УДК 004.274:004.056

Гільгурт С.Я.,
orcid.org/0000-0003-1647-1790

ПОРІВНЯЛЬНИЙ АНАЛІЗ ПІДХОДІВ ДО ПОБУДОВИ КОМПОНЕНТІВ РЕКОНФІГУРОВНИХ ЗАСОБІВ ТЕХНІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ

Інститут проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України

hilgurt@ipme.kiev.ua

Вступ

Останнім часом у зав'язку зі сталим зростанням мережевого трафіку, кількості та витонченості кібератак, а також через зупинення частоти універсальних мікропроцесорів програмна реалізація складних засобів технічного захисту інформації (таких як системи виявлення вторгнень, додатки проти вірусів та спаму) вже не відповідають вимогам щодо їх швидкодії. Тому розробники звертаються до апаратних, а саме до реконфігуровних рішень на базі програмованих логічних інтегральних схем (ПЛІС), які поєднують в собі продуктивність спецпроцесорів і гнучкість програмного забезпечення [1-2].

Найбільш ресурсномісткою задачею реального часу в апаратних засобах технічного захисту інформації є множинне розпізнавання фіксованих послідовностей символів (патернів) [3]. Від того, наскільки успішно вдасться вирішити цю задачу, залежить ефективність системи захисту в цілому.

Існує багато підходів до побудови апаратних схем розпізнавання патернів (pattern matching англійською), заснованих на різних засадах [4]. Для їх успішного використання необхідно, по-перше, визначитися з критеріями та показниками ефективності, по друге – порівняти властивості та особливості таких підходів.

Критерії та показники ефективності

Щоб мати змогу оцінювати властивості розробок реконфігуровних засобів технічного захисту інформації (РЗТЗІ) та їх компонентів і порівнювати різні підходи до їх побудови та окремі технічні рішення, потрібно визначити критерії ефективності та відповідні показники.

До критеріїв ефективності будь-якого цифрового пристрою належать обсяг потрібних апаратних витрат, швидкодія та набір виконуваних функцій. Стосовно РЗТЗІ аналіз наявного світового досвіду дозолив виявити наступну ієрархію відповідних показників ефективності (ПЕ).

Всі показники можна поділити на основні та проміжні, що пов'язують деякі з основних (рис.1).

До категорії основних ПЕ належать: вартісні показники, показники продуктивності та функціональні показники.

Вартісними ПЕ є: обсяги логічних ресурсів програмованої логіки, які задіяні для створення цифрової схеми, витрати на пам'ять (як внутрішню – тригери логічних комірок та блокову пам'ять BRAM, так і зовнішню відносно кристалу ПЛІС бортову пам'ять реконфігуровного прискорювача), а також інші витрати, що можуть бути оцінені в одиницях вартості.

До показників продуктивності відносяться, з одного боку – об'єм словнику патернів (кількість різних патернів, що розпізнає засіб), з іншого – швидкодію засобу, яка характеризується або часом обробки пристроєм одиниці даних, або пропускною здатністю. Важливим якісним ПЕ цієї категорії також є передбачуваність пропускної здатності, тобто здатність пристрою забезпечувати наперед задане значення пропускної здатності. Така властивість спрощує оперування іншими швидкісними характеристиками системи та полегшує інтеграцію модулю розпізнавання з рештою компонентів технічної системи.

До функціональних ПЕ відносяться переважно якісні показники, а саме: спроможність протидіяти атакам, що спрямовані на РЗТЗІ, здатність до оновлення

словнику патернів без припинення процесу розпізнавання (властивість динамічної реконфігурації), здатність до селективного розпізнавання (можливість зміни за зовнішнім сигналом підмножини патернів, що розпізнаються,), здатність системи виявлення вторгнень працювати у режимі запобігання вторгнень та ін. При створенні деяких РЗТЗІ висуваються спеціальні вимоги до їх здібностей, в результаті чого з'являються додаткові функціональні показники.

Рис. 1. Ієрархія показників ефективності РЗТЗІ

Проміжними ПЕ є масштабованість (трьох типів: за пропускною здатністю, за об'ємом словнику патернів і за довжиною патернів), здатність використання надлишковості словнику патернів для покращення швидкісних і ресурсних характеристик та похідні показника, що формуються з кількох інших.

Кількісні похідні показники ефективності мають вигляд певної функціональної залежності (математичного виразу довільної складності) від основних кількісних ПЕ. Потреба в таких показниках з'явилася в результаті накопичення досвіду створення та використання РЗТЗІ світовою дослідницькою спільнотою для більш витонченого вимірювання та порівняння кількісних характеристик технічних рішень для засобів захисту.

Підходи до побудови апаратних схем розпізнавання патернів

Як свідчить проведений автором аналіз численних публікацій, найкращі здібності в сенсі ефективності при побудові апаратних схем множинного розпізнавання патернів продемонстрували наступні три підходи:

- асоціативна пам'ять на базі цифрових компараторів;
- фільтр Блума на базі геш-функцій;
- алгоритм Ахо–Корасік на базі скінченних автоматів,

а також окремі технології, на яких вони ґрунтуються, і модифікації внаслідок застосування різноманітних технік покращення технічних характеристик відповідних рішень.

Формалізацію на якісному рівні кожного з перелічених напрямів проведено в публікаціях автора [5-7] відповідно. Головна увага при цьому була спрямована на особливості (переваги та недоліки) кожного підходу в сенсі показників ефективності, сформульованих вище, специфіку реалізації підходів на ПЛІС, складнощі та проблеми, що виникають під час ство-

рення реконфігуровних засобів розпізнавання на базі цих підходів, а також на шляхи їх усунення.

Використання асоціативної пам'яті на базі цифрових компараторів

Асоціативна пам'ять (АП) – англійською Content Addressable Memory (CAM), є класом пристроїв, що створювалися саме для швидкого розпізнавання кодів. Вони виконують функцію, протилежну традиційному ОЗП: за змістом відшукують місце розташування даних в запам'ятовуючому пристрої або свідчать про їх відсутність. Швидкодіючою основою АП є цифрові компаратори (ЦК).

Безпосереднім рішенням щодо виявлення збігу вхідних даних з патернами є набір цифрових компараторів, кожен з яких порівнює певний байт вхідної послідовності з відповідним символом патерну. Назвемо таку схему Базовою схемою на АП або схемою BsCAM (рис.2). Після подачі вхідних даних схема BsCAM здатна отримувати результат на виході за один такт синхронізації. Її перевагами є також простота та регулярність структури. Головний недолік – значне споживання логічних ресурсів ПЛІС: для її створення потрібно стільки схем ЦК, скільки символів загалом міститься у словнику патернів. Додатково також потрібні схеми об'єднання по "І" виходів компараторів.

Реалізація схеми BsCAM реконфігуровними засобами ускладнюється, по-перше, перенавантаженням виходів регістрів, побудованих на штатних компонентах ПЛІС, тому що сигнали з них подаються на входи багатьох компараторів, по-друге, необхідністю об'єднувати велику кількість бітових сигналів багатовходовими схемами "І" для довгих патернів. Обидві проблеми вирішуються побудовою конвеєрних схем розгалуження, що призводить до додаткового збільшення апаратних витрат, тобто є причиною відносно поганої масштабованості підходу за об'ємом словнику патернів.

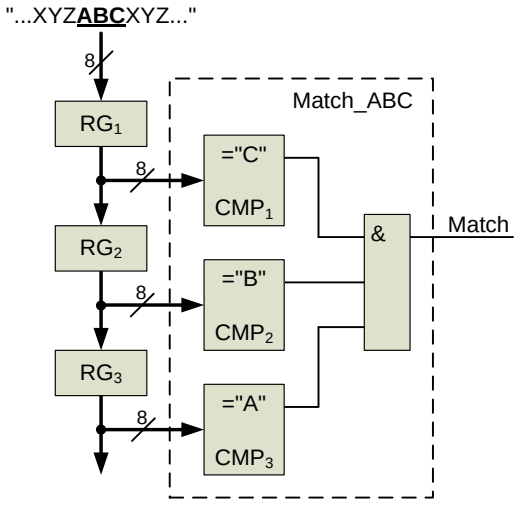

Рис. 2. Безпосереднє розпізнавання цифровими компараторами

Скоротити апаратні витрати АП дозволяє повторне використання компараторів, що в граничному випадку призводить до виділення лише одного ЦК на кожний символ алфавіту, а їх комбінації формуються за допомогою цифрових схем затримки (ЦСЗ) та схем "І", як показано на рис.3. На цьому рисунку DEL(1) та DEL(2) – ЦСЗ на один і на два такти відповідно. Таке рішення отримало назву *Схема розпізнавання на базі декодованої асоціативної пам'яті* або *схема DCAM* (Decoded Content-Addressable Memory), тому що повний комплект ЦК фактично є дешифратором (декодером).

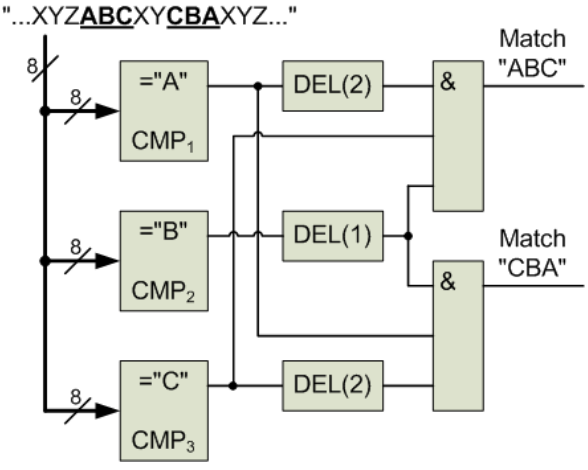

Рис. 3. Схемне рішення DCAM

Подальше зменшення складності схеми розпізнавання на базі DCAM можливе шляхом використання техніки часткового розпізнавання, коли довгі патерни

розбиваються на коротші фрагменти, що розпізнаються послідовно. При цьому достатньо затримувати лише сигнал часткового збігу замість використання для довгих патернів великих ЦСЗ. На рис.4 наведено схему розпізнавання 31-символьного патерну, що побудована за таким принципом. Це рішення отримало назву *Схема розпізнавання на базі частково декодованої АП* або *схема DpCAM* (Decoded partial Content-Addressable Memory).

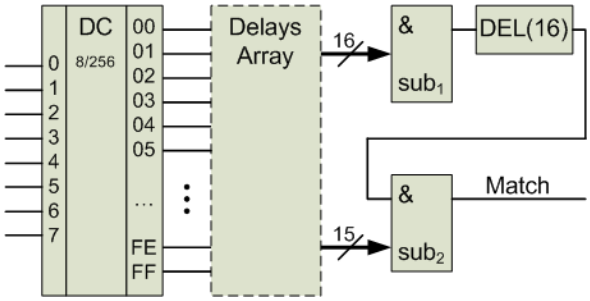

Рис. 4. Схемне рішення DpCAM

Ефект зниження апаратних витрат за рахунок повторного використання також працює при використанні паралельного з'єднання декількох блоків розпізнавання на ЦК для підвищення швидкодії. Дослідження свідчать, що в цьому випадку використання описаних вище технік зменшення витрат призводить навіть до більш високої економії ресурсів ніж лінійний зріст. Отже, схеми АП на ЦК мають добру масштабованість за пропускною здатністю.

В роботі [5] також розглянуто використання небайтової обробки даних, а саме, схеми на півбайтових компараторах, а також схеми, що будуються з використанням бінарних діаграм рішень.

Зменшити витрати можливо також шляхом кластеризації вхідного набору патернів для розбиття схеми на відповідну кількість підсхем меншого розміру.

Той факт, що зміст вхідних даних ніяким чином не впливає на характер роботи схем розпізнавання АП на базі ЦК, обумовлює наявність у цих схем показника передбачуваності пропускної здатності та робить системи, що використовують даний підхід, невразливими до атак на РЗТЗІ, що є здійсненням іншого функціонального ПЕ.

Як можна бачити по розглянутих рішеннях, при створенні засобів розпізнавання на ЦК інформація про патерни фактично "прошивається" в апаратну схему, що унеможливлює динамічну реконфігурацію та селективне розпізнавання.

Використання фільтрів Блума на базі геш-функцій

Фільтр Блума (ФБ) – англійською Bloom Filter (BF) – це абстрактний пристрій, який дозволяє виявити збіг фрагменту заданої послідовності бітів зі зразком із словнику. ФБ складається з двох ключових компонентів (рис.5): комплекту з K блоків, що обчислюють геш-функції $h_1(x)$, $h_2(x)$, …, $h_K(x)$, та масиву з M бітових комірок (компонент Rg на рисунку). В початковому стані цей регістр бітів (РБ) заповнений нулями.

На етапі програмування на входи блоків геш-функцій послідовно подається кожен з n елементів словнику патернів (довжиною W бітів), для кожного з цих елементів обчислюються всі K геш-функцій, значення котрих інтерпретуються як адреси комірок у РБ, в які заносяться значення "1".

В процесі функціонування фільтра Блума на його вхід подається фрагмент вхідної послідовності символів (також довжиною W бітів), і знову обчислюються значення всіх K геш-функцій (рис.6). 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 приймали участь у програмуванні ФБ. Але якщо хоча б одна геш-функція вкаже на комірку з нульовим значенням, це гарантовано свідчить про відсутність збігу. Отже, ФБ функціонує з деякою вірогідністю помилки розпізнавання другого роду (false positive), проте без помилок першого роду (false negative).

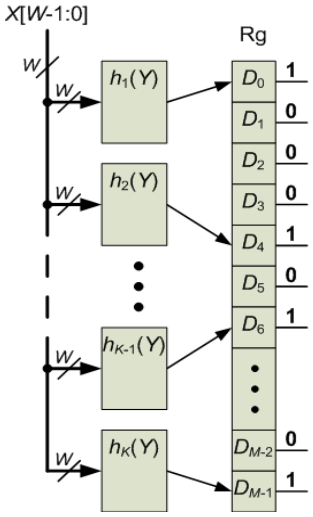

Рис. 5. Фільтр Блума в режимі програмування

ФБ дозволяє економно використовувати ресурси пам'яті (розмір словнику патернів не впливає напряму на розмір РБ: додання нових патернів до вже присутніх призводить лише до підвищення вірогідності помилки розпізнавання, але не до збільшення об'єму потрібної пам'яті). Кількість та складність геш-функцій, отже й апаратурна ресурсоємність і продуктивність при даному підході теж не залежать від об'єму словника патернів. Довжина патернів так само не впливає напряму ні на кількість ресурсів пам'яті, ні на продуктивність. Тобто ФБ добре масштабується по двох напрямах: за об'ємом словнику патернів та за довжиною патернів. Дійсно, навіть дуже довгі рядки після перетворення геш-функціями потребують для зберігання ті ж самі *K* комірок РБ.

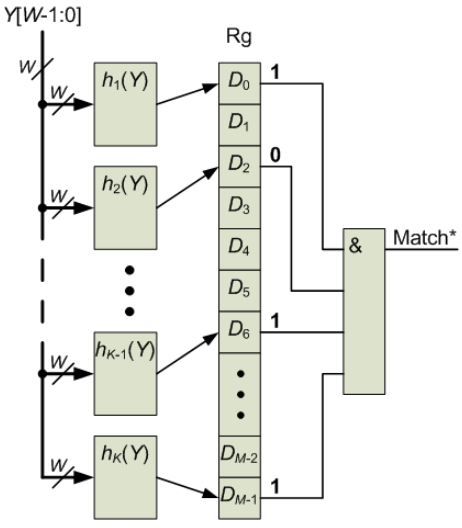

Рис. 6. Фільтр Блума в режимі розпізнавання

На жаль, фільтр Блума має важливий недолік, притаманний всім рішенням на базі гешування: розмір вхідної послідовності символів для аналізу має бути фіксованим для обраного набору геш-функції. Тобто, один ФБ здатен розпізнавати патерни тільки однакової довжини. Для подолання цього недоліку будують структуру, що містить кілька ФБ для патернів різної довжини. Це погіршує вартівні показники ефективності схем на фільтрах Блума, але не швидкісні.

Складності побудови ФБ на ПЛІС пов'язані насамперед с тим фактом, що в базовій схемі фільтра Блума кільком генераторам геш-функцій потрібно одночасно доступатися до регістру бітів Rg. Якщо реалізувати останній на єдиному запам'ятовуючому пристрої (наприклад, на базі блоку BRAM), виникатимуть колізії. Рішення полягає у розбитті РБ на декілька запам'ятовуючих пристроїв в залежності від числа геш-функції. Для цього потрібно, по-перше, розподілити виходи блоків реалізації 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. Доведено, що такі обмеження не суттєво підвищують вірогідність помилок другого роду фільтра Блума.

В загальному випадку для створення РБ потрібно кілька блоків BRAM. Відповідна схема отримала назву *Повнорозмірний фільтр Блума* або *схема LBF* (Large Bloom Filter). Але в більшості практичних застосувань для побудови модулю розпізнавання РЗТЗІ можна задіяти окремо так звані схеми міні-ФБ, які є складовими частинами LBF і містять лише по одному блоку BRAM. Назвемо таке рішення *Спрощеним фільтром Блума* або *схемою SBF* (Simplified Bloom Filter).

З метою підвищення швидкодії ФБ можна об'єднувати в паралельну структуру, але на відміну від ЦК властивості фільтрів Блума не дозволяють досягнути сублінейного закону зростання витрат – апаратні витрати спільної схеми строго

пропорційні досягнутому прискоренню, тобто за пропускною здатністю ФБ масштабується не так добре, як схеми на базі АП/ЦК.

Класична схема фільтра Блума не дозволяє в процесі функціонування вилучати патерни, додані під час програмування. Якщо це потрібно, застосовують *Фільтр Блума з лічильниками*, або *схему CBF* (Counting Bloom Filter). Таке рішення дозволяє здійснювати динамічну реконфігурацію без зупинення роботи РЗТЗІ.

Необхідність уточнення результатів внаслідок системної помилки ФБ, яке виконується повільніше, ніж розпізнавання, призводить до непередбачуваності пропускної здатності, та робить фільтр Блума вразливим до атак на РЗТЗІ.

Використання алгоритму Ахо–Корасік на базі скінченних автоматів

Математичний апарат цифрових автоматів (ЦА) вже багато років успішно використовується для створення обчислювальних пристроїв. Класичний ЦА задається шісткою елементів:

$$A = \{X, Y, S, S_0, f_s, f_y\}, \quad (1)$$

де X – множина вхідних сигналів, Y – множина вихідних сигналів, S – множина станів автомата, S_0 – початковий стан автомата, f_s – функція переходів з одного стану в інший, f_y – функція виходів автомата.

Але останнім часом в техніці частіше використовують так звані автомати – *розпізнавачі* (classifier). Такий автомат видає на виході активний сигнал тільки у випадку, якщо на його вхід надійшла одна з наперед заданих послідовностей символів (комбінацій вхідних сигналів). Коли ця ситуація трапляється, розпізнавач переходить в один з так званих *прийнятних* станів. Поки переходи здійснюються між неприйнятними станами, сигнали на виході відсутні. Саме такі автомати використовуються при створенні РЗТЗІ.

На відміну від класичного ЦА в автоматі – розпізнавачі замість множини вихідних сигналів Y та функції виходів f_y, з'являється множина прийнятних станів F:

$$A = \{X, S, S_0, f_s, F\}. \quad (2)$$

Отже, автомат – розпізнавач описується не шісткою, а п'ятіркою елементів.

Якщо у виразах (1) або (2) елементи X, Y, S та F є скінченними множинами, ЦА називають *скінченним автоматом* (СА) – англійською Finite Automaton (FA). Це стосується автоматів усіх типів, але останніми роками в технічній літературі словосполукою *скінченний автомат* частіше позначають саме автомати – розпізнавачі.

Скінченні автомати бувають *детермінованими* (ДСА) та *недетермінованими* (НСА). У детермінованому автоматі на кожному такті можливий лише один перехід лише до одного стану. Як наслідок в кожний окремий момент часу ДСА може перебувати тільки в одному стані. Для НСА вказані обмеження не виконуються, тобто він може переходити за один такт з одного стану одразу до кількох станів та находитися у кількох станах одночасно.

Алгоритм Ахо–Корасік (АК) англійською – Aho–Corasick (AC), є прикладом засобу, який на відміну від багатьох відомих алгоритмів одиночного розпізнавання виявляє у вхідних даних одночасно кілька зразків, тобто виконує множинне розпізнавання. На етапі побудови АК з наданого набору патернів за певними правилами створюється ДСА, який під час функціонування розпізнає потрібні патерни. Такий ДСА називають скінченним автоматом Ахо–Корасік (СА-АК) – англійською Aho–Corasick finite automaton (AC-FA)

Теорію та приклади формування СА-АК широко подано в літературі. Однак, в даному дослідженні виявилося корисним додатково проаналізувати функцію переходів f_s СА-АК. В [7] виокремлено чотири типи переходів, які присутні в такому автоматі: *прямі* (direct або goto-переходи), *перехресні* (cross), *хибні* (failure) та *післястартові* (restartable). Строго кажучи, останній тип є різновидом перехресних переходів, але внаслідок особливостей технічної реалізації його відділення в самостійний тип дозволяє спростити поводження з автоматом.

Узагальнену структуру апаратної реалізації СА-АК наведено на рис.7. Його основу складає запам'ятовуючий пристрій (ЗП) в якому зберігається таблиця переходив автомата. Кожна комірка ЗП містить номер наступного стану та вектор збігу. До значення номеру наступного стану, що витягується з пам'яті, шляхом конкатенації додається код символу із вхідній послідовності. Здобуте значення подається на адресний вхід ЗП та обирає відповідний рядок інформації. Вектори збігу містять одиницю у позиції, що позначає відповідний патерн. Керує роботою автомата керуючий пристрій. Якщо при реалізації схеми використовується зовнішня щодо кристалу ПЛІС пам'ять, називатимемо таке рішення *Базовою схемою СА-АК із зовнішньою пам'яттю* або *схемою ACRAM*. Рішення на базі внутрішньої блокової пам'яті ПЛІС будемо називати *Базовою схемою СА-АК із блоковою пам'яттю* або *схемою ACBRAM*.

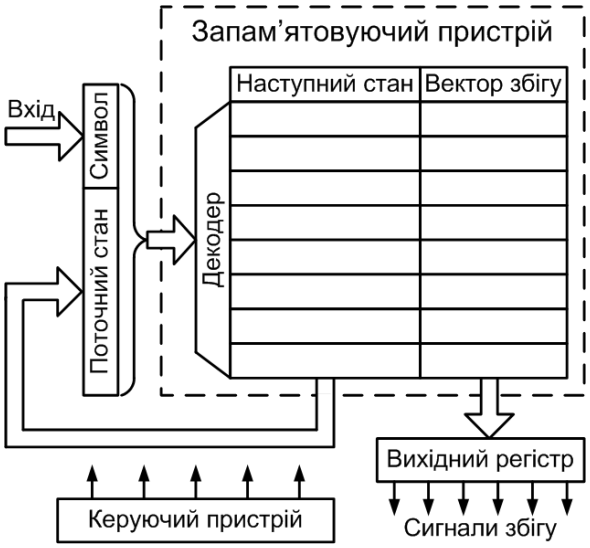

Рис. 7. Узагальнена структурна схема типової реалізації скінченого автомата Ахо–Корасік

Найважливіша перевага рішень на базі СА-АК – незалежність пропускної здатності від об'єму словнику сигнатур та від особливостей патернів, зокрема, від їх довжин, що має наслідком передбачуваність пропускної здатності. Теоретично СА за кожен такт приймає один символ зі вхідної послідовності. Але на практиці якщо розмір таблиці переходив завеликий, і для її зберігання потрібно використовувати зовнішню відносно ПЛІС пам'ять, кожне звернення до ЗП може займати декілька тактів. Крім того, зовнішня пам'ять сама по собі повільніша за внутрішню. Отже, зворотна сторона цієї переваги – відносно низька швидкодія, яку до того ж складно нарощувати, що дещо погіршує масштабованість за пропускною здатністю СА-АК.

Основні складнощі, які виникають при реалізації СА-АК на реконфігуровній платформі, пов'язані з організацією ефективного обміну даними із ЗП. Якщо при побудові схеми ACBRAM за рахунок гнучкості конфігурування блоків BRAM існує можливість синтезувати пристрій пам'яті майже довільної структури, то у випадку ACRAM проблема загострюється, тому що бортова пам'ять РУО має фіксовану архітектуру, не розраховану на ефективну взаємодію з керуючим пристроєм

Як можна бачити, СА-АК потребує незначну кількість ресурсів логіки для створення схеми керування, регістрів та контролера ЗП. Проте об'єм запам'ятовуючого пристрою може сягати дуже великих значень, тобто підхід характеризується високою ресурсоємністтю щодо пам'яті. Збільшення кількості патернів призводить до лавиноподібного зросту ресурсів ЗП, що означає дуже погану масштабованість за об'ємом словнику патернів. Тому переважна більшість досліджень щодо застосування СА-АК в РЗТЗІ спрямована саме на зменшення ресурсів пам'яті та стримування їх швидкого зростання.

Серед численних модифікацій СА-АК є рішення, в яких пропонуються різні способи кодування таблиці переходів, або аналізується збіг не тільки префіксів, а й інфіксів патернів. Але в більшості модифікацій зменшення пам'яті досягається шляхом певного поводження з переходами автомату – окремо для кожного типу. В низці робіт було запропоновано та розвинуто техніку застосування конвеєризації при побудові СА-АК: лінійний конвеєр з H ступенів дозволяє позбавитися всіх основних

перехресних переходів у структурі СА-АК від початкового стану до рівня *H*.

Окрема частка зусиль дослідників була спрямована на покращення не дуже гарних швидкісних ПЕ підходу. Оскільки СА-АК, як будь-який скінченний автомат, обробляє вхідну інформацію послідовно – символ за символом, були здійснені спроби прискорити роботу алгоритму АК за рахунок обробки більш, ніж одного символу за такт. Оскільки заздалегідь невідомо, з яким зсувом питомий патерн опиниться у вхідних даних, потрібно організувати паралельну роботу відповідної кількості однакових автоматів.

Ще одно рішення, пов’язане з небайтовою обробкою даних, так звана схема *Bit-split*, полягає в заміні одного автомата, що обробляє 8-бітні символи на декілька паралельно працюючих підавтоматів, які аналізують 1, 2 або 4 біти. За рахунок зменшення "символів", що обробляються, суттєво скорочується розмір алфавіту. Рішення Bit-split є протилежним щодо схем розпізнавання по кілька символів за такт, але воно спрямоване не на прискорення, а на скорочення ресурсів.

Використання зовнішнього ОЗП забезпечує просту реалізацію динамічної реконфігурації схеми СА-АК шляхом перезапису таблиці переходів, тобто повної зміни алгоритму роботи автомата, не зупиняючи процес функціонування РЗТЗІ. Щодо показника селективного розпізнавання, він також реалізується відносно просто – шляхом формування в пам'яті ЗП кількох таблиць переходів для різних підавтоматів.

Порівняння підходів до побудови реконфігуровних засобів розпізнавання

Результати формалізації опису властивостей схем розпізнавання на базі АП, ФБ та СА зведені до Таблиці 1.

Як свідчить порівняльний аналіз, жоден з досліджених підходів не демонструє явних переваг перед іншими. Кожен має позитивні риси та недоліки, Але жоден не перевершує конкурентні рішення за всіма ПЕ. Так, цифрові компаратори та різновиди асоціативної пам'яті на їх основі забезпечують максимальну швидкодію, але є більш витратними щодо споживання апаратних ресурсів і електроенергії, вони також програють в плані масштабування за об’ємом словнику патернів і за довжиною патернів. Фільтр Блума на базі геш-функцій більш економний та краще масштабується, але накладає обмеження по довжині сигнатур; він також вимагає додаткових витрат на доуточнення результатів роботи через системні помилоки розпізнавання другого роду. Скінчені автомати Ахо–Корасік забезпечують стабільну, але відносно невисоку пропускну здатність, складні в побудові, потерпають від "вибухового" зростання об’єму пам'яті для великих словників сигнатур.

Таблиця 1. Порівняння основних підходів до побудови РЗТЗІ

№	Показник ефективності		Підхід		
			Асоціативна пам'ять	Фільтр Блума	Скінчен. автомат Ахо–Корасік
1.	Витрати логіки		---	+	+++
2.	Витрати пам'яті	розподіленої	---	+	+++
3.		блокової	+++	+	---
4.		зовнішньої	+++	+++	---
5.	Швидкодія		+++	+	-
6.	Передбачуваність пропускної здатності		+++	---	+++

7.	Функціональні показники	здатність протидіяти атакам на РЗТЗІ	+++	---	+
8.		динамічна реконфігурація	---	+	+++
9.		селективне розпізнавання	---	+	+++
10.		режим запобігання вторгнень	+++	-	---
11.	Масштабованість	за пропускною здатністю	+	-	-
12.		за об'ємом словнику	-	+++	---
13.		за довжиною патернів	-	+++	+++
14.	Використання надлишковості		+++	---	+
15.	Суттєвий недолік, який зводить нанівець головні переваги підходу		Завелике споживання ресурсів	Фіксована довжина патернів	Вибухоподібний зріст об'єму пам'яті

Позначення: "+++" – суттєва перевага, "+" – помірна перевага, "---" – суттєвий недолік, "-" – помірний недолік.

Висновки

Отже жоден з досліджених підходів не демонструє явних переваг перед іншими. Тому існує потреба в методах, які дозволили би поєднати різні підходи в єдиному пристрої, максимізуючи ефективність за рахунок реалізації переваг кожного з них.

Література

1. *Палагин А.В., Опанасенко В.Н.* Реконфигурируемые вычислительные системы: Основы и приложения / К.: «Просвіта», 2006. – 280 с.

2. *Гильгурт С.Я.* Реконфигурируемые вычислители. Аналитический обзор // Электронное моделирование. – 2013. – Т.35, № 4. – С. 49-72.

3. *Смит Б.* Методы и алгоритмы вычислений на строках. Теоретические основы регулярных вычислений / Пер. с англ. – М.: Вильямс, 2006. – 496 с.

4. *Chen H., Chen Y., Summerville D.H.* Survey on the Application of FPGAs for Network Infrastructure Security // IEEE Communications Surveys and Tutorials, Article. – 2011. – Vol. 13, №4. – P. 541-561.

5. *Гільгурт С.Я.* Побудова асоціативної пам'яті на цифрових компараторах реконфігуровними засобами для вирішення задач інформаційної безпеки // Електронне моделювання. – 2019. – Т. 41, №3. – С. 59-80.

6. *Гільгурт С.* Побудова фільтрів Блума реконфігуровними засобами для вирішення задач інформаційної безпеки // Безпека інформації. – 2019. – Т. 25, №1. – С. 53-58.

7. *Гільгурт С.* Побудова скінченних автоматів реконфігуровними засобами для вирішення задач інформаційної безпеки // Захист інформації. – 2019. – Т. 21, №2. – С. 111-120.

Гільгурт С.Я.

ПОРІВНЯЛЬНИЙ АНАЛІЗ ПІДХОДІВ ДО ПОБУДОВИ КОМПОНЕНТІВ РЕКОНФІГУРОВНИХ ЗАСОБІВ ТЕХНІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ

У даній роботі досліджуються підходи до апаратної реалізації на базі ПЛІС засобів технічного захисту інформації сигнатурного типу, які в реальному часі вирішують обчислювально ресурсоємну задачу множинного розпізнавання патернів в інтенсивному потоці вхідних даних (системи виявлення вторгнень, засоби протидії вірусам та спаму). З метою оцінки властивостей таких засобів та їх компонентів, а також для порівняння різних підходів до побудови схем розпізнавання визначено та класифіковано потрібні показники ефективності. З'ясовано що найбільш ефективними при побудові апаратних

схем множинного розпізнавання патернів є наступні три підходи та технології, на яких вони ґрунтуються: асоціативна пам'ять на базі цифрових компараторів; фільтр Блума на базі геш-функцій; алгоритм Ахо–Корасік на базі скінченних автоматів. Для кожного з напрямів наведено формалізований на якісному рівні огляд характерних особливостей в сенсі визначених показників ефективності, специфіки його практичної реалізації на ПЛІС, складнощі та проблеми, що при цьому виникають, та шляхи їх усунення. Проведено порівняльний аналіз згаданих підходів. Зроблено висновок, що жоден з них не демонструє явних переваг перед іншими, тому існує потреба в методах, які дозволили би поєднати різні підходи в єдиному пристрої, максимізуючи ефективність за рахунок реалізації переваг кожного з них.

Ключові слова: *технічний захист інформації, сигнатурний аналіз, множинне розпізнавання патернів, ПЛІС, ефективність, порівняльний аналіз.*

Hilgurt S.Ya.

COMPARATIVE ANALYSIS OF APPROACHES TO THE BUILDING OF RECONFIGURABLE SECURITY TOOLS COMPONENTS

This paper investigates the approaches to hardware implementation on the reconfigurable (based on FPGAs) signature-based security tools, which in real time solve computationally resource-intensive problem of multi-pattern matching in an intensive input data stream (intrusion detection systems, anti-virus and anti-spam tools). In order to assess the properties of such tools and their components, as well as to compare different approaches to recognition schemes building, the performance indicators are identified and classified. It was found that the most effective in the construction of hardware schemes for multi-pattern matching are the following three approaches and corresponding underlying technologies: associative memory based on digital comparators; Bloom filter based on hash functions; Aho–Corasik algorithm based on finite automata. A qualitatively formalized review in terms of correspondent indicators is provided for each of these directions. The features of its implementation on FPGA, the difficulties and problems that arise, and ways to eliminate them are investigated as well. A comparative analysis of these approaches was performed. It is concluded that none of them demonstrates clear advantages over others, so the methods to combine different approaches in a single device, maximizing efficiency by realizing their advantages, are needed.

Keywords: *information security, signature analysis, multi-pattern matching, FPGA, effectiveness, comparative analysis.*