регистрируясь, он соглашается с предоставленным ему документом. В листинге 7 представлен текст составленного пользовательского соглашения.

1.6. Резюме

В данной главе рассмотрены основные методы реализации рекламы в интернете. Рассмотрены виды информации, которые позволяют осуществлять выбор целевой аудитории в задачах таргетированной рекламы. Поставлены цель и задачи выполняемой работы. Проанализирован юридический аспект в вопросе хранения и использования клиентской информации. Дальнейшим этапом станет рассмотрение интеллектуальной системы, как системы для формирования целевой рекламы на основе клиентской базы данных.

2. ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ В ЗАДАЧАХ ФОРМИРОВАНИЯ ЦЕЛЕВОЙ РЕКЛАМЫ

Одной из задач работы является создание программного продукта, способного на основе клиентской базы данных генерировать события с рекламными сообщениями. Для решения такой задачи может быть использована интеллектуальная система. Это следует из определения интеллектуальных систем и решаемых ими задач.

Интеллектуальная система (ИС) – это программный комплекс, который предназначен для решения задач различных типов в зависимости от потребностей пользователя. Интеллектуальная система в своей работе использует базу знаний и логико-математические средства для получения решения.

Интеллектуальная система способна решать различные задачи. Среди них: интерпретация данных, диагностика, мониторинг, проектирование, прогнозирование, планирование, обучение, управление, поддержка принятия решений и др [10].

В условиях данной работы, интеллектуальная система должна использовать базу знаний и логико-математические средства для решения задачи управления рассылкой рекламных сообщений.

Существуют разные типы интеллектуальных систем, каждая из которых более применима в той или иной ситуации. Для того чтобы понять, какая системы подходит для использования в таргетировании рекламы, необходимо рассмотреть различные типы интеллектуальных систем. Так же следует изучить вопрос устройства интеллектуальных систем. Это позволит разработать свой вариант ИС, применимый для решения поставленной задачи.

2.1. Типы интеллектуальных систем и их характеристики

Для интеллектуальных систем характерны такие свойства как развитая коммуникативная способность, умение решать плохо формализуемые задачи, способность к самообучению и адаптивности.

Коммуникативность ИС проявляется в способности взаимодействовать с конечным пользователем. Это позволяет ему формировать

свои запросы для получения решения.

Решение плохо формализуемых задач определяется способностью ИС к построению оригинальных алгоритмов с возможностью использовать информацию о постоянно меняющихся данных и окружающей среде.

Самообучение заключается в возможности использовать накопленный опыт для решения новых задач.

Адаптивность позволяет системе подстраиваться под изменения модели проблемной области.

Исходя из перечисленных признаков можно разделить ИС на следующие типы:

- системы с коммуникативными способностями (с интеллектуальным интерфейсом);
- экспертные системы (системы для решения сложных задач);
- самообучающиеся системы (системы, способные к самообучению);
- системы с адаптивными способностями.

Каждый из типов в свою очередь подразделяется на подтипы в зависимости от используемой технологии [10]. Полная классификация представлена на рис.2.1.

Системы с интеллектуальным интерфейсом направлены на улучшение способов взаимодействие с конечным пользователем.

Интеллектуальные базы данных. Применяются для получения выборки данных, которая не хранилась БД в явном виде. В отличие от обычных БД выборка может быть сформирована на основе хранимой информации.

Естественно-языковой интерфейс. Позволяет получать данные с помощью голосового ввода команд или машинного перевода с других языков. В работе таких систем используются модули морфологического, семантического и синтаксического анализа.

Гипертекстовые системы. Используют метод поиска данных по ключевым словам. Работа такой системы происходит в два этапа. На первом из входных данных выделяются ключевые слова. На втором обрабатывается информация извлеченная по выбранным ключевым словам.

Рис.2.1. Классификация интеллектуальных систем

Системы контекстной помощи. При работе с такой системой пользователь описывает свой запрос, а система задает ему вопросы для конкретизации проблемы. На основе этой информации генерируется решение исходной задачи.

Системы когнитивной графики. Используют графические образы для представления моделируемых и наблюдаемых процессов. Это позволяет пользователю легче воспринимать выводимую информацию и быстрее осваивать программный интерфейс интеллектуальной системы.

Экспертные системы создаются для частичной замены специалиста-эксперта в определенной области. Они позволяют формировать советы или принимать решения, основываясь на заранее записанном и формализованном опыте эксперта. Одним из важных качеств ЭС является возможность вывести ход своих рассуждений при принятии решения.

Самообучающиеся интеллектуальные системы используют методы автоматической классификации ситуаций из реальных примеров. Примеры реализованы в обучающей выборке и для каждого из них сформированы ожидаемые критерии результата. В процессе обу-

чения система сама генерирует правила, на основе которых способна классифицировать ту или иную исходную ситуацию.

Адаптивные информационные системы применяются в областях, где исходные данные постоянно изменяются. Адаптивные системы отвечают двум специфическим требованиям:

- способны отображать знания в каждый момент времени;
- пригодны для быстрой реконструкции при изменении проблемной среды.

Рассмотрев типы интеллектуальных систем, можно сделать вывод, что для решения задачи таргетирования рекламы подходит подтип экспертных системы, позволяющих управлять процессом генерации рекламных событий.

2.2. Типовая структура экспертных систем

Определившись с типом интеллектуальной системы, стоит рассмотреть ее типовую структуру. В дальнейшем, это позволит правильно спроектировать архитектуру программного приложения.

Экспертная система может быть двух видов: статическая и динамическая. Отличие динамической системы от статической в том, что она учитывает изменение знаний, которые могут произойти в процессе решения задачи.

В общем случае статическая ЭС состоит из 6 основных компонентов:

- база знаний;
- база данных или рабочая память;
- решатель (интерпретатор);
- система объяснений;
- компоненты приобретения знаний;
- интерфейс взаимодействия с пользователем.

База знаний ЭС включает в себя данные, описывающие некоторую предметную область. База знаний, так же содержит правила, позволяющие производить логическое рассуждение и преобразовывать хранимые знания в решения поставленных задач.

База данных (рабочая память) содержит данные, используемые в решении текущей задачи.

Решатель (интерпретатор) осуществляет процесс получения решения с помощью последовательной обработки правил, данных из базы знаний и рабочей памяти.

Система объяснений предназначена для визуализации процесса получения решений. Данная система отображает используемые в процессе решения данные и правила. Этот компонент может значительно повысить простоту тестирования системы и степень доверия к найденному решению.

Компоненты приобретения знаний служат для добавления новых правил и знаний в ЭС.

В процессе получения решения входные данные о задаче сохраняются в рабочую память. Затем данные из рабочей памяти обрабатываются решателем с использованием правил из базы знаний. Таким образом получается решение исходной задачи. В отличие от обычной программы ЭС может в процессе работы генерировать дополнительные последовательности операций, способствующих нахождению решения.

Динамические ЭС применяются для решения отдельного, широко распространенного класса задач, в которых во время нахождения решения могут возникнуть дополнительные факты, обусловленные внешним воздействием. Такие ЭС в дополнение к описанным компонентам включают в себя подсистему моделирования внешнего мира и подсистему связи с внешним окружением.

Подсистема моделирования внешнего мира служит для оценки, анализа и прогнозирования окружающей системы. Для того чтобы система использовала актуальные данные о внешней среде, необходимо постоянно обновлять знания, хранимые в ЭС.

Компонент связи с внешним миром необходим для интеллектуальных систем управления, а также для автономных интеллектуальных систем. Зачастую этот компонент реализован с использованием системы датчиков и контроллеров.

Опираясь на выше изложенный материал можно сказать, что разрабатываемый программный модуль будет иметь структуру статиче-

ской экспертной системы. Однако в нем так же будет и элемент динамической ЭС. Этим элементом является компонент связи с внешним миром, который реализован в виде Wi-Fi сканер. Именно Wi-Fi сканер постоянно добавляет информацию из окружающей среды в клиентскую базу знаний.

2.3. Инструментальные средства создания экспертных систем

Инструментальные средства, используемые для разработки ИС определяют степень трудозатратности, проделываемой работы. Поэтому их рассмотрение является важной частью создания интеллектуальных систем. Инструментальные средства, используемые при разработке приложений классифицируются по следующим параметрам:

- уровень используемого языка;
- способ представления знаний;
- механизмы вывода и моделирования;
- средства приобретения знаний;

Уровень используемого языка. На трудоемкость процесса разработки ЭС сильно влияет язык программирования(ЯП). Основными параметрами для оценки ЯП служат его универсальность и мощность.

С точки зрения языка разработки, можно выделить 5 подходов к созданию ЭС:

1. Традиционные языки программирования высокого уровня, такие как C++ и Java. Использование таких средств несет за собой как плюсы так и минусы. С одной стороны это сделает разработку более трудоемкой, однако с другой — позволит реализовывать экспертную систему без каких-либо ограничений.

2. Специальные языки программирования. К таким языкам относятся язык LISP, язык логического программирования PROLOG, язык рекурсивных функций РЕФАЛ и т.д. Недостатком данных языков является сложность в их использовании с другими языками, созданными для решения прикладных задач.

3. Инструментальные средства, уже содержащие большинство компонентов ЭС. Данным программным обеспечением может воспользоваться квалифицированный разработчик, владеющий навыками программирования и умеющий совмещать и использовать различные технологии и компоненты в одной системе.

4. Оболочки ЭС общего назначения. Такие системы включают в себя все программные компоненты, но не специализируются на применении в конкретной области. Обеспечение данного класса не требует от разработчика знания языков программирования.

   Использование средств этого класса могут привести к возникновению некоторых трудностей:

   - Заложенные в систему вывода механизмы могут ограничивать или вовсе не позволять генерировать решение, которое использует в своей работе эксперт. Это может привести к генерации неправильных или некачественных решений.
   - Способ представления знаний может оказаться непригодным для структурирования информации из конкретной предметной области.

5. Среды, ориентированные на решение задач в конкретной области:

   - Проблемно-ориентированные средства — содержат дополнительные программные компоненты, предназначенные для решения задач определенного типа. Например, задач управления, прогнозирования или поиска.
   - Предметно-ориентированные средства. В таких средствах типы предметных областей заранее известны, что значительно уменьшает время, затрачиваемое на разработку БЗ.

Для решения задачи таргетирования рекламы наиболее подходящими являются традиционные языки программирование высокого уровня. Это позволит создать свои варианты правил, задаваемых пользователем и свой интерпретатор этих правил.

Система представления знаний. Способов представления знаний достаточно много. Это вызвано стремлением качественно и целостно описать данные из различных прикладных областей. Зачастую

способ представления знаний в ЭС определяется методом представления знаний. К самым распространенным моделям представления знаний относятся:

- правила (продукции);
- фреймы (объекты);
- семантические цепи;
- логические цепи.

В разрабатываемой системе пользователи сами будут добавлять правила генерации рекламных событий для клиентов. Это позволяет выбрать в качестве модели представления знаний записи в виде правил.

Механизмы вывода и моделирования. Для статической ЭС характерно то, что единственным компонентом, изменяющим информацию, является механизм вывода. В динамических ЭС на данные так же влияют изменения окружающей среды, информация о которых поступает извне или эмулируется специальным компонентом. В различных системах механизмы вывода могут отличаться друг от друга вариантами реализации следующих процедур:

1. Структура процесса получения решения:
    - конструирование дерева вывода на основе обучающей выборки и дальнейший выбор для решения задачи.
    - построение сети вывода из специальных правил в процессе получения знаний и генерация решения на сети в процессе решения задачи.
    - компиляция сети вывода и поиск решения в режиме решения задачи. При этом сеть вывода генерируется исходя из данных удовлетворяющих условиям правил, применяемых к ним.
    - в процессе решения задач ЭС при отсутствии достаточного количества данных производит выборку наиболее корректных решений, приводит их обоснование, генерирует альтернативные сети вывода и осуществляет поиск решений в этих сетях.

2. Возможны три варианта направлений поиска решений: от цели к данным, от данных к цели и двунаправленный поиск.

Средства приобретения знаний. Средства приобретения знаний характеризуются следующими признаками:

1. Уровень языка приобретения знаний:

   - формальный язык;
   - ограниченный естественный язык;
   - язык пиктограмм и изображений;
   - естественно-языковой интерфейс и язык изображений.

   В реализуемом модуле будет использован формальный язык, позволяющий описывать некоторые характеристики.

2. Тип приобретаемых знаний:

   - информация в табличном виде, которая содержит параметры входных и выходных данных, по которым индуктивными методами компилируется дерево вывода;
   - специализированные правила;
   - общие и специализированные правила.

   Приобретаемыми знаниями в системе являются специализированные правила, задаваемые пользователем.

3. Тип приобретаемых данных:

   - атрибуты и значения;
   - объекты;
   - классы структурированных объектов и их экземпляры, получающие значения атрибутов путем наследования.

Результатом работы программного модуля должны стать объекты, позволяющие осуществить событие рекламного характера для конкретного клиента.

2.4. Резюме

В данной главе рассмотрены виды интеллектуальных систем, их основные компоненты и модели реализации. Это позволяет получить представление о том, как необходимо конструировать ИС для целей таргетирования рекламы. В результат принято решение, что разрабатываемый программный компонент станет статической экспертной системой в состав которой, войдет компонента связи с внешним миром. Моделью представления знаний для данной ЭС станет список записей в виде правил. Средством приобретения знаний для разрабатываемой системы будет инструмент формирования правил, которым сможет воспользоваться пользователь. В результате работы, ЭС должна генерировать рекламные события.

3. ОБЛАЧНЫЕ СИСТЕМЫ КАК ПЛАТФОРМА ДЛЯ АНАЛИЗА ДАННЫХ ЦЕЛЕВОЙ РЕКЛАМЫ

Целевой аудиторией для разрабатываемой системы являются малый и средний бизнес, которые не готовы тратить значительные денежные и временные ресурсы на покупку программного обеспечения и установку необходимого оборудования. С этой точки зрения, для данной аудитории выгодно использование системы как услуги. На сегодняшний день такой подход позволяют реализовывать облачные технологии, которые развиваются все стремительнее.

3.1. Выбор типа облачной технологии для использования в системе таргетирования рекламы

С появлением сверхскоростных каналов передачи данных, облачные системы получили сильный толчок в развитии. Большие скорости позволили устранить влияние больших расстояний между компьютерами на выполнение совместных задач. Такое развитие способствовало переносу большого количества программного обеспечения в облако. На сегодняшний день все чаще встречаются продукты имеющие как версию для стационарного компьютера, так и онлайн версию, созданную с использованием облачных технологий.

Облачные технологии удобны как для поставщика услуг так и для потребителя. Поставщик услуг, или провайдер, предоставляет пользователю часть ресурсов в аренду. Если число клиентов позволяет постоянно загружать рабочие ресурсы, то провайдер получает значительную прибыль от арендной платы. Это позволяет ему быстро окупить затраты на приобретение дорогостоящего оборудования и разработку программного обеспечения. С другой стороны, потребитель получает в свое распоряжение высокопроизводительные ресурсы, приобрести которые он не в состоянии.

Выделяют три основных типа облачных платформ [23]:

1. IaaS (Infrastructure-as-a-Service) — инфраструктура как сервис. Сюда относится организация аппаратной среды с организо-

ванными серверами, настроенными средствами коммуникации, устройствами хранения информации и средствами бесперебойного питаний. На потребителя ложится задача по развертыванию и настройке требуемой системы, установке общего и специализированного программного обеспечения;

2. PaaS (Platform-as-a-Service) — платформа как сервис. По сравнению с IaaS, в PaaS технологии уже настроена операционная система для выполнения приложений. Также, могут быть уже развернуты средства разработки, отладки и поддержки. Потребитель получает возможность развернуть и сопровождать специализированное программное обеспечение такое как, например, ПО для бухгалтерского учета;

3. SaaS (Software-as-a-Service) — программное обеспечение как сервис. В данном случае поставщик предоставляет пользователю доступ к программному обеспечению, которое разработано для решения определенного типа задач. Пользователь получает доступ только к эксплуатации программы. Ответственность за администрирование лежит на поставщике. К таким системам относятся многие CRM системы такие как amoCRM [12] или VtigerCRM [26].

Разрабатываемая система таргетирования рекламы является программным продуктом, который удобнее использовать, если он уже развернут и настроен. Это значит, что в качестве технологии для организации системы удобнее всего использовать третий подход — программное обеспечение как сервис (SaaS). Чтобы определить наиболее важные компоненты такой системы необходимо рассмотреть ее достоинства и недостатки.

3.2. Преимущества и недостатки размещения системы таргетированной рекламы в облачной инфраструктуре

Рассмотрим преимущества облачных систем.

Первое преимущество помещения программного продукта в облако следует из их популярности. Облачные технологии очень удобны для конечного пользователя, что привлекает к их использованию

большое количество компаний и частных лиц. Рассмотрим список наиболее важных преимуществ с точки зрения потребителя:

- низкие материальные затраты. В отличии от обычного ПО, клиент не покупает программу в облаке, а берет ее в аренду, что значительно дешевле. Уменьшение затрат происходит и за счет устранения необходимости в производительном аппаратном обеспечении и обслуживающем его персонале;
- временная аренда. Клиент может в любой момент отказаться от использования продукта, если он его не устраивает;
- высокая надежность. Данные, хранимые на удаленных серверах всегда имеют резервную копию, что минимизирует вероятность их потери;
- защита от кражи данных. Провайдер облачных вычислений всегда принимает меры по защите данных от вирусных и хакерских угроз;
- своевременное обновление используемого ПО;
- мобильность. Облачный сервис доступен из любой точки, в которой у клиента есть доступ к интернету. Также, большинство поставщиков услуг предоставляет возможность доступа из специально разработанного мобильного приложения.

Вторым преимуществом является возможность своевременного обновления программного обеспечения. Любая найденная ошибка или уязвимость может быть моментально устранена для всех клиентов облачной системы. Это является огромным плюсом с точки зрения безопасности, так как большинство эксплуатируемых уязвимостей программных и аппаратных систем находится именно в устаревшем ПО.

К третьему преимущество относится большое количество пользователей, являющихся в тоже время тестировщиками разработанной программы. Любая ошибка, случайным образом попавшая в рабочую версию системы будет быстро найдена и устранена благодаря обращениям клиентов. Однако это преимущество не должно исключать этап тестирования перед обновлением программы, т.к. выявленные клиентами ошибки плохо влияют на репутацию компании.

Рассмотрим недостатки облачных систем.

Главным недостатком облачной системы являются высокие требование к их надежности. Как было описано в преимуществах с точки зрения пользователя, облачная система должна гарантировать надежность хранимых данных. Для обеспечения этой надежности провайдерам необходимо заботиться о резервном копировании данных, бесперебойной работе оборудования, бесперебойном питании. Также необходимо постоянно поддерживать актуальность средств защиты от вирусных и хакерских атак.

Вторым недостатком является теоретическая возможность потери данных. Если шанс единовременной потери данных на основном и резервном сервере крайне мал, то возможность пострадать от уязвимости нулевого дня исключать не стоит [13]. Уязвимостью нулевого дня называется ошибка в программном коде или аппаратуре о которой стало публично известно и у разработчика было 0 дней на ее исправление. Потеря данных или взлом сервера является большим ударом по авторитету компании и неминуемо приведет к многочисленным затратам, в том числе и компенсации пользователям.

Третий минус относится как к недостаткам для поставщика услуги, так и для потребителя. В данной ситуации речь идет об интернет соединении. Разрыв в сети может возникнуть в любом месте между клиентом и сервером и это полностью прервет любые коммуникации между сторонами. Чтобы уменьшить риск отключения со стороны провайдера, ему необходимо позаботиться о резервной линии интернет соединения. В дополнение ко всему, поставщику услуг следует позаботиться не только о надежности соединения но и о ширине канала связи.

Рассмотрев основные недостатки облачных технологий становится понятна необходимость рассмотрения вопроса о безопасности пользовательских данных с которыми работает система. Следует отметить что под ответственность провайдера попадают как данные хранимые на сервере так и данные пересылаемые между сервером и клиентским приложением (браузером).

3.3. Проблемы безопасности облачных SaaS технологий и способы их решения

Следует понимать что вопросов, касающихся безопасности много и не за все из них отвечает поставщик услуги. Некоторая ответствен-

ность возлагается и на потребителя. Основными проблемами безопасности облачных вычислений являются следующие категории [16]:

- конфиденциальность и целостность персональных данных;
- защита от несанкционированного доступа;
- уязвимости программное обеспечение, используемого для поддержания работоспособности сервера;
- общие уязвимости web сервисов.

Конфиденциальность и целостность персональных данных.

Хранимая на сервере клиентская информация является самой ценной составляющей всей облачной платформы с точки зрения пользователя. Защищенность хранимых данных и их конфиденциальность зачастую является самым важным аспектом при выборе оператора облачных вычислений. Из этого следует необходимость уделения особого внимания к вопросам безопасности клиентских данных.

К угрозам персональной информации относятся угрозы целостности хранимой информации и их конфиденциальности.

Целостность данных — это гарантия, что данные не будут повреждены или потеряны в период действия договора между провайдером и потребителем облачных услуг. На целостность данных могут повлиять как физические так и программные факторы. Например, сбой в подаче электроэнергии может привести к потере несохраненной информации. Для минимизации такой угрозы следует иметь возможность перехода на резервный источник питания. Физическая невозможность считывания информации с носителя, связанная с его повреждением или устареванием приведет к потере части данных. Сбой в работе программного обеспечения способен навредить большому количеству клиентских данных. Потери такого характера практически невозможно восстановить имея единственный носитель информации. В связи с этим, лучшим вариантом обеспечения сохранности данных, является ее резервное копирование на дополнительных устройствах.

Другая угроза связана с конфиденциальностью информации.

Конфиденциальность информации — запрет передачи определенной информации посторонним лицам без согласия ее обладателя. Например, конфиденциальность персональных данных — обязательное

для соблюдения оператором или иным получившим доступ к персональным данным лицом требование: не допускать их распространения без согласия субъекта персональных данных или наличия иного законного основания [7].

Определение конфиденциальности информации накладывает на оператора ответственность за получение данных сторонними лицами. Утечка информации хоть и мало вероятна, однако возможна. Ее последствием может стать раскрытие такой информации о клиентах как номера телефонов, ФИО, год рождения, номера банковских карт. Это важная информация, которая может быть использована для кражи денежных средств или использования данных о личности в других махинациях. Существует много зафиксированных случаев утери информации с серверов крупных компаний. Например, крупная утечка информации с серверов компании eBay произошла в 2014 году [6].

Для защиты от подобных инцидентов принято использовать шифрование хранимой информации. В том же случае с компанией eBay, украденная информация была зашифрована, что позволило избежать серьезных последствий. Существует два основных способа шифрования — классический способ и использование гомоморфного шифрования.

Классический способ подразумевает шифрование данных одним из алгоритмов симметричного шифрования, таких как AES, DES или RC4 [14]. Проблема данного способа заключается в необходимости хранения ключей шифрования. Ключи шифрования должны храниться в месте, недоступном для злоумышленника, даже если он сумел получить данные клиентов.

Гомоморфное шифрование подразумевает возможность выполнения произвольных операций над зашифрованными данными, без необходимости расшифровывать их. Идея гомоморфного шифрования возникла в 1978 году а теоретическое доказательство ее возможности появилось только в 2009 году. Сейчас теория стремительно развивается и упрощается. Появляются некоторые реализации данного способа шифрования, например библиотека HElib от компании IBM [15]. Основной проблемой использования гомоморфного шифрования является его низкая производительность.

Защита от несанкционированного доступа.

Взлом учетных записи позволяет злоумышленнику получить доступ ко всему сервису. Доступ к приватной информации может быть получен как из внутренней, так и из внешней сети приложения.

Обиженные плохим отношением или малой зарплатой администраторы могут серьезно навредить безопасности организации. Права администратора позволяют делать что угодно во внутренней сети компании. Это может повлечь значительные финансовые потери или потери в производительности сервиса. В свою очередь это повлияет на репутацию бренда. Чтобы избежать проблемы с человеческим фактором необходимо тщательно подбирать рабочий персонал и следить за работой сотрудников.

Следует помнить и об API облачной системы. Обычно API находится в открытом доступе и имеет хорошую документацию. Это позволяет любому разработчику в своей программе взаимодействовать с облачной средой. API, легко может быть использовано и в десктопном и в мобильном приложении так как, зачастую, оно реализовано по протоколу http или https. Доступность и понятность программного интерфейса может сыграть и отрицательную роль, так как открывает новый вектор атак для злоумышленника. Плохо продуманный с точки зрения безопасности интерфейс привнесет в систему множество уязвимостей, недоступных в веб-интерфейсе системы. API позволяет злоумышленники многократно использовать токен доступа к системе, а при получении несанкционированный доступ к учетным записям скачивать конфиденциальную информацию и осуществлять любые манипуляции от имени пользователя.

Для того чтобы риск взлома был минимален, следует подобрать качественный способ авторизации и аутентификации пользователей. Существует три основных типа аутентификации, основанных на трех специальных факторах [22]:

- что-то что вы знаете — это может быть специальной секретной информацией, такой как пароль или ответ на секретный вопрос, который не известен никому. Это фактор знания;
- что-то что вы имеете — это предмет которым вы владеете, такой как смарт карта или другие подобные электронные устройства. Это фактор владения;
- что-то что является вашей часть — это физическое свойство, такое как отпечаток пальца или голос, которые позволяют вас идентифицировать. Это фактор неотъемлемости.

В веб-приложениях обычно используется первый тип аутентификации. К этому типу относятся следующие методы аутентификации,

которые давно и успешно используются во многих системах [4]:

- аутентификация по паролю;
- аутентификация по сертификатам. Как правило, используются сертификаты стандарта X.509, которые позволяют установить обе стороны общения;
- аутентификация по одноразовым паролям. Сюда относятся программные и аппаратные токены для генерации паролей, коды получаемые пользователем по смс, заранее подготовленные списки одноразовых паролей;
- аутентификация по ключам доступа. Этот способ часто применяется в API веб-приложений. Сервером генерируется специальный ключ, заменяющий логин и пароль, который в дальнейшем используется в приложении для взаимодействия с сервисом;
- аутентификация по токенам. Этот способ применяется при построении распределенных систем, где один сервис доверяет функцию аутентификации другому. Такой метод можно наблюдать при аутентификации в каком либо приложении через социальную сеть.

Уязвимости программного обеспечения.

Программное обеспечение, поддерживающее работоспособность сервера так же подвержено рискам быть атакованными [2]. Сюда относятся так же атаки на операционные системы и используемые сетевые протоколы. Для предотвращения таких атак достаточно использовать межсетевые экраны и системы обнаружения и предотвращения вторжений. Такие подсистемы позволяют экранировать уязвимости операционной системы и приложений до момента, когда будут установлены важные обновления. Такие системы устанавливаются в виде программного агента, что позволяет экранировать уязвимости, обнаруженные в ОС и приложениях: защита от любых атак на известные уязвимости без установки критических обновлений; блокировка атак типа XSS (Cross Site Scripting) и SQL-Injection.

Общие уязвимости веб-сервисов.

Все уязвимости веб-сервисов распространяются также и на облачные SaaS системы, так как с точки зрения внешнего пользователя

облако имеет такой же интерфейс. Ежегодно, сообществом OWASP публикуется Топ-10 уязвимостей веб-приложений.

OWASP – это открытый проект обеспечения безопасности веб-приложений. Топ-10 уязвимостей от OWASP представляет собой перечень наиболее критичных рисков безопасности приложений и отражает текущие проблемы в области ИБ.

В 2017 году этот список представлен следующими пунктами [25]:

- A1 — внедрение кода;
- A2 — некорректная аутентификация и управление сессией;
- A3 — межсайтовый скриптинг;
- A4 — небезопасные прямые ссылки на объекты;
- A5 — небезопасная конфигурация;
- A6 — утечка чувствительных данных;
- A7 — отсутствие контроля доступа к функциональному уровню;
- A8 — подделка межсайтовых запросов;
- A9 — использование компонентов с известными уязвимостями;
- A10 — невалидированные редиректы.

Большинство из этих уязвимостей давно известны, как и способы защиты от них. Проблема их распространения заключается в том, что при разработке веб-приложений безопасности уделяется мало внимания. Это приводит к тому что современные веб ресурсы могут быть подвержены уязвимости десятилетней давности.

Рассмотренные угрозы безопасности и методы их решения относились к части облачных вычислений, которые находятся на стороне сервиса. Далее рассмотрим не менее важный аспект безопасности пользовательских данных — это защита информации в каналах связи.

3.4. Безопасность клиентских данных в каналах связи

Самый распространенный протокол прикладного уровня для передачи данных в сети интернет это http протокол. Он включает в себя большое количество полей, что делает его удобным для различных задач — от отправки коротких сообщений, до передачи файлов. Тем не менее, с точки зрения безопасности http протокол никак не защищен. Простое прослушивание трафика между клиентом и сервером раскрывает всю информацию, передаваемую в канале связи. Для того чтобы защитить частную информацию был разработан протокол SSL, позднее переросший в TLS. В данной пункте будет рассмотрен принцип работы SSL/TLS протокола.

TLS протокол работает над транспортным TCP протоколом. Это делает его удобным для использования с вышестоящими протоколами такими как HTTP. Место протокола TLS показано на рис.3.1 Таким образом, при использовании TLS, HTTP протокол никак не изменяется. Именно комбинацией TLS и HTTP и является протокол HTTPS, который работает на 443 порту.

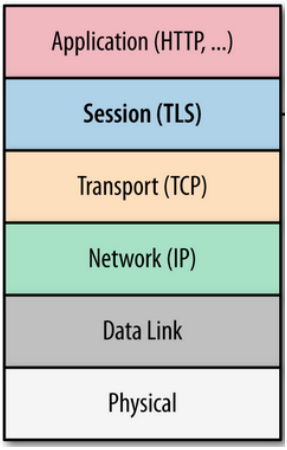


Рис.3.1. Место протокола TLS в стеке TCP/IP

Последней версией протокола на данный момент является TLS 1.3. Так как это протокол утвержден не так давно [21] — в начале

2018 года, наиболее популярной сейчас версией является TLS 1.2.

Протокол TLS обеспечивает три функции для вышестоящих протоколов:

- шифрование — информативная часть сообщения в канале связи не может быть прочитана третьей стороной;
- аутентификация — гарантия того, что собеседники являются именно теми за кого себя выдают;
- целостность — обнаружение подмены всего сообщения или его частей.

Взаимодействие клиента с сервером происходит в два этапа:

- установление TLS-сессии;
- передача данных по зашифрованному каналу.

Процесс установления сессии состоит из нескольких шагов:

- аутентификация;
- установление криптографических алгоритмов для обоих этапов;
- генерация общего секретного мастер-ключа;
- формирование на основе мастер-ключа общего сеансового ключа для защиты информационных сообщений.

Шифрование.

В протоколе используется как асимметричное, так и симметричное шифрование. Это связано с тем, что асимметричное шифрования более ресурсоемко по сравнению с симметричным. Проблемой же симметричного шифрования является необходимость в общем секретном ключе. Обе стороны должны знать секретный ключ для поддержания зашифрованного канала. Таким образом, комбинация двух типов шифрования позволяет исключить недостатки каждого из них. С помощью асимметричного шифрования стороны договариваются об общем секретном ключе, который в дальнейшем используют в шифрованном канале связи. В TLS используются системы асимметричного шифрования RSA, Диффи-Хеллмана и их модификации. Алгоритм

шифрования Диффи-Хеллмана является более современным и позволяет обеспечить прогрессивную секретность. Это значит, что при компрометации секретного ключа, записанные ранее сообщения расшифровать не удастся.

Аутентификация — сертификаты.

Сертификаты позволяют привязывать открытый ключ к некоторому сетевому имени. Сертификат не несет в себе какую-либо секретную информацию. Самая важная информация заключенная в нем — это открытый ключ, доменное имя или ip-адрес и цифровая подпись. Открытый ключ используется при шифровании сообщений и может быть подменена третьей стороной. Для того чтобы выявить факт подмены была разработана система удостоверяющих центров(УЦ). Для того чтобы система работала, вводится допущение о том, что пользователь доверяет центру сертификации. Центр сертификации гарантирует соответствие доменного имени и открытого ключа.

SSL сертификат состоит из двух условных частей. Первая часть эта передаваемая в открытом виде информации. Вторая часть — это цифровая подпись удостоверяющего центра. Если злоумышленник сумеет подменить сертификат, то принимающая сторона сможет выявить эту подмену. Цифровая подпись это контрольная сумма от открытой части сертификата, зашифрованная закрытым ключом центра сертификации. Для составления контрольный суммы используются криптосистемы RSA и ECDSA. Таким образом, для проверки валидности данных достаточно посчитать контрольную сумму от открытой части сертификата и сравнить ее с контрольной суммой, полученной после расшифровывания подписи открытым ключом центра сертификации.

Доменных имен в интернете очень много и поэтому центры сертификации всегда делегируют свои полномочия между другими организациями и дочерними компаниями. В свою очередь дочерние УЦ так же способны распределять свои обязанность. Это порождает цепочку из удостоверяющих центров, что благодаря системе подписей не мешает корректной проверке сертификата. Пользователю необходимо доверять лишь УЦ верхнего уровня, чтобы проверить всю последовательность.

Для обеспечения большей надежности, сертификаты выдаются на определенный срок. Также, они могут быть отозваны центром сертификации при их компрометации. Для проверки статуса сертификата используется протокол OCSP [8].

Целостность.

Для обнаружения подмены сообщения используется механизм HMAC (hash-based message authentication code — код аутентификации сообщения), называемый также имитовставкой. HMAC использует криптографическую хеш функцию для подсчета контрольной суммы передаваемого сообщения. Полученное значение вставляется в конец сообщения. Предполагается, что подделка HMAC вычислительно недостижима, без знания секретного ключа. Получается, что подмена сообщения, как и в случае с сертификатом, будет сразу выявлена на стороне получателя.

3.5. Резюме

В данной главе рассмотрены типы облачных систем их преимущества и недостатки. В результате, выделена технология SaaS, как наиболее подходящая для размещения компонента генерации рекламных событий. Рассмотрены угрозы безопасности данных на сервере облачных технологи и методы их устранения. Рассмотрены угрозы информации в каналах передачи данных. Рассмотрена структура TLS протокола, как способа защиты данных в каналах связи.

4. АНАЛИЗ СЕТЕВОЙ ИНФРАСТРУКТУРЫ

4.1. Обзор компонентов, необходимых для системы таргетирования рекламы

В предыдущих главах представлен обзор одного компонента всей системы таргетирования рекламы — компонента генерации рекламных событий. Этот компонент выполняет несколько функций:

- предоставляет пользователю возможность формировать правила для генерации рекламных событий;
- генерирует события исходя из правил и имеющихся данных;
- получает информацию о новых клиентах системы и регистрирует их;
- получает и сохраняет информацию о местах, посещенных клиентами;
- регистрирует новых пользователей системы.

Для того чтобы рассмотренный ранее компонент работал корректно, ему необходимы еще два дополнительных средства:

- компонент регистрации новых клиентов;
- Wi-Fi сканер.

Компонент регистрации новых клиентов.
Данный модуль является открытой Wi-Fi точкой доступа, которая установлена в помещении пользователя системы. Если организация пользователя работает в нескольких местах, то и точки доступа могут быть расположены в каждой из них. Если точка доступа установлена в помещении пользователя, то клиенты получают возможность бесплатно пользоваться интернетом. Это является хорошим стимулом для подключения к открытой сети.

Регистрация клиента происходит при первом его подключении к открытому Wi-Fi. Для получения доступа в интернет клиент должен

зарегистрироваться в системе. При регистрации запрашиваются такие данные как номер телефона, имя, возраст и пол человека, а также согласие на использование предоставленной информации. Полученная информация регистрируется на сервере и в дальнейшем используется для формирования рекламных сообщений.

Если клиент уже был ранее зарегистрирован в сети, то ему сразу предоставляется доступ в интернет без необходимости проходить процесс регистрации.

Wi-Fi сканер.

Wi-Fi сканер также работает по стандарту 802.11. Однако его задача — прослушивание эфира и выявление новых устройств, которые попадают в зону его досягаемости. Определив нового посетителя он формирует информацию о его идентификаторе, времени его появления и уходе из зоны досягаемости. После выхода посетителя из зоны сканирования, рассматриваемый компонент отправляет информацию на сервер, где она заносится в базу данных и в дальнейшем будет использована при формировании рекламных событий.

Wi-Fi сканер может быть установлен в любом месте и их количество может быть неограничено. Оптимальным вариантом являются места, где пользователю интересен факт появления и время появления его клиентов.

Общая схема работы все системы представлена на рис. 4.1

Далее необходимо рассмотреть некоторые вопросы связанные с работой двух описанных выше компонентов.

4.2. Обзор технологии Captive Portal для использования в целевой рекламе

Данный пункт относится к компоненту регистрации новых клиентов. Для того чтобы зарегистрировать клиента, его необходимо при первом же подключении перенаправлять на сервер регистрации. Для таких задач существует технология Captive Portal. Рассмотрим ее более детально.

Работа Captive Portal может базироваться на двух подходах:

- перенаправление за счет DNS запросов;
- перенаправление внутренними средствами маршрутизации.

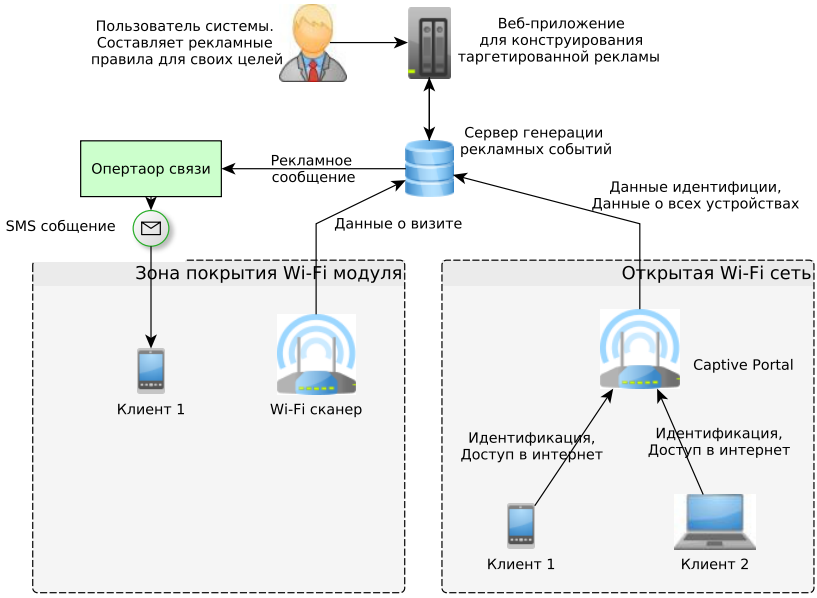


Рис.4.1. Комплексная архитектура всей системы

При первом варианте DNS сервер настраивается таким образом, что на все запросы нового клиента ответ содержит адрес сервера регистрации. После регистрации клиента, его устройство начинает получать корректные ответы от DNS сервера. Такая системы может быть легко обманута при статической настройке DNS сервера на устройстве клиента. Исключением является случай, когда сетевой экран настроен так, чтобы устранить возможность работы от стороннего DNS.

Второй вариант работы более надежен и подразумевает настройку маршрутизации таким образом, что все запросы по протоколу HTTP или HTTPS перенаправляются на локальный сервер. Captive Portal, организованный на платформе linux, может быть настроен с использованием утилиты iptables. В таком случае, после регистрации клиента в системе, для MAC адреса его устройства добавляется исключение, позволяющее ему использовать интернет.

Сценарий работы технологии Captive Portal существует достаточ-

но давно, что делает его известным и простым в использовании. Данная технология позволяет доносить до клиентов дополнительную рекламу, что выгодно для любого бизнеса. Так же, для Российской Федерации введен закон, запрещающий наличие открытой сети без регистрации. Это означает что любая компания, предоставляя бесплатный интернет, не только может регистрировать клиентов, но и обязана это делать. Эти факторы делают технологию Captive Portal популярной и распространенной.

Популярность сервиса привела к тому, что в устройства, способные подключаться к Wi-Fi, добавляется специальная функциональность, позволяющая проще и быстрее совершать регистрацию в сети.

В ОС Android старше версии 4, мобильное устройство запрашивает с одного из серверов Google файл с названием generate_204. В ответ оно ожидает код 204 (No Content). При отсутствии в ответе этого кода создается уведомление, по клику на которое запускается браузер со страницей из ответа. Поведение ОС Android может различаться для разных производителей. В некоторых вариантах вместо уведомления может сразу быть открыта страница авторизации. Так же различен и сам браузер — это может быть как стандартный браузер системы, так и специально созданной для этих целей простейший браузер.

В операционной системе iOS устройства также запрашивают файл с одного из сайтов Apple и сверяют его содержимое с ожидаемым. В случае обнаружения различий запускается утилита Captive Network Assistant, представляющий собой браузер без поддержки HTTP cookies.

Рассмотрение сервиса Captive Portal позволяет понять, что его функциональность полностью подходит для решаемой задачи. Рассмотренная технология позволяет регистрировать каждого нового клиента в системе, при этом запрашивая у него всю информацию, которая необходима для работы системы таргетирования рекламы, и которой клиент согласен поделиться.

4.3. Возможности стандарта IEEE 802.11 для выявления клиентов в зоне действия модуля Wi-Fi

Задачей Wi-Fi сканера является выявление новых устройств в зоне действия Wi-Fi модуля и последующая регистрация этого собы-

тия. Для того чтобы корректно выявлять новые устройства необходимо понимать работу протокола 802.11 и знать какие типы сообщений в нем используются.

IEEE 802.11 — это стандарт, описывающий коммуникацию оборудования по беспроводной сети в частотных диапазонах 0,9; 2,4; 3,6; 5 и 60 ГГц[1]. Стандарт состоит из набора других стандартов ориентирующихся на конкретные вопросы. Например стандарт 802.11i описывает безопасность связи, а 802.11n вариант коммуникаций со скоростью до 600 МБит/с. Стандарт описывает работу на канальном уровне модели OSI.

Стандарт 802.11 описывает три типа фреймов

1. Фреймы Управления (Management frames).
2. Фреймы Контроля (Control frames).
3. Фреймы Данных (Data frames).

Внутри каждого фрейма имеются поля, которые определяют версию протокола тип фрейма и различные идентификаторы. Также каждый фрейм несет в себе информацию о MAC-адресе отправителя и MAC-адресе получателя.

Фреймы Управления (Management Frames)

Фреймы управления используются для установления и поддержания канала общения между коммутирующими устройствами.

Всего стандарт 802.11 определяет 14 типов фреймов управления:

1. association request;
2. association response;
3. reassociation request;
4. reassociation response;
5. probe request;
6. probe response;
7. beacon;
8. ATIM (Announcement traffic indication message);

9. disassociation;
10. authentication;
11. deauthentication;
12. action;
13. action no ack;
14. timing advertisement.

Рассмотрим некоторые фреймы.

Фрейм аутентификации (Authentication frame) отправляется устройством, желающим подключиться к точке доступа. В фрейме присутствует идентификационная информация об устройстве. Точка доступа может принять идентификацию или отвергнуть.

Фрейм ассоциации (Association request frame). Этот фрейм отправляется от устройства к точке доступа. Данным фреймом устройство просит точку доступа зарезервировать ресурсы для будущей сессии. Внутри фрейма содержится информация о характеристиках канала, таких как скорость передачи, с которыми может работать устройство. Для идентификации точки доступа, в данном фрейме отправляется SSID(имя) сети. Если точка доступа принимает запрос ассоциации, то она отвечает устройству фреймом ответа на ассоциацию, внутри которого хранится идентификатор ассоциации.

Фрейм — маячок (Beacon frame) периодически отправляется точками доступа и содержит информацию, такую как SSID точки доступа, частотный канал, временные маркеры для синхронизации времени, поддерживаемые скорости и другое. Этот фрейм позволяет всем устройствам определять досягаемые точки доступа и каналы на которых они работают.

Фрейм — проба (Probe request frame) отправляется мобильными устройствами, для установления размещенных в зоне покрытия точек доступа. Запрос может быть как широковещательным, так и с указанием конкретных SSID точек доступа. В ответном фрейме точка доступа отправляет информацию о функциональности, поддерживаемых скоростях и другое.

Подключение Wi-Fi устройства к точке доступа происходит несколько этапов. Первым этапом является этап аутентификации, затем этап ассоциации. Если точка доступа защищена паролем, то тре-

тьим этапом является проверка пароля, а при наличии шифрования и установление его характеристик.

Фреймы контроля (Control Frames)

Фреймы контроля позволяют доставлять данные между Wi-Fi устройствами. В стандарте 802.11 определено 9 типов фреймов контроля. Однако в отличие от фреймов управления они не несут какой-либо полезной информации в условиях нашей задачи.

Фреймы данных Wi-Fi

Главной задачей протокола является передача информации протоколов вышестоящего уровня. Для этого и предназначены 15 типов фреймов данных, описанных в стандарте 802.11. Фреймы данных используются уже при установившемся соединении, из чего следует, что для задачи выявления новых устройств они не интересны.

Исходя из вышесказанного, следует, что наиболее интересным является управляющий фрейм Probe Request. Он несет в себе информацию о MAC-адресе мобильного устройства и отправляется постоянно при поиске доступных Wi-Fi сетей. Таким образом, устройство, пронесенное мимо Wi-Fi сканера со включенным модулем Wi-Fi будет рассылать запросы на поиск сетей и попадет в список зафиксированных устройств. Это позволит определить факт посещения клиентом определенного места.

4.4. Резюме

В данной главе рассмотрены все оставшиеся компоненты системы формирования целевой рекламы. Таким образом всего в системе три компонента: компонент регистрации новых пользователей, Wi-Fi сканер и сервер с интеллектуальной системой для формирования рекламных событий. Как было установлено в главе про облачные системы, оптимальным протоколом взаимодействия между компонентами является HTTPS протокол. Дальнейшим этапом станет подбор средств разработки и сама разработка описанных компонентов.

5. ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

5.1. Проектирование и разработка компонента регистрации новых клиентов

Компонент регистрации новых клиентов предназначен для решения четырех задач:

1. Перенаправление всех пользователей заданной подсети на сервер регистрации.
2. Поддержание работы сервера регистрации.
3. Регистрация новых клиентов на удаленном сервере.
4. Предоставление интернета зарегистрированным клиентам.

5.1.1. Проектирование архитектуры

Для того чтобы рассматриваемый компонент был способен реализовывать поставленные задачи, было принято решение, что архитектура приложения будет разделена на три основных модуля. На рис.5.1 изображена архитектура всего компонента и системы, с которыми он взаимодействует.

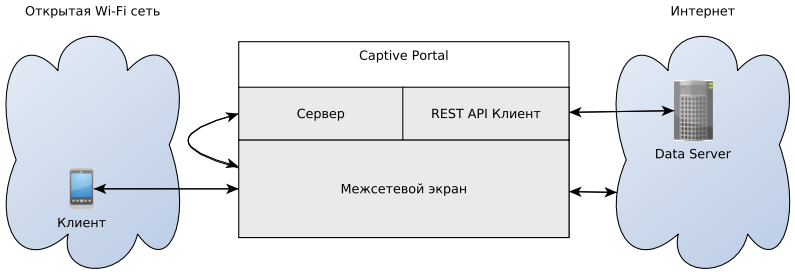


Рис.5.1. Архитектура компонента регистрации

Сервер необходим для первоначального обслуживания клиентов. Данный модуль слушает 80 и 443 порты и работает по протоколу HTTP и HTTPS соответственно. В его задачи входит:

- показ страницы регистрации;
- показ пользовательского соглашения;
- получение регистрационных данных;
- формирование события регистрации клиента;
- формирование события, разрешающего клиенту доступ в интернет.

REST API Клиент позволяет проверять MAK адрес устройства клиента в базе системы. При наличие адреса, клиенту предоставляется доступ в интернет. Данный модуль также отвечает за регистрацию новых клиентов. Передача данных между модулем и сервером системы осуществляется по протоколу HTTPS, что позволяет защитить информацию от злоумышленников.

Межсетевой экран осуществляет следующие функции:

- перенаправляет все запросы новых клиентов на сервер Captive Portal;
- блокирует доступ в интернет всем не авторизованным пользователям;
- по команде от сервера Captive Portal предоставляет клиентам доступ в интернет.

5.1.2. Разработка компонента

Для организации открытой Wi-Fi сети решено использовать утилиту hostapd [19]. Утилита позволяет создавать Wi-Fi точку доступа с различными параметрами путем конфигурации файла настроек. Используемый в компоненте файл настроек представлен в листинге 1.

Для корректной работы Wi-Fi сети необходимо наличие DHCP сервера. Роль DHCP сервера в рассматриваемом компоненте выполняет утилита dnsmasq. Для удобства обращение к серверу Captive Portal по установленному локальному имени (localcaptive) используется DNS

сервер, который также реализован в утилите dnsmasq. Конфигурационный файл dnsmasq представлен в листинге 2

Предполагается, что разрабатываемый программный компонент будет устанавливаться в устройства со слабой производительностью. К таким устройствам относятся большинство домашних роутеров. Ресурсы таких устройств сильно ограничены. Они могут иметь количество памяти до 20 мб и частоту одноядерного процессора до 500 MHz. В таких условиях использование таких серверов как apache и nginx существенно замедлит работу устройства. В связи с этим, принято решение разрабатывать свой сервер на языке C++.

Для ускорения разработки и уменьшения количества ошибок было решено использовать заготовку для веб-сервера[24], написанную на языке C++ и распространяемую по лицензии MIT[20]. В своей работе данная заготовка использует библиотеку boost.asio[17], которая позволяет при необходимости вносить изменения в исходный код как сервера так и библиотеки.

Технология Captive Portal подразумевает специфическую настройку межсетевого экрана в системе. В ядре Linux систем уже присутствует межсетевой экрана, который называется netfilter. Один из вариантов его настройки — использование консольной утилиты iptables. В листинге 4 приведен скрипт, который осуществляет необходимые настройка netfilter.

На рис.5.2 представлена диаграмма классов компонента регистрации

Рассмотрим каждый класс и наиболее значимые методы по отдельности.

CaptiveBaseServer.

Базовый класс выполняет функции по инициализации полей класса и настройке сетевого фильтра. Так же в его функции входит возврат исходных настроек netfilter, после прекращения работы программы.

CaptiveServer.

Функцией этого класса является поддержание работоспособности сервера и выдача команд межсетевому экрану и REST API клиенту.

`static void defaultGet (std::shared_ptr <typename SimpleWeb::Server <socket_type>::Response> response, std::shared_ptr <typename SimpleWeb::Server <socket_type>::Request> request)`. Метод вызывается при получении GET запроса на сервер. Его функция — в зависимости

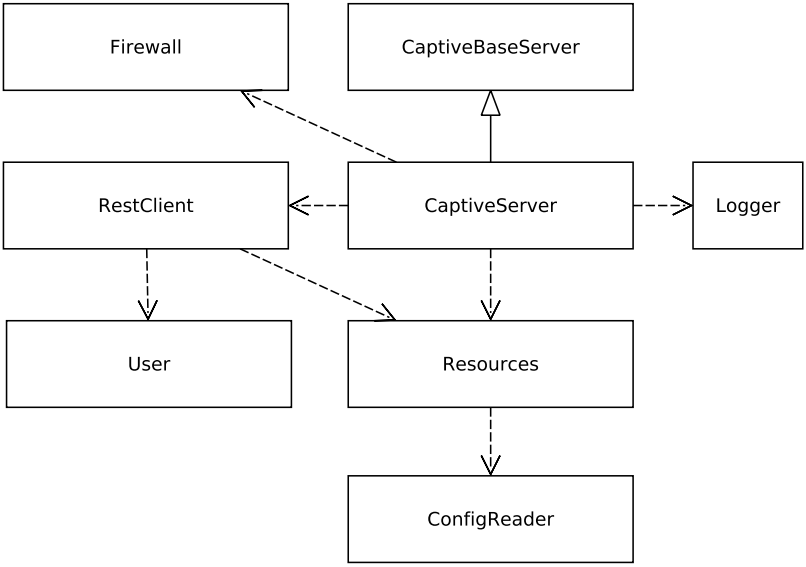


Рис.5.2. Диаграмма классов компонента регистрации

от mac-адреса устройства вернуть страницу регистрации, или пропустить клиента в интернет. Исходный код метода представлен в листинге 5

`template <class socket_type> static void registerPost (std::shared_ptr <typename SimpleWeb::Server <socket_type> ::Response> response, std::shared_ptr <typename SimpleWeb::Server <socket_type> ::Request> request)`. Метод вызывается при получении на сервер POST запроса. Это происходит во время регистрации клиентов. Полученные данные отправляются на сервер для регистрации нового клиента. При получении положительного ответа, клиенту предоставляется доступ в интернет.

Firewall.

Объект этого класса выполняет функции сетевого экрана, позволяющие блокировать клиентам доступ в интернет до момента получения команды из класса CaptiveServer.

`bool pass_mac(const std::string& ip)`. Метод позволяет от-

крыть доступ в интернет клиенту с адресом *ip*.

`bool saveIptablesRules(std::string fileName)`. Входным значением является имя файла, для сохранения предыдущих настроек сетевого фильтра. Метод использует утилиту iptables-save. Возвращает true при успешном завершении операции.

`bool Firewall::loadIptablesRules(std::string fileName)`. Входным значением является путь к файлу, в котором хранятся настройки сетевого фильтра. В работе используется утилита iptables-restore.

`std::string exec(const char cmd)`. Метод исполняет команду *cmd* в системе.

`std::string mac_by_ip(const std::string& ip)`. Позволяет получить mac-адрес клиента по заданному *ip*. Для получения mac-адреса используется arp таблица, хранящаяся в системе.

RestClient.

Данный класс регистрирует на сервере новых клиентов и проверяет наличие MAC адреса в базе данных сервера таргетированной рекламы.

Метод `bool RestClient::isClientExist(std::string mac)` получает в качестве параметра MAC адрес устройства клиента в виде AA:BB:CC:DD:EE:FF. Его задача установить зарегистрирован ли данный MAC в системе. Возвращаемое значение равно true если адрес уже существует. В противном случае возвращается false.

Метод `bool RestClient::registerClient(User user)` получает в качестве параметра объект класса User и совершает попытку его регистрации на сервере. В случае успеха возвращается true, иначе false.

User.

Отвечает за данные, полученные от клиента при регистрации. Поля данного класса отправляются на сервер для регистрации нового клиента.

Resources.

Класс, реализующий паттерн проектирования Singleton. Единственный экземпляр этого класса содержит информацию о всех ресурсах, записанных в конфигурационном файле и необходимых в других компонентах системы.

ConfigReader.

Объект этого класса позволяет считывать конфигурационный файл, сформированный по правилу «ресурс = значение».

`bool isKey(const std::string& s) const`. Метод проверяет наличие ключа *s* в конфигурационном файле.

Logger.

Еще один класс, реализующий паттерн проектирования Singleton. Позволяет вести логирование работы системы. Логируемые сообщения имеют один из четырех уровней, определенных в перечислении LogLevel: ERROR, WARNING, INFO, DEBUG.

Метод `Logger::get(LogLevel level)` возвращает единственный экземпляр класса Logger и задает уровень логирования.

Для объектов класса переопределен оператор «», что позволяет использовать класс следующим образом:

Logger::get(LogLevel::INFO) « System started « std::endl;

Метод `std::string Logger::getTimeForFileName()` позволяет получить время, используемое в имени файла логов.

Метод `std::string Logger::getTimeForFileName()` возвращает время, вставляемое перед каждым сообщением при логировании. Пример файла с логами представлен в листинге 3

5.2. Проектирование и разработка Wi-Fi сканера

В список задач Wi-Fi сканера входят:

- мониторинг радиоканала на частоте 2.4 ГГц;
- определение новых устройств в радиусе досягаемости Wi-Fi модуля;
- регистрация MAC-адреса устройства на сервере таргетированной рекламы.

5.2.1. Проектирование архитектуры

Архитектура Wi-Fi сканера состоит из двух модулей: сканера Wi-Fi сети и модуля регистрации посещений. На рис.5.3 представлена архитектура Wi-Fi сканера.

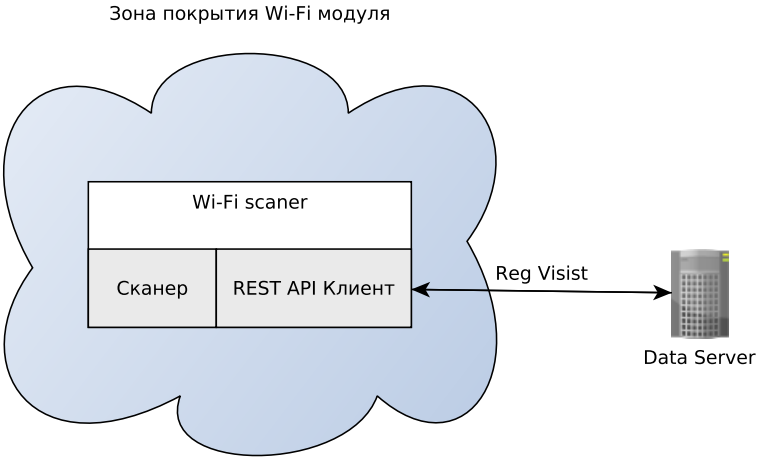


Рис.5.3. Архитектура Wi-Fi сканера

Сканер осуществляет мониторинг пакетов стандарта 802.11. При выявлении пакетов от нового клиента, MAC-адрес устройства отправителя сохраняется в память для дальнейшей обработки.

REST API клиент регистрирует новые устройства на сервере таргетированной рекламы.

5.2.2. Разработка компонента

Wi-Fi монитор, как и компонент регистрации должен работать на устройстве со слабой производительностью. Из этого следует, что рассматриваемый компонент также следует реализовать на языке C++. Для реализации функциональности монитора решено использовать библиотеку Pcap[5]. Библиотека позволяет задавать необходимые настройка интернет адаптеру и просматривать все полученные им пакеты. Pcap используют в своей работе такие программы как Wireshark и tcpdump. Для успешного захвата всех пакетов Wi-Fi модулем, необходимо предварительно перевести его в режим монитора. Библиотека Pcap позволяет легко сделать это.

На рис.5.4 представлена диаграмма классов Wi-Fi сканера.

Рассмотрим каждый класс в отдельности

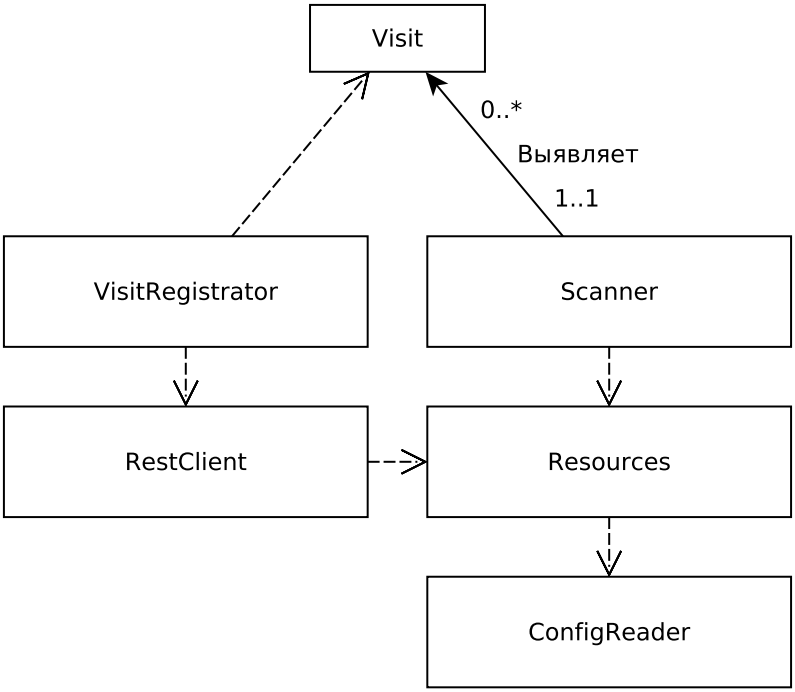


Рис.5.4. Диаграмма классов Wi-Fi сканера

Scanner.

Экземпляр данного класса устанавливает Wi-Fi модуль в режим монитора, что позволяет обрабатывать в программе все услышанные пакеты.

Метод `int Scanner::start()` запускает режим мониторинга трафика и настраивает фильтрующую функцию на прием пакетов только типа Probe Request. В листинге 6 приведен исходный код класса Scaner.

Метод `void Scanner::sniffCallback(u_char arg, const struct pcap_pkthdr pkthdr, const u_char packet)` позволяет осуществлять обработку каждого отловленного пакета. При нахождении нового устройства, во внутренней памяти создается запись типа

Visit с указанием MAC-адреса устройства и временем его появления. Если запись об устройстве уже есть, то обновляется время выхода устройства из зоны действия модуля Wi-Fi.

VisitRegistrator.

Класс осуществляет регистрацию на сервере всех устройств, которые были записаны во внутреннюю память объектом класса Scanner.

Метод `void VisitRegistrator::start()` запускает работу созданного объекта. В методе осуществляется постоянная проверка списка объектов Visit с целью выявить устройство, покинувшее зону мониторинга. Это осуществляется путем проверки времени последнего обновления записи. Если запись обновлялась больше 10 минут назад, то такое посещение регистрируется на сервере и удаляется из памяти.

Visit.

Класс является контейнером для информации о выявленном устройстве. Его полями являются: MAC-адрес устройства, время его появления, и время выхода из зоны мониторинга.

RestClient.

Класс предназначен для регистрации посещений на удаленном сервере с помощью его REST API.

Метод `bool RestClient::sendVisit(Visit visit)` получает обьет типа Visit и с его помощью формирует POST запрос для отправки на сервер таргетированной рекламы.

Классы **Resources** и **ConfigReader** выполняют ту же функцию что и в компоненте регистрации клиентов.

5.3. Проектирование и разработка компонента генерации рекламных событий

5.3.1. Проектирование архитектуры

Архитектура генератора событий спроектирована исходя из выводов, сделанных при рассмотрении типовой структуры экспертных систем. В компоненте должны быть реализованы такие модули как:

- модель представления знаний;
- средство приобретения знаний;
- решатель.

Решение использовать SaaS технологию для работы системы накладывает необходимость разработки веб-интерфейса. Веб-приложение, в свою очередь, является средством приобретения знаний для ЭС. В веб-интерфейсе пользователь имеет возможность сформировать правила, которые используются решателем для получения рекламных событий. Компонента связи с внешним миром для системы является Wi-Fi сканер.

На рис.5.5 представлена архитектура компонента генерации рекламных событий. Архитектура приложения реализована по шаблону MVC, где представлением является веб-интерфейс, моделью — база данных, а контроллером — REST API.

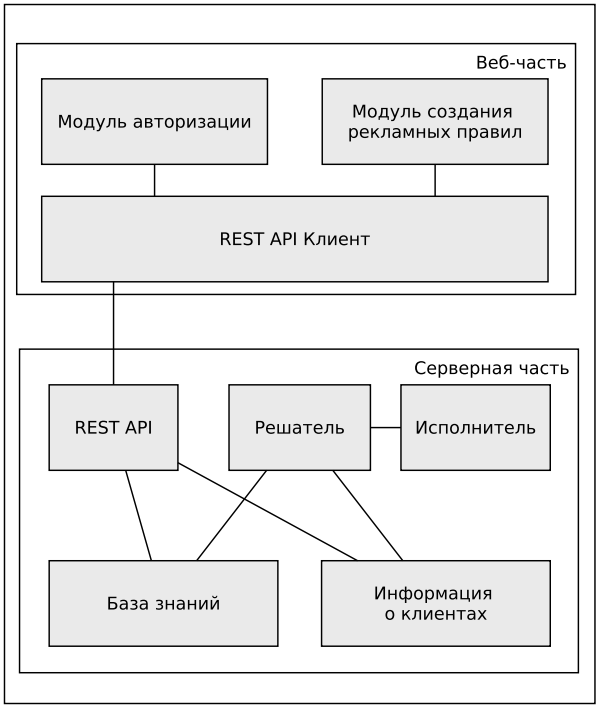


Рис.5.5. Архитектура генератора рекламных событий

5.3.2. Разработка компонента

Для разработки серверной части компонента решено использовать язык программирования Java и фреймворк Spring. Использование библиотеки Spring Boost значительно ускоряет этап настройки и развертывания, что делает разработку более быстрой. Одна из функций, реализованных в Spring — поддержка спецификации JPA. JPA позволяет сохранять в удобном виде java-объекты в базе данных(БД). При этом программисту не нужно задумываться о синхронизации объектов и базы данных. Это делает работу со слоем хранения программы удобной и исключает возможные ошибки при написании собственных компонентов взаимодействия с БД. В качестве реализации JPA интерфейса выбрана библиотека Hibernate. Spring фреймворк позволяет быстро настроить и использовать эту библиотеку.

Для создания веб-приложения решено использовать фреймворк Angular пятой версии. Он позволяет организовывать структуру веб приложения в компоненты, что делает ее более понятной и структурированной. Фреймворк подразумевает использование языка TypeScript, который исключает некоторые недочеты JavaScript, такие как динамическая типизация и отсутствие модульности.

Серверная часть компонента генерации рекламных событий.

Одним из важных компонентов серверной части является решатель, позволяющий генерировать рекламные события. Его работа независима и заключается в периодической фильтрации клиентской базы данных с использованием имеющихся в базе знаний правил. Исходный код решателя представлен в листинге 8.

Правила представлены в виде набора параметров, которыми может обладать клиент и специального типа, который позволяет фильтровать посещения клиентов. Правила бывают трех типов:

1. моментальное — выполняется во время появления клиента в области действия Wi-Fi сканера;

2. по дням недели — выполняется, если клиент посещает определенный набор мест в выбранные дни недели N недель подряд;

3. по последним дням — выполняется, если клиент посетил определенный набор мест последние N дней.

В серверной части реализованы два модуля паттерна MVC — модель и контроллер. Модель реализована с использованием библиотеки Hibernate, что, в свою очередь, выделяет в ней два блока: блок сущностей и блок реализации JPA интерфейса. Созданные в Java коде сущности преобразуются библиотекой в базу данных. Структура базы данных представлена на рис.5.6.

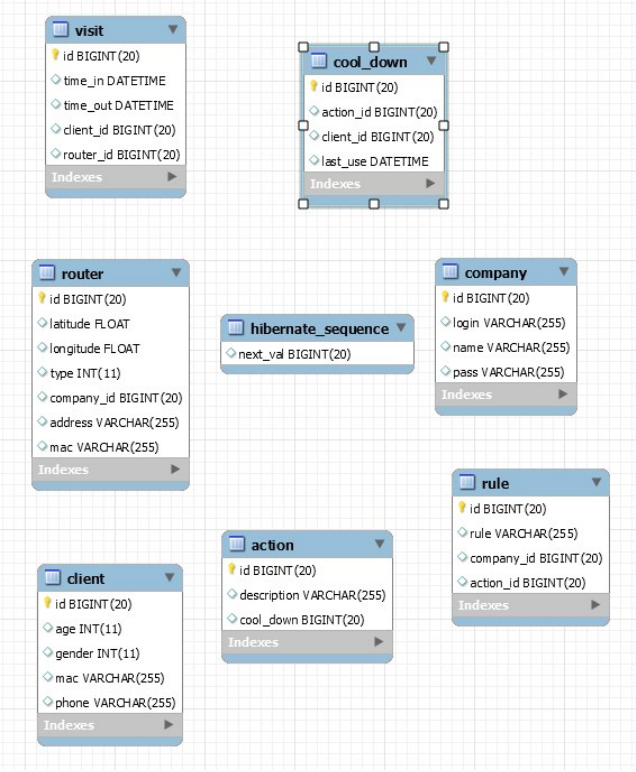


Рис.5.6. Структура базы данных генератора рекламных событий

Рассмотри каждую сущность более подробно.

Company — компания потребитель, которой принадлежат определенные компоненты регистрации и Wi-Fi сканеры.

Client — клиент системы, зарегистрировавшийся в одной из открытых Wi-Fi сетей. Сущность хранит информацию о номере телефо-

на, MAC-адресе устройства, возрасте и поле клиента.

Router — этой сущностью моет быть как Wi-Fi сканер так и компонент регистрации новых клиентов. Разграничение происходит по внутреннему полю type. Хранит информацию о своем географическом положении и компании владельце.

Rule — правило, сохраненное в виде набора параметров, по которому будет выявлен список целей для применения рекламного события.

Visit — запись о посещении клиентом определенного Wi-Fi сканера с указанием времени появления и убытия.

Action — объект, являющийся указанием к выполнению определенного рекламного действия. В рамках выполняемой задачи этот объект является SMS сообщением.

Контроллер позволяет совершать манипуляции с данными используя REST API интерфейс. Рассмотрим наиболее интересные контроллеры и их функции.

ClientController позволяет взаимодействовать с сущностью клиента. В его функции входит выполнение таких задач как создание нового клиента и проверка на наличие в базе уже существующего.

CompanyController взаимодействует с сущностью компании и позволяет создавать компанию и извлекать компанию по ее логину.

VisitController позволяет создавать посещения и просматривать имеющиеся.

Веб-часть компонента генерации рекламных событий.

Для веб части реализованы два модуля. Первый модуль отвечает за авторизацию представителя компании-потребителя в системе и регистрацию новых пользователей. Второй компонент предоставляет пользователям возможность добавлять в систему новые правила.

5.4. Резюме

В результате процесса проектирования и разработки реализованы следующие компоненты: Wi-Fi сканер, компонент регистрации новых клиентов и компонент генерации рекламных событий с интерфейсом для формирования рекламных правил. Следующим и завершающим станет этап тестирования разработанных компонентов.

6. ТЕСТИРОВАНИЕ, АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

Тестирование — важная часть разработки программного обеспечения, помогающая разработчику создавать работоспособное приложение. В данном разделе будет осуществлено функциональное тестирование компонентов системы. Функциональное тестирование позволяет рассмотреть варианты использования приложения и выявить недочеты в программе.

6.1. Тестирование компонента регистрации новых клиентов

Тестирование проводилось с использованием мобильных телефонов Sony Ericsson Xperia Neo V и HTC One M9. При подключении к открытой Wi-Fi сети появляется уведомление о необходимости авторизации. После нажатия на всплывающее уведомление в списке событий, открывается окно регистрации сервера Captive Portal. Окно регистрации представлено на рис.6.1

В окне регистрации появляется возможность ввести данные о номере телефона, возрасте и поле. Страница позволяет загрузить и прочитать пользовательское соглашение. В конце веб-страницы пользователь предупреждается о согласии с соглашением, если тот нажмет на кнопку регистрации.

При успешной регистрации в браузере смартфона открывается страница поиска Google. Результат перенаправления представлен на рис.6.2. С этого момента информация сохранена в клиентской базе данных, а клиенту предоставлен доступ в интернет. Запись о новом клиенте можно увидеть на рис.6.3.

Для того чтобы эмулировать новое подключение клиента к системе в которой он уже зарегистрирован необходимо перезапустить сервер регистрации, что приведет к сбросу локально сохраненных данных. При совершении нового подключения на устройстве клиента ничего не происходит, а интернет сразу становится доступным.

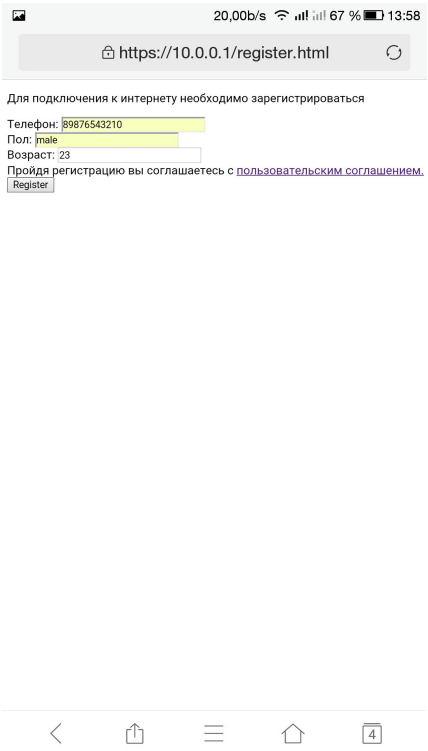


Рис.6.1. Окно регистрации сервера Captive Portal

6.2. Тестирование Wi-Fi сканера

Для того чтобы протестировать Wi-Fi сканер достаточно включить на недолгое время Wi-Fi в зарегистрированном устройстве. После этого запись о его появлении появится в клиентской базе данных. Запись о новом посещении показана на рис.6.4.

Включение других устройств в зоне действия сканера приводит к отправке сообщений на сервер, однако новых записей не появляется, так как эти устройства не зарегистрированны.

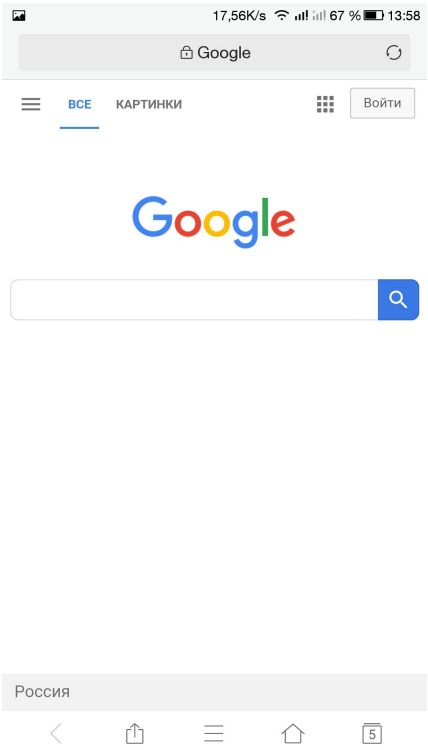
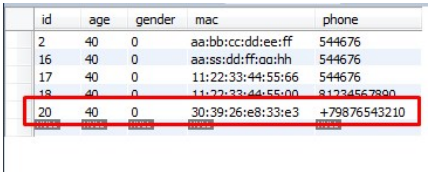


Рис.6.2. Результат успешной регистрации в системе

6.3. Тестирование компонента генерации рекламных событий

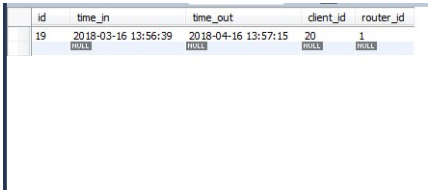
Тестирование этого компонента начинается со страницы логирования. На рис.6.5 показана ситуация введения пользователем неверного логина. А на рис.6.6 введение неверного пароля.

Успешно пройдя авторизацию пользователю предоставляется возможность осуществить добавление правила, по которому будут генерироваться рекламные сообщения. Здесь ему необходимо выбрать тип правила, выставить соответствующие поля фильтров и указать рекламное сообщение, которое будет отправлено клиенту. Добавление



id	age	gender	mac	phone
2	40	0	aa:bb:cc:dd:ee:ff	544676
16	40	0	aa:ss:dd:ff:gg:hh	544676
17	40	0	11:22:33:44:55:66	544676
18	40	0	11:22:33:44:55:00	81234567890
20	40	0	30:39:26:e8:33:e3	+79876543210

Рис.6.3. Результат успешной регистрации в системе



id	time_in	time_out	client_id	router_id
19	2018-03-16 13:56:39	2018-04-16 13:57:15	20	1

Рис.6.4. Запись о посещении клиентом зоны действия Wi-Fi сканера

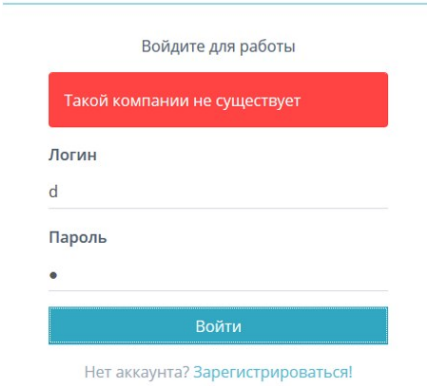


Рис.6.5. Введение неверного логина на странице авторизации

каждого типа правила представлено на рис.6.7, рис.6.8 и рис.6.9.

Увидеть добавленные в базу знаний правила можно на рис.6.10

Генератор рекламных событий работает с заданой частотой в 10 секунд. По истечении этого времени проверются все правила, присутствующие в базе знаний. Если в клиентской базе данных, присут-

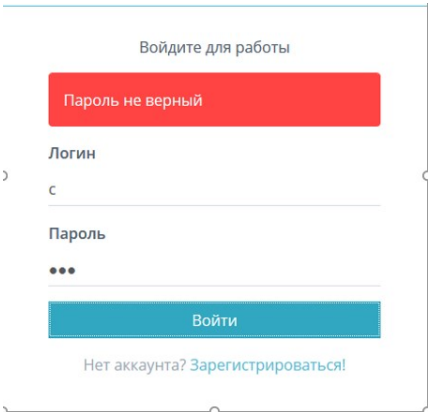


Рис.6.6. Введение неверного пароля на странице авторизации

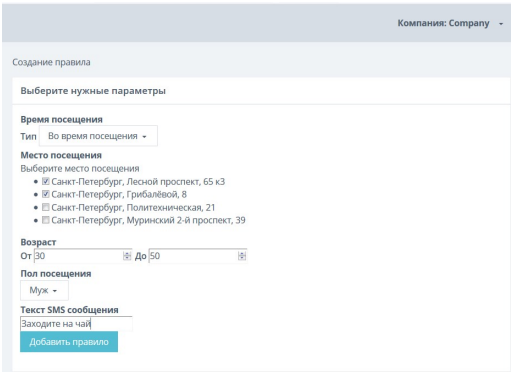


Рис.6.7. Добавление правила, типа: моментальное

ствует информация, соответствующая сформированному правилу, то генерируется событие на отправку SMS сообщения клиенту. В рамках разработанной программы, модуль отправки сообщения не реализован, так как его тестирование повлекло бы за собой значительные денежные затраты. Однако, сообщение о формировании события и текст сообщения, могут быть просмотрены в логе компонента генерации рекламных событий. Сообщения о генерации событий представлены на рис.6.11–6.13. Список событий может быть просмотрен и в графиче-

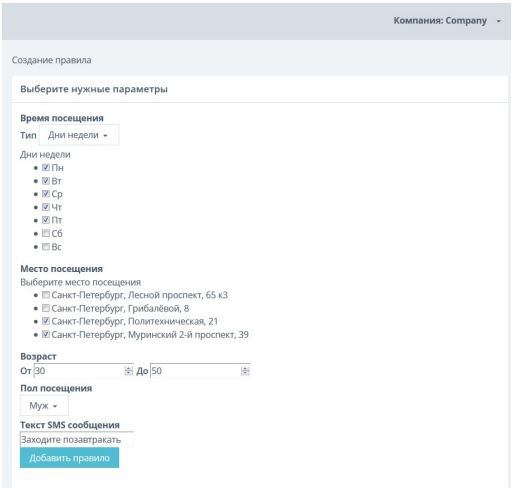


Рис.6.8. Добавление правила, типа: по дням недели

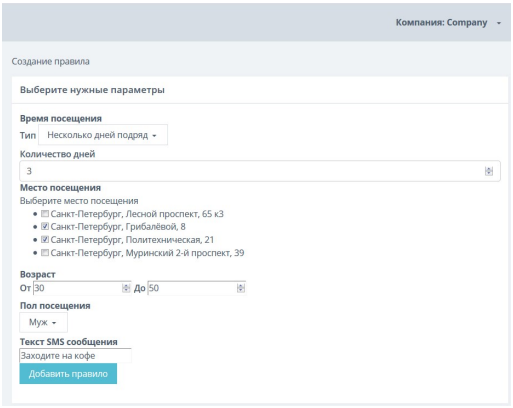
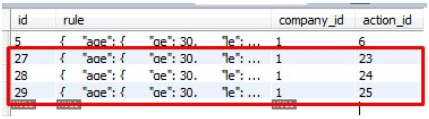


Рис.6.9. Добавление правила, типа: последние n дней

ском интерфейсе пользователя, который представлен на рис.6.14



id	rule	company_id	action_id
5	{ "age": { "oe": 30, "le": ...	1	6
27	{ "age": { "oe": 30, "le": ...	1	23
28	{ "age": { "oe": 30, "le": ...	1	24
29	{ "age": { "oe": 30, "le": ...	1	25

Рис.6.10. Отображение новых правил в базе знаний

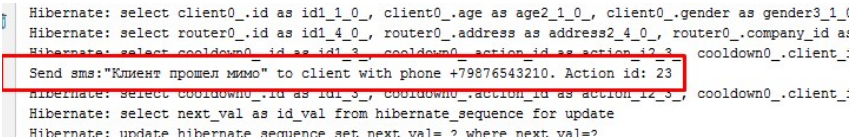


Рис.6.11. Результат тестирования правила типа: моментальное

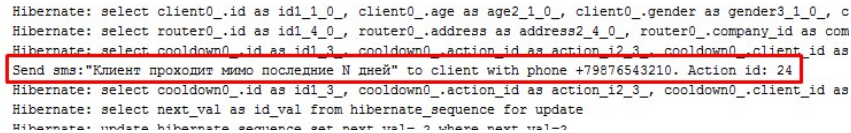


Рис.6.12. Результат тестирования правила типа: по дням недели

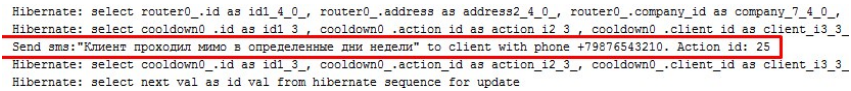


Рис.6.13. Результат тестирования правила типа: по последним дням

6.4. Анализ результатов

В результате тестирования установлено, что в системе решены все поставленные задачи. Осуществлена возможность регистрации клиента. Получен модуль, позволяющий отслеживать появление клиентов в зоне его досягаемости. Разработано приложение для формирования правил таргетирования сообщений. Разработан компонент, позволяющий обрабатывать внесенные правила и на их основе генерировать события о посылке SMS сообщений.

Созданная система может быть использована любой желающей

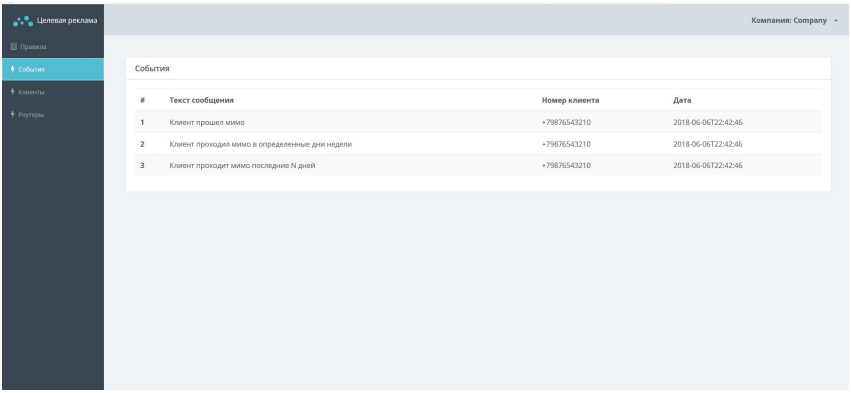


Рис.6.14. Отображение сгенерированных событий в графическом интерфейсе

компанией. Этому способствует реализация главной части системы с использованием облачной архитектуры. Для того чтобы начать использование, пользователю необходимо первым делом зарегистрироваться. Далее следует приобрести оборудование, на которое может быть установлен компонент регистрации новых клиентов и оборудование для установки Wi-Fi сканера. Для разработанных программных компонентов не требуется мощного оборудования и это делает систему более доступной для пользователей. Приобретя аппаратуру, пользователь должен разместить компонент регистрации в своей организации, а Wi-Fi сканеры в наиболее интересных для него местах — например, на улице возле совего офиса. После этого следует зарегистрировать настроенное оборудование в системе. В дальнейшем пользователь может осуществлять добавление рекламных правил и событий.

Недостатком данной системы является тот факт, что Wi-Fi модуль остается включенным в смартфонах не так уж и часто. Обычно его выключают сразу после использования. В дальнейшем можно разработать систему идентификации клиента по каким-либо другим признакам. Например Bluetooth устройство или радио-модуль смартфона.

Еще одним существенным ограничением является тенденция к анонимизации Wi-Fi модуля. Некоторые современные устройства рассылают пакеты Probe Request используя случайный MAC-адрес. Это

делает выявление клиентского устройства более трудным. Однако существуют исследования, утверждающие, что такой метод анонимизации легко обойти, путем создания "слепка"сообщений, рассылаемых смартфоном.

Дальнейшее развитие системы должно быть направлено на расширение функциональности веб-интерфейса и его привлекательность и удобности с точки зрения внешнего вида. Переработка страницы создания правил в полноценный конструктор была бы очень привлекательной для пользователей. Так же система позволяет реализовать новые виды фильтров для создаваемых правил. Возможно и добавление типов правил, связанных с фильтрацией по времени.

ЗАКЛЮЧЕНИЕ

Целевая реклама в будущем может выйти на новый уровень, позволяющий использовать так называемый геоповеденческий таргетинг. Он позволит выбирать целевую аудиторию исходя из информации о посещенных человеком мест. В данной работе реализована система, позволяющая воплотить идею геоповеденческого таргетинга.

В работе выполнено рассмотрение видов таргетированной рекламы и информации, которую они использует. Проанализирован вопрос законности сбора и использования персональных данных в маркетинговых целях. Составлено пользовательское соглашений, принимаемое клиентами во время регистрации.

Рассмотрены виды интеллектуальных систем, их основные компоненты и модели реализации. На основе полученной информации выделен тип ИИ - экспертные системы, как наиболее подходящий в целях формирования рекламных сообщений. С использованием типовой структуры ЭС спроектирован компонент генерации рекламных событий.

В качестве платформы для размещения компонента генерации рекламы выбрана SaaS технология. Предварительно рассмотрены преимущества и недостатки облачных систем. Проанализированы угрозы данным как на сервере облачных технологий, так и в каналах связи. Для защиты данных в каналах связи решено использовать протокол HTTPS.

В результате была разработана система, состоящая из трех компонентов: сервера регистрации новых клиентов для пополнения клиентской базы данных, Wi-Fi сканера, позволяющего получать данные о географических перемещениях клиентов и системы генерации рекламных сообщений. Последний компонент в своем составе так же имеет веб-интерфейс для формирования правил, являющихся базой знаний для генератора событий.

Проведенное тестирование показало работоспособность каждого компонента в отдельности и системы в целом. Дальнейшее развитие может быть осуществлено как в сторону повышения качества и удобства графического интерфейса, так и в сторону увеличения функциональных возможностей.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin.

Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. 802.11-2016 - IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications // IEEE Std. — 2016.
2. Волков В.А. АНАЛИЗ УГРОЗ И МЕТОДОВ ЗАЩИТЫ ОБЛАЧНЫХ СЕРВИСОВ // Молодий вчений. — 2015. — № 12-1. — С. 38–43.
3. Временной таргетинг, Яндекс. — URL: https://yandex.ru/support/direct/efficiency/timetargeting.html (дата обращения: 18.05.2018).
4. Выростков Д. Обзор способов и протоколов аутентификации в веб-приложениях. — 2015. — URL: https://habr.com/company/dataart/blog/262817/ (дата обращения: 11.05.2018).
5. Закон «Об авторизации пользователей в WiFi», 210fz. — URL: https://210fz.ru/zakon-ob-avtorizacii-polzovatelej-v-wifi/ (дата обращения: 13.05.2018).
6. Карасев С. eBay просит пользователей поменять пароли в связи с кражей базы данных, 3DNews. — URL: https://3dnews.ru/820608 (дата обращения: 11.05.2018).
7. Лопатников Л.И. Агент экономический/Экономико-математический словарь: Словарь современной экономической науки, 5-е изд., перераб. и доп // М.: Дело. — 2003.
8. OCSP — протокол проверки статуса SSL сертификата, Emaro. — 2015. — URL: https://www.boost.org/doc/libs/1_66_0/doc/html/boost_asio.html (дата обращения: 12.05.2018).
9. Селиванов В. 5 мифов о персональных данных. — 2015. — URL: https://habr.com/company/it-lex/blog/337354/ (дата обращения: 13.05.2018).
10. Семенов А.М. Соловьев Н.А. Чернопрудова Е.Н. Цыганков А.С. Интеллектуальные системы. — ЛитРес, 2013.
11. Целевая реклама в интернете, ЗЕКСЛЕР. — URL: http://zexler.ru/usefull/celevaya-reklama-v-internete (дата обра-

щения: 18.05.2018).
12. amoCRM, amoCRM. — URL: http://www.amocrm.ru/ (дата обращения: 10.05.2018).
13. Amoroso Edward G. From the enterprise perimeter to a mobility-enabled secure cloud // IEEE Security & Privacy. — 2013. — Т. 11, № 1. — С. 23–31.
14. Data Encryption in SQL Server, Microsoft. — URL: https://docs.microsoft.com/ru-ru/dotnet/framework/data/adonet/sql/data-encryption-in-sql-server (дата обращения: 11.05.2018).
15. HElib Documentation, HElib. — URL: https://shaih.github.io/HElib/ (дата обращения: 11.05.2018).
16. Khalil Issa M, Khreishah Abdallah, Azeem Muhammad. Cloud computing security: A survey // Computers. — 2014. — Т. 3, № 1. — С. 1–35.
17. Kohlhoff Christopher. Boost.Asio. — URL: https://www.boost.org/doc/libs/1_66_0/doc/html/boost_asio.html (дата обращения: 15.05.2018).
18. Kumar Subodha. Evolution of web advertising // Optimization Issues in Web and Mobile Advertising. — Springer, 2016. — С. 1–7.
19. Malinen Jouni. hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator. — 2013. — URL: https://w1.fi/hostapd/ (дата обращения: 15.05.2018).
20. MIT License. — URL: https://github.com/eidheim/Simple-Web-Server/blob/master/LICENSE (дата обращения: 15.05.2018).
21. An Overview of TLS 1.3 – Faster and More Secure, Kinsta. — URL: https://kinsta.com/blog/tls-1-3/ (дата обращения: 11.05.2018).
22. An overview of various authentication methods and protocols / Dwiti Pandya, Khushboo Ram Narayan, Sneha Thakkar и др. // International Journal of Computer Applications. — 2015. — Т. 131, № 9. — С. 25–27.
23. Sareen Pankaj. Cloud computing: types, architecture, applications, concerns, virtualization and role of it governance in cloud // International Journal of Advanced Research in Computer Science and Software Engineering. — 2013. — Т. 3, № 3.
24. Simple-Web-Server. — URL: https://github.com/eidheim/Simple-Web-Server (дата обращения: 15.05.2018).

25. Top 10-2017 Top 10, OWASP. — URL: https://www.owasp.org/index.php/Top_10-2017_Top_10 (дата обращения: 11.05.2018).
26. VtigerCRM, VtigerCRM. — URL: https://www.vtiger.com/ (дата обращения: 10.05.2018).
27. Wang Jun, Zhang Weinan, Yuan Shuai. Display advertising with real-time bidding (RTB) and behavioural targeting // arXiv preprint arXiv:1610.03013. — 2016.

Приложение

ЛИСТИНГИ

Листинг 1. Конфигурационный файл hostapd

```
interface=wlan1
bssid=66:66:66:66:66:66
driver=nl80211
ssid=CaptiveSpot
channel=6
hw_mode=g
beacon_int=100
ieee80211n=1
disassoc_low_ack=0
ap_max_inactivity=3000
auth_algs=3
logger_syslog=-1
logger_stdout=-1
logger_syslog_level=2
logger_stdout_level=2
ctrl_interface=/var/run/hostapd
ctrl_interface_group=0%
```

Листинг 2. Конфигурационный файл dnsmasq

```
dhcp-range=10.42.0.100,10.42.0.254,1h
dhcp-option=6,10.42.0.1 #DNS
dhcp-option=3,10.42.0.1 #Gateway
dhcp-authoritative
log-queries

# запрет считывать адреса DNS-серверов с файла resolv.conf
no-resolv

# запрещает считывать доменные имена из файла /etc/hosts
# no-hosts

# отключение отслеживание изменения файла /etc/resolv.conf или друго
    го файла выполняющего его функцию
no-poll

address=/localcaptive/10.42.0.1
server=8.8.8.8

# Для защиты от DNS атак необходимо запретить ответы от вышестоящих
    DNS серверов с IP адресами компьютеров локальной сети:
stop-dns-rebind


# очистка DNS-кэша при перезапуске сервиса
clear-on-reload
```

Листинг 3. Пример логов компонента регистрации

```
23:35:41 [DEBUG]: ServerBase->write_responce
23:35:41 [DEBUG]: Connection->set_timeout
23:35:41 [INFO]:  Get unknown :10.0.0.1//favicon.ico
```

```
23:35:41 [INFO]:  Returned file: getHttpsServer.html23:35:41 [DEBUG
    ]: Responce->write 7
23:35:41 [DEBUG]: Responce->write 3
```

Листинг 4. Скрипт настройки межсетевого экрана

```
#!/bin/sh
iptables -t mangle -F
iptables -t filter -F
iptables -t nat -F
iptables -t raw -F


iptables -A INPUT -i lo -j ACCEPT

# Настройка wifi

# Создаем цепочку internet
# Новые клиенты будут перенаправляться на 80 порт
# Маркированные пакеты не авторизованных клиентов будут отбрасыватьс
    я
iptables -N internet -t mangle

iptables -t mangle -A PREROUTING -i wlan1 -j internet

#маркируем все пакеты
iptables -t mangle -A internet -j MARK --set-mark 99

#TODO получить ip автоматически.
# все маркированные пакеты которые идут на 80 порт отправляем на наш
    сервер
iptables -t nat -A PREROUTING -m mark --mark 99 -p tcp --dport 80 -j
    DNAT --to-destination 10.0.0.1
iptables -t nat -A PREROUTING -m mark --mark 99 -p tcp --dport 443 -
    j DNAT --to-destination 10.0.0.1


#Разрешаем получение ответов на запросы с сервера
iptables -t filter -A INPUT -m conntrack --ctstate ESTABLISHED,
    RELATED -j ACCEPT

#Разрешаем доступ к веб серверу, серверам проверки сертификата и днс
#Запрещаем неизвестные/неожидаемые пакеты, направленные серверу

iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 8401 -j ACCEPT
iptables -t filter -A INPUT -p udp --dport 67 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 67 -j ACCEPT
iptables -t filter -A INPUT -p udp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -m mark --mark 99 -j DROP


# Enable Internet connection sharing
echo "1" > /proc/sys/net/ipv4/ip_forward

```

```
47 #Запрещаем все маркеровнные пакеты
48 iptables -t filter -A FORWARD -m mark --mark 99 -j DROP
49
50 iptables -t nat -A POSTROUTING -s 10.0.0.0/24 ! -d 10.0.0.0/24 -j
       MASQUERADE
```

Листинг 5. Исходный код метода defaultGet класса CaptiveServer

```
1  void CaptiveServer::defaultGet(std::sharedptr<typename SimpleWeb::
       Server<sockettype>::Response> response,
2                                 std::sharedptr<typename SimpleWeb::
                                      Server<sockettype>::Request>
                                      request) {
3      try {
4
5          RestClient rc;
6          if (rc.isClientExist(Firewall::getInstance().macbyip(
               response->getIp()))) {
7
8              std::string redirection = "https://";
9              redirection += mredirectionSite;
10             std::cout << "Redirection to: " << redirection << std::
                   endl;
11             SimpleWeb::CaseInsensitiveMultimap header;
12             header.emplace("Content-Length", std::tostring(0));
13             header.emplace("Access-Control-Allow-Origin", "*");
14             header.emplace("Access-Control-Allow-Methods", "*");
15             header.emplace("Access-Control-Allow-Headers", "Origin,
                   X-Requested-With, Content-Type, Accept");
16             header.emplace("Location", redirection);
17
18             Firewall::getInstance().passmac(response->getIp());
19
20             response->write(SimpleWeb::StatusCode::
                   redirectionseeother, header);
21
22         } else {
23
24             auto webrootpath = boost::filesystem::canonical(
                   mpathToResources + "/" + mpageFolder);
25             auto relativePath = request->path;
26
27             std::string host = "";
28             auto it = request->header.find("Host");
29
30             if (it == request->header.end())
31                 Logger::get(LogLevel::INFO) << "Get :/" <<
                       relativePath << std::endl;
32             else {
33                 Logger::get(LogLevel::INFO) << "Get unknown :" << it
                       ->second << "/" << relativePath << std::endl;
34             }
35
36             //The address not exists or is a directory
37             if (!boost::filesystem::exists(webrootpath /
                   relativePath)
```

```
38                  || boost::filesystem::isdirectory(webrootpath /
                        relativePath)) {
39
40                  relativePath = mfirstPage;
41              }
42
43              auto path = boost::filesystem::canonical(webrootpath /
                    relativePath);
44
45              // Check if path is within webrootpath
46              if (distance(webrootpath.begin(), webrootpath.end()) >
                    distance(path.begin(), path.end()) ||
47                  !equal(webrootpath.begin(), webrootpath.end(), path.
                        begin()))
48                  throw invalidargument("path must be within root path
                        ");
49
50              Logger::get(LogLevel::INFO) << "Returned file: " <<
                    relativePath;
51
52              SimpleWeb::CaseInsensitiveMultimap header;
53
54              auto ifs = makeshared<ifstream>();
55              ifs->open(path.string(), ifstream::in | ios::binary |
                    ios::ate);
56
57              if (*ifs) {
58                  auto length = ifs->tellg();
59                  ifs->seekg(0, ios::beg);
60
61                  header.emplace("Content-Length", tostring(length));
62                  header.emplace("Access-Control-Allow-Origin", "*");
63                  header.emplace("Access-Control-Allow-Methods", "*");
64                  header.emplace("Access-Control-Allow-Headers", "
                        Origin, X-Requested-With, Content-Type, Accept"
                        );
65
66
67                  response->write(header);
68
69                  // Trick to define a recursive function within this scope (for
                        example purposes)
70                  bool res = FileServer::readandsend(response, ifs);
71
72              } else
73                  throw invalidargument("could not read file");
74          }
75      } catch (const exception &e) {
76          Logger::get(LogLevel::ERROR) << "Http exception: " << e.what
                () << std::endl;
77          response->write(SimpleWeb::StatusCode::clienterrorbadrequest
                ,
78                          "Could not open path " + request->path + ":
                              " + e.what());
79      }
80  }
```

Листинг 6. Исходный код класса Scaner

```
1  //
2  // Created by root on 25.03.18.
3  //
4
5  #include "Scanner.h"
6  #include "System.h"
7  #include "RestClient.h"
8  #include <pcap.h>
9  #include <iostream>
10
11 struct ieee80211_radiotap_hdr {
12     uint8_t it_version;
13     uint8_t it_pad;
14     uint16_t it_len;
15     uint32_t it_present;
16 } ieee80211_radiotap_hdr;
17
18 struct ieee80211_probe_req {
19     uint16_t frame_control;
20     uint16_t duration_id;
21     uint8_t addr1[6];
22     uint8_t addr2[6];
23     uint8_t addr3[6];
24     uint16_t seq_ctrl;
25 } ieee80211_probe_req;
26
27 tbb::concurrent_unordered_map <std::string, Visit> Scanner::visits;
28
29 std::mutex Scanner::visitsMutex;
30
31 Scanner::Scanner() {
32
33
34 }
35
36 Scanner::~Scanner() {
37     for (auto a : visits) {
38         std::cout << a.first << ": " << a.second.get_time_in() <<
              std::endl;
39     }
40 }
41
42 int Scanner::start() {
43     char *dev, errbuf[PCAP_ERRBUF_SIZE];
44
45     dev = pcap_lookupdev(errbuf);
46     if (dev == NULL) {
47         fprintf(stderr, "Couldn't find default device: %s\n", errbuf
              );
48         return (2);
49     }
50
51     dev = "wlan1";
52     printf("Device: %s\n", dev);
53
54     /* open device for reading in no promiscuous mode */
```

```
55
56      pcap_t *descr;
57      descr = pcap_open_live(dev, BUFSIZ, 0, -1, errbuf);
58      if (descr == NULL) {
59          printf("pcap_open_live(): %s\n", errbuf);
60          exit(1);
61      }
62
63      struct bpf_program fp;          /* hold compiled program */
64      char *filter = "type mgt subtype probe-req"; //and ether host
            30:39:26:e8:33:e3
65      /* Now we'll compile the filter expression*/
66      if (pcap_compile(descr, &fp, filter, 0, PCAP_NETMASK_UNKNOWN) ==
            -1) {
67          fprintf(stderr, "Error calling pcap_compile\n");
68          exit(1);
69      }
70
71      /* set the filter */
72      if (pcap_setfilter(descr, &fp) == -1) {
73          fprintf(stderr, "Error setting filter\n");
74          exit(1);
75      }
76
77      /* loop for callback function */
78      pcap_loop(descr, -1, &Scanner::sniffCallback, NULL);
79
80      return 0;
81  }
82
83  void Scanner::sniffCallback(u_char *arg, const struct pcap_pkthdr *
        pkthdr,
84                              const u_char *packet) {
85
86      struct ieee80211_radiotap_hdr *rhdr = (struct
            ieee80211_radiotap_hdr *) (packet);
87      struct ieee80211_probe_req *prhdr = (struct ieee80211_probe_req
            *) (packet + rhdr->it_len);
88
89      char mac[18];
90      int n = sprintf(mac, "%.2x:%.2x:%.2x:%.2x:%.2x:%.2x ", prhdr->
            addr2[0], prhdr->addr2[1], prhdr->addr2[2],
91                      prhdr->addr2[3], prhdr->addr2[4], prhdr->addr2
                            [5]);
92
93      std::string s_mac = std::string(mac);
94      printf("Source m_mac: %s\n", mac);
95
96
97  // Visit visit(std::string(mac), System::getTime());
98
99      auto time = System::getTime();
100
101
102     visitsMutex.lock();
103     auto v = visits.find(s_mac);
104     if (v == visits.end()) {
105         visits.insert(std::pair<std::string, Visit>(s_mac, Visit(
```

```
            s_mac, time)));
106     } else {
107         v->second.setTime_out(time);
108     }
109     visitsMutex.unlock();
110 }
```

Листинг 7. Текст пользовательского соглашения

```
1   Согласие на обработку, передачу третьим лицам персональных данных и
        получение рекламно-информационных сообщений
2
3   Настоящим я, во исполнение требований Федерального закона от
        27.07.2006 г. N 152-ФЗ <<О персональных данных>> (с изменениями
         и дополнениями), Федерального закона от 13.03.2006 N 38-ФЗ <<О
         рекламе>> с изменениями и дополнениями) свободно, своей волей
        и в своем интересе даю свое согласие Компании на обработку свои
        х персональных данных, указанных при регистрации путем заполнен
        ия веб-формы на сайте.
4
5   Оператор обрабатывает Ваши следующие персональные данные: номер моби
        льного телефона, возраст и пол.
6
7   1. Я согласен с тем, что в рамках обработки персональных данных Обще
        ство, а также Операторы, поименованные выше, вправе осуществлят
        ь сбор, запись, систематизацию, накопление, анализ, использован
        ие, извлечение, распространение, передачу Операторам и / или лю
        бым иным третьим лицам (включая, но не ограничиваясь: страховым
         организациям; организациям владельцам-серверов; банковским и и
        ным кредитным организациям; организациям, оказывающим услуги по
         осуществлению звонков, смс - рассылок, любых иных видов рассыл
        ок и уведомлений; организациям, оказывающим услуги по проведени
        ю различных опросов и исследований и пр.), получение, обработку
        , хранение, уточнение (обновление, изменение), обезличивание, б
        локирование, удаление, уничтожение моих персональных данных пут
        ем ведения баз данных автоматизированным, механическим, ручным
        способами в целях:
8   1) ведения и актуализации клиентской базы;
9   2) получения и исследования статистических данных об объемах продаж
        и качестве оказываемых услуг;
10  3) проведения маркетинговых программ;
11  4) проведению опросов и исследований, направленных на выявление удов
        летворенности/неудовлетворенности Пользователя, постоянного сов
        ершенствования уровня предоставляемых услуг;
12  5) рекламирования и иного любого продвижения товаров и услуг на рынк
        е путем осуществления прямых контактов со мной и иными потребит
        елями;
13  6) технической поддержки при обработке информации, документации и пе
        рсональных данных с использованием средств автоматизации и без
        такого использования.
14
15  2. Я выражаю согласие на получение рекламы и разрешаю Обществу и Опе
        раторам осуществлять в мой адрес смс-рассылки, а также иные вид
        ы рассылок и уведомлений, в том числе рекламного характера, с и
        спользованием любых средств связи.
16
```

```
3. Я выражаю согласие и разрешаю Обществу и Операторам объединять пе
    рсональные данные в информационную систему персональных данных
    и обрабатывать мои персональные данные, с помощью средств автом
    атизации либо без использования средств автоматизации, а также
    с помощью иных программных средств, специально разработанных по
     поручению Общества и Операторов. Работа с информационными сист
    емами персональных данных осуществляется по предписанному Общес
    твом и/или Операторами алгоритму (сбор, систематизация, накопле
    ние, хранение, уточнение, использование, блокирование, уничтоже
    ние, др.).

4. Я ознакомлен(а), что :
1) настоящее согласие на обработку моих персональных данных и получе
    ние рекламы является бессрочным и может быть отозвано посредств
    ом направления в адрес Общества и Операторов, указанных в пункт
    е 1 настоящего Согласия, письменного заявления. Датой отзыва сч
    итается день, следующий за днём вручения Обществу и Операторам
    письменного заявления об отзыве согласия на обработку персональ
    ных данных и/или получение рекламы;
2) имею право на доступ к своим персональным данным, требовать уточн
    ения (обновление, изменение) моих персональных данных, а также
    удаления и уничтожения моих персональных данных в случае их обр
    аботки Обществом и вышеуказанными Операторами, нарушающих мои з
    аконные права и интересы, законодательство Российской Федерации
    .

5. Настоящим Согласием я подтверждаю, что являюсь субъектом предоста
    вляемых персональных данных, а также подтверждаю достоверность
    предоставляемых данных.
```

Листинг 8. Исходный код решателя

```
package server.service;

import java.lang.reflect.Type;
import java.time.Clock;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.temporal.ChronoUnit;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;
import java.util.concurrent.TimeUnit;

import com.google.gson.*;
import com.google.gson.reflect.TypeToken;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.scheduling.annotation.Async;
import org.springframework.stereotype.Service;
import server.dao.*;
import server.entity.*;

enum RepeatabilityType {
    once, daysOfWeek, lastDays
}
```

```
26
27  @Service
28  public class AsyncServices {
29
30      //Время в секундах между проверками
31      int CHECK_REPETETION_TIME = 10;
32
33      @Autowired
34      RuleDao ruleDao;
35
36      @Autowired
37      ClientDao clientDao;
38
39      @Autowired
40      CoolDownDao coolDownDao;
41
42      @Autowired
43      VisitDao visitDao;
44
45      @Autowired
46      RouterDao routerDao;
47
48      Logger log = LoggerFactory.getLogger(this.getClass().getName());
49
50      @Async
51      public void process() throws InterruptedException {
52          try {
53              while (true) {
54                  List<Rule> rules = ruleDao.findAll();
55
56                  if (rules.isEmpty()) return;
57
58                  for (Rule r : rules) {
59                      checkRule(r);
60                  }
61
62                  TimeUnit.SECONDS.sleep(CHECK_REPETETION_TIME);
63              }
64          } catch (InterruptedException e) {
65              System.out.println("Exception catched. Stack trace:");
66              e.printStackTrace();
67          }
68      }
69
70      private void checkRule(Rule rule) {
71          JsonParser parser = new JsonParser();
72          JsonObject mainObject = parser.parse(rule.getRule()).
                getAsJsonObject();
73
74          Long routerId = -1L;
75          int lowerBound;
76          int upperBound;
77          Gender gender;
78          RepeatabilityType repeatabilityType;
79
80          //Извлечение простых параметров
81          lowerBound = mainObject.getAsJsonObject("age").
                getAsJsonPrimitive("ge").getAsInt();
```

```
82          upperBound = mainObject.getAsJsonObject("age").
                getAsJsonPrimitive("le").getAsInt();
83          gender = Gender.valueOf(mainObject.getAsJsonPrimitive("
                gender").getAsString());
84
85          JsonObject repeatability = mainObject.getAsJsonObject("
                repeatability");
86          repeatabilityType = RepeatabilityType.valueOf(repeatability
87                  .getAsJsonPrimitive("type")
88                  .getAsString());
89
90          //Извлечение роутеров
91          ArrayList<Router> routersList = new ArrayList<>();
92          if (mainObject.has("router")) {
93              JsonElement jElement = mainObject.get("router");
94              Type listType = new TypeToken<List<Long>>() {}.getType()
                        ;
95              List<Long> routersIdList = new Gson().fromJson(jElement,
                        listType);
96              for (Long l : routersIdList) {
97                  Optional<Router> r = routerDao.findById(l);
98                  if (r.isPresent()) {
99                      routersList.add(r.get());
100                 }
101             }
102         }
103         if (routersList.isEmpty()) {
104             return;
105         }
106
107         List<Client> clients = clientDao.
                findAllByAgeBetweenAndGenderEquals(lowerBound,
                upperBound, gender);
108
109         if (clients.isEmpty()) return;
110
111         for (Client cl : clients) {
112             List<Visit> visits = getVisitsByClientAndRoutars(cl,
                        routersList);
113             if (visits.isEmpty()){
114                 return;
115             }
116
117             switch (repeatabilityType) {
118                 case once:
119
120                     if (checkOnce(cl, visits)) {
121                         doAction(cl, rule.getAction());
122                     }
123                     break;
124                 case daysOfWeek:
125                     if (checkDaysOfWeek(cl,
126                             repeatability.getAsJsonPrimitive("
                                    daysOfWeekMask").getAsString(),
127                             visits)) {
128                         doAction(cl, rule.getAction());
129                     }
130                     break;
```

```
131                 case lastDays:
132                     if (checkLastDays(cl,
133                             repeatability.getAsJsonPrimitive("
                                    repetitions").getAsInt(),
134                             LocalDate.now(),
135                             visits)) {
136                         doAction(cl, rule.getAction());
137                     }
138                     break;
139                 default:
140                     System.out.println("Exception: Unexpected
                            RepeatabilityType:" + repeatabilityType.
                            toString());
141                     break;
142             }
143         }
144     }
145
146     //Проверяет подходит ли клиент под роутер
147     private boolean checkOnce(Client client, List<Visit> visits){
148         if (visits.isEmpty())
149             return false;
150
151         for (Visit v : visits){
152             LocalDateTime nowTime =LocalDateTime.now(Clock.systemUTC
                        ()).minusSeconds(CHECK_REPETETION_TIME);
153             if (v.getTimeIn().isAfter(nowTime)) {
154                 return true;
155             }
156         }
157         return false;
158     }
159
160     private boolean checkLastDays(Client client, int number,
            LocalDate date, List<Visit> visits) {
161
162         List<LocalDate> dates = new ArrayList<LocalDate>();
163         for (int i = 1; i <= number; i++) {
164             dates.add(date.minusDays(i));
165         }
166
167         return checkDaysInVisits(dates, visits);
168     }
169
170     private boolean checkDaysOfWeek(Client client, String
            daysOfWeekMask, List<Visit> visits) {
171         List<LocalDate> searchDates = weekMaskTodates(daysOfWeekMask
                , LocalDate.now());
172         return checkDaysInVisits(searchDates, visits);
173     }
174
175     private boolean checkDaysInVisits(List<LocalDate> days, List<
            Visit> visits) {
176         boolean accept = true;
177         for (LocalDate date : days) {
178             boolean isPresent = false;
179             for (Visit visit : visits) {
180                 //Время входа
```

```
181                 LocalDate dayIn = visit.getTimeIn().toLocalDate();
182                 if (dayIn.compareTo(date) == 0) {
183                     isPresent = true;
184                     break;
185                 }
186                 //Время выхода
187                 LocalDate dayOut = visit.getTimeIn().toLocalDate();
188                 if (dayOut.compareTo(date) == 0) {
189                     isPresent = true;
190                     break;
191                 }
192             }
193             if (!isPresent) {
194                 accept = false;
195                 break;
196             }
197         }
198 
199         if (accept)
200             return true;
201         return false;
202     }
203 
204     private List<LocalDate> weekMaskTodates(String mask, LocalDate
          date) {
205 
206         List<LocalDate> list = new ArrayList<LocalDate>();
207 
208         if (mask.length() < 7 || mask.length() > 7) {
209             System.out.println("Error when parse dayOfWeekMask: " +
                   mask);
210         }
211 
212         //Начиная со вчерашнего дня
213         for (int i = 1; i <= mask.length(); i++) {
214             LocalDate newDay = date.minusDays(i);
215             int maskId = newDay.getDayOfWeek().getValue();
216             if (mask.charAt(maskId - 1) == '1') {
217                 list.add(newDay);
218             }
219         }
220 
221         return list;
222     }
223 
224     private boolean checkActionCD(Client client, Action action) {
225         Optional<CoolDown> optCd = coolDownDao.
                findByClientIdAndActionId(client.getId(), action.getId
                ());
226 
227         //Если записи еще не существует, то правило ранее не применялось
228         if (!optCd.isPresent())
229             return true;
230 
231         CoolDown cd = optCd.get();
232         long days = ChronoUnit.DAYS.between(cd.getLastUse(),
                LocalDateTime.now());
233         if (days < action.getCoolDown())
```

```
234             return false;
235         else
236             return true;
237     }
238
239     private void doAction(Client client, Action action) {
240         if (!checkActionCD(client, action)) {
241             return;
242         }
243
244         //TODO do action
245         System.out.println("Send sms:\"" + action.getDescription() +
                "\" "
246         + "to client with phone " + client.getPhone()
247         + ". Action id: " + action.getId());
248
249         CoolDown cd = new CoolDown(client.getId(), action.getId(),
                LocalDateTime.now());
250
251         Optional<CoolDown> optCd = coolDownDao.
                findByClientIdAndActionId(client.getId(), action.getId
                ());
252         if (optCd.isPresent()) {
253             cd.setId(optCd.get().getId());
254         }
255         coolDownDao.save(cd);
256     }
257
258     private List<Visit> getVisitsByClientAndRoutars(Client client,
            List<Router> routers) {
259         ArrayList<Visit> visits = new ArrayList<>();
260         for (Visit v: visitDao.findAllByClient(client)){
261             if (routers.contains(v.getRouter())) {
262                 visits.add(v);
263             }
264         }
265         return visits;
266     }
267 }
```