УДК 004.056.53

А.Т. Рахманов, Р.Х. Хамдамов, К.Ф. Керимов, Ш.К. Камалов

АЛГОРИТМ АВТОМАТИЧЕСКОГО ОБНАРУЖЕНИЯ УЯЗВИМОСТИ ВИДА SQL-ИНЪЕКЦИИ

Ключевые слова: SQL-инъекция, выявление угроз, адаптивный анализ, математическая модель.

Введение

Атаки на веб-приложения являются относительно новым видом угрозы. Если в веб-приложении должным образом не осуществляется фильтрация входящих параметров, то злоумышленники могут получить возможность фальсифицировать базу данных, используя форму на веб-странице либо изменяя другие входящие данные.

Математическое моделирование и идентификация информационных объектов играют важную роль при решении задач распознавания образов. Одной из таких задач является обнаружение атак на веб-приложения или нормальных запросов. Исследования, посвященные изучению обнаружения атак или нормальных запросов, начались сравнительно недавно. Но тем не менее существует много исследований в этом направлении.

Применение математических методов в решении таких задач рассматривалось во многих работах ученых [1–3]. Например, в [3] описаны два способа обнаружения атак SQL-инъекций, основанных на свойстве распределения символов при построении атак SQL-инъекций. В [4] предложено моделирование атак и нормальных запросов и рассмотрена их идентификация с помощью некоторой функции, нижняя граница которой зависит от длины входной строки.

В данной работе предлагается математический способ идентификации атак SQL-инъекций с помощью ограниченной снизу функции, которая зависит от входной строки. Для построения такой функции используются символы и ключевые слова, которые часто встречаются в построении атак злоумышленников.

В предлагаемом методе можно обнаружить атаки инъекций SQL с помощью одного символа. Тем не менее экспериментально показано, что метод обнаружения с использованием набора многочисленных символов позволяет более точно определить уязвимость вида SQL-инъекции.

1. Обнаружение атак вида SQL-инъекции

Атака вида SQL-инъекции выполняется посредством сигнатуры, которая введена через форму на веб-странице либо другим способом, позволяющим менять входящие параметры. Рассмотрим пример SQL-инъекции.

Запросы отправляем на сайт form.php. В данном файле запишем следующий код:

```
$test = $_GET['test'];

$query = "SELECT * FROM userlist WHERE user='$test".
```

Здесь делаем выборку данных по значению переменной $test. После этого посылаем запрос с кавычкой:

```
sqlinj/index2.php?user=AlexanderPHP'.
```

Если при выполнении данного запроса получаем ошибку, то уязвимость вида SQL-инъекции имеет место.

Чтобы определить, является ли вводимая строка атакой, используем комбинацию символов. Для этого необходимо разработать алгоритм, который исходя из символов определяет, является ли входящая строка нормальным запросом либо атакой.

1.1. Подготовка. Для определения SQL-инъекции вводим характеристики атак SQL-инъекций с помощью символов (табл. 1) и ключевых слов (табл. 2). Пусть наблюдается некоторая входная строка L, $x_1, x_2, ..., x_{20}$ — частота появления в L символов (табл. 1), $x_{21}, x_{22}, ..., x_{30}$ — частота появления ключевых слов (табл. 2), x_{31} — частота появления всех остальных знаков и чисел 0, 1, 2, ..., 9 в строке L. С точки зрения определения атак типа SQL-инъекций обычные символы $a, b, ..., z$ и числа 0, 1, …, 9 не играют важную роль. Поэтому в данной работе предполагается, что частота появления всех этих символов и чисел в наблюдаемой строке L равна единице, т.е. $x_{31} = 1$. Таким образом, любую строку L можно определить с помощью характеристик следующим образом: $L = (x_1, x_2, ..., x_{20}, x_{21}, ..., x_{30}, x_{31})$ как элемент некоторого фазового пространства X. Часто в построении атак типа SQL-инъекций используются символы и ключевые слова (табл. 1, 2).

Таблица 1

Переменная	Символ
u_1	Пробел
u_2	Точка-запятая (;)
u_3	Апостроф (’)
u_4	Правая скобка ())
u_5	Левая скобка (()
u_6	Правая фигурная скобка (})
u_7	Левая фигурная скобка ({)
u_8	Правая квадратная скобка (])
u_9	Левая квадратная скобка ([)
u_{10}	Диез (#)
u_{11}	Процент (%)
u_{12}	Кавычка (”)
u_{13}	Амперсанд (&)
u_{14}	Обратная косая (\)
u_{15}	Вертикальная линия (|)
u_{16}	Знак равенства (=)
u_{17}	Больше чем (>)
u_{18}	Меньше чем (<)
u_{19}	Звездочка (*)
u_{20}	Косая черта (/)

Таблица 2

Переменная	Ключевые слова
u_{21}	and
u_{22}	or
u_{23}	union
u_{24}	where
u_{25}	limit
u_{26}	group by
u_{27}	select
u_{28}	\'
u_{29}	hex
u_{30}	substr

1.2. Метод определения. Из определения L видно, что любой элемент L из построенного пространства X лежит на гиперплоскости $\Gamma = \{L = (x_1, x_2, ..., x_{20}, ..., x_{30}, x_{31}) : x_{31} = 1\}$. Поэтому можно предположить, что чем больше частота появления символов и ключевых слов во входной строке, тем очевиднее близость входной строки L к атакам типа SQL-инъекций. Поэтому функция определения атаки должна быть возрастающей по переменным $x_1, x_2, ..., x_{20}, x_{21}, ..., x_{30}$ и убывающей — по переменной x_{31}. Исходя из этого предлагаем следующую возрастающую по $x_1, x_2, ..., x_{20}, x_{21}, ..., x_{30}$ функцию

$$f(L) = f(x_1, x_2, ..., x_{20}, x_{21}, ..., x_{31}) = \frac{\sum_{i=1}^{30} x_i}{\sum_{i=1}^{30} x_i + x_{31}}$$

для определения атак типа SQL-инъекций. Так как в данной работе предполагается, что частота появления всех остальных знаков и чисел 0, 1, 2, ..., 9 в строке L равна единице, из последнего равенства получим

$$f(L) = f(x_1, x_2, ..., x_{20}, x_{21}, ..., x_{31}) = \frac{\sum_{i=1}^{30} x_i}{\sum_{i=1}^{30} x_i + 1}. \quad (1)$$

Данная функция имеет следующие свойства:

1) $0 \le f(L) < 1$ для всех $L \in \Gamma$;

2) для атак вида SQL-инъекций минимальное значение функции снизу ограничено числом 1/2.

Таким образом, если входная строка L — атака типа SQL-инъекции, то она, по крайней мере, должна содержать один символ (табл. 1) или одно ключевое слово (табл. 2). Поэтому $\sum_{i=1}^{30} x_i \ge 1$, и так как функция $f(L)$ является возрастающей по каждой из переменных x_i, ее минимум при $\sum_{i=1}^{30} x_i \ge 1$ достигается в точке L_0, для которой $\sum_{i=1}^{30} x_i = 1$.

Таким образом, если L — произвольная строка и $f(L) \ge 1/2$, то L, возможно, является атакой вида SQL-инъекции, или $f(L) < 1/2$ — входная строка нормальная, если при построении атак SQL-инъекций используются ключевые слова (табл. 2). Поэтому функцию (1) можно использовать для распознавания нормальных строк и атак SQL-инъекций, построенных с помощью символов и ключевых слов.

Таким образом, если L — произвольная строка, содержащая минимум два символа (табл. 1), то $f(L) \ge 2/3$ и L, возможно, является атакой SQL-инъекции. Если $f(L) < 2/3$, то входная строка, возможно, нормальная, если при построении атак SQL-инъекций используются только символы (табл. 1). Поэтому функцию (1) можно применять для распознавания нормальных строк и атак SQL-инъекций, построенных с помощью символов (табл. 1) и ключевых слов (табл. 2).

В обоих случаях, используя функцию (1), имеем критерий качества для определения угроз. Подобная функция построена и использована в работе [1], но там значение функции зависит от длины входной строки L и минимума такой функции для любой строки L не существует. Поэтому в [5] для определения границы разпознающей функции строятся дополнительные усредненные критерии качества, решая соответствующую оптимизационную задачу с помощью дополнительных математических аппаратов.

В данном случае границей разпознающей функции (1) является рациональное число 1/2. Таким образом, если строка L содержит хотя бы один символ или одно ключевое слово, то условие $f(L) \geq 1/2$ достаточно для определения угрозы.

Образцы строк, содержащих SQL-инъекции, и нормальных строк представлены в табл. 3, 4 соответственно.

Таблица 3

Номер	Строки атаки
1	id=1'
2	AlexanderPHP'
3	AlexanderPHP'%20--%20habrahabr
4	1 UNION SELECT 1,2
5	1 UNION SELECT 1,2,3
6	1 UNION SELECT 1,2,3,4,5
7	1 GROUP BY 2
8	1 GROUP BY 8
9	–1 UNION SELECT 1,2,3,4,5
10	–1 UNION SELECT 1,2,3,4,5 FROM users WHERE id=1
11	–1 UNION SELECT name,2,pass,4,5 FROM users WHERE id=1
12	–1' UNION SELECT name,2,pass,4,5 FROM users WHERE id=1 --%20
13	–1' UNION SELECT 1,'<?php eval($_GET[1]) ?>',3,4,5 INTO OUTFILE '1.php' --%20
14	–1' UNION SELECT 1,2,3,4,5 INTO OUTFILE '1.php' --%20
15	–1' UNION SELECT 1,LOAD_FILE('1.php'),3,4,5 --%20
16	4+OR+1
17	4+--
18	4+UNION+SELECT+*+FROM+news+WHERE+id=4
19	admin' --
20	admin' #
21	admin'/*
22	' or 1=1--
23	' or 1=1#
24	' or 1=1/*
...	…
39	') or ('1'='1—

Таблица 4

Номер	Нормальные строки
1	Test
2	password
3	kamil@
4	@kamil
5	{(1%2)+(3/4)}/5}
6	&temptest(URL){ width,height }

2. Вычисление коэффициента важности ключевых слов (символов)

Для определения коэффициента важности символов из табл. 1 проведены экспериментальные вычисления по 39 атакам инъекции SQL. При этом использована формула

$$K_B = \frac{K_U}{K_N},$$

где K_B — коэффициент важности символа U, K_U — количество атак SQL-инъекций L, при построении которых используется специальный символ u, K_N — общее количество атак SQL-инъекций.

Далее, используя введенные обозначения, определим входную строку как вектор с числовыми координатами:

$$L = (K_{U_1} x_1, K_{U_2} x_2, ..., K_{U_{30}} x_{30}, K_{U_{31}} x_{31}). \quad (2)$$

Такое определение входной строки отличается от предыдущего тем, что координаты входной строки (2) имеют различные значения, отличные от единицы. Поэтому входные строки имеют различные координаты. Но если в пространстве X определено понятие длины, то некоторые входные строки могут иметь одинаковую длину, что не помешает различать входные строки. Отличительной особенностью входной строки-атаки SQL-инъекций, определенной согласно (2), является то, что она имеет только неотрицательные координаты, и поэтому если длина строки строго положительна, то эта строка, возможно, близка к атаке SQL-инъекций.

Таким образом, можем построить новую функцию распознавания, используя определение (2) для входной строки. Тогда функция распознавания определяет входную строку более точно, чем (1), так как в этом случае при построении функции распознавания учитывается коэффициент важности всех символов (табл. 5). Но при этом важно отметить, что для определения коэффициента важности каждого из них необходимо провести экспериментальные вычисления над большим количеством атак SQL-инъекций (например, 500–600). И тогда коэффициент важности каждого будет неизменным почти для всех атак SQL-инъекций. Учитывая этот факт, построим новую функцию для определения атак SQL-инъекций:

$$f_K(L) = f(x_1, x_2, ..., x_{20}, x_{21}, ..., x_{31}) = \frac{\sum_{i=1}^{30} K_{u_i} x_i}{\sum_{i=1}^{30} K_{u_i} x_i + 1}. \quad (3)$$

Теперь для определения статуса входной строки (2) вместо функции (1) можно использовать функцию (3). Здесь так же, как и выше, можно определить нижнюю границу функции (3) для выявления атак SQL-инъекций.

Таблица 5

Символы	Коэффициент важности символов
=	0,4872
%	0,2051
‘	0,6923
*	0,0769
/	0,0513
]	0,0256
[	0,0257
{	0
}	0
&	0
\	0
#	0,0513
“	0
!	0
<	0,0256
>	0,0255
(	0,1538
)	0,1795
;	0
пробел	0,7949

Алгоритм распознавания состоит из следующих шагов.

Шаг 1. Используя реальную входную строку, определяем объект L согласно (2).

Шаг 2. Вычисляем значение функции (3).

Шаг 3. Определяем минимальное значение функции (3).

Шаг 4. Сравнивая значение функции (3) с ее минимальным значением, определяем статус входной строки.

Видно, что все функции определения статуса входных строк зависят от моделирования информационного объекта (входных строк). Поэтому очень важен принцип формализации определения входной строки, что во многом определяет и методы решения распознавания информационных объектов.

Заключение

В данной работе предложен алгоритм обнаружения атаки инъекций SQL с помощью функции распознавания и дана оценка эффективности предложенного алгоритма с помощью искусственных данных. В предлагаемом методе создан с помощью примерных данных атакующих и нормальных строк набор символов для распознавания как атаки, так и нормального запроса с ранее известным порогом. Согласно экспериментам с искусственными данными набор содержит пробел, точку с запятой и правую скобку, которая наиболее подходит для распознавания атаки и нормальной строки. Однако важен гибкий выбор набора в зависимости от наблюдаемых данных.

Проблемы предлагаемого метода — сбор информации об атаках и нормальных запросах и ее генерация. В экспериментах использовались искусственно сгенерированные данные, однако необходим реальный набор данных с серверов веб-приложений в эксплуатации.

А.Т. Рахманов, Р.Х. Хамдамов, К.Ф. Керімов, Ш.К. Камалов

АЛГОРИТМ АВТОМАТИЧНОГО ВИЯВЛЕННЯ ВРАЗЛИВОСТІ ВИДУ SQL-ІН’ЄКЦІЇ

Атаки на веб-додатки є відносно новим видом загрози. Якщо в веб-додатку належним чином не провадиться фільтрація вхідних параметрів, то зловмисники можуть отримати можливість фальсифікувати базу даних, використовуючи форму на веб-сторінці або змінюючи інші вхідні дані. Математичне моделювання та ідентифікація інформаційних об’єктів відіграють важливу роль при вирішенні задач розпізнавання образів. Однією з таких задач є виявлення атак на веб-додатки або нормальних запитів. Дослідження, присвячені вивченню виявлення атак або нормальних запитів, почалися порівняно недавно. Проте існує багато досліджень в цьому напрямку. Атака виду SQL-ін’єкції — поширений спосіб злому веб-додатків, які мають базу даних. Запропоновано математичний спосіб ідентифікації атак SQL-ін’єкцій за допомогою обмеженої знизу функції, що залежить від вхідного рядка. Для побудови такої функції використано символи і ключові слова, які часто зустрічаються в побудові атак зловмисників. За допомогою запропонованого методу можна виявляти атаки ін’єкцій SQL, використовуючи один символ. Проте експериментально показано, що даний метод виявлення з використанням набору численних символів дозволяє більш точно визначити вразливість виду SQL-ін’єкції. У запропонованому методі створено набір символів, що поєднується як з атакою, так і з нормальними запитами, з раніше відомим порогом, використовуючи приблизні дані атак та нормальних запитів. Згідно з експериментами з штучними даними набір містить пробіл, крапку з комою і праву дужку, що найбільше підходить для виявлення атаки чи нормального запиту.

Ключові слова: SQL-ін’єкція, виявлення загроз, адаптивний аналіз, математична модель.

A.T. Rakhmanov, R.Kh. Khamdamov, K.F. Kerimov, Sh.K. Kamalov

AUTOMATIC DETECTION ALGORITHM FOR VULNERABILITY OF SQL-INJECTION

Attacks to web applications are a relatively new type of attack. If the web application does not filter incoming parameters properly, then attackers can get the opportunity to falsify the database using the form on the web page or by changing other incoming data. Mathematical modeling and identification of information objects play an important role in solving problems of pattern recognition. One of these tasks is to detect attacks or normal requests for web applications. Studies on the detection of attacks or normal requests for web applications began relatively recently. Nevertheless, there is a lot of research in this direction. Attack of the form of SQL-injection is a common way of hacking web applications that have a database. Our paper proposes a mathematical method for identifying SQL-injection attacks using a function bounded below that depends on the input string. To build such a function, we used special characters and keywords that are often found in the construction of attacks by intruders. In our proposed method, we can detect SQL-injection attacks using a single character. Nevertheless, we experimentally show that the proposed detection method using a set of numerous symbols allows us to determine the vulnerability of the form of SQL-injection more accurately. In the proposed method, we created a character set that combines both attack and normal detections, and the previously known threshold, using the approximate data of the attackers and normal strings. According to our experiments with artificial data, the set contains a space, a semicolon, and the right bracket has worked well for a larger weight range for the attack and the normal string.

Keywords: SQL-injection, threat identification, adaptive analysis, mathematical model.

1. Керимов К.Ф. Модель выявления угроз информационной безопасности в электронных ресурсах. *Перспективы развития техники и технологии и достижения горно-металлургической отрасли за годы независимости Республики Узбекистан*: Тез. докл. Респ. науч. конф. 12-14 мая. Навои, 2011. С. 339–340.
2. Козлов Д.Д., Петухов А.А. Методы обнаружения уязвимостей в web-приложениях. *Программные системы и инструменты*. 2006. № 7. С. 156–166.
3. Рябко Д.М. Подход к тестированию уязвимости web-приложений от атак типа SQL-инъекций. *Проблеми програмування.* 2006. № 2-3. С. 585–591.
4. Керимов К.Ф., Мухсинов Ш.Ш., Исматуллаев С.О. Брандмауэр баз данных, основанный на обнаружении аномалий. *Проблемы информатики и энергетики*. 2016. № 1. С. 89–95.
5. Opanasenko V.N., Kryvyi S.L. Synthesis of adaptive logical networks on the basis of Zhegalkin polynomials. *Cybernetics and Systems Analysis*. 2015. **51**, N 6. P. 969–977. DOI: 10.1007/s10559-015-9790-1.

Получено 01.04.2019
После доработки 24.05.2019