Алгоритм шифрования NASH

ЗАЩИТА ИНФОРМАЦИИ

Алгоритм NASH

Алгоритм NASH – алгоритм блочного шифрования, предназначенный для реализации на широко распространенных микроконтроллерах общего назначения.

Алгоритм NASH

Алгоритм NASH основан на принципах современной «легковесной криптографии», но используют управляемые сдвиги, что позволяет при сохранении уровня стойкости ограничиваться меньшим числом раундов, повышая скорость обработки данных.

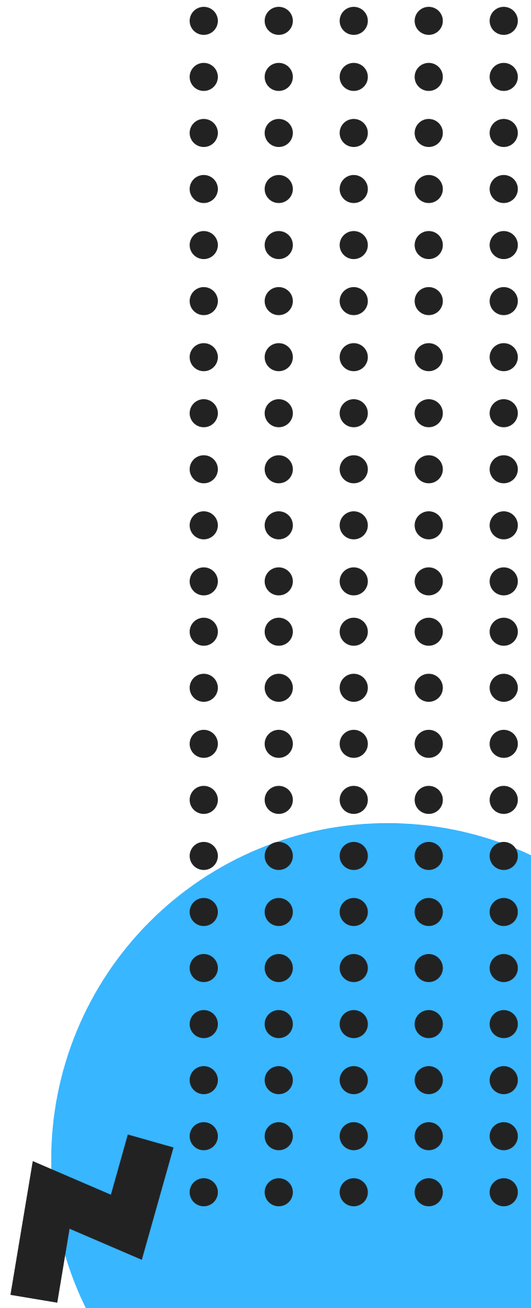
Алгоритм NASH

Алгоритм NASH может быть использован для защиты обмена данными между устройствами в так называемых сетях «интернета вещей», а также для защиты данных, записываемых на компактные персональные носители (флэш-память, карты MicroSD и т. д.).

Актуальность

Реализация на микроконтроллерах стандартов шифрования не может обеспечить приемлемой скорости шифрования.

Поэтому для достижения высокой скорости было предложено несколько специальных алгоритмов шифрования, получивших название «легковесных».



Актуальность

Наиболее эффективными среди них следует признать алгоритмы SPECK и SIMON, разработанные АНБ.

Мы поставили своей целью реализовать легковесный алгоритм блочного шифрования, который не уступал бы по стойкости упомянутым алгоритмам АНБ, но позволял бы несколько сократить количество раундов, что делает его еще более быстрым.

Схема алгоритма

Блок шифруется r раундов на последовательности раундовых ключей k(i), получаемых из главного ключа по алгоритму «расширения ключа»

Блок данных разбивается на левый и правый полублоки (L(i), R(i)) по 2^n бит каждый, с которыми на (i+1)-м раунде производятся следующие преобразования:

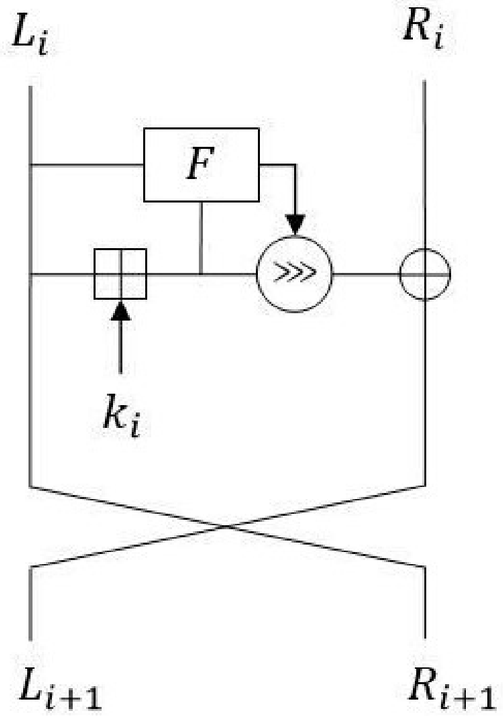
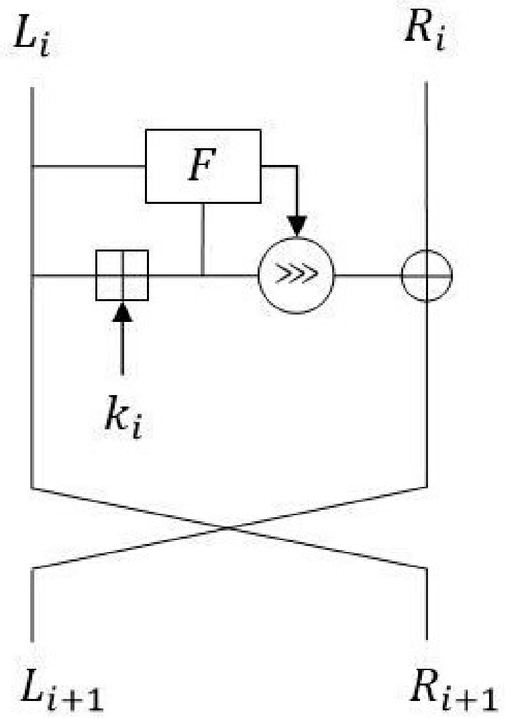


Схема алгоритма

Уравнения шифрования блока данных на (i+1)-м раунде выглядят так:

$$R(i+1) = L(i)$$
$$L(i+1) = ((L(i) \boxplus k(i)) \ggg F(L(i), L(i) \boxplus k(i))) \oplus R(i)$$

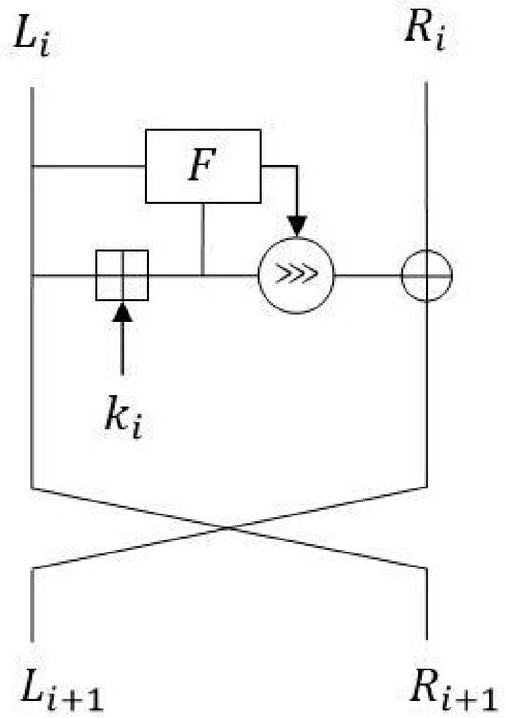
В последнем раунде шифрования блока не меняются местами полублоки L(i+1), R(i+1)



Детали раундового преобразования

Размер полублока равен 2^n, где n=5 или 6, соответственно размер полублока равен 32 или 64 бита. Соответственно предлагается <u>размер блока 64 или 128 бит</u>.

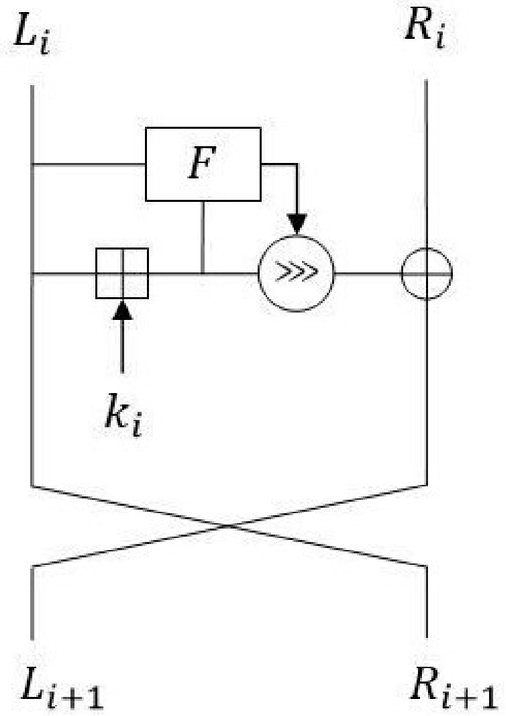
<u>Смешивание с раундовым ключом</u> k(i):
⊞ – функция сложение двух целых чисел по модулю 2^(2^n).



Детали раундового преобразования

Управляемый циклический сдвиг:

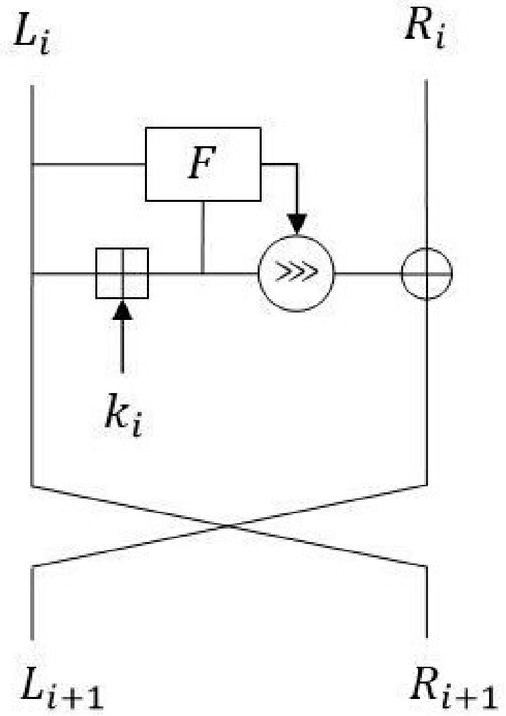
- **для размера блока 64 бита (полублока – 32 бита) — циклический сдвиг вправо на одно из 4 значений (11, 14, 10, 19).**
- **для размера блока 128 бит (размер полублока – 64 бита) — циклический сдвиг вправо на одно из 4 значений (37, 34, 38, 29).**



Функция управления сдвигами

Интерпретируем полублок L(i) как вектор значений булевой функции от n переменных.
Первый выходной бит F получаем как значение данной функции на наборе битов из L(i) ⊞ k(i) вида 2^i−1, где i=1, …, n,
то есть как значение L(i) ((L(i) ⊞ k(i)) [2^1−1,…,2^n−1]), нумерация битов полублока от 0 до 2**n−1.

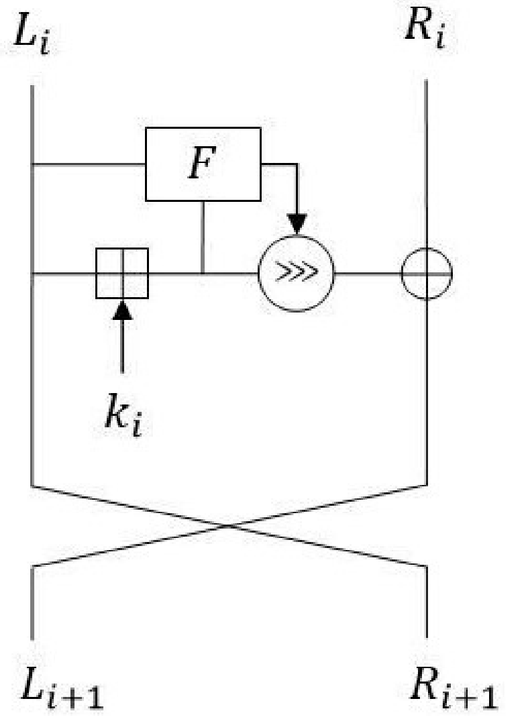
Интерпретируем L(i) ⊞ k(i) как вектор значений булевой функции от n переменных.
Второй выходной бит F получаем как значение данной функции на наборе битов из L(i) вида 2^i−1, где i=1,…, n,
то есть как (L(i) ⊞ k(i)) (L(i) [2^1−1,…,2^n−1]), нумерация битов полублока от 0 до 2^n−1.



Функция управления сдвигами

Для размера блока 64 бита (соответственно, полублока – 32 бита):

- **00 соответствует циклическому сдвигу на 11;**
- **01 соответствует циклическому сдвигу на 14;**
- **10 соответствует циклическому сдвигу на 10;**
- **11 соответствует циклическому сдвигу на 19.**



Число раундов r:
для размера блока 64 (полублока — 32): r=24;
для размера блока 128 (полублока — 64): r=28.

Размер ключа: 128, 192 или 256 бит.

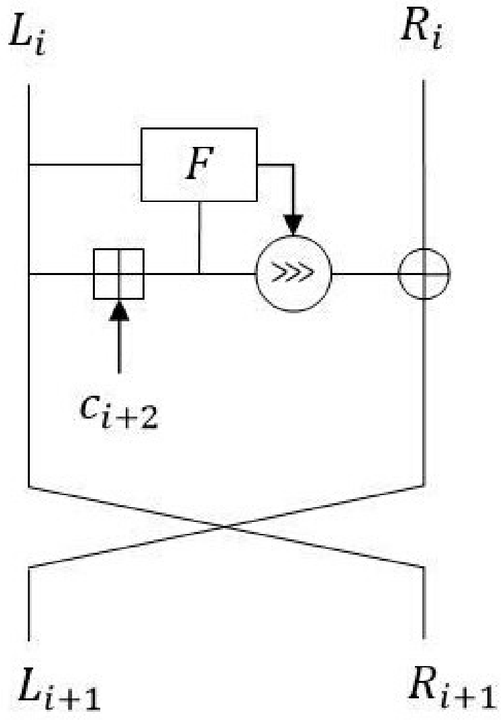
Функция выработки раундовых ключей

L(0)=c(0), R(0)=c(1), где значение константы c(i) получается следующим образом:

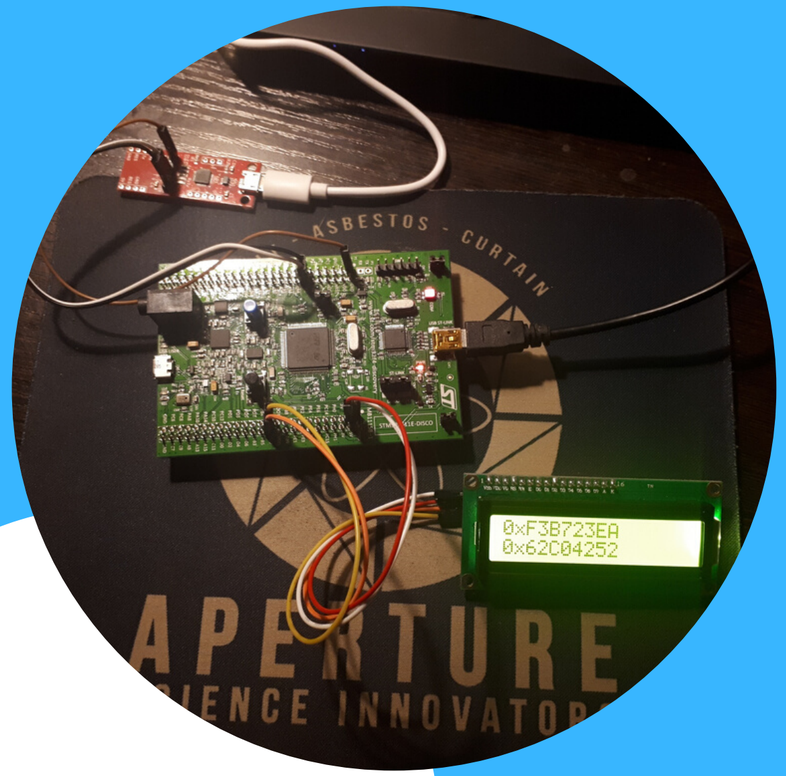
Ключ разбивается на L блоков длины 2^n, еще 8−L блоков получаем как значения квадратного корня из первых простых чисел.
Данные блоки соответствуют c(0), c(1), ..., c(7).

Далее при вычислении c(i) берем константу c(i) с индексом (i mod 6)+2 и складываем ее по модулю 2 с номером раунда c(i)=i⊕c((i mod 6)+2).

В качестве раундового ключа берем k(i)=L(i+1).



В результате работы мы:



- 1. Изучили схему и принцип работы алгоритма NASH
- 2. Реализовали алгоритм NASH
- 3. Установили связь с микроконтроллером
- 4. Установили связь с компьютером
- 5. Провели упрощенный анализ стойкости шифра

Спасибо за внимание!

РАБОТУ ВЫПОЛНИЛИ:

БИРЮКОВ АЛЕКСЕЙ, 618
КРОПАЧЕВА АЛИНА, 618
ГАЛЯЕВ ИВАН, 616
ВОЩИЛОВ ЕГОР, 612
ЗАРУБА ЮЛИЯ, 612