УДК 004.056.53 **DOI** 10.15622/sp.54.1

И.В. Котенко, А.А. Кулешов, И.А. Ушаков

СИСТЕМА СБОРА, ХРАНЕНИЯ И ОБРАБОТКИ ИНФОРМАЦИИ И СОБЫТИЙ БЕЗОПАСНОСТИ НА ОСНОВЕ СРЕДСТВ ELASTIC STACK

Котенко И.В., Кулешов А.А., Ушаков И.А. **Система сбора, хранения и обработки информации и событий безопасности на основе средств Elastic Stack.**

Аннотация. В статье рассматривается подход к построению системы сбора, хранения и обработки информации и событий безопасности на основе средств Elastic Stack. Анализируются задачи мониторинга и управления инцидентами безопасности, исследуются архитектуры систем мониторинга, выявляются требования к ним, и предлагается архитектура системы сбора, хранения и обработки информации и событий безопасности. Описывается разработанный программный прототип системы и представляются результаты экспериментов с разработанным прототипом.

Ключевые слова: мониторинг и управление инцидентами компьютерной безопасности, SIEM-системы, Elastic Stack, Elasticsearch, Logstash, Kibana.

1. Введение. Технологический прогресс не стоит на месте, и системы защиты информации развиваются и эволюционируют вместе с ним. Системы управления информацией и событиями безопасности (Security Information and Event Management, SIEM) не являются исключением. Ранее функционал классического SIEM-решения больших и средних компаний более-менее удовлетворял имеющимся требованиям. Однако в настоящее время необходимы новые механизмы и функции, способные своевременно и адекватно выявлять, обрабатывать и анализировать текущие потоки информации и событий безопасности и управлять инцидентами для гораздо большего количества устройств с учетом существенно возросших объемов информации, в том числе данных о пользователях, трафике, сервисах, событиях и так далее [1-4].

Проблема заключается в том, что современные SIEM-системы в недостаточной степени адаптированы к своевременной обработке больших объемов информации и событий безопасности, необходимой для оценки текущего состояния, управления инцидентами и выработки контрмер. В работе ставится задача разработки архитектуры и реализации исследовательского прототипа системы сбора, хранения и обработки информации и событий безопасности, базирующейся на технологиях больших данных как основы будущей SIEM-системы, а также проведения предварительного анализа параметров функционирования данной системы. Специфика поставленной задачи заключается в необходимости применения технологий больших данных для задач мониторинга и выборе наиболее производительной архитектуры. Предлагаемое в статье решение отличается от

существующих комплексированием программных продуктов Elastic stack, Nginx и Docker для решения задачи сбора, хранения и обработки большого количества данных в целях мониторинга событий информационной безопасности. Данное решение направлено на обеспечение высокой производительности обработки сообщений с учетом возможных перегрузок и дальнейшего расширения системы.

Статья имеет следующую структуру. Во *втором разделе* представляется краткий анализ релевантных работ и наиболее представительных SIEM-продуктов, рассматриваются их достоинства и недостатки. В *третьем разделе* выявляются технические требования к перспективной системе сбора, хранения и обработки информации и событий безопасности, а также на их основании формируется обобщенная архитектура предлагаемого решения, и с учетом характеристик продуктов, имеющихся на рынке, производится выбор средств реализации. В *четвертом разделе* описан предлагаемый подход к реализации системы сбора, хранения и обработки информации и событий безопасности, удовлетворяющий заданным требованиям, описана его реализация. В *пятом разделе* представлены результаты экспериментов, и проведено сравнение разрабатываемого прототипа с несколькими решениями, предложенными в научных публикациях. В *заключении* сделаны выводы и определены направления будущих исследований.

2. Релевантные решения. Выделим два направления обзора релевантных решений — исследовательские работы и программные реализации коммерческих продуктов и продуктов с открытым исходным кодом.

Рассмотрим вначале несколько исследовательских работ. В [5] описывается обобщенная архитектура SIEM-системы нового поколения, которую можно разбить на уровни сети, данных, событий и приложений. Выделяются следующие основные компоненты системы: коллектор, универсальный транслятор событий, высоконадежная шина данных, масштабируемый процессор событий, репозиторий, система принятия решений и реагирования, компонент моделирования атак и анализа защищенности, прогностический анализатор безопасности и система визуализации.

В [6] определены три архитектурных уровня построения SIEM-систем: анализа данных, управления данными и сбора данных. Данный подход дает возможность проанализировать сложность обработки и количество обрабатываемых событий на каждом из уровней. Делается вывод, что самым нагруженным и требовательным к вычислительным мощностям является уровень сбора данных.

Исследования, приведенные в [4], направлены на анализ центрального компонента любой SIEM-системы — системы хранения данных. Представлены преимущества хранения информации в гибридном репозитории. Предложенный подход к хранению обеспечивает удобный и надежный обмен данными между разнородными системами хранения и позволяет реализовать достаточно высокую производительность.

Системы, способные работать с большими данными, интегрируются во все большее количество различных продуктов, и SIEM-продукты не является исключением [7, 8]. Представителем такой системы является платформа Hadoop [9]. Она работает по принципу пакетной обработки, когда непосредственный анализ не зависит от потока данных. Платформа предоставляет средства обработки структурированных и неструктурированных файлов больших размеров. За данную функциональность отвечает компонент управления ресурсами MapReduce [10]. Основная идея MapReduce заключается в распределении задач с использованием большого количества узлов, организованных в кластер. В [11] представлен результат реализации такого подхода (скорость — до 30 000 MB обработанных данных в секунду). Такая производительность возможна на основе применения большого количества средств для вычисления, в частности, использовалось 1800 узлов, каждый из которых включал два процессора 2ГГц Intel Xeon и 4Гб оперативной памяти. В указанной работе рассмотрена технология пакетной обработки, однако для обработки данных в потоке используют потоковую обработку, позволяющую отслеживать сообщения в реальном времени. Такую задачу выполняет, например, система Apache Storm [12]. Компания Cisco представила анализ данного подхода [13], способного обрабатывать свыше миллиона пакетов за секунду на одном узле системы.

Компания Gartner [14] выделяет несколько наиболее продвинутых SIEM-систем, в том числе HP ArcSight, IBM Qradar (из коммерческих) и AlienVault OSSIM (с открытым исходным кодом).

SIEM-система HP ArcSight [15] обладает модулями мониторинга событий, поведенческого анализа и системой правил обработки событий безопасности. Из-за закрытого исходного кода затруднено добавление нового функционала. HP Arcsight в качестве СУБД использует собственную разработку CORR. В системе по умолчанию не реализована возможность хранения событий, поступающих без определенного шаблона либо маски, то есть для добавления принципиально новой информации необходимо дополнительное вмешательство в систему. В системе реализовано централизованное хранение данных. Для географически

распределенных организаций необходимо передавать все события в центральную базу данных, где и выполняется вся обработка. У ArcSight ESM ядро системы лицензируется по объему логов в день. Кроме ядра необходимо лицензировать набор различных параметров и опций, например количество пользователей, разработка собственных коннекторов, количество источников событий (считается раздельно по типам источников), модули соответствия требованиям, управление логами и так далее. Из чего следует, что ArcSight ориентируется на крупные корпорации.

Конкурентом ArcSight является IBM SIEM Qradar [16]. Данное средство представляет собой единое унифицированное решение для управления информацией и событиями безопасности, в том числе управления системными журналами, выявления аномалий, управления конфигурациями и устранения уязвимостей. Эта система позволяет выявлять признаки наиболее критичных инцидентов. Qradar использует преимущества единой архитектуры для анализа системных журналов, потоков данных, уязвимостей, данных пользователей и информационных ресурсов. Вместе с тем данный подход приводит к технически сложному процессу увеличения ресурсов, используемых системой. Кроме того, отсутствует программно-аппаратная реализация. Стоимость приобретения SIEM IBM Qradar складывается из многих факторов комплекта поставки и конфигурации самой системы. Цена лицензии зависит также от возможностей по обработке событий в секунду. Высокая цена не позволяет применять решение на малых и средних предприятиях.

OSSIM (Open Source Security Information Management) [17] — это SIEM-система на основе открытого исходного кода от компании AlientVault. OSSIM реализует функции сбора, анализа и корреляция событий, а также обнаружения вторжений. Она включает хостовую систему обнаружения вторжений (HIDS), сетевую систему обнаружения вторжений (NIDS), систему обнаружения вторжений для беспроводных сетей (WIDS), компоненты мониторинга узлов сети, анализа сетевых аномалий, сканер уязвимостей, систему обмена информацией об угрозах между пользователями, набор плагинов для синтаксического анализа и корреляции записей системных журналов с различных внешних устройств и служб. Основным недостатком перечисленных решений является ограниченная функциональность по агрегации получаемых сообщений.

Поскольку SIEM-продукты развивались эволюционно, то с приходом концепции больших данных в сферу информационной безопасности большинство из них были вынуждены внедрить

дополнительный набор методов Hadoop по работе с большим потоком информации от компании Apache Software Foundation. На сегодняшний день ситуация не изменилась, и Hadoop используют как основное решение. Стоит учитывать, что проект Hadoop был разработан с целью построения программной инфраструктуры распределенных вычислений. Выход осенью 2013 года модуля YARN, делающего технологию универсальной для обработки данных, считается большим развитием для проекта, однако, по нашему мнению, является избыточным для решения задач по сбору, индексации и распределению данных.

3. Требования, обобщенная архитектура и выбор средств для реализации системы сбора, хранения и обработки информации событий безопасности. С учетом результатов анализа релевантных работ в настоящей статье ставится цель разработать обобщенную архитектуру и прототип системы сбора, хранения и обработки информации и событий безопасности, базирующуюся на технологиях больших данных, как основу будущей SIEM-системы.

Перечислим базовые требования к системе сбора, хранения и обработки информации и событий безопасности: (1) обеспечение сбора информации от множества различных источников; (2) эластичность системы — обеспечение оптимального (рационального) распределения нагрузки и низкой зависимости показателей функционирования компонентов при изменении отдельных компонентов; (3) возможность эффективной интеграции собственных алгоритмов в подсистему аналитики для последующего развития SIEM-системы; (4) минимизация времени на развертывание и настройку инфраструктуры, сервисов и других системных задач; (5) реализация эффективных механизмов обеспечения отказоустойчивости и высокой доступности на программном уровне; (6) поддержка горизонтального масштабирования — способности разделения системы на отдельные компоненты и разнесения их по отдельным физическим машинам; (7) распространенность средств разработки системы на рынке и активное сообщество пользователей.

Архитектура системы должна быть микросервисной, что обеспечит системе виртуализацию, отказоустойчивость и быстрое горизонтальное масштабирование. Это также позволит автоматизировать и упростить развертывание. Для эффективной интеграции алгоритмов аналитики необходима поддержка развитого API на популярных языках программирования, таких как JAVA, C++ или Python. Не должно быть жесткой настройки прототипа к определенному источнику события, то есть прием информации должен осуществляться по маске или универсальному образцу. Должна быть обеспечена возможность использования нескольких сетевых потоков

или одинаковых компонентов для распределения нагрузки, во избежание перегрузок на стороне сервера.

Основным компонентом архитектуры SIEM-системы является набор средств, осуществляющий функции сбора, хранения, обработки информации и событий безопасности и функции аналитики, что в совокупности представляет систему мониторинга и управления инцидентами [1-4].

Рассмотрим представленную в [5] архитектуру SIEM-системы нового поколения (рисунок 1).

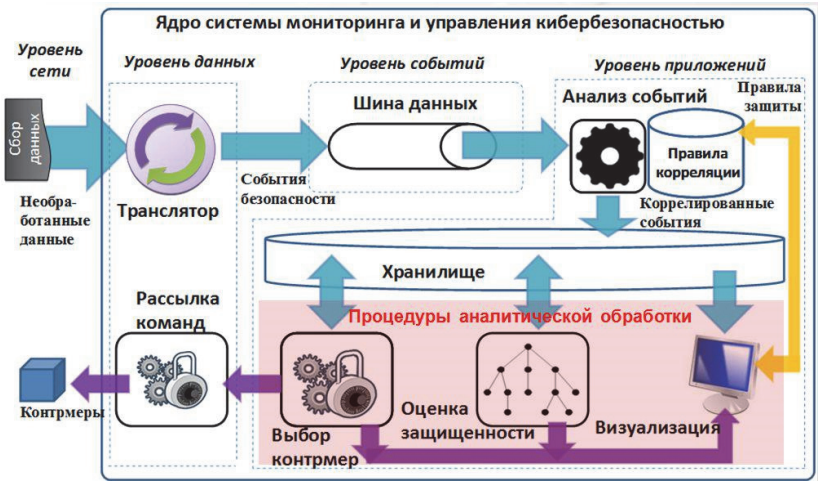

Рис. 1. Архитектура системы мониторинга и управления безопасностью [5]

В данной архитектуре можно выделить четыре уровня: (1) уровень сети, к которому относятся компонент сбора данных с устройств в сети и компонент реализации контрмер; (2) уровень данных, к которому относятся транслятор, необходимый для надежной передачи сообщений и компонент рассылки команд; (3) уровень событий, представляющий шину данных, служащий для оперативного хранения и передачи поступающей информации; (4) уровень приложений, состоящий из компонента корреляции событий, их анализа, долговременного хранилища и специальных процедур аналитической обработки.

Указанная архитектура является абстрактной и не раскрывает особенностей обработки большого потока входных данных. Предлагаемая обобщенная архитектура разрабатываемой SIEM-системы нового поколения, основанная на технологиях больших данных, представлена на рисунке 2.

SPIIRAS Proceedings. 2017. Issue 5(54). ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

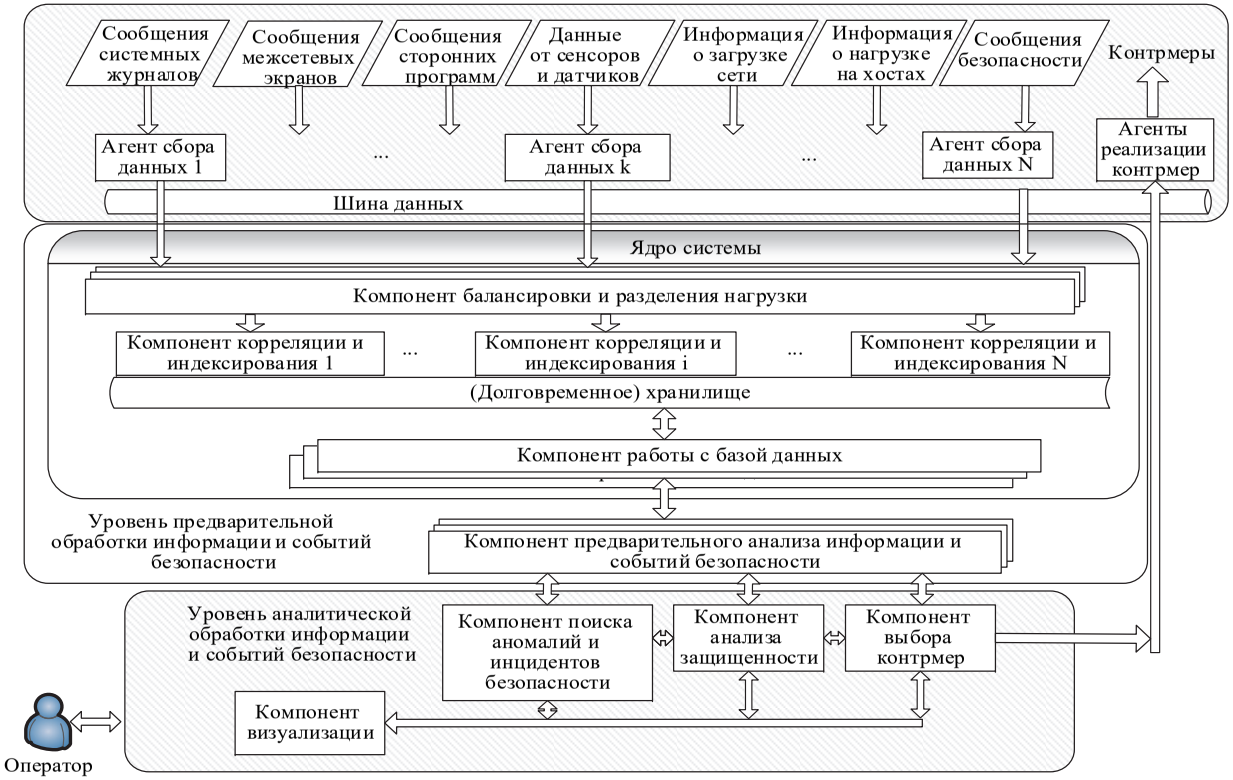

Рис. 2. Обобщенная архитектура SIEM-системы нового поколения, основанной на технологии больших данных

Архитектура включает три уровня обработки информации: (1) уровень сети и данных (данные от источников, специальные агенты сбора и отправки данных, агенты реализации контрмер); (2) уровень предварительной обработки информации и событий безопасности (компонент балансировки и разделения нагрузки, компоненты корреляции и индексирования, хранилище, компонент работы с базой данных, компонент предварительного анализа информации и событий безопасности); (3) уровень аналитической обработки информации и событий безопасности (компонент поиска аномалий и инцидентов безопасности, компонент анализа защищенности, компонент выбора контрмер, компонент визуализации).

Для обеспечения сбора информации от множества источников различной информации на уровне сети и данных используются агенты сбора данных, обеспечивающие отправку и сбор информации с устройств в сети. Далее полученная информация отправляется на шину данных, предназначенную для распространения информации о событиях безопасности и их гарантированной доставки требуемым компонентам SIEM-системы. Из-за большого количества входных данных возможны перегрузки системы. Для предотвращения и рационального распределения нагрузки в ядре системы весь входной поток информации принимается компонентами балансировки и разделения нагрузки, и корреляции и индексирования, которые находится на уровне предварительной обработки информации и событий безопасности. После этого данные попадают в хранилище. Для обеспечения быстрого поиска и выборки из хранилища используются компоненты работы с базой данных. Компонент предварительного анализа информации и событий безопасности позволяет собрать необходимую выборку событий и произвести первичный анализ полученной информации для дальнейшего предоставления компонентам уровня аналитической обработки информации и событий безопасности.

Уровень аналитической обработки информации и событий безопасности включает компоненты поиска аномалий и инцидентов, анализа защищенности, выбора контрмер и визуализации. Компонент поиска аномалий и инцидентов отправляет запросы компоненту предварительного анализа информации и событий безопасности на предоставление необходимых данных. Далее происходит оценка защищенности данных компонентом анализа защищенности и в случае обнаружения угроз безопасности данная информация передается компоненту выбора контрмер. Компонент выбора контрмер, в свою очередь, используется для выбора необходимых контрмер и отправляет инструкции агентам реализации контрмер. Оператор взаимодействует с

компонентами аналитической обработки информации и событий безопасности с помощью компонента визуализации.

Представленная архитектура позволяет реализовать комплексный подход к задаче мониторинга инцидентов информационной безопасности. В задачу ядра системы входит сбор и хранение как можно большего объема информации в реальном времени, чтобы впоследствии произвести комплексный анализ предоставленных данных на уровне аналитической обработки. В случае обнаружения инцидентов имеется возможность учесть при анализе ситуации и выработке контрмер изменения системы за длительное время.

Рассмотрим ниже практические решения по реализации предложенной архитектуры. Одним из возможных решений для реализации данной архитектуры является коммерческий программный продукт Splunk Enterprise [18]. Данный продукт — платформа аналитики для сбора и анализа машинных данных с автоматической балансировкой нагрузки. Он обладает возможностью наращивать производительность за счет добавления типовых серверов, что обеспечивает горизонтальную масштабируемость и отказоустойчивость. Splunk имеет документированный RESTful API и SDK для популярных языков программирования. Платформа обрабатывает данные любого формата, включая динамические данные программных приложений, серверов приложений, веб-серверов, операционных систем и многих других источников. Однако Splunk является коммерческим продуктом с закрытым исходным кодом, что ограничивает и замедляет развитие API и делает масштабирование системы платным.

Еще одно коммерческое решение — ManageEngine EventLog Analyzer [19], имеющее схожие достоинства и недостатки. Этот продукт поддерживает распределение нагрузки с использованием внутреннего сервиса. Алгоритмы, методы и основа платформы не доступны на программном уровне, а вся настройка и работа производится с помощью API, который развивается в зависимости от нужд потребителя, что уменьшает гибкость системы.

В 2014 году компания Cisco Systems опубликовала исходные коды системы OpenSOC [20], используемой для создания центра мониторинга киберугроз и основанной на системе Apache Big Data open source. Данные исходные коды стали основой проекта Apache Metron [21]. Система имеет архитектуру, изначально ориентированную на обработку массивов больших данных от множества распределенных источников. На текущий момент проект Apache Metron не является активно развиваемым открытым сообществом — статистика поиска Google и данные GitHub [22] по добавлению кода в проект показывают

слабый интерес со стороны потенциальных пользователей. Развитием проекта сейчас занимается практически единолично компания Hortonworks, предлагающая коммерческое внедрение данного продукта. К сожалению, адаптированная под задачи архитектура Apache Metron имеет плохо реализованную систему обратного масштабирования, и для построения простейшей системы мониторинга необходимо развертывание большого количества компонентов и ресурсоемких сервисов, которые большей частью не будут использоваться. В условиях среднестатистических ресурсов порой достаточно сложно обеспечить загрузку данными от источников, под которую рассчитана данная система. Одновременно с этим для функционирования компонентов системы требуются значительные человеческие ресурсы на развертывание и конфигурирование. Проект Apache Metron заслуживает внимания в случае необходимости решения задач, ориентированных именно на обслуживание высоконагруженных и распределенных информационных систем.

Одним из самых производительных из открытых программных продуктов, специализирующихся на решении обозначенных задач, является Graylog [23]. Это бесплатная система централизованного сбора, хранения и анализа информации, часто используемая в среде DevOps-команд. Эта система использует функции Elasticsearch. Стоит учесть, что, несмотря на частые обновления Graylog и развитое сообщество пользователей, интеграция актуальных версий Elasticsearch в проект требует много времени. Доказательство этому то, что на 2017 год последняя версия Graylog 2.2.1 работает только с Elasticsearch версии 2.4.4 [24], являющейся устаревшей.

В связи с частичной невозможностью удовлетворить всем требованиям, сформулированным выше, в работе был предложен следующий подход к реализации архитектуры системы.

Основываясь на результатах анализа имеющихся в индустрии средств выбран комплекс программного обеспечения Elastic Stack [25] с открытым исходным кодом, так как он подходит для удовлетворения ранее сформированных требований, является бесплатным и популярным решением, сумевшим доказать свою эффективность в большом количестве внедрений по всему миру. Основными программными компонентами являются: Elasticsearch — поисковое и аналитическое ядро системы; Logstash — программный конвейер обработки данных; Kibana — средство визуализации и навигации по системе; Beats — набор программ, необходимых для сбора и транспортировки системных журналов и файлов.

Программный стек, состоящий из Elasticsearch, Logstash и Kibana (далее ELK), специально разработан для решения задач сбора, хранения и обработки системных журналов. Необходимо отметить, что решение на базе программного стека ELK является достаточно зрелым продуктом, и на его базе реализован коммерческое решение Elastic X-Pack, объединяющее вышеперечисленные технологии с добавлением дополнительных функций аналитики.

Рассмотрим основные составляющие Elastic Stack, которые используются при реализации предлагаемой системы мониторинга.

Elasticsearch — распределенное, поисковое и аналитическое ядро системы, поддерживающее REST API и передачу данных через JSON. Elasticsearch централизованно хранит поступающие в него данные, поддерживает кластерную архитектуру, позволяя масштабировать систему, и является прозрачной и надежной системой, обладая средствами обнаружения сбоев, что, в свою очередь, обеспечивает высокую доступность. Ядро Elasticsearch выполняет в реальном времени поиск по большим объемам разнотипных структур данных — документов. Документ — базовая единица информации, которая может быть проиндексирована. Документы специфицируются в формате JSON. Система имеет развитый API, в список поддерживаемых языков для взаимодействия входят Java, Python, C++ и другие.

Logstash — программный конвейер обработки данных, он одновременно собирает данные из множества различных источников, обрабатывает их и отправляет в подсистему хранения. Logstash имеет встроенный синтаксический анализатор, позволяющий нормализовать разнотипные данные, производить определение географических координат по IP, обрабатывать информацию различных источников независимо от формата и структуры.

Kibana — программный компонент, реализующий функции визуализации и навигации в комплексе Elastic Stack. Kibana представляет данные в виде настраиваемой интерактивной панели индикаторов в реальном времени. Он реализует большое количество встроенных настраиваемых виджетов (гистограммы, графы, карты и другие стандартные инструменты) и имеет развитый API.

Beats — набор программ — коллекторов данных с низкими требованиями к ресурсам, которые устанавливаются на клиентских устройствах для сбора системных журналов и файлов. Имеется широкий выбор коллекторов, а также возможность написать свой коллектор. Filebeat транслируют на сервер информацию из динамических обновляемых журналов системы и файлов, содержащих текстовую информацию. Для аналогичных действий с журналами Windows-систем используется Winlogbeat. Packetbeat — сетевой

анализатор пакетов, который отправляет информацию о сетевой активности между серверами приложений. Он перехватывает сетевой трафик, декодирует протоколы и извлекает необходимые данные. Metricbeat собирает метрики операционных систем, характеризующие, например, использование CPU и памяти, количество переданных пакетов в сети, состояние системы и сервисов, запущенных на сервере.

4. Архитектура и реализация прототипа системы сбора, хранения и анализа информации и событий безопасности. Реализованный прототип имеет архитектуру, представленную на рисунке 3. Данные, которые собираются с помощью Winlogbeat и Metricbeat XML-документом, отправляются по сети Интернет на сервер, где развернут ELK-сервер. Данные принимает экземпляр Logstash, настроенный на сбор данных, который выполняет функцию коллектора. Затем специальный экземпляр Logstash индексирует поступающую информацию с коллекторов и отправляет полученные данные Elasticsearch для последующего сохранения и обработки. Вся аналитическая обработка происходит посредством программного компонента Kibana. Чтобы реализовать функцию просмотра представления Kibana по доменному имени сети Интернет, необходим сервер Nginx, на котором информация с порта вывода Kibana проксируется на 80-ый порт. Полученные команды пользователя с помощью API от Kibana передаются Elasticsearch, где обрабатываются и получают результирующие данные.

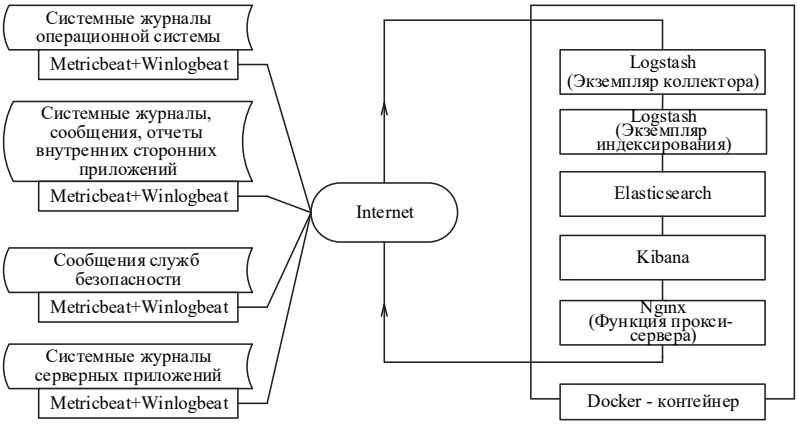

Рис. 3. Архитектура прототипа системы сбора, хранения и анализа информации и событий безопасности

Исходя из описания уровней обобщенной архитектуры, архитектура прототипа соответственно разделяется на уровень сети и данных, предварительной обработки информации и событий

безопасности, а также аналитической обработки. Агентами сбора данных являются Metricbeat и Winlogbeat, а вывод результатов анализа и мониторинга осуществляется с помощью сервера Nginx. Функцию балансировки и разделения нагрузки выполняют экземпляры коллектора и индексирования Logstash. Elasticsearch включает методы поиска и корреляции событий, индексирования перед хранением и хранение входной информации. С помощью открытого API Elastic Stack возможно внедрить скрипты, в том числе по машинному обучению, которые уже на основе полученной выборки будут проводить дополнительные расчеты.

Для полного описания и последующего анализа предлагаемых решений по созданию прототипа системы сбора, хранения и обработки информации и событий безопасности условно разделим прототип на следующие компоненты: (1) отправки данных с клиентских устройств; (2) конвейерной обработки и доставки данных; (3) обеспечения отказоустойчивости и балансировки нагрузки; (4) поискового и аналитического ядра и хранения; (5) визуализации.

4.1. Подсистема отправки данных с клиентских устройств. Система сбора, хранения и обработки информации и событий безопасности разрабатывается как ядро SIEM-системы, поэтому необходимо обеспечить в прототипе широкий охват доступной информации для анализа. На данный момент поддерживается сбор событий протокола syslog, журналов событий Windows, телеметрии оборудования, ОС и сервисов, а также информации о потоках сетевого трафика из протокола netflow/sflow с помощью filebeat, winlogbeat, metricbeat, packetbeat соответственно. В будущем, при необходимости отправки специфичных данных, планируется написание собственных Beat-коллекторов на основе библиотеки libbeat и развитого API.

4.2. Подсистема конвейерной обработки и доставки данных. Для обработки и доставки данных в Elastic Stack используется конвейер Logstash, показанный на рисунке 4.

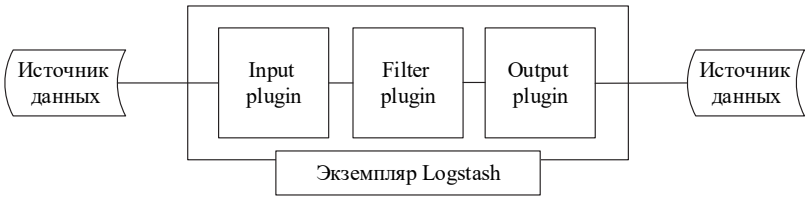

Рис. 4. Схема работы Logstash

Он разделен на три основных функциональных блока, реализованных в виде трех расширений (plugins): input; filter; output.

В функциональном блоке Input plugin указывается конкретный источник событий, который будет считываться Logstash-конвейером. В прототипе этими источниками являются beats. Используются документы JSON-формата, содержащие данные из журналов системы, метрики системы, информацию из протоколов и другие доступные данные в соответствии с выбранным коллектором.

Блок Filter plugin выполняет промежуточную обработку события. Это позволяет структурировать данные, извлекая только необходимую информацию, например, дату, время, IP-адрес, код ошибки и так далее, и, сохраняя их в структуры данных, отправлять далее информацию в output plugin для последующей передачи в Elasticsearch. Используемый фильтр выбирается в зависимости от характеристик события. Следует учитывать, что этот блок является ресурсоемким, в связи с этим Filter plugin активно использует распараллеливание вычислений.

В Output plugin указывается дальнейший маршрут обработки документов в JSON-формате. Это окончательный этап функционирования конвейера. В прототипе данные передаются в подсистему аналитики и хранения — Elasticsearch.

4.3. Механизм обеспечения отказоустойчивости и балансировки нагрузки. Архитектура конвейера обработки и доставки данных в прототипе показана на рисунке 5. При превышении скорости входящих событий над скоростью обработки данных конвейер Logstash начинает отбрасывать события. Для предотвращения потерь в качестве буфера был использован брокер сообщений Redis (возможно также использовать стандартные для индустрии решения, например Redis/ Kafka/ RabbitMQ).

В реализованном прототипе подсистема конвейерной обработки на базе Logstash была разделена на два отдельных программных сервиса: (1) сервис приема данных от источников событий beats, с последующей отправкой в буфер massage broker; (2) сервис приема из буфера, дальнейшей обработки и отправки в Elasticsearch. Использование нескольких экземпляров объектов Logstash с разделением функциональной роли позволяет выполнять балансировку нагрузки между источником данных и кластером Logstash.

Чтобы избежать невозможности ввода данных определенного типа, когда экземпляр Logstash данного типа не доступен, используются специально настроенные конвейеры Logstash, поддерживающие множество модулей input plugin. Например, если бы в подсистеме имелся только один экземпляр объекта Logstash с file input plugin, то при его сбое, исчезла бы возможность принимать данные от Filebeat. Увеличение количества модулей input plugin позволяет масштабировать горизонтально, а разделенные параллельные принимающие конвейеры увеличивают надежность системы и устраняют единую точку отказа.

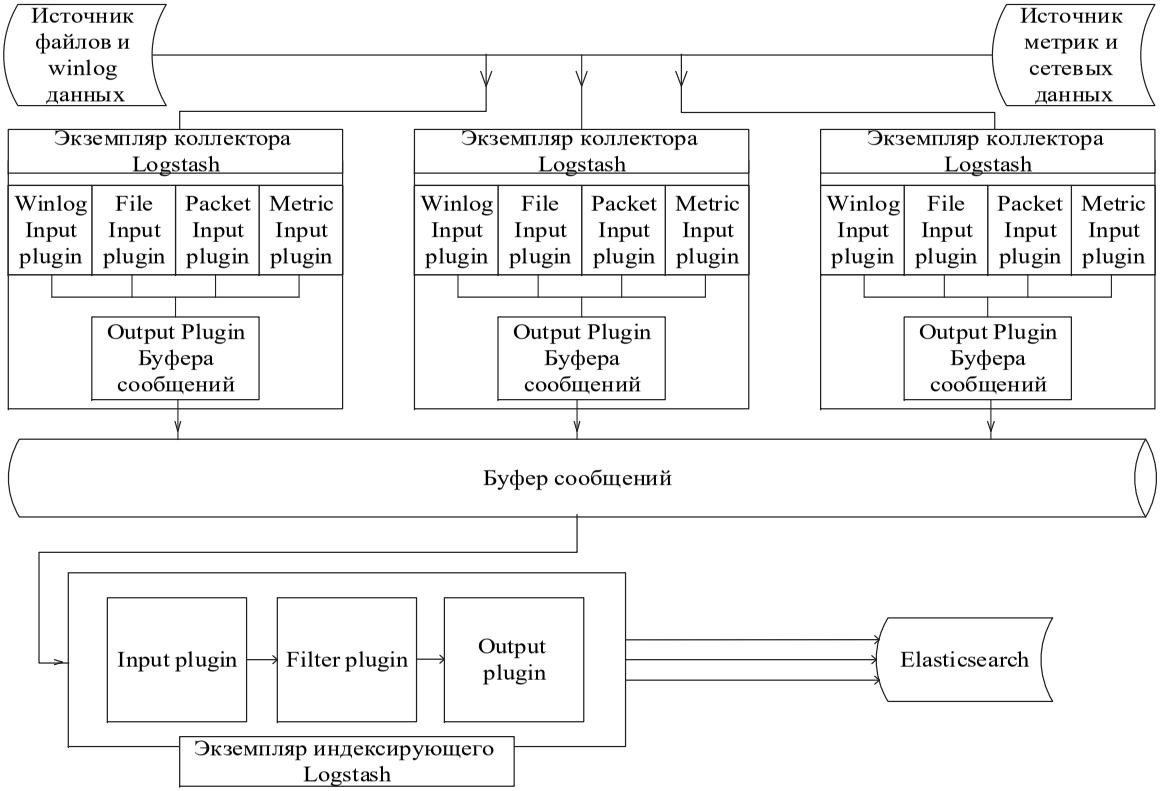

Рис. 5. Архитектура конвейера обработки и доставки данных

Модуль Elasticsearch output plugin также настроен на автоматическую балансировку нагрузки с использованием множества узлов в кластере Elasticsearch. При отказе одного из узлов поток данных не прерывается, что позволяет исключить единую точку отказа. Это обеспечивает высокую доступность кластера и маршрутизацию трафика к активным узлам кластера.

Использование нескольких экземпляров объектов Logstash с разделением функциональной роли позволяет использовать балансировку нагрузки между источником данных и кластером Logstash. Чтобы избежать невозможности ввода данных определенного типа, когда экземпляр Logstash данного типа не доступен, используются специально настроенные конвейеры Logstash, поддерживающие множество модулей input plugin. Например, если бы в подсистеме имелся только один экземпляр объекта Logstash с file input plugin, то при его сбое исчезла бы возможность принимать данные от Filebeat. Увеличение количества модулей input plugin позволяет выполнять горизонтальное масштабирование, а разделенные параллельные принимающие конвейеры увеличивают надежность системы и устраняют единую точку отказа.

4.4. Подсистема поискового и аналитического ядра, совмещенная с подсистемой хранения. В прототипе развернут кластер Elasticsearch. В терминологии Elastic Stack кластер представляет собой набор узлов (серверов), которые хранят всю информацию и предоставляют возможность индексирования и поиска по всем узлам. В терминологии ELK набор структур-документов, имеющих какие-либо похожие характеристики, называется *индексом*. Для удобства индексы разделяют на типы — документы, имеющие общие поля. Потенциально индекс может увеличиваться до больших размеров, превышающих физические возможности узла. Для этого индекс делят на несколько частей называемых *осколками* (shards). Это позволяет распределять данные на несколько узлов, а также распределить и распараллелить операции с осколками, что увеличивает производительность и пропускную способность. Для предотвращения сбоев и обеспечения отказоустойчивости, Elasticsearch позволяет делать копии осколков индекса, которые называются *репликами*. Осколок и его реплика никогда не располагаются на одном узле.

Элементы системы развернуты в контейнерах Docker, что позволяет автоматизировать функции развертывания системы и управления. Для ускорения разработки и тестирования, а также легкой модернизации до следующей версии сервисов в будущем, выбраны микросервисный подход и развертывание в среде контейнерной

виртуализации. Это разделило подсистему на модули, что упрощает обновление отдельных сервисов, а также тестирование их на совместимость друг с другом. В то же время контейнеры решают задачи резервирования инфраструктурных сервисов, позволяют инженерам сфокусироваться на логике решения и абстрагироваться от инфраструктурных проблем.

5. Результаты экспериментов. С целью тестирования предложенных решений был проведен ряд экспериментов.

Стенд для исследований имел следующие характеристики: 16 Гбайт RAM, использующей технологию ddr3; 4-х ядерный процессор частотой 1.9 ГГц; версии Elasticsearch 5.1.1, Kibana 5.1.1, Logstash 5.1.1, Winlogbeat 5.1.1, Nginx 1.10.2, Java(TM) SE Runtime Environment (build 1.8.0_73-b02); операционная система CentOS 7 на ядре Linux 3.10.0-327.36.3.el7.x86_64.

Одним из важных критериев работы системы мониторинга является пропускная способность системы (определяемая как количество обработанных пакетов от устройств за определенный промежуток времени). Когда пакет поступает на порт сервера ELK, Elasticsearch принимает его и начинает индексировать информацию. После индексирования происходит сохранение полученных данных в базе и обновление API. Обновление происходит не сразу, а через интервал времени, прописанный в настройках как refresh_interval, который по умолчанию равен 1 сек. На обновление API требуются вычислительные ресурсы, поэтому при загрузке большого количества пакетов, и при допустимости задержки мониторинга, интервал увеличивают, что повышает скорость индексирования. Все эксперименты проводились при refresh_interval, равном 1 сек.

Для проведения экспериментов к серверу по сети WLAN с максимальной пропускной способностью 10Мбит/с подсоединялся компьютер, с которого отправлялись данные из заранее подготовленных системных журналов. Во время экспериментов отправлялись данные без дополнительных сведений об источнике в виде одного сообщения. При этом средняя скорость индексирования и обработки составила 13000 пакетов в секунду при средней загрузке процессора 65%. На рисунке 6 показана зависимость загрузки CPU в процентном соотношении от времени. На рисунке 7 представлена зависимость количества обрабатываемых пакетов от времени.

Для того чтобы система могла отслеживать, от какого узла пришло сообщение, какой процесс отправитель, а также время происшествия и так далее, сообщения упаковываются в JSON-пакеты, содержащие дополнительную информацию. Во время экспериментов

отправлялась библиотека JSON-пакетов с дополнительной информацией. В результате размер пакета по сравнению с первым экспериментом увеличился и в среднем стал равен 330 байтам. Это сказалось на скорости обработки, которая в среднем стала равна 4000 пакетов в секунду, а загрузка процессора в среднем составила 40%, что показано на рисунках 8 и 9.

Важной функцией SIEM-системы является аналитическая обработка информации и событий безопасности, для которой критична возможность быстрого извлечения необходимых данных из базы данных. Чтобы проверить время поиска, системе экспериментально были посланы запросы выгрузки проиндексированных и сохраненных сообщений, содержащие слово из 5 букв. Из полученных результатов, представленных на рисунке 10 (где ось Y задает время, мс, а ось X — номер испытания), выявлено, что среднее время поиска сообщений по фразе составляет 1.28 мс.

Рис. 6. Зависимость загрузки CPU в процентном соотношении от времени при обработке пакетов

Рис. 7. Зависимость скорости обработки пакетов от времени

Рис. 8. Зависимость загрузки CPU при обработке пакетов с учетом отправки дополнительной информации от времени

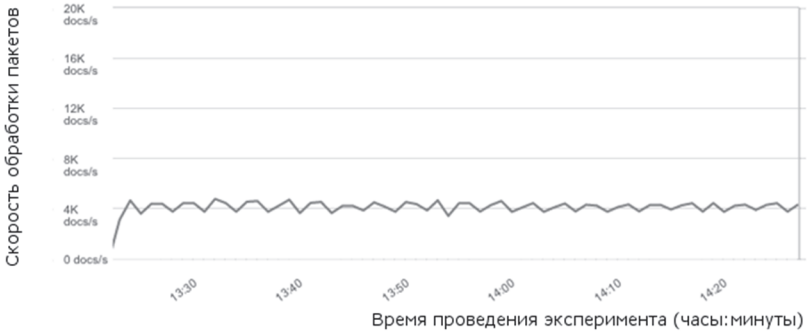

Рис. 9. Зависимость скорости обработки пакетов с учетом отправки дополнительной информации от времени

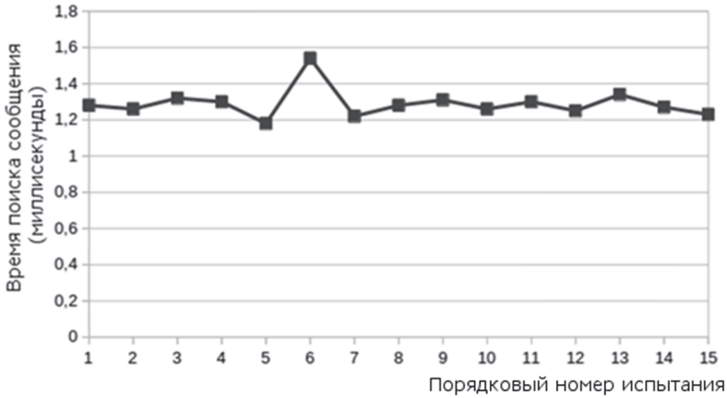

Рис. 10. Зависимость времени поиска сообщений по фразе от номера испытания

Эксперименты демонстрируют, что вычислительной мощности системы хватает для увеличения скорости обработки данных, однако увеличения скорости не происходит. Причина этому — ограничения пропускной способности сети. Однако поставленная цель обработки и представления большого количества пакетов в реальном времени достигнута. С учетом того, что в Elastic Stack есть поддержка протокола netflow, обеспечивается возможность анализа сетевых протоколов в реальном времени.

Рассмотрим несколько решений, предлагаемых в научных публикациях, и сравним их с разрабатываемым прототипом.

Сравнительные характеристики рассмотренных прототипов представлены в таблице 1.

Рассмотрим ниже показатели, использованные для сравнения.

1. *Объем входных данных*. В анализируемых работах [26-30] приведены результаты тестов и экспериментов, в которых предоставлены сведения о тестировании разрабатываемых прототипов с использованием различных объемов данных. В сравнении с этими работами наш прототип использует 1 Гб данных, обрабатываемых в потоковом режиме. Это несколько выше, чем в работе [26], но значительно ниже, чем в остальных работах [27-29], но достаточно для подтверждения работоспособности системы и позволяет сделать вывод об удовлетворении требованиям по обработке и анализу данных в реальном времени.

2. *Количество серверов/узлов*, используемых для построения прототипа. Большинство рассмотренных прототипов [26-29] используют распараллеливание и распределение нагрузки на несколько потоков. Данный параметр определяет способность системы распределять нагрузку между различными узлами сети. В нашем прототипе использовался один физический сервер с представленными в таблице характеристиками, однако различные компоненты системы находились на разных виртуальных машинах.

3. *Время обработки данных*. Данный параметр показывает скорость индексирования и анализа загружаемых в систему данных. Провести сравнительный количественный анализ по данному параметру для работ [26-30] (ввиду отсутствия возможности проведения экспериментов на едином стенде) не представляется возможным. Однако с учетом используемых вычислительных ресурсов и времени обработки данных, полученных в результате исследования, можно предположить, что Elastic Stack является одним из самых производительных решений.

4. *Способ обработки данных*. Как видно из таблицы, большинство исследователей используют потоковую обработку данных [26-29]. Разрабатываемый нами прототип использует потоковую обработку данных, что является ключевым параметром для построения SIEM-системы нового поколения.

5. *Задачи, решаемые с помощью прототипа*. Каждое исследование, рассмотренное в таблице, направлено на решение конкретной задачи. Все представленные решения способны справиться с целью обработки больших потоков данных для оперативного выявления инцидентов информационной безопасности. Разрабатываемый нами прототип призван решить задачу создания распределенного поискового и аналитического ядра системы мониторинга и управления инцидентами, которое выполняет поиск по большим объемам разнотипных данных безопасности.

Таблица 1. Сравнительные характеристики рассмотренных решений

Рассматриваемая система	Объем входных данных	Количество серверов/ узлов	Время обработки	Способ обработки	Задачи, решаемые с помощью прототипа
Massive Distributed and Parallel Log Analysis For Organizational Security [27]	От 100 Мб до 500 Мб в тестовой выборке	До 8 slave-серверов в эксперименте	Время обработки составило 450 секунд при обработке 500 Мб трафика на 8 серверах. Авторы отмечают, что скорость работы прототипа может быть увеличена, если использовать несколько серверов хранения	Потоковая обработка данных, данные журналов загружаются на один сервер хранения информации	Разработка архитектуры распределенной обработки журналов инцидентов безопасности, которая позволяет анализировать большие массивы данных в облаке
VSS Monitoring. Leveraging a Big Data Model in the Network Monitoring Domain [28]	Более 3.5 экзабайт	Данные не предоставлены	Поддерживается скорость обработки трафика от 100 Мбит/с до 100 Гбит/с	Возможность обрабатывать данные на основе как пакетной, так и поточной обработки. Поддержка Hadoop/Spark	Возможность отделить сетевую аналитику от системы хранения и получения большей эффективности с помощью интеграции различных данных

Продолжение таблицы 1

Toward a Standard Benchmark for Computer Security Research. The Worldwide Intelligence Network Environment (WINE) [29]	Обрабатываемая информация (более 100 Тб) включает: - до 10 миллионов доменов URL-репутаций (30 Терабайт); - 2,5 миллиона e-mail-спам аккаунтов; - данные телеметрии антивирусного ПО с 130 млн. хостов; - репутационные базы с 50 млн. хостов; -примеры вредоносного ПО с ПК 200 стран мира	240 000 сенсоров по всему миру	Данные не предоставлены	Возможность обрабатывать данные на основе как пакетной, так и поточной обработки. Поддержка Hadoop/Spark	Возможность обработки данных, собираемых с миллионов хостов по всему миру с использованием ключевых полей безопасности
Using Large Scale Distributed Computing to Unveil Advanced Persistent Threats [30]	Репрезентативная выборка составила 74 гигабайта данных о 144 миллионах событий	Один физический сервер с 16 ядрами процессора и распределением ресурсов	Время обработки информации - 1500 сек. при 15 ядрах. Авторы делают предположение о возрастании скорости обработки при использовании различных жестких дисков и распределении нагрузки между различными серверами	Пакетная обработка данных	Предполагаемая схема детектирования атак должна интегрировать все инциденты безопасности, собранные организацией с использованием контекста, и на основе различных алгоритмов, детектировать возможную подозрительную активность, применяя MapReduce

Продолжение таблицы 1

RF (Random Forest) [26]	500 Гб	10 устройств, объединенных в кластер, используется 500 деревьев принятия решений	Эксперимент проводился для приложения в области медицины, в качестве входных данных использовалась информация по пациентам. Время обработки информации - 517,8 секунд	Классификация объектов проводится путем голосования, где каждое дерево принятия решений сопоставляет каждый объект определенному классу, в итоге выбирается то соотношение, которое соответствует наибольшему количеству деревьев	Алгоритм машинного обучения, способный обрабатывать данные с большим числом признаков и классов
Spark-MLRF [26]	500 Гб	10 устройств, объединенных в кластер, используется 500 деревьев принятия решений	Эксперимент проводился на примере приложения для медицины. В качестве входных данных использовалась информация о пациентах. Время обработки информации - 186,2 секунд	Основан на библиотеке Apache Spark Mllib	Метод распараллеливания работы алгоритма Random Forest

Продолжение таблицы 1

PRF (Parallel Random Forest) [26]	500 Гб	10 устройств, объединенных в кластер, используется 500 деревьев принятия решений	Эксперимент проводился на примере приложения для медицины. В качестве входных данных использовалась информация о пациентах. Время обработки информации - 101,3 секунд	Разработан на платформе Apache Spark. Для предварительной обработки и машинного обучения строятся деревья регрессии и классификации	Гибридный метод оптимизации распараллеливания работы алгоритма Random Forest
Разрабатываемая система на базе комплексирования средств Elastic Stack	Поток данных при индексировании - 1 Гб. Для обработки и анализа данных использовалась библиотека из 2,5 млн. записей объемом 768 Мб	Один физический сервер: 16 Гигабайт RAM ddr3; 4-х ядерный процессор частотой 1.9 ГГц; пропускная способность передачи данных - 10 Мбит/с	Отправка JSON-пакетов размером 330 байт. Время обработки потока из таких пакетов - 794 мс	Потоковая обработка данных	Распределенное, поисковое и аналитическое ядро должно выполнять поиск по большим объемам разнотипных структур данных

Продемонстрированный в экспериментах и представленный при сравнительном анализе большой потенциал предлагаемой системы при малых затратах вычислительных ресурсов, позволяет получить существенный выигрыш по сравнению с аналогами и дает возможность для проведения дальнейших исследований при больших объемах поступающей информации и событий безопасности.

7. Заключение. В статье представлена архитектура и реализованный прототип системы сбора, хранения и обработки информации и событий безопасности, основанной на технологиях больших данных. Для решения этой задачи проведен анализ релевантных работ и современных SIEM-продуктов, реализующих сбор,

хранение и анализ информации и событий безопасности. На основе этого предложена обобщенная архитектура системы мониторинга, удовлетворяющая представленным требованиям. Выбрано решение Elastic Stack, и на его базе реализован прототип системы для исследовательских целей. Проведены эксперименты, демонстрирующие работоспособность реализованного прототипа системы. Способность реализованного прототипа собирать, хранить, структурировать и анализировать разнородные данные с высокой производительностью, а также программная расширяемость системы предоставляют широкие возможности для дальнейшей разработки перспективной SIEM-системы.

Будущие исследования и разработки будут направлены на дальнейшее совершенствование архитектуры системы, исследование взаимодействия компонентов для обработки информации и событий безопасности, а также анализ, экспериментальную оценку параметров функционирования системы для поиска величин, при которых система сохраняет работоспособность, и выявление зависимости производительности прототипа от увеличения потока данных.

Литература

1. *Котенко И.В., Саенко И.Б.* Создание новых систем мониторинга и управления кибербезопасностью // Вестник Российской академии наук. 2014. Том 84. № 11. С. 993–1001.
2. *Котенко И.В., Саенко И.Б.* SIEM-системы для управления информацией и событиями безопасности // Защита информации. Инсайд. 2012. № 5(47). С. 54–65.
3. *Котенко И.В., Саенко И.Б.* Архитектура системы интеллектуальных сервисов защиты информации в критически важных инфраструктурах // Труды СПИИРАН. 2013. № 1(24). С. 21–40.
4. *Kotenko I., Polubelova O., Saenko I.* Data Repository for Security Information and Event Management in Service Infrastructures // Proceedings of the International Conference on Security and Cryptography (SECRYPT 2012). 2012. pp. 308–313.
5. *Котенко И.В., Саенко И.Б., Юсупов Р.М.* Новое поколение систем мониторинга и управления инцидентами безопасности // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. 2014. № 3(198). С. 7–18.
6. *Котенко И.В., Саенко И.Б.* Построение системы интеллектуальных сервисов для защиты информации в условиях кибернетического противоборства // Труды СПИИРАН. 2012. № 3(22). С. 84–100.
7. Big Data Analytics for Security Intelligence. Cloud Security Alliance. 2013. pp. 1–12.
8. *Zuech R., Khoshgoftaar T.M., Wald R.* Intrusion detection and Big Heterogeneous Data: a Survey // Journal of Big Data. 2015. pp. 1–42.
9. Apache Hadoop 2.7.2. URL: http://hadoop.apache.org/docs/current/ (дата обращения: 20.02.2017).
10. *Dean J., Ghemawat S.* MapReduce: Simplified Data Processing on Large clusters // Communications of the ACM. 2008. vol. 51. no. 1. pp. 107–113.
11. *Shim K.* MapReduce algorithms for big data analysis // International Workshop on Databases in Networked Information Systems. 2013. LNCS 7813. pp. 44–48.
12. Apache Storm. URL: http://storm.apache.org/ (дата обращения 20.02.2017).

13. *Santos O.* Network Security with NetFlow and IPFIX: Big Data Analytics for Information Security // Cisco Press. 2015. 320 p.
14. *Kavanagh K.M., Rochford O., Bussa T.* Magic Quadrant for SIEM // Gartner. 2016.
15. HPE Security ArcSight ESM. URL: https://saas.hpe.com/en-us/software/siem-security-information-event-management (дата обращения: 20.02.2017).
16. IBM Security QRadar SIEM. URL: http://www-03.ibm.com/software/products/en/qradar-siem (дата обращения: 20.02.2017).
17. Alienvault OSSIM. URL: https://www.alienvault.com/products/ossim (дата обращения: 20.02.2017).
18. Splunk Enterprise. URL: https://www.splunk.com/ru_ru/products/splunk-enterprise.html (дата обращения: 20.02.2017).
19. ManageEngine EventLog Analyzer. URL: https://www.manageengine.com/products/eventlog/ (дата обращения: 20.02.2017).
20. Cisco Systems OpenSOC. URL: https://github.com/OpenSOC/ (дата обращения: 20.02.2017).
21. Apache Metron. URL: http://metron.incubator.apache.org/ (дата обращения: 20.02.2017).
22. GitHub Apache Metron. URL: https://github.com/apache/incubator-metron/pulls (дата обращения: 20.02.2017).
23. Graylog. URL: https://www.graylog.org/ (дата обращения: 20.02.2017).
24. Documentation Graylog. URL: http://docs.graylog.org/en/2.2/pages/configuration/elasticsearch.html (дата обращения: 20.02.2017).
25. Elastic Stack. URL: https://www.elastic.co/ (дата обращения: 20.02.2017).
26. *Chen J. et al.* A Parallel Random Forest Algorithm for Big Data in a Spark Cloud Computing Environment // IEEE Transactions on Parallel and Distributed Systems. 2017. vol. 28. no. 4. pp. 919–933.
27. *Shu X., Smiy J., Yao D., Lin H.* Massive Distributed and Parallel Log Analysis For Organizational Security // IEEE Globecom Workshops. December 2013. pp. 194–199.
28. Leveraging a Big Data Model in the Network Monitoring Domain. White Paper. VSS Monitoring. 2014. URL: http://www.vssmonitoring.com/resources/whitepapers/Leveraging-a-BD-Model-Whitepaper.pdf (дата обращения: 03.12.2016).
29. *Dumitras T., Shou D.* Toward a Standard Benchmark for Computer Security Research: the Worldwide Intelligence Network Environment (WINE) // Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS'11). 2011. pp. 89–96.
30. *Giura P., Wang W.* Using Large Scale Distributed Computing to Unveil Advanced Persistent Threats // Science Journal. 2013. vol. 1. no. 3. pp. 93–105.

Котенко Игорь Витальевич — д-р техн. наук, профессор, заведующий лабораторией проблем компьютерной безопасности, Федеральное государственное бюджетное учреждение науки Санкт-Петербургский институт информатики и автоматизации Российской академии наук (СПИИРАН). Область научных интересов: безопасность компьютерных сетей, в том числе управление политиками безопасности, разграничение доступа, аутентификация, анализ защищенности, обнаружение компьютерных атак, межсетевые экраны, защита от вирусов и сетевых червей, анализ и верификация протоколов безопасности и систем защиты информации, защита программного обеспечения от взлома и управление цифровыми правами, технологии моделирования и визуализации для противодействия кибертерроризму. Число научных публикаций — 450. ivkote@comsec.spb.ru, http://www.comsec.spb.ru; 14-я линия В.О., 39, Санкт-Петербург, 199178; р.т.: +7-(812)-328-71-81, Факс: +7(812)328–4450.

Кулешов Артем Андреевич — бакалавр, Федеральное государственное бюджетное образовательное учреждение высшего образования «Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича» (СПбГУТ). Область научных интересов: информационная безопасность. Число научных публикаций — 3. gart9515@gmail.com; пр. Большевиков, 22, к. 1, Санкт-Петербург, 193382; р.т.: +79997509515.

Ушаков Игорь Александрович — старший преподаватель кафедры защищенных систем связи, Федеральное государственное бюджетное образовательное учреждение высшего образования «Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича» (СПбГУТ). Область научных интересов: большие данные, информационная безопасность, безопасность компьютерных сетей, технологии виртуализации. Число научных публикаций — 20. ushakovia@gmail.com; пр. Большевиков, 22, к. 1, Санкт-Петербург, 193382; р.т.: +79214106586.

Поддержка исследований. Работа выполнена при финансовой поддержке РНФ (проект № 15-11-30029).

DOI 10.15622/sp.54.1

I.V. KOTENKO, A.A. KULESHOV, I.A. USHAKOV

A SYSTEM FOR COLLECTING, STORING AND PROCESSING SECURITY INFORMATION AND EVENTS BASED ON ELASTIC STACK TOOLS

Kotenko I.V., Kuleshov A.A., Ushakov I.A. **A System for Collecting, Storing and Processing Security Information and Events based on Elastic Stack Tools.**

Abstract. The paper considers an approach to the design of a system for data and security events collecting, storing and processing based on Elastic Stack tools. The tasks of monitoring and incident management are analyzed; architectural solutions for monitoring systems are studied; requirements to such systems are defined; and the architecture of systems for data and security events collecting, storing and processing is suggested. The developed software prototype of such system is described, and the results of experiments are specified.

Key words: security information and event management, Big Data, SIEM systems, Elastic Stack, Elasticsearch, Logstash, Kibana.

Kotenko Igor Vitalievich — Ph.D., Dr. Sci., professor, head of computer security problems Laboratory, St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences (SPIIRAS). Research interests: computer network security, including security policy management, access control, authentication, network security analysis, intrusion detection, firewalls, deception systems, malware protection, verification of security systems, digital right management, modeling, simulation and visualization technologies for counteraction to cyber terrorism. The number of publications — 450. ivkote@comsec.spb.ru, http://www.comsec.spb.ru; 39, 14-th Line V.O., St. Petersburg, 199178, Russia; office phone: +7-(812)-328-71-81, Fax: +7(812)328–4450

Kuleshov Artem Andreevich — bachelor, Federal State Budget-Financed Educational Institution of Higher Education, The Bonch-Bruevich Saint - Petersburg State University of Telecommunications (SPbSUT). Research interests: Big Data, information security. The number of publications — 3. gart9515@gmail.com; 22 (b.1) pr. Bolshevikov, St.Petersburg 193232 Russia; office phone: +79997509515.

Ushakov Igor Aleksandrovich — senior lecturer of secure communications system department, Federal State Budget-Financed Educational Institution of Higher Education, The Bonch-Bruevich Saint - Petersburg State University of Telecommunications (SPbSUT). Research interests: Big Data, information security, computer network security, virtualization technologies. The number of publications — 20. ushakovia@gmail.com; 22 (b.1) pr. Bolshevikov, St.Petersburg 193232 Russia; office phone: +79214106586.

Acknowledgements. This research is supported by RSF (project № 15-11-30029).

References

1. Kotenko I.V., Saenko I.B. [Creating New Generation Cybersecurity Monitoring and Management Systems]. *Sozdanie novyh system monitoring I upravleniya kiberbezopasnostyu – Herald of the Russian Academy of Sciences.* 2014. vol. 84. no. 6. pp. 993–1001. (In Russ.).
2. Kotenko I.V., Saenko I.B. [SIEM-systems for security information and events management]. *SIEM sistemy dlya upravleniya informaciej I sobytiyami bezopasnosti – Protection of the information. Inside.* 2012. vol. 5. pp. 54–65. (In Russ.).

3. Kotenko I.V., Saenko I.B. [Architecture of the system of intelligent services to protect information in cyber warfare]. *Trudy SPIIRAN – SPIIRAS Proceedings*. 2012. vol. 1(24). pp. 21–40. (In Russ.).
4. Kotenko I., Polubelova O., Saenko I. Data Repository for Security Information and Event Management in Service Infrastructures. Proceedings of the International Conference on Security and Cryptography (SECRYPT 2012). 2012. pp. 308–313.
5. Kotenko I.V., Saenko I.B., Yusupov R.M. [New Generation of Security information and Event Management Systems]. *Novoe pokolenie system monitoring I upravleniya incidentami bezopasnosti – St. Petersburg State Polytechnical University Journal. Computer Science. Telecommunication and Control Systems.* 2014. vol. 3(198). pp. 7–18. (In Russ.).
6. Kotenko I.V., Saenko I.B. [Developing the system of intelligent services to protect information in cyber warfare]. *Trudy SPIIRAN – SPIIRAS Proceedings.* 2012. vol. 3(22). pp. 84–100. (In Russ.).
7. Big Data Analytics for Security Intelligence. Cloud Security Alliance. 2013. pp. 1–12.
8. Zuech R., Khoshgoftaar T.M., Wald R. Intrusion detection and Big Heterogeneous Data: a Survey. *Journal of Big Data. Springer*. 2015. pp. 1–42.
9. Apache Hadoop 2.7.2. Available at: http://hadoop.apache.org/docs/current/ (accessed 20.02.2017).
10. Dean J., Ghemawat S. MapReduce: Simplified Data Processing on Large clusters. Google Inc. 2004. pp. 1–13.
11. Shim K. MapReduce algorithms for big data analysis. International Workshop on Databases in Networked Information Systems. 2013. LNCS 7813. pp. 44–48.
12. Apache Storm. Available at: http://storm.apache.org/ (accessed 20.02.2017).
13. Santos O. Network Security with NetFlow and IPFIX: Big Data Analytics for Information Security. Cisco Press. 2015. 320 p.
14. Kavanagh K.M., Rochford O., Bussa T. Magic Quadrant for SIEM. Gartner. 2016.
15. HPE Security ArcSight ESM. Available at: https://saas.hpe.com/en-us/software/siem-security-information-event-management (accessed 20.02.2017).
16. IBM Security QRadar SIEM. Available at: http://www-03.ibm.com/software/products/en/qradar-siem. (accessed 20.02.2017).
17. Alienvault OSSIM. Available at: https://www.alienvault. com/products/ossim (accessed 20.02.2017).
18. Splunk Enterprise. Available at: https://www.splunk. com/ru_ru/products/splunk-enterprise.html (accessed 20.02.2017).
19. ManageEngine EventLog Analyzer [Official web site of EventLog Analyzer]. Available at: https://www.manageengine.com/products/eventlog/ (accessed 20.02.2017).
20. Cisco Systems OpenSOC. Available at: https://github.com/OpenSOC/ (accessed 20.02.2017).
21. Apache Metron. Available at: http://metron.incubator.apache.org/ (accessed 20.02.2017).
22. GitHub Apache Metron. Available at: https://github.com/ apache/incubator-metron/pulls (accessed 20.02.2017).
23. Graylog. Available at: https://www.graylog.org/ (accessed 20.02.2017).
24. Documentation Graylog. Available at: http://docs.graylog.org /en/2.2/pages/configuration/elasticsearch.html (accessed 20.02.2017).
25. Elastic Stack. Available at: https://www.elastic.co/ (accessed 20.02.2017).
26. Chen J. et al. A Parallel Random Forest Algorithm for Big Data in a Spark Cloud Computing Environment. IEEE Transactions on Parallel and Distributed Systems. 2017. vol. 28. no. 4. pp. 919–933.

27. Shu X., Smiy J., Yao D., Lin H. Massive Distributed and Parallel Log Analysis For Organizational Security. IEEE Globecom Workshops. 2013. pp. 194–199.
28. Leveraging a Big Data Model in the Network Monitoring Domain. White Paper. VSS Monitoring. 2014. Available at: http://www.vssmonitoring.com/resources/whitepapers/Leveraging-a-BD-Model-Whitepaper.pdf (accessed 03.12.2016).
29. Dumitras T., Shou D. Toward a Standard Benchmark for Computer Security Research: the Worldwide Intelligence Network Environment (WINE). Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS'11). 2011. pp. 89–96.
30. Giura P., Wang W. Using Large Scale Distributed Computing to Unveil Advanced Persistent Threats. *Science Journal*. 2013. vol. 1. no. 3. pp. 93–105.