Т2. №2(34)’2024 https://doi.org/10.31474/2786-9024/v2i2(34).313785



# ОЦІНКА АПАРАТНИХ ВИТРАТ ДЛЯ ФІЛЬТРА БЛУМА З ЛІЧИЛЬНИКАМИ В СИСТЕМАХ ЗАХИСТУ ІНФОРМАЦІЇ НА ПЛІС

**С.Я. Гільгурт**[1]

[1] Department of Mathematical and Econometric Modeling, Georgy Pukhov Institute for Modelling in Energy Engineering of the National Academy of Sciences of Ukraine, Kyiv, Ukraine

E-mail: hilgurt@ukr.net

Отримано 21.10.2024
Прийнято до публікації 25.10.2024
Опубліковано 01.11.2024

**АНОТАЦІЯ**

*Мета роботи полягає в дослідженні рішень та шляхів практичної реалізації щодо організації динамічної зміни складу патернів, які розпізнаються фільтром Блума в апаратних (на базі ПЛІС) сигнатурних системах захисту інформації (в тому числі критичної інфраструктури), таких як мережеві системи виявлення вторгнень, антивіруси, спам-фільтри тощо. Під зміною складу мається на увазі як додавання, так і вилучання окремих елементів з переліку патернів, які відшукує сигнатурна система в потоці вхідних даних, зокрема в тілах пакетів мережевого трафіку.*

*Для досягнення мети в роботі проаналізовані принципи побудови та функціонування фільтра Блума, його переваги та недоліки порівняно з іншими схемами розпізнавання в термінах відомих показників ефективності. Розглянуті модифікації та різновиди, запропоновані розробниками з метою покращення його характеристик протягом всього періоду використання в системах кіберзахисту комп'ютерних мереж. Проаналізовано особливості апаратної реалізації фільтра Блума на ПЛІС.*

*З двох виявлених підходів до вирішення проблеми динамічного переналаштування в роботі розглянутий той, що забезпечує більшу швидкодію за рахунок зміни апаратної структури пристрою, а саме фільтр Блума з лічильниками. На прикладі однієї з модифікацій апаратної схеми, так званого спрощеного фільтра Блума, розглянуто можливий варіант побудови його цифрової структури. Для підвищення ефективності процесу розробки систем захисту інформації з використанням даної схеми сформована функція оцінки апаратних ресурсів, яка дозволяє знаходити кількісні характеристики витратна синтез цифрових пристроїв у ПЛІС без виконання витратної процедури повної компіляції проєкту. Проведено попереднє порівняння отриманого виразу з функцією оцінки для схеми спрощеного фільтра Блума без лічильників.*

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

***Ключові слова**: МСВВ, множинне розпізнавання патернів, ПЛІС, фільтр Блума з лічильниками, апаратні витрати.*

## ВСТУП

У 1970 році Бёртоном Говардам Блумом була запропонована схема обробки даних, заснована на використанні геш-функцій [1]. Сьогодні вона відома як фільтр Блума (ФБ), англомовна назва – Bloom Filter (BF). Її метою було забезпечити більш компактне зберігання даних в застосуваннях, коли є прийнятною певна частка хибних спрацювань. Винахід виявилася вдалим и почав широко використовуватися спочатку в сфері традиційного застосування гешування, а згодом і в суміжних технічних галузях.

Оскільки задачі обробки даних, де використовувався ФБ, пов'язані з інтенсивною обробкою інформації, виникла потреба у прискоренні операцій, які він здійснює. Ефективною основою для апаратного прискорення виявилися пристрої на базі програмованих логічних інтегральних схем (ПЛІС) типу Field-programmable gate array (FPGA). Почалося використання програмованої логіки для синтезу ФБ ще в середині 1990-х років. Історично першим застосуванням програмованої логіки для побудови цієї схеми обробки даних з метою швидкого розпізнавання тексту де-факто стала робота [2], хоча власне назва "фільтр Блума" в ній не згадувалася. Згодом подібні розробки почали з'являтися все частіше. Причому більша їх частка була присвячена мережевим застосуванням.

Починаючі с середини 2000-х, фільтр Блума активно використовується в галузі технічного захисту інформації, зокрема в якості основного засобу розпізнавання при побудові мережевих систем виявлення вторгнень (МСВВ). При цьому в переважній більшості застосувань ФБ вирішує ресурсоємну задачу множинного розпізнавання патернів (pattern matching) – ключову обчислювальну задачу сигнатурних засобів захисту інформації [3]. Сутність цієї задачі полягає в одночасному порівнянні фрагменту вхідних даних (наприклад, кількох байтів тіла мережевого пакету) з великою кількістю патернів – фіксованих послідовностей символів, що входять до складу сигнатур – описів відомих атак. Такому застосуванню ФБ свого часу посприяла активна діяльність проф. Джона Вільяма Локвуда з Вашингтонського університету та його численні наукові публікації за даною темою [4]. Але й досі розробники мережевих засобів захисту інформації не втратили інтерес до фільтра Блума, використовуючи його, зокрема, як "перший ешелон" захисту [5]. Найбільш поширеною платформою для використання ПЛІС на сьогодні є реконфігуровні обчислювачі або прискорювачі, англомовна назва – Reconfigurable Accelerator (RA), які крім кристала ПЛІС містять бортовий оперативний запам'ятовуючий пристрій (ОЗП), контролер зв'язку з хостовою комп'ютерною системою, у випадку мережевих застосувань – Ethernet-порти та інші компоненти [6].

Нижче наведено короткий літературний огляд, в якому висвітлені основні властивості фільтра Блума, його переваги та недоліки порівняно з іншими схемами розпізнавання, які використовуються в системах захисту інформації, шляхи усунення недоліків та підвищення ефективності застосування ФБ. Також розглянуто специфіку реалізації цифрової схеми ФБ на ПЛІС, складнощі та проблеми, що виникають при цьому, та можливі рішення щодо їх подолання. Розглянуто критерії оцінки технічних показників ФБ с точки зору застосування для побудови систем кіберзахисту й метод прискореного обчислення їх кількісних характеристик. Сформульована технічна проблема, пов'язана з неможливістю класичного ФБ змінювати набір даних, що підлягають виявленню.

## АНАЛІЗ ЛІТЕРАТУРНИХ ДАНИХ ТА ПОСТАНОВКА ПРОБЛЕМИ

На сьогодні застосування ФБ охоплює велику кількість предметних галузей (Рис. 1) [7].

Результати поглибленого дослідження існуючої літератури з оптимізації ФБ наведено в публікації [8].

Багато прикладів використання ФБ в галузі захисту інформації можна знайти у вичерпному дослідженні [9], а також в більш сучасному огляді [10]. Проводяться дослідження також щодо застосування ФБ для вирішення проблем кіберзахисту об'єктів критичної інфраструктури, зокрема, SCADA-систем [11].

**Структура та принцип дії класичного фільтра Блума**. ФБ складається з двох ключових компонентів, як показано на Рис. 2: комплекту з $K$ блоків, що обчислюють геш-функції $h_1(x)$, $h_2(x)$, $h_3(x)$, …, $h_K(x)$ та масиву з $M$ бітових комірок (компонент Rg на рисунку) [12]. Назвемо цей масив регістром бітів (РБ). В початковому стані він заповнений нулями.

На етапі програмування або навчання (Рис. 2, а) на входи всіх геш-функцій послідовно подається кожен з $N$ елементів словнику патернів (довжиною $W$ бітів). Для кожного елемента обчислюються значення всіх $K$ геш-функцій, отримані значення яких інтерпретуються як



ISSN 2786-9024

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

номер (адреса) комірки у РБ. У відповідні комірки заносяться значення "1".

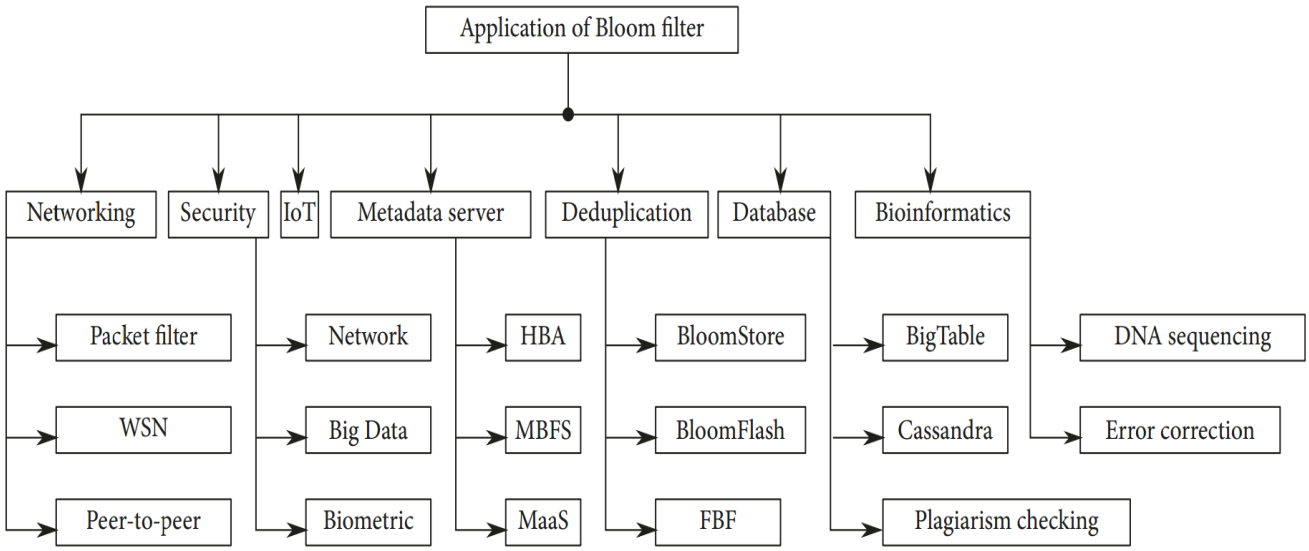


**Рис. 1. Галузі застосування фільтра Блума [7]**

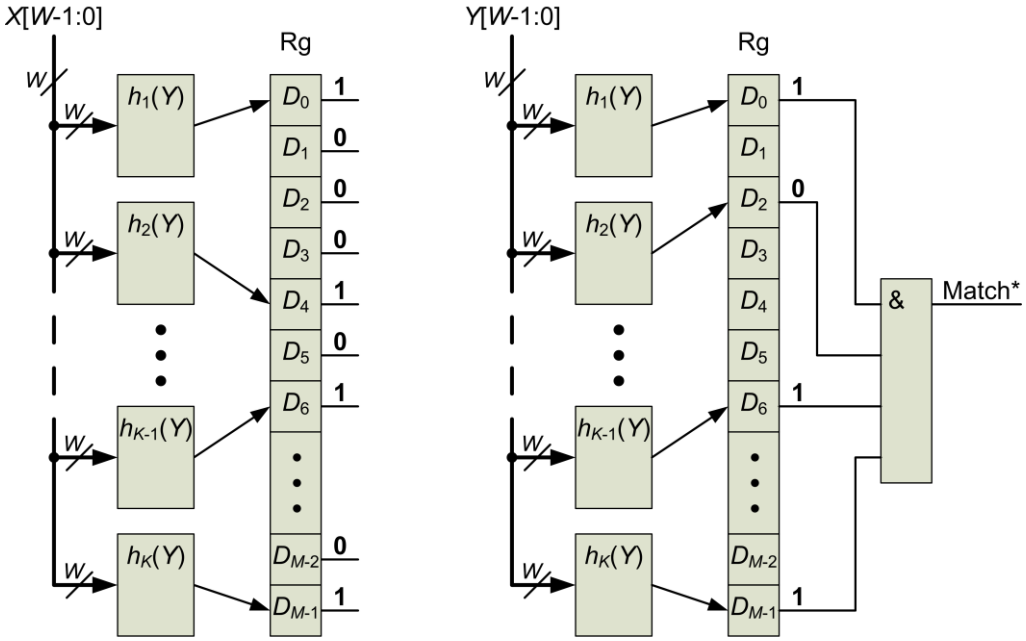


**Рис. 2. Функціонування фільтра Блума: а – в режимі програмування; б – в процесі розпізнавання**

В процесі функціонування фільтра Блума (Рис. 2, б) на його вхід подається фрагмент вхідної послідовності символів (також довжиною $W$ бітів). Для нього обчислюються значення всіх $K$ геш-функцій. По отриманим адресам здійснюється звернення до комірок РБ. Якщо у всіх позиціях, на які вкажуть геш-функції, містяться одиниці, вважається, що вхідна комбінація символів з певною вірогідністю співпадає з одним з патернів, що брали участь у програмуванні ФБ. Але якщо хоча б одна геш-функція вкаже на комірку з нульовим значенням, це гарантовано свідчить про відсутність

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

відповідної послідовності символів серед навчальної множини патернів.

Отже, ФБ функціонує з деякою вірогідністю помилки розпізнавання другого роду (false positive), але без помилок першого роду (false negative). Тобто інколи він може сигналізувати про наявність збігу, якого насправді немає, але ніколи не пропустить справжній збіг. Значення вірогідності помилки другого роду для заданих значень кількості патернів $N$, розміру $M$ регістра бітів, та кількості геш-функцій $K$ знаходиться згідно виразу [9]:

$$p_{пом.2} = \left(1 - \left(1 - \frac{1}{M}\right)^{KN}\right)^K \approx \left(1 - e^{-\frac{KN}{M}}\right)^K. \quad (1)$$

При цьому значення $M$ покладається досить великим, що відповідає дійсності.

Як можна бачити, збільшення кількості патернів призводить до збільшення системної помилки ФБ, в той час як збільшення розміру РБ – до зменшення помилки. Кількість геш-функцій $K$ ще більш суттєво впливає на вірогідність помилки розпізнавання другого роду. Доведено [9], що ця вірогідність мінімальна при певному співвідношенні значень $N$ та $M$:

$$K = \frac{M}{N} ln\, 2 \approx 0{,}6931 \frac{M}{N}. \quad (2)$$

При цьому вірогідність помилки

$$p_{пом.2} = 2^{-K} \approx 0{,}6185^{M/N}. \quad (3)$$

Оскільки геш-функції обчислюються цифровими схемами у двійковій системі числення, при забезпеченні даного співвідношення під час вибору параметрів ФБ розмір $M$ регістра бітів обирають, як правило, таким, що дорівнює ступеню двійки.

Аналізуючи наведені вище відомості щодо основ функціонування фільтра Блума можна зробити попередню оцінку його можливостей щодо розпізнавання.

По-перше, ФБ дозволяє дуже економно використовувати ресурси пам'яті. Розмір словнику патернів не впливає напряму на розмір РБ. Додання нових патернів до вже присутніх у цієї структурі даних призводить лише до підвищення вірогідності помилки розпізнавання, але не до збільшення об'єму запам'ятовуючого пристрою. По-друге, кількість та складність геш-функцій, які потрібно обчислювати в процесі розпізнавання, тобто, апаратурна витратність та продуктивність при даному підході теж не залежать від об'єму словника патернів. Нарешті, довжина патернів також не впливає напряму ні на кількість ресурсів пам'яті, ані на продуктивність. Отже ФБ добре масштабується по двох напрямках: за об'ємом словнику патернів та за довжиною патернів, що відшукуються. Остання властивість має важливе значення: навіть дуже довгі рядки після перетворення геш-функціями потребують для зберігання ті ж самі $K$ комірок РБ.

**Кластеризація**. На жаль, фільтру Блума притаманний важливий недолік (як будь-яким рішенням на базі геш-функцій): розмір вхідної послідовності символів, тобто довжина патерну має бути фіксованою для обраного набору геш-функції. Інакше кажучи, один ФБ здатен розпізнавати патерни тільки однакової довжини. Єдиною можливістю подолання даного недоліку є кластеризація, тобто ранжування патернів по довжіні та спільне застосування кількох ФБ, що розпізнають патерни різної довжини (див. Рис. 2. у [12]). Це призводить до збільшення витрат ресурсів, що значно погіршує вартівні показники ефективності ФБ. Але це погіршення не стосується швидкісних показників, про що свідчить, зокрема, той факт, що певна кількість успішних практичних розробок МСВВ на базі фільтра Блума фактично є системами запобігання вторгнень, для яких висуваються значно жорсткіші вимоги щодо продуктивності [4, 13, 14].

**Уточнення результатів**. Як було вказано вище, Фільтру Блума притаманні системні помилки розпізнавання другого роду. Тому результати розпізнавання потрібно уточнювати, тобто переконатися, чи насправді виявлена послідовність символів співпадає з патерном. Операція уточнення може бути виконана програмно на хост-комп'ютері, апаратними засобами із використанням частки ресурсів програмованої логіки [4] або за допомогою вбудованих в ПЛІС процесорних ядер універсальної архітектурі [13]. У двох останніх випадках цю операцію зазвичай виконують із застосуванням вторинної геш-таблиці, яка зберігається в бортовому ОЗП реконфігуровного обчислювача, зовнішньому по відношенню до мікросхеми ПЛІС.

Операція уточнення виконується значно повільніше аніж процедура розпізнавання. Це не тільки уповільнює роботу системи, але також підвищує вразливість від атак на МСВВ шляхом насичення мережевого трафіку пакетами, що імітують напад (тобто співпадають з патернами бази даних сигнатур МСВВ). Втім, вірогідність хибного спрацьовування ФБ можна зменшити до потрібного значення шляхом вибору належного співвідношення параметрів $K$ та $M$.

**Розпаралелювання**. Підвищити продуктивність фільтра Блума можливо за рахунок паралельного підключення із зсувом кількох однакових блоків, кожен з яких містить потрібний комплект фільтрів Блума. (див. Рис. 3 у [12]). Дана техніка збільшує апаратні витрати рівно в стільки разів, в скільки пришвидшує процес


ISSN 2786-9024

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

розпізнавання. Тобто фільтр Блума також має добру масштабованість за швидкодією.

**Подальше підвищення швидкодії**. Відомо про багато спроб дослідників підвищити швидкісні характеристики підходу до розпізнавання патернів на основі ФБ без розпаралелювання. Але чи не єдиним здобутком виявилася техніка каскадування, яка на практиці призвела до суттєвого зниження енергозбереження, але не до прискорення, на що вказано в роботі [15]. В якості ілюстрації докладених зусиль можна навести це ж саме дослідження, в якому хоч і вдалося підвищити продуктивність системи МСВВ у цілому, але за рахунок маніпулювань із запитами на обробку вхідних даних на рівні мережевих пакетів, не змінюючи власне структуру фільтра Блума.

**Особливості реалізації фільтра Блума на ПЛІС**. Для створення ефективного ФБ реконфігуровними засобами має важливе значення вибір геш-функцій. Ці функції мають, по-перше, однозначно відображати множину комбінацій вхідних символів на множину вихідних значень, по-друге, цифрова схема їх реалізації мати якомога простішу апаратну структуру, щоб забезпечити високу швидкодію при невеликих ресурсних витратах, по-третє, мають легко реалізовуватися на ПЛІС. Сформульованим вимогам задовольняють так звані геш-функції класу $H_3$ [16]. Притаманна ним властивість рекурсивності призводить до регулярності цифрової схеми та дозволяє знизити потрібні ресурси.

Іншою особливістю базової схеми фільтра Блума (Рис. 1) є необхідність одночасного доступу до масиву бітів Rg багатьох геш-функцій. У разі його реалізації на єдиному пристрої зберігання даних можуть виникнути колізії. Помірні вимоги ФБ до обсягу пам'яті дозволяють реалізовувати бітовий масив в на базі блоків вбудованої пам'яті BRAM мікросхеми ПЛІС. Такі блоки дозволяють синтезувати двопортові пристрої пам'яті, що забезпечує одночасний доступ до комірок бітового масиву двох геш-функцій. Але реальна кількість геш-функцій в практичних додатках зазвичай в кілька разів більша.

Рішення полягає у розбитті бітового масиву на декілька пристроїв, для чого потрібно, по-перше, розподілити виходи блоків геш-функцій між відповідними ОЗП, по-друге, на функціональність цих блоків накласти додаткові обмеження, щоб їх вихідний діапазон укладався у зменшений об'єм запам'ятовуючого пристрою. Такі обмеження впливають на вірогідність помилок другого роду ФБ, втім не суттєво [12].

**Оцінка ефективності ФБ та потреба в динамічному переналаштуванні**. В роботі [17] розглянуто критерії оцінки ефективності та відповідні показники для реконфігуровних сигнатурних систем захисту інформації. Потреба в засобах швидкої оцінки кількісних характеристик таких систем та їх компонентів обґрунтована у [3]. Там же розглянуто метод прискореного розрахунку таких характеристик без потреби виконання витратної за часом повної процедури компіляції проєкту щодо синтезу цифрової схеми в ПЛІС.

Одним з важливих некількісних показників ефективності сигнатурних реконфігуровних систем захисту інформації є здатність змінювати склад *патернів* (додавати та вилучати), що них має відшукувати система у вхідному потоку даних, без припинення її функціонування та без реконфігурації ПЛІС [17, 18]. Назвемо цю функцію здатністю до *динамічного переналаштування.* Класична структура ФБ програмується (конфігурується) перед початком роботи і принципово не здатна в процесі розпізнавання вилучати патерни, які Були неявно внесені до регістра бітів на етапі програмування. Однак мінлива та динамічна природа завдань захисту інформації в комп'ютерних мережах в деяких ситуаціях вимагає зміни набору патернів, що розпізнаються.

**Метою** дослідження є дослідження можливих рішень та шляхів їх практичної реалізації щодо організації динамічної зміни складу патернів, що розпізнаються реконфігуровним фільтром Блума на базі ПЛІС, без припинення його функціонування та без перезавантаження в ПЛІС конфігураційної послідовності. Важливим завданням також є знаходження функції оцінки для знайденого рішення, що дозволить задіяти метод прискореного розрахунку кількісних характеристик при проєктуванні цифрової схеми ФБ з новою властивістю під час практичної реалізації даного рішення.

## МАТЕРІАЛИ ТА МЕТОДИ ДОСЛІДЖЕНЬ

В роботі використовувалися методи булевої алгебри, цифрової схемотехніки, теорії обчислювальних систем, а також основи теорій обчислень на рядках та реконфігуровних обчислень. Втім, в якості основного інструменту застосовано метод прискореної оцінки кількісних характеристик реконфігуровних сигнатурних систем захисту інформації [3, 18]. Метод базується на використані так званих функцій оцінки (ФО), які дозволяють знаходити ресурсні та часові параметри модулів розпізнавання таких систем без виконання процедури компіляції проєкту створення в ПЛІС обчислювальної структури. Головну увагу в даному дослідженні зосереджено на ресурсній складовій ФО,

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

тобто на апаратних витратах на створення ФБ з лічильниками.

## РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

### *МОЖЛИВОСТІ ФІЛЬТРА БЛУМА ЩОДО ДИНАМІЧНОГО ПЕРЕНАЛАШТУВАННЯ*

Аналіз базової схеми ФБ свідчить про те, що якщо додавати патерни в процесі його функціонування можливо, то вилучати їх дана схема не дозволяє принципово. Розмаїття відомих модифікацій ФБ надає два варіанти вирішення проблеми його динамічного переналаштування.

Перший варіант пов'язаний з використанням так званого фільтра Блума з лічильниками, Counting Bloom Filter (CBF). Вперше подібна модифікація була запропонована ще в 2000-му році [19]. В схему додається $M$ лічильників (по числу розрядів регістра бітів). Кожний лічильник розташовується між виходом блоку обчислення геш-функцій та відповідним входом регістра бітів. В процесі програмування фільтра (або додання під час роботи) обчислені значення геш-функцій подаються на лічильників як вхідні імпульси в режимі додавання. Якщо в процесі функціонування ФБ виникає потреба вилучити патерн з набору, значення виходів геш-функцій керують лічильниками в режимі віднімання. Перехід числа, накопиченого в лічильнику, зі значення "0" у значення "1" призводить до запису "1" у відповідну комірку РБ фільтра Блума; перехід лічильника зі стану "1" в стан "0" – до запису "0". В іншому принцип функціонування схеми не змінюється. Таким чином, техніка використання лічильників дозволяє динамічно переналаштовувати ФБ в реальному часі без перепрограмування ПЛІС та тимчасового припинення функціонування сигнатурної системи захисту інформації.

Недоліком схеми фільтра Блума з лічильниками є додаткові витрати ресурсів ПЛІС. Причому точне значення розрядності лічильників складно передбачити, тому що кількість "влучань" різних геш-функцій в одну й ту саму комірку РБ є випадковою величиною. Якщо використати лічильники недостатньої розрядності, їх переповнення призводитиме до виникнення помилок першого роду (false negative), тобто, схема буде розпізнавання не всі патерни, що були запрограмовані. У загальному сенсі це гірше за притаманні ФБ помилки другого роду, які технічно можуть бути виправлені додатковими заходами. Але, як свідчать розрахунки, вірогідність таких подій невелика. В роботі [19] наведено вирази для оцінки вірогідності помилки першого роду. Виходячи з їх аналізу робиться припущення, що 4-розрядних лічильників достатньо для більшості практичних застосувань.

Другий варіант вирішення проблеми передбачає перерахунок значень РБ програмними засобами з послідовним завантаженням до нього вже модифікованого набору бітів. Такий прийом не потребує додаткових апаратних витрат порівняно з базовою схемою. Взагалі, будь-які зміни завантаженої в ПЛІС конфігурації не потрібні. (Відповідно, для функцій оцінки кількісних характеристик ФБ можуть бути використані відомі вирази для класичних схем ФБ). Але, по-перше, в цьому випадку необхідно задіяти спеціальний режим конфігурації, при якому всередині ПЛІС відновлюється тільки зміст внутрішньої, блокової пам'яті BRAM, а інші частини кристалу залишаються незмінними. По-друге, програмний режим оновлення значень РБ суттєво уповільнює процес динамічного переналаштування (У випадку потреби вилучити патерн з набору, що розпізнається, потрібно повністю повторити процедуру програмування). І, якщо першу умову в сучасних реконфігуровних прискорювачах реалізувати відносно нескладно, другий момент може звести нанівець всі переваги такого підходу, якщо на процедуру динамічного переналаштування накладаються суворі часові обмеження.

Тому нижче розглянуто реалізацію на ПЛІС саме схеми CBF. Також розраховано та наведено ресурсну ФО для даної схеми.

### *СТРУКТУРНА СХЕМА ФІЛЬТРА БЛУМА З ЛІЧИЛЬНИКАМИ*

В роботі [20] докладно досліджено різни схеми розпізнавання для реконфігуровних систем захисту інформації, включаючи різни модифікації ФБ.

Схема генераторів геш-функцій (ГГФ) не залежить від модифікації та має $B$ входів і $K$ виходів по $G$ розрядів кожний, де $B$ – кількість розрядів у патерні, яка дорівнює здобутку кількості символів у патерні та кількості бітів у слові (8 – при байтовому кодуванні), $K$ – кількість геш-функцій у ФБ, $G$ – розрядність геш-функцій (двійковий логарифм від довжини $M$ регістра бітів).

Як згадувалося вище під час обговорення особливостей реалізації ФБ на ПЛІС, вбудована блокова пам'ять BRAM зазвичай дозволяє створити, максимум, двопортові пристрої пам'яті, які надають одночасний доступ до комірок бітового масиву лише двох геш-функцій. З іншого боку, кількість $K$ геш-функцій визначає вірогідність помилки розпізнавання другого роду: ця вірогідність обернено пропорційна двійці у ступеню $K$.


ISSN 2786-9024

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

Наприклад, якщо задіяне 5 блоків BRAM, $K = 10$, і ця вірогідність $p_{пом.2} = 2^{-10} \approx 0,00098$.

При оптимальному співвідношенні основних параметрів ФБ для $K = 10$ і $M = 4096$ (типовий розмір блоку BRAM) схема буде здатна розпізнавати $N \approx 1420$ патернів. У випадку, коли цієї кількості патернів достатньо, схема РБ спрощується до модифікації так званого спрощеного фільтра Блума – Simplified Bloom Filter (SBF). Реалізація схеми регістра бітів для даної модифікації на ПЛІС наведена на Рис. 3.

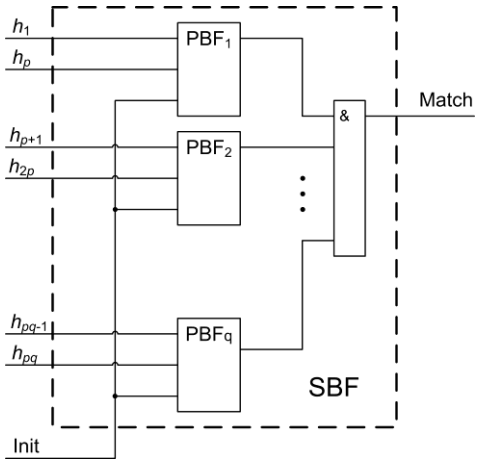


**Рис. 3. Схема SBF регістра бітів для спрощеного фільтра Блума**

Тут PBF (Partial Bloom Filter) – схема РБ для часткового ФБ, реалізованого на одному блоці BRAM (Рис. 4). Мультиплексор MUX дозволяє в режимі програмування завантажувати у BRAM через один з портів заздалегідь підготовлену комбінацію бітів (керує процесом завантаження інтерфейсний блок IU). В той час як в режимі функціонування обидва входи працюють на прийом значень двох геш-функцій [20].

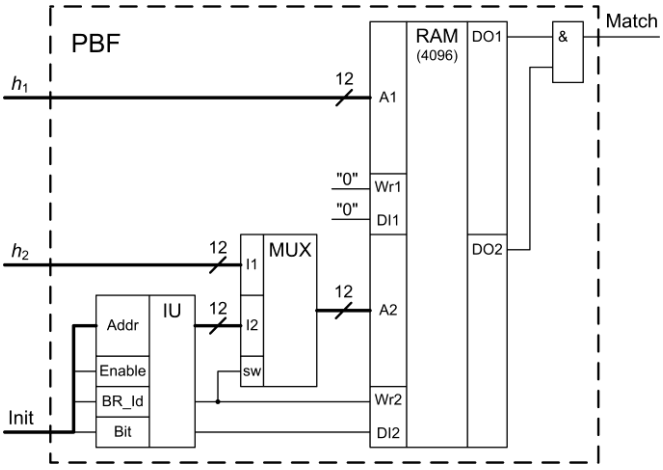


**Рис. 4. Схема PBF регістра бітів для часткового фільтра Блума**

Якщо ФБ повинен розпізнавати більшу кількість патернів, ніж здатен містити SBF, потрібна повнорозмірна схема РБ. Відповідна структура має назву Повнорозмірний фільтр Блума – LBF (Large Bloom Filter) і наведена на Рис. 5.

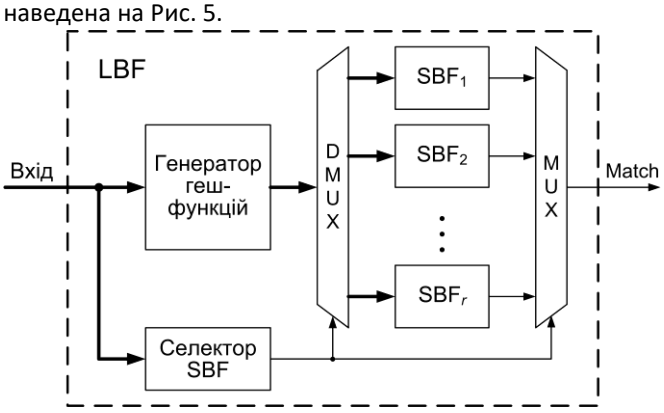


**Рис. 5. Схема повнорозмірного фільтра Блума LBF**

Тут поданням виходів ГГФ на входи потрібних спрощених схем регістрів SBF керує спеціальний селектор SBF (згідно алгоритму розподілу, про який йшлося наприкінці підрозділу "Особливості реалізації фільтра Блума на ПЛІС").

Розглянемо, як зміняться компоненти ФБ при реалізації рішення з використанням лічильників.

Схеми ГГФ, як і узагальнені схеми спрощеного SBF (рис. 3) та повнорозмірного LBF (Рис. 5) фільтрів Блума залишаться без змін. Модифікація торкнеться лише схеми регістра бітів для часткового фільтра Блума – PBF (Рис. 4). Назвемо цю схему CPBF (Counting Partial Bloom Filter). Оскільки додавати патерни тепер буде потрібно під час

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

функціонування ФБ, до схеми РБ додається ще один мультиплексор MUX (Рис. 6). Інтерфейсний блок IU замінюється блоком керування CU, який тепер керує не тільки процесом завантаження РБ перед початком роботи, але також доданням та вилучанням патернів.

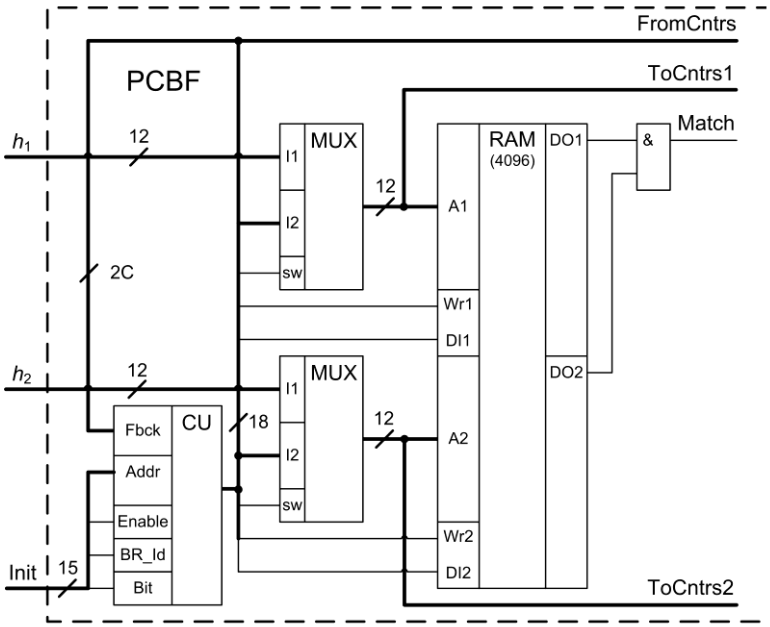


**Рис. 6. Схема регістра бітів для часткового CBF**

Оскільки кількість лічильників у CBF дорівнює числу $M$ бітів у РБ. значення якого може сягати кількох тисяч, синтезувати їх з конфігураційних блоків CLB мікросхеми ПЛІС не є доцільним. В даному дослідження пропонується задіяти для цього вбудовану швидкодіючу блокову пам'ять BRAM. Комірки ОЗП, що створюється на базі BRAM, будуть використовуватися як D-тригери, які за рахунок зворотного зв'язку функціонуватимуть як лічильні Т-тригери. Логіка з'єднань між каскадами лічильників, яка створює з Т-тригерів реверсивний асинхронний лічильник, зосереджена в блоках переносів $CL_1$ … $CL_{C-1}$ (Рис. 7). Тут $C$ – кількість розрядів лічильників. Перемикання між режимами додавання/віднімання, а також занесення нуля або одиниці у відповідну комірку РБ також здійснює CU.

Сумістити РБ з наймолодшим розрядом лічильників неможливо, оскільки в такому випадку не буде змоги відслідковувати події переходів лічильників в нульовій стан та – з нульового стану, в результаті яких в комірки РБ повинні заноситися нулі та одиниці відповідно.

## *ФУНКЦІЇ ОЦІНКИ ДЛЯ ФІЛЬТРА БЛУМА З ЛІЧИЛЬНИКАМИ*

Як було згадано вище, метод швидкої оцінки кількісних характеристик реконфігуровних сигнатурних систем захисту інформації базується на створенні та використанні функцій оцінки [3, 18]. Ці функції описують залежність від набору патернів, що розпізнаватимуться, таких параметрів синтезованих в ПЛІС цифрових схем, як апаратні (ресурсні) витрати та швидкодія.

Принцип побудови ФО апаратних витрат базується на підрахунку всіх апаратних ресурсів, які задіяні при створені відповідного компонента або системи в цілому. При цьому, щоб звести підрахунки до якоїсь універсальної одиниці, наприклад, "умовних LUT" – логічних таблиць (ЛТ) мікросхеми ПЛІС, різнорідні ресурси додаються з відповідними коефіцієнтами. Тоді загальну кількість апаратних витрат на синтез певної схеми можна підрахувати за формулою:

$$R = L + \alpha F + \beta B + \gamma S, \quad (4)$$

де $L$ – кількість ресурсів логіки ПЛІС (кількість ЛТ),

$F$ – кількість ресурсів розподіленої пам’яті ПЛІС (кількість тригерів),

$B$ – кількість ресурсів блокової пам’яті ПЛІС (кількість блоків BRAM),

$S$ – кількість ресурсів зовнішньої пам’яті – об’єм бортової пам’яті реконфігуровного прискорювача (у Мб),

$\alpha, \beta, \gamma$ – коефіцієнти нормалізації відносно ЛТ ресурсів різного типу, відповідно: кількості тригерів, числа блоків BRAM та об’єму бортової пам’яті RA.


ISSN 2786-9024

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

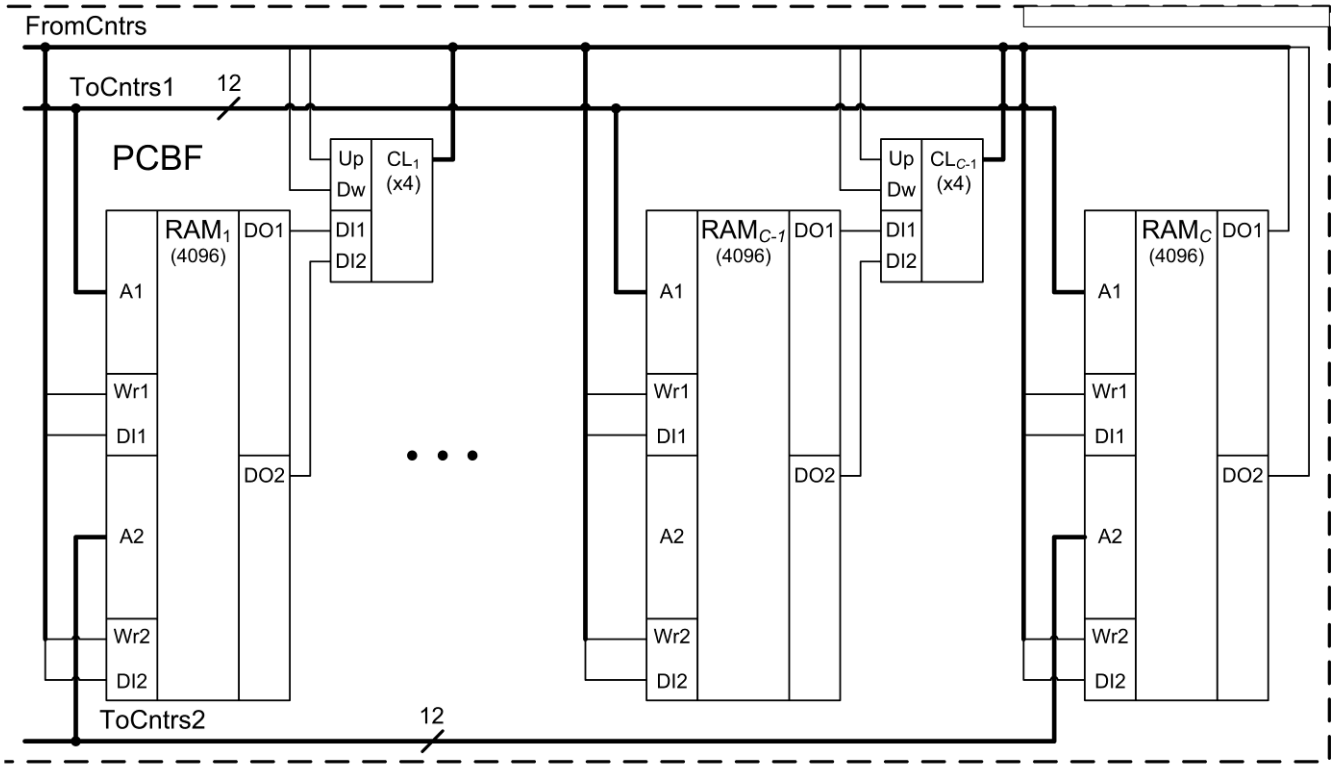


**Рис. 7. Схема лічильників для часткового CBF**

Слід зауважити, що формула (4) лише демонструє основну ідею підходу до оцінки апаратних витрат. Її нескладно розвинути шляхом додавання інших ресурсів ПЛІС та RA з відповідними коефіцієнтами.

Для складної цифрової схеми кількість використаних ресурсів дорівнює сумі ресурсів кожного компонента, з яких ця схема складається. Використовуючи такий підхід можна знайти ресурсні витрати $R_{BF}$ на синтез ФБ, які містять витрати на генератор геш-функцій $R_{ГФ}$, на регістр бітів $R_{РБ}$ та на допоміжні схеми $R_{доп}$:

$$R_{BF} = R_{ГФ} + R_{РБ} + R_{доп} = L_{ГФ} + \alpha F_{ГФ} + \beta B_{РБ} + L_{доп} + \alpha F_{доп} = \\ = L_{ГФ} + L_{доп} + \alpha(F_{ГФ} + F_{доп}) + \beta B_{РБ}, \quad (5)$$

де $L_{ГФ}$ і $L_{ДОП}$ – витрати ЛТ відповідно на ГГФ та на допоміжні схеми,

$F_{ГФ}$ і $F_{ДОП}$ – витрати тригерів відповідно на ГГФ та на допоміжні схеми,

$B_{РБ}$ – витрати на блокову пам'ять для РБ, які вимірюються кількістю задіяних блоків BRAM кристалу ПЛІС.

Якщо підрахувати всі види ресурсів, наприклад, для схеми спрощеного фільтра Блума SBF (див. докладні розрахунки у [20]), отримуємо наступну ресурсну функцію оцінки:

$$R_{SBF} = EG\left(\left\lceil\frac{8j}{x}\right\rceil + (\alpha+1)\left\lceil\frac{\left\lceil\frac{8j}{x}\right\rceil - 1}{x-1}\right\rceil - \alpha\right) + \\ + q\left(\alpha G + \beta + \left\lceil\frac{G}{\left\lfloor\frac{x-1}{2}\right\rfloor}\right\rceil + 4\right) + \left\lceil\frac{q-1}{x-1}\right\rceil, \quad (6)$$

де $E = -\lfloor \log_2 \rho_{дозв.} \rfloor$ – фактор хибного розпізнавання, який чисельно дорівнює кількості геш-функцій у ФБ і обернено пропорційний логарифму ймовірності хибного спрацьовування, прийнятного для конкретного застосування фільтра Блума;

$G = \left\lceil log_2 \frac{E \cdot \delta_j}{ln\, 2} \right\rceil$ – розрядність геш-функцій (дорівнює довжині адресу блоків BRAM), де $\delta_j$ – функція розподілу довжин патернів (кількість патернів довжиною j символів в наборі);

$j$ – довжина патернів, які розпізнає даний ФБ;

$x$ – число входів ЛТ конкретної ПЛІС, що використана в RA;

$q = \left\lceil \frac{E}{p} \right\rceil$ – кількість PBF в складі SBF, де $p$ – число портів блокової пам'яті BRAM.

Слід зауважити, що змінними у виразі (6) є: фактор хибного розпізнавання $E$, функція розподілу довжин патернів $\delta_j$ та довжина патернів $j$, тоді як решта змінних, які залежать від типу застосованої ПЛІС, є константами.

 https://doi.org/10.31474/2786-9024/v2i2(34).313785

При формуванні ресурсної ФО для схеми з лічильниками, побудованій на основі спрощеного ФБ – SCBF, деякі складові виразу (6) зміняться. У підрахунок витрат логіки на допоміжні схеми в складі часткового ФБ з лічильниками PCBF в схему CU додається $2(C-1)$ входів, кількість мультиплексорів MUX подвоюється, витрачаються логічні ресурси також на блоки переносів CL (по 4 ЛТ на кожний):

$$L_{\text{допPCBF}} = 3 + \left\lceil \frac{2(C-1)}{x} \right\rceil + 2\left\lceil \frac{G}{\left\lfloor \frac{x-1}{2} \right\rfloor} \right\rceil + 1 + \rightleftarrows 4(C-1) = \left\lceil \frac{2(C-1)}{x} \right\rceil + \rightleftarrows 2\left\lceil \frac{G}{\left\lfloor \frac{x-1}{2} \right\rfloor} \right\rceil + 4C. \quad (7)$$

Кількість тригерів в складі схеми CU збільшиться на 4:

$$F_{\text{допPCBF}} = G + 4. \quad (8)$$

Нарешті, у загальній схемі збільшиться кількість задіяних ресурсів вбудованої пам'яті – блоків BRAM:

$$B_{\text{SCBF}} = q(1 + C). \quad (9)$$

Тоді ресурсна ФО для схеми спрощеного ФБ з лічильниками матиме вигляд:

$$R_{\text{SCBF}} = EG\left(\left\lceil \frac{8j}{x} \right\rceil + \left\lceil \frac{\left\lceil \frac{8j}{x} \right\rceil - 1}{x-1} \right\rceil\right) + q\left(\left\lceil \frac{2(C-1)}{x} \right\rceil + 2\left\lceil \frac{G}{\left\lfloor \frac{x-1}{2} \right\rfloor} \right\rceil + 4C\right) + \left\lceil \frac{q-1}{x-1} \right\rceil +$$

$$+\alpha\left(EG\left(\left\lceil \frac{\left\lceil \frac{8j}{x} \right\rceil - 1}{x-1} \right\rceil - 1\right) + q(G+4)\right) + \beta q(1+C) =$$

$$= EG\left(\left\lceil \frac{8j}{x} \right\rceil + (\alpha+1)\left\lceil \frac{\left\lceil \frac{8j}{x} \right\rceil - 1}{x-1} \right\rceil - \alpha\right) + \alpha q(G+4) +$$

$$+q\left(\left\lceil \frac{2(C-1)}{x} \right\rceil + 2\left\lceil \frac{G}{\left\lfloor \frac{x-1}{2} \right\rfloor} \right\rceil + 4C + \beta(1+C)\right) + \left\lceil \frac{q-1}{x-1} \right\rceil. \quad (10)$$

Аналізуючи вираз (10), можна звернути увагу на відсутність в ньому, як і у виразі (6), коефіцієнта $\gamma$. Це означає, що схеми фільтра Блума, що описуються даними ФО, не споживають ресурсів зовнішньої по відношенню до ПЛІС пам'яті (бортового ОЗП прискорювача), а тільки логічні таблиці, тригери та блоки вбудованої пам'яті BRAM. Це позитивно впливає на швидкодію схем пристроїв.

## ОБГОВОРЕННЯ РЕЗУЛЬТАТІВ

Порівнюючи вирази (10) та (6) можна зробити висновок, що додавання лічильників у схему ФБ не призвело до драматичного збільшення апаратних витрат. Для порівняння конкретних значень, потрібно використати реальні набори патернів з баз даних сигнатур, наприклад, розповсюджених систем МСВВ.

Отримана функція оцінки дозволяє для будь-якого набору патернів розрахувати кількість апаратних ресурсів, потрібних для синтезу фільтра Блума з лічильниками, здатного розпізнавати даний набір. При цьому не потрібно кожного разу виконувати складну і часоємну процедуру компіляції проєкту для ПЛІС. Наведений приклад спрощеної схеми ФБ демонструє, яким чином можна формувати ФО для інших схем фільтрів Блума. В роботі [8] наведено декілька модифікацій ФБ з лічильниками для різних застосувань.

## ВИСНОВКИ

В роботі розглянуто особливості фільтра Блума, які слід враховувати при використанні даної схеми в реконфігуровних засобах захисту інформації безпеки. До переваг даного абстрактного пристрою відноситься економне споживання ресурсів та висока пропускна здатність у випадку його апаратної реалізації. ФБ добре масштабується за швидкодією, розміру словнику патернів та довжині патернів.

Регулярна та логічна структура обумовлює непогану здатність до реалізації на ПЛІС. Головна складність, втім, полягає в одночасному зверненні множини блоків генерації геш-функцій до багаторозрядного масиву бітів, якщо останні реалізований у вигляді ОЗП. Але ця проблема вирішується шляхом розбиття запам'ятовуючого пристрою на декілька окремих модулів та незначної корекції схематики обчислювачів геш-функцій.

До недоліків ФБ можна віднести здатність розпізнавати патерни тільки одного розміру.

Іншим обмеженням є неможливість змінювати склад патернів в процесі функціонування. З метою подолання цього обмеження використовуються модифікації ФБ з лічильниками. В роботі проаналізовані принципи побудови таких схем, розглянутий можливий варіант реалізації ФБ з лічильниками на базі ПЛІС.

Для швидкої оцінки апаратних витрат на синтез схеми, сформована функція оцінки, яка описує залежність значення ресурсів ПЛІС від набору патернів, що розпізнаватимуться. Подібним чином можна створити функції оцінки і для інших варіантів схеми ФБ з лічильниками.

Отримані в даному дослідженні результати дозволять розробникам цифрових пристроїв створювати більш ефективні реконфігуровні засоби інформаційної безпеки, в тому числі на об'єктах критичної інфраструктури, які висувають підвищені вимоги до їх продуктивності та швидкодії.

В подальшому планується проведення низки обчислювальних експериментів з метою оцінки збільшення апаратних витрат у схем з лічильниками на відкритих наборах патернів з баз даних сигнатур відомих МСВВ.


ISSN 2786-9024

**ФІНАНСУВАННЯ**

Статтю підготовлено за матеріалами дослідження, яке фінансується Національною академією наук України в рамках виконання науково-дослідної роботи "Розвиток методів і засобів підвищення рівня кіберзахищеності цифрових підстанцій (шифр: МОД-К)" протягом 2024-2026 років. Номер державної реєстрації: 0124U002384.

**ЛІТЕРАТУРА**

[1] B. H. Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", *Communications of the ACM*, Article vol. 13, no. 7, pp. 422-426, 1970, doi: 10.1145/362686.362692.

[2] D. V. Pryor, M. R. Thistle, and N. Shirazi, "Text searching on Splash 2", in *IEEE Workshop on FPGAs for Custom Computing Machines*, 1993, pp. 172-177.

[3] С. Я. Гільгурт, "Метод прискореної кількісної оцінки компонентів реконфігуровних сигнатурних систем кіберзахисту", *Електронне моделювання*, т. 44, № 5, с. 3-24, 2022. doi: 10.15407/emodel.44.05.003.

[4] S. Dharmapurikar, M. Attig, and J. Lockwood, "Design and Implementation of a String Matching System for Network Intrusion Detection using FPGA-based Bloom Filters", in *All Computer Science and Engineering Research*, Washington University in St. Louis, 2004, WUCSE-2004-12.

[5] R. Patgiri, S. Nayak, and N. B. Muppalaneni, "Is Bloom Filter a Bad Choice for Security and Privacy? ", *2021 International Conference on Information Networking (ICOIN)*, Jeju Island, Korea (South), pp. 648-653, 2021, doi: 10.1109/ICOIN50884.2021.9333950.

[6] S. Ya. Hilgurt, "Pattern Handling for Quantifying Hardware Components of Signature-Based Cybersecurity Systems", *Proceedings of the 2nd International Workshop on Information Technologies: Theoretical and Applied Problems (ITTAP 2022)*, Ternopil, Ukraine, Nov. 22-24, 2022. – CEUR Workshop Proceedings, vol. 3309, pp. 83-93, 2022, Available online: https://ceur-ws.org/Vol-3309/paper7.pdf.

[7] R. Patgiri, S. Nayak, and S. K. Borgohain, "Hunting the Pertinency of Bloom Filter in Computer Networking and Beyond: A Survey", *Journal of Computer Networks and Communications*, 2019, 2712417, 10 pages, doi: 10.1155/2019/2712417.

[8] L. Luo, D. Guo, R. T. B. Ma, O. Rottenstreich, and X. Luo, "Optimizing Bloom Filter: Challenges, Solutions, and Comparisons", in *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, 2019, pp. 1912-1949, doi: 10.1109/COMST.2018.2889329.

[9] S. Geravand, and M. Ahmadi, "Bloom filter applications in network security: A state-of-the-art survey", *Computer Networks*, Article vol. 57, no. 18, pp. 4047-4064, Dec 2013, doi: 10.1016/j.comnet.2013.09.003.

[10] M. A. Owaid, and O. A. Dawood, "A survey in privacy-preserving by bloom filters", in *Proceedings of the 4th international computer sciences and informatics conference (Icsic 2022)*, 28-29 June 2022, Amman, Jordan, 2022, doi: 10.1063/5.0174813.

[11] M. A. Ferrag, M. Babaghayou, and M. A. Yazici, "Cyber security for fog-based smart grid SCADA systems: Solutions and challenges", *Journal of Information Security and Applications*, Article vol. 52, 2020, Art no. 102500, doi: 10.1016/j.jisa.2020.102500.

[12] С. Гільгурт, "Побудова фільтрів Блума реконфігуровними засобами для вирішення задач інформаційної безпеки", *Безпека інформації*, т. 25, № 1, с. 53-58, 2019, doi: 10.18372/2225-5036.25.13594.

[13] J. Harwayne-Gidansky, D. Stefan, and I. Dalal, "FPGA-based SoC for Real-Time Network Intrusion Detection using Counting Bloom Filters", in *Proceedings of the IEEE Southeastcon 2009*, 2009.

[14] Y. Chen, A. Kumar, and J. Xu, "A new design of bloom filter for packet inspection speedup", in *IEEE Global Telecommunications Conference (GLOBECOM 07)*, Washington, DC, Nov 26-30 2007, NEW YORK: IEEE, in IEEE Global Telecommunications Conference (Globecom), 2007, pp. 1-5.

[15] N. S. Artan, K. Sinkar, J. Patel, and H. J. Chao, "Aggregated bloom filters for intrusion detection and prevention hardware", in *IEEE Global Telecommunications Conference (GLOBECOM 07)*, Washington, DC, Nov 26-30 2007, NEW YORK: IEEE, in IEEE Global Telecommunications Conference (Globecom), 2007, pp. 349-354.

[16] L. Carter, and M. Wegman, "Universal Classes of Hashing Functions", *Computer and System Sciences*, vol. 18, no. 2, pp. 143-154, 1979.

[17] С. Я. Гільгурт, "Порівняльний аналіз підходів до побудови компонентів реконфігуровних засобів технічного захисту інформації", *Проблеми інформатизації та управління*, т. 2, № 66, с. 17-26, 2021, doi: 10.18372/2073-4751.66.15712.

[18] S. Y. Hilgurt, A. M. Davydenko, T. V. Matovka, and M. P. Prygara, "Tools for Analyzing Signature-Based Hardware Solutions for Cyber Security Systems", *JCSANDM*, vol. 12, no. 03, pp. 339–366, 2023, doi: 10.13052/jcsm2245-1439.123.5.

[19] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: A scalable wide-area Web cache sharing protocol", *IEEE ACM Transactions on Networking*, Article vol. 8, no. 3, pp. 281-293, 2000, doi: 10.1109/90.851975.

[20] С. Я. Гільгурт, О. А. Чемерис, *Реконфігуровні сигнатурні засоби захисту інформації комп'ютерних систем*. Київ, Україна: Академперіодика, 2022, 297 с., doi: 10.15407/akademperiodyka.458.297.

## ESTIMATION OF COUNTING BLOOM FILTER HARDWARE COSTS FOR FPGA-BASED CYBERSECURITY SYSTEMS

Serhii Hilgurt

***The objective of the study is to research possible solutions and ways of practical implementation regarding the organization of dynamic changing the list of patterns that are matched by the hardware (FPGA-based) Bloom filter in the signature cybersecurity systems, such as network intrusion detection systems, antiviruses, spam filters, etc. Changing the list of patterns means both adding and removing elements from the list of patterns that the signature system searches in the incoming data stream, in particular in the bodies of network traffic packets.***

*To achieve the goal, the principles of construction and functioning of the Bloom filter, its advantages and disadvantages compared to other matching schemes in terms of known performance indicators are analyzed in the work. The modifications and variations proposed by developers to improve its characteristics during the entire period of use in computer network cybersecurity systems are considered. The features of the hardware implementation of the Bloom filter on the FPGAs are analyzed.*

*The one of two discovered approaches to solve the problem of dynamic readjustment, that provides greater speed due to changing the hardware structure of the device, namely the Bloom filter with counters, is considered in the work. On the example of one of the hardware scheme modifications, the so-called simplified Bloom filter, a possible option for building its digital structure is considered. To increase the efficiency of the process of developing cybersecurity systems using this scheme, a hardware resource evaluation function has been created, which allows you to find the quantitative characteristics of the cost for synthesizing FPGA-based digital devices without performing the time-consuming procedure of full project compilation. A preliminary comparison of the obtained expression with the evaluation function for the scheme of the simplified Bloom filter without counters was carried out.*

***Keywords****: NIDS, multi-pattern matching, FPGA, Bloom filter with counters, hardware costs.*

## REFERENCES

[1] B. H. Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", *Communications of the ACM*, Article vol. 13, no. 7, pp. 422-426, 1970, doi: 10.1145/362686.362692.

[2] D. V. Pryor, M. R. Thistle, and N. Shirazi, "Text searching on Splash 2", in *IEEE Workshop on FPGAs for Custom Computing Machines*, 1993, pp. 172-177.

[3] S. Ya. Hilgurt, "Accelerated Quantitative Evaluation of Components of FPGA-Based Security Systems", *Electronic Modeling*, vol. 44, no. 5, pp. 3-24, 2022, doi: 10.15407/emodel.44.05.003. (In Ukrainian).

[4] S. Dharmapurikar, M. Attig, and J. Lockwood, "Design and Implementation of a String Matching System for Network Intrusion Detection using FPGA-based Bloom Filters", in *All Computer Science and Engineering Research*, Washington University in St. Louis, 2004, WUCSE-2004-12.

[5] R. Patgiri, S. Nayak, and N. B. Muppalaneni, "Is Bloom Filter a Bad Choice for Security and Privacy? ", *2021 International Conference on Information Networking (ICOIN)*, Jeju Island, Korea (South), pp. 648-653, 2021, doi: 10.1109/ICOIN50884.2021.9333950.

[6] S. Ya. Hilgurt, "Pattern Handling for Quantifying Hardware Components of Signature-Based Cybersecurity Systems", *Proceedings of the 2nd International Workshop on Information Technologies: Theoretical and Applied Problems (ITTAP 2022)*, Ternopil, Ukraine, Nov. 22-24, 2022. – CEUR Workshop Proceedings, vol. 3309, pp. 83-93, 2022, Available online: https://ceur-ws.org/Vol-3309/paper7.pdf.

[7] R. Patgiri, S. Nayak, and S. K. Borgohain, "Hunting the Pertinency of Bloom Filter in Computer Networking and Beyond: A Survey", *Journal of Computer Networks and Communications*, 2019, 2712417, 10 pages, doi: 10.1155/2019/2712417.

[8] L. Luo, D. Guo, R. T. B. Ma, O. Rottenstreich, and X. Luo, "Optimizing Bloom Filter: Challenges, Solutions, and Comparisons", in *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, 2019, pp. 1912-1949, doi: 10.1109/COMST.2018.2889329.

[9] S. Geravand, and M. Ahmadi, "Bloom filter applications in network security: A state-of-the-art survey", *Computer Networks*, Article vol. 57, no. 18, pp. 4047-4064, Dec 2013, doi: 10.1016/j.comnet.2013.09.003.

[10] M. A. Owaid, and O. A. Dawood, "A survey in privacy-preserving by bloom filters", in *Proceedings of the 4th international computer sciences and informatics conference (ICSIC 2022)*, 28-29 June 2022, Amman, Jordan, 2022, doi: 10.1063/5.0174813.

[11] M. A. Ferrag, M. Babaghayou, and M. A. Yazici, "Cyber security for fog-based smart grid SCADA systems: Solutions and challenges", *Journal of Information Security and Applications*, Article vol. 52, 2020, Art no. 102500, doi: 10.1016/j.jisa.2020.102500.

[12] S. Hilgurt, "Constructing Bloom filters by reconfigurable means for solving information security tasks", *Ukrainian Scientific Journal of Information Security*, vol. 25, no. 1, pp. 53-58, 2019, doi: 10.18372/2225-5036.25.13594. (In Ukrainian).

[13] J. Harwayne-Gidansky, D. Stefan, and I. Dalal, "FPGA-based SoC for Real-Time Network Intrusion Detection using Counting Bloom Filters", in *Proceedings of the IEEE Southeastcon 2009*, 2009.

[14] Y. Chen, A. Kumar, and J. Xu, "A new design of bloom filter for packet inspection speedup", in *IEEE Global Telecommunications Conference (GLOBECOM 07)*, Washington, DC, Nov 26-30 2007, NEW YORK: IEEE, in IEEE Global Telecommunications Conference (Globecom), 2007, pp. 1-5.

[15] N. S. Artan, K. Sinkar, J. Patel, and H. J. Chao, "Aggregated bloom filters for intrusion detection and prevention hardware", in *IEEE Global Telecommunications Conference (GLOBECOM 07)*, Washington, DC, Nov 26-30 2007, NEW YORK: IEEE, in IEEE Global Telecommunications Conference (Globecom), 2007, pp. 349-354.

[16] L. Carter, and M. Wegman, "Universal Classes of Hashing Functions", *Computer and System Sciences*, vol. 18, no. 2, pp. 143-154, 1979.

[17] S. Ya. Hilgurt, "Comparative analysis of approaches to the building of reconfigurable security tools components", *Problems of informatization and management*, vol. 2, no. 66, pp. 17-26, 2021, doi: 10.18372/2073-4751.66.15712.

[18] S. Y. Hilgurt, A. M. Davydenko, T. V. Matovka, and M. P. Prygara, "Tools for Analyzing Signature-Based Hardware Solutions for Cyber Security Systems", *JCSANDM*, vol. 12, no. 03, pp. 339–366, 2023, doi: 10.13052/jcsm2245-1439.123.5.

[19] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: A scalable wide-area Web cache sharing protocol", *IEEE ACM Transactions on Networking*, Article vol. 8, no. 3, pp. 281-293, 2000, doi: 10.1109/90.851975.

[20] S. Ya. Hilgurt, O. A. Chemerys, *Reconfigurable signature-based information security tools of computer systems*. Kyiv, Ukraine: Akademperiodyka, 2022, p. 297, doi: 10.15407/akademperiodyka.458.297. (In Ukrainian).



ISSN 2786-9024