

BÀI 7. AN TOÀN AN NINH DỊCH VỤ WEB(1) TỔNG QUAN + HTTPS

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

1

1

1. TỔNG QUAN VỀ DỊCH VỤ WEB

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

2

2

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

Các đe dọa đối với dịch vụ web

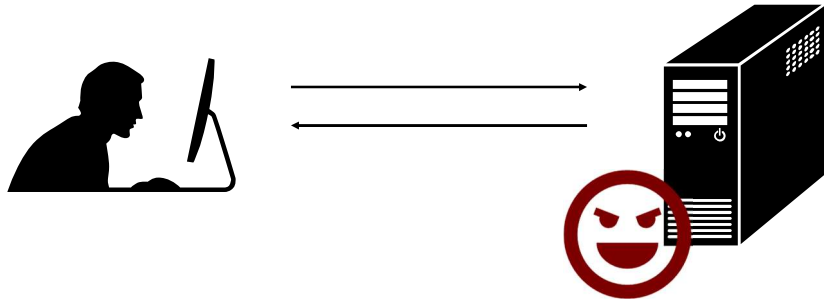


- Tấn công server từ phía client
 - Tấn công dạng Injection
 - File System Traversal
 - Broken Access Control

4

4

Các đe dọa đối với dịch vụ web

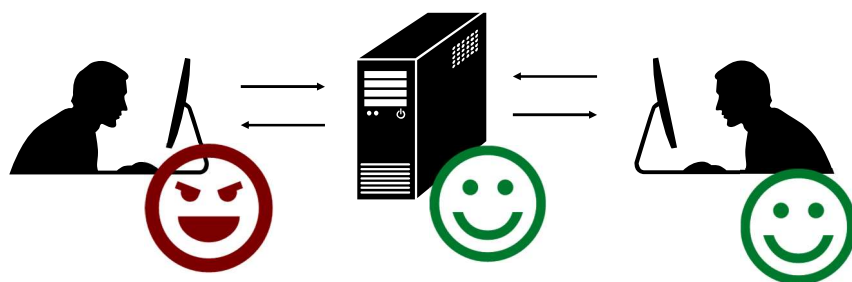


- Tấn công từ phía server:
 - Clickjacking
 - HistoryProbing
 - Phishing

5

5

Các đe dọa đối với dịch vụ web

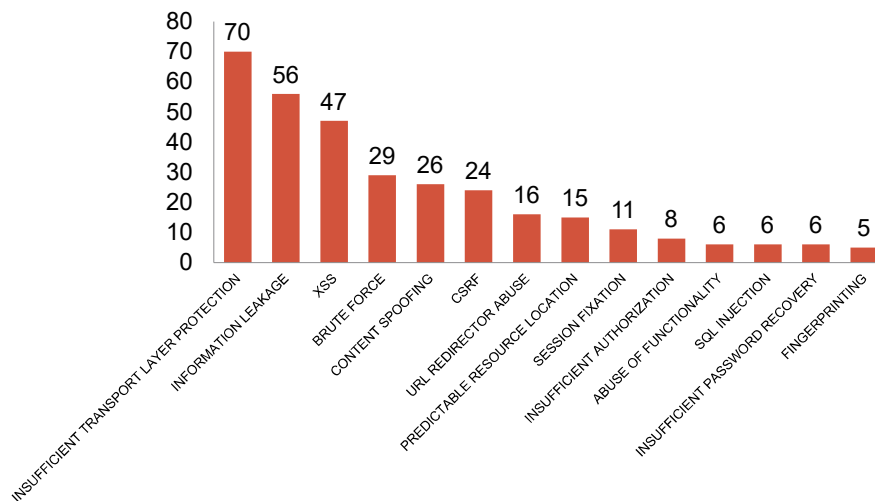


- Tấn công người dùng khác:
 - XSS
 - CSRF
 - Remote Script Inclusion

6

6

Top 15 lỗ hổng(2015 White Hat Security)



7

7

2020 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

8

8

2020 OWASP Top10 Project

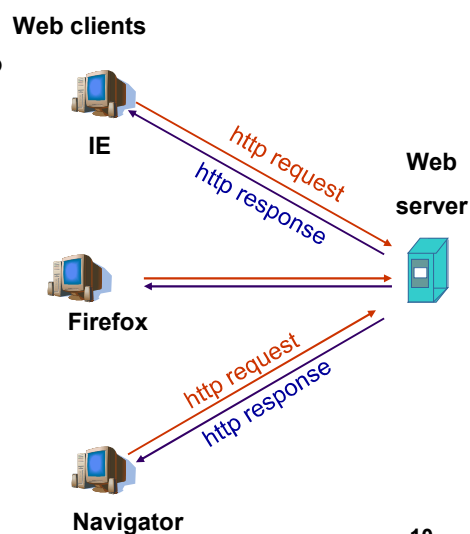
Mã	Tên	Mô tả
A-7	XSS	Ứng dụng Web không kiểm tra mã độc Javascript nhúng vào dữ liệu đầu vào
A-8	Insecure Deserialization	Không kiểm tra dữ liệu đóng gói trong các đối tượng Serialization
A-9	Using Components with Known Vulnerabilities	Sử dụng các công cụ, thành phần phần mềm chưa vá lỗ hổng bảo mật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

9

9

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



10

10

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
```

11

11

Thông điệp HTTP Response

status line
(protocol
status code
status phrase)

header
lines

data, e.g.,
requested
HTML file

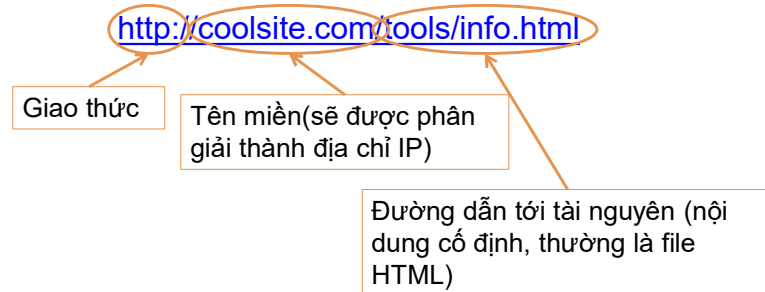
```
HTTP/1.1 200 OK\r\n
Date: Thu, 31 Jul 2015 00:00:14 GMT\r\n
Server: Apache/2.2.15 (CentOS)\r\n
Last-Modified: Wed, 30 Jul 2015 23:59:50 GMT\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Connection: close\r\n
Content-Type: text/html; charset=UTF-8\r\n
\r\n
data data data data data ...
```

12

12

Tương tác với web server

- Địa chỉ URL



- URL Encode: Biểu diễn các ký tự không hiển thị, ký tự trắng, ký tự đặc biệt
 - `%xx`: trong đó xx là mã ASCII của ký tự cần biểu diễn

13

13

Tương tác với web server (tiếp)

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

<http://coolsite.com/tools/doit.php?cmd=play&vol=44>

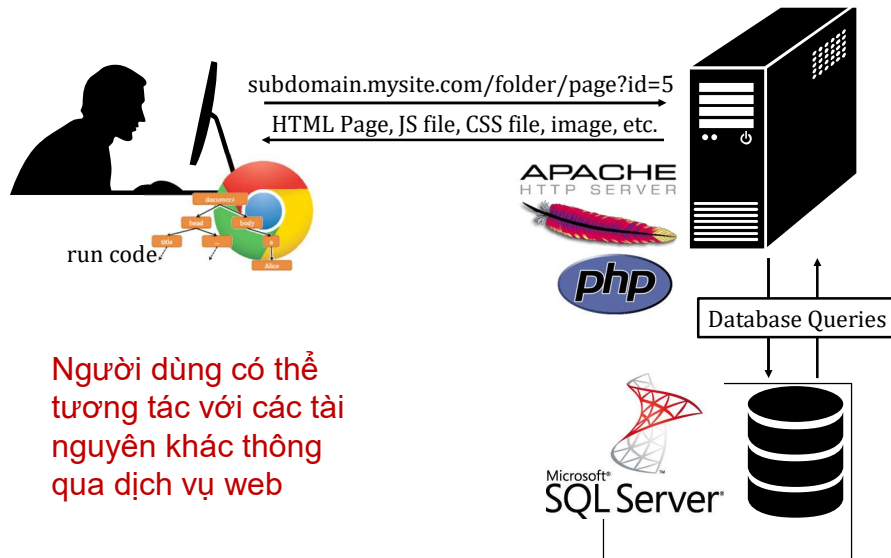
Đường dẫn tới servlet, cho phép server sinh nội dung trang web tùy thuộc các tham số:
CGI: *Common Gateway Interface*
PHP, JSP, ASP: ngôn ngữ kịch bản (scripts)

Các tham số cho quá trình sinh nội dung

14

14

Kiến trúc chung của các dịch vụ web



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

15

15

Hiển thị (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()...

16

16

Document Object Model(DOM)

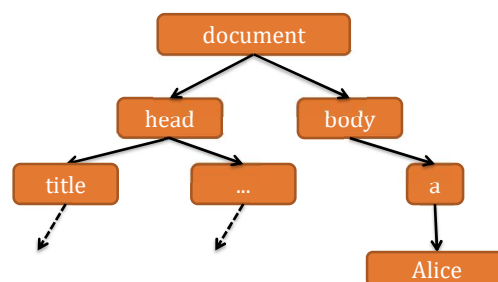
- Tổ chức các đối tượng của trang HTML thành cấu trúc cây
- Cung cấp API (hướng đối tượng) để tương tác
- Ví dụ:
 - Thuộc tính: document.alinkColor, document.URL, document.forms[], document.links[]...
 - Phương thức: document.write()...
- Bao gồm cả đối tượng của trình duyệt(Browser Object Model - BOM):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

17

17

DOM – Ví dụ

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

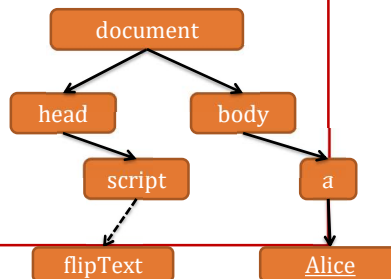


18

18

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>
```



19

19

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

20

20

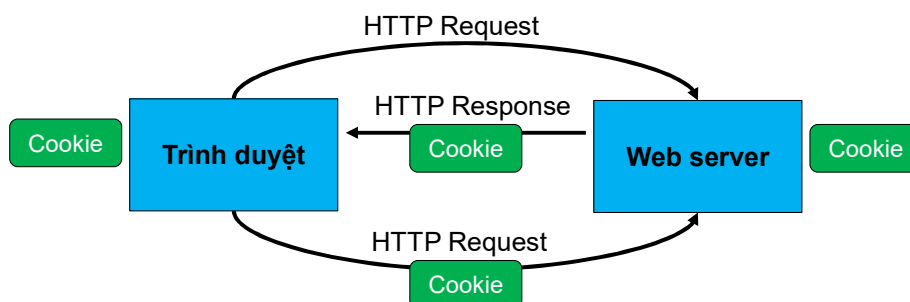
HTTP là giao thức stateless

- Một phiên hoạt động của HTTP:
 - Trình duyệt kết nối với Web server
 - Trình duyệt gửi thông điệp yêu cầu HTTP Request
 - Web server đáp ứng với một thông điệp HTTP Response
 - ...lặp lại...
 - Trình duyệt ngắt kết nối
- Các thông điệp HTTP Request được xử lý độc lập
- Web server không ghi nhớ trạng thái của phiên HTTP

21

21

HTTP Cookie



- Cookie: dữ liệu do Web server tạo ra, chứa thông tin trạng thái của phiên làm việc
 - Server có thể lưu lại cookie(một phần hoặc toàn bộ)
- Sau khi xử lý yêu cầu, Web server trả lại thông điệp HTTP Response với cookie đính kèm
 - Set-Cookie: key = value; options;
- Trình duyệt lưu cookie
- Trình duyệt gửi HTTP Request tiếp theo với cookie được đính kèm

22

22

HTTP Cookie - Ví dụ

HTTP Response

```
HTTP/1.1 200 OK
Server: nginx/1.10.1
Date: Mon, 14 Nov 2016 09:19:19 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/5.4.45
Set-Cookie: vflastvisit=1479115159; expires=Tue, 14-Nov-2017 09:19:19 GMT; path=/; domain=vozforums.com; secure
Set-Cookie: vflastactivity=0; expires=Tue, 14-Nov-2017 09:19:19 GMT; path=/; domain=vozforums.com; secure
Expires: 0
Cache-Control: private, post-check=0, pre-check=0, max-age=0
Pragma: no-cache
Content-Encoding: gzip
```

23

23

HTTP Cookie - Ví dụ

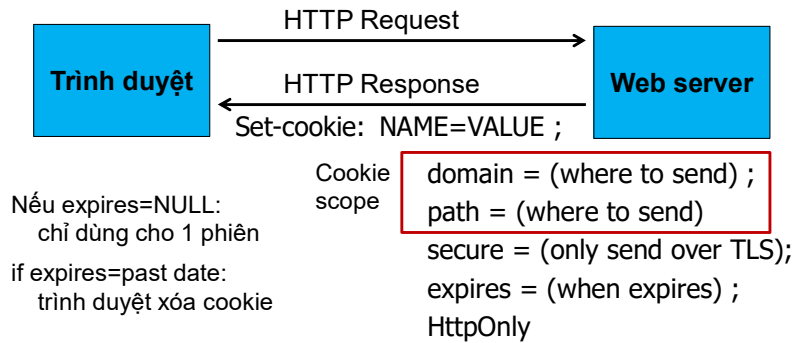
• HTTP Request

```
GET /clientscript/vbulletin_important.css?v=380 HTTP/1.1
Host: vozforums.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:49.0) Gecko/20100101 Firefox/49.0
Accept: text/css,*/*;q=0.1
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://vozforums.com/
Cookie: vflastvisit=1479115159; vflastactivity=0
Connection: keep-alive
```

24

24

HTTP Cookie



- Cookie scope: chỉ định các trang web sẽ gửi cookie tới
- HttpOnly: không thể đọc cookie bằng Javascript tại client
- secure: Chỉ gửi cookie trên kênh truyền TLS

25

25

Đọc ghi cookie tại trình duyệt

- Truy cập qua đối tượng DOM: `document.cookie`
- Thiết lập giá trị:
`document.cookie = "name=value; expires=...; "`
- Hiển thị cookie: `alert(document.cookie)`
 - Hiển thị dưới dạng 1 chuỗi gồm giá trị trong các thuộc tính của tất cả cookie đã lưu cho tài nguyên này
- Xóa cookie:
`document.cookie = "name=; expires= [Ngày trong quá khứ]"`

26

26

2. CHÍNH SÁCH SOP

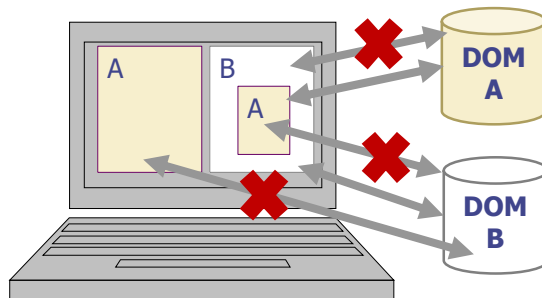
Bùi Trọng Tùng,
Viện Công nghệ thông tin và Truyền thông,
Đại học Bách khoa Hà Nội

27

27

Chính sách cùng nguồn(SOP)

- Same Origin Policy
 - Chính sách sử dụng trên trình duyệt Web
 - Nguồn(Origin) = 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



28

28

SOP – Ví dụ

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

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

29

29

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>
```



b.com

Xuất hiện
nguy cơ mới

30

30

Sử dụng thư viện bên thứ 3

- Lỗi hỏng: script có thể tương tác với trang web đã nạp nó
→ thiếu cơ chế cách ly do SOP không còn được áp dụng

Đánh cắp thông tin

```
var c = document.getElementsByName("password")[0]
```

Nhúng trực tiếp mã nguồn JavaScript của bên thứ 3 làm phát sinh nguy cơ cho trang đã sử dụng nó

Gửi lại cho kẻ tấn công

```

```



_ 1

31

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 `postMessage()`
- 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

32

32

Chính sách SOP cho cookie

- Địa chỉ URL: scheme://domain:port/path?params
- Nguồn(origin) của cookie được xác định bởi: domain, path và scheme(không bắt buộc)
- **Thiết lập cookie**: một trang web có thể thiết lập cookie cho các trang có cùng tên miền, hoặc mang tên miền cấp trên(trừ tên miền cấp 1)
- Ví dụ: trang web có domain là login.site.com:
 - Thiết lập được cookie với domain = login.site.com, site.com
 - Không thiết lập được với domain = othersite.com, other.site.com, .com
 - path: bất kỳ giá trị nào

33

33

Chính sách SOP cho cookie

- **Đọc cookie**: Server có thể đọc được tất cả cookie trong scope của nó
- Trình duyệt gửi tất cả cookie trong scope(domain và path) tới server:
- Nếu giá trị secure được thiết lập thì cookie chỉ được gửi nếu giao thức là HTTPS
- Ví dụ: cookie với domain = example.com và path = /some/path/ sẽ được đính kèm vào thông điệp HTTP Request tới địa chỉ
<http://foo.example.com/some/path/subdirectory/hello.html>

34

34

SOP cho cookie – Ví dụ khác

- Ba cookie được thiết lập bởi login.site.com

Cookie 1

value = a
domain = login.site.com
path = /
secure

Cookie 2

value = b
domain = site.com
path = /

Cookie 3

value = c
domain = site.com
path = /my/home

- Cookie được đặt trong HTTP Request như sau:

URL	Cookie 1	Cookie 2	Cookie 3
http://login.site.com	No	Yes	No
https://login.site.com	Yes	Yes	No
https://site.com	No	Yes	No
http://site.com/my/home	No	Yes	Yes
http://account.site.com	No	Yes	No
https://login.site.com/my/home	Yes	Yes	Yes

35

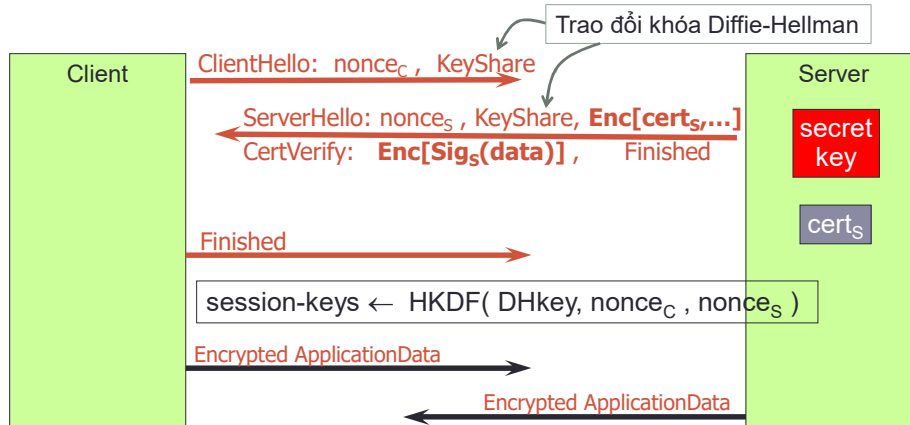
3. GIAO THỨC HTTPS

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

36

36

Thiết lập kết nối TLS 1.3



37

37

HTTPS trên trình duyệt Web

Truy cập Web với HTTPS

The screenshot shows a web browser with the address bar displaying `https://www.vietcombank.vn`. The browser's address bar shows a green padlock icon, indicating a secure connection. The page content includes a banner for Vietcombank and a login form.

- Toàn bộ nội dung website (bao gồm hình ảnh, CSS, Flash, scripts...) đã được trình duyệt thẩm tra tính toàn vẹn và nguồn gốc tin cậy.
- Mọi thông tin trao đổi giữa trình duyệt và Vietcombank được giữ bí mật.

38

38

Quá trình xác minh chứng thư số

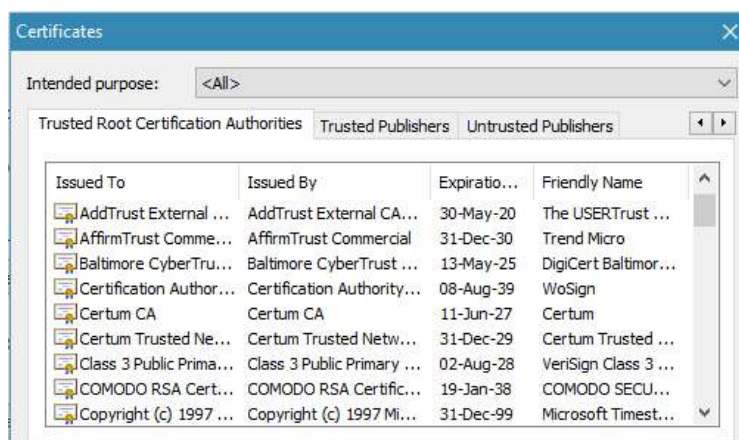
- Bước 1: Trình duyệt so sánh tên miền trong chứng thư số (Subject CN) và tên miền trên địa chỉ URL
 - Tên tường minh: dnsimple.com, hoặc
 - Tên đại diện: *.dnsimple.com, dn*.dnsimple.com
- Bước 2: Trình duyệt kiểm tra thời gian hiệu lực của chứng thư số
- Bước 3: Trình duyệt kiểm tra chứng thư số gốc 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 kiểm tra trạng thái của chứng thư số
- Bước 5: Trình duyệt sử dụng chứng thư số gốc của CA để thẩm tra chữ ký số trên chứng thư của server

39

39

Chứng thư số gốc

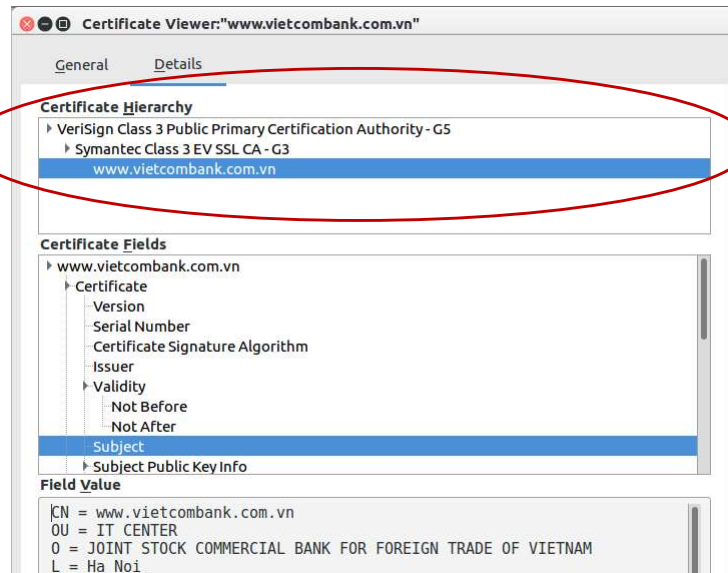
- Được tích hợp sẵn trên trình duyệt



40

40

Chuỗi chứng thực



41

41

Chuỗi chứng thực

- ✓  “I’m  because I say so!”
- ✓  “I’m  because  says so”
- ✓  “I’m  because  says so”

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

42

42

Chứng thư không hợp lệ

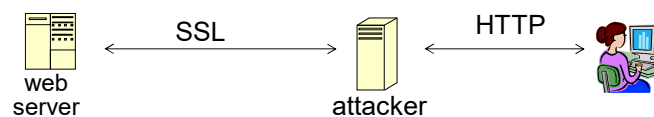


43

43

Tấn công sslstrip

- Tấn công sslstrip: ngăn cản nâng cấp từ HTTP → HTTPS



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

44

44

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

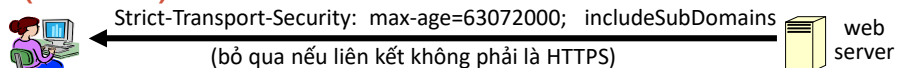


- Xóa cookie bằng cách chèn header “set-cookie” → người dùng bắt buộc đăng nhập lại
 - Phần lớn không phát hiện HTTPS → HTTP

45

45

Phòng chống: Strict Transport Security (HSTS)



- HSTS: Một trường trên tiêu đề của HTTP Response, yêu cầu trình duyệt luôn kết nối với máy chủ qua HTTPS
 - Từ chối các truy cập qua HTTP hoặc chứng thư số của server không hợp lệ
 - Yêu cầu toàn bộ nội dung website phải được truy cập qua 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

46

46

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">
```

```

```

- Nguy cơ: Kẻ tấn công thay thế những nội dung này
- Cảnh báo trên trình duyệt



Chrome(Các phiên bản cũ):

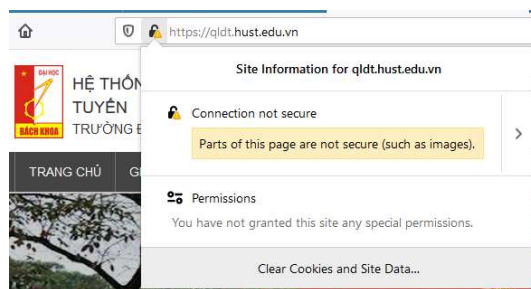


- Chính sách của Chrome: chặn CSS, mã Javascript, thẻ <frame>

47

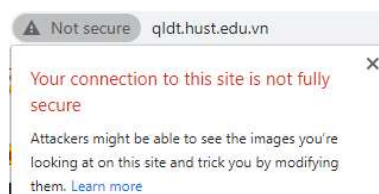
47

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



Mozilla Firefox

Chrome



48

48

CSP: upgrade-insecure-requests

- Các trang truy cập bằng HTTPS có thể trộn lẫn các URL của tài nguyên có thể truy cập bằng HTTP

Ví dụ: ``

- Thêm vào tiêu đề trong HTTP Response

Content-Security-Policy: upgrade-insecure-requests

→ Trình duyệt tự động chuyển URL sử dụng HTTP sang HTTPS, ngay cả với các tài nguyên ngoài (khác tên miền) trừ liên kết với thẻ `<a>`

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

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

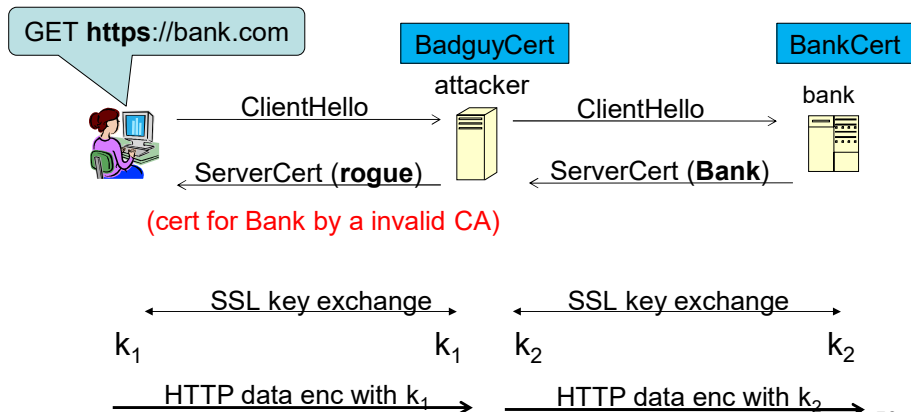
Luôn sử dụng URL tương đối: ``

49

49

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



50

50

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

51

51

Phòng chống chứng thư số giả mạo

- Giải pháp cũ: HTTP Public Key Pinning(HPKP)
 - 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
 - Hiện nay được khuyến cáo không nên sử dụng
- 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)

52

52

Tấn công phishing

- Homograph attack: Lợi dụng hình dáng giống nhau của một số ký tự. Ví dụ: vvesternbank, paypai, paypal, 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)

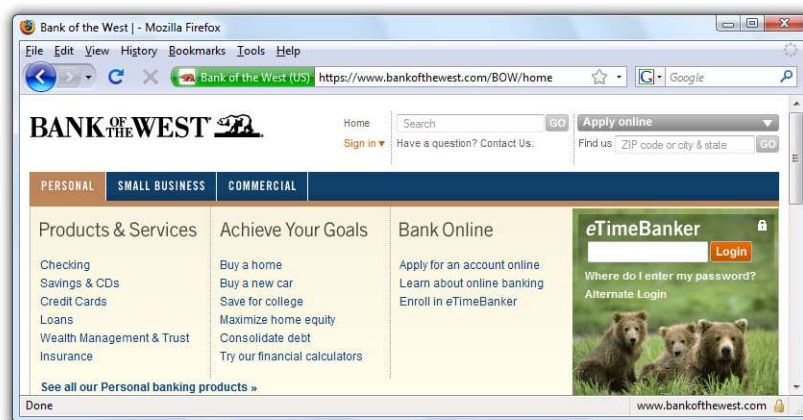
 Vietnam Joint Stock Commercial Bank For Industry and Trade [VN] | <https://ebanking.vietinbank.vn/ipay/login.do>

53

53

Homograph attack – Ví dụ

- Website bị giả mạo

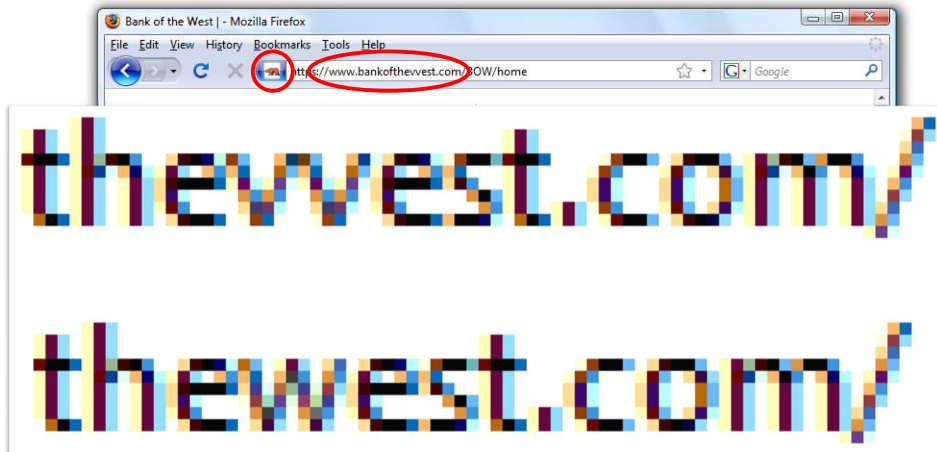


54

54

Homograph attack

- Website giả mạo



55

55

Semantic attack (ví dụ)



56

56

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



57

57

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
- Introduction to Computer Security, Carnegie Mellon University

58

58