ционных технологий Национального авиационного университета.
Korchenko Anna, PhD in Eng., Associate Professor of Academic Department of IT.

Гізун Андрій Іванович, асистент кафедри безпеки інформаційних технологій Національного авіаційного університету.
E-mail: andriy.gizun@gmail.com
Гизун Андрей Иванович, асистент кафедры безопасности информационных технологий Национального авиационного университета.
Gizun Andrii, Assistant of Academic Department of IT-security, National Aviation University.

УДК 004.056

СКОРОСТНОЕ УНИВЕРСАЛЬНОЕ ХЕШИРОВАНИЕ НА ОСНОВЕ МНОГО ПОТОКОВЫХ ВЫЧИСЛЕНИЙ

Евгений Котух, ВладимирКарташов, Денис Цапко, Олег Халимов, Алина Самойлова

Универсальное хеширование определяет доказуемо стойкую аутентификацию со счетчиком, обеспечивает высокую стойкость к коллизиям и скорость вычислений. Одним из наиболее перспективных направлений в решении задач высокоскоростных вычислений является использование технологии GPGPU (General-purpose graphics processing units). Технология GPGPU позволяет на одном вычислителе достигать высокого уровня параллелизма без временных затрат на передачу данных между узлами и синхронизацию результатов вычислений. Для повышения скорости универсального хеширования на основе скалярных и полиномиальных вычислений разработаны предложения по много потоковым вычислениям, вычислениям в модулярной арифметике и арифметике над расширенным конечным полем с использованием GPGPU технологии.

***Ключевые слова**: универсальное хеширование, много потоковые вычисления, архитектура массивно-параллельных вычислений CUDA.*

ВВЕДЕНИЕ. В качестве основных составляющих прироста мощности вычислительных систем можно выделить: рост производительности выделенных вычислительных устройств; организацию массовых вычислений в совокупности устройств, в том числе и мобильных; использование эффективных моделей вычислений на существующих классах архитектур. Одним из наиболее перспективных направлений в решении задач вычислений общего назначения является использование технологии GPGPU (General-purpose graphics processing units) [1, 2]. Графический процессор (GPU) обладает меньшим набором исполняемых команд (RISC-подобные архитектуры), чем CPU, но большей производительностью. Технология GPGPU позволяет на одном вычислителе достигать достаточно высокого уровня параллелизма без временных затрат на передачу данных между узлами и синхронизацию результатов вычислений. Относительная низкая стоимость, простота добавления вычислительных модулей и удельное энергопотребление в сочетании с высокой удельной производительностью GPU позволяют реализовать на практике массовые распределенные параллельные вычисления, которые доступны более широкому кругу потенциальных нарушителей. Использование вычислений на GPGPU в решение целого ряда вычислительно сложных задач в области проблем криптографии представляет практический и научный интерес.

Широкое распространение получили алгоритмы шифрования и цифровой подписи (Эль-Гамаль, DSA, ГОСТ), стойкость которых основана на сложности решения задачи дискретного логарифмирования в мультипликативной группе числового поля и в группе точек эллиптической кривой над конечным полем. Особенностью данной задачи является быстрый рост времени выполнения при увеличении размера задачи. Эффективный алгоритм параллельного выполнения на GPGPU архитектуре, предназначенный для ускорения решения задачи дискетного логарифмирования в различных группах с помощью распределённых многопроцессорных вычислений представлены в работе [1]. Для вычисления дискретного логарифма в группе точек эллиптической кривой наиболее эффективным считается ρ-метод Полларда реализация которого на GPU в работе [3] дала 100-кратный прирост производительности по сравне-

нию с CPU. Алгоритмы, предназначенные для мультипликативной группы числового поля, рассматриваются в работах [1, 2]. Быстродействующий алгоритм деления полиномов в арифметике по модулю два, позволяющий работать с данными в параллельном виде, а также матричный алгоритм деления полиномов представлен в работе [4]. Среди перспективных направлений применения GPU следует отметить криптоанализ методом прямого перебора, обслуживание криптовалют, построение rainbow-таблиц, подходы к «интеллектуальному» сокращению перебора. Тем не менее, существует целый ряд ограничений, связанных с особенностями архитектуры GPU: «SIMD-эффект» [5], произвольный доступ к памяти [6]. Так, алгоритм DPLL не выигрывает от переноса на GPU, в отличие от более простых способов криптоанализа. Архитектура современных GPU плохо приспособлена к условными переходам (часть вычислительных ядер группы вынуждена простаивать в ожидании необходимой команды) для выполнения DPLL и подобных ему сложных поисковых алгоритмов [6].

Успешные атаки и рост вычислительной емкости предъявляют высокие требования к стойкости криптографических примитивов. Недостатки, выявленные в криптографических стандартах 90-х годов и обоснованные теоретические атаки, обуславливают развитие новых методов реализации криптопримитивов. Применение методов криптоанализа дает возможность почти неограниченного масштабирования на параллельных вычислительных архитектурах. Многопоточное распараллеливание имеет значительную эффективность для атак полного перебора, что послужило причиной полного отказа от использования неактуальных хеш-функций (MD5, SHA-1). Наиболее эффективные результаты в скорости полного перебора MD5 (16 млрд. и более хеш/с) достигнуты в работе [7]. Известные результаты полного перебора для SHA-1 представлены в работе [8]. Обобщим эти данные в табл. 1 (в качестве стенда для воспроизведения результатов использовались: CPU - Core i7-3770K 3,5 Ггц, 8 ГБ, GPU - Nvidia GTX 680 с различными реализациями алгоритмов полного перебора.

Таблица 1

Эффективность атак полного перебора при многопоточном распараллеливании

Скорость перебора, млн хеш/сек	CPU	GPU	$GPU_{optimized}$	EGP	GPU_{IGHASH}	$2xGPU_{optimized}$	$2xGPU_{IGHASH}$
SHA-1	4	78	165	180	532	243	962

Конкурс SHA-3, организованный NIST, показал развитие требований к безопасности, архитектуре и производительности при реализации криптографических примитивов. В финале конкурса SHA-3 двое из пяти финалистов (Keccak и Skein) оказались универсальными криптопримитивами, которые могут использоваться не только для хеширования, но и для выполнения множества других криптографических операций, обеспечивая упрощение проектируемых криптографических протоколов.

Ранее считалось, что параллельные режимы возможны только для блочных шифров (OCB, GCM) или при использовании громоздкого режима древовидного хеширования. В работе [9] группой авторов был предложен способ параллельного вычисления MAC-кода на основе конструкции «губка» (алгоритм Keccak). Ключ объединяется с вектором инициализации (Nonce) и подается на вход множества параллельных конструкций, где предварительно объединяется еще и со значением счетчика каждой конструкции. Возможность использовать Keccak в параллельном режиме существенно упрощает создание протоколов, требующих шифрования с аутентификацией, и избавляет от множества потенциальных ошибок в их проектировании и реализации.

Целью исследований является повышение скорости универсального хеширования на основе реализации распараллеливания вычислений.

ПОСТАНОВКА ЗАДАЧИ ИССЛЕДОВАНИЙ. Научная задача построения доказуемо секретной аутентификации впервые сформулирована в работе [10]. Решение этой задачи было предложено в классе универсальных хеш-функций, как аутентификации с максимальной теоретически достижимой секретностью. Идеи универсальной аутентификации получили развитие в теории безусловной аутентификации с использованием строго универсального хеширования.

Для достижения высокой скорости аутентификации необходимо решить задачу разработки методов скоростного универсального хеширования на основе распараллеливания вычислений и оценки их параметров.

ОПРЕДЕЛЕНИЯ УНИВЕРСАЛЬНОГО ХЕШИРОВАНИЯ. Универсальное хеширование определяет доказуемо стойкую аутентифика-

цию со счетчиком в представлении Картера – Вегмана [10] и определяется семействами хеш функций с заданными комбинаторными свойствами. Основные результаты исследований представлены в работах Стинсона, Биербрауэра и др. [11].

Определение 1 [12]. $(N;n,m)$ хеш семейство является ε – универсальным, если для любых двух различных элементов $x_1, x_2 \in A$, существует самое большее εN функций $h \in H$ таких, что $h(x_1) = h(x_2)$. Аббревиатура $\varepsilon - U$ используется для обозначения ε – универсальных хеш функций.

Замечание 1. Массив значений MAC кодов состоит из N строк, n столбцов, элементы принимают одно из m значений. Каждая функция $h \in H$ определяется значением используемого ключа, связывается со строкой и определяет правило отображения элементов множества A (номеров столбцов массива) в элементы B (собственные значения элементов массива).

Утверждение 1 [12]. Пусть h выбирается случайно из заданного $\varepsilon - U(N;n,m)$ хеш семейства, тогда вероятность коллизии хеш значений для двух разных входных сообщений $x_1, x_2 \in A$ не превышает ε.

Замечание 2.

1. Первоначальное определение универсальных хеш функций было предложено Картером и Вегманом для $\varepsilon = 1/m$ [12].

2. Вероятность коллизии для универсальных хеш функций Картера и Вегмана является наименьшей и определяется мощностью пространства хеш значений $P_{кол} = 1/|B|$.

Определение 2 [12]. H является ε – почти универсальным семейством хеш функций $(\varepsilon - AU(N;n,m))$, если $P_{кол} = \Pr_{h \in H}[h(x_1) = h(x_2)] \le \varepsilon$ для $x_1, x_2 \in A$, $x_1 \ne x_2$, $1/m < \varepsilon \le 1$.

Замечание 3. Для почти универсальных семейств несколько ослабляются требования к вероятности коллизии.

Универсальное хеширование реализуется на основе следующих методов:

- метода скалярного произведения;
- метода полиномиального хеширования;
- метода скалярного произведения по рациональным функциям алгебраических кривых.

МЕТОД СКАЛЯРНОГО ПРОИЗВЕДЕНИЯ. Построение универсальных хеш- функций предложено Картером и Вегманом в методе скалярного произведения [13].

Хеш вычисление y над конечным полем F_q определяется функцией вида

$$y = \sum_{i=1}^{k} x_i m_i ,$$

где $y, x_i, m_i \in F_q, . m_i$ - слова сообщения, x_i - слова ключа, k - число слов сообщения.

Замечание 4.

1. Хеш функция на основе скалярного произведения определяет хеш класс $\varepsilon - U\left(q^k, q^k, q\right)$ с вероятностью коллизии $\varepsilon = 1/q$.

2. Решение задачи построения доказуемо секретной аутентификации с максимальной теоретически достижимой секретностью достигается в классе универсальных хеш-функций со скалярным произведением.

3. Решение задачи повышения быстродействия универсального хеширования в методе скалярного произведения возможно в реализации с распараллеливанием на графическом процессоре (GPU).

Модель вычислительного устройства (ядра) GPU-процессора определяется верхним уровнем ядра, который состоит из блоков одинакового размера, которые группируются в сетку (grid) размерностью $N_1 \times N_2$. При использовании GPU-процессора можно задействовать грид необходимого размера и при помощи CUDA-технологии сконфигурировать блоки под параметры поставленной вычислительной задачи.

Оптимизация производительности, как правило, сводится к следующим шагам:

1) максимальное использование параллелизма задачи;

2) оптимизация доступа в память;

3) оптимизация машинной арифметики.

Каждое ядро GPU-процессора может работать одновременно с очень большим числом нитей, поэтому, чтобы ядро могло однозначно определить номер нити, в CUDA используются встроенные переменные threadIdx и blockIdx.

При написании параллельного кода для GPU-процессора CUDA-технология имеет ряд дополнительных расширений языка C:

— Спецификаторы функций, которые показывают, как и откуда будут выполняться функции.

— Спецификаторы запуска ядра GPU.

— Спецификаторы переменных, которые служат для указания типа используемой памяти GPU.

— Встроенные переменные для идентификации нитей, блоков и др. параметров при исполнении кода в ядре GPU.

Спецификаторы функций определяют, как и откуда будут вызываться функции. Всего в CUDA три таких спецификатора:

— __host__ — выполняется на CPU, вызывается с CPU;

— __global__ — выполняется на GPU, вызывается с CPU;

— __device__ — выполняется на GPU, вызывается с GPU.

Спецификаторы запуска ядра служат для описания количества блоков, нитей и памяти, которые необходимо выделить при расчете на GPU-процессоре.

Общим приемом в CUDA-технологии является то, что исходная задача разбивается на набор отдельных подзадач, решаемых независимо друг от друга. Каждой такой подзадаче соответствует свой блок нитей. Причем каждой отдельной нити соответствует один элемент вычислительных данных.

Анализ решения задачи повышения быстродействия универсального хеширования в методе скалярного произведения предлагается реализацию следующих действий.

На первом шаге необходимо решить проблему «последовательного участка» параллельного алгоритма. Возможные варианты решения — это уменьшение такого участка либо попытка преобразовать его большую часть в параллельный код. Одно потоковый алгоритм универсального хеширования разбивается на много потоковый на p процессоров. Это в p раз уменьшает сложность вычислений.

Следующий этап заключается в скоростной реализации целочисленных операций умножения и сложения в модулярной арифметике. Известные алгоритмы с вычислениями 64 битных чисел на графических процессорах позволяют повысить быстродействие в 81,1 и более раз [14].

Недостатком универсального хеширования $y = \sum_{i=1}^{k} x_i m_i$ является требование - размер ключевого пространства должен быть не меньше пространства сообщения.

Ограничение на размер ключевого пространства снимается в методе полиномиального хеширования.

МЕТОД ПОЛИНОМИАЛЬНОГО УНИВЕРСАЛЬНОГО ХЕШИРОВАНИЯ. Полиномиальное универсальное хеширование PolyCW хеширование (polynomial Carter-Wegman hashing) предложено для снятия ограничения на пространство ключей.

Полиномиальное хеширование y над конечным полем F_q определяется функцией вида

$$y = \sum_{i=1}^{k} m_i x^i,$$

где $y, x, m_i \in F_q, . m_i$ - слова сообщения, x - ключевое слово, k - число слов сообщения $k < q$.

Замечание 5.

1. Хеш функция на основе полиномиального вычисления определяет хеш класс $\varepsilon - U\left(q, q^k, q\right)$ с вероятностью коллизии $\varepsilon = k / q$. Вероятность коллизии для полиномиальной хеш-функции ограничивается отношением k / q, где q простое число, определяющее поле F_q. Значение ε определяется фундаментальной теоремой алгебры: многочлен отличный от нуля, степени не меньше k, имеет не меньше k корней.

2. Множество ключей определяется значением q. Чем больше размер ключевого пространства, тем большее количество слов можно хешировать до достижения допустимой вероятности коллизии. Как следует из анализа зависимостей вероятности коллизии от значения поля вычислений, размер конечного поля F_q должен быть как можно большим.

3. Практическая схема эффективного алгоритма хеш вычисления должна включать ещё одно дополнительное преобразование. Векторы чисел с w битами, которые имеют элементы вне диапазона представления Z_q, необходимо преобразовать в вектор, который их не имеет с помощью так называемого двойного представления, как это сделано в UMAC алгоритме. Это приводит к удвоению размера данных и соответственно к увеличению вероятности коллизии и соответственно снижается скорость вычислений.

Анализ решения задачи повышения быстродействия полиномиального универсального хеширования предлагается реализацию следующих действий.

На первом шаге необходимо решить проблему распараллеливания алгоритма. Вычисление хеш-функции $y = \sum_{i=1}^{k} m_i x^i$ можно осуществить по итерационной схеме Горнера, с одной операцией умножения и сложения в конечном поле на каждом шаге. Лучший результат достигается в арифметике Z_q при как можно большом простом числе $q < 2^w$. Итерационная схема Гор-

нера вычисления хеш-функции предполагает вычисление

$$y \leftarrow xy + m \bmod q .$$

Полиномиальное вычисление разбивается на многопотоковое на p процессоров

$$y = \sum_{j=0}^{p-1} x^{jp} \sum_{i=1}^{k/p} m_i x^i .$$

Результат такого разбиения – двух параметрическая схема вычисления. Применение к полиномиальным вычислениям по каждой сумме итерационной схемы Горнера со сложностью $k/p + p$, практически в p раз повышает быстродействие полиномиального хеширования.

Следующий этап заключается в скоростной реализации операций умножения и сложения в модулярной арифметике или арифметики над расширенным полем F_q. Арифметика над расширенным полем F_q характеристики 2 является менее удобной для современных микропроцессоров, хотя обеспечивает легкое разделение хешируемых битовых строк на w битовые подстроки. При этом отсутствуют потери по вероятности коллизии и по объему ключевых данных, которые возникают при хешировании в арифметике Z_q. Известно несколько эффективных алгоритмов быстрого умножения в F_{2^w}, ориентированных на табличное умножение элементов поля или являющихся модификациями схемы Монтгомери. Известные алгоритмы с вычислениями над расширенным полем F_q характеристики 2 на графических процессорах позволяют повысить быстродействие в 10 и более раз по сравнению с однопроцессорными вычислениям при уменьшение количества операций обмена данными между основной оперативной памятью и внутренней памятью GPU и максимальным использованием параллелизма [15].

Недостатком полиномиального хеширования является требование - размер пространства сообщений ограничивается условием для вероятности коллизии $\varepsilon = k/q$ и размером поля вычислений.

Ограничение на размер пространства сообщений снимается в методе универсального хеширования по алгебраическим кривым.

МЕТОД УНИВЕРСАЛЬНОГО ХЕШИРОВАНИЯ ПО АЛГЕБРАИЧЕСКИМ КРИВЫМ. **Определение 3** [16]. Пусть задана абсолютно неразложимая, несингулярная проективная кривая χ над полем F_q с точками $P = \{P_1, P_2, ..., P_n\} \in \chi(F_q)$. Для каждой алгебраической кривой можно определить поле рациональных функций $F_q(\chi)$. В каждой точке P_j кривой χ можно вычислить оценку ϑ_P для рациональных функций $f_i \in F_q(\chi)$, которая определяет порядок нуля или полюса функции f_i в этой точке. Хеш значение $h_{P_j}(m) \in F_q$ для сообщения $m = (m_1, ..., m_k)$, $m_i \in F_q$ в точке $P_j \in F_q$ определяется выражением

$$h_{P_j}(m) = \sum_{i=1}^{k} f_i(P_j) m_i ,$$

где $f_i \in F_q(\chi)$ с упорядоченными порядками полюсов $0 < \rho_1 < ... < \rho_k$. Хеш функция $h_{P_j}(m)$ определяет универсальный хеш класс ε–$U(N, q^k, q)$, где вероятность коллизии $\varepsilon \le \rho_k / N$, N - число точек алгебраической кривой.

Замечание 6.

1. Выражение $h_{P_j}(m)$ определяет хеш вычисление на основе скалярного произведения по рациональным функциям алгебраических кривых. Метод универсального хеширования определяется последовательностью следующих действий:

- определить проективное многообразие – алгебраическую кривую и её точки;
- построить линейное векторное пространство для функционального поля алгебраической кривой;
- задать хеш функцию как скалярное произведение слов данных и значений рациональных функций в точке кривой.

2. Параметры универсального хеш класса $\varepsilon - U(N, q^k, q)$ на основе хеширования по рациональным функциям определяются свойствами алгебраической кривой. Подгруппа Вейерштрасса $H(P_\infty) = \{\rho_0 = 0 < \rho_1 < ...\}$ определяется полюсами рациональных функций в особой точке кривой и рациональные функции упорядоченные по значениям полюсов образуют векторное линейное пространство размерности $\dim(L(G) = \nu_\ell := \{(i, j) \in N^2 : \rho_i + \rho_j = \rho_{\ell+1}\}$.

3. Ключевой параметр хеш функции $h_{P_j}(m)$ определяется вычислением в точке алгебраической кривой.

ОЦЕНКИ СЛОЖНОСТИ УНИВЕРСАЛЬНОГО ХЕШИРОВАНИЯ ПО АЛГЕБРАИЧЕСКИМ КРИВЫМ.

Замечание 7.

1. Сложность универсального хеширования по алгебраическим кривым определяется размерностью функционального поля ассоциированного с кривой.

2. Вычисление хеш значений $h_{P_j}(m)$ определяется многопараметрическим скалярным произведением рациональных функций алгебраических кривых со словами сообщения. В конструкции с прямым вычислением $h_{P_j}(m)$ требуется m (m - размерность поля рациональных функций) умножений в конечном поле для одного значения суммы $h_{P_j}(m)$ с предварительным вычислением значения рациональной функции в точке кривой.

Алгоритмы вычислений хеш функций по максимальным кривым Эрмита, Судзуки и кривым с большим числом точек Ферма и оценки сложности хеширования представлены в табл. 2.

Алгоритмы вычисления хеш функций $h_{x,y}(m)$ по алгебраическим кривым построены с использованием вычислений по многопараметрической схеме Горнера, допускают такое же распараллеливание, как в случае полиномиального хеширования и итерационные вычисления хеш-функции $y \leftarrow xy + m \bmod q$. Универсальное хеширование по кривой Судзуки является 4-х параметрическим, но сложнее только в два раза по сравнению с хешированием по плоским кривым. На каждом этапе хеш вычислений возможна дополнительная параметризация для грид вычислений со сложностью $k/p + p$ для GPU-процессора Двух шаговое применение грид технологии для хеширование по плоским кривым (кривым Эрмита, Ферма) приводит к уменьшению сложности $\approx 2(\sqrt{k}/p + p)$ раз.

Таблица 2

Алгоритмы вычислений хеш функций по максимальным кривым

Уравнение кривой	Определение $h_{x,y}(m)$	Оценки сложности хеш вычислений
Проективная прямая $X+Y+Z=0$, F_q	$\sum_{i=0}^{k-1} m_i \cdot x^i$	k
Кривая Эрмита $y^q + y = x^{q+1}$, F_{q^2}	$\sum_{j=0}^{s} y^j \cdot \sum_{i=0}^{s-j} m_{i,j} \cdot x^i$	$k+s$
Максимальные кривые $y^q + y = x^d$, F_{q^2}, $d \mid q+1$	$\sum_{i=0}^{s_1-1} x^i \cdot \sum_{j=0}^{m(s_1-1-i)+t+ind} m_{i,j} \cdot y^j$, $t = \lfloor (k - m(s_1-1)s_1/2)/s_1 \rfloor$, $m = (q+1)/d$, $ind = 0,-1$	$k+s_1$
Кривая Судзуки $y^q - y = x^{q_0}(x^q - x)$, F_q, $q = 2q_0^2$, $q_0 = 2^s$	$\sum_{t=0}^{1} y^t \sum_{i=0}^{s-t} v^s \sum_{r=0}^{\min\{s-t,q_0-t\}} (x/v)^r \sum_{j=0}^{\min\{s-r,q_0-1\}} m_{t,i,r,j}(w/v)^j$, $s = (3k)^{1/3}$	$k + s^3/3 + s^2/2 - 1$
Кривая Ферма $X^{(q-1)/3} + Y^{(q-1)/3} + Z^{(q-1)/3} = 0$, F_q, $q \equiv 1 (\bmod 3)$	$\sum_{j=0}^{s} y^j \cdot \sum_{i=0}^{s-j} m_{i,j} \cdot x^i$	$k+s$

$s = \left\lfloor (2k + 1/4)^{1/2} - 1/2 \right\rfloor$, $s_1 = \left\lfloor (2k/m + 1/4)^{1/2} - 1/2 \right\rfloor$, $\lceil \cdot \rceil$ – округление к большему целому числу, $\lfloor \cdot \rfloor$ – округление к меньшему целому числу.

Решение задачи быстрых вычислений в модулярной арифметике и арифметики над расширенным полем F_q является такой же как в случае полиномиального хеширования.

Выводы. Алгоритмы универсального хеширование допускают эффективное разбиение одно потокового процесса хеширования на много потоковые. Программно-аппаратная архитектура массивно-параллельных вычислений CUDA хорошо приспособлена к таким вычислениям позволяет эффективно увеличить скорость универсального хеширования. Практический выигрыш приближается к размеру грида GPU-процессора.

ЛИТЕРАТУРА

[1]. Бабенко Л.К. Ускорение вычислений дискретного логарифма с помощью технологии CUDA./ Л.К.Бабенко, И.Д. Сидоров, А.С. Кириллов // Материалы XI Международной научно-

практической конференции «Информационная безопасность». Ч.3.Таганрог: Изд-во ТТИ ЮФУ, -2010. -С. 58-63.

[2]. Красилов А. А. Использование технологии CUDA для неграфических вычислений на GPU / А.А.Красилов // Информационные технологии и системы 2013 (ИТС 2013) : материалы международной научной конференции, БГУИР, Минск, Беларусь, 23 октября 2013.– С. 276-277.

[3]. Lei Xu. ECDLP on GPU. [Электронный ресурс] / Lei Xu, Dongdai Lin, Jing Zou //IACR Cryptology ePrint Archive -2011, P-146.

[4]. Буркатовская Ю.Б.. Быстродействующие алгоритмы деления полиномов в арифметике по модулю два/ Ю.Б.Буркатовская, А.Н.Мальчуков, А.Н.Осокин //Известия Томского политехнического университета, 2006. - Т.309. - № 1. - С. 19-24.

[5]. Lee V. W. et al. Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU //ACM SIGARCH Computer Architecture News. – ACM, -2010. – Т. 38. – №. 3. – С. 451-460.

[6]. Булавинцев В. Г.. О GPU реализации ограниченной версии нехронологического алгоритма DPLL / В. Г.Булавинцев, А.А Семенов // ПДМ. Приложение - 2013. -Вып. 6. –С. 111–112.

[7]. Результаты представленные на электронном ресурсе для вычислений с использованием CUDA [Эл. ресурс] http://hashcat. Net /hashcat/

[8]. Marc Stevens. New collision attacks on SHA-1 based on optimal joint local-collision analysis / Marc Stevens // EUROCRYPT 2013, Lecture Notes in Computer Science, vol. 7881, Springer, -2013. - P. 245-261.

[9]. P. Morawiecki./Parallel authenticated encryption with the duplex construction [Электронный ресурс] // P. Morawiecki, J. Pieprzyk, Cryptology ePrint Archive: Report 2013/658.

[10]. Carter J. L. Universal classes of hash functions / J. L.Carter, M.N.Wegman // Journal of Computer and Systems Science. -1979. - V.18. -P.143-154

[11]. Bierbrauer J. On families of hash functions via geometric codes and concatenation. / J.Bierbrauer, T.Johansson, G.Kabatianskii, B.Smeets //Advances in Cryptology-CRYPTO '93 Proceedings, Springer-Verlag.-1994.- P. 331-342.

[12]. Carter J. L. Universal classes of hash functions /J. L.Carter, M.N.Wegman // Journal of Computer and Systems Science. -1979. - V.18. -P.143-154.

[13]. Wegman. M. N. New hash functions and their use in authentication and set equality / M.N.Wegman, J.L.Carter // Journal of Computer and Systems Science.– 1981. – V. 22. – P. 265–279.

[14]. Беспалов Д. В. Использование графических ускорителей в решении задач криптоанализа / Д.В.Беспалов, В.Г.Булавинцев, А.А Семенов // Прикладная дискретная математика. Приложение. - 2010. -№ 3. -С. 86–87.

[15]. Желтов С.А. Некоторые оценки эффективности параллельных вычислений в архитектуре CUDA при решении задачи факторизации целых чисел / С.А.Желтов // Тр. IV Междунар. конгресса по интеллект. системам и информ. технол. // XII Междунар. научн.-техн. конф. «Интеллектуальные системы» (AIS'12). – М.: Физматлит, -2012. – Т. 2. – С.191-196.

[16]. Халимов Г.З. Универсальное хеширование по максимальным кривым Гурвица / Г.З. Халимов // Журнал "Прикладная радиоэлектроника". Харьков: ХНУРЭ. - 2010. - Т.9. -№ 3. - С.365-370.

REFERENCES

[1]. Babenko LK Accelerating the computation of the discrete logarithm with technology CUDA. / L.K.Babenko, ID Sidorov, AS Kirillov // Proceedings of the XI International scientific-practical conference "Information Security". Ch.3.Taganrog Univ Tsure, -2010. -FROM. P.58-63.

[2]. Krasilov AA Using CUDA technology for non-graphics computing on GPU / A.A.Krasilov // Information Technologies and Systems 2013 (ITS 2013): Proceedings of the International Conference, BSUIR, Minsk, October 23, 2013.- With . 276-277.

[3]. Lei Xu. ECDLP on GPU. [Electronic resource] / Lei Xu, Dongdai Lin, Jing Zou // IACR Cryptology ePrint Archive -2011, P-146.

[4]. Burkatovsky JB .. Fast algorithms polynomial division in arithmetic modulo two / Yu.B.Burkatovskaya, A.N.Malchukov, A.N.Osokin // Bulletin of the Tomsk Polytechnic University, 2006., T.309, № 1, P. 19-24.

[5]. Lee V. W. et al. Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU // ACM SIGARCH Computer Architecture News. - ACM, -2010. - T. 38. - №. 3. - P. 451-460.

[6]. Bulavintsev VG .. About GPU implementation of a limited version nehronologicheskogo algorithm DPLL / V. G.Bulavintsev, AA Semenov // PDM. Appendix - 2013, vol. 6. P. 111-112.

[7]. The results are shown on electronic resources for computing with CUDA [electronic resource] http://hashcat.net/hashcat/

[8]. Marc Stevens. New collision attacks on SHA-1 based on optimal joint local-collision analysis / Marc Stevens // EUROCRYPT 2013, Lecture Notes in Computer Science, vol. 7881, Springer, -2013. - P. 245-261.

[9]. P. Morawiecki. / Parallel authenticated encryption with the duplex construction [Electronic resource] // P. Morawiecki, J. Pieprzyk, Cryptology ePrint Archive: Report 2013/658.

[10]. Carter JL Universal classes of hash functions / JLCarter, MNWegman // Journal of Computer and Systems Science. -1979. - V.18. -P.143-154

[11]. Bierbrauer J. On families of hash functions via geometric codes and concatenation. / J.Bierbrauer, T.Johansson, G.Kabatianskii, B.Smeets // Advances in Cryptology-CRYPTO '93 Proceedings, Springer-Verlag.-1994.- P. 331-342.

[12]. Carter JL Universal classes of hash functions / J. L.Carter, MNWegman // Journal of Computer and Systems Science. -1979. - V.18. -P.143-154.
[13]. Wegman. MN New hash functions and their use in authentication and set equality / MNWegman, JLCarter // Journal of Computer and Systems Science.- 1981. - V. 22. - P. 265-279.
[14]. Bespalov DV Using graphics accelerators in the task of cryptanalysis / D.V.Bespalov, V.G.Bulavintsev, AA Semenov // Applied discrete mathematics. Application. -2010. -№ 3. -P. 86-87.
[15]. Zheltov SA Some estimates of the efficiency of parallel computing CUDA architecture to solve the problem of factoring integers / S.A.Zheltov // Tr. IV Mezhdunar. Congress on intelligence. systems and inform. tehnol. // XII Intern. nauchn. techno. Conf. "Intelligent systems» (AIS'12). - M .: FIZMATLIT, -2012. - T. 2. -P.191-196.
[16]. Halimov GZ Universal hashing for maximum Hurwitz curves / GZ Halimov // Journal "Applied electronics." Kharkov: KNURE. - 2010. - v.9. -№ 3. - P.365-370.

ШВИДКІСНЕ УНИВЕРСАЛЬНЕ ХЕШУВАННІ НА ОСНОВІ БАГАТО ПОТОКОВИХ ОБЧИСЛЕНЬ

Універсальне хешування визначає доказово стійку аутентифікацію з лічильником, забезпечує високу стійкість до колізій і швидкість обчислень. Одним з найбільш перспективних напрямків у вирішенні завдань високошвидкісних обчислень є використання технології GPGPU (General-purpose graphics processing units). Технологія GPGPU дозволяє на одному обчислювачі досягати високого рівня паралелізму без тимчасових витрат на передачу даних між вузлами і синхронізацію результатів обчислень. Для підвищення швидкості універсального хеширования на основі скалярних і поліноміальних обчислень розроблені пропозиції щодо багато потоковим обчисленням, обчисленням в модулярной арифметиці та арифметики над розширеним кінцевим полем з використанням GPGPU технології.

Ключові слова: універсальне хешування, багато потокові обчислення, архітектура масивно-паралельних обчислень CUDA.

HIGH SPEED UNIVERSAL HASHING BASED ON MANY STREAM COMPUTING

Universal hashing determines provably resistant authentication with the meter, provides high resistance to collisions and computing speed. One of the most promising areas in addressing the challenges of high-speed computing is the use of technology GPGPU (General-purpose graphics processing units). GPGPU technology enables one solver to achieve a high level of parallelism without the time-consuming data transfer and synchronization between the nodes of the calculation results. To enhance the universal hash-based scalar polynomial computations and developed proposals for a lot of stream computing, calculations in modular arithmetic and arithmetic over the extended finite field with GPGPU technology.

Index terms: universal hashing, a lot of stream computing architecture massively parallel computing CUDA.

Котух Євген Володимирович, аспірант кафедри безпеки інформаційних технологій, Харківський національний університет радіоелектроніки.
E-mail: yevgenkotukh@gmail.com.
Котух Евгений Владимирович, аспирант кафедры безопасности информационных технологий, Харьковский национальный университет радиоэлектроники.
Kotukh Yevgen, postgraduate student,Department of Information Technology Security, Kharkiv National University of Radio Electronics.

Карташов Володимир Михайлович, доктор технічних наук, завідувач кафедри радіоелектронних систем, Харківський національний університет радіоелектроніки.
E-mail: kartashov@kture.kharkov.ua.
Карташов Владимир Михайлович, доктор технических наук, заведующий кафедры радиоэлектронных систем, Харьковский национальный университет радиоэлектроники.
Kartashov Vladimir, Doctor of Engineering, Head of electronic systems department, Kharkiv National University of Radio Electronics.

Цапко Денис Петрович, аспірант кафедри безпеки інформаційних технологій, Харківський національний університет радіоелектроніки.
E-mail: denys.tsapko@gmail.com.
Цапко Денис Петрович, аспирант кафедры безопасности информационных технологий, Харьковский национальный университет радиоэлектроники.
Denis Tsapko, postgraduate student,Department of Information Technology Security, Kharkiv National University of Radio Electronics.

Халімов Олег Геннадійович, аспірант кафедри безпеки інформаційних технологій, Харківський національний університет радіоелектроніки.
E-mail: o.khalimov@mail.ru.
Халимов Олег Геннадьевич, аспирант кафедры безопасности информационных технологий, Харьковский национальный университет радиоэлектроники.
Khalimov Oleg, postgraduate student,Department of Information Technology Security, Kharkiv National University of Radio Electronics.

Самойлова Аліна Вадимівна, аспірант кафедри безпеки інформаційних технологій, Харківський національний університет радіоелектроніки.
E-mail: hitori26@mail.ru.
Самойлова Алина Вадимовна, аспирант кафедры безопасности информационных технологий, Харьковский национальный университет радиоэлектроники.
Samoilova Alina, postgraduate student,Department of Information Technology Security, Kharkiv National University of Radio Electronics.