БЕЗОПАСНОСТЬ ПЕРЕДАЧИ МЕДИА ДАННЫХ ПО СЕТИ В РЕЖИМЕ РЕАЛЬНОГО ВРЕМЕНИ

Баклановская Б. М.

bella.baklanovskaya@gmail.com

Под руководством М.В. Баклановского

Аннотация

В данной работе рассмотрена реализация решения для организации защищённой видеосвязи в браузере средствами молодой, активно развивающейся технологии WebRTC (Web Real-Time Communication).

Введение

На сегодняшний день существует ряд программных продуктов и технологий, таких как Skype, которые активно используются для видеообщения в сети Интернет. Однако данные продукты имеют закрытый исходный код и протокол передачи данных, что не даёт возможности осуществить проверку защиты передаваемой информации или провести работы по повышению безопасности передачи данных. Поэтому существующие закрытые программные продукты не могут использоваться компаниями, которым требуется обеспечивать высокий уровень безопасности передаваемой информации. Возникает необходимость разработки решения для организации защищённой видеосвязи через Интернет. Разработка такого решения возможна на основе открытой технологии WebRTC (Web Real-Time Communication) [1], которая предназначена для организации пирингового соединения между двумя браузерами для передачи медиа данных в режиме реального времени.

Технология организации пирингового соединения средствами WebRTC

Технология WebRTC интегрирована с DOM (Document Object Model) и её основные возможности реализуются при помощи Java Script API [2]. Перед открытием соединения средствами WebRTC между двумя браузерами необходимо обеспечить участникам возможность договориться о формате передаваемых медиа данных и обменяться адресной информацией. О формате передаваемых данных, согласно WebRTC,

участники соединения договариваются, обмениваясь сообщениями предложение/ответ по протоколу SDP (Session Description Protocol) [3]. Адресные данные, как возможные маршруты для установления соединения, согласно стандартам технологии, формируются по протоколу ICE (Interactive Connectivity Establishment) [4]. Каждый отдельный маршрут называется ice-кандидатом. Указанными данными участники обмениваются через веб-сервер, который в терминологии WebRTC называется сигнальным (Signaling server). Реализация сигнального сервера возложена на разработчика.

После обмена адресной информацией участники пробуют установить соединение используя полученные ice-кандидаты. Если подходящий маршрут найден, участники начинают обмен медиа данными. (Рис. 1)

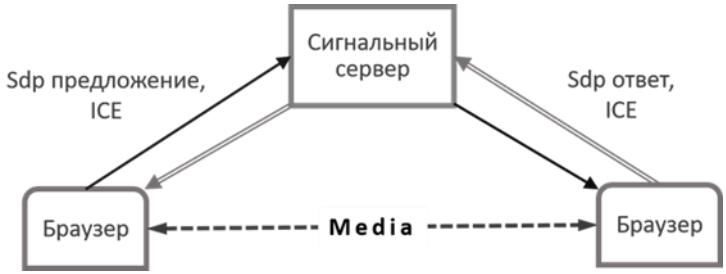

Рисунок 1: Схема организации пирингового соединения между двумя браузерами

Обмен медиа данными между участниками происходит по протоколу RTP (Real-time Transport Protocol), который базируется на протоколе UDP (User Datagram Protocol).

Помешать установить соединение напрямую может NAT (Network Address Translation) [5], находящийся на пути соединения. Проблема наличия NAT заключается в том, что участники соединения знают только свои локальные адреса и, поэтому, ни один из сформированных ими маршрутов не подходит для установления соединения, за исключением случая, когда участники находятся в одной локальной сети. Данную проблему можно решить, сообщив участникам соединения их публичные адреса. Для этого используется STUN (Session Traversal Utilities for NAT) сервер [6].

Однако в случае, когда оба участника соединения находятся в разных локальных сетях за NAT, реализованными таким образом, чтобы исключить возможность инициализации соединения из публичной сети, использование STUN сервера недостаточно для установления соединения.

Для обхода таких ограничений необходимо использовать relay-сервер. В рамках технологии WebRTC для ретрансляции передаваемых данных используется TURN (Traversal Using Relays around NAT) сервер [7]. Протокол TURN впервые был описан в документе RFC 5766 в 2010 году.

Установление соединения с использованием TURN сервера осуществляется следующим образом (Рис. 2). Обмен данными между TURN сервером и клиентом осуществляется по транспортному протоколу UDP.

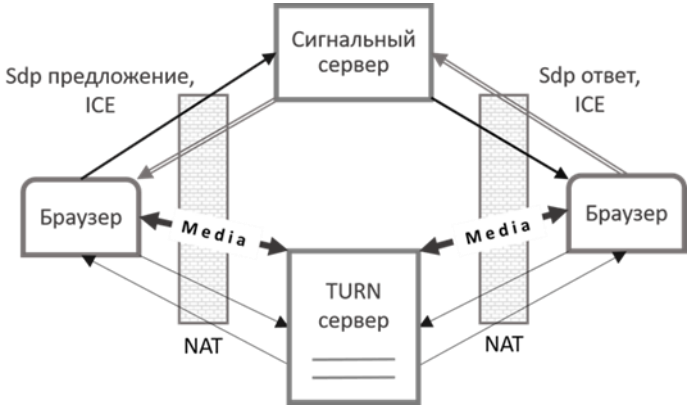

Рисунок 2: Схема организации соединения с использованием TURN сервера

1) Каждый участник проходит авторизацию на TURN-сервере. Авторизовавшемуся участнику сервер выделяет ретрансляционный адрес.

2) Участники обмениваются полученными на TURN-сервере ретрансляционными адресами через сигнальный сервер.

3) Участники начинают обмен данными через TURN-сервер.

Существующие реализации данного сервера являются частью коммерческих разработок и доступ к ним предоставляется на платной основе. В открытом доступе можно найти исходные коды реализаций сервера, которые написаны для использования на машинах с операционной системой Ubuntu. Например, coTURN, Restund, и ряд других.

В ходе данной работы был реализован TURN сервер, работающий на машине с операционной системой Windows. Реализация проводилась по документу RFC 5766.

Безопасность передаваемых данных

Описанное в данной работе программное решение для организации видеосвязи в браузере с использованием TURN сервера (Рис. 2) позволяет решить ряд серьёзных вопросов безопасности передачи данных по Сети:

- При открытии соединения и передаче данных используются открытые протоколы, при этом все медиа данные проходят через TURN сервер. Таким образом, в отличие от ситуации с закрытыми программными продуктами, например, Skype, при использовании данного решения известно, как идёт весь трафик между участниками соединения. Это позволяет вести работы по защите передаваемой информации.

- Для использования данного решения не требуется установка дополнительных внешних бинарных модулей и плагинов. Поэтому при его использовании в браузерах с открытым исходным кодом, например, Firefox, есть возможность организовать эффективную проверку данного программного продукта на наличие угроз безопасности данных изнутри.

- В протоколе, описывающем работу TURN сервера, предусмотрена авторизация участников на сервере перед началом обмена данными. При реализации данного решения система авторизации пользователей может быть расширена, в том числе возможна интеграция базы пользователей описанного программного решения с уже существующей системой каталогов.

- Также использование TURN сервера для ретрансляции передаваемых медиа данных позволяет осуществлять видеологирование. Это значит, что к давно освоенным возможностям записи данных, передаваемых через веб-сервер (сигнальный сервер), таких как данные для установления соединения (адресная информация и данные с описанием сессии), текстовые файлы, картинки и др., добавилась возможность записи всех передаваемых аудио и видеопотоков. Таким образом, при использовании описанного решения для организации видеосвязи, можно записывать и сохранять все данные, которыми обмениваются участники общения.

- Обмен данными с веб-сервером (сигнальным сервером) осуществляется по транспортному протоколу TCP, поэтому данные могут быть защищены SSL, TLS шифрованием. Передача медиа данных через TURN сервер осуществляется по

транспортному протоколу UDP, что не даёт возможности использовать SSL, TLS шифрование. Для защиты данных во время их передачи между браузером и сервером можно использовать VPN.

Заключение

TURN сервер, который изначально был реализован для обхода ограничений NAT при установлении соединения, в результате оказался продуктом, который является важной частью, описанного в данной работе, программного решения, так как позволяет контролировать передаваемые медиа потоки в достаточной степени для проведения работ по повышению безопасности передачи данных по Сети.

Литература

1. WebRTC Home // URL: https://webrtc.org
2. Mozilla Developer Network, WebRTC API // URL: https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API
3. Handley M., Jacobson V., Perkins C. SDP: Session Description Protocol, RFC 4566, July 2006 // URL: https://www.rfc-editor.org/rfc/rfc4566.txt
4. Rosenberg J. Interactive Connectivity Establishment (ICE): A protocol for Network Address Translator (NAT) Traversal for offer/answer protocols, RFC 5245, April 2010 // URL: https://www.rfc-editor.org/rfc/rfc5245.txt
5. Олифер В., Олифер Н. Компьютерные сети. Принципы, технологии, протоколы. Учебник для вузов. – 5-е изд. – СПб.: Питер, 2016.
6. Rosenberg J., Mahy R., Matthews P., Wing D. Session Traversal Utilities for NAT (STUN), RFC 5389, October 2008 // URL: https://www.rfc-editor.org/rfc/rfc5389.txt
7. Mahy R., Matthews P., Rosenberg J. Traversal Using Relays around NAT (TURN): Relay extensions to Session Traversal Utilities for NAT (STUN), RFC 5766, April 2010 // URL: https://www.rfc-editor.org/rfc/rfc5766.txt