УДК 681.324

И.Ю. Половко

МЕТОДЫ ТЕСТИРОВАНИЯ ЭФФЕКТИВНОСТИ СЕТЕВЫХ СИСТЕМ ОБНАРУЖЕНИЯ АТАК

Описываются подходы к тестированию производительности сетевых систем обнаружения атак, приводятся результаты тестов и их анализ.

Системы обнаружения атак; СОА; производительность; эффективность СОА; тестирование СОА.

I.Y. Polovko

METHODS OF TESTING THE EFFECTIVENESS OF NETWORK INTRUSION DETECTION SYSTEMS

Describe approaches to testing the performance of network systems to detect attacks, the results of tests and their analysis.

Intrusion detection system; IDS, productivity; IDS efficiency; IDS testing.

1. Тестирование эффективности

Разработка методики тестирования

На сегодняшний день на рынке присутствует большое количество коммерческих систем обнаружения вторжений (атак). Однако до сих пор не существует стандартизированной методики их тестирования, позволяющей выявить все достоинства и недостатки тестируемой системы обнаружения атак (СОА). Тесты, рекомендуемые производителем, как правило, служат рекламным целям, и не могут помочь оценить функциональные возможности системы [1].

Общие положения

Целью разработки средств тестирования системы обнаружения атак является создание программного комплекса, предназначенного для выполнения независимого тестирования систем для выработки заключения об их функциональных возможностях [2, 3].

Предлагаемые средства тестирования позволяют получить оценки выполнения/невыполнения СОА установленного набора функций, определяемого выполненными настройками, а также оценки уязвимости данных средств [4, 5].

Предполагается, что к моменту применения комплекса средств тестирования СОА развернута и настроена в соответствии с требованиями эксплуатационных документов и политики безопасности организации, эксплуатирующей СОА.

Критерии оценки для функциональных тестов

Обнаружение атак – способность СОА обнаруживать различные типы атак на основе анализа различных частей пакетов. Должны учитываться следующие основные характеристики СОА:

Способность анализировать заголовки – позволяет обнаруживать атаки, связанные со значениями заголовков IP-пакетов. Типичный представитель –

LAND-атака. LAND-атака посылает SYN пакет с одинаковыми значениями IP-адресов и портов источника и приёмника. Машина-приёмник попадает в бесконечный цикл, что может вызвать крах сети.

Сборка пакетов – возможность СОА собирать фрагментированный трафик и обнаруживать атаки, заключённые в нескольких пакетах. Типичный представитель – TearDrop-атака. Эта атака инициируется посылкой нескольких IP-пакетов, которые после сборки имеют перекрывающиеся данные пакетов. Это может вызвать нестабильность в функционировании сети или всей системы.

Анализ данных пакета – позволяет СОА обнаруживать атаки, связанные с данными пакетов. Типичный представитель такой атаки – так называемая phf-атака на HTTP. CGI-программа phf является примером сценария, который можно использовать для работы с адресной книгой, и эксплуатировать уязвимость, позволяющую локально запускать любые команды.

Также необходимо учитывать следующие критерии:

а) **Способность СОА определять IP Dsync** – обнаружение атак, при которых с целью маскировки задаются нестандартные значения номера последовательности и размера.

б) **Способность СОА обнаруживать распределённые атаки** – выявляются характеристики СОА, используя метод корреляции, обнаруживать атаки, распределённые во времени (между этапами атаки проходит какое-то время) или в пространстве (атака осуществляется с нескольких хостов с различными IP-адресами).

в) **Возможность сохранения информации для анализа** – характеризует возможности программы по сохранению информации об инцидентах для дальнейшего анализа.

г) **Наличие распределённой архитектуры** – очень важное свойство для СОА, применяющихся в больших сетях. Этот тест определяет архитектуру СОА и показывает способность работы консоли с несколькими сенсорами.

д) **Архитектура системы принятия решения** – показывает, где принимается окончательное решение об обнаружении атаки – на сенсорах или на консоли управления.

е) **Пропускная способность** – позволяет оценить способность СОА перехватывать пакеты, не вызывая их потерю. Для этого теста был использован только чистый трафик, не содержащий атак.

ж) **Влияние на производительность системы** – тест для оценки влияния работы СОА на загруженность центрального процессора и памяти, и общую производительность хоста.

Методические рекомендации по тестированию систем обнаружения атак

Проверка СОА осуществляется путем имитации атак или аномальных действий. При этом для успешного прохождения проверок, в базе сигнатур СОА должны присутствовать сигнатуры для каждого из производимых действий. Если эти сигнатуры отсутствуют, то они должны быть созданы вручную при подготовке и настройке СОА для тестирования. Топология стенда для тестирования представлена на рис. 1.

Каждый тест представляет собой внедрение специальных пакетов в сеть, в которой функционирует тестируемая система обнаружения атак. Результаты тестирования можно отслеживать на консоли управления системой. Тесты универсальны, они воспринимают тестируемую СОА как некий «чёрный ящик». Все тесты используют протокол TCP. В большинстве случаев в тестах используется взаимодействие между внедрёнными пакетами и третьей стороной – так называемым

«целевым» хостом, подвергающимся «атаке». Этот хост является целью назначения всех тестовых пакетов. Наличие такого «целевого» хоста позволяет сымитировать «реальное» TCP соединение с точки зрения тестируемой СОА. Для этих целей служит эмулятор сетевых сервисов.

Кроме того, целевой хост также играет роль проверки эффективности эксперимента. Его реакция на внедрённые пакеты позволяет наблюдать за поведением «реального» TCP-соединения и сравнивать его с информацией о поведении, отображаемой на консоли управления тестируемой СОА.

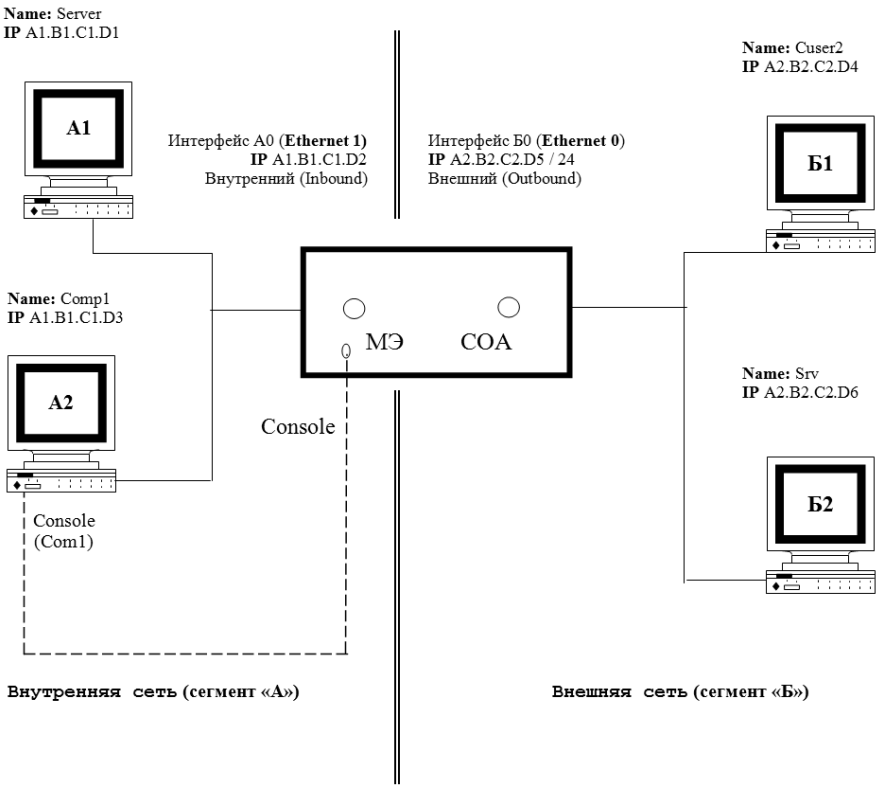

Рис. 1. Схема стенда для тестирования

В каждом тесте в сети при помощи программного обеспечения хоста-генератора нагрузки моделируется такой трафик, который мог бы появиться при нормальном функционировании сети.

Пакеты, внедряемые в трафик, слегка отличаются для каждого теста. Тестируемая СОА реагирует на каждый тест обнаружением или необнаружением атаки. Анализируя сообщения СОА и тип пакетов, использованных в конкретном тесте, можно сделать вывод о возможностях и особенностях тестируемой СОА. Перед тем, как проводить комплексные, специальные тесты, должна проводиться серия тестов базовых функций СОА. Цель этих тестов – убедиться, что СОА настроена правильно и функционирует во время проведения тестов, и что СОА, в принципе, способна обнаружить атаку. На целевом хосте для визуализации сетевой активности используется перехватчик пакетов. Контролируя эту активность, можно сказать, должна ли тестируемая СОА вообще обнаружить сымитированную атаку.

2. Тестирование производительности

Цели тестирования производительности

Тесты производительности проводятся в условиях, когда атакующий, жертва и СОА располагаются в одной сети. В этом случае тестам не мешают устройства, ограничивающие пропускную способность. Цель тестов производительности – определить характеристики работы СОА с пакетами [6]. Рис. 2 иллюстрирует топологию стенда для таких тестов.

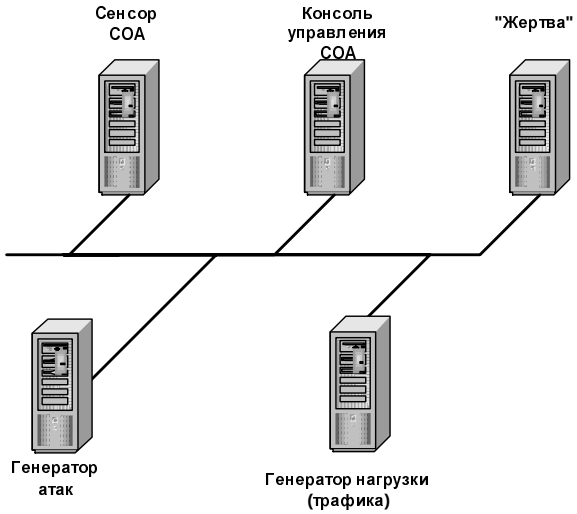

Рис. 2. Топология стенда для проведения тестов производительности

Методы тестирования производительности

Необходимо проводить несколько серий тестов при различной загрузке сети. Как минимум, необходимы две серии при загрузке 100 Mbps сети 6100 пакетов в секунду (~3%) и 25000 пакетов в секунду (~15%). Во всех тестах должны использоваться пакеты длиной 64 байта. Пакеты, содержащие атаки, генерируются с заданной скоростью 100 пакетов в секунду.

Размеры пакета должны быть выбраны таким способом, чтобы генерировать максимальное количество пакетов в секунду, что обычно является самым жестким режимом работы сетевого анализатора. Размер пакета, при котором отправлено максимальное количество пакетов, равен 101 байту.

Генератор трафика обычно использует всю доступную пропускную способность, не давая возможности передачи другого трафика. Для повышения точности результатов тесты были проведены несколько раз, после чего были выведены средние значения.

Трафик посылается несуществующему хосту, таким образом исключается любое взаимодействие между рабочими станциями. На втором хосте интерфейс устанавливается в различные режимы и производится захват трафика с использованием Win-Dump/TCPdump в различных конфигурациях. В зависимости от типа теста захваченные пакеты могут сохраняться на диск, выводиться на экран или пропускаться.

Для получения значения загрузки системы используются следующие программы: netperf в UNIX, task manager в Windows XP. Первая утилита разработана автором, вторая поставляется вместе с операционной системой. В тестировании были использованы: WinDump версии 4.0; TCPdump версии 3.4, libpcap версии 0.4.

Даже если в испытаниях суметь изолировать воздействие каждой из подсистем (BPF и фильтрация BPF, передача наверх), результаты не способны продемонстрировать производительность каждого компонента. Это происходит из-за различий в архитектуре разных версий, и не возможности изолировать каждый компонент от взаимодействия с другими и операционной системой. Самый представительный тест – это тест номер 3, который измеряет производительность "в целом", включая драйвер пакета, libpcap, WinDump, а также операционную систему (обработка ядром, запись данных на диск, и т.д). Причина этого в том, что производительность "целой системы" наиболее интересна пользователям.

Критерии оценки для тестов на производительность

Критерии оценки производительности следующие:

а) *Пропускная способность* – позволяет оценить способность СОА перехватывать пакеты, не вызывая их потерю. Для этого теста был использован только чистый трафик, не содержащий атак.

б) *Сборка пакетов* – тест для определения производительности сенсора при сборке пакетов. Для теста использовалась атака TearDrop, в базу сигнатур была загружена сигнатура только этой атаки.

в) *Эффективность фильтрации* – тест предназначен для оценки общей эффективности системы при решении задачи перехвата, разбора пакета и реагирования на атаку. Для тестирования использовалась атака LAND.

г) *Влияние на производительность системы* – тест для оценки влияния работы СОА на загруженность центрального процессора и памяти, и общую производительность хоста.

Описание тестов производительности

Тест 1: Производительность фильтра

Этот тест измеряет воздействие BPF-фильтра на процесс захвата. Пакеты, полученные из сети, перехватываются и проверяются фильтром BPF. Фильтр получает и обрабатывает все посланные пакеты. WinDump/TCPdump запущен со следующей командной строкой:

```
tcpdump 'filter',
```

где `'filter'` – фильтр пакетов в формате TCPdump. Этот тест выполняется с двумя различными фильтрами:

– `'udp'`: принимать только UDP-пакеты. Это сделано 5-ю командами.

– `'src host 1.1.1.1 and dst host 2.2.2.2'`: принимать пакеты от 1.1.1.1 для 2.2.2.2. Этот фильтр немного более сложен и устанавливается 13-ю командами.

Так как нет пакетов, удовлетворяющий этому фильтру, (потому что все пакеты сгенерированы повреждёнными специальным образом), фильтр отклоняет все пакеты. Таким образом, они не будут скопированы, и не будет взаимодействия с приложением. Только функции фильтрации будут использовать системные ресурсы.

Функция фильтрации не использует память, так что интересным является увидеть использование ею процессора.

Отличия между различным ОС очень ограничены. Это показывает, что реализация BPF в Windows оправдана и вполне способна конкурировать с оригинальным BPF. Загрузка центрального процессора изменяется на различных платформах, однако остается на приемлемых уровнях.

Это из-за того, что NDIS обычно вызывает функцию `packet_tap` прежде, чем DMA передача пакета закончена, давая небольшое преимущество BPF, в то время как `bpf_tap` вызывается в конце DMA передачи.

Значения производительности для двух фильтров очень похожи. Это подтверждает, что машина BPF-фильтрации хорошо оптимизирована, и что ее эффективность увеличивается с увеличением длины фильтра.

Тест 2: производительность драйвера

Для второго теста используется программа перехвата трафика. Это приложение получает все пакеты от драйвера (установка фильтра – принимать все), но дальнейшая обработка их не осуществляется, потому что libpcap функция `'callback'` пустая. Все пакеты обрабатываются на основных сетевых уровнях, затем драйвером, но на пользовательском уровне обработки нет. Этот тест показывает глобальную оценку эффективности драйвера, включая процесс копирования из интерфейса драйвера в буфер ядра и затем в пользовательский буфер.

UNIX имеет лучшую производительность, чем Windows, главным образом потому что `tap`-функция в UNIX очень проста и быстра, в Windows более сложная и медленная. Для более «длинных» пакетов использование центрального процессора под FreeBSD уменьшается, чего не происходит в Windows. Так получается из-за возможности UNIX BPF "отсроченной записи". При высокой интенсивности трафика загрузка центрального процессора различных операционных систем подобна. Однако частота системных вызовов (отсюда и загрузка центрального процессора) под UNIX значительно уменьшается, когда размер входящих пакетов увеличивается (то есть частота уменьшается), в то время как в Windows остается устойчивой.

Такое поведение – не проблема для Windows реализации, потому что время ЦП используется только, когда есть возможность. Все системы теряют малое количество пакетов [7].

Тест 3: Производительность захвата трафика

Это самый важный тест, потому что измеряет полный процесс сбора данных, для чего используется tcpdump. Фильтр не определен. Захваченные пакеты сохраняются на диск, для чего используем опцию `"-w"` tcpdump.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Network Based Intrusion Detection A review of technologies, Denmac Systems, Inc. [Электронный ресурс]. Режим доступа: www.denmac.com, свободный.

2. *Абрамов Е.С.* Разработка методов функционального тестирования СОА. // Сборник научных трудов XI Всероссийской научной конференции «Проблемы информационной безопасности в системе высшей школы». – М., 2004.

3. *Абрамов Е.С.* Система анализа защищённости от сетевых атак // Известия ТРТУ. Тематический выпуск. «Информационная безопасность». – 2003. – № 4 (33).

4. Профиль защиты БИЗКТ.МЭ.2.ПЗ. Межсетевые экраны. Класс защищенности – второй. Версия 1.0.

5. Профиль защиты БИЗКТ.СОА.2.ПЗ. Системы обнаружения сетевых атак. Класс защищенности – второй. Версия 1.0.

6. Network Based Intrusion Detection. A review of technologies. [Электронный ресурс] / Режим доступа: http://linkinghub.elsevier.com/retrieve/pii/S016740489980131X, свободный. – Загл. с экрана.

7. Benchmarking network IDS. [Электронный ресурс] / Режим доступа: http://archives.neohapsis.com/archives/sf/ids/2000-q4/0244.html, свободный. – Загл. с экрана.

Половко Иван Юрьевич
Технологический институт Федерального государственного образовательного учреждения высшего профессионального образования «Южный федеральный университет» в г. Таганроге.
E-mail: ivan.polovko@gmail.com.
347928, г. Таганрог, пер. Некрасовский, 44.
Тел.: 8(8634)371-905.
Кафедра безопасности информационных технологий; аспирант.

Polovko Ivan Yurevich
Taganrog Institute of Technology – Federal State-Owned Educational Establishment of Higher Vocational Education “Southern Federal University”.
E-mail: ivan.polovko@gmail.com.
44, Nekrasovskiy, Taganrog, 347928, Russia.
Phone: 8(8634) 371-905.
The Department of IT Security; post-graduate student.

УДК 681.324

К.И. Емельянов

РЕАЛИЗАЦИЯ АТАКИ НА ПРОТОКОЛ WPA2

В статье описывается практическая реализация атаки на подсистему обеспечения безопасности беспроводных протоколов TKIP (Temporal Key Integrity Protocol), использующуюся в WPA/WPA2.

Безопасность беспроводных протоколов; WPA; WPA2; TKIP; атака; точка доступа.

K.I. Emelianov

IMPLEMENTATION OF THE ATTACK ON THE PROTOCOL WPA2

The article describes the practical realization of the attack on the subsystem wireless security protocols, TKIP (Temporal Key Integrity Protocol), used in WPA/WPA2.

Wireless security protocols; WPA; WPA2; TKIP; attack; access point.

Анализ безопасности протокола WPA

Протокол WPA, хоть и является более защищённым, чем WEP, но имеет уязвимости, обусловленные используемым алгоритмом шифрования RC4 и использованием алгоритма CRC32, который не дотягивает по требованиям до криптографической хеш-функции.

На рис. 1 блок Network Data идет после MAC Header (чек-сумма FCS здесь не показана, так как ее обработка ведется на нижележащем уровне модели OSI). IV – это тот же 24-битовый WEP Initialization Vector, только смысл его несколько иной, структура данных же пакета значительно усложнилась.