CRYPTONET LABS

Web App Security

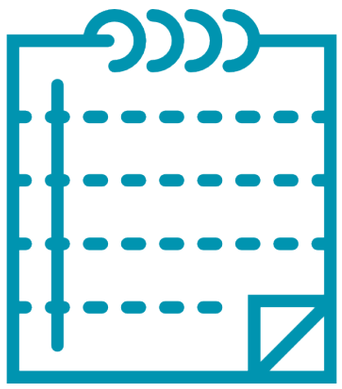
Vulnerability Assessment e Penetration Test di web app

Luca Capacci

Bologna, 13/11/2017, 20/11/2017

AGENDA

- Vulnerability Assessment & Penetration Test
- OWASP Top Ten Web (e oltre)
- Tools



Target & Tools

Target

php.testsparker.com

google-gruyere.appspot.com

Burp Suite

https://portswigger.net/burp/download.html

https://support.portswigger.net/customer/portal/articles/1783055-configuring-your-browser-to-work-with-burp

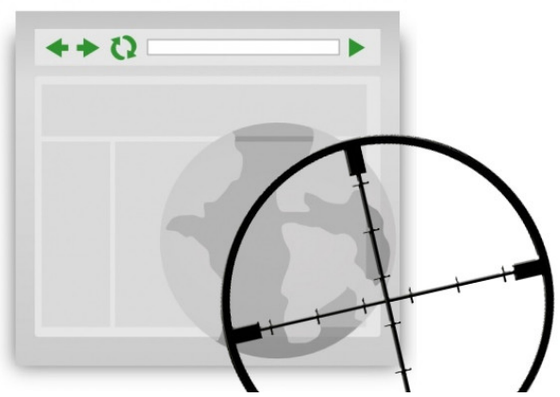
https://support.portswigger.net/customer/portal/articles/1783075-Installing_Installing%20CA%20Certificate.html

Nmap

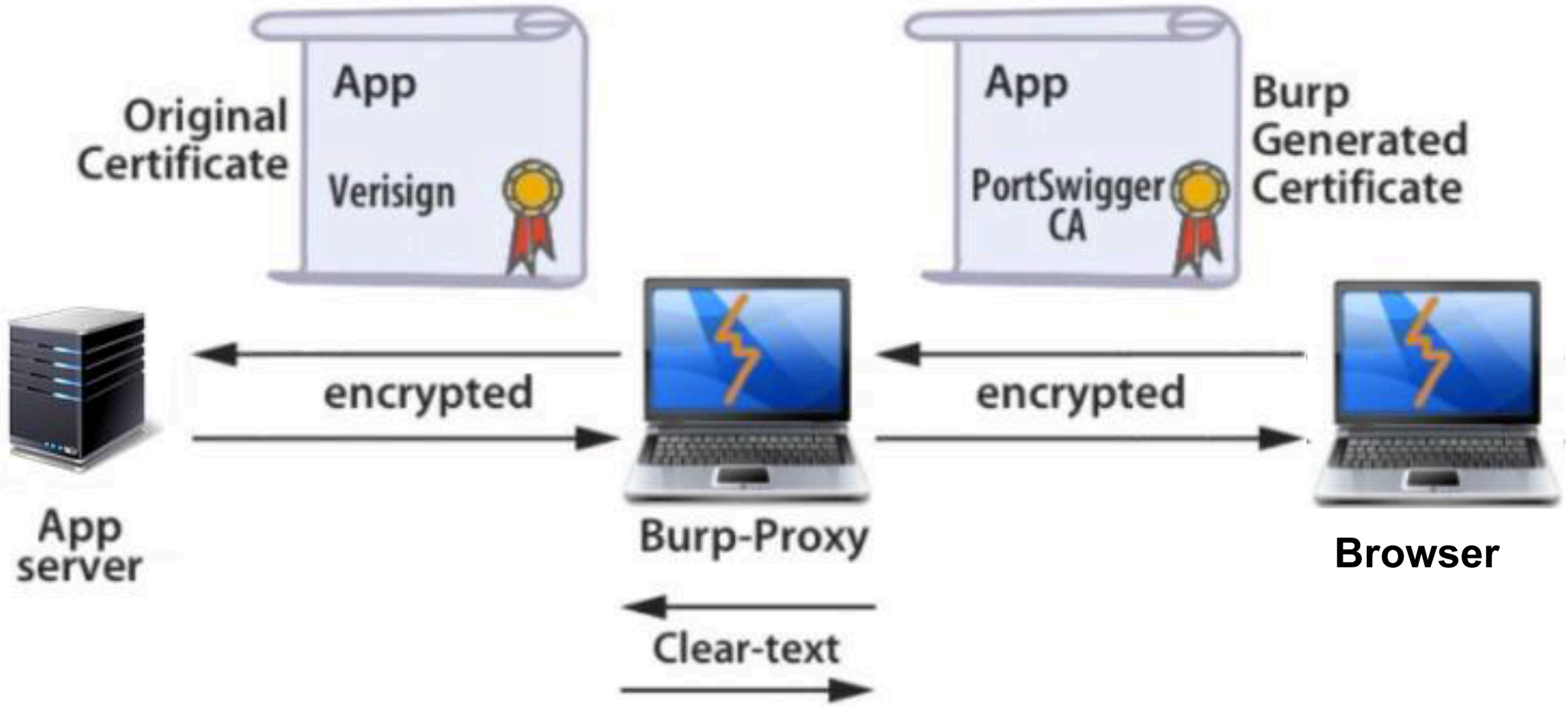
https://nmap.org/download.html

SqlMap

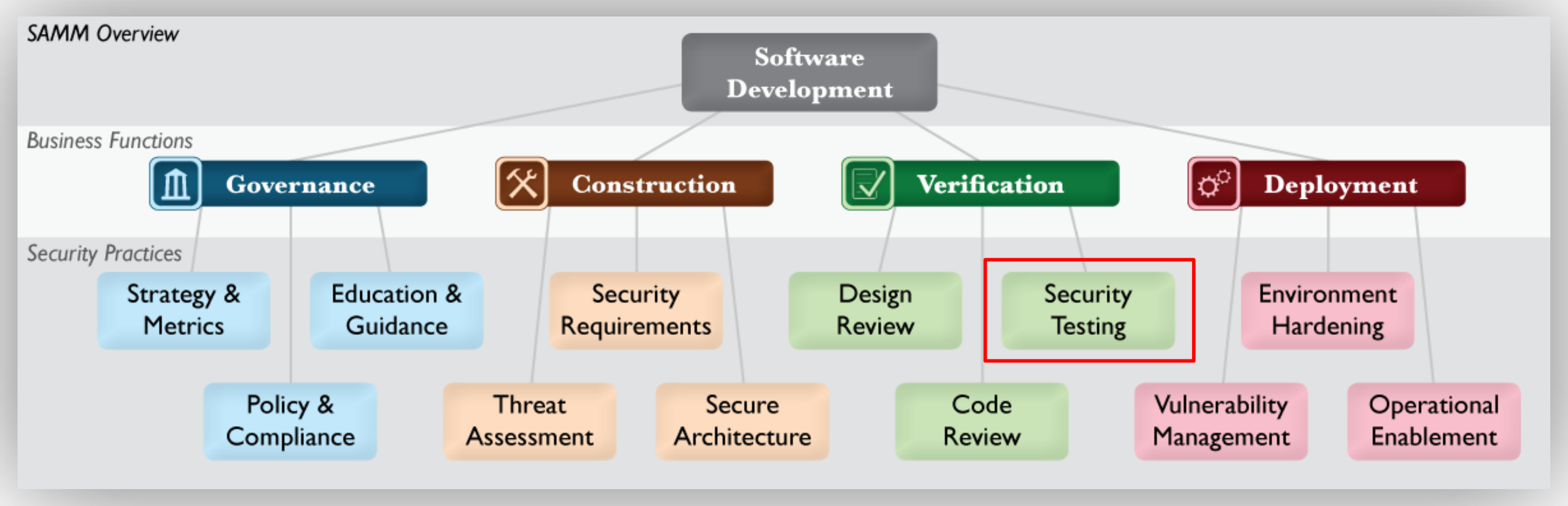
https://github.com/sqlmapproject/sqlmap



Burp Suite


Original Certificate
App
Verisign
App
PortSwigger CA
Burp Generated Certificate
encrypted
encrypted
App server
Burp-Proxy
Browser
Clear-text

Security Testing



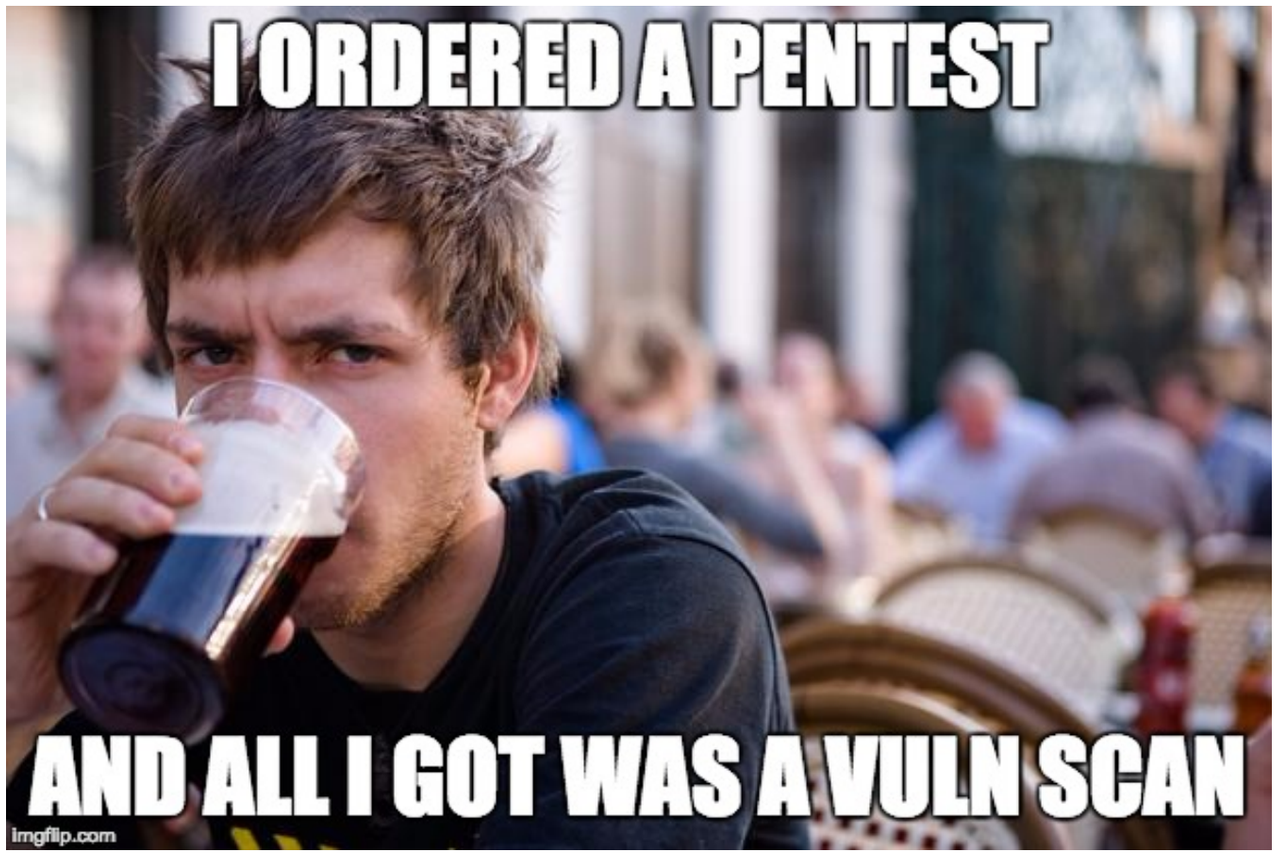
Security Testing is a process intended to reveal flaws in the security mechanisms of an information system

VA vs PT

https://www.htbridge.com/blog/how_to_keep_your_website_safe_in_2015.html

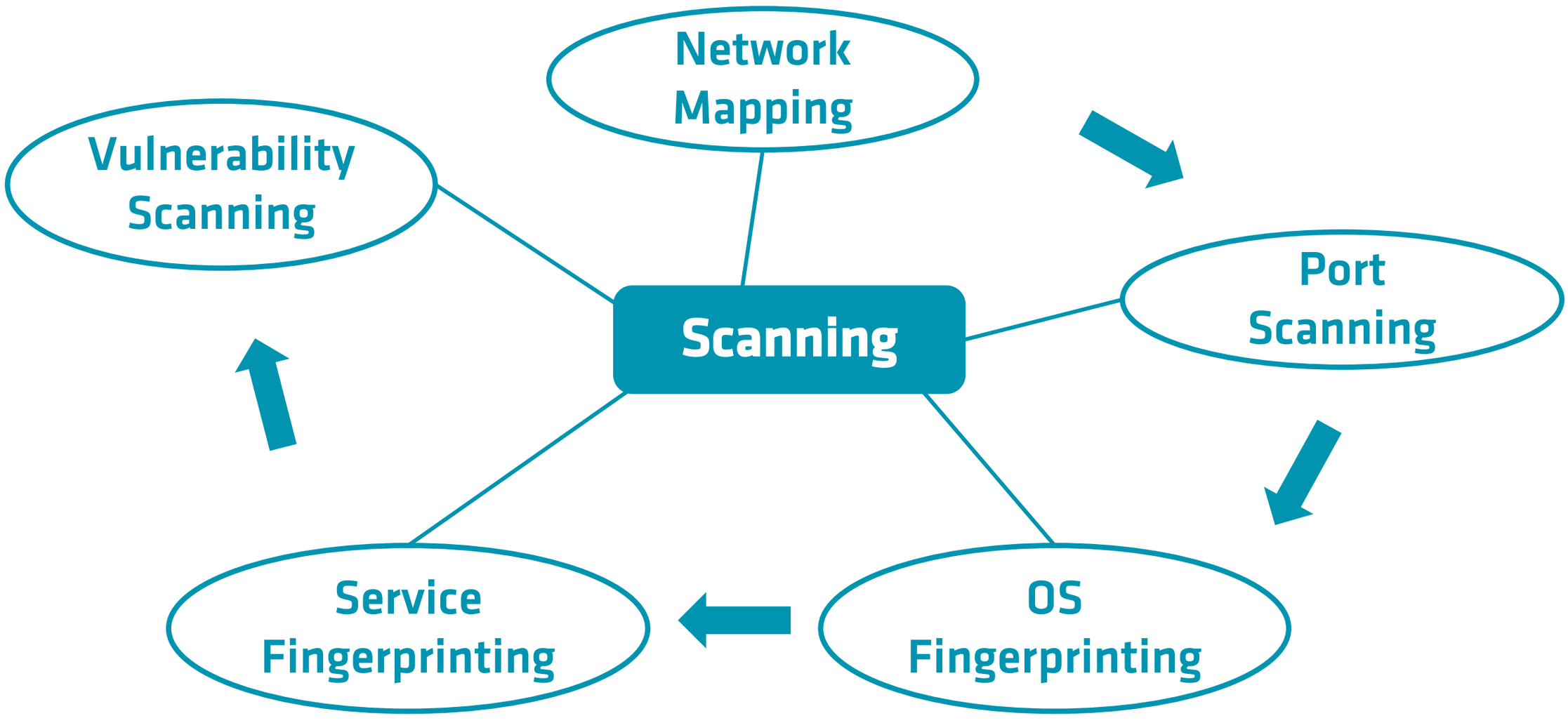
VA vs PT

- **Vulnerability Assessment (VA)**
  - Vulnerability assessments discover which vulnerabilities are possibly present, but don't try to exploit them
  - Typically automated
- **Penetration Test (PT)**
  - Penetration tests simulate hacker attempts to get into a system to find exploitable flaws and measure the severity of each
  - Conducted by human beings

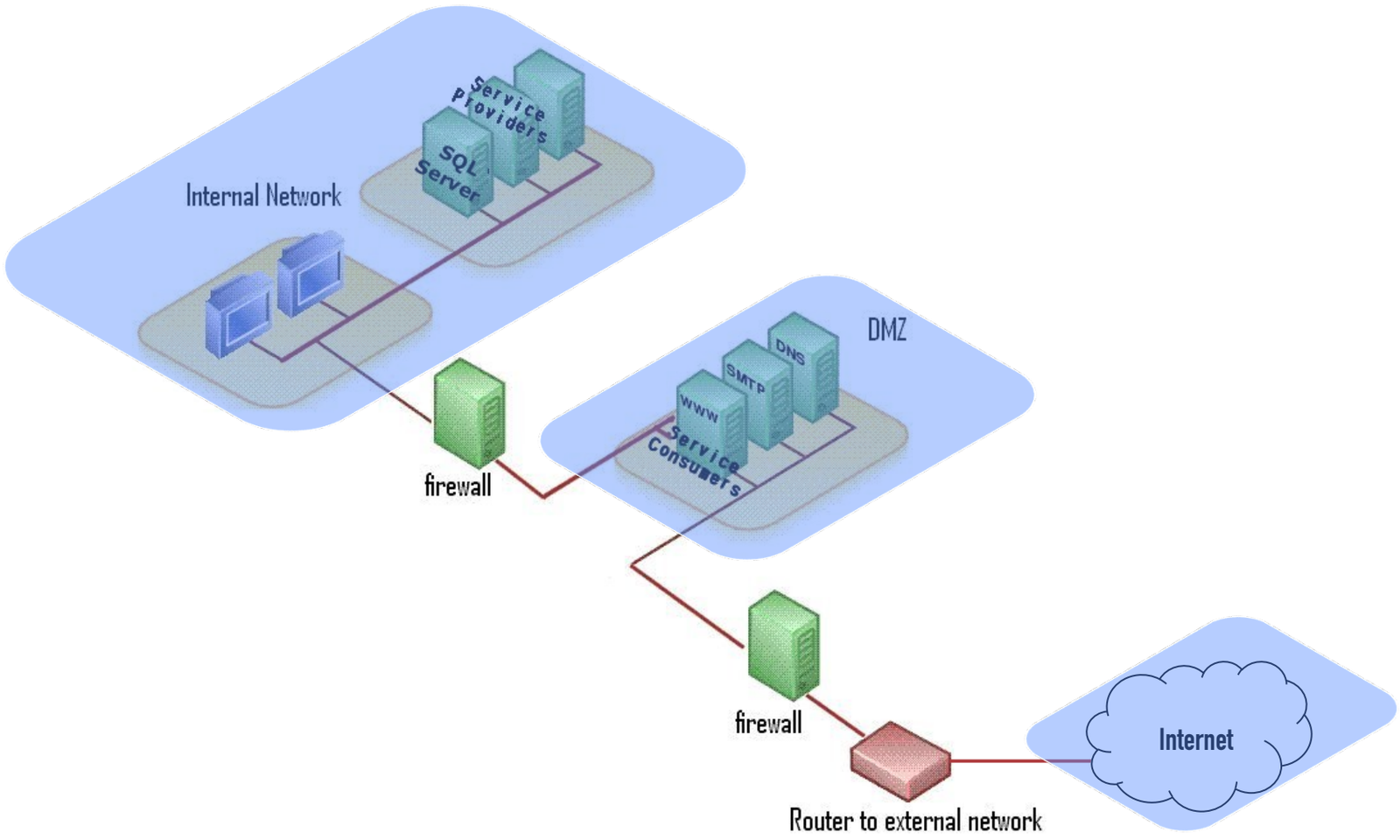


https://www.alienvault.com/blogs/security-essentials/penetration-testing-vs-vulnerability-scanning-whats-the-difference

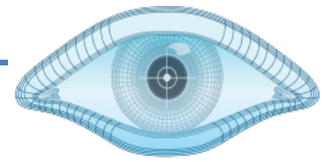
VAPT



Discovery


Internal Network
Service Providers
SQL Server
firewall
DMZ
DNS
SMTP
WWW
Service Consumers
firewall
Router to external network
Internet

Discovery



```
nmap -sS -sU -O -sV mysite.com

...

PORT      STATE SERVICE    VERSION
123/udp   open  ntp        udp-response ttl 57 NTP v4
22/tcp    open  ssh        OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.3
80/tcp    open  http       Apache httpd 2.4.7 ((Ubuntu))
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.2 - 3.19

...
```

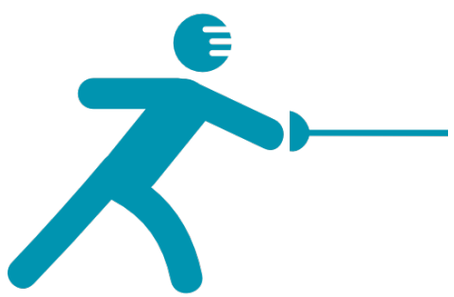

Vulnerability Scanning

In generale

Controlli differenti a seconda del tipo di servizio

Controlli comuni

- Versioni dei software
- Cifratura delle comunicazioni
- Robustezza dei meccanismi di autenticazione



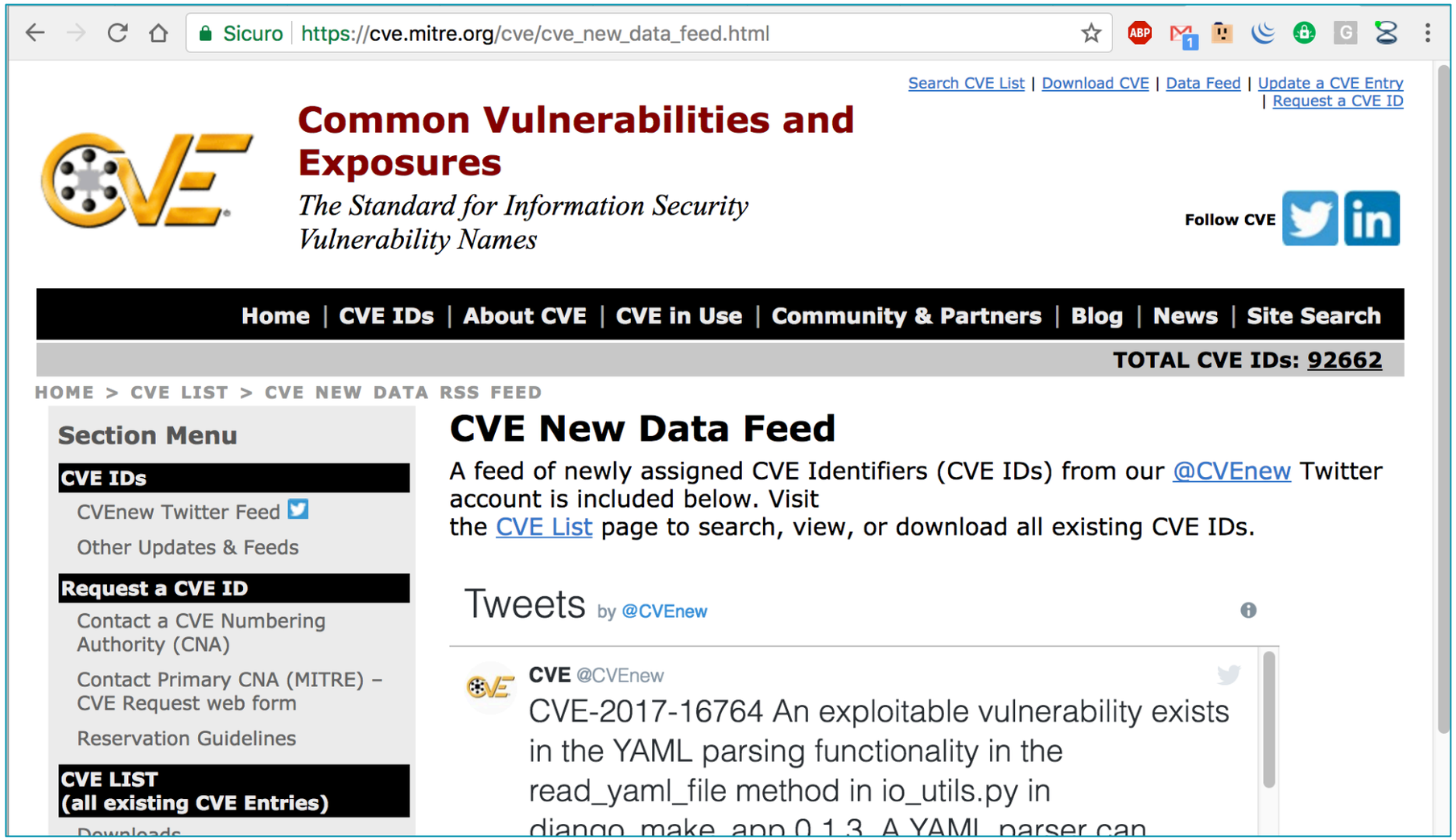
Web application VAPT

Analisi dei servizi web

Versioni di software vulnerabili

CVE (Common Vulnerabilities and Exposures)

- CVE is a list of common identifiers for publicly known cyber security vulnerabilities



Versioni di software vulnerabili

https://www.cvedetails.com/

Sicuro | https://www.cvedetails.com/vulnerability-list/vendor_id-45/produ

CVE Details
The ultimate security vulnerability datasource

Search
(e.g.: CVE-2009-1234 or 2010-1234 or 20101234)
View CVE

Log In Register

Vulnerability Feeds & WidgetsNew www.itsecdb.com

Switch to https://
Home
Browse :
Vendors
Products
Vulnerabilities By Date
Vulnerabilities By Type
Reports :
CVSS Score Report
CVSS Score Distribution
Search :
Vendor Search
Product Search
Version Search
Vulnerability Search
By Microsoft References
Top 50 :
Vendors
Vendor Cvss Scores
Products
Product Cvss Scores
Versions
Other :
Microsoft Bulletins
Bugtraq Entries
CWE Definitions

Apache » Tomcat » 9.0.0 M1 : Security Vulnerabilities

Cpe Name:*cpe:/a:apache:tomcat:9.0.0:m1*
CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9
Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending
Copy Results Download Results

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2017-12617	434		Exec Code	2017-10-03	2017-10-23	6.8	None	Remote	Medium	Not required	Partial	Partial	Partial

When running Apache Tomcat versions 9.0.0.M1 to 9.0.0, 8.5.0 to 8.5.22, 8.0.0.RC1 to 8.0.46 and 7.0.0 to 7.0.81 with HTTP PUTs enabled (e.g. via setting the readonly initialisation parameter of the Default servlet to false) it was possible to upload a JSP file to the server via a specially crafted request. This JSP could then be requested and any code it contained would be executed by the server.

2	CVE-2017-7675	22		Dir. Trav. Bypass	2017-08-10	2017-11-03	5.0	None	Remote	Low	Not required	Partial	None	None

The HTTP/2 implementation in Apache Tomcat 9.0.0.M1 to 9.0.0.M21 and 8.5.0 to 8.5.15 bypassed a number of security checks that prevented directory traversal attacks. It was therefore possible to bypass security constraints using a specially crafted URL.

3	CVE-2017-7674	345			2017-08-10	2017-11-03	4.3	None	Remote	Medium	Not required	None	Partial	None

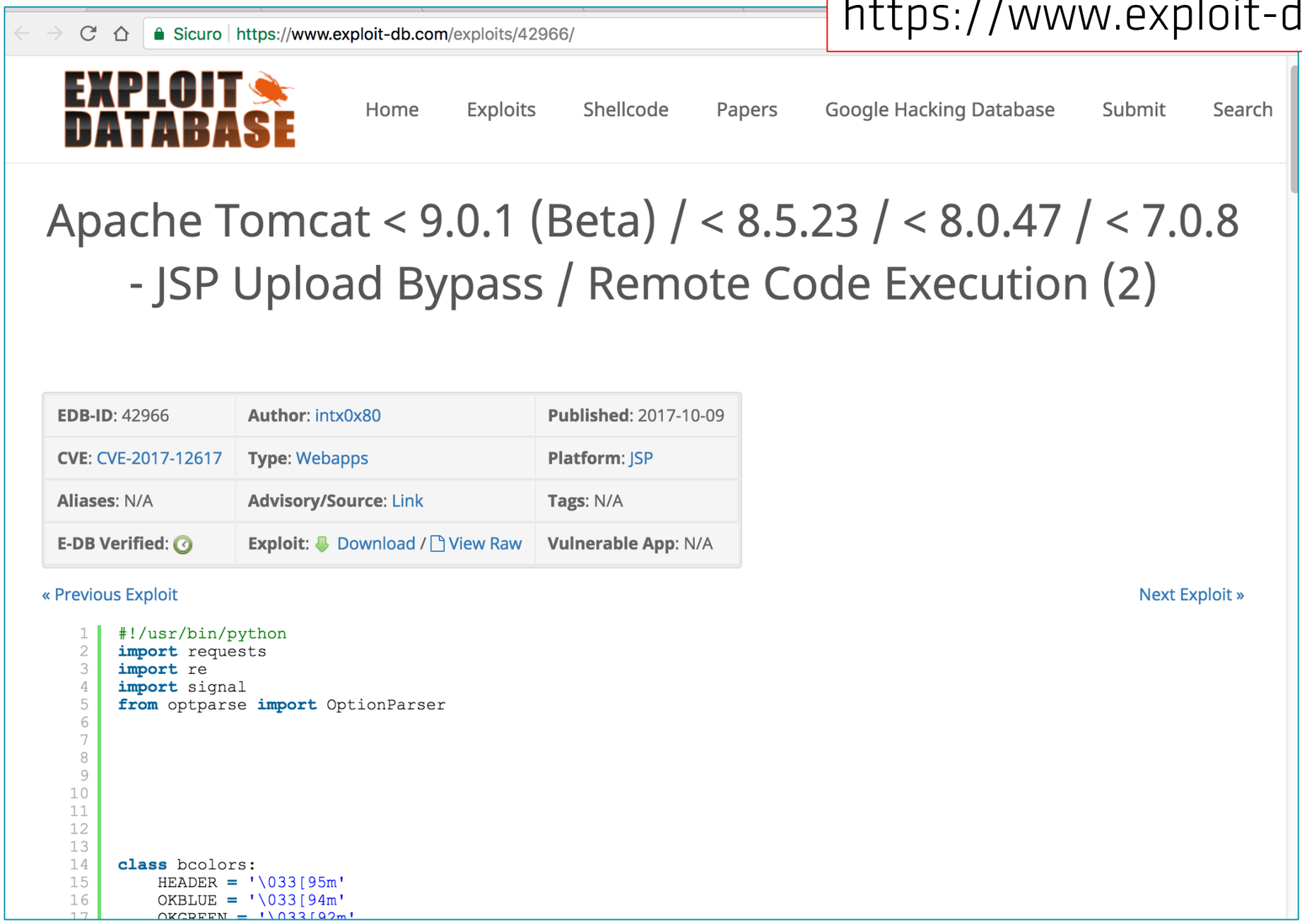
The CORS Filter in Apache Tomcat 9.0.0.M1 to 9.0.0.M21, 8.5.0 to 8.5.15, 8.0.0.RC1 to 8.0.44 and 7.0.41 to 7.0.78 did not add an HTTP Vary header indicating that the response varies depending on Origin. This permitted client and server side cache poisoning in some circumstances.

4	CVE-2017-5664	254			2017-06-06	2017-11-03	5.0	None	Remote	Low	Not required	None	Partial	None

The error page mechanism of the Java Servlet Specification requires that, when an error occurs and an error page is configured for the error that

Versioni di software vulnerabili

https://www.exploit-db.com/



Laboratorio – Discovery

Test 1

```
sudo nmap -sV -O php.testsparker.com
```

Test 2

```
sudo nmap -sV -O php.testsparker.com -Pn
```

Test 3

In base ai risultati di nmap, ricerca di vulnerabilità note dei software rilevati, su www.cvedetails.com e exploit su www.exploit-db.com

OWASP

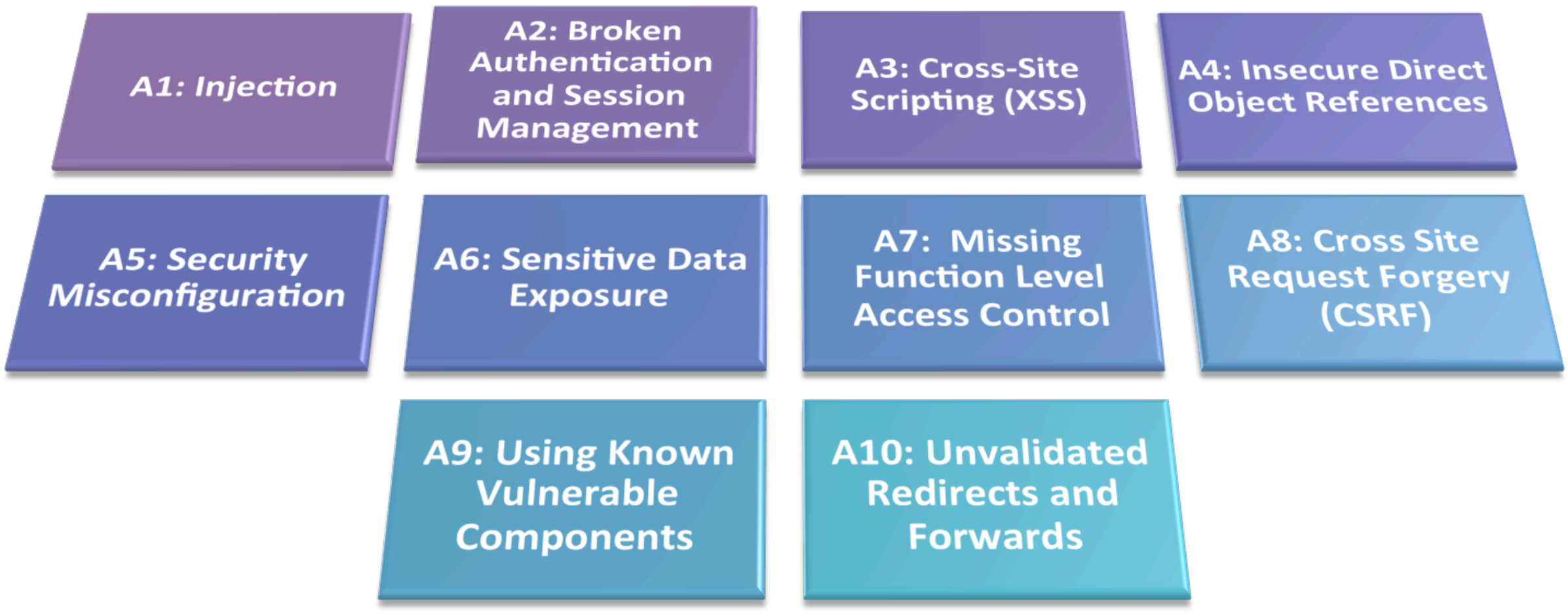
Open Web Application Security Project

- Since 2001
- Not-for-profit charitable organization
- Dedicated to enabling organizations to conceive, develop, acquire, operate, and maintain applications that can be trusted

OWASP
Open Web Application Security Project

https://www.owasp.org/index.php/Marketing/Resources

OWASP Top Ten 2013



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

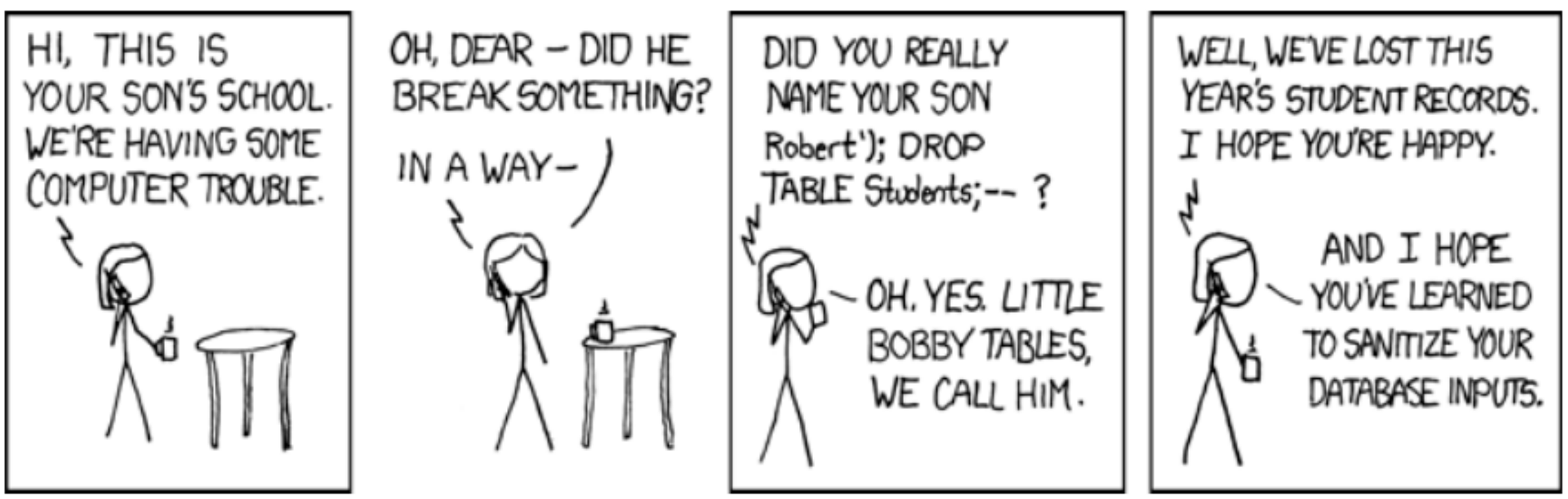
OWASP Top Ten 2017 RC2

OWASP Top 10 2013	±	OWASP Top 10 2017
A1 – Injection	→	A1:2017 – Injection
A2 – Broken Authentication and Session Management	→	A2:2017 – Broken Authentication and Session Management
A3 – Cross-Site Scripting (XSS)	↘	A3:2013 – Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017 – XML External Entity (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017 – Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017 – Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017 – Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	✗	A8:2017 – Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017 – Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	✗	A10:2017 – Insufficient Logging & Monitoring [NEW, Comm.]

https://github.com/OWASP/Top10/blob/master/2017/OWASP%20Top%2010%202017%20RC2%20Final.pdf

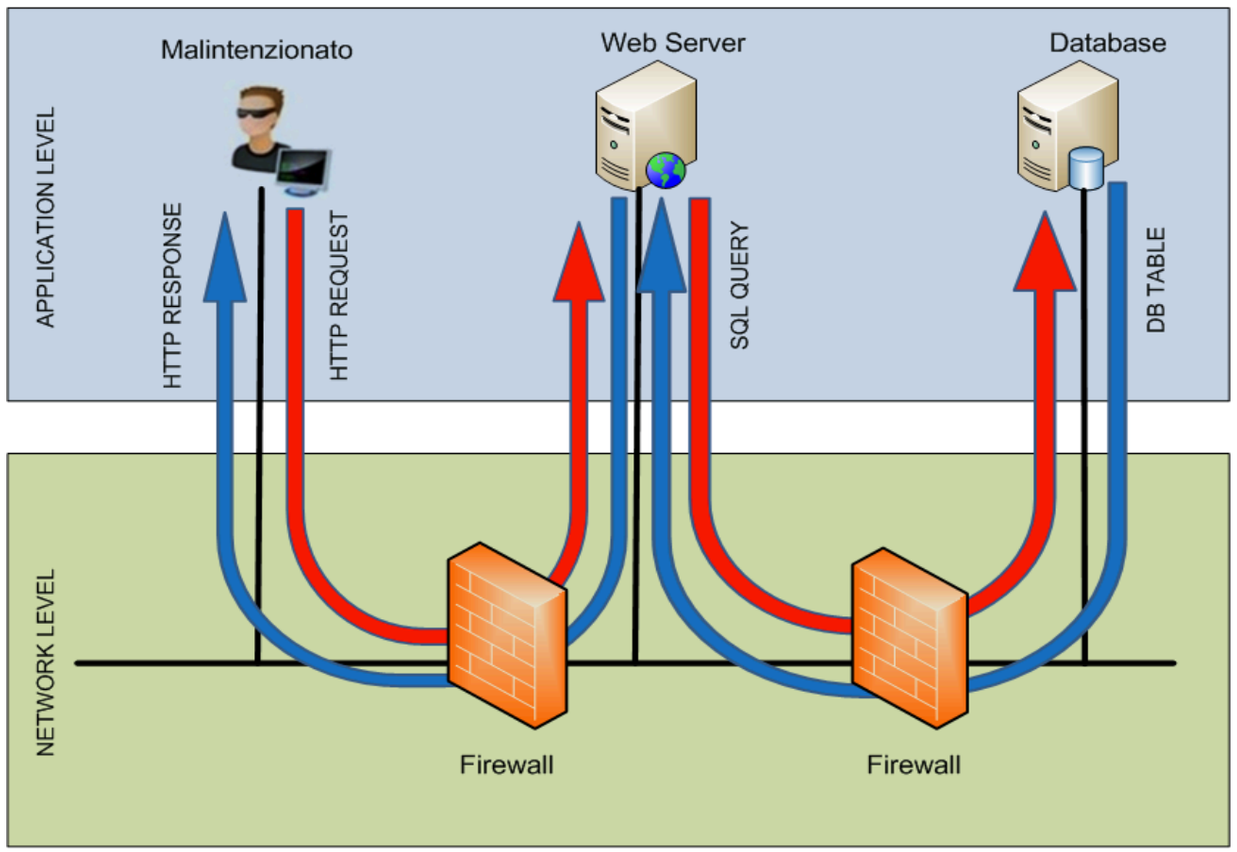
A1 – Injection

Untrusted data is sent to an **interpreter** as part of a command or query. The data can trick the interpreter into **executing unintended commands** or **accessing data** without proper authorization.



https://xkcd.com/327/

A1 – Injection (SQL Injection)



1

Username
Password
Submit

2-3-4

```
String query = "SELECT * FROM accounts
   WHERE acct='" + httpParam + "'";
```

↓

"SELECT * FROM accounts WHERE acct='' OR 1=1--'"

5-6

Account Summary

Acct:5424-6066-2134-4334
Acct:4128-7574-3921-0192
Acct:5424-9383-2039-4029
Acct:4128-0004-1234-0293

1. Application presents a form to the attacker
2. Attacker sends an attack in the form data
3. Application forwards attack to the database in a SQL query
4. Database runs query containing attack and sends results back to application
5. Application processes data as normal and sends results to the user

A1 – Injection (SQL Injection)

Blind SQL Injection

L'applicazione vulnerabile non scrive direttamente i risultati su di una pagina web, ma mostra una pagina di errore generica.

Per estrarre dei dati bisogna fare un grande numero di richieste al server cercando di ottenere delle risposte vero/falso, discriminando i risultati in base a:

- Tempi di risposta, usando le funzioni SLEEP (e simili) → **Time Based**
- Differenze nelle pagine di errore → **Boolean Based**

https://www.acunetix.com/websitesecurity/sql-injection2/

A1 – Injection

Impact

- Entire database can usually be read or modified
- May also allow full database schema, or account access, or even OS level access

Recommendations

- Encode all user input before passing it to the interpreter
- Use an interface that supports bind variables (e.g., prepared statements, or stored procedures)
- Always minimize database privileges to reduce the impact of a flaw

Laboratorio – SQL Injection (1)

URL di partenza

```
http://php.testsparker.com/artist.php?id=test
```

Test 1

```
http://php.testsparker.com/artist.php?id=1
```

Test 2

```
http://php.testsparker.com/artist.php?id=1%20OR%201=1
```

Test 3

```
http://php.testsparker.com/artist.php?id=1 OR 1=(SELECT
SLEEP(10))
```


Laboratorio – SQL Injection (2)

Test 4

```
http://php.testsparker.com/artist.php?id=1 OR 1=(SELECT COUNT(*)
FROM information_schema.SCHEMATA)
```

Test 5

```
http://php.testsparker.com/artist.php?id=1 OR 2=(SELECT COUNT(*)
FROM information_schema.SCHEMATA)
```

……..

Test 9

```
http://php.testsparker.com/artist.php?id=1 OR 6=(SELECT COUNT(*)
FROM information_schema.SCHEMATA)
```


Laboratorio – SQL Injection (3)

Test 10

Inserimento in SqlMap dell'URL `http://php.testsparker.com/artist.php?id=test` ed exploitation automatica della vulnerabilità:

- `sqlmap -u "http://php.testsparker.com/artist.php?id=test" --level=3 -v 3`
- `sqlmap -u "http://php.testsparker.com/artist.php?id=test" --level=3 -v 3 --tables`

Laboratorio – Command Injection

URL di partenza contenente il form su cui eseguire i test

`http://php.testsparker.com/nslookup.php`

Test 1

192.168.0.1

Test 2

& DIR &

Test 2

& CD.. & DIR &

Test 3

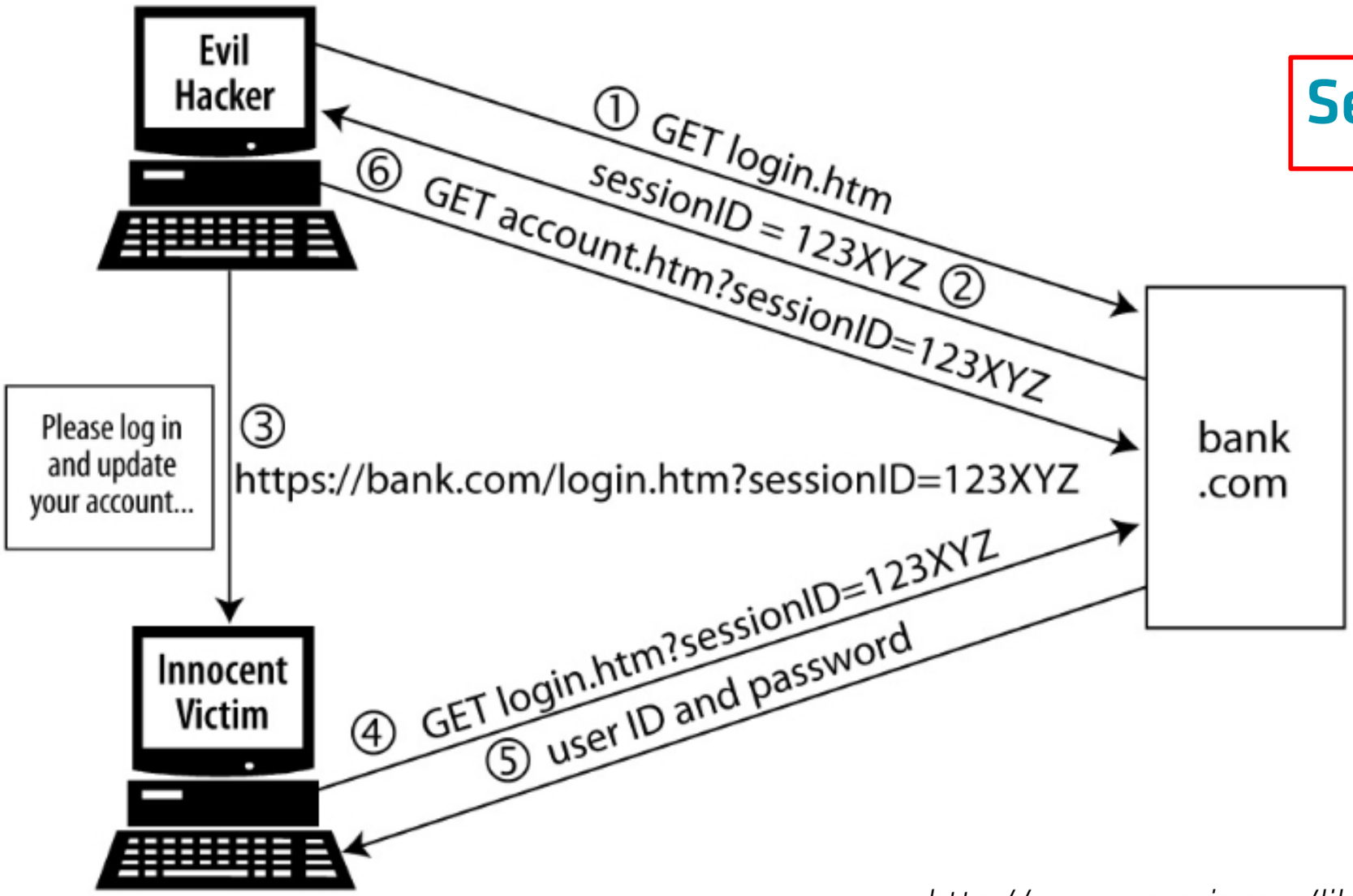
& TYPE nslookup.php &

A2 – Broken Authentication and Session Management

Application functions related to authentication and session management are not implemented correctly.

Attackers can **compromise passwords, keys, or session tokens** to assume other users' identities.

A2 – Broken Authentication and Session Management



Session Fixation

http://www.maravis.com/library/session-fixation-attack/

A2 – Broken Authentication and Session Management

Email di conferma vulnerabile

1. L'attaccante si registra al sito

2. Il server assegna all'attaccante l'id **2000** e gli invia una mail di conferma con il link:
   `http://www.site.com/userconfirmation/2000`

3. L'attaccante visita il link ricevuto

4. Il server risponde con un cookie di sessione associato all'utente 2000

5. L'attaccante prova a visitare l'URL:
   `http://www.site.com/userconfirmation/1999`

6. Il server risponde con un cookie di sessione associato all'utente 1999

A2 – Broken Authentication and Session Management

Impact

- User accounts or user sessions compromised

Recommendations

- Use the standard session id provided by your container (JESSIONID…)
- Use HTTPS and set HttpOnly and Secure flags on cookies
- Verify that logoff actually destroys the session
- **Beware the side-doors:** change my password, forgot my password, secret question, logout, email address verification…

A2 – Broken Authentication and Session Management

Cookie Flags

- HttpOnly
- The cookie cannot be accessed through client side script
- Bypass via XST (Cross-site tracing) → Disable HTTP TRACE
- Secure:
- The cookie will only be sent over an HTTPS connection

```
Set-Cookie: <name>=<value>[; <Max-Age>=<age>]
[; expires=<date>][; domain=<domain_name>]
[; path=<some_path>][; secure][; HttpOnly]
```


Laboratorio – Session Fixation (1)

URL con form di login

http://php.testsparker.com/auth/login.php

Test 1

Visitare l'URL e analizzare con Burp la generazione del cookie di sessione

Test 2

Inserire le credenziali e visualizzare in Burp la mancata generazione di un nuovo cookie di sessione

Laboratorio – Session Fixation (2)

Test 3:

Scenario in cui attaccante e vittima condividono lo stesso pc

Attaccante:

- Apertura dell'URL `http://php.testsparker.com/auth/login.php`
- Copia in un file di testo del cookie generato dal server, salvataggio del file su una chiavetta USB ed allontanamento dal pc

Vittima (dallo stesso pc appena usato dall'attaccante):

- Apertura dell'URL `http://php.testsparker.com/auth/login.php`
- Login

Attaccante (da un altro pc):

- Apertura dell'URL `http://php.testsparker.com/auth/internal.php`, ottenendo un redirect alla pagina di login
- Impostazione nel proprio browser del cookie salvato in precedenza
- Apertura dell'URL `http://php.testsparker.com/auth/internal.php`, ottenendo la stessa pagina mostrata alla vittima autenticata

Laboratorio – Cookie Flags

URL con form di login

`http://php.testsparker.com/auth/login.php`

Test 1

Visitare l'URL e analizzare con Burp l'assenza di flag sul cookie di sessione generato dal server

Test 2

Visitare **`https://studenti.unibo.it`** e analizzare con Burp la presenza di flag sul cookie di sessione generato dal server

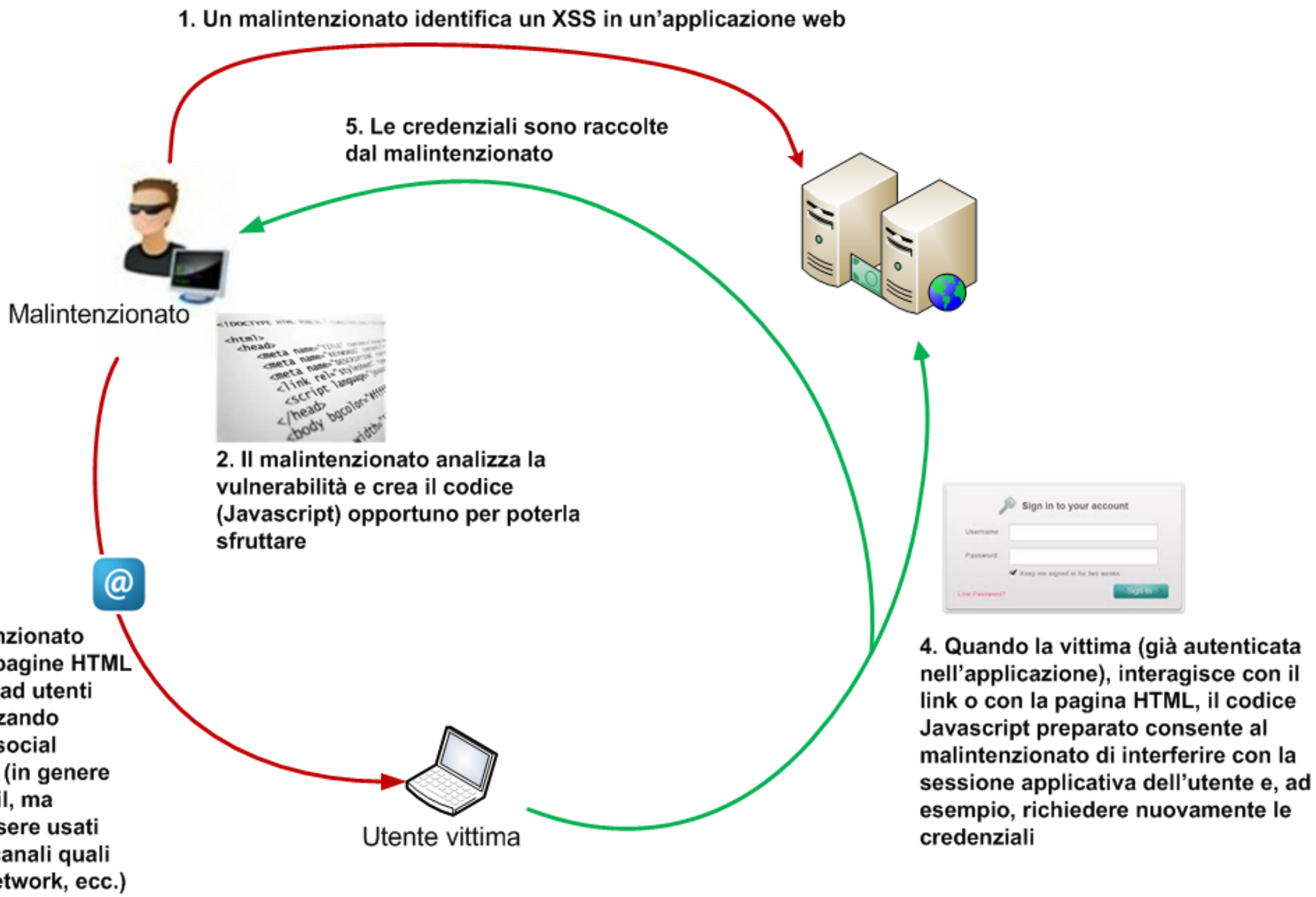
A3 – Cross-Site Scripting

An application takes untrusted data and writes it to a web page without proper validation or escaping.

XSS allows attackers to **execute scripts in the victim's browser** which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

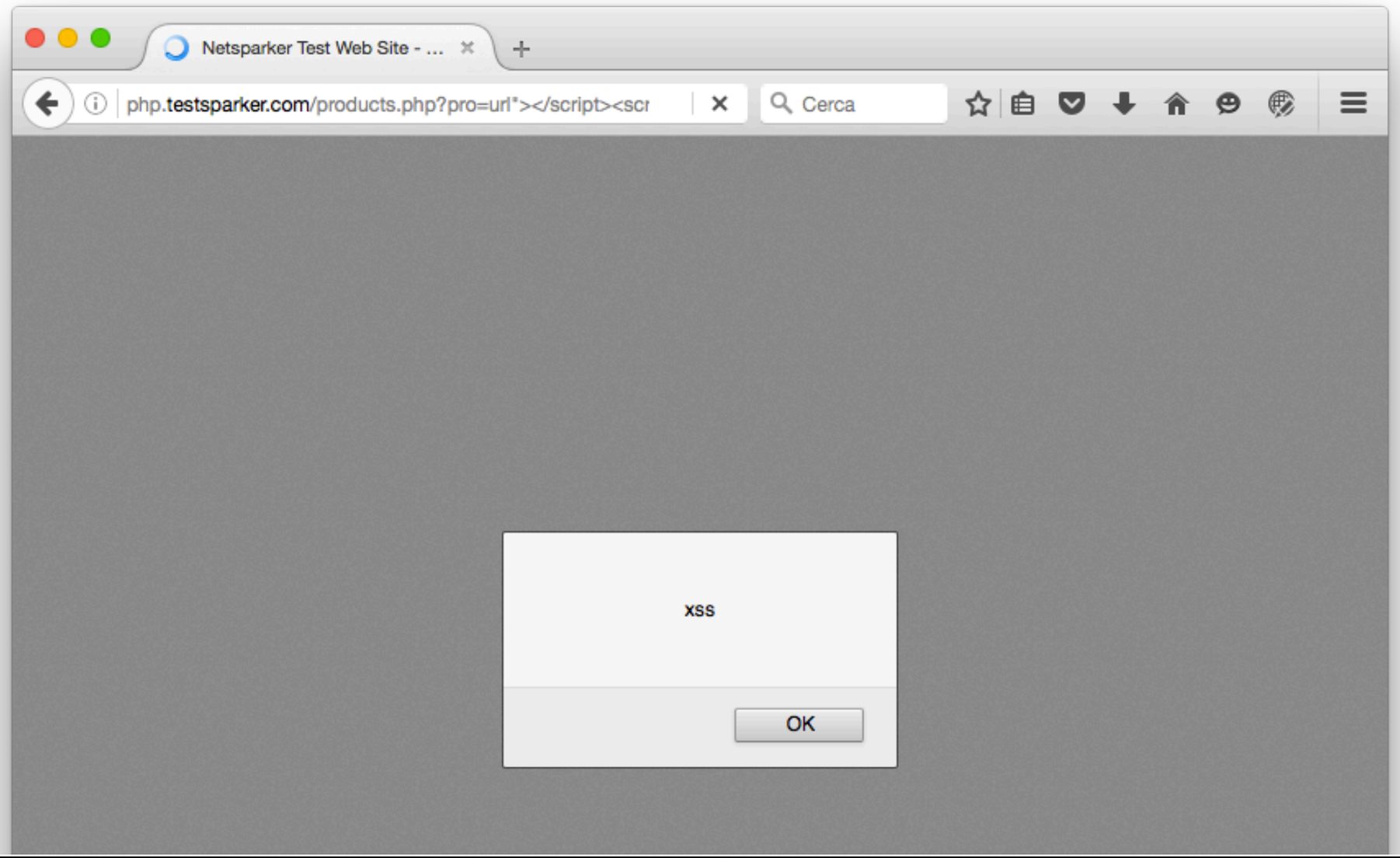
http://memeshappen.com/meme/futurama-fry/script-alert-xss-script-60649

A3 – Cross-Site Scripting (reflected)



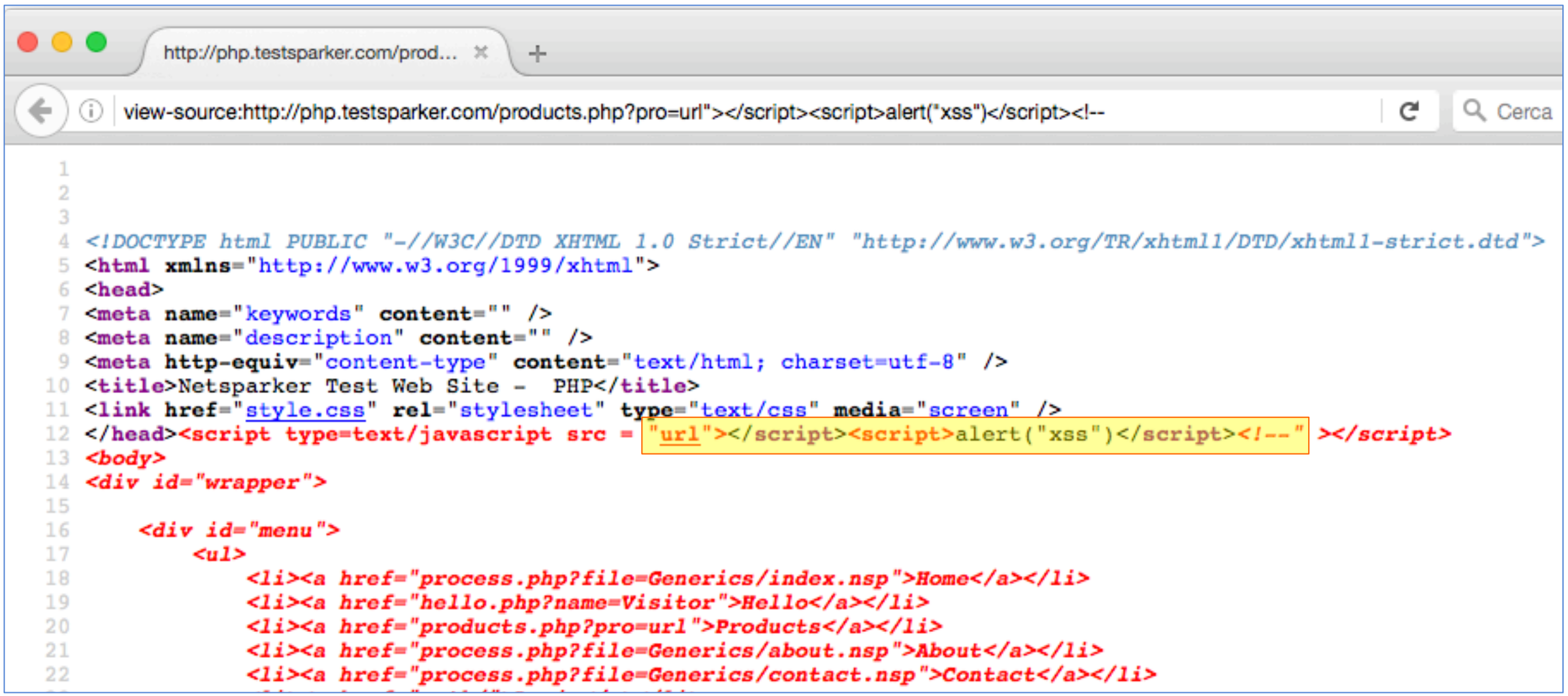
A3 – Cross-Site Scripting (reflected)

http://php.testsparker.com/products.php?pro=**url"></script><script>alert("xss")</script><!--"**

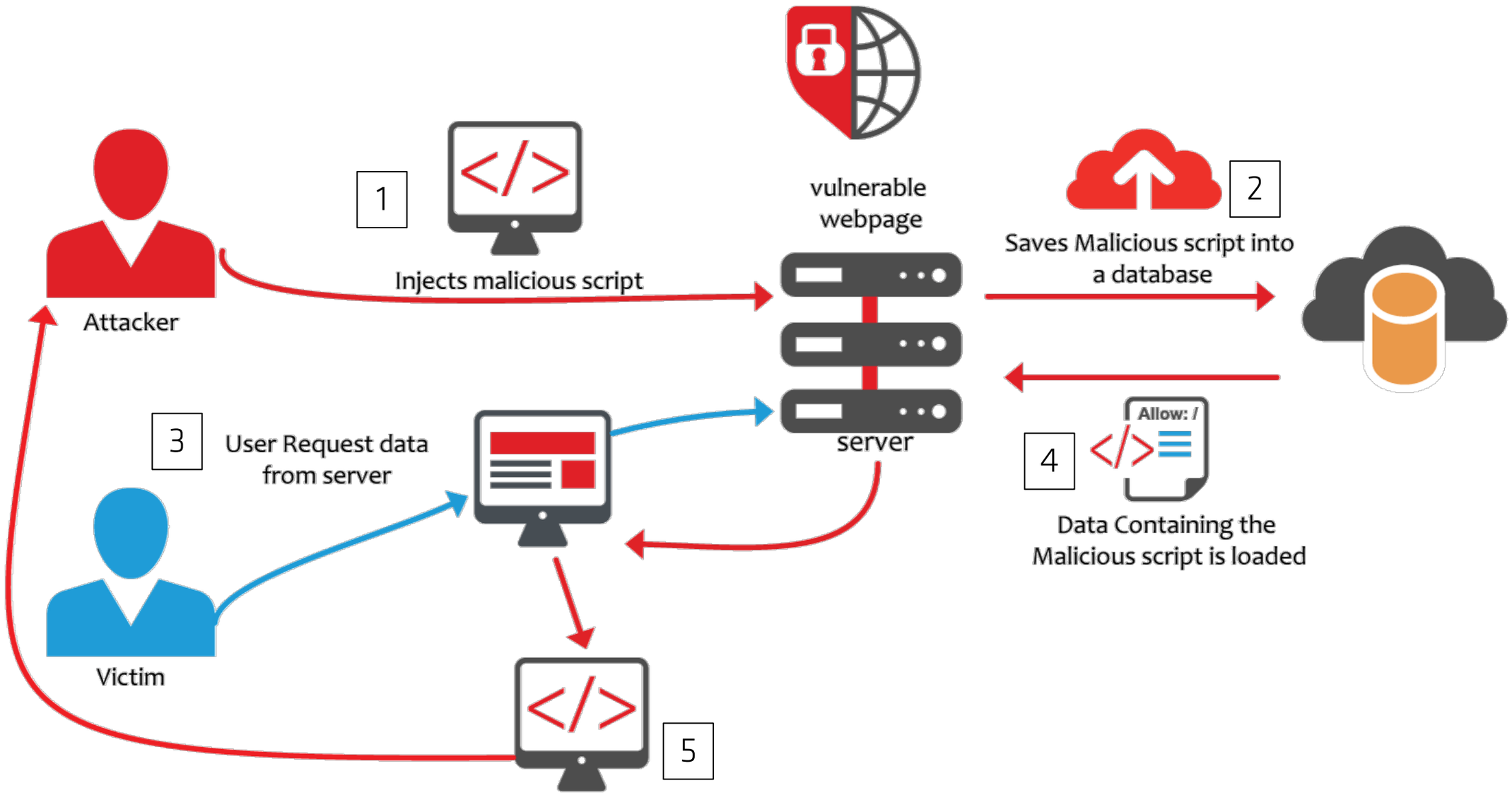


A3 – Cross-Site Scripting (reflected)

http://php.testsparker.com/products.php?pro=**url"></script><script>alert("xss")</script><!--**



A3 – Cross-Site Scripting (stored)



http://www.acunetix.com/blog/articles/blind-xss/

A3 – Cross-Site Scripting

Impact

- Steal user's session, steal sensitive data, rewrite web page, redirect user to phishing or malware site

Recommendations

- Output encode all user supplied input
- Perform 'white list' input validation on all user input to be included in page
- Use tested and well-known third party libraries to filter input data

Laboratorio – Cross-Site Scripting Reflected (1)

URL di partenza

```
http://php.testsparker.com/hello.php?name=Visitor
```

Test 1

```
http://php.testsparker.com/hello.php?name=Visitor"<'>
```

Test 2

```
http://php.testsparker.com/hello.php?name=Visitor<script>alert('xss')</script>
```


Laboratorio – Cross-Site Scripting Reflected (2)

URL di partenza

```
http://php.testsparker.com/products.php?pro=url
```

Test 1

```
http://php.testsparker.com/products.php?pro=url"></script><scrip
t>alert("xss")</script><!--
```

Test 2

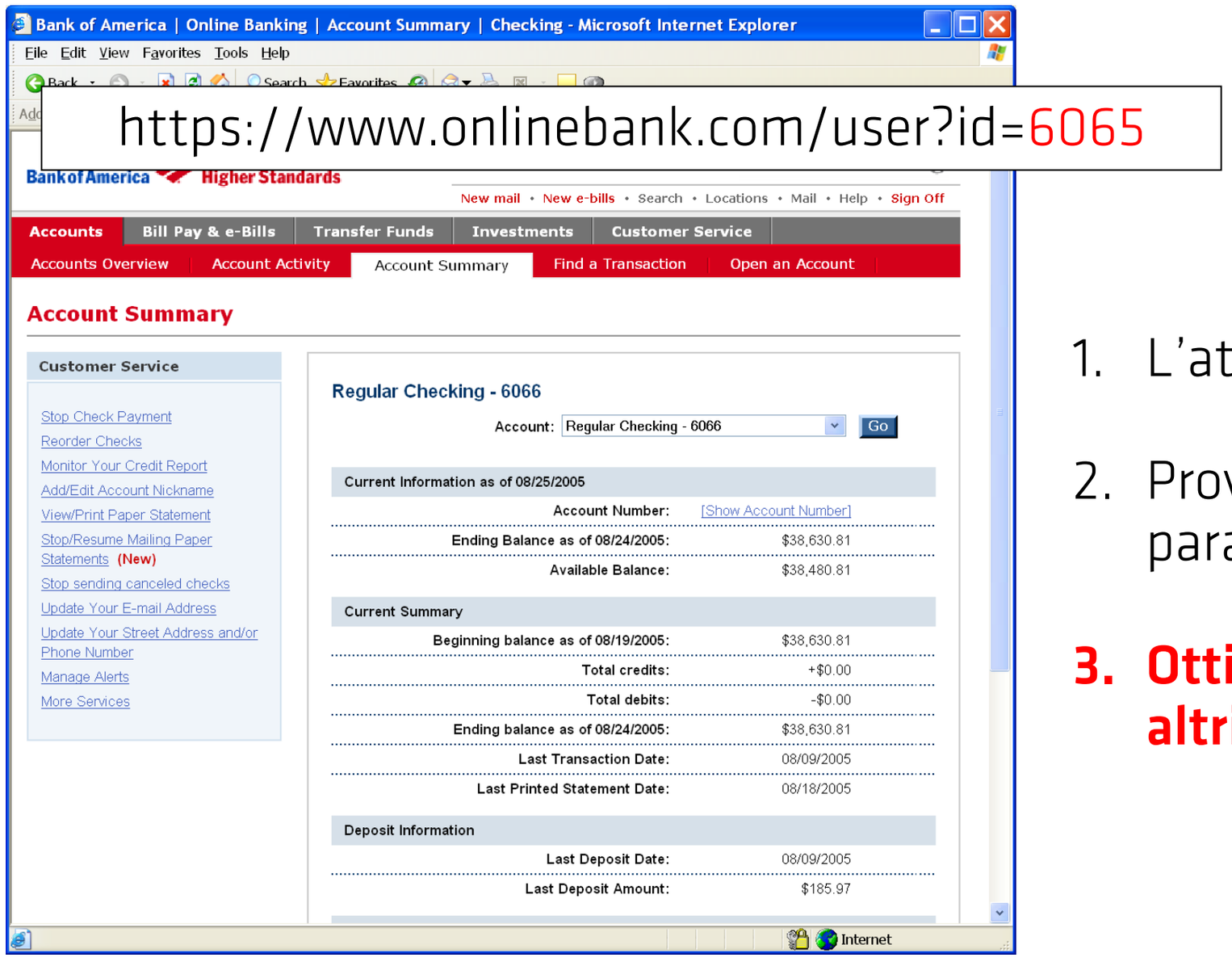
```
http://php.testsparker.com/products.php?pro=url"></script><scrip
t>alert("xss")</script><script src="x
```


A4 – Insecure Direct Object References

An application **exposes a direct reference to an internal implementation object**, such as a file, directory, or database key.

Attackers can manipulate these references to access unauthorized data.

A4 – Insecure Direct Object References



1. L'attaccante è un utente del sito
2. Prova a modificare il valore del parametro ID
3. **Ottiene l'accesso alle pagine degli altri utenti**

A4 – Insecure Direct Object References

Impact

- Users are able to access unauthorized files or data

Recommendations

- Eliminate the direct object reference
- Replace them with a temporary mapping value (e.g. 1, 2, 3)
- Validate the direct object reference
  - Verify the parameter value is properly formatted
  - Verify the user is allowed to access the target object

A5 – Security misconfiguration

The application is missing the proper security hardening across one or more parts of the application stack:

- OS
- Web/App Server
- DBMS
- Code libraries
- Unnecessary features enabled or installed (e.g.: services, pages)
- Default accounts enabled with passwords unchanged
- Error pages showing overly informative error messages to users

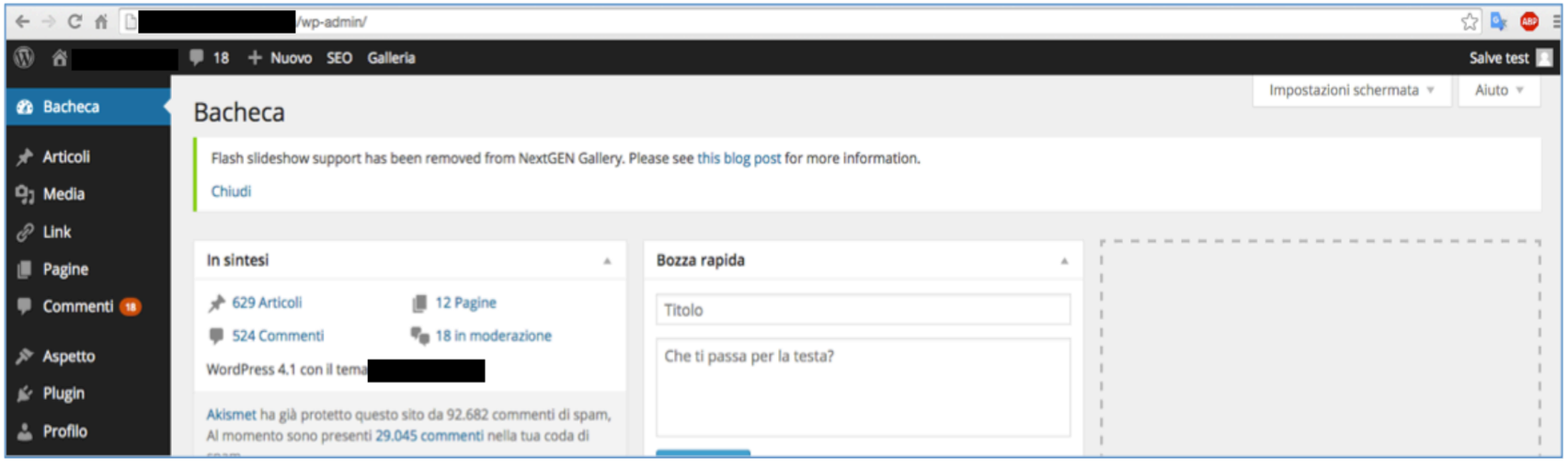
A5 – Security misconfiguration

Test Account

1. L'attaccante fa enumeration degli utenti di WordPress

Id	Login	Name
0	[redacted]	[redacted]
2	[redacted]	[redacted]
3	[redacted]	[redacted]
5	[redacted]	[redacted]
6	[redacted]	[redacted]
8	[redacted]	[redacted]
9	[redacted]	[redacted]
10	[redacted]	[redacted]
11	test	test, Author at
12	[redacted]	[redacted]
13	[redacted]	[redacted]

2. L'attaccante prova l'account di test con:
- **Username: test**
- **Password: test**



A5 – Security misconfiguration

Impact

- Install backdoor through missing OS or server patches
- Unauthorized access to default accounts, application functionality or data

Recommendations

- Verify your system's configuration management
- Be aware of the configurations of your components

A5 – Security misconfiguration

~~SSL~~/TLS

- No SSL, no TLSv1.0, ok TLSv1.1, TLSv1.2
- Ciphers:
  - No RC2, RC4, Null, Export…
  - Possibly no DES
  - 128+ bit ciphers
  - Forward Secrecy ciphers
- Use 2048-bit Private Keys
- Signature: no SHA-1, ok SHA-2 family
- No Client-Initiated Renegotiation
- **New Vulnerability Alerts!**

Check HTTPS configuration

https://www.ssllabs.com/ssltest/analyze.html

https://www.htbridge.com/ssl/

A5 – Security misconfiguration

HTTP Headers

- Browser-based layer of security
- Not completely cross-browser

- X-Frame-Options:
  - Allow/deny embedding within `<frame>, <iframe>, <object>`
  - IE 8+, Chrome 4.1.249.1042+, Firefox 3.6.9+, Safari 4.0+, Opera 10.50+
  - `X-Frame-Options: DENY`
  - `X-Frame-Options: SAMEORIGIN`
  - `X-Frame-Options: ALLOW-FROM http://trusted-origin.com`

A5 – Security misconfiguration

HTTP Headers

- Cache-Control:
  - Permit/prevent the caching of responses containing sensitive data
  - IE 6+, Chrome, Firefox, Safari, Opera
  - `Cache-Control: no-store`
  - `Pragma: no-cache` (for HTTP 1.0 compatibility)
  - `Expires: 0` (for HTTP 1.0 compatibility)
- Other:
  - HTTP Strict-Transport-Security (HSTS)
  - X-Content-Type-Options
  - X-XSS-Protection
  - https://appsec-labs.com/portal/improve-your-web-apps-security-with-http-headers/

Check HTTP headers configuration

https://securityheaders.io/

Security Report Summary

A

Site:	https://lastpass.com/
IP Address:	23.207.38.173
Report Time:	05 Jul 2016 11:05:56 UTC
Headers:	✔ X-Frame-Options ✔ Content-Security-Policy ✔ Strict-Transport-Security ✔ X-XSS-Protection ✔ X-Content-Type-Options ✖ Public-Key-Pins

Raw Headers

HTTP/1.1	200 OK
Content-Type	text/html; charset=UTF-8
X-Frame-Options	SAMEORIGIN

Laboratorio – Configurazione HTTPS

Test 1

Verifica della configurazione HTTPS di `studenti.unibo.it` tramite `https://www.ssllabs.com/ssltest/analyze.html`

Test 2

Verifica della configurazione HTTPS di `studenti.unibo.it` tramite `https://www.htbridge.com/ssl/`

Laboratorio – Configurazione Header HTTP

Test 1

Verifica della configurazione degli header HTTP di `php.testsparker.com` tramite `https://securityheaders.io/`

Test 2

Verifica della configurazione degli header HTTP di `studenti.unibo.it` su tramite `https://securityheaders.io/`

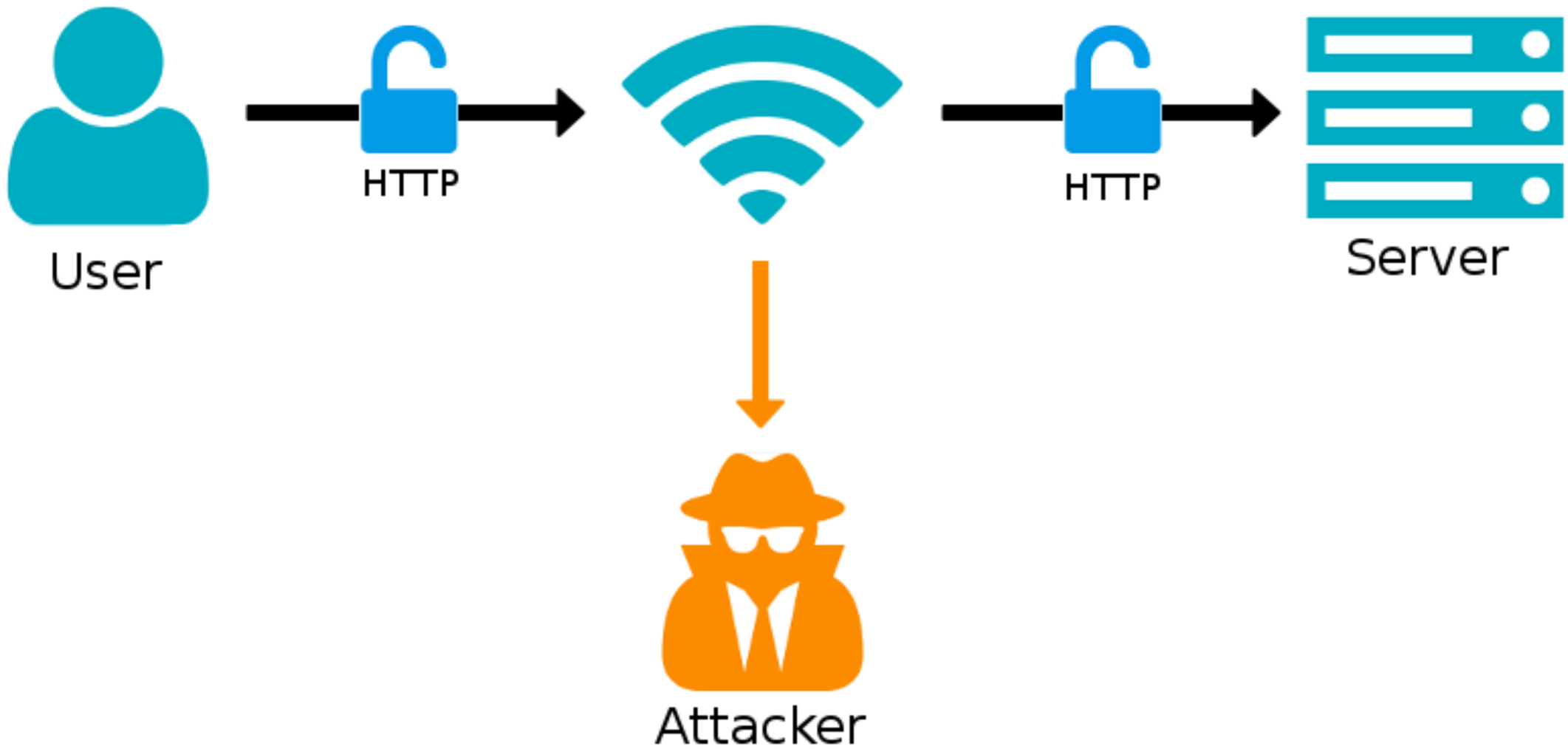
A6 – Sensitive Data Exposure

The application does not properly protect sensitive data (credit cards, authentication credentials…).

- Failure to **identify all the places** that data is stored or sent (web pages, third party websites…)
- Failure to protect **data at rest** (databases, log files, backups…)
- Failure to protect **data in transit** (browser ↔ web server, web server ↔ database server…)

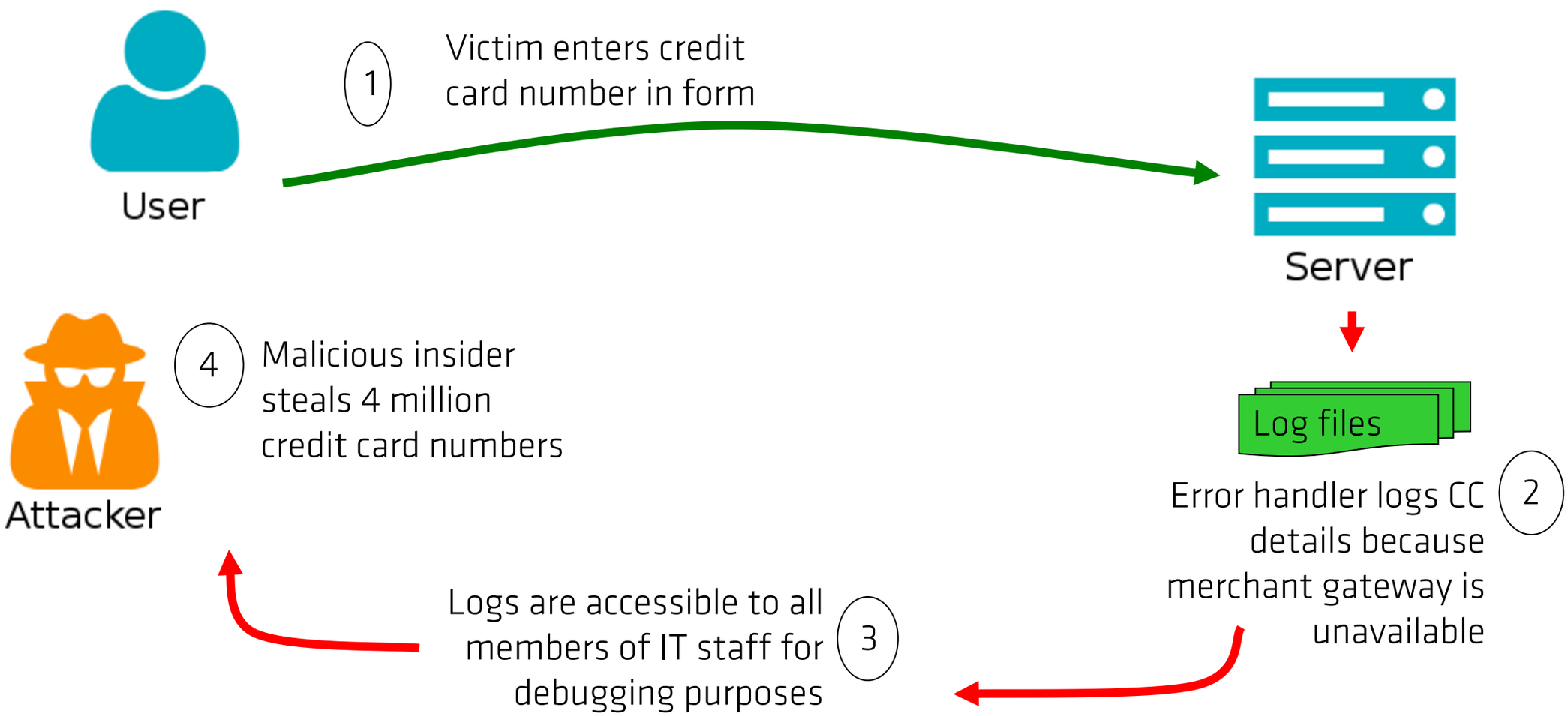
A6 – Sensitive Data Exposure

Man in the Middle



A6 – Sensitive Data Exposure

Insecure Storage



A6 – Sensitive Data Exposure

Impact

- Attackers access or modify confidential or private information
- Attackers extract secrets to use in additional attacks

Recommendations

- Identify all sensitive data
- Identify all the places that data is stored
- Identify all the places that data is sent
- Use encryption
  - Correctly implement encryption!

Laboratorio – Man in the middle

Test 1

Verifica tramite Burp dell'invio in chiaro via POST delle credenziali a partire dal form di login all'URL `http://php.testsparker.com/auth/login.php`

Test 2

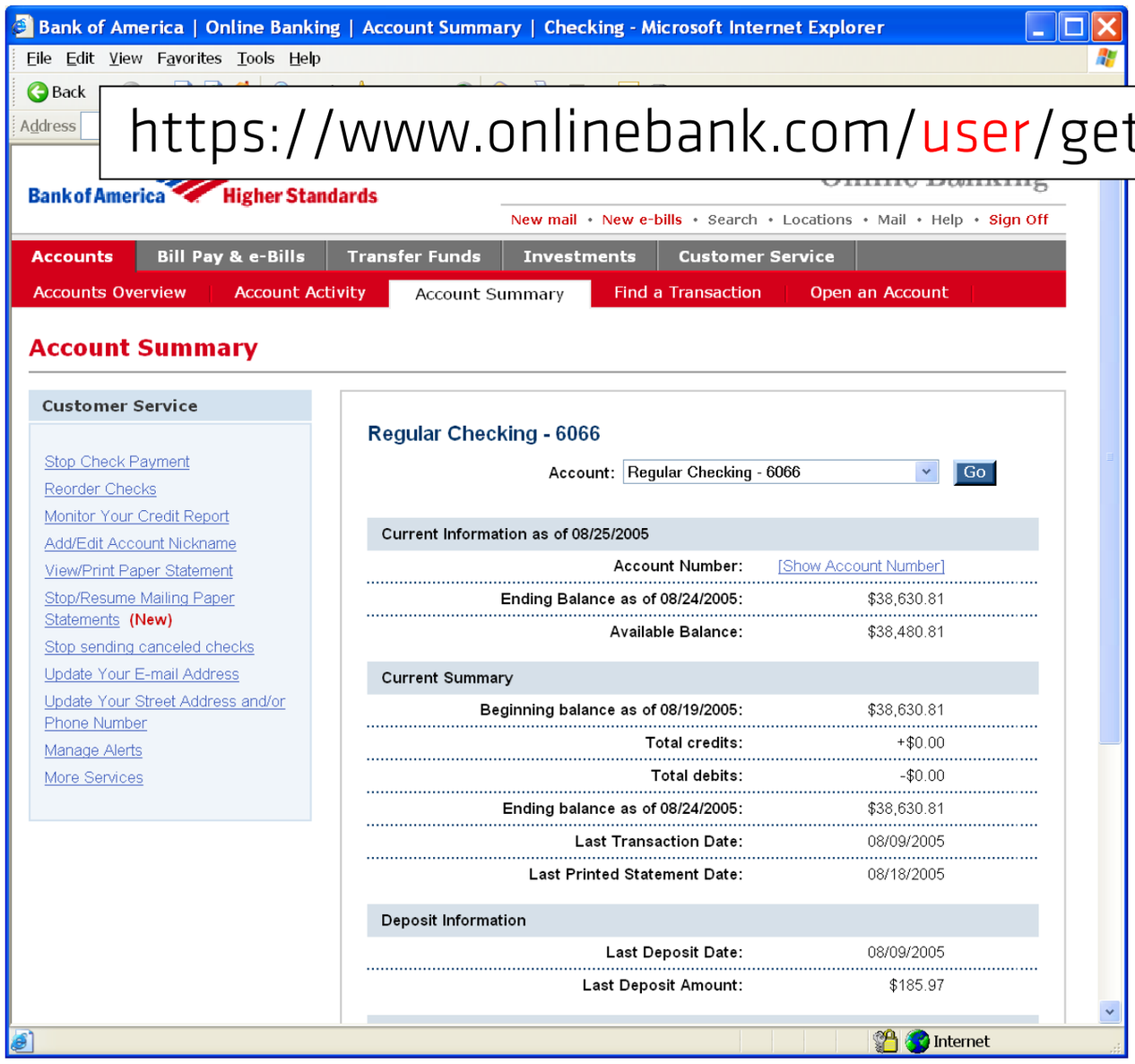
Verifica tramite Burp dell'invio in chiaro via GET delle credenziali a partire dal form di login all'URL `http://google-gruyere.appspot.com/XXXXXXXXXXXXXXX/login` (raggiungibile da `https://google-gruyere.appspot.com/start`)

A7 – Missing Function Level Access Control

The application fails to verify function level access rights when each function is accessed.

Example: application verifies function level access rights just before making that functionality visible in the UI.

A7 – Missing Function Level Access Control



1. L'attaccante è un utente del sito

2. Prova a sostituire alla parola 'user' la parola 'admin': `https://www.onlinebank.com/admin/getAccounts`

3. **Ottiene l'accesso alla pagina di amministrazione del sito**

A7 – Missing Function Level Access Control

Impact

- Attackers invoke functions and services they're not authorized for
- Attackers Perform privileged actions

Recommendations

- Restrict access to authenticated users (if not public)
- Enforce any user or role based permissions (if private)
- Completely disallow requests to unauthorized page types (e.g., config files, log files, source files, etc.)
- Be sure you actually have a mechanism at every layer

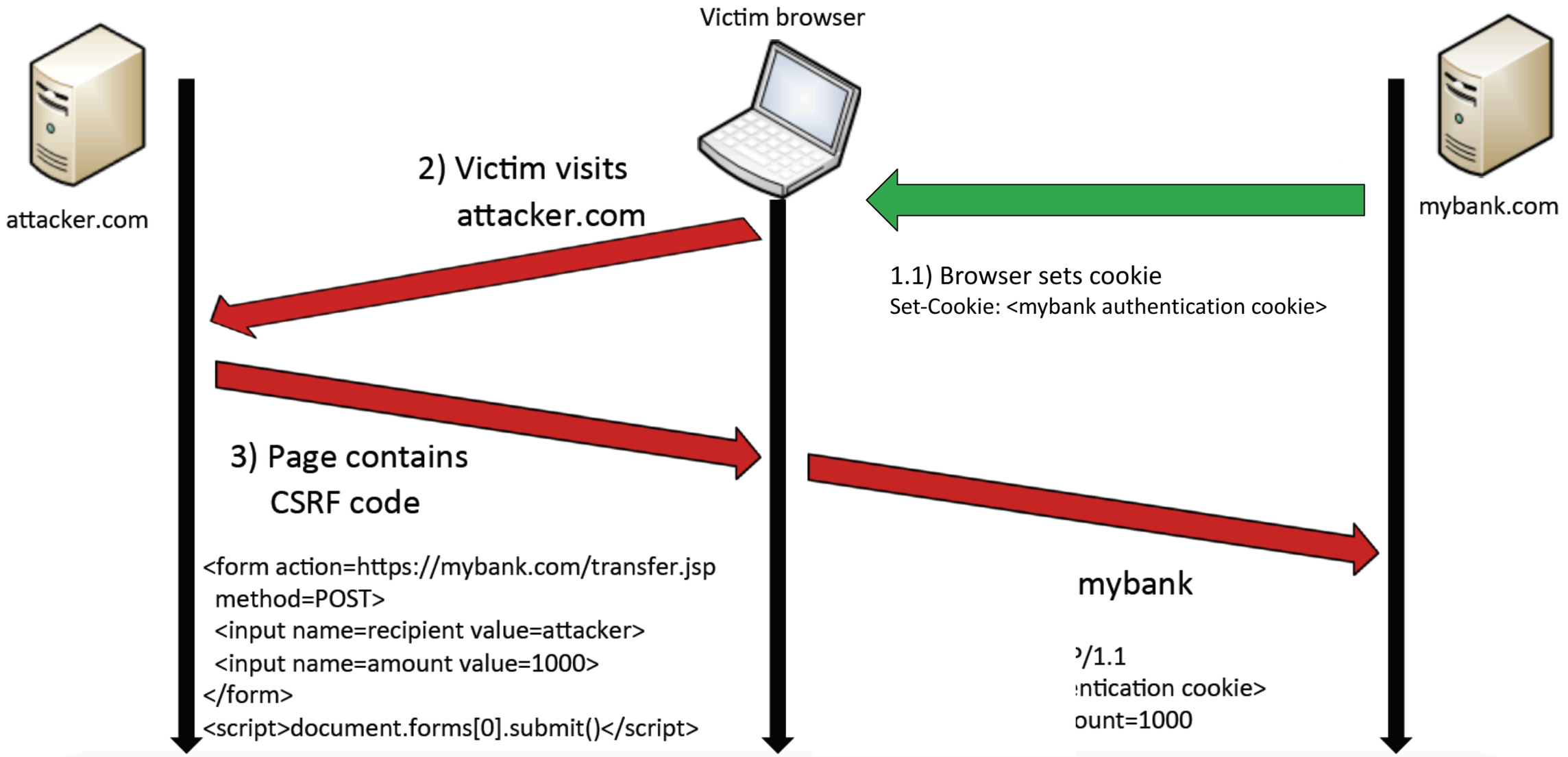
A8 – Cross Site Request Forgery (CSRF)

The victim's browser is tricked into issuing a command to a vulnerable w application, without the user being aware.

The vulnerable application thinks they are legitimate requests from the victim.

Vulnerability is caused by browsers automatically including user authentication data (session ID, IP address, Windows domain credentials, ...) with each request.

A8 – Cross Site Request Forgery (CSRF)



http://www.slideshare.net/fykim/rest-security-with-jaxrs

A8 – Cross Site Request Forgery (CSRF)

Impact

- Initiate transactions (transfer funds, close account…)
- Access sensitive data
- Change account details

Recommendations

- Add a secret token, not automatically submitted, to ALL sensitive requests
- Tokens should be cryptographically strong
- Beware exposing the token in a referer header
- Hidden fields are recommended

Laboratorio – Cross-Site Request Forgery (1)

Operazioni di partenza

- Creazione di un account su `https://google-gruyere.appspot.com/`
- (raggiungibile da `https://google-gruyere.appspot.com/start`)
- Login
- Creazione di un nuovo snippet
- Eliminazione dello snippet e analisi della richiesta HTTP relativa all'eliminazione (`https://google-gruyere.appspot.com/XXXXXXXXXXXXXXXX/deletesnippet?index=0`)

Laboratorio – Cross-Site Request Forgery (2)

Test

Vittima:

- Login e creazione di un nuovo snippet

Attaccante:

- Creazione di un link che, se visitato dalla vittima, faccia in modo che lo snippet sia automaticamente eliminato (`http://php.testsparker.com/hello.php?name=Visitor<script>function redirect() {window.location.replace("https://google-gruyere.appspot.com/XXXXXXXXXXX/deletesnippet?index=0");}window.onload = redirect;</script>`)
- Invio del link alla vittima

Vittima:

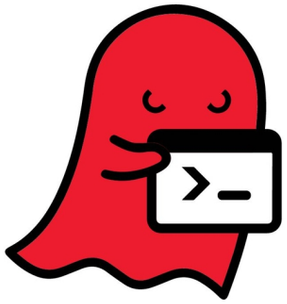
- Click sul link e verifica dell'avvenuta eliminazione dello snippet

A9 – Using Components with Known Vulnerabilities

The application relies on components (frameworks, libraries, products…) with known vulnerabilities.

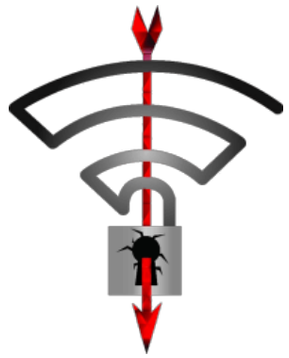
Heartbleed
CVE-2014-0160
(OpenSSL)

ShellShock
CVE-2014-6271
(Bash)



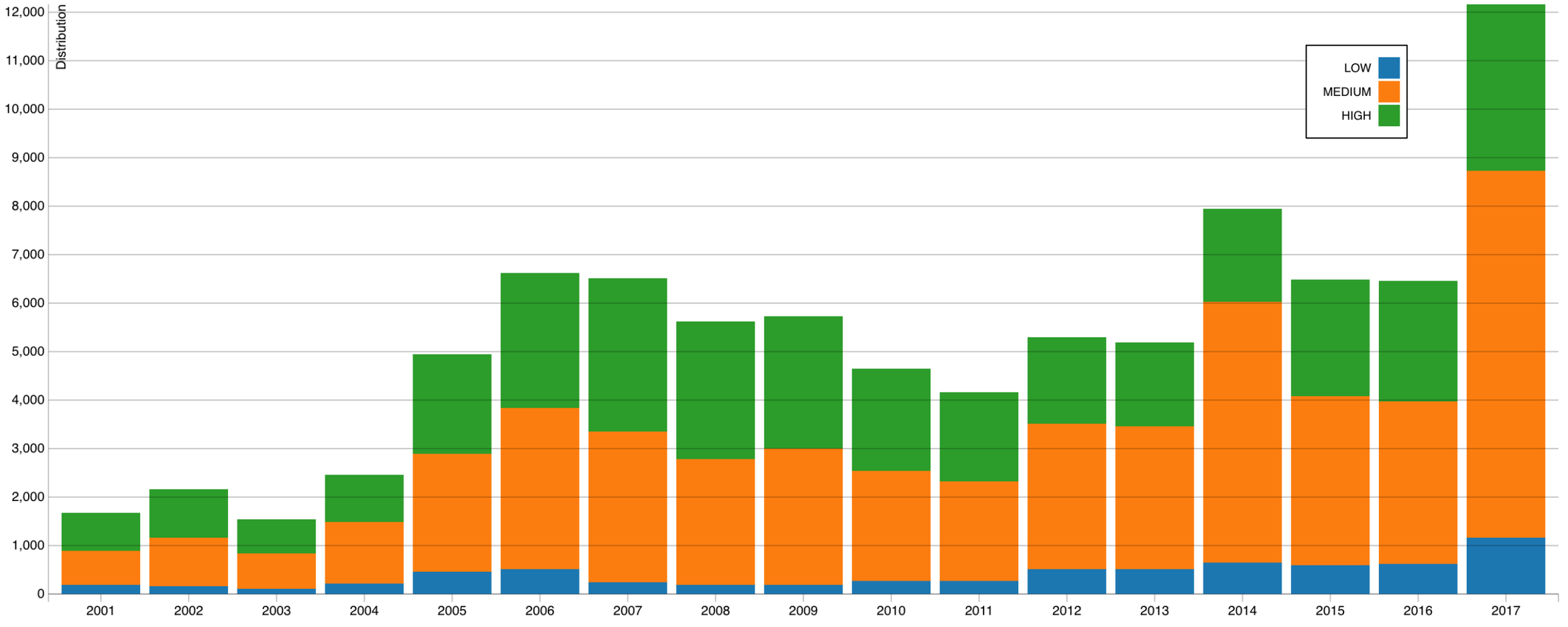
GHOST
CVE-2015-0235
(Linux)

DROWN
CVE-2016-0800
(OpenSSL)



KRAK
Vari CVE-2017
(WPA2)

A9 – Using Components with Known Vulnerabilities



cve.mitre.org

https://nvd.nist.gov/vuln-metrics/visualizations/cvss-severity-distribution-over-time

A9 – Using Components with Known Vulnerabilities

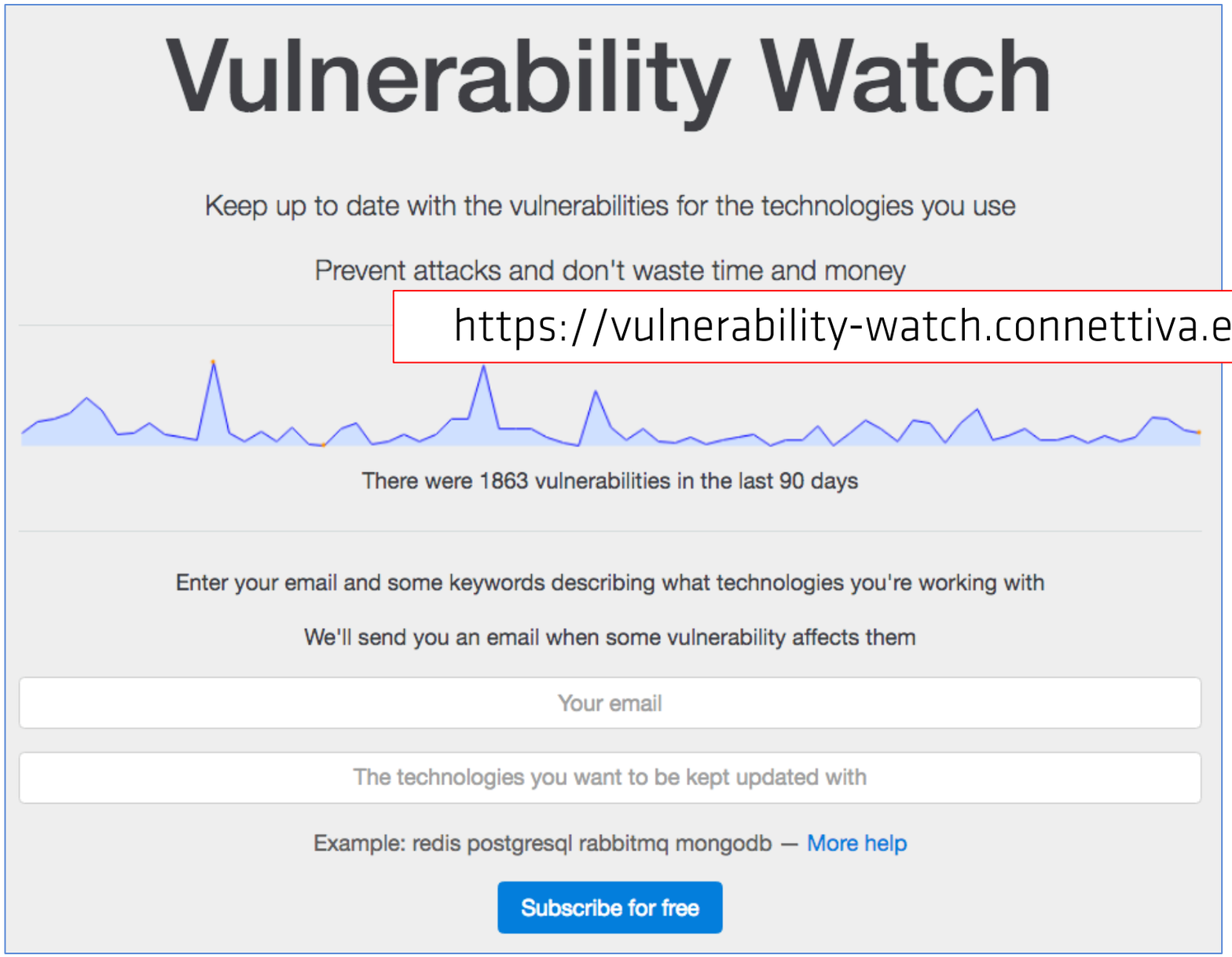
Impact

- Full range of weaknesses is possible, including injection, broken access control, XSS …
- The impact could range from minimal to complete host takeover and data compromise

Recommendations

- Perform periodic checks to see if your libraries:
  - are out of date
  - have known vulnerabilities

A9 – Using Components with Known Vulnerabilities



Laboratorio – Software Vulnerabili

Test

Analisi via Burp delle versioni dei software rivelate dagli header HTTP delle risposte fornite da `http://php.testsparker.com` e ricerca su CveDetails ed ExploitDb di informazioni sulle vulnerabilità.

A10 – Unvalidated Redirects and Forwards

The application redirects and forwards users to other pages and websites, and uses untrusted data to determine the destination pages.

A10 – Unvalidated Redirects ~~and Forwards~~

A10 – Unvalidated ~~Redirects and~~ Forwards

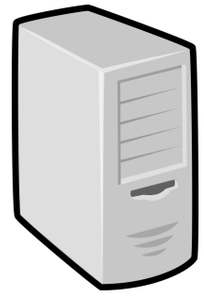
L'attaccante è un utente **non admin** del sito

1. L'attaccante prova a visitare http://www.example.com/admin.jsp

2. Il server fa il check sui privilegi dell'utente e verifica che l'utente non può visitare la pagina di amministrazione

```
public void adminPage
  (HttpServletRequest request,
  HttpServletResponse response) {
    try {
        // Do sensitive stuff here.
        ...
    }
    catch ( ...
```

3. Il server risponde con un Forbidden



A10 – Unvalidated ~~Redirects and~~ Forwards

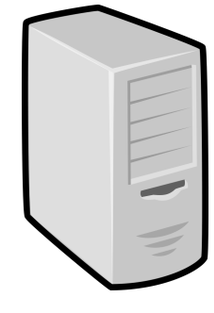
L'attaccante è un utente **non admin** del sito

4. L'attaccante prova a visitare `http://www.example.com/boring.jsp?` **`fwd=admin.jsp`**

```
public void doGet(
  HttpServletRequest request,
  HttpServletResponse response) {
    try {
      String target =
  request.getParameter("fwd"));
      ...
       request.getRequestDispatcher
      (target).forward(request,
      response);
    }
    catch ( ...
```

5. Il server non fa il check sui privilegi dell'utente e permette il caricamento della pagina di amministrazione

```
public void adminPage(
  HttpServletRequest request,
  HttpServletResponse
  response) {
    try {
      // Do sensitive
  stuff here.
      ...
    }
    catch ( ...
```



6. Il server risponde con la pagina di amministrazione, come se l'attaccante avesse visitato **`http://www.example.com/admin.jsp`**

A10 – Unvalidated Redirects and Forwards

Impact

- Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages.

Recommendations

- Avoid using redirects and forwards as much as you can
- If used, don't involve user parameters in defining the target URL
- If you 'must' involve user parameters, then either:
  - Validate each parameter to ensure its valid and authorized
  - Use server side mapping to translate choice provided by user with actual target page

Laboratorio – Unvalidated Redirect

URL di partenza

`http://php.testsparker.com/redir.php?param=test`

Test

`http://php.testsparker.com/redir.php?param=http://www.unibo.it`

Tutto qui?

La Top Ten considera le vulnerabilità più diffuse

Altre

- Abuso di funzionalità
- HTML Injection
- SSRF (Server Side Request Forgery)
- Local & Remote File Inclusion
- Information Leakage
- Frameable Response
- Errori nella business logic
- Denial of Service
- ...

HTML Injection

A type of injection issue that occurs when a user is able to control an input point and is able to inject arbitrary HTML code into a vulnerable web page.

Spesso derivato da una protezione parzialmente inefficace contro i Cross site scripting.

Abuso di funzionalità

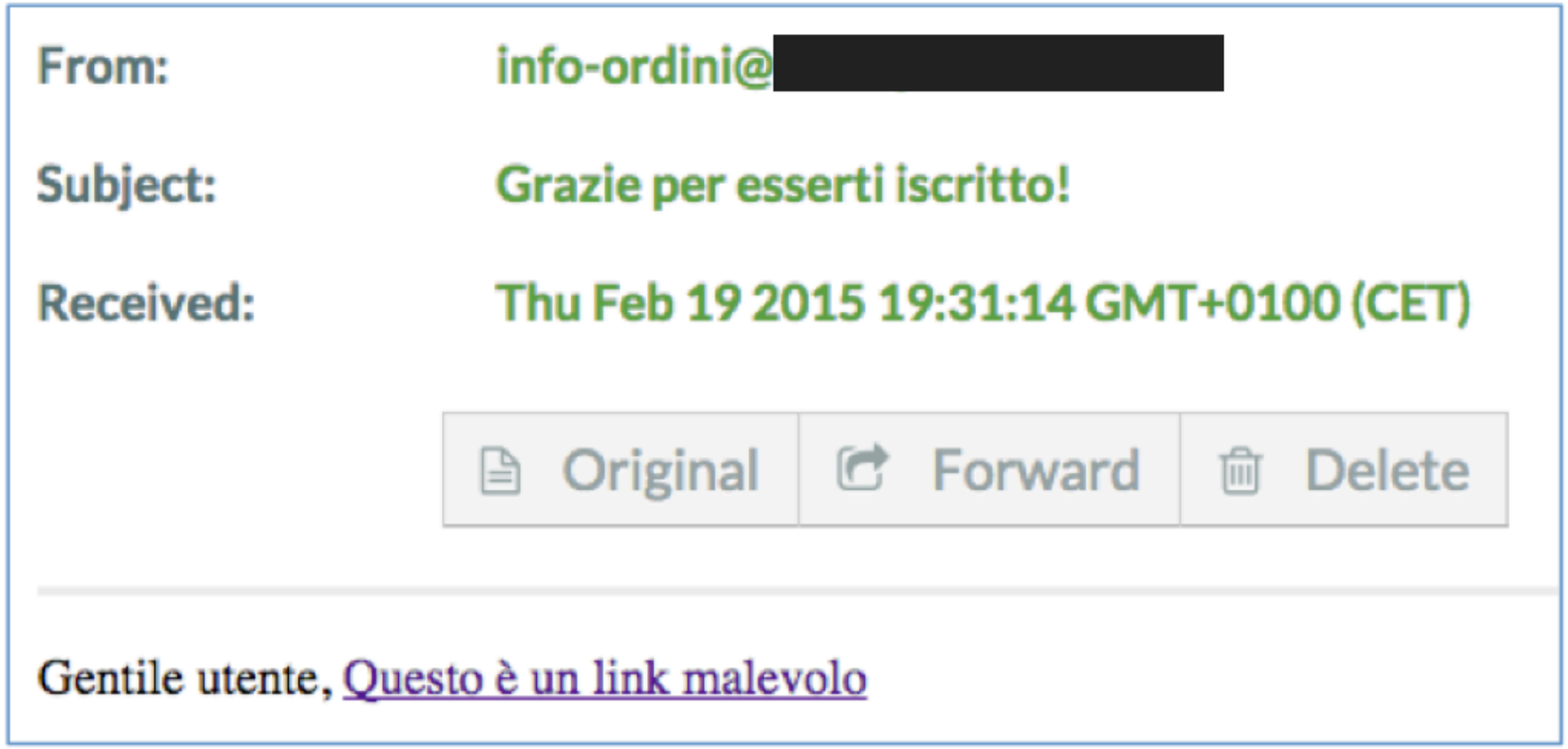
Un attaccante riesce a utilizzare in modo malevolo una funzionalità di una applicazione web

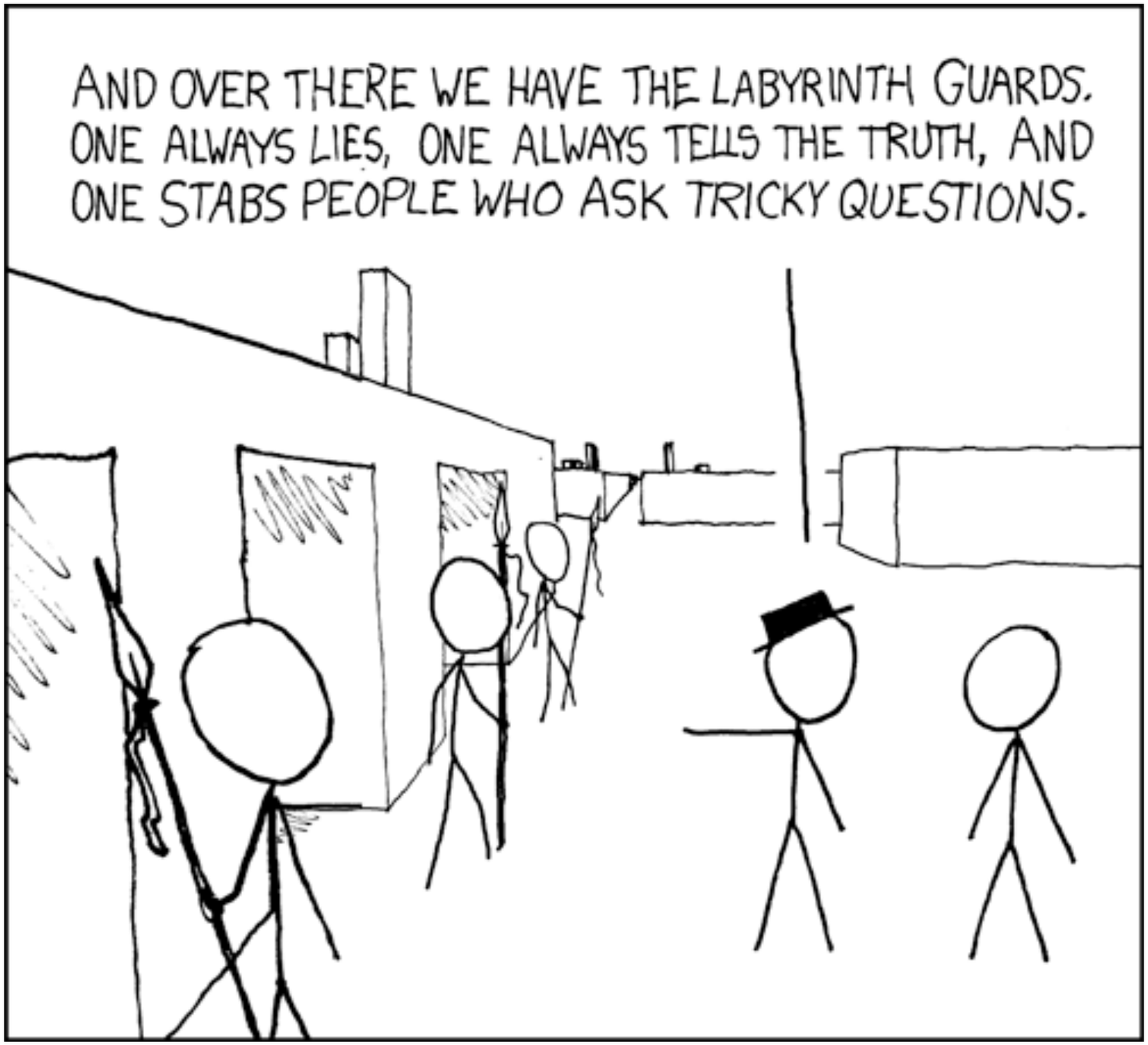
Casi tipici

- Form di registrazione
- Form di richiesta contatti

HTML Injection + Abuso di funzionalità

possibile realizzare uno scenario in cui un attaccante riesce in modo anonimo ad inviare email con contenuto arbitrario a qualunque indirizzo mail di persone non iscritte a [redacted], spacciandosi per info-ordini@[redacted]
Qui viene mostrato un esempio di email ricevuta da una vittima, in cui il contenuto del corpo del messaggio è completamente sotto il controllo dell'attaccante.





luca.capacci@cryptonetlabs.it

