**УДК 004.9**

***Левщанова Е.В,***

***студентка,***

***Ульяновский государственный университет,***

***Россия, г. Ульяновск***

# РАЗРАБОТКА КЛИЕНТ-СЕРВЕРНОЙ СИСТЕМЫ, ОБЕСПЕЧИВАЮЩЕЙ ЗАЩИЩЕННУЮ АВТОРИЗАЦИЮ НА ОСНОВЕ КРИПТОГРАФИЧЕСКОГО АЛГОРИТМА SRP-6

***Аннотация:*** *SRP протокол описывает установление безопасного соединения, используя для аутентификации пару логина и пароля, при этом пароль используется только на стороне клиента. Надежность сохраняется даже при использовании относительно коротких, подходящих для запоминания человеком паролей. Даже имея полную запись обмена по каналу, злоумышленник не получает информации, достаточной для аутентификации данным пользователем. В статье рассматривается 6 версия протокола SRP.*

***Ключевые слова:*** *информационные технологии, клиент-серверная система, криптографический алгоритм, протокол SRP-6.*

***Annotation:*** *The SRP protocol describes the establishment of a secure connection using a pair of login and password for authentication, in which the password is used only on the client side. Reliability is maintained even when using relatively short, suitable passwords for memorizing by human. Even with a complete record of the exchange through the channel, the attacker does not receive information sufficient to authenticate as this user. This article discusses about 6$^{th}$ version of the SRP protocol.*

***Key words:*** *information technology, client-server system, cryptographic algorithm, SRP-6 protocol.*

Secure Remote Password protocol впервые был опубликован в 1998. Это протокол парольной аутентификации, устойчивый к прослушиванию и Man in the middle (MITM)-атаке и не требующий третьей доверенной стороны. SRP-

6 содержит некоторые элементы из других протоколов обмена ключами, при этом вносит небольшие усовершенствования и уточнения.

Протокол SRP-6 позволяет пользователю идентифицировать себя на сервере, при этом не передавая своего пароля, то есть подтвердить тот факт, что он знает свой пароль, и только этот факт. SRP-6 эффективно реализует доказательство с нулевым разглашением между пользователем и сервером, хранящим информацию о его пароле.

В результате работы данного протокола обе стороны получают длинный секретный ключ, проверяемый на соответствие между сторонами после получения. В случаях, когда помимо аутентификации необходимо шифрование данных, SRP-6 предоставляет более быстрые, чем Deffie-Hellman, средства для достижения этой цели.

Далее рассматриваются особенности программной реализации клиент-серверной системы, использующей для авторизации данный протокол.

**Описание программных инструментов для реализации системы.**

Для демонстрации работы SRP-6 протокола требуется разработать две программы: клиент и сервер, обеспечивающие друг с другом обмен данными по локальной сети.

Серверное приложение осуществляет проверку (авторизацию) приложения клиента на основе криптографического протокола SRP-6, который обеспечивает безопасность этой процедуры.

В работе будут использованы следующие программные инструменты:

1. Среда разработки Visual Studio (версии 2015);

2. Программные библиотеки Qt;

3. Среда разработки Qt Creator;

4. Программная библиотека OpenSSL - сторонняя криптографическая библиотека

Структура проекта в виде дерева каталогов представлена на рисунке 1.

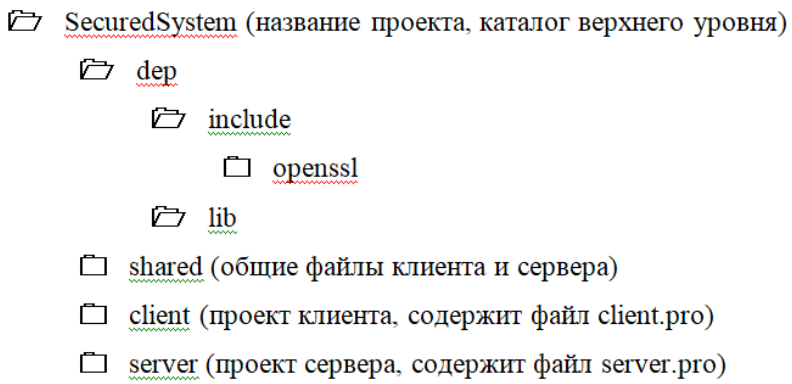


***Рисунок 1. Структура проекта***

Каталог dep содержит сторонние библиотеки. В каталоге include находятся заголовочные файлы библиотек. Openssl - каталог .h-файлов библиотеки OpenSSL. Каталог lib включает в себя бинарные .lib-файлы библиотек.

**Создание приложений клиента и сервера.**

Разрабатываются два программных приложения, обеспечивающие обмен данными друг с другом по локальной сети. Внешний вид серверного приложения представлен на рисунке 2.

Объект Spin Box позволяет устанавливать значение порта, к которому будет подключаться клиент. В журнале выводятся данные, которые приходят от подключенных клиентов. В таблице содержатся пары идентификатор-пароль, используя которые возможно пройти авторизацию на сервере. Кнопка «старт» запускает сервер, создает слушающий сокет. Сокеты - название программного интерфейса для обеспечения обмена данными между процессами, это абстрактный объект, представляющий конечную точку соединения.

Для взаимодействия между машинами с помощью стека протоколов TCP/IP используются адреса и порты. Первое на текущий момент представляет собой 32-битный адрес (для протокола IPv4, 128-битный для IPv6), наиболее часто его представляют в символьной форме mmm.nnn.ppp.qqq (адрес, разбитый на четыре поля, разделённых точками, по одному байту в

поле). Второе — это номер порта в диапазоне от 0 до 65535. Эта пара и есть сокет («гнездо», соответствующее адресу и порту).

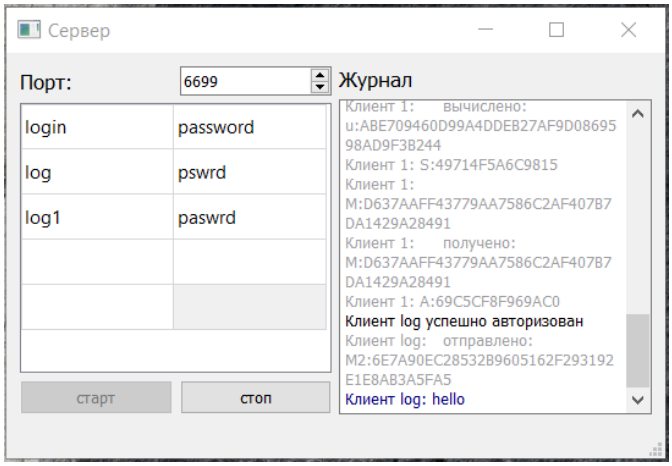


***Рисунок 2. Серверное приложение***

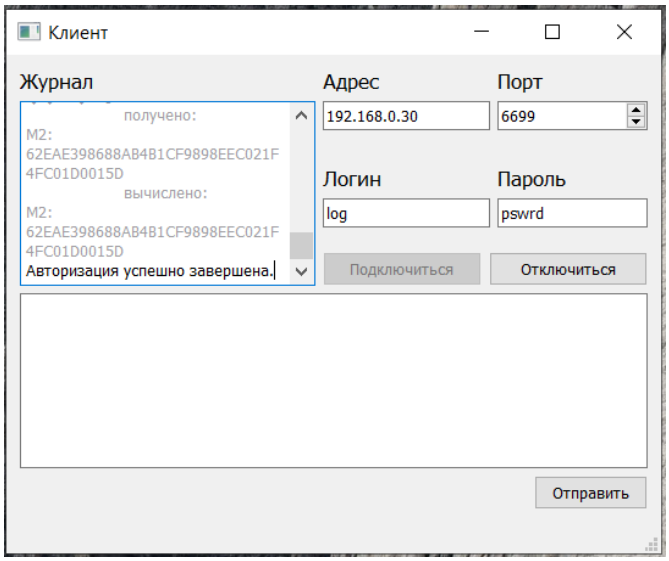


***Рисунок 3. Клиентское приложение***

Кнопка «стоп» останавливает сервер и прослушивание данного порта.

Внешний вид клиентского приложения, представленный на рисунке 3, построен по схожей схеме.

**Исходные данные, необходимые для алгоритма протокола SRP-6:**

1. q и N = 2q + 1, такие, что N и q простые. N должно быть достаточно большим, чтобы дискретное логарифмирование по модулю N было практически неосуществимо. Так же должно выполняться N = 2 (mod 3). Известно обеим сторонам.

2. Вся арифметика выполняется по модулю N (поле $F_N$).
3. g — первообразный корень по модулю N, известно обеим сторонам.
4. k — постоянный параметр, известный обеим сторонам, и равный 3 в данной ревизии протокола.
5. s — соль (строка данных).
6. I — идентификатор пользователя в системе сервера.
7. p — пароль пользователя, соответствующий I.
8. H() — криптографическая хеш-функция (в этом проекте использована SHA-1)
9. x — секретный ключ, x = H(s, I, p).
10. v — верификатор пароля на стороне сервера, $v = g^x$.
11. u — произвольный параметр для кодирования.
12. a,b — секретные одноразовые числа

Понятия пароля и верификатора соответствуют общепринятым понятиям секретного и открытого ключей, с двумя оговорками: пароль, как правило, меньше секретного ключа, так как его должен помнить пользователь. В свою очередь, верификатор по математическим свойствам схож с открытым ключом, так как он легко получается из пароля, а обратная операция является вычислительно неразрешимой. Однако вместо того, чтобы быть общеизвестным, верификатор хранится сервером в тайне в наборе (I, s, v). Способ аутентификации, который предполагает хранение сервером верификатора, но не пароля, называется основанным на верификации (verifier-based).

**Алгоритм протокола SRP-6**

Авторизация происходит по следующей схеме:

1. Клиент -> Сервер: $I, A = g^a$
2. Сервер -> Клиент: $s, B = kv + g^b$
3. На обеих сторонах: $u = H(A, B)$
4. На стороне клиента вычисляются:

$x = H(s, I, p)$

$S = (B — kg^x)^{(a + ux)}$ (вычисляется ключ сессии)

$M_1 = H(A, B, S)$

$K = H(S)$ (K — это искомый ключ для шифрования)

5. Клиент -> Сервер: $M_1$
6. На стороне сервера:

$S = (Av^u)^b$ (вычисление ключа сессии)

$M_2 = (A, M_1, S)$

$K = H(S)$ ($K$ — это искомый ключ для шифрования)

Проверяется равенство $M_1$ и $M_2$; если они равны – авторизация прошла успешно.

Дополнительно сервер может передать клиенту свое $M_2$, и тогда клиент сможет убедиться, что сервер тоже «настоящий».

**Реализация протокола SRP-6**

Клиент и сервер ведут обмен по протоколу TCP путем посылки и приема информационных пакетов. Каждый информационный пакет состоит из заголовка и данных. Заголовок информационного пакета имеет размер 4 байта и представлен структурой вида:

```
struct MessageHeader
{
    quint16 size;
    quint8 type;
    quint8 error;
};
```

Поле size содержит размер данных информационного пакета (заголовок не является частью данных).

Поле error содержит код ошибки, который устанавливается сервером в ответ на запросы клиента. Это поле может принимать значения, представленные в перечислении ниже:

```
enum PacketError
{
	PACKET_ERROR_NONE = 0,
	PACKET_ERROR_ACCESS_DENIED = 1,
	PACKET_ERROR_FORBIDDEN_MESSAGE = 2
};
```

Поле type несет информацию о типе пакета. В процессе авторизации и дальнейшего обмена данными клиента и сервера используются следующие типы пакетов:

```
enum PacketType
{
	PACKET_TYPE_EMPTY = 0,
	PACKET_TYPE_LOGIN = 1,
	PACKET_TYPE_AUTHORIZATION = 2,
	PACKET_TYPE_DATA = 3,
	PACKET_TYPE_DISCONNECTION = 4
};
```

Пакет PACKET_TYPE_LOGIN посылается клиентом сразу после успешного подключения к серверу и содержит информацию, требующуюся на начальном этапе авторизации информацию. В ответ сервер посылает пакет такого же типа с ненулевым кодом ошибки (если логин не найден в базе), либо с нулевым кодом ошибки и информацией, необходимой для работы алгоритма SRP-6.

В случае приема пакета с нулевым кодом ошибки, клиент посылает серверу пакет PACKET_TYPE_AUTHORIZATION с информацией, необходимой для продолжения работы алгоритма SRP-6. В ответ сервер посылает пакет такого же типа с нулевым кодом ошибки и информацией о верификации SRP-6 в случае успеха (этот момент считается окончанием авторизации на стороне сервера). Клиент, получив этот пакет с нулевым

кодом ошибки, проводит верификацию серверных данных (при необходимости) и получает возможность отправлять серверу информационные сообщения в пакетах PACKET_TYPE_DATA. Ненулевой код ошибки в пакете PACKET_TYPE_AUTHORIZATION свидетельствует о некорректном пароле.

При попытке клиентом отправить пакет PACKET_TYPE_DATA до завершения процесса авторизации, сервер возвращает в ответ сообщение PACKET_TYPE_DATA с кодом ошибки PACKET_ERROR_ FORBIDDEN_MESSAGE.

Любой ненулевой код ошибки со стороны сервера сопровождается разрывом соединения с клиентом (сразу после отправки сообщения с кодом ошибки).

PACKET_TYPE_DISCONNECTION предназначен для оповещения клиента о том, что сервер отключился.

Таким образом, на клиенте и сервере реализуется протокол SRP-6, использующий 3 обмена сообщениями в процессе авторизации.

**Использованные источники:**

1. SRP Protocol Design [Электронный ресурс] http://srp.stanford.edu/design.html
2. The Secure Remote Password Protocol, Thomas Wu, Computer Science Department, Stanford University [Электронный ресурс] http://srp.stanford.edu/ndss.html
3. Документация OpenSSL [Электронный ресурс] https://www.openssl.org/docs/
4. T. Wu, SRP-6: Improvements and Refinements to the Secure Remote Password Protocol, Представление рабочей группе IEEE P1363, Октябрь 2002.