Аппаратно защищенные микросхемы семейства CryptoAuthentication: потенциальные применения ATSHA204A

Игорь КРИВЧЕНКО
ik@efo.ru

Как известно, аутентификация может быть выполнена двумя основными способами — симметричным и асимметричным. Основное отличие между ними состоит в том, каким образом используются секретные ключи. Если и на стороне хоста, и на стороне клиента применяется один и тот же ключ, то аутентификация — симметричная. Если же используется математически связанная пара открытого и закрытого ключей, то аутентификация — асимметричная.

Асимметричная аутентификация также носит название инфраструктуры открытых ключей (или PKI). Она очень хорошо подходит под требования окружающего нас мира. Фактически Интернет является одним из главных пользователей PKI. На практике столь же часто применяются комбинации симметричного и асимметричного подходов с их плюсами — скоростью работы симметричного и безопасностью асимметричного. Компания Atmel выпускает микросхемы для обоих типов аутентификации.

В статье рассмотрены некоторые примеры организации дополнительного уровня защиты информации во встраиваемых системах на базе недорогой криптографической микросхемы ATSHA204A. Она поддерживает криптографический алгоритм SHA-256 и отлично подходит для задач симметричной аутентификации.

Симметричная аутентификация по схеме «запрос — ответ» с хранением секретного ключа на стороне хоста

При взаимной авторизации между хостом и клиентом обе стороны используют один и тот же секретный ключ шифрования. В нашем примере и клиент, и хост содержат микросхемы ATSHA204A, в защищенную память которых заранее запрограммирован одинаковый секретный ключ (рис. 1).

Процесс начинается, когда хост посылает клиенту случайное число-запрос, созданное встроенным генератором случайных чисел микросхемы ATSHA204A (шаг 1). Например, это происходит в тот момент, когда аккумуляторный блок вставляется в брендовый шуруповерт. Клиент, получив этот запрос, подает его на вход хеш-алгоритма SHA-256 вместе с хранящимся у него в памяти секретным ключом. В результате получается новое число-ответ, которое также называют кодом аутентификации сообщения, или MAC (Message Authentication Code). Ответ пересылается в хост. Вместе оба действия составляют шаг 2. На следующем, шаге 3, хост у себя прогоняет то же самое случайное число, которое он посылал клиенту, через хеш-алгоритм с использованием секретного ключа, хранящегося в его памяти. Затем хост сравнивает два дайджеста — только что вычисленный и полученный от клиента. В случае совпадения клиент считается верифицированным, то есть прошедшим аутентификацию.

Основные особенности:

- быстрый процесс симметричной аутентификации;
- защищенные криптографические микросхемы на обеих сторонах обеспечивают безопасное хранение секретных ключей.

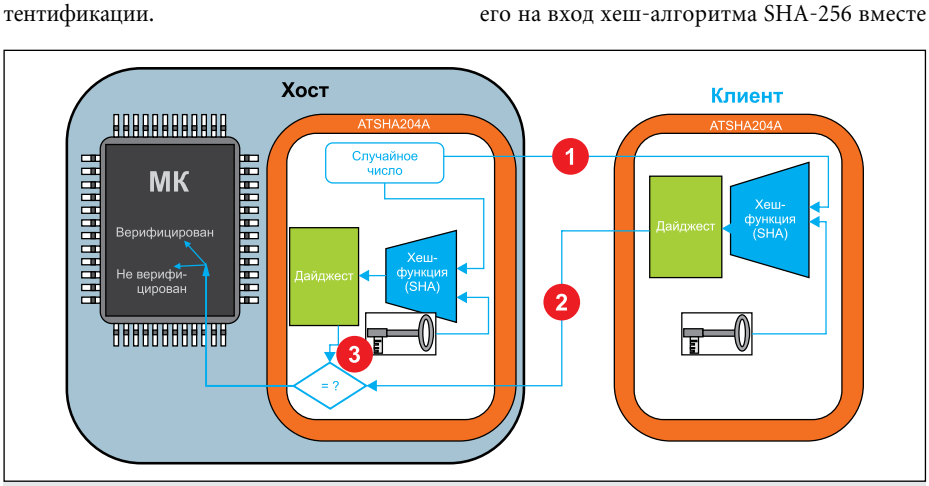

Рис. 1. Симметричная аутентификация по схеме «запрос — ответ» с хранением секретного ключа на стороне хоста

Симметричная аутентификация по схеме «фиксированный запрос — ответ» без хранения секретного ключа на стороне хоста

Симметричная аутентификация может быть выполнена и без криптографической микросхемы на стороне хоста. Такой подход носит название «фиксированный запрос». Здесь хост уже не использует случайные числа — вместо этого применяются определенные пары заранее вычисленных чисел (рис. 2).

Для вычисления ответа каждый фиксированный запрос заранее прогоняется при производстве конечного изделия через хеш-алгоритм с помощью определенного секретно-

го ключа. Значения запросов и предварительно вычисленных, соответствующих им ответов записываются в энергонезависимую память микроконтроллера на стороне хоста. Во все микросхемы ATSHA204A на стороне клиентов системы записывается этот же секретный ключ.

Когда хост посылает предварительно загруженный в его память фиксированный запрос клиенту с целью аутентификации (шаг 1), клиент должен запустить у себя такой же хеш-алгоритм над этим фиксированным запросом для генерации ответа. Затем он отсылает полученный ответ обратно (шаг 2). Хост сравнивает полученный и предварительно вычисленный ответы (шаг 3). В случае совпадения клиент считается верифицированным.

Основные особенности:

- быстрый процесс симметричной аутентификации;
- не требуется безопасное хранение секретных ключей на стороне хоста.

Такой подход может применяться для защиты программного обеспечения (ПО) и в проектах, не предусматривающих необходимость хранения секретных данных на стороне хоста. Он может реализовываться на простых маломощных микроконтроллерах с очень низкой стоимостью, поскольку на стороне хоста не требуется выполнение объемных математических вычислений. Но у аутентификации с фиксированным запросом есть и серьезный недостаток — относительно невысокая криптостойкость. Злоумышленник, хакер или криптоаналитик (общий термин — «атакующий») может использовать логический анализатор на шине информационного обмена для перехвата сообщений и раскрытия секретов.

К счастью, существует достаточно простой способ повышения уровня секретности к сценарию «фиксированный запрос – ответ»: добавление промежуточного ключа.

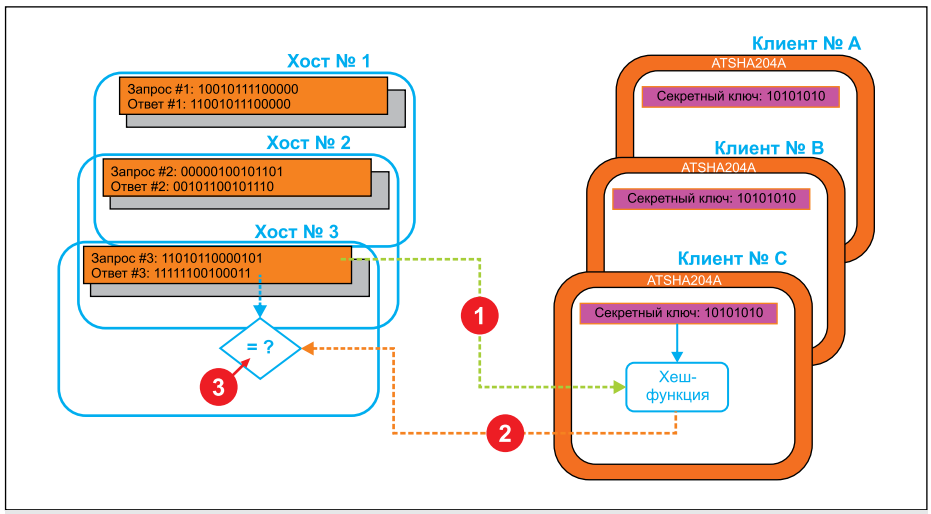

Рис. 2. Симметричная аутентификация по схеме «фиксированный запрос — ответ» без хранения секретного ключа на стороне хоста

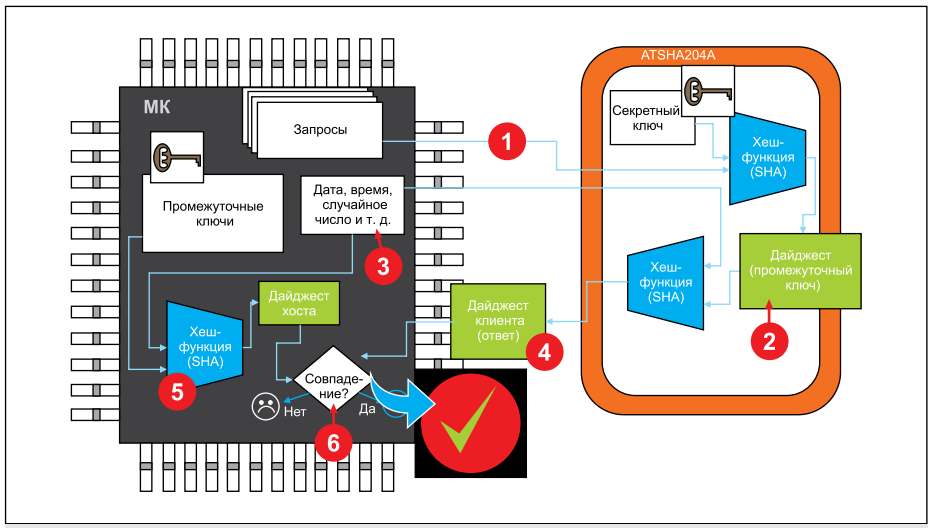

Рис. 3. Симметричная аутентификация по схеме «фиксированный запрос — ответ» с промежуточным ключом и без хранения секретного ключа на стороне хоста

Симметричная аутентификация по схеме «фиксированный запрос — ответ» с промежуточным ключом и без хранения секретного ключа на стороне хоста

Этот подход реализуется путем дополнительной процедуры хеширования с промежуточным ключом (рис. 3). Как и в предыдущем случае, значения запросов и предварительно вычисленных и связанных с ними ответов заранее записываются в энергонезависимую память микроконтроллера хоста. Но теперь эти ответы становятся промежуточными ключами шифрования.

Процесс начинается с запроса, который хост пересылает клиенту (шаг 1). Этот запрос хешируется в ATSHA204A на стороне клиента с секретным ключом (шаг 2). Полученный дайджест (то есть промежуточный ключ) затем вновь хешируется, но уже со случайным числом (например, комбинацией текущих значений даты и времени), которое генерируется хостом и посылается клиенту (шаг 3). Это уникальное случайное число используется лишь один раз, поэтому Atmel называет его Nonce (Number used once). Дайджест промежуточного ключа и Nonce является окончательным ответом клиента, который отправляется хосту (шаг 4) и сравнивается там с дайджестом, полученным хостом путем таких же вычислений с этими же данными (шаг 5). При совпадении дайджестов (шаг 6) клиент считается верифицированным.

Отметим, что ответ клиента каждый раз будет новым, потому что случайное число Nonce каждый раз тоже будет разным. Это значительно повышает уровень информационной безопасности системы — теперь применение логического анализатора не даст атакующему требуемого результата.

Основные особенности:

- быстрый процесс симметричной аутентификации;
- не требуется безопасное хранение секретных ключей на стороне хоста.

Защита загружаемого кода при помощи симметричных ключей

Защита интеллектуальной собственности в виде кода микропрограммы (firmware), по-видимому, является самой обширной областью приложений. Этот сегмент включает практически любую встраиваемую систему с обновляемым или модифицируемым ПО, базирующуюся на микропроцессоре или микроконтроллере. В настоящее время также все чаще применяется удаленное внесение исправлений, обновлений и коррекция ошибок микропрограмм. И здесь необходимо помнить, что базирующиеся только на ПО решения информационной безопасности, как правило, беззащитны перед атакующим, потому что могут быть при желании легко взломаны.

Устройства CryptoAuthentication помогают защищать загружаемую в систему микропрограмму путем недопущения получения

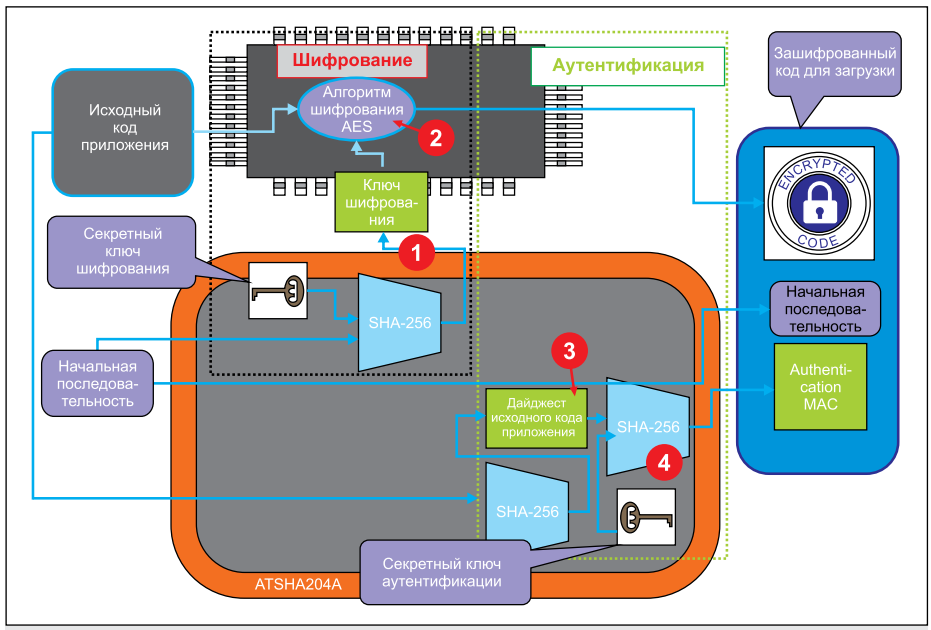

Рис. 4. Защита загружаемого кода при помощи симметричных ключей. Шифрование кода приложения и получение MAC

хакерами ее образов и алгоритмов. Это гарантирует, что только оригинальное, не модифицированное firmware от производителя загружается в энергонезависимую память системы для последующего исполнения.

Данный подход использует и шифрование, и аутентификацию. На первом этапе (рис. 4) на стороне хоста осуществляется шифрование кода приложения и получение MAC. Шифрование микропрограммы перед отправкой целевому клиенту не позволяет хакерам использовать ее в случае перехвата. Для зашифровки исходного кода разработчику потребуется создать ключ шифрования. Это реализуется в микросхеме ATSHA204A, которая хранит секретный ключ и хеширует его вместе с некоторой начальной последовательностью (шаг 1). Полученный дайджест (ключ шифрования) подается на вход алгоритма шифрования, после чего разработчик получает загружаемую в устройство целевого клиента зашифрованную копию (шаг 2). Начальная последовательность будет переслана клиенту вместе с зашифрованным кодом, чтобы можно было осуществить расшифровку.

Аутентификация подготавливается путем хеширования (или дополнения) оригинального кода приложения до размера, при котором он может быть корректно хеширован с секретным аутентификационным ключом (шаг 3). Отметим, что аутентификационный ключ не совпадает с предыдущим ключом, использовавшимся исключительно для шифрования. Дайджест исходного кода приложения хешируется в микросхеме ATSHA204A еще раз, теперь совместно с ключом аутентификации (шаг 4). Результирующий дайджест называется Authentication MAC, который, по сути, представляет собой некий аналог цифровой подписи. Он отправляется клиенту вместе с зашифрованным кодом приложения и использовавшейся при шифровании начальной последовательностью.

Второй этап (рис. 5) на стороне клиента включает операции загрузки, дешифрования и аутентификации. Полученный код сначала расшифровывается, а затем проводится его аутентификация для подтверждения подлинности. Целевой клиент защищенного ПО получает от хоста зашифрованную версию кода, начальную последовательность и Authentication MAC. Для расшифровывания кода начальная последовательность хешируется с секретным ключом, который хранится в микросхеме ATSHA204A на стороне клиента (шаг 1). Напомним, что это тот же самый секретный ключ, который использовался и на стороне хоста. Результатом хеширования является ключ расшифровки, естественно, совпадающий с ключом зашифровки. Микропроцессор встраиваемой системы (клиента) восстанавливает код приложения (шаг 2), используя данный ключ.

Затем процесс переходит в стадию аутентификации. Расшифрованный код обрабатывается (шаг 3) по такой же схеме, по какой это осуществлялось на стороне хоста, а затем хешируется в криптографическом устройстве (шаг 4) вместе с секретным аутентификационным ключом клиента, который совпадает с таким же ключом хоста. Результирующий клиентский Authentication MAC сравнивается с полученным Authentication MAC хоста (шаг 5). Если дайджесты совпадают, то полученный исходный код считается подлинным и может быть загружен в микроконтроллер целевой системы (шаг 6).

Основные особенности:

- можно создавать уникальные копии загрузок для каждого клиента путем привязки, например, к его серийному номеру.
- все загрузки будут верифицированными;
- пользователь может свободно разместить зашифрованный образ загружаемого кода в Сети.

Обмен симметричным сеансовым ключом (с криптографическими устройствами на обеих сторонах)

Идея реализации (рис. 6) состоит в создании ключа шифрования, который изменяется для каждой сессии или сеанса обмена данными. Отсюда и происходит название «сеансовый ключ».

С помощью встроенного генератора случайных чисел микросхема ATSHA204A соз-

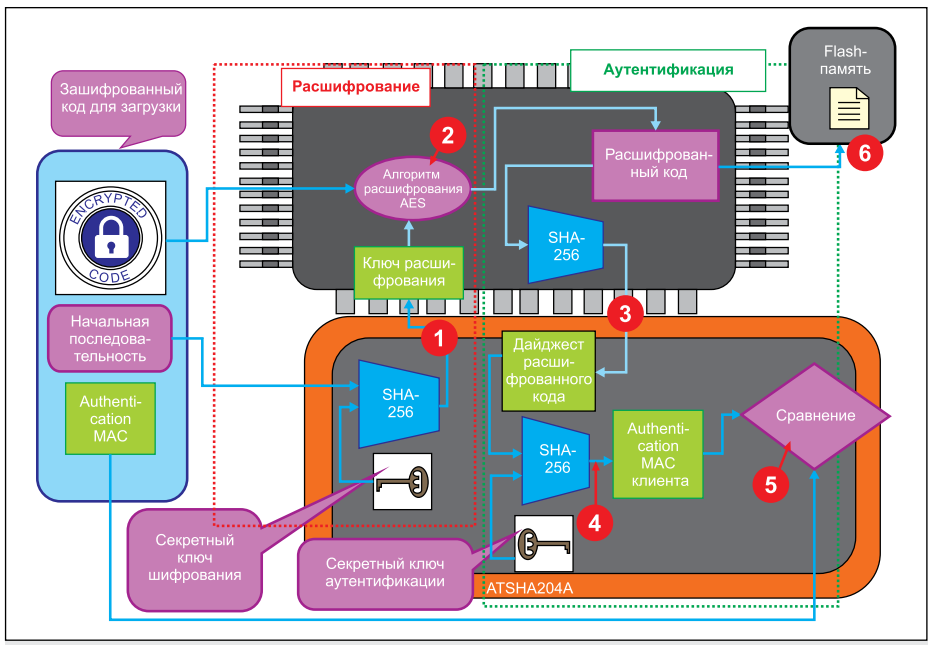

Рис. 5. Защита загружаемого кода при помощи симметричных ключей. Загрузка, дешифровка и аутентификация

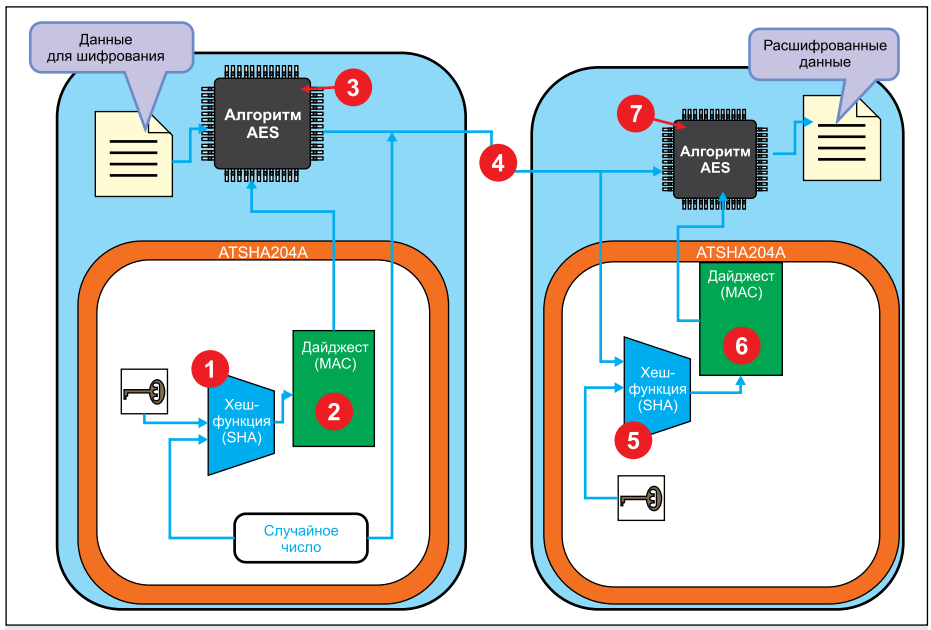

Рис. 6. Обмен симметричным сеансовым ключом (с криптографическими устройствами на обеих сторонах)

дает некоторое случайное число и хеширует его с хранящимся в ее памяти секретным ключом (шаг 1). От полученного дайджеста берутся первые или вторые 16 байт (128 бит), которые становятся сеансовым ключом шифрования AES (шаг 2). Этот ключ пересылается в управляющий микроконтроллер для выполнения процедуры шифрования требуемых исходных данных по алгоритму AES-128 (шаг 3).

Зашифрованные данные и использовавшееся случайное число затем пересылаются на другую сторону (шаг 4). Конечно, для расшифровки данных требуется еще точно такой же ключ, использованный при шифровании. Именно поэтому второй стороне пересылается случайное число — для восстановления сеансового ключа с помощью такого же секретного ключа, хранящегося здесь в микросхеме ATSHA204A (шаг 5). Поскольку секретные ключи на обеих сторонах одинаковы, то при операции хеширования по алгоритму SHA-256 с одним и тем же случайным числом на каждой из сторон будут получены одинаковые дайджесты. Теперь на стороне расшифровки точно так же первые или вторые 16 байт полученного дайджеста берутся в качестве сеансового ключа для алгоритма AES-128 (шаг 6). Этот ключ используется управляющим микроконтроллером на второй стороне для получения искомых данных из зашифрованного сообщения путем выполнения алгоритма дешифрования AES-128 (шаг 7).

Основные особенности:

- ключи шифрования изменяются для каждого сеанса, что повышает уровень безопасности;
- очень простая, но защищенная методология;
- легко может быть реализовано с помощью микросхем ATSHA204A.

Защита информационного канала между криптографическим устройством и микроконтроллером

Иногда возникают ситуации, когда нужно периодически проверять, не атакуется ли канал связи между криптографическим устройством, отвечающим за аутентификацию, и управляющим микроконтроллером.

При использовании в качестве устройства аутентификации микросхем семейства CryptoAuthentication подтверждение того, что аутентификация прошла успешно, может поступать на внешний вывод управляющего микроконтроллера в виде обычного логического сигнала «0» или «1». Этот сигнал является результатом выполнения встроенной функции CheckMAC. Поэтому атакующий может просто присоединить кнопку к линии связи между двумя устройствами и постоянно подавать на нее сигнал успеха, «обманывая» управляющий микроконтроллер.

Проверка может быть осуществлена путем выполнения дополнительного цикла верификации результата, полученного от функции CheckMAC. Для этого требуется второй секретный ключ, который хранится в памяти криптографической микросхемы и вычисляется программой микроконтроллера.

На рис. 7 показано, что после выполнения команды CheckMAC в ATSHA204A (шаг 1) наступает второй этап — проверка подлинности полученного логического ответа. Для этого дополнительный ключ 2, хранящийся в памяти микросхемы ATSHA204A, сразу же загружается во внутренний служебный регистр TempKey (шаг 2). В то же время микроконтроллер, получив логический сигнал предполагаемого подтверждения успешно прошедшей перед тем аутентификации, генерирует некоторое случайное число (например, текущее время дня). Это число пересылается в криптографическую микросхему (шаг 3) и хешируется там вместе с ключом 2. Полученный дайджест (шаг 4) отправляется обратно в микроконтроллер для последующего сравнения. Микроконтроллер одновременно с ATSHA204A выполняет такую же операцию хеширования (шаг 5) над этим же случайным числом и ключом 2, вычисляя его «на месте» в коде программы. Затем результаты сравниваются (шаг 6), и в случае совпадения полученный ранее логический сигнал считается верным.

Описанный процесс — стандартная процедура поверки выдаваемого аутентифицируемым устройством логического сигнала подтверждения. Атакующему становится гораздо сложнее «обманывать» систему, постоянно посылая сигнал успеха «1» в линию связи между микроконтроллером и криптографической микросхемой.

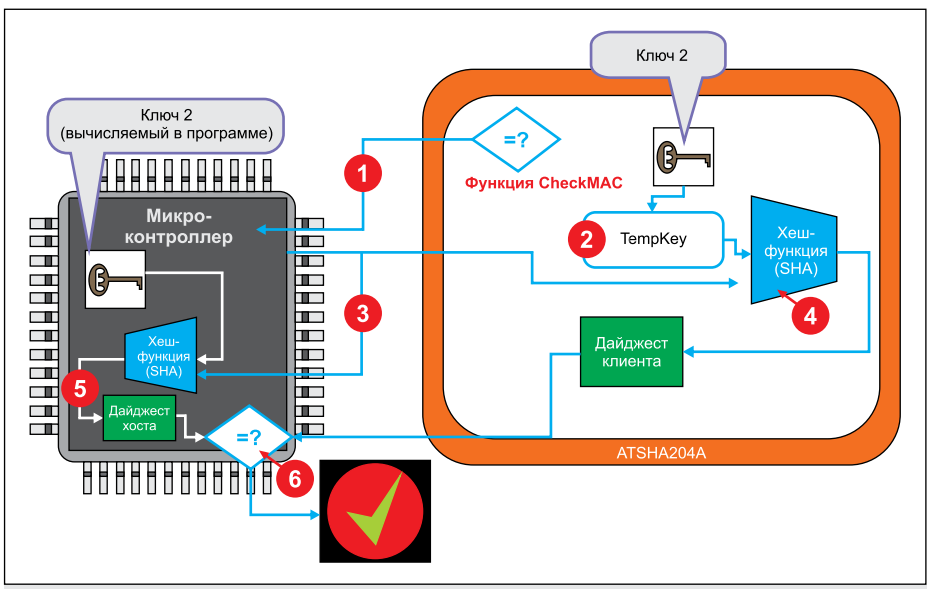

Рис. 7. Защита информационного канала между криптографическим устройством и микроконтроллером

Основные особенности:
- обеспечивается проверка подлинности канала связи между микроконтроллером и аутентифицируемым устройством;
- функционал аппаратно реализован в любой микросхеме семейства CryptoAuthentication для простоты использования.

Защищенное хранение данных с использованием симметричного шифрования

Напомним, что одна из основных функций микросхем семейства CryptoAuthentication — безопасное хранение ключей шифрования в защищенном аппаратном окружении. При работе распределенных систем сбора и обработки данных, привязанных к банкам данных на удаленных серверах или облаках, защищенный с точки зрения информационной безопасности обмен данными очень важен. Он предполагает использование проверенного и надежного алгоритма шифрования и ключей шифрования, которые должны храниться в безопасном месте. Возможности криптографических устройств CryptoAuthentication в этом смысле выглядят более предпочтительными по сравнению с традиционными решениями на базе ПО, потому что ПО гораздо проще взломать и, следовательно, раскрыть ключи шифрования.

Пусть удаленному клиенту необходимо получить некоторую информацию от корпоративного сервера, и ее по каким-то причинам нежелательно в открытом виде передавать по Сети или общедоступным каналам связи. Следовательно, данные нужно предварительно зашифровать. Но для последующего расшифровывания на стороне удаленного клиента потребуется ключ шифрования. Возникает задача его пересылки по тем же каналам связи без риска раскрытия, решить которую могут помочь микросхемы CryptoAuthentication.

В приведенном примере (рис. 8) используется симметричное шифрование. Информация, предназначенная для отправки клиенту, зашифровывается на стороне сервера быстрым симметричным алгоритмом AES. Сеансовый ключ шифрования для этой операции формируется путем хеширования некоторой начальной последовательности (случайного числа) и секретного ключа, надежно и безопасно хранящегося на стороне шифрования. Зашифрованные данные и начальная последовательность сохраняются на сервере, а сеансовый ключ уничтожается.

Затем клиенту пересылается искомая начальная последовательность (шаг 1) и собственно зашифрованная информация (шаг 2). Зашифрованные данные принимаются системным микроконтроллером, а начальная последовательность — микросхемой CryptoAuthentication, в памяти которой хранится тот же самый секретный ключ, что и на удаленном сервере. После хеширования в защищенной аппаратной среде криптографического устройства из секретного ключа и начальной последовательности воссоздается сеансовый ключ шифрования AES (шаг 3), который передается в управляющий микроконтроллер (шаг 4). Полученные на шаге 2 данные расшифровываются микроконтроллером с помощью созданного сеансового ключа по алгоритму симметричного шифрования AES (шаг 5) и затем могут быть использованы по назначению в открытом, исходном виде.

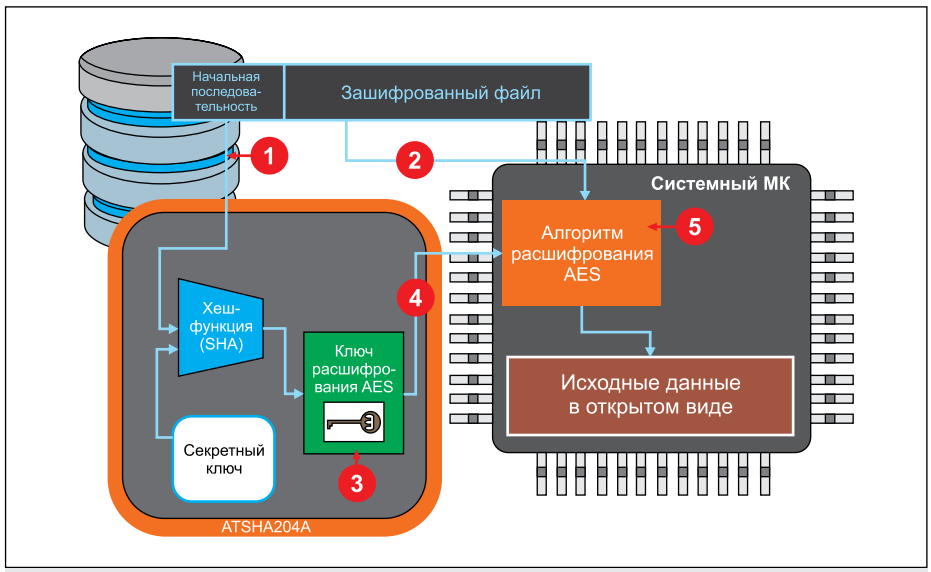

Рис. 8. Защищенное хранение данных с использованием симметричного шифрования

Отметим, что поскольку многие современные 32-разрядные и даже некоторые 8-разрядные микроконтроллеры имеют на кристалле аппаратный криптоакселератор AES, то реализация подобной схемы защищенного обмена данными достаточно проста. В систему добавляется один недорогой кристалл, который также можно использовать для хранения других критичных данных небольшого объема — логов событий, калибровочных коэффициентов и т. п.

Основные особенности:
- безопасное хранение ключей шифрования в защищенной аппаратной среде;
- функционал аппаратно реализован в любой микросхеме семейства CryptoAuthentication для простоты использования.

Защита паролей

Иногда требуется сравнивать вводимый в систему пароль с хранящимся в ней искомым значением таким образом, чтобы при этом пароль нельзя было скопировать

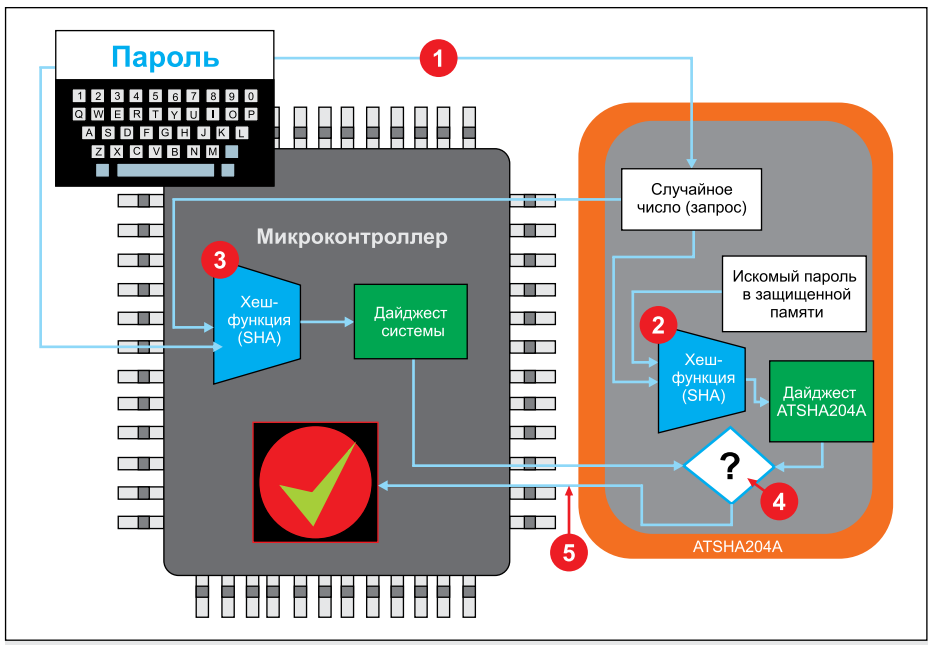

Рис. 9. Защита паролей

или прочитать. Поскольку микропрограмма стандартного микроконтроллера может быть взломана, в таких случаях рекомендуется как хранить пароль в защищенной аппаратной среде, так и проводить в ней сравнение вводимого значения пароля с эталонным (осуществлять аутентификацию). Для этого могут использоваться микросхемы семейства CryptoAuthentication. В примере на рис. 9 криптографическое устройство посылает системному микроконтроллеру запрос в виде случайного числа в тот момент времени, когда в систему извне начинает вводиться пароль (шаг 1). Это случайное число генерируется и затем хешируется в микросхеме ATSHA204A с корректным значением пароля, который хранится в ее защищенной энергонезависимой памяти (шаг 2), создавая первый дайджест. После окончания ввода пароля в систему микроконтроллер хеширует полученное значение (шаг 3) со случайным числом, заранее пересланным ему из криптографической микросхемы. Полученный второй дайджест пересылается из микроконтроллера в криптографическое устройство, которое сравнивает его с уже вычисленным первым дайджестом (шаг 4). При их совпадении введенный пароль считается подлинным, криптографическая микросхема сигнализирует об этом микроконтроллеру, разрешая тем самым работу системы (шаг 5).

Основные особенности:

- безопасное хранение системных паролей;
- недорогое, надежное и простое в реализации решение.

Использование диверсифицированных ключей

Хорошим способом для повышения информационной безопасности в системах с симметричным шифрованием и аутентификацией является применение так называемых диверсифицированных ключей. При таком подходе ключ, находящийся на стороне хоста, называется корневым. Клиенты системы не имеют к нему доступа. Вместо этого каждый клиент системы получает некоторую производную данного корневого ключа, которая базируется на уникальном числе, ассоциированном только с этим определенным клиентом. Например, это может быть 72-битный сериальный номер криптографической микросхемы. Уникальное число заранее хешируется при производстве конечного изделия совместно с корневым ключом и записывается в криптографическую микросхему клиента как диверсифицированный (производный) ключ. Этот ключ называется диверсифицированным именно потому, что он уникален, так как каждый клиент в системе имеет свой персональный, неповторяющийся, идентификационный номер. Для аутентификации клиента с диверсифицированным ключом хост должен уметь генерировать именно этот ключ таким же способом, каким

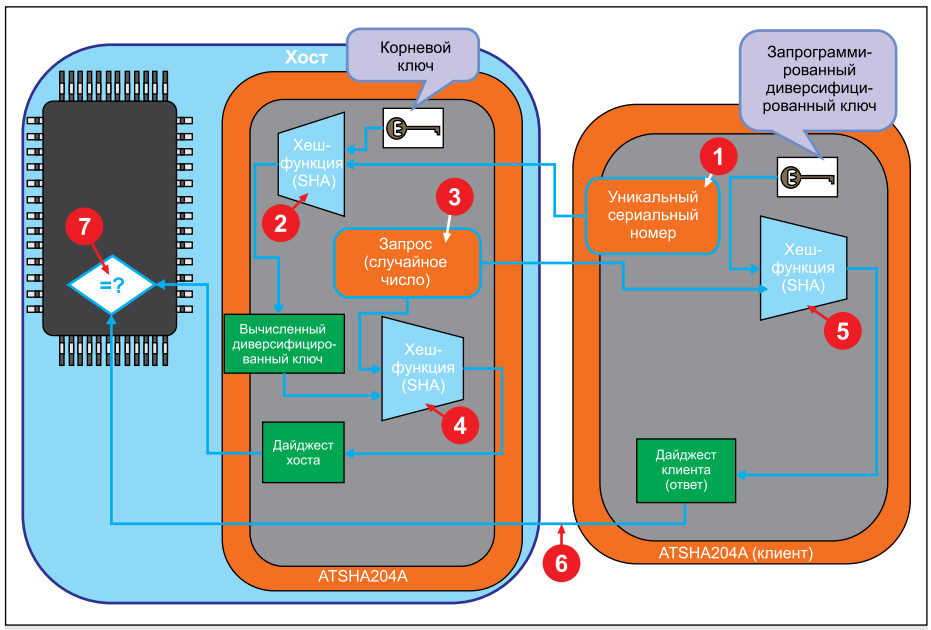

Рис. 10. Использование диверсифицированных ключей для повышения уровня информационной безопасности

он создавался при производстве конечного изделия. Следовательно, ему потребуется уникальный номер клиента.

В процедуре аутентификации, показанной в примере на рис. 10, используются две одинаковые криптографические микросхемы — по одной на стороне клиента и на стороне хоста. Сначала клиент по запросу хоста посылает ему уникальный сериальный номер своей микросхемы ATSHA204A (шаг 1). В микросхеме ATSHA204A на стороне хоста это число хешируется вместе с хранящимся там секретным корневым ключом. Полученный дайджест (шаг 2) будет представлять собой «кандидата» на диверсифицированный ключ этого клиента. Следующий этап — проверка того, действительно ли ключи одинаковы и, следовательно, что клиент является подлинным. Для этого выполняется дополнительная операция «запрос – ответ». Микросхема ATSHA204A на стороне хоста генерирует запрос в виде некоторого случайного числа (шаг 3) и отправляет его клиенту. Одновременно в ней выполняется хеширование этого случайного числа и вычисленного ранее диверсифицированного ключа. Полученный дайджест (ответ) пересылается в управляющий микроконтроллер (шаг 4).

В это же самое время микросхема ATSHA204A на стороне клиента выполняет похожие действия. Получив от хоста случайное число (запрос), она хеширует его вместе с хранящимся у нее в памяти заранее запрограммированным при производстве конечного изделия диверсифицированным ключом клиента (шаг 5). Вычисленный дайджест пересылается в управляющий микроконтроллер хоста (шаг 6), где сравнивается с полученным ранее ответом от своей микросхемы ATSHA204A. В случае совпадения клиент считается подлинным.

Основные особенности:

- все клиенты будут иметь различные ключи, поэтому потеря одного из них не влияет на информационную безопасность остальных клиентов в системе;
- простые в реализации операции, позволяющие существенно повысить уровень информационной безопасности системы;
- технология может быть использована для создания клиентских черных списков.

Вместо заключения

AES — очень хороший и широко применяемый во всем мире быстрый алгоритм симметричного шифрования, который был досконально изучен криптоаналитиками на предмет наличия слабых мест. Поэтому часто априори предполагается, что системы, в которых используется AES, безопасны по умолчанию. Но это предположение не всегда является верным.

Во-первых, подобно всем криптографическим системам и алгоритмам, криптостойкость AES зависит от ключа. Если атакующий может достать ключ, он способен притвориться аутентичной стороной, расшифровывать все сетевые сообщения и в общем случае разрушить безопасность системы. Ключом к безопасности является безопасность ключа. Проблема и состоит в том, что очень немногие системы имеют действительно безопасное место для хранения ключей, защищенное от атак. Одновременно с развитием распределенных систем и неуклонным усложнением ПО количество недоработок в программах тоже растет. Яркий пример тому — последствия недавно обнаруженной ошибки Heartbleed в криптографическом ПО OpenSSL, которая позволяла несанкционированно просматривать память на стороне

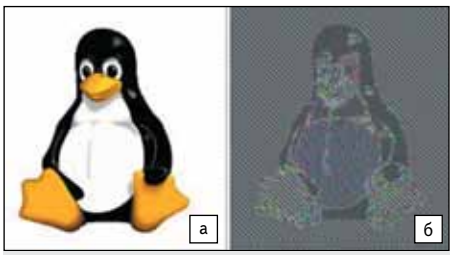

Рис. 11. Результат некорректного применения алгоритма AES:
а) оригинальное изображение;
б) изображение, зашифрованное в режиме ECB

Рис. 12. Генератор случайных чисел Lavarand

хоста и клиента, извлекая хранящиеся там ключи шифрования.

Во-вторых, как и все криптографические алгоритмы, AES может применяться множеством вариантов и способов. Здесь скрывается еще одно слабое место — система безопасности может быть взломана только из-за неправильного или непродуманного интерфейса, организации режима работы устройства, процедуры обмена данными и т. п. Криптография должна быть исполнена предельно корректно и профессионально, чтобы секреты действительно надежно сохранялись. Это очевидно в теории, но далеко не всегда легко осуществимо на практике. Приведенная на рис. 11 картинка показывает, что может произойти, если криптография не была реализована корректно.

В-третьих, когда что-то зашифровывается с помощью AES, большинство режимов требуют начальную последовательность (или инициализирующий вектор IV). Эта последовательность никогда не должна повторяться и во многих режимах обязательно должна быть случайным числом. Проблемы с повторяющимся IV следующие:

- если атакующий смог раскрыть исходный текст первого сообщения, он сможет раскрыть и второе;
- если одно и то же сообщение обрабатывается с одним и тем же IV, то зашифрованный текст будет абсолютно одинаковым, что является жизненно важной информацией для вычисления ключа шифрования злоумышленниками.

Получить действительно случайное число весьма трудно. Вселенная может двигаться вперед в направлении увеличения энтропии, но на этом пути она заполнена всевозможными шаблонами с кажущейся случайностью. Действительная случайность должна быть «схвачена» очень аккуратно. Один из знаменитых генераторов случайных чисел использовал дайджесты от изображения лавалампов (рис. 12), и многие годы сайт Lavarand поддерживался фирмой Silicon Graphics для предоставления случайных чисел в режиме онлайн.

Допустим, что у компании-разработчика нет генератора настоящих случайных чисел. Тогда возникает соблазн использовать «случайные» строковые комбинации, шум на сигнальных линиях, текущее время в миллисекундах или что-то подобное. Проблема состоит в том, что пока результирующие числа являются похожими на случайные, объем их выборки сильно ограничен. При высокой производительности современных компьютеров атакующий может перепробовать миллионы комбинаций за секунды и угадать «случайное» число.

Принимая во внимание эти и многие другие аспекты безопасности, многие разработчики в настоящее время полагаются на специализированные аппаратные криптографические средства, включая законченные устройства и защищенные интегральные микросхемы различного класса. В общем случае такие средства предлагают следующие решения:

- сильную защиту криптографических ключей, которая не зависит от недоделок в ПО, устойчива к вредоносному ПО (вирусам) и другим агрессивным атакам;
- надлежащее использование режимов и протоколов для криптографических операций;
- высококачественные генераторы случайных или псевдослучайных чисел, которые основываются на случайных физических событиях и которые досконально протестированы.

Аппаратно защищенные микросхемы семейства CryptoAuthentication компании Atmel относятся к таким специализированным криптографическим средствам. Они могут применяться в широком диапазоне конечных приложений. В случаях, когда в систему требуется добавить начальный, но при этом реально работающий уровень информационной безопасности, не базирующийся только на программных способах защиты, эти микросхемы могут оказаться хорошим, удобным и недорогим решением.

Литература

1. www.atmel.com