Туртыгин А.А. Реализация протокола аутентификации с нулевым разглашением с использованием меток // Международный журнал информационных технологий и энергоэффективности. – 2024. – Т. 9 № 10(48) с. 83–89

Международный журнал информационных технологий и энергоэффективности

Сайт журнала: http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.438.31

РЕАЛИЗАЦИЯ ПРОТОКОЛА АУТЕНТИФИКАЦИИ С НУЛЕВЫМ РАЗГЛАШЕНИЕМ С ИСПОЛЬЗОВАНИЕМ МЕТОК

Туртыгин А.А.
ФГБОУ ВО "УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ", Ульяновск, Россия, (432017, Ульяновская область, город Ульяновск, ул. Льва Толстого, д. 42), e-mail: alex.mad.turt@gmail.com

Целью статьи является исследование протокола аутентификации с нулевым разглашением на основе шифра Эль-Гамаля. Протокол аутентификации, основанный на асимметричном шифровании с дополнительным свойством нулевого разглашения, позволяет проверяющей стороне убедиться в достоверности некоторого высказывания доказывающей стороны, не допуская при этом утечки информации о секрете. Для обеспечения взаимной аутентификации применяется механизм меток, добавляемых в проверочное сообщение перед его зашифрованием. Использование меток в протоколе аутентификации позволяет снизить нагрузку на вычислительные ресурсы вследствие отказа от традиционно применяемых хеш-функций.

Ключевые слова: Протокол аутентификации; нулевое разглашение; алгоритм Эль-Гамаля; серверная аутентификация; клиент-серверное приложение.

IMPLEMENTATION OF ZERO-KNOWLEDGE AUTHENTICATION PROTOCOL USING LABELS

Turtygin A.A.
ULYANOVSK STATE UNIVERSITY, Ulyanovsk, Russia, (432017, Ulyanovsk region, Ulyanovsk city, Lva Tolstoy str., 42), e-mail: alex.mad.turt@gmail.com

The purpose of the paper is to study the zero-knowledge authentication protocol based on the ElGamal cipher. The authentication protocol, based on asymmetric encryption with an additional zero-knowledge property, allows the verifier to check the authenticity of some statement of the prover, while preventing leakage of secret. To ensure mutual authentication, a mechanism of tags is used, which are added to the verification message before its encryption. The use of tags in the authentication protocol reduces the load on computing resources due to the abandonment of traditionally used hash functions.

Keywords: Authentication protocol; zero-knowledge proof; ElGamal algorithm; server authentication; client-server application.

Метки в протоколах аутентификации

В последнее время набирают популярность протоколы аутентификации с нулевым разглашением на основе асимметричного шифрования. В данном случае «нулевое разглашение» говорит об отсутствии утечки какой-либо информации о секретном ключе в процессе обмена сообщениями типа «запрос-ответ» в процессе аутентификации [1]. Причём допускается возможная первоначальная утечка информации о секретном ключе в случае, если злоумышленник предложит владельцу открытого ключа расшифровать поддельное проверочное сообщение M [2].

Такие протоколы должны обеспечивать взаимную аутентификацию, т.е. оба участника по результатам обмена сообщениями в процессе аутентификации получают информацию о возможности доверия к противоположной стороне-участнику.

Односторонние протоколы аутентификации построены на методах зашифрования некоторых сообщений проверяющим, их последующей расшифровке и передаче обратно доказывающей стороной. Такой подход позволяет проверяющей стороне убедиться в подлинности доказывающей, но не наоборот. Проблема возникает в случае, когда вместо проверяющей стороны окажется злоумышленник – он сможет получить информацию о секрете.

Для решения проблемы взаимной аутентификации к зашифрованным сообщениям добавляют хеш-код, полученный в результате работы хеширующей функции от этих сообщений [3], тем самым предотвращая возможность выбора злоумышленником случайного запроса в качестве проверочного. Корректно сформированный запрос от проверяющего участника представляется в виде пары значений: зашифрованного на открытом ключе сообщения и его хеша. При получении этой пары значений в качестве запроса, доказывающий расшифровывает сообщение и вычисляет его хеш. Затем он сравнивает полученный и вычисленный хеши, и убеждается в том, что расшифрованное им сообщение уже было известно проверяющему.

Но применение хеш-функций не является единственным способом каждой из сторон убедиться в правдивости утверждения противоположной стороны. В качестве доказательства того, что переданное сообщение было известно проверяющему до процесса зашифрования, может применяться механизм меток μ, встраиваемых в сообщение [1]. Наличие метки в расшифрованном сообщении позволяет убедиться в том, что проверяющая сторона изначально владела информацией об исходном сообщении. На роль метки при использовании асимметричного шифрования хорошо подходит некоторая часть открытого ключа доказывающей стороны. На практике длина метки ($|\mu|$) в двоичном представлении выбирается от 80 до 512 бит, что позволяет значительно снизить вероятность расшифрования случайного запроса злоумышленника в сообщение с корректной меткой ($P = 2^{-|\mu|}$) [1].

Протокол Эль-Гамаля

Схема Эль-Гамаля – это криптосистема с открытым ключом, основанная на сложности вычисления дискретных логарифмов в конечном поле [4]. Шифрование в ней – вероятностное, т.е. одинаковое сообщение может быть зашифровано в различные криптограммы, при расшифровании которых всегда получается одно и то же изначальное сообщение.

Рассмотрим протокол аутентификации с нулевым разглашением, использующий асимметричный шифр Эль-Гамаля и метку в качестве механизма взаимной аутентификации (Рис. 1). В нашем случае меткой будут являться младшие 160 бит открытого ключа, что говорит о крайне низкой вероятности совпадения меток для случайного запроса ($P = 2^{-160}$).

В качестве открытых (известных обоим участникам) параметров протокола используются p – большое простое число, где p-1 содержит простой делитель длиной не менее 160 бит в двоичном исчислении, g – первообразный корень по модулю p, y – открытый ключ доказывающей стороны и μ – метка, где $\mu = y \pmod{2^{160}}$.

Секретный ключ x доказывающая сторона будет хранить в тайне.

Шаги протокола аутентификации можно записать следующим образом:

1. Проверяющая сторона генерирует случайным образом разовый секретный ключ отправителя k и сообщение M со следующим ограничением его длины: $|M|<|p|-|2*\mu|$.
2. Далее вычисляет разовый открытый ключ отправителя R и разовый общий секретный ключ Q, где $R = g^k \pmod p$ и $Q = y^k \pmod p$.
3. Затем осуществляет зашифрование битовой строки $M||\mu$, состоящей из сообщения M и метки μ, в число C, где $C = Q*(M||\mu) \pmod p$.
4. Проверяющая сторона передаёт доказывающей полученную в предыдущих шагах пару чисел (R, C).
5. Доказывающая сторона получает значения (R, C) и приступает к вычислению общего секретного ключа $Q = R^x \pmod p$ и Q^{-1} (обратного к нему числа по модулю p), используя расширенный алгоритм Эвклида [5].
6. Далее расшифровывает полученную криптограмму C и получает некоторую битовую строку, содержащую в себе сообщение и метку, следующим образом: $M'||\mu' = C*Q^{-1} \pmod p$.
7. Затем проверяет равенство вычисленной ранее и расшифрованной меток, если равенство выполняется, то доказывающая сторона отправляет проверяющей значение M', а в противном случае объявляет запрос некорректным.
8. Проверяющая сторона принимает решение о легитимности доказывающей на основе равенства исходного и полученного сообщений.

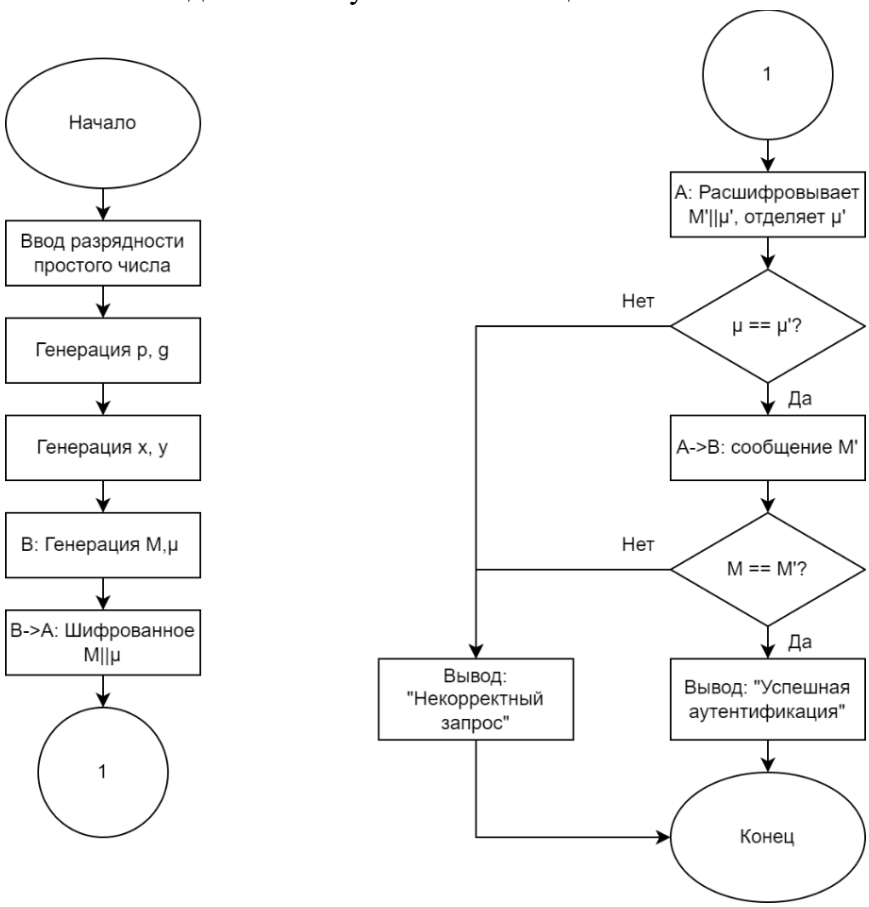

Рисунок 1. - Схема протокола аутентификации

Источник: анализ автора

Реализация протокола

Протокол аутентификации был реализован в виде исполняемого файла для операционных систем семейства Windows на языке программирования Visual C# (.Net Framework 4.7.2). Поскольку размеры стандартных числовых типов в данном языке ограничены 64 битами, а для криптостойкости необходима работа с числами размером, превышающим 1024 бита, было решено использовать тип данных BigInteger.

Все основные методы и функции работают в нескольких потоках, поскольку приложение содержит в себе клиентскую и серверную части – выполнение всех функций обоих частей в одном потоке невозможно по причине постоянных его блокировок. Пример запуска серверных функций во вспомогательных потоках:

```
            private void awaitConnectButton_Click(object sender,
EventArgs e)
            {
                new Thread(MainServerThread).Start();
            }
            private void MainServerThread()
            {
                Invoke(() => { consoleRichTextBox.Text = "*** Создан
именованный канал ***\n"; });
                Invoke(() => { consoleRichTextBox.Text += "Ожидание
подключения клиентов...\n"; });
                new Thread(ServerThread).Start();
                Thread.Sleep(250);
            }

            private void newClientWindowOpenButton_Click(object
sender, EventArgs e)
            {
                Thread cfT = new Thread(ClientStart);
                cfT.Start();

      }
```

Основное окно серверной части программы позволяет задать размер числа p, запустить протокол аутентификации и открыть клиентскую часть. Клиентское приложение содержит строку установления связи с сервером и выступает в роли доказывающей стороны.

Межпроцессный обмен информацией в процессе аутентификации происходит посредством именованных каналов. По этим каналам информация передаётся в виде некоторой последовательности байт. Для восстановления чисел из данных последовательностей предложено следующее представление числа:

“длина массива длины” [*“массив длины”*] [*“массив числа”*]

Это позволяет передавать числа размерами до 255*255*8 = 520200 бит, что достаточно для современных криптосистем. Ниже представлен фрагмент программы, отвечающий за восстановление числа из последовательности байт:

```
            private BigInteger GetFromPipe(NamedPipeServerStream
pipe)
            {
                // [length of length] [...length...] [...number...]
                int lengthOfLength = pipe.ReadByte();

                byte[] lengthBytes = new byte[4] { 0, 0, 0, 0 };
                for (int i = 0; i < lengthOfLength; i++)
                {
                    lengthBytes[i] = (byte)pipe.ReadByte();
                }
                int length = BitConverter.ToInt32(lengthBytes, 0);
                byte[] number = new byte[length];
                pipe.Read(number, 0, number.Length);

                return new BigInteger(number);
            }
            private void SendToPipe(NamedPipeServerStream pipe,
BigInteger bigInt)
            {
                byte[] number = bigInt.ToByteArray();
                byte[] length =
BitConverter.GetBytes(number.Length);
                // [length of length] [...length...] [...number...]
                pipe.WriteByte((byte)length.Length);
                pipe.Write(length);
                pipe.Write(number);
            }
```

Именованный канал создаётся при запуске аутентификации либо на сервере, либо на клиенте, и выбранная сторона начинает следить за поступающей в канале информацией. Фрагмент программы, отвечающий за создание именованного канала:

```
            private void ServerThread(object? data)
            {
                try
                {
                    // Pipe initialization
                    NamedPipeServerStream pipeServer =
                    new NamedPipeServerStream("authpipe",
PipeDirection.InOut);

                    // Wait for a client to connect
                    pipeServer.WaitForConnection();
```

```
                    Invoke(() => { consoleRichTextBox.Text +=
$"Клиент успешно подключен.\n"; });
                }
            }
```

После того, как сервер и клиент запустят процесс аутентификации, происходит обмен «приветствием» друг с другом для того, чтобы убедиться в идентичности выбранного протокола:

```
            // Get greeting <---
            BigInteger hello = new BigInteger(788580748674 85);
            if (hello == GetFromPipe(pipeServer))
            {
                Invoke(() => {
                    consoleRichTextBox.Text += $"Получено
приветствие от клиента!\n\n";
                });
            }
```

После утверждения типа протокола между клиентом и сервером начинается обмен параметрами и криптограммами в соответствии с блок-схемой алгоритма (Рис. 1). По результатам выводится отчёт об успешности аутентификации (Рис. 2).

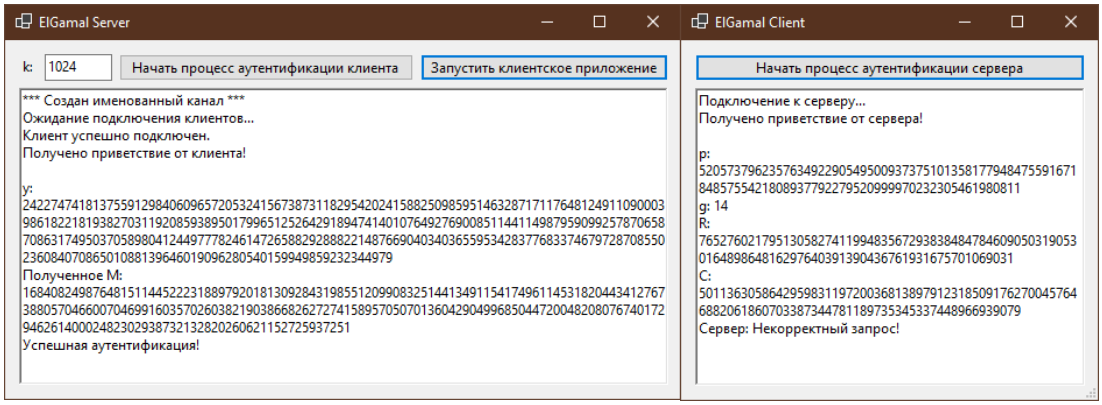

Рисунок 2. - Примеры отчётов программы

Источник: анализ автора

Заключение

Протоколы аутентификации с нулевым разглашением часто применяются в системах защиты информации. С их помощью одна из сторон-участников убеждается в том, что некоторый секрет уже известен второй стороне без необходимости раскрытия самого секрета.

В статье приводится программная реализация протокола аутентификации на основе асимметричного шифра Эль-Гамаля с использованием меток, который позволяет снизить нагрузку на вычислительные ресурсы путём отказа от традиционно используемых хеш-функций. Также приведён способ решения возникающих проблем при обработке больших чисел в процессе разработки.

Список литературы

1. Молдовян, А. А. Протоколы аутентификации с нулевым разглашением секрета / А. А. Молдовян, Д. Н. Молдовян, А. Б. Левина. – Санкт-Петербург : Университет ИТМО, 2016. – 55 с. – EDN XFXBRL.
2. Рацеев, С. М. Математические методы защиты информации / С. М. Рацеев. – Санкт-Петербург : Издательство "Лань", 2022. – 544 с. – ISBN 978-5-8114-8589-5. – EDN QZANSJ.
3. Рацеев, С. М. О протоколах аутентификации с нулевым разглашением знания / С. М. Рацеев, М. А. Ростов // Известия Саратовского университета. Новая серия. Серия: Математика. Механика. Информатика. – 2019. – Т. 19, № 1. – С. 114-121. – DOI 10.18500/1816-9791-2019-19-1-114-121. – EDN ULRWTX.
4. T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in IEEE Transactions on Information Theory, vol. 31, no. 4, pp. 469-472, July 1985, doi: 10.1109/TIT.1985.1057074.
5. Сикорская, Г. А. Мультипликативно обратный элемент для вычета по модулю m / Г. А. Сикорская // Инновационная наука. – 2016. – № 4-4. – С. 37-39. – EDN VSUOND.

References

1. Moldovyan, A. A. Authentication protocols with zero-knowledge of a secret / A. A. Moldovyan, D. N. Moldovyan, A. B. Levina. – St. Petersburg : ITMO University, 2016. – p.55 – EDN XFXBRL.
2. Ratseev, S. M. Mathematical methods of information protection / S. M. Ratseev. – St. Petersburg : Lan Publishing House, 2022. – p. 544– ISBN 978-5-8114-8589-5. – EDN QZANSJ.
3. Ratseev, S. M. On authentication protocols with zero knowledge disclosure / S. M. Ratseev, M. A. Rostov // Izvestiya Saratov University. A new series. Series: Mathematics. Mechanics. Computer science. - 2019. – Vol. 19, No. 1. – pp. 114-121. – DOI 10.18500/1816-9791-2019-19-1-114-121. – EDN ULRWTX.
4. T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in IEEE Transactions on Information Theory, vol. 31, no. 4, pp. 469-472, July 1985, doi: 10.1109/TIT.1985.1057074.
5. Sikorskaya, G. A. Multiplicatively inverse element for deduction modulo m / G. A. Sikorskaya // Innovative science. - 2016. – No. 4-4. – pp. 37-39. – EDN VSUOND.