Таким образом, если пользователь авторизуется без доверенности или авторизуется с некорректной доверенностью, то он получает ровно те права, которые дают ему группы, в которых он изначально состоит (посредством функции UA). Однако при предъявлении корректной доверенности он может авторизоваться на некий набор групп, которым он изначально не обладал, но который делегировал ему пользователь-доверитель. Требования 1–4 выполняются благодаря использованию доверенностей, в которых явно указаны доверенное лицо, доверитель, определён срок действия, и все эти поля подписаны на закрытом ключе доверителя.

В реализации данного решения для операционной системы Linux в качестве доверенностей используются сертификаты X.509 версии 3 [1]. Сертификаты данного стандарта содержат следующие ключевые поля:

— имя эмитента (кто выдал сертификат);
— имя субъекта (кому выдан сертификат);
— период действия;
— расширения;
— подпись сертификата (с указанием алгоритма хэширования и подписи).

Поле «Расширения» представляет собой набор троек $(OID, criticalityFlag, Value)$, где OID (Object Identifier) используется для именования расширения; $criticalityFlag$ — флаг критичности; $Value$ — значение расширения. Расширения предоставляют возможность внедрения в сертификат произвольной информации до его создания.

Таким образом, сертификаты стандарта X.509 v3 могут использоваться в качестве доверенностей. Для этого в поле «Имя эмитента» необходимо указать имя пользователя-доверителя, в поле «Имя субъекта» — имя доверенного лица, в поле «Расширения» — набор делегируемых прав в системе. Доверителю необходимо также указать период действия доверенности в поле «Период действия» и подписать сертификат на своём закрытом ключе.

Создание доверенностей осуществляется при помощи криптографического инструмента OpenSSL [2]. Функция *assign*, авторизующая пользователя на делегированные группы (при предъявлении корректной доверенности), реализована в виде модуля PAM [3] — элемента ядра Linux.

ЛИТЕРАТУРА

1. `https://www.ietf.org/rfc/rfc5280.txt` — RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.
2. `http://www.openssl.org/` — OpenSSL: The Open Source toolkit for SSL/TLS.
3. `http://www.linux-pam.org/` — A Linux-PAM page.

УДК 004.94

ФОРМИРОВАНИЕ ВЕКТОРОВ ПОКАЗАТЕЛЕЙ ДЛЯ ОБУЧЕНИЯ НЕЙРОННЫХ СЕТЕЙ ПРИ ОБНАРУЖЕНИИ АТАК НА WEB-ПРИЛОЖЕНИЯ

С. Н. Сорокин

Представлен подход к оценке качества и выбора наиболее подходящих показателей для обучения нейронных сетей при решении задач обнаружения атак на web-приложения, предложена методика формирования векторов показателей для классов атак, позволяющая уменьшить количество нейронных сетей, используемых для обнаружения различных атакующих воздействий.

Ключевые слова: *обнаружение атак, обнаружение злоупотреблений, нейронная сеть, вектор показателей, классы атак, web-приложение.*

Обнаружение различных классов атак на web-приложения является актуальной задачей. Одним из перспективных подходов к построению систем обнаружения атак является подход, предполагающий использование нейронных сетей для поиска злоупотреблений [1 – 3].

Для создания систем обнаружения атак на базе нейронных сетей, работающих по принципу обнаружения злоупотреблений, целесообразно решать следующие задачи:

— формирование множества показателей для обучения нейронной сети, описывающих состояние наблюдаемой системы;
— формирование векторов показателей для обучения нейронной сети, позволяющих проводить обнаружение различных классов атакующих воздействий.

При формировании множества показателей, описывающих состояние web-приложения, нужно учитывать, что активность пользователей изменяется в зависимости от времени суток, дня недели или ввиду естественного изменения популярности web-приложения. Более подробно данные вопросы рассмотрены автором в [4].

Рассмотрим процесс формирования векторов показателей для обучения нейронной сети (рис. 1).

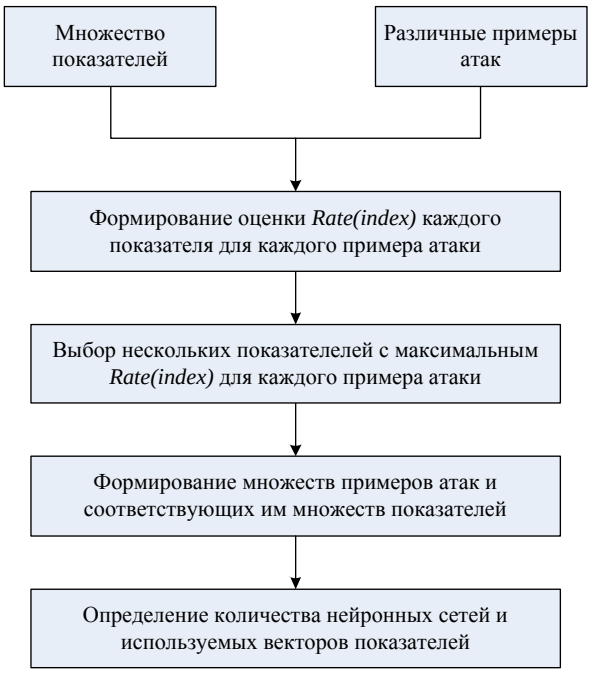

Рис. 1. Формирование векторов показателей

При формировании оценки показателя используются следующие свойства:

— амплитуда (разброс значений показателя);
— дифференциация (различие в среднем значении показателя на данных о поведении пользователей web-приложения и данных атаки);
— цикличность (свойство показателя иметь схожие значения в одинаковое время суток).

Общая оценка показателя вычисляется по формуле

$$Rate(Index) = \\ = (a \cdot Amplitude'(Index) + d \cdot Differentiation'(Index) + r \cdot Rhythm'(Index))/3,$$

где $Amplitude'(Index)$, $Differentiation'(Index)$, $Rhythm'(Index)$ — оценки в баллах амплитуды, дифференциации и цикличности соответственно; a, d, r — поправочные коэффициенты амплитуды, дифференциации и цикличности соответственно.

Для обнаружения атакующего воздействия выбираются показатели с наибольшей оценкой $Rate(Index)$. Заметим, что при таком подходе для каждого атакующего воздействия используется отдельная нейронная сеть со своим вектором показателей. Для уменьшения количества нейронных сетей может быть использована методика формирования векторов показателей для классов атак.

Введём следующие обозначения:

— $AttackQuantity$ — количество видов атак, для которых производится обучение нейронных сетей;
— $AttackType_i$ — некоторый вид атаки, $i \in \{1, \ldots, AttackQuantity\}$;
— $Indexes$ — множество всех показателей;
— $IndexQuantity$ — количество показателей во множестве показателей;
— $Index_j \in Indexes$ — некоторый показатель, $j \in \{1, \ldots, IndexQuantity\}$;
— $Rate_{AttackType_i}(Index_j)$ — оценка $Rate(Index_j)$, полученная при сравнении показателя $Index_j$ на статистике нормального поведения пользователей и статистике атаки вида $AttackType_i$.

Методика формирования векторов показателей для классов атак заключается в следующем:

1) Для каждого показателя $Index_j$ и вида атаки $AttackType_i$ вычислить оценку $Rate_{AttackType_i}(Index_j)$.
2) Для каждой атаки $AttackType_i$ сформировать множество $\{Index_{i_1}, \ldots, Index_{i_t}\}$ показателей, удовлетворяющих условию

$$\forall Index \in \{Index_{i_1}, \ldots, Index_{i_t}\}\ \forall Index' \in Indexes \setminus \{Index_{i_1}, \ldots, Index_{i_t}\} \\ (Rate_{AttackType_i}(Index) \geqslant Rate_{AttackType_i}(Index')).$$

3) Создать классы атак $AttackClass_k$ и соответствующие им векторы $\{Index_{i_1}, \ldots, Index_{i_k}\}_k$, помещая в один класс атаки $AttackType_i$, содержащие наибольшее количество v одинаковых показателей во множествах $\{Index_{i_1}, \ldots, Index_{i_t}\}$. Для создания классов атак можно использовать итерационную процедуру (аналогичную алгоритму кластеризации методом k-средних [5]):
   а) принять количество классов $K = 1$. Поместить все виды атак в один класс;
   б) провести оценку качества обучения нейронной сети при использовании вектора показателей $\{Index_{i_1}, \ldots, Index_{i_k}\}_k$ (обычно для оценки качества обучения нейронной сети вычисляют процент правильных срабатываний и ошибок первого и второго рода [6]);
   в) увеличить количество классов K на 1. Поместить все виды атак в K классов таким образом, чтобы два вида атак из одного класса содержали больше совпадений во множествах $\{Index_{i_1}, \ldots, Index_{i_t}\}$, чем два вида атак из разных классов;

г) провести оценку качества обучения нейронной сети при использовании векторов показателей $\{Index_{i_1}, \ldots, Index_{i_k}\}_k$ для каждого класса атак $AttackClass_k$ на отдельной нейронной сети;
д) если качество тестов ухудшилось, то вернуть множество классов атак, полученное на предыдущем шаге (при $K-1$);
е) если качество тестов улучшилось и ошибки лежат в установленных пределах, то вернуть текущее множество классов атак;
ж) перейти к шагу «в».

В результате использования методики получено множество классов атак и соответствующее каждой атаке множество показателей для работы нейронной сети. Каждый класс атак обрабатывается отдельной нейронной сетью.

После формирования классов атак необходимо с помощью топологических тестов нейронной сети убедиться, что для различных атакующих воздействий внутри одного класса атак оптимальными являются схожие параметры архитектуры нейронной сети. В противном случае атакующие воздействия с отличными оптимальными параметрами архитектуры нейронной сети выделяются в отдельный класс атак.

ЛИТЕРАТУРА

1. *Жульков Е. В.* Построение нейронных сетей для обнаружения классов сетевых атак: дис. ... канд. техн. наук. СПб., 2007. 155 с.
2. *Александров И. С.* Разработка системы защиты web-приложений от автоматизированного копирования информации: дис. ... канд. техн. наук. М., 2003. 127 с.
3. *Хафизов А. Ф.* Нейросетевая система обнаружения атак на www-сервер: дис. ... канд. техн. наук. Уфа, 2004. 172 с.
4. *Сорокин С. Н.* Метод обнаружения атак типа «отказ в обслуживании» на web-приложения // Прикладная дискретная математика. 2014. № 1(23). С. 55–64.
5. *Menasce D. A. and Almeida V. A. F.* Capacity Planning for Web Services. Metrics, Models, and Methods. New Jersey: Prentice Hall PTR, 2001. 608 p.
6. *Хайкин С.* Нейронные сети: полный курс. 2-е изд., испр. М.: ООО «И.Д. Вильямс», 2006. 1104 с.

УДК 004.94

РЕАЛИЗАЦИЯ МОНИТОРА БЕЗОПАСНОСТИ СУБД MySQL В DBF/DAM-СИСТЕМАХ

Н. О. Ткаченко

Предлагается прототип механизма, реализующего политику мандатного управления доступом типа multilevel security (MLS) и type enforcement (TE) на основе разработанной ранее формальной ДП-модели, а также механизм сокрытия структуры базы данных на основе метода переписывания запросов. Прототип реализован в виде DBF/DAM-модуля — для MySQL-proxy, функционирующей между клиентом и сервером системы управления базами данных (СУБД) MySQL как прокси-сервер. При реализации модели безопасности предложен и использован подход, при котором функции в коде соответствуют де-юре правилам формальной ДП-модели.

Ключевые слова: *компьютерная безопасность, управление доступом, реализация моделей безопасности, DBF/DAM-системы, СУБД MySQL, MySQL-proxy.*