МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра Информатики и программирования

СОЗДАНИЕ ПРИЛОЖЕНИЯ ДЛЯ БЕЗОПАСНОГО СОЕДИНЕНИЯ С СЕРВЕРАМИ: ИССЛЕДОВАНИЕ НА ПРИМЕРЕ SHADOWSOCKS

ОТЧЕТ О ПРАКТИКЕ

студента 3 курса 341 группы
направления Математическое обеспечение и администрирование
информационных систем
факультета КНиИТ
Филиппенко Дмитрий Александрович

Научный руководитель
Доцент _____ Кудрина Е.В.

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

Саратов 2024

СОДЕРЖАНИЕ

Цель и задачи

1. Исследовать протокол Shadowsocks, его механизм шифрования и основные принципы работы.
2. Провести сравнение Shadowsocks с другими протоколами безопасности.
3. Реализовать прототип приложения, обеспечивающего безопасное соединение с серверами.

Актуальность

С ростом интернет-угроз и потребностью в защите персональных данных пользователей, разработка методов безопасной передачи данных через интернет становится крайне важной задачей. Shadowsocks представляет собой мощный инструмент для защиты трафика, особенно в условиях высоких требований к приватности.

1 Теоретическая часть

1.1 Введение

Virtual Private Network (VPN, виртуальная частная сеть) — это технология, позволяющая создать зашифрованный канал связи между клиентом и сервером, обеспечивая конфиденциальность и целостность передаваемых данных. VPN применяется для обхода цензуры, защиты трафика в открытых сетях и скрытия реального IP-адреса пользователя.

Shadowsocks — это прокси-сервер с открытым исходным кодом, специально разработанный для обхода интернет-цензуры. Он поддерживает VPN-функциональность, обеспечивая шифрование и передачу данных через TCP и UDP. Shadowsocks отличается от традиционных VPN своей легковесностью, высокой скоростью и устойчивостью к блокировке. В этой работе подробно рассматривается, как Shadowsocks реализует сетевые функции и обеспечивает безопасность соединений.

1.2 Основные принципы работы протокола Shadowsocks

Shadowsocks был разработан китайским программистом под псевдонимом Clowwindy в 2012 году. Изначально протокол создавался для обхода интернет-цензуры, а также для защиты приватности пользователей. В отличие от классических VPN, Shadowsocks является высокоспециализированным прокси-протоколом, нацеленным на скрытие интернет-активности пользователей через зашифрованные туннели, что позволяет избежать блокировок и фильтрации трафика.

Shadowsocks работает на основе модели клиент-сервер. Клиентская программа Shadowsocks устанавливает соединение с сервером, расположенным за пределами заблокированной сети, и перенаправляет через него сетевой трафик. Схема взаимодействия построена на использовании протокола SOCKS5, который поддерживает различные типы трафика, включая HTTP, FTP и другие виды сетевых запросов. На стороне сервера Shadowsocks принимает зашифрованные пакеты данных от клиента, расшифровывает их и перенаправляет в интернет, а затем отправляет ответ обратно клиенту в зашифрованном виде.

Shadowsocks использует технику маскировки трафика, чтобы он выглядел как обычный HTTPS-трафик, что делает его менее заметным для

систем глубокого анализа пакетов (Deep Packet Inspection, DPI), которые применяются интернет-провайдерами и государственными органами для блокировки прокси-сервисов и VPN. Shadowsocks не устанавливает постоянное соединение, как это делает VPN, а вместо этого подстраивается под каждую сессию, что делает его менее уязвимым для анализа.

Shadowsocks поддерживает современные симметричные алгоритмы шифрования, включая AES (Advanced Encryption Standard) и ChaCha20. Эти алгоритмы обеспечивают высокий уровень безопасности и быстродействия.

— AES используется в режиме CBC (Cipher Block Chaining) или CTR (Counter Mode), что позволяет обеспечить защиту данных от различных атак, включая атаку повторения.
— ChaCha20 является альтернативой AES, особенно эффективной на устройствах с ограниченными вычислительными ресурсами, таких как мобильные устройства.

Эти методы шифрования позволяют шифровать каждый пакет данных до его отправки на сервер, что обеспечивает целостность данных и конфиденциальность передачи.

Управление трафиком и параметры конфигурации: Shadowsocks обладает гибкими настройками для конфигурирования уровня шифрования, скорости передачи и маскировки трафика. Пользователь может настроить Shadowsocks для работы на определенных портах, что позволяет обходить блокировки через изменение параметров соединения.

Обфускация (скрытие) и устойчивость к обнаружению: Shadowsocks применяет дополнительные методы обфускации, чтобы замаскировать трафик под обычный HTTP или HTTPS. Этот подход делает трафик Shadowsocks практически неотличимым от легитимного трафика и усложняет его обнаружение для DPI-систем. Shadowsocks способен адаптироваться к разным методам блокировки за счет динамической настройки соединения, что делает его стойким к анализу со стороны провайдеров и государственных органов.

Таким образом, Shadowsocks представляет собой мощный инструмент для безопасного и неприметного соединения с интернетом. Его архитектура и функции маскировки делают его устойчивым к методам блокировки,

а поддержка современных алгоритмов шифрования гарантирует высокую степень безопасности передаваемых данных. “‘

1.3 Обзор атак на Shadowsocks и защита от них

Shadowsocks был разработан как инструмент для обхода интернет-цензуры, однако его использование сталкивается с различными типами угроз. Понимание этих угроз и способов защиты от них является ключевым для успешного внедрения и эксплуатации протокола.

1.3.1 Угрозы и атаки

1. Анализ трафика (Traffic Analysis) Многие провайдеры и государственные органы используют анализ трафика для обнаружения использования прокси. Shadowsocks, несмотря на шифрование, оставляет характерные признаки, которые можно обнаружить. Пример атаки: обнаружение стандартных временных интервалов пакетов или постоянной скорости передачи данных.
2. Глубокий анализ пакетов (Deep Packet Inspection, DPI) DPI позволяет анализировать содержимое пакетов. Shadowsocks может быть идентифицирован, если пакеты не обфусцированы должным образом. Реальная угроза: блокировка Shadowsocks на уровне провайдера через DPI.
3. Brute Force и атаки на ключи Если используется слабый пароль или устаревший алгоритм шифрования, злоумышленники могут осуществить атаку перебором.
4. DDoS-атаки на серверы Направленные атаки, блокирующие сервер Shadowsocks, могут нарушить его работу.

1.3.2 Методы защиты Shadowsocks

— Обфускация трафика Для маскировки трафика Shadowsocks могут использоваться плагины, такие как obfs или v2ray-plugin. Эти инструменты изменяют заголовки пакетов, делая их похожими на HTTPS-трафик.

— Использование современных алгоритмов шифрования Рекомендуется использовать ChaCha20 или AES-256-GCM. Эти алгоритмы обеспечивают надежное шифрование и устойчивость к переборам.

— Ротация ключей и паролей Регулярная смена ключей шифрования предотвращает их возможный компромет.
— Динамические порты Постоянная смена портов делает обнаружение Shadowsocks сложнее.

1.4 Проблемы производительности и оптимизация Shadowsocks

1.4.1 Затраты ресурсов

1. Производительность процессора При использовании тяжелых шифров, таких как AES-256, на серверах с ограниченными ресурсами могут наблюдаться задержки.
2. Задержка и скорость передачи данных Проблемы с производительностью могут вызывать увеличение задержек или снижение скорости передачи данных.

1.4.2 Оптимизация

— Выбор легких шифров Алгоритмы, такие как ChaCha20, более эффективны на устройствах с низкой производительностью, например, на мобильных платформах.
— Использование аппаратного ускорения Включение аппаратного AES (через инструкции AES-NI) или использование GPU для обработки трафика.
— Настройка MTU (Maximum Transmission Unit) Правильная настройка MTU позволяет уменьшить фрагментацию пакетов, что ускоряет соединение.
— Кэширование DNS Локальное кэширование запросов DNS на сервере сокращает задержки при соединении.

1.5 Применение Shadowsocks в корпоративной среде

1.5.1 Преимущества для бизнеса

1. Безопасность данных Shadowsocks обеспечивает шифрование трафика, защищая конфиденциальную информацию сотрудников от утечек.
2. Обход корпоративной цензуры Некоторые компании используют строгие политики фильтрации трафика, ограничивающие доступ к ресурсам. Shadowsocks может стать инструментом для работы в таких

условиях.

1.5.2 Практические примеры

— Удаленная работа сотрудников: настройка собственного Shadowsocks-сервера позволяет сотрудникам безопасно подключаться к корпоративным ресурсам.
— Защита корпоративных серверов: Shadowsocks может быть использован для доступа к внутренним ресурсам компании через зашифрованные каналы.

1.6 Практические аспекты развертывания

1.6.1 Минимальные системные требования

— Сервер: CPU с поддержкой AES-NI (рекомендуется), минимум 512 МБ ОЗУ.
— Клиент: Поддержка протоколов Shadowsocks. Рекомендуются официальные или сторонние клиенты.

1.6.2 Шаги по настройке

1. Установка сервера Shadowsocks:

```
sudo apt update
sudo apt install shadowsocks-libev
```

2. Конфигурация:

```
{
    "server": "0.0.0.0",
    "server_port": 8388,
    "password": "your_password",
    "method": "aes-256-gcm",
    "timeout": 300
}
```

3. Настройка клиента: установите приложение Shadowsocks, введите IP сервера, порт, пароль и метод шифрования.

1.7 Сравнение с другими протоколами безопасности

Для анализа Shadowsocks можно сравнить со следующими популярными протоколами:

- OpenVPN — один из наиболее известных VPN-протоколов, который обеспечивает высокую безопасность за счет использования шифрования на базе SSL/TLS и поддержки различных алгоритмов шифрования. Однако OpenVPN требует значительных вычислительных ресурсов и может демонстрировать высокие задержки, особенно при использовании на мобильных устройствах или в условиях низкой пропускной способности сети. Shadowsocks, в отличие от OpenVPN, был создан как более легковесное и гибкое решение для обхода блокировок, оптимизированное для минимальной задержки и низкой заметности, что делает его предпочтительным выбором в условиях строгой цензуры.
- SOCKS5 — классический прокси-протокол, поддерживающий маршрутизацию различных типов трафика и позволяющий обфусцировать данные для сокрытия IP-адреса отправителя. Тем не менее, SOCKS5 не предоставляет встроенного шифрования, что делает его менее защищенным по сравнению с Shadowsocks. Shadowsocks использует алгоритмы шифрования, такие как AES и ChaCha20, что делает его более безопасным вариантом для защиты данных в сети.
- WireGuard — современный VPN-протокол, известный своей высокой скоростью и простотой настройки. WireGuard применяет передовые методы криптографии и обеспечивает быстрые соединения при сравнительно небольшой нагрузке на систему. Однако, в отличие от Shadowsocks, WireGuard более заметен для систем DPI, так как не предназначен для обфускации трафика и легко распознается по своим характеристикам. Shadowsocks же разработан специально для обхода обнаружения, делая трафик схожим с обычным HTTPS, что помогает избежать блокировок и фильтров в сетях с высоким уровнем контроля.
- V2Ray — это платформа для создания защищённых соединений, поддерживающая несколько протоколов, включая VMess и SOCKS. Она предлагает более гибкие возможности настройки по сравнению с Shadowsocks, позволяя адаптировать маршрутизацию и обфускацию трафика в за-

висимости от ситуации. V2Ray может скрывать трафик от систем DPI и хорошо справляется с обходом блокировок. Однако для его настройки требуется больше технических знаний и ресурсов по сравнению с Shadowsocks, что может быть сложным для пользователей без опыта.

Таким образом, каждый из этих протоколов отличается по критериям безопасности, скорости, удобства использования и требованиям к ресурсам. Shadowsocks выгодно выделяется среди них как гибкое решение с акцентом на неприметность и низкую задержку, что особенно важно для пользователей, которым требуется незаметное и безопасное соединение в условиях ограниченного доступа.

1.8 Основы криптографии в Shadowsocks

Алгоритмы шифрования: Shadowsocks использует симметричные шифры, такие как AES и ChaCha20, обеспечивающие высокую степень защиты данных. Эти шифры:

- AES — позволяет обеспечить защиту на уровне CBC и CTR режимов, предлагая высокую безопасность с минимальной нагрузкой на систему.
- ChaCha20 — известен своей эффективностью на мобильных устройствах и платформах с ограниченными ресурсами, обеспечивая защиту от большинства видов криптоанализа.

Задачи криптоанализа: Криптоанализ позволяет выявить слабые места в шифровании и обеспечить защиту трафика от различных видов атак.

1.9 Сетевые функции VPN

VPN-сервисы обеспечивают передачу данных между клиентом и сервером через защищенный туннель, который шифрует и анонимизирует трафик пользователя. Основные компоненты VPN:

- Шифрование: Шифрование обеспечивает защиту данных от перехвата и модификации. Применяются такие алгоритмы, как AES (Advanced Encryption Standard) и ChaCha20.
- Маршрутизация трафика: VPN перенаправляет сетевой трафик пользователя через удаленный сервер, скрывая реальный IP-адрес.
- Аутентификация: Обеспечивает доступ к VPN только авторизованным пользователям.
- Проверка целостности: Используются хеш-функции и алгоритмы MAC (Message Authentication Code), чтобы предотвратить изменение данных в процессе передачи.

1.10 Архитектура и функции Shadowsocks

Shadowsocks представляет собой прокси-сервер, который работает на основе SOCKS5 и поддерживает передачу данных через зашифрованные соединения. Основные функции Shadowsocks включают:

- Поддержка протоколов TCP и UDP: Shadowsocks может обрабатывать как TCP, так и UDP-пакеты, что позволяет ему работать с широким диапазоном сетевых приложений.
- Шифрование данных: Shadowsocks шифрует данные перед передачей, чтобы защитить их от перехвата.
- Поддержка многопоточности: Shadowsocks поддерживает одновременную обработку нескольких подключений, что повышает его производительность.
- API для управления ключами доступа: API-интерфейс позволяет создавать, обновлять и удалять ключи доступа, что необходимо для управления пользователями.

1.11 Сетевые аспекты работы Shadowsocks

Shadowsocks работает как туннельный прокси, перенаправляя трафик через защищенное соединение между клиентом и сервером. Это достигается за счет нескольких ключевых элементов:

1. Создание зашифрованного туннеля: Shadowsocks использует шифрование на уровне транспорта. Применяются такие алгоритмы, как AES и ChaCha20, которые обеспечивают как высокую скорость, так и стойкость к взлому.
2. Маршрутизация и перенаправление: Shadowsocks работает как SOCKS5-прокси, перенаправляя TCP- и UDP-трафик через сервер. Когда пользователь отправляет данные, клиент Shadowsocks шифрует их и отправляет на сервер, который затем перенаправляет их на конечный адрес.
3. Аутентификация и доступ: Shadowsocks использует механизм ключей доступа. Каждый ключ доступа привязан к пользователю и может иметь собственные лимиты данных, что позволяет администратору контролировать и управлять доступом.
4. Устойчивость к блокировке: Shadowsocks был разработан с учетом требований обхода цензуры. Для этого он маскирует трафик, делая его схожим с обычным HTTPS-трафиком, что затрудняет его обнаружение и блокировку.

1.12 Принцип работы Shadowsocks

1.13 Процесс шифрования и передачи данных

Shadowsocks шифрует данные перед отправкой их на сервер. Основной алгоритм шифрования — AES (Advanced Encryption Standard) или ChaCha20. Процесс включает следующие этапы:

1. Инициализация шифрования: Клиент и сервер согласовывают параметры шифрования, такие как ключ и метод. Это позволяет создавать уникальное защищенное соединение для каждого сеанса.
2. Шифрование данных: Клиент шифрует каждый пакет перед отправкой. Шифрованные данные отправляются на сервер Shadowsocks через SOCKS5-прокси.
3. Передача и маршрутизация: Сервер принимает шифрованный пакет, расшифровывает его и перенаправляет на конечный IP-адрес. Ответ от конечного сервера проходит обратный процесс — шифруется сервером и передается клиенту.
4. Дешифрование: Клиент расшифровывает полученные данные и пе-

редает их приложению.

1.14 Маршрутизация трафика и управление подключениями

Shadowsocks поддерживает многопоточность, что позволяет обрабатывать несколько подключений одновременно. При подключении клиента сервер выделяет ему уникальный канал связи и следит за состоянием соединения. Shadowsocks также поддерживает UDP, что делает его совместимым с широким спектром приложений.

1.15 API для управления ключами доступа

Shadowsocks имеет REST API, который позволяет управлять ключами доступа, обеспечивая удобное администрирование и контроль за использованием сети. Основные функции API:

- Создание ключей доступа: С помощью API можно создавать новые ключи доступа, привязывая их к определенным пользователям.
- Ограничение по трафику: Администратор может устанавливать лимиты на объем данных, передаваемых каждым ключом.
- Управление ключами: API позволяет обновлять, удалять и переименовывать ключи доступа, обеспечивая гибкость в управлении пользователями.

1.16 Прокси-серверы

Прокси-серверы выступают в роли посредников между клиентом и сервером, выполняя задачи скрытия IP-адреса клиента и обеспечения анонимности при подключении к интернет-ресурсам. В схеме сети с использованием прокси можно выделить следующие элементы:

$$\text{Клиент} \rightarrow \text{Провайдер} \rightarrow \text{Прокси-сервер} \rightarrow \text{Сервер}.$$

Такой подход скрывает исходный IP-адрес клиента, что позволяет конечному серверу видеть только IP-адрес прокси-сервера.

Прокси-серверы различаются по ряду критериев, таких как размещение, уровень анонимности и доступность. По типу размещения прокси-серверы делятся на централизованные и децентрализованные. В централизованной модели один прокси-сервер обслуживает множество клиентов, тогда как децентрализованные прокси-серверы распределены по разным

географическим точкам, что позволяет распределить нагрузку и повысить отказоустойчивость системы.

В зависимости от уровня анонимности, прокси-серверы бывают следующих типов:

— Прозрачные — не изменяют данные клиента, включая IP-адрес, и не обеспечивают скрытия, но являются быстрым и экономичным вариантом маршрутизации.
— Анонимные — скрывают IP-адрес клиента, но оставляют факт использования прокси-сервера.
— Искажающие — скрывают IP-адрес клиента и факт использования прокси, повышая уровень анонимности.
— Приватные — полностью скрывают данные клиента и периодически меняют IP-адрес, обеспечивая наибольшую степень анонимности.

По уровню доступности прокси-серверы делятся на:

— Публичные — открытые и бесплатные, но часто медленные и менее безопасные.
— Приватные — требуют оплаты, но обеспечивают высокую скорость и безопасность.
— Выделенные — предоставляют индивидуальные ресурсы для одного клиента, повышая производительность.
— Общие — предназначены для ограниченной группы, такой как сотрудники компании, и поддерживают баланс между безопасностью и доступностью.

1.17 Сравнение прокси и VPN

Основное различие между прокси и VPN заключается в уровне их функционирования в сетевой модели OSI: прокси работают на уровне приложений (уровень 7), тогда как VPN функционирует на уровнях 3 и 4. В VPN используется шифрование трафика с помощью криптографических алгоритмов, что обеспечивает высокую степень защиты от перехвата. VPN-сервисы зачастую дороже прокси, но предоставляют более высокий уровень безопасности за счёт шифрования данных, хотя в сравнении с прокси могут работать медленнее.

1.18 Типы прокси по протоколам

Различные типы прокси-серверов поддерживают специфические протоколы для работы с сетевыми запросами:

- HTTP-прокси — используется для обработки HTTP-запросов и подходит для базового веб-серфинга, такого как кеширование страниц и контроль доступа. Этот тип часто используется в корпоративных сетях для ограничения доступа к определённым ресурсам.
- HTTPS-прокси — представляет собой усовершенствованный HTTP-прокси, так как шифрует передаваемый трафик, что делает его подходящим для задач, требующих безопасности данных.
- SSL-прокси — обеспечивает создание TCP-канала для защищённого соединения, создавая одно соединение от клиента к серверу и позволяя безопасно использовать как HTTP, так и HTTPS.
- SOCKS-прокси — работает на уровне TCP и подходит для интенсивного трафика, включая потоковую передачу данных и P2P-соединения. SOCKS скрывает IP-адрес клиента, обеспечивая базовую конфиденциальность.

2 Практическая часть

1. Изучение существующих Open Source-решений:

— Outline — VPN-проект от Google, использующий Shadowsocks для обеспечения конфиденциальности.
— Shadowsocks-qt5 — кроссплатформенный клиент, написанный на Qt, подходит для изучения клиентской части Shadowsocks.

Документацию можно найти на официальном GitHub Shadowsocks (https://github.com/shadowsocks) и в репозиториях Outline и Shadowsocks-qt5.

2. Реализация клиентского приложения: Разработка клиентского приложения включает в себя несколько этапов, каждый из которых направлен на создание стабильного и защищённого соединения с серверами. Для этого потребуются следующие шаги:

— Определение интерфейса для подключения к серверу и визуализации состояния соединения: На этом этапе разрабатывается графический интерфейс пользователя (GUI), обеспечивающий удобное подключение к выбранному серверу и отображение текущего статуса соединения. Интерфейс должен включать:
  – Поле для ввода IP-адреса или доменного имени сервера и порта.
  – Настройки для выбора метода шифрования и ввода ключа (пароля) для шифрования соединения.
  – Индикатор состояния соединения (например, «Подключено», «Отключено», «Ошибка подключения»).
  – Опционально: возможность просмотра журналов (логов) соединений и ошибок для диагностики.
— Настройка параметров безопасности и конфигурации Shadowsocks: Здесь настраиваются криптографические параметры и методы шифрования, которые используются для защиты передаваемых данных. Важно обеспечить корректную конфигурацию для максимальной безопасности:
  – Выбор метода шифрования (например, AES-256-GCM или ChaCha20) для обеспечения стойкости к криптоанализу.
  – Настройка режима маскировки трафика для защиты от глубокого анализа пакетов (DPI).
  – Конфигурация параметров подключения (включая настройки

для сокетов) и каналов передачи данных.
  - Опционально: настройка методов обфускации трафика для повышения устойчивости к блокировкам и фильтрации на уровне сети.
- Оптимизация производительности и шифрования: Этот этап необходим для обеспечения быстрой работы приложения даже при высоких нагрузках и на устройствах с ограниченными ресурсами:
  - Настройка параметров буферизации для минимизации задержек при передаче данных.
  - Оптимизация алгоритмов шифрования для обеспечения баланса между безопасностью и скоростью (например, при использовании ChaCha20, который эффективен для мобильных устройств).
  - Проведение тестирования производительности, анализ слабых мест и оптимизация критических функций (например, настройка потоков для многопоточности, если это возможно).
  - Опционально: реализация алгоритмов сжатия данных для увеличения пропускной способности и снижения трафика.

В результате выполнения этих этапов будет создано приложение, которое предоставляет пользователю удобный интерфейс для безопасного подключения к серверам с высокой производительностью и низкой задержкой, устойчивое к различным методам блокировки и анализа трафика.

Заключение

Обзор источников

- GitHub https://github.com/shadowsocks
- Статья "ACER: Detecting Shadowsocks Server Based on Active Probe Technology— о методах анализа и обнаружения Shadowsocks-трафика.
- Книга "Cryptography and Network Security: Principles and Practice"(William Stallings) для глубокого понимания основ криптографии.