MỤC LỤC

LỜI NÓI ĐẦU

Cùng với sự phát triển của công nghệ thông tin, công nghệ mạng máy tính và sự phát triển của mạng internet ngày càng phát triển đa dạng và phong phú. Các dịch vụ trên mạng đã thâm nhập vào hầu hết các lĩnh vực trong đời sống xã hội. Các thông tin trên Internet cũng đa dạng về nội dung và hình thức, trong đó có rất nhiều thông tin cần được bảo mật cao hơn bởi tính kinh tế, tính chính xác và tính tin cậy của nó.

Bên cạnh đó, các hình thức phá hoại mạng cũng trở nên tinh vi và phức tạp hơn. Rất nhiều các cuộc tấn công với website đã được thực hiện trong suốt những năm gần đây. Xuất phát từ những thực tế đó, chúng ta sẽ: ***"Tìm hiểu tấn công https bằng cache injection"***. Đây là tấn công đã được xếp hạng 4 trong bảng xếp hạng 10 kĩ thuật hack hay nhất trong năm 2010 được công bố bởi tổ chức bảo mật Whitehat Security. Với đề tài này, chúng ta sẽ được tiếp cận với một kỹ thuật tấn công khá mới và rất hấp dẫn này. Đồng thời cũng có thể thấy được mức độ nguy hiểm của tấn công đối với các website.

Do thời gian có hạn và cũng là một đề tài chưa được nghiên cứu phổ biến, nên hiểu biết của nhóm về tấn công còn hạn chế và bài báo cáo còn nhiều thiếu sót. Vì vậy, chúng em rất mong cô giáo và các bạn đưa ra những nhận xét, đóng góp ý kiến để đề tài trên được hoàn thiện hơn nữa.

Chúng tôi xin chân thành cảm ơn sự hướng dẫn của là cô giáo trực tiếp hướng cho chúng tôi, giúp chúng tôi có thể hoàn thành bài báo cáo này.

DANH MỤC VIẾT TẮT

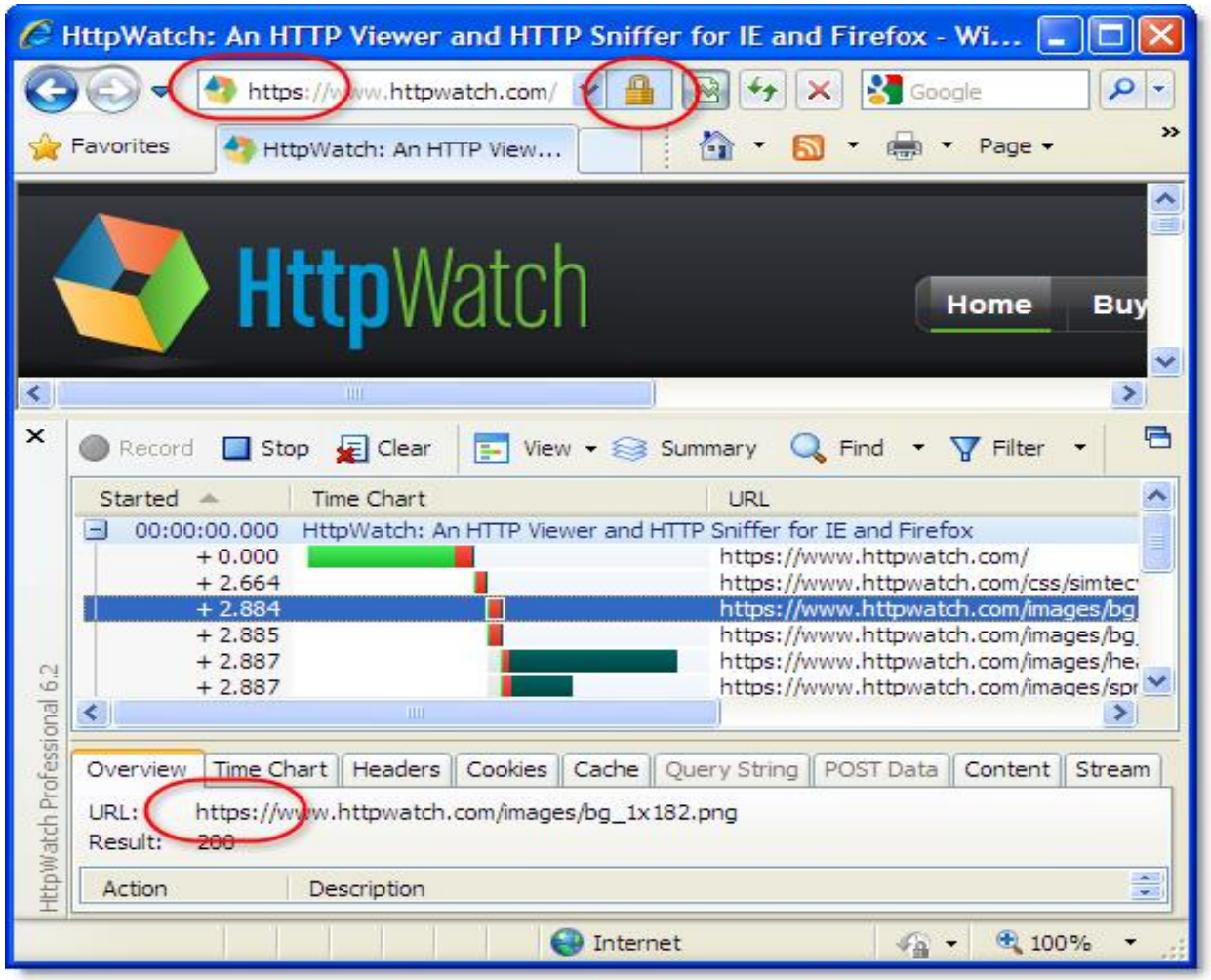
Từ viết tắt	Từ đầy đủ
HTTPS	Hypertext Transfer Protocol Secure
SSL	Secure Socket Layer
TLS	Transport Layer Security
CA	Certificate Authority

DANH MỤC HÌNH ẢNH

PHẦN 1: TỔNG QUAN VỀ HTTPS

1.1. Định nghĩa https.

HTTPS là viết tắt của "HyperText Transfer Protocol Secure", Nó là một sự kết hợp giữa giao thức HTTP và giao thức bảo mật SSL hay TLS cho phép trao đổi thông tin một cách bảo mật trên Internet. Giao thức HTTPS thường được dùng trong các giao dịch nhạy cảm cần tính bảo mật cao.



Hình 1.1. Trang web sử dụng https.

HTTPS là sự kết hợp giữa giao thức HTTP và giao thức bảo mật SSL. Vậy giao thức SSL là gì và có tác dụng gì?

SSL là viết tắt của từ Secure Sockets Layer. Đây là một tiêu chuẩn an ninh công nghệ toàn cầu tạo ra một liên kết được mã hóa giữa máy chủ web và trình duyệt. Liên kết này đảm bảo tất cả các dữ liệu trao đổi giữa máy chủ web và trình duyệt luôn được bảo mật và an toàn. SSL đảm bảo rằng tất cả các dữ liệu được truyền giữa các máy chủ web và các trình duyệt được mang tính riêng tư, tách rời. SSL là một chuẩn công nghiệp được sử dụng bởi hàng triệu trang web trong việc bảo vệ các giao dịch trực tuyến với khách hàng của họ.

Giao thức HTTPS sử dụng port 443, và cung cấp các dịch vụ hay đảm bảo tính chất sau của thông tin:

- Confidentiality: sử dụng phương thức encryption để đảm bảo rằng các thông điệp được trao đổi giữa client và server không bị kẻ khác đọc được.
- Integrity: sử dụng phương thức hashing để cả client và server đều có thể tin tưởng rằng thông điệp mà chúng nhận được có không bị mất mát hay chỉnh sửa.
- Authenticity: sử dụng digital certificate để giúp client có thể tin tưởng rằng server/website mà họ đang truy cập thực sự là server/website mà họ mong muốn vào, chứ không phải bị giả mạo.

1.2. Sử dụng HTTPS như thế nào?

Trước hết, muốn áp dụng HTTPS thì trong quá trình cấu hình Webserver, bạn có thể dễ dàng tự tạo ra một SSL certificate dành riêng cho website của mình và nó được gọi là self-signed SSL certificate.

Hình 1.2. Hoạt động của SSL certificate.

SSL certificate tự cấp này vẫn mang lại tính Confidentiality và Integrity cho quá trình truyền thông giữa server và client. Nhưng rõ ràng là không đạt được tính Authenticity bởi vì không có bên thứ 3 đáng tin cậy nào (hay CA) đứng ra kiểm chứng sự tính xác thực của certificate tự gán này. Điều này cũng giống như việc một người tự làm chứng minh nhân dân cho mình rồi tự họ ký tên, đóng dấu luôn vậy!

Vì vậy, đối với các website quan trọng như E-Commerce, Online Payment, Web Mail,… thì họ sẽ mua một SSL certificate từ một Trusted Root CA nổi tiếng như VeriSign, Thawte, và ít tên tuổi hơn thì có GoDaddy, DynDNS… Các CA có 2 nhiệm vụ chính sau:

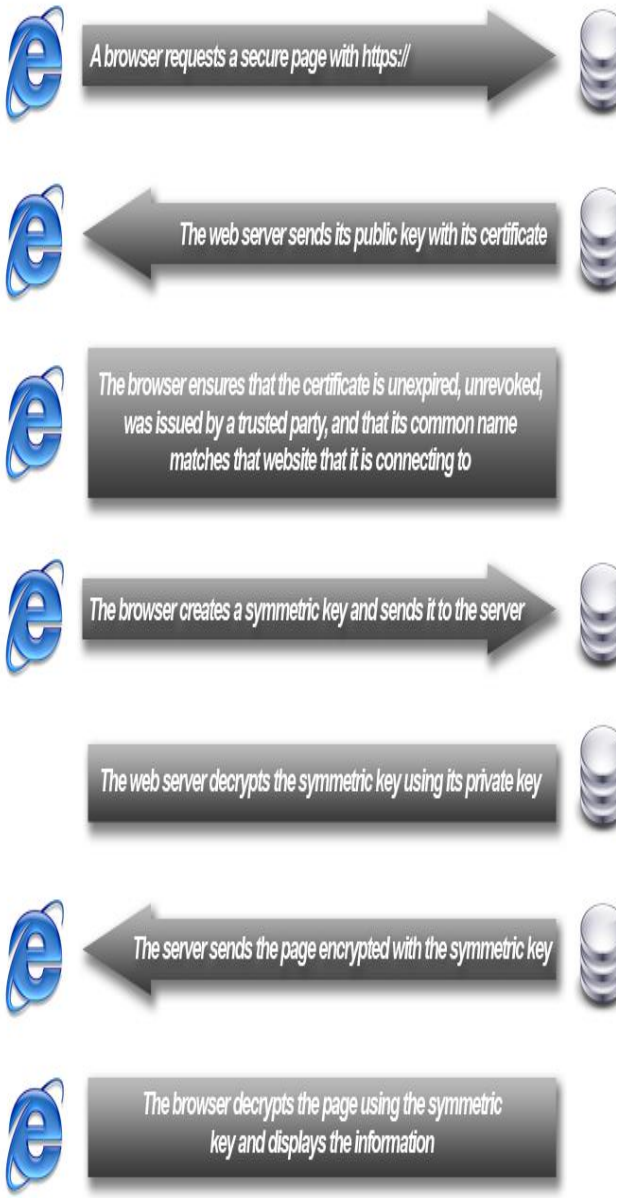
- Cấp phát và quản lý SSL certificate cho server/website.
- Xác thực sự tồn tại và tính hiệu lực của SSL certificate mà Web client gửi tới cho nó.

Thực chất thì SSL certificate cũng là một loại digital certificate (một loại file trên máy tính). Vì giao thức HTTPS có dính tới giao thức SSL nên người ta mới đặt tên cho nó là SSL certificate để phân biệt với các loại digital certificate khác như Personal Certificate, Server Certificate, Software Publisher Certificate, Certificate Authority Certificate.

Dưới đây là một số thông tin quan trọng được chứa trong SSL certificate mà bất cứ client nào cũng có thể xem được bằng cách click vào biểu tượng padlock trên thanh Address của Web browser:

- Thông tin về owner của certificate (như tên tổ chức, tên cá nhân hoặc domain của website…).
- Tên và digital signature của CA cấp certificate.
- Khoảng thời gian mà certificate còn hiệu lực sử dụng.
- Public key của server/website. Còn private key được lưu trữ trên chính server (không có trong certificate) và tuyệt đối không thể để lộ cho bất cứ client nào.
- Một số thông tin phụ khác như: loại SSL certificate, các thuật toán dùng để encryption và hashing, chiều dài (tính bằng bit) của key, cơ chế trao đổi key (như RSA, DSA…)…..

1.3. Quá trình giao tiếp giữa client và server thông qua HTTPS.



Hình 1.3. Giao tiếp giữa client và server thông qua https.

1. Client gửi request cho một secure page (có URL bắt đầu với https://)

2. Server gửi lại cho client certificate của nó.

3. Client gửi certificate này tới CA (mà được ghi trong certificate) để kiểm chứng. Giả sử certificate đã được xác thực và còn hạn sử dụng hoặc client vẫn cố tình truy cập mặc dù Web browser đã cảnh báo rằng không thể tin cậy được certificate này (do là dạng self-signed SSL certificate hoặc certificate hết hiệu lực, thông tin trong certificate không đúng…) thì mới xảy ra bước 4 sau.

4. Client tự tạo ra ngẫu nhiên một symmetric encryption key, rồi sử dụng public key (trong certificate) để mã hóa symmetric key này và gửi về cho server.

5. Server sử dụng private key (tương ứng với public key trong certificate ở trên) để giải mã ra symmetric key ở trên.

6. Sau đó, cả server và client đều sử dụng symmetric key đó để mã hóa/giải mã các thông điệp trong suốt phiên truyền thông.

Và tất nhiên, các symmetric key sẽ được tạo ra ngẫu nhiên và có thể khác nhau trong mỗi phiên làm việc với server. Ngoài encryption thì cơ chế hashing sẽ được sử dụng để đảm bảo tính Integrity cho các thông điệp được trao đổi.

PHẦN 2: TỔNG QUAN VỀ CACHE INJECTION.

2.1. Định nghĩa cache injection.

Cache là bộ nhớ đệm – nơi lưu trữ các dữ liệu nằm chờ các ứng dụng hay phần cứng xử lý. Mục đích của nó là để tăng tốc độ xử lý. Nói một cách chính xác hơn cache là một cơ chế lưu trữ tốc độ cao đặc biệt. Nó có thể là một vùng lưu trữ của bộ nhớ chính hay một thiết bị lưu trữ tốc độ cao độc lập. Có hai dạng lưu trữ cache được dùng phổ biến trong máy tính cá nhân là memory caching (bộ nhớ cache hay bộ nhớ truy xuất nhanh) và disk caching (bộ nhớ đệm đĩa).

Sử dụng cache injection, dữ liệu mạng gửi đến được đặt trực tiếp vào bộ nhớ cache của bộ xử lý. Khi có một yêu cầu xử lý dữ liệu, không cần phải gọi nó khỏi bộ nhớ vì nó đã có trong bộ nhớ cache. Mục tiêu của việc này là để chứng minh rằng các chính sách hệ điều hành khác nhau liên quan đến cache injection, có thể làm giảm việc sử dụng băng thông bộ nhớ và cải thiện hiệu suất ứng dụng trên các hệ thống hiệu năng cao.

Các kiến trúc hiện tại đặt dữ liệu mạng gửi đến vào bộ nhớ chính để cho một bộ xử lý tìm nạp dữ liệu vào bộ nhớ cache của nó. Tìm nạp dữ liệu vào một bộ nhớ cache thường được thực hiện bằng cách nạp trước trong đó dự đoán các truy cập đến các khối bộ nhớ dựa trên các mẫu sử dụng. cache Injection cung cấp một tiếp cận thay thế bằng cách đặt dữ liệu mạng gửi đến trực tiếp vào bộ nhớ cache của bộ xử lý từ I/O bus. Kỹ thuật này làm giảm băng thông bộ nhớ bằng cách loại bỏ lấy dữ liệu từ bộ nhớ chính. Một trong những thách thức chính của Cache injection là để quyết định khi nào và ở đâu để inject dữ liệu. Nếu ứng dụng không sử dụng các dữ liệu kịp thời, Cache injection có thể gây đầu độc bằng cách thiết lập công việc của ứng dụng của bộ nhớ cache. Vì vậy, chính sách injection là phụ thuộc vào mô hình sử dụng của ứng dụng. Hơn nữa, những lợi ích thực hiện các kỹ thuật này dựa trên một chính sách injection tốt.

Đặc điểm của cache injection:

- Cache injection làm giảm sử dụng băng thông mạng(có thể lên đến 96%).
- Cache injection nhanh hơn so với cache nạp trước
- Cache injection có thể có hại nếu không có một chính sách inject tốt
- Cache injection cải thiện hiệu suất ứng dụng sử dụng hệ điều hành và trình biên dịch thông tin.

2.2. Phân loại cache injection.

2.2.1. Cache injection using Speculation.

Một phương pháp, hệ thống, sản phẩm và chương trình máy tính được cung cấp cho việc chèn vào cache sử dụng suy đoán. Phương pháp này bao gồm việc tạo ra một bảng gián tiếp các dòng bộ nhớ cache ở trung tâm vào ra I/O, trong đó bao gồm các lĩnh vực, các mục địa chỉ, bộ xứ lí ID, các loại bộ nhớ cache và bao gồm cache giới hạn đường truyền. Phương pháp này cũng bao gồm các thiết lập giới hạn dòng bộ nhớ cache để xác định các CLL và nhận được một luồng các địa chỉ lân cận trong bảng. Đối với mỗi địa chỉ trong 1 luồng, phương pháp này bao gồm: tra cứu các địa chỉ trong bảng; nếu như địa chỉ có trong bảng thì chèn và cache các dòng địa chỉ tương ứng với các địa chỉ phức tạp trong bộ vi xử lí; nếu địa chỉ không có mặt trong bảng, tìm kiếm giá trị từ bộ nhớ cache mức thấp nhất đến bộ nhớ cache mức cao nhất; và chèn địa chỉ không có trong bảng để hệ thống phân cấp bộ nhớ cache của bộ xử lí chèn vào cuối các dòng địa chỉ liền kề nhau

2.2.2. Cache injection using Clustering.

Một phương pháp chèn vào cache sử dụng clustering, bao gồm: một giao dịch vào ra (I/O) tại một hệ thống vào ra chứa ít nhất một trong những hệ thống chipset và một IO trung tâm. Một giao dịch tại I/O bao gồm địa chỉ; tìm kiếm các địa chỉ trong một bảng tra cứu gián tiếp các địa chỉ trong khối bộ nhớ cache, bảng bộ nhớ cache gián tiếp bao gồm các lĩnh vực các lĩnh vực và mục địa chỉ hoặc dãy địa chỉ và định danh cluster, admin đáp ứng một yêu cầu từ việc tra cứu, nhiều hoạt động chèn vào các đơn vị xử lí được xác định bởi ID cluster.

2.2.3. Cache injection using semi-synchronous memory copy operation.

Một hệ thống, phương pháp và một máy tính có thể đọc được để chèn dữ liệu vào bộ nhớ cache sử dụng bản sao thao tác bán đồng bộ bộ nhớ được tiết lộ. Phương pháp này bao gồm việc xác định một sự khởi đầu của một bản sao thao tác bán đồng bộ bộ nhớ. Bản sao thao tác bán đồng bộ bộ nhớ là kiểm tra một giá trị nhất định trong một bit chèn vào bộ nhớ cache. Để đáp ứng với các giá trị được đưa ra trong các bit chèn vào bộ nhớ cache, một số được xác định trong dòng dữ liệu đích được sao chép vào ít nhất một cấp của bộ nhớ cache

2.3. Các kỹ thuật cache injection

2.3.1. Kỹ thuật Cache Injection trong một hệ thống xử lý.

Đây là kỹ thuật thực hiện cache injection bao gồm địa chỉ giám sát trên một bus. Quyền sở hữu của đầu vào/đầu ra dữ liệu trên bus thu được bởi cache khi một

địa chỉ trên bus (được kết hợp với dữ liệu đầu vào/đầu ra) tương ứng với một địa chỉ của một khối dữ liệu được lưu trữ trong bộ nhớ cache.

2.3.1.1. Kỹ thuật Cache Injection trong một hệ thống xử lý sử dụng một Cache Injection Instruction.

Đây là kỹ thuật thực hiện cache injection bao gồm địa chỉ giám sát trên một bus để đáp ứng với một hướng dẫn cache injection. Quyền sở hữu của đầu vào/đầu ra dữ liệu trên bus thu được bởi cache khi một địa chỉ trên bus (được kết hợp với dữ liệu đầu vào/đầu ra) tương ứng với một địa chỉ của một khối dữ liệu liên quan đến việc hướng dẫn cache injection.

2.3.1.2. Kỹ thuật Cache Injection trong một hệ thống xử lý đáp ứng một hướng dẫn trình tự cụ thể.

Một kỹ thuật để thực hiện cache injection bao gồm theo dõi một luồng hướng dẫn cho một trình tự hướng dẫn cụ thể. Địa chỉ trên một bus sau đó được theo dõi, bởi cache, để đáp ứng với phát hiện trình tự hướng dẫn cụ thể một số lần xác định. Quyền sở hữu của đầu vào/đầu ra dữ liệu trên bus sau đó thu được bởi bộ nhớ cache khi một địa chỉ trên bus (được kết hợp với dữ liệu đầu vào/đầu ra) tương ứng với một địa chỉ của một khối dữ liệu được lưu trữ trong bộ nhớ cache.

2.3.2. Kỹ thuật Cache Injection trong một hệ thống xử lý dựa trên một trạng thái chia sẻ.

Một kỹ thuật thực hiện cache injection bao gồm giám sát, tại giao diện host, phản ứng với nghe lén một địa chỉ trên một bus. Khi phản ứng nghe lén cho thấy một khối dữ liệu liên quan đến địa chỉ ở trong trạng thái chia sẻ, đầu vào/đầu ra dữ liệu liên quan đến địa chỉ trên bus hướng đến bởi cache trong đó bao gồm các khối dữ liệu trong trạng thái chia sẻ và vị trí địa lý gần với giao diện host hơn một hoặc nhiều lưu trữ khác bao gồm các khối dữ liệu liên quan đến các địa chỉ trong trạng thái chia sẻ.

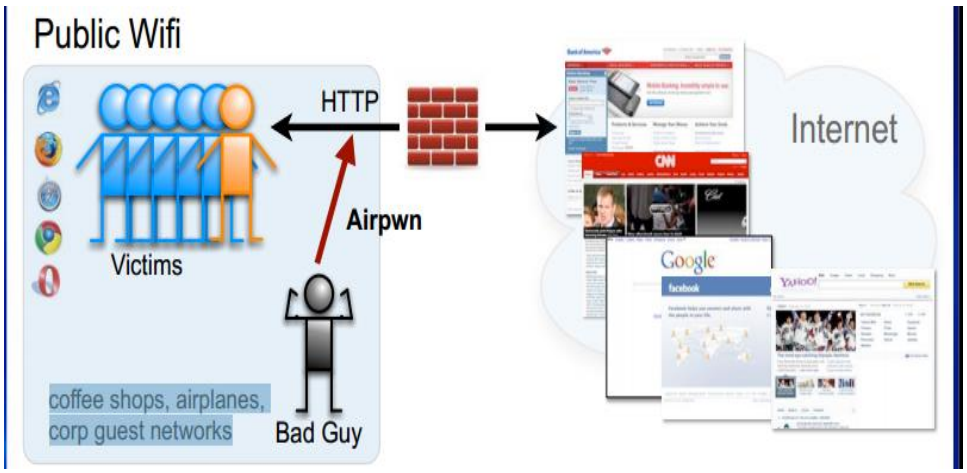
2.3.3. Kỹ thuật Cache Injection trong một hệ thống xử lý từ một nút từ xa.

Một kỹ thuật thực hiện cache injection trong một hệ thống xử lý bao gồm giám sát, bởi cache, địa chỉ trên một bus. Đầu vào/đầu ra dữ liệu liên kết với một địa chỉ của một khối dữ liệu được lưu trữ trong bộ nhớ cache sau đó yêu cầu từ một nút từ xa, thông qua một bộ điều khiển mạng. Quyền sở hữu của các dữ liệu vào/ra được thu được bởi cache khi một địa chỉ trên bus được kết hợp với dữ liệu đầu vào/đầu ra tương ứng với địa chỉ của khối dữ liệu được lưu trữ trong cache.

PHẦN 3: TẤN CÔNG HTTPS BẰNG CACHE INJECTION

3.1. Ý tưởng

Ý tưởng là lợi dụng các library javascript thông dụng thường được chèn vào các trang web (ví dụ jquery, google analytic, etc.). Giả sử bạn kết nối đến 1 trang web thông thường (không có HTTPS), trang web này có sử dụng thư viện javascript google analytic. Lúc này attacker đóng vai trò man in the middle sẽ trả lại cho bạn thư viện javascript trên được chèn thêm code của attacker. Browser sẽ lưu lại thư viện này trong cache.



Hình 3.1. Mô tả ý tưởng tấn công https bằng cache injection.

Khi bạn kết nối đến 1 trang web HTTPS có sử dụng cùng thư viện javascript trên, browser sẽ gọi luôn thư viện này từ cache, chứ không kết nối đến server để lấy về nữa, vì nó tin tưởng nội dung trong cache. Như vậy, đoạn code javascript của attacker sẽ được load vào trang web, mặc dù nó được bảo vệ bởi HTTPS, i.e. mặc dù attacker không thể tấn công được đường truyền HTTPS, nhưng vẫn có thể chèn đoạn code của mình vào trang web một cách hợp lệ.

Tiếp đó khi đã có thể thực thi được javascript trong trang web, attacker có thể thực hiện các tấn công lên trang web như đánh cắp user/pass,…

3.2. Định nghĩa

Tấn công HTTPS bằng cache injection: "tiêm" mã độc vào thư viện Javascript nằm trong cache của trình duyệt, do đó hacker có thể "phá" trang web dù được bảo vệ bởi SSL, và khiến cache bị xóa sạch. Gần một nửa trong 1 triệu trang web hàng đầu sử dụng thư viện mở rộng của Javascript. (Người tạo: Elie Bursztein, Baptiste Gourdin và Dan Boneh).

3.3. Cache injection hoạt động như thế nào?

Một cache injection làm việc như sau: Trong bước đầu tiên người dùng kết nối đến một địa điểm không an toàn như một quán cà phê. Sau đó, attacker thực hiện các tấn công man in the middle đối với người sử dụng để có thể inject một mã độc bên ngoài javascript vào trong một trang web được xem bởi người sử dụng. Thư viện javascript độc hại được phục vụ với các HTTP header đúng để đảm bảo rằng nó sẽ được lưu trữ bởi trình duyệt nạn nhân. Một khi thư viện được lưu trữ thành công, tất cả các kết nối HTTPS đến một trang bao gồm các thư viện mục tiêu sẽ bị tổn hại đến khi nạn nhân xóa bộ nhớ cache trình duyệt của mình. Thư viện sẽ được lấy từ bộ nhớ cache có chứa phiên bản độc hại đã bị inject bởi những kẻ tấn công. Hơn nữa kết nối với một trang bao gồm các thư viện javascript đã bị inject, sẽ không bao giờ gây ra một cảnh báo trình duyệt vì là trình duyệt tin tưởng một cách mù quáng nội dung bộ nhớ cache của nó.

3.4. Tại sao cache injection là nguy hiểm.

Inject một thư viện độc hại là một cuộc tấn công rất hiệu quả, vì ba lý do sau:

- Đầu tiên cache injection tấn công các hoạt động trên mọi trình duyệt, bộ nhớ đệm như là một trong những cơ chế chính được sử dụng để hiển thị các trang web nhanh hơn.
- Thứ hai nó có thể được sử dụng để nhắm đến phần lớn các trang web. Trong thu thập của Alexa trên 10000 trang web, chúng tôi thấy rằng trong số những người sử dụng SSL, ít nhất 60% trong số đó bao gồm ít nhất một thư viện javascript bên ngoài trang chính của họ, làm cho họ dễ bị tấn công bộ nhớ đệm.
- Cuối cùng kẻ tấn công có thể tận dụng thực tế là nhiều trang web sử dụng thư viện javascript bên ngoài như Google Analytics và jQuery để nhắm đến nhiều trang web cùng một lúc.

3.5. Khai thác trình duyệt UI không thống nhất.

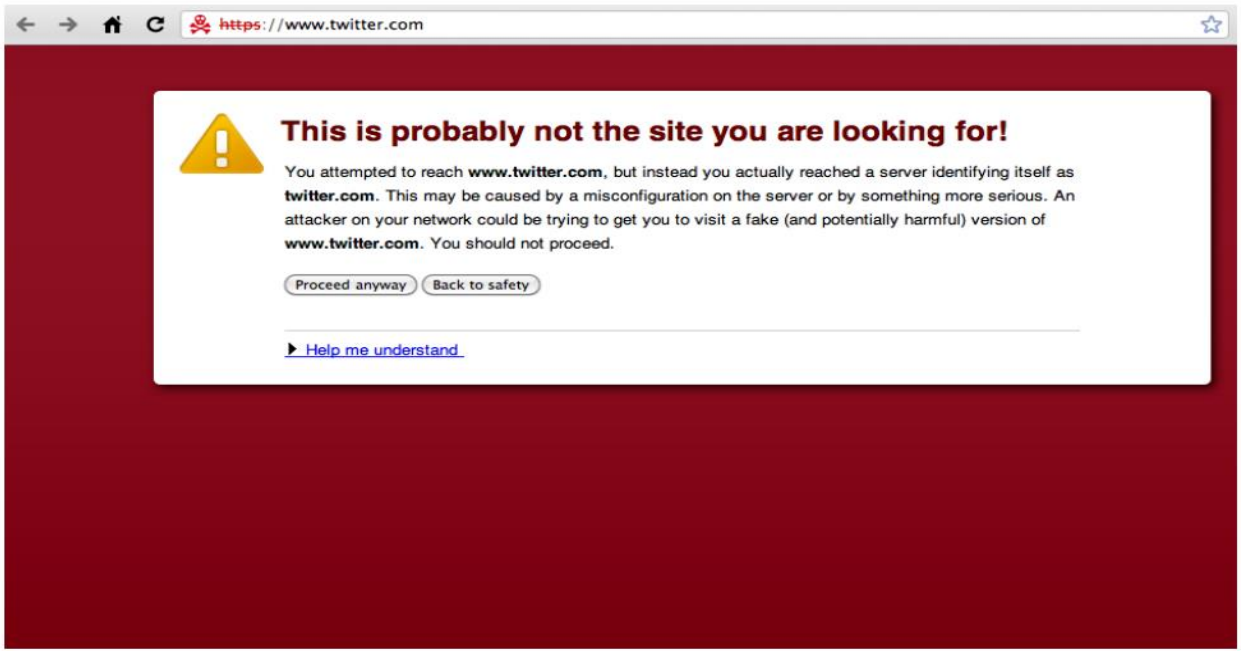
Bảo vệ duy nhất chống lại các cuộc tấn công injection là cache cảnh báo hiển thị bởi trình duyệt khi một chứng chỉ SSL là không hợp lệ (xem *hình 3.2*). Tuy nhiên đây không phải là một trở ngại lớn cho những kẻ tấn công vì hai lý do sau:

- Thứ nhất là trọng tâm trong các số liệu gần đây được thực hiện bởi Comodo cho thấy rằng khoảng 50% các trang web có chứng chỉ không hợp lệ, người sử dụng được sử dụng để bấm vào thông qua các cảnh

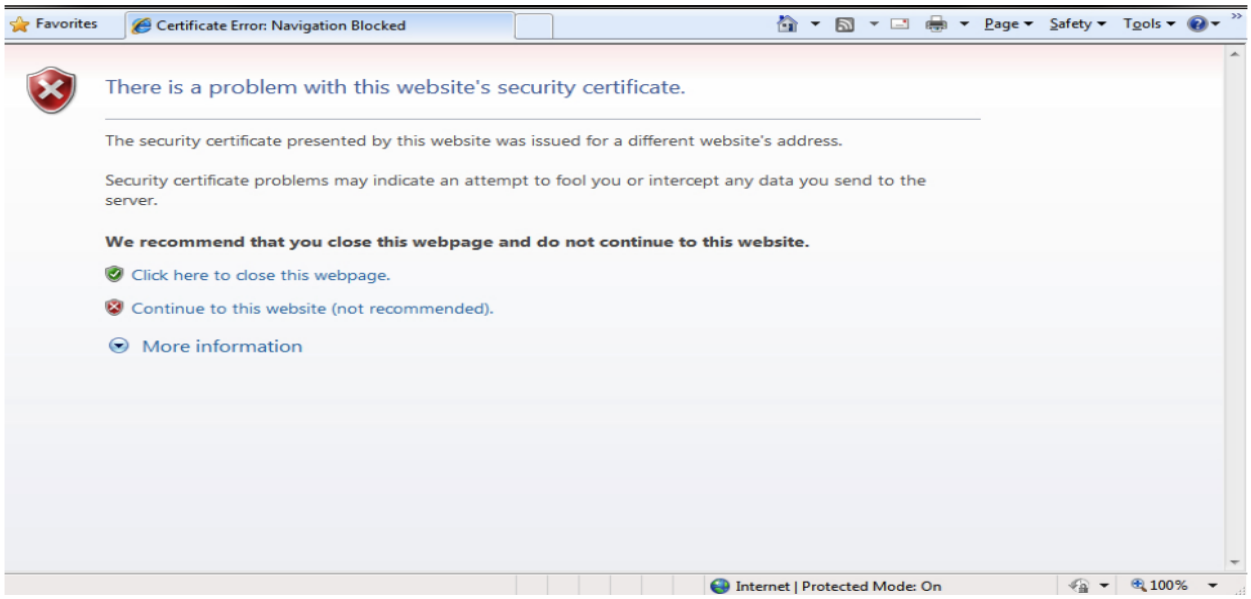
báo này. Thậm chí các trang web phổ biến như https://www.twitter.com và https://www.youtube.com cũng hiển thị một cảnh báo vì không phù hợp cname.

- Thứ hai và thậm chí nguy hiểm hơn, chúng ta có thể tạo ra những tình huống mà các cảnh báo SSL tiêu chuẩn đã không được hiển thị đúng.

Internet Explorer

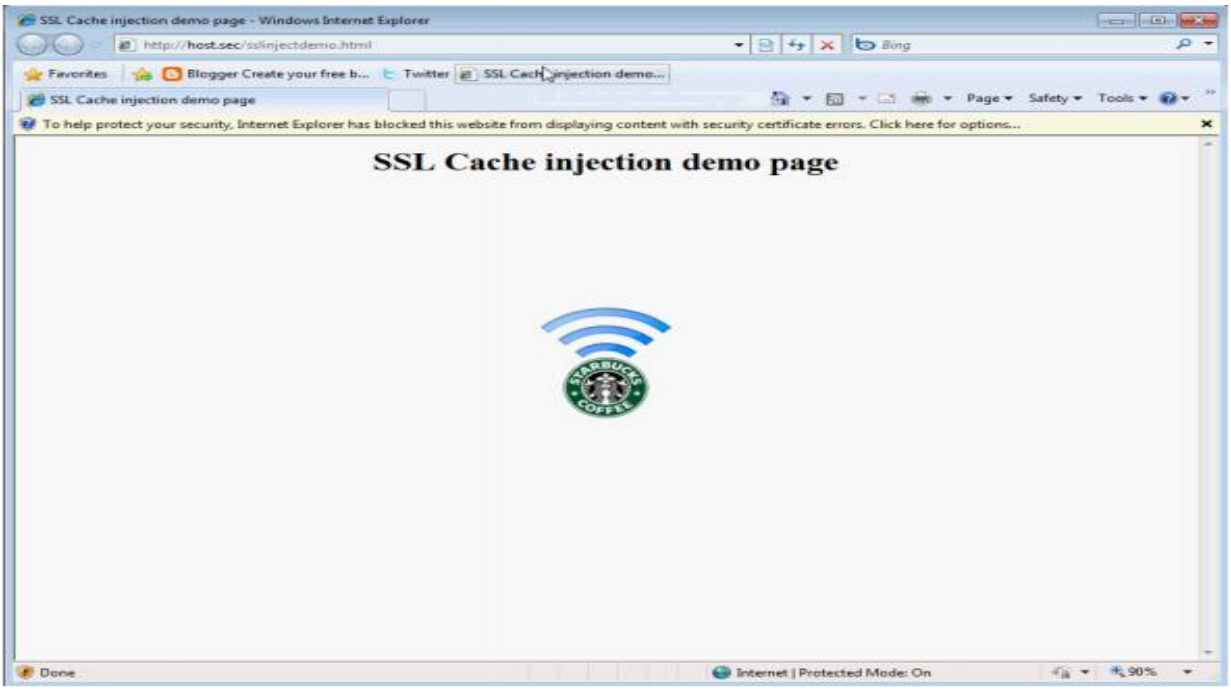


Hình 3.2. Chrome SSL warning.



Hình 3.3. Internet Explorer 8 standard SSL warning

Hình 3.3 cho thấy Internet Explorer 8 tiêu chuẩn cảnh báo SSL. Hình 3.4 cho thấy những gì xảy ra khi chứng nhận xấu được nhúng vào iframe ẩn. Như có thể thấy trong trường hợp này người dùng chỉ nhìn thấy một thanh cảnh báo màu vàng nhỏ ở đầu mà không cung cấp thông tin về các nội dung đã bị chặn. Có thể nhìn thấy loại cảnh báo này trong khi truy cập vào trang khởi động của một hotspot sẽ không gây ra sự nghi ngờ của người dùng. Vấn đề tồi tệ nhất là chấp nhận một lần để hiển thị nội dung bị chặn sẽ áp dụng trong thực tế cache nhiều thư viện javascript từ nguồn gốc khác nhau cùng một lúc. Điều này cho phép kẻ tấn công đầu độc tất cả các thư viện chia sẻ phổ biến nhất cùng một lúc mà làm cho Cache injection là tấn công thậm chí còn nguy hiểm hơn khi người dùng đang sử dụng Internet Explorer.

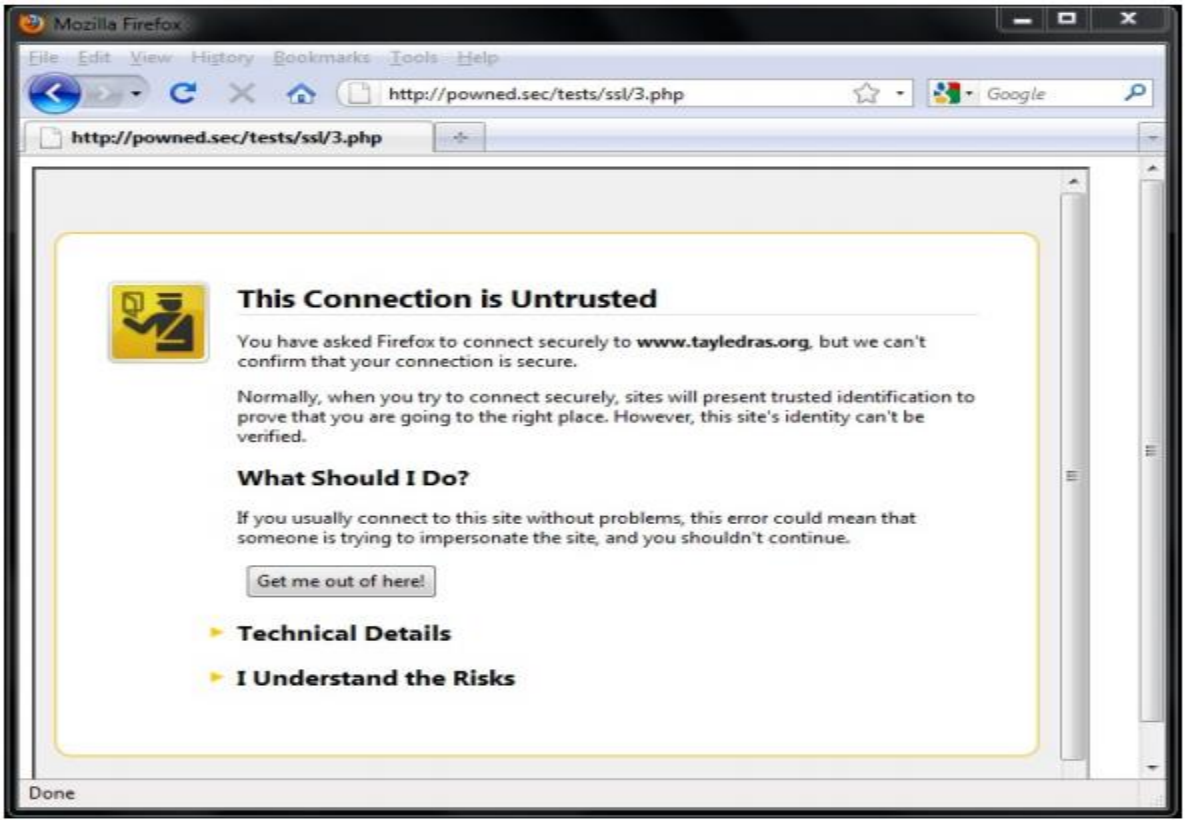


Hình 3.4: Internet Explorer 8 corner case.

Firefox.

Hình 3.5 hiển thị cảnh báo SSL tiêu chuẩn cho Firefox 3.6 là gì. Trong khi chúng tôi đã không thể tìm thấy một trường hợp cảnh báo không được hiển thị đúng, vẫn còn là một trường hợp thú vị mà những kẻ tấn công có thể khai thác. Như có thể nhìn thấy trong hình 3.6 khi một trang với một chứng nhận xấu được nạp vào iframe, cảnh báo sẽ được hiển thị bên trong iframe là tốt. Kể từ khi cảnh báo không có bảo vệ clickjacking, kẻ tấn công có thể sử dụng một cuộc tấn công clickjacking để giảm các cảnh báo bằng chạm ba lần nhấp chuột vào một.

Hình 3.5. Firefox standard SSL warning



Hình 3.6. Firefox standard SSL warning Framed

Nạn nhân sẽ nhìn thấy cửa sổ popup cuối cùng chỉ yêu cầu để chấp nhận một chứng nhận mà không có cảnh báo nào (hình 3.7).

Hình 3.7. Firefox Remaining popup after the clickjacking attack warning

PHẦN 4: DEMO TẤN CÔNG HTTPS BẰNG CACHE INJECTION

KẾT QUẢ ĐẠT ĐƯỢC VÀ HƯỚNG PHÁT TRIỂN

A. Kết quả đạt được.

Tất cả các trình duyệt phổ biến đều dùng bộ nhớ đệm để hiển thị trang web nhanh hơn. Chúng tôi đã chứng minh làm thế nào một kẻ tấn công có thể tiêm một thư viện độc hại có khả năng ảnh hưởng đến phiên SSL tiếp theo bằng cách tận dụng thực tế là các trang web tin cậy thư viện javascript bên ngoài, chẳng hạn như Google Analytics. Sau đó chúng tôi đã mô tả cách dễ dàng đánh lừa người sử dụng phải chấp nhận thư viện javascript độc hại này bằng cách khai thác trình duyệt ở góc độ giao diện người dùng. Tấn công cache injection sẽ tấn công phần lớn các trang web phổ biến vì chúng được bao gồm cả tập tin javascript trong trang web của mình. Hơn nữa tiêm một thư viện độc hại duy nhất cho phép kẻ tấn công nhắm mục tiêu đến nhiều trang web khi họ đang sử dụng của bên thứ ba cùng một thư viện javascript như Google Analytics.

B. Hướng phát triển.

- Thực hiện thành công demo tấn công https bằng cache injection.
- Tìm hiểu thêm các tấn công https khác.
- Đưa ra biện pháp phong thủ đối với tấn công https bằng cache injection nói riêng và các tấn công https nói chung.

TÀI LIỆU THAM KHẢO

[1] Video "Attacking HTTPS with Cache Injection" của Elie Bursztein

[2] BlackHat-USA-2010-Bursztein-Bad-Memories-wp[PDF]

[3] Bad Memories - Black Hat [PDF]

[4]https://docs.google.com/viewer?url=www.google.com/patents/US7836254.pdf

[5]https://docs.google.com/viewer?url=www.google.com/patents/US7836255.pdf

[6]https://docs.google.com/viewer?url=www.google.com/patents/US7484062.pdf

[7]https://docs.google.com/viewer?url=www.google.com/patents/US20100262787.pdf

[8]https://docs.google.com/viewer?url=www.google.com/patents/US20100268896.pdf

[9]https://docs.google.com/viewer?url=www.google.com/patents/US20100070717.pdf

[10]https://docs.google.com/viewer?url=www.google.com/patents/US20100070717.pdf

[11] http://manthang.wordpress.com/2011/08/23/phan-tich-hoat-dong-cua-giao-thuc-https/