УДК 004.274:004.056

В.Ф. Евдокимов, Киев
А.Н. Давиденко, Киев
С.Я. Гильгурт, Киев

ДОПОЛНИТЕЛЬНЫЕ ЭТАПЫ ПРОЦЕДУРЫ ОПЕРАТИВНОЙ РЕКОНФИГУРАЦИИ АППАРАТНЫХ УСКОРИТЕЛЕЙ ЗАДАЧ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Abstract. A procedure of dynamic resynthesizing reconfigurable information security accelerators is analyzed. The preprocessing and post-processing steps are investigated. The partitioning of large intrusion detection pattern databases and the elimination of false positives of Bloom filters are considered as examples of preprocessing and post-processing operations respectively.

Введение

В последние годы для создания систем киберзащиты, в частности, сетевых систем обнаружения вторжений (ССОВ), соответствующий англоязычный термин – Network Intrusion Detection System (NIDS), все чаще используют аппаратные решения. При этом в качестве платформы разработчики отдают предпочтение реконфигурируемым ускорителям на базе ПЛИС [1, 2]. Близкое к аппаратному быстродействие программируемой логики в сочетании с ее высокой гибкостью наиболее полно отвечает требованиям такой динамично развивающейся области, как защита информации. Сказанное особенно актуально для задачи множественного распознавания (Multi-Pattern String Matching) – наиболее ресурсоемкой операции, выполняемой в сигнатурных системах киберзащиты [3, 4].

К сожалению, широкое распространение реконфигурируемых средств сдерживается рядом факторов, обусловленных как особенностями самой программируемой логики, так и спецификой реализации сигнатурных распознающих систем.

В предыдущих работах авторов [5 – 8] была предложена и развита концепция системы централизованного синтеза аппаратных ускорителей для решения задач информационной безопасности с использованием распределенной суперкомпьютерной сети. Для проверки идеи о целесообразности переноса ресурсоемкой операции создания аппаратных схем с локальных устройств киберзащиты на высокопроизводительные компьютерные системы был создан экспериментальный грид-сервис для реконфигурируемых ускорителей задач информационной безопасности STRAGS (Security Tasks Reconfigurable Accelerators Grid-Service).

Специфической особенностью применения реконфигурируемых вычислителей для аппаратного ускорения систем обнаружения вторжений и других сигнатурных средств защиты информационных объектов является регулярно

возникающая необходимость повторного синтеза некоторых аппаратных компонентов. В качестве причины потребности в подобной процедуре *оперативной реконфигурации* может выступить либо обновление базы данных сигнатур, либо изменения условий работы защищаемого объекта, например, модификация локальной сети, изменения ее структуры и состава, переустановка программного обеспечения и т.п.

Следует заметить, что оперативная реконфигурация не приводит к необходимости проведения полного цикла создания цифрового устройства. Поэтому процесс синтеза можно упростить, сократив большую часть этапов разработки, которые могут быть реализованы заранее. В результате, как показано в работе [8], вся технологическая цепочка синтеза цифровой схемы распознавания сводится всего лишь к двум этапам. На первом этапе по исходному набору сигнатур компонуется вычислительная структура в виде текста на языке описания аппаратуры (HDL-языке от англ. Hardware Definition Language). На втором – вычислительные модули в автоматическом режиме преобразуются в загружаемые в ПЛИС конфигурации.

Однако на практике при создании аппаратных ускорителей информационной безопасности возникает необходимость в дополнительных программных компонентах, выполняющих как функции предварительной обработки исходных данных (препроцессинг), так и некоторые завершающие операции (постпроцессинг).

Целью данной работы является более полное рассмотрение цепочки преобразования информации на пути от исходной базы данных сигнатур к загружаемым в ПЛИС файлам конфигурациям (bitstream), реализуемой в процессе оперативной реконфигурации аппаратных ускорителей задач информационной безопасности, а также в процессе их функционирования.

Прежде чем перейти непосредственно к исследованию операций препроцессинга и постпроцессинга, уточним некоторые положения, связанные с аппаратной реализацией сетевых систем обнаружения вторжения сигнатурного типа.

1. Сетевые СОВ

В публикации [9] проведено исследование и обобщение основных принципов построения систем обнаружения вторжений на базе программмируемой логики по результатам анализа накопленного в мире опыта применения ПЛИС типа FPGA в составе ССОВ.

Сетевые системы обнаружения вторжений, разрабатываемые в настоящее время, в зависимости от метода анализа событий подразделяются на два основных класса:

– выявляющие аномалии (Statistical anomaly based IDS);

– использующие сигнатуры (Signature-based IDS).

Системы выявления аномалий в отличие от подходов, использующих сигнатуры, способны обнаруживать новые, неизвестные ранее атаки. Однако на сегодняшний день они приводят к недопустимо большому количеству

ошибок распознавания как первого, так и – особенно – второго рода, то есть допускают неприемлемо много ложных срабатываний [10]. Поэтому в большинстве анализируемых источников под сетевыми СОВ понимаются системы, основанные на распознавании сигнатур. В данном исследовании также рассматриваются только системы такого класса.

Механизм функционирования сетевой системы обнаружения вторжений в общем случае состоит из 3-х этапов:

– захват сетевых пакетов (packet capture);

– фильтрация и сборка пакетов (filtering / fragmentation reassembly);

– распознавание (pattern matching).

Самым ресурсоемким является последний этап, который сводится к выполнению большого количества операций сравнения содержимого сетевых пакетов с паттернами – последовательностями символов из базы данных сигнатур.

Анализ сетевого трафика может осуществляться двумя способами:

– путем тотального захвата и инспектирования всех (необработанных – raw) пакетов сетевого трафика;

– с учетом сетевых протоколов – stateful подход, основанный на анализе заголовков сетевых пакетов с целью полного восстановления сеансов сетевого обмена.

Системы, основанные на первом способе, распознают большее число атак. Для них не являются проблемой нестандартные номера портов, потерянные и намеренно искаженные злоумышленником сетевые пакеты. С другой стороны, такие ССОВ намного более ресурсоемки в своей реализации.

2. Подходы к построению аппаратных схем распознавания.

Как показывает анализ информационных источников, в существующих на сегодняшний день системах аппаратного распознавания задействованы разнообразные подходы, приемы и технические решения. Наиболее распространенными из них являются [11]:

– ассоциативная память на параллельных дискретных компараторах и ее разновидности [12 – 14];

– схемы на базе хэш-функций, в частности, фильтр Блума [15 – 18];

– конечные автоматы (finite automaton), реализующие в большинстве случаев алгоритм Ахо-Корасик [19 – 22].

Как показывает сравнительный анализ, ни один из подходов не демонстрирует явных преимуществ перед другим и не удовлетворяет в полной мере требованиям, предъявляемым сетевыми системами обнаружения вторжений к их реализации на реконфигурируемых устройствах.

Например, фильтр Блума за счет применения функций хеширования позволяет уменьшить число сравнений, что дает возможность снизить аппаратные затраты без потери быстродействия, но этой схеме присущи системные ошибки распознавания второго рода, что требует дополнительного

доуточнения результатов распознавания. При некоторых условиях аппаратное выполнение данной вспомогательной функции становится нецелесообразным, поэтому она переносится на дополнительный программный модуль постпроцессинга, включаемый в состав аппаратно-программного комплекса киберзащиты. Функциональность и параметры данного модуля необходимо конкретизировать на этапе синтеза цифровой схемы аппаратного ускорителя после того, как будет выбран подход к построению блока сигнатурного распознавания, который зависит от исходных данных (состав набора сигнатур, тип ПЛИС, а также дополнительные опции, задаваемые пользователем).

3. Параллелизм задачи множественного распознавания

Как указывалось в работе [23], задаче множественного распознавания строк присущ внутренний параллелизм, обусловленный эффектом самоподобия, за счет которого большое число сигнатур можно распознавать одновременно. Данный эффект заключается в том, что многочисленные паттерны, входящие в состав сигнатур (которых в современных ССОВ может насчитываться нескольких десятков тысяч, а для антивирусов – нескольких сотен тысяч и даже миллионов), в значительной степени повторяют друг друга, обладая либо общими префиксами, либо суффиксами, либо инфиксами. Перечисленные в предыдущем разделе подходы позволяют в значительной степени его задействовать. Усилить эффект распараллеливания (с целью улучшения показателей быстродействия и экономичности) можно путем кластеризации исходного набора распознаваемых сигнатур, в результате чего они разбиваются на группы, в каждой из которых свойство самоподобия проявляется наиболее выражено.

Данная задача кластеризации словаря сигнатур для конкретного подхода построения модуля распознавания может рассматриваться в качестве одной из возможных процедур препроцесинга.

4. Кластеризация базы данных сигнатур

В работах [24, 25] рассматриваются варианты решения задачи кластеризации (partitioning) применительно к схемам ассоциативной памяти на цифровых компараторах. При создании аппаратных систем обнаружения и предотвращения вторжений наиболее эффективно выполнить кластеризацию базы данных сигнатур позволяет использование методов теории графов. Отмечается, что за счет разбиения процесса распознавания на подпроцессы, одновременно обрабатывающие несколько групп паттернов, удается достичь ускорения от двух до восьми раз.

При таком подходе база данных сигнатур сначала преобразуется в граф, отражающий свойство самоподобия входящих в нее записей: паттерны задают вершины графа; дуги связывают каждую пару вершин, в паттернах которых имеются одинаковые символы. Разбиение на группы производится таким образом, чтобы максимизировать число дуг внутри группы,

минимизируя при этом внешние связи между группами. В качестве инструмента решения данной задачи может быть использована, например, программная библиотека кластеризации графов METIS [26].

Следует заметить, что при таком подходе граф получается очень тесносвязанным. (В экспериментальных разработках для 361 вершины получалось порядка 40000 дуг.) По мере роста числа сигнатур наступает момент, паттерны невозможно разбить на группы так, чтобы, элементы каждой не содержали бы полный набор символов алфавита, что приводит к невозможности непосредственного использования данного алгоритма для словарей объемом примерно в 500 паттернов и более.

Модифицировать подход таким образом, чтобы сделать его применимым для больших размеров наборов сигнатур, позволяет добавление к дугам весовых функций, учитывающих не только наличие одинаковых символов, но и их местоположение в паттерне [25]. Такое взвешивание позволяет алгоритму кластеризации более тесно группировать паттерны, содержащие совпадающие подстроки, поскольку весовая функция слабо реагирует на значительно отличающиеся сигнатуры, но существенно возрастает при наличии совпадений символов по местоположению.

5. Обобщенная схема построения реконфигурируемой структуры

Как неоднократно указывалось ранее, базы данных сигнатур современных ССОВ насчитывают многие тысячи записей. При реализации модуля распознавания на ПЛИС все они "прошиваются" в вычислительную структуру на аппаратном уровне [5]. Очевидно, что разработку настолько сложной схемы нецелесообразно выполнить вручную, не столько из-за трудоемкости, сколько из-за недопустимо высокого числа ошибок, неизбежно возникающих при этом [27]. К счастью, все подходы, использующиеся в настоящее время при построении систем аппаратного распознавания строк, приводят к созданию достаточно регулярных структур, представление которых на языке описания аппаратуры может быть автоматизировано посредством специально созданного программного обеспечения. Пример подобной разработки можно найти в публикации [28].

С учетом вышесказанного представим полную обобщенную схему процесса построения реконфигурируемой структуры аппаратного устройства информационной безопасности (рисунок).

Процесс начинается с процедуры анализа базы данных сигнатур, которые должны распознаваться создаваемой системой защиты. В большинстве научных исследований по реконфигурируемым ССОВ в качестве подобной базы используются наборы сигнатур свободно распространяемых систем обнаружения вторжений, таких как Bro, Hogwash, Snort либо Suricata [29 – 32]. Однако в большинстве случаев исследователи используют в качестве устоявшегося стандарта де-факто набор сигнатур от Snort [33, 34]. Такой подход позволяет, с одной стороны, приблизить проводимые исследования к реальной жизни, благодаря распространенности

и интенсивному практическому применению данной системы. С другой стороны, использование одинаковых наборов исходных данных дает возможность сравнить различные методы, алгоритмы и решения для ССОВ, объективно оценить достигнутые ими технические показатели.

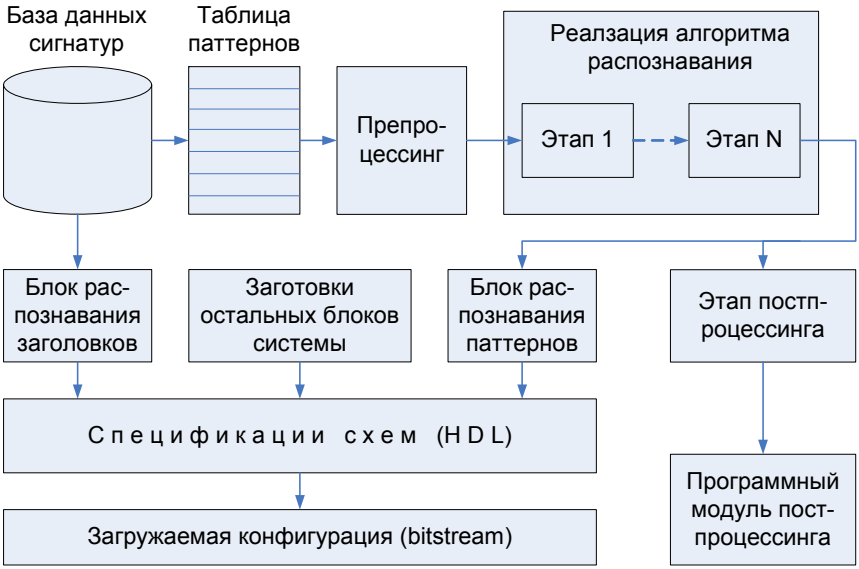

Рис. Этапы синтеза реконфигурируемой структуры ССОВ

В работе [34] подробно рассмотрена структура записей базы данных сигнатур ССОВ Snort. Напомним, что под понятием *сигнатура* мы понимаем всю совокупность информации о конкретной угрозе, содержащуюся в одной записи базы данных Snort. В этом смысле понятие *запись базы данных* эквивалентно термину «сигнатура». Уточним также, что под словом *паттерн* мы понимаем образец текстовой строки (последовательность символов, закодированных байтами), входящий в состав сигнатуры, который ищется в теле сетевого пакета, анализируемого системой.

В общем случае сигнатура содержит один или несколько паттернов, а также вспомогательные правила, регламентирующие условия поиска паттернов. Для сетевых систем обнаружения вторжений по данным правилам обрабатываются заголовки сетевых пакетов при анализе трафика в режиме учета сетевых протоколов (см. раздел 1). В результате анализа правил базы данных сигнатур генерируется схема Блока распознавания заголовков.

Одновременно в процессе анализа базы данных сигнатур формируется актуальная таблица паттернов, над которой затем производится процедура препроцессинга, например, как описано выше в разделе 4.

Дальнейшие этапы синтеза зависят от способа распознавания, выбран-

ного в качестве основы в данной ССОВ. В общем случае они состоят из нескольких шагов различной сложности и ресурсоемкости. В результате генерируется схема Блока распознавания паттернов.

Затем к вновь сгенерированным блокам добавляются фиксированные схемные заготовки остальных компонентов, не зависящих от набора распознаваемых правил [35]. Далее, для полученных вычислительных структур составляются спецификации синтезируемых схем на одном из HDL-языков описания аппаратуры.

На завершающей стадии HDL-описание дополняется как файлами ограничений (ucf-файлами и т.п.), так и другими компонентами проекта создания цифрового устройства, после чего запускается процедура автоматической генерации загружаемых в ПЛИС конфигураций, например, с применением фирменного пакета САПР производителя микросхем программируемой логики, используемых в ускорителе.

Программный модуль постпроцессинга выбирается, подключается и конфигурируется на этапе реализации алгоритма распознавания и в зависимости от результатов выполнения этого этапа.

В заключение отметим различие времени жизни рассмотренных в данном исследовании этапов пре- и постпроцессинга. Если препроессинг осуществляется во время динамической реконфигурации, инициированной изменениями условий эксплуатации системы киберзащиты, то постпроцессинг выполняется в процессе технической эксплуатации системы, притом, что настройки и сам факт наличия данного этапа определяются в период реконфигурации.

Выводы

В данной роботе на основе анализа мирового опыта в области разработок средств киберзащиты на базе ПЛИС исследован в общем виде процесс построения реконфигурируемой структуры аппаратного устройства информационной безопасности. Данный процесс включает в себя помимо прочего этап предварительной обработки базы данных сигнатур – препроцессинг. Завершающий этап преобразования информации данных в сигнатурной системе распознавания – постпроцессинг – выполняется в реальном времени в процессе ее функционирования.

1. *Paxson V., Asanovic K., Dharmapurikar S., Lockwood J., etc.* Rethinking Hardware Support for Network Analysis and Intrusion Prevention // USENIX First Workshop on Hot Topics in Security (HotSec), Vancouver, B.C., July 31, 2006.
2. Реконфигурируемые вычислительные системы: Основы и приложения / А.В. Палагин, В.Н. Опанасенко. – К.: «Просвіта», 2006. – 280 с.
3. Методы и алгоритмы вычислений на строках / Б. Смит / Пер. с англ. – М.: Вильямс, 2006. – 496 с.
4. *Гільгурт С.Я.* Множинне розпізнавання рядків у системах виявлення вторгнення на базі реконфігуровних обчислювачів // Сучасні комп'ютерні системи та мережі: розробка та використання: матеріали 5-ої Міжнар. наук.-техн. конф. ACSN-2011, 29

вересня – 01 жовтня 2011, Львів, Україна. – Л.: Вид-во Нац. ун-ту «Львів. політехніка», 2011. – С.54-56
5. *Гильгурт С.Я.* Организация вычислительного процесса синтеза файлов конфигураций для аппаратных ускорителей при решении задач информационной безопасности // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2015. – Вип. 74. – Київ. – С.29-33.
6. *Євдокимов В.Ф., Давиденко А.М., Гільгурт С.Я.* Створення на базі грід-сайту ІПМЕ ім. Г.Є. Пухова НАНУ системи централізованого синтезу апаратних прискорювачів для вирішення задач інформаційної безпеки в енергетичній галузі // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2017. – Вип. 79. – С.3-8.
7. *Евдокимов В.Ф., Давиденко А.Н., Гильгурт С.Я.* Организация централизованной генерации файлов конфигураций для аппаратных ускорителей задач информационной безопасности // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2017. – Вип. 81. – С.3-11.
8. *Евдокимов В.Ф., Давиденко А.Н., Гильгурт С.Я.* Централизованный синтез вычислительных структур реконфигурируемых ускорителей задач информационной безопасности // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2018. – Вип. 82. – С.3-11.
9. *Коростиль Ю.М., Гильгурт С.Я.* Принципы построения сетевых систем обнаружения вторжений на базе ПЛИС // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2010. – Вип. 57. – С.87–94.
10. *Abdulhammed R., Faezipour M., Elleithy K.M.* Network Intrusion Detection Using Hardware Techniques: A Review // IEEE Long Island Systems, Applications and Technology Conference (LISAT '16), April 2016.
11. *Гильгурт С.Я.* Аппаратное распознавание строк в интеллектуальных системах защиты информации // Штучний інтелект. – 2012. – № 1. – С.259-266.
12. *Yu F., Katz R.H., Lakshman T.V.* "Gigabit rate packet pattern matching using TCAM" Network Protocols, 2004. ICNP 2004. Proceedings of the 12th IEEE International Conference on, 2004, pp. 174–183.
13. *Sourdis I., Pnevmatikatos D.* "Pre-decoded cams for efficient and high-speed NIDS pattern matching" Field-Programmable Custom Computing Machines, 2004. FCCM 2004. 12th Annual IEEE Symposium on, 2004, pp. 258–267.
14. *Yusuf S., Luk W.* "Bitwise optimized CAM for network intrusion detection systems" Field Programmable Logic and Applications, 2005. International Conference on, 2005, pp. 444–449.
15. *Dharmapurikar S., Lockwood J.W.* "Deep packet inspection using parallel bloom filters" Micro, IEEE, vol. 24, no. 1, pp. 52–61, 2004.
16. *Dharmapurikar S., Lockwood J.W.* "Fast and scalable pattern matching for network intrusion detection systems" Selected Areas in Communications, IEEE Journal on, vol. 24, no. 10, pp. 1781–1792, 2006.
17. *Suresh D.C., Guo Z., Buyukkurt B., Najjar W.A.* "Automatic compilation framework for bloom filter based intrusion detection" Lecture notes in computer science. Berlin; New York: Springer-Verlag, 2006, pp. 413–418.
18. *Baker Z.K., Prasanna V.K.* "High-throughput linked-pattern matching for intrusion detection systems" Architecture for networking and communications systems, 2005. ANCS 2005. Symposium on, 2005, pp. 193–202.
19. *Aho A.V., Corasick M.J.* Efficient string matching: an aid to bibliographic search. – II

Communications of the ACM, 1975. – Vol. 18, № 6. – P. 333–340.
20. *Bos H., Huang K.* "A network instruction detection system on ixp1200 network processors with support for large rule sets" Technical Report 2004-02. Leiden University, 2004.
21. *Alicherry M., Muthuprasanna M., Kumar V.* "High speed pattern matching for network IDS/IPS" ICNP '06: Proceedings of the 2006 IEEE International Conference on Network Protocols. IEEE Computer Society, 2006, pp. 187–196.
22. *Jiang W., Prasanna V.* Scalable Multi-Pipeline Architecture for High Performance Multi-Pattern String Matching // IEEE International Parallel and Distributed Processing Symposium (IPDPS '10), April 2010.
23. *Гильгурт С.Я., Коростиль Ю.М., Дурняк Б.В., Билак Ю.Ю.* Анализ реализации множественного распознавания строк на реконфигурируемых устройствах // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2011. – Вип. 60. – С.71–77.
24. *Baker Z.K., Prasanna V.K.* A Methodology for Synthesis of Efficient Intrusion Detection Systems on FPGAs, IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM), April 2004.
25. *Baker Z.K., Prasanna V.K.* Automatic Synthesis of Efficient Intrusion Detection Systems on FPGAs. IEEE Trans. Dependable Sec. Comput. 3(4): 289-300 (2006).
26. *Karypis G., Aggarwal R., Schloegel K., Kumar V., etc.* "METIS Family of Multilevel Partitioning Algorithms," http://www-users.cs.umn.edu/~karypis/metis/, 2004.
27. *Sourdis I., Pnevmatikatos D.* Fast, large-scale string match for a network intrusion detection system // Proceedings of 13th International Conference on Field Programmable Logic and Applications, 2003.
28. *Mitra A., Najjar W., Bhuyan L.* Compiling PCRE to FPGA for accelerating SNORT IDS, Proceedings of the 3rd ACM/IEEE Symposium on Architecture for networking and communications systems // December 03-04, 2007, Orlando, Florida, USA.
29. The Bro Network Security Monitor [Электронный ресурс]. – Режим доступа: http://www.bro-ids.org. – Загл. с экрана. – (Дата обращения: 15.09.2018).
30. Welcome to Hogwash [Электронный ресурс]. – Режим доступа: http://hogwash.sourceforge.net/oldindex.html. – Загл. с экрана. – (Дата обращения: 15.09.2018).
31. SNORT [Электронный ресурс]. – Режим доступа: http://snort.org. – Загл. с экрана. – (Дата обращения: 15.09.2018).
32. Suricata. Open Source IDS / IPS / NSM engine [Электронный ресурс]. – Режим доступа: http://suricata-ids.org. – Загл. с экрана. – (Дата обращения: 15.09.2018).
33. *Давиденко А.Н., Гильгурт С.Я., Сабат В.И.* Аппаратное ускорение алгоритмов сигнатурного обнаружения вторжений в открытой системе информационной безопасности Snort // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2012. – Вип. 65. – С.94–103.
34. *Коростиль Ю.М., Гильгурт С.Я., Назаренко О.М.* Анализ базы данных системы информационной безопасности Snort и вопросы быстродействия // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ ім. Г.Є. Пухова НАН України. – Київ, 2012. – Вип. 66. – С.77-84.
35. *Katashita T., Yamaguchi Y., Maeda A., Toda K.* FPGA-Based Intrusion Detection System for 10 Gigabit Ethernet // IEICE - Transactions on Information and Systems, v.E90-D n.12, p.1923-1931, December 2007.

Поступила 20.09.2018р.