Оглавление

Введение

В настоящее время передача коммерческой, научно-технической, личной информации по различным сетям стала фактом нашей жизни. В связи с этим большое значение приобретает сокрытие самого факта контакта фирм, частных лиц. Множество фирм (e-commerce, e-finance e-health), использующих глобальные и локальные сети, заинтересовано в решении вопросов безопасности и анонимности. Некоторые фирмы и интернет-компании используют виртуальные частные сети (VPN), финансируя развитие новых средств безопасности.

Шифрование является надежным средством защиты от третьих лиц, но с ростом объема информации и числа пользователей глобальной сети инструменты защиты информации требуют развития. Возникают требования не только к сокрытию данных, но и к анонимности в сети. Необходимо исключить возможность утечки информации и установления факта контакта. Эти задачи в настоящее время решаются с помощью анонимных сетей.

Анонимные сети могут служить различным интересам для различных групп пользователей:

- затрудняют возможность отслеживания рядовых граждан;
- позволяют сохранить коммерческую тайну компаниям;
- способствуют анонимной работе правоохранительных органов;
- защищают от несанкционированной утечки информации;
- затрудняют работу вредоносных приложений;
- ограничивают доступ и делают невозможным определить источник информации.

Обеспечение анонимности в сети является многоаспектной задачей и требует комплексных решений. Важнейшей криптографической проблемой остается проблема анализа трафика (the traffic analysis problem)[1].

Дэвид Чаум (David Chaum) предложил концепцию перемешивающих узлов, актуальную и по сей день. Оригинальная концепция взаимодействия в сети, предложенная в работе[1], работоспособна и в задачах реального времени, таких как обмен мгновенными сообщениями и просмотр веб-страниц[2].

В соответствии с техническим заданием на выполнение квалификационной работы бакалавра необходимо разработать одноранговую анонимную сеть с зашифрованным трафиком.

1. Аналитический раздел

1.1 Постановка задачи

В соответствии с техническим заданием на квалификационную работу, необходимо разработать одноранговую анонимную сеть с зашифрованным трафиком. Разрабатываемая система должна обладать следующими свойствами:

a) обеспечивать потоковый обмен информацией между участниками;
b) быть устойчивой к выходу из строя узлов;
c) препятствовать установлению личности;
d) обеспечивать недоступность третьим лицам;
e) обеспечивать шифрование трафика.

Для решения поставленной задачи необходимо:

a) Проанализировать особенности построения сети и разработать структуру системы.
b) Исследовать и выбрать методы построения анонимной сети, разработать алгоритмы, обеспечивающие анонимность.
c) Исследовать и выбрать методы, препятствующие перехвату сообщений, разработать алгоритмы сокрытия трафика.
d) Исследовать и выбрать методы шифрования трафика.

1.2 Анализ проблемы сокрытия трафика на уровне узлов

Проблема сокрытия трафика является ключевой проблемой сохранения конфиденциальности контактов в сети и приобретает все большее значение с развитием электронного общения. Решение проблемы анализа трафика должно удовлетворять следующим требованиям[1]:

1. Ни один промежуточный узел не должен иметь доступ к информации об отправителе и получателе сообщения, а также должна быть невозможной подмена сообщения при отсутствии закрытого ключа.
2. Каждый узел должен иметь возможность создавать, транслировать и представлять сообщения в телекоммуникационной сети и каждый может создавать свои собственные сообщения.

Анонимность обеспечивается за счет так называемых перемешивающих узлов. Они должны обеспечивать побитовую неразличимость передаваемых сообщений и перемешивать потоки сообщений. Пользователи такой криптосистемы состоят не только из источников и приемников сообщений, но и содержат узлы, не участвующие напрямую в общении, называемые перемешивающими.

Один из участников – источник, готовит сообщение M для передачи участнику с псевдонимом A. Для этого сообщение дополняется строкой случайной длины $R0$ и зашифровывается открытым ключом Ka. Затем сообщение дополняется новой строкой случайной длины $R1$ и зашифровывается открытым ключом $K1$. При получении, сообщение расшифровывается с помощью закрытого ключа:

$$K1(R1, Ka(R0, M)A) \rightarrow Ka(R0, M)A \qquad (1)$$

Цель перемешивания узлов состоит в том, чтобы скрыть маршрут следования сообщения, источник и приёмник. Шифрование на каждом этапе ограничивает доступ узла к информации, содержащейся в сообщении. Тот факт, что сообщение на каждом этапе при расшифровке изменяет длину и не обрабатывается в текущем виде более одного раза, несет в себе дополнительную защитную функцию.

Эффективность перемешивающих узлов сети базируется на сложности определения случайного пути, зависящего от степени связности в графе, формирующемся исходя из структуры сети. Даже с учетом того, что сеть ограничена, а маршрут может быть построен только с использованием путей текущей топологии, подобная структура эффективна при расширении. Это следует из леммы о перемешивании. Для любых двух подмножеств вершин *S* и *T* графа *G*, $S,T \subseteq V$, число ребер между *S* и *T* с вероятностью *d/n* равно $\frac{d}{n} \cdot |S| \cdot |T|$ - числу ребер в случайном d-регулярном графе, где d – степень регулярности графа. Таким образом, эффективная в подграфе сеть будет также эффективна при ее расширении.

Существующие реализации анонимных сетей используют в своей работе перемешивающие узлы. Реализация этих узлов отличается в основном, схемой составления маршрута сообщения. Существуют алгоритмы, генерирующие маршрут случайным образом или использующие алгоритм подражания муравьиной колонии (*ant colony optimization*, ACO).

1.3 Анализ проблемы сокрытия трафика на уровне пакетов

Обеспечение анонимности требует учета большого числа факторов, способных ее нарушить. К таким факторам относятся[2]:

1. возможность анализ трафика по длине пакетов;
2. перехват сообщения на промежуточном узле;
3. возможность анализа трафика на основе порядка приема пакетов и промежутков времени между последовательным отправлением сообщений;
4. «бомбардировка» узлов пакетами для обнаружения топологии сети;
5. перехват пакетов с целью их повторной отправки (Replay Attack).

Для создания препятствий к анализу трафика по длине передаваемых пакетов, устанавливается фиксированный размер сообщения. Размер пакета определяется таким образом, чтобы реальное сообщение было меньше передаваемого. В этом случае, при меньшей длине сообщения неиспользуемые байты заполняются случайным образом, а при расшифровке на стороне получателя – отбрасываются. При большей длине пакета происходит разбиение сообщения на части одинакового размера. Вместо фиксированного размера сообщения может быть применен механизм генерации случайного размера сообщения. В этом случае длина пакета случайна, что также не позволяет отследить зависимость размера пакетов от исходной длины сообщения.

Для ограничения доступа к сообщению на промежуточном узле осуществляется послойное шифрование сообщения. Пусть узел – источник имеет информацию об *n* узлах, доступных для перемешивания. Пусть узлу – источнику также известны открытые ключи всех узлов *K1…Kn*. Тогда перед отправкой, сообщение должно быть зашифровано *n-1* раз. В соответствии с требованием равной длины сообщений, все данные дополняются набором случайных байт *R1…Rn*. После передачи сообщения на первый узел в построенной последовательности узлов и расшифровки верхнего слоя данных, сообщение может быть представлено следующим образом:

$$K2\left(R2,\ldots,K\langle n-1\rangle\left(R\langle n-1\rangle,Kn\left(Rn,X\right)\right)\right),\text{ где} \qquad (2)$$

K_i – открытый ключ, $i \notin 2..n$

R_i – случайная последовательность знаков, $i \notin 2..n$.

В результате перехвата и анализа трафика невозможно определить отправителя и получателя, поскольку нет доступа к закрытым ключам каждого слоя шифрования. Текущий узел имеет сведения только о том, с какого

промежуточного узла пришло сообщение и на какой следующий промежуточный узел его необходимо отправить.

Важнейшим фактором обеспечения анонимности в сети является изменение последовательности передачи пакетов. Считается[2], что изменение последовательности передачи пакетов приближает анонимность сети к теоретическому пределу. Для изменения порядка следования пакетов по сети применяются временные задержки. Временные задержки, генерируются случайным образом для каждого сообщения. Это позволяет нарушить порядок следования пакетов по сети, установленный отправителем. Кроме того, в работе [1] предлагается использовать ложные сообщения, не несущие в себе полезной информации, но позволяющие обеспечить однородность заполнения сети пакетами. Также важно, чтобы сеть работала одинаково с зашифрованным и открытым трафиком, не делая различий. Сложность анализа пакетов в системах с такими механизмами заключается в равной вероятности порядка и направления следования пакетов.

Для получения несанкционированного доступа к информации может быть применен механизм перехвата сообщений во время аутентификации узлов. Повторная передача злоумышленником перехваченных пакетов в сеть не должна приводить к успешной аутентификации компьютера злоумышленника. Необходимо гарантировать, что сообщение будет получено, обработано и отправлено только один раз[2]. Уникальность передаваемых пакетов затрудняет атаку, позволяющую получить несанкционированный доступ перехватом пакетов. Эта мера реализуется за счет послойного шифрования и случайно генерируемых различий в сообщениях, а также тщательным сокрытием закрытых ключей.

1.4 Анализ и выбор функциональных элементов анонимной сети.

Анонимные сети, построенные по принципу перемешивающих узлов имеют общую архитектурную основу. Типовая архитектура такой сети представлена на рисунке 1.1[2]:

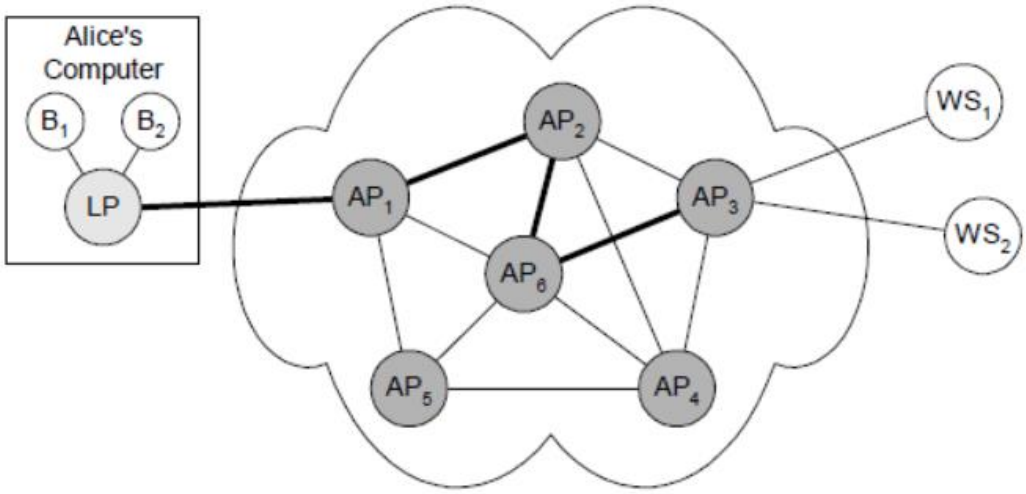

Рисунок 1.1. Структура анонимной сети с перемешивающими узлами.

Ядро анонимной сети составляют анонимные прокси-сервера (AP). Анонимные прокси-сервера – это распределенные по сети точки доступа, обеспечивающие функциональные возможности перемешивающего узла. Такие точки доступа должны обладать доверием, регулярно поддерживаться и обслуживаться.

Пользователь обращается к анонимной сети через локальный прокси – сервер (LP), который представляет собой запущенную на компьютере пользователя программу (приложение или службу). При отправке сообщения, пользователь определяет множество доверенных анонимных прокси-серверов и организует анонимный туннель.

Анонимный туннель – это конкретная последовательность, состоящая из доверенных прокси-серверов. Все соединения с компьютерами в локальной или глобальной сети осуществляются через этот туннель. По завершении

передачи сообщения, туннель разрушается. Аналогичные действия по построению туннеля выполняются также приёмником для отправки ответа получателю.

Важным аспектом является возможность использования туннеля несколькими локальными соединениями. Возникает задача мультиплексирования пакетов при отправке, и распределения данных при получении.

Точки доступа могут работать на прикладном уровне сетевой модели OSI, так как требуется, чтобы защищенные и незащищенные данные передавались одинаково. Следует отметить, что в сетях с большим числом пользователей не обязательно подключение каждого клиента к каждому, но при создании подключения между клиентами должно быть ровно одно соединение. В результате этого анонимные туннели нельзя отличить от открытых сетевых соединений, что позволяет сохранить топологию сети в тайне.

У каждого узла сети имеется набор сервисов. Сервисы имеют разные требования к пропускной способности, надежности передачи, безопасности и стабильности. Тем не менее, они должны работать в общей инфраструктуре. Инфраструктура во многом определяет топологию сети, а топология напрямую влияет на степень анонимности. К инфраструктуре предъявляются определенные требования[3]:

a) Высокая скорость работы. Сеть должна быть в состоянии обработать пакеты с достаточной скоростью, учитывая, что в системе применяются искусственные задержки отправления пакетов.
b) Изоляция. Любой узел в рамках общей инфраструктуры должен иметь уникальный псевдоним, обеспечивающий надежную адресацию.
c) Гибкость. Сеть должна обеспечивать работу с разными типами пакетов и допускать появление новых типов пакетов.

d) Масштабируемость. Необходимо, чтобы сеть можно было расширять, осуществляя минимум технологических и структурных изменений в ней.

Выбор функциональных элементов анонимной сети осуществляется так, чтобы полученная инфраструктура соответствовала требованиям, указанным выше. Важно, чтобы инфраструктура соответствовала задачам сети с учетом приоритетности требований.

1.5 Анализ и выбор сетевых протоколов

С целью сокрытия различий в передаче зашифрованных и открытых данных, целесообразно использовать стандартные сетевые протоколы TCP/IP. В рамках реализации данной работы, рассматривается разработка локальной сети на существующей виртуальной платформе, поэтому протоколы канального, сетевого и транспортного уровня могут быть продекларированы.

На сетевом уровне модели OSI предполагается использование протокола IPv4. Для передачи сообщений в зашифрованном виде между всеми участниками анонимной сети предполагается использовать протокол TCP, так как его механизмы обеспечивают надежную передачу информации и сохранение целостности данных.

Для обеспечения гибкости системы и ускорения развертывания сети рассылка информации о доступных узлах будет осуществляться по протоколу UDP. Этот протокол не требует предварительного установления соединения и поддерживает широковещательную рассылку.

Задачи обеспечения анонимности в сети отличаются от задач незащищенной сети, поэтому требуют реализации особого протокола прикладного уровня. В работе[4] предлагается обобщенный вариант протокола, учитывающий особенности построения анонимных сетей.

Основной сложностью при реализации протокола прикладного уровня является необходимость мультиплексирования сообщений в одно соединение от разных прокси-серверов. Для этого используется уникальный идентификатор узла сети для каждого анонимного туннеля.

Поскольку пакет формируется из сообщения и набора случайных байт, обеспечивающих равный размер пакетов, то требуется указание количества байт полезной нагрузки. Для корректной работы протокола требуется включение данного поля в состав сетевого кадра.

Тип передаваемых пакетов не является обязательным параметром, но присутствует во всех существующих реализациях анонимных сетей. Поле, характеризующее тип пакета, позволяет ускорить работу промежуточного узла, облегчая обработку данных. Пример формата сообщения с указанием примерной длины полей для анонимной сети, основанной на перемешивающих узлах представлен на рисунке 1.2[4]:

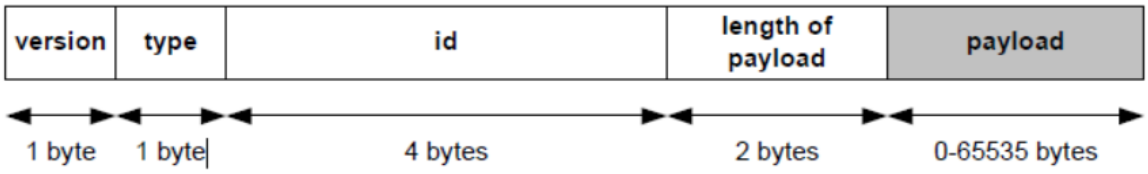

Рисунок 1.2. Формат сообщения в анонимной сети

Таким образом, выбранный протокол позволяет реализовать анонимную сеть, учитывая особенности инфраструктуры. Предложенный формат кадра позволяет добиться устойчивой анонимности, допуская реализацию защиты от всех опасных факторов, рассмотренных в данном разделе.

1.6 Анализ и выбор алгоритмов шифрования данных

Шифрование является необходимым условием анонимности и решает многие проблемы обеспечения безопасности в сети. Современные алгоритмы шифрования исключают практическую целесообразность расшифровки сообщений методом подбора, даже при известной структуре алгоритма[5].

Важной проблемой является проблема безопасного обмена ключами, поскольку обмен происходит в незашифрованной сессии. Проблема решается использованием асимметричных алгоритмов с открытым ключом. В этом случае открытый ключ может быть передан без опасения раскрытия зашифрованного сообщения в дальнейшем. Однако, существует опасность подмены ключа, когда злоумышленник представляется узлом сети и перенаправляет на себя сообщения.

Поскольку каждый узел самостоятельно выбирает маршрут в сети и доверенные узлы, пакет может быть направлен в обход узла злоумышленника. Послойное шифрование, ограничивающее доступ промежуточного узла к информации в пакете также способствует сохранению анонимности.

Функционирующие реализации анонимных сетей используют для шифрования пакетов стандартные алгоритмы AES, RSA и ElGamal, надежность которых установлена и подтверждена экспериментально. На различных этапах передачи информации в этих сетях (от клиента ко входному шлюзу, между узлами туннеля, от выходного шлюза к получателю) применяются различные алгоритмы. Выбор алгоритма зависит от задач, решаемых на конкретном этапе передачи. Клиент при шифровании должен использовать установленные алгоритмы, поэтому используемые алгоритмы шифрования определяются заранее.

Advanced Encryption Standard (AES, Rijndael) — симметричный алгоритм блочного шифрования (размер блока 128 бит, ключ 128/192/256 бит). Алгоритм криптографически проанализирован и широко используется. AES

является одним из самых распространённых алгоритмов симметричного шифрования[5].

RSA (аббревиатура от фамилий Rivest, Shamir и Adleman) — криптографический алгоритм с открытым ключом, основывающийся на вычислительной сложности задачи факторизации больших целых чисел. Криптосистема RSA пригодна для шифрования и цифровой подписи. Алгоритм используется в большом числе криптографических приложений. Из-за низкой скорости шифрования сообщения обычно шифруют с помощью более производительных симметричных алгоритмов со случайным ключом (сеансовый ключ), а с помощью RSA шифруют лишь этот ключ, таким образом реализуется гибридная криптосистема.

Схема Эль-Гамаля (Elgamal) — криптосистема с открытым ключом, основанная на трудности вычисления дискретных логарифмов в конечном поле. Криптосистема включает в себя алгоритм шифрования и алгоритм цифровой подписи. Сравнение алгоритмов шифрования приводится в таблице 1.1.

Таблица 1.1 Сравнение алгоритмов шифрования

Параметр	AES	RSA	ElGamal
Тип алгоритма	Симметричный	Асимметричный	Асимметричный
Размер ключа	до 256 бит	До 4096 бит	До 4096 бит
Назначение	Шифрование	Шифрование и подпись	Шифрование и подпись
Криптостойкость, MIPS	$7.2 \cdot 10^{16}$	$2,7 \cdot 10^{28}$ для ключа 1300 бит	При одинаковой длине ключа $2,7 \cdot 10^{28}$ для ключа 1300 бит
Математическая база	Основан на сложности решения определённых видов уравнений	Основан на трудности задачи факторизации больших чисел	Основан на трудной задаче вычисления дискретных логарифмов в конечном поле
Скорость работы	49,1 Мбит/с, ключ 256 бит	около 30 Кбит/с при 512 битном ключе	около 20 Кбит/с при 512 битном ключе

Чтобы обеспечить необходимый уровень защиты и скорость работы анонимной сети, в качества алгоритма шифрования был выбран асимметричный алгоритм RSA. Алгоритм предоставляет широкие возможности для использования, высокую скорость работы среди асимметричных алгоритмов, а также высокую криптоустойчивость и надежность.

Если каждой рабочей станции будет известен открытый ключ узла сети, это позволит осуществлять обмен информацией между двумя рабочими станциями с сохранением анонимности на промежуточных узлах.

В некоторых анонимных сетях ключи имеют ограниченный срок службы. Данный подход повышает надежность системы, но требует дополнительных накладных расходов в виде рассылки квитанций об изменении ключа. Пакеты, зашифрованные старой версией ключа, также должны быть приняты, поэтому возникает проблема идентификации ключа для данного пакета.

Выводы

В результате анализа предметной области были определены правила и методы реализации одноранговой анонимной сети, осуществлен выбор сетевых протоколов и базовых функциональных элементов. Были выявлены проблемы обеспечения анонимности и предложены пути их решения в рамках данной квалификационной работы.

2. Конструкторский раздел

2.1 Разработка диаграммы работы узла анонимной сети

Участникам обмена информацией недоступны данные об источнике и приемнике сообщений. Для обеспечения работы сети, промежуточному узлу достаточно знать адрес узла, которому следует передать сообщение для дальнейшей обработки. Изменение схемы маршрутизации обеспечивает надежное сокрытие отправителя, получателя, а также текста сообщения. Для организации этой схемы можно применить механизм туннелей[6].

Туннель представляет собой последовательность узлов, участвующих в пересылке сообщений. Данная последовательность определяется на стороне отправителя и больше никому не доступна. Каждый участник сети является одновременно промежуточным узлом, возможным источником и приемником сообщений. Адреса узлов не могут быть секретными, так как иначе сеть не могла бы функционировать.

В целях сохранения конфиденциальности пересылаемых данных, вся пересылаемая информация должна подвергаться шифрованию. Каждый узел помимо IP-адреса, идентификатора и порта публикует открытые ключи шифрования. Все сообщения, адресованные узлу, должны подвергаться шифрованию его открытым ключом.

Для усиления безопасности все туннели должны обновляться и перестраиваться после каждой отправки сообщения. Входящие и исходящие туннели должны быть различными.

В качестве основных параметров управления сетью выбираются следующие:

а) степень перемешивания сообщений, регулирует уровень вложенности шифрования, определяет длину туннеля и число промежуточных передач пакетов;

b) время обновления списка доступных узлов, способствует экономии трафика за счет регулирования периодичности обновления списка узлов и запроса ключей;
c) порт широковещательной рассылки и порт передачи по TCP, регулирование которых необходимо для обеспечения гибкости и переносимости приложения.

Автоматическим управляющим параметром является тип пакета. Данный параметр указывает узлу, какие действия необходимо выполнить для обработки пакета. Возможные типы пакетов будут представлены при описании структур данных.

IDEF0 диаграмма работы маршрутизатора нулевого и первого уровня представлена на рисунке 2.1.

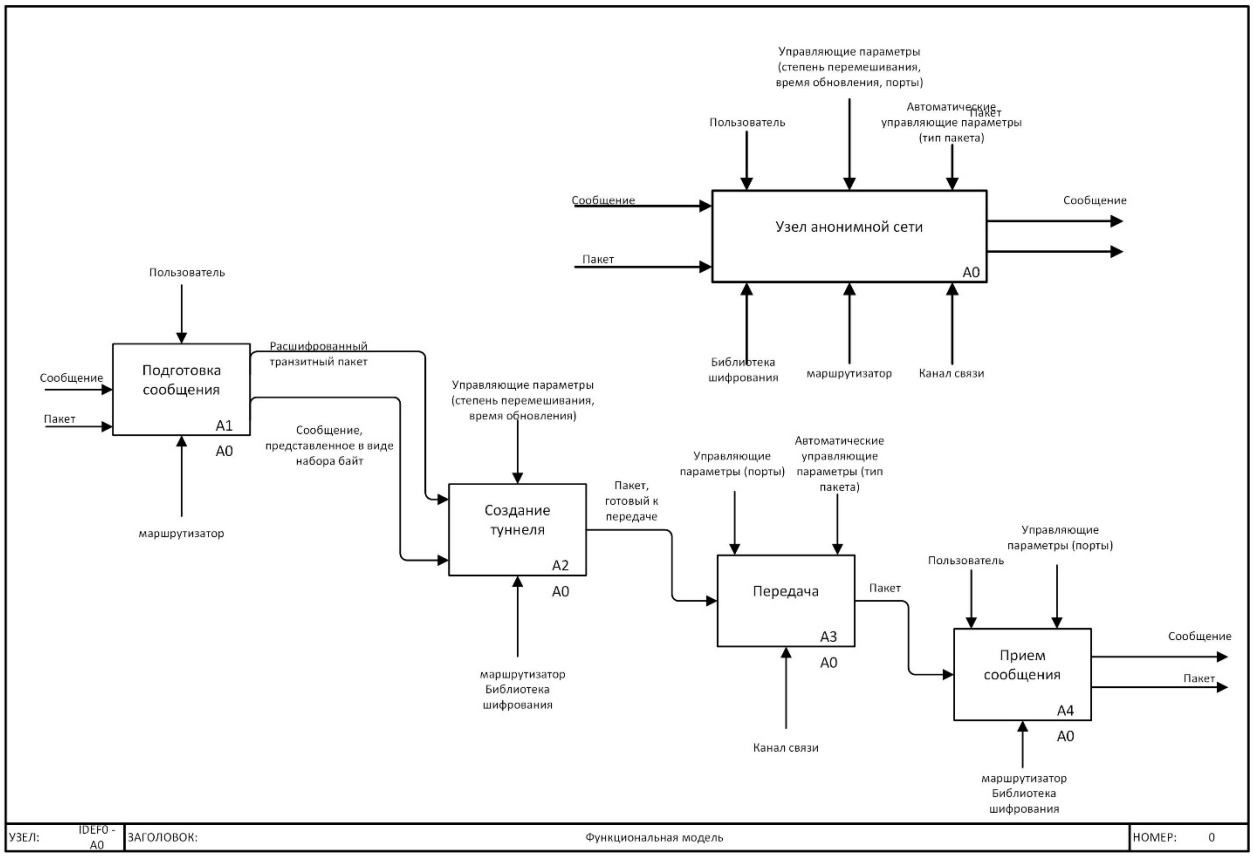

Рисунок 2.1 IDEF0-диаграмма работы узла.

Этап подготовки сообщения заключается в приеме сообщения от пользователя через интерфейс программы и в преобразовании его в формат, подлежащий передаче. При получении транзитного пакета, подготовка сообщения заключается в расшифровке слоя данных, предназначенного для этого узла и передача транзитного пакета на следующий уровень.

Преобразованное сообщение, пригодное для отправки, упаковывается для помещения в туннель. На этом этапе определяется последовательность узлов передачи и осуществляется послойное шифрование.

Механизм приема сообщений должен обеспечивать возможность мультиплексирования и разбиения входных данных по типу пакета. После получения набора байт из канала передачи, приемник данных формирует пакет. На этом этапе пакет представляет из себя сообщение, полученное полностью и готовое к сериализации. Этот пакет передается в блок обработки, который обеспечивает корректное представление данных в установленном формате прикладного протокола (сериализацию).

При получении пакета и определении его, как финального, данные для отображения передаются в блок отображения сообщения, связанный с пользовательским интерфейсом. Этот блок обеспечивает корректное отображение данных для пользователя и оповещение пользователя о появлении нового сообщения. Диаграмма последовательностей действий при приеме сообщения представлена на рисунке 2.2.

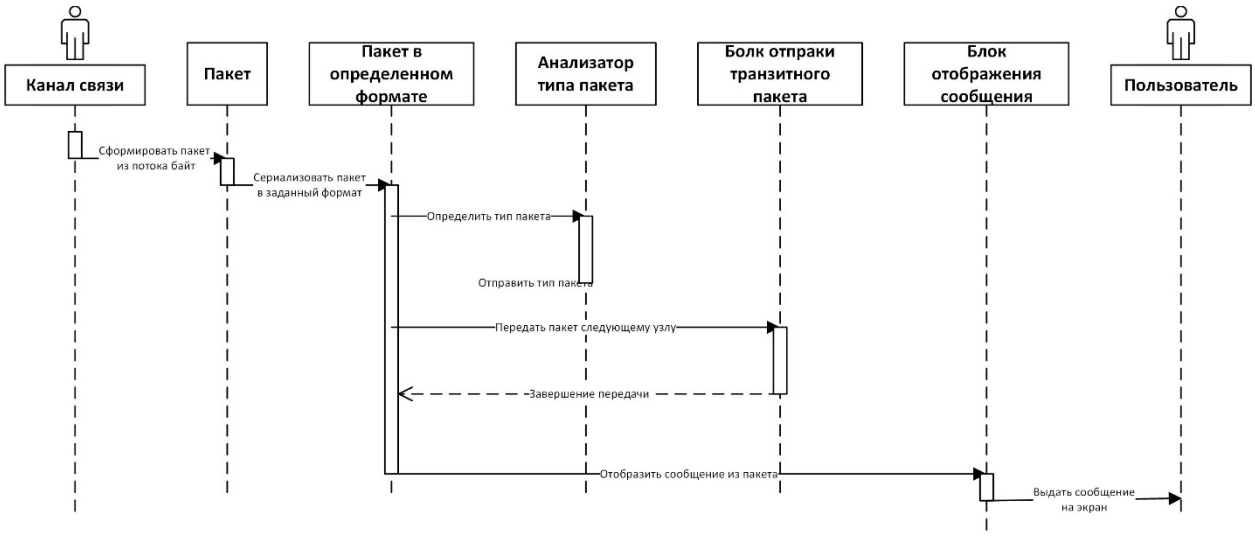

Рисунок 2.2 Разработка диаграммы последовательностей действий при получении сообщения.

Одноранговая анонимная сеть, разрабатываемая в рамках данного дипломного проекта, подразумевает реализацию функций получения, отправки и обработки сообщений в едином приложении. Поэтому функции сети, представленные в этом разделе можно рассматривать в качестве блоков одной общей системы.

В сочетании с базовыми функциональными элементами, описанными в пункте 1.3 аналитического раздела пояснительной записки, можно определить функциональные блоки, которые необходимо реализовать:

- анонимный прокси-сервер, реализующий прием и обработку транзитных сообщений;
- локальный прокси – сервер, обеспечивающий подготовку сообщений, шифрование и отправку;
- туннель, обеспечивающий анонимность передачи сообщения.

Диаграмма прецедентов для указанных блоков представлена на рисунке 2.3.

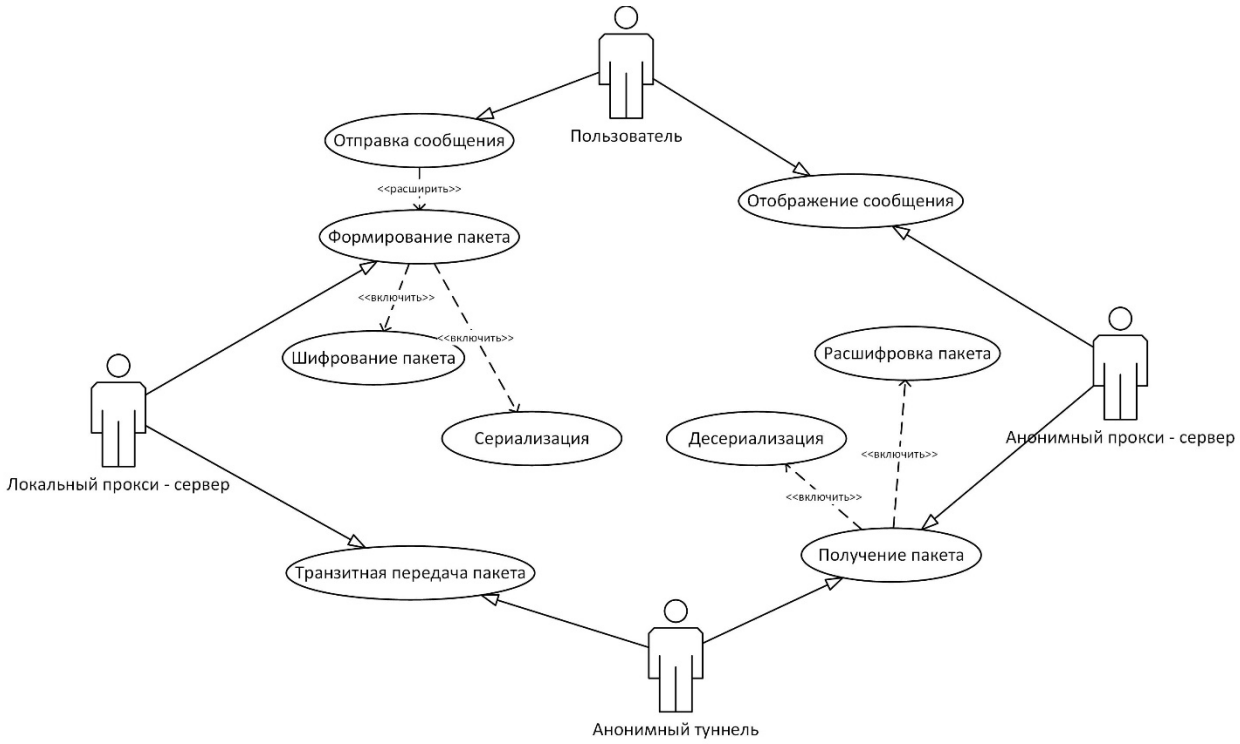

Рисунок 2.3 Диаграмма прецедентов функциональных блоков анонимной сети.

2.2 Алгоритм функционирования анонимного прокси – сервера

Основная нагрузка по приему, передаче сообщения лежит на анонимном прокси-сервере. Обеспечение безопасности и анонимности также осуществляется этим узлом. Алгоритм работы анонимного прокси-сервера представлен на рисунке 2.4.

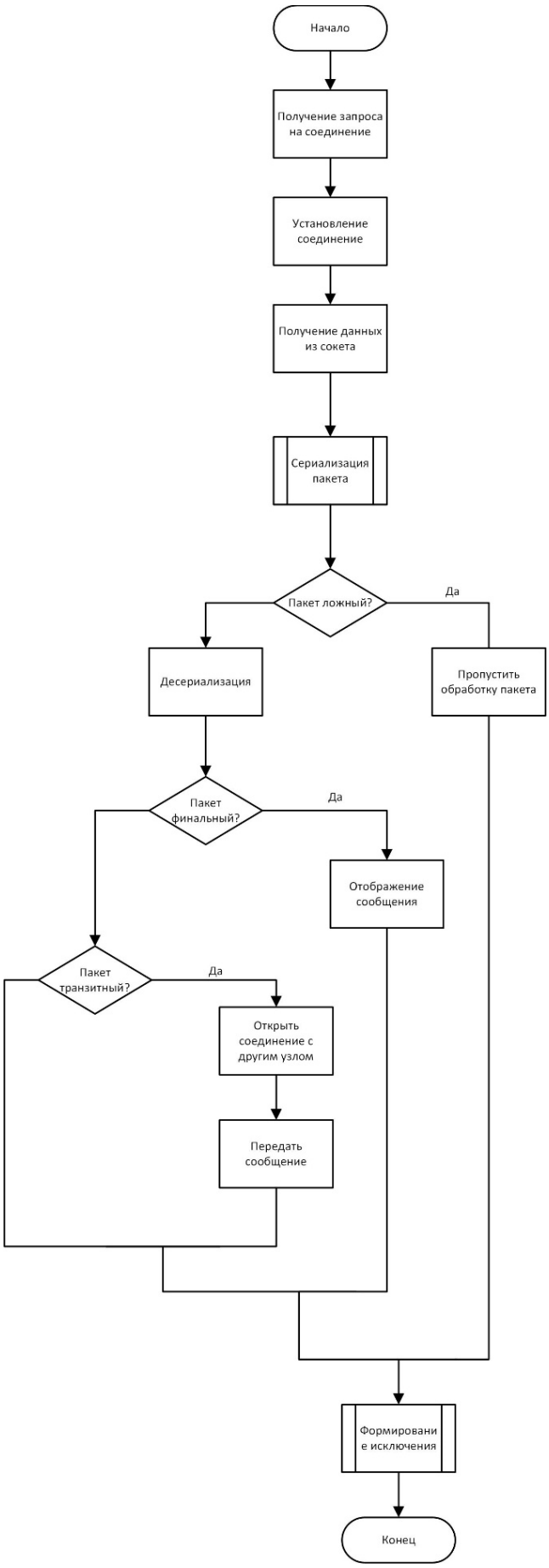

Рисунок 2.4 Схема алгоритма работы анонимного прокси-сервера.

После установления соединения и получения зашифрованного пакета происходит расшифровка и анализ пакета. Для ускорения анализа, ложные пакеты, не содержащие полезной информации, отбрасываются. Пакеты, содержащие полезную информацию, подразделяются на несколько типов. Тип пакета указывает серверу, какой способ обработки пакета выбрать.

Пакет типа FINALLY

Сообщения, помечаемые данным флагом, позволяют дать сигнал серверу о необходимости расшифровки и отображения сообщения. Пакет с типом FINALLY не подлежит дальнейшей передаче по сети и означает достижение конечной точки туннеля (приемника). Задачей анонимного прокси-сервера на данном этапе является оповещение интерфейса о необходимости вывода сообщения на экран пользователя. Поскольку до прибытия на конечный узел, пакет остается зашифрованным, промежуточный узел не может определить, будет ли пакет являться финальным на следующем шаге.

Пакет типа TRANSIT

Пакет данного типа указывает серверу, что необходимо расшифровать пришедший пакет, определить адрес следующего получателя и передать сообщение ему. Пакет типа TRANSIT имеет только два открытых поля: тип пакета и адрес следующего получателя. Поскольку транзит пакета между двумя конкретными узлами осуществляется нерегулярно, целесообразно открывать соединение отдельно для каждого сообщения данного типа.

Пакет типа FAKE

Пакет типа FAKE предназначен для пометки ложных пакетов. Так как транзитные узлы не имеют доступ к финальной метке пакета, для них любой пересылаемый пакет имеет тип TRANSIT. Пакете типа FAKE указывает серверу, что пакет прибыл в узел назначения, но нет необходимости в обработке данных внутри него. Это экономит время анализа пакетов.

2.3 Формальное описание построения туннеля.

Туннель представляет собой последовательность узлов, через которую проходит пакет, прежде чем достичь адресата. Каждый туннель работает в одну сторону и предназначен для обеспечения анонимности на уровне пакетов. Теоретически возможна ситуация, что какой-то из узлов будет скомпрометирован, однако этому узлу не будет доступна информация об отправителе и получателе и анонимность туннеля сохранится[7].

Кроме того, при каждой пересылке, сообщение подвергается зашифровке, что обеспечивает сокрытие содержимого пакетов. Длина туннеля, положение узла в туннеле и адрес получателя известны только отправителю сообщения. Диаграмма последовательностей действий при пересылке сообщения по туннелю представлена на рисунке 2.5.

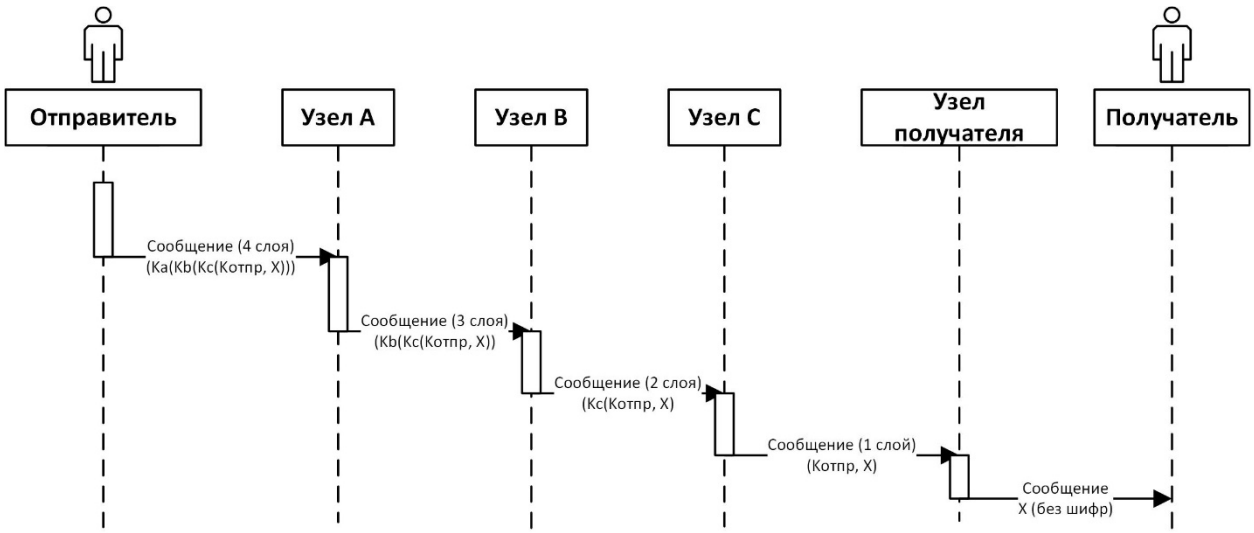

Рисунок 2.5 Разработка диаграммы последовательностей действий работы туннеля.

Данная реализация туннеля обладает следующими особенностями:

1. Ограничений на число вхождений одного и того же узла в туннель нет. Слои данных, доступные при первичной и повторной расшифровке различаются и воспринимаются узлом, как различные

пакеты. У такого транзитного узла нет информации, позволяющей сопоставить два пакета, как пакеты, следующие в одном туннеле.

2. Выбор узлов туннеля нельзя возложить на сторонний узел, так как тот может быть скомпрометирован и использовать для построения узлы, не обладающие доверием. Поэтому выбор узлов для туннеля осуществляется непосредственно отправителем. Для этого из набора доверенных узлов выбирается случайным образом последовательность узлов для создания туннеля.
3. Нет необходимости в поиске доступных узлов при создании каждого нового туннеля. Можно полагать, что выход из строя узлов является маловероятным событием. Тем не менее, требуется периодическое обновление списка доступных узлов. Обновление должно происходить автоматически через заданный промежуток времени и вручную при необходимости срочного обновления. Алгоритм планирования туннеля представлен на рисунке 2.6.

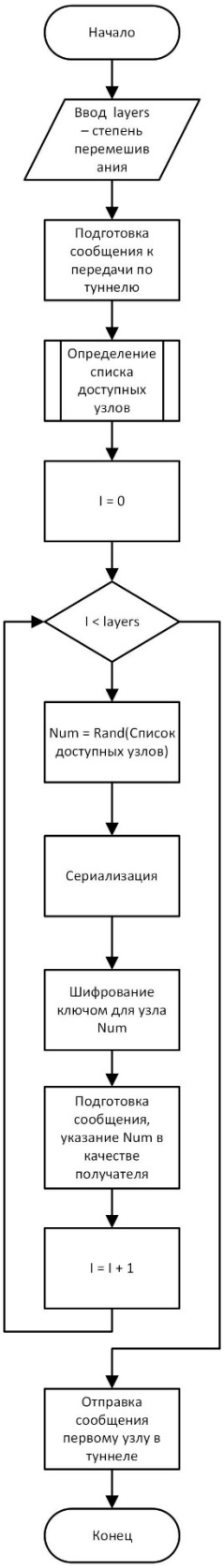

Рисунок 2.6 Схема алгоритма планирования туннеля.

2.4 Алгоритм функционирования локального прокси – сервера

Локальный прокси сервер обеспечивает взаимодействие сети с пользователем. Данный сервер обеспечивает отправку сообщения, планирование туннеля и оповещение интерфейса пользователя о событиях, происходящих в сети.

Одной из функций локального прокси-сервера является получение и обновление списка доступных узлов. Каждый узел при запуске отправляет пакет по широковещательному UDP соединению, оповещающий другие узлы. Локальный прокси-сервер получает информацию о доступном узле и проверяет, является ли узел доверенным. Схема алгоритма обнаружения новых узлов представлена на рисунке 2.7.

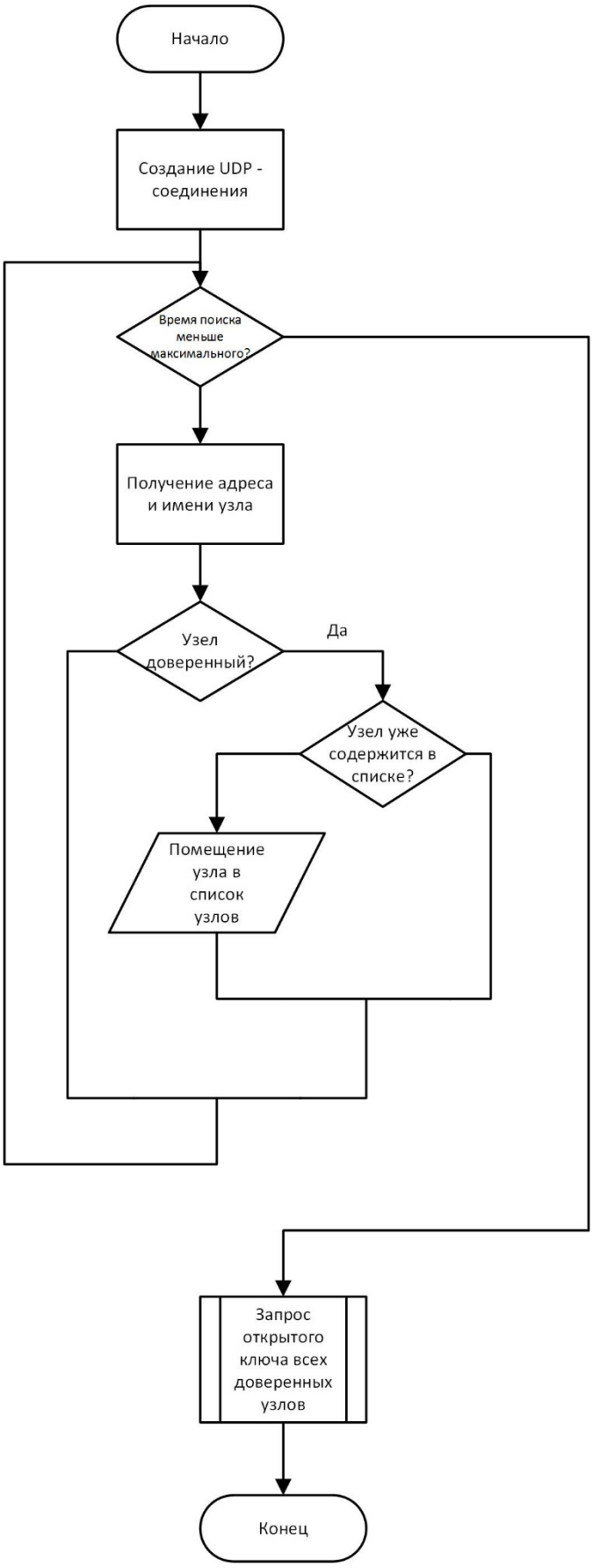

Рисунок 2.7 Схема алгоритма получения списка доверенных узлов.

Локальный прокси-сервер составляет сообщение, в котором помимо зашифрованного поля данных содержится адрес отправителя, чтобы получатель (и только он) смог его идентифицировать. Для пересылки информации по туннелю отправитель последовательно шифрует сообщение соответствующим набором ключей в нужной последовательности.

2.5 Разработка структуры ПО

В соответствии с алгоритмами, представленными в конструкторской части пояснительной записки, учитывая особенности реализации туннеля, анонимного и локального прокси-сервера, была разработана структура программного обеспечения.

Для реализации структуры одноранговой сети, в соответствии с заданием на квалификационную работу, требуется объединить клиент и сервер в одном приложении. Для этого предлагается разделить логически блоки клиента и сервера, объединив их с помощью программного интерфейса. Структура программного обеспечения, реализующего одноранговую анонимную сеть, представлена на рисунке 2.8.

Рисунок 2.8 Разработка структуры ПО.

3. Технологический раздел.

3.1 Выбор языка и среды программирования

Для реализации программного продукта была выбрана технология объектно-ориентированного программирования. При выборе языка было учтено, что язык должен иметь стандартные компоненты, облегчающие работу с сетью, криптографией и сериализацией данных.

В соответствии с выбором объектно-ориентированного подхода к разработке ПО, для реализации проекта был выбран язык C#. Язык C# является полностью объектно-ориентированным, а кроме того, в нем реализована технология сборки мусора, что позволяет избежать многих случаев утечек памяти. Потеря быстродействия по сравнению с языками низкого уровня не является критичной, так как по сравнению с временными издержками при передаче пакетов, выигрыш за счет оптимизации кода будет ничтожно малым.

В качестве среды разработки была выбрана платформа Visual Studio 2013. Данная платформа предоставляет полный набор инструментов и служб для разработки приложений для настольных компьютеров, интернета и сетевых систем. Платформа обладает гибкими инструментами разработки для языка C# и инструментами интеграции с операционной системой Windows.

Для тестирования приложения была развернута виртуальная локальная сеть на базе виртуальной платформы VMware. Данная виртуальная платформа обеспечивает широкие возможности обмена данными между виртуальными машинами, а также автоматическую настройку сети.

3.2 Диаграммы классов системы, реализующей анонимную сеть

В соответствии с представленной в пункте 2.5 конструкторского раздела пояснительной записки структурой программного обеспечения, разработана диаграмма классов.

В качестве базового класса выбран класс «Базовый прокси-сервер», инкапсулирующий в себе основные параметры рабочей станции как сервера для отправки и получения сообщений. Данный базовый класс расширяется блоком шифрования, поскольку шифрование применяется во всех серверных блоках.

Классы «Анонимный прокси-сервер» и «Локальный прокси-сервер» наследуют базовый класс, что позволяет реализовать конкретные функциональные возможности для разных типов задач с использованием одинаковых параметров рабочей станции.

Вспомогательные классы «Узел» и «Сообщение» представляют собой структуры данных, используемые в приложении.

Одноранговая структура сети подразумевает объединение функциональности локального и анонимного прокси-сервера в одном узле. При этом необходимо контролировать общие данные разных блоков одной системы и исключить их дублирование.

Диаграмма классов представлена на рисунке 3.1.

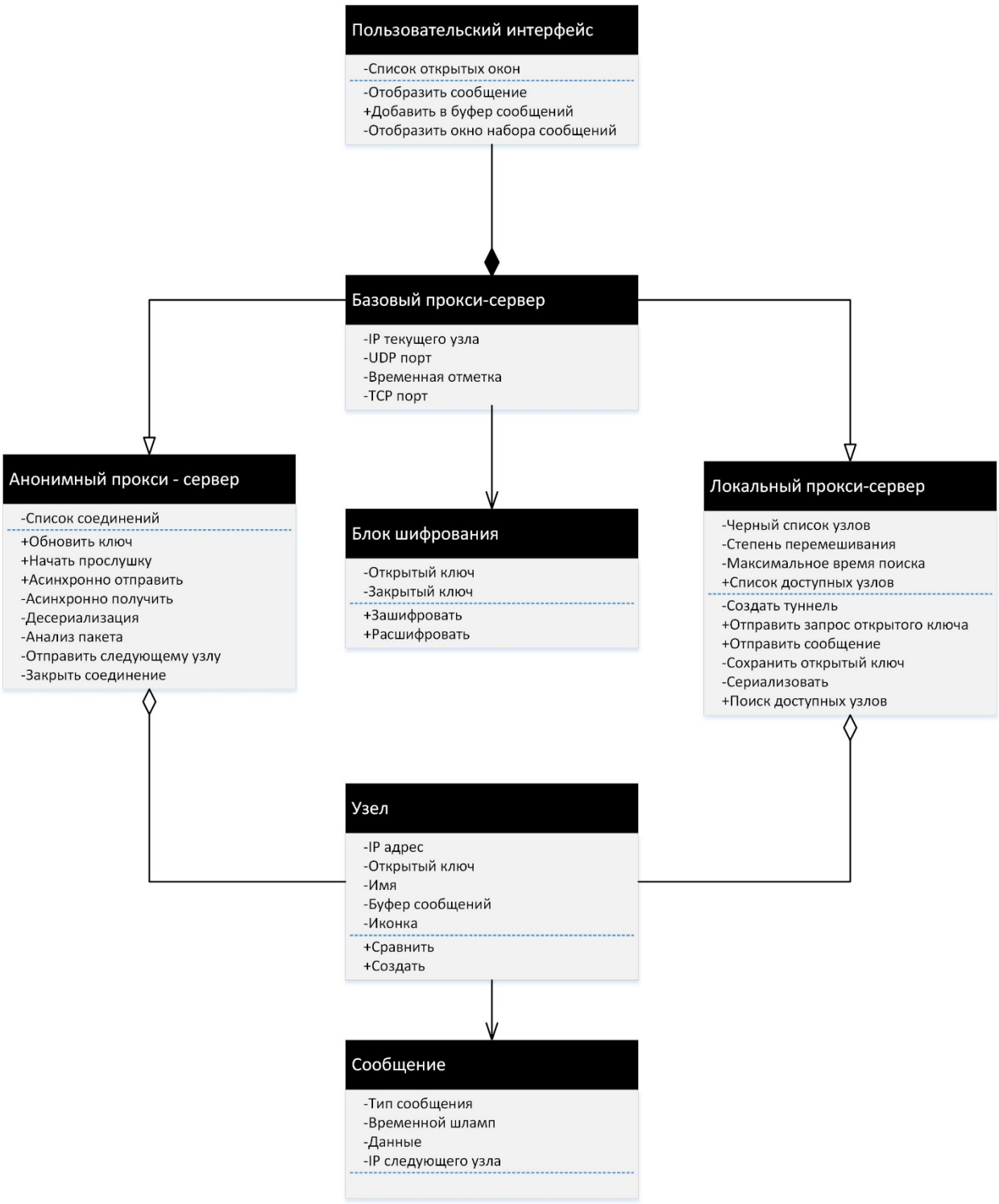

Рисунок 3.1 Диаграмма классов системы, реализующей анонимную сеть.

3.3 Структуры, реализующие анонимный прокси-сервер

Анонимный прокси-сервер в приложении реализован классом TCPserver. Сервер запускается при старте приложения и прослушивает заданный TCP порт.

Особенностью реализации анонимного прокси-сервера является асинхронное получение и передача сообщений. Асинхронная модель ввода-вывода устраняет необходимость в создании потоков и управления ими, что позволяет значительно упростить код и повышает эффективность ввода-вывода. При асинхронном вводе-выводе для обработки входящих данных и соединений используются методы обратного вызова[8].

Механизм асинхронного обратного вызова реализован в .NET Framework. С каждым .NET-приложением сопоставлен пул потоков. Когда у функции асинхронного ввода-вывода имеются готовые к обработке данные, поток из пула потоков .NET выполняет функцию обратного вызова. После завершения обратного вызова поток возвращается в пул.

Все асинхронные методы принимают объект состояния контекста, который может представлять любые данные. В текущей реализации контекст составляется из тела сообщения. Диаграмма зависимостей методов класса TCPServer представлена на рисунке 3.2.

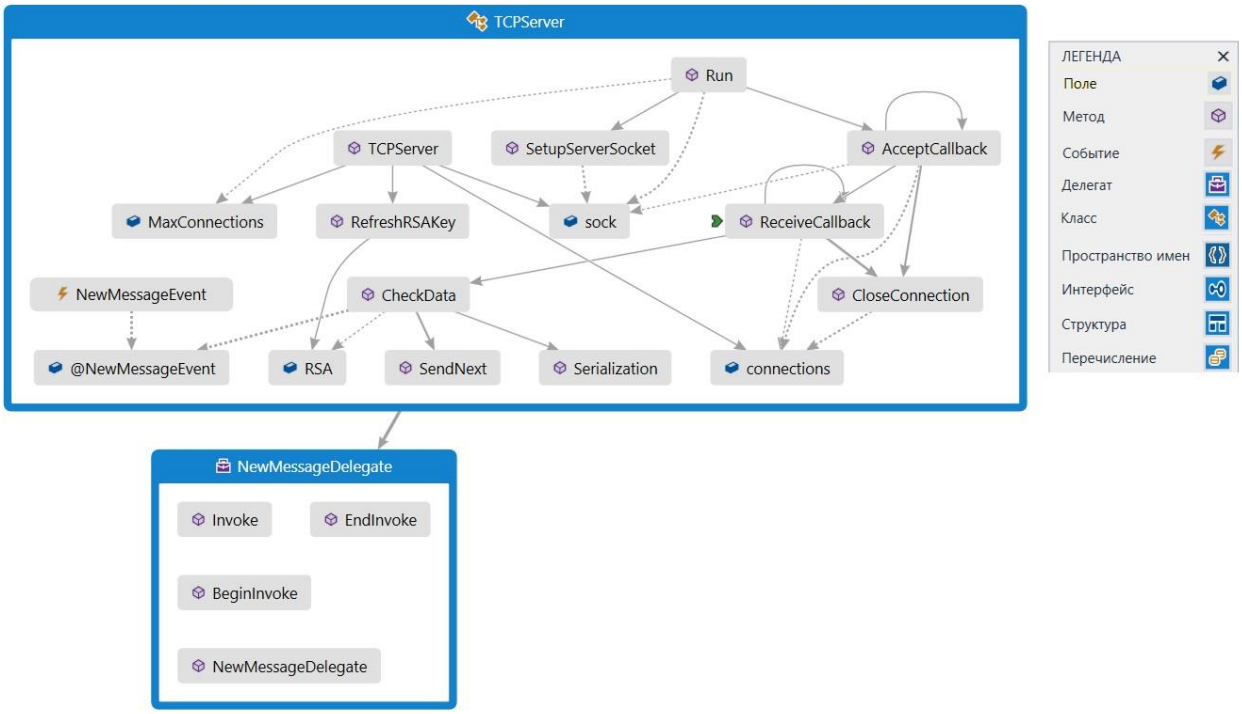

Рисунок 3.2 Диаграмма зависимостей методов класса TCPServer.

Пунктирной линей указано направление чтения данных, сплошной линией – запись данных

При получении нового сообщения типа FINALLY, сервер генерирует событие. На данное событие подписан интерфейс приложения, который обеспечивает отображение его на экран. Реализация основных методов класса представлена ниже:

```
    public class TCPServer: BasicClient
    {
        Socket sock = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
        List<ConnectionInfo> connections = new List<ConnectionInfo>();

        RSACryptoServiceProvider RSA;
        private void ReceiveCallback(IAsyncResult result)
        {
            ConnectionInfo connection =
                (ConnectionInfo)result.AsyncState;
            try
            {
                int bytesRead =
                    connection.Socket.EndReceive(result);
                if (0 != bytesRead)
                {
                    lock (connections)
                    {
```

```
                    //Здесь происходит обработка
                    CheckData(connection.Buffer);
                }
                connection.Socket.BeginReceive(
                    connection.Buffer, 0,
                    connection.Buffer.Length, SocketFlags.None,
                    new AsyncCallback(ReceiveCallback),
                    connection);
            }
            else CloseConnection(connection);
        }
        catch (SocketException exc)
        {
            CloseConnection(connection);
            MessageBox.Show ("Socket exception: " +
                exc.SocketErrorCode);
        }
    }
  private DataType.PackageLabel CheckData(byte[] Serialized)
    {

        NetworkMessage wst = (NetworkMessage)deserializer.Deserialize(BytesStream);

        if (wst.label == DataType.PackageLabel.final)
        {
            if (NewMessageEvent != null)
            {
                NewMessageEvent(wst, wst.NextIP);
            }
            return DataType.PackageLabel.final;
        }
        else if (wst.label == DataType.PackageLabel.transit)
        {
            SendNext(wst.NextIP, wst.Data);
            return DataType.PackageLabel.transit;
        }
        else if (wst.label == DataType.PackageLabel.getkey)
        {
             NetworkMessage msg = new DataType.NetworkMessage();
             msg.NextIP = CurrentIP;
             msg.label = DataType.PackageLabel.openkey;
             msg.TimeStamp = DateTime.Now;
             msg.Data = Encoding.UTF8.GetBytes(RSA.ToXmlString(false));
             SendNext(wst.NextIP, Serialization(msg));
            //Отправка ключа штатными средствами
            return DataType.PackageLabel.getkey;
        }
        else if (wst.label == DataType.PackageLabel.openkey)
        {
                if (NewMessageEvent != null)
                {
                    NewMessageEvent(wst, wst.NextIP);
                }
                return DataType.PackageLabel.final;
        }
        return DataType.PackageLabel.final;
    }

}
```

3.4 Реализация туннеля

Туннель реализуется на стороне отправителя. Метод, организующий построение туннеля входит в класс TCPClient, обеспечивающий работу локального прокси-сервера.

Особенностью реализации туннеля является многократная сериализация. Каждый набор данных для транзитного узла представляет собой сериализованный объект для следующего узла. Данный механизм очень удобен и обеспечивает универсальность обработки пакетов всех этапов передачи по туннелю. Метод, реализующий туннель, представлен ниже:

```
        public void SendString(String str, IPAddress IP)
        {
            Random rnd = new Random();

            /*Разметка приемочного пакета*/
            NetworkMessage Sendmsg = new NetworkMessage();
            Sendmsg.label = PackageLabel.final;     //Финальный пакет
            Sendmsg.TimeStamp = DateTime.Now;       //Будет иметь временной штамп,
            Sendmsg.NextIP = CurrentIP;             //IP отправителя (то есть текущий)
            Encoding Utf8 = Encoding.UTF8;
            Sendmsg.Data = Utf8.GetBytes(str);      //и данные.

            /*Этот пакет должен иметь "транзитника последней мили"
             чтобы адес отправителя всегда был равен нужному приемнику*/

            NetworkMessage Buffer = new NetworkMessage();
            Buffer.Data = Serialization(Sendmsg);
            Buffer.TimeStamp = DateTime.Now;
            Buffer.label = PackageLabel.transit;
            Buffer.NextIP = IP;
            Sendmsg = Buffer;
            for (int i = 0; i < layernum-1; i++)
            {
                int rand = rnd.Next(workstations.Count);
                RSACryptoServiceProvider RSAProvider = new RSACryptoServiceProvider();
                RSAProvider.FromXmlString(workstations[rand].publickey);
                Buffer.Data = Serialization(Sendmsg);
                Buffer.Data = RSAProvider.Encrypt(Serialization(Sendmsg), false);
                Buffer.TimeStamp = DateTime.Now;
                Buffer.label = PackageLabel.transit;
                Buffer.NextIP = workstations[rand].IPAddr;
                Sendmsg = Buffer;
            }


            SendToServer(workstations[rnd.Next(workstations.Count)].IPAddr,
Serialization(Sendmsg));
        }
```

3.5 Реализация локального прокси-сервера

Локальный прокси-сервер отвечает за подготовку сообщения к передаче, отображение сообщения пользователю, шифрование и сериализацию. Локальный прокси-сервер также отвечает за обработку события получения сообщения. В этом случае сервер предоставляет клиенту расшифрованные данные для отображения. Диаграмма зависимостей методов класса представлена на рисунке 3.3.

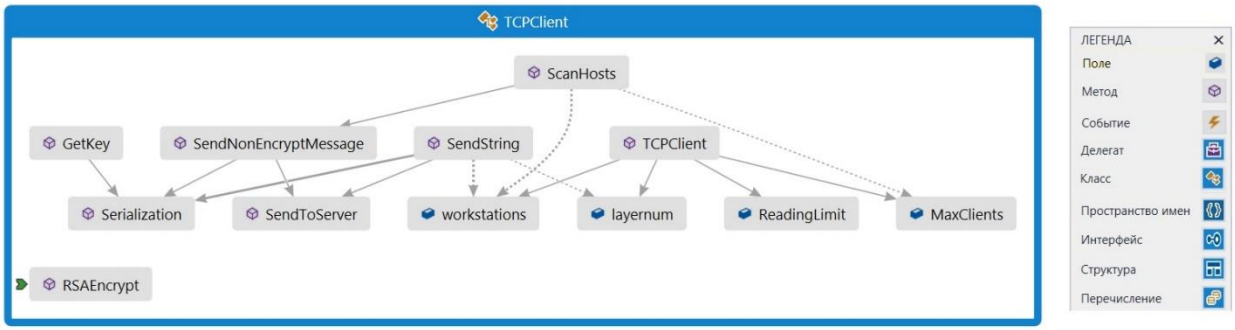

Рисунок 3.3 Диаграмма зависимостей методов класса TCPClient.

Пунктирной линией показано направление чтения данных, сплошной линией – запись данных

Также локальный сервер отвечает за поиск новых узлов. Функция, обеспечивающая поиск, представлена ниже:

```
        public void ScanHosts() {
            //workstations = new List<Workstation>();

            UdpClient listener = new UdpClient(UDPport);
            IPEndPoint groupEP = new IPEndPoint(IPAddress.Parse("192.168.88.255"),
UDPport);
            DateTime startTime = DateTime.Now;
            DateTime TimeStep = startTime;
            try
            {
                while (TimeStep.Millisecond - startTime.Millisecond < TimeLimit)
                {
                    byte[] bytes = listener.Receive(ref groupEP);
                    String[] IP = groupEP.ToString().Split(new Char[] {':'});

                    Workstation NeoFit = new Workstation(IPAddress.Parse(IP[0]), null);
                    NeoFit.name = Encoding.UTF8.GetString(bytes);
                    if (!workstations.Contains(NeoFit, new
                              WorkstationEqualityComparer()) && Trusted(NeoFit))
                    {
                        workstations.Add(NeoFit);
```

```
                }
                TimeStep = DateTime.Now;
            }

            foreach (Workstation w in workstations)
            {
                NetworkMessage Message = new NetworkMessage();
                Message.Data = null;
                Message.label = PackageLabel.getkey;
                Message.TimeStamp = DateTime.Now;
                Message.NextIP = CurrentIP;
                SendNonEncryptMessage(Message, w.IPAddr);
            }
        }
        catch (Exception e)
        {
            Console.WriteLine(e.ToString());
        }
        finally
        {
            listener.Close();
        }
    }
```

3.6 Входные данные и управляющие параметры

Диапазоны входных данные и управляющие параметры представлены в таблице 3.1.

Таблица 3.1. Входные данные и управляющие параметры.

Данные	Тип данных	Диапазон значений	Примечание
Текст передаваемого сообщения	Byte[]	[0..1024]	В качестве ограничения была принята максимальная целесообразная длина сообщения
Степень перемешивания	Integer	[3..150], По умолчанию: 3	По результатам тестирования
Время обновление списка доступных узлов	Integer	Более 10 секунд По умолчанию: 60 секунд	Время указывается в секундах. Зависит от максимального времени поиска узлов
Порт широковещательной рассылки	Integer	По умолчанию 19999	По правилам, определенным протоколом UDP
Порт TCP	Integer	По умолчанию 1459	По правилам, определенным протоколом TCP
Тип пакета	Enum	{FINALLY, TRANSIT, FAKE}	Назначается автоматически

3.7 Тестирование приложения

Для тестирования приложения была развернута виртуальная локальная сеть на базе виртуальной платформы VMware. Трудность тестирования сетевого приложения заключается в невозможности отследить поведение других элементов в сети в зависимости от работы конкретного узла. В рамках решения задач данной работы было проведено локальное функциональное тестирование, интеграционное системное тестирование в виртуальной локальной сети и нагрузочное тестирование.

Локальное функциональное тестирование выполнялось вручную и с помощью автоматизированных тестов. Автоматизированные тесты предназначены для контроля над разработкой и позволяют быстро искать логические ошибки в методах классов. Автоматизированные тесты запускаются после каждого исправления кода. Разработанное приложение не подлежит объемному автоматизированному тестированию из-за сложности архитектуры и сетевой природы приложения. В текущей реализации автоматически тестируются методы шифрования, методы, разделяющие данные на блоки для шифрования и методы сериализации. Модуль тестирования со списком тестов представлен на рисунке 3.4.

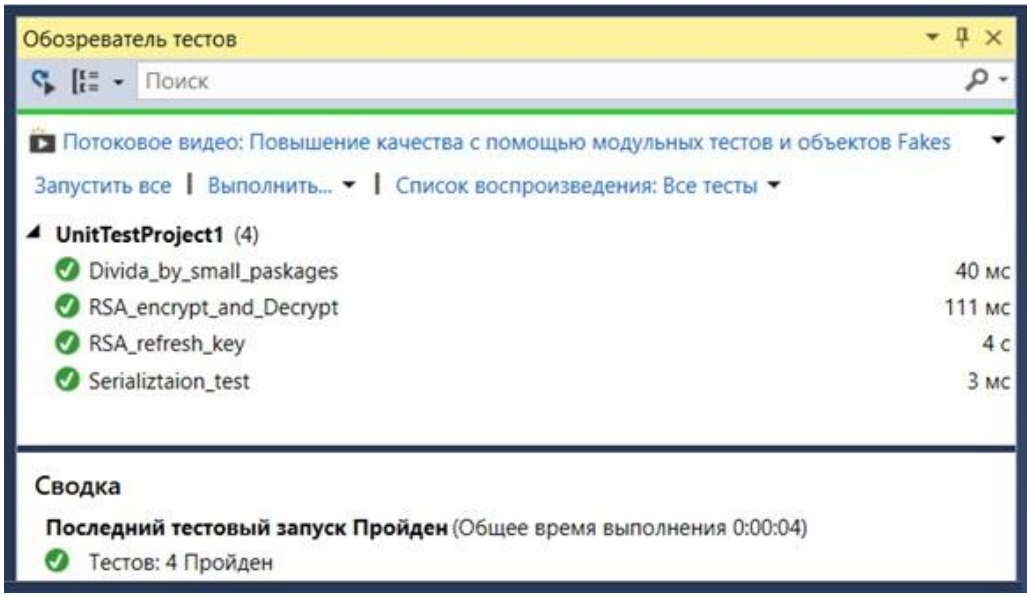

Рисунок 3.4. Список пройденных тестов в окне обозревателя.

В рамках интеграционного тестирования был организован перехват сообщений локальной сети с помощью сниффера Wireshark. Был проанализирован UDP сигнал от рабочих станций. UDP сигнал в данной реализации не зашифрован. Перехваченный UDP-пакет с указанием маски рассылки и источника рассылки представлен на рисунке 3.5.

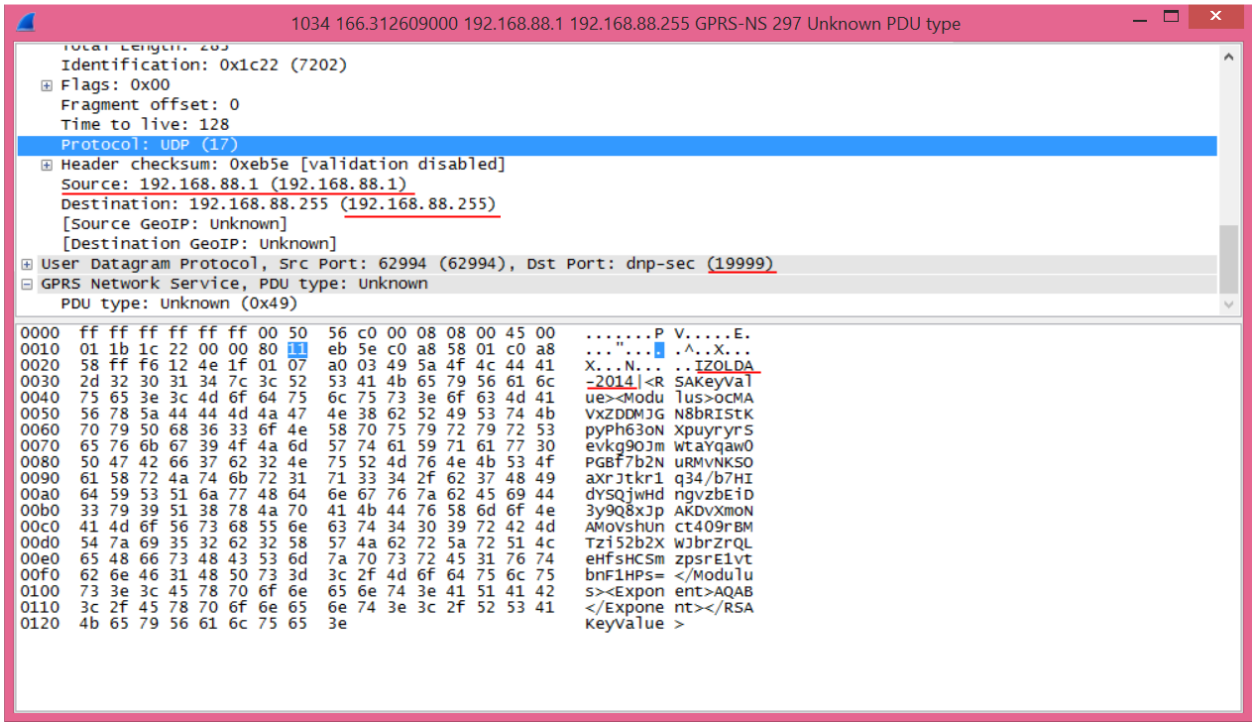

Рисунок 3.5. Перехват UDP пакета с помощью сниффера Wireshark.

TCP соединение, через которое передается анонимная информация, принимает зашифрованные сериализованные объекты. При перехвате трафика нельзя сказать точно, какой пакет отправлен. Зашифрованный TCP-пакет представлен на рисунке 3.6.

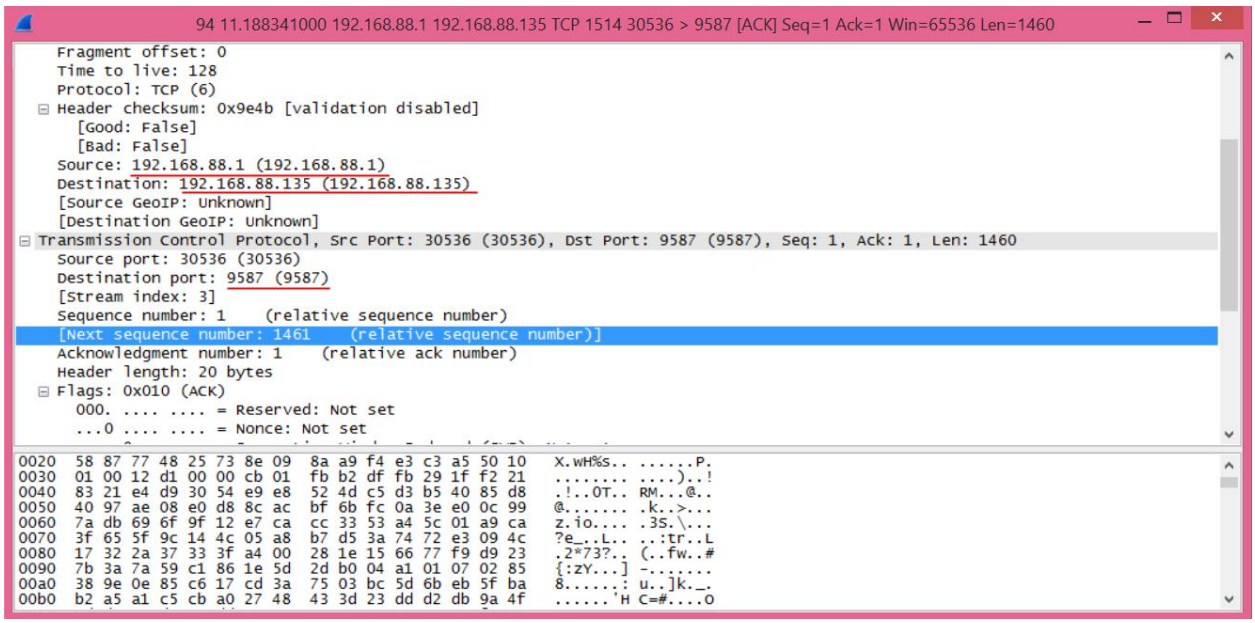

Рисунок 3.6. TCP – пакет с зашифрованным трафиком.

Нагрузочное тестирование приложения позволило определить предельные возможности по приему и отправке сообщений. Функция отправки сообщения используется в локальном и анонимном прокси-серверах и является универсальной. Поэтому данная функция буферизует запросы на отправку штатными средствами языка C#. В результате эксперимента удалось установить, что одновременно можно отправить 210 сообщений. Увеличить число одновременно отправляемых сообщений можно за счет реализации асинхронного вызова функции, но в рамках данной работы в этом нет необходимости.

Анонимный прокси-сервер реализован с использованием асинхронных функций .NET Framework. Это позволило принять одновременно более 2000 соединений. Эффективность асинхронного подхода рассматривается в работе [7].

4. Экспериментальный раздел

4.1 Метрики анонимности

Для сравнения различных способов обеспечения анонимности и проверки эффективности разработанной сети, необходимо определить количественные характеристики анонимности.

Одной из очевидных количественных характеристик является количество сообщений, участвующих в смешивании, но этого недостаточно, поскольку все сообщения могут быть отправлены с одного узла. Тогда источник будет скомпрометирован.

Для определения степени анонимности требуется знать мощность анонимного множества[9]. Если сообщение может быть отправлено только одним узлом, мощность такого множества равна единице, что означает отсутствие анонимности. Если же все узлы сети могут отправлять и принимать сообщения, то мощность анонимного множества будет равна количеству узлов.

В настоящее время исследование мощности различных элементов сети недостаточно для ее оценки. Поскольку работа анонимной сети подчиняется вероятностным распределениям, необходимо рассуждать об энтропии, как о количественной оценке неопределенности сети [10].

Пусть, по модели злоумышленника множество всех пользователей сети равно Ψ, пусть пользователю может принадлежать роль $r \in R$, где $R = \{отправитель, получатель\}$ в отношении сообщения M. Пусть $u \subseteq \Psi$ - пользователь, которому обязательно назначена роль, а $U : \Psi \times R \to [0.1]$ - отображение декартова произведения множества пользователей на множество ролей в отношении сообщения M. В приведённой модели имеет место вероятность распределения возможных отправителей и получателей. Если для всех пользователей вероятность принадлежности к роли одинакова, то такая

система может называться системой с постоянной вероятностью и считается безопасной с точки зрения анонимности.

Пусть S – множество вершин с нулевой вероятности роли. Тогда эффективный размер этого множества будет равен энтропии распределения:

$$S = -\sum_{u \in \Psi} p_u \log_2(p_u), \text{ где} \qquad (3)$$

$$p_u = U(u, r). \qquad (4)$$

Можно интерпретировать этот результат, как количество дополнительной информации, которая нужна злоумышленнику для идентификации пользователя u с ролью r для конкретного сообщения *M*. В случае единственного пользователя вероятность p = 1 (S=0), злоумышленник имеет достаточно информации для идентификации пользователя:

1. Если S = 0, то канал коммуникации не анонимный
2. S всегда принадлежит интервалу $[0.\log_2|\Psi|]$
3. Для всех моделей злоумышленника, в которых $S = \log_2|\Psi|$, сеть считается полностью анонимной.

Так как в анонимной сети может быть реализовано несколько уровней перемешивания, энтропия в таких сетях может быть получена путем суммирования энтропий каждого уровня. Предложенные методы оценки анонимности носят приближенный характер.

4.2 Оценка анонимности сети по степени перемешивания узлов.

В рамках данной квалификационной работы были проведены эксперименты по изменению времени доставки пакетов в зависимости от степени перемешивания узлов. Данные изменения проводились с целью

определить оптимальную степень перемешивания, позволяющую обеспечить высокий уровень анонимности и в то же время работать достаточно быстро.

В соответствии с указанной степенью перемешивания сообщений, была определена суммарная энтропия сети по формуле (3). Результаты измерений и расчетов представлены в таблице 4.1.

Таблица 4.1. Зависимость времени передачи сообщения от степени перемешивания узлов

Степень перемешивания узлов	Суммарная энтропия, S	Среднее время передачи сообщения, мс
2	0,301	507
3	1,204	574
4	2,334	633
5	3,612	741
6	5,0	860
7	6,475	1070
8	8,023	1082
9	9,633	1593
10	11,297	1604

В качестве параметра по умолчанию была выбрана степень перемешивания, равная трем. При этом энтропия равна 1,204, что можно интерпретировать, как невозможность определить, какой узел точно является источником, какой узел является приемником, а какой узел выполняет исключительно транзитные функции.

Также невозможно определить, какие узлы и в каком порядке входят в туннель. При степени перемешивания, равной трем, среднее время доставки

сообщения равно 574 миллисекундам, что можно считать допустимым для программ обмена сообщениями. График изменения энтропии и график изменения времени доставки сообщения представлен на рисунке 4.1.

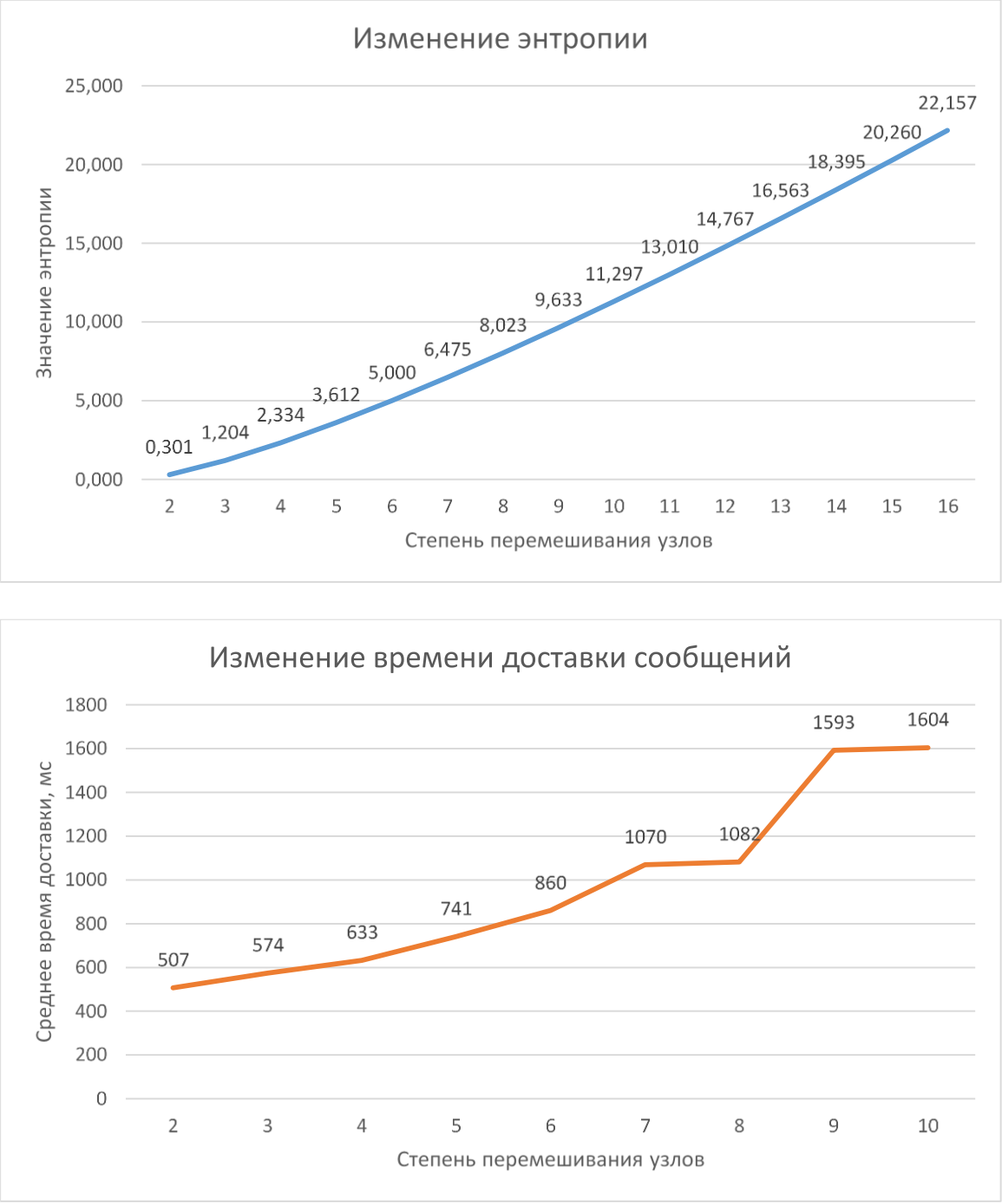

Рисунок 4.1. График зависимости суммарной энтропии и времени доставки сообщений от степени перемешивания узлов сети.

5. Организационно – экономическая часть

5.1 Исходные данные для расчета стоимости разработки ПО

С позиции рынка и рыночных отношений программный продукт (ПП) является товаром с особой инфраструктурой. Для оценки целесообразности реализации программного продукта, требуется оценить его себестоимость, а также объем временных и трудовых затрат.

Организация и планирование процесса разработки предусматривает следующие работы:

- формирование состава выполняемых работ, и группировка их по стадиям разработки;
- расчет трудоемкости выполнения работ;
- установление профессионального состава и расчет количества исполнителей;
- определение продолжительности выполнения отдельных этапов разработки;
- построение календарного графика выполнения разработки;
- контроль выполнения календарного графика.

На предпроектном этапе создания программного продукта требуется определить цену в отсутствие достоверной информации. Целью такого расчета будет принятие управленческого решения о целесообразности разработки.

Реализуемый программный продукт в рамках данной курсовой работы имеет действующие аналоги, поэтому данную реализацию можно отнести к группе новизны В.

Реализация программного продукта предусматривает решение задач стандартными методами с учетом существующих программных библиотек. Таким образом, уровень сложности алгоритма можно оценить, как 3.

Так как алгоритмы, реализуемые на клиентской и серверной части предусматривают преимущественно шифрование разных типов, а также передачу шифрованных данных разного объема, можно определить тип задачи программного продукта, как задачи расчетного характера.

Исходные данные для расчета стоимости разработки ПО указаны в таблице 5.1.

Таблица 5.1. Исходные данные для расчета стоимости разработки ПО.

№ п/п	Наименование	Значение	Примечание
1	Степень сложности алгоритма разработки	3	Исходя из ТЗ
2	Группа новизны	В	-«-
3	Тип задач	Расчетного характера	-«-
4	Количество наборов данных переменной информации	4	Определено разработчиком
5	Количество наборов данных нормативно – справочной информации	4	-«-
6	Количество наборов данных баз данных	2	-«-
7	Степень сложности входной информации	12	-«-
8	Степень сложности выходной информации	21	-«-

5.2 Расчет трудоемкости разработки.

Трудоемкость разработки программной продукции $\tau_{ПП}$ может быть определена как сумма величин трудоемкости выполнения отдельных стадий разработки ПП из выражения:

$$\tau_{ПП} = \tau_{ТЗ} + \tau_{ЭП} + \tau_{ТП} + \tau_{РП} + \tau_{В}, \text{ где} \qquad (5)$$

$\tau_{ТЗ}$ – трудоемкость разработки технического задания на создание ПП;

$\tau_{ЭП}$ – трудоемкость разработки эскизного проекта ПП;

$\tau_{ТП}$ – трудоемкость разработки технического проекта ПП;

$\tau_{РП}$ – трудоемкость разработки рабочего проекта ПП;

$\tau_{В}$ - трудоемкость внедрения разработанного ПП.

Трудоемкость разработки технического задания рассчитывается по формуле:

$$\tau_{ТЗ} = T_{РЗ}^{3} + T_{РП}^{3}, \text{ где} \qquad (6)$$

Значения величин $T_{РЗ}^{3}$ и $T_{РП}^{3}$ рассчитываются по формулам

$$T_{РЗ}^{3} = t_{З} \cdot K_{РЗ}^{3}; \qquad (7)$$

$$T_{РП}^{3} = t_{З} \cdot K_{РП}^{3}, \qquad (8)$$

Значения величин, указанных в формулах, представлены в таблице 5.2.

Таблица 5.2. Затраты времени разработчиков на разработку технического задания.

$T_{РЗ}^{3}$	30,55	затраты времени разработчика постановки задач на разработку ТЗ, чел.-дни;
$T_{РП}^{3}$	16,45	затраты времени разработчика программного обеспечения на разработку ТЗ, чел.-дни.
$t_{З}$	47	норма времени на разработку ТЗ на программный продукт в зависимости от функционального назначения и степени новизны разрабатываемого ПП, чел.-дни;
$K_{РЗ}^{3}$	0,65	коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком постановки задач на стадии ТЗ;
$K_{РП}^{3}$	0,35	коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком программного обеспечения на стадии ТЗ.

$$\tau_{ТЗ} = 47 \cdot (0,65 + 0,35) = 47 \text{ [чел.-дн.]}$$

Аналогично рассчитывается трудоемкость эскизного проекта ПП $\tau_{ЭП}$:

$$\tau_{ЭП} = T_{РЗ}^{Э} + T_{РП}^{Э} \quad (9)$$

$$T_{РЗ}^{Э} = t_{Э} \cdot K_{РЗ}^{Э} \quad (10)$$

$$T_{РП}^{Э} = t_{Э} \cdot K_{РП}^{Э} \quad (11)$$

Результаты расчетов и исходные данные представлены в таблице 5.3.

Таблица 5.3 Затраты времени на разработку эскизного проекта(ЭП)

$T_{РЗ}^{Э}$	46,9	затраты времени разработчика постановки задач на разработку ЭП, чел.-дни;
$T_{РП}^{Э}$	20,1	затраты времени разработчика программного обеспечения на разработку ЭП, чел.-дни.
$t_{Э}$	67	норма времени на разработку эскизного задания, чел-дни
$K_{РЗ}^{Э}$	0,7	коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком постановки на стадии ЭП;
$K_{РП}^{Э}$	0,3	коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком программного обеспечения на стадии ЭП.

$$\tau_{ЭП} = 67 \cdot (0,7 + 0,3) = 67 \text{ [чел.-дн.]}$$

Трудоемкость разработки технического проекта $\tau_{ТП}$ зависит от функционального назначения ПП, количества разновидностей форм входной и выходной информации и определяется как сумма времени, затраченного разработчиком постановки задач и разработчиком программного обеспечения, т.е.

$$\tau_{ТП} = (t_{РЗ}^{Т} + t_{РП}^{Т}) \cdot K_{В} \cdot K_{Р}, \text{ где} \quad (12)$$

$K_{В}$ – коэффициент учета вида используемой информации. Значение коэффициента $K_{В}$ определяется из выражения:

$$K_{В} = (K_{П} \cdot n_{П} + K_{НС} \cdot n_{НС} + K_{Б} \cdot n_{Б}) / (n_{П} + n_{НС} + n_{Б}) \quad (13)$$

Результаты расчетов и исходные данные представлены в таблице 5.4.

Таблица 5.4. Затраты времени на разработку технического проекта (ТП)

$t_{РЗ}^{Т}$	14	норма времени, затрачиваемого на разработку ТП разработчиком постановки задач, чел.-дни;
$t_{РП}^{Т}$	12	норма времени, затрачиваемого на разработку ТП разработчиком программного обеспечения, чел – дни;
$K_В$	1,104	коэффициент учета вида используемой информации;
$K_Р$	1,26	коэффициент учета режима обработки информации.
$K_П$,	1	значение коэффициента учета вида используемой информации для переменной информации
$K_{НС}$	0,72	значение коэффициента учета вида используемой информации для нормативно-справочной информации
$K_Б$	2,08	значение коэффициента учета вида используемой информации для баз данных;
$n_П$	4	количество наборов данных переменной информации
$n_{НС}$	4	количество наборов данных нормативно-справочной информации
$n_Б$	2	количество наборов данных баз данных

$$\tau_{ТП} = (14 + 12) \cdot 1{,}104 \cdot 1{,}26 = 36 \text{ [чел.-дн.]}$$

Трудоемкость разработки рабочего проекта $\tau_{РП}$ зависит от функционального назначения ПП, количества разновидностей форм входной и выходной информации, сложности алгоритма функционирования, сложности контроля информации, степени использования готовых программных модулей, уровня алгоритмического языка программирования и определяется по формуле:

$$\tau_{РП} = K_К \cdot K_Р \cdot K_Я \cdot K_З \cdot K_{ИА} \cdot (t_{РЗ}^{Р} + t_{РП}^{Р}), \text{ где} \qquad (14)$$

$K_{ИА}$ – коэффициент учета вида используемой информации и сложности алгоритма ПП;

Значение коэффициента $K_{ИА}$ определяется из выражения

$$K_{ИА} = (K_П' \cdot n_П + K_{НС}' \cdot n_{НС} + K_Б' \cdot n_Б) / (n_П + n_{НС} + n_Б) \qquad (15)$$

В таблице 5.5 представлены исходные значения коэффициентов для расчёта трудоемкости разработки рабочего проекта.

Таблица 5.5. Расчет трудоемкости разработки рабочего проекта (РП)

$K_К$	1,08	коэффициент учета сложности контроля информации;
$K_Р$	1,32	коэффициент учета режима обработки информации
$K_Я$	1	коэффициент учета уровня используемого алгоритмического языка программирования;
$K_З$	0,6	коэффициент учета степени использования готовых программных модулей;
$K_{ИА}$	0,672	коэффициент учета вида используемой информации и сложности алгоритма ПП;
$t_{РЗ}^{Р}$	22	норма времени, затраченного на разработку РП на алгоритмическом языке высокого уровня разработчиком постановки задач, чел.-дни.
$t_{РП}^{Р}$	87	норма времени, затраченного на разработку РП на алгоритмическом языке высокого уровня разработчиком программного обеспечения соответственно, чел.-дни.
$K_П$`	1	значения коэффициентов учета сложности алгоритма ПП и вида используемой информации для переменной информации
$K_{НС}$`	0,48	значения коэффициентов учета сложности алгоритма ПП и вида используемой информации для нормативно-справочной информации
$K_Б$`	0,4	значения коэффициентов учета сложности алгоритма ПП и вида используемой информации для баз данных

$$\tau_{РП} = (22 + 87) * 1{,}08 * 1.32* 1 * 0.6 * 0.672=63 \text{ [чел.-дн.]}$$

Трудоемкость выполнения стадии внедрения $\tau_В$ может быть рассчитана по формуле:

$$\tau_В = (t_{РЗ}^{В} + t_{РП}^{В}) \cdot K_К \cdot K_Р \cdot K_З. \qquad (16)$$

Параметры формулы представлены в таблице 5.6.

Таблица 5.6. Значения коэффициентов для расчета трудоемкости проекта на стадии внедрения

$t_{РЗ}^{В}$	8	норма времени, затрачиваемого разработчиком постановки задач
$t_{РП}^{В}$	24	разработчиком программного обеспечения соответственно на выполнение процедур внедрения ПП, чел.-дни.
$K_Р$	1,21	коэффициент учета режима обработки информации. для стадии внедрения

$$\tau_В = (8 + 24) \cdot 1 \cdot 1{,}21 \cdot 0{,}6 = 25 \text{ [чел.-дн.]}$$

Подставляя полученные данные в исходную формулу получим:

$$\tau_{ПП} = 24 + 67 + 36 + 63 + 25 = 214 \text{ [чел.-дн.]}$$

Распределение трудоемкости по стадиям разработки проекта представлено в таблице 5.7.

Таблица 5.7. Трудоемкости по стадиям разработки проекта.

Этап	Трудоемкость этапа	№ работы	Содержание работы	Трудоемкость чел-дн.
1 (ТЗ)	47	1	Постановка задачи	30
		2	Выбор средств проектирования, разработки и СУБД	17
2 (ЭП)	67	3	Разработка структурной схемы системы	20
		4	Разработка структуры базы данных	20
		5	Разработка алгоритмов доступа к данным	15
		6	Разработка алгоритмов решения частных задач	12
ТП	36	7	Реализация алгоритмов доступа к данным	10
		8	Реализация алгоритмов решения частных задач	17
		9	Разработка пользовательского интерфейса	3
		10	Реализация пользовательского интерфейса	6
РП	63	11	Отладка и тестирование всего комплекса информационной среды	25
		12	Исправление ошибок и недочетов	15
		13	Разработка документации к системе	15
		14	Итоговое тестирование системы	8
4 (В)	25	15	Установка и настройка ПП	25
Всего	238 чел-дн			238 чел-дн

5.3 Расчет количества исполнителей

Средняя численность исполнителей при реализации проекта разработки и внедрения ПО определяется соотношением:

$$N = \frac{Q_p}{F}$$,где: (17)

Q_p - затраты труда на выполнение проекта,

F - фонд рабочего времени.

Величина фонда рабочего времени определяется соотношением:

$$F = T * F_m$$, где (18)

T - время выполнения проекта в месяцах, равное 6 месяцам;

F_M - фонд времени в текущем месяце, который рассчитывается из учета общества числа дней в году, числа выходных и праздничных дней:

$$F_м = \frac{t_p \cdot (D_K - D_B - D_П)}{12}$$, где (19)

t_p - продолжительность рабочего дня;

D_K - общее число дней в году;

D_B - число выходных дней в году;

$D_П$ - число праздничных дней в году.

$$F_м = \frac{t_p \cdot (D_K - D_B - D_П)}{12} = \frac{8 \cdot (365 - 103 - 13)}{12} = 166 \text{[часы]}$$

$F = T \cdot F_M = 4 \cdot 166 = 664$ [часы] – количество рабочих часов

Затраты труда оценим исходя из квалификации сотрудников:

Q = 237*2.3*1.15 *2 = 1259 [чел-часы]

$N = \frac{Q}{F} = \frac{1259}{664} = 1.89 \approx 2$ - число исполнителей проекта.

5.4 Календарный план-график разработки ПП

Планирование и контроль хода выполнения разработки проводится по календарному графику выполнения работ. Для построения календарного графика требуется распределить трудоемкость операций на каждой стадии по исполнителям. Распределение трудоемкости представлено в таблице 5.8.

Таблица 5.8. Планирование разработки по стадиям с учетом трудоемкости работ.

Стадия разработки	Трудоемкость, чел-дн.	Должность исполнителя	Распределение трудоемкости, чел – дн.	Численность
1.Разработка технического задания	47	Ведущий инженер Программист	30 17	1 1
2.Разработка эскизного проекта	67	Ведущий инженер Программист	45 22	1 1
3.Разработка технического проекта	36	Ведущий инженер Программист	25 11	1 1
3.Разработка рабочего проекта	63	Ведущий инженер Программист	33 30	1 1
5.Внедрение	25	Ведущий инженер Программист	10 15	1 1
Итого:	238			2

Календарный план – график выполнения работ представлен на рисунке 5.1.

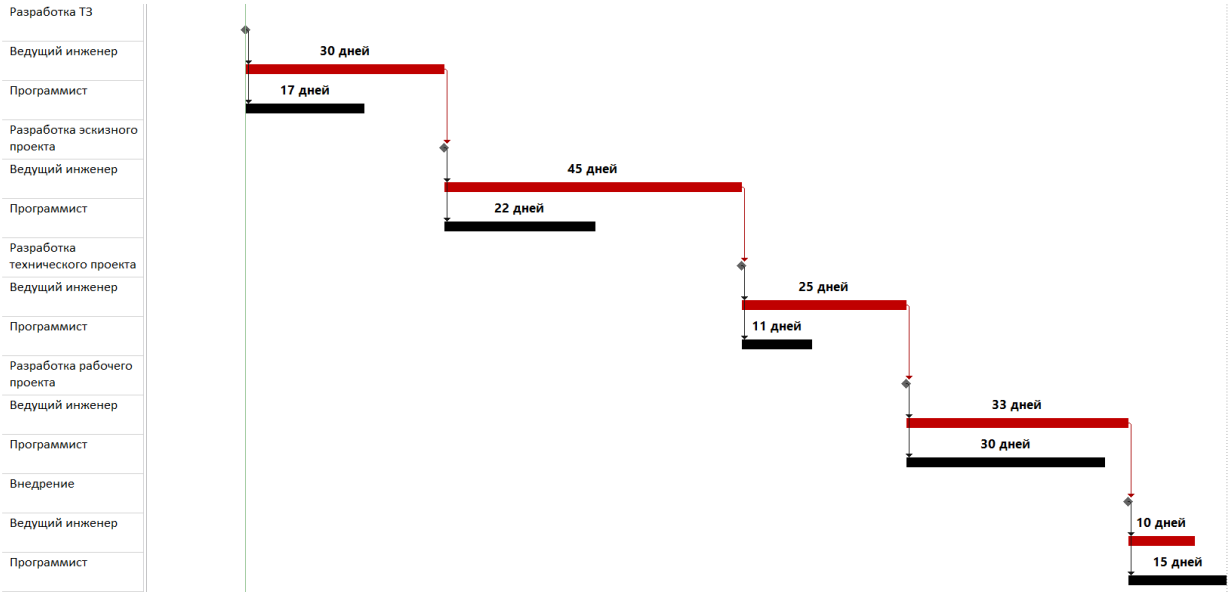

Рисунок 5.1. Календарный план – график выполнения работ

Вывод: при использовании оптимизации в результате параллельной работы ведущего инженера и программиста можно добиться сокращения срока разработки и внедрения программного продукта с 238 дней до 148 дней, т. е. в 1,6 раз.

5.5 Определение цены программной продукции

Затраты на выполнение проекта состоят из затрат на заработную плату исполнителям, затрат на закупку или аренду оборудования, затрат на организацию рабочих мест, и затрат на накладные расходы.

В таблице 5.9 приведены затраты на заработную плату и отчисления в пенсионный фонд (26%) и фонд обязательного медицинского страхования (5,1%) и фонд социального страхования (2,9%).

Таблица 5.9. Расчет заработной платы и других отчислений

Расчетные показатели	Должность – ведущий инженер	Должность – программист	ИТОГО
Квалификационный разряд	13	11	-
Оклад с учетом МРОТ	7800	6600	-
Число рабочих дней месяца	20	20	-
Заработная плата за день (расчётное)	390	330	-
Число фактически отработанных дней	143	95	238
Заработная плата	55770	31350	87120
Коэффициент премии	1	1	-
Премия	55770	31350	87120
Отчисления в пенсионный фонд	-	-	45302,4
Отчисления в ФОМС			8886,2
Отчисления в ФСС			5053,0

В рамках данной курсовой работы нет необходимости производить расчет амортизационных отчислений. Расходы на материалы, необходимые для разработки программной продукции, указаны в таблице 5.10.

Таблица 5.10. Материальные затраты.

Наименование	Ед. изм	Кол-во	Стоимость за ед.	Цена
Flash USB Drive	Шт.	1	300	300
Бумага А4	Пачка.	1	120	120
Картридж для струйного принтера HP Z602	Шт.	1	900	900
Канцелярские товар	Шт.	8	150	1200
Диск	Шт.	10	50	500
Компьютерная мебель	Шт.	2	7000	14000
Персональный компьютер	Шт.	1	35000	35000
Принтер	Шт.	1	3500	3500
Мышь 4TECH	Шт.	1	500	500
ИТОГО				56 020

Обобщенно все затраты представлены в таблице 5.11.

Таблица 5.11. Затраты на создание программного продукта.

Наименование	Стоимость, руб.
Зарплата основная	87 120
Зарплата дополнительная	87 120
Материальные затраты	56 020
Отчисления в ПФ	45302
ФОМС	8886
ФСС	5053,0
ИТОГО	289501
Плановая прибыль	100 000
Цена реализации	389 501

5.6 Расчет экономической эффективности

Расчет экономической эффективности проведем, предполагая цену разработанной программной продукции 262 619 рублей.

Основными показателями экономической эффективности является чистый дисконтированный доход (ЧДД) и срок окупаемости вложенных средств.

Чистый дисконтированный доход определяется по формуле:

$$ЧДД = \sum_{t=0}^{T}(R_t - З_t) * \frac{1}{(1+E)^t}, \qquad (20)$$

где T – горизонт расчета по месяцам;

t – период расчета;

R_t – результат, достигнутый на t шаге (стоимость);

$З_t$ – затраты;

E – приемлемая для инвестора норма прибыли на вложенный капитал.

Коэффициент *E* установим равным ставке рефинансирования ЦБ РФ – 8,25% годовых (или 0,69 % в месяц). В результате анализа рынка программной продукции, аналогичной разрабатываемой, планируется единовременная продажа всего программного комплекса. Коэффициент дисконтирования равен 1/(1 + E) = 0.993

В таблице 5.12 приведен расчет ЧДД по месяцам работы над проектом.

Таблица 5.12. Чистый дисконтированный доход по месяцам.

	Текущие затраты	Доход	ЧДД
Февраль	105225,00	0,00	-104172,75
Март	46069,00	0,00	-148283,25
Апрель	46069,00	0,00	-191501,12
Май	46069,00	0,00	-233839,81
Июнь	46069,00	389501,00	95099,00

График изменения ЧДД представлен на рисунке 5.2.

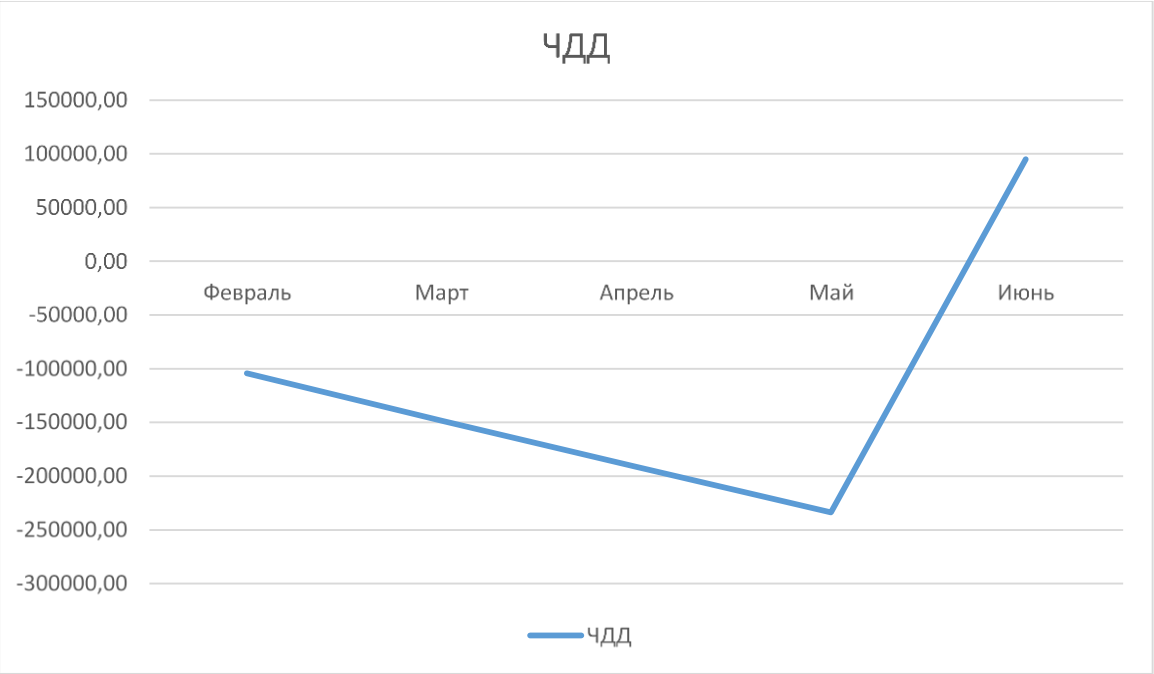

Рисунок 5.2. Чистый дисконтированный доход по месяцам, график

Выводы

В рамках данной работы был представлен расчет трудоемкости и стоимости реализации программного продукта. Таким образом, затраты на реализацию продукции с учетом амортизационных отчислений и материальных затрат составят 289501 руб. Чистый дисконтированный доход через пять месяцев составит 95099 руб.

6. Безопасность жизнедеятельности и охрана труда.

6.1 Анализ опасных и вредных факторов при разработке программного обеспечения и мероприятия по их устранению

Разработка программного обеспечения ведется с использованием персональных электронно-вычислительных машин (ПЭВМ). Работа с ПЭВМ связана с рядом вредных и опасных факторов, таких как статическое электричество, излучение в широких спектрах, электромагнитные поля, блики и отраженный свет, ультрафиолетовое излучение.

В данном разделе рассматриваются вредные и опасные факторы, влияющие на здоровье работников при разработке программного обеспечения, а также определяются требования к помещению, в котором будет вестись разработка. Требования к помещениям сформулированы в соответствии с санитарно-эпидемиологическими правилами и нормативами СанПиН 2.2.2/2.4.1340-03.

6.1.1 Общие требования к помещениям для работы с ПЭВМ

Помещения для эксплуатации ПЭВМ должны соответствовать нормам безопасности для деятельности и здоровья рабочих. Естественное и искусственное должно соответствовать требованиям и сопровождаться расчетной документацией. В рамках реализации данного курсового проекта используются ПЭВМ с плоским дискретным (жидкокристаллическим) экраном, что определяет площадь на 1 рабочее место 4,5 м2. Нормы коэффициентов отражения для этих помещений указаны в таблице 6.1:

Таблица 6.1. Нормы коэффициентов отражения для помещений с ПЭВМ

Для потолка	0,7-0,8
Для стен	0,5-0,6
Для пола	0,3-0,5

Помещения для работы с ПЭВМ должны быть оборудованы защитным заземлением (занулением) в соответствии с техническими требованиями.

6.1.2 Требования к микроклимату

Помещение разработчиков относится к производственным помещениям, в которых работа с использованием ПЭВМ является основной и связана с нервно-эмоциональным напряжением. Такие помещения согласно СанПиН должны обеспечиваться оптимальными параметрами микроклимата для категории работ 1а. Оптимальные параметры микроклимата указаны в таблице 6.2.

Таблица 6.2. Оптимальные параметры микроклимата во всех типах учебных и дошкольных помещений с использованием ПЭВМ

Температура воздуха, ^{o}C	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
19-21	40-60	< 0,1
22-24	40-60	< 0,1
23-25	40-60	< 0,1

В помещениях, оборудованных ПЭВМ должна проводиться ежедневная влажная уборка и систематическое проветривание. Содержание вредных химических веществ в воздухе производственных помещений с ПЭВМ не должно превышать предельно допустимых концентраций в соответствии с действующими гигиеническими нормативами.

6.1.3 Требования к уровням шума и вибрации

В производственных помещениях при выполнении основных или вспомогательных работ с использованием ПЭВМ уровни шума на рабочих местах не должны превышать предельно допустимых значений, установленных для данных видов работ в соответствии с действующими санитарно-эпидемиологическими нормативами. Шумящее оборудование (печатающие устройства, серверы и т.п.), уровни шума которого превышают нормативные, должно размещаться вне помещений с ПЭВМ. Допустимые значения уровней звукового давления в октавных полосах частот даны в таблице 6.3.

Таблица 6.3. Допустимые значения уровней звукового давления в октавных полосах частот и уровня звука, создаваемого ПЭВМ

Уровни звукового давления в октавных полосах со среднегеометрическими частотами									Уровни звука в дБА
31,5 Гц	63 Гц	125 Гц	250 Гц	500 Гц	1000 Гц	2000 Гц	4000 Гц	8000 Гц	
86 дБ	71 дБ	61 дБ	54 дБ	49 дБ	45 дБ	42 дБ	40 дБ	38 дБ	50

При выполнении работ с использованием ПЭВМ в производственных помещениях уровень вибрации не должен превышать допустимых значений вибрации для рабочих мест (категория 3, тип "в") в соответствии с действующими санитарно-эпидемиологическими нормативами СНиП 2.2.4/2.1.8.566—96.

6.1.4 Требования к освещению

Наиболее важным условием эффективной работы разработчиков ПО является соблюдение оптимальных параметров системы освещения в рабочих помещениях. Естественное освещение осуществляется через светопроемы, ориентированные в основном на север и северо-восток.

Искусственное освещение в помещениях для эксплуатации ПЭВМ должно осуществляться системой общего равномерного освещения. Освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300 - 500 лк. Освещение не должно создавать бликов на поверхности экрана. Освещенность поверхности экрана не должна быть более 300 лк.

Параметры освещенности представлены в таблице 6.4:

Таблица 6.4. Требования к освещениюна рабочих местах, оборудованных ПЭВМ

Освещенность на поверхности стола	300 - 500 лк.
Освещенность поверхности экрана	<300 лк.
яркость светящихся поверхностей (окна, светильники и др.), находящихся в поле зрения	<200 кд/м2.
Показатель ослепленности	<20
Яркость светильников общего освещения	<200 кд/м2
Коэффициент запаса (Кз)	1.4
Коэффициент пульсации	5%

6.1.5 Требования к уровням электромагнитных полей на рабочих местах, оборудованных ПЭВМ

Работник, для которого действия с ПЭВМ являются основными повергается воздействию электромагнитных полей, излучаемых корпусом ПЭВМ и устройствами отображения информации (ВДТ). Уровни электромагнитных полей нормируются по СанПиН 2.2.2/2.4.1340-03 и представлены в таблицах 6.5, 6.6:

Таблица 6.5. Временные допустимые уровни ЭМП, создаваемых ПЭВМ

Наименование параметров		ВДУ ЭМП
Напряженность электрического поля	в диапазоне частот 5 Гц - 2 кГц	25 В/м
	в диапазоне частот 2 кГц - 400 кГц	2,5 В/м
Плотность магнитного потока	в диапазоне частот 5 Гц - 2 кГц	250 нТл
	в диапазоне частот 2 кГц - 400 кГц	25 нТл
Электростатический потенциал экрана видеомонитора		500 В

Таблица 6.6. Визуальные параметры ВДТ, контролируемые на рабочих местах

N п/п	Параметры	Допустимые значения
1	Яркость белого поля	Не менее 35 кд/кв. м
2	Неравномерность яркости рабочего поля	Не более +/- 20%
3	Контрастность (для монохромного режима)	Не менее 3:1
4	Временная нестабильность изображения (мелькания)	Не должна фиксироваться
5	Пространственная нестабильность изображения (дрожание)	Не более 2 x 1E(-4L), где L - проектное расстояние наблюдения, мм

6.2 Расчет системы искусственного освещения для помещения разработчиков.

В рамках реализации проекта был разработан план помещения, в котором реализуется разработка программного обеспечения с использованием ПЭВМ. Расчет освещения проводится с использованием программы DIALux 4.12.01. План помещения представлен на рисунке 6.1. Ведомость объектов помещения представлена в таблице 6.7.

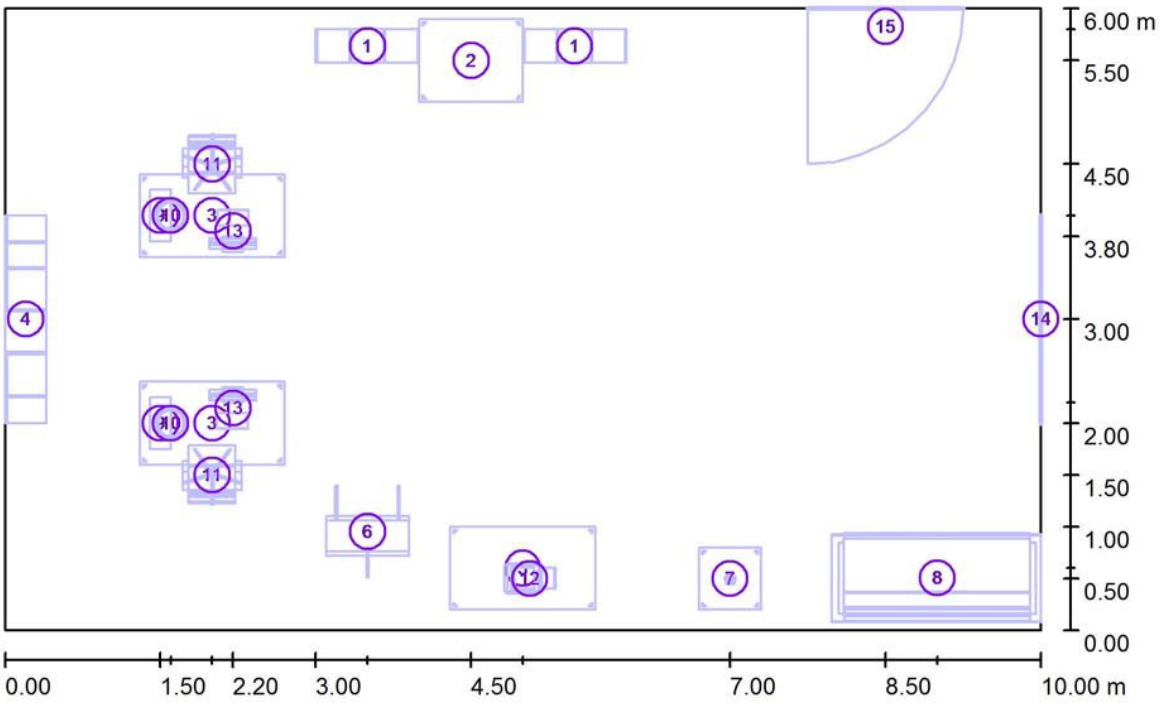

Рисунок 6.1. План помещения разработчиков с указанием офисных объектов.

Таблица 6.7. Ведомость объектов помещения офиса.

№ п/п	Количество	Объект
1	2	100*200 квадрат стеллаж
2	1	100*80 стол стандартный
3	3	140*80 стол для разработчика, стандартный
4	1	200*120 перегородка
5	1	60*80
6	1	Флипчарт
7	1	Горшечное растение
8	1	Диван широкий
9	2	Компьютер (системный блок)
10	2	Корзина для бумаг
11	2	Офисный стул
12	1	принтер
13	2	Экран и клавиатура
14	1	Окно
15	1	Дверь

Светотехнические результаты расчетов:

Общий световой поток: 74254 lm
Общая мощность: 1416.0 W
Коэффициент эксплуатации: 0.80
Краевая зона: 0.000 m

Поверхность	Средние освещенности [lx]			Коэффициент отражения [%]	Средние Яркость [cd/m²]
	Напрямую	Опосредовано	Всего		
Рабочая плоскость	169	384	553	/	/
Расчетные поверхности 1	192	488	680	/	/
Полы	132	306	438	30	42
Потолок	714	167	881	70	196
Стенка 1	65	276	340	50	54
Стенка 2	57	276	333	50	53
Стенка 3	58	269	327	50	52
Стенка 4	50	258	308	50	49

Равномерность на рабочей плоскости
E_{min} / E_{cp}: 0.383 (1:3)
E_{min} / E_{max}: 0.287 (1:3)

Удельная подсоединенная мощность: 23.60 W/m² = 4.27 W/m²/100 lx (Поверхность основания: 60.00 m²)

План помещения с указанием изолиний освещенности представлен на рисунке 6.2.

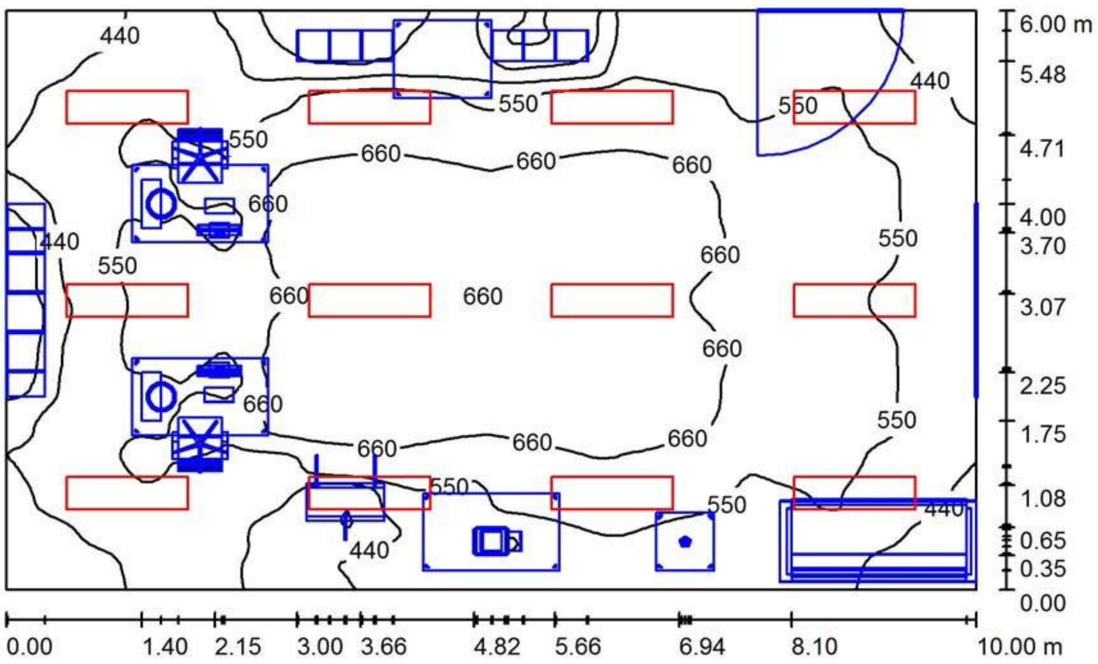

Рисунок 6.2. План помещения офиса с указанием изолиний освещенности.

Данные о помещении:

Высота помещения: 3.500 m,

Монтажная высота: 3.000 m,

Коэффициент эксплуатации: 0.80

Уровни освещенности поверхностей помещения представлены в таблице 6.8:

Таблица 6.8. Уровни освещенности поверхностей помещения.

Поверхность	Коэффициент поглощения	Ecp [lx]	Emin [lx]	Emax [lx]	Emin/Ecp
Полы	30	553	212	738	0.383
Потолок	70	438	108	638	0.247
Стенки (4)	50	881	287	2548	0.326
Рабочая плоскость: (0.800 m)		329	101	761	/

Ведомость светильников представлена в таблице 6.9:

Таблица 6.9. Ведомость светильников.

№	Шт.	Обозначение (Поправочный коэффициент)	E (Светильник) [lm]	E(Лампы) [lm]	P [W
1	12	DIAL 26 USN-ID 2/54W T16 EVG ASQ500 (1.000)	6188	8840	118.(
			Всего: 74254	Всего: 106080	1416 0

Удельная подсоединенная мощность: 23.60 W/m² = 4.27 W/m²/100 lx (Поверхность основания: 60.00 m²) Визуализация помещения и план помещения представлены на рисунке 6.3.

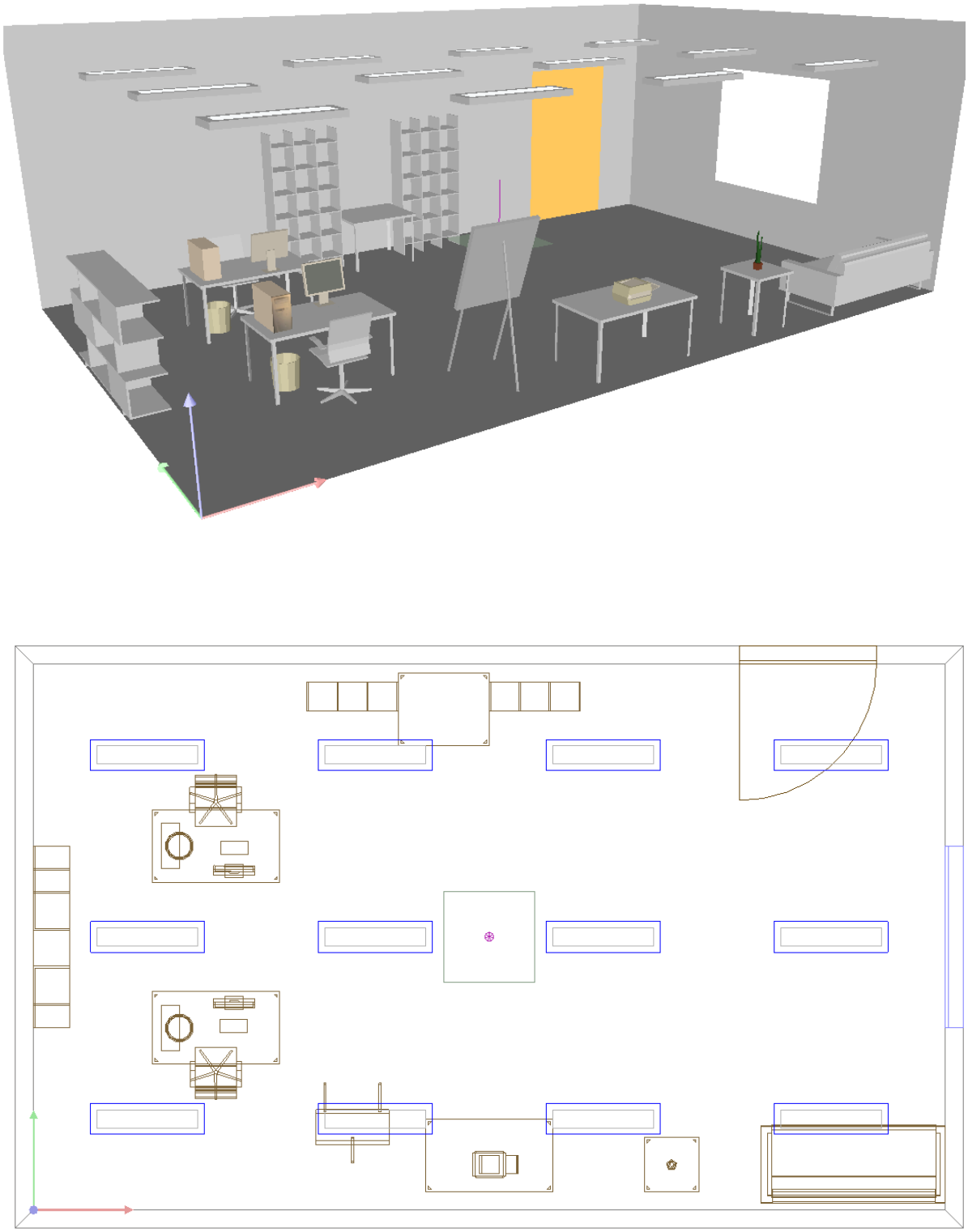

Рисунок 6.3. 3D визуализация и план помещения.

6.3 Расчет уровней шума и вибрации в помещении серверов.

Данная дипломная работа подразумевает использование серверов в качестве промежуточных маршрутизаторов сети. Рассматриваются источники шума, данные о которых соответствуют индивидуальному варианту и являются приблизительными. Схема расположения источников шума представлена на рисунке 6.4:

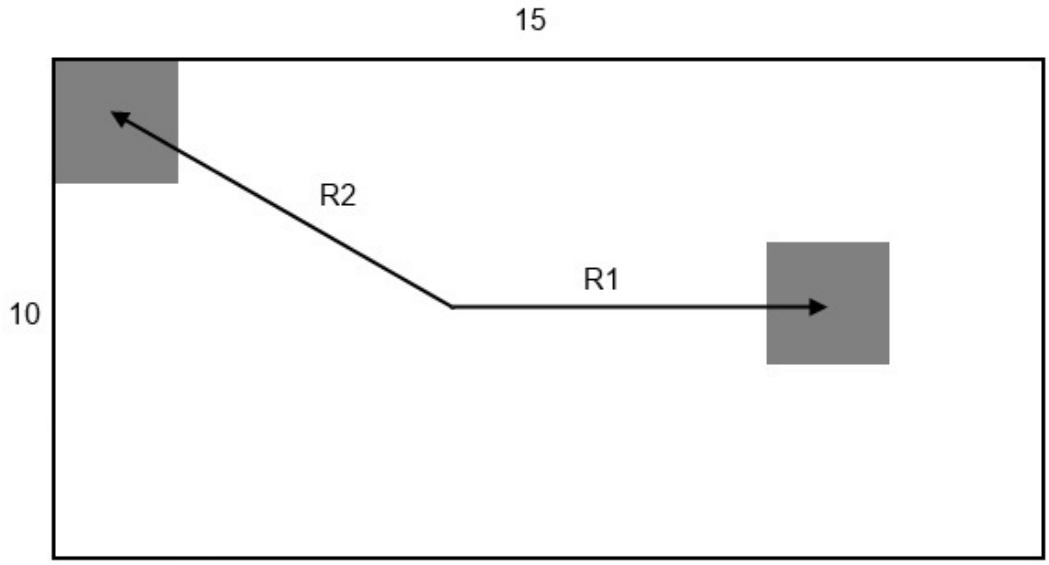

Рисунок 6.4. Расположение источников шума.

Уровни звуковой мощности каждого из источников шума представлены в таблице 6.10. Уровни звуковой интенсивности источников шума и расчеты даны в таблице 6.11.

Таблица 6.10. Уровни звуковой мощности источников шума.

№ источника	Расположение источника	Расстояние до источника	Уровни звуковой мощности, Lw								
			63	125	250	500	1000	2000	4000	8000	Площадь S
1	на полу	5	72	72	68	68	68	68	71	70	157
2	на полу	5	83	87	85	85	85	82	83	83	39,25

Таблица 6.11. Уровни звуковой интенсивности источников шума.

Параметры расчетов								
Октавные полосы	63	125	250	500	1000	2000	4000	8000
y	0,65	0,62	0,64	0,76	1	1,6	2,4	4,2
B	32,5	31	32	38	50	80	120	210

№ ист.	Уровни звуковой интенсивности, Lj							
	63	125	250	500	1000	2000	4000	8000
1	63,1209	63,31624	59,18494	58,47791	57,3636	55,51044	56,98821	54,05125
2	74,71886	78,88956	76,77472	76,16411	75,23161	70,77819	70,69459	69,48607

Параметры комнаты и расчетные коэффициенты представлены на рисунке 6.5. Предельный спектр уровня шума в помещении с серверами представлен на рисунке 6.6.

Габариты комнаты	Значение	Ед. изм.	
Длина	10	м	
Ширина	20	м	
Высота	5	м	
объем	1000	м3	
Прочие параметры			
Ф	1	-	Фактор направленности источника шума
x	1	-	Коэффициент влияния ближнего акустического поля
B1000	50	м2	Постоянная помещения
æ	1	-	Диффузность звукового поля в помещении

Рисунок 6.5. Габариты комнаты и расчётные коэффициенты.

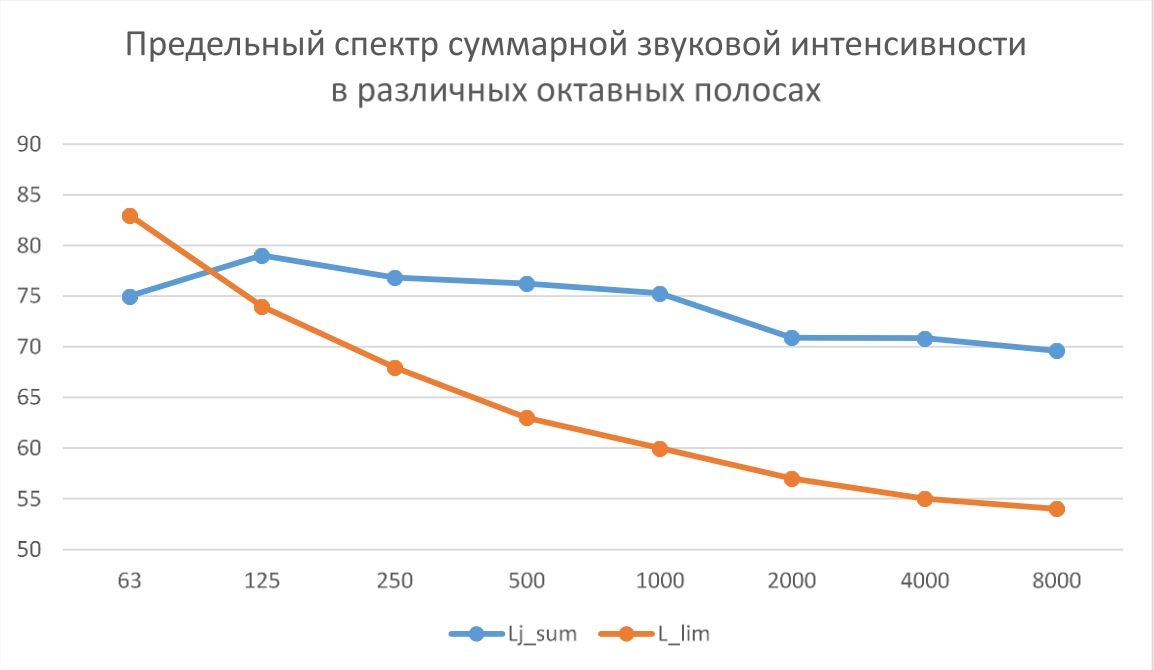

Рисунок 6.6.

Расчетный спектр превышает предельно допустимый уровень шума. Определено, что увеличение Ri - расстояния до источников шума слабо влияет на уровень шума. Изменение габаритов комнаты, однако, может в значительной мере повлиять на уровень шума в помещении, но по результатам расчетов также не позволяет снизить его до допустимых значений. Действенным защитным мероприятием в этом случае будет являться введение звукоизоляции, акустическая обработка помещения.

Помещение, в котором расположены сервера не соответствует нормам, установленным в СанПиН для помещений работниов ПЭВМ. Нахождение сервера и разработчика в одной комнате недопустимо.

Для снижения уровня шума требуется установка ограждающих конструкций, предназначенных для защиты от шума. Эффективными являются такие конструкции, однослойные с пустотами или из бетонов на пористых заполнителях и ячеистых бетонов, или однослойные конструкции с тонкой облицовкой и воздушным промежутком.

При невозможности организации ограждающих конструкций следует предусмотреть для машин звукопоглощающие кожухи и корпуса. Кожухи должны полностью закрывать оборудование. На внутренних поверхностях стенок кожухов следует предусматривать облицовку из звукопоглощающего материала.

Также можно применять звукопоглощающие экраны. Экраны следует применять для снижения уровня звукового давления на рабочих местах и в местах постоянного пребывания людей от источников шума.

Выводы.

В разделе сформулированы требования к помещению по основным пунктам, указанным в санитарно-эпидемиологических правилах и нормативах. На основании данных требований был построен план помещения с указанием позиций размещения офисных объектов. План составлен с учетом эргономических потребностей разработчиков, и соответствует условиям по безопасности помещений и охране труда.

Для составленного плана был проведен расчет освещения в программе DIALux. Установлены координаты расположения источников освещения, их общая мощность. Уровень освещенности в каждой точке помещения показан на плане помещения изолиниями.

В соответствии с индивидуальным вариантом проведен расчет уровней шума в помещении, оборудованном серверами. Установлен перечень октавных полос, в которых уровень шума превышает допустимые нормы. Были предложены средства снижения уровня шума и даны рекомендации по размещению группы разработчиков вне помещения с серверами.

Заключение

В ходе проделанной работы был осуществлен анализ предметной области, рассмотрены существующие подходы к обеспечению анонимности и методы их реализации. Были проанализированы проблемы сокрытия трафика на уровне узлов и на уровне пакетов, предложены пути их решения.

В соответствии с техническим заданием на квалификационную работу, разработана структура одноранговой анонимной сети, реализованы алгоритмы сокрытия данных в сети.

В соответствии с выбранной структурой было реализовано программное обеспечение, выполняющее поставленные перед ним задачи по созданию и поддержке функционирования анонимной одноранговой сети с зашифрованным трафиком. Программное обеспечение позволяет отправлять и получать сообщения по сети анонимно, скрывая факт обмена данными.

Список литературы

1. Chaum David., “Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms”/ David Chaum// Communications of the ACM.-1981.-№2
2. Marc Rennhard, Sandro Rafaeli, Laurent Mathy., “Design, Implementation, and Analysis of an Anonymity Network for Web Browsing” // Swiss Federal Institute of Technology, Computer Engineering and Networks Laboratory, Lancaster University, Faculty of Applied Sciences. Technical Report.-2002.-№129
3. Sapan Bhatia, Murtaza Motiwala, Vytautas Valancius and more. Hosting Virtual Networks on Commodity Hardware.// Princeton University, Georgia Institute of Technology,TLabs/TU Berlin.- 2008
4. Marc Rennhard, Sandro Rafaeli, Laurent Mathy and more. «An Architecture for an Anonymity Network»/Swiss Federal Institute of Technology, Computer Engineering and Networks Laboratory, Lancaster University, Faculty of Applied Sciences// Enabling Technologies: Infrastructure for Collaborative Enterprises, 2001. WET ICE 2001. Proceedings. Tenth IEEE International Workshops on .-2001.- p.165 – 170
5. Барычев С. Основы современной криптографии/С. Барычев, Р. Серов.-М.:2001.-152с.
6. Wiangsripanawan, Rungrat and Susilo, Willy and Safavi-Naini, Rei. Design Principles for Low Latency Anonymous Network Systems Secure Against Timing Attacks// Proceedings of the Fifth Australasian Symposium on ACSW Frontiers - Volume 68.- 2007.-pp 183—191
7. David M. Goldschlag, Michael G. Reed, and Paul F Syverson. "Hiding Routing Information," Workshop on Information Hiding, Cambridge, UK, May, 1996.
8. Дэрин Кили. «Winsock. Преимущества быстродействующих сокетов в .NET», [Электронный ресурс]//MSDN Magazine.-2014.-Режим доступа:

http://msdn.microsoft.com/ru-ru/library/dd335942.aspx .- Загл. с экрана (дата обращения 22.06.2014)

9. D. L. Chaum. The Dining Cryptographers Problem: Unconditional Sender and Receiver Untraceability. *Journal of Cryptology*, 1(1):66–75, 1988
10. George Danezis “Mix-networks with Restricted Routes”/ Privacy Enhancing Technologies.-2003
11. Andrei Serjantov and George Danezis. Towards an Information Theoretic Metric for Anonymity// Proceedings of the 2Nd International Conference on Privacy Enhancing Technologies.- 2003.-pp 41--53
12. Tor. Anonymity Online. [Электронный ресурс]//Официальный сайт.- Режим доступа: https://www.torproject.org/index.html.en .- Загл. с экрана (Дата обращения 22.06.2014)
13. I2P. Проект невидимый интернет. [Электронный ресурс]//Официальный сайт.-Режим доступа: https://geti2p.net/ru.-Загл. с экрана (Дата обращения 22.06.2014)

Приложение А. Руководство пользователя.

Установка программы

Программа поставляется в виде инсталлятора InstallShield для операционной системы Windows с предустановленным .NET Framework 4.5. Поскольку приложение выполняет роль клиента и сервера, нет необходимости развертывания дополнительного программного обеспечения. Для корректной работы программы также необходима настроенная локальная сеть. Окно инсталлятора представлено на рисунке А1.

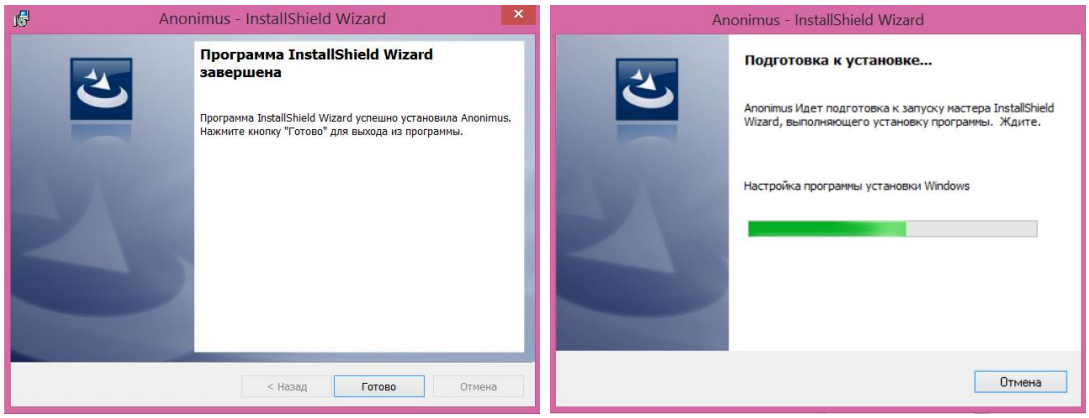

Рисунок А1. Окно установщика приложения

Передача сообщения

При запуске программы открывается окно со списком доступных доверенных узлов. Для передачи сообщения любому из узлов достаточно дважды щелкнуть по узлу в списке. При этом откроется окно чата с полем для ввода сообщения. После набора сообщения необходимо нажать кнопку «Отправить». Сформируется туннель и сообщение будет отправлено получателю. Текст сообщения с пометкой «Я» будет перенесен в окно чата. Окно со списком доверенных узлов и окно чата представлено на рисунке А2.

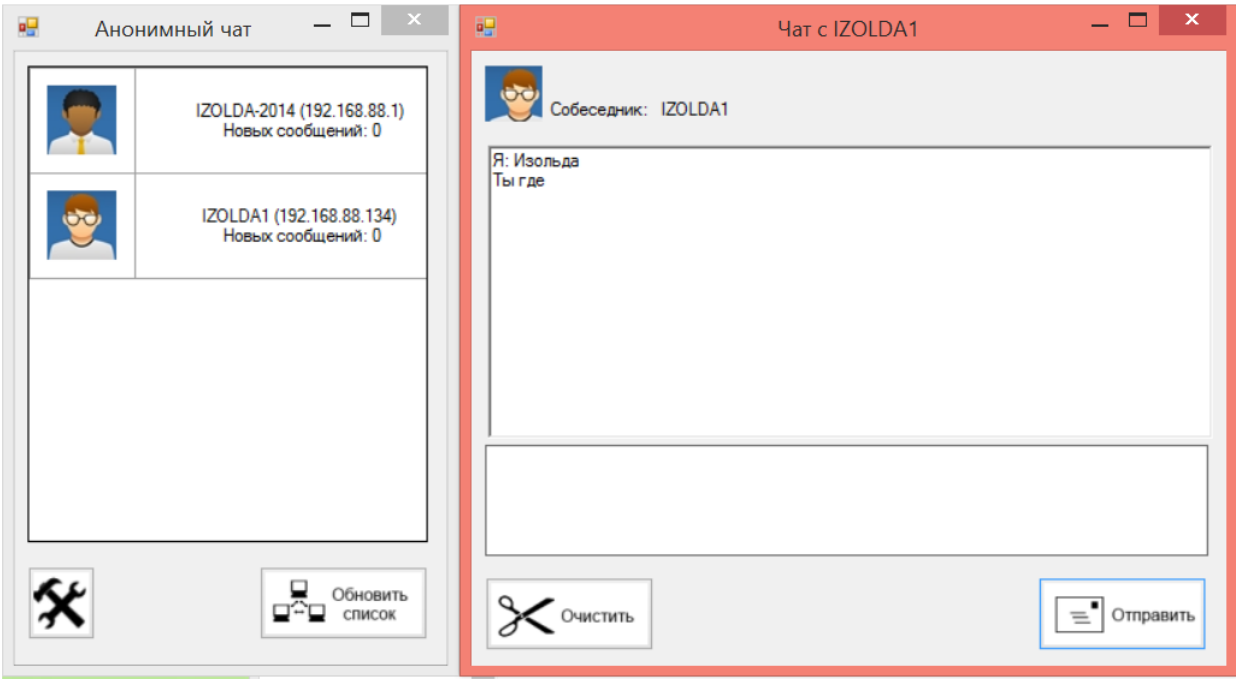

Рисунок А2. Окно со списком доверенных узлов и окно чата.

Получение сообщения

Если на стороне приемника сообщения уже открыто окно чата, соответствующее источнику, то сообщение отобразится в окне чата с пометкой имени отправителя. Если окно закрыто, то при следующем обновлении списка доступных узлов в таблице отобразится информация о новых сообщениях, пришедших с узла. Чтобы их прочесть, необходимо дважды щёлкнуть по имени узла в таблице. Откроется окно чата с отображением всех отправленных сообщений. Окно списка узлов с информацией о новых сообщениях представлено на рисунке А3.

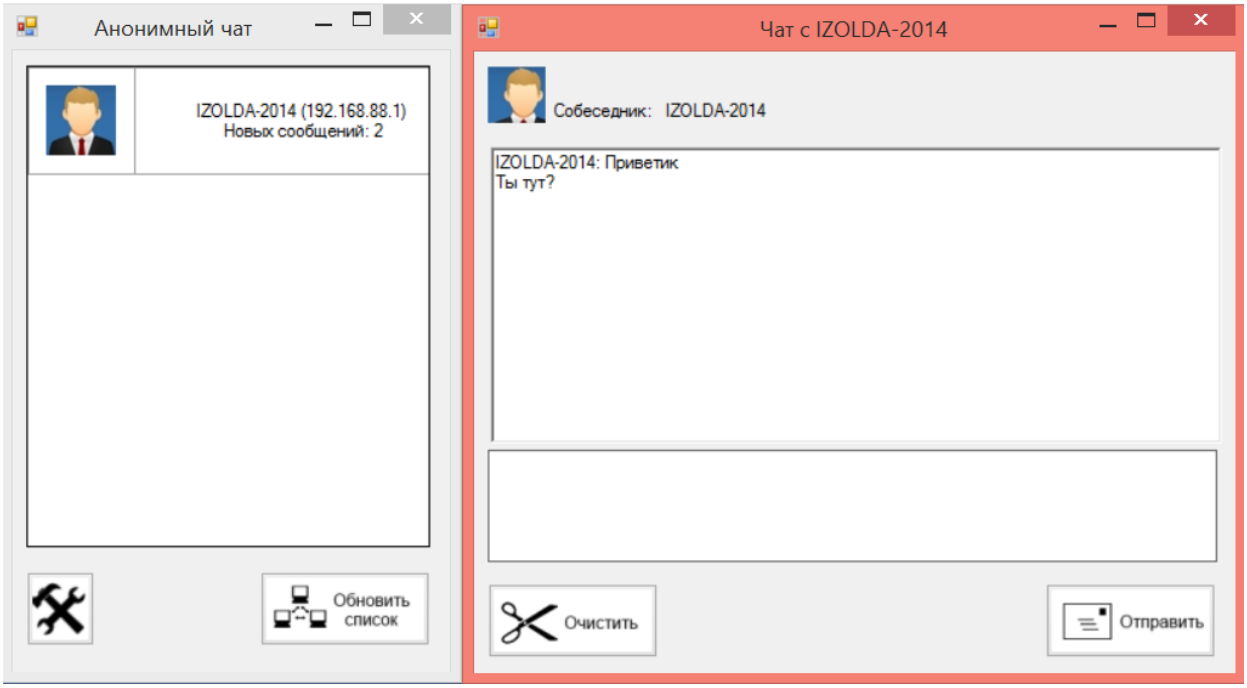

Рисунок А3. Окно списка узлов с информацией о новых сообщениях

Завершение работы и удаление программы

Для завершения работы программы достаточно закрыть приложение. При следующем обновлении списка доступных узлов в приложениях собеседников, завершивший работу узел не будет отображен.

Для удаления программы необходимо воспользоваться штатными средствами операционной системы Windows. Для этого необходимо выбрать в меню «Пуск» - «Панель управления», затем «Программы и компоненты», найти в списке программ приложение «Anonimus» и удалить его нажатием клавиши «Удалить». Служба «Программы и компоненты с установленным приложением представлена на рисунке А4.

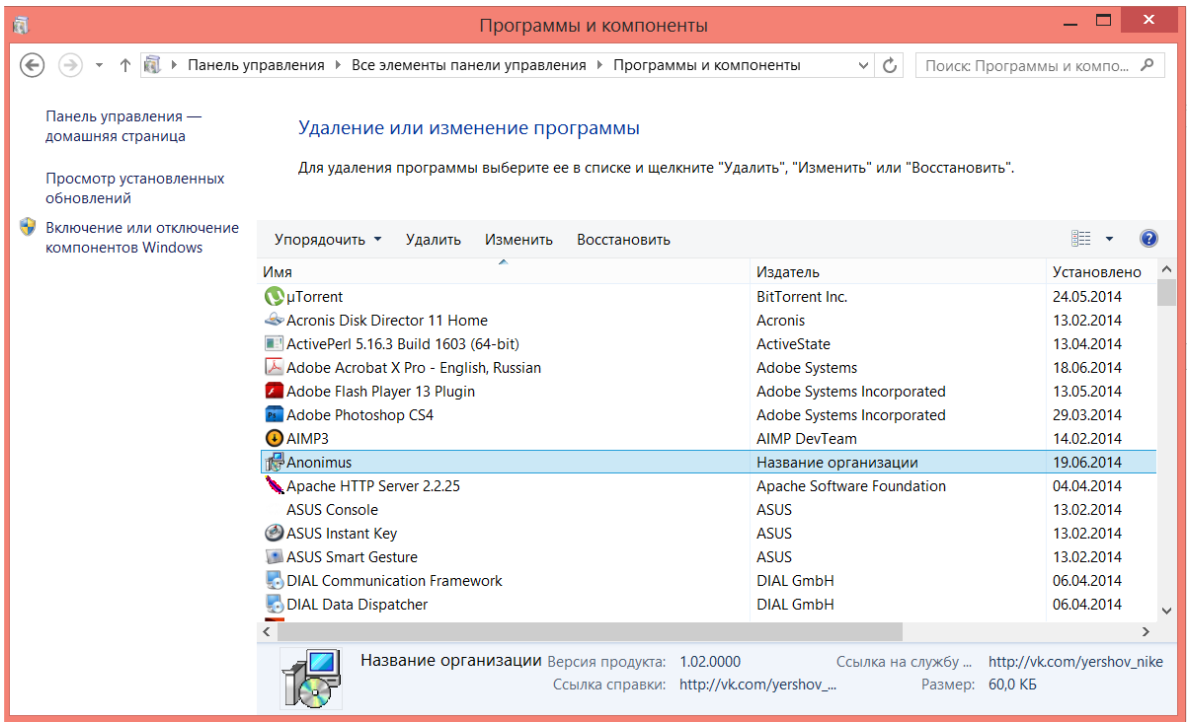

Рисунок А4. Окно системной службы «Программы и компоненты» с установленным приложением