

**Санкт Петербургский государственный
политехнический университет
Факультет технической кибернетики
Кафедра информационно-измерительных технологий**

**Методические указания для
лабораторных работ
по курсу
«Системы обнаружения вторжений»**

Составил:

доц. каф. ИИТ, к.т.н., Шалаевский О.Н.

Санкт-Петербург

2012

Содержание

Содержание.....	2
1. Введение.....	3
2. Система обнаружения вторжений Snort 6	
3. Лабораторная работа № 1. Организация виртуальной сети.....	14
4. Лабораторная работа № 2. Обнаружение атаки по сигнатуре.....	20
5. Лабораторная работа № 3. Обнаружение сканирования сети.....	22
6. Литература.	25

1. Введение.

Система обнаружения вторжений является программной или аппаратной системой, которая автоматизирует процесс анализа событий, возникающих в компьютерной системе или сети с точки зрения безопасности. Соответствующий английский термин — Intrusion Detection System (IDS). В дальнейшем для краткости изложения систему обнаружения вторжений будем называть COB.

На современном рынке существует множество различные COB. Приведём некоторые из них:

1. OSSEC является масштабируемой, кросс-платформенной хостовой системой обнаружения вторжений. Она имеет мощный компонент анализа, в неё встроен анализ логов, проверка целостности файлов, централизованная политика, обнаружение руткитов (rootkit), оповещение в режиме реального времени и активные ответные меры. COB работает в большинстве операционных систем, широко используемая. OSSEC очень активно развивается. Для корпоративных клиентов существует коммерческая поддержка. Данный продукт хорошо задокументирован.
2. Wro является сетевой системой обнаружения вторжений с открытым исходным кодом. Она является пассивной COB только для пользователей unix-подобных операционных систем. На сайте производителя утверждается, что данный программный продукт настоятельно рекомендуется использовать только как дополнение к уже установленной COB. Документация весьма скудная.
3. CATNET – это интеллектуальная система для обнаружения, анализа и регистрации инцидентов в сети. Она может обнаруживать аномалии и попытки сетевых вторжений, контролировать инфраструктуру

организации. Продукт предлагает быстрый и эффективный мониторинг сети, мониторинг безопасности, журнал событий для последующего анализа, поддержка продукта. К сожалению данная система весьма скудно документирована.

4. Snort – это продукт с открытым исходным кодом для обнаружения и предотвращения вторжений. Изначальной система умела только обнаруживать вторжения, но затем переросла в зрелую и более многофункциональную систему предотвращения вторжений. Система способна выполнять в режиме реального времени анализ трафика и регистрацию по ip-сети. Она заслужила всемирную известность, в связи с этим существует довольно большое число сообществ по поддержке данного продукта.

Ниже, в таблице 1 приводится сравнение этих COB по 4 признакам.

COB Параметр	Bro	CATNET	OSSEC	Snort
Бесплатность	+	–	+	+
Открытость исходных кодов	+	–	+	+
Мультиплатформенность	–	+	+	+
Графический интерфейс	–	+	+	–
Тип по мониторингу системы	сетевая	сетевая	хостовая	сетевая, хостовая

Табл.1.Сравнение COB

Для рассмотрения COB предпочтение было отдано Snort. При выборе системы руководствовались следующими принципами:

1. Бесплатность.

Данный фактор являлся существенным, так как все COB работают по одному принципу. При рассмотрении структуры и организации работы

одной из систем, можно легко ориентироваться в других в будущем.

2. Открытость исходного кода.

Данный продукт с открытым исходным кодом. Преимущество в том, что любой желающий может просмотреть и редактировать исходный код программы, внося свой вклад в развитие проекта. Благодаря этому Snort развивается быстрыми темпами.

3. Мультиплатформенность.

Snort разрабатывается для разных операционных систем. Проще говоря, установив и настроив COB на одном компьютере, его можно перенести без особых трудностей на другую машину с отличной от этой операционной системой. Данный продукт существует под такие операционные машины как Windows и семейство UNIX-подобных систем.

4. Простота в настройке.

Программный продукт хорошо документирован. На официальном сайте разработчика можно скачать руководство пользователя, в котором доступно описаны все возможности Snort и каким образом его можно сконфигурировать. Язык написания собственных правил гибкий и мощный.

5. Наличие различных сообществ.

Существуют много различных сообществ поддерживающих пользователей Snort. Также есть сообщества разрабатывающие правила для Snort, тем самым база правил постоянно совершенствуется для распознавания новых видов атак.

6. Включение в себя системы определения и предотвращения вторжений.

Snort включает в себя не только систему определения вторжения, но и систему предотвращения вторжений.

2. Система обнаружения вторжений Snort

Snort выполняет протоколирование, анализ и поиск по содержимому. Она также используется для активного блокирования или пассивного обнаружения целого ряда нападений и зондирований, таких как переполнение буфера, стелс-сканирование портов, атаки на веб-приложения, SMB-зондирование и попытки определения ОС. Программное обеспечение в основном используется для предотвращения проникновения в информационные системы, а также, блокирования атак, если они имеют место.

COB Snort в зависимости от параметров настройки можно классифицировать и как узловую, и как сетевую. Обычно она защищает определённый сегмент локальной сети от внешних атак из интернета. На Рис. 2.1. показана схема типовой сети. В рамках ЛВС, Snort выполняет функции узловой COB, а перед межсетевым экраном (МЭ) — функции сетевой COB. МЭ – это комплекс аппаратных и программных средств, осуществляющий контроль и фильтрацию проходящих через него сетевых пакетов в соответствии с заданными правилами. Переданный через сеть интернет пакет попадает в маршрутизатор и передаётся дальше в нужную подсеть. После этого пакет попадает на машину с COB Snort, которая просматривает не подпадает ли пакет под какое-либо, существующее в её базе, правило. Если такого правила не существует, то пакет передаётся дальше получателю, иначе же Snort передаёт межсетевому экрану соответствующие команды.

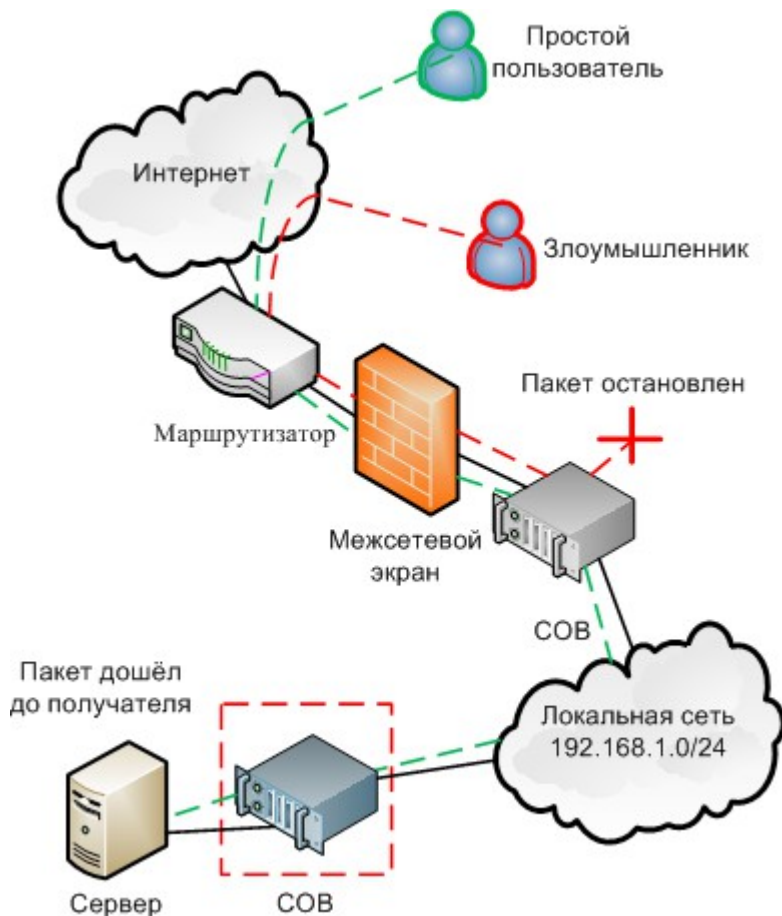


Рис. 2.1. Построение сети с установленной сетевой COB Short

Команды, передаваемые МЭ могут быть двух видов:

1. Система позволяет передавать пакет получателю;
2. Система запрещает передавать пакет получателю.

Существует также и другой способ, в котором сетевые пакеты из интернета попадают сначала не межсетевому экрану, а системе обнаружения вторжений. Первый способ более целесообразен нежели второй, так как нагрузка на машину со Snort уменьшается. На Рис. 2.1.

межсетевой экран, маршрутизатор и COB показаны как логические блоки, физически же они могут заключаться все в одном устройстве. Посмотрим, из каких же функциональных блоков состоит COB Snort. На Рис. 2.2. показаны компоненты, которых включает в себя Snort.

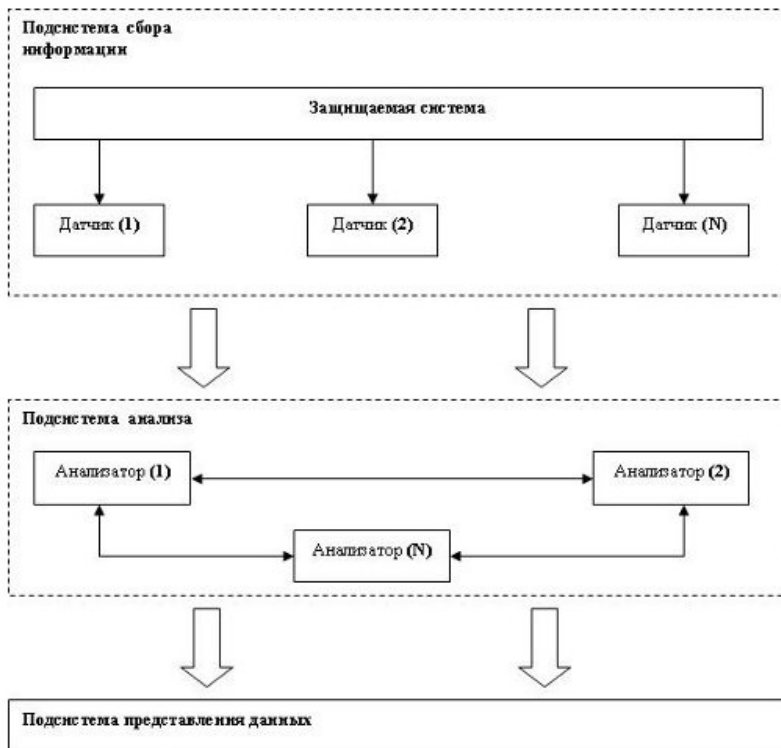


Рис. 2.2. Подсистемы COB Snort

Подсистема сбора информации – подсистема, занимающаяся сбором событий, которые связаны с безопасностью защищаемой системы.

Подсистема анализа – подсистема, предназначенная для выявления атак и подозрительных действий на основе данных сенсорной подсистемы.

Подсистема представления данных – подсистема, обеспечивающая накопление первичных событий и результатов анализа.

COB Snort включается в себя сниффер пакетов, который перехватывает все пакеты в своей подсети. На Рис. 2.2. показан принцип прохождения данных, полученных сниффером, через COB Snort.

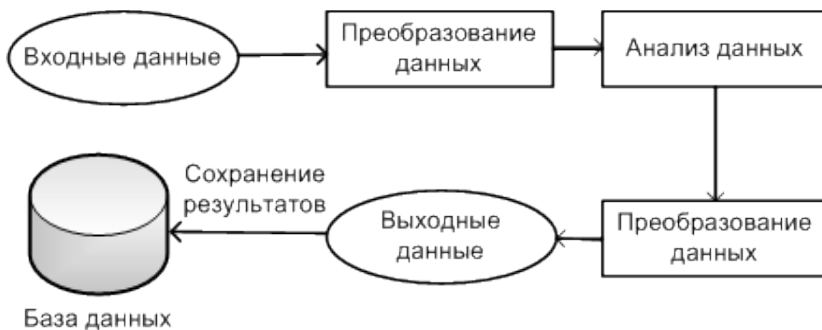


Рис. 2.2. Прохождение данных через Snort

Этапы прохождения данных через COB Snort можно представить в следующем виде:

1. Преобразование данных к пригодному для анализа виду. Это преобразование производится с помощью так называемого декодера;
2. Анализ данных с помощью на предмет выявления вторжений и атак;

Получение результатов и преобразование их к понятному для аналитика виду;

Фиксация результатов анализа в БД.

Snort может быть запущен в четырёх режимах.

1. **сниффер** – просто читает пакеты из сети и показывает их на экране в виде продолжительного потока в консоли.

Для запуска в данном режиме используется команда:

`./snort -v` – на экран выводятся только IP и TCP/UDP/ICMP заголовки пакетов, которые программа перехватывает в сети;

`./snort -vd` – в отличие от предыдущего случая ключ `d` обеспечивает ещё вывод на экран пакетных данных;

`./snort -dev` – ключ `e` обеспечивает дополнительный вывод данных на канальном уровне.

2. **пакетное журналирование** – читает пакеты из сети и записывает их на диск в так называемый лог-файл. Отличается от режима sniffера только тем, что вывод осуществляется не на экран, а в файл на диск.

Запуск Snort в данном режиме осуществляется таким же образом как и в режиме sniffера с добавлением ключа `l` после которого следует указать путь к каталогу где будут храниться лог-файлы. Если же указанного каталога не существует, то запуск программы будет завершён с ошибкой. Ниже приведён пример.

`./snort -dev -l /usr/local/var/logs/snort;`

`./snort -dev -l /home/user/logs -h 192.168.1.0/24`

`./snort -l /home/user/logs -b` – запись данных в бинарный лог-файл;

`./snort -dv -r packet.log` – чтение бинарного файла для дальнейшего анализа.

3. **сетевая система обнаружения вторжений** – Snort анализирует сетевой трафик и выполняет какие-либо действия в зависимости от вида атак.

`./snort -dev -c /usr/local/etc/snort/snort.conf`

4. **inline** – режим работы совместно с файерволом iptables. Перед запуском в этом режиме необходимо убедиться, что программа установлена с поддержкой данного режима. После этого следует настроить файервол для взаимодействия со Snort.

Три типа переменных могут быть определены в COB Snort:

⌚ var

⌚ portvar

⌚ ipvar

Три ключевые слова, приведённые выше, предназначены для присвоения указанным нами переменным значений. Синтаксис у них одинаков, отличие состоит в том, что `var`/`portvar`/`ipvar` используются для разных типов данных, и выглядит так:

`<var | portvar | ipvar> <название_переменной> <значение_переменной>`

Слово `var` используется для присвоения переменной пути к файлу или директории и для назначения переменной `ip`-адресов. Заметим, что ключевое слово `ipvar` применяется к переменным также для указания `ip`-адресов, но только с поддержкой IPv6. Слово `portvar` используется для задания переменных с номерами портов. Теперь приведём отрывок из файла конфигурации `snort.conf`. Полная его версия находится в приложении 1.

`var RULE_PATH /usr/local/etc/snort/rules` – указание расположения директории с правилами.

`var HOME_NET [192.168.1.0/24,!192.168.1.23]` – указание диапазона `ip`-адресов, которые мы будем защищать. При этом мы исключили один `ip`-адрес. Ничего не мешает здесь указать только `ip`-адрес своей машины, тогда Snort использовался бы как узловая СОВ.

`var EXTERNAL_NET any` – указание `ip`-адресов от которых мы будем защищать нашу сеть. В данном случае от всех адресов.

`portvar HTTP_PORTS [80,2301,3128,7777,7779,8000,8008,8028,8080,8180,8888,9999]`

`portvar FTP_PORTS 21`

`portvar SMB_PORTS [139,445]`

`portvar SSH_PORTS 22`

В предыдущих четырёх примерах мы просто определяем порты. Здесь тоже может использоваться отрицание, как и в случае с `ip`-адресами.

Для указания последовательности портов, запись может выглядеть так:
[12:17,1024:].

С помощью ключевого слова `include` можно подключать дополнительные файлы с настройками. В данном случае мы подключили несколько правил:

```
include $RULE_PATH/ftp.rules
```

```
include $RULE_PATH/ssh.rules
```

Сейчас рассмотрим, что же из себя представляют правила. Они состоят из заголовка и опционального поля. Опции заключены в круглые скобки. Общий синтаксис правил таков:

<действие_программы> <протокол> <ip-адреса> <порт> <направление действия правила> <ip-адреса> <порт> (опции).

К действиям программы относится:

1. `alert` – вывести предупреждающее сообщение, а потом записать данные пакета в лог-файл;
2. `log` – просто записать данные пакета в лог-файл;
3. `pass` – игнорировать пакет.

Протоколы с которыми работает Snort

1. `tcp`;
2. `udp`;
3. `icmp`;
4. `ip`.

В поле `ip-адреса` может указываться как диапазон `ip-адресов`, так и отдельные адреса.

В поле `порт` можно указать любой порт от 1 до 65535.

Направления действия правила может быть двух видов:

1. В одну сторону, обозначается “->”;
2. В обе стороны, обозначается “<>”.

Опций в программе Snort большое множество, приведём лишь основные:

1. msg – информирующее сообщение об угрозе;
2. content – поиск определённой сигнатуры в файле;
3. sid – указание номера правила;
4. rev – указание номера версии правила;
5. ref – указание ссылки с подробным описанием угрозы.

3. Лабораторная работа № 1.

Организация виртуальной сети.

Для практического применения системы Snort нам необходимо несколько компьютеров, объединённых в сеть. Отличительной особенностью является необходимость одновременного доступа к нескольким машинам в этой компьютерной сети. Для целей обучения нет необходимости получения доступа к какой-нибудь реальной компьютерной сети, поэтому для решения поставленной задачи будем строить модель вычислительной сети.

На современном рынке существует множество программных продуктов, созданных специально для решения подобных задач. Примером компании, занимающихся разработкой продуктов виртуализации, могут служить VMware, Parallels, Sun Microsystems. С помощью данных программ можно эмулировать оборудование реального компьютера. Созданная в этой среде машина называется виртуальной машиной. Виртуальная машина захватывает не все ресурсы компьютера сразу, пользователь может самостоятельно их ограничивать. К примеру, при создании виртуальной машины пользователь указывает объём жесткого диска, оперативной памяти, видео-памяти. На созданную виртуальную машину возможно произвести установку какого-нибудь программного обеспечения и комфортно с ним работать из своей же родной операционной системы. Для большей наглядности проиллюстрируем это рисунком 3.1.

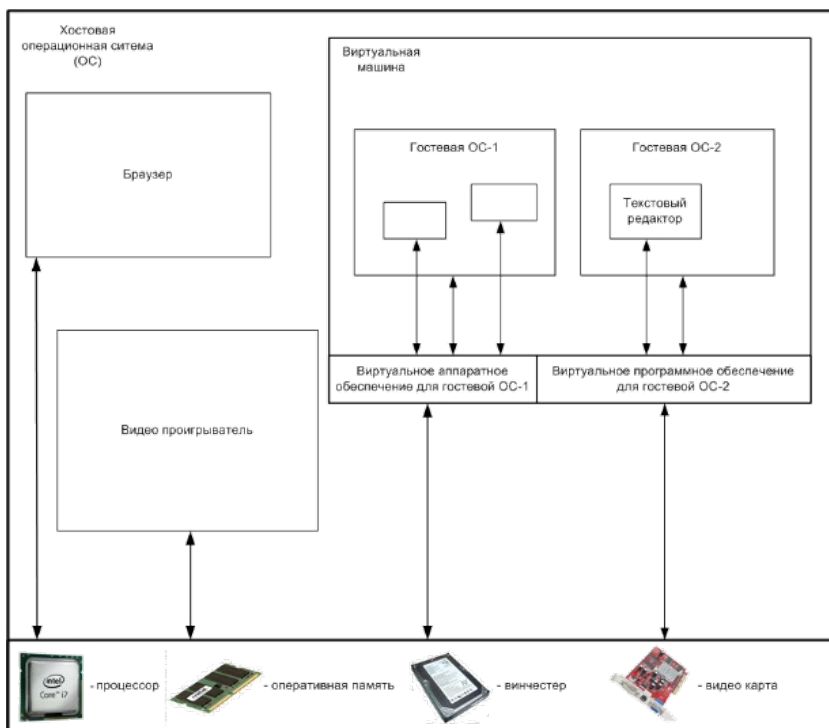


Рис. 3.1. Ресурсы компьютера виртуальной машиной

Здесь показан реальный компьютер, на котором запущены такие программы, как браузер, видео проигрыватель и соответственно виртуальная машина. Все они используют реальные ресурсы компьютера. Виртуальная машина же эмулирует оборудование этого компьютера для своих гостевых операционных систем. На рисунке видно, что разным системам виртуальной машины отводится разное количество системных ресурсов. Операционная система реального компьютера по отношению к системам виртуальной машины называется хостовой.

Для моделирования компьютерной сети надо создать несколько виртуальных машин, установить операционные системы на них и

объединить наши виртуальные компьютеры во внутреннюю сеть. Рассмотрим этапы моделирования сети с использованием выбранного нами программного продукта.

Выбор пал на кроссплатформенный продукт VirtualBox. Данная программа разрабатывается компанией Sun Microsystems. На виртуальную машину, созданную с использованием VirtualBox, можно установить такие операционные системы как Microsoft Windows, DOS, GNU/Linux, FreeBSD, Mac OS X, Sun Solaris/OpenSolaris. Приступим к созданию виртуальной машины.

По ходу создания виртуальной машин, система попросила нас установить объём оперативной памяти (Рис. 3.2.). Ограничимся 512 Мб памяти.

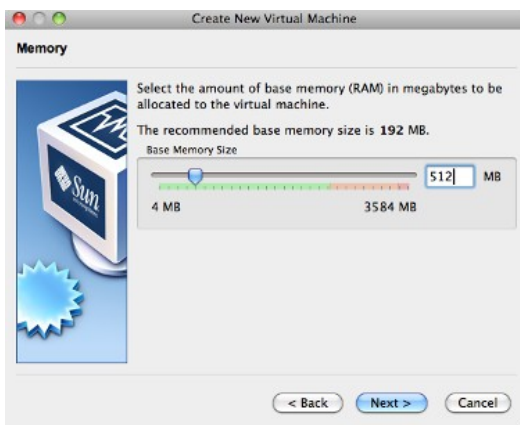


Рис. 3.2. Определение оперативной памяти

Затем указываем расположение файла с виртуальным диском на нашем физическом носителе, его название, а также его размер (Рис. 3.3.) Установим объём виртуального винчестера равный 10 Гб.



Рис. 3.3. Определение объёма жесткого диска

В настройке программы настраиваем сетевой интерфейс, выбрав внутренняя сеть (Рис. 3.4.).

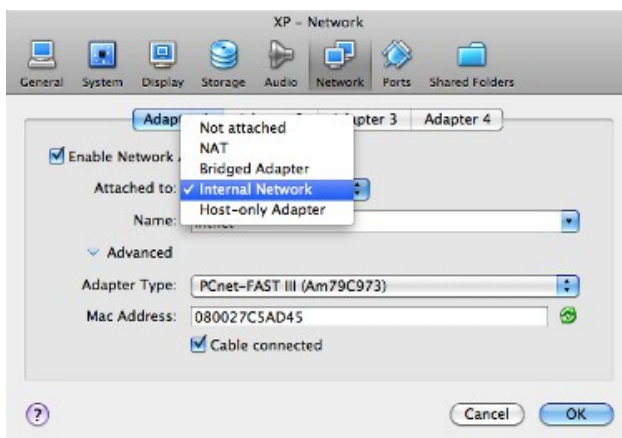


Рис. 3.4. Выбор сетевого интерфейса

В конечном итоге на Рис. 3.5. показаны параметры созданной виртуальной машины:



Рис. 3.5. Параметры виртуальной машины

В результате получили две виртуальные машины с установленными операционными системами. В каждой виртуальной машине надо настроить сеть, указав ip-адреса 192.168.1.2 и 192.168.1.3, соответственно, и маску подсети 255.255.255.0. После этого обе машины окажутся в одной подсети и смогут друг с другом взаимодействовать. Созданная нами вычислительная сеть будет являться полем для экспериментов.

Теперь посмотрим структуру созданной компьютерной сети (Рис. 3.6.). Три компьютера соединены между собой и находятся в одной подсети. Через хост-машину они также имеют доступ в интернет.



Рис. 3.6. Структура компьютерной сети

4. Лабораторная работа № 2.

Обнаружение атаки по сигнатуре.

Определение атак по сигнатурам довольно известная практика. В первых версиях Snort только по этому принципу и определялись атаки. Рассмотрим как же работает эта возможность в данном программном продукте. Возьмём какой-нибудь любой файл и откроем его в шестнадцатеричном редакторе (Рис. 4.1.). В этот файл любое место вставим нашу сигнатуру, на рисунке она выделена и представляет 4-байтовую последовательность: 0A 2D 42 C8.

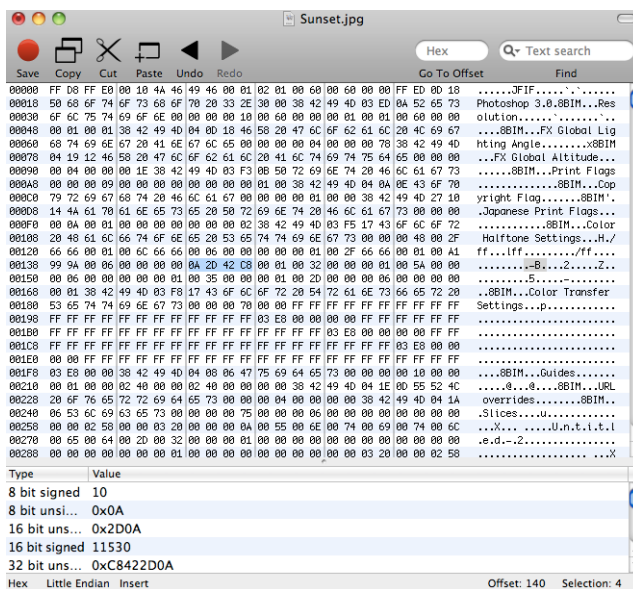


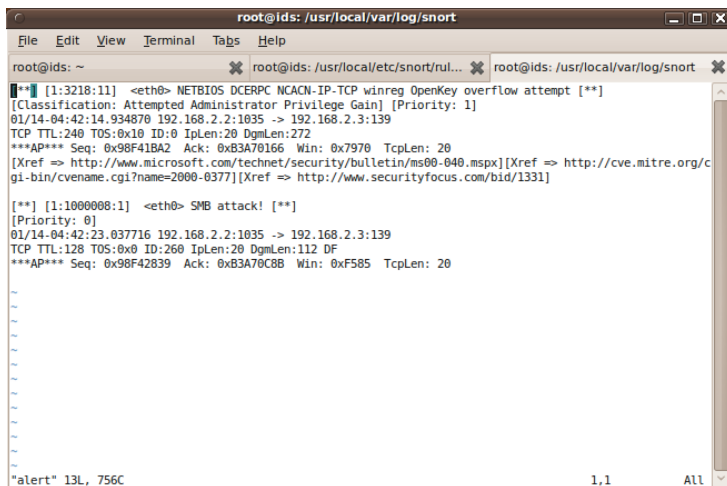
Рис. 4.1. Шестнадцатеричный редактор

После напишем своё правило, чтобы Snort все файлы с такой сигнатурой отлавливал и сохранял соответствующее предупреждающее сообщение. Правило будет выглядеть так:

alert tcp any any -> \$HOME_NET \$SMB (msg:"SMB attack!"; content: "0A 2D 42 C8"; sid: 1000004; rev:2;).

Данное правило добавим в файл local.rules, которая находится в директории с различными правилами.

Затем по протоколу SMB отправляем наш модифицированный файл. В результате система выдает нам сообщение, которое показано на Рис. 4.2.

A screenshot of a terminal window showing a Snort alert. The window title is 'root@ids: /usr/local/var/log/snort'. The alert message is: '[**] [1:3218:11] <eth0> NETBIOS DCERPC NCACN-IP-TCP winreg OpenKey overflow attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] 01/14-04:42:14.934870 192.168.2.2:1035 -> 192.168.2.3:139 TCP TTL:240 TOS:0x10 ID:0 Iplen:20 Dgmlen:272 ***AP*** Seq: 0x98F41BA2 Ack: 0xB3A70166 Win: 0x7970 TcpLen: 20 [Xref => http://www.microsoft.com/technet/security/bulletin/ms00-040.msp] [Xref => http://cve.mitre.org/cve/bin/cvename.cgi?name=2000-0377] [Xref => http://www.securityfocus.com/bid/1331]'. Below this, another alert is partially visible: '[**] [1:1000008:1] <eth0> SMB attack! [**] [Priority: 0] 01/14-04:42:23.037716 192.168.2.2:1035 -> 192.168.2.3:139 TCP TTL:128 TOS:0x0 ID:260 Iplen:20 Dgmlen:112 DF ***AP*** Seq: 0x98F42839 Ack: 0xB3A70C0B Win: 0xF585 TcpLen: 20'. At the bottom of the terminal, it says '"alert" 13L, 756C' on the left and '1.1 All' on the right.

```
root@ids: /usr/local/var/log/snort
root@ids: ~
root@ids: /usr/local/etc/snort/rul...
root@ids: /usr/local/var/log/snort

[**] [1:3218:11] <eth0> NETBIOS DCERPC NCACN-IP-TCP winreg OpenKey overflow attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
01/14-04:42:14.934870 192.168.2.2:1035 -> 192.168.2.3:139
TCP TTL:240 TOS:0x10 ID:0 Iplen:20 Dgmlen:272
***AP*** Seq: 0x98F41BA2 Ack: 0xB3A70166 Win: 0x7970 TcpLen: 20
[Xref => http://www.microsoft.com/technet/security/bulletin/ms00-040.msp] [Xref => http://cve.mitre.org/cve/bin/cvename.cgi?name=2000-0377] [Xref => http://www.securityfocus.com/bid/1331]

[**] [1:1000008:1] <eth0> SMB attack! [**]
[Priority: 0]
01/14-04:42:23.037716 192.168.2.2:1035 -> 192.168.2.3:139
TCP TTL:128 TOS:0x0 ID:260 Iplen:20 Dgmlen:112 DF
***AP*** Seq: 0x98F42839 Ack: 0xB3A70C0B Win: 0xF585 TcpLen: 20

"alert" 13L, 756C                                     1.1      All
```

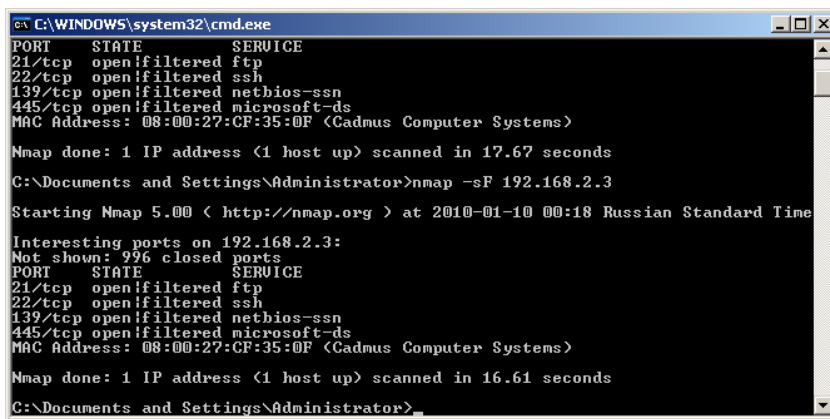
Рис. 4.2. Ответные действия системы на модифицированный файл

Из этого можно сделать вывод, что программный продукт Snort с лёгкостью обнаружил сигнатуру в файле. Это полезная опция, так как в любой файл злоумышленник может вмонтировать свой зловредный код и передавать его ни о чём не подозревающим пользователям.

5. Лабораторная работа № 3.

Обнаружение сканирования сети.

Рассмотрим одно из существующих аномальных правил. Предположим, что злоумышленник решил выявить уязвимость компьютеров сети путём сканирования их портов. Популярной утилитой с такими возможностями является Nmap. Она предназначена для настраиваемого сканирования IP-сетей с любым количеством объектов. Nmap использует множество различных методов сканирования. Для проверки работоспособности СОВ просканируем компьютер с ip-адресом 192.168.2.3 “невидимым” FIN методом. Для этого введём команду `nmap -sF 192.168.2.3` (Рис. 5.1.). После сканирования мы видим на этом же рисунке, что программа нам выдала результаты.



```
C:\WINDOWS\system32\cmd.exe
PORT      STATE      SERVICE
21/tcp    open|filtered ftp
22/tcp    open|filtered ssh
139/tcp   open|filtered netbios-ssn
445/tcp   open|filtered microsoft-ds
MAC Address: 08:00:27:CF:35:0F (Cadmus Computer Systems)

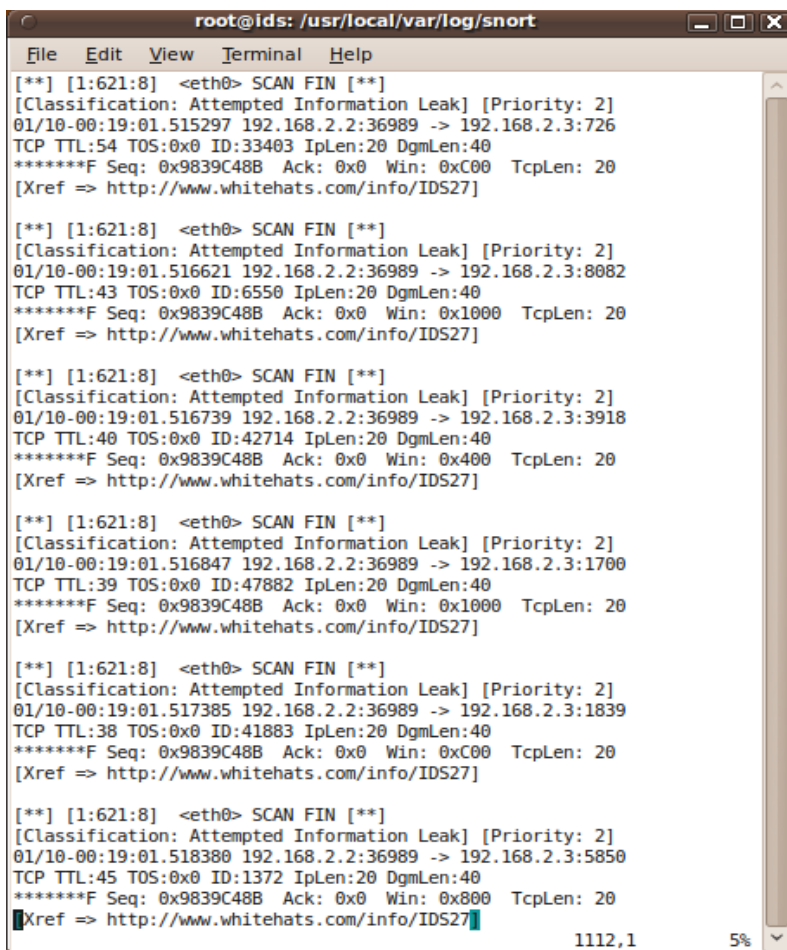
Nmap done: 1 IP address (1 host up) scanned in 17.67 seconds
C:\Documents and Settings\Administrator>nmap -sF 192.168.2.3
Starting Nmap 5.00 < http://nmap.org > at 2010-01-10 00:18 Russian Standard Time
Interesting ports on 192.168.2.3:
Not shown: 996 closed ports
PORT      STATE      SERVICE
21/tcp    open|filtered ftp
22/tcp    open|filtered ssh
139/tcp   open|filtered netbios-ssn
445/tcp   open|filtered microsoft-ds
MAC Address: 08:00:27:CF:35:0F (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 16.61 seconds
C:\Documents and Settings\Administrator>
```

Рис. 5.1. Сканирование портов с помощью nmap

После проведённых действий обратимся к сканируемому компьютеру и проверим, обнаружил ли Snort “неправильную” активность со стороны другого компьютера. После просмотра файла alert с предупреждениями (Рис. 5.2.) обнаруживаем, что СОВ засекла сканирование портов с

указанием ip-адресов сканирующей и сканируемой машины. Также в этом файле указан адрес в интернет с подробным описание атаки. Правила, отвечающие за выявление атак связанных со сканированием портов, хранятся в файле scan.rules.



```
root@ids: /usr/local/var/log/snort
File Edit View Terminal Help
[**] [1:621:8] <eth0> SCAN FIN [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/10-00:19:01.515297 192.168.2.2:36989 -> 192.168.2.3:726
TCP TTL:54 TOS:0x0 ID:33403 IpLen:20 DgmLen:40
*****F Seq: 0x9839C48B Ack: 0x0 Win: 0xC00 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS27]

[**] [1:621:8] <eth0> SCAN FIN [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/10-00:19:01.516621 192.168.2.2:36989 -> 192.168.2.3:8082
TCP TTL:43 TOS:0x0 ID:6550 IpLen:20 DgmLen:40
*****F Seq: 0x9839C48B Ack: 0x0 Win: 0x1000 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS27]

[**] [1:621:8] <eth0> SCAN FIN [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/10-00:19:01.516739 192.168.2.2:36989 -> 192.168.2.3:3918
TCP TTL:40 TOS:0x0 ID:42714 IpLen:20 DgmLen:40
*****F Seq: 0x9839C48B Ack: 0x0 Win: 0x400 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS27]

[**] [1:621:8] <eth0> SCAN FIN [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/10-00:19:01.516847 192.168.2.2:36989 -> 192.168.2.3:1700
TCP TTL:39 TOS:0x0 ID:47882 IpLen:20 DgmLen:40
*****F Seq: 0x9839C48B Ack: 0x0 Win: 0x1000 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS27]

[**] [1:621:8] <eth0> SCAN FIN [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/10-00:19:01.517385 192.168.2.2:36989 -> 192.168.2.3:1839
TCP TTL:38 TOS:0x0 ID:41883 IpLen:20 DgmLen:40
*****F Seq: 0x9839C48B Ack: 0x0 Win: 0xC00 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS27]

[**] [1:621:8] <eth0> SCAN FIN [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/10-00:19:01.518380 192.168.2.2:36989 -> 192.168.2.3:5850
TCP TTL:45 TOS:0x0 ID:1372 IpLen:20 DgmLen:40
*****F Seq: 0x9839C48B Ack: 0x0 Win: 0x800 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS27]
1112,1 5%
```

Рис. 5.2. Ответные действия СОВ на сканирование портов

Следует также отметить, что обнаружение аномалий является серьёзной компонентой выявления сетевых атак, так как она позволяет выявлять подозрительное, отклонённое от нормы поведение сетевого трафика в корпоративных сетях.

6. Литература.

1. Жигулин Г.П., Новосадов С.Г., Яковлев А.Д. Информационная безопасность, СПб: СПб ГУ ИТМО, 2003
2. SNORT Users Manual 2.5.8, October 22, 2009 (<http://www.snort.org>)
3. Русская группа пользователей Snort (<http://snortgroup.ru>)
4. В.Г. Олифер, Н.А. Олифер Компьютерные сети, 3-е издание ПИТЕР, 2006