Giải pháp kiểm tra đồng thời mức độ an toàn và khả năng tiếp cận của trang web

Vũ Thị Hương Giang, Phan Văn Huy, Vũ Văn Trung

***Tóm tắt*— An toàn và dễ tiếp cận là hai khía cạnh độc lập của chất lượng trang web. Tuy nhiên, nếu kiểm tra riêng từng tiêu chí này thì sẽ khó đánh giá mối tương quan giữa các khía cạnh này. Bài viết này mô tả một cách tiếp cận cho phép kiểm tra tính an toàn và tính dễ tiếp cận cho các nội dung web, được hiển thị ở trình duyệt phía máy khách (client). Chúng tôi đề xuất hai phương pháp kiểm tra. Thứ nhất, các vi phạm về thuộc tính và giá trị thuộc tính của từng nội dung web (HTML node) được kiểm tra bằng các luật. Các luật được định nghĩa dựa theo chuẩn ISO/IEC 40500 [9] để phát hiện các vi phạm tính dễ tiếp cận và dựa theo dấu hiệu nhận biết vi phạm của OWASP [12, 13, 14] để phát hiện các vi phạm tính an toàn. Thứ hai, sử dụng các bản thảo (scripts) và dữ liệu do người dùng đưa vào để so khớp với các mẫu tấn công đã được chúng tôi định nghĩa sẵn. Hai phương pháp này có thể được thực hiện riêng rẽ hoặc đồng thời và đã được cài đặt thử nghiệm dưới dạng ứng dụng web.**

***Abstract*— Accessibility and security are two independent aspects of the website quality. For every web content, if they are separately considered and evaluated, the joint violation could not be highlighted. This paper proposes an approach for customizing the multi-aspects evaluation of web contents that are displayed at the client's browser. This approach is composed of two methods. In the first method, we define two rules sets to check the violation of the HTML nodes' attributes and values. The ISO 40500 [13] - based rules allow detecting accessibility violations. The OWASP [12] based rules allow detecting security violations. In the second method, we define the attack patterns for checking the conformance of the scripts and inputs data from users. These checking methods could be jointly or separately operated. The approach is experimented in the form of a web application.**

Từ khóa— nút HTML; ISO/IEC 40500; lỗ hổng web; kiểm thử mờ; OWASP top 10.

Keywords— HTML node; ISO/IEC 40500; web vulnerabilities; fuzzing; OWASP Top 10.

I. GIỚI THIỆU

Trang web đã trở thành một kênh thông tin quan trọng, phổ biến trong mọi lĩnh vực của đời sống. Tuy nhiên, một thực trạng hiện nay là số lượng các trang web tồn tại lỗ hổng bảo mật chiếm một tỷ lệ khá cao. Theo nghiên cứu [16] về tình trạng an toàn thông tin của các trang web trên thế giới, thì có tới 22% các trang web được khảo sát tồn tại các lỗ hổng an toàn. Nguyên nhân chính của tình trạng này là do vấn đề bảo mật của nhiều trang web chưa được quan tâm đúng mức.

Bên cạnh đó, phần lớn các trang web hiện nay quan tâm đến khía cạnh kết xuất nội dung web (thiết kế giao diện đồ họa đẹp, thu hút người đọc cho các nội dung cần hiển thị trực quan) hơn là khía cạnh điều hướng, hỗ trợ tương tác (hỗ trợ khả năng tiếp cận nội dung của người dùng).

Một trang web được coi là dễ tiếp cận nếu nó không tạo bất cứ rào cản nào cho người dùng trong mọi bối cảnh sử dụng [17]. Trang web dễ tiếp cận cần đảm bảo phục vụ không chỉ cho đối tượng người dùng thông thường, mà kể cả cho người khuyết tật với những năng lực hành vi vốn có của họ.

Thực tế cho thấy, các trang web dễ tiếp cận thường không an toàn và các trang web an toàn thường khó tiếp cận. Ví dụ, trên một trang web, người dùng dễ tiếp cận các hình ảnh có kích thước lớn, đặt ở đầu trang (header) hay chính giữa màn hình hơn là các hình ảnh có kích thước khiêm tốn, đặt tại các vị trí khác. Các trang web có nội dung không lành mạnh thường áp dụng nguyên tắc này để thiết kế những hình ảnh nhạy cảm thu hút sự tò mò của người dùng. Các nội dung này dễ tiếp cận với người dùng nên thường bị chèn thêm nhiều đường dẫn không an toàn, giả mạo trang web khác hay đường dẫn quảng cáo. Mặt khác chúng ta thường thấy các trang web yêu cầu người dùng nhập mã CAPTCHA khi bình luận, điền thông tin vào biểu mẫu để đảm bảo dữ liệu là do người dùng thực đưa vào. Tuy nhiên, phần lớn các mã CAPTCHA hiện nay được biểu diễn dưới dạng hình ảnh cách điệu hay che khuất (để tránh các công cụ nhận dạng tự động), hoặc chỉ được phát âm bằng một ngôn ngữ thông dụng như tiếng Anh. Điều này dễ gây nhầm lẫn, khó khăn cho người dùng, làm người dùng phải thử nhiều lần mới nhập đúng CAPTCHA, thậm chí không thể hoàn

thành việc này để truy cập vào trang web.

Khi khảo sát mã nguồn các trang web hiển thị trên phía máy khách, nhiều nội dung ảnh trên các trang web vừa chứa lỗ hổng bảo mật, vừa vi phạm các quy định về tính dễ tiếp cận, ví dụ trong Bảng 1 dưới đây.

BẢNG 1. NỘI DUNG ẢNH VI PHẠM

Nội dung web	<img src="http://bank.com/transfer.do?acct=MARIA&amount=100000">
Vi phạm tính an toàn	Node <img> chứa lỗ hổng Cross-Site Request Forgery[4]. Cụ thể, thuộc tính src của node <img> trong mã nguồn ở trên có giá trị không hợp lệ, sử dụng phương thức GET để thực hiện các giao dịch tự động mà không có sự đồng ý của chủ tài khoản.
Vi phạm tính dễ tiếp cận	Node <img> vi phạm tiêu chí SC 1.1.1 của ISO/IEC 40500[10] vì chưa có thuộc tính alt để mô tả nội dung ảnh. Mô tả này được coi là văn bản thay thế cho nội dung ảnh: nếu vì lý do nào đó trình duyệt không thể hiện thị ảnh, người dùng vẫn nhận biết được ảnh này dùng vào mục đích gì nhờ vào giá trị thuộc tính alt.

Nhiều nghiên cứu đã sử dụng tập luật để kiểm tra tính dễ tiếp cận của trang web, ví dụ như các công cụ Accessibility Developer Tools [1] và AInspector Sidebar [7]. Tương tự, cũng có thể sử dụng tập luật được định nghĩa sẵn để kiểm tra tính an toàn, ví dụ như các luật phát hiện và phòng chống xâm nhập trái phép vào các ứng dụng web của OWASP ModSecurity [11]. Bên cạnh đó, kỹ thuật kiểm thử mờ (fuzzing) cũng hứa hẹn phát hiện được các lỗ hổng bảo mật mà các luật định nghĩa sẵn thường bỏ sót như SQL Injection, Cross-site scripting.... Kỹ thuật này được thực hiện bằng cách liên tục gửi các yêu cầu chứa bản thảo tấn công lên máy chủ, sau đó căn cứ vào kết quả trả về để xác định lỗ hổng phía máy chủ. Các công cụ tiêu biểu là Zed Attack Proxy [14], Webscarab [6], Acunetix [8].

Hướng nghiên cứu đề cập đến cả hai tiêu chí an toàn và dễ tiếp cận của trang web còn mới, chưa có nhiều công trình được công bố. Chúng tôi đề xuất hai phương pháp kiểm tra.

Với phương pháp thứ nhất, các vi phạm về thuộc tính và giá trị thuộc tính của từng nội dung web được kiểm tra bằng các tập luật. Chúng tôi định nghĩa 3 tập luật như sau. Tập luật đầu tiên nhằm kiểm tra vi phạm tính dễ tiếp cận, dựa trên chuẩn ISO/IEC 40500 [9]. Tập luật thứ hai là tập luật kiểm tra các vi phạm về tính an toàn, được định nghĩa dựa trên các dấu hiệu nhận biết vi phạm của OWASP [12]. Tập luật thứ ba bao gồm các luật kết hợp, được xây dựng theo mô hình kết nối vi phạm mô tả trong Mục II.C.

Còn đối với phương pháp thứ hai, các bản thảo và dữ liệu do người dùng đưa vào được so khớp với các mẫu tấn công (attack patterns). Các mẫu tấn công này được định nghĩa dưới dạng cấu trúc XML trong Mục II.D.

Cuối cùng, chúng tôi đề xuất mô hình kiểm tra, cho phép thực hiện riêng rẽ hoặc đồng thời hai phương pháp nói trên. Mô hình này được giới thiệu trong Mục III. Một số kết quả thực nghiệm được giới thiệu trong Mục IV.

Theo [13], có 8 yếu tố liên quan đồng thời tới cả tính an toàn và tính dễ tiếp cận của trang web, bao gồm: Trường hợp bổ sung nội dung (Additional text instances), các hình thức thay thế khác của CAPCHA (Alternate forms of CAPTCHA), bổ sung các tập tin (Additional files), sử dụng dịch vụ của bên thứ ba (Use of third-party services), thêm kịch bản phía máy khách (Additional client-side scripting), thời gian chờ của phiên linh hoạt (Flexible session timeouts), phục hồi tái xác thực (Re-authentication recovery), tính hợp lệ của mã nguồn (Code validity). Thông qua các yếu tố này, sẽ định nghĩa mối quan hệ giữa các kỹ thuật đảm bảo tính dễ tiếp cận WCAG 2.0 và các lỗ hổng bảo mật thuộc OWASP TOP 10 2007 [19]. Bên cạnh đó, trong [15]] phân tích ảnh hưởng của các công cụ đảm bảo an toàn mà người dùng nhận biết được trên giao diện website như CAPTCHA hay Virtual Keyboard đến tính dễ tiếp cận của trang web. Từ đó đưa ra khuyến cáo sử dụng các biện pháp đảm bảo an toàn mà người dùng không nhận biết được qua giao diện như SSL Connection, Server Authenticity Certificate hay các phương pháp mã hóa sử dụng khóa bí mật khi phát triển các trang web dễ tiếp cận. Trong [18], các tác giả chỉ ra sự cần thiết của chuẩn WCAG 1.0 đối với trang web, đồng thời nêu ra các điểm yếu thường dẫn dụ tấn công vào website chính phủ Kyrgyzstan như SQL injection, XSS. Tuy nhiên, ngay cả trong tình huống cụ thể này, mối quan hệ giữa khả năng truy cập và bảo vệ trang web chưa được làm rõ.

Có thể thấy, các nghiên cứu trên mới chỉ xét tới một vài trường hợp quan hệ giữa tính an toàn và tính dễ tiếp cận, chưa đưa ra được mô hình kết nối việc kiểm tra tính an toàn và tính dễ truy cập của trang web. Vì lý do đó, hiện nay chưa có công cụ nào có khả năng đánh giá đồng thời cả hai tiêu chí này. Về mặt ứng dụng, việc để cho người phát triển tự tổng hợp kết quả từ các công cụ kiểm tra riêng từng khía cạnh không mang lại nhiều ý nghĩa, do chúng không có các tiêu chí chung để đánh giá. Kết quả là, trang web an toàn vẫn khó tiếp cận hoặc trang web dễ tiếp cận vẫn chứa nhiều lỗ hổng.

Như vậy, cần công cụ cho phép kiểm tra đồng thời tính an toàn và tính dễ tiếp cận của từng nội dung web khi cần thiết. Với người dùng cuối, công cụ này giúp họ lựa chọn được các website an toàn và dễ tiếp cận nội dung. Với người phát triển, đây là công cụ hữu ích để xác định nguyên nhân gây ra vi phạm, vị trí vi phạm, có gợi ý khắc phục vi phạm và hỗ trợ hiệu chỉnh mã nguồn để loại bỏ vi phạm. Dưới đây đóng góp một giải pháp để xây dựng công cụ như vậy.

II. PHƯƠNG PHÁP KIỂM TRA TÍNH AN TOÀN VÀ TÍNH DỄ TIẾP CẬN

A. Đối tượng được kiểm tra

Đối tượng được kiểm tra trong bài báo này là các nội dung web một chiều hoặc hai chiều được mô tả dưới dạng các HTML node trong mã nguồn phía máy khách. Một HTML Node bao gồm các thuộc tính như ID, name, value (nội dung web hiện thị trên trình duyệt), type (tên thẻ), parent node, child node [3].

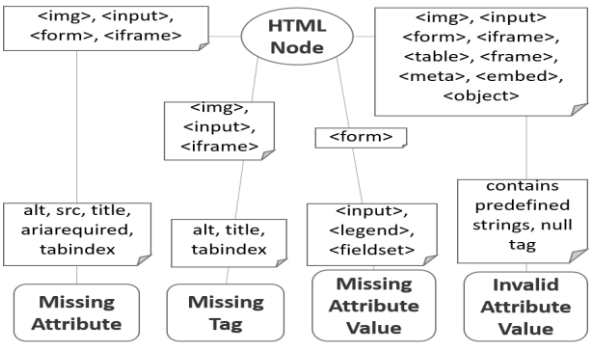
Nội dung web một chiều tương ứng với các HTML node có giá trị do phía máy chủ trả về, ví dụ như <image>, <table>, <a>. Phương pháp kiểm tra sử dụng luật xét đến các nội dung một chiều.

Còn các nội dung web hai chiều tương ứng với các HTML node có giá trị được xác định thông qua tương tác và nhập liệu của người dùng, ví dụ các thẻ <input>, <form>. Phương pháp kiểm tra sử dụng mẫu tấn công xét đến nội dung web hai chiều.

Việc vi phạm tính dễ tiếp cận xuất hiện khi hiển thị nội dung một chiều trên trình duyệt web và được xác định theo chuẩn ISO/IEC 40500.

Việc vi phạm tính an toàn xuất hiện khi hiện thị nội dung một chiều trên trình duyệt web, hoặc khi người dùng tương tác với nội dung hai chiều – được xác định theo OWASP top 10.

B. Mô hình kết nối vi phạm tính an toàn và vi phạm tính dễ tiếp cận



Hình 1. Mô hình kết nối

Hình 1 minh họa 4 kiểu vi phạm: thiếu thuộc tính (missing attribute), thiếu thẻ (missing tag), không có giá trị thuộc tính (missing attribute value), sai giá trị thuộc tính (invalid attribute value). Các kiểu vi phạm này có thể xảy ra với các HTML node cùng loại. Mô hình kết nối chỉ ra các vi phạm tính an toàn và tính dễ tiếp cận thuộc cùng 1 trong số 4 kiểu nói trên, có khả năng xảy ra trên cùng một loại HTML node. Các vi phạm như vậy sẽ có khả năng được kiểm tra đồng thời.

Bảng 2 minh họa hai trường hợp kết nối. Trường hợp đầu tiên, node <iframe> liên quan đến 2 vi phạm cùng kiểu thiếu thuộc tính. Trường hợp thứ hai, node <img> liên quan đến 2 vi phạm cùng kiểu sai giá trị thuộc tính. Các vi phạm loại này có thể được kiểm tra bằng các luật riêng hoặc sử dụng luật kết hợp.

BẢNG 2. KẾT NỐI VI PHẠM CHO NODE <IFRAME> VÀ <IMG>

HTML Node	Loại vi phạm	Vi phạm tính dễ tiếp cận	Vi phạm tính an toàn
<iframe>	Missing Attribute	Node <iframe> thiếu thuộc tính title → vi phạm tiêu chí SC2.4.1[20] Ví dụ vi phạm: <iframe src="banner-ad.html" > Advertisement</iframe>	Node <iframe> thiếu thuộc tính sandbox. Giải thích: thuộc tính sandbox giúp ngăn chặn các script thực thi được nhúng bên trong iframe, thuộc tính sandbox sẽ giúp node <iframe> trở nên an toàn hơn.

<img>	Invalid Attribute Value	Node <img> có thuộc tính longdesc với URI không tồn tại → vi phạm tiêu chí SC 1.1.1[21] [22]	Node <img> có giá trị của thuộc tính chứa lỗ hổng CSRF. Ví dụ: <img src="http://bank.com/transfer.do?acct=MARIA&amount=100000">

C. Mẫu tấn công

Các mẫu tấn công chính là các đoạn bản thảo được mô tả bằng cấu trúc XML. Chúng tôi định nghĩa 4 loại mẫu tấn công bằng bản thảo như sau: SQL Injection, XSS, XML Injection và Xpath Injection. Hình 2 minh họa một mẫu tấn công loại SQL Injection.

```
<attack_pattern>
<attack_category>
    <attack_type>SQL Injection</attack_type>
</attack_category>
<patterns>
    <pattern>1 or 1=1</pattern>
    <pattern>1' or '1'='1</pattern>
    <pattern>select @@servernamee</pattern>
    <pattern>select @@version</pattern>
    <pattern>select session_user;</pattern>
    <pattern>select current_setting('password_encryption');</pattern>
    <pattern>select current_setting('config_file');</pattern>
    <pattern>+if(benchmark(3000000,MD5(1)),NULL,NULL))%20%23</pattern>
    <pattern>and 0=benchmark(3000000,MD5(1))%20/*</pattern>
</patterns>
</attack_pattern>
```

Hình 2. Mẫu tấn công SQL injection sử dụng trong kỹ thuật fuzzing

III. MÔ HÌNH KIỂM TRA ĐỒNG THỜI TÍNH AN TOÀN VÀ TÍNH DỄ TIẾP CẬN

A. Mô hình kiểm tra

Mô hình kiểm tra đồng thời tính an toàn và tính dễ tiếp cận được mở rộng từ framework [3]. Framework này gồm các thành phần:

(1) Bộ phân tích mã nguồn (Parser);

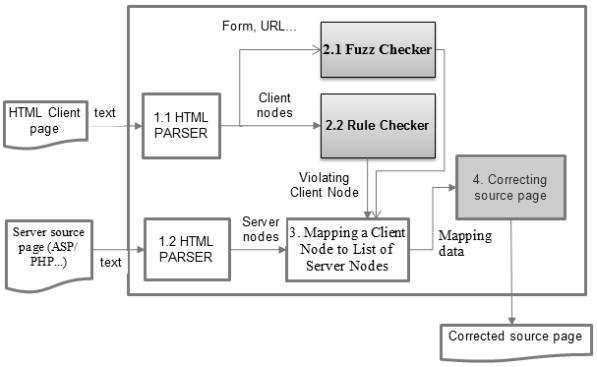
(2) Bộ phát hiện vi phạm (Rule checker);

(3) Bộ ánh xạ mã nguồn máy khách với mã nguồn máy chủ (Mapper);

(4) Cuối cùng là bộ chỉnh hiệu chỉnh vi phạm (Corrector).

Bộ phân tích mã nguồn sẽ phân tích mã nguồn thành danh sách các node máy khách. Sau đó, danh sách này được sử dụng làm đầu vào cho bộ phát hiện vi phạm. Khi đó ta sẽ thu được danh sách node máy khách vi phạm. Từ danh sách này, ta sử dụng bộ ánh xạ mã nguồn máy khách với mã nguồn máy chủ để tìm ra các node máy chủ tương ứng gây ra vi phạm. Cuối cùng, ta sử dụng bộ hiệu chỉnh vi phạm để hiệu chỉnh các vi phạm từ mã nguồn máy chủ.

Hình 3 minh họa cách thức để mở rộng framework này. Chúng tôi giữ nguyên các thành phần (1) và (3), thay thế thành phần (2), điều chỉnh cách thức hoạt động của thành phần (4) và quy trình thực hiện của các thành phần.



Hình 3. Mô hình kiểm tra vi phạm

Các bước xử lý trong mô hình kiểm tra và hiệu chỉnh lỗi bao gồm:

Bước 1: Sử dụng bộ Parser để phân tích trang web phía máy khách thành các node máy khách và phân tích trang web phía máy chủ thành các node máy chủ.

Bước 2: Phát hiện vi phạm. Nhận đầu vào là các node máy khách thu được trong bước 1 và trả về danh sách các node máy khách chứa vi phạm.

Chúng tôi sử dụng 2 bộ kiểm tra để phát hiện vi phạm: bộ Fuzz checker và bộ phát hiện vi phạm. Bộ phát hiện vi phạm có vai trò phát hiện vi phạm theo mô hình kết nối vi phạm tính an toàn và vi phạm tính dễ tiếp cận, được cài đặt bằng cách sử dụng luật kiểm tra. Bộ Fuzz checker được cài đặt bằng kỹ thuật Fuzzing, có vai trò phát hiện các vi phạm tính an toàn bằng cách so khớp các mẫu tấn công và kiểm tra kết quả trả về.

Thành phần kiểm tra có thể hoạt động theo 3 cách như sau:

- Sử dụng bộ Fuzz Checker để phát hiện các vi phạm tính an toàn.
- Sử dụng bộ bộ phát hiện vi phạm để phát hiện các vi phạm có trong node máy khách dựa theo các luật được xây dựng từ mô hình kết nối vi phạm tính an toàn và vi phạm tính dễ tiếp cận.
- Sử dụng kết hợp bộ Fuzz Checker và Bộ phát hiện vi phạm để phát hiện các vi phạm tính an toàn và tính dễ tiếp cận.

Bước 3: Sử dụng bộ Mapper để ánh xạ node máy khách vi phạm (thu được từ bước 2) với node máy chủ (thu được từ bước 1) gây ra vi phạm.

Bước 4: Hiệu chỉnh vi phạm. Từ dữ liệu ánh xạ thu được sau bước 3, ta sử dụng bộ hiệu chỉnh vi phạm để hiệu chỉnh node máy chủ đã gây ra vi phạm ở phía máy khách. Cụ thể:

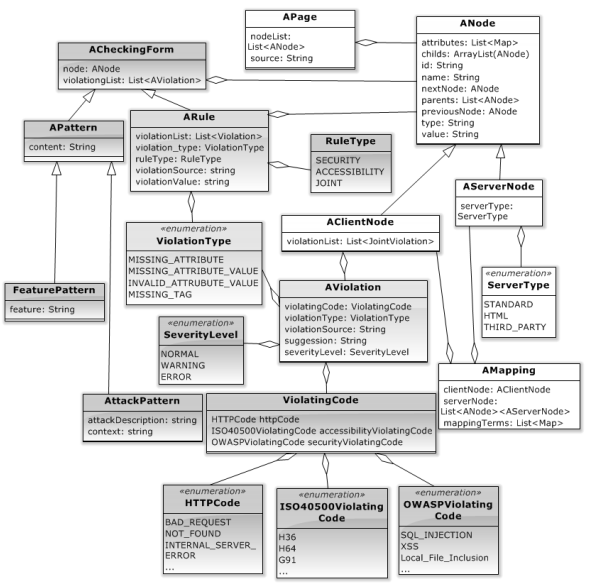
- Trường hợp node máy khách có kiểu vi phạm Mising Attribute thì node máy chủ được hiệu chỉnh bằng cách thêm thuộc tính bị thiếu.
- Trường hợp node máy khách có kiểu vi phạm Missing Tag thì node máy chủ được hiệu chỉnh bằng cách thêm thẻ bị thiếu.
- Trường hợp node máy khách có kiểu vi phạm Missing Attribute Value thì node máy chủ được hiệu chỉnh bằng cách thêm giá trị cho thuộc tính chưa được gán giá trị.

Kết quả trả về của bộ chỉnh hiệu chỉnh vi phạm là danh sách node máy chủ đã được hiệu chỉnh.

B. Các lớp mô tả dữ liệu xử lý

Chúng tôi đã định nghĩa lại và bổ sung một số cấu trúc dữ liệu như trong Hình 4. Cụ thể:

AViolation : là một vi phạm. Chúng tôi bổ sung thuộc tính severityLevel để biểu diễn mức độ vi phạm tính an toàn và thuộc tính suggestion để lưu giữ thông tin về cách hiệu chỉnh vi phạm.



Hình 4. Các lớp mô tả dữ liệu xử lý

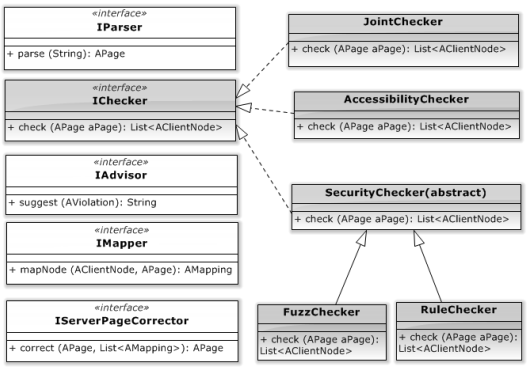
ViolationType: là các kiểu vi phạm, bao gồm: thiếu thuộc tính (*Missing attribute*), thiếu thẻ hỗ trợ (*Missing tag*), giá trị thuộc tính không hợp lệ (*Invalid attribute value*) và thiếu giá trị cho thuộc tính (*Missing attribute value*).

ACheckingForm: là lớp định nghĩa cách thức kiểm tra một node, có 2 lớp con kế thừa là ARule và APattern.

- ARule: định nghĩa luật để phát hiện vi phạm có trong một node máy khách.
- APattern: lớp định nghĩa mẫu tấn công, sử dụng khi kiểm tra theo kỹ thuật fuzzing.

AViolatingCode: là các mã vi phạm, bao gồm: các mã vi phạm theo chuẩn ISO/IEC 40500 (ISO ViolatingCode), các vi phạm tính an toàn đưa ra bởi OWASP (OWASPViolatingCode) và các vi phạm tính an toàn được trả về từ máy chủ (HTTPCode).

C. Giao diện cài đặt



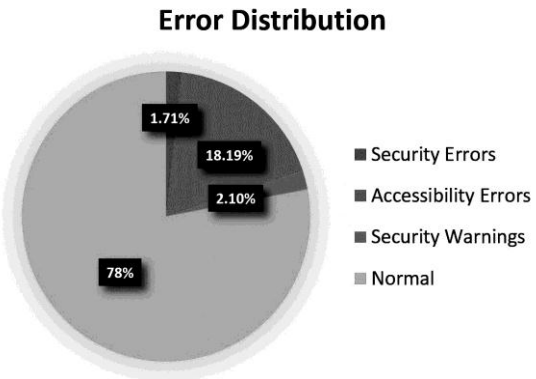
Hình 5. Các giao diện và cài đặt của giao diện IChecker

Như minh họa trong Hình 5, chúng tôi cài đặt lại giao diện IChecker. Các lớp đã cài đặt bao gồm: JointChecker (kiểm tra đồng thời tính an toàn và tính dễ tiếp cận), AccessibilityChecker (kiểm tra riêng tính dễ tiếp cận), SecurityChecker (kiểm tra riêng tính an toàn). SecurityChecker có 2 lớp con kế thừa là RuleChecker (cài đặt phương thức kiểm tra vi phạm bằng luật) và Fuzz Checker (cài đặt phương thức so khớp mẫu tấn công bằng kỹ thuật fuzzing).

IV. THỰC NGHIỆM

Chúng tôi đã sử dụng công cụ này để kiểm tra một số trang web thông dụng tại Việt Nam. Kịch bản thử nghiệm nhằm làm rõ mục đích nghiên cứu công cụ kiểm tra đồng thời hai khía cạnh dễ tiếp

cận và an toàn của chất lượng trang web. Cần giải quyết mối tương quan sau: nếu một trang web đã an toàn, nó có dễ tiếp cận hay không? Và nếu một trang web đã dễ tiếp cận, nó có an toàn hay không? Kết quả kiểm tra sẽ được minh họa dưới dạng biểu đồ và dạng bảng thống kê (Bảng 3 và Hình 6).



Hình 6. Biểu đồ thống kê tỷ lệ các loại vi phạm

BẢNG 3. KẾT QUẢ THỰC NGHIỆM VỚI CÁC TRANG WEB THÔNG DỤNG

STT	Trang web	Thống kê vi phạm tính dễ tiếp cận	Thống kê vi phạm tính an toàn	
			Sử dụng luật	Sử dụng mẫu tấn công
1	vnexpress.net	255	3	0
2	dantri.com.vn	158	0	0
3	lazada.vn	550	3	2
4	moet.gov.vn	154	1	1
5	thanhnien.vn	343	2	1
6	raovat123.com	146	0	18
7	megacard.vn	94	1	2
8	enbac.vn	592	4	0

Từ kết quả thực nghiệm cho thấy, công cụ kiểm tra đã cung cấp được phát hiện được các vi phạm tính an toàn và vi phạm tính dễ tiếp cận, các vi phạm tính dễ tiếp cận chiếm phần lớn tỷ lệ trong tổng số các vi phạm phát hiện được. Biểu đồ trên chỉ ra rằng hiện nay số lượng trang web ở Việt Nam hầu như chỉ chú trọng tính an toàn (tỷ lệ lỗi bảo mật nhỏ, ~1.7%) nhưng chưa chú trọng tới tính dễ tiếp cận của người dùng (tỷ lệ lỗi là 18.2%). Điều đó chứng tỏ các trang web được xây dựng theo các khuyến cáo đảm bảo an toàn thường bỏ sót các yêu cầu về tính dễ tiếp cận.

VI. KẾT LUẬN

Bài báo đã đề cập đến một giải pháp cho phép kiểm tra đồng thời hoặc riêng rẽ tính an toàn và tính dễ tiếp cận của trang web. Chúng tôi sử dụng các đối tượng HTML node thuộc mã nguồn phía máy khách để xác định vi phạm theo mô hình kết nối 2 tiêu chí an toàn – dễ tiếp cận, kết hợp với kỹ thuật Fuzzing. Các đối tượng này sau đó sẽ được ánh xạ sang mã nguồn phía máy chủ (nếu mở) để thực hiện bước hiệu chỉnh vi phạm. Việc phát hiện vi phạm từ mã nguồn có ưu điểm là dễ dàng xác định được nguyên nhân gây ra vi phạm, để từ đó xác định cách hiệu chỉnh vi phạm. Các nguyên nhân gây ra vi phạm có thể là do thiếu thuộc tính, do thiếu thẻ, thiếu giá trị hay do giá trị không hợp lệ. Bên cạnh đó, kỹ thuật Fuzzing cho phép phát hiện các vi phạm tính an toàn dựa trên các mẫu tấn công định nghĩa sẵn.

Kết quả thực nghiệm cho thấy, việc áp dụng mô hình kiểm tra này cho phép phát hiện đồng thời các vi phạm về tính an toàn và tính dễ tiếp cận của cùng một nội dung web. Hơn thế nữa, việc sử dụng kết hợp các phương pháp kiểm tra đã đề xuất cho phép phát hiện các nội dung vi phạm tính an toàn mà phương pháp phân tích mã nguồn chưa kiểm tra được.

Công cụ thử nghiệm của nghiên cứu đã kiểm tra được 39 tiêu chí trong tổng số 51 tiêu chí dễ tiếp cận theo chuẩn ISO/IEC 40500, trong đó có 24 tiêu chí được kiểm tra kết hợp với tính an toàn, đồng thời có 16 loại vi phạm tính an toàn đã được kiểm tra. Chức năng hiệu chỉnh lỗi đã được thử nghiệm độc lập với mã nguồn phía máy chủ viết bằng ngôn ngữ ASP.Net. Chúng tôi vẫn đang tiếp tục bổ sung các luật phát hiện việc vi phạm tính an toàn để nâng cao khả năng phát hiện lỗi vi phạm của công cụ kiểm tra cài đặt theo kết quả nghiên cứu, đề xuất ngưỡng chấp nhận được cho tỷ lệ lỗi vi phạm/ an toàn của website, đồng thời mở rộng chức năng hiệu chỉnh mã nguồn máy chủ áp dụng cho nhiều ngôn ngữ lập trình.

TÀI LIỆU THAM KHẢO

[1]. Google Accessibility Developer Tools. Chrome Web Store.[Online]

https://chrome.google.com/webstore/detail/accessibility-developer-t/fpkknkljclfencbdbgkenhalefipecmb?hl=en

[2]. Bypass Blocks.[Online], https://www.w3.org/TR/UNDERSTANDING-WCAG20/navigation-mechanisms-skip.html

[3]. Thi Huong Giang Vu, Dat Trinh Tuan, Van Hung Phan, "Checking and Correcting the Source Code of Web Pages for Accessibility" 2012. IEEE, Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF). pp. 1-4, 2012.

[4]. Cross-Site Request Forgery (CSRF).[Online]

https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)

[5]. Document Object Model (DOM). W3C.[Online] https://www.w3.org/DOM/

[6]. Fuzzing with WebScarab. OWASP.[Online] https://www.owasp.org/index.php/Fuzzing_with_WebScarab

[7]. AInspector Sidebar. Hoyt, Nicholas.[Online] https://addons.mozilla.org/enUS/firefox/addon/ainspector-sidebar/

[8]. HTTP Fuzzer Tool. Acunetix.[Online] http://www.acunetix.com/blog/docs/http-fuzzer-tool/

[9]. ISO/IEC 40500:2012. ISO.[Online] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=58625

[10]. Non-text Content.[Online] https://www.w3.org/TR/UNDERSTANDING WCAG20/text-equiv-all.html

[11]. ModSecurity Core Rule Set Project. OWASP. [Online] https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project

[12]. 2013 OWASP Top Ten Project. OWASP.[Online] http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202013.pdf

[13]. Web Application Security Accessibility Project. OWASP. [Online]https://www.owasp.org/index.php/OWASP_Web_Application_Security_Accessibility_Project

[14]. Zed Attack Proxy Project. OWASP.[Online] https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

[15]. Edward Rolando Núñez-Valdéz, Oscar Sanjuán Martínez, Gloria García Fernández, Luis Joyanes Aguilar, Juan Manuel Cueva Lovelle , "Security Guidelines for the Development of Accessible Web Applications through the implementation of intelligent systems". IJIMAI 1, pp. 79-86, 2009.

[16]. Symantec. 2016 Internet Security Threat Report.

[17]. Vũ Thị Hương Giang, Nguyễn Thị Thu Trang. "Hướng dẫn thiết kế trang web cho người khiếm thị". ISBN: 978-604-938-730-2: NXB Bách Khoa, 2015.

[18]. Ismailova, Rita, "Web site accessibility, usability and security: a survey of government websites in Kyrgyz Republic". Universal Access in the Information Society, pp. 1-8, 2015.

[19]. 2007 OWASP Top Ten Project. OWASP.[Online] 2007. https://www.owasp.org/index.php/Top_10_2007

[20]. Using the title attribute of the frame and iframe elements.W3C.[Online]. https://www.w3.org/TR/WCAG20-TECHS/H64.html

[21]. Using longdesc W3C.[Online]. https://www.w3.org/TR/WCAG20-TECHS/H45.html

[22]. Understanding SC 1.1.1 W3C.[Online] https://www.w3.org/TR/UNDERSTANDING-WCAG20/text-equiv-all.html

SƠ LƯỢC VỀ TÁC GIẢ

TS. Vũ Thị Hương Giang

Đơn vị công tác: Viện Công nghệ Thông tin và Truyền thông, ĐH Bách Khoa Hà Nội.

Email: giangvth@soict.hust.edu.vn

Nhận bằng đại học năm 2001 và nhận bằng thạc sĩ năm 2003 chuyên ngành Công nghệ thông tin tại ĐH Bách Khoa Hà Nội. Nhận bằng tiến sĩ chuyên ngành Hệ thống và phần mềm tại Đại học Bách Khoa Hà Nội năm 2008.

Hướng nghiên cứu hiện nay: các hệ thống hướng dịch vụ đảm bảo an toàn thông tin,các phương pháp phát triển phần mềm tiên tiến và ứng dụng.

KS. Phan Văn Huy

Email: phanhuy.bkhn@gmail.com

Tốt nghiệp chuyên ngành Công nghệ thông tin tại ĐH Bách Khoa Hà Nội năm 2016.

Hướng nghiên cứu hiện nay: an toàn thông tin.

KS.Vũ Văn Trung

Email: trungvu.inside@gmail.com

Tốt nghiệp chuyên ngành Công nghệ thông tin tại ĐH Bách Khoa Hà Nội năm 2016

Hướng nghiên cứu hiện nay: an toàn thông tin.