Параллельное и распределенное программирование, грид-технологии, программирование на графических процессорах

УДК: 004.056.53 DOI 10.25559/SITITO.2017.3.629

Бондяков А.С.[1,2]

[1] Объединенный институт ядерных исследований, г. Дубна, Россия
[2] Институт физики НАН Азербайджана, г. Баку, Азербайджан

ОСНОВНЫЕ РЕЖИМЫ РАБОТЫ СИСТЕМЫ ПРЕДОТВРАЩЕНИЯ ВТОРЖЕНИЙ (IDS/IPS SURICATA) ДЛЯ ВЫЧИСЛИТЕЛЬНОГО КЛАСТЕРА

Аннотация

В данной статье ставится задача протестировать систему предотвращения вторжений Suricata и рассмотреть возможности ее использования в качестве основного или дополнительного инструмента для обеспечения безопасности вычислительного кластера.

Для решения поставленной задачи, протестирована работа данной системы в режимах IDS и IPS. Для оценки производительности тестируемой системы, приведены графики счетчиков мониторинга CPU utilization и CPU load average. Данные получены посредством системы мониторинга дата-центра института Физики НАН Азербайджана на базе платформы ZABBIX. Описана установка Suricata и настройка ее основных параметров. Показана возможность оптимизации режимов работы Suricata в зависимости от аппаратных ресурсов, например, количество ядер. В качестве полигона использовался облачный сегмент дата-центра института Физики НАН Азербайджана, который в свою очередь является частью облачной инфраструктуры ЛИТ ОИЯИ. Полученные результаты демонстрируют возможности Suricata обрабатывать поступающие данные не нагружая систему в целом, детектировать угрозы и своевременно реагировать на них что существенно повышает уровень безопасности. Кроме того, в данной статье показаны возможности облачного сегмента и системы мониторинга дата-центра института Физики НАН Азербайджана, с помощью которых проводилось тестирование.

Ключевые слова

Дата-центр; система предотвращения вторжений; облачные технологии; вычислительный кластер.

Bondyakov A.C.[1,2]

[1] Joint Institute for Nuclear Research, Dubna, Russia
[2] Institute of Physics, Azerbaijan National Academy of Sciences, Baku, Azerbaijan

THE BASIC MODES OF THE INTRUSION PREVENTION SYSTEM (IDS/IPS SURICATA) FOR THE COMPUTING CLUSTER

Abstract

This article aims to test the Suricata intrusion prevention system and consider its use as a primary or additional tool for securing the computing cluster. For solve this task, we tested the operation of this system in the IDS and IPS modes. To evaluate the performance of the system under test, the CPU utilization and CPU load average counters were used. The data was obtained through the system of monitoring the data center of the Institute of Physics of the National Academy of Sciences of Azerbaijan on the basis of the ZABBIX platform.

In this article, describes the installation of Suricata and the configuration of its main parameters and the possibility of optimizing the operating modes of Suricata is shown depending on the hardware resources, for example, the number of cores. The cloud segment of the data center of the Institute of Physics of the National Academy of Sciences of Azerbaijan was used as a testing ground, which in its turn is a part of the JINR LIT's cloud infrastructure.

The results demonstrate the ability of Suricata detecting threats and responding to them in a timely manner, which significantly increases the level of security. In addition, this article shows the capabilities of the cloud segment and the monitoring system of the data center of the Institute of Physics of NAS of Azerbaijan, with the help of which testing was conducted.

Keywords

Data center; intrusion prevention system; cloud technologies; computing cluster.

Введение

Вычислительный кластер представляет собой группу серверов, объединенную в единую высокоскоростную сеть. Основная задача вычислительного кластера – увеличение скорости обработки данных с помощью различных технологий распараллеливания. В силу ряда преимуществ, таких как масштабируемость, отказоустойчивость, а также экономических – хорошее соотношение стоимость/производительность, кластерные решения находят применение в различных областях информационных технологий.

В связи с этим наиболее важным компонентом кластерной системы считается вопрос информационной безопасности, а именно предотвращение несанкционированного доступа к его ресурсам. Для предотвращение несанкционированного доступа используются как правило различные системы брандмауэров такие как Netfilter управляемые утилитами iptables/firewalld и в некоторых случаях антивирусы. Однако, как показывает практика, возможностей этих систем оказывается недостаточно, например, в случае скрытого сканирования портов, кроме того, сетевой трафик пользователей регулярно повышается что в свою очередь увеличивает нагрузку на брандмауэры, которые могут не справится с обработкой большого потока данных используя стандартные возможности. Одним из решений данного вопроса может быть применение дополнительной системы предотвращения вторжений, которая могла бы работать в связке с брандмауэром и не нагружать при этом аппаратные ресурсы кластера.

Цель исследования

Цель исследования данной статьи состоит в подробном изучении возможностей системы предотвращения вторжений IDS/IPS Suricata, которое можно использовать в качестве основной или дополнительной системы повышающей безопасность доступа к ресурсам вычислительного кластера.

Основная часть

В настоящее время существует несколько IDS/IPS систем как коммерческих, так и open source. Долгое время одним из лидеров open source IDS/IPS систем был проект Snort [1] но на сегодняшний день возможности данной системы не позволяют обрабатывать большие потоки данных в силу плохой поддержки многопоточности. Система IDS/IPS Suricata [2] о которой пойдет речь далее, изначально создавалась для работы в многопоточном режиме, что является основным определяющим фактором для ее использования в кластерной системе. Возможности IDS/IPS Suricata дополняются также использованием GPU (CUDA, OPENCL). Многопоточный режим работы позволяет ей обрабатывать потоки данных до 10 Гбит/с., что является очень важным показателем применительно к вычислительному кластеру.

Suricata представляет собой модульную систему. Для сбора данных, захвата, декодирования, обнаружения вторжения используется отдельный модуль. Есть возможность извлекать и проверять файлы, передающиеся по HTTP, проверять сжатые данные, производить распознавание по URI, по данным cookie, заголовкам и многое другое. Благодаря модульной системе можно оперативно подключить новый модуль для перехвата и анализа данных. Для перехвата потоков имеются следующие интерфейсы: AF_PACKET, NFQueue, IPFRing, IPFW, Libpcap, PF_RING.

Suricata, в зависимости от конфигурации, может работать как в режиме IDS/IPS (режим предотвращения вторжений и режим детектирования) так и в режиме IDS, т.е. используя только средства детектирования.

Режим IPS реализуется посредством интерфейсов NFQueue, AF_PACKET, IPFW, PF_RING.

Рассмотрим реализацию IPS посредством NFQueue.

NFQueue, представляет собой связку с iptables в котором NFQueue становится частью цепочки правил. Пакет попадает в iptables и фильтруется по правилам NFQueue. При фильтрации пакет может получить статус NF_DROP, если соответствует таковой записи в правилах о блокировании, либо NF_ACCEPT – результат

успешной проверки, а также NF_REPEAT – повторная проверка. При включении большого количества правил, это наименее быстродействующий режим работы.

Режим IPS посредством AF_PACKET, представляет собой наиболее быстродействующий режим работы, его особенность заключается в использовании двух сетевых устройств, когда система работает в режиме роутера или шлюза. Пакет, в результате проверки, помеченный как небезопасный не пересылается на другое сетевое устройство.

Для тестирования вышеуказанных режимов Suricata, была использована облачная платформа дата центра института Физики НАН Азербайджана, являющаяся сегментом облачной инфраструктуры ОИЯИ [3], на которой была создана виртуальная машина со следующими параметрами: 10 ГБ дискового пространства, 16 ГБ ОЗУ, 16 ядер ЦПУ. Операционная система виртуальной машины: Centos 6.8 x86_64.

Облачные ресурсы дата центра института Физики НАН Азербайджана [4-6] представлены платформой OpenNebula [7]. Аппаратные возможности ресурса – 50 ТБ дискового пространства, 256 ГБ ОЗУ, 32 ядра ЦПУ. Вычислительный кластер дата центра, построен по технологии torque/pbs+openmpi с интегрированным планировщиком заданий Maui. Мониторинг ЦПУ, локальной сети, интернет соединения, в режиме реального времени осуществляется средствами ZABBIX.

Рассмотрим установку Suricata из исходников на виртуальную машину с ОС Centos 6.8 x86_64, а также работу данной системы в выше перечисленных режимах. Suricata также может быть установлена в операционных системах: macOS, BSD и Windows.

Предварительно устанавливаются необходимые программы:

yum install wget gcc libpcap-devel pcre-devel libyaml-devel file-devel zlib-devel jansson-devel nss-devel libcap-ng-devel libnet-devel tar make libnetfilter_queue-devel lua-devel

В процессе конфигурирования и сборки скаченного и разархивированного пакета Suricata указываются следующие параметры:

./configure –prefix=/usr –sysconfdir=/etc –localstatedir=/var –enable-nfqueue –enable-lua && make && make install-full

Для формирования кэша динамических библиотек после установки рекомендуется воспользоваться утилитой *ldconfig*, которая сформирует необходимы ссылки для корректной работы Suricata.

В результате выполнения вышеуказанной команды конфигурирования и сборки Suricata устанавливается в каталог: */usr*

Конфигурационные файлы содержатся в каталоге: */etc/suricata/*

Каталог лог-файлов: */var/log/suricata/*

Параметр конфигурации *–enable-nfqueue* включает поддержку режимов NFQueue, AF_PACKET, параметр *–enable-lua* включает режим детектирования Lua-скриптов для мониторинга угроз различной сложности.

Скомпилировать программу можно стандартными командами *make, make install*, кроме того разработчики Suricata предлагают дополнительно три варианта автоматической установки:

1. *make install-conf*: выполняет стандартную команду *make install* и настраивает все необходимые для работы директории;
2. *make install-rules*: выполняет стандартную команду *make install*, а также загружает актуальный набор правил для режимов детектирования и предотвращения угроз;
3. *make install-full*: выполняет предыдущие две команды и настраивает необходимые параметры для первого запуска Suricata.

Проверить результат установки можно командой: *suricata –build-info.*

Основным конфигурационным файлом Suricata является файл *suricata.yaml*, расположенный в каталоге */etc/suricata/.* Для просмотра конфигурационных параметров используется команда *suricata –dump-config.*

В файле *suricata.yaml* необходимо указать значения переменным и выбрать режим работы Suricata. Наиболее важными переменными являются:

- *vars.address-groups.HOME_NET: указывается диапазон защищаемой сети*
- *vars.port-groups.HTTP_PORTS: указываются необходимые порты*
- *default-rule-path: указываются необходимые правила*
- *host-mode: данный параметр определяет режим работы Suricata*
- *host-os-policy: определяет политику для ОС*
- *app-layer: осуществляет проверку по определенным протоколам.*

Основные переменные отвечающие за настройку производительности:

1. *threading*
2. *detect-thread-ratio*
3. *defrag*
4. *stream*
5. *runmode*

Настройка переменных *threading* и *detect-*

thread-ratio позволяет оптимизировать режим работы Suricata в зависимости от аппаратных ресурсов, например, количество ядер. Переменные *stream* и *defrag* отвечают за оптимальную работу с сетевыми данными. Переменная *runmode* определяет режим поточности и очередности обрабатываемых задач.

Команда *suricata --list-runmodes* показывает возможности различных режимов реализации с помощью настройки данной переменной. Данная переменная может использовать следующие режимы реализации: auto, autofp, workers, single. Режим single – однопоточный режим, workers – многопоточный режим при котором захват пакета и его последующая обработка разделены.

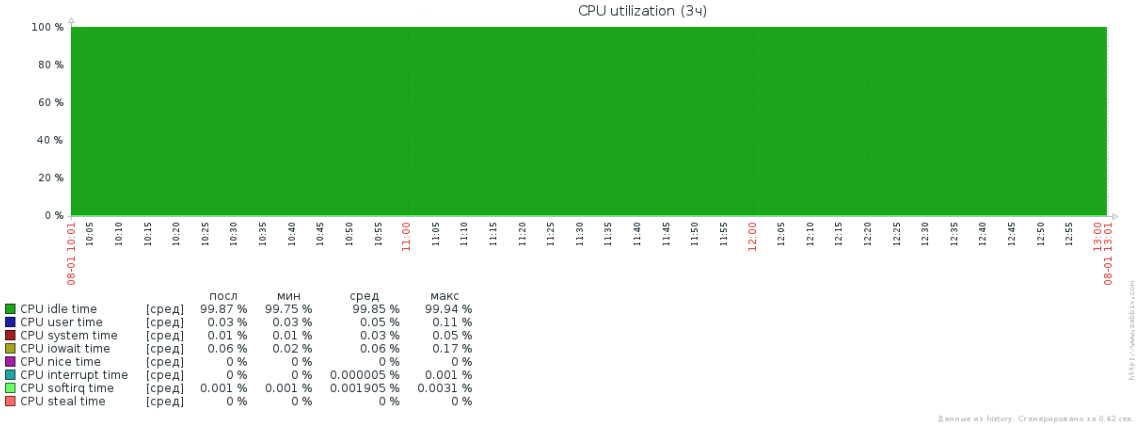
В зависимости от аппаратных возможностей можно подобрать оптимальный вариант. В данном случае выбран режим autofp при котором Suricata, работая в многопоточном режиме, распределяет все обрабатываемые сетевые потоки по принципу: один поток – одно ядро.

Рассмотрим режим работы IDS. В данном режиме, если система не выполняет функцию роутера, переменной *host-mode* можно указать значение «*sniffer only*», так как основная задача данного режима – детектирование угроз. Для запуска Suricata в данном режиме необходимо выполнить команду:

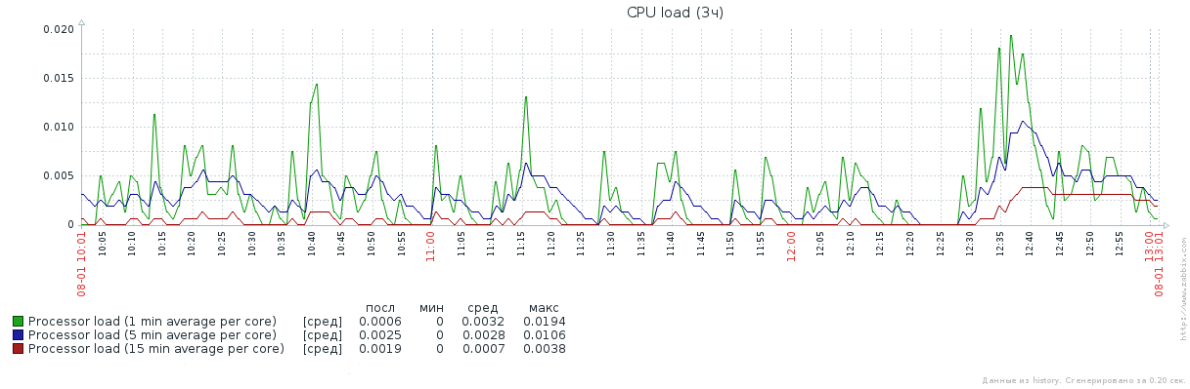
suricata –D –c /etc/suricata/suricata.yaml –i eth0

Предварительно, для выявления ошибок в конфигурационном файле рекомендуется выполнить команду: *suricata –c /etc/suricata/suricata.yaml –i eth0 –init-errors-fatal*

Для оценки производительности CPU при тестировании Suricata в режимах IDS и IPS использовались возможности системы мониторинга дата-центра института Физики НАН Азербайджана на базе платформы ZABBIX, а именно регистрировалась загрузка CPU посредством счетчиков CPU utilization и CPU load average, которые показывают загрузку CPU виртуальной машины, в рассматриваемых режимах. Счетчик CPU-utilization показывает среднее значение загрузки процессора в процентном эквиваленте, фиксируя время простоя процессора, нагрузку в пользовательском и системном режимах и т.д. CPU-load average фиксирует среднее значение загрузки системы за определенный период времени. С помощью данных счетчиков можно определить, с некоторой долей погрешности, нагрузку тестируемой системы на ресурсы виртуальной машины.



а)



б)

Рис.1 Загрузка CPU в режиме работы IDS (а – CPU utilization, б – CPU load average)

```
08/08/2017-12:43:14.375112  [**] [1:2010935:2] ET POLICY Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/08/2017-12:50:54.597223  [**] [1:2010936:2] ET POLICY Suspicious inbound to Oracle SQL port 1521 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/08/2017-12:52:54.276860  [**] [1:2008578:4] ET SCAN Sipvicious Scan [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/08/2017-12:52:54.276860  [**] [1:2011716:3] ET SCAN Sipvicious User-Agent Detected (friendly-scanner) [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/08/2017-13:13:37.755626  [**] [1:2010935:2] ET POLICY Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/08/2017-13:23:21.889824  [**] [1:2008578:4] ET SCAN Sipvicious Scan [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/08/2017-13:23:21.889824  [**] [1:2011716:3] ET SCAN Sipvicious User-Agent Detected (friendly-scanner) [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/08/2017-13:30:08.627127  [**] [1:2018789:3] ET POLICY TLS possible TOR SSL traffic [**] [Classification: Misc activity] [Priority: 3] {TCP}
08/08/2017-13:35:34.067533  [**] [1:2008578:4] ET SCAN Sipvicious Scan [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/08/2017-13:35:34.067533  [**] [1:2011716:3] ET SCAN Sipvicious User-Agent Detected (friendly-scanner) [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:22:15.630382  [**] [1:2010937:2] ET POLICY Suspicious inbound to mySQL port 3306 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:22:17.303143  [**] [1:2010936:2] ET POLICY Suspicious inbound to Oracle SQL port 1521 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:22:17.436085  [**] [1:2002911:6] ET SCAN Potential VNC Scan 5900-5920 [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP}
08/21/2017-15:22:18.034155  [**] [1:2002910:6] ET SCAN Potential VNC Scan 5800-5820 [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP}
08/21/2017-15:22:18.192780  [**] [1:2010939:2] ET POLICY Suspicious inbound to PostgreSQL port 5432 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:22:18.365210  [**] [1:2010935:2] ET POLICY Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:22:40.964671  [**] [1:2010935:2] ET POLICY Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:22:51.571858  [**] [1:2001219:20] ET SCAN Potential SSH Scan [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP}
08/21/2017-15:22:52.083640  [**] [1:2101390:6] GPL SHELLCODE x86 inc ebx NOOP [**] [Classification: Executable code was detected] [Priority: 1] {UDP}
08/21/2017-15:22:52.083640  [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:22:54.778772  [**] [1:2101390:6] GPL SHELLCODE x86 inc ebx NOOP [**] [Classification: Executable code was detected] [Priority: 1] {UDP}
08/21/2017-15:22:54.778772  [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
```

Рис.2 Вывод файла /var/log/suricata/fast.log (Suricata в режиме IDS)

На рисунке 1 показаны графики загрузки процессора виртуальной машины при запуске Suricata в режиме IDS, при тестовом сканировании портов сетевым сканером nmap в течение трех часов.

Как видно из графика – загрузка процессора виртуальной машины в данном режиме минимальна. Результаты детектирования, а именно выявление сканирования портов было зарегистрировано системой Suricata (рис 2), в файле */var/log/suricata/fast.log.*

На Рис.2 Suricata в режиме IDS детектирует сканирование портов при тестовом сканировании сетевым сканером nmap.

Как было сказано выше, одной из основных задач Suricata является предотвращение угроз, а именно режим IPS. Рассмотрим реализацию данного режима посредством NFQueue.

Для реализации режима IPS посредством NFQueue выполним команду:

suricata –c /etc/suricata/suricata.yaml –q 0 –D

Установим новую цепочку правил для iptables:

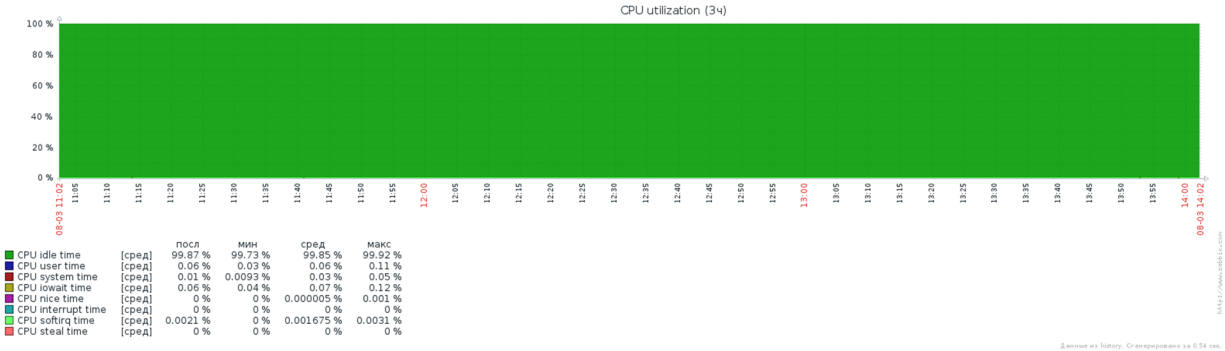
iptables –I INPUT –j NFQUEUE

Данная цепочка устанавливает очередь *–q 0* которую сканирует Suricata.

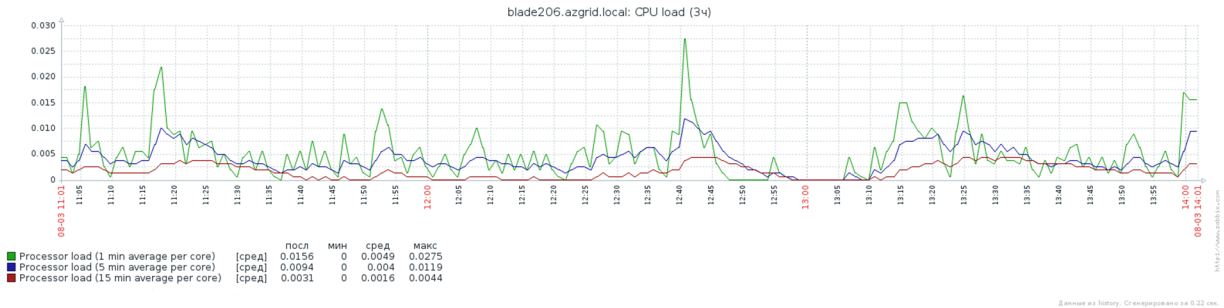
Предварительно, в конфигурационном файле suricata.yaml требуется указать переменные относящиеся к работе NFQueue:

NFQueue предлагает три варианта исполнения:

- accept: при котором пакет принимается или отклоняется как угроза на основании действующих правил Suricata и не фильтруется цепочками iptables;
- repeat: пакет маркируется и фильтруется всеми цепочками iptables;
- route: пакет после того как принимается, перенаправляется в другую очередь отличную от *–q 0.* Обычно такой вариант используется если в системе установлено несколько сетевых сканеров.



а)



б)

Рис.3 Загрузка CPU в режиме работы IPS (а – CPU utilization, б – CPU load average)

```
08/21/2017-15:47:18.295042  [Drop] [**] [1:2002911:6] ET SCAN Potential VNC Scan 5900-5920 [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP}
08/21/2017-15:47:18.600972  [Drop] [**] [1:2002910:6] ET SCAN Potential VNC Scan 5800-5820 [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP}
08/21/2017-15:47:19.092484  [Drop] [**] [1:2010936:2] ET POLICY Suspicious inbound to Oracle SQL port 1521 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:47:19.237441  [Drop] [**] [1:2010936:2] ET POLICY Suspicious inbound to Oracle SQL port 1521 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:47:19.239290  [Drop] [**] [1:2010935:2] ET POLICY Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:47:19.398225  [Drop] [**] [1:2010935:2] ET POLICY Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:47:19.571055  [Drop] [**] [1:2010939:2] ET POLICY Suspicious inbound to PostgreSQL port 5432 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:47:19.710663  [Drop] [**] [1:2010939:2] ET POLICY Suspicious inbound to PostgreSQL port 5432 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
08/21/2017-15:47:28.060927  [**] [1:2402000:4502] ET DROP Dshield Block Listed Source group 1 [**] [Classification: Misc Attack] [Priority: 2] {TCP}
08/21/2017-15:48:12.175996  [Drop] [**] [1:2001219:20] ET SCAN Potential SSH Scan [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP}
08/21/2017-15:48:12.694123  [Drop] [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:48:13.128240  [Drop] [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:48:13.460615  [Drop] [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:48:13.793110  [Drop] [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:48:16.171932  [Drop] [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:48:16.605429  [Drop] [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:48:16.938577  [Drop] [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:48:17.270920  [Drop] [**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP}
08/21/2017-15:49:18.965179  [Drop] [**] [1:2001219:20] ET SCAN Potential SSH Scan [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP}
08/21/2017-15:49:44.295427  [**] [1:2402000:4502] ET DROP Dshield Block Listed Source group 1 [**] [Classification: Misc Attack] [Priority: 2] {TCP}
```

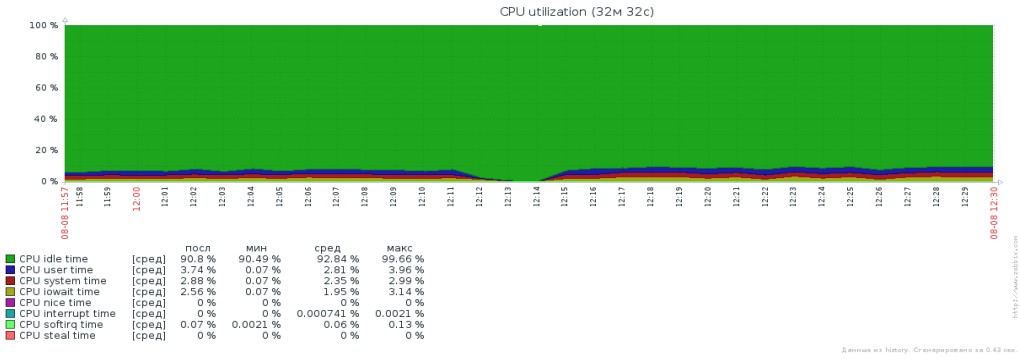
Рис.4. Вывод файла /var/log/suricata/fast.log (Suricata в режиме IPS)

В данном случае использовался вариант исполнения accept, как наиболее привлекательный с точки зрения нагрузки на систему и быстродействия в детектировании и предотвращении угроз.

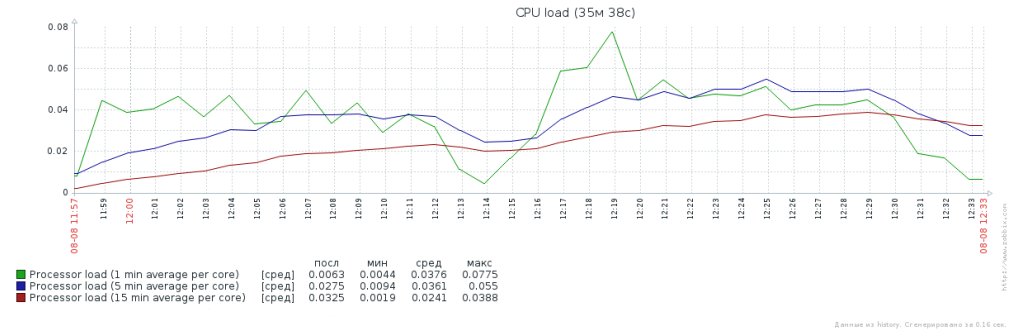
На рис 3 показаны графики загрузки процессора виртуальной машины при запуске Suricata в режиме IPS, при тестовом сканировании портов сетевым сканером nmap в течение трех часов. Загрузка процессора, как и в случае с IDS также минимальна.

Рис.4 наглядно демонстрирует возможности IPS по предотвращению вторжений, в данном случае выявлено сканирование портов и произведена блокировка атакующего.

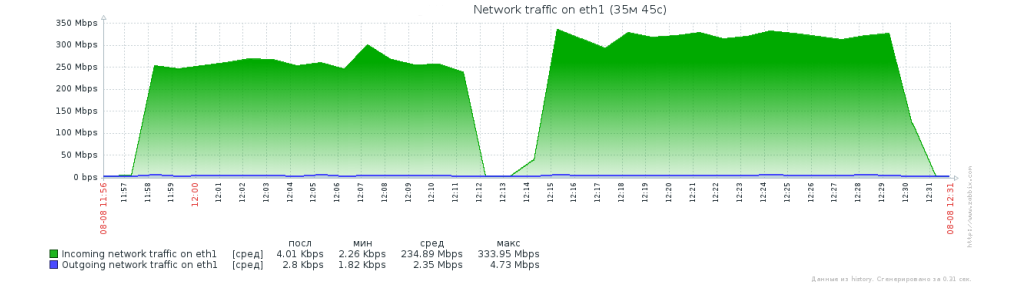
В рамках тестирования возможностей Suricata по обработке большого потока данных, производилась сетевая нагрузка посредством копирования группы файлов, размер каждого файла составлял более 20 ГБ. Результаты продемонстрированы на Рис.5.



а)



б)



в)

Рис.5 Загрузка CPU в режиме работы IPS при передаче больших файлов по сети (а – CPU utilization, б – CPU load average, в – Network traffic)

Полученные результаты

В результате проведенных исследований, получены данные счетчиков CPU utilization и CPU load average, которые показывают загрузку CPU виртуальной машины, в рассматриваемых режимах:

- графики загрузки процессора виртуальной машины при запуске Suricata в режиме IDS и IPS, при тестовом сканировании портов сетевым сканером nmap в течение трех часов, (рис.1- 4).
- график загрузки процессора виртуальной машины при тестировании возможностей Suricata по обработке большого потока данных (рис.5).

Заключение

Как видно из полученных данных, нагрузка на CPU, при использовании Suricata в режимах IDS и IPS, увеличивается незначительно, демонстрируя таким образом возможности Suricata обрабатывать поступающие данные не нагружая систему в целом. Для вычислительных кластеров, которые постоянно принимают и передают большое количество данных и кроме того подвергаются различным сетевым атакам, такое быстродействие Suricata, позволяет значительно повысить уровень безопасности. Резюмируя полученные результаты можно также отметить возможности облачного сегмента и системы мониторинга дата-центра института Физики НАН Азербайджана, с помощью которых проводились вышеописанные тесты.

Литература

1. Snort [электронный ресурс] // URL: https://www.snort.org (дата обращения 25.09.2017)
2. Suricata-ids [электронный ресурс] // URL: https://suricata-ids.org (дата обращения 25.09.2017)
3. Baranov A.V., Balashov N.A., Kutovskiy N.A., Semenov R.N. JINR cloud infrastructure evolution //Physics of Particles and Nuclei Letters. — 2016. — Vol. 13, Issue 5. — P. 672-675.
4. Abdinov O., Bondyakov A., Khalilova Sh., Orujova N. XXIV International Symposium NEC 2013, Conception GRID Infrastructure in Azerbaijan, p.9-12.
5. Bondyakov A.S. Basic directions of information technology in National Academy of Sciences of Azerbaijan // Computer Research and Modeling, 2015, T.7, №3,C657-660. (in Russian)
6. Bondyakov A.S. CEUR Workshop Proceedings, Vol-1787, urn:nbn:de:0074-1787-5, Инфраструктура и основные задачи дата-центра института физики НАН Азербайджана, P. 150-155 //http://ceur-ws.org/Vol-1787/150-155-paper-25.pdf
7. Opennebula [электронный ресурс] // URL: https://opennebula.org (дата обращения 25.09.2017)

References

1. Snort [jelektronnyj resurs] // URL: https://www.snort.org (data obrashhenija 25.09.2017)
2. Suricata-ids [jelektronnyj resurs] // URL: https://suricata-ids.org (data obrashhenija 25.09.2017)
3. Baranov A.V., Balashov N.A., Kutovskiy N.A., Semenov R.N. JINR cloud infrastructure evolution //Physics of Particles and Nuclei Letters. — 2016. — Vol. 13, Issue 5. — P. 672-675.
4. Abdinov O., Bondyakov A., Khalilova Sh., Orujova N. XXIV International Symposium NEC 2013, Conception GRID Infrastructure in Azerbaijan, p.9-12.
5. Bondyakov A.S. Basic directions of information technology in National Academy of Sciences of Azerbaijan // Computer Research and Modeling, 2015, T.7, №3,S657-660. (in Russian)
6. Bondyakov A.S. CEUR Workshop Proceedings, Vol-1787, urn:nbn:de:0074-1787-5, Infrastruktura i osnovnye zadachi data-centra instituta fiziki NAN Azerbajdzhana, P. 150-155 //http://ceur-ws.org/Vol-1787/150-155-paper-25.pdf
7. Opennebula [jelektronnyj resurs] // URL: https://opennebula.org (data obrashhenija 25.09.2017)

Поступила: 1.10.2017

Об авторе:

Бондяков Алексей Сергеевич, инженер-программист, Объединенный институт ядерных исследований; Институт Физики НАН Азербайджана, aleksey@jinr.ru

Note on the author:

Bondyakov Aleksey S., Software Engineer, Joint Institute for Nuclear Research, Institute of Physics; Azerbaijan National Academy of Sciences, aleksey@jinr.ru