УДК 004.4+ 004.65+004.056.5

Погромська Г.С.
Миколаївський національний університет імені В.О. Сухомлинського

ЗАСТОСУВАННЯ ВАЛІДАТОРІВ ДЛЯ РЕАЛІЗАЦІЇ МОДЕЛІ ДОСЛІДЖЕННЯ SQL-ІН'ЄКЦІЙ ДО БАЗ ДАНИХ

У статті дано опис розробленого програмного забезпечення для реалізації методів SQL-ін'єкцій з метою використання в навчальних цілях. Моделюються випадки реалізації SQL-вставок, виділяються їх особливості для розуміння механізмів роботи. Наведено особливості реалізації моделі дослідження SQL-ін'єкцій за допомогою валідаторів. Розглянуто роботу моделі для дослідження ін'єкцій SQL. У моделі реалізована можливість перегляду результату ін'єкції у разі захищеної та незахищеної від SQL ін'єкцій бази даних.

Ключові слова: *база даних, SQL ін'єкція (SQL Injection), програмне забезпечення, захист даних, атака.*

Постановка проблеми. SQL-ін'єкції є одним з найбільш поширених загроз інформаційній безпеці завдяки своїй простоті. Атака типу SQL-Injection – це атака, за якої проводиться вставка шкідливого коду в рядки, що передаються потім в екземпляр системи управління базою даних (СУБД) для синтаксичного аналізу і виконання. Будь-яка процедура, яка створює інструкції SQL, повинна розглядатися на предмет уразливості до вставки небезпечного коду, оскільки СУБД виконує всі одержувані синтаксично правильні запити. Навіть параметризовані дані можуть стати предметом маніпуляцій досвідченого зловмисника. Вивчення основних відомостей про атаки типу SQL-Injection та загальних принципів захисту від SQL Injection є невід'ємним складником дисциплін у ЗВО, пов'язаних з реалізацією та експлуатацією баз даних. Тому проведення аналізу кода, котрий містить вразливість, демонстрація основних прийомів захисту від атак SQL-Injection у межах відповідних навчальних курсів ЗВО потребує практичної спрямованості та засобів унаочнення.

Аналіз останніх досліджень і публікацій. Базову інформацію по атакам типу SQL-Injection зустрічаємо в роботі [7], в якій пропонуються механізми та способи попередження атак типу SQL-Injection. Серед інших питань розглядається ін'єкція підзапитів в запиті Insert. У роботах автора Т. Макарова [1] описано основні техніки SQL-Injection та специфічні особливості MySQL. Авторами О. Маор та Е. Шулман у [4] розглядаються методи здійснення SQL-Injection в умовах, коли немає будь-якої інформації про додаток, тип бази даних, структури таблиці тощо, відсутні повідомлення про помилки, а у роботі [5] розглядаються методи обходу сигнатурного захисту, націлених на попередження атак SQL-Injection. Робота [6] присвячена не тільки методам здійснення атаки, але приділяє увагу процесу дослідження уразливого додатка. Інформативною з точки зору атаки на систему є стаття автора Ц. Церрудо [3], в якій розглядаються атаки на системи, що застосовують в якості бекенд-сервера MS SQL-Server, зокрема, як за допомогою SQL-Injection проникнути за міжмережевий екран у внутрішню мережу (отримання доступу через SQL-Injection до інших джерел даних, витягування бази користувачів MS SQL, засоби з захисту SQL-Server тощо). У статті [2] розглянуто важливий метод здійснення атаки SQL-Injection у випадках, коли немає виведення повідомлень про помилки, неможливо використати Union Select, відсутня інформація про структуру бази, зокрема, функції MySQL, корисні при здійсненні SQL-Injection, посимвольний перебір значень будь-яких даних з бази даних, використання підзапитів тощо. Отже, SQL Injection – це серйозна помилка в програмуванні, широко розповсюджена та вкрай небезпечна. Правильно побудований процес Software Development Life Cycle значною мірою знижує ймовірність появи вразливості у коді. Тому захист додатків, також як і інформаційна безпека в цілому, повинні бути комплексним, а їх вивченню слід приділяти належну увагу в межах відповідних навчальних курсів ЗВО.

Постановка завдання. Метою статті є описати розроблений програмний засіб для реалізації різних методів SQL-ін'єкцій з метою використання в навчальних цілях для моделювання випадків реалізації SQL-вставок, виділення їх особливостей, розуміння механізмів роботи.

Виклад основного матеріалу дослідження. Програмне забезпечення (ПЗ) «Моделювання атак типу SQl-ін'єкцій до баз даних» – модель, що дозволяє наочно оцінити методи і способи захисту баз даних від найбільш поширеної загрози їх інформаційній безпеці – атак типу SQL Injection.

Програмне забезпечення наочно показує можливості злому бази даних за методом SQl-ін'єкцій (SQL-Injection), що реалізоване за допомогою валідаторів. У програмному забезпеченні використовується база даних (БД), що складається з декількох таблиць, одна з яких є загальнодоступною, одна – приватною, і третя, до якої користувач може отримати доступ (рис. 1).

Як засіб розробки було обране середовище розробки Embarcadero RAD Studio 10 Seattle та СУБД MySQL.

Реалізовані функції:

– можливість перегляду результату ін'єкції за незахищеної бази даних;

– реакція на запит захищеної і незахищеної від SQL-ін'єкцій БД у разі реалізації однієї і тієї ж ін'єкції;

– під час демонстрації використовується SQL-запит з об'єднанням рядків публічної і прихованої таблиці, що дозволяє отримати список користувачів з таблиці user, який зазвичай є прихованим;

– демонстрація реакції захищеної системи у відповідь на запит з SQL-ін'єкцією (видається помилка) тощо.

Розглянемо особливості реалізації моделі дослідження SQL-ін'єкцій за допомогою валідаторів. Для захисту системи від можливості виконання SQL-ін'єкцій необхідний попередній аналіз значень параметрів, що надходять від користувача (строкових параметрів). Не можна безпосередньо використовувати значення параметрів в SQL-запитах. Було прийнято рішення розробити ряд класів, які виконують валідацію. Розроблено три валідатора (рис. 2):

– валідатор строкових значень «TStringValidator»;

– валідатор цілочисельних значень «TIntValidator»;

– валідатор значень з плаваючою точкою «TFloatValidator».

З діаграми класів слідує, що всі три валідатора успадковують свою поведінку від класу «TCustomValidator». У нашій архітектурі даний клас є «батьком» для всіх валідаторів і концентрує в собі базові функції для здійснення валідації. Клас «TCustomValidator» визначає не тільки базовий метод, який буде виконувати перевірку введених параметрів, але і механізм оповіщення про негативні результати валідації (властивість «UseException» типу «Boolean» визначає, чи буде використаний механізм винятків, або ж буде використаний «тихий режим», функція «RaiseException» – генерує або не генерує виключення залежно від значення властивості «UseException»).

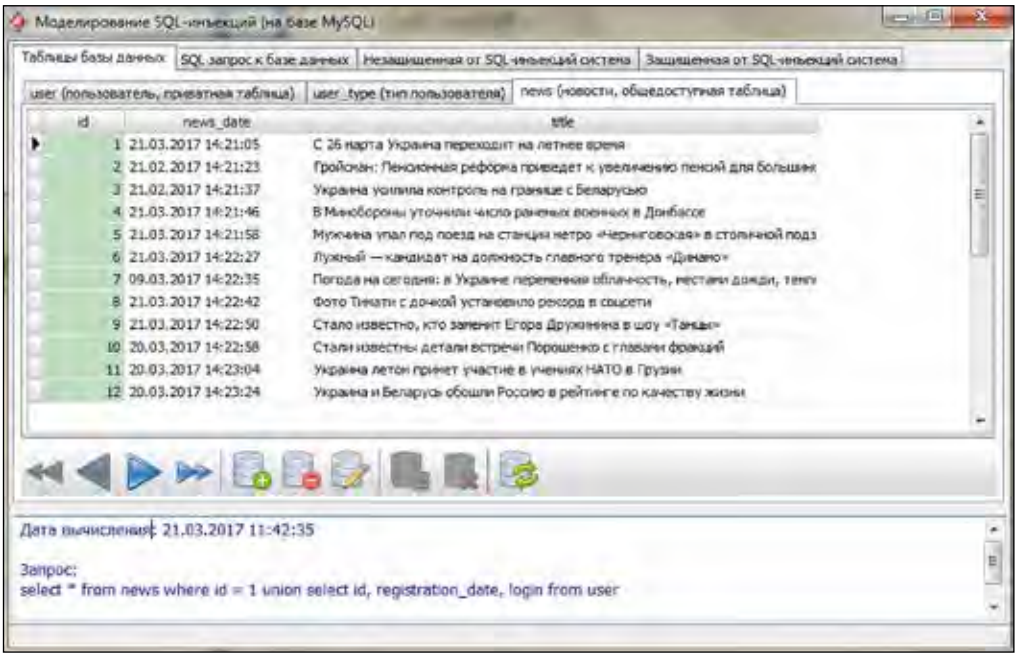

Рис. 1. Таблиці програмного засобу

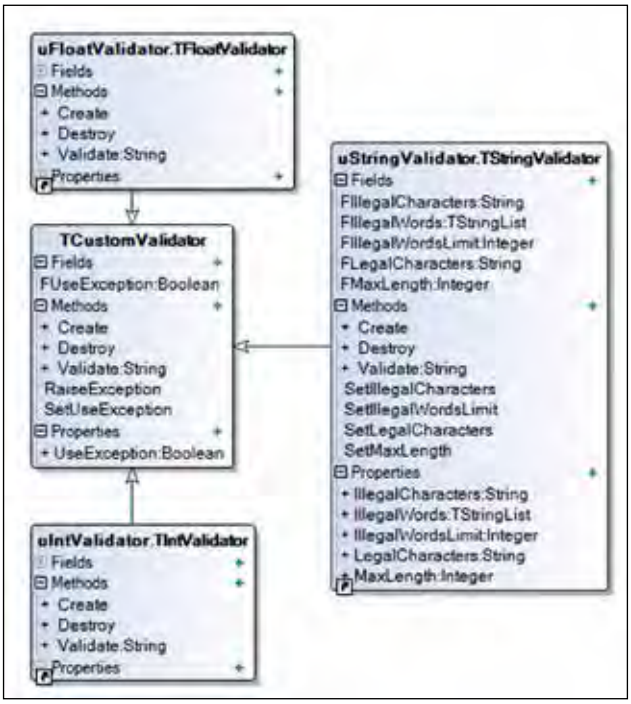

Рис. 2. Діаграма класів

Клас «TStringValidator» слугує для аналізу строкових параметрів і концентрує в собі такі принципи боротьби з SQL-ін'єкціями: фільтрація параметрів, відсікання параметрів, фільтрація зарезервованих слів. Цей клас реалізує в собі можливості для запобігання всіх видів SQL-ін'єкцій через рядкові змінні (використання UNION, екранування хвоста запиту, розщеплення SQL-запиту тощо).

Принцип валідації строкових значень, реалізований в даному класі, ґрунтується на використанні трьох наборів:

– набір дозволених символів (властивість «LegalCharacters»);

– набір заборонених символів (властивість «IllegalCharacters»);

– набір заборонених слів (властивість «IllegalWords»).

Дані властивості будуть визначені програмістом в процесі використання валідатора, тому що для різних СУБД існують свої особливості синтаксису. У разі, якщо якийсь набір не було наведено, то відповідна перевірка просто не буде виконуватися.

Процедуру валідації строкового параметра можна розділити на наступні етапи:

1. Перевірка довжини рядка (обмеження розміру рядка визначається властивістю «MaxLength»);

2. Перевірка наявності неприпустимих символів за розділами дозволених і заборонених символів (визначається властивостями «LegalCharacters» і «IllegalCharacters» відповідно);

3. Перевірка наявності заборонених слів (властивість «IllegalWords» визначає перелік зарезервованих слів, а властивість «IllegalWordsLimit» визначає їх гранично-допустиму кількість в аналізованому рядку).

Валідатор цілочисельних значень «TIntValidator» (як і «TFloatValidator») влаштований значно простіше свого аналога «TStringValidator». Це пояснюється тим, що під час виконання валідації цілого числа (або числа з плаваючою точкою) перевіряється лише можливість його перетворення. Ніяких додаткових обчислень, крім виконання спроби перетворення строкового значення, не проводиться.

Приклад використання строкового валідатора:

```
// Визначення валідатора
StringValidator : TStringValidator;
…
// Ініціалізація валідатора
StringValidator := TStringValidator.Create;
// Визначення набору заборонених символів
StringValidator.IllegalCharacters := '\|/;*()[]"'=-%' + Char(39);
// Визначення припусимої кількості заборонених слів
StringValidator.IllegalWordsLimit := 0;
// Визначення набору заборонених (зарезервованих) слів
StringValidator.IllegalWords.Add('or');
StringValidator.IllegalWords.Add('and');
StringValidator.IllegalWords.Add('select');
StringValidator.IllegalWords.Add('union');
StringValidator.IllegalWords.Add('from');
StringValidator.IllegalWords.Add('where');
StringValidator.IllegalWords.Add('insert');
StringValidator.IllegalWords.Add('delete');
StringValidator.IllegalWords.Add('update');
…
// Використання валідатора
StringValidator.Validate (Text);
```

Приклад використання цілочисельного валідатора:

```
IntValidator : TIntValidator;
…
IntValidator := TIntValidator.Create;
…
IntValidator.Validate (Text);
```

Приклад використання валідатора значень з плаваючою точкою:

```
...
```

FloatValidator : TFloatValidator;

…

FloatValidator := TIntValidator.Create;

…

FloatValidator.Validate (Text);

Клас «TCustomValidator» – базовий клас для реалізації будь-яких валідаторів:

Field Summary: internal Boolean (FUseException – використовувати механізм виключень для повідомлення про некоректні значення).

Property Summary: public Boolean (UseException – використовувати механізм виключень для повідомлення про некоректні значення).

Constructor Summary: Create() (конструктор класу).

Method Summary: public Sub (Destroy() – деструктор класу); protected internal Sub (RaiseException(Msg: string) – процедура створення виключень); internal Sub (SetUseException(Value: Boolean) – метод встановлення властивості UseException); public function String (Validate(Value: String) – метод валідації).

Клас «TStringValidator» – валідатор строкових параметрів для SQL-запитів.

Field Summary: internal String (FIllegalCharacters – заборонені символи); internal TStringList (FIllegalWords – заборонені слова); internal Integer (FIllegalWordsLimit – ліміт заборонених слів під час їх підрахунку); internal String (FLegalCharacters – дозволені символи); internal Integer (FMaxLength – обмеження довжини рядка).

Property Summary: public String (IllegalCharacters – заборонені символи); public TStringList (IllegalWords – заборонені слова); public Integer (IllegalWordsLimit– ліміт заборонених слів під час їх підрахунку); public String (LegalCharacters – дозволені символи); public Integer (MaxLength – обмеження довжини рядка).

Constructor Summary: Create() – конструктор класа.

Method Summary: public Sub (Destroy()–деструктор класа); internal Sub (SetIllegalCharacters(Value: String) – метод визначення заборонених символів); internal Sub (SetIllegalWordsLimit(Value: Integer) – ліміт заборонених слів під час їх підрахунку); internal Sub (SetLegalCharacters (Value: String) – встановити дозволені символи); internal Sub (SetMaxLength (Value: Integer) – встановити обмеження довжини рядка); public function String (Validate(Value: String) – метод валідації).

Клас «TIntValidator» – валідатор цілочисельних значень для SQL-запитів.

Constructor Summary: Create() – конструктор класу.

Method Summary: public Sub (Destroy() – деструктор класу); public function String (Validate(Value: String) – метод валідації).

Клас «TFloatValidator» – валідатор значень з плаваючою точкою SQL-запитів.

Constructor Summary: Create() – конструктор класу.

Method Summary: public Sub (Destroy () – деструктор класу); public function String (Validate(Value: String) – метод валідації).

Розглянемо більш детально роботу моделі для дослідження SQL-ін'єкцій. Як зазначалося вище, у моделі реалізована можливість перегляду результату ін'єкції за незахищеної бази даних. При цьому в разі реалізації однієї і тієї ж ін'єкції можна бачити, як реагує на запит захищена база даних і незахищена від SQL-ін'єкцій база даних (рис. 3).

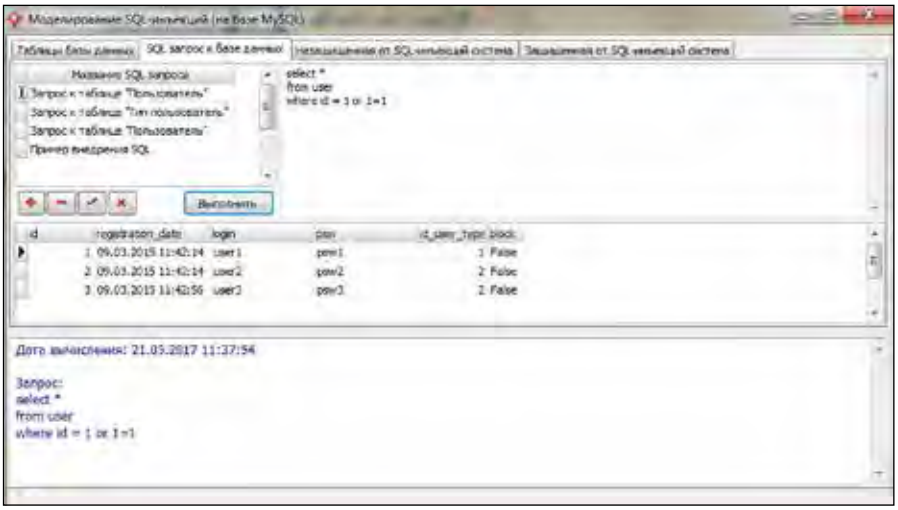

Рис. 3. SQL-запити

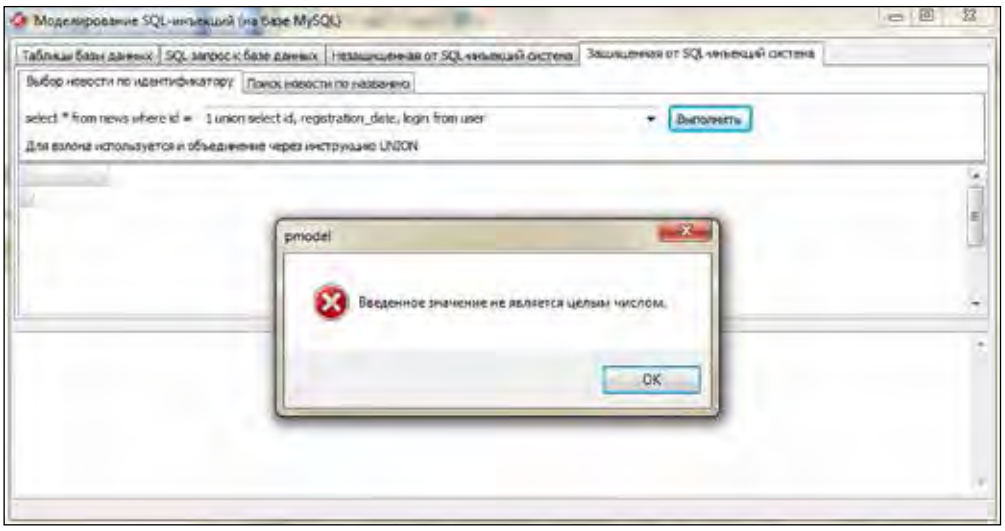

Рис. 4. Результат ін'єкції до захищеної системи

Є можливість демонстрації використання SQL-запитів з об'єднанням рядків публічної і прихованої таблиці, що дозволяє отримати список користувачів з таблиці (user), який зазвичай є прихованим. У разі впровадження SQL-ін'єкцій до захищеної системи у відповідь на запит видається помилка (рис. 4).

Висновки. Пропоноване ПЗ «Моделювання атак типу SQl-ін'єкцій до баз даних» впроваджено у навчальний процес МНУ ім. В.О.Сухомлинського у 2017–2018 н. рр. Останнє дозволило збільшити продуктивність використання навчального часу на 40% під час вивчення теми «Методи та способи захисту даних баз даних» (дисципліна «Бази даних та інформаційні системи») та «Розробка програмного забезпечення з підключенням баз даних» (дисципліна «Програмування»).

Список літератури:

1. Макаров Т. MS SQL Injection, 2010. URL: http://whatis.ru/razn/razn13.shtmll (дата звернення: 14.12.2018).
2. Посимвольный перебор в базах данных на примере MySQL. URL: https://www.securitylab.ru/contest/212099.php (дата звернення: 17.01.2019).
3. Cerrudo C. Manipulating Microsoft SQL Server Using SQL Injection. Application Security, Inc., 2005. 14 p.
4. Maor O., Shulman A. Blindfolded SQL Injection. Imperva. Inc, 2003. 17 p.
5. Maor O., Shulman A. SQL Injection Signatures Evasion. Imperva. Inc, 2004. 17 p.
6. SQL-Injection на примерах. URL: http://injection.rulezz.ru/sql-injection-by-example.html (дата звернення: 11.01.2019).
7. SQL-Injection. Are Your Web Applications Vulnerable? // SPI Labs. SPI Dynamics, 2002. 31 p.
8. SQL-инъекция в MySQL. *SecurityLab.* 2004. URL: https://www.securitylab.ru/contest/212101.php (дата звернення: 23.11.2018).

ПРИМЕНЕНИЕ ВАЛИДАТОРОВ ДЛЯ РЕАЛИЗАЦИИ МОДЕЛИ ИССЛЕДОВАНИЯ SQL-ИНЪЕКЦИЙ В БАЗАХ ДАННЫХ

В статье описана разработка программного обеспечения для реализации методов SQL-инъекций с целью использования в учебных целях. Моделируются случаи реализации SQL-вставок, выделяются их особенности для понимания механизмов работы. Приведены особенности реализации модели исследования SQL-инъекций с помощью валидаторов. Рассмотрена работа модели для исследования SQL-инъекций. В модели реализована возможность просмотра результата инъекции в случае защищенной и незащищенной от SQL-инъекций базы данных.

***Ключевые слова**: база данных, SQL-инъекция (SQL-Injection), программное обеспечение, защита данных, атака.*

APPLICATION OF VALIDATORS FOR THE IMPLEMENTATION OF THE MODEL OF RESEARCH OF SQL INJECTIONS IN DATABASES

The article describes the development of software for the implementation of SQL injection methods for educational purposes. The cases of implementing SQL inserts are modeled, their features are highlighted to understand the mechanisms of operation. The features of the implementation of the SQL injection study model using validators are given. Considered the work of the model for the study of SQL injection. The model has the ability to view the injection result in the case of a database protected and unprotected from SQL injection.

***Key words**: database, SQL Injection, software, data protection, attack.*