Микросхемы семейства CryptoAuthentication компании Atmel: вопросы и ответы

Игорь КРИВЧЕНКО
ik@efo.ru

Во время общения с разработчиками встраиваемых систем на темы аутентификации, аппаратной реализации криптографических алгоритмов и безопасного хранения данных применительно к микросхемам CryptoAuthentication компании Atmel обычно возникает целый ряд однотипных вопросов, обсуждаются возможные решения и особенности применения. В приведенной статье предпринята попытка сгруппировать и обобщить наиболее популярные темы обсуждения в формате «вопрос — ответ».

Вопрос № 1

Я уже использую AES-криптографию. Зачем мне что-то еще? Ведь я зашифровываю данные в программе, разве это не вся информационная безопасность, которая требуется?

Симметричный алгоритм AES осуществляет шифрование путем выполнения логических операций перестановки и инвертирования групп символов исходных данных. Зашифрованная таким образом информация может затем храниться или передаваться, оставаясь при этом непригодной для использования до момента ее расшифровки и перевода в оригинальную форму.

Алгоритму AES для работы требуется секретный ключ, который должен быть одним и тем же для зашифровки и расшифровки информации. Этот секретный ключ необходимо надежным образом сохранять от раскрытия как на передающей, так и на принимающей стороне. Если ключ не держится в секрете на любой из сторон, тогда зашифрованная информация может быть раскрыта кем-то еще, что полностью разрушает всю идею шифрования. Безопасное, защищенное хранение любого ключа шифрования (в том числе и для алгоритма AES) является одной из самых критичных и уязвимых составляющих информационной безопасности. Область памяти, где хранится ключ, должна быть способна противостоять различного вида атакам, задачей которых является прочтение секретных данных.

Защита информации на физическом уровне обычно гораздо сильнее защиты на уровне программного обеспечения (ПО), потому что на кристалл микросхемы встраиваются сразу несколько взаимосвязанных и совместно работающих аппаратных блоков, которые обеспечивают надежную защиту всей микросхемы от атак различных типов. Поэтому для безопасного хранения информации все чаще используют специализированные, аппаратно защищенные криптографические устройства разного уровня сложности. Например, микросхемы семейства CryptoAuthentication компании Atmel предлагают метод защиты секретных ключей, который ограничивает несанкционированный доступ к памяти извне, а также обеспечивает генерацию ключей и управление ими непосредственно на кристалле.

Вопрос № 2

Говорят, что алгоритмы информационной безопасности ранжируются по надежности, например, в таком порядке: SHA, AES, RSA, ECC. А какой уровень надежности у предлагаемых микросхем Atmel?

По существу, безопасность является функцией двух важных критических факторов:

1) длины ключа, используемого для работы криптографического алгоритма;
2) набора математических операций, которые применяются этим криптографическим алгоритмом.

Следовательно, эффективность информационной безопасности зависит и от размера ключа, и от применяемых специальных вычислительных процедур. Любая из этих вычислительных процедур может оказаться сильнее или слабее, чем другая.

Среди большинства криптографических экспертов следующие размеры ключей считаются приблизительно одинаковыми по надежности: SHA-256, AES-128, RSA-3072 и ECC-256. Буквы в названиях отображают название алгоритма (например, SHA), а цифры — длину ключа (скажем, 256). Эти общепринятые стандартные алгоритмы шифрования используются в конечных приложениях самыми разными способами, в зависимости от задачи. Более того, многие криптографические системы применяют сразу несколько алгоритмов, чтобы объединить локальные преимущества каждого из них для достижения как можно более высокого уровня информационной безопасности и для повышения эффективности (например, скорости выполнения).

Вопрос № 3

Симметричные алгоритмы, такие как AES и SHA, используют один и тот же ключ для хоста конечной системы и всех клиентов в этой системе. Но если ключ в каком-то месте данной системы был однажды взломан, то он уже не будет являться секретным. Должен ли я по-прежнему использовать симметричные алгоритмы?

Сущность криптографии состоит в том, что секретная информация должна храниться в секрете (дословно). Безопасное хранение ключей — то, что поддерживает целостность системы вне зависимости от того, какие алгоритмы шифрования в ней используются. Тем не менее затронутая проблема действительно существует. Но ее негативное влияние можно значительно снизить.

На самом деле, для систем, использующих симметричные алгоритмы, нет необходимости везде хранить абсолютно одинаковый ключ, даже если это может показаться алогич-

ным или интуитивно непонятным. Используя технологию, называемую «диверсификация ключей» (искусственное внесение разнообразия для обеспечения отказоустойчивости системы), каждый клиент в системе может хранить некий уникальный ключ, который является производной своего серийного номера и секретного ключа хоста. Во время процедуры аутентификации с диверсифицированными ключами клиент посылает хосту свой уникальный серийный номер. Хост хеширует это число с хранящимся у него секретным корневым ключом, формируя предполагаемый диверсифицированный ключ клиента. Далее на базе некоторого случайного числа и диверсифицированного ключа и хост, и клиент вычисляют «цифровые подписи» — дайджесты, которые сравниваются между собой. Если вычисленный на стороне хоста и хранимый на стороне клиента диверсифицированные ключи совпадают, то дайджесты получатся одинаковыми. В этом случае хост принимает решение, что данный клиент — реальный (то есть аутентифицированный).

Безотносительно к типу применяемого алгоритма, виду криптографического протокола или к архитектуре всей системы лучшим решением является недопущение взлома на самом первом шаге — хранении ключей. В настоящее время эта задача решается с помощью различных защищенных аппаратных устройств, включая специализированные интегральные микросхемы.

Вопрос № 4

Каким образом можно обеспечить шифрование с помощью «одностороннего» алгоритма типа SHA и не является ли базовая для криптографии логическая операция XOR слабым местом?

Многие алгоритмы шифрования, подобно DES и AES, применяют базовую операцию XOR, с помощью которой данные объединяются с сеансовым ключом. Конфиденциальность зашифрованных данных зависит от того, насколько хорошо сам криптографический алгоритм математически скремблирует эти данные, а не от того, насколько сильной является сама операция XOR. То есть насколько выбранный алгоритм шифрования не имеет с точки зрения криптографии слабых мест.

Микросхемы Atmel (например, ATSHA204A) используют хеш-алгоритм SHA-256 при защищенном обмене данными между криптографическим устройством и микроконтроллером. По запросу хоста на выполнение шифрованной операции чтения криптографическое устройство генерирует некоторое случайное число, хешируемое непосредственно на кристалле с внутренним секретом. Полученный дайджест используется в качестве ключа при выполнении операции XOR над исходными данными для последующей передачи. (Другими словами, для зашифровывания исходного открытого текста он прогоняется через операцию XOR с уникальным ключом, полученным как дайджест секретного ключа и случайного числа.) Когда эти зашифрованные данные попадают из криптографической микросхемы в приемник, ему требуется знать тот же самый секрет и текущее случайное число, чтобы воспроизвести исходные сведения при помощи операции XOR. Аналогично, при запросе на шифрованную операцию записи криптографическое устройство математически реконструирует принятые данные, используя операцию XOR и вычисляемый ключ шифрования. Естественно, что секретные ключи на обеих сторонах должны быть одинаковыми.

Поскольку для каждой шифрованной передачи данных создается совершенно новое случайное число и поскольку дайджест этого случайного числа и секретного ключа используется как ключ XOR, то попытка взлома такой зашифрованной передачи равнозначна попытке взлома самого алгоритма SHA-256 или попытке подобрать секретный ключ методом прямого перебора.

Вопрос № 5

В чем состоит различие между MAC и HMAC?

MAC (Message Authentication Checksum) — это, по сути, криптографическая контрольная сумма сообщения, которая вычисляется совместно с некоторым секретным ключом путем шифрования, хеширования или выполнения специальных алгоритмов аутентификации. MAC предоставляет возможность обнаружить изменения в исходных данных (проверка их целостности) и обычно применяется именно для этой цели, гораздо реже — для идентификации сообщений или пользователей. HMAC означает алгоритм аутентификации сообщения MAC на основе вычисления хеш-функции, отсюда и добавление «H» перед «MAC». Этот алгоритм не единственный среди алгоритмов проверки целостности данных на основе хеш-функций, но он является одним из наиболее распространенных.

Большинство алгоритмов HMAC работает следующим образом. Две стороны совместно владеют секретным ключом и хотят обменяться некоторым сообщением. Сначала на стороне отправки вычисляется дайджест секретного ключа и сообщения. Затем при пересылке сообщения к нему прибавляется этот дайджест. Принимающая сторона тоже вычисляет дайджест полученного сообщения вместе с тем же самым секретным ключом. Затем принятый и вычисленный дайджесты сравниваются. Если сообщение было изменено, то дайджесты не совпадут. Что именно вызвало несовпадение — неизвестно и не суть важно: что-то пошло не так, процесс передачи нужно начинать заново.

Дайджест зависит и от содержания исходного сообщения, и от ключа, поэтому злоумышленник, хакер или криптоаналитик (атакующий) должен знать секретный ключ, чтобы изменить сообщение, а затем вычислить и присоединить к нему правильную контрольную сумму. В этом и состоит проверка целостности данных. Если есть опасение, что информация может быть изменена, данные пересылаются вместе с контрольным значением HMAC. Если контрольное сообщение показывает отсутствие изменений, данные считаются подлинными.

Отметим, что HMAC в качестве цифровой подписи (то есть для целей идентификации) имеет слабые места. Поэтому на практике контрольные значения HMAC обычно используются для контроля неизменности сообщения при передаче. Для создания уникальной цифровой подписи нужен другой способ — например, шифрование дайджеста с помощью закрытого ключа автора исходного сообщения. В отличие от алгоритмов электронной цифровой подписи, в которых для создания подписи используется закрытый ключ, а для ее проверки — соответствующий ему открытый ключ, при вычислении и проверке HMAC применяется один и тот же секретный ключ. Следовательно, HMAC не гарантирует уникальность авторства сообщения — секретный ключ принадлежит не одному, а как минимум двум пользователям.

Вопрос № 6

Мне говорили, что система не может считаться безопасной, если в ней не используется защищенный криптографический микроконтроллер. Это правда?

Специализированные защищенные криптографические микроконтроллеры (ЗКМК) с открытой архитектурой являются отличным решением в случае, когда вся конечная система, включая приложение и операционную систему, может быть реализована внутри предоставленных ресурсов этого ЗКМК и когда итоговая цена вопроса приемлема для данной системы.

Для многих современных приложений в качестве основы требуются высокопроизводительные микропроцессоры с внешней расширяемостью или периферией специального назначения. Ресурсы ЗКМК обычно невелики, так как целевая функция у них совершенно другая. Поэтому они используются в качестве вспомогательного устройства или сопроцессора для мощного системного ядра, увеличивая тем самым проблемы безопасности системы: теперь нужно объединять работу ЗКМК с плохо защищенными от атак операциями основного процессора и защищать шину информационного обмена между ними.

Тем не менее добавление в проект любой криптографической микросхемы (например, из семейства CryptoAuthentication или профессионального ЗКМК с открытой архитек-

турой) считается хорошим решением. Это как минимум гарантирует безопасное хранение конфиденциальных данных и ключей шифрования. Функционирование криптографических блоков в любой из специализированных микросхем для шифрования и/или аутентификации тщательно проверяется и верифицируется каждым производителем подобных устройств, который должен иметь соответствующие международные сертификаты безопасности. Для разработчика конечной системы таким образом снижается вероятность появления в его прикладном ПО слабых мест с точки зрения информационной безопасности. Это полезно на этапах разработки и тестирования, но особенно актуальным может оказаться в случаях, когда микропрограмму приходится обновлять или заменять «в поле», в том числе и удаленно.

Вопрос № 7

Как защитить коммуникационную шину между системным микроконтроллером и внешней криптографической микросхемой?

Этот вопрос обычно возникает, когда требуется периодически проверять, не атакуется ли канал связи между криптографическим устройством, отвечающим за аутентификацию, и управляющим микроконтроллером.

В случаях, когда в качестве такого криптографического устройства используется микросхема CryptoAuthentication, сигнал успешно проведенной проверки может быть выдан в виде обычного логического уровня (импульса), подаваемого на внешний вывод микроконтроллера. Но если, например, присоединить кнопку к линии связи между двумя устройствами, то можно попытаться имитировать сигнал успеха и ввести систему в заблуждение. Для противодействия подобным хакерским манипуляциям микросхемы этого семейства содержат встроенный механизм проверки истинности логического сигнала успеха, над которым выполняется дополнительный цикл верификации. Для этого нужен второй секретный ключ, хранящийся в памяти криптографической микросхемы и вычисляемый в программе микроконтроллера.

После получения извне сигнала успешной аутентификации микроконтроллер генерирует некоторое случайное число. Оно передается в криптографическую микросхему и хешируется там совместно со вторым секретным ключом. Полученный дайджест возвращается в микроконтроллер, который за это время выполняет такую же последовательность действий над случайным числом и вычисляемым секретом. Дайджесты сравниваются, и при их совпадении логический сигнал успеха от аутентифицируемого устройства считается подлинным. Такой подход существенно усложняет задачу атакующему, пытающемуся заменить проверяемое устройство неаутентичным.

Микросхемы CryptoAuthentication также могут быть использованы для организации процесса защищенной начальной загрузки микроконтроллера. В этом режиме они подтверждают аутентичность (подлинность) загружаемого кода.

Вопрос № 8

Почему считается, что только специализированные защищенные микросхемы могут успешно противостоять попыткам взлома? Современные микроконтроллеры имеют однократно программируемую блокировку доступа к области памяти на кристалле. «Залочив» такую микросхему, добыть что-то из нее невозможно. Или это не так?

Методы, применяемые для раскрытия хранимой и обрабатываемой в микроконтроллерах информации (атаки), делятся на две основные группы: неагрессивные (атаки по сторонним каналам; атаки, основанные на инициировании ошибок) и агрессивные.

Атаки по сторонним каналам основаны на корреляции между значениями физических параметров, измеряемых в разные моменты работы устройства, и его внутренним логическим состоянием. Например, информация о времени выполнения определенных операций, потреблении тока, электромагнитном излучении и т. п. может быть использована для вычисления секретного ключа.

Успех такой атаки зависит, в частности, от того, содержит ли конкретная программная или аппаратная реализация атакуемого криптографического алгоритма слабые места с точки зрения информационной безопасности.

Атаки, основанные на инициировании ошибок, изменяют нормальное поведение микроконтроллера с помощью внешнего воздействия. Информация, получаемая во время сбоев, может быть полезна для дальнейшего криптоанализа. Так, можно изменять напряжение питания и/или тактовую частоту; точечно облучать микросхему лазером, ультрафиолетовым или другим излучением; применять локальный нагрев определенной области кристалла.

Существенно усложнить успешное проведение подобных атак можно разными способами. Например, добавить на кристалл детекторы напряжения питания, частоты и освещенности для останова блока шифрования при обнаружении воздействия. Или снабдить микросхему экраном, при попытке снятия которого она выходила бы из строя. Можно реализовать аппаратное дублирование внутренних вычислений со сравнением результатов. И так далее.

К агрессивным атакам относятся активные, разрушающие воздействия на открытый кристалл интегральной схемы. Данные атаки требуют наличия дорогостоящего оборудования, поэтому чаще всего их применяют для изучения и копирования внутренней структуры атакуемого устройства.

Методы борьбы с агрессивными атаками: гибкое и настраиваемое шифрование пользовательских данных; уменьшение размеров кристалла, скремблирование внутренних шин и шифрование передаваемых по ним данных; нестандартное расположение компонентов интегральной схемы и ряд других.

Все ли стандартные микроконтроллеры общего назначения имеют на кристалле хоть какое-либо подмножество средств защиты от атак или хотя бы сертифицированные аппаратные блоки шифрования? Даже в последнем случае можно ли быть полностью уверенным в том, что разработчик встраиваемого ПО корректно реализует программный интерфейс взаимодействия с ними? А ведь специализированное ПО и аппаратные пробники для проведения неагрессивных атак свободно продаются. Они весьма недешевы, но доступны и частным лицам, и фирмам-производителям микросхем. Поэтому здесь все определяется конечной выгодой, которую получит злоумышленник при раскрытии чужого секрета. Нельзя забывать и о том, что после успешного взлома какого-либо микроконтроллера все остальные кристаллы такого же наименования будут легко вскрываться по найденному проверенному сценарию. В Азиатском регионе, например, подобный сервис реверсивного инжиниринга поставлен на коммерческую основу: в списке устройств, «готовых к вскрытию под заказ», находится множество стандартных микроконтроллеров Microchip, STM, NXP, Freescale, TI, Renesas, Atmel и других.

Все технологические дополнения по защите кристалла и последующие процедуры обязательной сертификации требуют от фирм-производителей и времени, и значительных инвестиций. Поэтому и выделяются в отдельный класс специализированные микросхемы, ориентированные только на квалифицированное выполнение определенных криптографических операций, а также на безопасное хранение информации.

Так что ответ на поставленный вопрос будет следующим: скорее нет, чем да. Взломать в конечном счете можно все, если заказчик готов заплатить за это определенные деньги и подождать некоторое время. Очевидно, что вскрыть специализированные защищенные микросхемы гораздо сложнее, дольше и дороже, чем стандартные.

Вопрос № 9

Как мне следует защитить свою интеллектуальную собственность, которая сохранена в программном обеспечении?

Единственным способом для полной защиты ПО конечной системы является применение специализированных защищенных микроконтроллеров (например, таких, которые широко используются в смарт-картах и сер-

тифицированы по Common Criteria/FIPS). В этом случае системный код и секретные данные хранятся в их внутренней защищенной памяти и, следовательно, не могут быть скопированы, прочитаны или изменены атакующей стороной. Но из-за высокой стоимости такого решения и других трудностей (необходимость наличия специального оборудования для разработки и персонализации, ограниченный доступ к информации, защищенные помещения, уровни доступа к средствам разработки, лицензирование и разрешительная деятельность) это не может быть приемлемым решением для подавляющего большинства распределенных систем удаленного контроля и управления.

Промежуточный вариант — использовать «более простую» специализированную криптографическую микросхему для аутентификации и безопасного хранения конфиденциальной информации. Это обеспечит достаточный уровень аппаратной защиты в совокупности с простотой разработки и очень низкой добавленной в решение стоимостью. Такой подход может использоваться для того, чтобы воспрепятствовать несанкционированному копированию и последующему клонированию ПО системы, а также для проверки подлинности ПО. Аутентификация микропрограммы может осуществляться путем регулярной посылки запросов криптографической микросхеме во время работы системы и последующей проверкой корректных ответов на них. В случае любого несовпадения или отказа встроенное ПО должно прекратить работу.

Если в вышеприведенной схеме микропрограмма регулярно обновляется и при этом новые пары «запрос — ответ» сохраняются в хост-системе, то для хакера становится очень трудной задача анализа кода и удаления из него аутентификационных закладок. Если доступно удаленное соединение, по которому некоторые из запросов случайным образом посылаются от доверенного сервера, то такая система может считаться хорошо защищенной.

Многие современные микропроцессоры и микроконтроллеры имеют на кристалле в памяти ROM область загрузчика. Эта память может быть запрограммирована на использование внешней криптографической микросхемы для проверки цифровой подписи рабочей программы перед ее выполнением. Если были осуществлены какие-либо несанкционированные модификации, это мгновенно обнаружится и все системные операции могут быть запрещены. Когда разработчики ищут способ защитить чувствительный алгоритм, данные или протокол обмена, подобная методология защищенной системной загрузки может применяться и для расшифровки программного модуля, хранимого в зашифрованном виде во внешней памяти. Но при этом сам ключ шифрования должен находиться в криптографической микросхеме, потому что в системном процессоре или его внешней памяти обычно нет специальных аппаратно защищенных областей на кристалле для хранения секретной информации.

Вопрос № 10

Все труднее и труднее доверять чему-либо в нашем стремительно изменяющемся цифровом мире. Как мы можем проектировать распределенные системы контроля и управления, которые предполагают сохранение доверия к себе в течение продолжительного времени?

Очевидно, что уровень доверия — это одна из наиболее важных составляющих любого проекта, потому что если передаваемым данным нельзя доверять, то и цена им ноль. В хорошо спроектированной системе должны присутствовать все три кита безопасности: конфиденциальность, целостность и аутентичность данных. При этом основополагающим является требование обеспечить гарантированное, безопасное сохранение конфиденциальной информации в секрете. Хранение цифровых ID, паролей, ключей шифрования, имен и других важных личных данных только в программном обеспечении на сегодняшний день уже не считается совершенно надежным решением, потому что любое ПО имеет недоработки и, следовательно, может быть вскрыто. Хранение секретных данных на защищенных аппаратных устройствах, реализующих гораздо более сильный уровень защиты от атак, является лучшим решением, если цена вопроса при реализации такого проекта может считаться оправданной.

Вопрос № 11

В приложениях, где требуется организовать асимметричную аутентификацию, необходимы сертификаты. Могут ли микросхемы CryptoAuthentication работать со стандартными сертификатами инфраструктуры открытых ключей?

Для задач формирования/проверки цифровой подписи и распределения открытых ключей компания Atmel выпускает микросхемыATECC508A и ATECC108A. Они поддерживают алгоритм ECDSA на эллиптических кривых, работают с ключами длиной 256 бит и ориентированы на работу с сертификатами в формате наиболее распространенного в мире стандарта X.509. Микросхемы могут хранить в своей защищенной энергонезависимой памяти до двух сертификатов. Большего уровня вложенности Atmel не предусматривает, считая двухуровневую схему достаточной для широкого круга прикладных задач.

Поскольку размер сертификата стандарта X.509 может быть довольно большим, в ряде случаев появляются некоторые ограничения на структуру и размер хранимых в микросхеме CryptoAuthentication сертификатов. В определенной области памяти тогда сохраняется собственно криптографический компонент сертификата — цифровая подпись ECDSA, обычно вместе с открытым ключом. Остальные данные сертификата (даты начала и окончания действия, алгоритм, имя, ограничения по использованию и т. п.) могут размещаться либо в других областях защищенной памяти этой же микросхемы CryptoAuthentication, либо во внешних устройствах хранения информации. Процесс формирования полного сертификата стандарта X.509 должен выполнять внешний по отношению к ATECC508A / 108A микроконтроллер в соответствии с синтаксисом ASN.1.

Вопрос № 12

Распределенные системы удаленного контроля, управления, доступа и т. п., включая решения для IoT, предполагают постоянное обновление микропрограммы и данных, в том числе удаленно. Каким образом можно защитить процесс обновления прошивки?

Наверное, существует много вариантов решения. Например, не позволять обновляемому коду загружаться в клонированные устройства. Проверять перед загрузкой, не была ли модифицирована прошивка. Регулярно верифицировать код, хранящийся в энергонезависимой памяти системы. Применять только защищенную загрузку. И так далее.

Рекомендуется использовать цифровые подписи для проверки корректности образов кода или фрагментов кода. Не стоит «играть» с криптографическими алгоритмами и с длиной ключа. Следует помнить о постоянном обновлении ключей или увеличении их количества в будущем.

Формируйте многоуровневый процесс обновления встроенного ПО так, чтобы неизменяемые куски кода были небольшими, с высокой плотностью кода и всесторонне протестированными. Реализуйте второй уровень загрузки, который может быть верифицирован основным уровнем. Создавайте методику, которая позволит обеспечить гибкость на стадии загрузки. Поддерживайте как широковещательные образы, так и (возможно фрагментированные) целевые образы загружаемой микропрограммы.

Вопрос № 13

Оправдано ли неизбежное увеличение стоимости моих изделий при добавлении в них того или иного уровня информационной безопасности?

Добавление безопасности в систему в чем-то похоже на обычную добровольную страховку в повседневной жизни: КАСКО, ДМС, страхование квартиры от кражи, дачи от по-

жара и т. п. «Стоимость» результатов взлома информации можно оценить по аналогии с потерями в случае аварии, кражи, пожара, затопления, землетрясения, угона, внезапной болезни и т. д. Небольшая цена за добавленное в систему специализированное криптографическое устройство (микросхему) может предотвратить ущерб от возможных непредвиденных, иногда катастрофических, потерь.

Вопрос № 14

Требуется ли лицензирование при покупке микросхем семейства CryptoAuthentication компании Atmel?

Криптографические алгоритмы, аппаратно реализованные в микросхемах Atmel семейства CryptoAuthentication, являются стандартными, одобренными и проверенными во всем мире процедурами, которые не требуют для своего использования приобретения лицензии или лицензионных отчислений ни компании Atmel, ни какой-либо другой компании. В дополнение Atmel предоставляет свободно распространяемые примеры реализации уровней коммуникационных протоколов в исходных кодах.

Вопрос № 15

Почему криптографические микросхемы Atmel могли бы быть предпочтительными для применения?

Сейчас выпускается уже третье поколение криптографических продуктов Atmel, которые заключают в себе годы опыта и многочисленные ноу-хау по созданию и серийному производству защищенных от атак криптографических устройств безопасного хранения и обработки информации. Atmel использует собственные сертифицированные методы тестирования при производстве кристаллов, не допуская возможности несанкционированного доступа на этапах тестирования.

Микросхемы Atmel используют апробированные и надежные алгоритмы SHA-256 и ECC-P256. Для сравнения: решения, построенные на базе CRC, не могут успешно противостоять современным приемам криптоанализа, а хеш-алгоритм SHA-1 не рекомендован для новых разработок. Микросхемы Atmel содержат на кристалле многоуровневый генератор псевдослучайных чисел, сертифицированный в соответствии с требованиями FIPS. Это позволяет добиваться высокого качества генерируемых случайных чисел.

Atmel постоянно уделяет огромное внимание анализу, тестированию и совершенствованию своих микросхем на предмет наличия слабых мест. На кристаллы добавляются специальные аппаратные узлы для противодействия неагрессивным и агрессивным атакам; JTAG не используется; не существует отладочных, тестовых или других точек отладки и контроля и т. д.

Литература

1. www.atmel.com