# Требования к легковесным алгоритмам и их реализации

Легковесная криптография изучает криптографические алгоритмы, ориентированные на использование в небольших устройствах, заведомо не обладающих значительными ресурсами. Специфика устройств определяет ограничения (требования), которым должны соответствовать легковесные алгоритмы и их программная реализация.

* Бизнес-требования:
  + Требуемый уровень безопасности устройства.
  + Стоимость устройства.
  + Итоговая производительность устройства.
* Ограничения аппаратной части:
  + Ограничения энергетических ресурсов.
  + Ограничения по объему ОЗУ.
  + Ограничения размера микросхемы (GE-мера).
* Ограничения программной реализации:
  + Объем программного кода.
  + Количество потребляемой оперативной памяти.
  + Время работы.
* Возможные дополнительные ограничения:
  + Ширина полосы рабочих частот и пропускная способность канала связи.

## Бизнес-требования

Бизнес-требования определяют необходимость поиска компромисса между стоимостью, безопасностью и производительностью устройства. Достаточно легко оптимизировать два из трех свойств, а вот оптимизировать все три – обычно весьма трудная задача для разработчиков.

Процедура определения требуемого уровня безопасности определяется в соответствии с конкретной областью применения конкретной системы. В коммерческих структурах может использоваться, например, модель системы безопасности *с полным перекрытием*. В рамках данной модели перечисляются все защищаемые объекты системы и возможные атаки на них, а также вероятность появления и наносимый ущерб для каждой из атак. После чего определяются приоритетные направления защиты. В контексте систем интернета вещей представляется разумным ввести атаки, общие для всех уровней системы (например, DoS-атаки) и атаки, специфичные для конкретного слоя (например, кодовые инъекции нацелены главным образом на прикладной уровень). Атаки на каждый уровень и меры защиты от них кратко приведены в первой главе данной работы.

Стоимость и итоговая производительность устройства определяется, главным образом, используется ли в этом качестве устройство «общего назначения» (например, смартфон) или же некоторое «кастомное», уникальное аппаратное решение.

В первом случае и процессор, и системы сетевого взаимодействия, и иногда даже системы безопасности уже реализованы, и требуется реализовать только модули для сбора данных и для их шифрования. Обратной стороной являются возможные скрытые уязвимости или сознательно добавленные «бэкдоры» устройства, которые крайне трудно обнаружить. Кроме этого, использование функционала «из коробки» означает необходимость положиться на разработчиков этого функционала и потерю полного контроля над аппаратным обеспечением.

Во втором случае разработка начинается «с нуля» (или почти с нуля). С одной стороны, это требует гораздо больших навыков от команды разработчиков и больших затрат времени и денежных средств на разработку. Однако, с другой стороны, это может позволить создать в итоге более дешевое и надежное устройство – опять же в зависимости от уровня компетенций команды.

## Требования к аппаратной части

В случае использования «кастомной» аппаратной реализации, аппаратная часть должна удовлетворять требованиям по количеству энергетических ресурсов, объему ОЗУ и площади микросхемы. При использовании аппаратного решения «общего назначения» количество имеющихся ресурсов обычно значительно больше, чем требуется для работы криптографических алгоритмов (но, естественно, тоже ограниченно, что необходимо учитывать).

Ограничения энергетических ресурсов могут явно задаваться конкретным технологическим решением (например, RFID-метки рассчитаны не более чем на 15 микроватт[[1]](#footnote-1)). Кроме того, для предупреждения *атак по энергетическим ресурсам* требуется по возможности уменьшить скачки в энергопотреблении устройства. Это отдельная тема, выходящая за рамки данной работы.

Ограничения размера (площади) микросхемы связаны с тем, что зачастую в слабых устройствах (в том числе в устройствах интернета вещей) используются «кастомные» аппаратные решения, направленные на удешевление и повышение надежности устройств. Это означает, что требуется, насколько возможно, уменьшить количество используемых функциональных элементов. Оно определяется, во-первых, используемым алгоритмом, а во-вторых, его программной и аппаратной реализацией. В качестве единицы измерения данного ресурса используется количество элементов NAND (Not And), необходимое для реализации. Обозначается эта величина GE (logic Gate Elements number, количество логических вентилей). Требуемый размер микросхемы для легковесного алгоритма для устройств интернета вещей не должен превышать 1000 GE.

При создании конкретной аппаратной реализации необходимо искать баланс между скоростью работы и размером микросхемы. Оптимизация размера микросхемы достигается за счет использования последовательной архитектуры (обрабатывающей информацию побайтно), оптимизация скорости – за счет распараллеливания и конвейеризации, что влечет увеличение размера.

Основной стандарт, определяющий количество доступных ресурсов для легковесных криптографических алгоритмов, это ISO 29192[[2]](#footnote-2).

## Требования к программной части

При разработке программной реализации алгоритма необходимо оптимизировать те же направления, что при разработке любого ПО: оптимизация потребления памяти и времени, а также специфичный для данной области параметр – объем кода.

Количество потребляемой памяти особенно критично для «кастомных» реализаций, при использовании более мощной аппаратной части уже не столь критично, хотя все равно подлежит оптимизации.

Время работы состоит из двух величин. Задержка (англ. latency) – время инициализации алгоритма, и пропускная способность (англ. throughput) – количество информации, обрабатываемой в единицу времени. Обычно требования по каждому из этих параметров формулируются отдельно. Например, для систем автоматического осуществления дорожных сборов время реакции устройства должно быть менее 10 миллисекунд [Жуков, стр 29], это ограничение на задержку. С другой стороны, если система должна еще и зафиксировать изображение машины, нарушившей ПДД, то требуется передать достаточно большое количество информации, а это требует относительно большой пропускной способности.

Объем программного кода тесно связан с количеством используемого кода. Так, S-блоки, определяющие порядок перестановки блочного шифра, можно хранить в оперативной памяти, а можно напрямую «зашить» в код. Второй способ быстрее, однако компиляция и исполнение кода тоже требует определенных накладных расходов (при исполнении код также хранится в ОЗУ). Поэтому конкретное соотношение хранимых в ОЗУ и заданных в коде данных определяется под конкретную программу.

# Возможность использования различных типов криптографических алгоритмов в качестве легковесных

1. ISO/IEC 18000-3:2004 Information technology – Radio frequency identification for item management – Part 3: Parameters for air interface communications at 13.56 MHz [↑](#footnote-ref-1)
2. ISO/ IEC FDIS 29192 – Information technology – Security techniques – Lightweight cryptography. [↑](#footnote-ref-2)