чем, наибольшая эффективность достигается при их комплексной и продуманной реализации.

Литература

1. Смирнов, С.Н. Безопасность систем баз данных/С.Н.Смирнов .— М.: Гелиос АРВ, 2007, — 352 с.

Е.Э. Елисеев

АДАПТИВНЫЙ АЛГОРИТМ В СИСТЕМЕ ОБНАРУЖЕНИЯ ВТОРЖЕНИЙ SNORT ДЛЯ ПРЕДОТВРАЩЕНИЯ WEB УГРОЗ

(Самарский университет)

Система обнаружения вторжений (сокращённо СОВ) – программное или аппаратное средство, предназначенное для выявления фактов неавторизованного доступа в компьютерную систему или сеть либо несанкционированного управления ими через Интернет. СОВ обеспечивают дополнительный уровень защиты компьютерных систем [2].

СОВ используются для обнаружения некоторых типов вредоносной активности, которая может нарушить безопасность компьютерной системы. К такой активности, в том числе, относятся:

1. Сетевые атаки против уязвимых сервисов.
2. Атаки, направленные на повышение привилегий.
3. Неавторизованный доступ к важным файлам.
4. Действия вредоносного программного обеспечения (компьютерных вирусов, троянов и червей).

Обычно архитектура СОВ включает в себя четыре основных подсистемы:

1. Сенсорная подсистема, предназначенная для сбора событий, связанных с безопасностью защищаемой системы.
2. Подсистема анализа, предназначенная для выявления атак и подозрительных действий на основе данных сенсоров.
3. Хранилище, обеспечивающее накопление первичных событий и результатов анализа.
4. Консоль управления, позволяющая конфигурировать СОВ, наблюдать за состоянием защищаемой системы и СОВ, просматривать выявленные подсистемой анализа инциденты.

Каждая подсистема непосредственно связана с остальными, работа СОВ заключается в их совместном функционировании.

Существует несколько способов классификации СОВ в зависимости от типа и расположения сенсоров, а также методов, используемых подсистемой анализа для выявления подозрительной активности.

Snort — свободная сетевая система предотвращения вторжений (IPS) и обнаружения вторжений (IDS) с открытым исходным кодом, способная выполнять регистрацию пакетов и в реальном времени осуществлять анализ трафика в IP-сетях. Создана Мартином Рёшем, в дальнейшем развивалась и поддерживалась основанной им компанией Sourcefire (поглощена Cisco в 2013 году) [3]. Snort позволяет в реальном времени регистрировать и анализировать трафик в IP-сетях, выполнять определённые действия при обнаружении той или иной угрозы согласно составленным правилам (например, уведомление администратора, запись в журнал или блокирование подозрительных запросов).

Актуальные правила находятся в свободном доступе и обновляются самим сообществом Snort по мере обнаружения новых векторов атак и уязвимостей на различные сервисы и приложения.

Предлагается алгоритм для определения того, как Snort справляется с предотвращением WEB-угроз. Оценка эффективности набора правил Snort проводится на нескольких различных наборах данных (датасетах), каждый датасет состоит из некоторого количества HTTP-запросов, каждый запрос представляет из себя путь к запрашиваемому документу или скрипту на WEB-сервере и параметры. Запрос может быть GET или POST. Реакция Snort на другие данные в заголовках не проверяется. Определим тестовый датасет как датасет, в котором для каждого HTTP-запроса известен его тип – нормальный или аномальный (вредоносный). Реакция СОВ может быть двух типов – игнорирование запроса (Snort определяет запрос как нормальный) и уведомление о вероятной атаке (Snort определяет запрос как аномальный).

Под эффективностью набора правил Snort для тестового датасета будем понимать величину $E \in [0, 1]$, которая вычисляется по формуле (1). Чем выше эффективность – тем лучше СОВ справляется со своей задачей: обнаруживает подозрительный трафик и игнорирует нормальный.

$$E = \frac{T}{A}\ , \qquad (1)$$

где E – эффективность набора правил;
T – количество HTTP-запросов с правильно определённым типом;
A – общее количество HTTP-запросов в датасете.

Для определения реакции СОВ на запросы разворачивается сервер с ОС Linux с СОВ Snort с оцениваемыми набором правил и настройками СОВ (СОВ-сервер), затем с другого сервера (тестирующий сервер) каждый HTTP-запрос из датасета отправляется на порт 80 СОВ-сервера с помощью скрипта на ЯП Python. Предварительно проверяется, что Snort запущен и детектирует какую-либо подозрительную активность – в частности, пинг и сканирование портов. Используется следующий алгоритм:

1. Проверяется весь диапазон портов тестирующего сервера, из них отбираются те, которые могут использоваться как исходящие порты в скрипте

для тестирования. Обозначим количество таких портов N, а множество этих портов OP.

2. Тестируемый датасет разбивается на части, каждая размером не более чем N.
3. Для каждой части S датасета выполняется следующее:
   3.1. Фиксируется время начало тестирования, T0.
   3.2. Каждому запросу из части ставится в соответствие свой исходящий порт, т.е. строится биекция из множества запросов данной части во множество P.
   3.3. Выполняются все запросы данной части датасета в любом порядке таким образом, чтобы минимизировать время тестирования.
   3.4. Фиксируется время окончания тестирования, T1.
   3.5. Анализируется syslog СОВ-сервера на наличие уведомлений Snort во временном интервале [T0, T1] с учётом возможных погрешностей T0 и T1. В каждом уведомлении Snort сообщает в том числе исходящий порт, с которого был произведён запрос. Исходящие порты тестирующего сервера в уведомлениях данного временного интервала образуют множество D. Таким образом, зная биекцию между номером порта и запросом, можно определить, как Snort определил тип данного запроса – как нормальный (номер порта отсутствует во множестве D) или аномальный (номер порта присутствует во множестве D). Далее считается количество правильно определённых запросов. Также можно увидеть явно, на какой запрос было срабатывание и какой вид атаки СОВ Snort в нём обнаружил.
4. Объединяются результаты всех частей датасета.
5. Вычисляется эффективность набора правил.
6. Дополнительно вычисляется False Positive (FP, ошибка первого рода) – количество нормальных запросов, определённых как аномальные, False Negative (FN, ошибка второго рода) – количество аномальных запросов, определённых как нормальные, True Positive (TP, количество верно определённых нормальных запросов), True Negative (TN, количество верно определённых аномальных запросов). Затем вычисляется точность, полнота, F-мера.

Данный алгоритм позволяет проводить тестирование одного СОВ-сервера сразу несколькими тестирующими серверами, а также подключить дополнительные СОВ-сервера для распределения нагрузки во время тестирования. В этом случае при анализе уведомлений в выбранном временном интервале учитываются только те, где IP-адресом источника является конкретный тестирующий сервер. Это позволяет оценивать реакцию правил на DDOS-атаки, а также проводить моделирование работы СОВ-сервера, когда в одно и то же время поступают как нормальные запросы, так и аномальные.

Для оценки эффективности настройки системы Snort использовались по умолчанию, syslog отправлялся на тестирующий сервер. В качестве набора пра-

вил использовался Talos Rules от 2 мая 2019 года и дополнительно 5 правил обнаружения SQL-инъекций, полученные и экспериментально проверенные исследователями [7].

В результате проведения эксперимента получены результаты на датасетах (таблица 1) и вычислены показатели точности (Precision), полноты (Recall) и F-меры для датасета №1 (таблица 2). Для датасета №2 и датасета №3 данные показатели Precision и F-меры не вычислялись по причине неопределённости значения Precision.

Таблица 1 – Результат тестирования датасетов

№	Описание датасета	Количество обнаружений атак	False Positive	False Negative	Эффективность
1	CSIC 2010. Тестовый датасет, нормальный и аномальный трафик [8]	308	0	24129	0.601
2	Набор конструкций для XSS-атак, аномальный трафик [9]	22	-	313	0.066
3	Набор конструкций для SQL-атак, аномальный трафик [9]	44	-	264	0.143
4	GET-запросы некоторого WEB-сервера за год, неизвестный трафик	491	-	-	-

Таблица 2 – Результат вычисления точности, полноты и F-меры

№	Описание датасета	Precision	Recall	F-мера
1	CSIC 2010. Тестовый датасет [8]	1	0.599	0.749
2	Набор конструкций для XSS-атак [9]	-	0	-
3	Набор конструкций для SQL-атак [9]	-	0	-

Система обнаружения и предотвращения вторжений способна обнаружить и нейтрализовать некоторую часть уязвимостей WEB-приложений. Наличие системы обнаружения и предотвращения вторжений, подобной системе Snort, позволяет уменьшить количество возможных векторов атак. Разработана методика оценки эффективности набора правил и протестировано несколько датасетов, в результате которого был сделан вывод, что набор правил по умолчанию требует доработки для повышения уровня защиты от атак на WEB-приложения. Данную методику можно использовать для анализа поступающих запросов в реальном времени, чтобы избежать нагрузки на защищаемый сервер. Алгоритм тестирования можно использовать также на кластере с несколькими СОВ-серверами и тестирующими серверами, проводить моделирование атаки и

генерировать новые правила по новым поступающим данным в целях обнаружения большего количества актуальных векторов атак.

Литература

1. Anderson, James P., "Computer Security Threat Monitoring and Surveillance" Washing, PA, James P. Anderson Co., 1980.

2. Википедия. Система обнаружения вторжений [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Система_обнаружения_вторжений (дата обращения: 06.05.2019).

3. Википедия. Snort [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Snort (дата обращения: 06.04.2019).

4. OWASP Top 10 - 2017. The Ten Most Critical Web Application Security Risks [Электронный ресурс]. Режим доступа: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf (дата обращения: 06.04.2019).

5. OWASP Top 10 - 2013. The Ten Most Critical Web Application Security Risks [Электронный ресурс]. Режим доступа: https://www.owasp.org/images/f/f8/OWASP_Top_10_-_2013.pdf (дата обращения: 07.04.2019).

6. OWASP Топ-10 2017: обзор изменений [Электронный ресурс]. Режим доступа: https://www.securitylab.ru/blog/company/a1qa/343176.php (дата обращения: 06.04.2019).

7. Hussein Alnabulsi, Md Rafiqul Islam, Quazi Mamun. Detecting SQL injection attacks using SNORT IDS. URL: https://www.researchgate.net/publication/278677876_Detecting_SQL_injection_attacks_using_SNORT_IDS (дата обращения: 06.04.2019).

8. HTTP DATASET CSIC 2010 [Электронный ресурс]. Режим доступа: http://www.isi.csic.es/dataset/ (дата обращения: 12.05.2019).

9. GitHub. SecLists, the security tester's companion [Электронный ресурс]. Режим доступа: https://github.com/danielmiessler/SecLists (дата обращения: 12.05.2019).

Д.А. Зенцов

РАЗРАБОТКА АЛГОРИТМА ПРОВЕРКИ ПОДЛИННОСТИ ВИДЕОПОСЛЕДОВАТЕЛЬНОСТЕЙ ПУТЕМ ОБНАРУЖЕНИЯ В НИХ ПРЕДНАМЕРЕННЫХ ИЗМЕНЕНИЙ

(Самарский университет)

Сходство между соседними кадрами используется видеокодерами путем прогнозирования конкретного кадра в зависимости от его соседей и кодирования ошибок прогнозирования. Закодированное видео состоит из последовательности I-кадров (начало видеопоследовательности, содержит изображение