УДК 004.41:004.056 **doi: 10.26906/SUNZ.2019.6.043**

О. В. Коваленко

Центральноукраїнський національний технічний університет, Кропивницький, Україна

МАТЕМАТИЧНА МОДЕЛЬ ТЕХНОЛОГІЇ ТЕСТУВАННЯ ВРАЗЛИВОСТІ ДО SQL ІН'ЄКЦІЙ

Анотація. В роботі представлені результати дослідження та алгоритми тестування на вразливість до однієї з найбільш поширених видів атак на Web-застосунки SQL ін'єкції. На основі аналізу методології тестування уразливості Web-застосунків до DOM XSS і матеріалів Open Web Application Security Project, розроблений алгоритм аналізу уразливості Web-застосунків до SQL ін'єкцій. Відмінною особливістю даного алгоритму є облік тільки уразливості, яка є в GET параметрах URL і використовує тільки сліпий метод ін'єкції SQL коду, що використовує особливість використання булевих операторів в SQL запитах (Boolean blind SQL injection). На підставі поданого алгоритму розроблена GERT-модель технології тестування уразливості до SQL ін'єкцій. В розробленій моделі вузли графа інтерпретуються станами комп'ютерної системи в процесі тестування уразливості до SQL ін'єкцій, а гілки графа – ймовірносно-тимчасовими характеристиками переходів між станами. Таким чином, на основі експоненційної GERT-мережі розроблено математичну модель технології тестування уразливості до SQL ін'єкцій, яка відрізняється від відомих, вдосконаленим способом визначення відстані між результатами ін'єкції. Використання в запропонованому способі критерію Джаро-Вінклера, для порівняння результатів ін'єкції SQL коду і введення порогового значення дозволить підвищити точність результатів тестування безпеки програмного забезпечення. Розглянуто приклад атаки SQL ін'єкцій, суть яких – впровадження в дані (передані через GET, POST запити або значення Cookie) довільного SQL коду. Побудовано графіки, зовнішній вигляд кривих яких дає підстави припустити, що не всі знайдені рішення застосовні при математичному та імітаційному моделюванні в якості вхідних даних. У той же час зовнішній вигляд графіків, отриманих для інших значень дає підстави припустити, що випадкова величина часу виконання технології тестування уразливості до SQL ін'єкцій відповідає гамма-розподілу (близьке до експоненціального). Перевірка цієї гіпотези проведена за критерієм χ^2 Пірсона.

Ключові слова: вразливості до SQL ін'єкцій, GERT-мережі, вразливості безпеки, алгоритми тестування, математична модель тестування.

Вступ

В даний час великий попит на Web-застосунки і Web-послуги обумовлює великий інтерес зловмисників до їх можливих вразливостей. При цьому основні загрози в напрямку серверних компонентів трансформується в атаки, спрямовані проти звичайних користувачів. Проведений аналіз матеріалів Open Web Application Security Project (OWASP TOP-10) [1] показав, що одним з найбільш небезпечних видів атак (вразливостей) є міжсайтовий скриптинг – XSS (Cross Site Scripting).

Аналіз літератури [1-8] показав, що міжсайтовий скриптинг – це помилка валідації призначених для користувача даних, яка дозволяє передати JavaScript код на виконання в браузер користувача. Атаки такого роду часто також називають HTML-ін'єкціями, адже механізм їх впровадження дуже схожий з SQL-ін'єкціями, але на відміну від останніх, впроваджуваний код виповнюється в браузері користувача. У ряді Інтернет-статей докладно описані основні механізми виникнення подібного роду загроз, а також шляхи можливого блокування. Однак, щоб ідентифікувати ці загрози і можливі наслідки їх поширення в процесі безпечного управління IT-проектами, а також запропонувати оптимальні шляхи вирішення цієї проблеми, існує необхідність математичної формалізації процесу їх ініціалізації і поширення.

Особливо актуальним завданням в цьому напрямку є моделювання DOM (Document Object Model) XSS уразливості. Для математичної формалізації алгоритму аналізу DOM XSS уразливості скористаємося основними положеннями мережевого GERT-моделювання, докладно описаними в роботах [2-5]. Проведені дослідження [3, 5] показали, що на основі аналізу методології тестування уразливості Web-застосунків до DOM XSS і матеріалів Open Web Application Security Project [6, 7], можна розробити алгоритм аналізу уразливості Web-застосунків до SQL ін'єкцій. Відмінною особливістю даного алгоритму є облік тільки уразливості, яка є в GET параметрах HTTP запиту тобто передається серверу за допомогою посилання Uniform Resource Locator і використовує тільки сліпий метод ін'єкції SQL коду, що використовує особливість використання булевих операторів в SQL запитах (Boolean blind SQL injection) [7].

Мета роботи: розробити математичну модель технології тестування вразливості до SQL ін'єкцій, яка відрізняється від відомих, удосконаленим способом визначення відстані між результатами ін'єкції.

Алгоритм аналізу уразливості до SQL ін'єкцій

Відповідно до алгоритму аналізу уразливості Web-застосунки до SQL ін'єкцій, його етапи можна описати таким чином:

1. З введеного URL посилання виходить список GET параметрів.

2. Виконується перевірка стабільності Web-сторінки. Для цього виконується два послідовних запити в Web-сторінки і обчислюється відстань між вмістом HTML коду сторінки за допомогою критерію Джаро-Вінклера [8]. Якщо значення критерію менше певного порогового значення, виконувати подальший аналіз неможливо.

3. У параметр GET запиту виконується ін'єкція SQL коду, який не змінює результат запиту до бази даних і зберігається результуючий HTML код.

4. У параметрі GET запиту виконується ін'єкція SQL коду, який змінює результат запиту до бази даних, призводить або до отримання повного набору даних з таблиці, або до відсутності результату, після чого зберігається результуючий HTML код.

5. За допомогою критерію Джаро-Вінклера виконується порівняння результатів ін'єкції SQL коду. Якщо значення критерію менше певного порогового значення, то в даному GET параметрі є можлива вразливість до SQL ін'єкції.

6. Кроки 2-5 повторюються для всіх параметрів GET запиту наданого URL.

На підставі поданого алгоритму розробимо GERT-модель технології тестування уразливості до SQL ін'єкцій.

GERT-модель технології тестування уразливості до SQL ін'єкцій

Побудуємо, відповідно до представленого описом мережеву GERT-модель технології тестування уразливості до SQL ін'єкцій. Графічне зображення GERT-моделі представлено на рис. 1.

У представленій мережі вузли графа інтерпретуються станами комп'ютерної системи в процесі тестування уразливості до SQL ін'єкцій, а гілки графа – ймовірносно-тимчасовими характеристиками переходів між станами.

Зокрема гілка (1,2) характеризує час отримання і аналізу GET-параметрів з введеного URL посилання. Гілка (2,3) відображає час відправлення первинних і вторинних запитів в Web-сторінки. Гілка (3,4) задає випадкове час порівняння сторінок (час обчислення відстані між вмістом HTML коду сторінки за допомогою критерію Джаро-Вінклера). Гілка (4,5) характеризує час, за яке виконується ін'єкція SQL коду, який не змінює результат запиту до бази даних, а також який змінює результат запиту до бази даних відповідно. Далі гілка (5,6) характеризує час порівняння результатів ін'єкції SQL коду. Гілка (4,2) характеризує тимчасові характеристики повернення системи в початковий стан, коли значення критерію Джаро-Вінклера менше певного порогового значення, в той же час гілка (6,2) відображає тимчасові характеристики переходу до нової перевірки в разі якщо значення критерію Джаро-Вінклера більше певного порогового значення. Характеристики гілок моделі наведені в табл. 1.

Таблиця 1. **Характеристики гілок моделі технології тестування вразливості до SQL ін'єкцій**

№ п/п	Гілка	*W*-функція	Ймовірність	Функція що походить від моментів
1	(1,2)	W_{12}	p_1	$\lambda_1/(\lambda_1-s)$
2	(2,3)	W_{23}	p_2	$\lambda_2/(\lambda_2-s)$
3	(3,4)	W_{34}	p_3	$\lambda_3/(\lambda_3-s)$
4	(4,5)	W_{45}	p_4	$\lambda_4/(\lambda_4-s)$
5	(5,6)	W_{56}	p_5	$\lambda_3/(\lambda_3-s)$
6	(4,2)	W_{42}	$1-p_4$	$\lambda_5/(\lambda_5-s)$
7	(6,2)	W_{62}	p_6	$\lambda_6/(\lambda_6-s)$

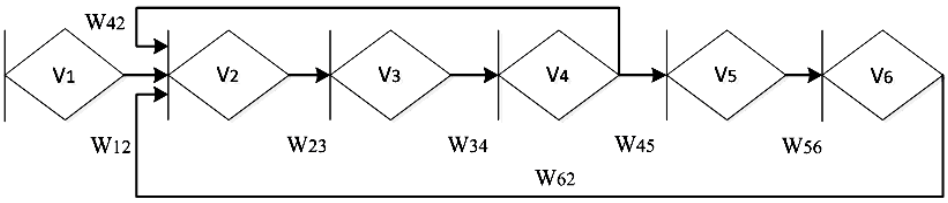

Рис. 1. GERT-модель технології тестування вразливості до SQL ін'єкцій

Еквівалентна *W*-функція часу виконання технології тестування вразливості до SQL ін'єкцій дорівнює:

$$W_E(s)=W_{12}W_{23}W_{34}W_{45}W_{56}/\left(1-W_{12}W_{23}W_{34}W_{42}-W_{12}W_{23}W_{34}W_{45}W_{56}W_{62}\right)=$$

$$=\frac{p_1p_2p_3p_4p_5\lambda_1\lambda_2\lambda_3^2\lambda_4\left(\lambda_3-s\right)\left(\lambda_5-s\right)\left(\lambda_6-s\right)}{\left[\begin{array}{c}\left(\lambda_1-s\right)\left(\lambda_2-s\right)\left(\lambda_3-s\right)^2\left(\lambda_4-s\right)\left(\lambda_5-s\right)\left(\lambda_6-s\right)-\\-\left(p_1p_2p_3\lambda_1\lambda_2\lambda_3\times\left(q_1\lambda_5\left(\lambda_3\lambda_4-\lambda_4s-\lambda_3s-s^2\right)\left(\lambda_6-s\right)-p_4p_5p_6\lambda_3\lambda_4\lambda_6\left(\lambda_5-s\right)\right)\right)\end{array}\right]}, \quad (1)$$

де $1-p_4=q_1$.

Аналогічно алгоритму аналізу DOM XSS вразливості [2, 5] виконуючи комплексне перетворення $z=-s$, отримаємо

$$\Phi(z)=\frac{vz^2+bz+k}{\left(z^7+rz^6+cz^5+dz^4+gz^3+hz^2+wz+m\right)}, \quad (2)$$

де $v=-p_1p_2p_3p_4p_5\lambda_1\lambda_2\lambda_3^2\lambda_4$,

$$b=p_1p_2p_3p_4p_5\lambda_1\lambda_2\lambda_3^2\lambda_4\left(\lambda_5+\lambda_6\right),$$

$$k=-p_1p_2p_3p_4p_5\lambda_1\lambda_2\lambda_3^2\lambda_4\lambda_5\lambda_6,$$

$$r=\lambda_1+\lambda_2+\lambda_4+\lambda_5-2\lambda_3-\lambda_6,$$

$$c=\left(\begin{array}{c}\lambda_1\lambda_4+\lambda_2\lambda_4+\lambda_1\lambda_5+\lambda_2\lambda_5+\lambda_3^2+\\+2\lambda_3\lambda_6-\lambda_4\lambda_6-\lambda_5\lambda_6-\lambda_1\lambda_6-\lambda_4\lambda_5-\\-2\lambda_3\lambda_4-2\lambda_3\lambda_5-\lambda_1\lambda_2-2\lambda_1\lambda_3-2\lambda_2\lambda_3\end{array}\right),$$

$$d = -\lambda_1\lambda_2\lambda_3\lambda_4\lambda_5\lambda_6 \times$$
$$\times\left(\begin{array}{l}\frac{1}{\lambda_2\lambda_3\lambda_5}+\frac{1}{\lambda_1\lambda_3\lambda_5}+\frac{1}{\lambda_2\lambda_3\lambda_4}+\frac{1}{\lambda_1\lambda_3\lambda_4}+\\ +\frac{\lambda_3}{\lambda_1\lambda_2\lambda_4\lambda_5}+\frac{1}{\lambda_2\lambda_3\lambda_6}+\frac{1}{\lambda_1\lambda_3\lambda_6}+\frac{1}{\lambda_3\lambda_5\lambda_6}+\\ +\frac{2}{\lambda_2\lambda_5\lambda_6}+\frac{2}{\lambda_1\lambda_5\lambda_6}+\frac{1}{\lambda_3\lambda_4\lambda_6}+\frac{2}{\lambda_2\lambda_4\lambda_6}+\\ +\frac{2}{\lambda_1\lambda_4\lambda_6}-\frac{1}{\lambda_1\lambda_2\lambda_3}-\frac{2}{\lambda_1\lambda_2\lambda_5}-\frac{2}{\lambda_1\lambda_2\lambda_4}-\\ -\frac{1}{\lambda_3\lambda_4\lambda_5}-\frac{2}{\lambda_2\lambda_4\lambda_5}-\frac{2}{\lambda_1\lambda_2\lambda_6}-\frac{\lambda_3}{\lambda_1\lambda_2\lambda_5\lambda_6}-\\ -\frac{\lambda_3}{\lambda_1\lambda_2\lambda_4\lambda_6}-\frac{2}{\lambda_4\lambda_5\lambda_6}-\frac{\lambda_3}{\lambda_2\lambda_4\lambda_5\lambda_6}-\frac{\lambda_3}{\lambda_1\lambda_4\lambda_5\lambda_6}\end{array}\right),$$

$$g = \left[\begin{array}{l}\left(\begin{array}{l}\lambda_1\lambda_2\lambda_3\lambda_4\lambda_5\lambda_6 \times\\ \times\left(\begin{array}{l}\frac{2}{\lambda_1\lambda_2}+\frac{\lambda_3}{\lambda_1\lambda_2\lambda_5}+\frac{\lambda_3}{\lambda_1\lambda_2\lambda_4}+\frac{2}{\lambda_4\lambda_5}+\\ +\frac{\lambda_3}{\lambda_2\lambda_4\lambda_5}+\frac{\lambda_3}{\lambda_1\lambda_4\lambda_5}+\frac{\lambda_3}{\lambda_1\lambda_2\lambda_6}-\\ -\frac{1}{\lambda_2\lambda_3}-\frac{1}{\lambda_1\lambda_3}-\frac{1}{\lambda_3\lambda_5}-\frac{2}{\lambda_2\lambda_5}-\\ -\frac{2}{\lambda_1\lambda_5}-\frac{1}{\lambda_3\lambda_4}-\frac{2}{\lambda_2\lambda_4}-\frac{2}{\lambda_1\lambda_4}-\\ -\frac{1}{\lambda_3\lambda_6}-\frac{2}{\lambda_2\lambda_6}-\frac{2}{\lambda_1\lambda_6}-\frac{2}{\lambda_5\lambda_6}-\\ -\frac{\lambda_3}{\lambda_2\lambda_5\lambda_6}-\frac{\lambda_3}{\lambda_1\lambda_5\lambda_6}-\frac{2}{\lambda_4\lambda_6}-\\ -\frac{\lambda_3}{\lambda_2\lambda_4\lambda_6}-\frac{\lambda_3}{\lambda_1\lambda_4\lambda_6}-\frac{\lambda_3}{\lambda_4\lambda_5\lambda_6}\end{array}\right)\end{array}\right)+\\ +p_1p_2p_3q_1\lambda_1\lambda_2\lambda_3\lambda_5\end{array}\right];,$$

$$h = -\left[\begin{array}{l}\left(\begin{array}{l}\lambda_1\lambda_2\lambda_3\lambda_4\lambda_5\lambda_6 \times\\ \times\left(\begin{array}{l}\frac{\lambda_3}{\lambda_1\lambda_2}+\frac{\lambda_3}{\lambda_4\lambda_5}-\frac{1}{\lambda_3}-\frac{2}{\lambda_2}-\frac{2}{\lambda_1}-\frac{2}{\lambda_5}-\\ -\frac{\lambda_3}{\lambda_2\lambda_5}-\frac{\lambda_3}{\lambda_1\lambda_5}-\frac{2}{\lambda_4}-\frac{\lambda_3}{\lambda_2\lambda_4}-\frac{\lambda_3}{\lambda_1\lambda_4}-\\ -\frac{2}{\lambda_6}-\frac{\lambda_3}{\lambda_2\lambda_6}-\frac{\lambda_3}{\lambda_1\lambda_6}-\frac{\lambda_3}{\lambda_5\lambda_6}-\frac{\lambda_3}{\lambda_4\lambda_6}\end{array}\right)\end{array}\right)+\\ +\left(\begin{array}{l}p_1p_2p_3q_1\lambda_1\lambda_2\lambda_3\lambda_5\lambda_6 \times\\ \times\left(\frac{1}{\lambda_5\lambda_6}+\frac{\lambda_3}{\lambda_4\lambda_5\lambda_6}-\frac{1}{\lambda_4\lambda_5}\right)\end{array}\right)\end{array}\right];$$

$$w = \left[\begin{array}{l}\left(\begin{array}{l}\lambda_1\lambda_2\lambda_3\lambda_4\lambda_5\lambda_6 \times\\ \times\left(\begin{array}{l}2+\frac{\lambda_3}{\lambda_2}+\frac{\lambda_3}{\lambda_1}+\\ +\frac{\lambda_3}{\lambda_5}+\frac{\lambda_3}{\lambda_4}+\frac{\lambda_3}{\lambda_6}\end{array}\right)\end{array}\right)-\left(\begin{array}{l}p_1p_2p_3q_1\lambda_1\lambda_2\lambda_3\lambda_5\times\\ \times\lambda_6\cdot\left(-1-\frac{\lambda_3}{\lambda_4}-\frac{\lambda_3}{\lambda_6}\right)\end{array}\right)+\\ +p_4p_5p_6\lambda_3\lambda_4\lambda_6\end{array}\right];$$

$$m = -\lambda_1\lambda_2\lambda_3^2\lambda_5\lambda_6\left(\lambda_4 + p_1p_2p_3q_1 - \frac{p_4p_5p_6}{\lambda_1\lambda_2}\right).$$

Аналогічно алгоритму аналізу DOM XSS вразливості [2, 5] щільність розподілення ймовірностей часу виконання технології тестування вразливості до SQL ін'єкцій дорівнює:

$$\phi(x) = 1/(2\pi i)\times$$
$$\times\int_{-i\infty}^{i\infty} e^{zx}\frac{vz^2+bz+k}{\left(z^7+rz^6+cz^5+dz^4+gz^3+hz^2+wz+m\right)}dz, \quad (3)$$

де операція інтегрування виконується за допомогою інтегралу Бромвича-Вагнера [7]. Тоді

$$e^{zx}\Phi(z) = \frac{e^{zx}\left(vz^2+bz+k\right)}{z^7+rz^6+cz^5+dz^4+gz^3+hz^2+wz+m} = \frac{\mu(z)}{\psi(z)}. \quad (4)$$

Тоді щільність розподілення часу виконання алгоритму тестування вразливості до SQL ін'єкцій

$$\phi(x) = \sum_{k=1}^{7}\mathrm{Re}\,s\left[e^{zx}\Phi(z)\right] = \sum_{k=1}^{7}\frac{\mu(z_k)}{\psi(z_k)} = \sum_{k=1}^{7}\frac{e^{zx}\left(vz^2+bz+k\right)}{7z_k^6+6rz_k^5+5cz_k^4+4dz_k^3+3gz_k^2+2hz_k+w}. \quad (5)$$

Багаточлен породжує сім полюсів

$$rz^6+cz^5+dz^4+gz^3+hz^2+wz+m\,.$$

Рішення рівняння

$$rz^6+cz^5+dz^4+gz^3+hz^2+wz+m=0\,. \quad (6)$$

Може бути знайдено будь-яким методом, наприклад, по формулам Вієта [7]. В результаті обчислюються особливі точки $z_1, z_2, z_3, z_4, z_5, z_6, z_7$.

Таким чином, на основі експоненційної GERT-мережі розроблено математичну модель технології тестування уразливості до SQL ін'єкцій, яка відрізняється від відомих, вдосконаленим способом визначення відстані між результатами ін'єкції. Використання в запропонованому способі критерію Джаро-Вінклера, для порівняння результатів ін'єкції SQL коду і введення порогового значення дозволить підвищити точність результатів тестування уразливості до SQL ін'єкцій. Дослідження GERT-моделі технології тестування уразливості до SQL ін'єкцій.

Розглянемо приклад атаки SQL ін'єкцій. Суть таких ін'єкцій – впровадження в дані (передані через GET, POST запити або значення Cookie) довільного SQL коду. Знайдемо щільності розподілення $\phi(x)$ ймовірностей часу виконання алгоритму при умові, що z обирається як корені рівняння

$$(z^7+rz^6+cz^5+dz^4+gz^3+hz^2+wz+m) = 0,$$

умовні ймовірності та інтенсивності у гілках *GERT*-мережах мають значення:

$$p_1=p_2=p_3=p_4=p_5=0{,}999999,$$
$$p_6=0{,}9,\ \lambda_1=\lambda_2=\lambda_3=0{,}9999,\ \lambda_4=0{,}8,\ \lambda_5=0{,}1,\ \lambda_6=0{,}999999.$$

Враховуючи приведені ознаки *GERT*-мережі, у відповідності з (2), а також використовуючи спеціалізований математичний пакет Mathcad, отримаємо, що в знаменнику виразу (3) сформований поліном

$$x^7 - 0.1x^6 - 4.174x^5 + 2.471x^4 - 4.509x^3 + 4.128x^2 + 2.014x - 0.169 = 0. \quad (7)$$

Корені цього полінома (та відповідно функція $\Phi(z)$) дорівнюють:

$x1 \approx -2.11254039866286$, $(P(x1) \approx 0$; iter = 1);

$x2 \approx -0.561885634027132$, $(P(x2) \approx 0$; iter = 4);

$x3 \approx -0.208185977139001 - i\ 0.60944124336833$, $(P(x3) \approx 0$; iter = 5);

$x4 \approx -0.208185883644938 + i \cdot 0.609441306673327$, $(P(x4) \approx 0$; iter = 4);

$x5 \approx -0.103581224605665$, $(P(x5) \approx 0$; iter = 3);

$x6 \approx 1.64718955898524 - i \cdot 0.775107663208$, $(P(x6) \approx 0$; iter = 1);

$x7 \approx 1.64718955909435 + i\ \ 0.775107667929698$, $(P(x7) \approx 0$; iter = 4).

Досліджуємо залежність функції $\Phi(z)$ від інтенсивності z. На рис. 2 представлена крива графіку залежності функції $\Phi(z)$ від z у розглянутих вище умовах. Як видно з рисунку випадкова величина z розподілена у відповідності до показового закону. Аналогічно алгоритму розрахунку $\phi(x)$, використовуваному в GERT-моделі технології тестування DOM XSS уразливості [3, 5] знайдемо цю функцію і дослідимо її з використанням математичного пакета Mathcad.

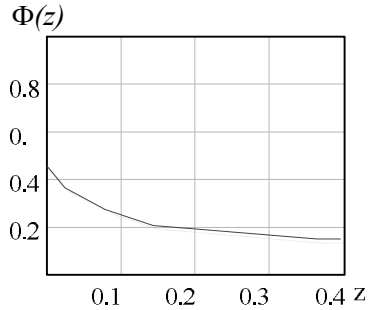

Рис. 2. Графік залежності функції $\Phi(z)$ від інтенсивності z

Відповідно до формули (5) $\phi(x)$ дорівнює:

$$\phi(x) = \sum_{k=1}^{6} \operatorname{Res}_{z=z_k} \left[e^{zx} \Phi(z) \right] = \frac{e^{(a+\delta i)x} \left(v(a+\delta i)^2 + b(a+\delta i) + k \right)}{\left(7u(a+\delta i)^6 + 6r(a+\delta i)^5 + 5c(a+\delta i)^4 + 4d(a+\delta i)^3 + 3g(a+\delta i)^2 + 2h(a+\delta i) + w \right)} - \frac{e^{(a-bi)x} \left(v(a-\delta i)^2 + b(a-\delta i) + k \right)}{\left(7u(a-\delta i)^6 + 6r(a-\delta i)^5 + 5c(a-\delta i)^4 + 4d(a-\delta i)^3 + 3g(a-\delta i)^2 + 2h(a-\delta i) + w \right)}. \quad (8)$$

Аналогічно підходу, використовуваному для GERT-моделі технології тестування DOM XSS уразливості використовуючи вирази Ейлера [2], отримаємо:

$$\phi(x) = \sum_{k=1}^{6} \operatorname{Res}_{z=z_k} \left(e^{zx} \Phi(z) \right) = e^{(a+\delta i)x} \frac{\tau + i\beta}{\gamma + i\theta} + e^{(a-\delta i)x} \frac{\tau - i\beta}{\gamma - i\theta} = = \frac{2e^{ax}}{\gamma^2 + \theta^2} \left((\tau\gamma + \beta\theta)\cos(\delta x) + (\tau\gamma - \beta\theta)\sin(\delta x) \right), \quad (9)$$

де $\tau = a^2 v - \delta^2 v + ab + k$, $\beta = 2a\delta v - \delta b$,

$$\gamma = 7ua^6 - 10ua^4\delta^2 + 105ua^2\delta^4 - 7u\delta^6 + 6ra^5 - 60ra^3\delta^2 + 30ra\delta^4 + 5ca^4 - 30ca^2\delta^2 + 5c\delta^4 + 4da^3 - 12da\delta^2 + 3ga^2 - 3g\delta^2 + 2ha + w,$$

$$\theta = 49ua^5\delta - 140ua^3\delta^3 + 49ua\delta^5 + 30ra^4\delta - 60ra^2\delta^3 + 6r\delta^5 + 20ca^3\delta - 20ca\delta^3 + 12da^3\delta - 4d\delta^3 + 6ga\delta + 2h\delta.$$

На рис. 3 представлені криві щільності розподілу ймовірностей часу виконання технології тестування уразливості до SQL ін'єкцій для наведених вище умов (в якості вхідних даних використовувалися корені полінома (7)).

При цьому рис. 3 *а* відповідає випадку коли в якості вхідних даних використовувалося значення $x1$, б – $x2$, в – $x3$, г – $x4$, д – $x5$, е – $x6$, ж – $x7$. Як і у розглянутому вище прикладі, зовнішній вигляд кривих графіків рис. 3 дає підстави припустити, що не всі знайдені вище рішення (коріння полінома (7)) застосовні при математичному та імітаційному моделюванні в якості вхідних даних.

Так значення *x3, x4, x6 і x7* неможливо надалі використовувати при аналізі і моделюванні. У той же час зовнішній вигляд графіків, отриманих для значень *x1, x2 і x5* дає підстави припустити, що випадкова величина часу виконання технології тестування уразливості до SQL ін'єкцій відповідає гамма-розподілу (близьке до експоненціального).

Результати перевірки цієї гіпотези за критерієм χ^2 Пірсона [8] підтвердили її правдоподібність. Так при досить великому значенні довірчої ймовірності Q=0,95 для всіх розглянутих *x1, x2 і x5* відповідні значення χ^2:

$$\chi_1^2 = 19,3,\ \chi_2^2 = 15,1,\ \chi_5^2 = 25,6) << \overline{\chi^2} = 101,9.$$

Висновки

У роботі розроблена математична модель технології тестування Web-застосунків. В основу математичного моделювання покладено підхід GERT-мережевого синтезу. В результаті розроблено математичну модель технології тестування уразливості до SQL ін'єкцій, яка відрізняється від відомих, вдосконаленим способом визначення відстані між результатами ін'єкції.

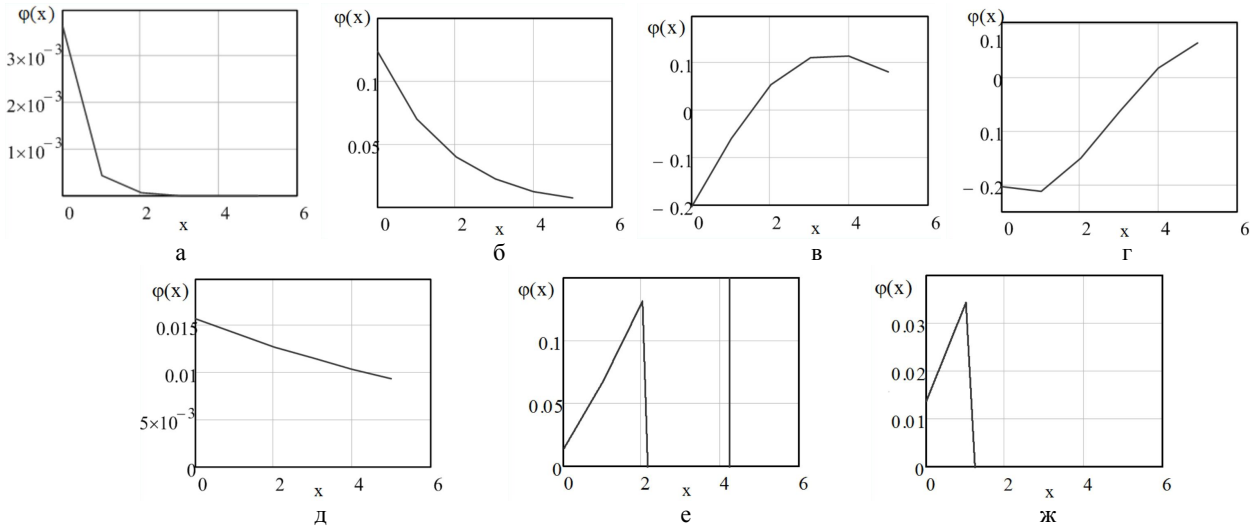

Рис. 3. Графіки щільності розподілу $\phi(x)$
ймовірності часу виконання технології тестування вразливості до SQL ін'єкцій

Використання в запропонованій математичній моделі критерію Джаро-Вінклера, для порівняння результатів ін'єкції SQL коду і введення порогового значення дозволило підвищити точність результатів тестування безпеки програмного забезпечення.

В ході дослідження представленої математичної моделі було визначено, що випадкова величина часу виконання даного процесу тестування в цілому відповідає гамма-розподілу. Перевірка цієї гіпотези була проведена за критерієм χ^2 Пірсона.

СПИСОК ЛІТЕРАТУРИ

1. OWASP Top 10 – 2017 RC1: [Електронний ресурс]. – Режим доступу: https://github.com/OWASP.
2. Kovalenko O., Smirnov O., Kovalenko A., Smirnov S., Vialkova, V. The mathematical model of the testing technology for DOM XSS vulnerabilities. Scientific & practical cyber security journal (SPCSJ). Tbilisi: SCSA, 2018Vol. 2, Is. 1, pp. 22-28.
3. Коваленко А.В. Методы качественного анализа и количественной оценки рисков разработки программного обеспечения / А.А. Смирнов, А.В. Коваленко // Системи обробки інформації. – Вип.. 5(142). – Х.: ХУПС, 2016. – С. 153-157.
4. Технология тестирования DOM XSS уязвимости / А.В. Коваленко, А.С. Коваленко, А.А. Смирнов, С.А. SPCSJ. № 1. [Электронный Журнал]. Грузия. Тбилиси: SCSA – 2017. URL: http://journal.scsa.ge/ru/issues/2017/09/484
5. Коваленко А.В. Проблемы анализа и оценки рисков информационной деятельности / А.А. Смирнов, А.В. Коваленко, Н.Н. Якименко, А.П. Доренский // Системи обробки інформації. – Випуск 3(140). – Х.: ХУПС – 2016. – С. 40-42.
6. About The Open Web Application Security Project – OWASP: [Електронний ресурс]. – Режим доступу: https://www.owasp.org/index.php/About_The_Open_Web_Application_Security_Project.
7. Testing for SQL Injection (OTG-INPVAL-005) – OWASP: [Електронний ресурс]. – Режим доступу: https://www.owasp.org/ index.php/103 Testing_for_SQL_Injection_(OTG-INPVAL-005).

Received (Надійшла) 30.09.2019
Accepted for publication (Прийнята до друку) 12.11.2019

Mathematical model for testing sql injection vulnerability technology

O. Kovalenko

Abstract. The paper presents the results of the study and testing algorithms for vulnerability to SQL injection, one of the most common types of attacks against Web-based applications. Based on the analysis of the methodology for testing the vulnerability of Web applications to the DOM XSS and the materials of the Open Web Application Security Project, an algorithm of analysis of the vulnerability of Web applications to SQL injections has been developed . A distinctive feature of this algorithm is consideration of only the vulnerability that is present in the GET URL parameters and uses only the blind SQL injection of code that uses the feature of using Boolean operators in SQL queries (Boolean blind SQL injection). Based on the presented algorithm, a GERT-model of a technology for testing vulnerabilities to SQL injections was developed, in which the nodes of the graph are interpreted by the states of the computer system in the process of testing the vulnerability to SQL injections, and the branches of the graph are interpreted as probabilistic-temporal characteristics of transitions between states. Thus, on the basis of the exponential GERT network, a mathematical model was developed for testing SQL injection vulnerability technology, which differs from the known technologies by improved method for determining the distance between injection results. The use of the Jaro – Winkler criterion in the proposed method for comparing the results of an injection of SQL code and introducing a threshold value will improve the accuracy of the results of software security testing. An example of an SQL injection attack, the essence of which is the introduction of arbitrary SQL code into data (transmitted via GET, POST or Cookie values), is considered. Graphs were constructed, the curves of which suggest that not all of the found solutions are applicable in mathematical and simulation modeling as input data. At the same time, the appearance of the graphs obtained for other values suggests that the random value of the execution time of the technology for testing the vulnerability to SQL injection corresponds to the gamma distribution (close to exponential). Testing of this hypothesis is made by the χ^2 Pearson criterion.

Keywords: SQL Injection Vulnerability, GERT-networks, security vulnerabilities, testing algorithms, mathematical testing model.