УДК 004.492:[004.774+004.439 DOI: 10.31673/2412-4338.2023.045963

Залива В.В.
Державний університет інформаційно-комунікаційних технологій, Київ

МЕТОД ВИЯВЛЕННЯ ТА УСУНЕННЯ ВРАЗЛИВОСТІ INPUT КОМПОНЕНТІВ У МОДЕЛІ DOM ЗА ДОПОМОГОЮ fDOM

Анотація: *У статті розглядається прогресивний підхід до забезпечення безпеки веб-додатків шляхом ідентифікації та ліквідації вразливостей вхідних компонентів у моделі Документного Об'єктового Моделювання (DOM). Основна увага приділяється розробці fDOM - формальної версії DOM, яка автоматизує процес "санітації" атрибутів, активно видаляючи потенційно небезпечний вміст, зокрема у випадках, коли мова йде про теги <script> і пов'язані з ними загрози, як-от міжсайтовий скриптинг (XSS).*

У статті детально розглядається існуюча структура DOM, виявляються основні вразливості, а також вносяться нововведення у вигляді методу sanitize для запобігання можливим атакам. Технічні аспекти включають алгоритми для очищення атрибутів та елементів, а також процедуру очищення вхідних даних. Демонструється ефективність запропонованої моделі за допомогою серії тестів та лем, які підтверджують відсутність шкідливого вмісту після процедури санітації.

Також розглядається верифікація методу виявлення та усунення вразливостей за допомогою системи формальної верифікації Isabelle/HOL, що підкреслює значення формальних методів у забезпеченні безпеки веб-додатків. У статті оцінюються сильні та слабкі сторони fDOM, висвітлюються потенційні напрямки для подальшого розвитку моделі, адаптації під динамічний контент та дослідження інтеграцій зі сторонніми службами.

Демонстрація ефективності fDOM за допомогою набору тестів підтвердила його здатність нейтралізувати потенційно небезпечний код, вказуючи на успіх цього підходу у забезпеченні вищого рівня безпеки веб-додатків.

Ключові слова: *fDOM, інформаційні системи, міжсайтовий скриптинг, DOM, Isabelle/HOL, інформаційна технологія, веб-додатки.*

Zalyva V.V.
State University of Information and Communication Technologies, Kyiv

METHOD OF DETECTION AND ELIMINATION OF VULNERABILITIES IN INPUT COMPONENTS OF THE DOM MODEL USING fDOM

Abstract: *The article discusses a progressive approach to securing web applications by identifying and eliminating vulnerabilities in input components of the Document Object Model (DOM). The main focus is on the development of fDOM - a formal version of DOM that automates the process of "sanitizing" attributes, actively removing potentially dangerous content, especially in cases involving <script> tags and associated threats such as cross-site scripting (XSS).*

The article thoroughly examines the existing structure of DOM, identifies key vulnerabilities, and introduces innovations in the form of a sanitize method to prevent potential attacks. Technical aspects include algorithms for cleaning attributes and elements, as well as a procedure for sanitizing input data. The effectiveness of the proposed model is demonstrated through a series of tests and lemmas that confirm the absence of harmful content after the sanitization procedure.

The verification of the method for detecting and eliminating vulnerabilities using the formal verification system Isabelle/HOL is also discussed, highlighting the importance of formal methods in securing web applications. The article evaluates the strengths and weaknesses of fDOM, sheds light on potential directions for further development of the model, adaptation for dynamic content, and exploration of integrations with third-party services.

Demonstration of the effectiveness of fDOM using a set of tests confirmed its ability to neutralize potentially dangerous code, indicating the success of this approach in providing a higher level of security for web applications.

Keywords: *fDOM, information systems, cross-site scripting, DOM, Isabelle/HOL, information technology, web applications.*

1. Вступ

Сучасний Інтернет-простір, насичений складними веб-додатками, стикається з безпрецедентними викликами у сфері кібербезпеки. Однією з ключових областей, де безпека постійно піддається випробуванням, є модель Документного Об'єктового Моделювання (DOM) - фундаментальний елемент веб-технологій, який визначає структурну ієрархію веб-сторінок. Незважаючи на її важливість, DOM не застрахована від вразливостей, які можуть бути використані для атак на веб-додатки.

Однією з найбільш поширених та ризикованих вразливостей є можливість вставки шкідливого коду через атрибути елементів. Така вразливість виявляється зокрема у випадках використання тегів <script>, що дозволяє виконувати JavaScript на стороні клієнта, потенційно призводячи до атак, таких як міжсайтовий скриптинг (XSS). Цей вступ розкриває потребу у розробці нових, більш ефективних методів захисту веб-додатків.

У відповідь на цю проблематику, була розроблена нова модель, відома як fDOM (формальний DOM). Ця модель вводить додатковий рівень безпеки, який діє автоматично, "санітарно" очищаючи атрибути від потенційно шкідливого вмісту, зокрема видаляючи теги <script>. Основою fDOM є функція sanitize, яка автоматично виконує цю роль, гарантуючи, що навіть у випадку спроб вставки шкідливого коду, він буде видалений до того, як потрапить у фінальну структуру DOM.

2. Аналіз літературних даних і постановка проблеми

Розробка веб-компонентів HTML5: Goldstein і Sutton (2018) описують використання Polymer для створення веб-компонентів HTML5. Їх робота може бути корисною для розуміння загальних принципів створення веб-компонентів, що є основою для аналізу та усунення вразливостей.

Безпека веб-додатків: Книга Hoffman (2020) охоплює різні аспекти безпеки веб-додатків, включаючи вразливості та способи їх усунення. Це може бути особливо релевантно для розуміння потенційних вразливостей у компонентах DOM.

Практичне використання веб-компонентів: Grigsby (2019) розглядає веб-компоненти у практичному аспекті. Ця робота може надати інсайти щодо реалізації та можливих викликів, пов'язаних з веб-компонентами.

Формалізація веб-компонентів: Дослідження Brucker та Herzberg (2020) вносить важливий вклад у розуміння теоретичних аспектів веб-компонентів. Це може бути корисно для глибшого аналізу структури та вразливостей DOM.

Стандарти DOM: WHATWG (2019) пропонує "живий стандарт" DOM, який є актуальним для розуміння поточних специфікацій та практик у роботі з DOM.

Перебудова веб-додатків з використанням Shadow DOM: Дослідження Oh, Ahn і Kim (2017) фокусується на використанні Shadow DOM для покращення підтримки веб-додатків. Цей аспект може бути релевантним для аналізу методів усунення вразливостей в DOM.

"Towards a Lightweight, Hybrid Approach for Detecting DOM XSS": Ця робота фокусується на виявленні вразливостей DOM XSS, використовуючи підходи динамічного відстеження забруднень, що дозволяє аналізувати код для виявлення вразливостей DOM XSS під час виконання. Однак, такий підхід веде до значного збільшення часу завантаження сторінок (на 16,8%), що може обмежити його застосування в певних умовах, наприклад, як захисту в браузері.

"Exposing and Addressing Security Vulnerabilities in Browser Text Input Fields", Asmit Nayak, Rishabh Khandelwal, Kassem Fawaz. У публікації виявлено дві вразливості в полях

вводу, включаючи наявність паролів у відкритому тексті в HTML-коді веб-сторінок. Дослідження включає розробку доказу концепту розширення, яке використовує методи статичного та динамічного внедрення коду для обходу процесу перегляду веб-магазину. Описано два контрзаходи для вирішення цих ризиків: JavaScript-пакет для негайного впровадження розробниками веб-сайтів для захисту чутливих полів вводу та рішення на рівні браузера, яке попереджає користувачів, коли розширення отримує доступ до чутливих полів вводу.

3. Метод виявлення та усунення вразливості input компонентів

Модель DOM (Документного Об'єктового Моделювання) відіграє ключову роль у веб-технологіях, формуючи структурну ієрархію веб-сторінок. Ця технологія, як і багато інших, не застрахована від вразливостей, що можуть бути використані для нападів на веб-додатки. Зокрема, поширеною проблемою є вставка шкідливого коду через атрибути елементів, наприклад, за допомогою тегів <script>, що виконують JavaScript, що може призвести до таких атак, як міжсайтовий скриптинг (XSS).

При аналізі існуючої моделі DOM, що включає елементи з атрибутами та дочірніми вузлами, розглядається наступне визначення:

datatype Node = Element string "Attribute list" "Node list"

and Attribute = Attr string string

Мета полягає в тому, щоб гарантувати, що жоден атрибут не містить шкідливий рядок <script>, який може бути використаний для вставки шкідливого коду:

definition has_malicious_content :: "Attribute ⇒ bool" where "has_malicious_content (Attr _ value) = (substring '<script>' value)"

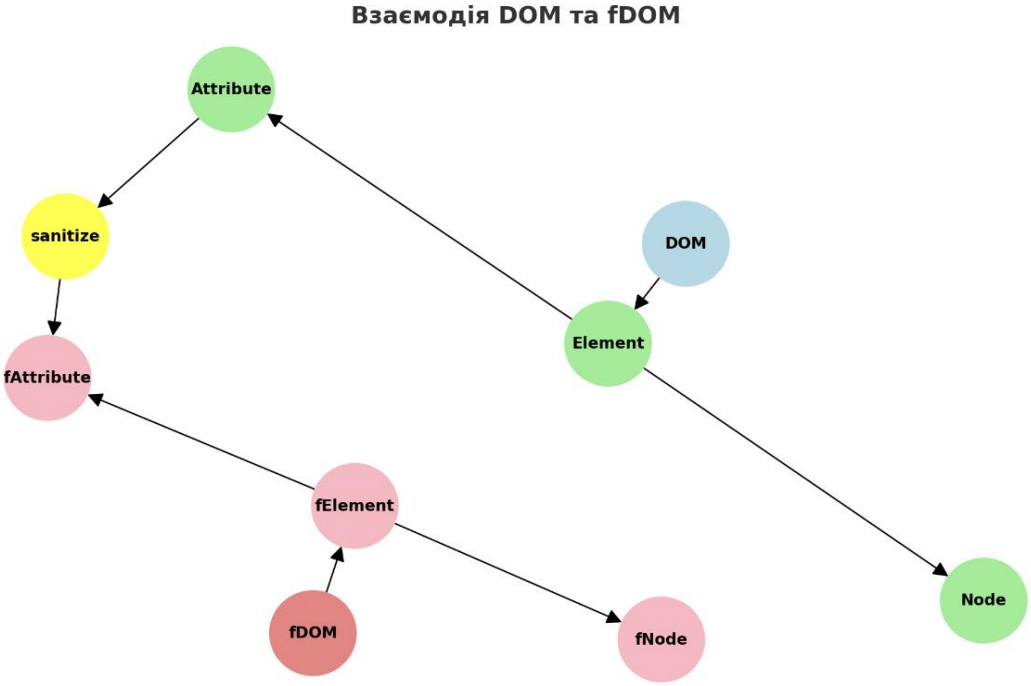

Рис. 1. Взаємодія DOM та fDOM

Уведення нової моделі fDOM та функції sanitize для видалення потенційно небезпечного вмісту включає:

datatype fNode = fElement string "fAttribute list" "fNode list"

and fAttribute = fAttr string string

definition sanitize :: "string ⇒ string" where "sanitize s = replace_all '<script>' '' s"

definition safe_attr :: "Attribute ⇒ fAttribute" where "safe_attr (Attr name value) = fAttr name (sanitize value)"

Лема, що підтверджує ефективність розробленої моделі:

"¬ has_malicious_content (safe_attr a)" by (simp add: has_malicious_content_def safe_attr_def sanitize_def)

Ця лема демонструє, що атрибут, оброблений функцією санітування, не містить шкідливих елементів.

4. Імплементація методу

У цій частині представлено реалізацію методу для очищення тегів <script> з атрибутів та введено функцію sanitize, яка призначена для вилучення або нейтралізації можливо небезпечних елементів з вхідних даних.

Структура fDOM включає:

1. Метод очищення атрибутів елемента (sanitizeAttributes): Проходить через атрибути елемента, видаляючи ті, що містять початкові символи 'on' або включають 'javascript:'.

2. Метод очищення елементів (sanitizeElements): Усуває елементи з тегом 'script' та застосовує метод sanitizeAttributes до кожного елемента.

3. Основна функція очищення (sanitize): Використовує DOMParser для аналізу вхідних даних як 'text/html'.

Код імплементації методу:

```
const fDOM = (function() {
  function sanitizeAttributes(element) {
    const attributes = element.attributes;
    for (let i = 0; i < attributes.length; i++) {
      const attribute = attributes[i];
 if (attribute.name.startsWith('on') || attribute.value.includes('javascript:')) {
        element.removeAttribute(attribute.name);
      }
    }
  }
  function sanitizeElements(element) {
    if (element.tagName.toLowerCase() === 'script') {
      element.remove();
      return;
    }
    sanitizeAttributes(element);
    for (let i = 0; i < element.childNodes.length; i++) {
      sanitizeElements(element.childNodes[i]);
    }
  }
  function sanitize(input) {
    const parser = new DOMParser();
    const doc = parser.parseFromString(input, 'text/html');
    sanitizeElements(doc.body);
    return doc.body.innerHTML;
  }
  return { sanitize: sanitize };
})();
```

5. Тестування імплементованого методу

Для валідації та демонстрації функціональності розробленої моделі була розроблена серія тестів, які демонструють її здатність ефективно працювати. Спочатку потрібно визначити набір даних для тестування:

```
const testData = [
  '<script>alert("Атака XSS")</script>',
```

```
    'Hello <script src="malicious.js"></script> World',
];
```

Процедура тестування включає наступні кроки:

```
function runTests() {
    testData.forEach((data, index) => {
        const sanitizedData = fDOM.sanitize(data);
        console.log(`Тест ${index + 1}:`);
        console.log('Original:', data);
        console.log('Sanitized:', sanitizedData);
        console.log('-------------------------');
    });
}
runTests();
```

Цей процес дозволяє оцінити, наскільки ефективно розроблена система може очищати тестові дані від потенційно шкідливих елементів, тим самим демонструючи її практичну придатність.

6. Висновки

У статті представлено розробку та впровадження моделі fDOM, спрямованої на зміцнення безпеки веб-додатків шляхом ефективного виявлення та усунення вразливостей, особливо пов'язаних з вставкою шкідливого коду через атрибути елементів у моделі DOM. Основою моделі є механізм автоматичного "санітування" вхідних даних, зокрема видалення тегів <script>, що є критичним для запобігання атакам типу міжсайтового скриптингу (XSS). Демонстрація ефективності fDOM за допомогою набору тестів підтвердила його здатність нейтралізувати потенційно небезпечний код, вказуючи на успіх цього підходу у забезпеченні вищого рівня безпеки веб-додатків.

Список використаної літератури

1. Goldstein, A., & Sutton, P. “ Developing HTML5 Web Components with Polymer.” 2018.
2. Hoffman, A. "Web Application Security: Exploitation and Countermeasures for Modern Web Applications.", 2020.
3. Grigsby, J. “Веб-компоненти на практиці.” Manning Publications, 2019.
4. Achim D. Brucker and Michael Herzberg. “A Formalization of Web Components”, 2020.
5. WHATWG. DOM – “Living Standard.”, 2019.
6. J. Oh, W. H. Ahn and T. Kim. “Web app restructuring based on shadow DOMs to improve maintainability.”, 2017
7. Moshenchenko M., Zhurakovskyi B., Poltorak V., Bondarchuk A., Korshun N. Optimization Algorithms of Smart City Wireless Sensor Network Control / CEUR Workshop Proceedings, 2021, 3188, p. 32–42
8. Zhebka V., Gertsiuk M., Sokolov V., Malinov V., Sablina M. Optimization of Machine Learning Method to Improve the Management Efficiency of Heterogeneous Telecommunication Network CEUR Workshop Proceedings, 2022, 3288, p. 149–155
9. Moshenchenko M., Zhurakovskyi B., Poltorak V., Bondarchuk A., Korshun N. Optimization Algorithms of Smart City Wireless Sensor Network Control // CEUR Workshop Proceedings, 2021, 3188, p. 32–42/
10. Shevchenko O., Bondarchuk A., Polonevych O., Zhurakovskyi B.,Korshun N. Methods of the objects identification and recognition research in the networks with the IoT concept support // CEUR Workshop Proceedings, 2021, 2923, страницы 277–282