/Министерство образования и науки Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕУЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

ТЕХНОЛОГИИ ДЕЦЕНТРАЛИЗОВАННЫХ ПРИЛОЖЕНИЙ

АВТОРЕФЕРАТ МАГИСТЕРСКОЙ РАБОТЫ

Студента 2 курса 273 группы
направления 01.04.02 Прикладная математика и информатика
факультета компьютерных наук и информационных технологий
Совы Игоря Олеговича

Научный руководитель
старший преподаватель _____ М. С. Портенко

Зав. кафедрой
доцент, к.ф.-м.н. _____ М. В. Огнева

Саратов 2017

ВВЕДЕНИЕ

Актуальность темы. Современные веб-сервисы, нацеленные на массовую аудиторию, представляют собой весьма сложные продукты. Они содержат огромную кодовую базу, требуют внушительных аппаратных ресурсов для работы с возрастающим изо дня в день количеством пользователей. Бесчисленные средства требуются на безопасность, программную и аппаратную поддержку масштабируемости. Однако несмотря на все развитие в сфере веб-технологий, классическая «клиент-серверная» архитектура, на основе которой построена большая часть сервисов в сети Интернет, изначально обладает рядом недостатков[1]:

- Неработоспособность сервера может сделать неработоспособной всю вычислительную сеть;
- Как следствие к предыдущему пункту, высокая стоимость поддержки масштабирования;
- Поддержка работы сервиса требует отдельного специалиста – системного администратора;
- Высокая стоимость серверного оборудования.

Практическая значимость магистерской работы. Созданное децентрализованное средство обмена сообщениями представляет собой защищенное и отказоустойчивое решение, не требующее дополнительных надстроек безопасности, так как в своей архитектуре уже содержит ассиметричное абонентское шифрование, и выгодно отличающется от аналогов в разрезе устойчивости к цензурированию трафика в сети.

Цель магистерской работы – изучение существующих технологий создания децентрализованных технологий, применение их на практике для создания системы обмена сообщениями, построенную на децентрализованных технологиях и удовлетворяющую современным требованиям безопасности и реализующую потенциал децентрализованного технологического стека.

Поставленная цель определила **следующие задачи**:

1. Исследовать существующие решения в области безопасности обмена сообщениями;
2. Изучить архитектуру и возможности децентрализованных систем;
3. Спроектировать децентрализованную систему обмена сообщениями;
4. Разработать концепт системы, предоставляющий необходимый базовый функционал;
5. Сравнить возможности спроектированной системы и существующих систем.

Структура и объём работы. Магистерская работа состоит из введения, 5 разделов, заключения, списка использованных источников и 1 приложения. Общий объем работы – 79 страниц, из них 58 страниц – основное содержание, включая 17 рисунков, список использованных источников информации – 25 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Технологии децентрализованных приложений» посвящен сути децентрализованных приложений.

Основная идея децентрализованных приложений – распределение функциональности сервера по всем клиентам сети [2]. В общем случае, каждый узел сети выполняет серверные функции для всех остальных, одновременно будучи клиентом каждого, как сервера. Таким образом, из сети исчезает сервер, как выделенная единица, что сразу же снимает проблемы, описанные в предыдущем параграфе. Производительность сети становится пропорциональна количеству узлов в ней, критические точки – отсутствуют как таковые, а масштабируемость выходит на новый уровень, где проблемы заключаются не в объеме аппаратных средств и количестве портов сервера, а в архитектурных решениях ПО.

Децентрализованные приложения является крайне устойчивыми: выход из строя одного узла мало сказывается на других и практически не сказывается на функциональности сети в целом. Любой пользователь являет собой самостоятельный узел сети, организованной приложением [3]. В таких условиях в сети существует постоянная миграция узлов: подключение новых, выход старых, внезапные отключения и т.п.

Была рассмотрена классификация существующих решении, рассмотрены Mesh-сети и P2P-сети.

Mesh-сеть – это распределенная, одноранговая, ячеистая сеть, представленная на 3 уровне (сетевом) сетевой модели OSI.

Одним из главных принципов построения mesh-сети является принцип самоорганизации архитектуры, обеспечивающий возможности [4, 5], как:

- реализацию топологии сети "каждый с каждым";
- устойчивость сети при отказе отдельных компонентов;
- масштабируемость сети – увеличение зоны информационного покрытия в режиме самоорганизации;
- динамическую маршрутизацию трафика.

P2P-сеть (peer-to-peer) также строится вокруг понятия равноправных узлов – клиенты и серверы одинаково взаимодействуют с другими узлами сети, таким образом являясь одноранговыми. P2P-сети имеют много общего с mesh-сетями касательно свойств, таких как отказоустойчивость, масштабируемость и динамическая маршрутизация трафика внутри сети, однако при этом данный тип сетей работает на 4 уровне (транспортном) сетевой модели OSI и выше [6].

Второй раздел «Современные технологии обмена сообщениями» посвящен изучению и сравнению существующих средств для обмена сообщениями.

На данный момент существует достаточное число популярных продуктов для обмена сообщениями, часть из них можно считать в некотором роде децентрализованными. В текущем разделе рассмотрены такие популярные решения, как:

- Email + PGP;
- XMPP;
- Telegram.

В итоге можно констатировать, что современные средства обмена сообщениями, представленные выше, едва ли удовлетворяют всем условиям надежности и безопасности и лишь отчасти могут быть надежны, особенно учитывая, что в основном им требуется дополнительная настройка безопасности для полноценного использования, что накладывает некоторую ответственность еще и на пользователя [7, 8, 9, 10]. Важной особенностью является то, что они слабо защищены от возможности цензуры контента как частичной, так и на уровне протокола в целом, при этом у большинства решений существует проблема точки отказа – «бутылочного горлышка».

Третий раздел «Технологии маршрутизации децентрализованных приложений» посвящен изучению одного из важнейших аспектов децентрализованного приложения аспекта – маршрутизации сети. В этом разделе рассмотрены наиболее яркие из вариантов построения систем на основе распределенных хеш-таблиц (DHT).

DHT (Distributed Hash Table – «распределённая хеш-таблица») – децентрализованные распределённые системы, которые обеспечивают поисковый сервис, похожий по принципу работы на хеш-таблицу, структуру данных, широко используемую в языках программирования и именуемую иначе ассоциативным массивом, содержащую пары: (ключ, значение), хранящиеся в DHT, где каждый участвующий узел сети может искать значение, ассоциированное с данным ключом [11]. Ответственность за поддержку связи между именем и значением распределяется между узлами, таким образом изменение набора участников является причиной минимального количества разрывов. Это позволяет легко масштабировать DHT и постоянно отслеживать добавление/удаление узлов и ошибки в их работе.

В данном разделе были также рассмотрены детали работы наиболее ярких представителей DHT-технологий: Chord DHT (метрика хода стрелки часов) [12, 13, 14] и Kademlia (XOR-метрика) [15, 16, 17, 18].

Стоит сказать, что оба варианта рассмотренных решений имеют схожие идеи в основе маршрутизации, несмотря на разные метрики, однако стоит отметить то, что Kademlia имеет ряд преимуществ: простота XOR-метрики для подсчетов, более гибкий механизм управления таблицей маршрутизации и меньшие задержки при осуществлении маршрутизации в силу грамотного кеширования адресации узлов.

Четвертый раздел «Технология Blockchain» посвящен Blockchain, перспективной децентрализованной технологии, которая не является сама по себе технологией маршрутизации, но при этом содержит ряд идей, необходимых для разработки безопасного децентрализованного приложения. Blockchain – своего рода распределенная база данных, хранящая все операции, совершенные узлами сети [2, 3]. Исторически технология довольно тесно связана с криптовалютами.

Blockchain формируется в виде как распределенной непрерывно растущая цепочки блоков с записями о всех транзакциях. Для транзакций в блоке используется древовидное хеширование, аналогичное формированию хеш-

суммы для файла в протоколе BitTorrent [19]. Копия базы или её части одновременно хранится на множестве узлов сети и синхронизируются согласно формальным правилам построения цепочки блоков. Информация в блоках (списки транзакций и конечный баланс узлов) не зашифрована и может быть получена в открытом виде, но защищена от изменений криптографически через хэши, связующие блоки друг с другом [20].

Пятый раздел «Реализация децентрализованного средства обмена сообщениями» посвящен реализации децентрализованного средства обмена сообщениями с веб-интерфейсом. Были сформированы технические требования к системе, среди них можно особенно выделить такие аспекты, как анонимность при передаче сообщений в сети (невозможность вычислить адресата или отправителя сообщения, невозможность прочтения самого сообщения), защита от спама в сети, что в свою очередь накладывает определенные требования на структуру сети.

Для реализации использовался компилируемый строго типизированный язык общего назначения Go от компании Google в силу своей простоты, наличия автоматического управления памятью (сборщик мусора) и быстродействия бинарных файлов, компилируемых Go [21].

Сетевое взаимодействие узлов построено на протоколе TCP. TCPвполне применим в данной реализации системы обмена сообщениями, так как отдельный узел не поддерживает тысяч соединений в один момент времени, соответственно дополнительные расходы по сравнению с протоколом UDP не страшны, но при этом TCPдает гарантию правильного порядка пакетов и надежной их доставки[22].

При децентрализации приложения отпадает всякая необходимость и возможность использовать крупные реляционные базы данных вроде MS SQL, MySQL и других промышленные решения. Поэтому в качестве хранилища данных используется Bolt. Bolt – это простое и быстрое key-value локальное хранилище данных, написанное на Go [23].

В качестве идентификатора пользователя предлагается использовать хеш

публичного ключа пользователя, который также предлагается использовать в качестве адреса для обмена с другими пользователями. Для шифрования сообщений используется схема шифрования с публичным ключом.

Для решения проблемы спама в качестве прототипа использовался механизм поиска SHA-256 хеша с заданным количеством префиксных нулей, с возможностью оптимизации вычисления с помощью OpenCL [24].

Для взаимодействия же программной части на Go с веб-интерфейсом использовался протокол полнодуплексный протокол WebSocket [25].

Стоит отметить, что реализация децентрализованного средства обмена сообщениями в итоге не подвержено таким проблемам рассмотренных аналогов, как:

- Спам в сети,
- Уязвимость к атакам Man-In-The-Middle,
- Наличие критической точки отказа,
- Проблемы масштабируемости.

ЗАКЛЮЧЕНИЕ

В данной работе были рассмотрены принципы создания децентрализованных приложений, описаны существующие технологии разработки децентрализованных приложений. Детально были разобраны наиболее перспективные из них, изучены и выбраны для дальнейшего использования в решении практической задачи, которые, несмотря на большую сложность понимания, несет в себе массу практических возможностей для сетевых приложений и может вывести их на новый уровень. Были изучены современные популярные средства обмена сообщениями с целью выявления их положительных и отрицательных качеств, чтобы в дальнейшем учесть их при создании децентрализованного их аналога.

В качестве практического примера был разработан децентрализованный сервис обмена сообщениями, являющийся актуальной задачей для иллюстрации особенностей децентрализованных приложений.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Архитектура «клиент-сервер» [Электронный ресурс] URL: http://www.4stud.info/networking/lecture5.html (дата обращения: 13.11.2016). Загл. с экрана. Яз. рус.

2. Raval S. Decentralized application – Sebastopol: O'Reilly Media, 2016. – 103 с.

3. The General Theory of Decentralized Applications, Dapps [Электронный ресурс] URL: https://github.com/DavidJohnstonCEO/DecentralizedApplications (дата обращения: 10.11.2016). Загл. с экрана. Яз. англ.

4. Mesh-СЕТИ [Электронный ресурс] URL: http://www.lastmile.su/files/article_pdf/2/article_2099_241.pdf (дата обращения 13.11.2016). Загл. с экрана. Яз. рус.

5. Mesh Networks: The Future of Communication [Электронный ресурс] URL: http://www.makeuseof.com/tag/mesh-networks-future-communication/ (дата обращения 13.11.2016). Загл. с экрана. Яз. англ.

6. Peer-to-Peer (P2P) Technologies and Services [Электронный ресурс] URL: https://tech.ebu.ch/docs/techreports/tr009.pdf (дата обращения: 13.11.2016). Загл. с экрана. Яз. англ.

7. SimpleMailTransferProtocol [Электронный ресурс] URL: https://www.ietf.org/rfc/rfc0821.txt (дата обращения: 20.02.2017). Загл. с экрана. Яз. англ.

8. How PGP works [Электронный ресурс] URL: http://www.pgpi.org/doc/pgpintro/ (дата обращения 20.02.2017). Загл с экрана. Яз. англ.

9. Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence [Электронный ресурс] URL: https://xmpp.org/rfcs/rfc3921.html (дата обращения: 20.02.2017). Загл. с экрана. Яз. англ.

10. MTProto Mobile Protocol [Электронный ресурс] URL: https://core.telegram.org/mtproto (дата обращения: 21.02.2017). Загл. сэкрана. Яз. англ.

11. DHT Protocol [Электронный ресурс] URL: http://www.bittorrent.org/beps/bep_0005.html (дата обращения: 21.02.2017). Загл. с экрана. Яз. англ.

12. Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications [Электронный ресурс] URL: https://pdos.csail.mit.edu/papers/ton:chord/paper-ton.pdf (дата обращения: 22.04.2017). Загл. сэкрана. Яз. англ.

13. Chord Theory [Электронный ресурс] URL: http://www.springer.com/cda/content/document/cda_downloaddocument/9781461490074-c1.pdf?SGWID=0-0-45-1421110-p175429135 (дата обращения: 22.04.2017). Загл. с экрана. Яз. англ.

14. Building a Replicated Data Store using Berkeley DB and the Chord DHT [Электронный ресурс] URL: http://www.diva-portal.org/smash/get/diva2:348050/FULLTEXT01.pdf (дата обращения: 25.04.2017). Загл. с экрана. Яз. англ.

15. Таненбаум Э., Уэзеролл Д. Компьютерные сети. 5-е изд. – СПб.: Питер, 2012. – 960 с.: ил.

16. Kademlia: APeer-to-peerInformationSystemBasedontheXORMetric [Электронныйресурс] URL: https://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf (дата обращения: 27.04.2017). Загл. сэкрана. Яз. англ.

17. Kademlia: A Design Specification [Электронныйресурс] URL: http://xlattice.sourceforge.net/components/protocol/kademlia/specs.html (дата обращения: 28.04.2017). Загл. с экрана. Яз. англ.

18. Kademlia-Based Fundamentals [Электронный ресурс] URL: https://github.com/telehash/telehash.github.io/blob/master/v2/dht.md (дата обращения: 28.04.2017). Загл. с экрана. Яз. англ.

19. Bittorrent Protocol Specification [Электронный ресурс] URL: https://wiki.theory.org/BitTorrentSpecification (дата обращения: 13.11.2016). Загл. с экрана. Яз. англ.

20. Стек приложений Blockchain [Электронный ресурс] URL: https://habrahabr.ru/company/piter/blog/277625/ (дата обращения: 15.11.2016)

21. The Go Programming Language [Электронный ресурс] URL: https://golang.org/doc/ (дата обращения: 21.05.2017). Загл. с экрана. Яз. англ.

22. TRANSMISSIONCONTROLPROTOCOL [Электронный ресурс] URL: https://www.ietf.org/rfc/rfc793.txt (дата обращения: 21.05.2017). Загл. с экрана. Яз. англ.

23. Intro to BoltDB: Painless Performant Persistence [Электронный ресурс] URL: https://npf.io/2014/07/intro-to-boltdb-painless-performant-persistence/ (дата обращения: 21.05.2017). Загл. с экрана. Яз. англ.

24. OpenCL – The open standard for parallel programming of heterogeneous systems [Электронный ресурс] URL: https://www.khronos.org/opencl/ (дата обращения: 21.05.2017). Загл. с экрана. Яз. англ.

25. WebSocket [Электронный ресурс] URL: https://developer.mozilla.org/ru/docs/WebSockets (дата обращения: 21.05.2017) Загл. с экрана. Яз. англ.