С.С. Федоров, Ю.А. Жомов, О.В. Ниссенбаум, И.Р. Зулькарнеев

Тюменский государственный университет, г. Тюмень
УДК 519.256, 004.056

ПРИМЕНЕНИЕ ФИЛЬТРОВ БЛУМА И COUNT-MIN SKETCH ДЛЯ ХРАНЕНИЯ СТАТИСТИКИ СТОХАСТИЧЕСКОГО ПРОТОКОЛА

Аннотация. В статье представлено описание результатов анализа применения фильтров Блума и Count-min sketch в стохастических протоколах анализа встречаемости паролей, как структур для хранения собираемой статистики.

Ключевые слова: стохастические протоколы, фильтр Блума, Count-min sketch, пароли, вероятностное хранение, хеш-функции.

Введение. На данный момент существуют протоколы, которые способны анализировать встречаемость паролей, при этом не осуществляя прямую их передачу и хранение. Цель таких протоколов, это безопасное обеспечение, контроля над распространением паролей. Один из таких протоколов, описанный в статье [1], обеспечивает высокую конфиденциальность при сборе паролей, но при практической реализации, могут возникнуть проблемы, для того чтобы это понять, рассмотрим кратко протокол далее.

Стохастический анонимный алгоритм. Рассматриваемый стохастический анонимный алгоритм предложен в работе [1] является частью специального протокола и применяется в нем для обеспечения конфиденциальности субъекта, пароль которого добавляется в систему. Схема протокола:

1. Клиент j отображает свой пароль ($\mathrm{pass_j}$) к секретному ℓ - битному значению $\mathrm{V_j} = \mathrm{H}(\mathrm{pass_j})$.

2. Клиент получает от сервера равномерно распределенное случайное ℓ-битное значение r_j. Устройство отправляет обратно однобитное значение скалярного произведения v_j и r_j в GF[2], обозначенное как $\langle v_j, r_j \rangle$.

3. Сервер хранит таблицу T[x] из 2^{ℓ} счетчиков, соответствующих всем возможным ℓ-битным значениям x (значения счетчиков инициализируется нулем при первоначальной настройке системы).

4. Для каждого значения x, если $\langle x, r_j \rangle = \langle v_j, r_j \rangle$ соответствующий счетчик увеличивается на единицу, в противном случае он уменьшается на единицу [1].

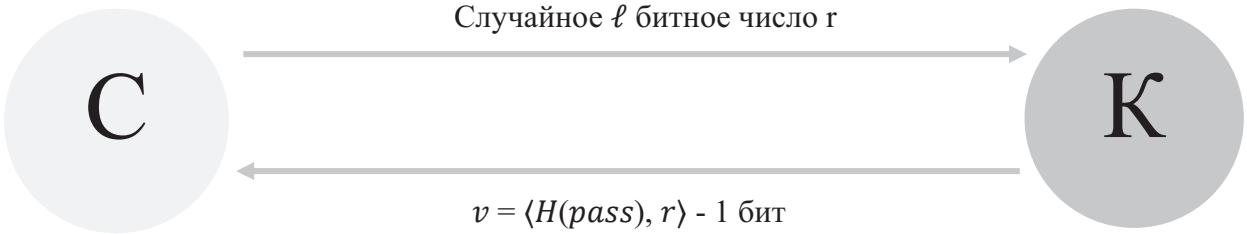


Рис. 1. Схема протокола.

Шаг 4 – наиболее важен в контексте применения структур хранения данных, и потому стоит еще раз отметить, что добавление одного пароля, провоцирует обновление значений абсолютно всех счетчиков, то есть их увеличение или уменьшение с вероятностью 0.5 для всех значений не являющихся реально добавляемым паролем, и с увеличение вероятностью 1 для счетчика отвечающего за добавляемый пароль. Благодаря этому, при многократном добавлении одного и того же пароля, значение его счетчика будет постоянно увеличиваться, в отличии от прочих.

Для обеспечения защиты от подмены отправляемого значения клиентом, в итоговом протоколе, применяется специальный алгоритм основанный на кодировании бит в квадраты и псевдо квадраты, однако

общая схема не меняется. Протокол по данной схеме должен выполняться каждый раз при добавлении нового пароля.

Проблема хранения данных в протоколе. Проблема заключается в хранении сервером таблицы счетчиков T, так как ее размер фактически составит 2^{ℓ} * n байт, где ℓ - размер используемой хеш-функции, n – количество байт выделяемой памяти под значение одного счетчика. Таким образом даже уже при $\ell = 32$ и $n = 4$ – размер таблицы уже превышает 16 гигабайт, и при этом каждое добавление пароля потребует перерасчет всех этих данных.

Для того чтобы уменьшить таблицу до N ($N < \ell$), необходимо чтобы все значения из 2^{ℓ} могли быть отображены в 2^{N} – то есть необходимо применения хэш-функции. Однако, использование хэш-функции с меньшей таблицей приведет к возникновению коллизий. Для того чтобы уменьшить влияние коллизий на итоговые значения счетчиков было решено рассмотреть такие структуры хранения данных как «фильтр Блума» и «Count-min sketch».

Фильтры Блума. Фильтр Блума представляет собой вероятностную структуру данных. В такой структуре информация хранится в массиве из n ячеек, которые изначально равны нулю. Перед применением фильтра задаются h независимых хэш-функций (на рисунке обозначены как x, y и z), размер результата этих хэш-функций равен размеру фильтра. При добавлении элемента в фильтр происходит последовательное вычисление хэш-значений от него. Вычисленные хэш значения являются индексами для массива фильтра [2; 116].

В результате эти хэш-функции отображают элементы в фильтр. Хэш-функции также вычисляются для того, чтобы получить значение элемента. Таким образом фильтр Блума позволяет обходится меньшим объемом

памяти при хранении, но с возможностью появления ложноположительных ошибок.

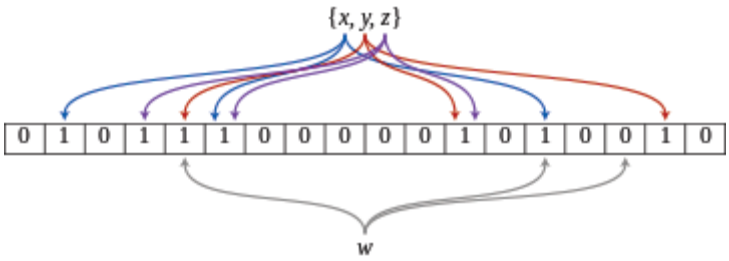


Рис. 2. Отображение в фильтре Блума.

Задача о мячах и корзинах. Для того чтобы иметь возможность анализировать распределение паролей в таблице, необходимо рассмотреть задачу о мячах и корзинах. В задаче распределения мечей по корзинам есть n мячей и m корзин. Каждый мяч попадает в одну случайную корзину независимо от других. При этом распределение мячей в корзине будет равно биномиальному распределению. Параметрами биномиального распределения являются количество испытаний и вероятность успеха. Соответственно в задаче эти параметры будут равны n и 1/m [2; 99].

Биномиальное распределение при достаточно большом числе испытаний m и низкой вероятности успеха p может быть использовано приближение распределением Пуассона [2; 101]:

$$n \to \infty, p \to 0$$

$$\frac{n!}{x!\,(n-x)!}p^x(1-p)^{n-x} \to \frac{e^{-\lambda}\lambda^x}{x!}$$

Параметром такого распределения Пуассона будет λ=np. И соответственно в задаче мячей и корзин параметр будет равен $\lambda = \frac{n}{m}$:

$$P_r = (X = r) = \frac{e^{-\frac{n}{m}}(\frac{n}{m})^r}{r!}$$

Связь задачи о мячах и корзинах с фильтром Блума. Определение вероятности аналогично задаче о мячах и корзинах. Количество корзин — это размер фильтра Блума, которое равно размеру результата хэш-функции. Количество мячей — это количество добавляемых значений m умноженное на количество хэш функций h, так как хэш-функции выдают все значения с равной вероятностью.

$$P_r = (X = r) = \frac{e^{-\mu}\mu^r}{r!}, \text{ где } \mu = \frac{mh}{n}$$

Применение фильтра Блума для хранения информации о паролях. Теперь рассчитаем μ при добавлении s штук R-битных паролей, с вероятностью добавления в следствии случайного совпадения результатов скалярного произведения p, с h штук n-битными хэш функциями:

$$\mu = sph(2^{R-n} - 2^{-n})$$

Для счетчиков, которые никогда не содержат значения реально добавляемых паролей, распределение инкремента и декремента будет соответствовать распределению Пуассона, которое приведено выше, и будет отличаться только лишь параметром μ. Однако для счетчиков содержащих паролей значения инкремента и декремента будут распределены по смещенному распределению Пуассона:

$$P_r = (X = r) = \frac{e^{-\mu}\mu^{(r-s)}}{(r-s)!}$$

Соответственно значения математического ожидания и дисперсии, при добавлении одного пароля s раз, будут равны:

$$E_{\text{инкПароль}} = sph(2^{R-n} - 2^{-n}) + s$$

$$E_{\text{декПароль}} = sph(2^{R-n} - 2^{-n}) - s$$

$$E_{\text{инкШум}} = E_{\text{декШум}} = sph(2^{R-n} - 2^{-n})$$

$$D = (2^{R-n} - 2^{-n})$$

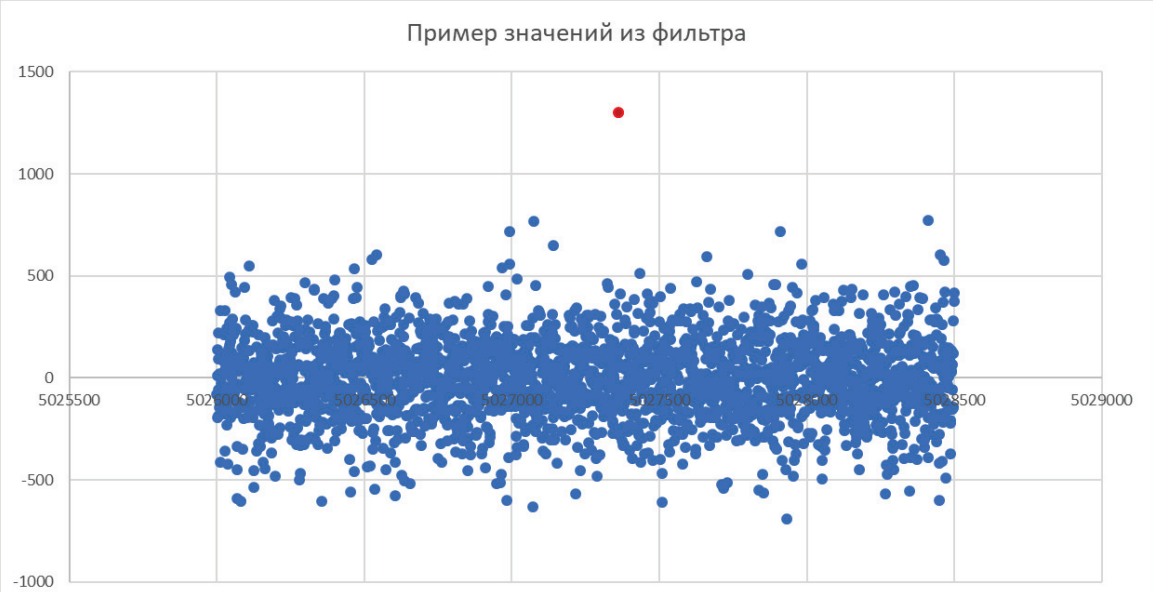
Дисперсия приведена только для одного добавления. Так же стоит отметить, что при добавлении разных паролей, необходимо учитывать и их количество. Так как распределения вычитаются, то итоговые значения математического ожидания и дисперсии будут таковы:

$$E_{\text{итогПароль}} = 2s$$

$$E_{\text{итогШум}} = 0$$

$$D = 2shp(2^{R-n} - 2^{-n})$$

Проведя эксперименты, данные значения подтвердились, однако стоит отметить наличие пока что не вычисленной зависимости между паролями. На основе полученных результатов были составлены графики, параметры системы при расчетах: размер хэша пароля был 2^{24}, размер фильтра Блума 2^{23}:



По оси X - номер ячейки фильтра
По оси Y - значение ячейки, после добавления 2000 случайных паролей и 1200 одинаковых

Рис. 3. Явное выделение добавленного пароля, фрагмент выборки.

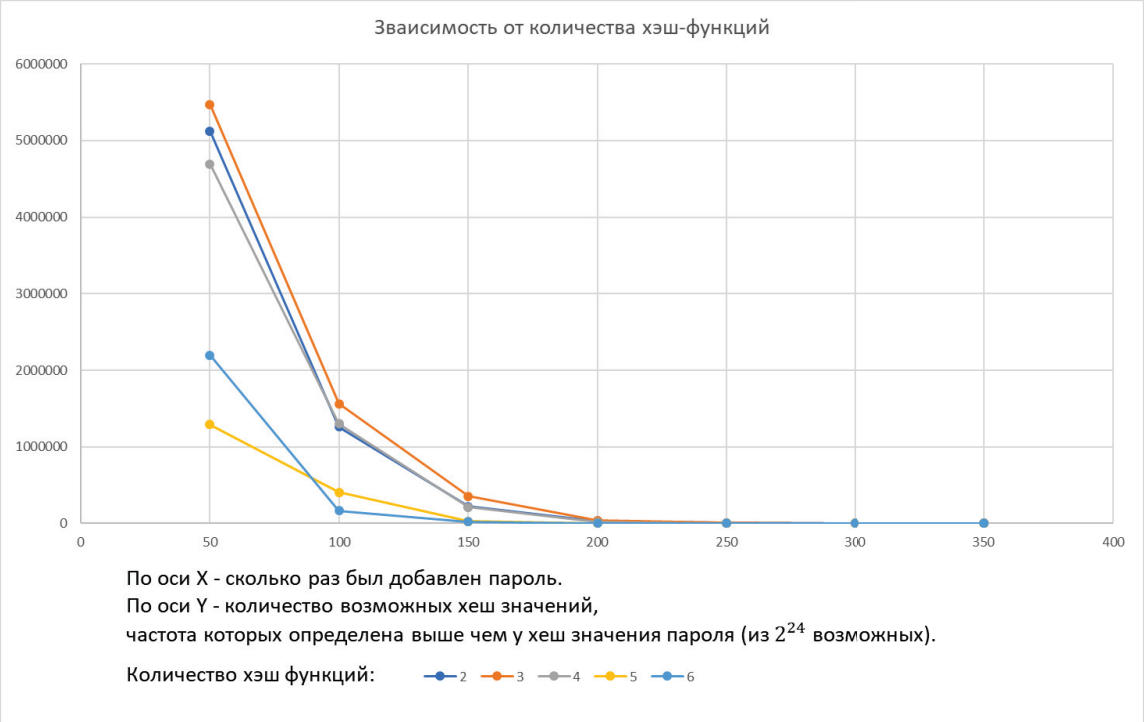


Рис. 4. Влияние количества хэш функций на выявление пароля.

Count-min sketch. Структура данных была разработана в 2003 году, она развивает применение фильтров Блума для оценки встречаемости событий. Эта структура состоит из набора $\{h_1, h_2 \dots h_d\}$ хеш-функций и таблиц счетчиков, где за каждой из хеш-функций закреплена одна таблица. Хеш-функции отображаются входные значения в номера счетчиков из соответствующей таблицы [3].

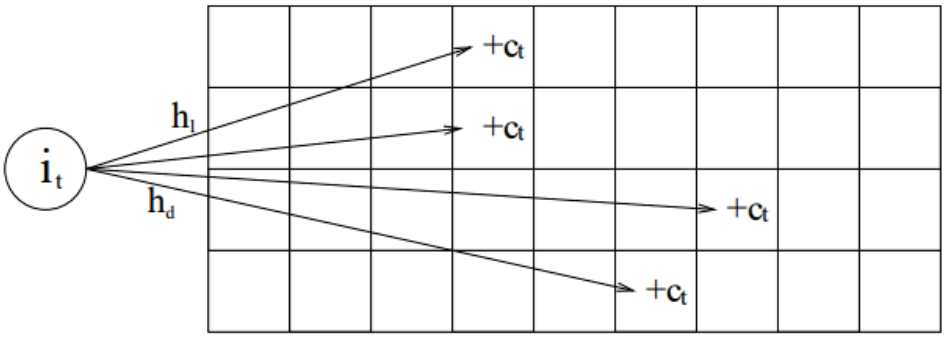


Рис. 5. Отображение в Count-min sketch.

На рис. 5 i_t – некое добавляемое значение, в нашем случае это хэш от пароля, $+c_t$ это оказываемое влияние, то есть ± 1. Итоговое значение определяется по минимальному из значений счетчиков:

$$count_i = \min_j(count[j, h_j(i)]).$$

Стоит отметить, что для каждого $count_i$ существует реальное значение a_i, такое, что $count_i \geq a_i$. Также, это гарантирует, что $count_i \leq a_i + \varepsilon N$ с вероятностью $1 - \delta$, где $N = \sum_{i=0}^{n} a_i$ – количество добавляемых значений. Основными параметрами такой структуры являются количество хеш-функций d и размеры таблиц - w. Увеличение диапазона значений хеш-функций повышает точность итоговой оценки встречаемости, а увеличение количества хеш-функций снижает вероятность неверной оценки. Параметры w и d могут быть выбраны таким образом: $w = \lceil e / \varepsilon \rceil$ и $d = \lceil \ln 1 / \delta \rceil$, где ошибка в ответе на запрос находится в пределах аддитивного коэффициента ε с вероятностью $1 - \delta$ [4].

Реализовав хранение счетчиков в данной структуре, были проведены эксперименты, которые позволили получить графическое представление эффективности данной структуры, при w равно 2^{20} и различных параметрах d:

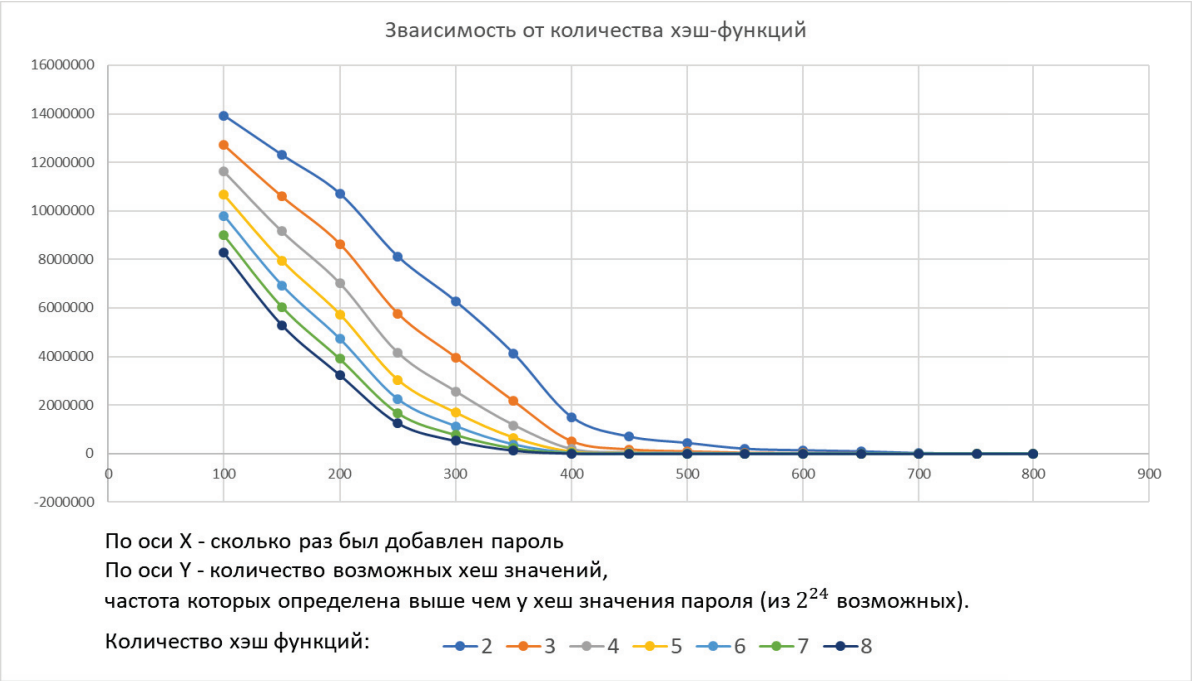


Рис. 6. Чувствительность в Count-min sketch.

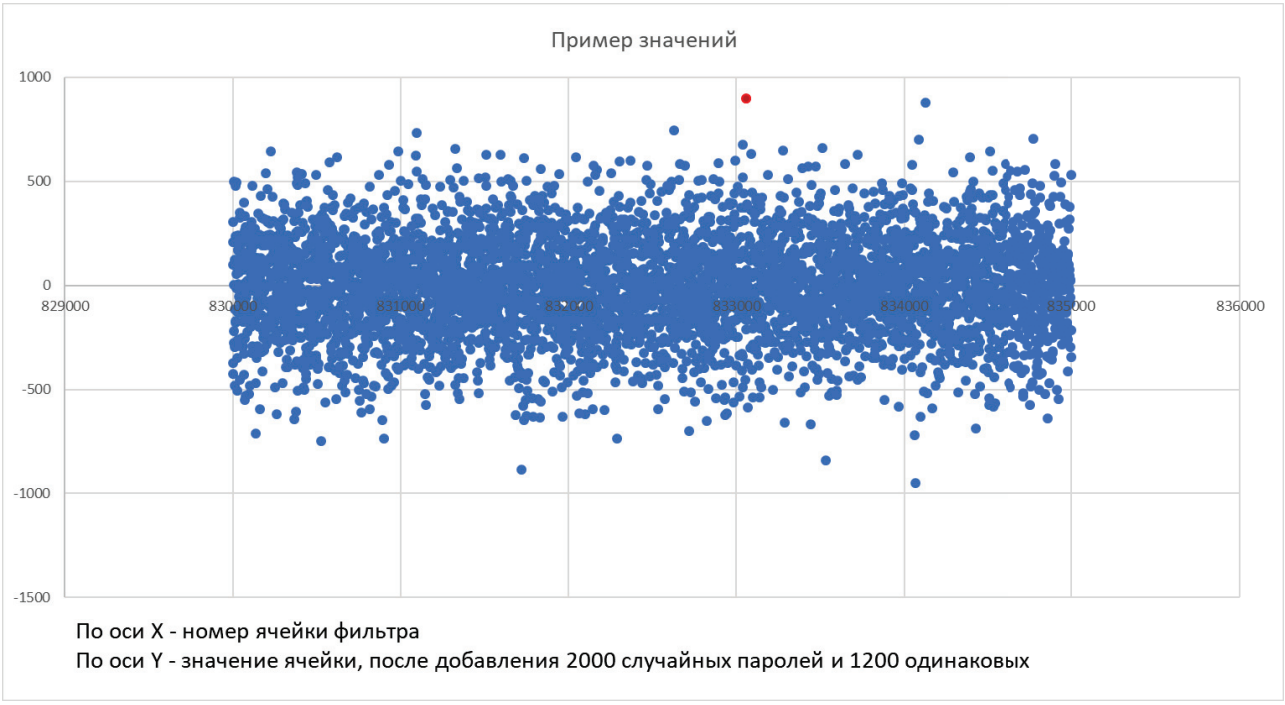


Рис. 7. Выделение счётчика пароля среди шумов в Count-min sketch.

Направление дальнейших исследований. Проведя данную работу, были выявлены некоторые зависимости между паролями и хеш-функциями, которые требуют дальнейшего изучения, в частности

получение более точных формул для математического ожидания и дисперсии в фильтрах Блума, характеристики распределения, коррелированность счетчиков, вероятности ошибок, границы чувствительности, более подробное изучение Count-min sketch и других структур данных, к примеру PaXoS [5], которые бы могли применяться для компактного хранение статистики.

Результаты и выводы. Подводя итог работы, можно сказать, что применение фильтров Блума или Count-min sketch для хранения статистики и выявления наиболее часто встречаемых паролей возможно, однако требуется провести дальнейшие исследования для точного определения чувствительности при различных параметрах системы. Приведем также график чувствительности к паролю этих структур:

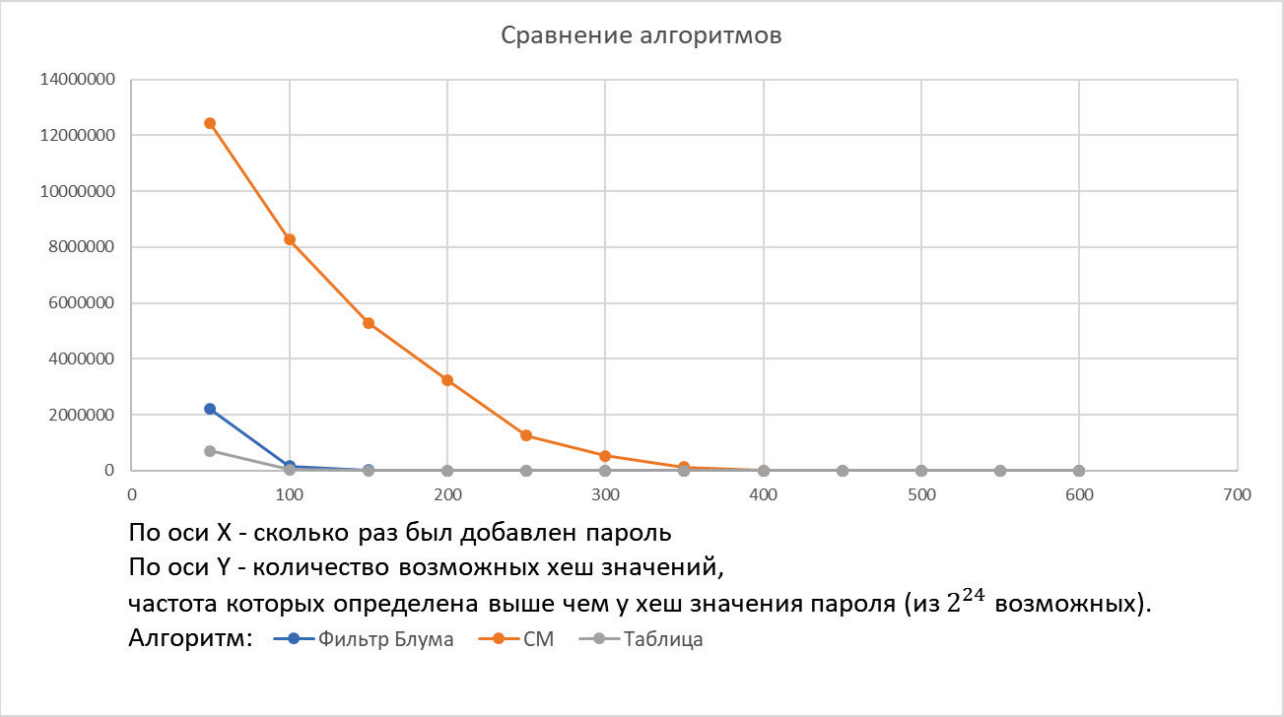


Рис. 8. График выделения пароля в разных структурах.

СПИСОК ЛИТЕРАТУРЫ

1. Moni Naor, Benny Pinkas, Eyal Ronen. How to (not) share a password [Электронный ресурс]: Privacy preserving protocols for finding heavy hitters with adversarial behavior. Real World Crypto, 2019. - URL: https://eprint.iacr.org/2018/003.pdf

2. Mitzenmacher M., Upfal E. Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis. – Cambridge university press, Sheridan Books, 2017. 467 с.

3. Graham Cormode, AT&T Labs–Research [Электронный ресурс]: Count-Min Sketch - URL: http://dimacs.rutgers.edu/~graham/pubs/papers/cmencyc.pdf

4. Graham Cormode, S. Muthukrishnan An Improved Data Stream Summary [Электронный ресурс]: The Count-Min Sketch and its Applications – URL: https://sites.google.com/site/countminsketch/cm-latin.pdf

5. Pinkas B. et al. PSI from PaXoS: Fast, Malicious Private Set Intersection //Annual International Conference on the Theory and Applications of Cryptographic Techniques. – Springer, Cham, 2020. – С. 739-767.