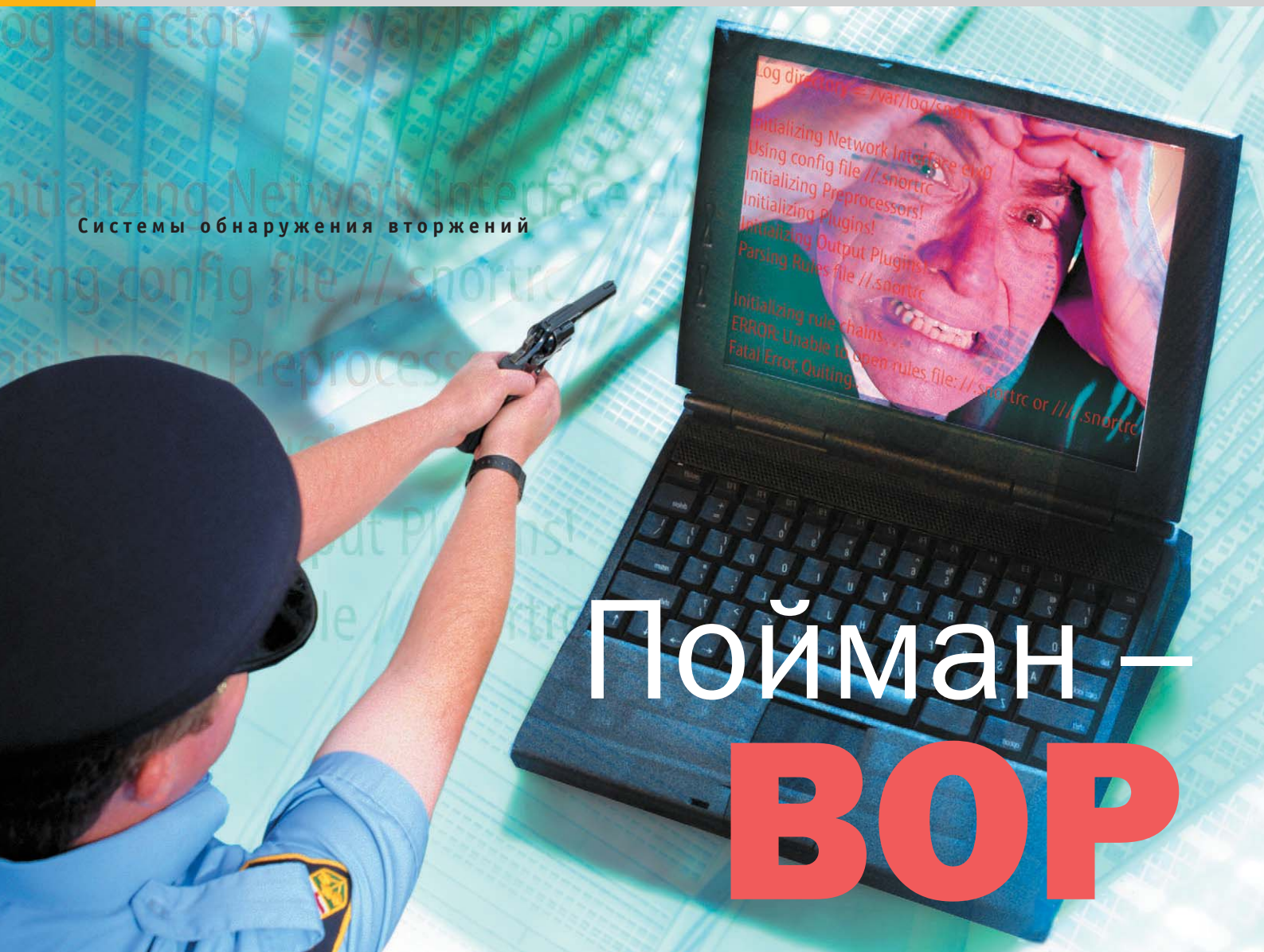
Системы обнаружения вторжений



Пойман — ВОР

Одной из важнейших задач обслуживания корпоративной сети является защита ее ресурсов от несанкционированного доступа извне. Существует множество способов отражения атак. Однако гораздо разумнее своевременно отслеживать попытки взлома и предупреждать предстоящее вторжение, чем потом устранять последствия неожиданного проникновения в систему.

IDS Snort

Snort — это система, которая, сочетая в себе свойства IDS двух видов, в режиме реального времени производит анализ сетевого трафика. Основной задачей Snort является обнаружение не только попыток взлома, но и попыток поиска незащищенных мест сети. Открытая архитектура Snort позволяет при необходимости наращивать ее функциональность для решения более широкого спектра задач.

Особенности инсталляции и конфигурирования

Дополнительные библиотеки

В поисках исходного кода Snort мы посетили ресурс www.snort.org и загрузили оттуда архив snort-1.8.2.tar.gz в каталог /export/install. Действуя в стандартной последовательности, мы его распаковали и попробовали создать файл конфигурации makefile. Результат оказался плачевным: выяснилось, что в архиве отсутствовала необходимая библиотека libcap.

Пришлось скачать ее исходный код по адресу ftp://ftp.ee.bl.lbl.gov/libcap.tar.z (указывал- »

» ся в распечатке ошибки). Оказалось, что это версия 0.4. В процессе конфигурирования библиотеки и при сборке (make) никаких нештатных ситуаций не возникло.

После установки библиотеки файл конфигурации, необходимый для инсталляции Snort, был создан, а вот запуск make выдал сообщение об отсутствии трех заголовочных файлов: pcap.h, pcap-named.h и bpf.h. Анализ проблемы показал, что для Solaris 8 они должны находиться в /usr/include, а не в /usr/local/include, как это было прописано по умолчанию, и при сборке libcap они не были установлены на место. Мы нашли данные файлы там, где собирали libcap (/export/install/libcap-0.4), и вручную скопировали pcap.h и pcap-named.h в /usr/include, а файл bpf.h в /usr/include/net.

После того как заголовочные файлы были перемещены в нужный каталог, сборка Snort завершилась успешно, и мы получили в каталоге /usr/local/bin исполняемый файл snort.

Первый запуск

Для начала мы решили запустить snort без ключей. Это дало следующий результат:

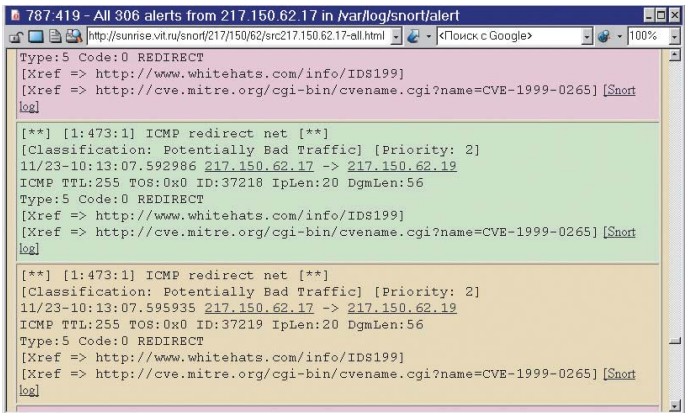
```
Log directory = /var/log/snort

Initializing Network Interface elx0
Using config file //.snortrc
Initializing Preprocessors!
Initializing Plugins!
Initializing Output Plugins!
Parsing Rules file //.snortrc

Initializing rule chains...
ERROR: Unable to open rules file: //.snortrc or ////.snortrc
Fatal Error, Quiting..
```

Итак, необходимо создать журнал в каталоге /var/log/snort. Следующая проблема — Snort не может найти файл с описаниями сигнатур атак, который должен называться .snortrc.

Но прежде всего необходимо было удостовериться, что сам snort собран правильно и может захватывать пакеты. Для этого необходимо вывести все перехваченные пакеты на консоль. Этот режим необходим, чтобы избежать создания огромного по размеру LOG-файла. После запуска весь экран мгновенно заваливает сообщениями.



▲ **Рис. 1.** Фрагмент отчета SnortSnarf

Intrusion Detection Systems

Платить или не платить...

Существует два вида систем обнаружения вторжения (IDS, Intrusion Detection Systems): к первому относят программы, следящие за функционированием системы и выявляющие различные подозрительные аномалии (например, слишком большое количество одновременно работающих процессов, увеличение исходящего сетевого трафика и т. п.); ко вторым относят IDS, которые ориентированы на поиск заранее известных признаков атаки.

В настоящее время на рынке систем IDS можно найти продукты самых различных производителей: eTrust фирмы Computer Associates International, Secure IDS от Cisco Systems, Centrax от CyberSafe, Dragon от Enterasys Networks, BlackICE компании Internet Security Systems, продукт Snort с открытым исходным кодом и многие другие.

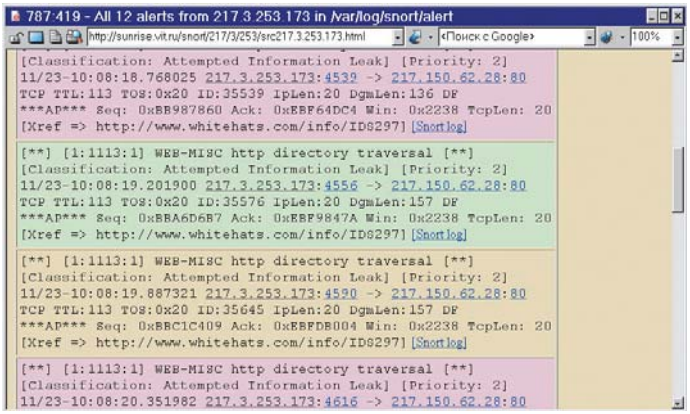
Уровень системы IDS определяется целым рядом параметров: надежность ядра, возможность удаленного управления, масштабируемость архитектуры, эргономичность интерфейса, способы обработки и корреляции данных. Согласно результатам тестирования, проведенного в университете DePaul в Чикаго, лидером стал продукт компании Enterasys Networks, также очень хорошие результаты показали IDS фирмы Cisco Systems и Snort.

Продукт	Производитель	Цена, $		Сайт
		сервер	консоль	
Dragon 4	Enterasis Networks	8500	7500	www.enterasys.com
Snort 1.7		free	free	www.snort.org
Real Secure 5.5	Internet Security	8995	free	www.iss.net
	Systems			
NFR Network	NFR Security	12500		www.nfr.com
Intrusion Detection				
NetProwler 3.5	Symantec	2995		www.symantec.com
SecureNet Pro 3.5	Intrusion.com	8495		www.intrusion.com
Centrax 2.4	CyberSafe	960	3000	www.cybersafe.com

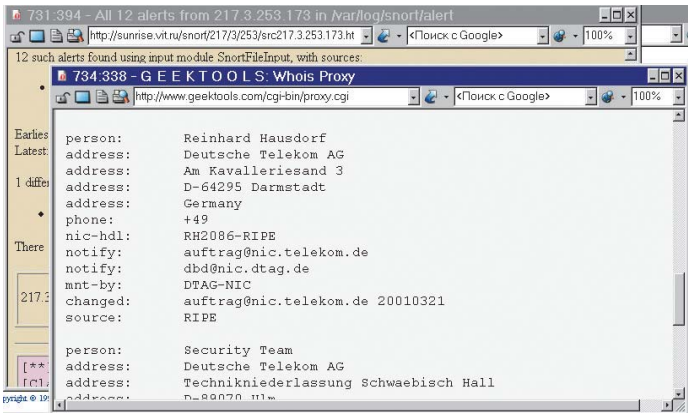
Учитывая, что нести какие-либо финансовые затраты на приобретение системы IDS нам не очень хотелось, нетрудно догадаться, какой именно продукт привлек наше внимание. Конечно, мы отдавали себе отчет в том, что установка и настройка Snort может повлечь за собой определенные затраты времени и усилий (обычно фирма — производитель коммерческого продукта обеспечивает покупателей документацией и технической поддержкой). Однако нас это сильно не пугало, и мы занялись сбором информации по данному вопросу.

Ссылки

- **http://www.silicondefense.com/software/snort_windows_installer** — Snort Windows Installer (beta)
- **http://www.silicondefense.com/software/snortsnarf** — генератор отчетов SnortSnarf
- **http://www.nessus.org** — сканер защищенности сети Nessus
- **http://www.vit.ru/vit/security** — системы защиты информации
- **http://www.compulog.ru/hackzone** — журнал HackZone
- **http://www.ypoku-xakepa.ru** — гуманистические уроки хакера



▲ Рис. 2. Атака на web-сайт с адреса 217.3.253.173



▲ Рис. 3. Geektools: информация о нарушителе

Файлы .conf и .rules

Попытки найти файл .snortrc ни к чему не привели. Начали искать по смыслу и обнаружили в каталоге файл snort.conf и множество файлов .rules. Предположим, что имеется две сети класса C, адреса которых условно можно обозначить как A.B.C.0 и X.Y.Z.0. В соответствии с этим устанавливаем следующие значения переменных в файле snort.conf:

```
var HOME_NET        [A.B.C.0/24,X.Y.Z.0/24]
var EXTERNAL_NET    any                     #пакеты из всех сетей
var SMTP            A.B.C.19,X.Y.Z.24       #наш почтовый сервер
var HTTP_SERVERS    [A.B.C.19, A.B.C.20,    #наши web-сайты
                    A.B.C.21,X.Y.Z.21]
var DNS_SERVERS     [A.B.C.19, X,Y,Z.24]    #наши DNS
```

Все остальные настройки можно оставить по умолчанию.

После очередного запуска snort мы получили в каталоге /var/log/snort все, что хотели, при этом для каждого подозреваемого в попытке атаки хоста создался подкаталог с именем, соответствующим его IP-адресу, и примерно таким содержимым.

```
-rw------- 1 root other 291 Nov 21 11:39 TCP:10018-80
-rw------- 1 root other 291 Nov 21 11:39 TCP:10035-80
...
-rw------- 1 root other 299 Nov 21 11:39 TCP:9775-80
```

В качестве примера приводим содержимое первого из них:

```
[**] WEB-IIS cmd.exe access [**]
11/21-11:39:09.023690 K.L.M.189:10018 -> A.B.C.19:80
TCP TTL:114 TOS:0x20 ID:51982 IpLen:20 DgmLen:136 DF
***AP*** Seq: 0x50AA695E Ack: 0x29274E7A Win: 0x27B4 TcpLen: 20
```

Смысл данного сообщения таков: хост, принадлежащий некоторой сети (условно обозначим его как K.L.M.189), пытался атаковать наш web-сайт, считая его построенным на базе Microsoft ISS.

Так как мы не используем Microsoft ISS, получать эту информацию нам нет необходимости. Открываем файл web-iis.rules и видим там 85 строк, не закрытых комментарием, каждая из которых представляет собой правило для проверки атаки на сервер ISS. При этом есть возможность добавления своих собственных правил.

Теперь необходимо минимизировать количество получаемых предупреждений. Внимательно просмотрев файл web-iss.rules и не найдя там сигнатур, которые имели бы отношение к нашему сайту, мы просто исключили его из перечня, поставив комментарий перед его названием в файле snort.conf. Таким же образом был исключен файл web-frontpage.rules, поскольку мы не пользуемся FrontPage. После этого мы проанализировали содержимое файла web-misc.rules и закомментировали в нем строки, содержащие сигнатуры атак, которые не имеют к нашим сервисам никакого отношения.

Язык описания атак

После мы занялись более тщательным изучением языка описания атак, так как появилась необходимость отслеживания трафика, не подпадающего под стандартно заданные описания. Для этого пришлось тщательно проанализировать правила, используемые в файлах, имеющих расширение .rules. Как выяснилось, синтаксис написания правил для Snort достаточно прост и в то же время эффективен. Рассмотрим, к примеру, правило, взятое из файла web-misc.rules:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:»WEB-MISC Cisco
IOS HTTP configuration attempt"; uricontent: "/level/"; uricontent:
"/exec/"; flags:A+; classtype:web-application-attack; reference:bugt-
raq,2936; sid:1250; rev:2;)
```

- Alert — вариант действия, которое будет выполняться, если проверяемый пакет соответствует условию, заданному правилом; в данном случае запись в журнале с уведомлением. Tcp определяет протокол передачи данных (возможны варианты: tcp, udp, icmp). Другие протоколы на данный момент не поддерживаются.
- $EXTERNAL_NET — это переменная, описанная в файле snort.conf. В общем случае этот параметр задает IP-адрес. Здесь возможны несколько вариантов: отдельный адрес, список адресов, any — все адреса, any — номер порта (any — любой порт). "->" — параметр, указывающий на направление трафика. В данном случае условию удовлетворяют все пакеты, идущие с адреса $EXTERNAL_NET (порт не имеет значения) на 80 порт web-сервера (или серверов, если их несколько) $HTTP_SERVERS (переменная $HTTP_SERVERS также определена в файле конфигурации snort.conf). Если данный параметр принимает значение <>, то направление трафика не учитывается.

Затем в круглых скобках указываются параметры, непосредственно характеризующие сам тип атаки:

- "msg:"WEB-MISC Cisco IOS HTTP configuration attempt"" — текст сообщения, обычно дает краткое описание вида атаки;
- "uricontent: "/level/"; uricontent: "/exec/"" — задает искомый шаблон в IRL (шаблон можно задавать как в текстовом виде, так и в шестнадцатеричном);
- "flags:A+" — задает условие на наличие TCP-флага (в данном случае проверяется TCP-флаг Ack, а пакеты с флагами Syn, Fin, Urg, Rst, Push игнорируются);
- "classtype:web-application-attack" — категория атаки;
- "reference:bugtraq,2936; sid:1250;rev:2" — необязательные параметры, характеризующие стандартные сигнатуры.

Пример правила

Определившись таким образом с форматом правила, мы решили отлавливать почтовые сообщения, приходящие с mail.ru на наш почтовый сервер, определенный как $SMTP в файле. Правило получилось таким:

```
alert tcp $EXTERNAL_NET any -> $SMTP 25 (msg: "SMTP Mail from
        mail.ru ";flags: A+; content: "@mail.ru"; nocase; classty-
        pe:attempted-user;)
```

Необходимо отметить, что здесь присутствуют дополнительные параметры: nocase — отключает чувствительность к регистру при анализе содержимого пакета; content: "@mail.ru" — задает искомый шаблон в содержимом пакета. Существует еще целый ряд параметров, задающих дополнительные условия, однако их немало, и мы не будем их рассматривать.

Генерация отчетов с помощью SnortSnarf

На сайте www.silicondefence.com мы нашли пакет SnortSnarf, позволяющий генерировать HTML-отчеты на основе LOG-файлов Snort. В нашем случае формат запуска оказался следующим (фрагмент полученного HTML-отчета показан на рис. 1.):

```
"/usr/local/snort/snortsnarf.pl -d /usr/local/apahe/htdocs/snort -split
        10 /var/log/snort/alert".
```

Пример практического использования

Мы решили вычислить кого-нибудь из нарушителей спокойствия наших сетевых ресурсов. Долго ждать попыток проникновения в систему не пришлось. Рис. 2 демонстрирует атаку на наш web-сайт с адреса 217.3.253.173. Сгенерированные с помощью SnortSnarf отчеты, помимо всего прочего, содержат ссылки на поисковые системы. Активизировав Geektools, мы попытались найти нарушителя в базе известных этой системе IP-сетей, и результат оказался успешным (рис. 3).

Мы не удержались и послали по этому адресу письмо с вложенными в него скриншотами из этой статьи и с краткими комментариями.

Заключение

Что ж, на этот раз результат оказался показательным. Несмотря на то, что продукт имеет статус freeware и для его установки пришлось приложить определенные усилия, мы наконец-то получили работающий инструмент для обнаружения вторжений. Хочется надеяться, что проделанная работа поможет вам оценить возможности Snort и контролировать сетевые ресурсы.

Анастасия Пшеницына, Юрий Калганников