РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

РЕФЕРАТ

ПО ТЕМЕ “Установление подлинности запрос-ответ”

дисциплина: Математические основы защиты информации и информационной безопасности

Студент: Пиняева Анна Андреевна

Группа: НПИмд-01-24

МОСКВА

2025

Оглавление

Введение

В условиях растущих киберугроз, фишинга, атак «человек посередине» (MITM), массовых утечек паролей и внедрения моделей безопасности Zero Trust, особенно остро встаёт задача надёжной аутентификации. Одним из эффективных и хорошо изученных решений является метод «вызов–ответ» (challenge–response), при котором пользователь доказывает своё право на доступ, не раскрывая секретной информации напрямую. Метод основан на генерации уникального случайного «вызова» со стороны сервера и последующего вычисления клиентом «ответа» с использованием известного только ему секрета (ключа или пароля). Такой подход исключает передачу постоянных паролей по сети, тем самым предотвращая их перехват. Challenge–response широко используется в протоколах удалённой аутентификации, одноразовых паролях, смарт-картах, банковских токенах и криптографических схемах. Его изучение остаётся актуальным в контексте развития методов активного перехвата, социальных атак и компрометации данных. Современные рекомендации в области ИБ (NIST, ISO/IEC 27001) прямо указывают на необходимость использования механизмов, исключающих повторное использование аутентификационной информации — что и обеспечивает данный метод.

1. Теоретические основы аутентификации

1.1 Понятие аутентификации и её роль в информационной безопасности

Аутентификация — это процесс подтверждения подлинности субъекта (пользователя, сервера, устройства или программного агента), который запрашивает доступ к информационному ресурсу. Она относится к базовым механизмам обеспечения безопасности наряду с авторизацией, контролем доступа и аудитом. Аутентификация отвечает на вопрос: «Кто ты?» И служит «точкой входа» в любые защищённые системы. От корректной работы механизма аутентификации зависит общая устойчивость системы к: подмене личности, перехвату учётных данных, доступу злоумышленника к критическим узлам, фишинговым атакам, манипуляциям внутри сети. По ГОСТ Р 50922–2006 и ISO/IEC 27001 аутентификация является обязательным атрибутом любого защищённого взаимодействия в информационных системах.

1.2 Модель факторов аутентификации

Все методы аутентификации подразделяются на три большие категории: 1. Фактор знания — “что пользователь знает” - пароли, - PIN-коды, - ответы на секретные вопросы. Слабость: легко забыть, можно украсть, перехватить, подобрать. 2. Фактор владения — “что у пользователя есть” - токен (OTP), - смарт-карта, - USB-ключ (например, FIDO U2F), - мобильное приложение-аутентификатор. Преимущество: физически привязан к пользователю. Недостаток: можно потерять или украсть. 3. Биометрический фактор — “кем является пользователь” - отпечаток пальца, - лицо, радужка, - голосовая биометрия Недостаток: нельзя сменить “биометрический пароль”, если он утёк.

Комбинация факторов

Метод “вызов-ответ” может использовать любой из факторов, либо их комбинацию внутри многофакторной аутентификации (MFA).

1.3 Требования к надёжной аутентификации

Согласно NIST SP 800-63 и ISO/IEC 29115, надежная аутентификация должна обеспечивать: 1. Конфиденциальность секрета – он не должен передаваться по сети. 2. Защиту от повторного воспроизведения (Replay) – каждое взаимодействие должно быть уникальным. 3. Доказательство владения секретом – злоумышленник, перехватив данные, не должен иметь возможность воспроизвести корректный ответ. 4. Защиту от MITM – данные не должны поддаваться подмене в процессе обмена. 5. Устойчивость к подбору и перебору – криптографические функции должны быть стойкими.

Метод «вызов–ответ» удовлетворяет каждому из этих требований.

1.4 Сравнительная классификация методов аутентификации

Сравнительная классификация методов представлена в таблице 1.

Таблица 1 - Сравнительная классификация

Критерий	Пароль	Биометрия	Токен OTP	Вызов-ответ
Защита секрета	низкая	средняя	высокая	**очень высокая**
Защита от Replay	нет	частично	да	**да, по определению**
Требует криптографии	нет	да	да	**да**
Устойчивость к MITM	низкая	средняя	средняя	высокая (при взаимной аутентификации)
Применимость	локально/онлайн	локально	онлайн/офлайн	**онлайн**
Степень доверия	средняя	высокая	высокая	**очень высокая**

2. Метод «вызов–ответ»

2.1 Историческая справка: как появился и развивался метод «запрос-ответ»

1970-е – первые идеи Идея «запрос-ответ» родилась почти одновременно с появлением компьютерных сетей и необходимостью защищённой удалённой аутентификации.

- 1976–1978 – Needham и Schroeder в своей знаменитой статье «Using Encryption for Authentication in Large Networks of Computers» (Communications of the ACM, Dec 1978) впервые формально описали протокол с nonce (number used once): сервер посылает случайное число R, клиент зашифровывает его общим ключом и возвращает. Это и есть первый в истории документированный протокол challenge-response.
- 1979 – в системе Kerberos (проект Athena, MIT) уже использовался полноценный симметричный challenge-response на основе DES и timestamp + nonce.

1980-е – стандартизация и первые реализации

- 1981 – Leslie Lamport публикует статью «Password Authentication with Insecure Communication» (CACM, Nov 1981) → хэш-цепочки Лампорта (S/KEY). Это первый широко известный однонаправленный challenge-response без передачи пароля в открытом виде.
- 1984–1988 – Bell Labs и AT&T Unix разрабатывают протокол S/KEY на основе идей Лампорта. Первое массовое внедрение одноразовых паролей как частного случая запрос-ответ.
- 1987–1993 – серия стандартов ISO/IEC 9798 «Entity Authentication»
    - Часть 2 (1993) – симметричные механизмы challenge-response
    - Часть 3 (1993) – механизмы на основе цифровой подписи
    - Часть 4 (1995) – с использованием криптографических check-функций Эти стандарты до сих пор (2025 год) являются базовыми для большинства современных протоколов.

1990-е – переход к публичным ключам

- 1994 – появление SSL 3.0 (Netscape) → первые массовые асимметричные challenge-response в веб-браузерах (сервер посылает nonce, клиент подписывает его своим сертификатом).
- 1996–1999 – протоколы EAP (Extensible Authentication Protocol) для PPP и позже для Wi-Fi (802.1X): EAP-MD5 (простейший challenge-response), EAP-TLS (сертификаты), EAP-TTLS, PEAP.

2000-е – борьба с фишингом и массовое внедрение

- 2003–2005 – RSA SecurID (аппаратные токены) → миллионы пользователей в банках и корпорациях используют TOTP-подобный challenge-response.
- 2008 – OpenID 2.0 и первые попытки убрать пароли из веба.

2010-е – рождение современных безпарольных стандартов

- 2012 – основана FIDO Alliance (Fast IDentity Online).
- 2014 – FIDO U2F (Universal 2nd Factor): первый массовый асимметричный challenge-response с аппаратным ключом (YubiKey, Google Titan). Схема: сайт посылает challenge → ключ подписывает его ECDSA → сайт проверяет.
- 2019 – выход стандартов FIDO2 + WebAuthn (W3C Recommendation) и CTAP2. Впервые браузер может делать полноценный challenge-response без плагинов.

2020-е – глобальный переход на Passkeys

- 2022 – Apple, Google и Microsoft объявляют о поддержке Passkeys (синхронизируемые ключи на основе WebAuthn).
- 2023–2024 – начало массового отключения паролей: Google (май 2023 – май 2025), Apple (iOS 16+), Microsoft (Entra ID passwordless by default).
- 2024 – NIST стандартизует постквантовые алгоритмы (Dilithium, Falcon, SPHINCS+). Chrome 128, Android 16, YubiKey 5.7 уже поддерживают Dilithium в WebAuthn.
- 2025 – Passkeys становятся самым распространённым способом аутентификации в мире (по данным FIDO Alliance – более 70 % новых регистраций). ### 2.2 Концепция и базовый принцип В компьютерной безопасности аутентификация по принципу «запрос — ответ» — это семейство протоколов, в которых одна сторона задаёт вопрос (“запрос”), а другая сторона должна предоставить верный ответ (“ответ”), чтобы пройти аутентификацию.

Самый простой пример протокола «запрос — ответ» — это аутентификация по паролю, где запрос — это запрос пароля, а действительный ответ — это правильный пароль.

Злоумышленник, который может подслушать аутентификацию по паролю, может пройти аутентификацию повторно, используя перехваченный пароль. Одно из решений — использовать несколько паролей, каждый из которых помечен идентификатором. Затем проверяющий может указать идентификатор, а проверяющий должен ответить правильным паролем для этого идентификатора. Если предположить, что пароли выбираются независимо друг от друга, то у злоумышленника, перехватившего одну пару «запрос — ответ», не будет подсказок для другого запроса в другое время.

Например, когда другие методы обеспечения безопасности связи недоступны, вооружённые силы США используют AKAC-1553 цифровой шифр TRIAD для аутентификации и шифрования некоторых сообщений. TRIAD включает в себя список трёхбуквенных проверочных кодов, которые проверяющий должен выбирать случайным образом, и случайные трёхбуквенные ответы на них. Для дополнительной безопасности каждый набор кодов действителен только в течение определённого периода времени, обычно 24 часов.

Другой базовый метод «запрос — ответ» работает следующим образом. Боб контролирует доступ к какому-то ресурсу, а Алиса пытается войти. Боб отправляет запрос «52w72y». Алиса должна ответить строкой символов, которая «соответствует» запросу Боба. «Соответствие» определяется заранее заданным алгоритмом, известным и Бобу, и Алисе. Правильный ответ может быть таким же простым, как «63x83z», при этом алгоритм изменяет каждый символ запроса с помощью шифра Цезаря. На самом деле

алгоритм гораздо сложнее. Боб каждый раз отправляет новый запрос, поэтому знание предыдущего правильного ответа (даже если он не зашифрован с помощью средств коммуникации) не позволяет злоумышленнику определить текущий правильный ответ.

При использовании метода «запрос-ответ» в КС заблаговременно создается и особо защищается массив вопросов, включающий в себя как вопросы общего характера, так и персональные вопросы, относящиеся к конкретному пользователю, например, вопросы, касающиеся известных только пользователю случаев из его жизни. Для подтверждения подлинности пользователя система последовательно задает ему ряд случайно выбранных вопросов, на которые он должен дать ответ. Опознание считается положительным, если пользователь правильно ответил на все вопросы. Основным требованием к вопросам в данном методе аутентификации является уникальность, подразумевающая, что правильные ответы на вопросы знают только пользователи, для которых эти вопросы предназначены. На рисунке 2.1 представлена общая схема метода.

Рис.2.1 Общая схема метода "вызов-ответ"

2.3 Использование шифра с симметричным ключом

Несколько подходов к установлению подлинности с помощью вызова-ответа используют шифрование с симметричными ключами. Секретность здесь - открытый ключ засекречивания, известный и претендент у и верификатору. Функция - алгоритм шифрования, с помощью которого обрабатывается вызов перед посылкой ответа.

Первый подход. В первом подходе верификатор отправляет претенденту nonce — случайное значение, используемое однократно и генерируемое заново для каждого сеанса аутентификации. Это число выступает в роли вызова (challenge). Получив его, претендент формирует ответ, используя секретный симметричный ключ, который известен только ему и верификатору. На рис. 2.2 показан принцип работы данного метода.

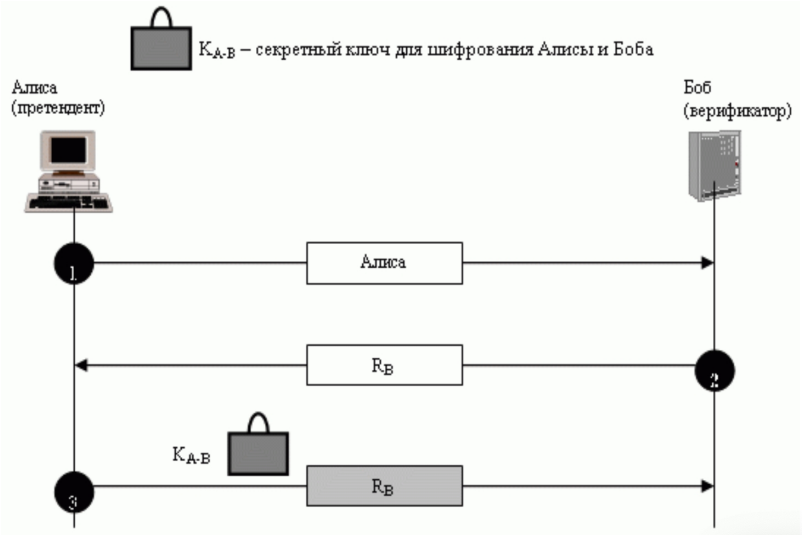


Рис.2.2 Принцип первого подхода

Начальное сообщение, отправляемое претендентом, не входит в сам механизм «вызов–ответ»; оно лишь уведомляет верификатор о намерении пройти аутентификацию. Второе сообщение — это собственно вызов: Rb, случайный nonce, выбранный верификатором (Бобом). Претендент шифрует этот nonce с помощью общего секретного ключа и отправляет полученный результат верификатору. После этого верификатор расшифровывает сообщение и сравнивает полученное значение с исходным nonce. Если значения совпадают, доступ претенденту (Алисе) предоставляется. Важно отметить, что для корректной работы обе стороны должны хранить один и тот же симметричный ключ, используемый для шифрования и расшифрования. Кроме того, верификатор обязан сохранять сгенерированный nonce до получения ответа, чтобы иметь возможность проверить его корректность. Следует подчеркнуть, что применение nonce исключает возможность повторного использования ответа злоумышленником. Даже если Ева перехватит третье сообщение, она не сможет выдать его за новый запрос аутентификации, поскольку каждый ответ действителен только для конкретного вызова. При следующей попытке верификатор сформирует новый nonce, и перехваченное значение станет бесполезным.

Второй подход. Во втором подходе для формирования вызова используется параметр, изменяющийся во времени, — метка времени. Поскольку текущее время непрерывно обновляется, оно естественным образом выполняет роль динамического вызова (implicit challenge). Верификатор передаёт претенденту значение времени, и предполагается, что часы обеих сторон синхронизированы, что является обязательным условием корректной работы метода. Благодаря этому претендент заранее знает ожидаемое значение вызова. Такая организация взаимодействия позволяет отказаться от явной передачи вызова как отдельного сообщения. Первое сообщение (инициирующее запрос) и третье сообщение (ответ) могут быть объединены, так как функция аутентификации рассчитывается на основе текущего времени. Таким образом, установление подлинности выполняется посредством одного сообщения — ответа на неявный вызов, которым выступает метка времени. На рисунке 2.3 схематично представлена работа данного подхода.

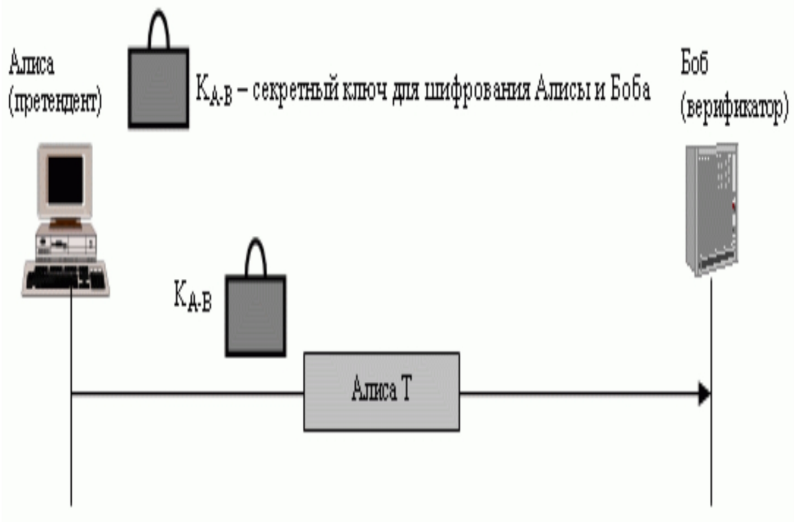
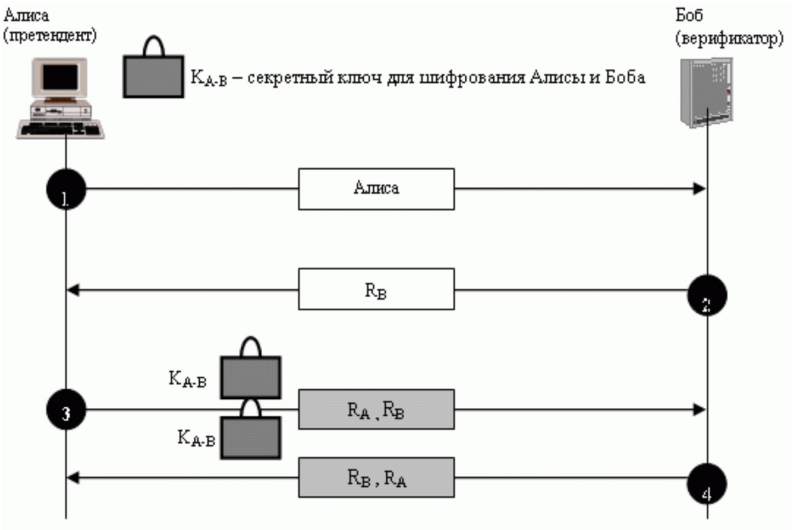


Рис.2.3 Принцип второго подхода

Третий подход. В первом и втором подходах описывается схема одностороннего установления подлинности: Боб проверяет, действительно ли перед ним Алиса, однако обратная проверка отсутствует. Если же необходимо, чтобы обе стороны убедились в подлинности друг друга, требуется использовать механизм взаимной (двунаправленной)

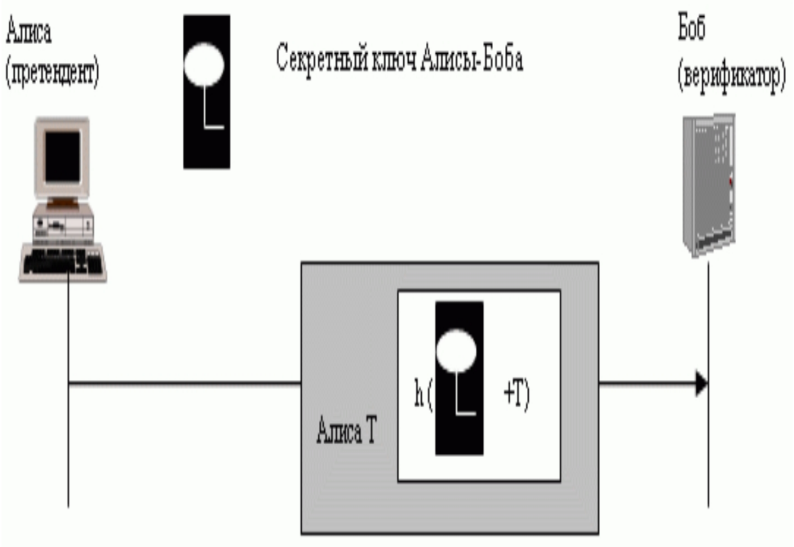
аутентификации. На рисунке 2.4 представлена соответствующая схема обмена.



Во втором сообщении Боб посылает Алисе свой вызов R^B. В третьем сообщении Алиса не только формирует ответ на вызов Боба, но и добавляет собственный вызов R^A, адресованный Бобу. Ответ Боба на этот новый вызов содержится в четвёртом сообщении. При этом порядок передачи значений R^A и R^B в последнем сообщении специально меняется для предотвращения атаки повторного воспроизведения: злоумышленник не сможет подставить фрагмент предыдущего сообщения как корректный ответ.

2.4 Использование функций ключевого хэширования

Вместо использования шифрования и дешифрования для проверки подлинности объекта можно применять ключевую хэш-функцию (MAC). Такой подход обладает важным преимуществом: он обеспечивает не только проверку подлинности, но и контроль целостности сообщений, поскольку результат вычисления MAC зависит как от данных, так и от секретного ключа. На рисунке 2.5 представлена схема формирования ответа на вызов, основанного на метке времени, с использованием ключевой хэш-

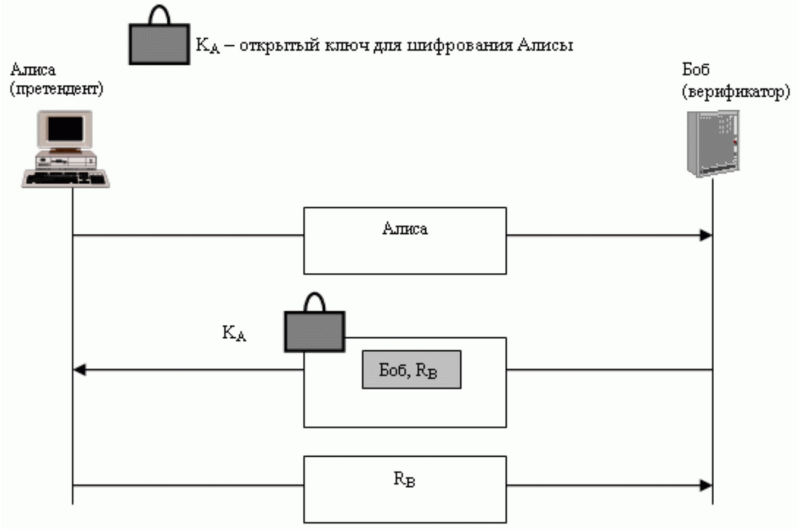


функции.

В данном случае текущее время передаётся одновременно в открытом виде и в виде значения, полученного путём вычисления MAC по этому же времени с использованием секретного ключа. Получив сообщение, Боб извлекает переданную метку времени и самостоятельно применяет к ней ту же ключевую хэш-функцию. Затем он сравнивает полученный результат с тем значением, которое прислала Алиса. Если оба значения совпадают, Боб считает сообщение подлинным, а Алису — успешно прошедшей аутентификацию.

2.5 Шифр, использующий асимметричный ключ

Вместо симметричного шифрования для установления подлинности может быть использована схема на основе асимметричных криптосистем. В этом случае секретом является закрытый ключ претендента, а его открытый ключ доступен верификатору и может быть опубликован. Задача претендента — доказать владение закрытым ключом, соответствующим известному открытому ключу. Процесс взаимодействия строится следующим образом: верификатор шифрует вызов (challenge) с помощью открытого ключа претендента; получив зашифрованное значение, претендент расшифровывает его своим закрытым ключом и возвращает результат — расшифрованный вызов. Если значение совпадает с тем, которое отправлял верификатор, подлинность претендента считается подтверждённой. Далее рассматриваются два варианта реализации такого подхода: один предназначен для одностороннего подтверждения подлинности, второй обеспечивает взаимную (двунаправленную) аутентификацию. **Первый подход (односторонняя аутентификация).** В этом варианте Боб шифрует случайно выбранный параметр, используя открытый ключ Алисы. Получив сообщение, Алиса расшифровывает его своим закрытым ключом и отправляет значение nonce обратно Бобу. Если переданное Алисой значение совпадает с исходным вызовом, Боб подтверждает её подлинность. Принцип работы этого метода показан на рис. 2.6.



Второй подход. Во втором подходе реализуется схема взаимной аутентификации на основе асимметричных ключей, где в каждом направлении используется свой открытый ключ. Алиса отправляет Бобу свой идентификатор и значение nonce, предварительно зашифровав их открытым ключом Боба. После получения сообщения Боб расшифровывает его с помощью своего закрытого ключа, формирует собственный

вызов и передаёт Алисе свой nonce, зашифровав его открытым ключом Алисы. В завершение Алиса расшифровывает присланный ей вызов и возвращает Бобу исходное значение nonce, подтверждая тем самым подлинность своей личности и корректность обмена. Такой подход представлен на рисунке 2.7.

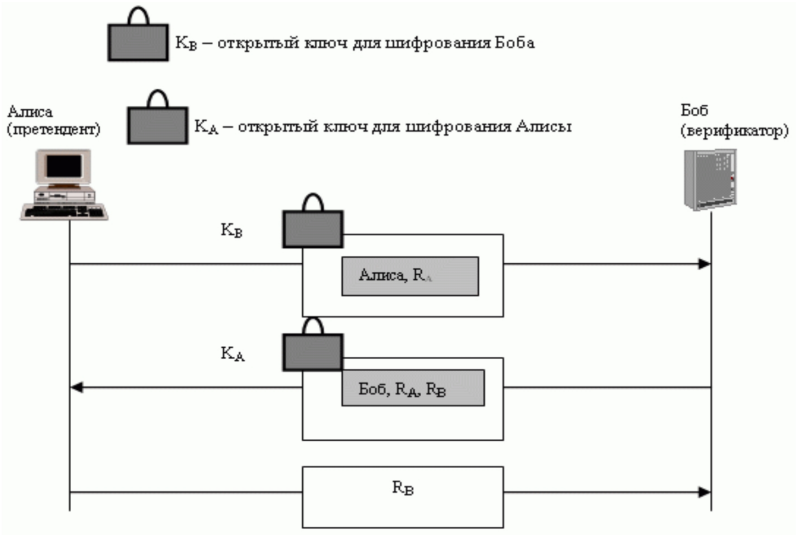
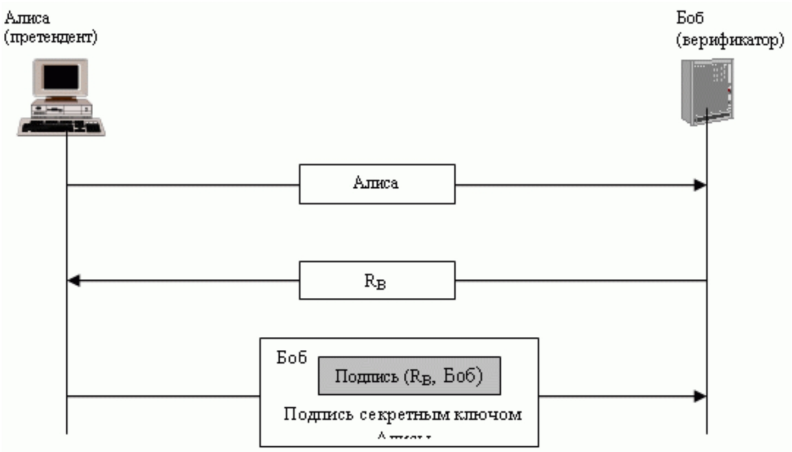


Рис.2.7 Схема первого подхода

2.6 Использование цифровой подписи

Установление подлинности объекта может быть реализовано также с использованием цифровой подписи. В такой схеме претендент применяет свой закрытый ключ для создания подписи, демонстрируя тем самым владение секретом, связанным с общедоступным открытым ключом. Рассмотрим два варианта реализации данного подхода; остальные схемы могут быть выведены аналогично. **Первый подход.** В первом варианте, представленном на рисунке 2.8, Боб формирует вызов в виде исходного открытого текста, а Алиса использует свой закрытый ключ для подписания ответа. После получения сообщения Боб проверяет подпись с помощью открытого ключа Алисы и тем самым подтверждает её подлинность.



Второй подход. Во втором варианте, показанном на рисунке 2.9, реализуется взаимная аутентификация. Алиса и Боб последовательно обмениваются вызовами и соответствующими подписями, каждая из которых формируется закрытым ключом отправителя и проверяется открытым ключом получателя. В результате обе стороны убеждаются в подлинности друг друга.

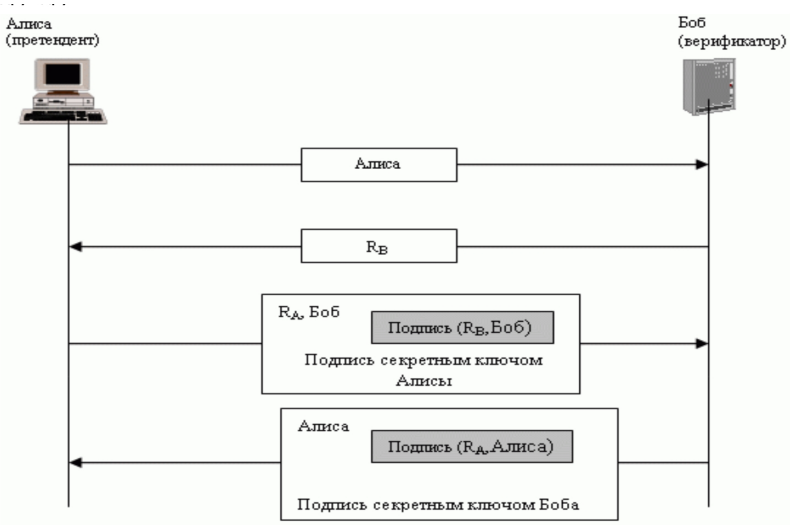


Рис.2.9 Схема второго подхода

2.7 Проблемы и ограничения метода “запрос–ответ”

Несмотря на высокую криптографическую стойкость и широкую применимость, метод установления подлинности «запрос–ответ» имеет ряд ограничений и потенциальных уязвимостей, которые необходимо учитывать при проектировании защищённых систем. Эти проблемы обусловлены как особенностями криптографических примитивов, так и архитектурой взаимодействия между участниками протокола. 1. Зависимость от качества генератора случайных чисел Корректность работы challenge–response напрямую зависит от того, насколько непредсказуемым и случайным является генерируемый вызов (nonce). Если генератор случайных чисел слабый, детерминированный или предсказуемый, злоумышленник может: - заранее угадать вызов; - провести атаку по предсказанию следующего nonce; - подменить или подстроить challenge так, чтобы получить валидный ответ.

Примеры подобных атак фиксировались в устаревших реализациях Kerberos и PPP CHAP, где слабая энтропия приводила к снижению стойкости всей системы.

2. Проблема хранения секретов Для любого варианта реализации challenge–response требуется хранение секретных данных:

- симметричного ключа,
- приватного ключа,
- мастер-ключа (в OTP-системах),
- ключей серверной базы (Kerberos KDC).

Основные риски: - компрометация ключа приводит к мгновенному нарушению безопасности; - на сервере часто приходится хранить секреты в явном виде (для MAC); - устройства (токены, смартфоны) могут быть украдены или подвергнуты взлому.

Поэтому рекомендуются безопасные среды хранения: HSM, TPM, Secure Enclave, SGX, аппаратные токены.

3. Уязвимость к MITM при отсутствии взаимной аутентификации Односторонние схемы (client → server) подвержены атаке «человек посередине» (MITM).

Злоумышленник может: - перехватывать вызовы и ответы, - проксировать запросы к настоящему серверу, - маскироваться под легитимного участника.

Без взаимной аутентификации сервер не может доказать свою подлинность клиенту, и пользователь может взаимодействовать с поддельным узлом, что особенно опасно при фишинге и spoofing-атаках.

4. Требования к синхронизации времени Для схем на основе временной метки (например, TOTP):

- необходима точная синхронизация часов,
- отклонение всего в несколько секунд может привести к отказу в доступе,
- необходимо управление «временными окнами»,
- возникает риск дрейфа времени у слабых устройств.

Это актуально для мобильных токенов, банкоматов, смарт-карт и IoT-устройств.

5. Зависимость от каналов и протоколов передачи Challenge–response защищает от повторного воспроизведения, но не защищает сам канал передачи.

Если данные идут по незашифрованному соединению, злоумышленник может: - перехватить challenge, - подменить сообщения, - и даже не нарушив протокола, организовать MITM-посредничество.

Поэтому всегда требуется использование защищённого канала: TLS, DTLS, IPSec, WPA2-Enterprise.

6. Уязвимости реализации (implementation flaws) Даже идеальная криптография не защищает от:

- ошибок в коде (например, неверная обработка padding в RSA);
- утечек информации через побочные каналы время выполнения, энергопотребление, электромагнитные излучения;
- недоверенных библиотек (старые версии OpenSSL, GnuTLS, NSS);
- ошибок в протоколах как в старых реализациях MS-CHAP v2 или WEP.

7. Проблемы масштабируемости В симметричных схемах:

- для N пользователей необходимо хранить N ключей,
- при прямой аутентификации между всеми парами объектов количество ключей растёт как $O(N^2)$.

Это делает схему масштабируемой только при наличии централизованного сервера доверия (KDC).

8. Ограничения производительности Некоторые реализации накладывают высокие вычислительные требования:

- асимметричные операции RSA/ECDSA требуют значительных ресурсов;
- на IoT-устройствах асимметрия часто невозможна из-за недостатка процессорной мощности;
- системы с высокой частотой запросов (аутентификация на портах, DDoS-защита) могут столкнуться с нагрузками.

9. Зависимость от криптографической стойкости используемых примитивов Протокол становится небезопасным, если:

- используется устаревший алгоритм (MD5, SHA-1, DES, RSA < 1024 бит);
- длина ключа недостаточна;
- хэш-функция уязвима к коллизиям или атаке расширения сообщения.

Особенно актуально в условиях появления: - квантовых вычислений, - атак на эллиптические кривые, - публичных взломов старых PKI-систем.

10. Влияние человеческого и организационного фактора Challenge–response не защищает от:

- социальных атак (social engineering),
- компрометации конечных устройств,
- неправильного администрирования,
- несвоевременного обновления ключей,
- плохого управления жизненным циклом сертификатов.

Например, пользователи могут потерять токены или игнорировать уведомления о недействительных сертификатах.

Заключение

В ходе проведённого исследования была рассмотрена проблема обеспечения подлинности субъектов и объектов в информационных системах, а также изучены современные подходы к реализации механизмов аутентификации, основанных на методе «запрос–ответ» (challenge–response). Данный метод является одним из наиболее надёжных и теоретически обоснованных способов подтверждения владения секретной информацией без её раскрытия, что особенно важно в условиях возрастания количества кибератак и постоянного усложнения моделей угроз. Анализ существующих решений показал, что challenge–response имеет универсальную природу и может быть реализован на основе различных криптографических примитивов: симметричного шифрования, ключевого хэширования (HMAC), асимметричного шифрования и цифровой подписи. Каждый из этих подходов обладает своими преимуществами, ограничениями и характерными областями применения — от простых смарт-карт и банковских токенов до высокозащищённых корпоративных инфраструктур, протоколов TLS/SSH, систем многофакторной аутентификации и государственных информационных систем. Историческая справка, включающая анализ работ Лесли Лампорта, протоколов CHAP, Kerberos, HOTP/TOTP и других схем, позволяет увидеть эволюцию метода: от простых текстовых пар «вызов–ответ» до современных криптографически стойких систем, работающих в рамках Zero Trust-архитектур. Особо важно, что при правильной реализации challenge–response способен обеспечить защиту от таких угроз, как перехват, повторное воспроизведение, подбор, подмена, атаки посредника и фишинг, что делает его ключевым компонентом современных систем информационной безопасности. Актуальность исследуемой темы подтверждается тенденциями: массовыми утечками паролей, развитием распределённых вычислений, ростом удалённого доступа, а также переходом организаций на архитектуры, исключающие implicit trust. В этой среде challenge–response выступает одним из центральных механизмов, позволяющих поддерживать высокий уровень доверия между взаимодействующими сторонами. Подводя итог, можно утверждать, что метод «запрос–ответ» остаётся фундаментальным криптографическим инструментом обеспечения подлинности, устойчивым к широкому спектру атак и адаптивным к различным технологиям и протоколам. Его использование представляет собой необходимый компонент надёжной аутентификации в современных информационных системах и будет сохранять свою значимость в дальнейшем, по мере усложнения цифровой среды и усиления требований к безопасности.

Список используемых источников

1. Кушниренко, А. Г. Криптографические методы защиты информации: учебное пособие. — Санкт-Петербург: Университет ИТМО, 2018. — 210 с.

2. Объектная аутентификация: лекция 15 // Материалы курса “Математические основы защиты информации”. — mathsec, 2020. (файл: mathsec_lection15-object-authentication_16x9.pdf)

3. Лекция «Установление подлинности объекта» // Учебные материалы по криптографии. — 2019. (файл: Установление подлинности объекта.pdf)

4. Stallings, W. Cryptography and Network Security: Principles and Practice. — 7th ed. — Pearson, 2017. — 752 p.

5. Kaufman, C., Perlman, R., Speciner, M. Network Security: Private Communication in a Public World. — 3rd ed. — Prentice Hall, 2016. — 888 p.

6. Menezes, A., van Oorschot, P., Vanstone, S. Handbook of Applied Cryptography. — CRC Press, 1996. — 816 p.

7. Wikipedia contributors. Challenge–response authentication // Wikipedia, The Free Encyclopedia. URL: https://en.wikipedia.org/wiki/Challenge–response_authentication

8. NIST Special Publication 800-63-3. Digital Identity Guidelines. — NIST, 2017. URL: https://pages.nist.gov/800-63-3

9. RFC 4226 — OATH HOTP Algorithm: An HMAC-Based One-Time Password Algorithm. — IETF, 2005

10. RFC 6238 — TOTP: Time-Based One-Time Password Algorithm. — IETF, 2011

11. Переводная статья: Challenge–response authentication // Wikipedia (RU). URL: https://translated.turbopages.org/... (дата обращения: 20.01.2025)

12. Studfile. Аутентификация. Методы и протоколы. URL: https://studfile.net/preview/321440/page:4/

13. Лампорта, Л. One-Time Password Authentication. — Bell Labs Technical Report, 1981

14. Dworkin, M. Recommendation for Block Cipher Modes of Operation. — NIST SP 800-38A, 2001

15. ISO/IEC 9798-4. Entity Authentication Mechanisms Using Symmetric Encryption Algorithms. — International Organization for Standardization, 1999