О создании эффективных программных реализаций отечественных криптографических стандартов

Казимиров Александр Владимирович
(Центр Сельмера по криптологии, Университет Бергена, Норвегия)
Леонтьев Сергей Ефимович
Попов Владимир Олегович
Смышляев Станислав Витальевич

Описание проблематики

- Использование ресурса параллелизма, доступного в целевой системе.
- Использование архитектурных расширений, позволяющих использовать графические процессоры: Nvidia CUDA, AMD FireStream.
- Возможность получения рекордных значений производительности при шифровании/хэшировании.
- Использование для подбора паролей.
- Проблемы при использовании в СКЗИ, в протоколах: накладные расходы, переносимость, необходимость отдельной обработки потока ключей.

Использование архитектурных расширений ЦП

- Одним из методов обеспечения параллельного выполнения операций является использование расширений, позволяющих реализовывать операции с векторами данных. Архитектурными расширениями, обеспечивающими эту возможность, доступными на большинстве современных вычислительных систем на платформах x86/x64, являются расширения SSE/AVX.

- Использование векторных регистров длины 128/256 бит – разбиение потока данных на однобайтовые потоки, одновременная обработка нескольких таких потоков.

- При шифровании: одновременная работа с несколькими блоками текста. Зашифрование: в режиме простой замены и гаммирования, расшифрование: в режимах простой замены, гаммирования, гаммирования с обратной связью и зацепления блоков (CBC).

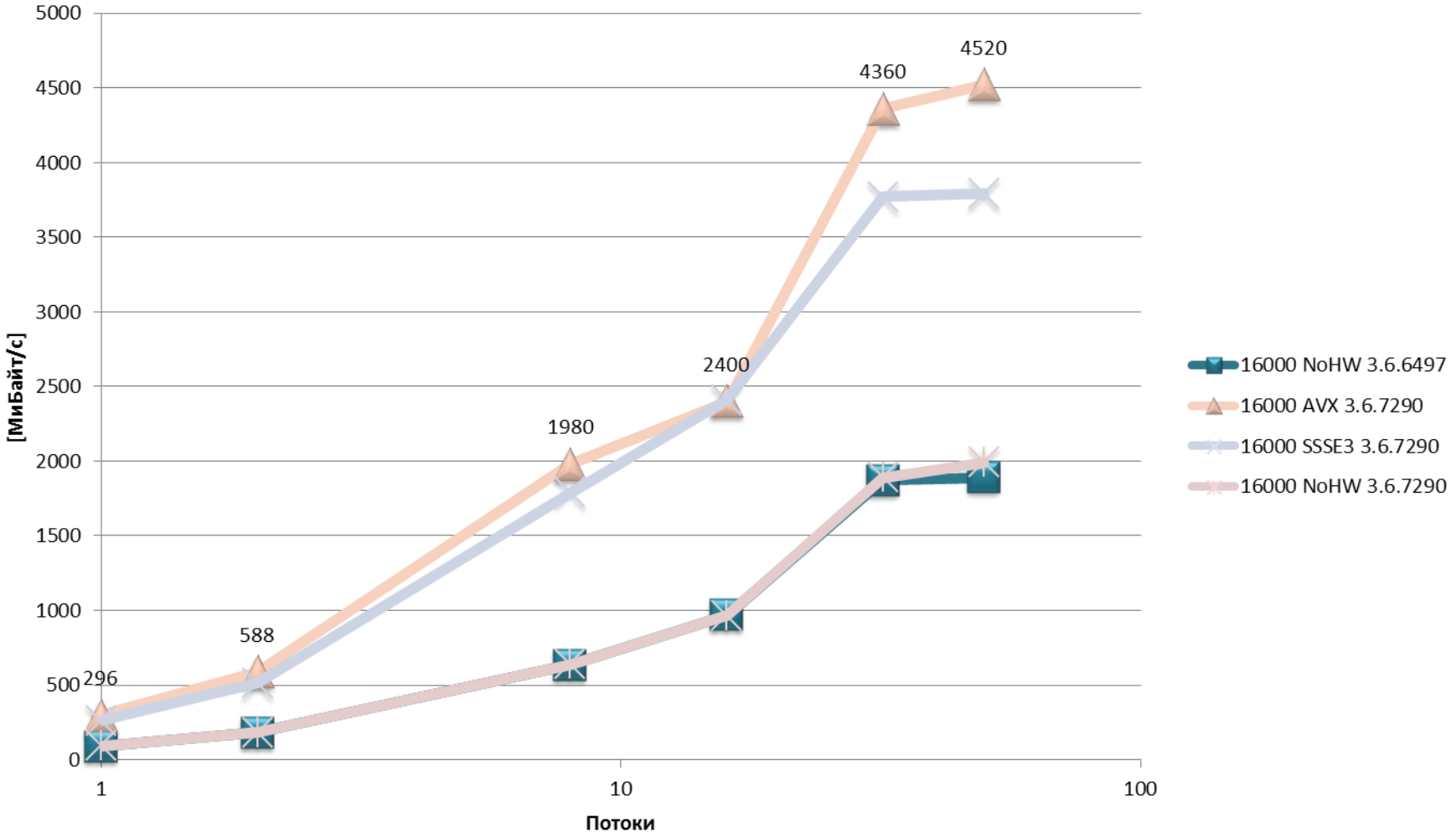
Использование архитектурных расширений ЦП

- С использованием данных расширений оказывается возможным ускорить шифрование по ГОСТ 28147-89 (непосредственно, без учета вспомогательных операций) более чем в 5 раз.

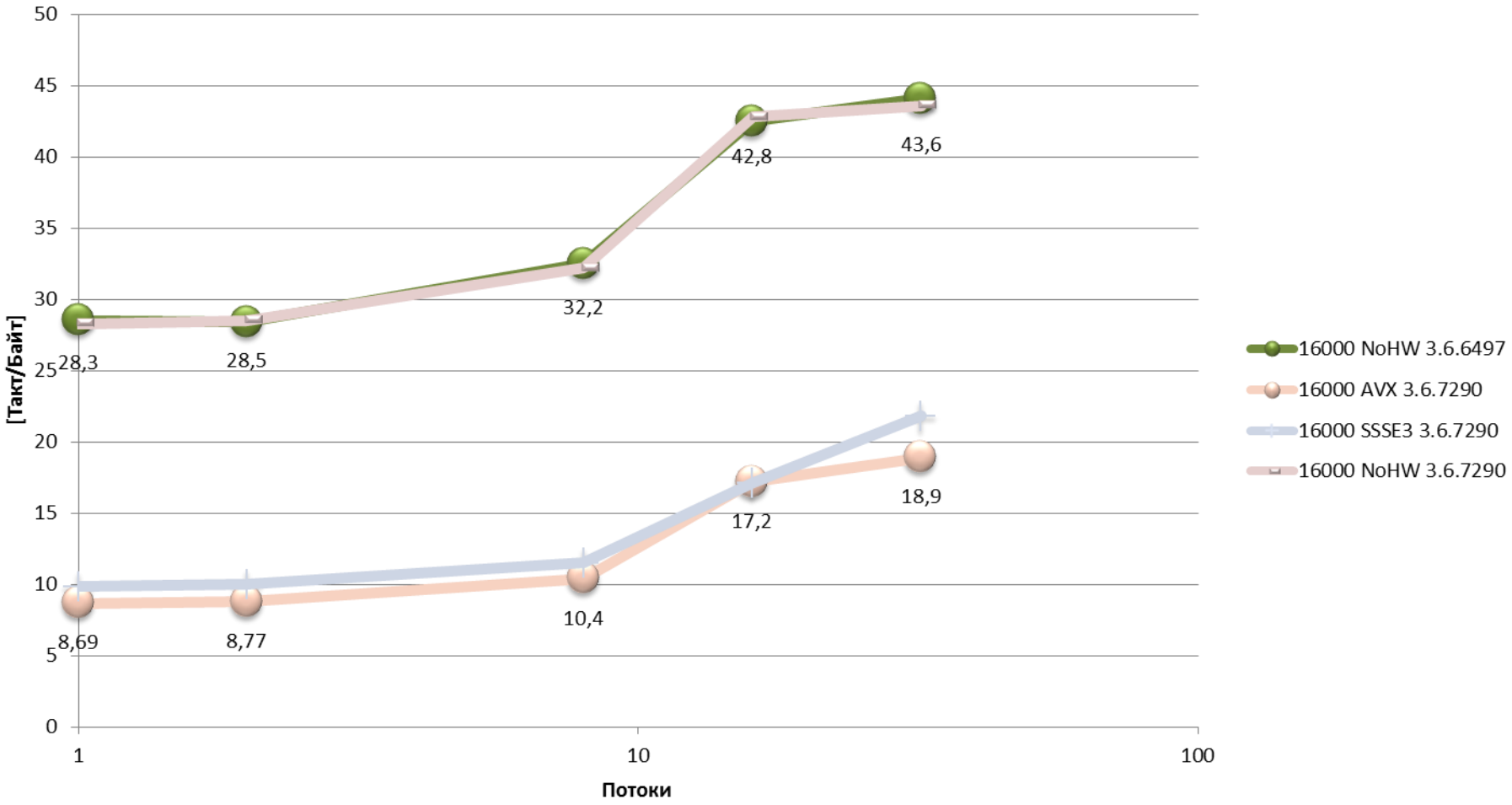
- С использованием данных расширений оказывается возможным ускорить шифрование по ГОСТ 28147-89 (при полноценном использовании в составе СКЗИ) более чем в 3.2 раза.

- Потенциальная возможность ускорить ГОСТ Р 34.11-94 в $\approx 4/(4-3s)$ раз, где s – доля шифрующего преобразования при вычислении функции сжатия.

Экспериментальные данные: шифрование в режиме гаммирования



Экспериментальные данные: шифрование в режиме гаммирования



Имитозащита

- При обработке данных в ряде криптографических протоколов (ESP IpSec, TLS и т.д.) требуется не только обеспечение конфиденциальности данных, но и их целостности, для чего применяются коды аутентификации сообщения, в случае российской криптографии – имитовставка.

- Заметим, что режим выработки имитовставки по ГОСТ 28147-89 предполагает зависимость каждого шага ее вычисления от предыдущих, что не позволяет реализовывать данный режим с помощью векторных операций как описано выше. Существует два принципиально разных подхода к решению проблемы аутентификации сообщений с помощью средств российской криптографии.

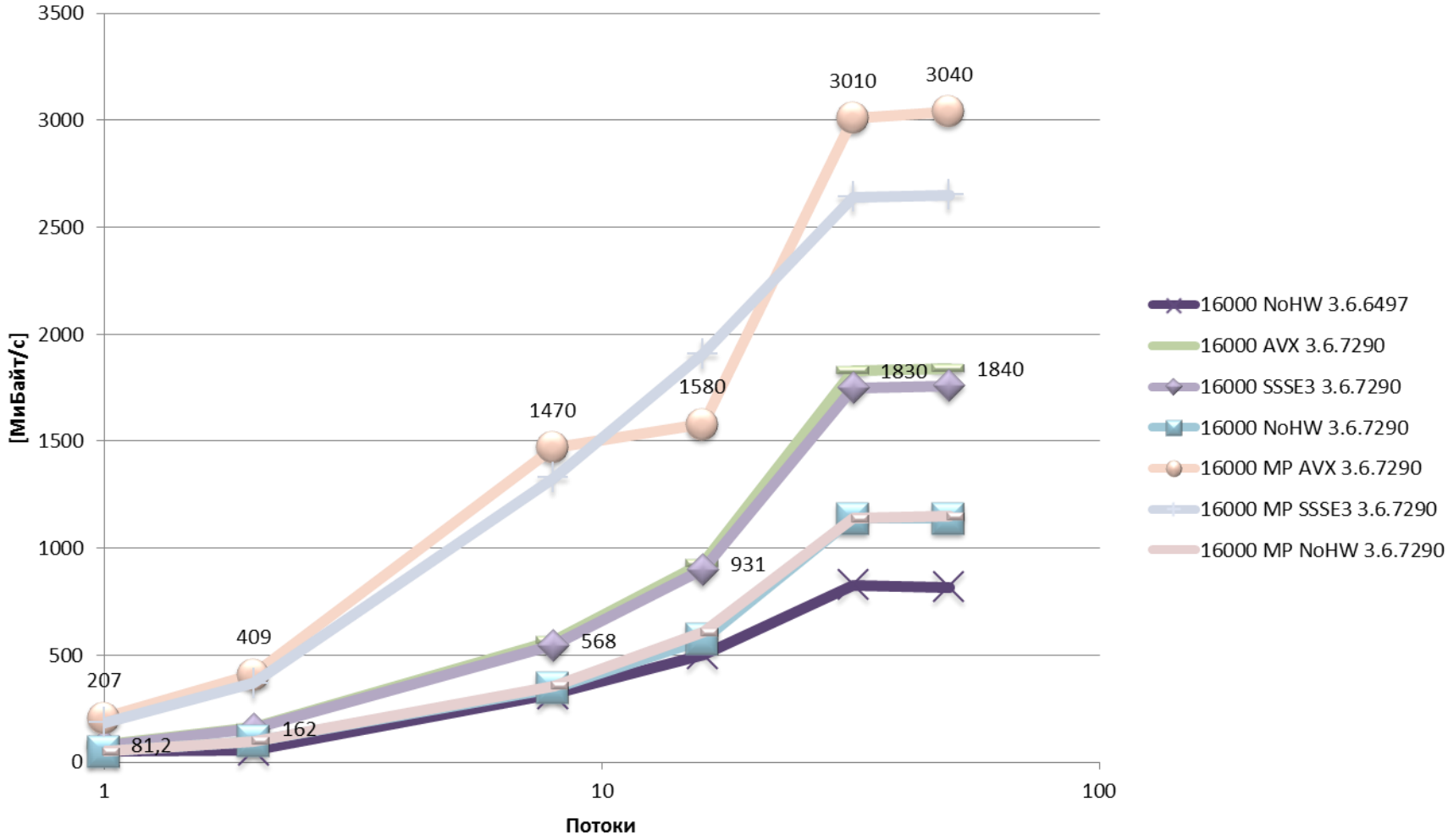
- Первый подход состоит в разработке и внедрении альтернативного режима выработки кода аутентификации сообщения. Данный альтернативный режим может быть построен, например, на основе режима PMAC.

Имитозащита

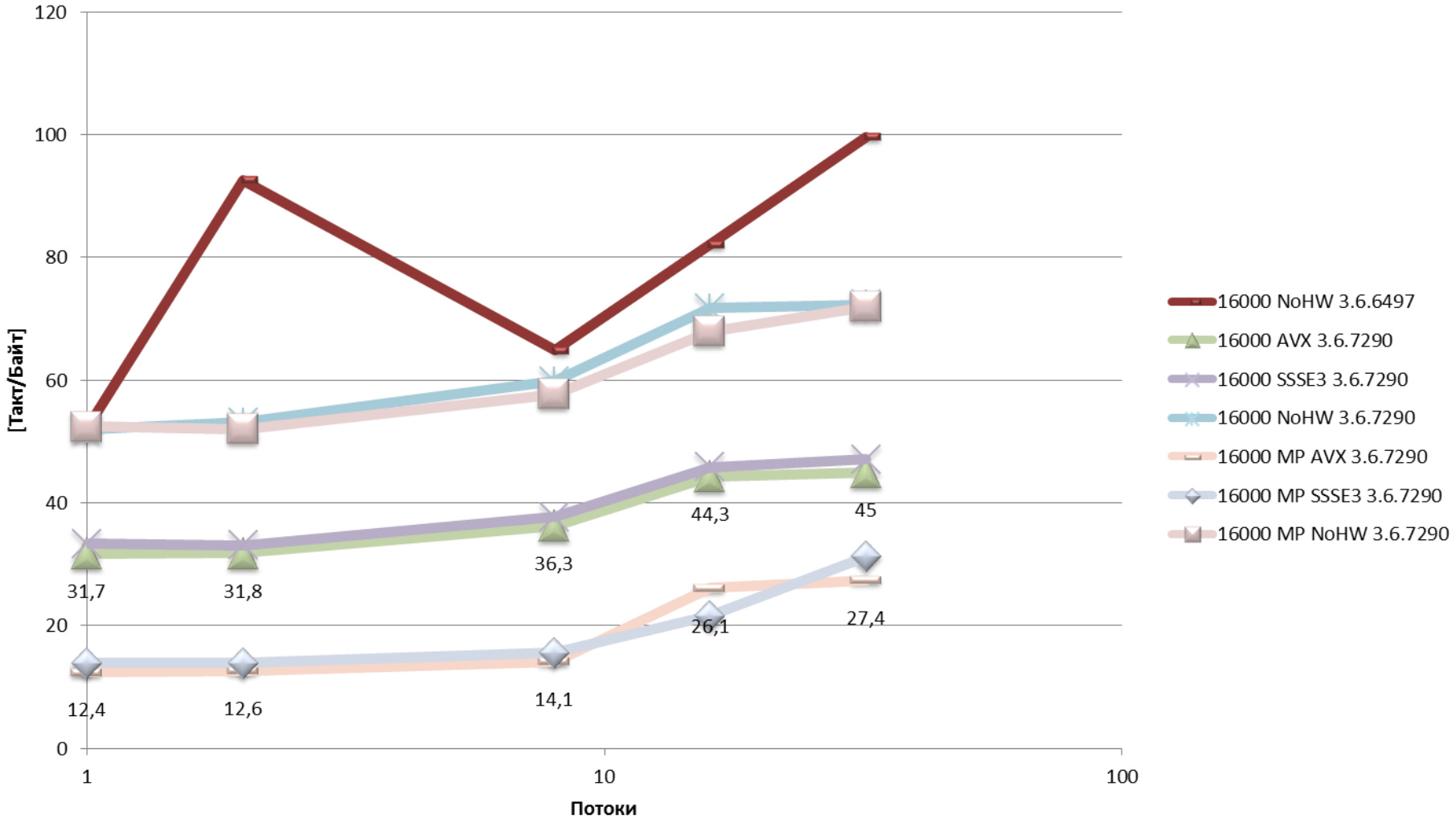
- Второй подход состоит в переходе в целях распараллеливания преобразований от одновременной обработки последовательных блоков одного пакета к одновременной обработке блоков с одинаковыми номерами различных пакетов. Данный способ применим в случае работы произвольного протокола, обрабатывающего большое количество пакетов так, что смена ключа производится раз в фиксированное количество пакетов.

- Очевидно, что эксплуатация данного подхода требует изменений как в реализации протокола, так и в интерфейсе криптографических вызовов. Примером интерфейсного решения может являться функция, на вход которой кроме ключа поступает массив пакетов, в котором каждый шифруется независимо от других (в случае режима гаммирования шифрование каждого из пакетов возможно производить также с одновременной обработкой нескольких блоков), подающийся далее целиком на обработку алгоритму выработки имитовставки.

Экспериментальные данные: шифрование и имитовставка



Экспериментальные данные: шифрование и имитовставка



Экспериментальные данные: шифрование в режиме гаммирования

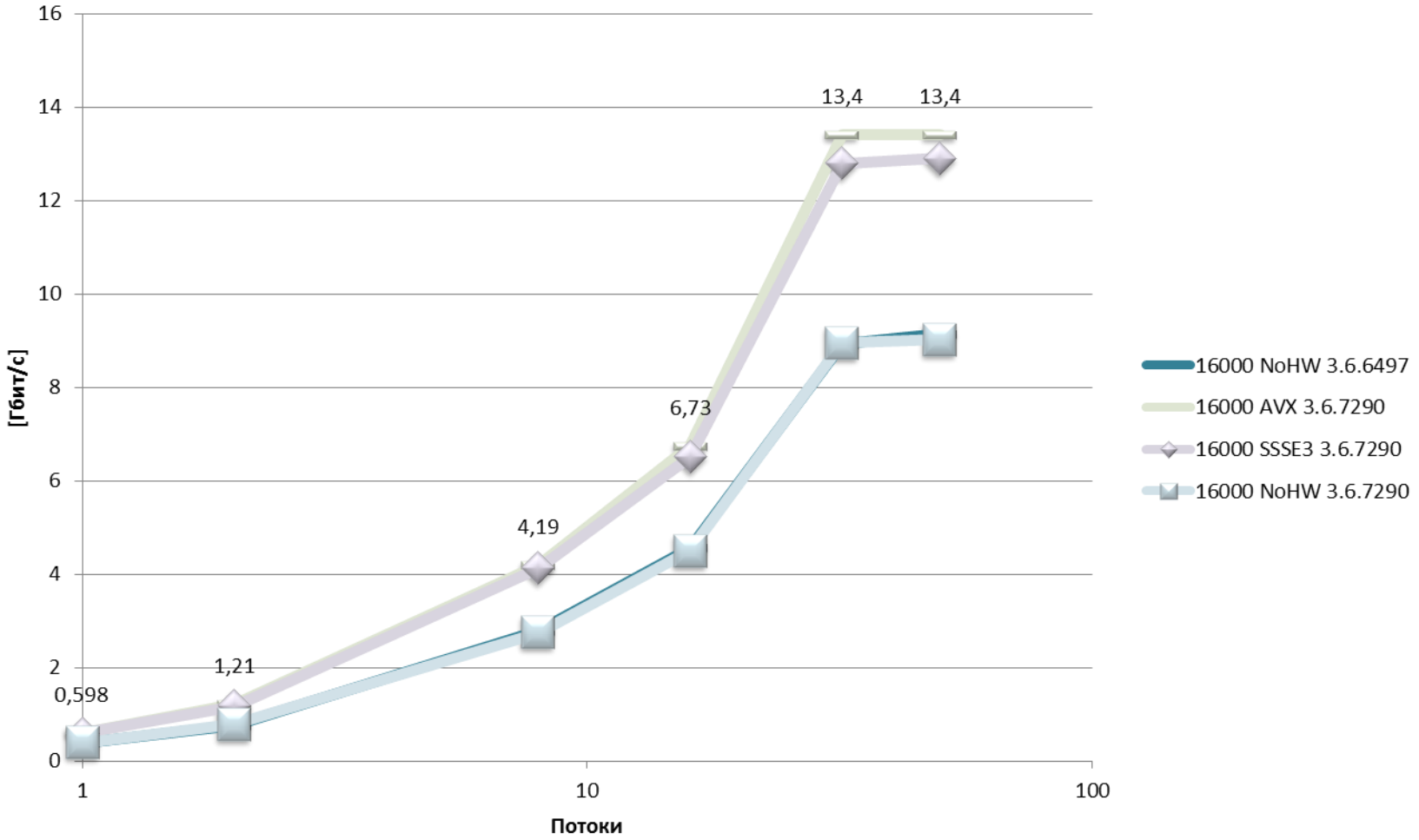
Процессор: Xeon(R) E5-2680 2.7 ГГц

В скобках указывается производительность одного потока, как абсолютная, так и в тактах процессора.

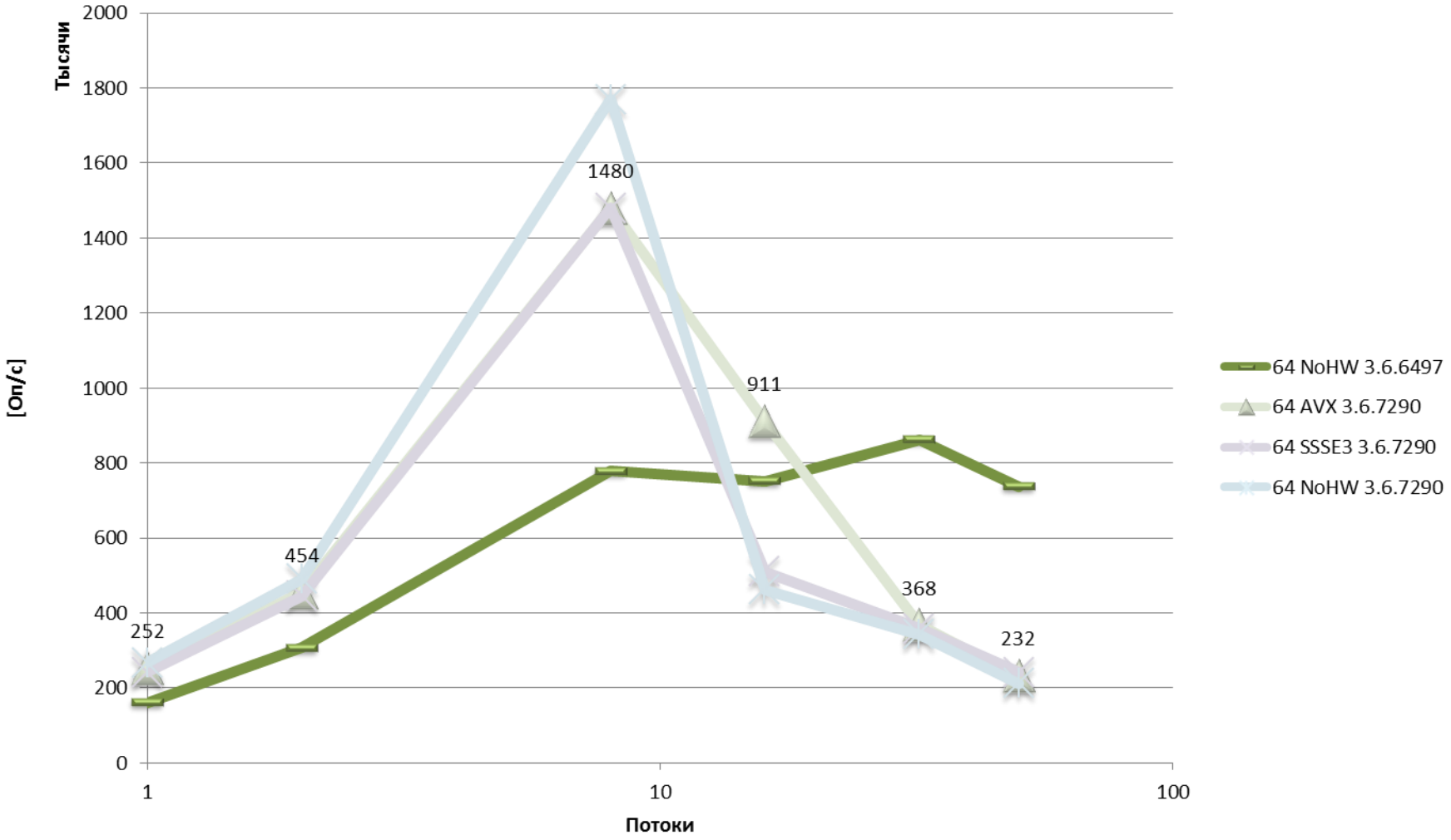
- x64: Шифрование в режиме гаммирования:
  4300 МБ/с (290 МБ/с, 8.9 такт/Байт);
- x64: Шифрование с одновременным вычислением имитовставки:
  3000 МБ/с (210 МБ/с, 12.2 такт/Байт);

- Win32: Шифрование в режиме гаммирования:
  3500 МБ/с (240 МБ/с, 10.3 такт/Байт);
- Win32: Шифрование с одновременным вычислением имитовставки):
  2500 МБ/с (170 МБ/с, 14.9 такт/Байт);

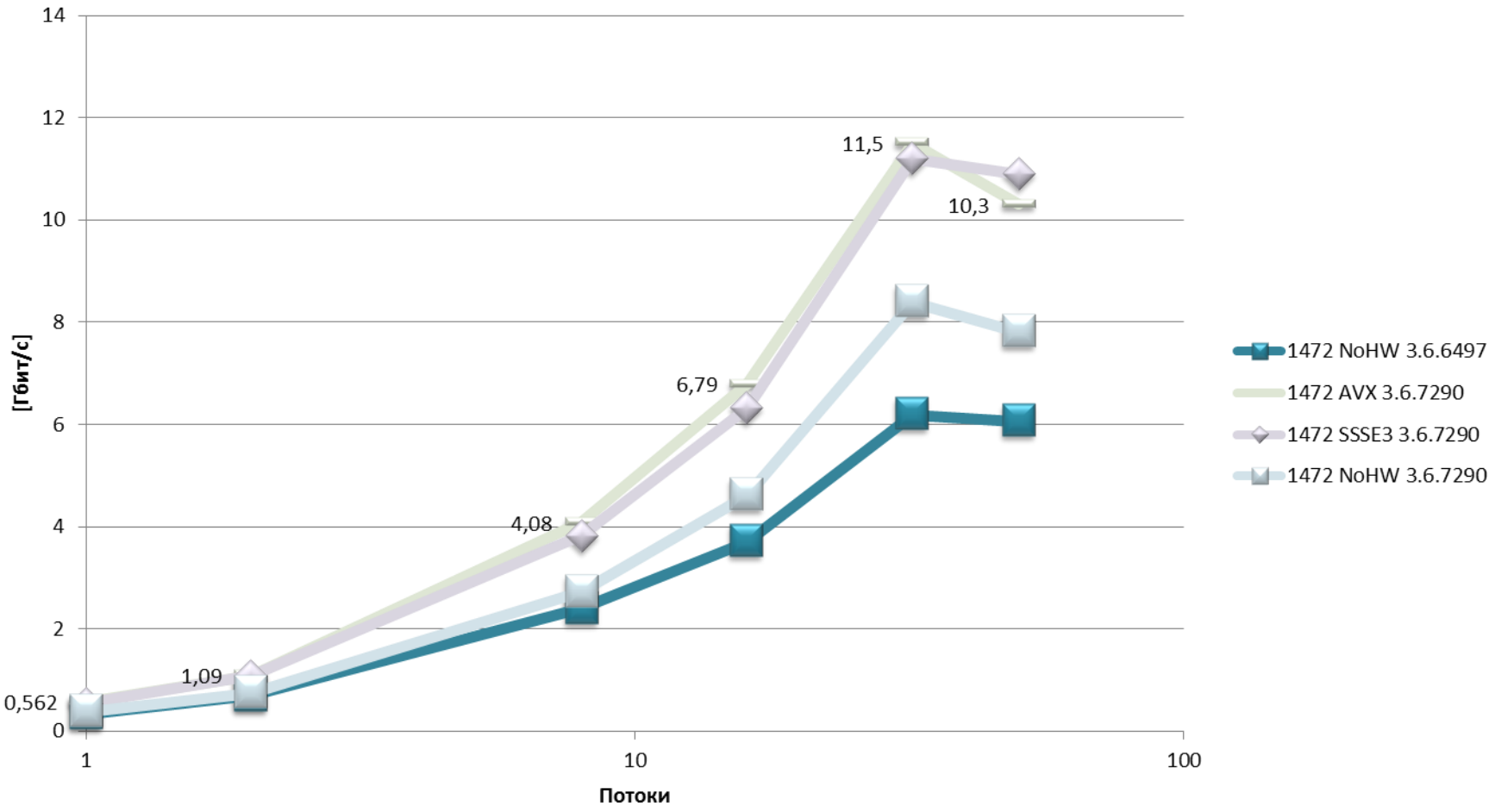
Экспериментальные данные: TLS, фаза передачи данных



Экспериментальные данные: ESP IPSec, пакеты по 64 байта



Экспериментальные данные: ESP IPSec, пакеты по 1472 байта



Функция хэширования

- По описанным ранее причинам (проблемы с переносимостью, накладные расходы, требование постоянного наличия многих независимых потоков данных) не будем рассматривать реализации на GPU, рассмотрим только реализации на центральном процессоре.

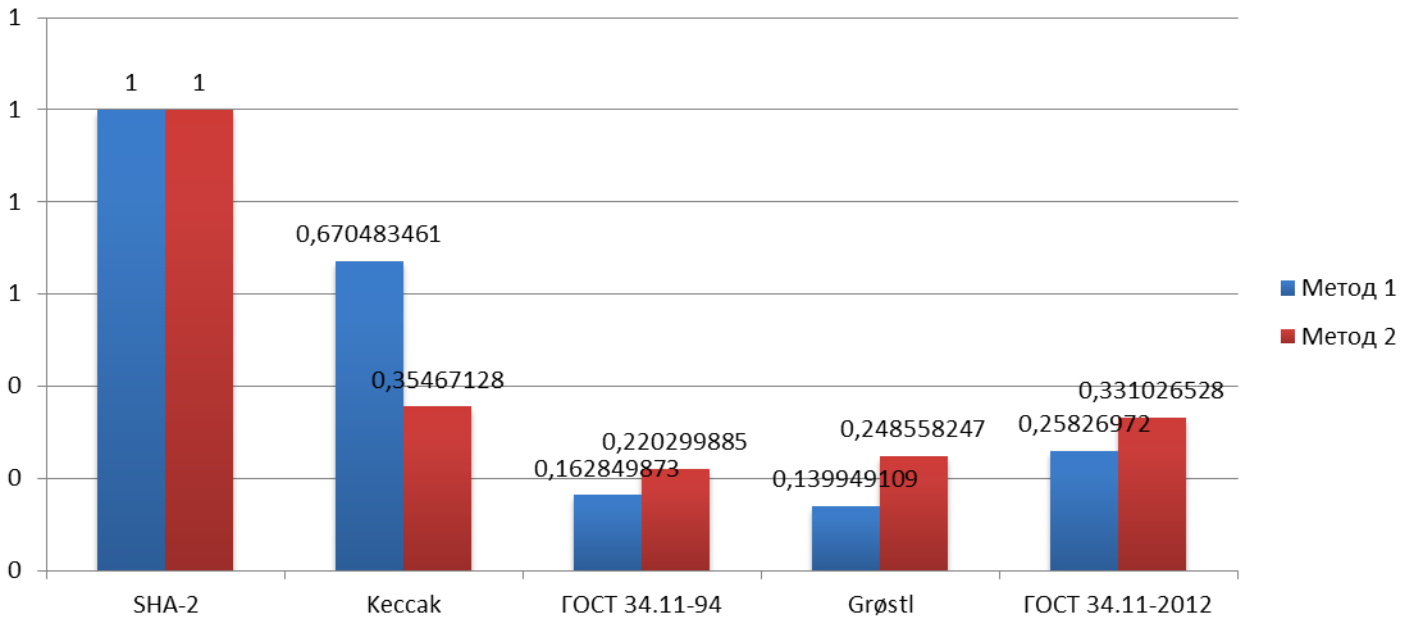
- Приведем экспериментальные данные с относительной производительностью полученных реализаций российских стандартов функции хэширования, а также хэш-функций SHA-2, SHA-3 (Keccak) и Grostl на Intel i7 3615QM 2.30 Гц 64 бит с 8 Гб ОЗУ с ОС Darwin Kernel Version 12.2.1:18.10.2012 x86_64, при сборке с помощью gcc version 4.2.1.

- Проводились исследования по двум методам:
  1) многоразовый подсчёт хэш-значения одного блока сообщения;
  2) одноразовое вычисление хэш-значения длинного сообщения.

1) тестирует всю хэш-функцию, начиная от этапа инициализации параметров, заканчивая выдачей хэш-значения. 2) ориентирован на тестирование производительности функции сжатия.

Функция хэширования

Проводились исследования по двум методам:
1) многоразовый подсчёт хэш-значения для одного блока сообщения;
2) одноразовое вычисление хэш-значения для длинного сообщения.



Функция хэширования

- Таким образом, реализацию нового стандарта функции хэширования ГОСТ Р 34.11-2012 благодаря эксплуатации свойств преобразования LPSX возможно построить примерно в полтора раза более быстрой, чем не использующую ресурс параллелизма реализацию ГОСТ Р 34.11-94.

- Заметим, что в предыдущем стандарте функции хэширования ГОСТ Р 34.11-94 основное преобразование состоит из четырех выполняющихся параллельно и независимо зашифрований блоков по ГОСТ 28147-89. Таким образом, данный этап потенциально возможно ускорить в 4 раза, а работу функции хэширования в целом в 1/(0.25*d+(1-d)) раз, где d – доля выполнения кода шифрования при работе функции сжатия хэш-функции.

- Однако, в отличие от ГОСТ Р 34.11-94, параллелизуемых блоков существенного размера в новом стандарте нет, из чего представляется уместным сделать вывод об отсутствии в конструкции ГОСТ Р 34.11-2012 существенного ресурса для ускорения его реализаций.

Задачи

- Эффективное использование GPU для работы в составе СКЗИ: уменьшение накладных расходов, переносимость, обеспечение дополнительных требований.

- Создание реализаций с помощью фреймворков, обеспечивающих относительно простую переносимость (OpenCL) для решения части проблем.

- Динамический выбор аппаратных средств (GPU/ SSE3 / AVX/…) в процессе работы для оптимизации производительности решения конкретной криптографической задачи на конкретной системе.

- Интерфейсные решения для поддержки одновременной обработки нескольких потоков данных.

СПАСИБО ЗА ВНИМАНИЕ!

КРИПТО-ПРО – ключевое слово в защите информации

http://www.cryptopro.ru
vpopov@cryptopro.ru
svs@cryptopro.ru

Тел./факс:
+7 (495) 780-48-20
+7 (495) 660-23-30