УДК 004.056

АРХИТЕКТУРА КЛАССИФИКАЦИИ СЕТЕВЫХ ПАКЕТОВ НА ОСНОВЕ ИСПОЛЬЗОВАНИЯ FPGA

Ташев К.А.

В настоящее время системы обнаружения сетевых вторжений/атак являются неотъемлемой частью современных информационно-коммуникационных систем. Потому что, именно они защищают от внешних влияний информационную инфраструктуру для создания условий эффективного функционирования. При этом классификация пакетов с несколькими совпадениями является важной функцией в сетевых системах обнаружения вторжений (NIDS), где необходимо сообщать о всех правилах соответствия для пакета. В данной работе приведены проблемы, связанные с классификацией заголовок пакетов и фильтрации контента. Но при этом рассмотрена проблема связанное с повышением эффективности классификации сетевых пакетов. Анализированы работы, посвященные к проблемам классификацию сетевых пакетов и реализация их на платформе FPGA. Но большинство предыдущих работ основаны на тройной ассоциативной памяти (ternary content addressable memory - TCAM), которые являются дорогостоящими и не масштабируемы в отношении тактовой частоты, энергопотребления и области схема техники. В этой статье изучаются характеристики реальных наборов правил Snort NIDS и предлагается новая архитектура на основе SRAM. Предложенная архитектура называется параллельным битовым вектором с разбиением на подполя, где некоторые поля заголовка пакета дополнительно разделяются на подполя уровня бит. В отличие от алгоритмов классификации пакетов с несколькими совпадениями таких как BV-TCAM, TCAM-SSA, MX-MN-IP и BV, которые страдают из-за переполнения памяти, требования к памяти предложенного алгоритма линейна по количеству правил. Кроме того, предложена использования технологии FPGA для обеспечения высокой пропускной способности и поддержки динамических обновлений. Приведены рекомендации по внедрения, что показывают предложенная архитектура может хранить на одном FPGA Xilinx Virtex-5 полный набор правил заголовков пакетов, извлеченных из последних Snort NIDS, и поддерживает пропускную способность 100 Гбит/с для пакетов минимального размера (40 байт). Приведены результаты сравнения производительности алгоритма на платформе FPGA фирмы XILINX семейства Virtex.

Ключевые слова: системы обнаружения атак, сетевые пакеты, классификация пакетов, Snort NIDS, FPGA, SRAM

Ҳозирги кунда тармоқ суқилиб киришлари/хужумларини аниқлаш тизимлари замонавий ахборот коммуникация тизимларининг ажралмас қисми бўлиб ҳисобланади. Чунки айнан улар ахборот инфратузилмасига самарали ишлаши учун шарт-шароитларини яратиш мақсадида ташқи таъсирлардан ҳимоялайди. Бунда бир нечта мосликлар асосида тармоқ пакетларини классификациялаш хужумларни аниқлашнинг тармоқ тизимларининг (NIDS) муҳим функцияларидан бири ҳисобланади. Чунки бунда тармоқ пакети учун мосликларнинг барча қоидалари ҳақида маълумот бериши зарур бўлади. Ушбу ишда тармоқ пакетлари сарлавҳаларини классификациялаш ва контентни фильтрлаш билан боғлиқ муаммолар келтирилган. Лекин ушбу ишда тармоқ пакетларини классификациялаш самарадорлигини ошириш билан боғлиқ муаммолар кўриб чиқилган. Тармоқ пакетларини классификациялаш ва уларни FPGA платформасида амалга ошириш масалаларига боғлиқ ишлар таҳлил қилинган. Лекин аввалги ишларнинг асосий қисми учталик ассоциатив хотирага (ternary content addressable memory - TCAM) асосланган ва уларнинг тактли частотаси, энерготаъминоти ва схемотехника соҳасига нисбатан таннархининг қимматлилиги ва масштабланмаслиги ажралиб туради. Ушбу мақолада Snort NIDSнинг реал қоидалар тўпламининг характеристикалари ўрганилади ва SRAM асосида самарали архитектура таклиф этилади. Таклиф этилган архитектура қисммайдонларга ажритишли параллель битли вектор деб номланиб, унда тармоқ пакетларининг сарлавҳалари қўшимча тарзда бит даражасида қисммайдонларга ажратилади. BV-TCAM, TCAM-SSA, MX-MN-IP ва BV каби хотирани тўлиб тошишидан азият чекувчи бир нечта мосликли тармоқ пакетларини классификациялаш алгоритмларидан фарқли томони бу таклиф этилган алгоритмда хотирага бўлган талаб қоидаларнинг сонига кўра чизиқлидир. Ундан ташқари, юқори ўтказувчанлик хусусиятини таъминлаш ва динамик янгиланишни қўллаб қувватлаш учун FPGA технологиясини қўллаш таклиф этилган. Тадбиқ этиш учун келтирилган тавсиялар шуни кўрсатадики, таклиф этилган архитектура ягона FPGA Xilinx Virtex-5да Snort NIDSнинг охирги версиясидан олинган тармоқ пакетлари сарлавҳаларининг тўлиқ қоидалар тўпламини сақлаши мумкин ва тармоқ пакетларининг минималь ўлчами (40байт) учун 100Гбит/с ўтказувчанлик хусусиятини қўллаб қувватлайди. Шунингдек XILINX фирмасининг Virtex оилассига мансуб FPGA платформасида алгоритмни унумдорлигини таққослаш натижалари келтирилган.

Таянч иборалар: хужумларни аниқлаш тизимлари, тармоқ пакети, пакетларни классификациялаш, Snort NIDS, FPGA, SRAM

Nowadays, network intrusion/attack detection systems are an integral part of modern information and communication systems. Because exactly network intrusion/attack detection systems protect the information infrastructure from external influences to create conditions for effective functioning. In this case, packet classification with multiple matches is an important function in network intrusion detection systems (NIDS), where all packet matching rules must be reported. This scientific article shows the problems related to packet header classification and content filtering. But at the same time, the problem associated with improving the efficiency of the classification of network packets was considered. Analyzed the works devoted to the problems of the classification of network packets and their implementation on the FPGA platform. But most of the previous works are based on ternary content addressable memory (TCAM), which are expensive and not scalable in terms of the clock frequency, power consumption, and circuit technology area. Also, this scientific article explores the characteristics of real Snort NIDS rule sets and proposes a new architecture based on SRAM. The proposed architecture is called a parallel bit vector split into subfields, where some packet header fields are further divided into bit level subfields. Unlike packet classification algorithms with several matches such as BV-TCAM, TCAM-SSA, MX-MN-IP and BV, which suffer from memory overflow, because the memory requirements of the proposed algorithm are linear in the number of rules. In addition, the proposed use of FPGA technology to ensure high bandwidth and support for dynamic updates. The implementation recommendations are shown that the proposed architecture can store the full set of packet header rules extracted from the latest Snort NIDS on one Xilinx Virtex-5 FPGA and supports 100 Gbps throughput for minimum packet size (40 bytes). The results of the comparison of the algorithm on the XILINX FPGA platform of the Virtex family are given.

Keywords: intrusion detection system, network packets, packet classification, Snort NIDS, FPGA, SRAM.

I. ВВЕДЕНИЕ

С развитием сети Интернет, началось широкое применение информационно-коммуникационных технологий (ИКТ) в различные сферы экономики. В итоге любая сфера человеческой деятельности стало бурно развиваться и формировать гигантский объем данных. К сожалению, на основе использования современных ИКТ технологий появились и информационные угрозы, которые негативно влияют на функционирования информационно-коммуникационных систем. Тем более, сетевые атаки в настоящее время имеют преимущества такие, как требование мало усилий и затрат для создания, трудное отслеживание, и могут быть запущены практически из любой точки мира [6]. Для устранения таких угроз обычно используется системы обнаружения атак, которые делятся в основном на два типа: сетевые и хостовые. Сетевая система обнаружения вторжений (Network

Intrusion Detection system - NIDS) [1,2] является важным средством сетевой безопасности, которое функционирует для защиты высокоскоростных компьютерных сетей от сетевых угроз [14]. Системы такого рода контролирует сетевые коммуникации, идентифицируют шаблоны атак, а затем выполняют действия по установленной политике: или прекращение соединения, или оповещение системного администратора. В настоящее время на ИТ-рынке существует много систем обнаружения атак/вторжений (коммерческие и некоммерческие) и в данной статье не рассматривается коммерческие системы из-за закрытой информации. Вместо этого, используется некоммерческая система с открытым кодом, которая называется Snort [2]. Система Snort, использует несколько тысячи правил, которые содержат шаблоны атак/вторжений. При этом каждое правило имеет две части: заголовок правила и параметр правила. Заголовок правила является фильтром классификации, состоящий из пяти фиксированных полей: 1) Prtcl – протокол; 2) SA (source address) – IP-адрес источника; 3) DA (destination address) – IP-адрес назначения; 4) SP (source port) – порт источника; 5) DP (destination port) – Порт назначения.

Параметр правила указывает шаблоны атак/вторжений, используемые для сканирования полезной нагрузки пакетов (payloads). Результаты классификации заголовка идентифицируют связанные параметры правил, которые будут проверяться при сопоставлении последующего шаблона.

Введение в правила Snort

На рис. 1 показано пример правила Snort, где раздел вложен в круглые скобки, является характеристикой правила, а оставшаяся часть является заголовком правила. В классификации пакетов с несколькими совпадениями 32-битные IP-адреса источника и назначения (обозначенные SA/DA), 16-битные номера портов источника и назначения (обозначенные как SP/DP) и 8-битные поля протоколов (обозначенные Prtcl) из заголовка правила системы Snort сопоставляется по заголовку входного пакета.

IP-адреса в полях SA/DA указаны как префиксы, которые могут представлять любую сеть или один хост. Номера портов в полях SP/DP могут содержать или одно число, или диапазон. Поле протокола в Snort имеет только четыре значения: tcp, udp, icmp и ip. Для полей SA/DA и SP/DP Snort поддерживает задание списка значений, вставленных в квадратные скобки. Snort также предоставляет оператор отрицания «!». Например, ![60,80] указывает любой номер порта, кроме 60 и 80. Кроме того, Snort использует «EXTERNAL_NET» как неявное отрицание «HOME_NET» в полях SA/DA [29].

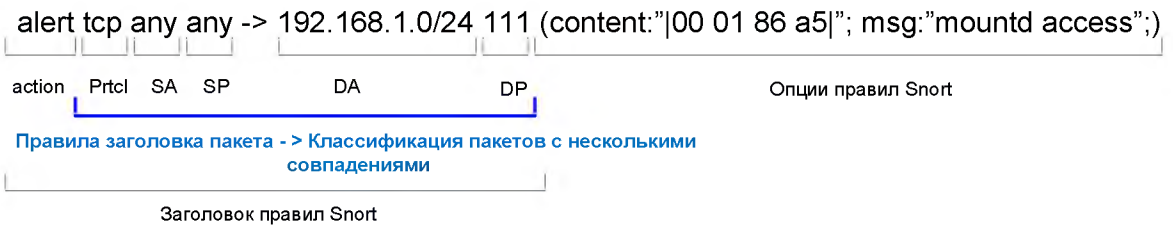


Рис.1. Пример структуры правил Snort

Исходя из этого, можно указать, что в сетевых системах обнаружения атак при анализе угроз существует две проблемы:

Проблемы по фильтрацию заголовка сетевых пакетов.

Проблемы по анализу контента сетевого пакета.

В данной работе рассматривается только первая проблема и предлагается решения в данном направлении.

Анализ предыдущих работ по классификации пакетов

Пакет может соответствовать нескольким заголовкам правил. Традиционным сетевым приложениям, такие как межсетевой экран требует отчетность только совпадающих правил с наивысшим приоритетом, которое называется лучшим совпадением (best matching) классификации пакетов [8,9,12,15]. Напротив, NIDS требует классификации пакетов с несколькими совпадениями, чтобы обнаружит все заголовки правил, которые соответствуют данному пакету [13, 17, 18]. В данной работе основное внимание уделяется классификации заголовков пакетов с несколькими совпадениями в NIDS. Решения для сопоставления шаблонов по пакетным нагрузкам не рассматриваются.

Классификация пакетов хотя и является широко изученной областью исследований [8, 12, 15], большинство предыдущих работ была сфокусирована на лучшие совпадения классификации пакетов, и очень мало работ посвящена к проблемам классификации пакетов с несколькими совпадениями [18]. При бурном росте сетевого трафика требуется классификации пакетов с несколькими совпадениями, которые должны выполняться в аппаратном обеспечении. Например, если пропускная способность продвинута до 100 Гбит/с, то требуется обработка пакета каждые 3.2 нс при минимальном размере пакета 40 байтов. Необходимо отметить, что существующие работы по классификации пакетов с несколькими совпадениями в основном основаны на тройной ассоциативной памяти (TCAM), которые могут выполнять быстрый параллельный поиск по всем записям за один такт, которые являются дорогостоящими и не масштабируемыми в отношении тактовой частоты, энергопотребления или области схемы по сравнению со статическими запоминающими устройствами с произвольным доступом (SRAM) [9].

Исходя из вышеизложенного, на данной работе рассматривается отображение современных алгоритмов классификации пакетов с несколькими совпадениями на SRAM-основанную архитектуру. Алгоритм параллельного битового вектора (BV) и его варианты [5,10,13] являются одним из немногих существующих алгоритмов классификации пакетов, которые поддерживают одновременное возвращение всех правил сопоставление для пакета. Алгоритм BV сначала выполняет параллельный поиск на каждом отдельном поле заголовка пакета. Поиск на каждом поле возвращает бит-вектор с каждым битом, представляющий правила. Бит получает значения «1», если соответствующее правила совпадает в этом поле; а если нет бит получает значения «0». Результат побитовой операции И(AND) на этих битовых векторах дает набор правил, что совпали данному пакету. Алгоритм BV может обеспечить высокую пропускную способность за счет низкой эффективности памяти. Учитывая N правил с D полями, так как проекция N правил на каждое поле может иметь U=O(N) уникальные значения и каждое значение соответствует одному N-битовому вектору, общая требования к памяти алгоритмов BV не менее U*N*D=O(N2), что нежелательно [8, 15]. Поэтому, предлагается разделить W-битное поле на подполя W, где каждое подполе принимает только 1 бит, количество уникальных значений в каждом подполе будет не более 2, то есть значение подполя ∈ [0,1]. Затем каждое подполе соответствует 2N-битным векторам, а общая потребность в памяти для этого W-битового поля равна 2W*N=O(WN) вместо U*N=O(N2). Это уменьшает требование к памяти при 2W <U. Так как W - фиксированное число, алгоритм может добиться линейной увеличении памяти с количеством правил.

Разделение поля приводит к большему количеству подполей, ведущей к большому количеству бит векторов для того, чтобы объединить с помощью побитовых операций И(AND). Для этого предлагается использовать реконфигурируемость и массивный параллелизм платформ (FPGA). Современные устройства FPGA на базе SRAM, такие как более новые платформы семейства Virtex фирмы XILINX [3], обеспечивают высокую тактовую частоту и большое количество встроенной двух-портовой памяти с настраиваемой шириной слова. Он занимает для перенастройки всей ПЛИС несколько миллисекунд, в то время как частота обновления правил систем NIDS составляет день.

Большинство существующих методов классификации пакетов с несколькими совпадениями основаны на TCAM, где каждый вход выполняет параллельный поиск по всем записям за один такт и выводится только первый индекс соответствия [11,18]. Недавние работы Лакшминараянана (Lakshminarayanan) и других [16] использует дополнительные биты в каждой записи TCAM и обнаруживает все совпадающие правила для пакета в нескольких тактах. Кроме того, один из самых современных решений

классификации пакетов с несколькими совпадениями на основе TCAM предложен со стороны Ю(Yu) и др. [17], который основан на геометрическом пересечении правил. После того, что авторы позже предлагают алгоритм установления разделений (Set Splitting Algorithm SSA) [18], чтобы разбить набор правил на две группы, чтобы удалить по крайней мере половину пересечений между правилами. Фаэзипур (Faezipour) и др. [6] предлагают подход максимальный-минимальный разбиение пересечения (MX-MN-IP) для классификации пакетов с несколькими совпадениями на основе TCAM. Объединив TCAM и оригинальный алгоритм BV Сонг (Song) и др. [13] представляют архитектуру под названием BV-TCAM для классификации пакетов с несколькими совпадениями на FPGA.

В вышеприведенных работах получены результаты по классификацию пакета, являющийся линейным по количеству совпадающих правил. Но у них все равно существует проблемы связанный или с низкой скоростью обработки, или с высоким энергопотреблением, даже существует проблема, связанная с недостаточностью памяти.

Анализ набора заголовка правил Snort

Хотя есть некоторые опубликованные замечания о характеристиках реальных наборов правил заголовков пакетов [4,7,12,16], большинство из них используются для классификации наилучшего соответствия, а не для множественного соответствия.

В данной работе показано 10 наборов правил с сайта Snort [2] (табл.1). В таблице показано количество уникальных значений для каждого элемента и приведены:

• Количество правил Snort намного больше, чем количество уникальных заголовков правил. Другими словами, заголовок правила разделяется по многим правилам в Snort.

• Количество заголовков правил остается довольно небольшим, хотя оно постепенно увеличивается.

• Хотя количество заголовков правил постепенно увеличивается, количество уникальных значений для полей SA/DA остаются небольшими, менее 15.

• Количество значений для полей SP/DP увеличивается с такой же скоростью, как и заголовки правил. Количество уникальных значений SP/DP также находится в том же порядке, что и заголовки правил.

• Значение поля протокола ограничено tcp, udp, icmp и ip. Таким образом, число уникальных значений этого поля остается равным 4.

• Большинство полей портов указаны как одно значение (т.е. более 85% как уникальное значение, около 10% как диапазоны).

• Каждое поле использует несколько списков значений. Количество списков значений в полях SA/DA/SP/DP равно 0/3/1/10, соответственно. Списки значений в полях SP/DP имеют не более чем 4 значения, а список значений в поле DA содержит до 18 IP-адресов.

Таблица 1. Статистика набора правил SNORT

Набор правил	Версия Snort	Дата	Кл. Правил	Кл. Заголовков правил	Кл. SA	Кл. DA	Кл. SP	Кл. DA	Кл. Prtcl
R0	2.3.0	20050405	3182	323	11	13	87	173	4
R1	2.4.0	20050722	3462	340	11	13	91	183	4
R2	2.3.0	20070911	8171	589	10	14	198	316	4
R3	2.4.0	20070911	8346	594	10	14	198	320	4
R4	2.6.0	20080924	9290	613	10	13	203	330	4
R5	2.7.0	20081017	9244	594	10	13	190	327	4
R6	2.8.0	20080122	9040	600	10	14	202	321	4
R7	2.8.0	20080826	9277	620	10	13	204	336	4
R8	2.8.0	20081017	9257	597	10	13	187	332	4
R9	2.8.4	20090421	5662	609	10	13	184	344	4

II. ОСНОВНАЯ ЧАСТЬ

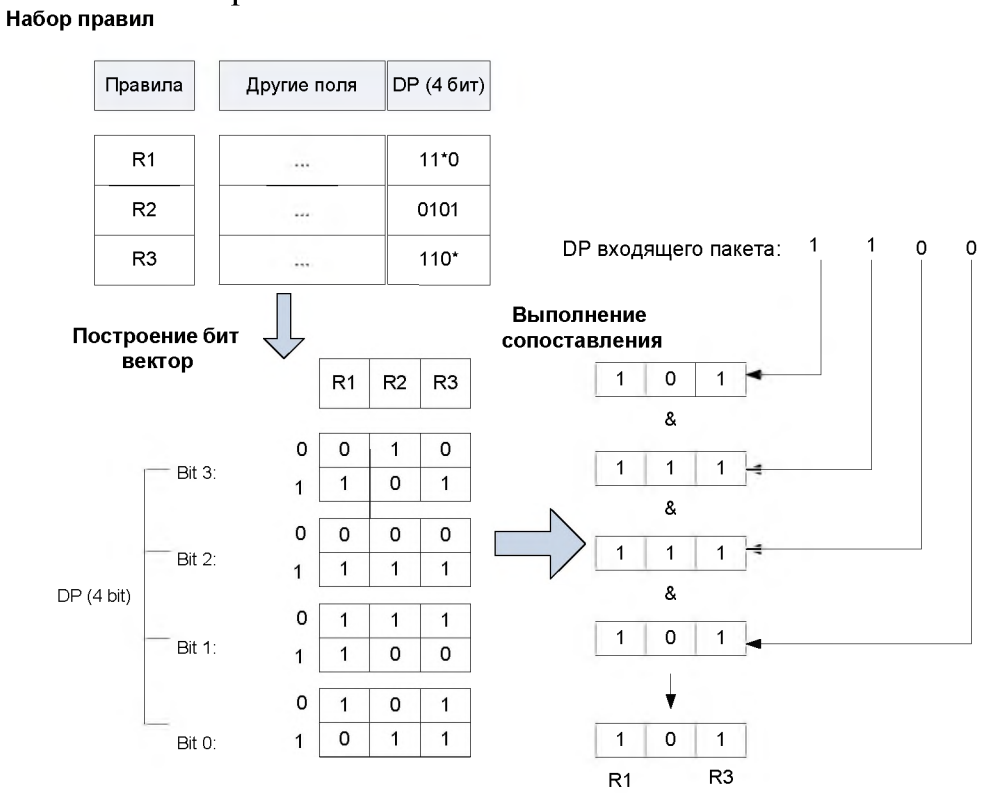
Алгоритм битового вектора с разделением на подполя

Как ранее обсуждалось, многие алгоритмы классификации пакетов страдают от переполнения памяти O(N2). Согласно статистикам набора правил Snort, показанный в таблице 1, большое количество значений поля порта приведет к большому количеству N-битных векторов, поскольку каждое уникальное значение соответствует одному N-битному вектору. Кроме того, нам нужно построить структуру поиска, такую как четырехскатное дерево поиска или сжатая мультибитная структура [13], для входного пакета, чтобы найти соответствующий номер порта. Учитывая уникальные значения U в поле порта W-бит, нам нужна память O(U) и O(UN) для сохранения структуры поиска и UN-битных векторов соответственно. Время поиска в этом поле - это O(logU), использующее двоичное (или квадратное) дерево поиска или O(W) с использованием структуры.

Отметив, что большинство значений в полях порта являются только одиночными числами, можно представлять их в двоичном формате. Даже для диапазонов можно представить их в нескольких тройных строках, где значение каждого бита равно 0, 1 или «*». Каждый бит поля W-бит-порта может выполнять независимое сопоставление на соответствующем бите входного пакета. Другими словами, можно разделить W-бит на W 1-битные подполя. Каждое подполе соответствует к 2N-битным векторам: одно для битового значения 0, а другое для значения 1. Значение «*» в тройной строке отображается с обоими битными векторами. Требование к памяти становится 2WN. Чтобы соответствовать входному пакету в этом поле W-бит, каждый бит поля входного пакета будет обращаться к соответствующей памяти с глубиной 2 и возвращать один N-битный вектор. Финале, W N-битные

векторы выполняют операцию побитовые AND, чтобы получить соответствующий вектор для этого поля. Время поиска - O(W), так как побитовая операция И(AND) может выполняться в O(1) раз в аппаратном обеспечении.

На рисунке 2 показан пример применения предложенного алгоритма для сопоставления DP поля пакета с тремя правилами. Формальные алгоритмы построения битных векторов и для выполнения классификации пакетов показаны в алгоритмах 1 и 2 соответственно.



2-расм. Пример предложенного алгоритма классификации пакетов

Алгоритм 1. Построение бит-вектора.

Input: N правил, каждое из которых представлено в виде W-битной тройной строки:

$Ri = Ti,W-1Ti,W-2 \cdots Ti,0,\ i = 0, 1, \cdots, N-1.$

Output: 2W N-битные векторы: $Vi = Bi,N-1Bi,N-2 \cdots Bi,0,\ i = 0, 1, \cdots, 2W - 1.$

1: Инициализация: $Vi \leftarrow 00 \cdots 0,\ i = 0, 1, \cdots, 2W - 1.$
2: for $i \leftarrow 0$ to $N - 1$ do {Process R i}
3: for $j \leftarrow 0$ to $W - 1$ do
4: if $Ti,j == *$ then
5: $B2j,i \leftarrow 1$
6: $B2j+1,i \leftarrow 1$
7: else {Ti,j == 0 or 1}
8: $B(2j+Ti,j),i \leftarrow 1$

9: B(2j+1−Ti,j),i ← 0
10: end if
11: end for
12: end for

Алгоритм 2. Классификация пакетов

Input: A W-битный заголовок пакета: PW −1PW−2 ⋯P0.

Input: 2W N -битные векторы: Vi = Bi,N−1Bi,N −2 ⋯Bi,0, i = 0, 1,⋯, 2W − 1.

Output: A N-битный вектор Vp указывающий все соответствующие правила.

1: Инициализировать N-битный вектор: Vp ← 11⋯1.
2: for i ← 0 to W − 1 do {bit-wise AND}
3: Vp ←Vp & V2i+Pi
4: end for

Необходимо обратит внимание, что предложенный алгоритм с разделением на подполя обеспечивает эффективность памяти при U>2W, то есть количество уникальных значений поля больше, чем в два раза количеств битов поля. Согласно характеристикам набора правил Snort, применение предложенного алгоритма к полям SP/DP будет значительно сократит потребление памяти по сравнению с предыдущими алгоритмами бит-вектора [10, 13]. Также обратите внимание, что общее число бит-векторов в предложенном алгоритме ограничено общим количеством битов в заголовке пакета. Ограниченный параллелизм в алгоритме делает большое отличие от TCAM, где массивный параллелизм по всем входам приводит к большому потреблению энергии и низкой масштабируемости [15].

Основная архитектура

Предложенный алгоритм может быть легко отображен на архитектуре аппаратного обеспечения. На рисунке 3 показана базовая архитектура предложенного алгоритма для соответствия N правил, каждая из которых может быть представлена в виде тройной строки. Согласно характеристикам, количество уникальных значений в полях SA/DA/protocol намного меньше, чем количество бит этих полях. Таким образом, используется TCAM и CAM, которые эффективны при сопоставлении ввода с небольшим количеством записей для этих полей. Небольшие TCAM выполняют сопоставление префикса в полях SA/DA, в то время как CAM с 4 входами выполняет точное сопоставление в поле протокола.

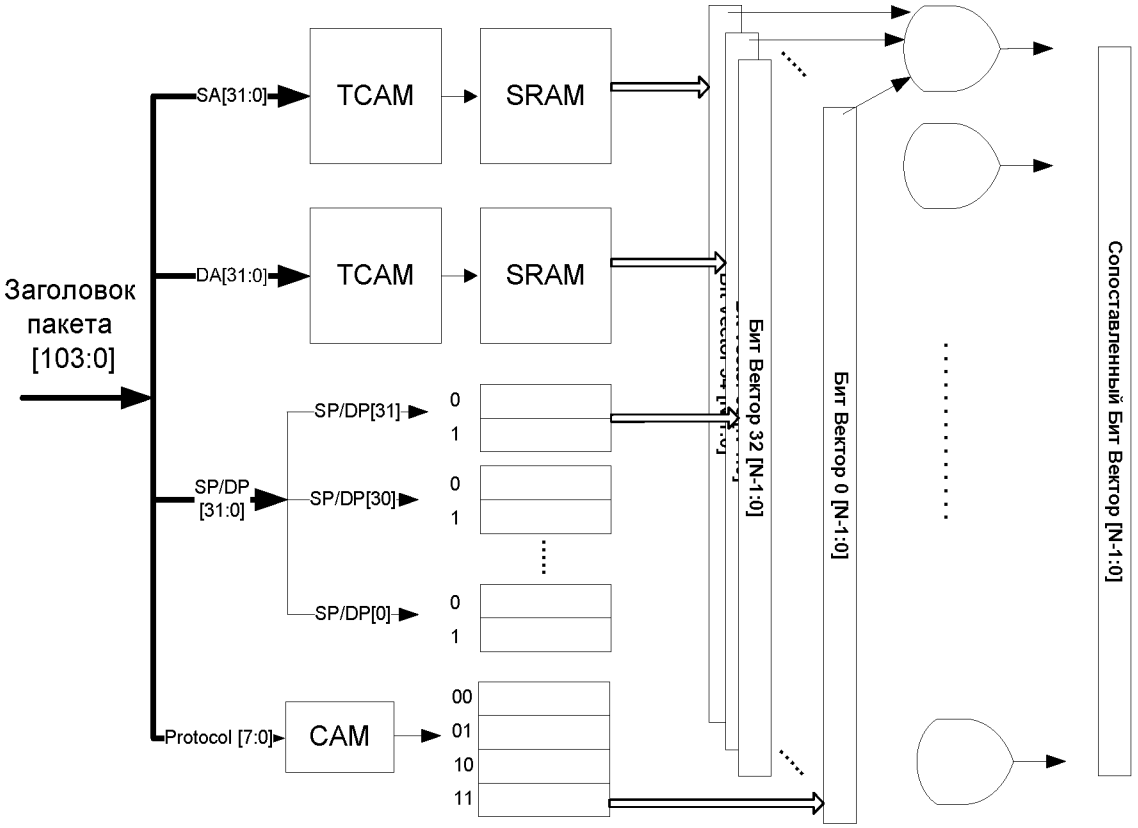


Рис.3. Основная архитектура алгоритма FSBV

В этой архитектуре все битовые векторы хранятся в SRAM. Для полей SA/DA/ protocol количество записей, сохраненные в SRAM, равно к тому, что в TCAM/CAM. Поля SP/DP имеют 16+16=32 бита в общем, и требуется 32 памятей, каждая из которых имеет глубину 2 для сохранения бит-векторов. При наличии одного входного пакета 3 и 32 N-битные векторы будут выводиться соответственно из полей SA/DA/protocol и SP/DP. Эти 35 N-битные векторы будут объединены в конечный N-битный вектор совпадения с помощью побитовых операций И(AND) на аппаратном уровне.

Поддержка функций Snort

Как ранее обсуждалось, правила Snort предоставляют несколько уникальных функций, включая операторов диапазонов и отрицаний и список значений. Архитектура предложенного алгоритма может поддерживать их, хотя эти функции широко не используются в текущих правилах Snort.

Диапазон. нужно преобразовать диапазон в тройную строку, чтобы удовлетворить предположения предложенного алгоритма, что каждое правило представляется как тройная строка.

Отрицание. Оператор отрицания редко используется в текущих наборах правил Snort. Однако отрицательное значение может быть преобразовано в несколько тройных строк. Это значительно воздействует на эффективность памяти [17].

Список значений. Snort позволяет указать поля SA/DA/SP/DP, используя список значений. Текущий набор правил Snort использует несколько списков значений. Для этого можно просто расширить список значений до нескольких значений. Однако преобразование диапазона в строку также приводит к списку значений для одного поля.

Сбор все вместе. После интеграции вышеуказанных методов в базовую архитектуру предложенного алгоритма с разделением на подполя получается архитектуру алгоритма, поддерживающую все функции правил Snort, как показано на рисунке 4.

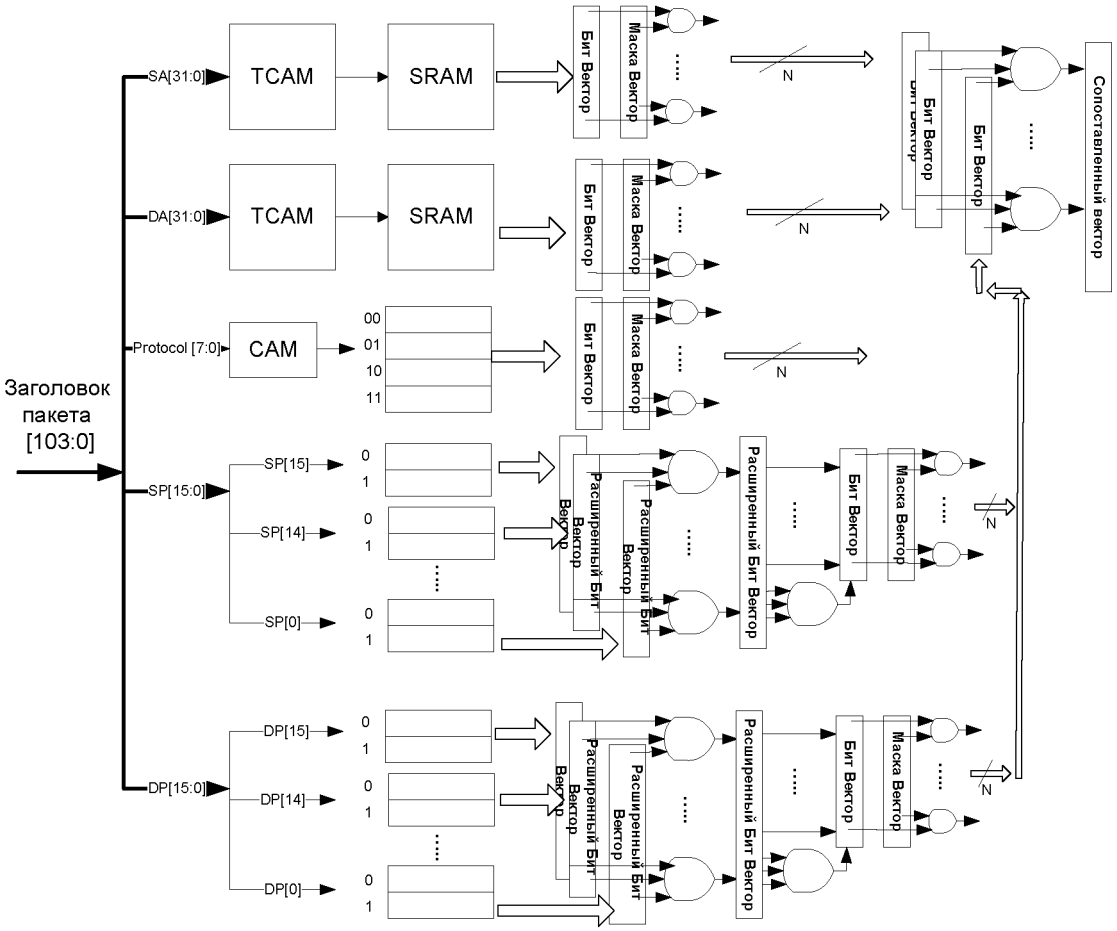


Рис.4. Архитектура алгоритма, поддерживающий функций Snort правил

FPGA реализация

Технология FPGA стала оптимальным выбором для реализации реальных методов обработки сетевых процессов [9,13], благодаря своей способности быстро реконфигурации и для предложения массивного параллелизма.

Формирование блоков. Согласно рис.4, архитектуре алгоритма необходимо большое количество широких блоков памяти с глубиной 2, для хранения бит-векторов полей SP/DP. Однако, большая часть встроенной памяти FPGA организована в виде блоков. Например, минимального размера блока памяти на FPGA Xilinx Virtex 5 составляет 18 Кбайт, с программным

управлением с 16Кх1 бит до 512х36 бит [3]. Другими словами, минимальная длина памяти равна 512, который требует ширину адреса не менее 9 бит. Для использования больших блоков Block RAM, предоставляемых текущими FPGA, предлагается схему формирования блоков, которая рассматривает B биты вместо 1 бит как подполе внутри W-бит поля. После формирования блоков имеется [W/B] подполя, каждый из которых потребляет 2BN бит, данном N правил. Общее требование к памяти является 2^B N W/B=2^B/B WN, в то время как нужно W/B побитовые операции И (AND), чтобы объединить битных векторов, для получения вектор частичного сопоставления этого поля. B данном реализации FPGA B=9, равно к минимальной ширине адреса Block RAMs на FPGA Xilinx.

Формирования блока обеспечивает гибкий способ для контроля степени параллелизма в архитектуре алгоритма. Больше B приведет к меньше N-битным векторам, которые должны быть объединены за счет увеличения потребления памяти. Такой контролируемый параллелизм делает архитектуру алгоритма отличительным от других решений, включающие алгоритм BV, который использует ограниченный параллелизм и TCAM, имеющий массивный параллелизм.

Конвейерная обработка. Дано N правил, один входной пакет извлекает 3 N-битных векторов из полей SA/DA/protocol через доступ TCAM /CAM. Для полей SP/DP с размером формирования блоков B = 9, [32/9]=4N-битный вектора выводятся из SRAM. Однако, это приводит к низкой тактовой частоте, если эти 3+4=7 N-битные векторы выполняются операция побитный AND в одном такте.

При этом используется конвейерной обработки, чтобы минимизировать задержку маршрутизации для достижения высокой тактовой частоты. Конвейерная обработка состоит из 7 этапов: 2 этапа для SP, 2 этапа для DP, 1 этап для SA, 1 этап для DA и 1 этап для протокола. На каждом этапе некоторые биты заголовка входного пакета обращаются к локальной памяти этого этапа для извлечения соответствующего битного вектора, который затем выполняет операцию побитовый AND с вектором бит, перенесенным с предыдущего этапа. Полученный бит-вектор переносится на следующий этап вместе с заголовком пакета. Путем дальнейшего использования двухпортовых BlockRAM, предоставляемых Xilinx FPGA, можно вводить два пакета на каждый такт, для удваивания пропускной способности.

Оценка производительности

Если сравнить предложенный алгоритм с алгоритмами BV и HyperCuts [22] и TCAM-базированными решениями, по отношению среднего требования к памяти для каждого правила. Алгоритм HyperCuts рассматривается как один из наиболее масштабируемых алгоритмов для классификации пакетов наилучшего соответствия [9].

Результаты на правилах Snort. Мы отображали 620 заголовков правил из R7 (самый большой набор правил заголовка Snort, показанный в таблице 1),

на архитектуру предложенного алгоритма. CAM для протокола потребовало 4 записей и TCAM для SA поля 11 записей. Для поля DA были некоторые списки значений, содержащие до 18 IP-адресов. После расширения TCAM для поля DA потребовалось 53 записей. Каждый бит 16-битного SP поля соответствовал двум расширенным 649-битным векторам, в то время как каждый бит DP поля соответствовал двум расширенным 668-битным векторам. Кроме того, для полей SA/DA/SP было три 620-битных векторов маски. Таким образом, для хранения всех битных векторов требовалось 4*620+11*620+53*620+16*2*649+16*2*668+3*620=86164 бит. Средний размер памяти для каждого правила составлял 17.4 байт, включая служебные данные памяти для поддержки уникальных функций в Snort. Служебными данными является FPGA. В документе BV-TCAM не представлены результаты реализации фактической пропускной способности. Используется заранее вычисленное значение, приведенное в [13]. Согласно Таблице 2, архитектура алгоритма достигла гораздо большей пропускной способности, чем современные решения.

Таблица 2. Сравнение производительности

Подходы	**Платформы**	**Память**	**Пропускная способность**	**Поддержка для функций Snort**
BV-TCAM[13]	FPGA	73.8 байты/правил	10 Гб/с	-
TCAM-SSA[18]	ASIC	13 байты/правил	20 Гб/с	-
MX-MN-IP[6]	ASIC	13 байты/правил	80 Гб/с	-
Предложенный алгоритм	FPGA	17.4байты/правил	100 Гб/с	+

III. ЗАКЛЮЧЕНИЕ

Классификация пакетов с несколькими совпадениями является важной функцией в NIDS. В этой статье предложена архитектура параллельных бит-векторов с разбиением на подполя для классификации пакетов с несколькими совпадениями. Этот подход требует линейной увеличении памяти с количеством правил. Изучив характеристики наборов правил Snort NIDS, объединяется они в небольшие TCAM и CAM архитектуры, которые были использованы для сопоставления IP-адреса и полям протокола, соответственно. Результаты реализации FPGA показали, что архитектура предложенного алгоритма может хранить полный набор текущих заголовков правил Snort, используя небольшое количество встроенных ресурсов и

поддерживает пропускную способность 100 Гбит/с для пакетов минимального размера (40 байт).

ЛИТЕРАТУРА

[1] Bro intrusion detection system. http://www.bro-ids.org.
[2] Snort: the de facto standard for intrusion detection/prevention. http://www.snort.org.
[3] Xilinx virtex-5 multi-platform FPGA. www.xilinx.com/products/virtex5/.
[4] F. Baboescu, S. Singh, and G. Varghese. Packet classification for core routers: Is there an alternative to CAMs? In INFOCOM '03: Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies, volume 1, pages 53–63. IEEE, March/April 2003.
[5] F. Baboescu and G. Varghese. Scalable packet classification. IEEE/ACM Trans. Netw., 13(1):2–14, Feb. 2005.
[6] M. Faezipour and M. Nourani. Wire-speed TCAM-based architectures for multimatch packet classification. IEEE Trans. Comput., 58(1):5–17, Jan. 2009.
[7] P. Gupta and N. McKeown. Packet classification on multiple fields. In SIGCOM M '99: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication, pages 147–160. ACM, 1999.
[8] P. Gupta and N. McKeown. Algorithms for packet classification. IEEE Network, 15(2):24–32, March/April 2001.
[9] W. Jiang and V. K. Prasanna. Large-scale wire-speed packet classification on FPGAs. In FPGA '09: Proceeding of the ACM/SIGDA intenational symposium on Field programmable gate arrays, pages 219–228. ACM, 2009.
[11] K. Lakshminarayanan, A. Rangarajan, and S. Venkatachary. Algorithms for advanced packet classification with ternary CAMs. In SIGCOMM '05: Proceedings of the confere nce on Applications, technologies, architectures, and protocols for computer communications, pages 193–204. ACM, 2005.
[12] L. Qiu, G. Varghese, and S. Suri. Fast firewall implementations for software and hardware-based routers. In ICNP '01: Proceedings of the Ninth International Conference on Network Protocols, pages 241–250. IEEE Computer Society, 2001.
[13] H. Songand J .W .Lockwood. Efficient packet classification for network intrusion detection using FPGA. In FPGA '05: Proceeding of the ACM/SIGDA international symposium on Field programmable gate arrays, pages 238–245. ACM, 2005.
[14] L. Tan and T. Sherwood. A high throughput string matching architecture for intrusion detection and prevention. In ISCA '05: Proceedings of the 32nd annual international symposium on Computer Architecture, pages 112–122. IEEE Computer Society, 2005.
[15] D. E. Taylor. Survey and taxonomy of packet classification techniques. ACM Comput. Surv., 37(3):238–275, 2005.

[16] D. E. Taylor and J. S. Turner. Scalable packet classification using distributed crossproducing of field labels. In INFOCOM ’05: Proceeding s of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies, volume 1, pages 269–280. IEEE, March 2005.

[17] F. Yu, R. H. Katz, and T. V. Lakshman. Efficient multimatch packet classification and lookup with TCAM. IEEE Micro, 25(1):50–59, 2005.

[18] F. Yu, T. Lakshman, M. Motoyama, and R. Katz. Efficient multimatch packet classification for network security applications. IEEE Journal on Selected Areas in Communications, 24(10):1805–1816, Oct. 2006.