PHƯƠNG PHÁP PHÁT HIỆN VÀ CẢNH BÁO SỰ THAY ĐỔI CỦA WEBSITE

Vũ Đức Thịnh*, Trần Đắc Tốt, Vũ Văn Vinh
Trường Đại học Công nghiệp Thực phẩm TP.HCM
*Email: *thinhvd@cntp.edu.vn*
Ngày nhận bài: 10/8/2017; Ngày chấp nhận đăng: 20/9/2017

TÓM TẮT

Lỗ hổng website luôn là mục tiêu tiềm tàng của các cuộc tấn công vì các mục đích khác nhau. Tấn công thay đổi nội dung website là một trong những hình thức tấn công rất phổ biến hiện nay. Bài báo này sẽ nghiên cứu về các giải thuật chính được sử dụng để phát hiện sự thay đổi về nội dung của website, từ đó đưa ra đánh giá và đề xuất phương pháp cải tiến thuật toán Rabin Fingerprint giúp tăng cường khả năng giám sát, phát hiện và cảnh báo, nhằm hỗ trợ cho người quản trị có thể phản ứng nhanh hơn trong các trường hợp website của mình bị tấn công.

Từ khóa: Cảnh báo, tấn công, thay đổi website, thuật toán Rabin fingerprint, phát hiện thay đổi.

1. MỞ ĐẦU

Những cuộc tấn công thay đổi website đã được thực hiện để xâm phạm tính toàn vẹn của web bằng một trong những hình thức sau [1]:

- Thay đổi nội dung của trang web.
- Thay đổi bất kỳ phần nào của nội dung trang web.
- Thay thế toàn bộ trang web.
- Chuyển hướng trang web.
- Phá hủy hoặc xóa bỏ trang web.

Các hệ thống kiểm soát an ninh mạng như Firewall, VPN, PKI (Public Key Infrastructure)…, là những công cụ quan trọng để giữ cho web được an toàn hơn, tuy nhiên chúng không đủ để đảm bảo an ninh website, do đó cần những cơ chế an ninh tốt hơn [2].

Việc kiểm soát sự thay đổi về nội dung và cảnh báo ngay tới người quản trị là một trong những giải pháp để giải quyết vấn để về an ninh website. Có rất nhiều các nghiên cứu để đánh giá, so sánh giữa 2 trang web đã được đề xuất. DaisyDiff là một dự án của Google đã được công bố vào năm 2007, dự án phát triển dựa trên việc so sánh các định dạng HTML, phát hiện các nội dung đã thay đổi. Thêm vào đó DaisyDiff cũng so sánh hai văn bản để tìm ra những điểm khác biệt giữa chúng một cách nhanh chóng. HTML Match cũng là công cụ khá mạnh đã được thương mại hóa vào năm 2005 của Salty Brine có thể so sánh được 2 trang HTML, thậm chí so sánh được cả các tài liệu có định dạng MS Word. Một số lượng lớn các bài báo đã công bố nghiên cứu và tìm ra được cách cách xử lý, đưa ra được thuật toán hiệu quả để phát hiện sự thay đổi của trang web. Trong [2], [3] và [4], các thuật toán khác nhau đã được đề xuất để phát hiện sự thay đổi trong các tài liệu XML. Trong [4], thuật toán dựa trên việc tìm kiếm và rút ra những sư thay đổi dựa trên việc so sánh cấu trúc cây. Việc so khớp 2 node của cây dựa trên chữ ký (là hàm của node đang xét và con của chúng)

và thứ tự xuất hiện của node đó trong chuỗi node. Từ những node của cây không khớp nhau, sự thay đổi của tài liệu sẽ được phát hiện.

Thuật toán GNU diffutility và AT & Ts HTMLDiff [5, 7, 8] là các thuật toán phát hiện thay đổi dựa vào việc tính toán sự khác nhau giữa các thông số phẳng của tập tin. Những thuật toán này áp dụng thuật toán chuỗi chung dài nhất để so sánh hai hay nhiều tập tin phẳng (tập tin không phân cấp về nội dung như XML) hoặc trang HTML. Những chuỗi chung dài nhất này thực hiện hiệu quả trên các tập tin phẳng nhưng lại gặp những thiếu sót trong các tập tin có cấu trúc phân cấp như XML [6].

Trong bài báo này, phần 2 giới thiệu các khái niệm cơ bản về dấu vân tay tài liệu, thuật toán Rabin Fingerprint và đề xuất thuật toán Rabin Fingerprint cải tiến áp dụng xây dựng hệ thống giám sát website nhằm phát hiện kịp thời các cuộc tấn công để đảm bảo tính toàn vẹn của trang web đồng thời tạo ra thông điệp cảnh báo có ý nghĩa khi trang web đã bị tấn công. Phần 3 trình bày các kết quả thực nghiệm đạt được. Phần 4 là kết luận và hướng nghiên cứu tiếp theo.

2. GIẢI PHÁP PHÁT HIỆN VÀ CẢNH BÁO SỰ THAY ĐỔI WEBSITE

2.1. Dấu vân tay tài liệu

Trong khoa học máy tính, dấu vân tay nhận dạng duy nhất dữ liệu gốc cho tất cả các mục đích thực tiễn giống như là việc nhận dạng duy nhất dấu vân tay người trong thực tế. Dấu vân tay tài liệu là một tập hợp các số nguyên đại diện cho một số khóa nội dung của tài liệu đó. Mỗi số nguyên được gọi là một giá trị băm.

Thông thường, một dấu vân tay tài liệu được tạo ra bằng cách chọn chuỗi con từ văn bản đó và áp dụng một hàm toán học cho mỗi chuỗi con đã chọn. Hàm này, giống như một hàm băm, tạo ra một giá trị băm. Giá trị băm này sau đó được lưu trữ trong một chỉ mục (index) để truy cập nhanh khi truy vấn. Khi một tài liệu truy vấn (query document) sẽ được so sánh với tập hợp các số nguyên đã được lưu trữ đó, dấu vân tay tài liệu cho phép các truy vấn đó sẽ được tạo ra. Đối với mỗi giá trị băm trong dấu vân tay tài liệu, chỉ mục của truy vấn và một danh sách các dấu vân tay đối sánh được lấy ra. Số lượng giá trị băm chung giữa dấu vân tay truy vấn và mỗi dấu vân tay trong tập hợp đã lưu trữ xác định tài liệu tương ứng đó.

Có một vài phương pháp để lấy dấu vân tay tài liệu dựa trên 4 sự biến đổi của các thông số thiết kết sau:

- Chiến lược lựa chọn (được sử dụng để chọn các chuỗi con từ tài liệu đã cho).
- Kích thước của các chuỗi con (được trích ra từ tài liệu).
- Số lượng giá trị băm (được sử dụng để xây dựng một tài liệu dấu vân tay).
- Hàm Fingerprint (được sử dụng để tạo ra một giá trị băm từ chuỗi con trong tài liệu, như là các checksum, hàm băm, hàm băm mật mã, và chữ kí số).

2.2. Thuật toán Rabin Fingerprint

Thuật toán Rabin Fingerprint là một trong nhiều thuật toán Fingerprint thực hiện khóa công khai sử dụng các đa thức trên một trường giới hạn.

Thuật toán Rabin Fingerprint điển hình tạo ra một giá trị băm từ chuỗi con trong các trang web (web pages), bởi vì đây là một thuật toán nhanh, dễ dàng thực thi, và nó cũng đi kèm với một phân tích chính xác toán học của xác suất đụng độ (hai tập tin có dấu vân tay giống nhau).

Thuật toán được sử dụng trong hệ thống như sau:

Đầu vào: Tài liệu (trang web công khai)

Đầu ra: Dấu vân tay tài liệu (các giá trị băm của tài liệu đó)

Bước 1: Bắt đầu.

Bước 2: Xử lý văn bản, xoá hết tất cả khoảng trắng và các kí tự đặc biệt (như: <, >, %, !, …) từ mã HTML (mã trang web) để thu được một khối văn bản thuần túy (pure text block).

Bước 3: Chia khối văn bản đã xử lý đó thành các chuỗi con có độ dài K.

// Số lượng chuỗi con có độ dài K và số lượng giá trị băm (mã băm) bằng (m-K+1),với m là kích thước của tài liệu.

Bước 4: Tính toán giá trị băm đối với mỗi chuỗi con bằng cách tính H(P) như sau:

// H(P) là một tuyến tính trong n (n là độ dài của P)

Khởi tạo:

Count=K

Tr = T[r..r+n-1]

$H(S) = S(n) + 2*S(n-1) + 4*S(n-2) + \ldots + 2^{n-1}*S$ (1)

Do while Count > 0

Sử dụng $H_p(P) = H(P) \bmod p$ như là một giá trị băm (fingerprint) của P

$H_p(T_r) = [2*H_p(T_{r-1}) - (2^n \bmod p) * T(r-1) + T(r+n-1)] \bmod p$

//Tính giá trị băm cho các chuỗi con tiếp theo.

Until Count = 1

Bước 5: Lưu lại tất cả các giá trị băm của văn bản.

Bước 6: Kết thúc.

2.3. Thuật toán Rabin Fingerprint cải tiến

Thuật toán cải tiến được đề xuất trong hệ thống như sau:

Đầu vào: Tài liệu (trang web công khai)

Đầu ra: Dấu vân tay tài liệu (các giá trị băm của tài liệu đó)

Bước 1: Bắt đầu.

Bước 2: Xử lý văn bản, xoá hết tất cả khoảng trắng và các kí tự đặc biệt (như: <, >, %, !, …) từ mã HTML (mã trang web) để thu được một khối văn bản thuần túy (pure text block).

Bước 3: Chia văn bản M thành K khối con, mỗi khối con có kích thước là n. K=m/n với m là kích thước của văn bản M, n là số nguyên dương cho trước là kích thước của mỗi chuỗi con.

Bước 4: Tính mã băm H(P) cho các chuỗi con như sau:

Khởi tạo:

Tr = T[r..r+n-1];

K=0;

$H(S) = S(n) + 2*S(n-1) + 4*S(n-2) + \ldots + 2^{n-1}*S(1)$;

While (K<m/n)

{

for (r=K*n; r<=K*n+n; r++)

{

$H_p(T_r)= (H_p(T_r) + T(r))$ mod p *//Tính giá trị băm cho các chuỗi con, p là số nguyên tố lớn.*

}

K++;

}

Bước 5: Lưu lại tất cả các giá trị băm của văn bản.

Bước 6: Kết thúc.

Thuật toán Rabin Fingerprint được nhóm tác giả đề xuất cải tiến cụ thể từ bước thứ 3, thay vì chia khối văn bản thành các chuỗi con với một độ dài như nhau và sau đó thực hiện tính toán giá trị băm với mỗi chuỗi con thì nhóm tác giả chia khối văn bản thành các khối con với kích thước như nhau. Sau khi tiến hành chạy thử nghiệm 2 thuật toán thì kết quả đạt được cho thấy thuật toán Rabin Fingerprint cải tiến có thời gian tính toán nhanh hơn khá nhiều so với thuật toán Rabin Fingerprint cũ. Kết quả thực nghiệm sẽ được trình bày cụ thể trong phần 3 của bài báo.

3. KẾT QUẢ THỬ NGHIỆM

Chương trình thử nghiệm được phát triển bằng ngôn ngữ C#. Với cấu hình máy được sử dụng như sau:

- Bộ xử lý: Intel(R) Core(TM) i5 CPU M450 @ 2.40GHz
- Bộ nhớ Ram: 8.00 GB.
- Loại hệ thống: hệ điều hành 64-bit.
- Hệ điều hành: Windows 10 Professional.

Chương trình được thử nghiệm kiểm tra về thời gian xử lý của thuật toán Rabin Fingerprint và thuật toán Rabin Fingerprint cải tiến với dữ liệu vào là 3 website (sử dụng hàm stopwatch() trong C# để đo thời gian xử lý của thuật toán).

Kết quả thử nghiệm của chương trình với 3 website về thời gian tính toán của thuật toán Rabin Fingerprint và thuật toán Rabin Fingerprint cải tiến như bảng sau:

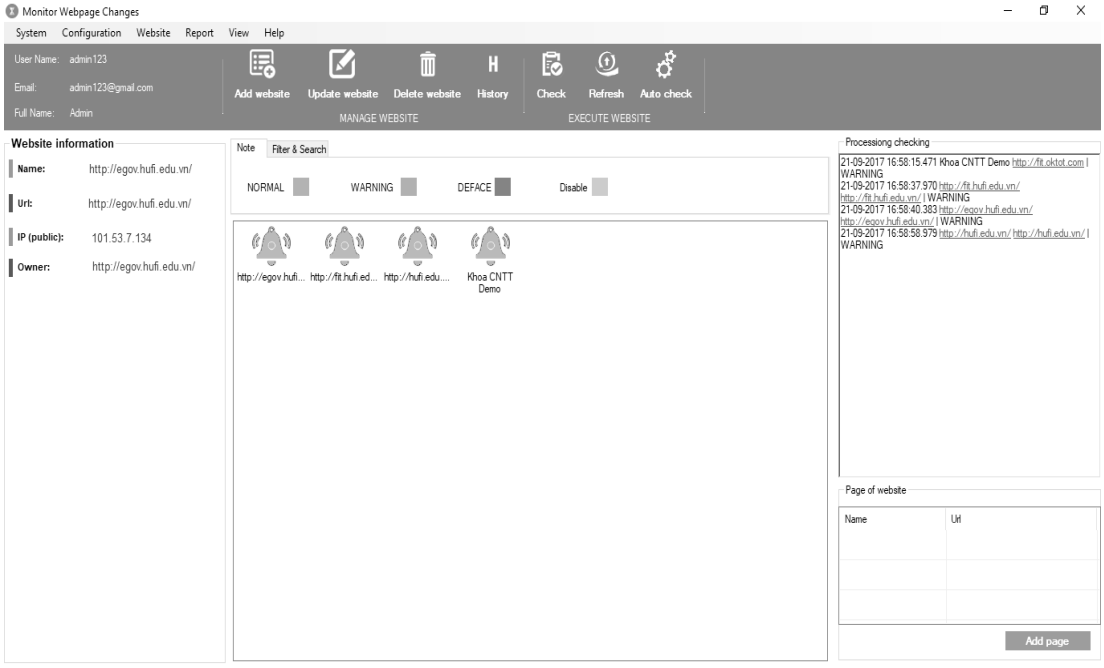
Bảng 1. Kết quả tính toán của thuật toán Rabin Fingerprint và Rabin Fingerprint cải tiến

Website	Rabin fingerprint (Thời gian)	Rabin fingerprint cải tiến (Thời gian)	Kích thước (kí tự)
fit.hufi.edu.vn	00:00:52.2659048	00:00:00.0179612	197342
hufi.edu.vn	00:00:56.6508695	00:00:00.0237493	260318
oktot.com	00:00:07.4765326	00:00:00.0032503	96819

Theo kết quả thực nghiệm ở Bảng 1 cho thấy rằng với thuật toán Rabin Fingerprint cải tiến thời gian phát hiện ra sự thay đổi nội dung của website được cải hiện đáng kể, điều này sẽ hỗ trợ rất lớn cho các quản trị viên trong việc phản ứng khi website của họ bị tấn công thay đổi nội dung.

Chương trình đã cài đặt thực nghiệm trên trang chủ của khoa Công nghệ Thông tin Trường Đại học Công nghiệp Thực phẩm thành phố Hồ Chí Minh (http://fit.hufi.edu.vn) và đã đạt được kết quả như sau:

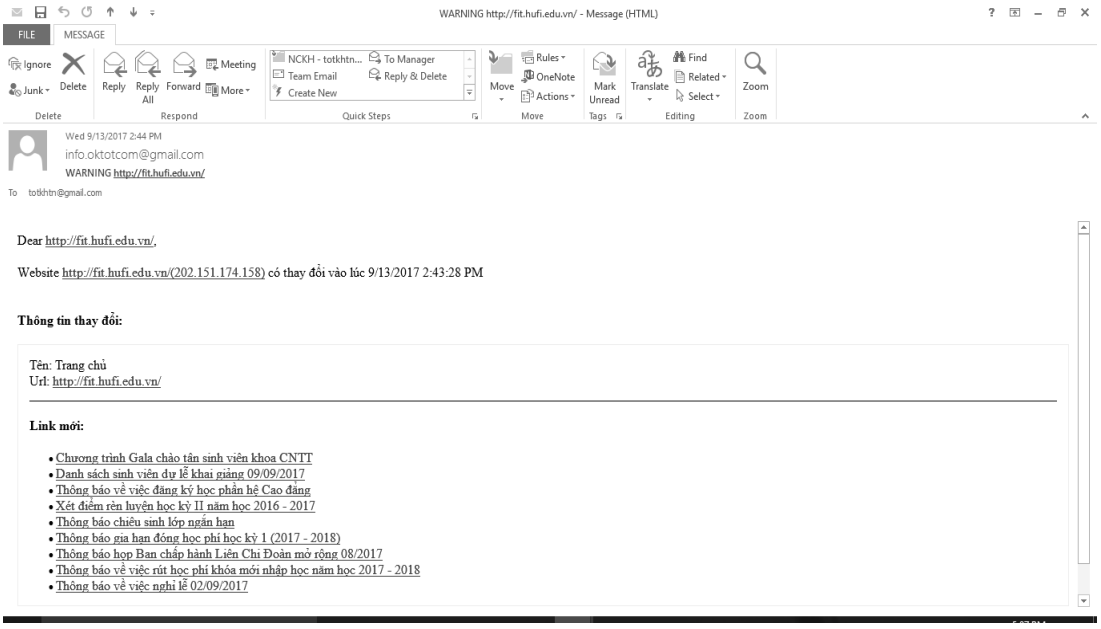
- Phát hiện được tất cả các thay đổi xảy ra của website
- Gửi cảnh báo về email cho quản trị viên mỗi khi có sự thay đổi.
- Giao diện ứng dụng khá thuận tiện
- Dễ dàng cho quản trị viên kiểm tra và phát hiện vị trí cần khắc phục khi có sự cố
- Tốc độ chương trình tương đối ổn định



Hình 1. Giao diện phần mềm

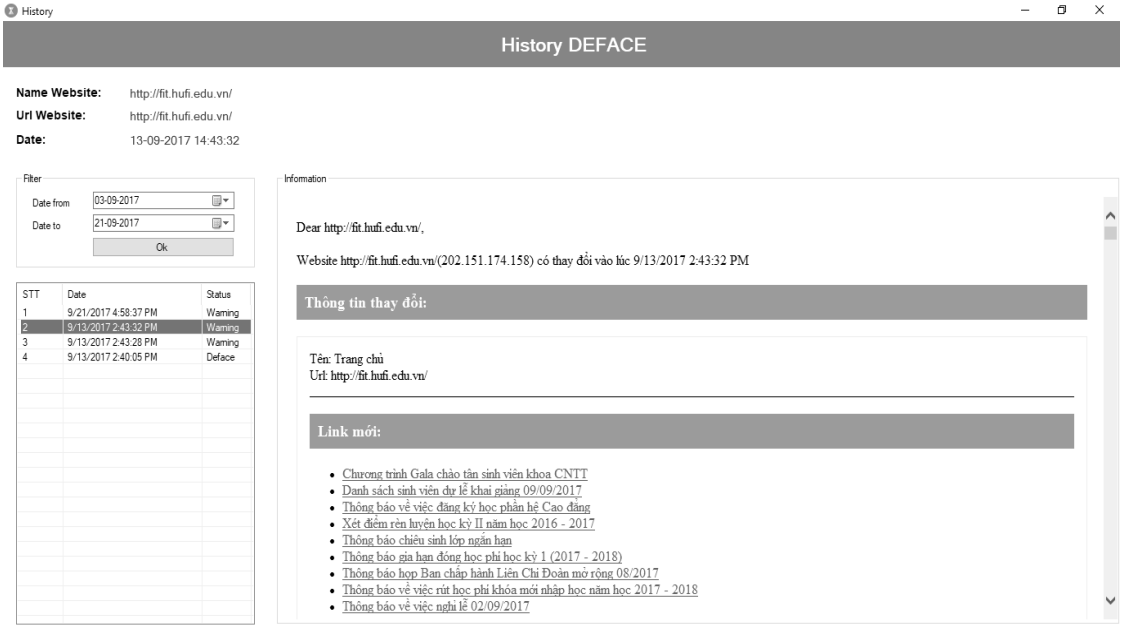
Căn cứ vào các thông tin trên Hình 1, quản trị viên có thể biết được rõ các website mình đang quản trị đang ở trạng thái an toàn hay không dựa vào màu sắc được hiển thị trong các cảnh báo về trạng thái như:

- Màu xanh có nghĩa là website đang ở trang thái an toàn;
- Màu vàng có nghĩa là trong website có xảy ra sự thay đổi về nội dung (tuy nhiên chưa hẳn là bị tấn công, nội dung bị thay đổi trong website có thể được thực hiện bởi chính quản trị viên);
- Màu đỏ có nghĩa là website đã bị tấn công;
- Màu trắng có nghĩa là phần mềm đã bị vô hiệu hóa, không còn thực hiện được chức năng theo dõi với website.



Hình 2. Email cảnh báo khi có sự thay đổi nội dung website

Hình 2 hiện thị nội dung cảnh báo của email gửi tới quản trị viên khi phần mềm phát hiện ra sự thay đổi trên website, giúp cho quản trị viên phản ứng nhanh hơn nếu có sự cố xảy ra, thông báo cho biết có sự thay đổi trên website vào lúc 2:43 PM ngày 09/13/2017 và được gửi tới quản trị viên vào lúc 2:44 PM cùng ngày, có nghĩa là chỉ khoảng chưa đầy một phút sau khi phát hiện có sự thay đổi.



Hình 3. Danh sách thông tin và thời điểm xảy ra thay đổi trên website

Hình 3 cho thấy danh sách các thời điểm xảy ra thay đổi trên website và trạng thái của website tại các thời điểm bị cảnh báo, kèm theo đó là các thông tin về sự thay đổi và danh sách các đường links mới xuất hiện trên website luôn được cập nhật thường xuyên.

4. KẾT LUẬN

Dựa trên các kết quả của thực nghiệm có thể kết luận rằng thời gian xử lý của thuật toán Rabin Fingerprint cải tiến là nhanh hơn rất nhiều so với thuật toán Rabin Fingerprint (đặc biệt khi kích thước trang web lớn). Kết quả thực nghiệm cho thấy với thuật toán Rabin Fingerprint cải tiến mang lại độ lợi về mặt thời gian, các cảnh báo được gởi đi sớm hơn, giúp người quản trị có thể phản ứng tốt hơn khi website bị tấn công thay đổi nội dung. Trong thời gian tới nhóm tác giả sẽ áp dụng thêm một số thuật toán như vector hỗ trợ vào bài toán để việc so khớp trở nên hiệu quả hơn.

TÀI LIỆU THAM KHẢO

1. Charles P. Pfleeger and Shari Lawrence. - Security in Computing, 3rd Edn, Prentice Hall, 2003. (Available at http://books.google.com/books?id=O3VB-zspJo4C&dq=%22web+site+defacement+attack+%22&source=gbs_navlinks_s).
2. Cobena G., Abiteboul S., and Marian A. - Detecting changes in XML documents, Proceedings of 18th International Conference on Data Engineering (2002) 41-52.
3. Wang Y., DeWitt D., and Cai J. - X-Diff: An effective change detection algorithm for XML documents, Proceedings of 19th International Conference on Data Engineering (2003) 519-530.
4. Jyoti J., Sachde A., and Chakravarthy S. - CX-DIFF: A change detection algorithm for XML content and change visualization for WebVigiL, Data and Knowledge Engineering **52** (2) (2005) 209-230.
5. Berk E. - HtmlDiff: A differencing tool for HTML documents", Student Project, Princeton University, http://www.htmldiff.com
6. Chawate S., Rajaraman A., Garcia-Molina H. and Widom J. - Change detection in hierarchical structured information", Proceedings of the ACM SIGMOD International Conference on Management of Data, Monteal, June 1996.
7. Douglis F., Ball T. - Tracking and viewing changes on the Web, USENIX Annual Technical Conference, 1996.
8. Douglis F., Ball T., Chen Y. F., Koutsofios E. - The AT&T internet difference engine: Tracking and viewing changes on the Web", World Wide Web **1** (1) (1998) 27-44.

ABSTRACT

METHOD OF DETECTING AND WARNING CHANGED WEBSITE

Vu Duc Thinh*, Tran Đac Tot, Vu Van Vinh
Ho Chi Minh City University of Food Industry
*Email: *thinhvd@cntp.edu.vn*

Website vulnerabilities are always a potential target of attacks for different purposes. Hackers always have tools to find vulnerabilities in the Web, and then they will exploit those vulnerabilities. This paper studies the main algorithms used to detect changes in the content of the website, then evaluates and proposes an improved Rabin Fingerprint algorithm, enhances the ability of monitoring, detecting and warning, to support the administrator in responding more quickly in case their websites were attacked.

Key words: Attacks, change detection, Rabin fingerprint algorithm, warnings, website changes.