BÀI 7.

AN TOÀN DỊCH VỤ WEB
HTTPS

Bùi Trọng Tùng,
Viện Công nghệ thông tin và Truyền thông,
Đại học Bách khoa Hà Nội

1. TỔNG QUAN VỀ DỊCH VỤ WEB

Bùi Trọng Tùng,
Viện Công nghệ thông tin và Truyền thông,
Đại học Bách khoa Hà Nội

World Wide Web

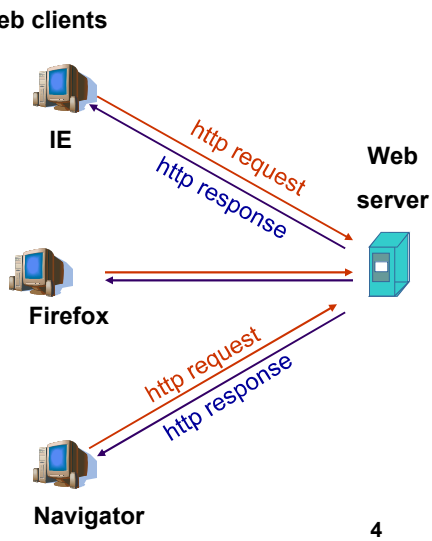
- Ra đời năm 1990
- Hệ thống các siêu văn bản trình bày bằng ngôn ngữ HTML được liên kết với nhau
- Cho phép truy cập đến nhiều dạng tài nguyên thông tin khác nhau (văn bản, hình ảnh, âm thanh, video...) qua URL (Uniform Resource Location) và URI (Uniform Resource Identifier)
- Đang được điều hành bởi W3C
- Các công nghệ liên quan: CSS, XML, JavaScrips, Adobe Flash, Silverlight...
- Hiện tại đã trở thành nền tảng (Web-based service)

3

3

Giao thức HTTP

- Sử dụng TCP, cổng 80
- Trao đổi thông điệp HTTP (giao thức ứng dụng)
  - HTTP Request
  - HTTP Response
- Lưu ý: có rất nhiều cách để tương tác với Web server ngoài trình duyệt



4

4

Thông điệp HTTP Request

- Mã ASCII (dễ dàng đọc được dưới dạng văn bản)

request line (GET, POST, HEAD commands)

header lines

carriage return, line feed at start of line indicates end of header lines

```
GET /~tungbt/index.htm HTTP/1.1\r\n
Host: nct.soict.hust.edu.vn\r\n
User-Agent: Mozilla/5.0
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

5

5

Thông điệp HTTP Response

status line (protocol status code status phrase)

header lines

data, e.g., requested HTML file

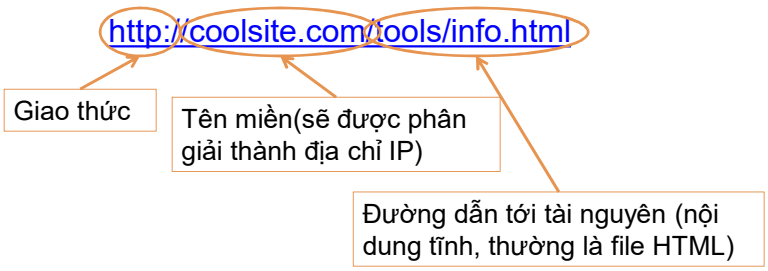
```
HTTP/1.1 200 OK\r\n
Date: Thu, 31 Jul 2015 00:00:14 GMT\r\n
Server: Apache/2.2.15 (CentOS)\r\n
Last-Modified: Wed, 30 Jul 2015 23:59:50 GMT\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Connection: close\r\n
Content-Type: text/html; charset=UTF-8\r\n
\r\n
data data data data data ...
```

6

6

Tương tác với web server

- Địa chỉ URL

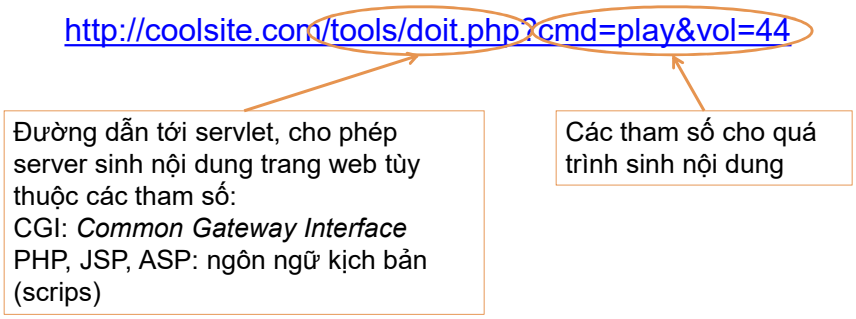


7

7

Tương tác với web server (tiếp)

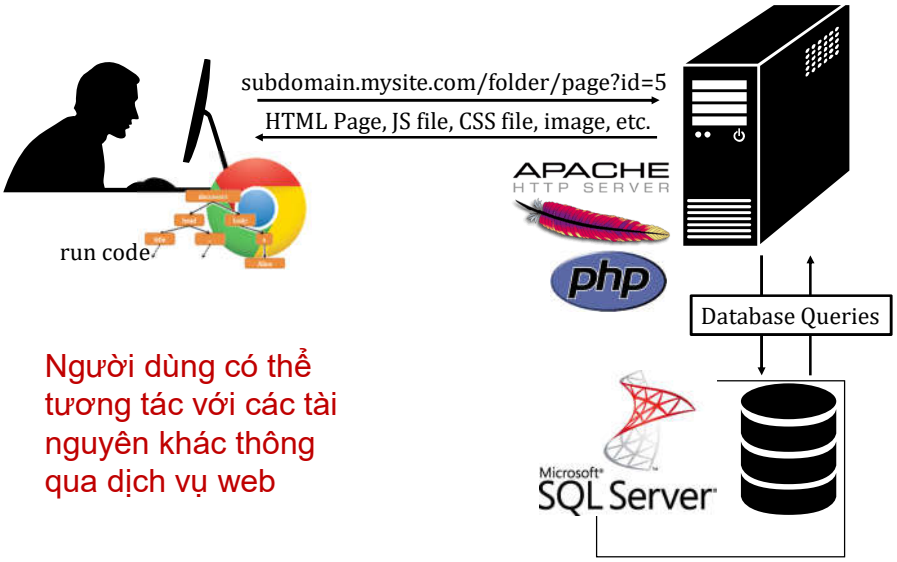
- Tương tác với các kịch bản được thực thi trên server (servlet)



8

8

Kiến trúc chung của các dịch vụ web



Người dùng có thể tương tác với các tài nguyên khác thông qua dịch vụ web

9

9

Hiển trị (rendering) nội dung trang web

- Mô hình xử lý cơ bản tại trình duyệt: mỗi cửa sổ hoặc 1 frame:
  - Nhận thông điệp HTTP Response
  - Hiển thị:
    - Xử lý mã HTML, CSS, Javascripts
    - Gửi thông điệp HTTP Request yêu cầu các đối tượng khác(nếu có)
    - Bắt và xử lý sự kiện
- Các sự kiện có thể xảy ra:
  - Sự kiện của người dùng: OnClick, OnMouseOver…
  - Sự kiện khi hiển thị: OnLoad, OnBeforeUnload…
  - Theo thời gian: setTimeout(), clearTimeout()…

10

10

Ví dụ

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<button type='button' onclick="document.write(5 + 6)">
                                                    Try it</button>

</body>
</html>
```

11

11

Document Object Model(DOM)

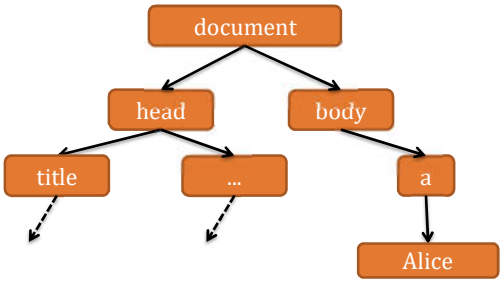
- Tổ chức các đối tượng của trang HTML thành cấu trúc cây
- Cung cấp API (hướng đối tượng)để tương tác
- Ví dụ:
  - Thuộc tính: document.alinkColor, document.URL, document.forms[ ], document.links[ ]…
  - Phương thức: document.write()…
- Bao gồm cả đối tượng của trình duyệt(Browser Object Model - BOM): window, document, frames[], history, location…

12

12

DOM – Ví dụ

```
<html>
<head><title>Example</title> ... </head>
<body>
<a id="myid" href="javascript:flipText()">Alice</a>
</body></html>
```

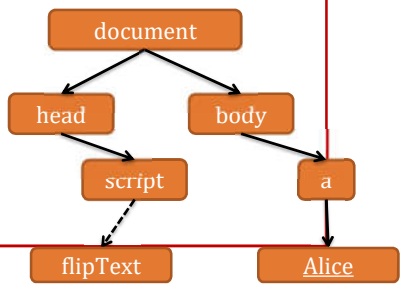


13

13

DOM – Ví dụ khác

```
<head> ...
<script type="text/javascript">
  flip = 0;
  function flipText() {
   var x = document.getElementById('myid').firstChild;
   if(flip == 0) { x.nodeValue = 'Bob'; flip = 1;}
   else { x.nodeValue = 'Alice'; flip = 0; }
  }
</script>
</head>
<body>
<a id="myid"
   href="javascript:flipText()">
   Alice
</a>
</body>
```



14

14

Javascript

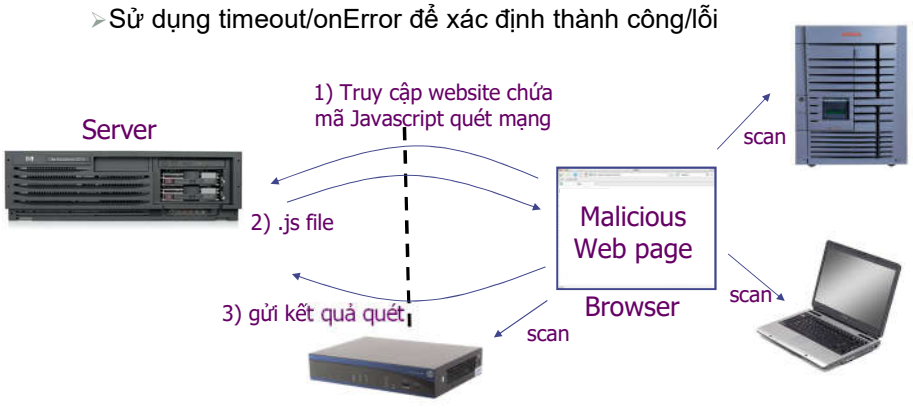
- Ngôn ngữ cho phép xây dựng các script để trình duyệt thực thi
- Được nhúng vào các trang web mà server trả về cho client
- Thường sử dụng để tương tác với DOM
- Khả năng tương tác rất mạnh:
  - Thay đổi nội dung trang web trên trình duyệt
  - Theo dõi và xử lý các sự kiện trên trang web (bao gồm cả hành vi của người dùng: rê chuột, nhấp chuột, gõ phím)
  - Đọc, thiết lập cookie
  - Duy trì kết nối (AJAX)
  - Sinh các HTTP Request khác và đọc HTTP Response trả về

15

15

Javascript – Quét mạng

- Cách thức thực hiện:
  - Gửi yêu cầu tới một ảnh với địa chỉ cục bộ
    - ✓ Ví dụ:
  - Sử dụng timeout/onError để xác định thành công/lỗi



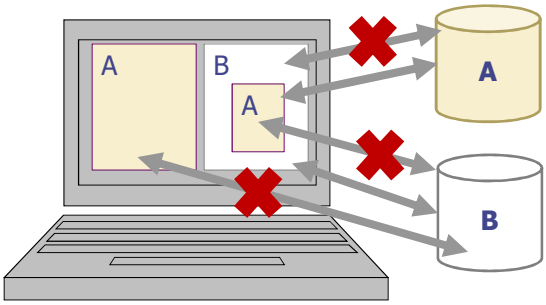
16

16

Chính sách cùng nguồn(SOP)

- Same Origin Policy
- Chính sách sử dụng trên trình duyệt Web
- Nguồn = Giao thức + Hostname + Cổng ứng dụng
- Chính sách SOP: Trang web của nguồn này không được phép đọc, thay đổi nội dung trang web của nguồn khác

→ cách ly các trang web khác nguồn



17

17

SOP – Ví dụ

Kiểm tra khả năng truy cập từ
http://www.example.com/dir/page.html

Compared URL	Outcome
http://www.example.com/dir/page2.html	Success
http://www.example.com/dir2/other.html	Success
http://username:password@**www.example.com**/dir2/other.html	Success
http://www.example.com:**81**/dir/other.html	Failure
https://www.example.com/dir/other.html	Failure
http://**en.example.com**/dir/other.html	Failure
http://**example.com**/dir/other.html	Failure
http://**v2.www.example.com**/dir/other.html	Failure
http://www.example.com:**80**/dir/other.html	Depends

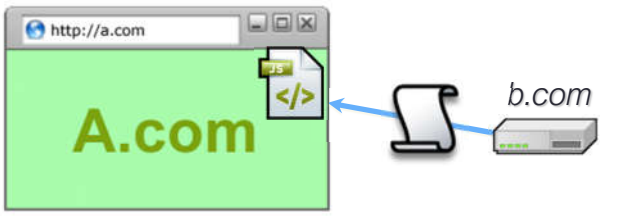
18

18

SOP – Sử dụng thư viện bên thứ 3

- Thư viện Javascript của bên thứ 3 có thể được nào vào trang Web <script> src = “URL to .js” </script>
- Script có quyền thực thi trên trang đã nạp, không phải trên server nguồn
- Ví dụ: script khi thực thi trên a.com có quyền tương tác với DOM của trang này nhưng không có quyền trên DOM của trang b.com

```
<script src=https://b.com/script/example.js></script>
```



Xuất hiện nguy cơ mới

19

19

Sử dụng thư viện bên thứ 3

- Lỗ hổng: script có thể tương tác với trang web đã nạp nó

→ thiếu cơ chế cách ly do SOP không còn được áp dụng

0

20

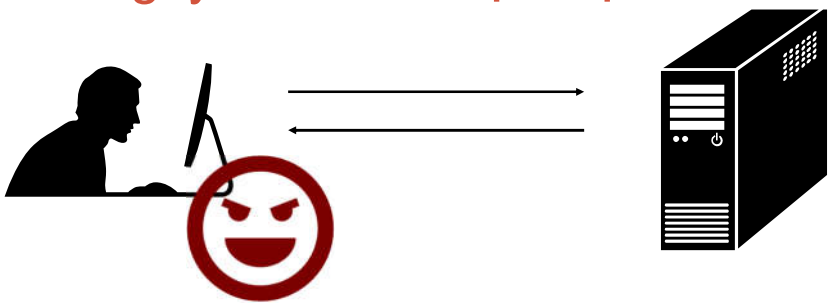
Giải pháp

- HTML5 Web Workers: cách ly script bằng cách thực thi trong luồng riêng, vẫn cùng nguồn với frame đã tạo luồng đó
  - Tương tác với luồng chính bằng postMesage()
- Thu hẹp phạm vi thực thi và giao tiếp:
  - HTML5 Sandbox
  - Content Security Policy
- Sub Resource Integrity(SRI): lập trình viên cung cấp mã băm của tài nguyên đã nạp trên trang Web, trình duyệt kiểm tra tính toàn vẹn
- Cross-Origin Resource Sharing: kiểm soát chia sẻ tài nguyên giữa các trang Web khác nguồn

21

21

Các nguy cơ đối với dịch vụ web

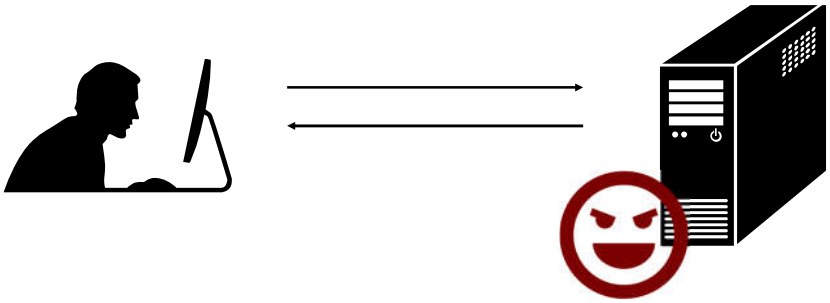


- Tấn công server từ phía client
  - Tấn công dạng Injection
  - File System Traversal
  - Broken Access Control

22

22

Các nguy cơ đối với dịch vụ web

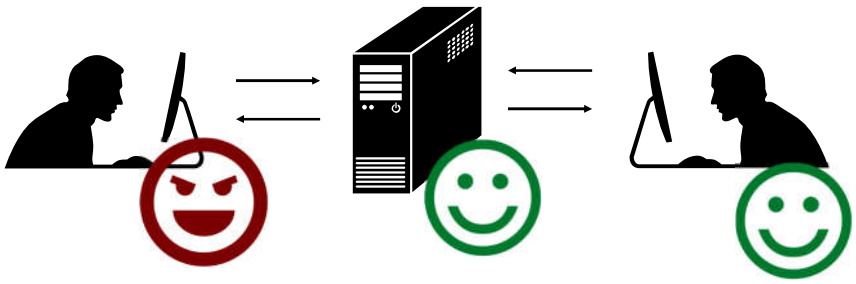


- Tấn công từ phía server:
  - Clickjacking
  - HistoryProbing
  - Phishing

23

23

Các nguy cơ đối với dịch vụ web



- Tấn công người dùng khác:
  - XSS
  - CSRF
  - Remote Script Inclusion

24

24

2017 OWASP Top10 Project

Mã	Tên	Mô tả
A-1	Injection	Cho phép chèn dữ liệu độc hại vào câu lệnh hoặc truy vấn
A-2	Broken Authentication	Đánh cắp mật khẩu, khóa, thẻ phiên, hoặc khai thác lỗ hổng để giả mạo người dùng
A-4	XML External Entities (XXE)	Khai thác lỗ hổng xử lý XML
A-5	Broken Access Control	Các dạng tấn công vượt quyền truy cập
A-6	Security Misconfiguration	Lỗ hổng khi cấu hình, triển khai website

25

25

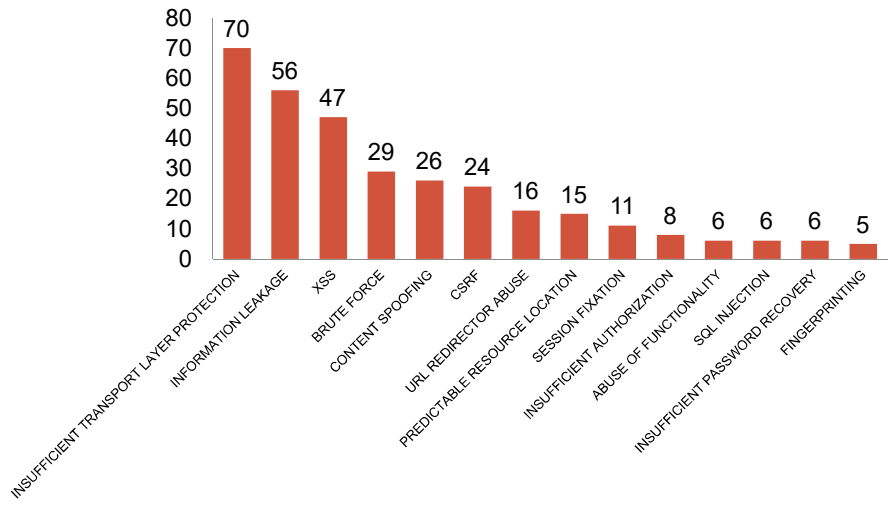
2017 OWASP Top10 Project

Mã	Tên	Mô tả
A-7	XSS	Ứng dụng Web không kiểm tra mã độc Javascript nhúng vào dữ liệu đầu vào
A-8	Insecure Deserialization	Không kiểm tra dữ liệu đóng gói trong các đối tượng Serialization
A-9	Using Components with Known Vulnerabilities	Sử dụng các công cụ, thành phần phần mềm chưa vá lỗ hổng bảo mật đã được công bố
A-10	Insufficient Logging & Monitoring	Không ghi nhật ký và giám sát đầy đủ

26

26

Top 15 lỗ hổng(2015 White Hat Security)



27

27

2. GIAO THỨC HTTPS

Bùi Trọng Tùng,
Viện Công nghệ thông tin và Truyền thông,
Đại học Bách khoa Hà Nội

28

28

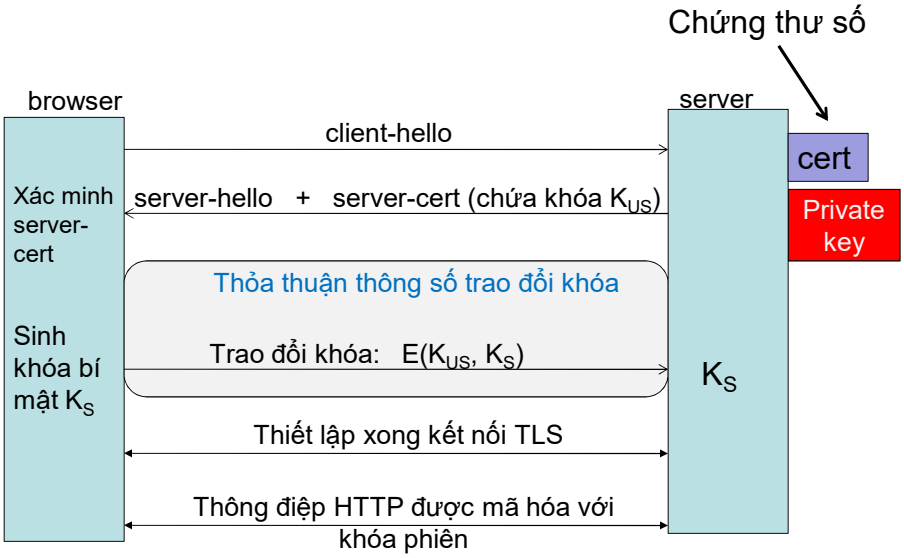
Giới thiệu chung về HTTPS

- Hạn chế của HTTP:
  - Không có cơ chế để người dùng kiểm tra tính tin cậy của Web server → lỗ hổng để kẻ tấn công giả mạo dịch vụ hoặc chèn mã độc vào trang web HTML
  - Không có cơ chế mã mật → lỗ hổng để kẻ tấn công nghe lén đánh cắp thông tin nhạy cảm
- Secure HTTP: Kết hợp HTTP và SSL/TLS:
  - Xác thực
  - Bảo mật

29

29

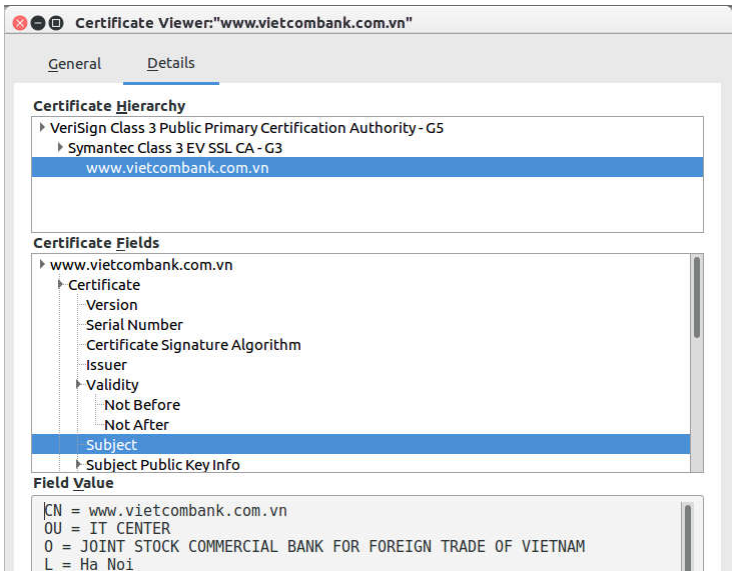
Thiết lập liên kết SSL/TLS



30

30

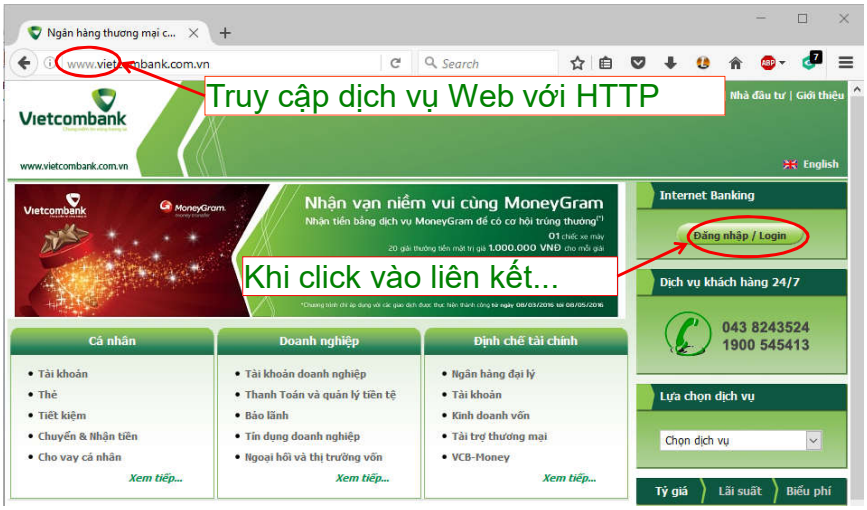
Chứng thư số - Ví dụ



31

31

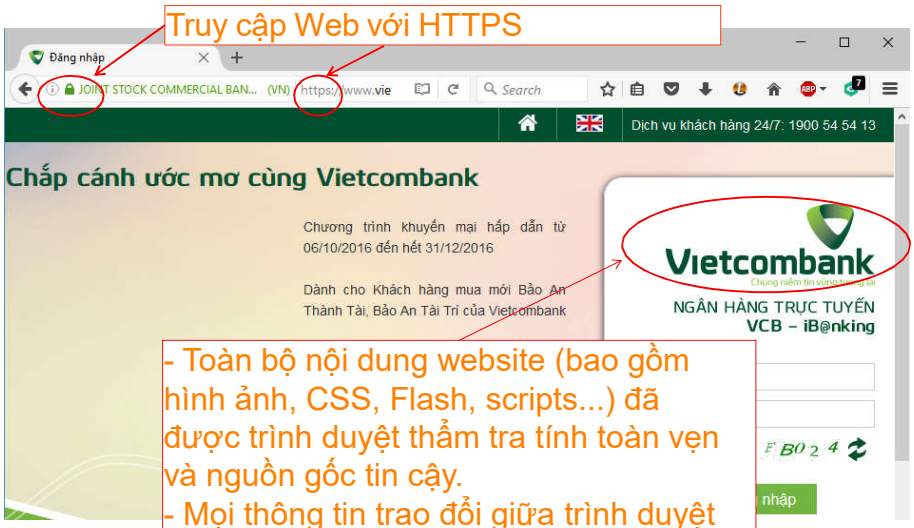
HTTP trên trình duyệt Web



32

32

HTTPS trên trình duyệt Web



33

33

Tại sao HTTPS an toàn?

- ~~Chặn bắt dữ liệu~~
- ~~Chèn mã độc vào nội dung website khi truyền từ server tới trình duyệt web~~
- Tấn công DNS cache poisioning
  - ~~Client truy cập vào Web server của kẻ tấn công~~
- Tấn công DHCP Spoofing
  - ~~Client truy cập vào Web server của kẻ tấn công~~
- ~~Tấn công định tuyến để chuyển hướng truy cập~~
- ~~Tấn công man-in-the-middle~~
- ~~Tấn công phát lại~~

34

34

Quá trình xác minh chứng thư số

- Bước 1: Trình duyệt so sánh tên miền trong chứng thư số (Subject CN)và tên miền trên địa chỉ URL
  - Tên tường minh: dnsimple.com, hoặc
  - Tên đại diện: *.dnsimple.com, dn*.dnsimple.com
- Bước 2: Trình duyệt kiểm tra thời gian hiệu lực của chứng thư số
- Bước 3: Trình duyệt kiểm tra chứng thư số gốc của CA chứng thực cho server
  - Để xem các chứng thư số gốc trên trình duyệt Firefox

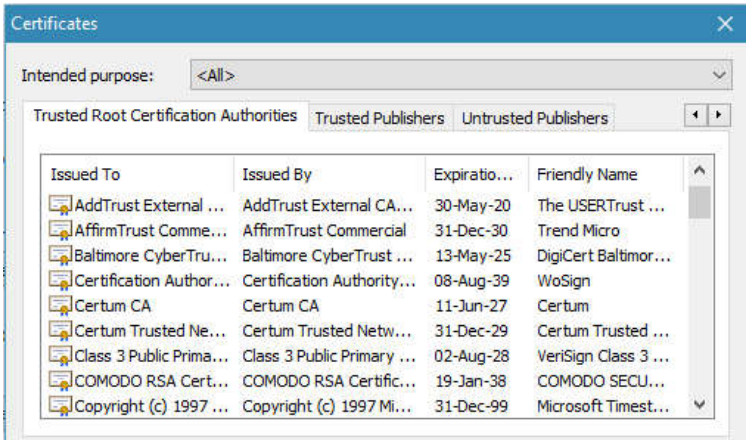
  Options→Advanced→View Certificates→Authorities
- Bước 4: Trình duyệt sử dụng chứng thư số gốc của CA để thẩm tra chữ ký số trên chứng thư của server

35

35

Chứng thư số gốc

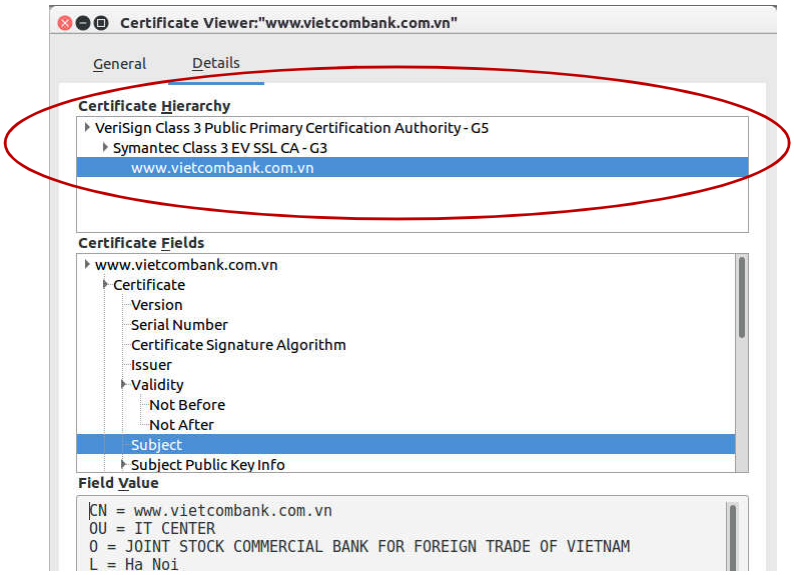
- Được tích hợp sẵn trên trình duyệt



36

36

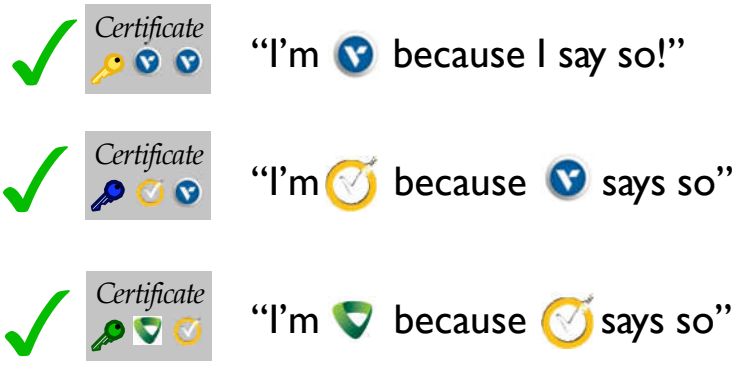
Chuỗi chứng thực



37

37

Chuỗi chứng thực



Chuỗi xác thực từ chối chứng thư số nếu có bất kỳ bước nào cho kết quả xác thực thất bại

38

38

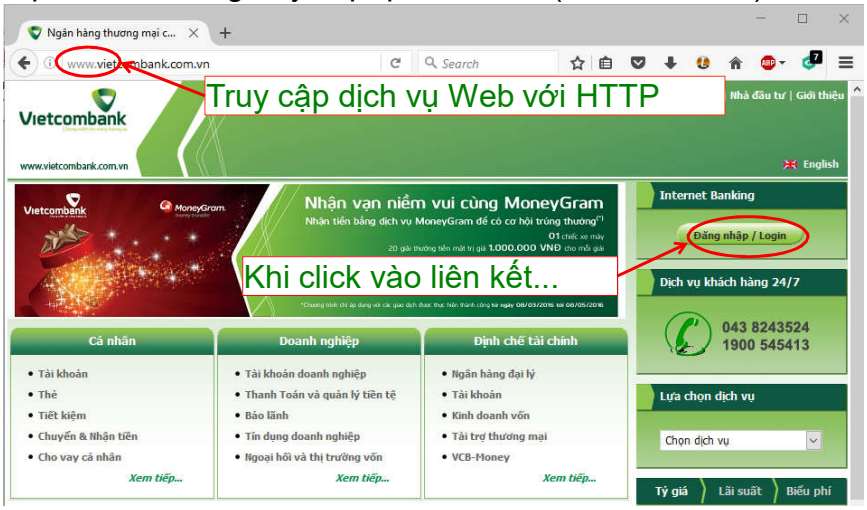
Không tìm thấy chứng thư số gốc

39

39

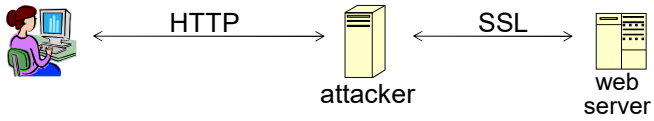
Tấn công vào HTTPS

- Tấn công sslstrip: lợi dụng lỗ hổng chuyển từ truy cập qua HTTP sang truy cập qua HTTPS (như thế nào?)



40

Tấn công sslstrip



Nội dung trả về từ web server		Nội dung thay thế bởi attacker
<a href=https://…>	→	<a href=http://…>
Location: https://...	→	Location: http://... (redirect)
<form action=https://… >	→	<form action=http://…>

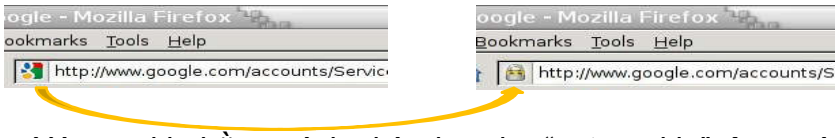
- Các trang có lỗ hổng này: ebay.com, rất nhiều ngân hàng lớn ở Việt Nam (Agribank, Vietinbank, Vietcombank, BiDV, ACB bank...)

41

41

Tấn công sslstrip

- Thậm chí, trên các website hỗ trợ đầy đủ HTTPS, nhưng có thể lợi dụng người dùng không cập nhật trình duyệt phiên bản mới
- Thay fav icon

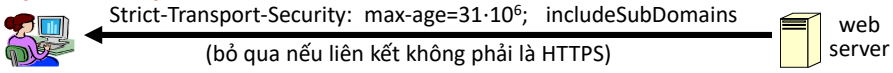


- Xóa cookie bằng cách chèn header “set-cookie” → người dùng bắt buộc đăng nhập lại
  - Phần lớn không phát hiện HTTPS → HTTP

42

42

Phòng chống: Strict Transport Security (HSTS)



- Một trường trên tiêu đề của HTTPS Response, yêu cầu trình duyệt luôn kết nối với máy chủ qua HTTPS
- HSTS chỉ được chấp nhận trên các liên kết HTTPS
- Max-age: thời gian duy trì
- Kiểm tra danh sách các website hỗ trợ HSTS:
  - chrome://net-internals/#hsts
  - %APPDATA%\Mozilla\Firefox\Profiles\...\SiteSecurityServiceState.txt

43

43

CSP: upgrade-insecure-requests

- Các trang thường trộn lẫn các URL của tài nguyên với HTTPS

Ví dụ: <img src="http://site.com/img">

- Thêm vào tiêu đề trong HTTP Response

Content-Security-Policy: upgrade-insecure-requests

- Trình duyệt tự động chuyển các liên kết sử dụng HTTP sang HTTPS
- Không áp dụng với thẻ <a>

<img src="http://site.com/img">	<img src="https://site.com/img">
<img src="http://othersite.com/img">	<img src="https://othersite.com/img">
<a href="http://site.com/img">	<a href="https://site.com/img">
<a href="http://othersite.com/img">	<a href="http://othersite.com/img">

Luôn sử dụng URL tương đối: <img src="//site.com/img">

44

44

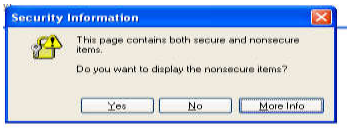
Tấn công vào HTTPS

- Giao thức truy cập là HTTPS nhưng nội dung website không được chứng thực đầy đủ
- Ví dụ:

  <script src="http://.../script.js>

  <img src="http://site.com/img">
- Nguy cơ: Kẻ tấn công thay thế những nội dung này
- Cảnh báo trên trình duyệt

IE7:



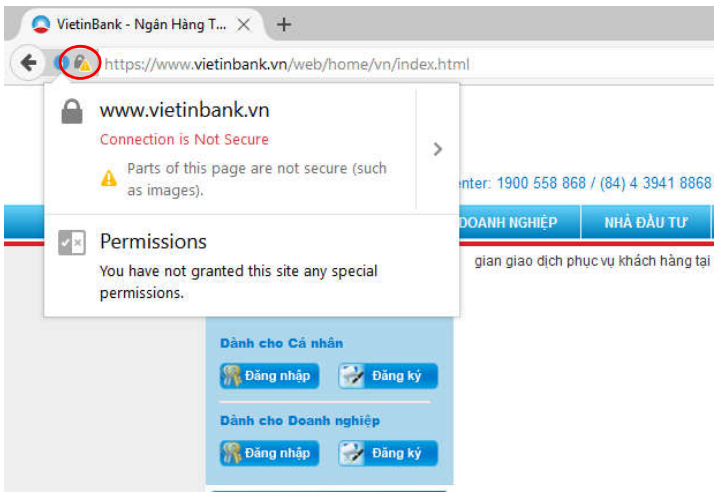
Chrome(Các phiên bản cũ):

- Chính sách của Chrome: chặn CSS, mã Javascript, thẻ <frame>

45

45

Tấn công lợi dụng website không được chứng thực đầy đủ-Ví dụ



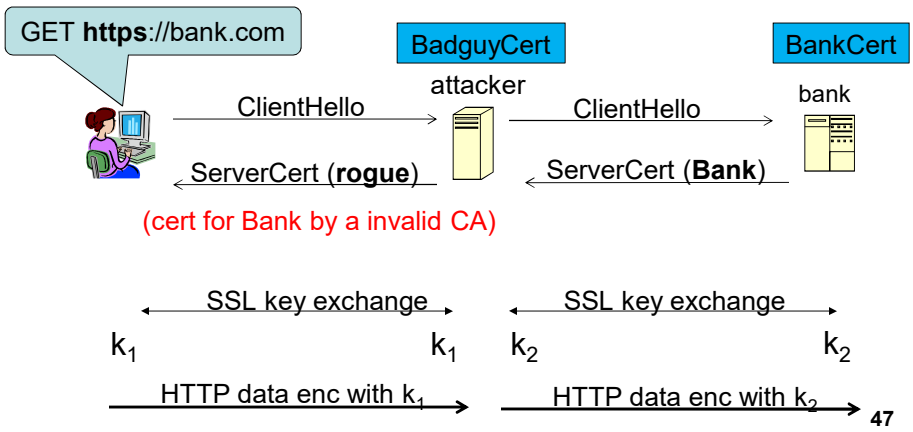
46

46

Tấn công vào HTTPS

- Sử dụng CA giả mạo để phát hành chứng thư giả mạo
- Người dùng sử dụng phiên bản trình duyệt không an toàn

→ chứng thư gốc bị thay thế



47

47

Phát hành chứng thư số sai cách

- 2011: Comodo and DigiNotar CAs bị tấn công, phát hành chứng thư số cho các tên miền của Gmail, Yahoo! Mail, …
- 2013: TurkTrust phát hành chứng thư số cho gmail.com (phát hiện nhờ cơ chế Dynamic HTTP public-key pinning)
- 2014: Indian NIC phát hành chứng thư số cho các tên miền của Google
- 2015: MCS phát hành chứng thư số cho các tên miền của Google

⇒ trình duyệt không phát cảnh báo khi kẻ tấn công thay thế chứng thư số

48

48

Phòng chống chứng thư số giả mạo

- Dynamic HTTP public-key pinning:
  - TOFU(Trust on First Use):
    - Trường Public-Key-Pins trong tiêu đề của HTTP Response chỉ ra CA đã cấp phát chứng thư số cho website
    - Nếu các phiên tiếp theo, sử dụng chứng thư được chứng thực bởi CA khác → từ chối
- Giao thức "Certificate Transparency" cho phép CA công bố toàn bộ bản ghi nhật ký (log) về các chứng thư đã phát hành
- Giao thức OCSP (Online Certificate Status Protocol)

49

49

HPKP – Ví dụ

Public-Key-Pins: max-age=2592000;
pin-sha256="E9CZ9INDbd+2eRQozYqqbQ2yXLVKB9+xcprMF+44U1g=";
pin-sha256="LPJNul+wow4m6DsqxbninhsWHlwfp0JecwQzYpOLmCQ=";
report-uri="https://example.net/pkp-report"

- chrome://net-internals/#hsts
- https://dxr.mozilla.org/mozilla-central/source/security/manager/tools/PreloadedHPKPins.json

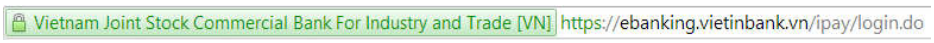
50

50

Tấn công phishing

- Homograph attack: Lợi dụng hình dáng giống nhau của một số ký tự. Ví dụ: vvesternbank, paypai, paypail, paypa1...
  - Sử dụng mã Unicode trong URL: paypal.com
- Semantic attack: lợi dụng các trình duyệt cũ không phân biệt các dấu đặc biệt trên tên miền. Ví dụ: Kẻ tấn công có thể mua tên miền var.cn và sử dụng tên miền con

  www.pnc.com/webapp/homepage.**var.cn**
- Phòng chống: sử dụng chứng thư hỗ trợ chứng thực mở rộng (Extended validation certificate)

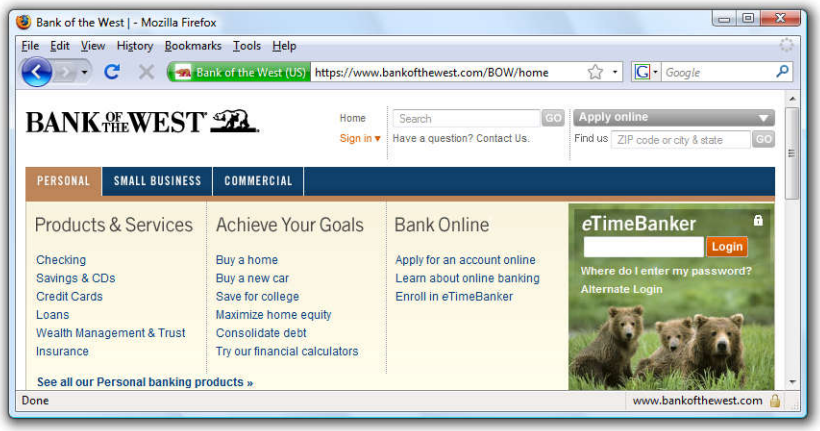


51

51

Homograph attack – Ví dụ

- Website bị giả mạo

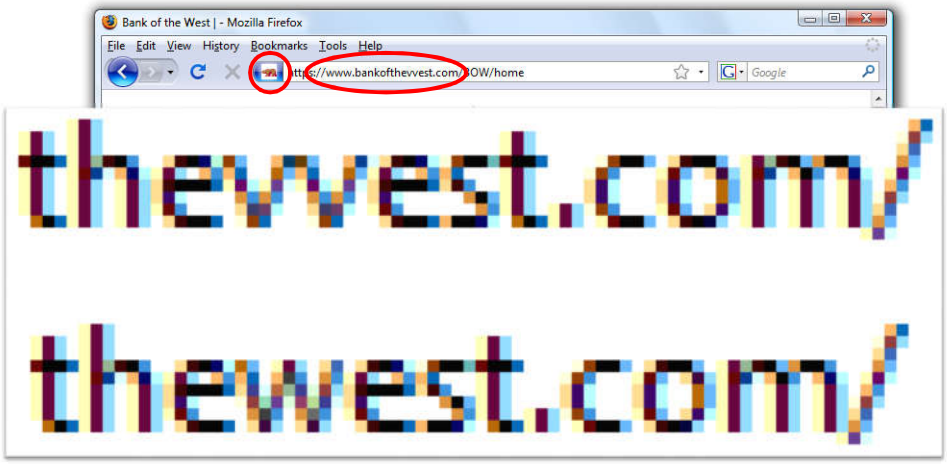


52

52

Homograph attack

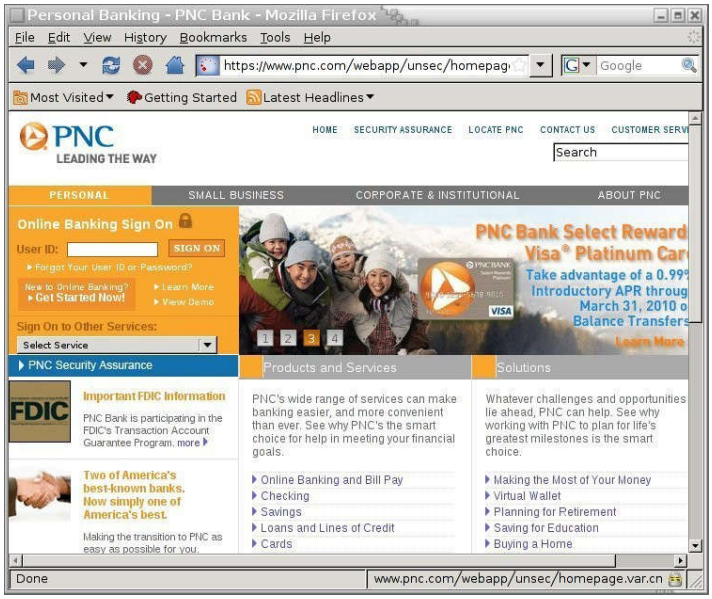
- Website giả mạo



53

53

Semantic attack (ví dụ)

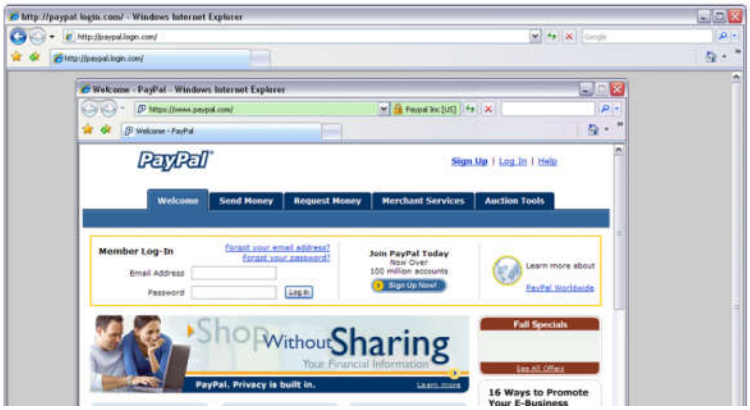


54

54

Tấn công phishing (tiếp)

- Picture-in-picture attack: dựng một frame chứa cửa sổ giao diện của một website đã được chứng thực →lợi dụng người dùng bất cẩn không quan sát khi duyệt web



55

55

Bài giảng sử dụng một số hình vẽ và ví dụ từ các khóa học:

- Computer and Network Security, Stanford University
- Computer Security, Berkeley University

56

56